Faculty of Electronics and Computer Technology and Engineering

Development of Smart Package Delivery System using Line Following Robot with Passcode Verification

KHIDHIR BIN KHIBIR

Bachelor of Electronics Engineering Technology with Honours

2023

Development of Smart Package Delivery System using Line Following Robot with Passcode Verification

KHIDHIR BIN KHIBIR

A project report submitted
in partial fulfillment of the requirements for the degree of
Bachelor of Electronics Engineering Technology with Honours

Faculty of Electronics and Computer Technology and Engineering

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2023

UNIVERSITI TEKNIKAL MALAYSIA MELAKA
FAKULTI TEKNOLOGI KEJUTERAAN ELEKTRIK DAN ELEKTRONIK

BORANG PENGESAHAN STATUS LAPORAN
PROJEK SARJANA MUDA II

Tajuk Projek : Smart Package Delivery System Using Line Following Robot with Passcode Verification

Sesi Pengajian : 2023

Saya …………Khidhir Bin Khibir……….. mengaku membenarkan laporan Projek Sarjana Muda ini disimpan di Perpustakaan dengan syarat-syarat kegunaan seperti berikut:

1. Laporan adalah hakmilik Universiti Teknikal Malaysia Melaka.
2. Perpustakaan dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan laporan ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. Sila tandakan (✓):

☐ **SULIT*** (Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)

☐ **TERHAD*** (Mengandungi maklumat terhad yang telah ditentukan oleh organisasi/badan di mana penyelidikan dijalankan)

☑ **TIDAK TERHAD**

Disahkan oleh:

(TANDATANGAN PENULIS)

Alamat Tetap:
51, Jalan Aman Perdana 9G/KU5 , Taman Aman Perdana , 41050 Klang , Selangor

(COP DAN TANDATANGAN PENYELIA)

TS. MOHD ANUAR BIN ADIP
Jurutera Pengajar
Fakulti Teknologi Dan Kejuruteraan Elektronik Dan Komputer (FTKEK)
Universiti Teknikal Malaysia Melaka (UTeM)

Tarikh:15/1/2024

Tarikh:

*CATATAN: Jika laporan ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali tempoh laporan ini perlu dikelaskan sebagai SULIT atau TERHAD.

DECLARATION

I declare that this project report entitled "Smart Package Delivery System Using Line Following Robot with Passcode Verification" is the result of my own research except as cited in the references. The project report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature :

Student Name : Khidhir Bin Khibir

Date : 12/1/2024

APPROVAL

I hereby declare that I have checked this project report and in my opinion, this project report is adequate in terms of scope and quality for the award of the degree of Bachelor of Electrical Engineering Technology with Honours.

Signature : TS. MOHD ANUAR BIN ADIP
Jurutera Pengajar
Fakulti Teknologi Dan Kejuruteraan Elektronik Dan Komputer (FTKEK)
Universiti Teknikal Malaysia Melaka (UTeM)

Supervisor Name : Ts Mohd Anuar Bin Adip

Date :

Signature :

Co-Supervisor Name (if any) :

Date :

DEDICATION

To my beloved Mother and Father
My dedicated lecturers, and my supportive friends

ABSTRACT

Universities, colleges, and independent boarding institutions now receive a sizable number of packages on a regular basis. The staff has a limited amount of time to track down names and receive packages. There is a substantial likelihood that a package will go missing, and the frequency of missing packages is rising. This Report Represent an idea on Developing a Smart Package Delivery System using Line Following Robot with Passcode Verification and integrate a system that offers a new delivery package using robots and Internet of things security system. This Proposed system uses line sensor that help to navigate the microcontroller Arduino Mega 2560 and to control the motors using Motor Driver. The Internet of things in this system is the Passcode Verification that will is developed with Blynk Application as the interface of the system and for sending the TAC code message the system is develop by using ESP32 SIM800L microcontroller that will notify user once the package arrives. When the robot arrives at its destination, the receiver are required to input the TAC code into the system by pressing the keypad. If the TAC code entered is correct,the system will wait for the receiver to press the "#" button to incdicate that the package have been received and the robot will be going back to its original position. This Project need to achieve the three objectives on designing and developing a Smart Package Delivery System using Line Following Robot with Passcode Verification with the helps on programming system that send the instruction to every component used that helps to overcome the student and staff problem. Next, the integrated system programmed in this project has succeeded in analyzing the analysis to get the its overall accuracy and effeciency of Smart Package Delivery System using Line Following System with Passcode Verification.

ABSTRAK

Universiti, kolej dan institusi pelajaran kini menerima banyak bungkusan pakej dengan kerap. Kakitangan mempunyai masa yang terhad untuk mengesan nama dan menerima pakej. Terdapat kemungkinan besar bahawa pakej akan hilang, dan kekerapan kehilangan pakej akan semakin meningkat. Laporan ini Mewakilkan idea tentang Membangunkan Sistem Penghantaran Pakej Pintar menggunakan Robot Mengikut garisan dengan Pengesahan Kod Laluan dan menyepadukan sistem yang menawarkan pakej penghantaran baharu menggunakan robot dan sistem keselamatan yang mengintegrasikan ‘Internet of things’. Sistem Cadangan ini menggunakan Line Sensor, pengesan garisan yang telah akan membantu mengemudi sistem robot ini dengan bantuan komponen lain seperti Pemacu Motor untuk membantu mengawal motor. Projek ini mencadangkan sebuah sistem Pengesahan Kod Laluan yang akan dibangunkan dengan Aplikasi Blynk sebagai pemuka untuk sistem ini dan ESP32 SIM800L untuk menghantar mesej TAC kepada penerima. Apabila penerima menerima pakej, dia perlu menekan butang keypad untuk memasukan kod TAC yang betul dan penerima juga perlu menekan butang “#” untuk memberitahu bahawa pakej telah diterima dan robot akan kembali ke kedudukan asal. Projek ini perlu mencapai tiga objektif untuk mereka bentuk dan membangunkan Sistem Penghantaran Pakej Pintar menggunakan Robot Mengikut Garisan dengan Pengesahan Kod Laluan dengan bantuan sistem pengaturcaraan yang menghantar arahan kepada setiap komponen yang digunakan yang membantu mengatasi masalah pelajar dan staf.Seterusnya, sistem bersepadu yang diprogramkan dalam projek ini telah berjaya membuat analisis terhadap ketepatan dan kecekapan sistem Penghantaran Pakej Pintar menggunakan Robot Mengesan Garisan dan Pengesahan Kod Laluan.

ACKNOWLEDGEMENTS

Alhamdulillah, I have taken so many efforts in this project. However, it would not have been possible without the kind support and help of many individuals. I would like to express my special thanks to my supervisor Ts Anuar Bin Adip whose help in simulating suggestions and gave the encouragement to complete this project and the process of writing my report.

Thank you for giving me the opportunity to do this wonderful project in the topic of Development of Smart Package Delivery System using Line Following Robot with Passcode Verification" which helped me to learn many new things. I also want to thanks to my dear friend Mohd Adib Najmi Bin Zamri for her support in completing my report and project. I really appreciate the time and effort of her discussing and giving me idea in this project.

Again, thank you for your kindness throughout the journey. Next, I would like to thank my parents and family who always understand, appreciated and support my journey on Final Year Project 1. Without their support and love, this project would be hard and possible to complete. I hope this report would give benefit to people and would help people gain more knowledge. I wish this project would have a place and would be useful to people in the future.

TABLE OF CONTENTS

LIST OF TABLES

TABLE **TITLE** **PAGE**

LIST OF FIGURES

LIST OF ABBREVIATIONS

0TP	-	One Time Password
LCD	-	Liquid Crystal Display
IOT	-	Internet Of Things
AI	-	Artificial Intelligence
IDE	-	Integrated Development Environment
GSB	-	Global Smart Box
LMP	-	Last Mile Problem
SMS	-	Short Message Services
LIDAR	-	Light Detection and Raging
LMD	-	Last Mile Delivery
TSP	-	Travelling Salesman Problem
TSP-MD	-	Traveling Salesman Problem-Moving Depot
VRP	-	Vehicle Routine Problem
GAMS	-	General Algebraic Modelling System
BDT	-	Bangladesh TAKA
INR	-	Indian Rupee
USD	-	United State Dollar
PID	-	Proportional Integral Derivative
DC	-	Direct Current
RF	-	Radio Frequency
GPS	-	Global Positioning System
GUI	-	Graphical User Interface
GPIO	-	General Purpose Input/Output
PWM	-	Pulse Width Modulation
UART	-	Universal Asynchronous Receiver/Transmitter
ICSP	-	In Circuit Serial Programming
USB	-	Universal Serial Bus
PCB	-	Printed Circuit Board
VSM	-	Virtual System Modelling

CHAPTER 1

INTRODUCTION

1.1 Background of Project

A significant market in the coming years will be online marketing. A smart, secure delivery system ought to be suggested at the client side, nevertheless, as technology advances in the direction of the internet of things. This clever solution should be practical, quicker to use, and secure the delivery of the package. A secure delivery box that generates an OTP for each active session while informing the client of the orderly process flow up until the session is ended. With the least amount of operational delay, the system operates as a secured Box. The delivery vault's authenticity is increased by using a time-based One Time Password.Text messages are sent over the GSM network since it has the benefit of spanning a larger area of operation even while the customer is on the go. [1]

It have a big transportation capacity and can move bulky objects over long distances. The development of technology has led to automation, like driverless robots. The suggested model shows a robotic device that moves automatically between points A and B without requiring human intervention. Obstacles are recognized using the sensors. The robot will halt if there are any impediments in its path, then resume moving once they have been removed. A self-driving system's fundamental premise is the existence of a robot that can move tangible goods from one location to another. A robot that makes deliveries is referred to as a self-contained delivery robot. Artificial intelligence is used by self-driving robots, which arrive at their destination after taking the right route. The robot is employed to deliver the packages securely [2] Urban logistics is the control of the pickup and delivery of packages and items to meet human needs in a metropolis. Urban logistics typically rely on professional transportation services with privately owned vehicles and couriers; as a result, this strategy is expensive and challenging to expand. [3]

1.2 Motivation

Line Based on Artificial Intelligence (AI) The latest technology offered by Following Robot has the potential to alter the global environment and improve human lives. Only artificial intelligence can realize the dream of giving intelligence to our robots so they can comprehend challenges in real life and respond appropriately. It can be utilized on the production or assembly line buses that pick up passengers..

1.3 Problem Statement

Universities, colleges, and independent boarding institutions now receive a sizable number of packages on a regular basis. This issue causes a variety of problems for the on-site workers in university post rooms, residence halls, and student housing. If internet shopping continues to increase as predicted each year, the situation will only become worse.

The staff has a limited amount of time to track down names and receive packages. As the number of parcels increases, more hours must be devoted to managing and distributing them. Staff that provide manual parcel management services incur time and financial costs. Since traditional parcel management services are typically provided for free, there is no way to profit financially from them or to offset their costs, thus when parcel quantities rise, so do the costs.

There is a substantial likelihood that a package will go missing, and the frequency of missing packages is rising. To avoid missing packages, you need proof of delivery and pickup. You can show them on the archived record that they picked up the package and signed for it if the package was collected. You can also prove that it was never delivered.

The existence of a log that building managers may consult helped us to settle a great deal of problems. With the help of modern IoT, this project system offers to combine a program that offers high security together with the password verification implementation to notify that the parcel has been delivered to the owner.

In most cases, mailrooms has lack sufficient area to divide packages up by level. To get out of this predicament, there is a system on sale that uses a Line Following Robot with Passcode Verification to provide a Smart Package Delivery System that provides a solution to the problem of managing a parcel delivery

1.4 Recent Objectives

The goal of this project is to develop a project and a system for managing parcel delivery utilizing intelligent robots that aid in the safe delivery of all packages. Therefore, the objective can be subdivided as follows:

i. To design Line Following Robot and (IoT) Passcode Verification using ESP32.

ii. To develop a Smart Package Delivery System using Line Following Robot with Passcode Verification.

iii. To analyze the accuracy and efficiency of the Smart Package Delivery System using Line Following Robot with Passcode Verification.

1.5 Scope of Work and Limitation of the Project

The main goal of this project is to create a line-following robot-based smart package delivery system with passcode verification. A key component of the project's architecture is the integration of Internet of Things (IoT) technologies with hardware and software. The Arduino Mega microcontroller serves as the core component which is configured to give the line-following robot signals and instructions. Furthermore, the ESP32 microcontroller and the Arduino Mega connect as part of the Internet of Things architecture, enabling data transfer and package delivery coordination. This project's main goal is to develop a completely working system that combines password authentication with a line-following robot to deliver packages in a safe and effective manner. An additional degree of protection is provided by the TAC (Transaction Authorization Code), the scope of this project is as follows:

1. To construct the hardware circuit connection, software coding and product design using engineering software such as AutoCAD, Arduino IDE and Proteus.
2. To build Smart Package Delivery System using Line Following Robot with Passcode Verification
3. To determine the accuracy for Smart Package Delivery System using Line Following Robot to deliver package to each room.

1.6 Thesis Outline

This thesis is divided into five chapters. The project's objective, problem statement, and scope and limitation are briefly described in Chapter 1. The literature review study on smart delivery systems is mentioned in Chapter 2, however the method of system identification will primarily be the focus. In Chapter 3, the project approach was discussed using a flowchart and in accordance with the project's goals. The idea of the project flow, method used, and software used is explained and discussed. The functionality and preliminary results are covered in detail in this chapter, which is Chapter 4, which presents the project's outcomes and discussion. The project's conclusion and research findings are covered in detail in Chapter 5, along with future suggestions for how to enhance the project moving forward

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

A literature review is a compilation of scholarly articles that are directly related to the subject of your study. This chapter will discuss more relevant ideas from earlier authors in order to strengthen the notion of this project. To develop and obtain the results, a select few strategies were used. The literature reviews, which included a range of research or studies pertinent to the project, are the main topic of Chapter 2. In this chapter, the project has been broken up into three sections. The first part discusses the design and development of smart delivery system using Line Following Robot with Passcode Verification. The second part is the research on the programming and system applied in this smart delivery system using Line Following Robot with Passcode Verification. Last but not least, the third part of this literature review is the research on the smart delivery system using Line Following Robot with Passcode Verification Internet of Things Application performance

2.2 Smart Delivery System Project

Ahmed Abdulmahdi Abdulkareem Alawsi et al. 2020 proposed a new global smart box (GSB) was presented as a way to resolve LMP. The GSBs can be connected to a central controller via the internet and managed by multiple offices that provide simple delivery services spread out close to or inside trading complexes. The results obtained in this study demonstrate that the GSB is trustworthy and can handle the majority of LMP subproblems, such as reducing delivery costs, doubling the maximum delivery distance, and saving quadcopters and packages. To address the issues with the previously described last mile problem, a worldwide smart box has been devised and installed. Depending on the size of the population density in the installation area, the GSB can be built in a variety of sizes.

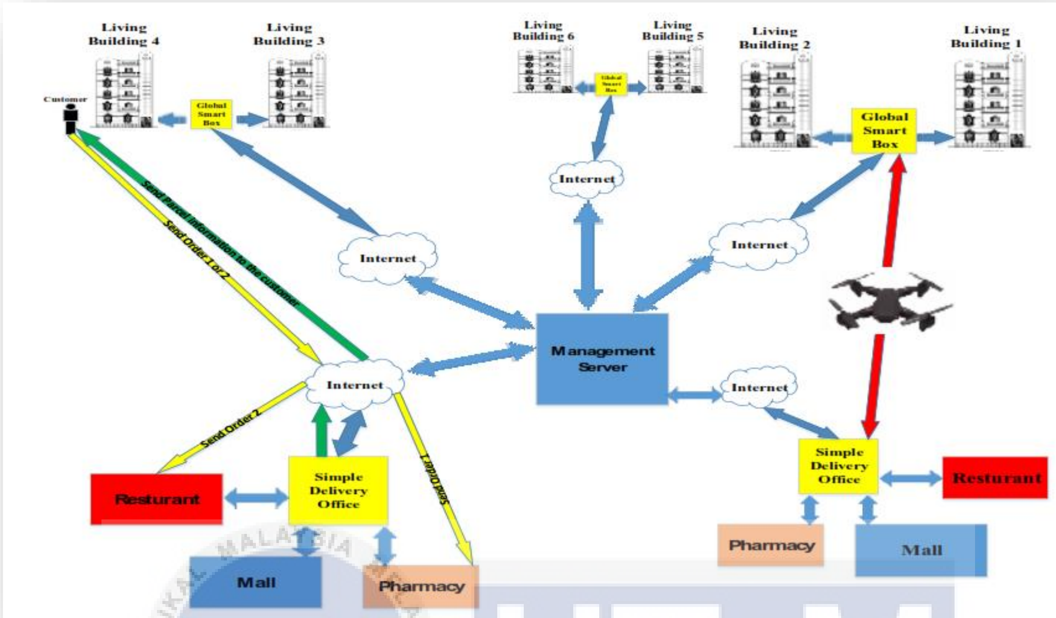


Figure 1 Architecture of the proposed system to solve LMP

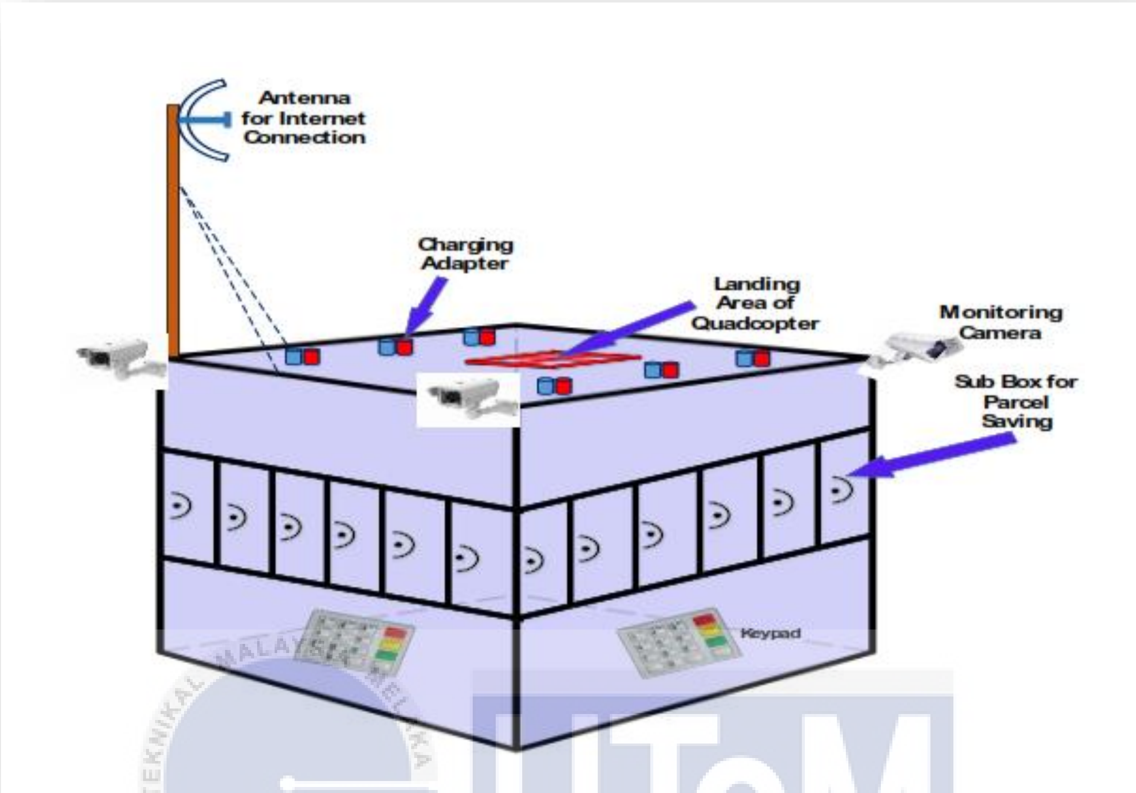


Figure 2 An illustration of the core components of the proposed global smart box (GSB)

The main server using the C (C##) programming language for its design and programming. The primary software for managing and controlling the majority of delivery processes is installed on the server. The server can be put at any simple\delivery office and connect with all delivery offices and GSBs through internet by using global IP with username and password to handle the delivery operations. The delivery server's home page, as depicted in the figure, allows users to choose the GSM and sub box that are nearest to the customer. In order to enter the customer's information, including their e-mail and cell phone, a new page opens when the book button is pressed as seen in the picture. After hitting the book button, the server randomly generates the password (code) depicted in figure 5. After the customer clicks the save button, the server will transmit the password and subbox number to his email address via the internet and to his mobile phone as an SMS message via GSM networks. Figure 6 displays an SMS message that the customer's mobile phone got, along with the password and subbox number (Misled in yellow). Additionally, the client will get the second message (Misled in blue) from the server after opening the sub box and receiving his package, as shown in figure 6. The delivery system software is very straightforward, making it easy for support offices to use and lowering the likelihood of issues. Through the server's home page, a unique password, GSB address and number, and subbox number are selected. These details are then sent to the client via email and mobile phone, where the password is entered to retrieve the package. In order to be able to take legal action against anyone who attempts to tamper with the box, all GSB can be watched using security cameras and the server at the delivery office [10]

Figure 3 Home page of the main delivery system

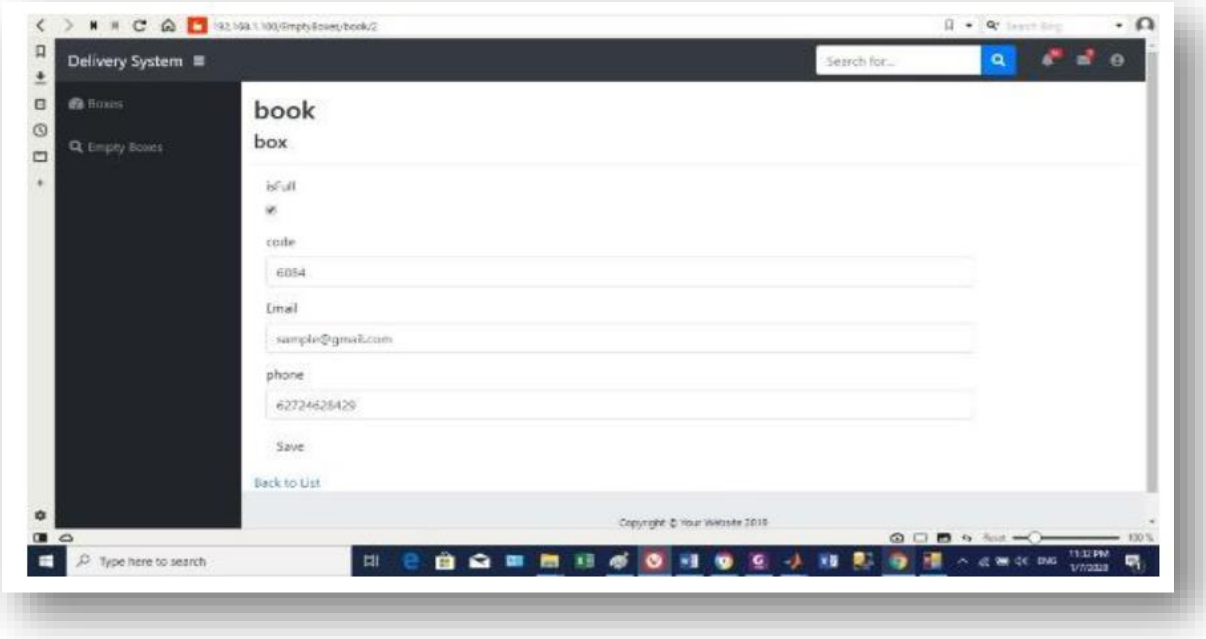


Figure 4 Server page used to determine sub box to the customer and enter his information

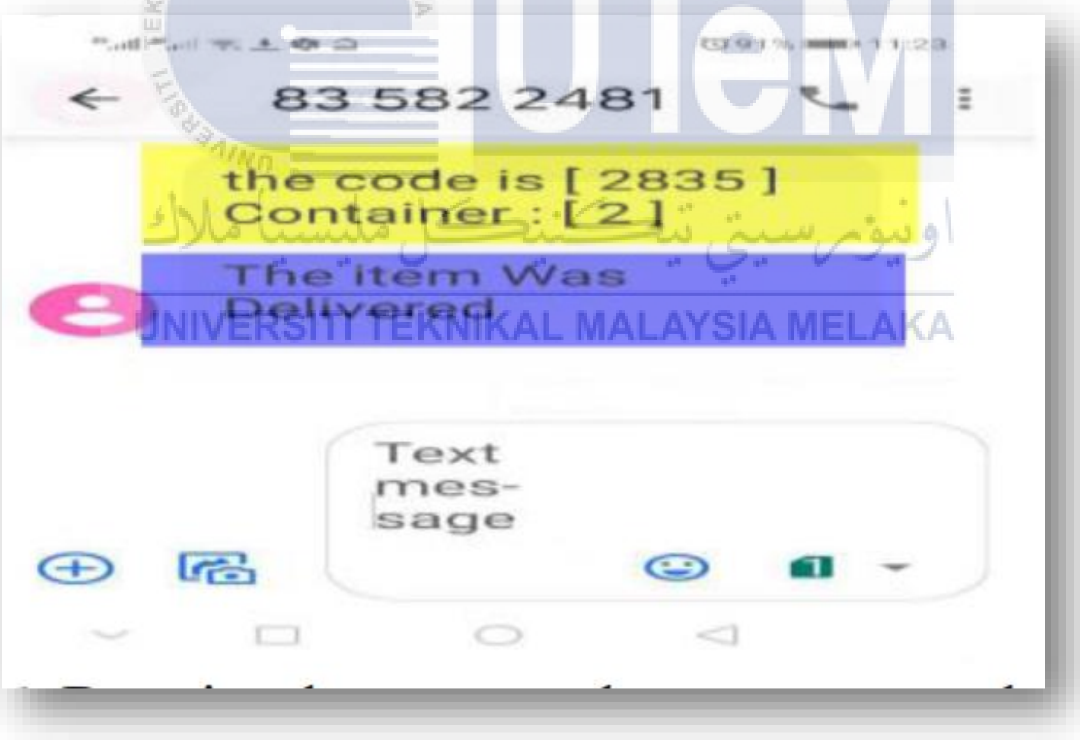


Figure 5 Figure 5:Received message by customer phone from the server.

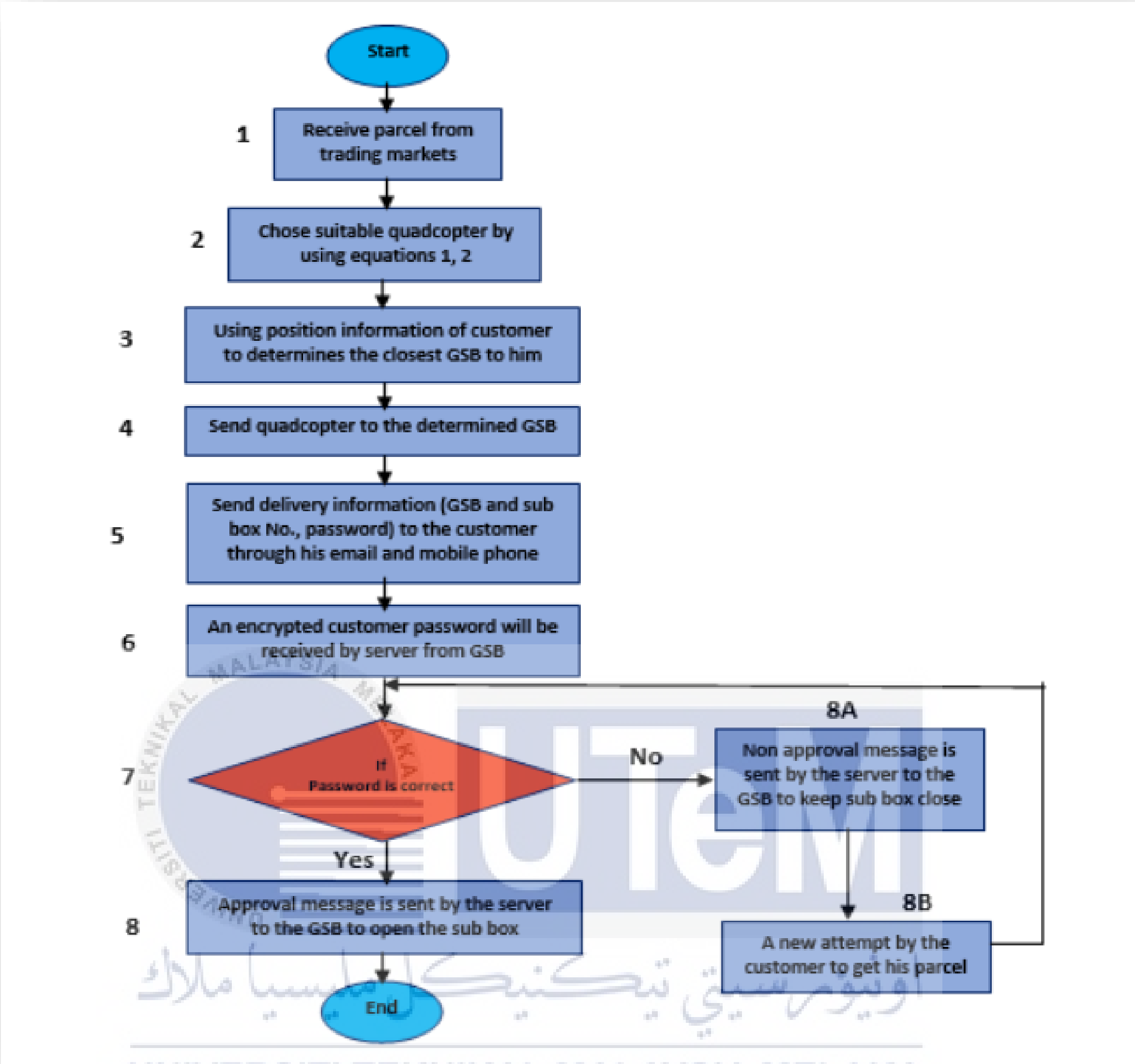


Figure 6 The algorithm of proposed GSB delivery system by quadcopter

Mayank Agarwal et. Al. 2015 proposed an IOT based Smart Delivery Box. The ease of use of the delivery system software makes it convenient for help centers to use and lowers the likelihood of issues. On the server's main page, a unique password, GSB address and number, and subbox number are selected. These details are sent to the client via email and mobile phone, and they are then entered to recover the package. Security cameras and the server at the delivery office can watch all GSB, allowing for the prosecution of any attempts to tamper with the box. The box needs to be set up and linked to the home wifi network in the customer's residence. The front box door provides entry to the box's only side. Initialization of the transportation box's sensors, which can recognize where a package is placed, is done. The door is opened by the delivery boy using the order Id to gain entry to the box, and after placing the package, the door is shut. When the customer enters their mobile number, a one-time password is created without time restrictions and sent to the customer's phone along with a notification that a package has been delivered. When the client gets home, he uses the generated one-time password to retrieve the package. An additional confirmation email and message are sent to the shipping portal service provider and e-commerce gateway when the package is retrieved. The amount of times the box was accessed and the delivery boy's credentials. Dimension of the parcel, time spent placing it, and session length can all be documented and saved for use in the future[11]

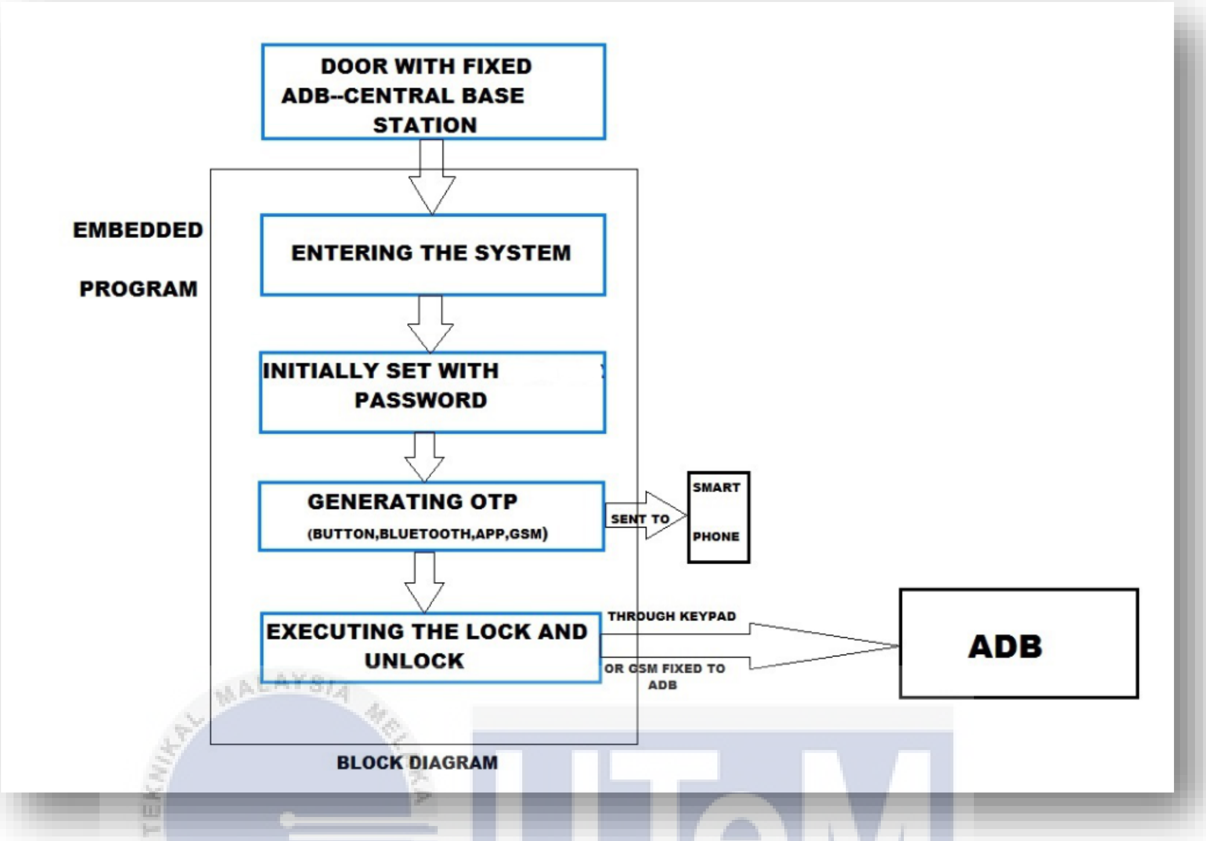


Figure 7 Block Diagram of the system.

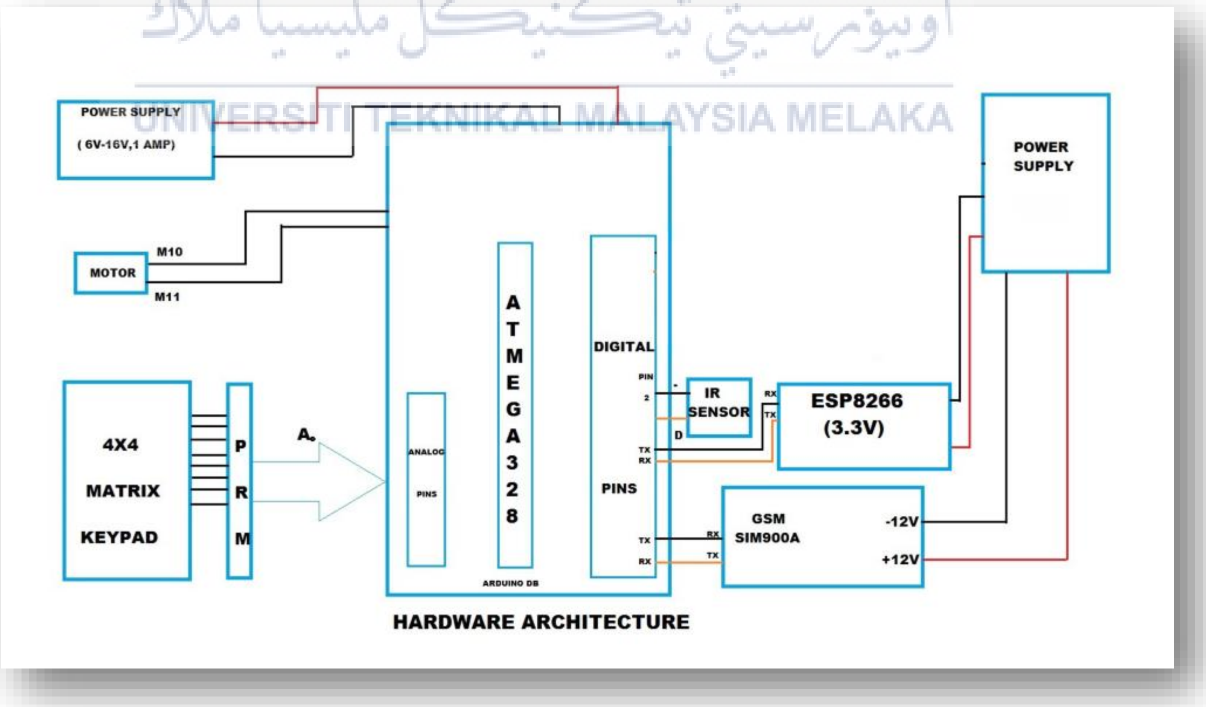


Figure 8 Hardware Architecture of the system

Ruslan Safin et al. 2021 proposed a system Modelling a TurtleBot3 Based Delivery System for a Smart Hospital in Gazebo. A system in which Mobile ground robots perform transportation tasks between multiple stations located in different rooms while navigating in an environment with moving objects such as humans and other mobile robots. The robot is equipped with a distance sensor (LIDAR), based on the indicators of which objects are detected and the robot is localized. Robots can be assigned tasks to be executed through a centralized interface. Tasks are assigned to a specific robot, and, depending on the type of task, the robot is autonomously directed to the station associated with the task in the building. We are considering the concept of defining possible robot behaviors as a finite set of states with certain transitions. To test the system, a hospital map was constructed in a Gazebo simulation.[12]

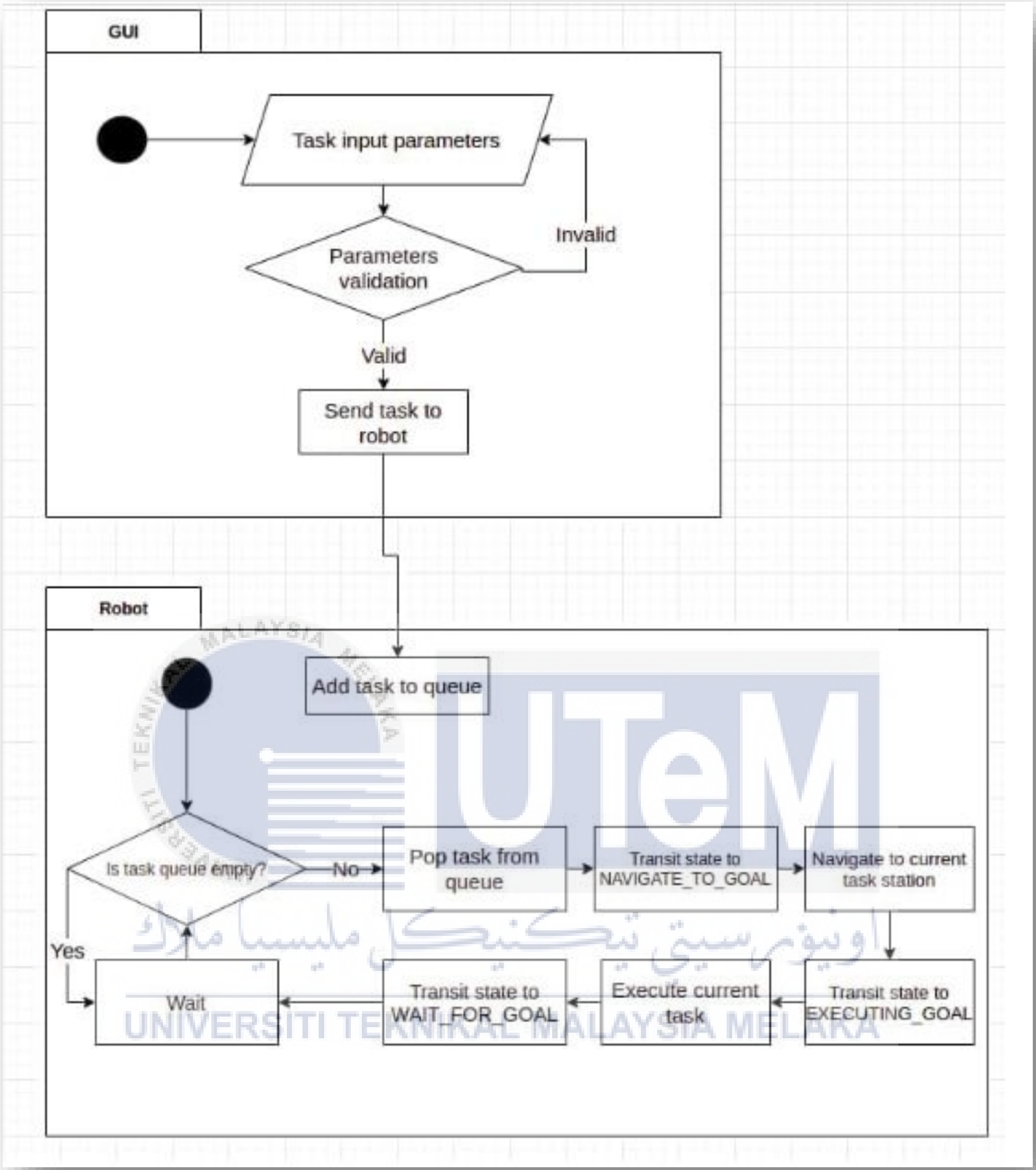


Figure 9 Block Diagram of proposed system FuF

Figure 10 TurtleBot3 Waffle is approaching the station in Gazebo simulation.

Mamatha KR et al. 2022 proposed a Smart AI Based Delivery Robot. A robot that moves automatically between two locations, requiring no human intervention. Obstacles are recognized using the cameras. The robot will halt if there are any impediments in its route, then resume moving once they have been removed. A self-driving system's fundamental premise is the existence of a robot that is capable of moving tangible items from one location to another. A robot that delivers deliveries is referred to as a self-contained delivery robot. Artificial intelligence is used by the self-driving robots, which arrive at their location after taking the right route. The robot is used to transport the packages secure [2]

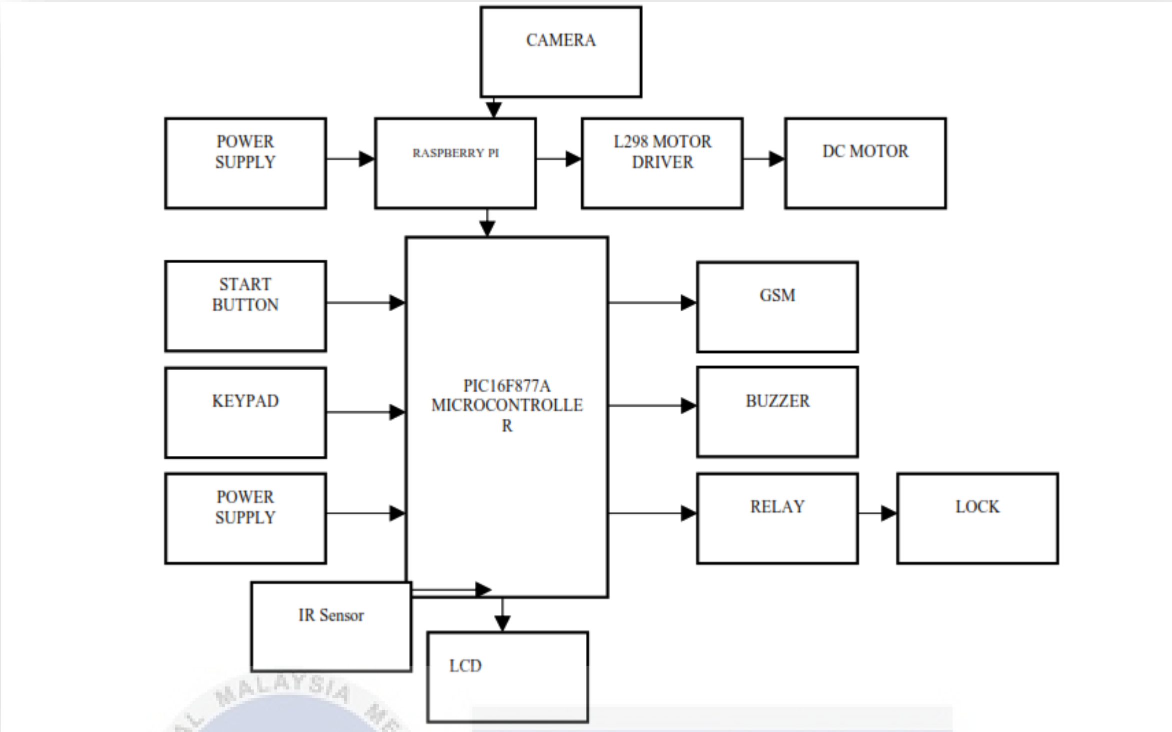


Figure 11 Block Diagram of Robotic Unit

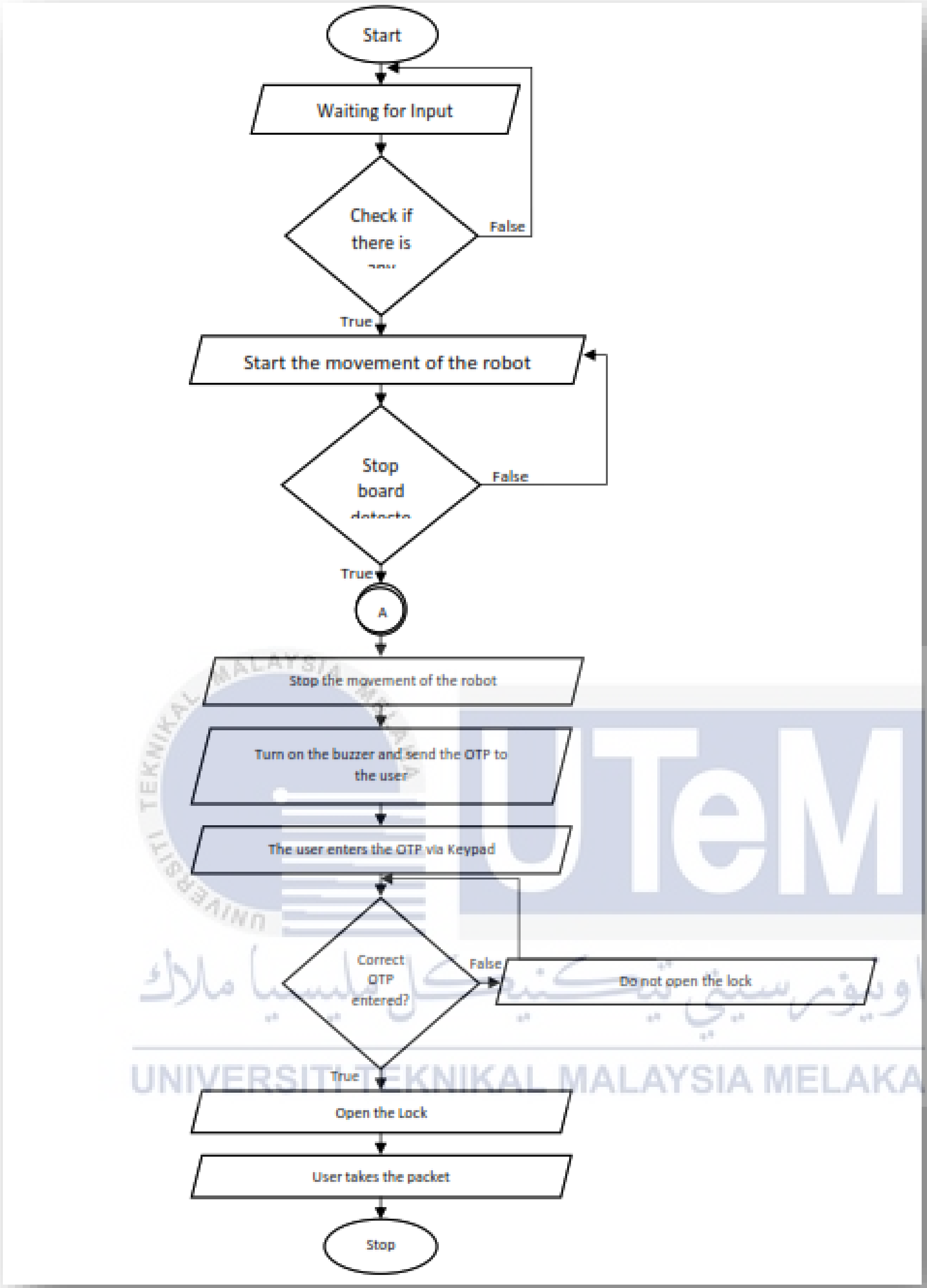


Figure 12 Flowchart of proposed project.

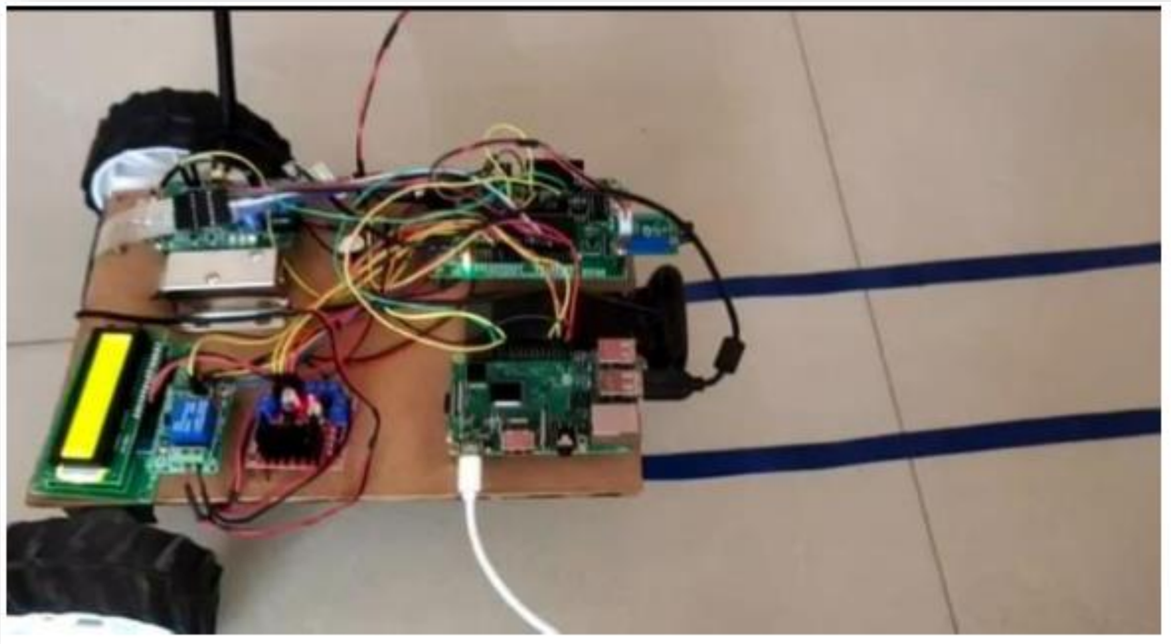


Figure 13 Model of proposed system following the line

Figure 14 Images showing the messages displayed on LCD

Sujay s n et al. 2021 proposed a Smart Box Delivery System for Faster And Safer Delivery. it is Intelligent and secure electronic locker systems are used for parcel transportation and collection. Allow residents of your multifamily housing to accept deliveries from any source at any moment. A flexible and expandable parcel gathering system created just for you is the smart locker. Local controls can be used to modify the features. We abide by all data security laws. A cloud-based management system designed for businesses makes sure that no resident data is kept on-site. The need to effectively handle online order delivery has increased with the rise of e-commerce. Nowhere is this more obvious than in residential buildings across the nation. During the busiest holiday seasons, package and parcel deliveries can easily rise by a factor of five. This places more responsibility on your staff as the owner or manager of a multi-family housing unit to handle these packages until they are picked up. In addition to the time and effort required to handle this process, you are also accountable for the package's security [13]

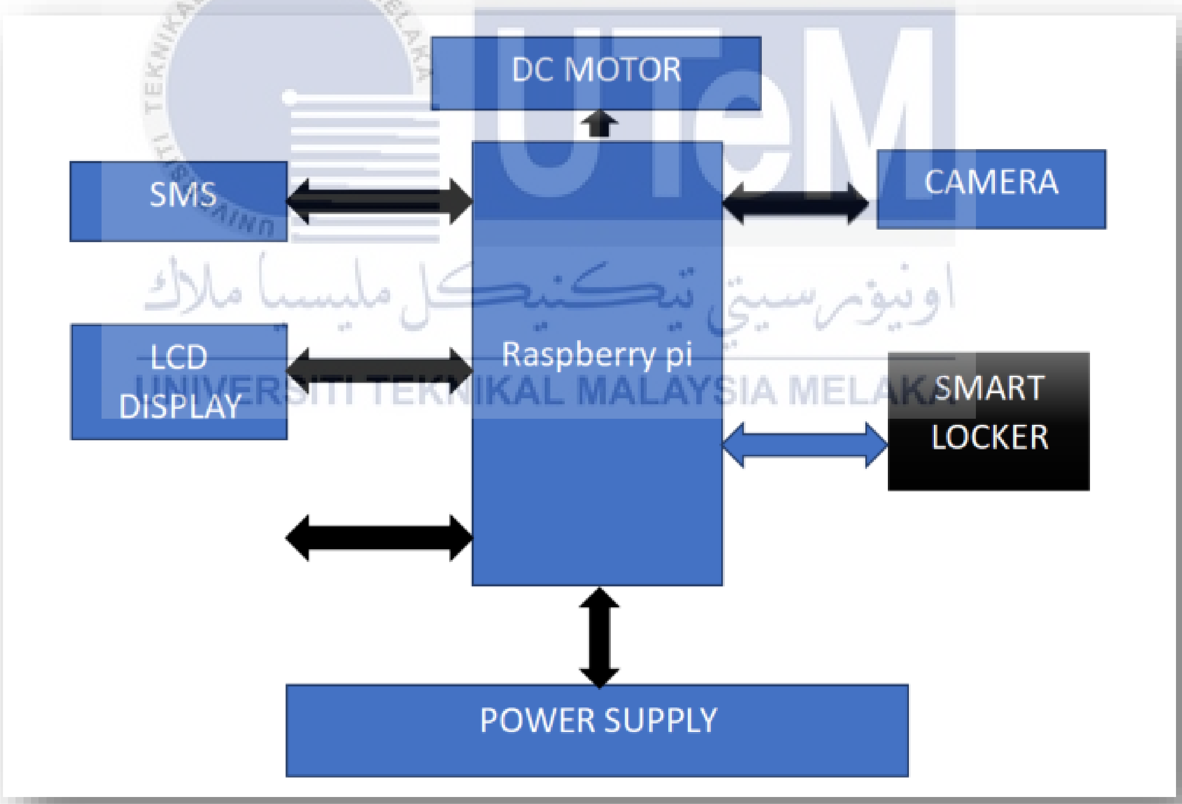


Figure 15 Block Diagram of proposed project

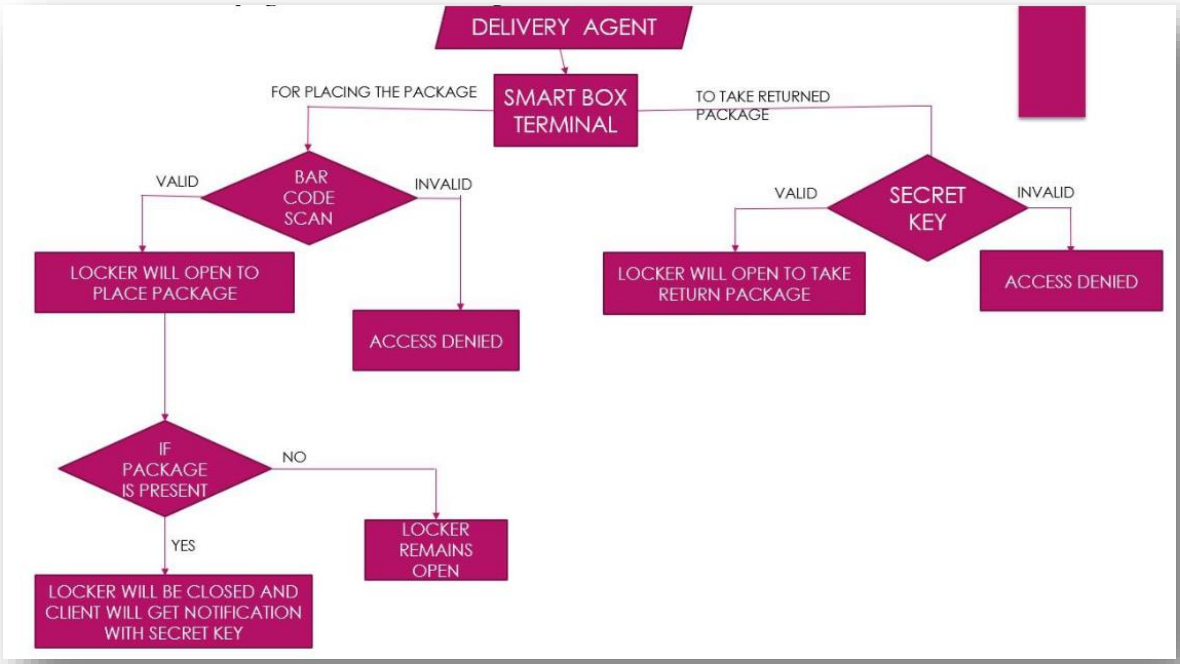


Figure 16 The flow chart of the delivery agent

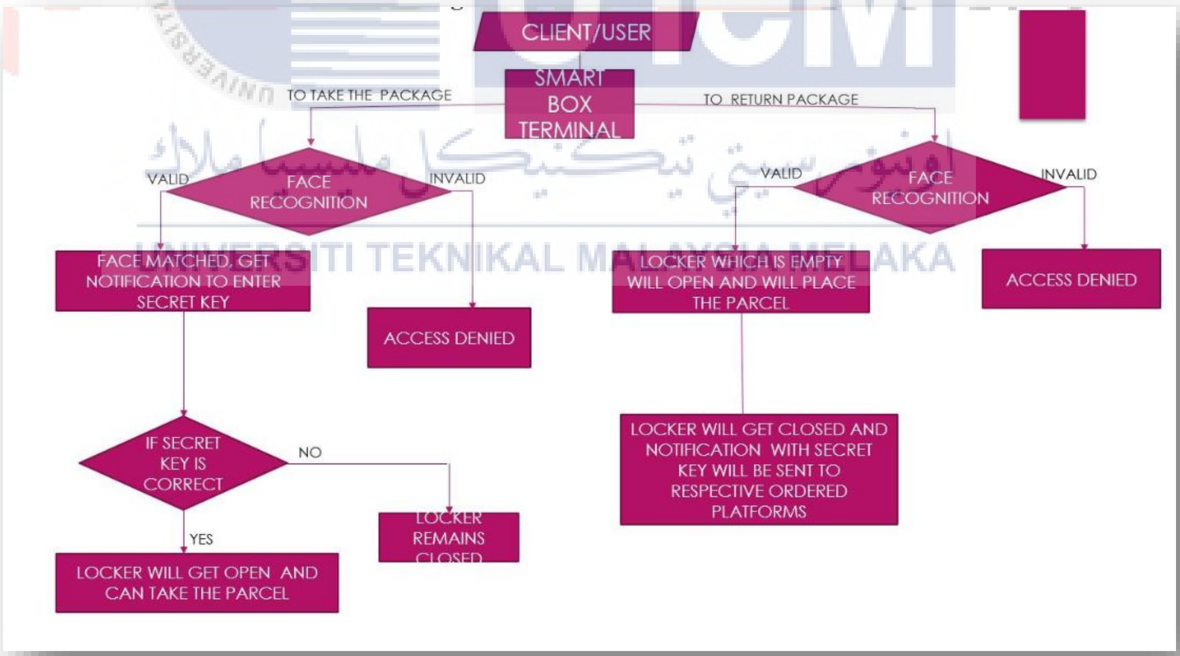


Figure 17 The flow chart of the client/user

Figure 18 Training Face Data Set

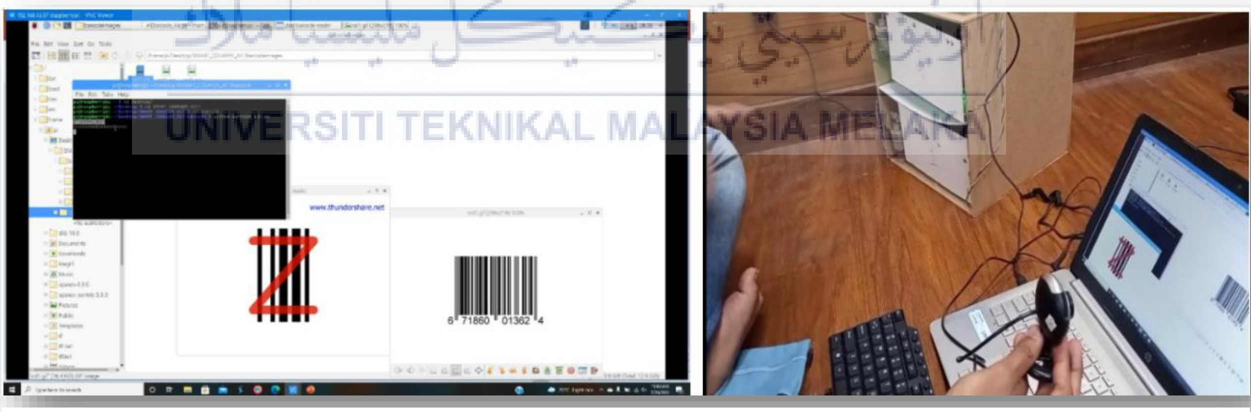


Figure 19 Barcode Scanning

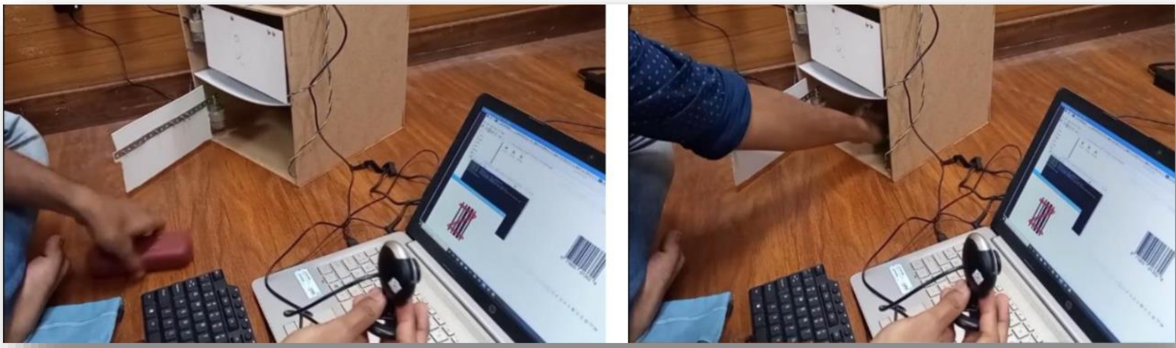


Figure 20 Opening and Placing Package in Smart Box

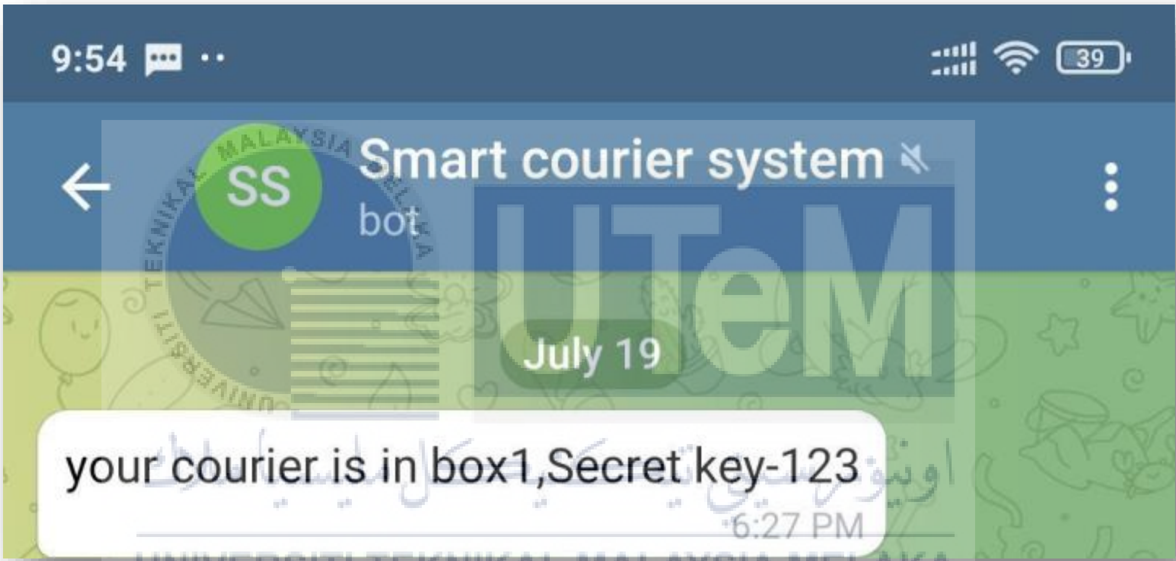


Figure 21 Client Receiving Notification about Parcel with Secret key

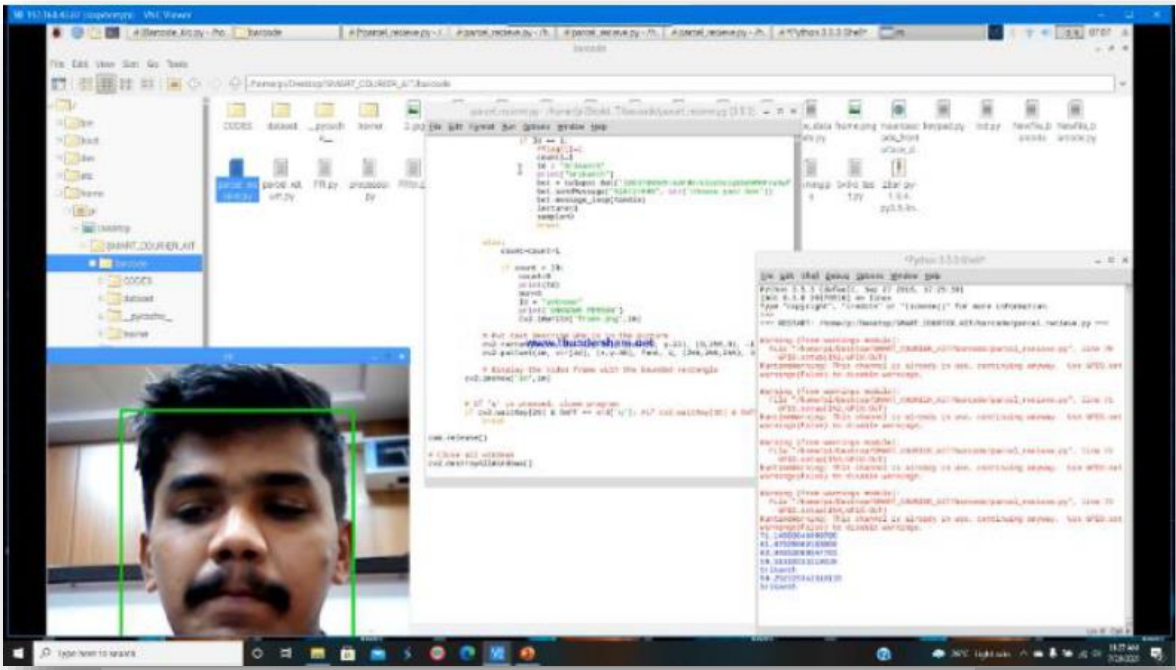


Figure 22 Face Verification

Figure 23 Client Entering Valid Secret Key

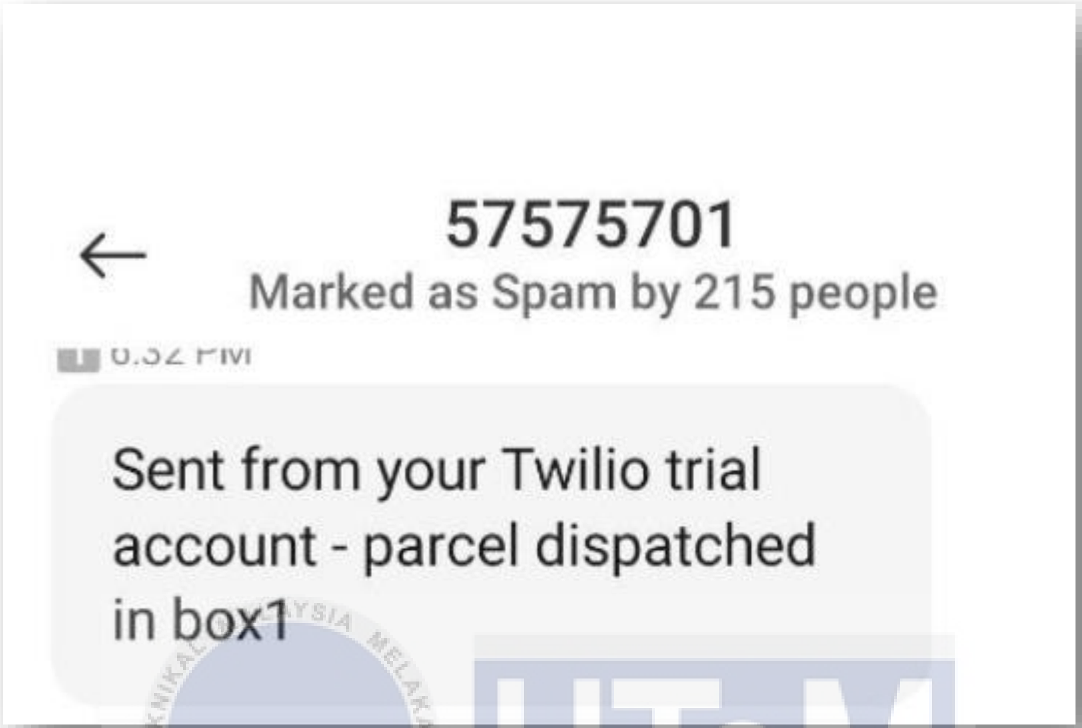


Figure 24 Delivery Agent Receiving Dispatched message

Figure 25 Someone trying to open smart box Forcefully

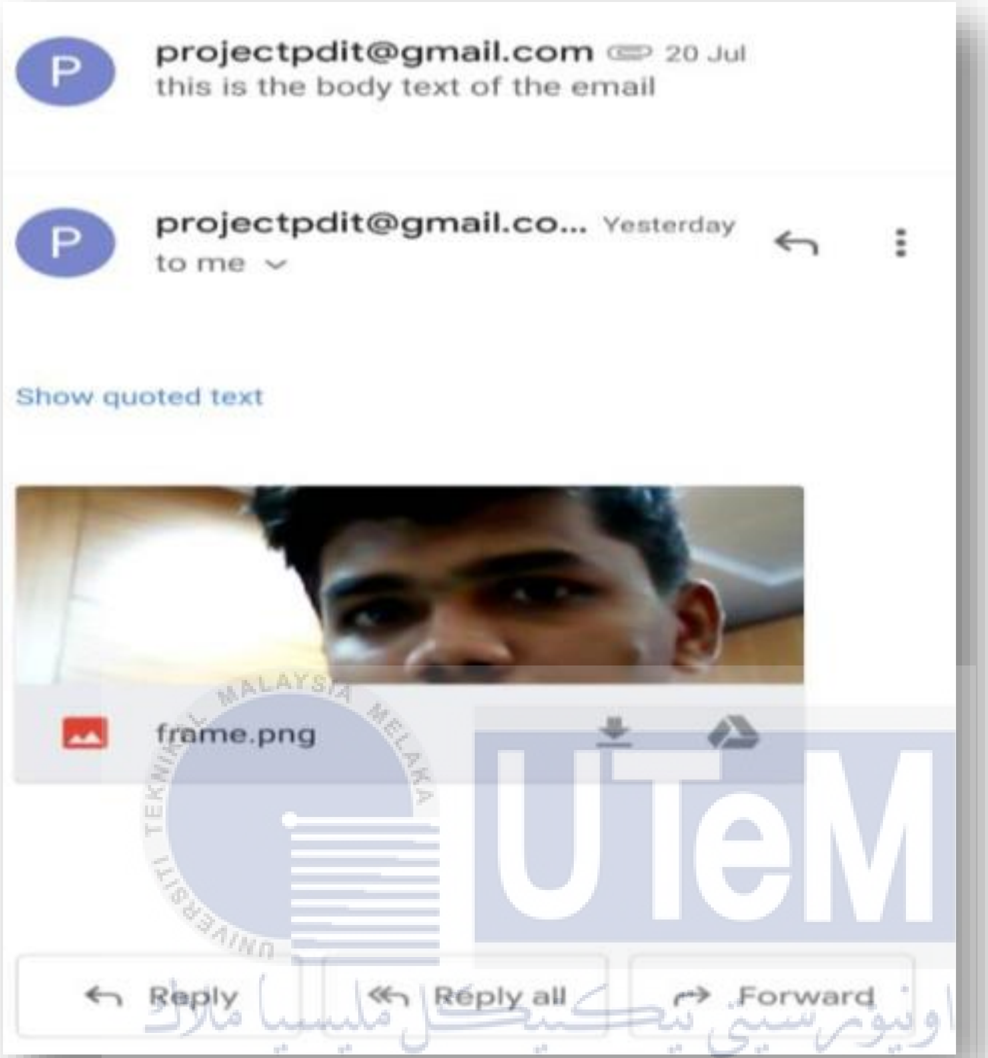


Figure 26 Mail attached with Snapshot during the theft.

Sreenivas Eeshwaroju et al 2021 proposed An IoT based Three-Dimensional Dynamic Drone Delivery (3D4). System Cities will have more high-rise structures due to the population growth. Additionally, the quick development of information and communication technology will call for clever solutions to address people's delivery needs. To facilitate vertical deliveries, this study suggests a "Three-Dimensional Dynamic Drone Delivery (3D)" system that expands on the traditional two-dimensional delivery methods with a third dimension (Z-axis). The suggested method gives the user the ability to accept shipments wherever they are needed. Additionally, this system improves customer satisfaction and addresses delivery issues like traditional fixed-point deliveries and package security [14]

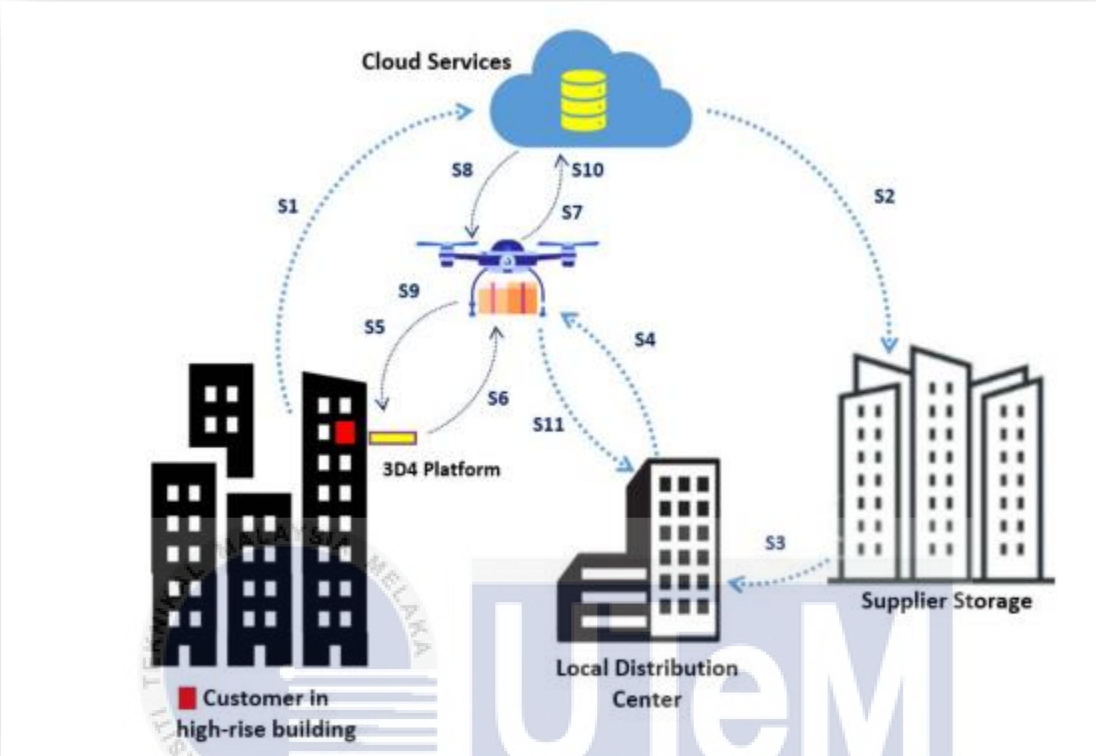


Figure 27 3D4 System Functional Overview Diagram

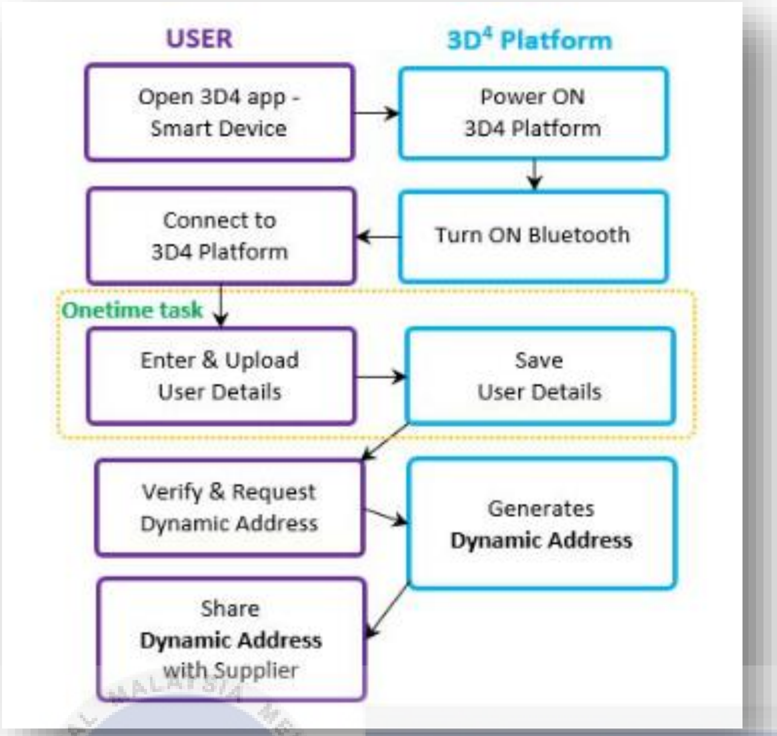


Figure 28 3D4 Platform Configuration Flow Diagram

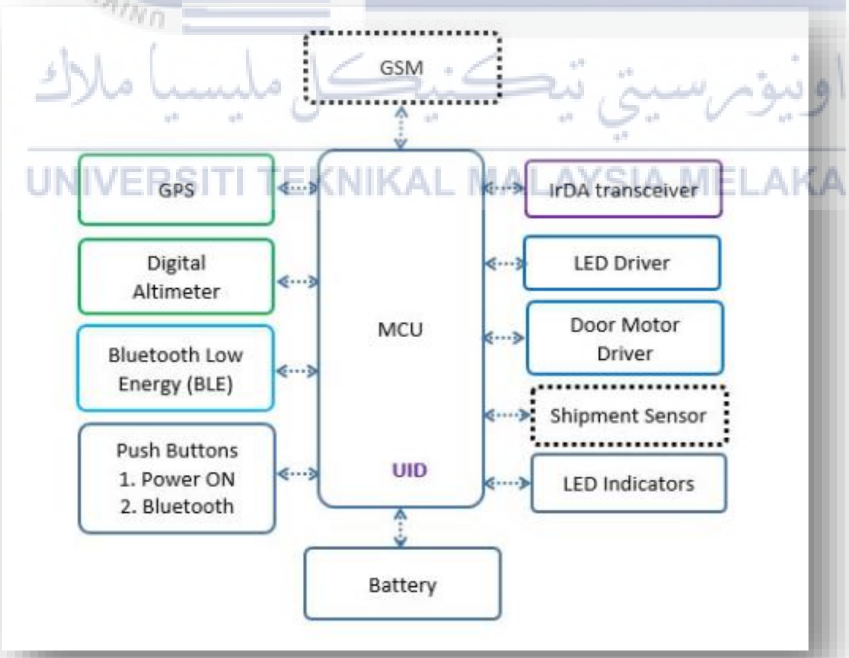


Figure 29 3D4 Platform Electrical Block Diagram

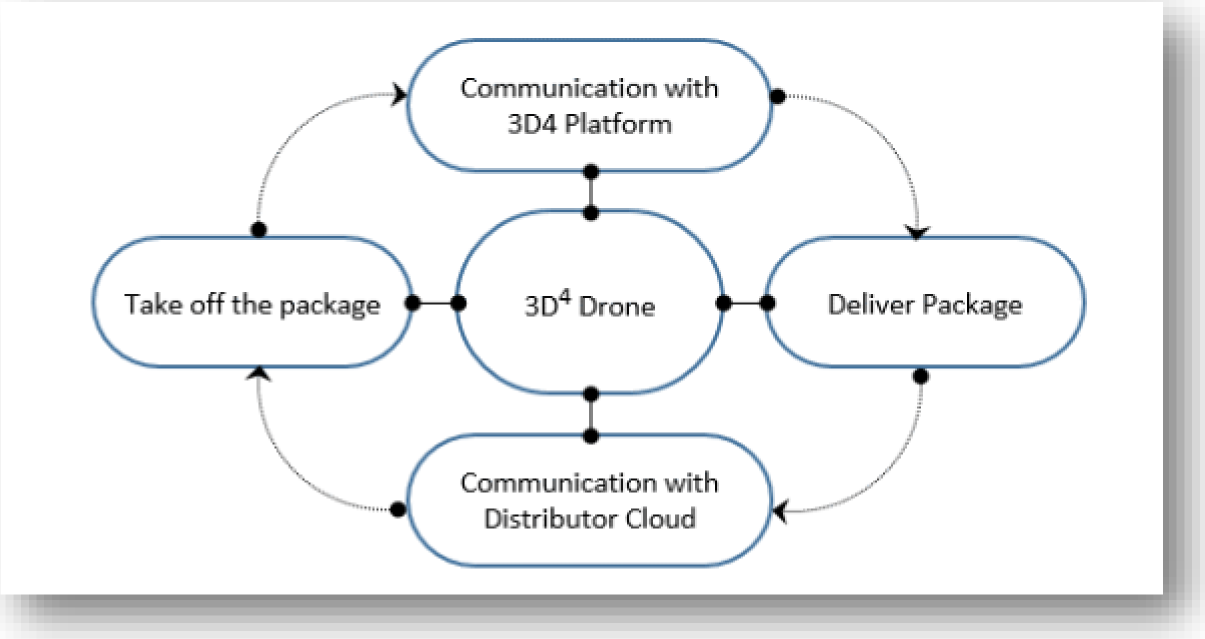


Figure 30 3D4 Drone Functional Block Diagram

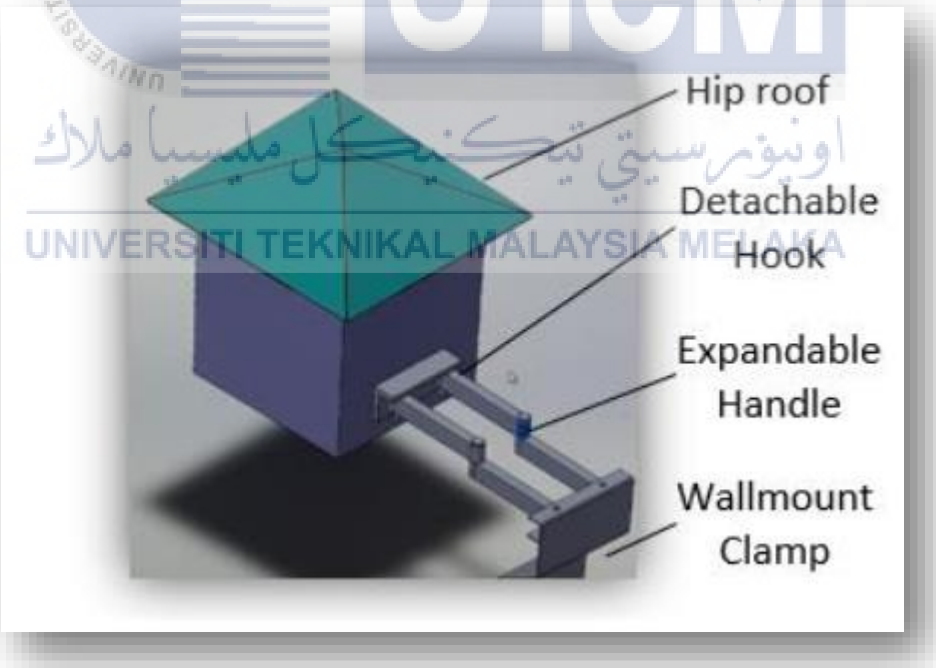


Figure 31 3D4 platform physical model

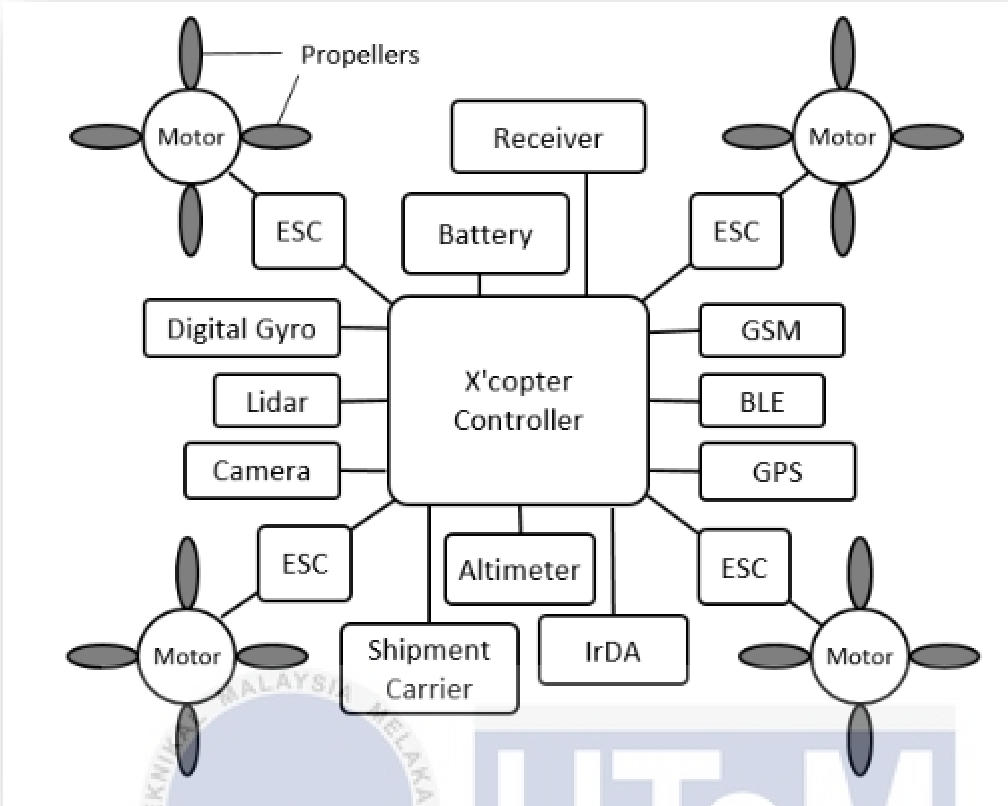


Figure 32 3D4 Drone Electrical Block Diagram

Fei Li et al 2013 proposed a system called Web-Scale Service Delivery for Smart Cities. A new methodology called "web-scale service delivery for smart cities" aims to provide open and scalable smart-city services by promoting cooperation between cloud and Internet of Things partners. In order to disrupt the traditional vertical service-delivery model, it takes advantage of the widespread distribution of computing resources and software services on the Web.[15]

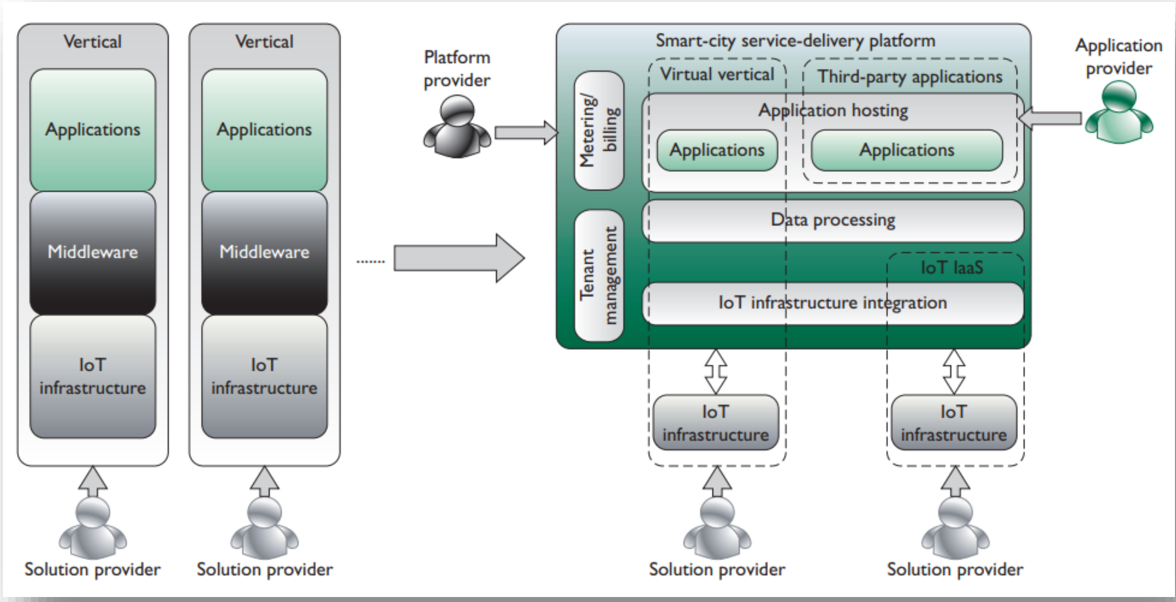


Figure 33 In smart cities, the emergence of platform providers for service delivery. The platform is a fresh take on a platform-as-a-service (PaaS) solution that combines IoT infrastructure with cloud and IoT resource on-demand utilization services for application

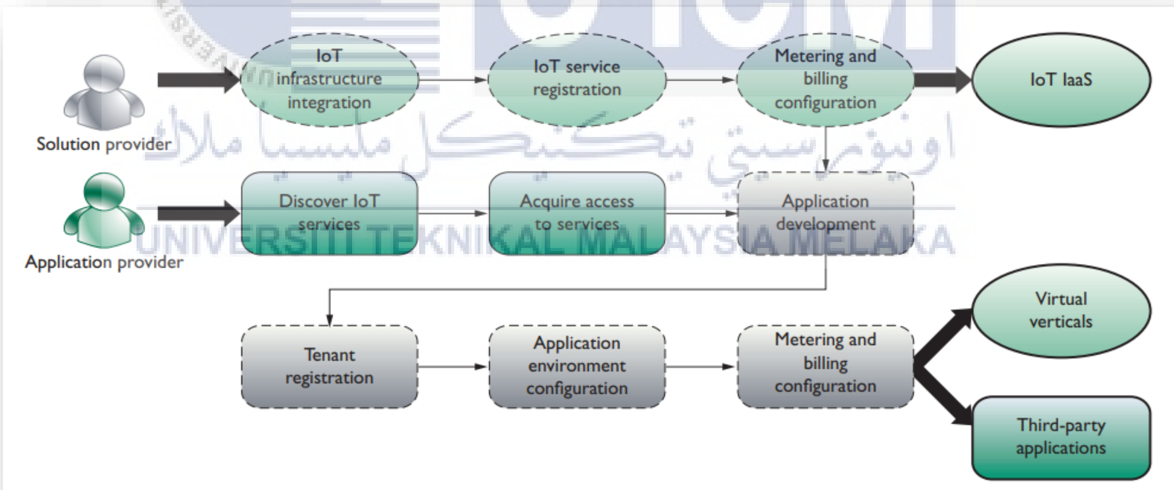


Figure 34 Service-delivery workflows. The workflows result in three service-delivery models — virtual verticals, third-party applications, and Internet of Things infrastructure as a service (IoT IaaS).

Wei Tu et al 2019 this project demonstrates a crowdsourced internet food delivery method. With the help of Internet-of-Things and 3G/4G/5G technology, crowdsourced workers can be attracted to deliver food using shared bicycles or electric motorbikes. We describe an online dynamic optimization framework that consists of procedures for order gathering, solution production, and sequential delivery. To assign food delivery duties and produce high-quality delivery routes in real time, a hybrid metaheuristic solution technique is created that combines the adaptive big neighborhood search and tabu search approaches. Different food providers dynamically share the crowdsourced riders. The effectiveness of the suggested approach is assessed using both simulated small-scale and actual large-scale on-demand food delivery situations. The outcomes show that the proposed crowdsourced food delivery strategy performs better than conventional urban logistics. In less than 120 seconds, the devised hybrid optimization method can generate reliable crowdsourced delivery routes. The outcomes show that the proposed online crowdsourced delivery approach can make on-demand food delivery possible at a city-wide scale [16]

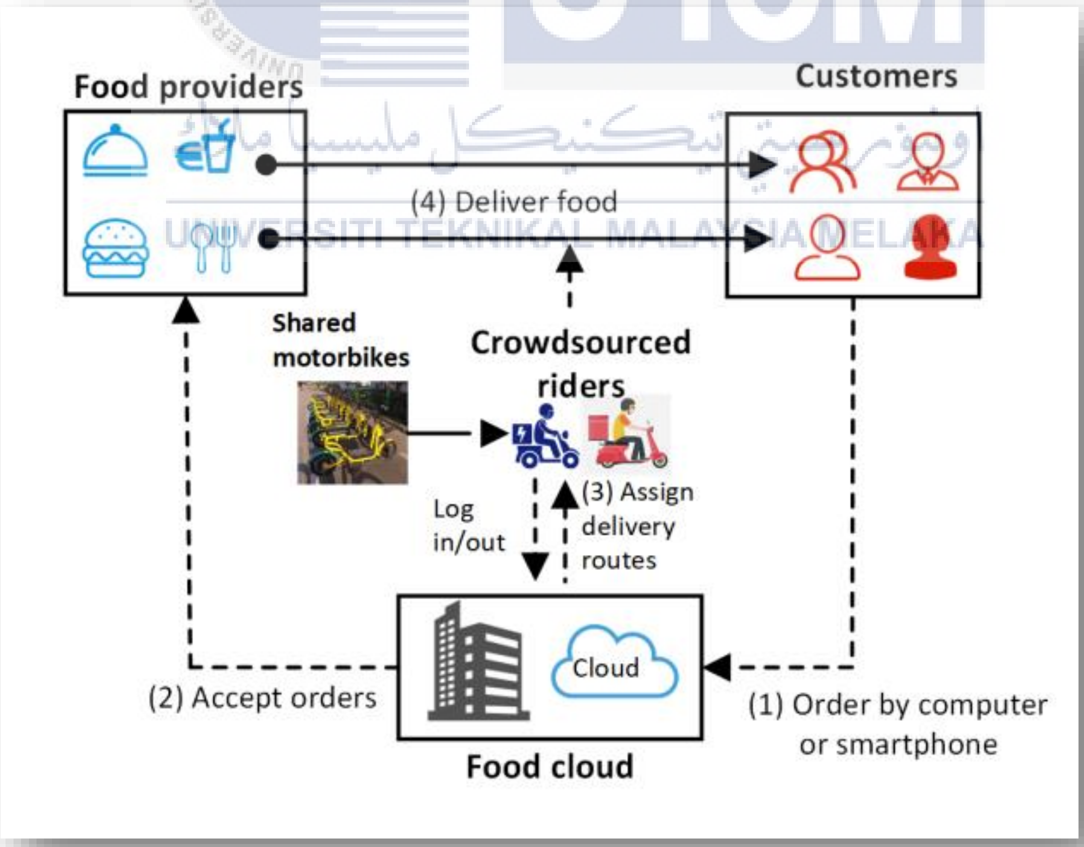


Figure 35 The crowdsourced on-demand food delivery system

B. Madani et al 2019 proposed a project called Autonomous Vehicles Delivery Systems Classification: Introducing a TSP With a Moving Depot. The challenge of getting items from distribution centers to end destinations (clients) in a timely manner has garnered more attention due to the massive growth of the e-commerce business. The chain of supply. Increasing supply chain performance and lowering operational costs or ecological impact are two major obstacles facing the Last Mile Delivery (LMD) dilemma. By creating new delivery systems to replace conventional delivery methods (delivery person, postal/delivery boxes), new technologies like drones, robots, and the internet of things assist address these difficulties. However, utilizing these technologies poses additional operational difficulties. For instance, in a truck-drone delivery system, the vehicle acts as a depot where the merchandise is loaded before being flown to the clients. In contrast to a standard Vehicle Routing Problem (VRP) where we always assume a stationary depot, the depot is now moving. This study examines how using autonomous vehicles (AV) will affect the LMD's logistical setup. In order to classify delivery systems based on the handover of packages or products at the time of the final handling before delivery to clients, we first give a technological assessment of the usage of AVs in logistics. In this article, we discuss the three types of handovers: machine-to-person, machine-to-machine, and person-to-machine. This study develops two iterations of a discrete traveling salesman problem (TSP-MD) with a moving depot and introduces a new class of vehicle routing issues with a moving depot. Using the program General Algebraic Modeling System (GAMS), we resolve the issue and talk about some of its features.[17]

R.Sathish Kumar et al 2017 proposed a project called Predicting Shortest Path for goods delivery to fair price shops in India. One of the main objectives of the development of smart cities is to design a smart system for efficiently delivering items to diverse fair-price stores. The smart city environment is viewed as a dispersed environment for moving goods between urban areas in this article. Thus, to compute the shortest path, the widely used Dijkstra's method is implemented as a MapReduce model utilizing the Hadoop environment. afor quick product delivery. A front end is made to convey goods both inside and between states. For the simulation, a distance matrix made up of 50, 100, 150, etc., 500 cities is taken into account. The experiment shows that the Dijkstra's algorithm used in the map-reduce model achieves the goal of delivering the materials to the correct location in the shortest amount of time. Other well-known algorithms as the Bellman-Ford, Throup, and Gobow algorithms are used to compare the performance of the proposed MapReduce Dijkstra's algorithm. The proposed MapReduce Dijkstra's algorithm is found to provide a trustworthy shortest path between any source and destination city while using less CPU time than the alternative algorithms.[18]

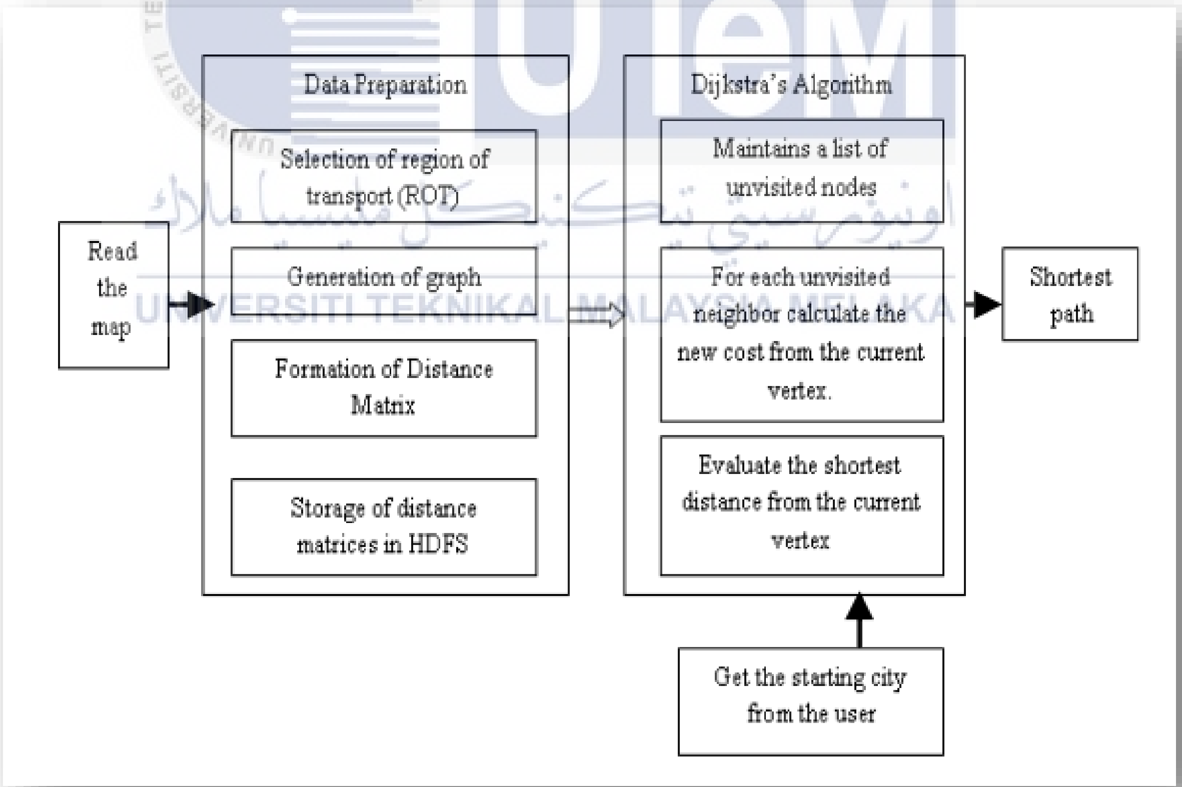


Figure 36 Architecure of Smart Logistics Transport System

2.3 Line Following Robot Project System.

Hasan U. Zaman et al 2016 proposed a Novel Design of Line Following Robot with Multifarious Function ability In this paper, the design and execution of a robot that follows a line—specifically, a black line on a white surface—are described. There are two distinct settings for this line-following robot, including line-following mode and obstacle detection mode. It is comparable to a self-driving robot vehicle. While following its intended route on its pathway, this robot can detect obstacles to the right, left, and in front of it. Overall, it can be said that the robot has the ability to travel along a black line while sensing obstacles from three directions. The creation of the design using Arduino Uno requires knowledge of Arduino programming, the integration of electrical circuits with the designed coding, as well as some architectural and fundamental mechanical engineering knowledge. For improved obstacle detection, the robot has three sonars linked in three different directions, and two infrared sensors are connected via a motor driver IC. (L293D). A 6-volt rechargeable lithium battery powers this follower with numerous modes. It costs 4500 BDT (Bangladeshi Taka), 4253.15 INR (Indian Rupees), and 64.136 USD (United States Dollars) to purchase the automaton in its entirety. The robot can carry up to 600 grams of component above and recognize lines and obstacles with ease, changing its course to follow the intended path.[19]

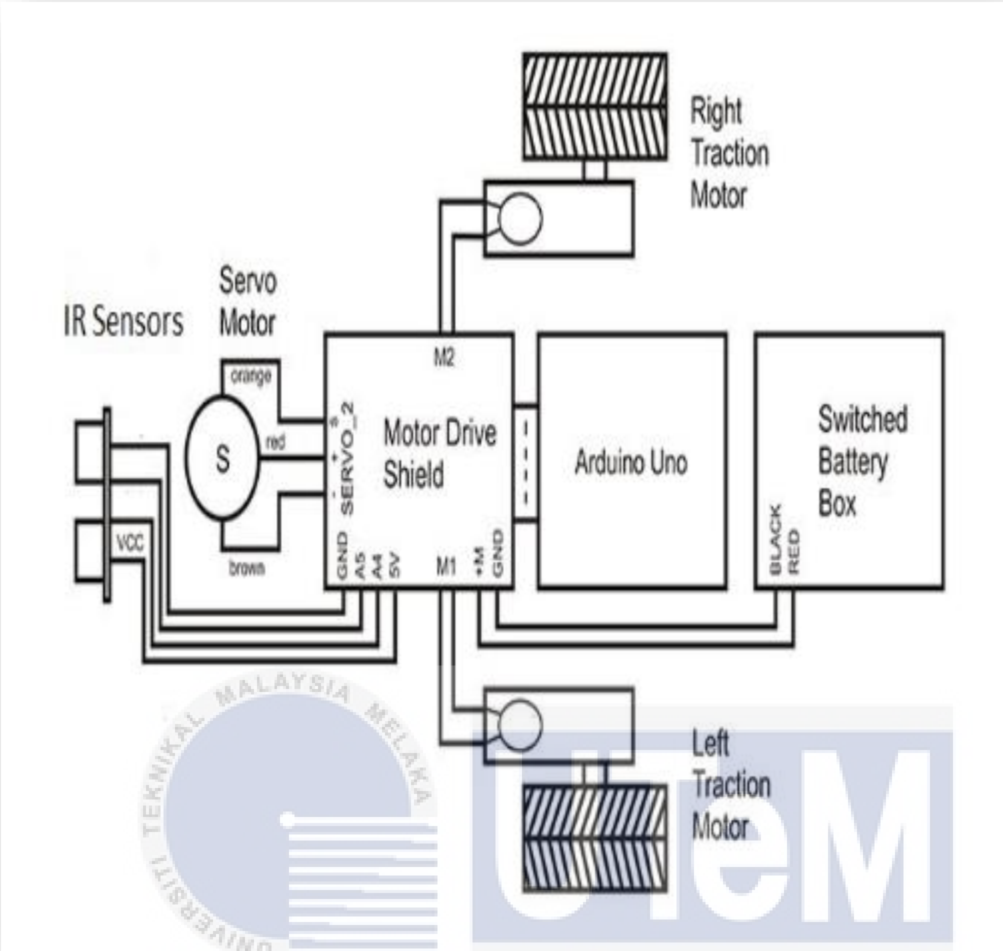


Figure 37 Block diagram of a line follower robot

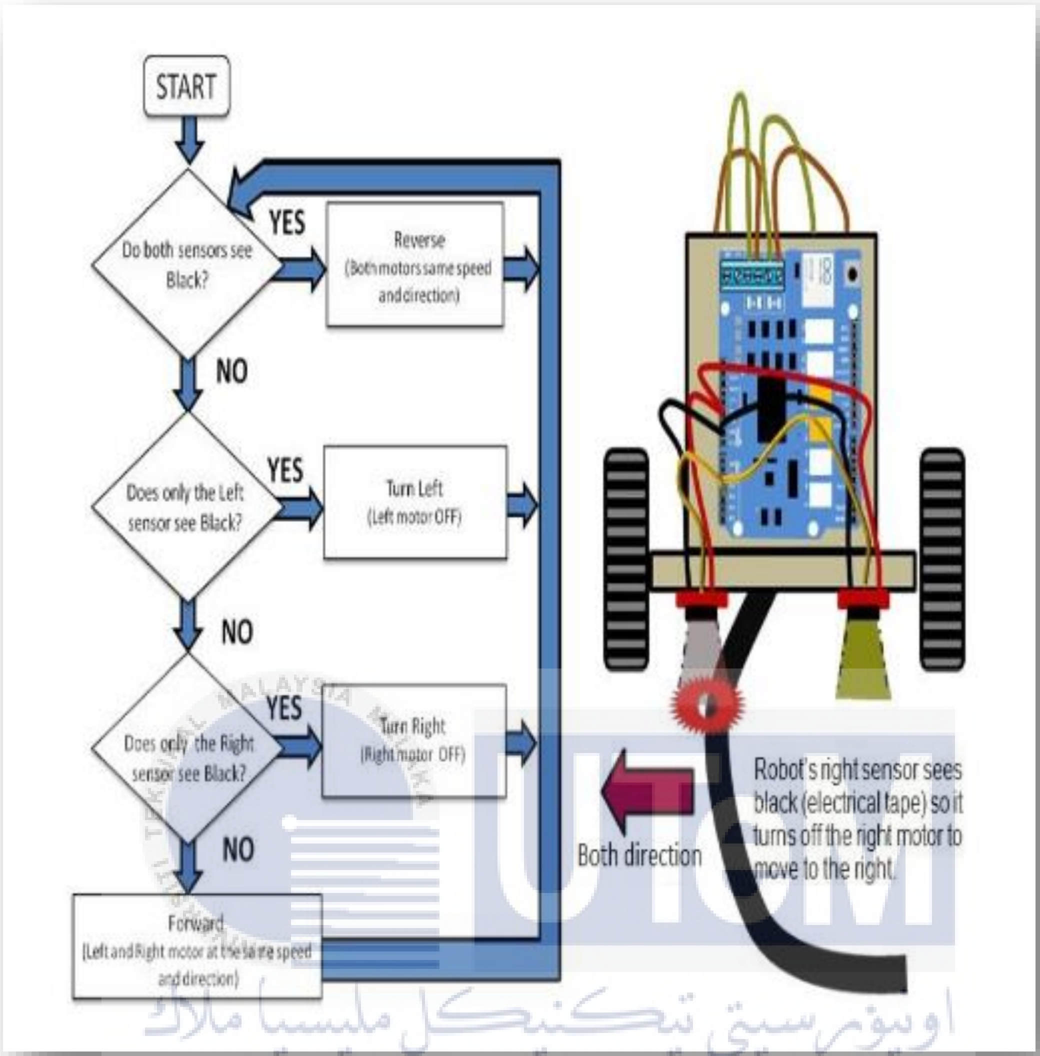


Figure 38 Flow chart of line follower

Milan Shah et al 2015 proposed a system which Design Implementation of High-Performance Line Following Robot. Robotics are being used more and more frequently today in hospitals, medical facilities, signals for microcontroller depending on their position over farming, in the military, on factory floors, and in every other area. Robots that follow lines are among the most common. A line-following robot is essentially a mobile autonomous device that tracks a line that is a different color from its surroundings. Thus, how well these line-following robots can distinguish the line from the background has a significant impact on how well they work. Around the globe, a lot of line-following robot-based robotic competitions are organized at both the college and professional levels. As we developed the line following system for the robot for ROBOCON 2016 (an international robotics event), we ran into a number of issues. In this paper, we discuss these issues, how we overcame them, and how we created the most optimized, effective, and high performance line following system.[20]

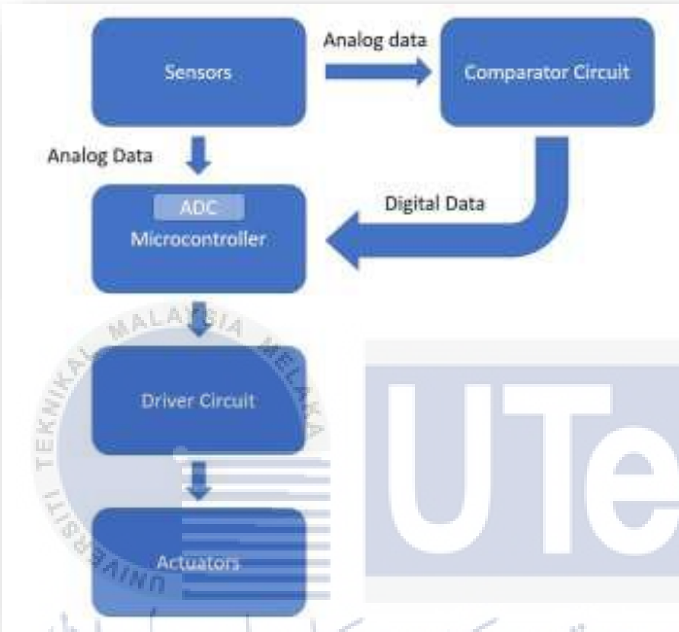


Figure 39 Schematic Block Diagram of Basic Line Following Robot

Mohammed Abdul Kader et al 2018 proposed a Line Following Autonomous Office Assistant Robot with PID Algorithm. In this paper, a line-following robot for office assistance is created. The robot can safely move paper documents, tea snacks, and other office supplies from one table to another within an office on its own, following user instructions. By hitting a PUSH button from any table, anyone can summon the robot. In response to a call, the robot will follow a predetermined route to the caller table and will automatically open a locker it has been given. The caller then put the document or other items in the container and could move the robot to a different table by pushing a different push button. To guarantee security, the locker won't open until the robot arrives at the designated table. Additionally, the robot has obstacle detection capabilities and can alarm if something blocks its route. The PID method is used by the robot to smoothly follow the route. Microcontroller, IR sensor module, RF Tx-Rx module, ultrasonic sensor, buzzer, and DC motor are the primary parts used to build the robot. Every workplace needs a peon to move the files or other materials, but they also have other duties to complete. This robot can help office workers by lightening their burden and allowing them to focus on more important tasks [21]

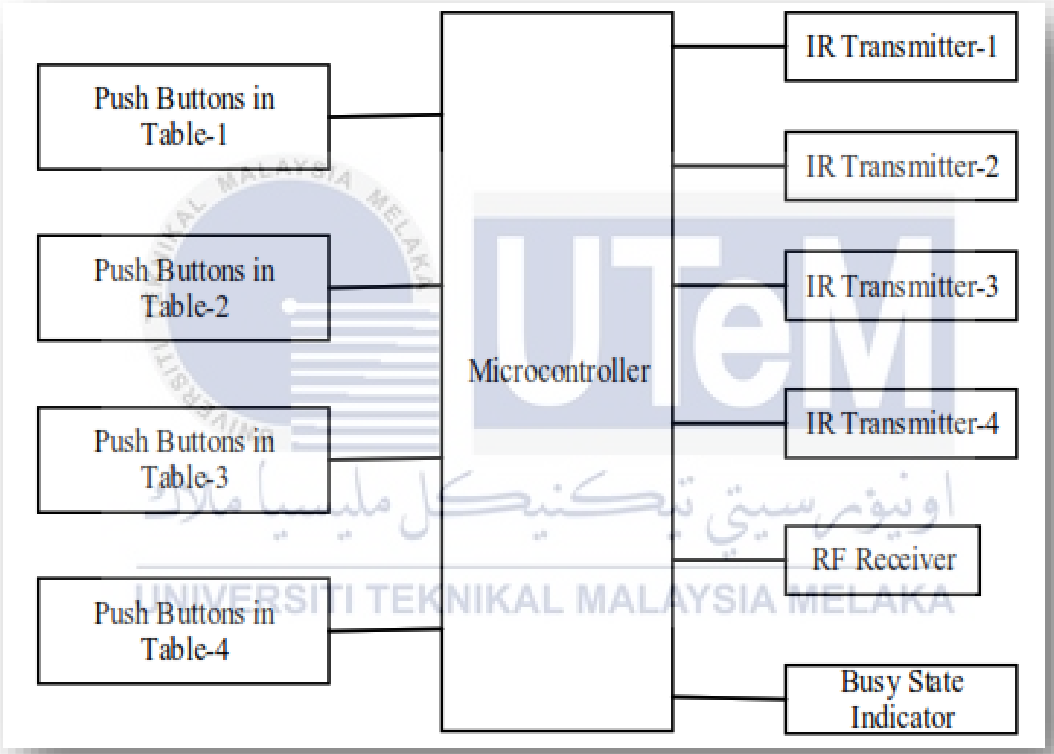


Figure 40 Block diagram of calling circuit

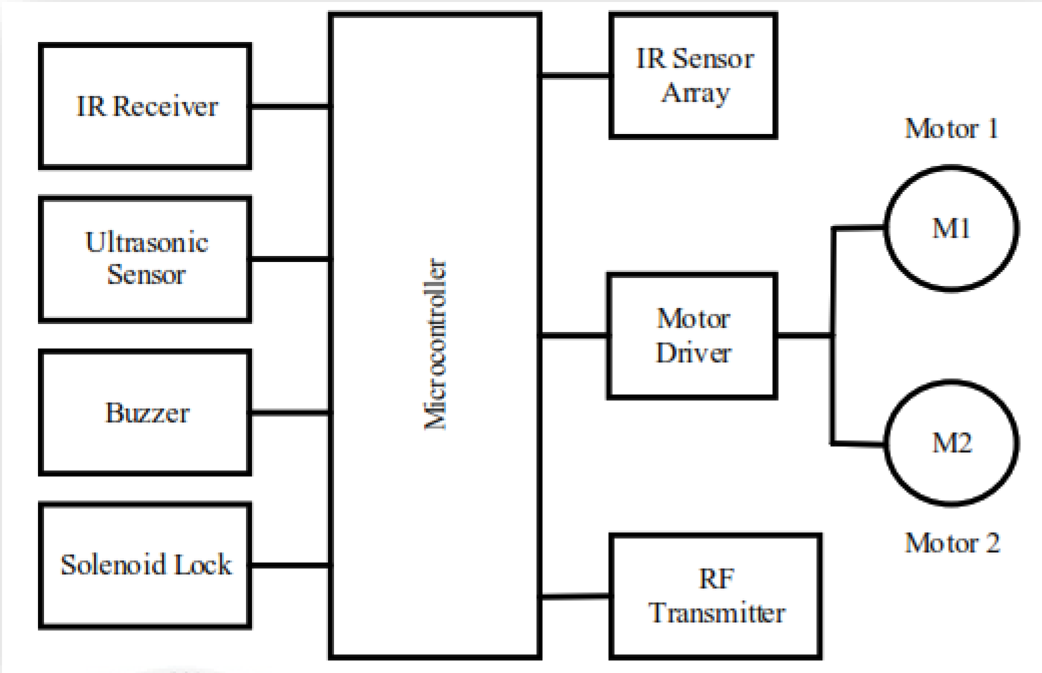


Figure 41 Block diagram of robot circuit.

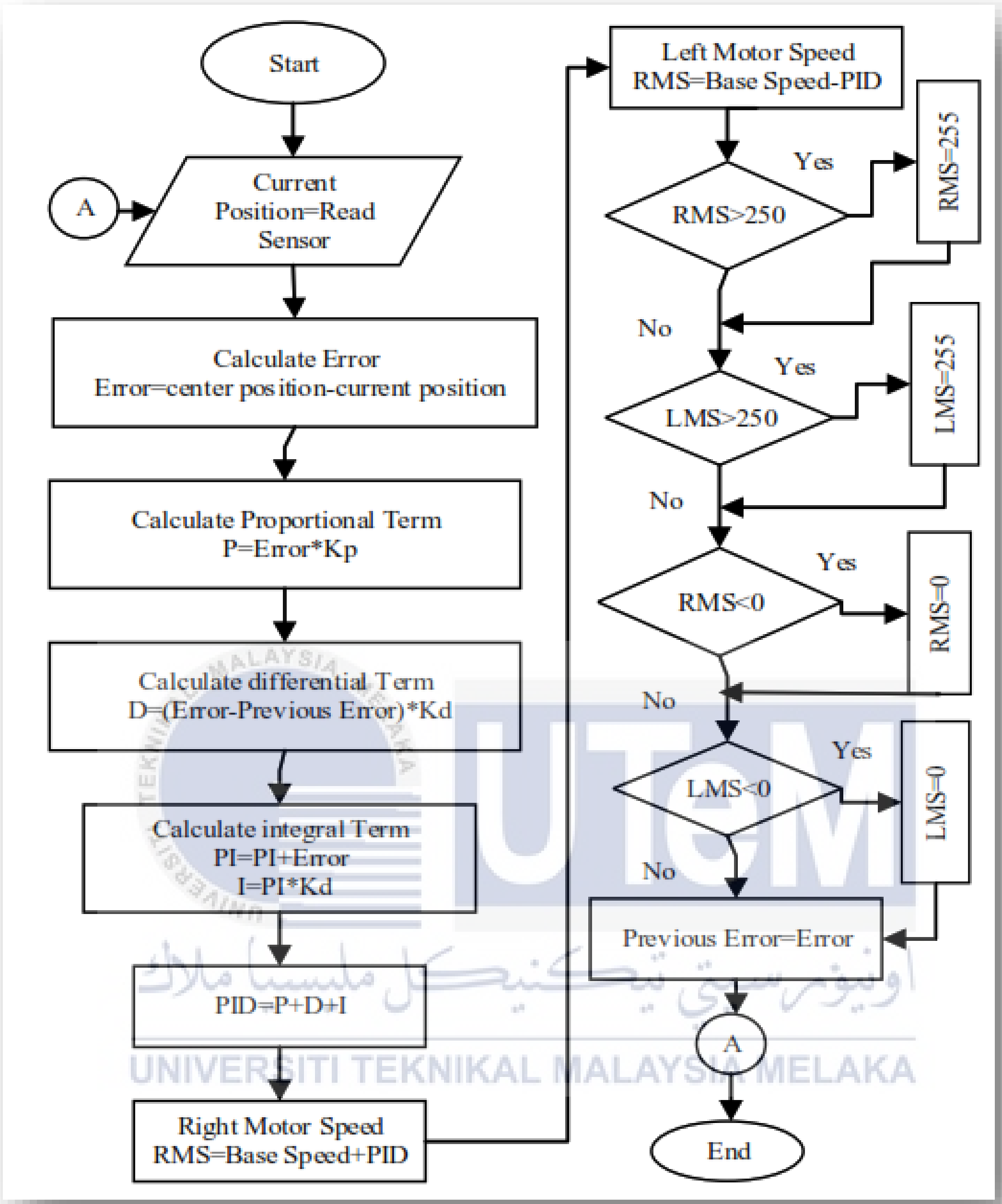


Figure 42 Flow chart of calculating motor speed by PID algorithm.

Endrowednes Kuantama et al 2014 proposed a Simple Delivery Robot System Based on Line Mapping Method. Because of the robot can transport items to another location as long as it is on the same level and has line mapping, the simple delivery robot system was created to make daily tasks simpler. The robot used in this study differs from previous line follower studies in that it is 40 cm long, 40 cm wide, and 40 cm high, can be called from multiple rooms using an RF remote, and can be instructed to transport items and send messages. The line is within 1.5 centimeters of the robot's body and is traced using seven infrared sensors. The robot has four rooms it can access specifically for this study, in addition to a main camp. The robot also has a number of other features, such as an alarm system that can let the user know when it has reached its location, an LCD and keyboard so they can type messages, an obstacle sensor to prevent collisions, and an emergency system that will activate if the robot veers off the path. The robot can be remotely controlled to be positioned back in the line map after the emergency system triggers the video camera and alarm. The AT89C51 microprocessor is in charge of all operations. This study concentrates on robot mapping systems that use line mapping, special remote systems that use radio frequency, message delivery systems that use LCDs, and emergency systems that use remote controls and video cameras. The robot has a top speed of 1.045 m/s and a maximum weight capacity of 3.5 kg.[5]

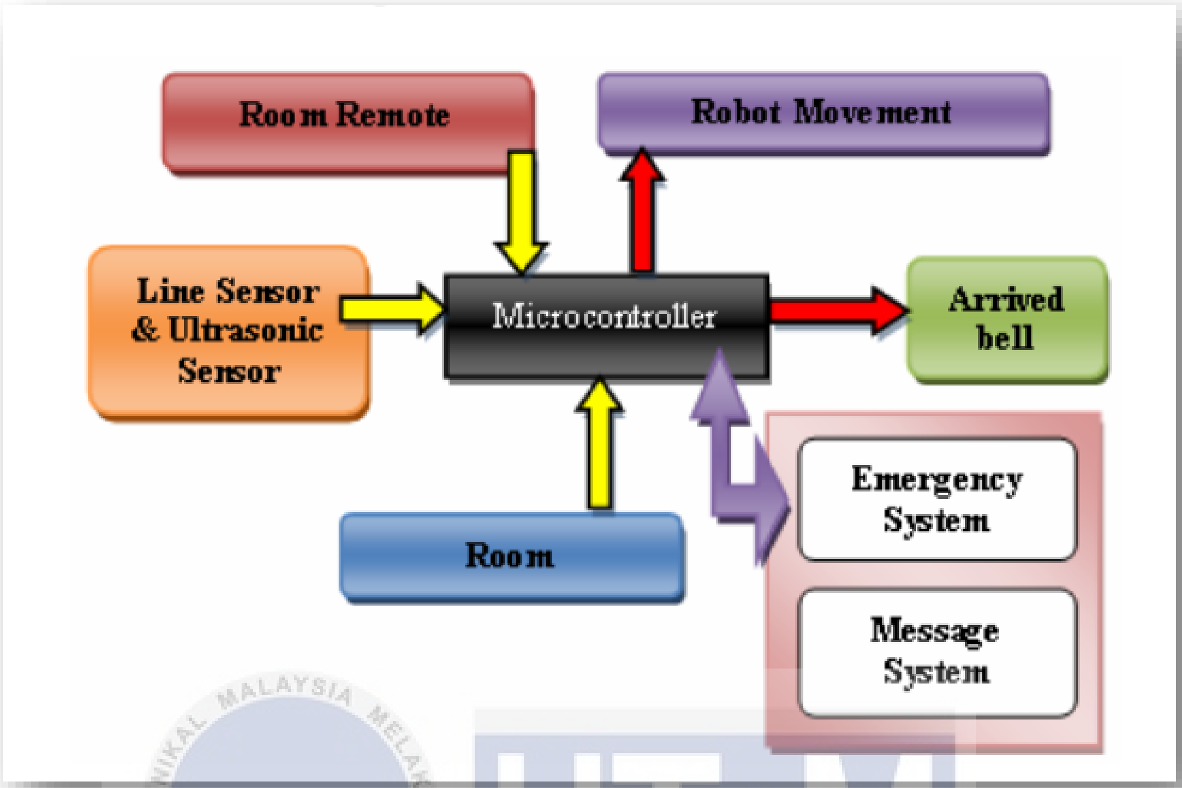


Figure 43 Flow diagram for delivery robot

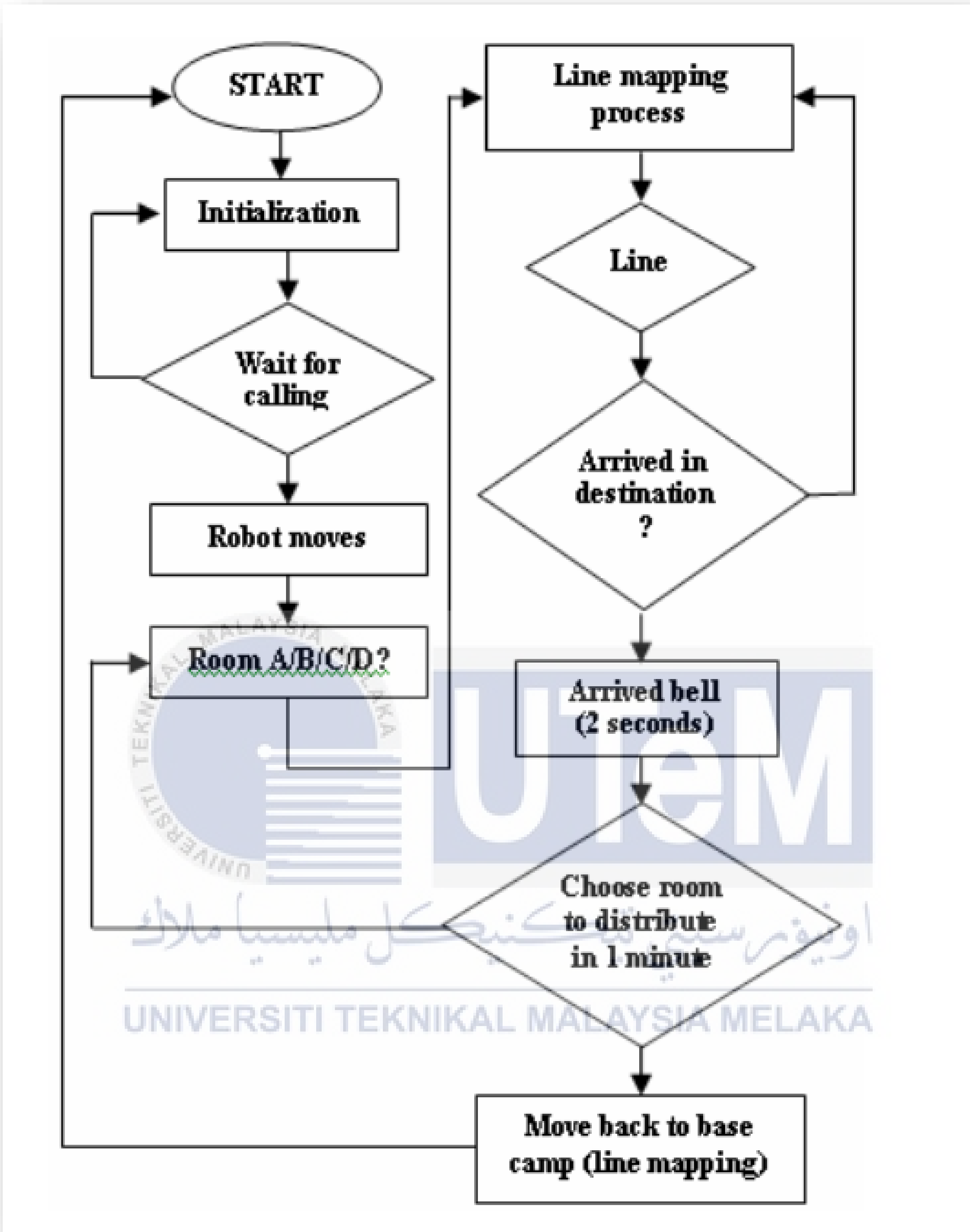


Figure 44 Flowchart microcontroller for line mapping.

Nor Maniha Abdul Ghani et al 2011 proposed a Two Wheels Balancing Robot with Line Following Capability. The creation of a line follower algorithm for a two-wheeled balancing robot is the main goal of this project. In this project, the ATMEGA32 is selected as the brain board controller to respond to the data received from the balance processor chip on the balance board in order to monitor environmental changes using two infrared distance sensors and address the inclination angle issue. Due to the application of internal PID algorithms at the balance board, the system will consequently instantly return to the set point (balance position). To create a seamless line follower robot, infrared light sensors used in conjunction with PID control are essential. It is able to create a dynamically stabilized balancing robot with line follower function by combining a line follower software with internal self-balancing algorithms.[22]

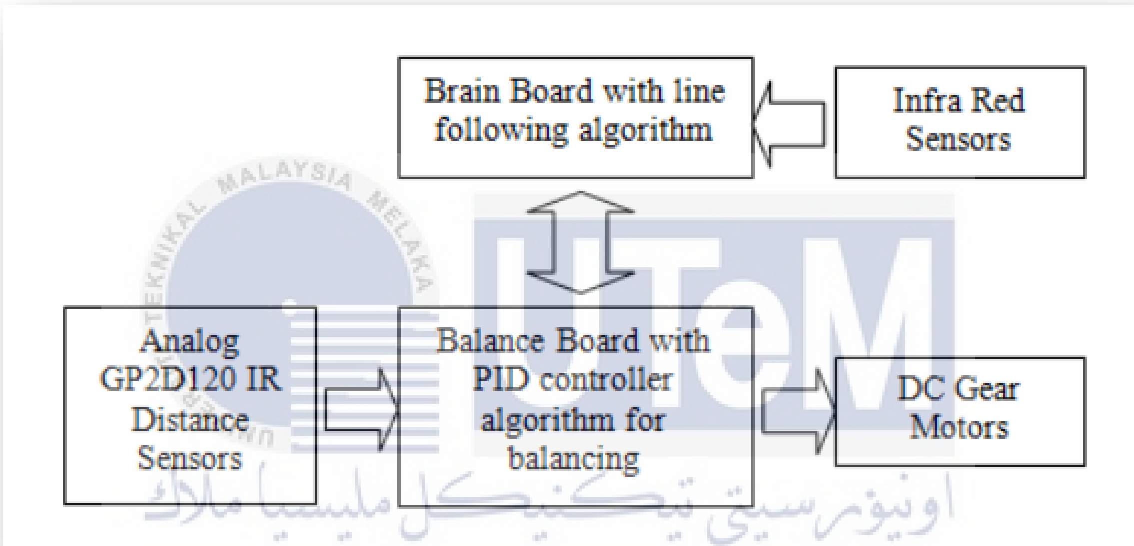


Figure 45 Robot's General System Block Diagram

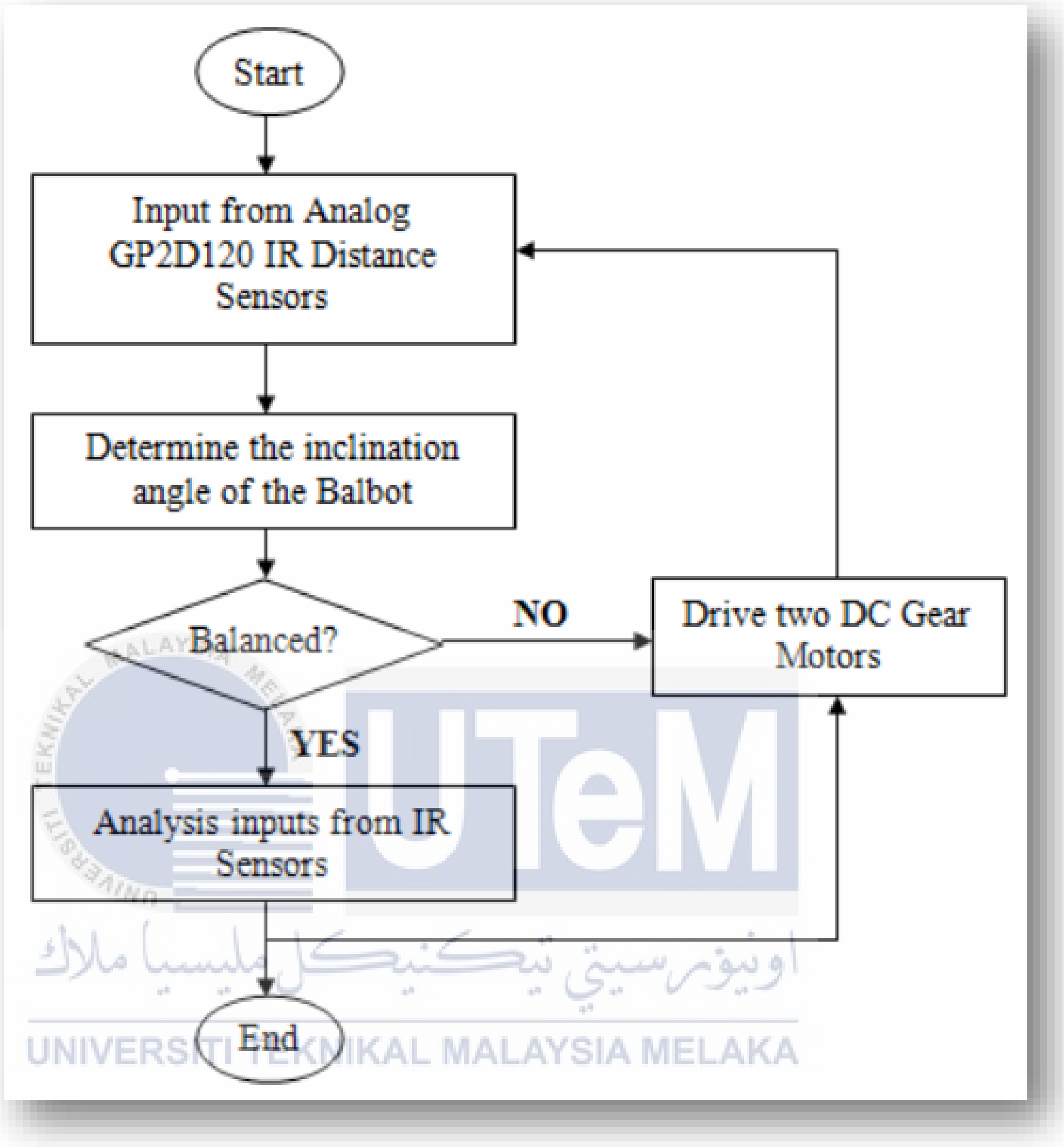


Figure 46 Robot's Internal System Flow

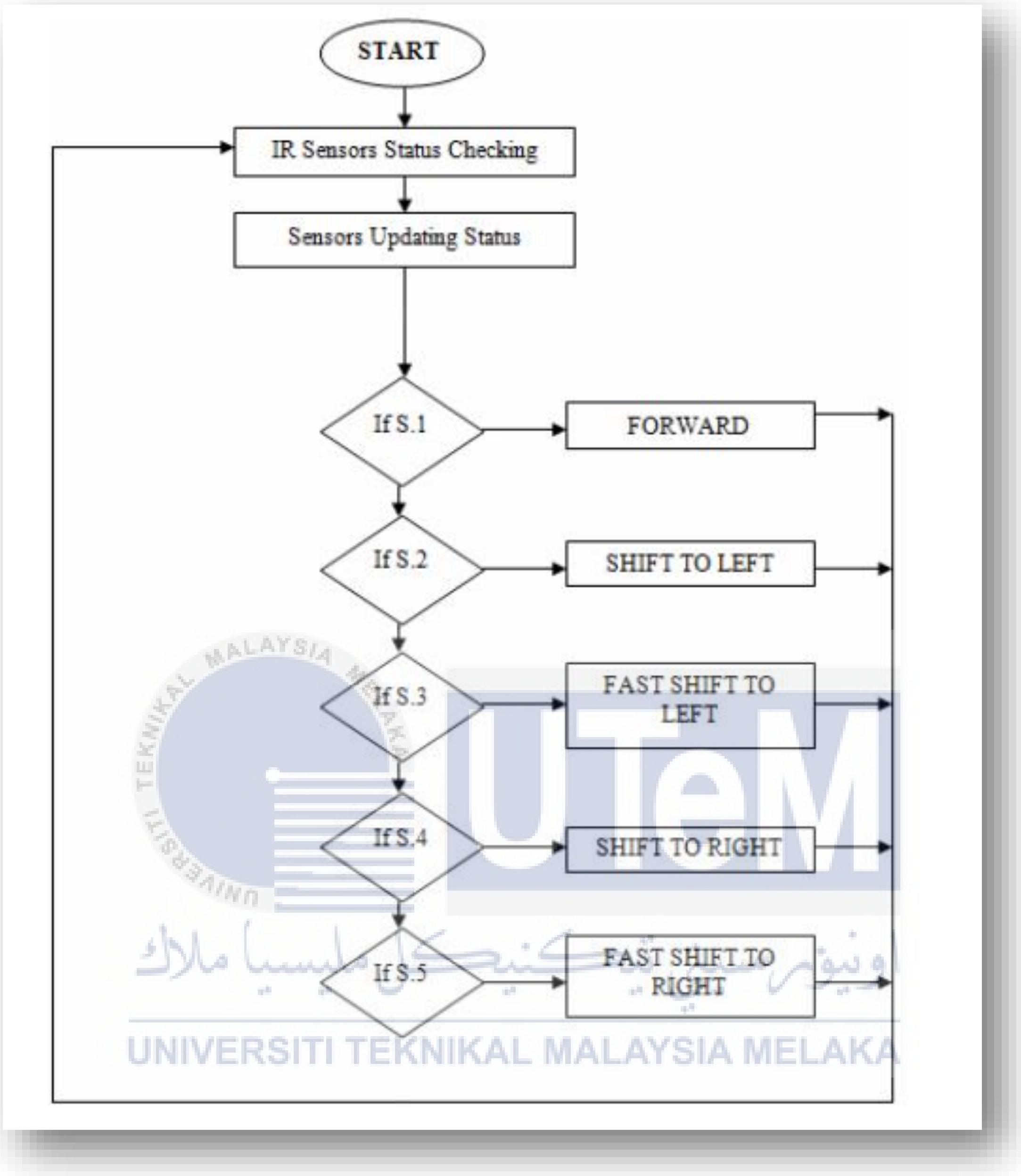


Figure 47 Flow Chart for the Line Following System

Alfian Ma'arif et al. 2020 proposed a Vision-Based Line Following Robot in Webots. A robot with robotic vision can gather data by having a camera analyze images. The line-following robot's camera is intended to identify image-based lines and guide the robot along the path. In this paper, a technique for image preprocessing and its robot action for line-following robots were suggested. To find path lines and choose the appropriate robot action, picture preprocessing techniques like dilation, erosion, Gaussian filtering, contour search, and centerline definition are used. Webots' simulator is used to practice the robot's implementation. Designing robot motions and line detection systems uses Python and OpenCV. The simulation results demonstrate that the technique has been properly applied, and the robot is capable of following various path types, including zigzag, dotted, and curved lines. The key factor in identifying path lines is the sharpness of the cropped-image frame.[23]

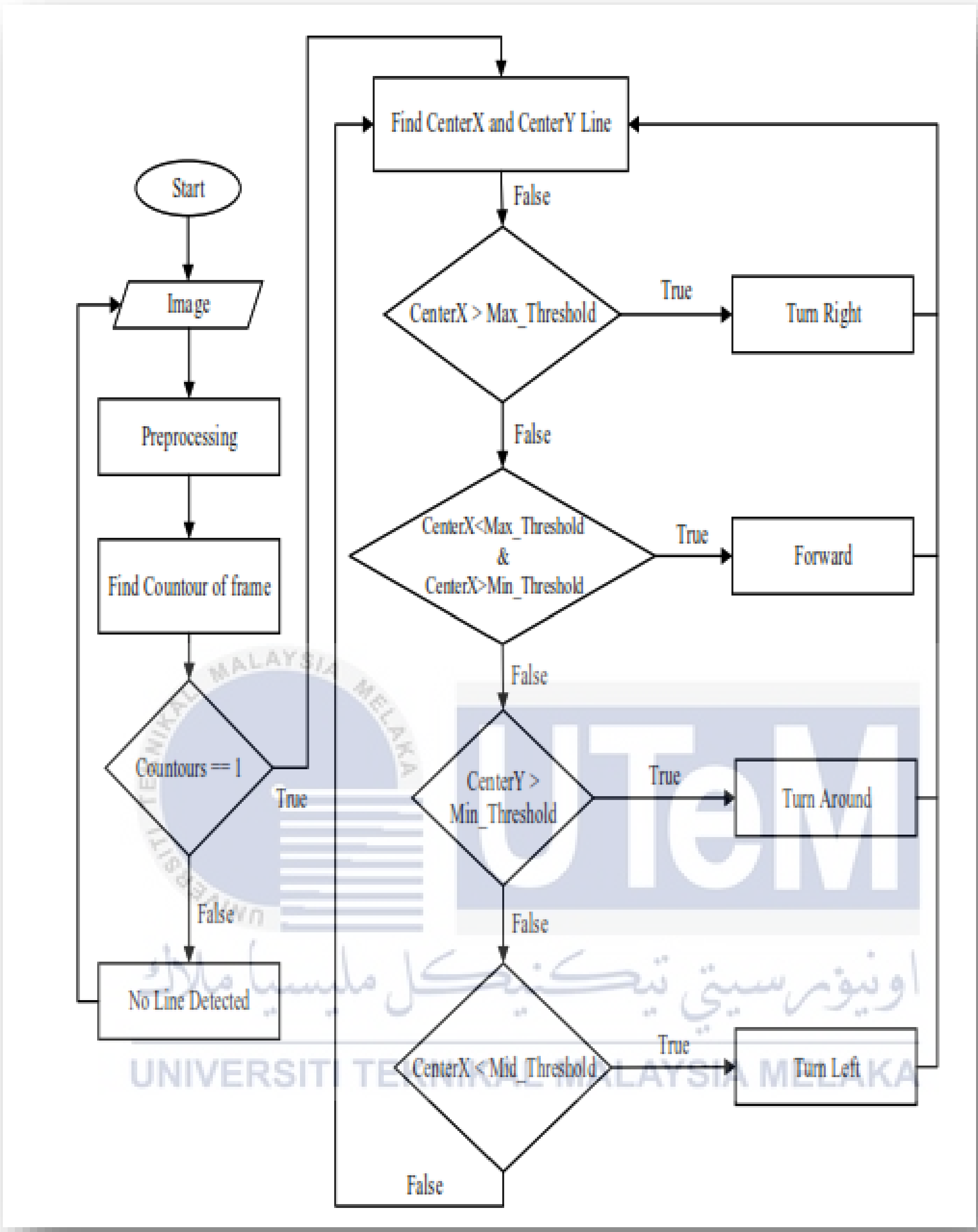


Figure 48 Line following detection system diagram

2.4 Blynk Application system

Wan Norsyafizan W.Muhamad et al 2020 proposed mart Bike Monitoring System for Cyclist via Internet of Things (IoT). This project showcases a smart bike with an internet of things monitoring device for cyclists. (IoT). The system is created for the purpose of tracking a cyclist's performance and overall wellbeing in real time. The entire system is made up of a number of instruments, including a GPS module, magnet reed sensor, pulse oximetry sensor, and heart rate sensor. The sensors are linked to the Wi-Fi module and microcontroller, which are accessible through the blynk program, an IoT platform. Both the cyclist and their coach can keep an eye on the statistics related to the cyclist's health and performance. First, a heart rate sensor was used to track the cyclist's heart rate over time, and a pulse oximetry sensor was used to gauge the cyclist's body's oxygen concentration. In the meantime, a magnet reed sensor was attached to the bike's frame to track the cyclist's speed and distance. Additionally, a GPS module was used to monitor the cyclist's location. Through the blynk application, the transmitted data can be synchronized with or viewed on a smartphone. So that the cyclist and coach could keep an eye on the cyclist's health while they were practicing or competing, the GUI interface of the Blynk application showed parameters like heart rate, pulse oximetry, speed, distance traveled, and position.[24]

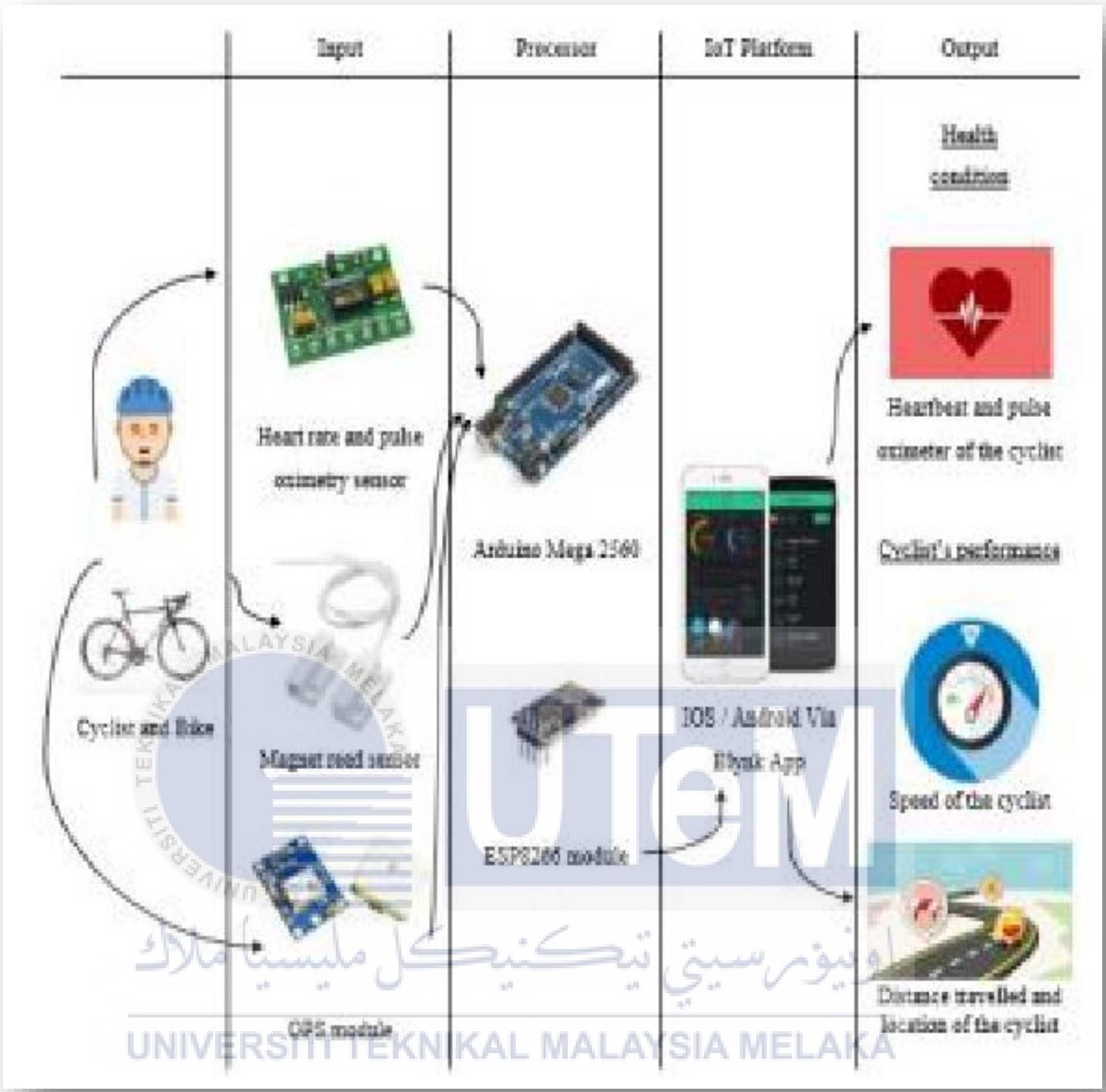


Figure 49 Block diagram of operation in Smart Bike Monitoring System for Cyclist via Internet of Things (IoT)

Prof. Mitul Sheth et al 2019 proposed a Smart Gardening Automation using IoT with BLYNK App which develops a smart automated system for providing water to fields, farms, and other places. We use soil moisture detectors, temperature detectors, and humidity detectors in this device, which are mounted at the plant roots. The system transmits to the base station the numbers it has identified. The goal is to use WiFi to retrieve info and sync those values with the internet. As soon as the water level drops below the set limit, it alerts the user. This article demonstrates how wireless circuit diagram observation using NodeMCU is possible, and it displays the results using the Blynk App.[25]

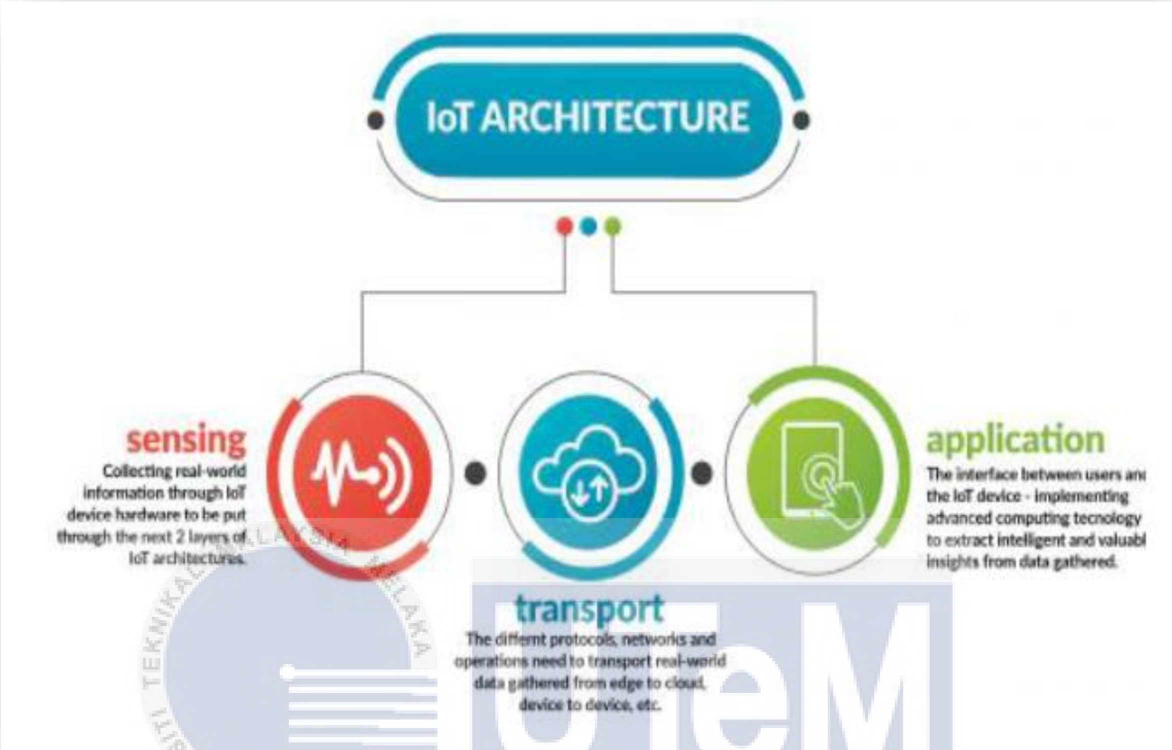


Figure 50 IoT Architecture

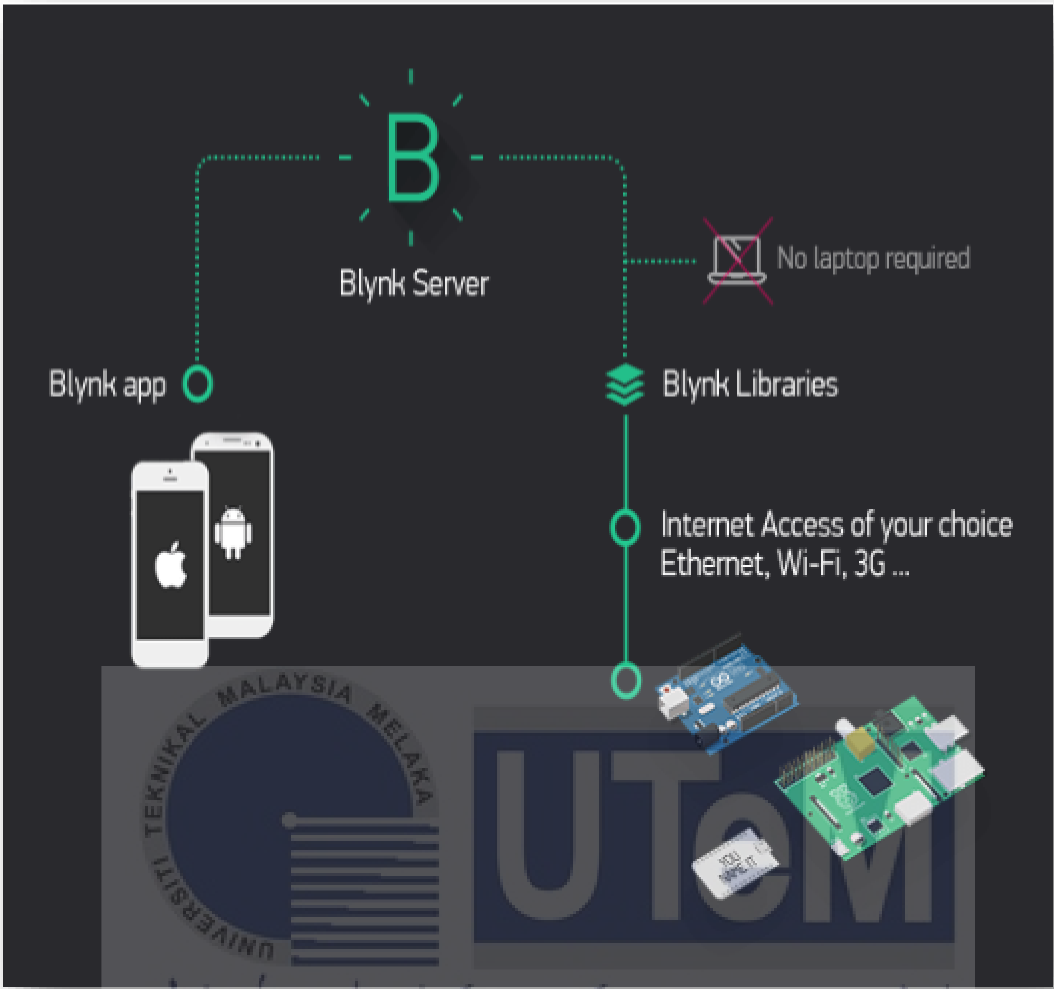


Figure 51 Blynk App working diagram.

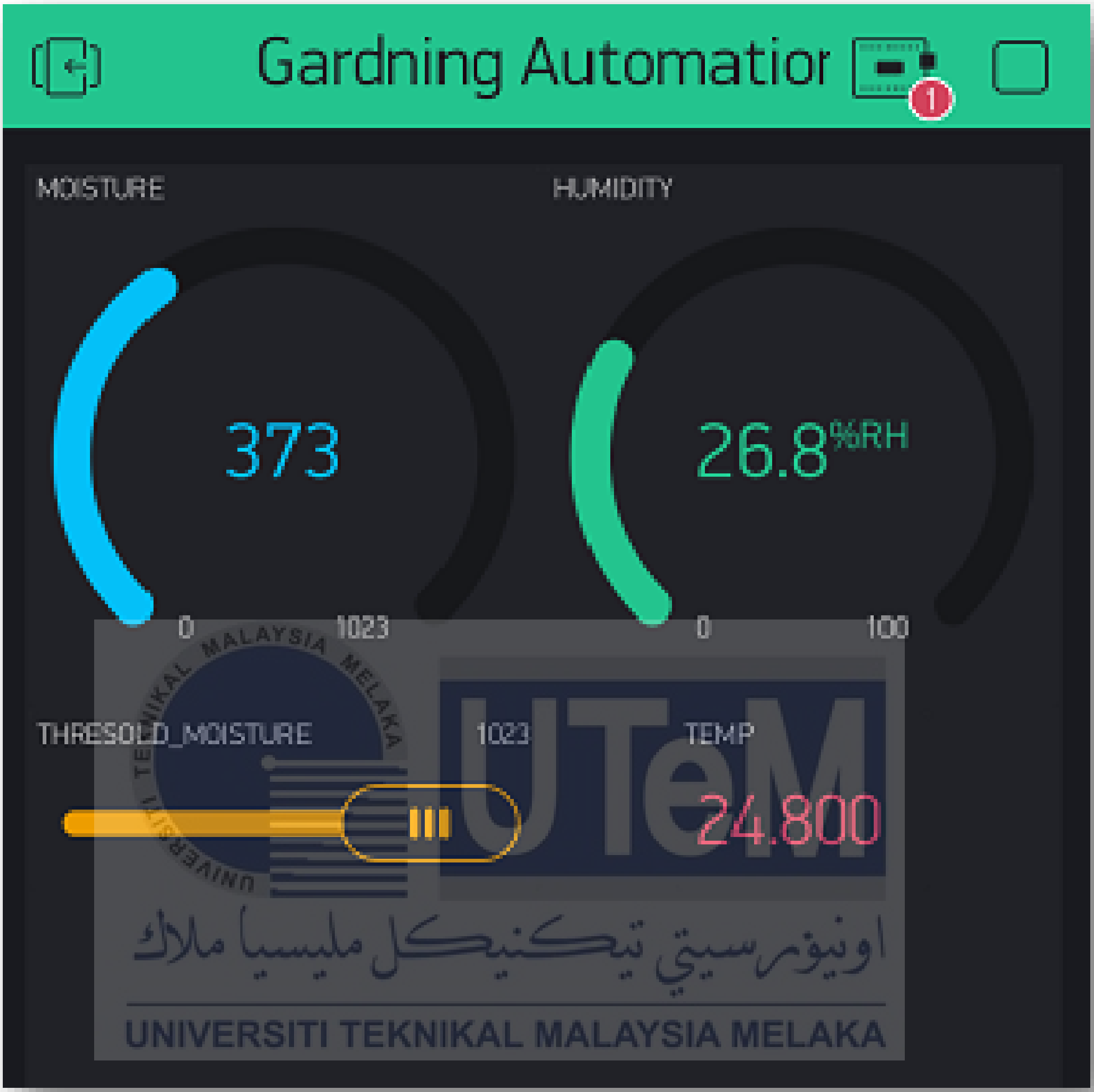


Figure 52 Blynk App live Project Status

Sreenivas Eeshwaroju et al 2021 proposed An IoT based Three-Dimensional Dynamic Drone Delivery (3D4). System Cities will have more high-rise structures due to the population growth. Additionally, the quick development of information and communication technology will call for clever solutions to address people's delivery needs. In order to facilitate vertical deliveries, this study suggests a "Three-Dimensional Dynamic Drone Delivery (3D)" system that expands on the traditional two-dimensional delivery methods with a third dimension (Z-axis). The suggested method gives the user the ability to accept shipments wherever they are needed. Additionally, this system improves customer satisfaction and addresses delivery issues like traditional fixed-point deliveries and package security.[14]

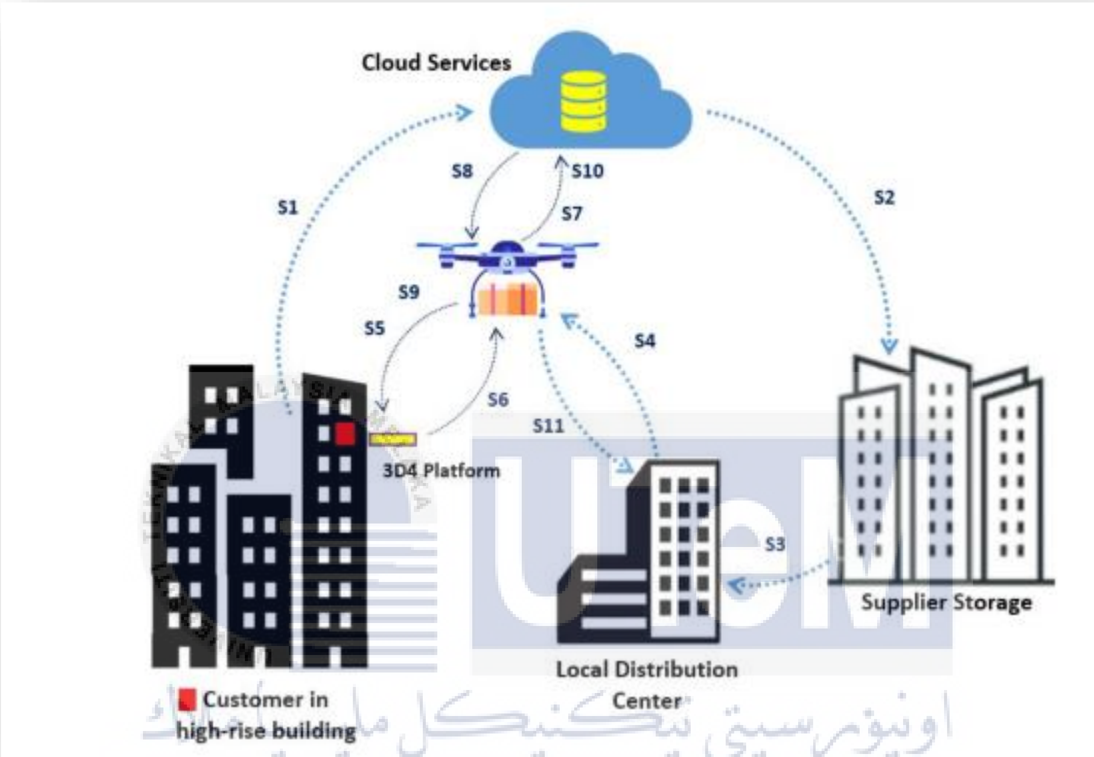


Figure 53 3D4 System Functional Overview Diagram

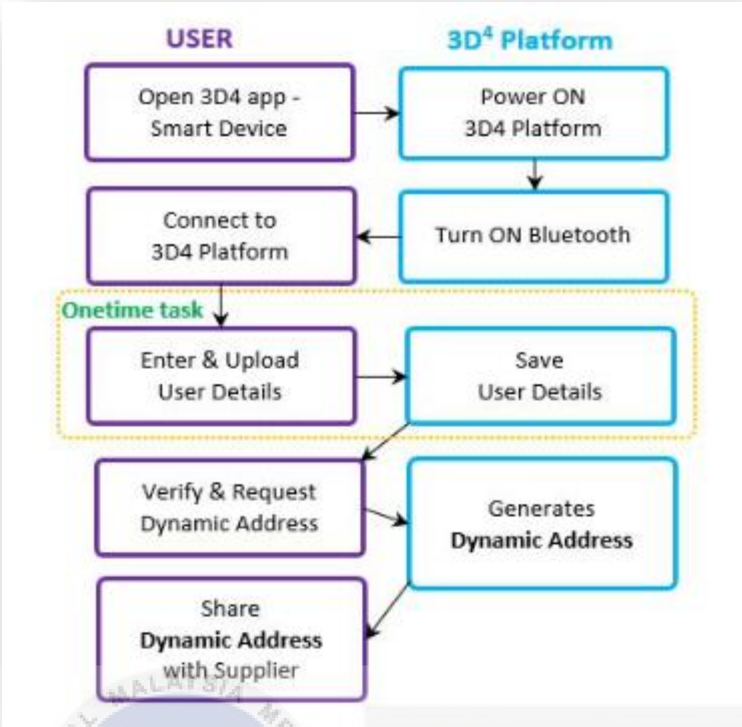


Figure 54 3D4 Platform Configuration Flow Diagram

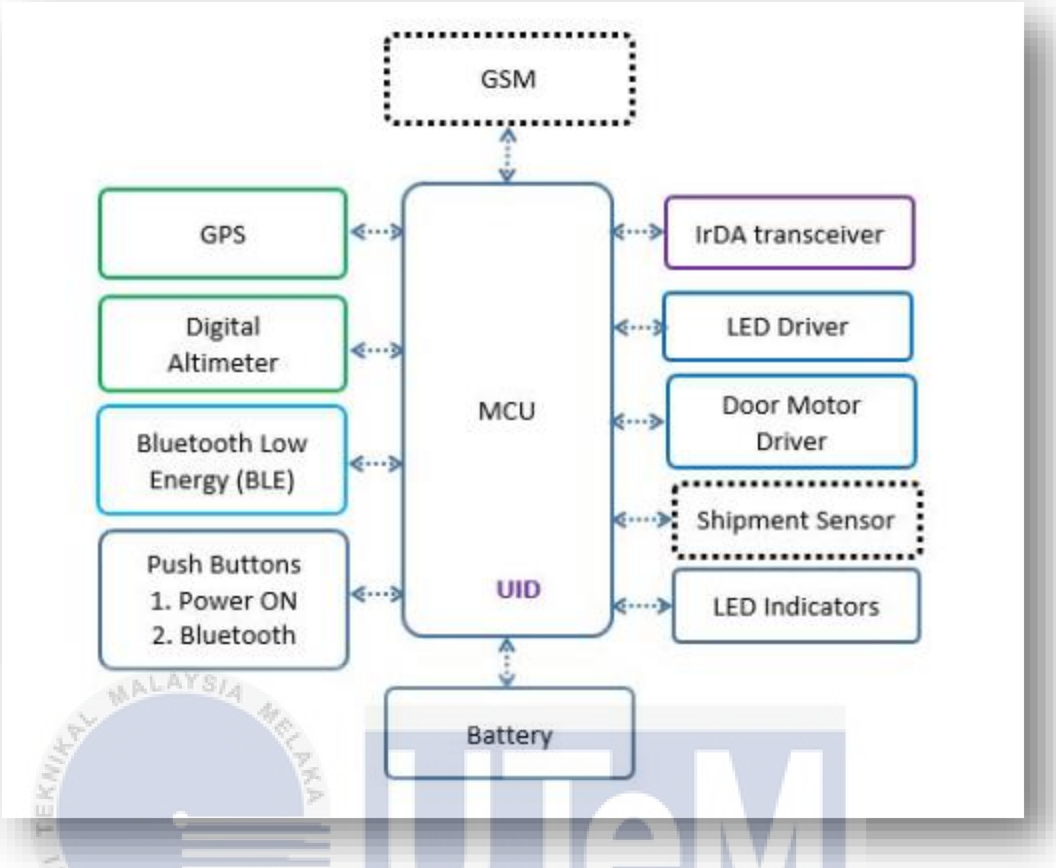


Figure 55 3D4 Platform Electrical Block Diagram

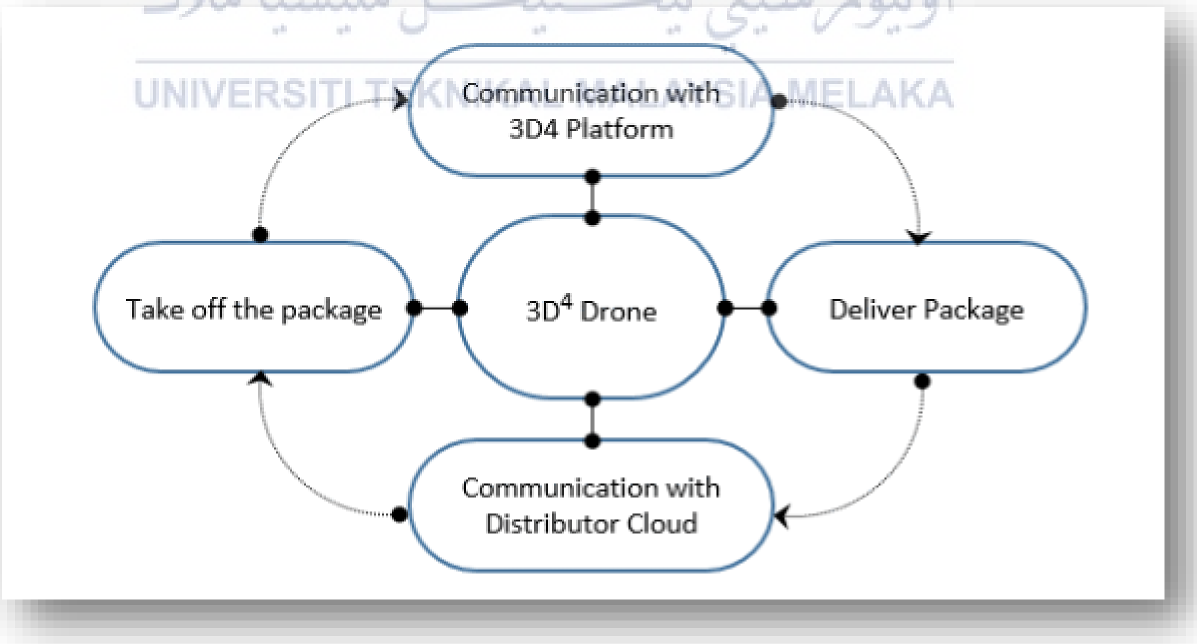


Figure 56 3D4 Drone Functional Block Diagram

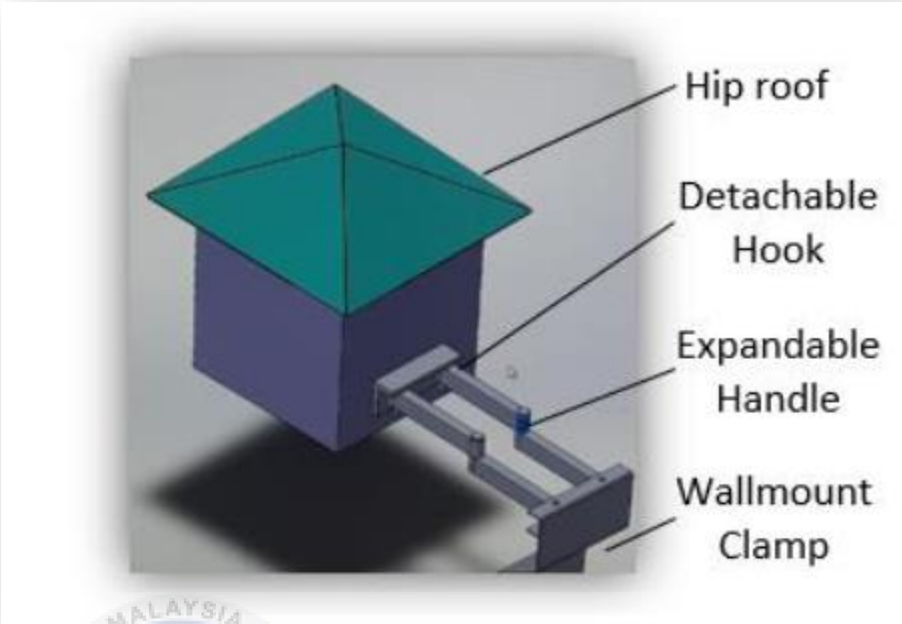


Figure 57 3D4 platform physical model

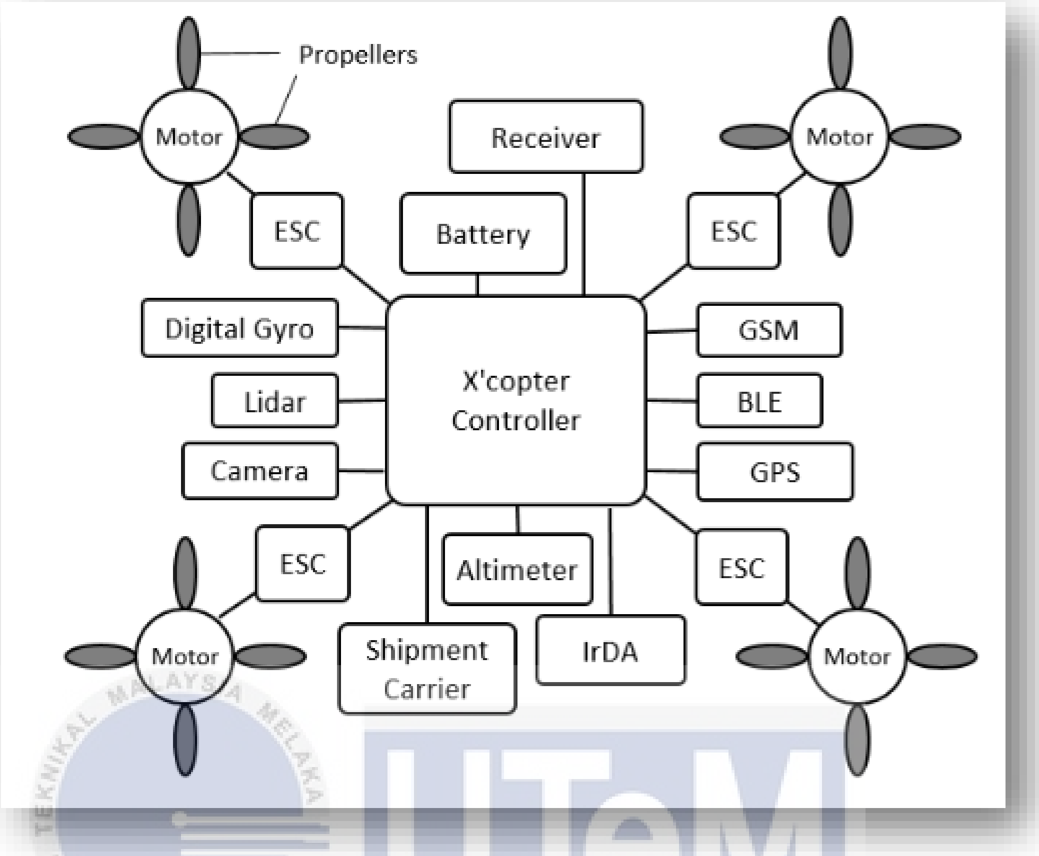


Figure 58 3D4 Drone Electrical Block Diagram

2.5 Analysis on Literature Review

Studies Presented	Method Used	Strength	Weaknesses	Findings
Smart Ai Based Delivery Robot	Artificial intelligence algorithms.	To increase delivery effectiveness and guarantee safe and secure package delivery, a self-driving robot prototype was developed. One of the navigational systems used is the Global System for Mobile Communications (GSM).	Self-driving delivery robots decreased travelling speed and limited walking range.	The right destination will be reached by the smart delivery robot as stated and trained prior to deployment, according to the overall research. The concept that is being presented can be used in a variety of settings, including hospitals, autonomous vehicles, food delivery, and others.

Smart Box Delivery System For Faster And Safer Delivery	GPS and GSM	Create a Smart Courier Box that can authenticate, receive, and deliver the ordered goods promptly while also valuing customers and online retailers. The Smart Courier Box was developed with the intention of decreasing the number of humans.	Enhancing the Security and Scalability of the Smart Courier Box. Very effective in terms of commodity protection as well as ensuring the safety and delivery of things to relevant firms.	When it comes to protecting commodities and assuring the delivery of goods to appropriate firms, this application has the potential to be highly effective. Technology such as GPS and GSM will be utilized to accurately track the vehicle.
Modelling a TurtleBot3 Based Delivery System for a Smart Hospital in Gazebo	Gazebo Simulation	Advancing the use of robots as helpers, working alongside humans and other robots in the same setting	When compared to drones, they often have a lesser working speed and range.	The idea of specifying potential robot actions as a finite set of states with certain transitions. In a Gazebo simulation, a hospital map was built to evaluate the technology.
IOT based Smart Delivery Box	IOT, MQTT Technology	This clever mechanism needs to be quick, easy, and secure enough to hold the delivered package.	MQTT is not developer friendly. The lack of security incorporated into MQTT in IoT. It does not arrive ready to use out of the box, thus it is up to the end user to handle. This necessitates the creation of a security layer on top of the MQTT.	A secured delivery Box which generates OTP for every active session and notifying the customer about the systematic process flow till the session is terminated.
Vision-Based Line Following Robot in Webots	Line mapping	Designing robot movements and line detection systems using Python and OpenCV.	Speed of the robot cannot be controlled well. The noise in the image will affect the visual	The simulation results demonstrate that the strategy has been effectively applied, and the robot is capable of following various path types, including zigzag, dotted, and curved lines. The key factor in identifying path lines is the resolution of the cropped-image frame.

Smart Gardening Automation using IoT With BLYNK App	Blynk Application	Using the slider, we can manually adjust Soil-Threshold value by analyzing the soil.	Poor Internet Connectivity in Farms, Disrupted Connectivity to the Cloud Take up a lot of space to grow the plant	Utilizing a NodeMCU ESP8266 microcontroller chip through WiFi, smart gardening automation is implemented. For the user interface, we are employing the iOS version of the BLYNK application, which allows users to effortlessly control and monitor farms or gardens from anywhere in the world.
Smart Bike Monitoring System for Cyclist via Internet of Things (IoT)	Blynk Application	The technology is made to track a cyclist's performance in addition to their general health in real time. The entire system is made up of a number of sensors, including a heart rate sensor, a pulse oximetry sensor, a magnet reed sensor, and a GPS module.	The system also will be more reliable if Raspberry Pi is used as a microprocessor due to advance technology.	This system helped in connecting the data between people and sensor to obtain specific data such as health condition and cyclist's performance.

CHAPTER 3

METHODOLOGY

3.1 Introduction

Every task that needs to be carried out in order to achieve the aim of this study case will be covered in this chapter. This study case's methodology includes project planning, system design, component specification, software setup and simulations, and analysis. According to the three project objectives that have been implemented, Chapter 3 is separated into three types of methods, as indicated below:

Table 1 Objectives and Classification

Objectives	Classification
To design Line Following Robot and (IoT) Passcode Verification using Node MCU.	Software Programming and Flowchart development
To develop a Smart Package Delivery System using Line Following Robot with Passcode Verification.	Component and Hardware
To create the line following system for Smart Package Delivery System and improve it using PID (Proportional-Integrative-Derivative) Algortihm into the line following system in order to maximize its performance.	Comparing line following system that implement PID algorithm and without implementing PID algorithm

3.2 Consideration for Social Sustainability

Social impact will be taken into account during the project's design decision-making process. Building a low-cost project that is still marketable and effective, for instance, would allow people to continue using the system. The project will took into account the environment's safety by designing the line-following robot's circuit path in a way that avoids any obstacles.

3.3 Consideration for Societal and Global Issue

It is essential to safeguard user data and ensure secure transfer and storage of personal information because the system uses passcode verification (OTP). To protect sensitive data from unauthorised access, use strong encryption techniques, data anonymization, and rigorous access controls. Moreover, make sure that everyone, including those with impairments, can access and use the smart package delivery system. To accommodate varied users and encourage inclusivity, take into account elements like user interfaces, visual clues, and alternate communication techniques.

3.4 System Architecture

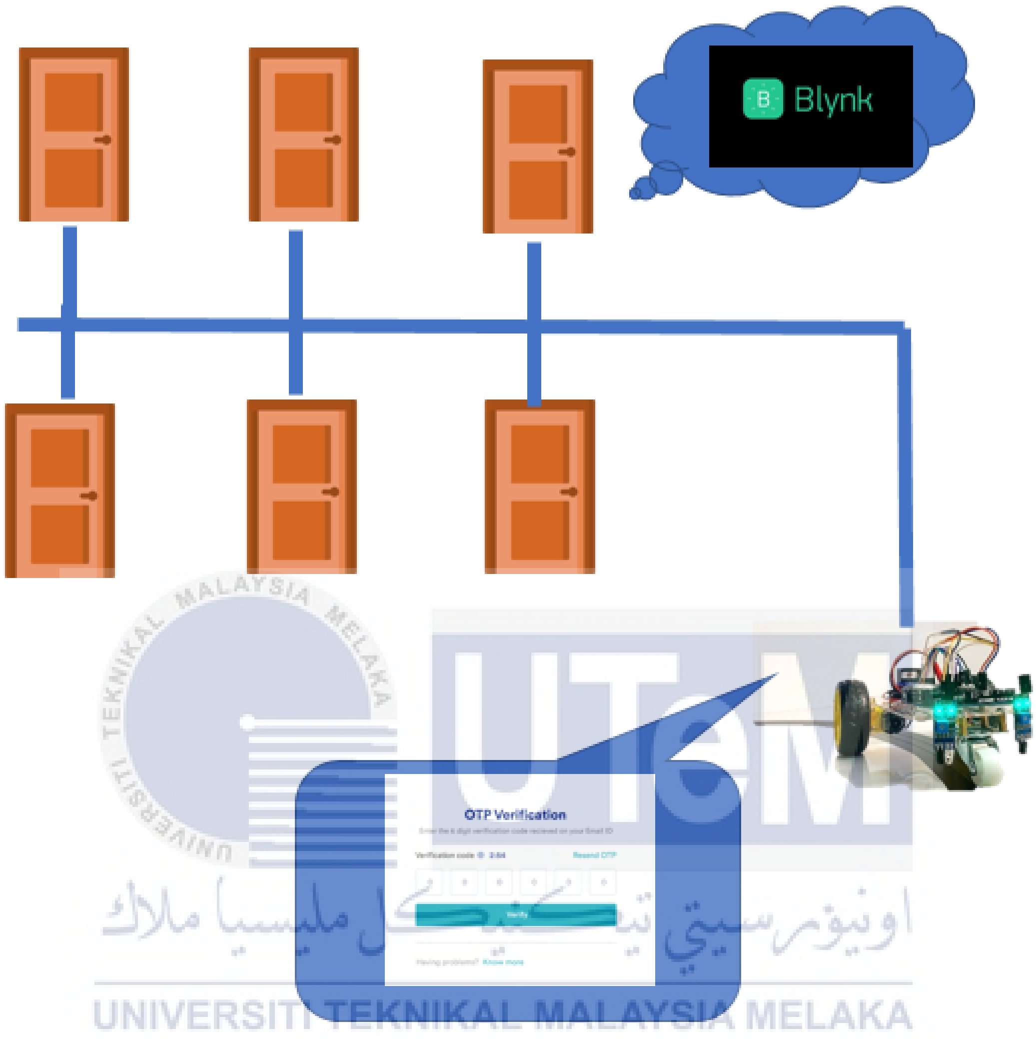


Figure 59 Develop a Smart Package Delivery System using Line Following Robot with Passcode Verification.

Figure abover shows the block diagram and the system architecture of this project. .

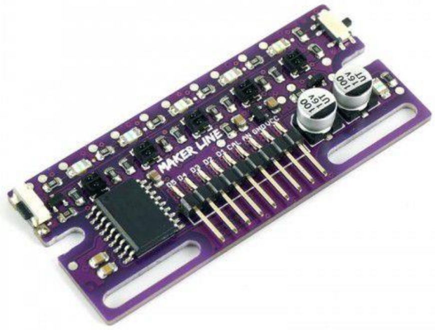
3.5 Hardware Requirements

Table 1 summarized the list of hardware with specification and quantity used to implement in the system. Selection of components is needed to ensure compatibility in the system for better performances

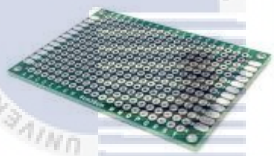
Table 2 Hardware Requirement

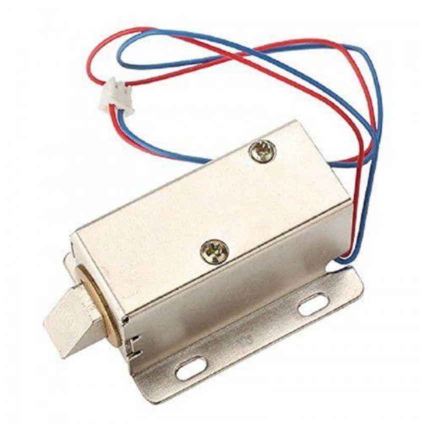
Components	Units	Specification
Arduino ESP 32 TTGO T-Call SIM800L 	1	• Chipset: ESPRESSIF-ESP32 240MHz Xtensa® single-/dual-core 32-bit LX6 microprocessor • FLASH Memory: QSPI flash 4MB / PSRAM 8MB • SRAM: 520 KB SRAM • Button: Reset • USB to TTL IC: CP2104 • Modular interface: UAR, SPI, SDIO, I2C, LED PWM, TV PWM, I2S, IRGPIO, capacitor touch sensor, ADC, DACLNA pre-amplifier • On-board clock: 40MHz crystal oscillator • Working voltage: 2.7V-3.6V • Working current: About 70mA • Sleep current: About 300uA • SIM card: Only supports Nano SIM card • Working temperature range: -40℃ ~ +85℃ • Size&Weight: 74.78mm*29.01mm*8.06mm(11.77g)
Cytron H-Bridge Motor Driver	1	• Driver: H-Bridge • + Driver power supply: +5V~+35V • + Driver peak current: 2A

		• + Logic power output Vss: +4.5~+5.5V (internal supply +5V) • + Logic current: 0~36mA • + Controlling level: Low -0.3V~1.5V, high: 2.3V~Vss • + Enable signal level: Low -0.3V~1.5V, high: 2.3V~Vss • + Max drive power: 20W (Temperature 75 ℃) • + Working temperature: -25℃~+130℃ • + Dimension: 60mm*55mm • + Driver weight
Arduino Mega 2560 (ATmega2560) 	1	• Operating Voltage -5V • Input Voltage(recommended) -7-12V • Input Voltage (limits) -6-20V • Digital I/O Pins -54 (of which 14 provide PWM output) • Analog Input Pins-16 • DC Current per I/O Pin-40 mA • DC Current for 3.3V Pin -50 mA • Flash Memory -256 KB of which 8 KB used by bootloader • SRAM -8 KB • EEPROM -4 KB • Clock Speed -16 MHz

Cytron Line Sensor 	1	• dimensions: 2.95" x 0.5" x 0.125" (without header pins installed) • Operating voltage: 3.3-5.0 V • Supply current: 100 mA • Output format: 8 digital I/O-compatible signals that can be read as a timed high pulse • Optimal sensing distance: 0.125" (3 mm) • Maximum recommended sensing distance: 0.375" (9.5 mm) • Weight without header pins: 0.11 oz (3.09 g)
Motor Gear With Tyre 	4	• Voltage: 3-6VDC • Current: 80-150mA • No Load Speed: 3V-125 rev/min5V-200 rev/min6V-230 rev/min • Load Speed: 3V-95 rev/min5V-160 rev/min6V-175 rev/min • Output Torque: 3V-0.8kg.cm5V-1.0kg.cm6V-1.1kg.cm • Wheel Diameter: 65mm including tyre • Wheel Width: 25mm • Gearbox/Motor Dimensions: 20mm x 22mm x 65mm • Weight: 50grams • Kit include:

Battery Lipo 11.1v 12v battery lithium ion  	1	• Capacity: 9800mah • Dimension: ~13x7x2.5cm • Weight: ~0.35kg • Battery type: Rechargeable Li-on battery • Input voltage: 12.6V • Output voltage: 10.8~12.6 DC • Output current: 1-3A(max) • DC Connector: 5.5 x 2.1mm • Max Support Power: 50W
Casing Box & Hinge Playwood 1FT X 2FT 	1	• 1 FT X 2 FT

		
Caster Wheel 	1	
Doublesided PCB Board 	1	• 70 mm x 90 mm
Female and male pin 	1	• Female & male
Solenoid Lock 12V	1	• Voltage: 12VDC • Current: 0.6A • Size: 53 x 26 x 23mm • Weight: 142g • Latch telescopic length: 10mm

<table>
<tr>
<td></td>
<td></td>
<td>

Energized forms: intermittent;
Unlocking time: 1S
Continuously energized <10S;
Temperature: -40 ~ +50
Applications: Cabinet lock,locker locks,file cabinet locks ,luggage locks,electric locks,door locks,solenoid locks,drawer,newspaper boxes lock ,sauna lock,locker electromagnetic locks,electric locks,newspaper boxes,sauna electronics lock.

</td>
</tr>
<tr>
<td>Keypad

</td>
<td></td>
<td>

SIze:6.5 x 6.4 CM
Weight:13.3g

</td>
</tr>
</table>

3.5.1 Microcontroller

- ESP32 (TTGO-TCALL SIM800L

The ESP32 is a low-cost, low-power microcontroller that includes Bluetooth and Wi-Fi. It is the replacement for the ESP8266, a low-cost Wi-Fi microprocessor with extremely restricted features. It includes a power amplifier, low-noise amplifiers, filters, and a power management module in addition to an integrated antenna and RF balun. The solution uses the least amount of space on the printed circuit board as a whole. This board uses TSMC 40nm low power 2.4 GHz dual-mode Wi-Fi and Bluetooth chips, which have the best power and RF attributes and are secure, dependable, and expandable to a range of applications.

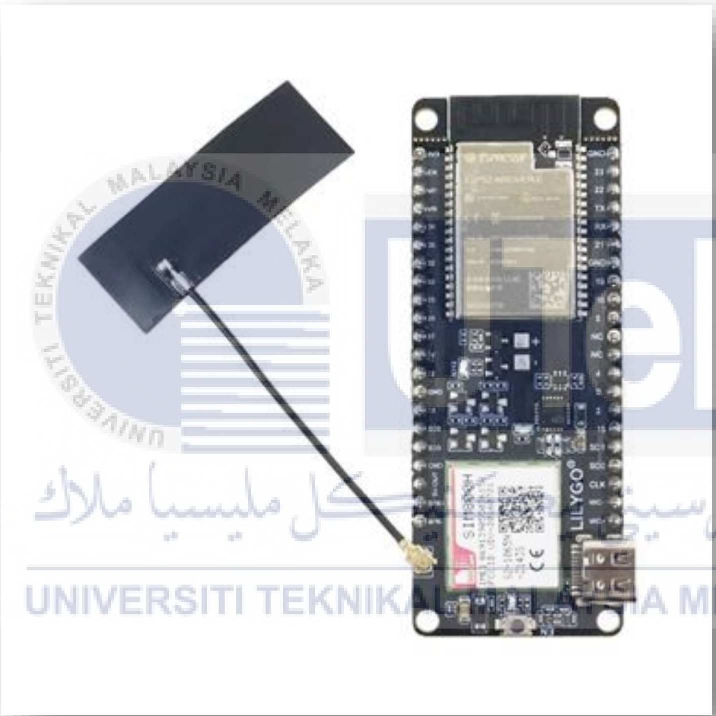


Figure 60 ESP32 (TTGO T-CALL SIM800L)

The control board for the Line Follower Robot IOT is an ESP32 board. All the parts needed to operate an onboard ESP32 microprocessor are present in the ESP32. The 3.3V output of the DC-DC converter is where is linked and operates at. The pins for the motor and sensors are attached to digital GPIOs and PWM-capable GPIOs, respectively.

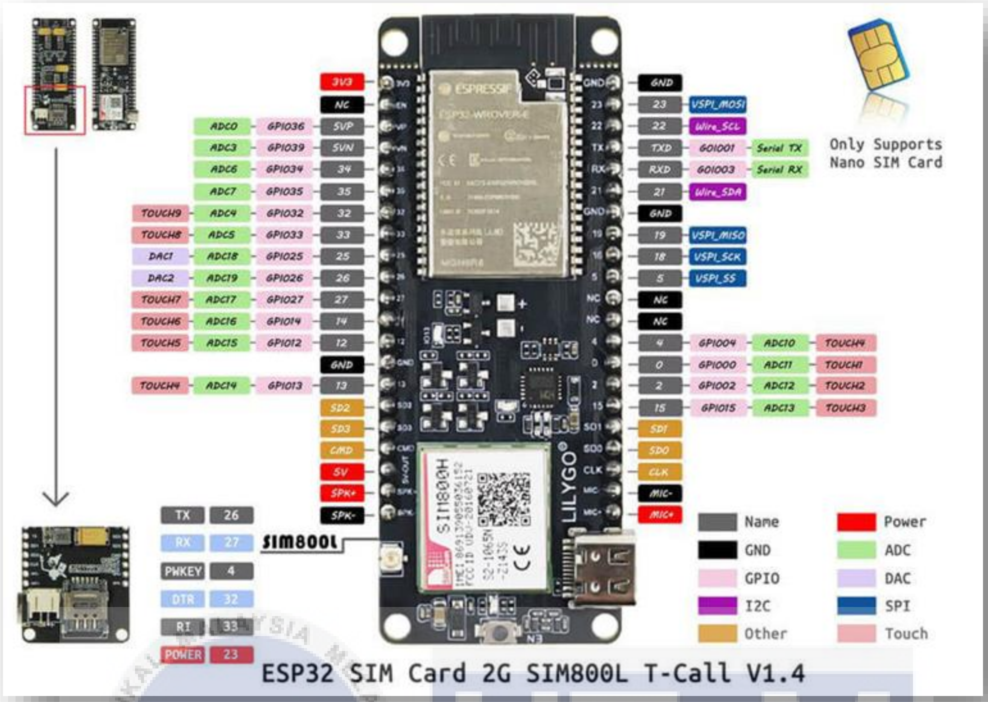


Figure 61 ESP32 SIM800L Pin Out

With each GPIO pin, the ESP32 offers three different I/O modes: internal sensors, analog, and digital. ESP32 was selected because it have many input/output pins compare to Nodemcu. Secondly, it cost about the same as ESP8266 and relatively cheap for its price. It is more stable compared to ESP8266 or any other WiFi module.In this project, ESP32 is used for IoT of the project, it is used mainly to connect to Blynk application and to send user the TAC code when the robot has arrived at the room. The ESP32 will always be connected to the Arduino Mega through the communication pins that is the RX, TX of the ESP32 for any necessary data communication.

- Arduino Mega 2560

A microcontroller board based on the ATmega2560 is called the Arduino Mega 2560. It contains 16 analog inputs, 4 hardware serial ports (UARTs), a 16 MHz crystal oscillator, 54 digital input/output pins (of which 15 can be used as PWM outputs), a USB connector, a power jack, an ICSP header, and a reset button.

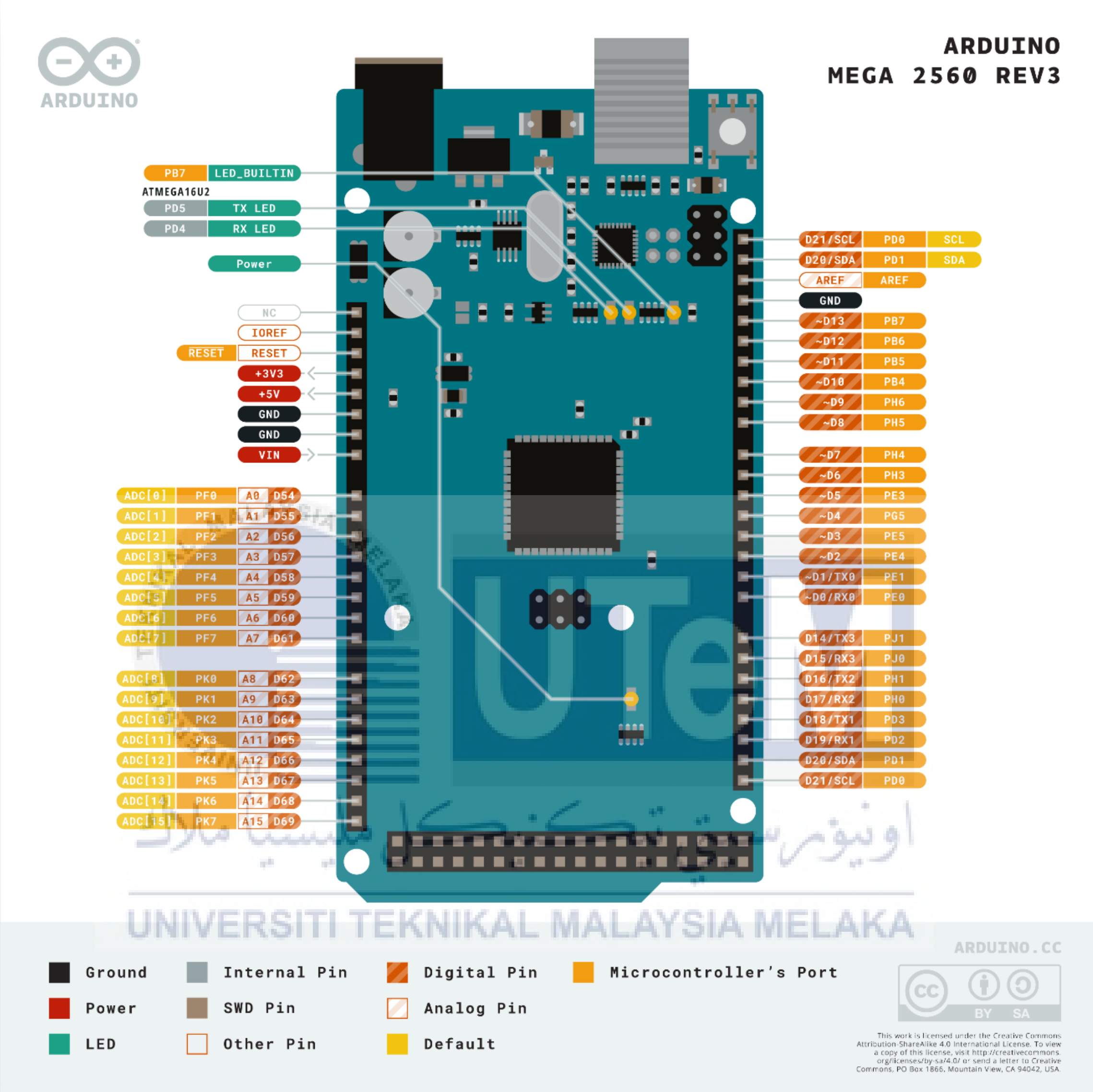


Figure 62 Arduino Mega 2560 Pin Out

Arduino Mega 2560 was selected because it has many input and output to work on, Secondly, it have 3 communications pins for sending data to ESP32. Lastly it is durable and reliable. In this project the Arduino Mega is used mainly for processing the input and output for the line following robot. The Arduino Mega will also handle the input of the keypad and the package lock system using the servo motor.

3.5.2 Cytron Line Sensor

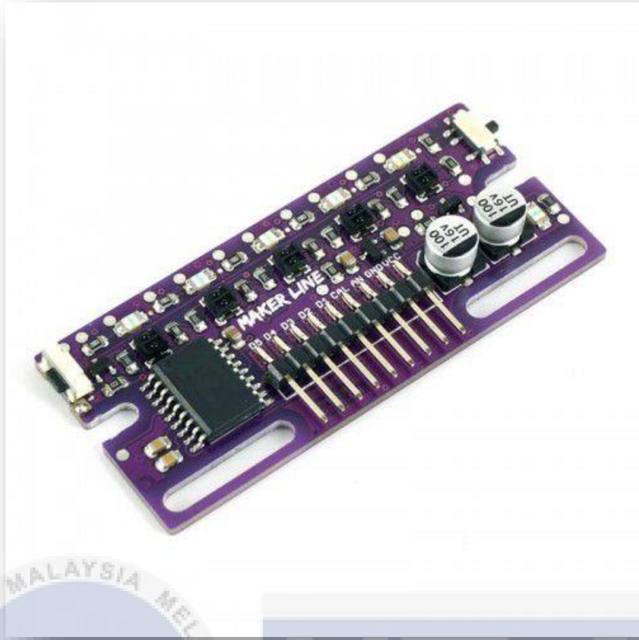


Figure 63 Cytron Line Sensor

A reflectance sensor array called the Cytron Line Sensor was created particularly for use as a line sensor by robots that follow lines. Five IR emitter and receiver (phototransistor) pairs are uniformly distributed throughout the Cytron Line Sensor at intervals of 0.375″ (9.525 mm). If necessary, the sensor can be divided into two pieces. The sensor, which includes five IR emitters and receivers, can distinguish between a light surface with high IR reflectivity and a dark surface with low IR reflectivity.

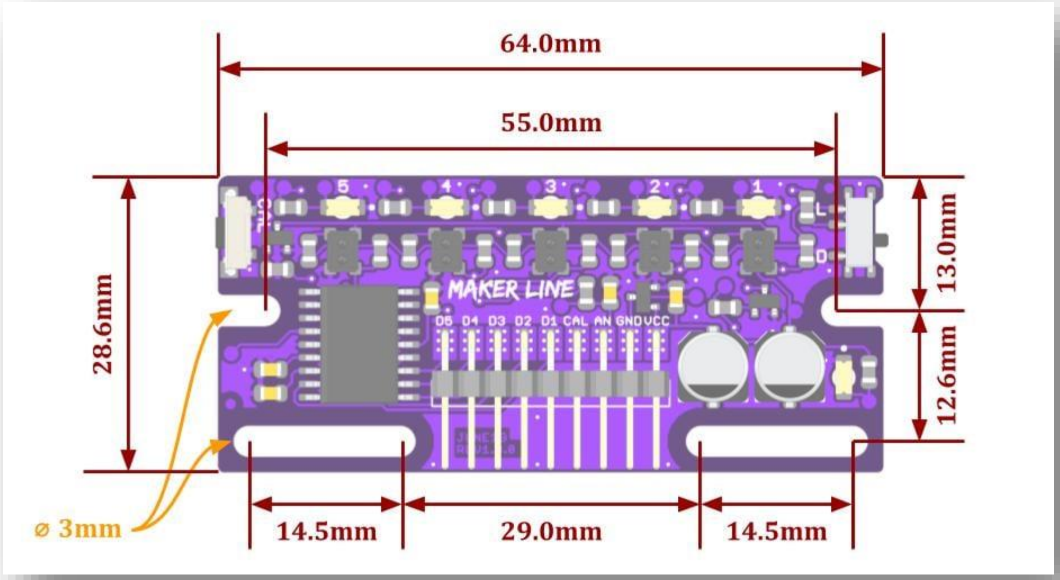


Figure 64 Cytron Line Sensor Dimension

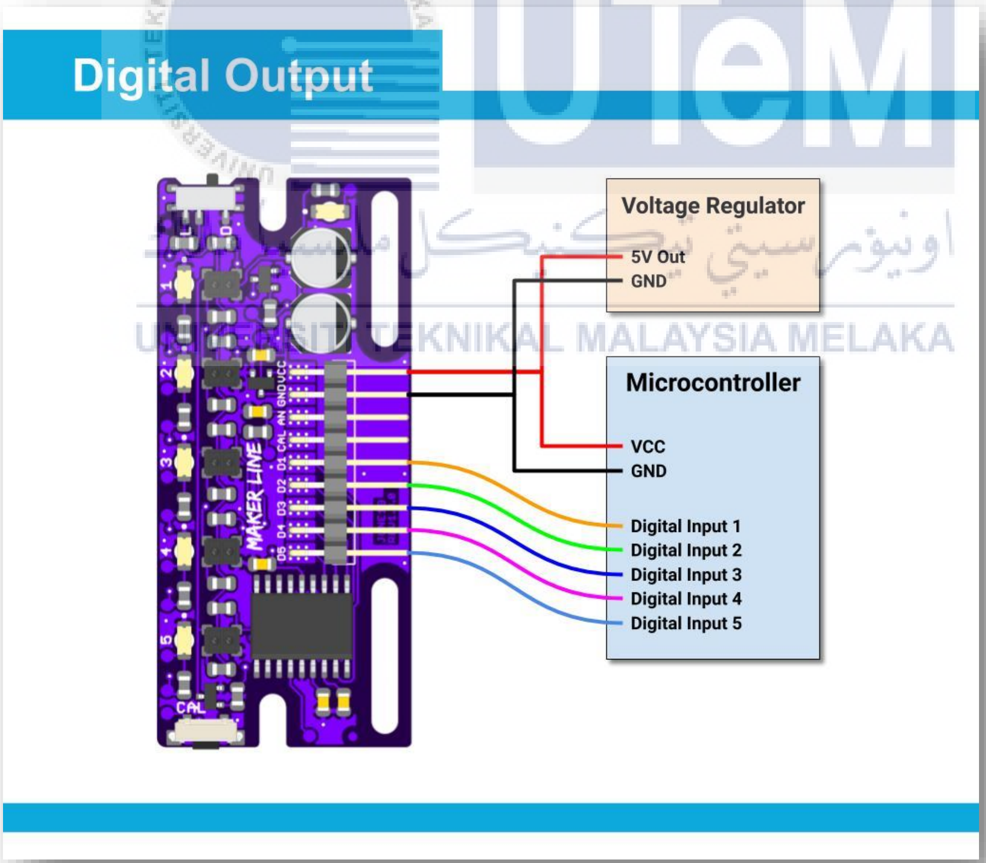


Figure 65 Cytron Line Sensor Pin Out

A QRE1113GR infrared (IR) reflecting sensor is used by the Line Sensor to ascertain the reflectivity of the surface beneath it. The reflectance is extremely low when the sensor is placed over a black surface and extremely high when the Line Sensor is placed over a white surface, resulting in a difference in the sensor's reading.

Line Sensor was selected it has many sensor and the detection is more accurate. The line sensing height is adjustable and suitable for PID system design for line following robot. In this project the line sensor will be place in front of the robot so that the sensing system will work as intended. The 5 IR sensors will make it easier for the processor to sense the black line.

3.5.3 Cytron H-Bridge Motor Driver

The DC Motor Driver IC TB6612FNG has a constant current capability of 1.2A (3.2A peak) and can drive up to two DC motors. There are four possible modes for the two input signals IN1 and IN2, including CW, CCW, short brake, and stop mode.

A PWM input signal with a frequency of up to 100kHz controls the speed of each motor, and the two motor outputs (A and B) can be independently regulated. To exit standby mode and activate the motor, the STBY pin must be lifted high.

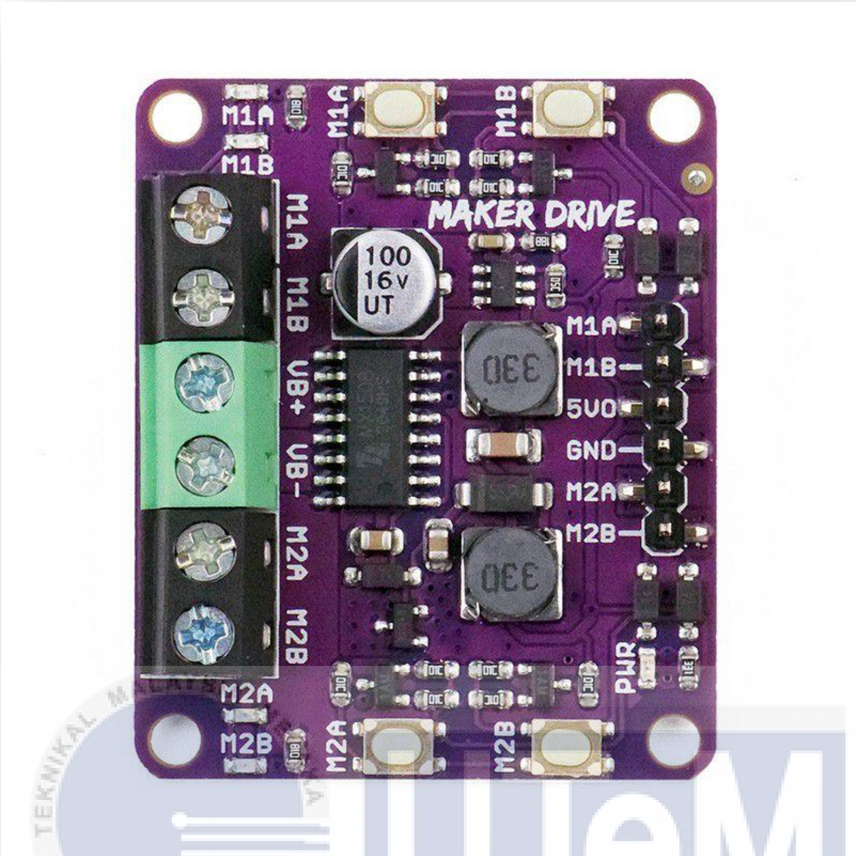


Figure 66 Cytron H-Bridge Motor Driver details

Table 3 Cytron H-Bridge Motor Driver

• Input voltage: 5-35V
• Output current: 2A per channel
• Control logic: standard 5V TTL
• Power consumption: 36ma for logic
• Size: 55mm * 60mm * 30mm
• Weight: 33g

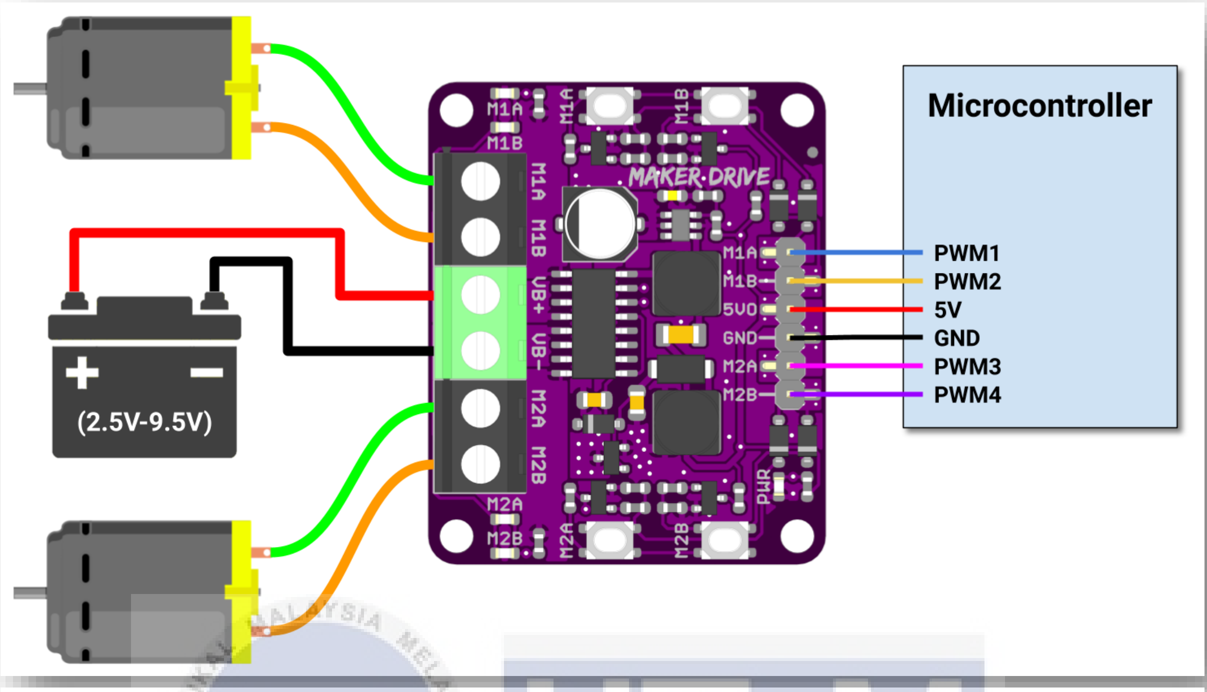


Figure 67 PINOUT of Cytron H-Bridge Motor Driver IC

H-bridge motor driver was selected because of it is low cost and cheap to be use. Next, H-bridge able to use higher voltage and can reach until 35 voltage. It is one of the very stable & reliable of motor driver. Last but not least, the motor driver's size makes it manageable. In this project the H-bridge motor driver will be used for controlling the motor speed and the motor direction so that the robot will follow the line easily.

3.5.4 12v Battery Lithium Ion

Figure 68 12V Battery Lithium Ion

Table 4 12v Battery Lithium Ion Specification

Capacity: 9800mah	Capacity: 9800mah
Dimension: ~13x7x2.5cm	Dimension: ~13x7x2.5cm
Weight: ~0.35kg	Weight: ~0.35kg
Battery type: Rechargeable Li-on battery	Battery type: Rechargeable Li-on battery
Input voltage: 12.6V	Input voltage: 12.6V
Output voltage: 10.8~12.6 DC	Output voltage: 10.8~12.6 DC

Lithium Ion 12v batteries were chosen since they are simple to charge and have a longer usable life. It also shows up in tiny sizes. In this project, the battery will supply the output power for the line following robot.

3.5.5 Solenoid Lock 12V

As is well known, solenoids are essentially large coils of copper wire with an armature centered within that function as electromagnets. The slug is drawn into the coil's center when it becomes electrified. The solenoid can now pull from one end as a result.

Because it has a robust mounting bracket and a slug with a slanted cut, this solenoid is nice and powerful. It is essentially an electronic lock. Normally, the solenoid slug gets in the way and prevents you from opening the door since the lock is engaged. At this point, it consumes no electricity. Nevertheless, the door can be opened when 9–12VDC is applied since the slug pushes in.

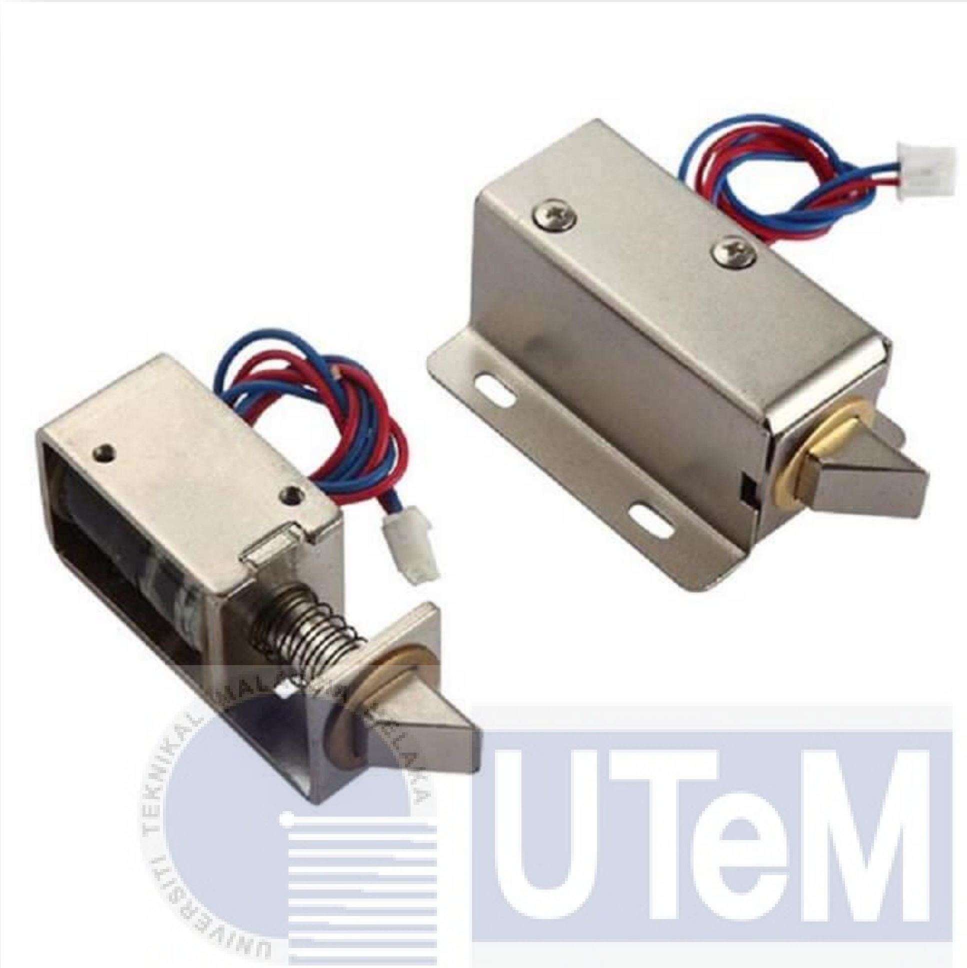


Figure 69 Solenoid 12V Lock

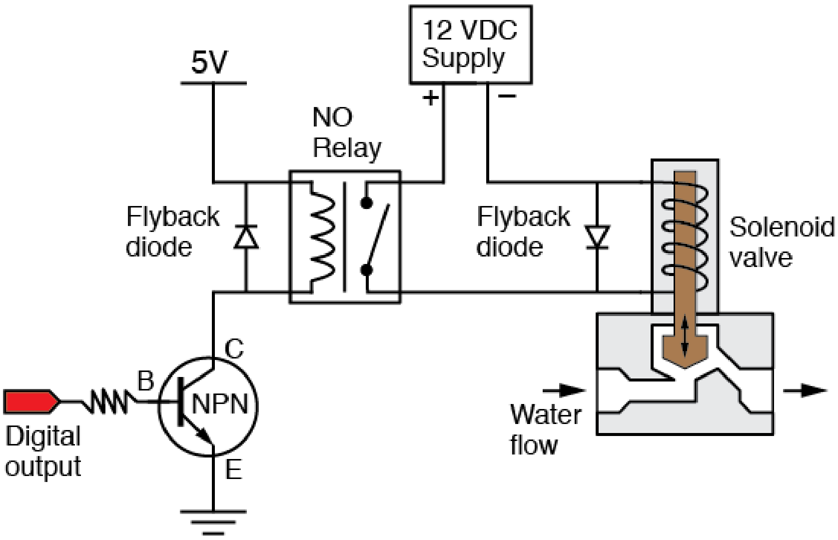


Figure 70 Solenoid 12V Lock Pinout

3.5.6 Motor Gear with Tyre

A 1:48 gear ratio and two 200mm wires with breadboard-friendly 2.54mm male connections are features of this TT DC Gearbox Motor. Perfect for connecting to a terminal block or breadboard.

Figure 71 Motor Gear With Tyre

Table 5 Motor Parameter

Motor Parameter:
• Rated Voltage: 3~6V
• Continuous No-Load Current: 150mA +/- 10%
• Min. Operating Speed (3V): 90+/- 10% RPM
• Min. Operating Speed (6V): 200+/- 10% RPM
• Torque: 0.15Nm ~0.60Nm

• Stall Torque (6V): 0.8kg.cm
• Gear Ratio: 1:48
• Material: Plastic
• Body Dimensions: 70 x 22 x 18mm
• Product Weight: 29g / 1.02oz
Wheel Parameter:
• Center hole: 5.3MM x 3.66MM
• Wheel diameter: 65mm
• Wheel thickness: 28mm

Motor Gear with Tyre was selected because it is easy to use and uses low voltage and current also it save battery. motor gear also low cost and cheap.In the project, the Motor Gear with Tyre will be used for the robot to move following the line.

3.5.7 Casing Box & Hinge Plywood 1FT X 2FT

Figure 72 Casing Box & Hinge Plywood 1FT X 2FT

Casing Box & Hinge Plywood 1FT X 2FT was selected because of the LOW COST to build the case. It also can cut easily to any shape & the wood very strong. In this project the Plywood will act as the box for the package lock system for the user to store the package in.

3.5.8 COM-08653 Matrix Keyboard Keypad Module

This is 4×4 Matrix 16 Keyboard Keypad Telephone Style Black. It is 16 Button Keypad switch, ideal for code or data entry. Each key is rated for up to 1,000,000 life-stroke and also features a durable high-quality material contact surface for best environmental resistances.

-This 4×4 Keypad Matrix lets you quickly add controls to your electronics projects. It offers 0-9 numerals, A-D letters, and standard star(*) and hash(#) symbols. It benefits is durable and high quality

In this project the keypad will be used to input the user telephone number and the room for the robot to send the package to.

Figure 73 COM-08653 Matrix Keyboard Kepad Module

3.6 Software Requirements

This section will include and complete the project's second objective, which is to program and create a Smart Package Delivery System using Line Following Robot with Passcode Verification.

3.6.1 Arduino Integrated Development Environment (Arduino IDE)

The Arduino Software (IDE), also referred to as the Arduino Integrated Development Environment (IDE), is another option that comes with a text editor for writing code, a message box, a text console, a toolbar with buttons for frequently used actions, and a number of menus. It connects to the Arduino device in order to code and upload the programming.

The Arduino Software (IDE) is used to build computer programs known as sketches. These illustrations are produced in a text editor, then saved as files with the.ino extension. Text replacement and text searching options are available in the editor. The message area reveals issues and offers feedback when saving and exporting. The terminal displays text produced by the Arduino Software (IDE), together with additional data including complete error messages. The assembled board

The bottom right corner of the pane shows the and serial port. Using the toolbar buttons, you may create, open, and save sketches, validate and upload programs, open the serial monitor, and more. The software used by Arduino to develop and execute code is depicted in the figure below.

Figure 74 Arduino IDE Software

This project uses the Arduino IDE to develop the coding and programming for each component used. The Arduino IDE executes all the component code. The component will then receive the uploaded coding. The element reads the information and produces results. The ESP32 , BLYNK examples of components that require code to operate.

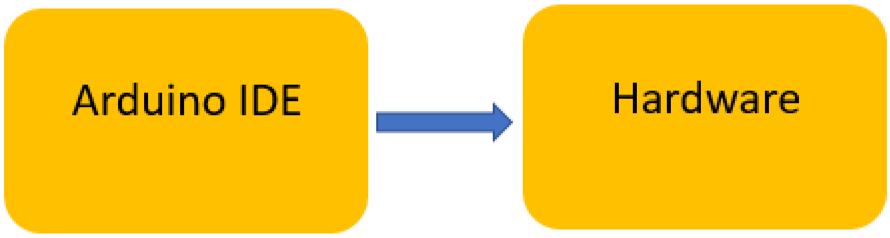


Figure 75 Block Diagram of software system

3.7 Proteus Software

An exclusive tool set for automating electronic design is called the Proteus Design Suite. Electronic design experts and technicians mostly use the programmed to develop schematics and electronic prints for printed circuit board manufacture. There are two type of proteus software use in this project which is Proteus PCB Design, Proteus Circuit Simulation. Proteus PCB tools offers a robust, integrated, and user-friendly collection of tools for professional PCB Design by seamlessly combining schematic capture and PCB layout. Next, Proteus Circuit Simulation is before ordering a physical prototype, design, test, and debug full embedded systems inside schematic capture. With VSM, the embedded process may now use AGILE development.

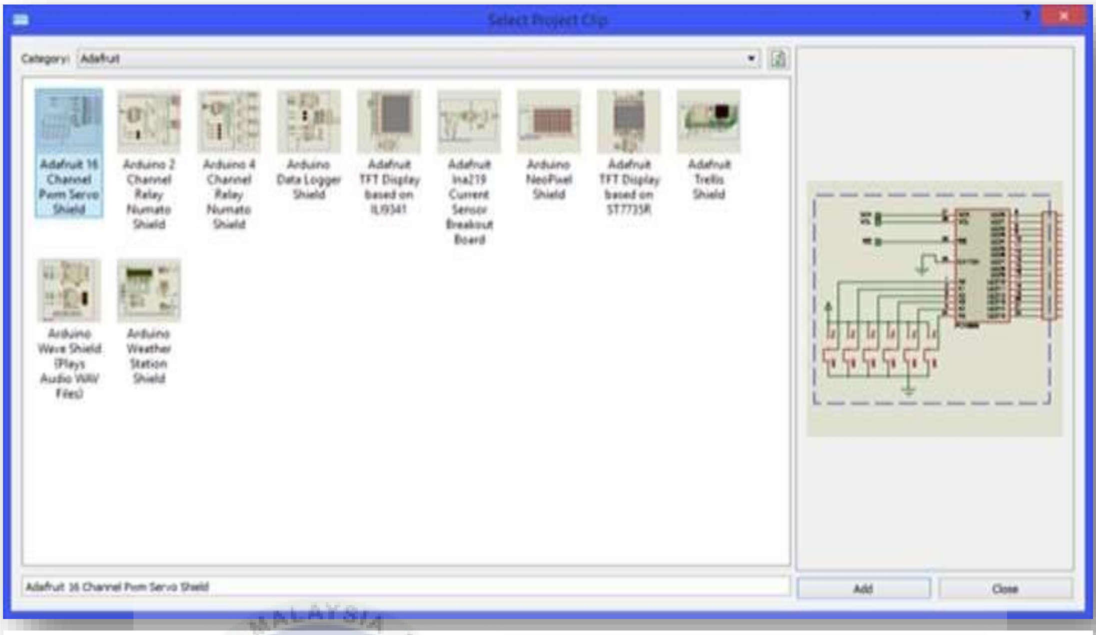
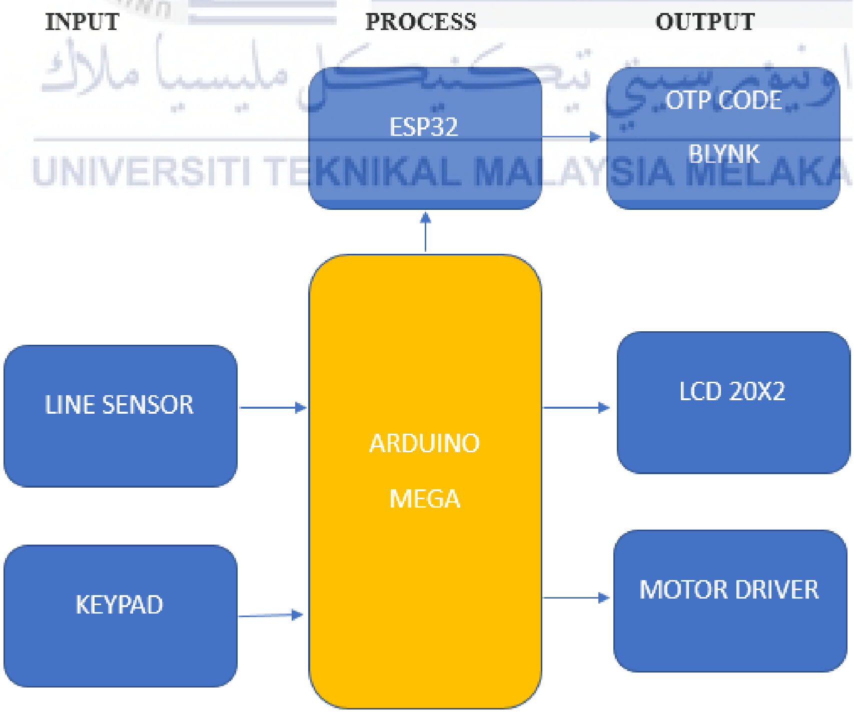


Figure 76 Proteus Software

3.8 Block Diagram of Project



3.9 Function Flowchart

For this project, the the overall system is represented by a general flowchart, illustrating the sequential steps and interactions among key components.The flowchart begins with the robot waits for sender to put package inside the robot storage. After that, the sender will input the destination and the telephone number of the receiver.

Furthermore, the robot will lock the storage for the package and follows the line towards the receiver's destination. When the robot arrives the receiver will receive a TAC code through the receiver's message. Later on, the receiver will input the TAC code and if the code is correct the robot will open the lock to the storage and the user can take the package, if the code is incorrect the robot will display incorrect code. After that, the robot will follow the path line back to its main position to receiver other deliveries.

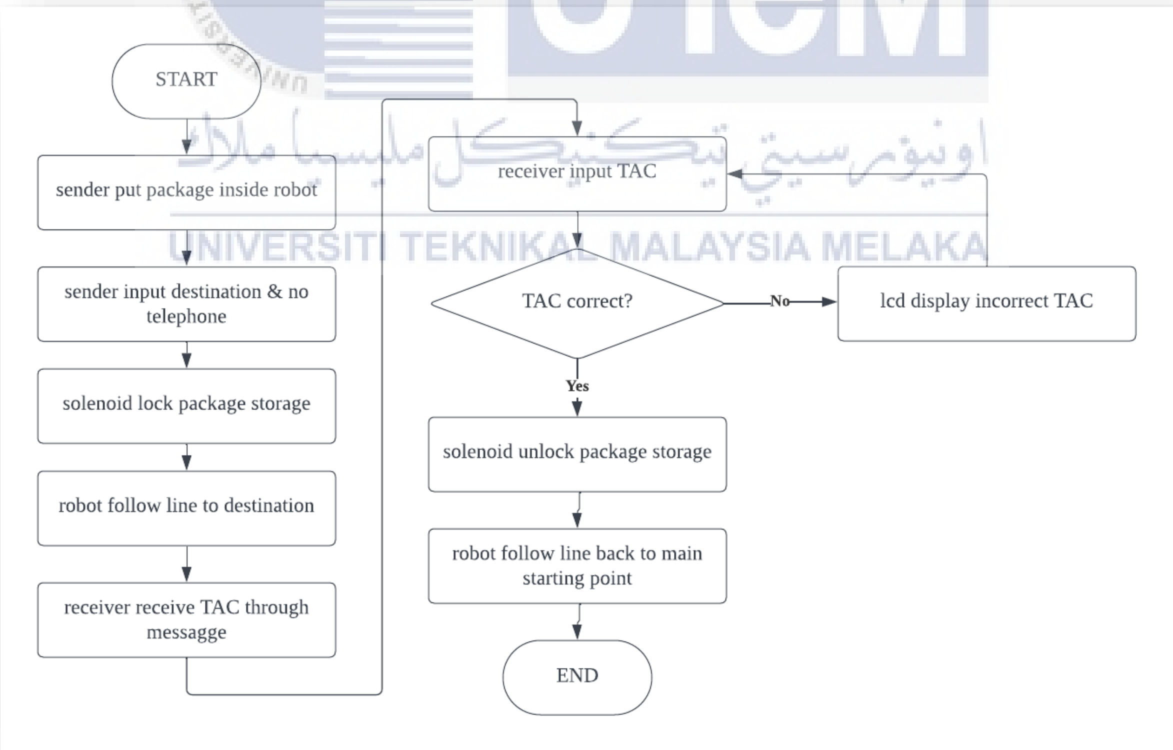


Figure 77 General Flowchart of the Project

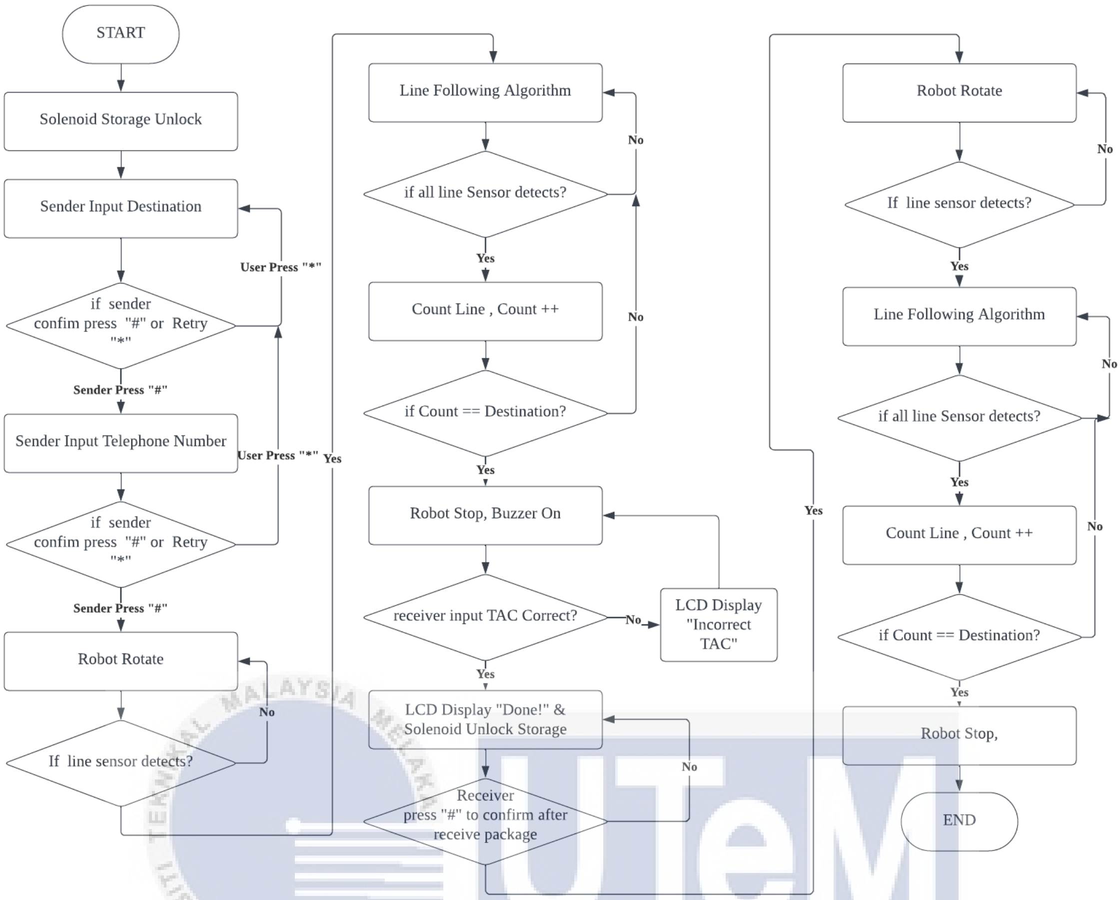


Figure 78 Package Delivering System Flowchart

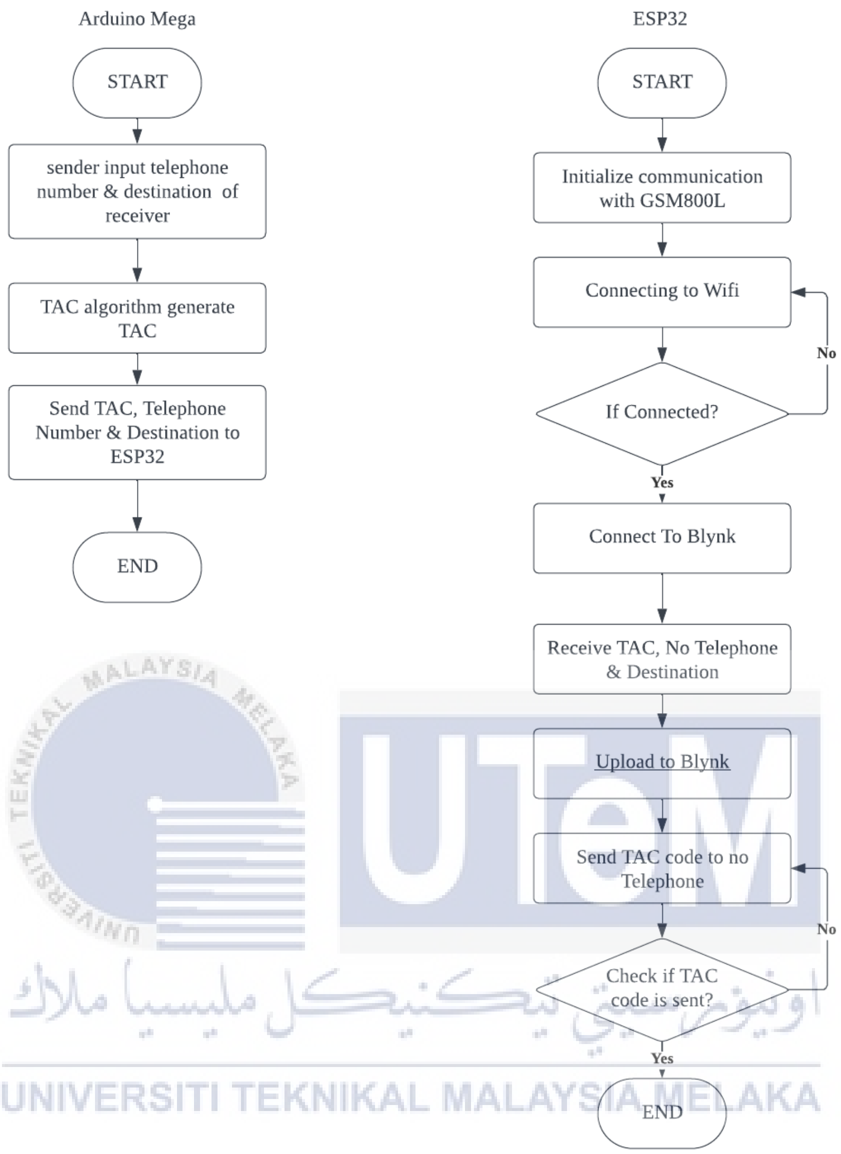


Figure 79 TAC message Sender & Blynk System Flowchart

3.10 Schematic Circuit

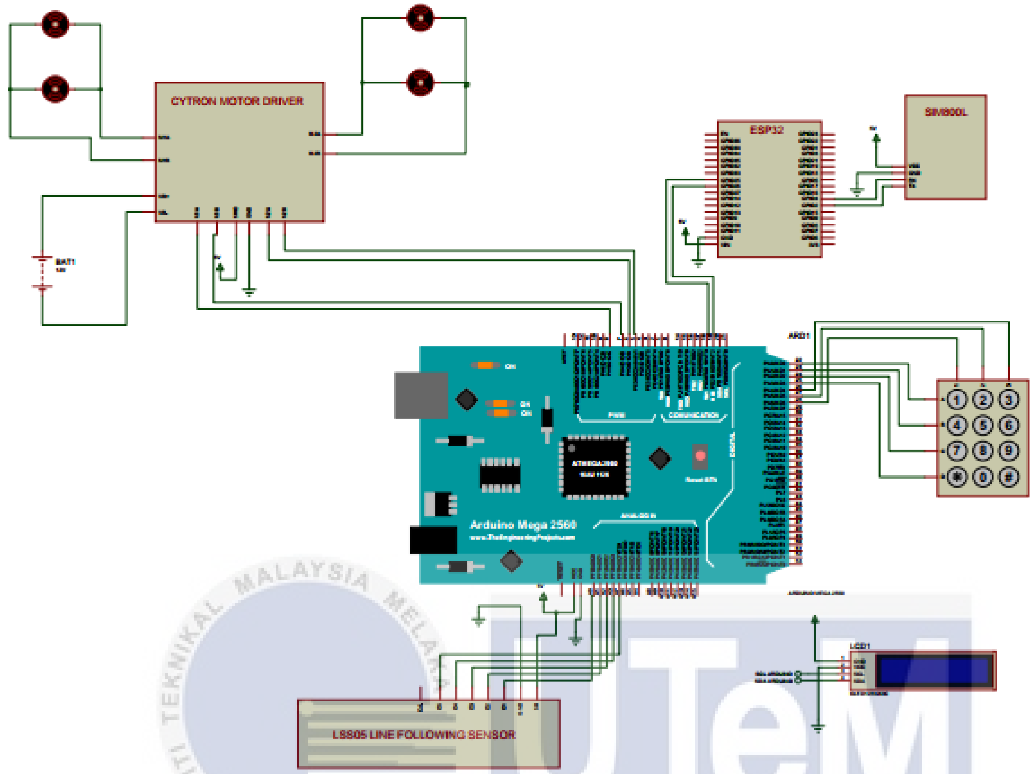


Figure 80 Schematic System of Project

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Introduction

This chapter focuses on the results and discussion obtained on the design and Development of Smart Package Delivery System using Line Following Robot with Passcode Verification. The system programmed for this project also will be discussed in this section. Integration of system on package delivery are done to prove that this project is well functioned. The results achieved are based on three objectives of this project which is to design and developed of Smart Package Delivery System. Next, to program a system on the Line Following Robot with Passcode Verification. Finally, to integrate a system on Passcode Verification.

4.2 Line Following Robot Design

This project core component is the line-following technique, which i use to improve autonomous robotic navigation. An extensive summary of the design factors, hardware elements, and control algorithms used to accomplish accurate and flexible line tracking is given in this section.

For the prototype model of this project,this project use plywood board to form this prototype design because it uses low cost and has strong durability. The design for the robot is trapezoidal and has a height of 19cm x 30cm x 30cm in length and width, I use this size so that documents or paperwork can fit. The shape of the diagonal surface on the front is also because the LCD and keypad surface can be seen clearly by the user.

Our line-following robot integrates four key components essential for its design, the Arduino Mega microcontroller, line-following sensor, motor driver, and DC motor. This four main components is the core for the line following system. The user interface incorporates an LCD display for real-time feedback and a keypad for user input. Additionally, the inclusion of GSM and Blynk establishes a communication and IoT interface, enabling the robot to connect with external networks, providing opportunities for remote monitoring and control.

Moreover, the combination of the Arduino Mega, line-following sensor, motor driver, and DC motor follow the line efficiently, while the The LCD display, keypad, GSM, and Blynk enhance the robot, making it user-friendly and versatile. This setup not only ensures precise movement but also enables interactive and remote functionality.

Figure 81 Full design Project

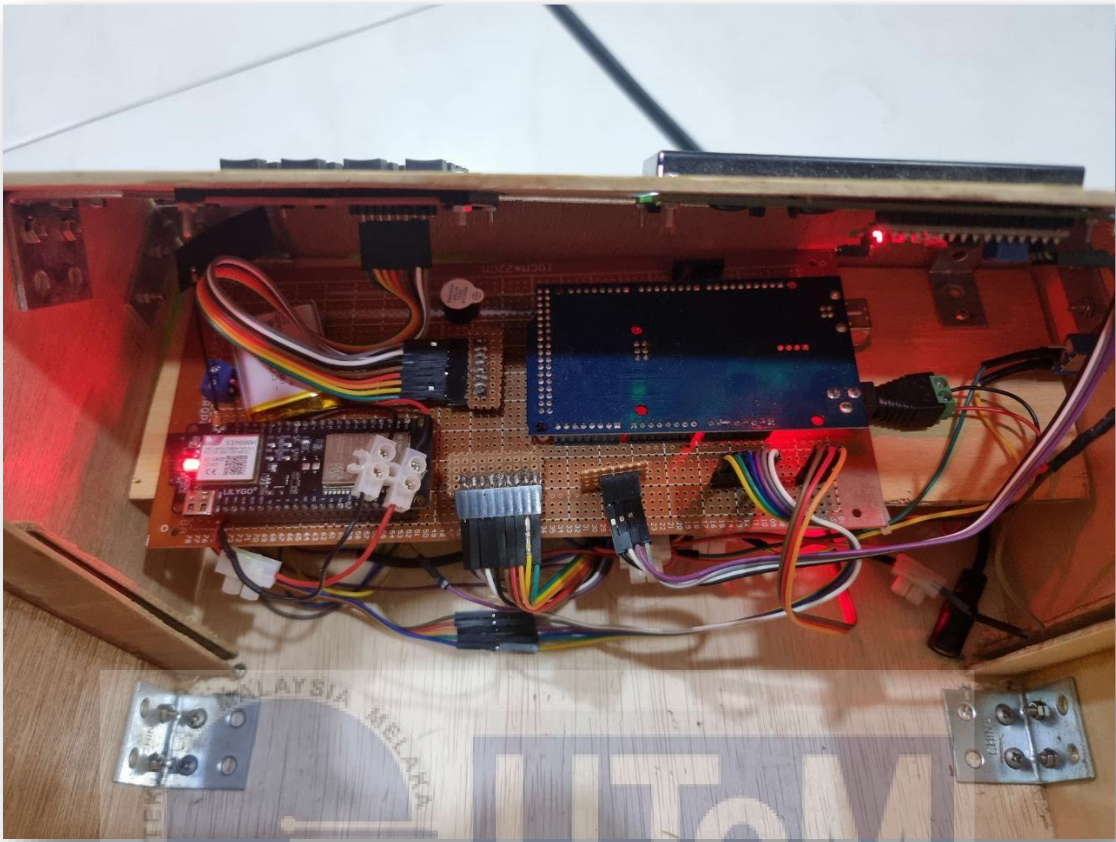


Figure 82 Project Internal Circuit

Figure 83 Side of design

4.2.1 LCD Display

Liquid crystals are used in LCD (Liquid Crystal Display), a kind of flat-panel display technology, to create images. LCDs are frequently used to provide a visual interface in a variety of electronic devices, including Arduino projects.In this project, the LCD will serve as the system's interface. It shows "Destination" at first to prompt the sender to enter the recipient's location.

Figure 84 LCD Display for Input Destination

After that, the sender will have to press the “#” button on the keypad for the system to detect that the destination have been inserted. The input of the “#” button also will indicate the system to move on to the next step that is to wait for the sender to input the receiver’s telephone number

Moreover, when the sender have inserted the telephone number into the system, the robot will then wait for input of the “#” button to start the robot to follow the line untill the receiver’s destination.

Furthermore, when the robot arrived at the destination it will then send a TAC number to the receiver's phone. The receiver will then need to input the TAC number into the robot to unlock the robot storage to receive the package.

Figure 85 LCD Display Input for Phone Number

Figure 86 LCD Display TAC

For the software side of the LCD display, the microcontroller is the core components for the LCD to work. For the program to be compiled and uploaded to the microcontroller, Arduino IDE is used because of its simple to understand design and uses C++ basic coding The coding for the LCD consists of the library "#include<LiquidCrystal_I2C.h>" and the initialize of the library that is "LiquidCrystal_I2C lcd(0x27,20,4)", to initialize the LCD we will have to use “lcd.begin()” and “lcd.backlight()” in the void setup of the arduino program to initialize the use of LCD in the program.

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

// Set the LCD address to 0x27 for a 16 chars and 2 line display
LiquidCrystal_I2C lcd(0x27, 20, 4);

//JSON
#include <ArduinoJson.h>

void setup() {
  lcd.begin();
  lcd.backlight();
```

Figure 87 Coding For LCD

Moreover, the LCD library uses “lcd.setCursor(0,0)” to set the row and column for the lcd before we want to display anything to the LCD. And use “lcd.print()” to display the output on the LCD


```
lcd.setCursor(0,0);
lcd.print("Press '#' for OK");
```

Figure 88 LCD print function

4.2.2 Keypad

One popular input device for Arduino projects is a keypad. It is made out of a 4x4 grid with a matrix of buttons grouped within, for a total of 16 buttons. The Arduino can determine which button was pressed by using a switch that, when pressed, connects a particular row pin to a particular column pin.

In this project we have used a 4x4 keypad for easier use of the project for the sender to input the receiver's destination and telephone number. The keypad also are used for the

receiver to input the TAC number into the system to unlock the package so that the receiver can receive the package

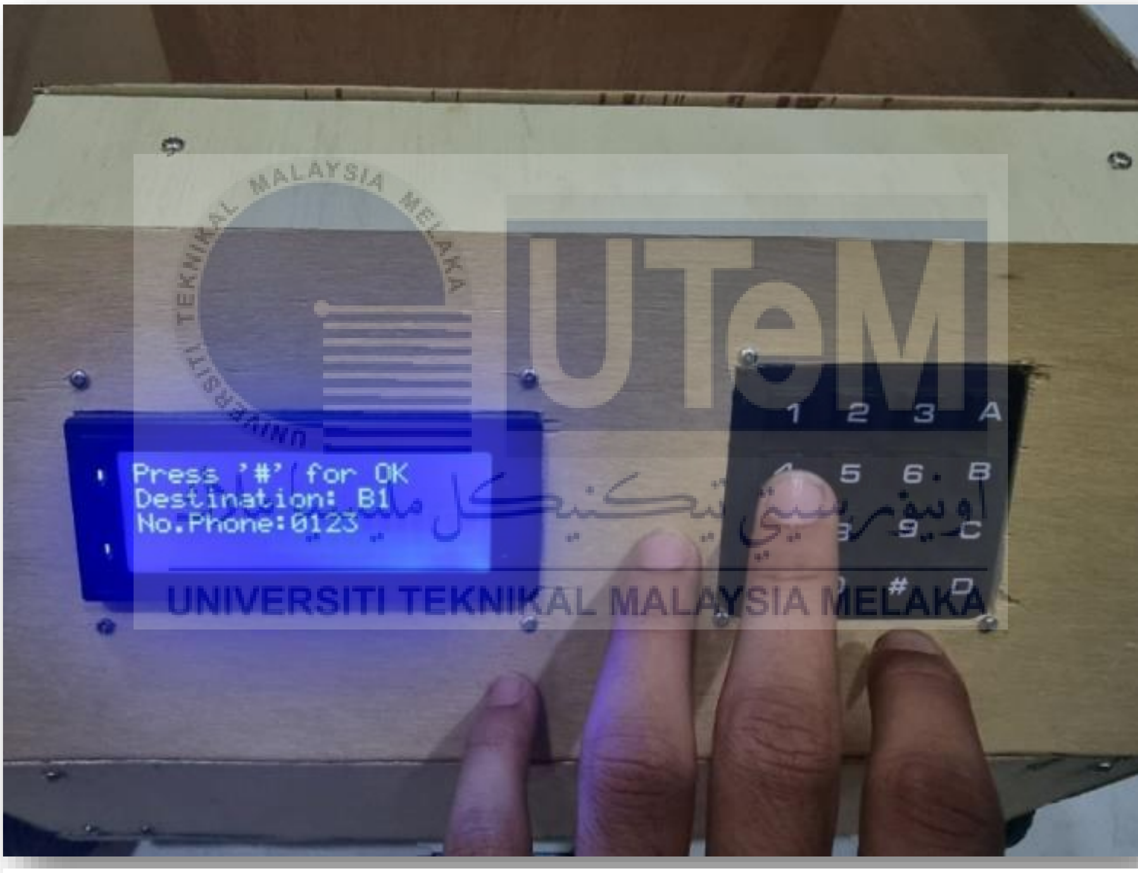


Figure 89 Keypad Input

For the software side of the keypad, in this project the keypad will be used 3 times. Firstly, the keypad will be used to input the destination of the receiver. The program for the keypad, the library and the library initialization for the keypad must be done first.

```
 //KEYPAD
#include <Keypad.h>

const byte ROWS = 4; //four rows
const byte COLS = 4; //three columns
char keys[ROWS][COLS] = {
  {'1','2','3' , 'A'},
  {'4','5','6' , 'B'},
  {'7','8','9', 'C' },
  {'*','0','#' , 'D'}
};
byte rowPins[ROWS] = {34, 35, 36, 37};
byte colPins[COLS] = {30, 31, 32 , 33};
```

Figure 90 Keypad Coding Declaration

After that, when the keypad detects any input it will run through this program to detect and save the data to the microcontroller. In the first keypad input the data will be stored in a String called “location”. After that, the input of the data from the keypad will be stored in other String named “noPhone” and “TAC_String”.


```
case '0':
location = location + "0";
lcd.print(location);
break;

case '1':
location = location + "1";
lcd.print(location);
break;

case '2':
location = location + "2";
lcd.print(location);
break;

case '3':
location = location + "3";
lcd.print(location);
break;
```

```
case 'A':
location = location + "A";
lcd.print(location);
break;

case 'B':
location = location + "B";
lcd.print(location);
break;

case 'C':
location = location + "C";
lcd.print(location);
break;
```

Figure 91 Keypad Coding Input

4.2.3 Motor Configuration

Line following systems are a kind of robotics application where a DC (direct current) motor is frequently utilized. A robot or vehicle with a line following system is intended to follow a path or line on the ground that is usually indicated by a contrasting color. Because DC motors are versatile, easy to regulate, and suitable for a wide range of robotic activities, they are the favored choice for these kinds of applications.

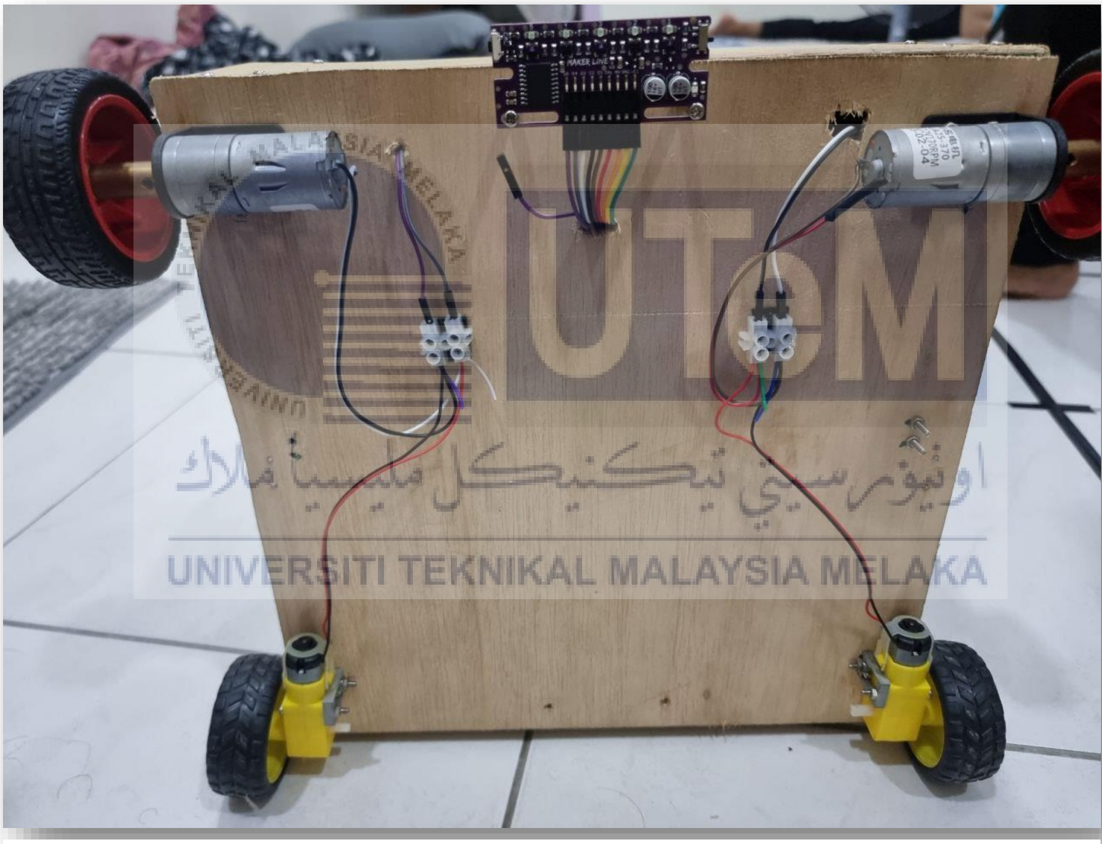


Figure 92 Motor Connection

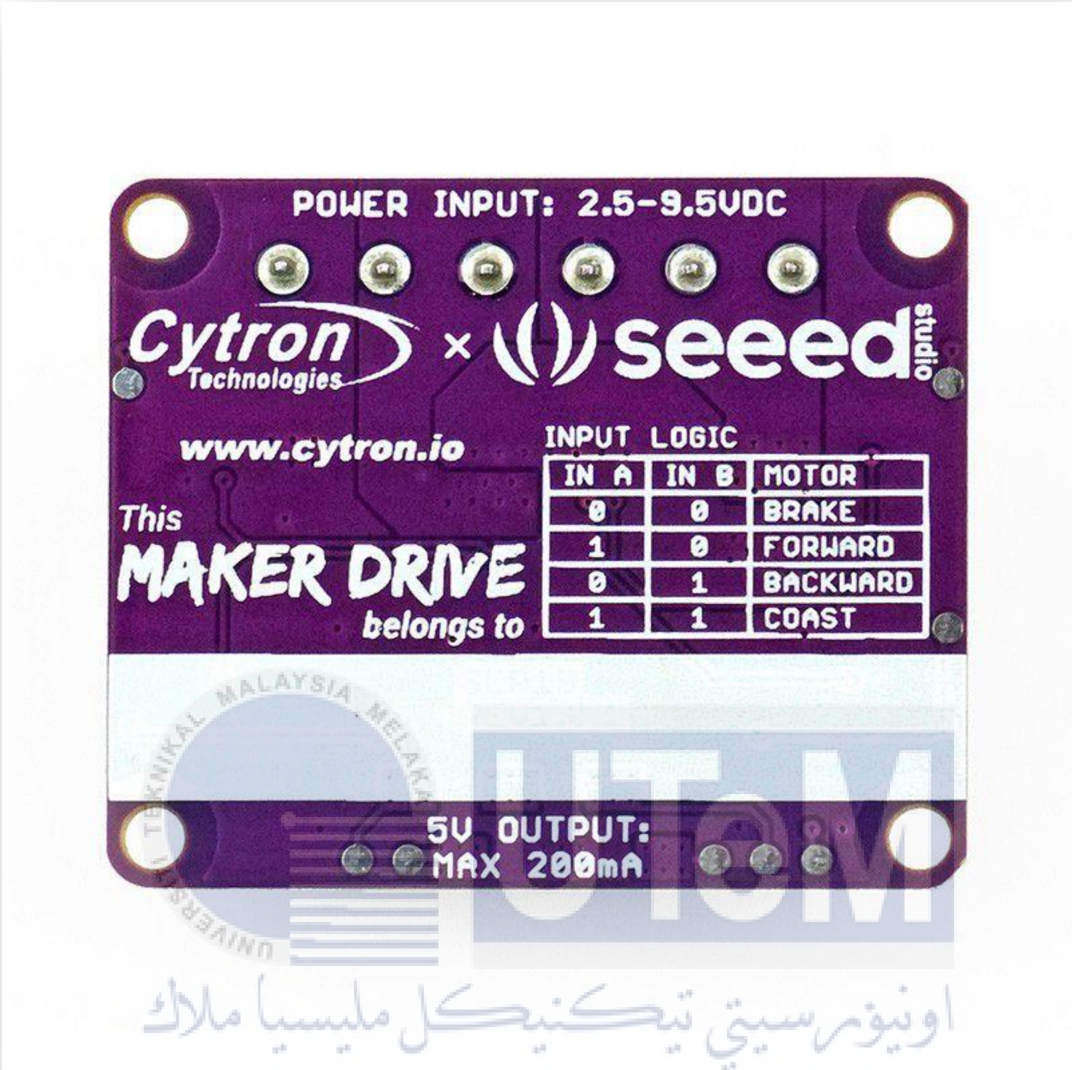


Figure 93 Input Logic for Cytron Motor Driver

For this project DC motors are used to move the robot to its destination. 4 DC motors are used that are connected in a way that 2 motors are connected together at the left side and 2 motors are connected together at the right side. This connection design is because in order to move the robot left or right, either right connected motor will move clockwise and left connected motor will turn counter clockwise.To control the DC motor we have to use motor driver, for this project Cytron Motor Driver H-Bridge is used. The configuration for the motor driver is as below.

In A and In B will determine the position for the motor direction, for example to move left or right, the microcontroller will send 1 to In A and In B. For the software side of the program, the library will have to be define in “#include "CytronMotorDriver.h"” and intialize the left side and the right side of the motor in “CytronMD motor1(PWM_PWM, 5, 6);” and “CytronMD motor2(PWM_PWM, 7, 8);”.

```
//MOTOR DRIVER
#include "CytronMotorDriver.h"

CytronMD motor1(PWM_PWM, 5, 6);   // PWM 1A = Pin 5, PWM 1B = Pin 6.
CytronMD motor2(PWM_PWM, 7, 8);  // PWM 2A = Pin 7, PWM 2B = Pin 8.
```

Figure 94 Motor Driver Coding Declaration

After that, in order to move the motor clockwise or counter clockwise and the speed of each motor “motor1.setSpeed(0);” and “motor2.setSpeed(0);” is use.

```
motor1.setSpeed(-180);  // Motor 1 runs forward at full speed.
motor2.setSpeed(180);  // Motor 2 runs backward at full speed.
```

Figure 95 Motor Speed & Direction Coding

4.2.4 User Interface (Blynk)

A platform called Blynk facilitates simple and adaptable communication between mobile devices and hardware such as Arduino, Raspberry Pi or ESP8266. It enables you to remotely operate and see your Arduino-based projects by creating an interface on your tablet or smartphone.

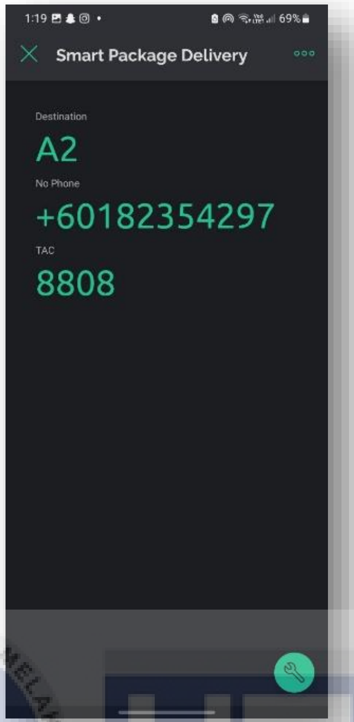


Figure 96 Blynk Interface

In this project the Blynk serves as an interface for the technician or the person in charge of the delivery package robot, to help troubleshoot problems when the robot is delivering packages. The Blynk will display the Destination, Telephone Number and TAC code for the receiver. This data will further help technician to locate the robot if it is stuck in its way to the destination or when the TAC code to unlock the package storage is not unlocked when user did not received the TAC code or TAC code received is wrong.

For the software side of the project, the libraries is first included in the coding such as "#include <BlynkSimpleEsp32.h>" and "#include <WiFiClient.h>". After that, to use the Blynk interface it is required to insert the Template ID , Name and the Authentication Token from the Blynk Interface.

```
//BLYNK
#define BLYNK_PRINT Serial

#define BLYNK_TEMPLATE_ID "TMPL6emIpelGl"
#define BLYNK_TEMPLATE_NAME "Smart Package Delivery"
#define BLYNK_AUTH_TOKEN "feHp7NksJhlaJBp9D8TUQABnP3yzz2Nx"

#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>

char auth[] = "feHp7NksJhlaJBp9D8TUQABnP3yzz2Nx";
char ssid[] = "wifitest";
char pass[] = "wifitest";
```

Figure 97 Blynk Coding Declaration

Furthermore, to connect to the network the Blynk is have to connect to the wifi network, so the wifi name and password must be included such as “ssid[] = wifitest” and “pass[] = wifitest”. So in this coding, the wifi name and password is set to “wifitest”.

```
Blynk.begin(auth, ssid, pass);
```

```
Blynk.run();
```

Figure 98 Blynk Coding

Moreover, to initialize the blynk program “Blynk.begin()” is inserted in the void setup and "Blynk.run()" is included to run the program.

4.2.5 TAC Verification

The GSM Association (GSMA) provides mobile devices with a unique identity called the Type Allocation Code (TAC). In the context of mobile telecommunications, it is used to identify the type of device and manufacturer.

To verify and certify the legality of mobile devices, TAC verification is an essential process that is carried out by industry entities, network operators, and regulatory bodies. By verifying compliance with standards and protecting against the use of illegal or counterfeit devices, this verification maintains mobile networks' security and stability.

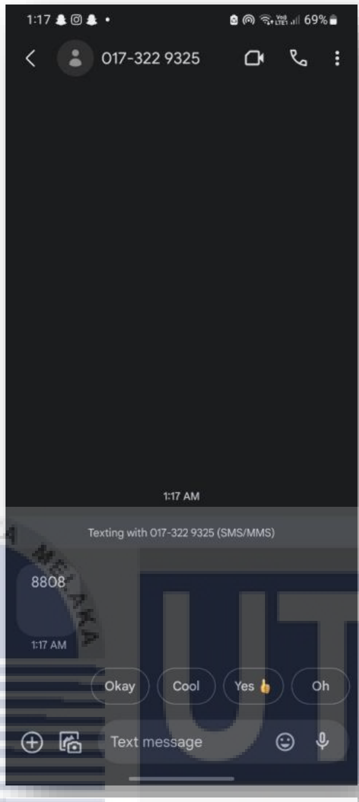


Figure 99 SMS Received

In this project TAC Verification is used to verify and unlock the package storage of the smart package delivery robot. The ESP32 used here is equipped with SIM800L GSM that can act as mobile device to send, receive message or make and receive calls. The SIM800L combined with ESP32 is used to send message containing the TAC Verification to the receiver. For the receiver side of the project the LCD will display “TAC: ” for the receiver to input the TAC code that they receive.

Arduino Mega, the TAC is shown on the LCD display.

Figure 100 LCD Display TAC Input

In the software, the coding for the TAC verification is set at the microcontroller Arduino Mega. TAC code is set to random in “TAC = random(1000,9999);” in range from 1000 untill 9999.

```
if(done_tac == 0){
TAC = random(1000,9999);
done_tac = 1;
}
```

Figure 101 TAC Algorithm Coding

4.3 Line Following System

A line-following robot is a kind of robot that follows a predetermined path or track indicated by a surface line that contrasts with its surroundings. Applications for line following robots are numerous and include educational initiatives and robotics contests. The core idea is to use sensors to identify the line, then modify the robot's actions to maintain its course.

In this project, line following system is used for robot to navigate along the line and to determined which route to take in order to reach its pre-determined destination. Line sensor is used to detect the path so that the robot is able to navigate along the line and avoiding getting off track from its path

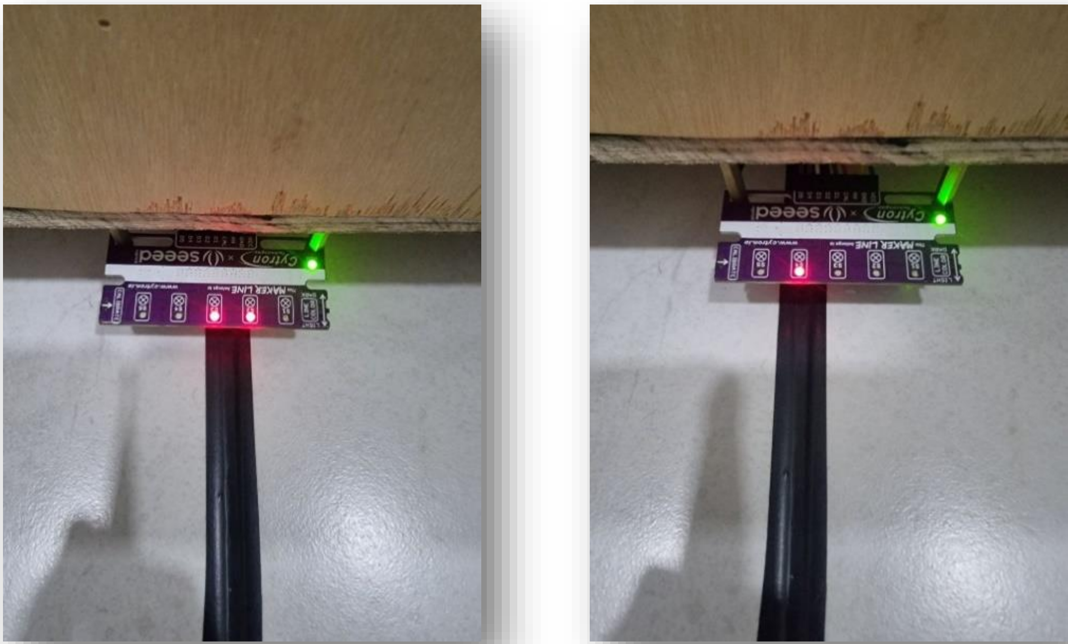


Figure 102 Line Following Sensor Detects

The line sensor is equipped with 5 IR sensors to detect the path line. When the sensor detects change in contrast the LED on each sensor will light up to signal that the IR sensor is detecting. For the line following system in this project, the analog signal from the line following sensor are fed into the microcontroller. The output of this signal will range from 0 – 1000, so if the output of the analog signal is at 500 it means that the line sensor is on the centre of the line.

For the coding of the line following algorithm, this line following system implemented a simple algorithm to improve the line following accuracy. In the coding P, I and D formula is initialized, to obtain the speed of the motor the algortihm is set as follows “int motorspeed = P*Kp + I*Ki + D*Kd;” Moreover, when the basic motor speed is acquired this simple calculation is set to get the motor speed for each motor “int motorspeeda = basespeeda + motorspeed;” and “int motorspeedb = basespeedb - motorspeed;”

```
  uint16_t position = analogRead(AN);

//set point = 500

  int error = 500 - position;

  P = error;
  I = I + error;
  D = error - lastError;
  lastError = error;

  int motorspeed = P*Kp + I*Ki + D*Kd;

  //Serial.println(motorspeed);
  int motorspeeda = basespeeda + motorspeed;
  int motorspeedb = basespeedb - motorspeed;

  if (motorspeeda > maxspeeda) {
    motorspeeda = maxspeeda;
  }
  if (motorspeedb > maxspeedb) {
    motorspeedb = maxspeedb;
  }
  if (motorspeeda < 0) {
    motorspeeda = 0;
  }
  if (motorspeedb < 0) {
    motorspeedb = 0;
  }
```


Figure 103 Line Following Algorithm Coding

Additionaly, the line following system uses horizontal black lines so that the system is able to track the position of the robot in the circuit path. This is because in order to make any decision or action for the robot, its is required to know its location in the circuit path, for example when in the circuit the robot should make a left or right turn. Besides that, the system must also be able to recognise if the robot has reached its destination. Therefore, the tracking system of the robot is required, the operation of this system is when the line sensor detects any horizontal black line, the system will count the number of horizontal black line in the circuit path.

In the software side, "Count += 1; " is used to count the line to ensure the system to track the number of each horizontal line and will able to decide for its next decision.

```
//STOP LINE
  if(right == 1 && midright == 1 && mid == 1 && midleft == 1 && left == 1){
  unsigned long currentMillis = millis();

  if(currentMillis - previousMillis >= 500){
  previousMillis = currentMillis;
  motor1.setSpeed(0);
  motor2.setSpeed(0);
  Count += 1;
  }
```

Figure 104 Line Following System Coding

4.4 Package Delivery Accuracy

A line-following package delivery robot's accuracy must be measured in order to evaluate its overall effectiveness and performance. Precise line following guarantees that the robot can follow designated routes without deviating, which is essential for dependable and prompt delivery of packages. Delivery times and productivity are maximized by the efficiency that comes from precise navigation.

Furthermore, assessing accuracy provides information on the line-following system's overall reliability. For a delivery robot to operate consistently, its navigation system must be well-designed and set up, as shown by consistent accuracy. It enables continuous modifications to increase the robot's accuracy and overall performance by pointing out areas that need development.

For the scope of this analysis, 4 destinations of the circuit path is created with length of 2.5m and each junction with width of 1m. The aim of the analysis is to obtain the accuracy the package delivery robot to each destination. Besides that, the efficiency of the package delivery robot to each destination will also be analyzed.

For the analysis, the package delivery robot will deliver package to one destination repeatedly for a test span of 10 times. After that, the destination is change and the test will be repeated for each of the destination.

Figure 105 Line Following Robot Delivering Package

Table 6 A1 Destination Table

Destination Room 1 (A1)	Result
Test 1	PASS
Test 2	PASS
Test 3	PASS
Test 4	PASS
Test 5	PASS
Test 6	PASS
Test 7	FAIL – STUCK WHEN TURNING
Test 8	PASS
Test 9	PASS
Test 10	FAIL - SENSOR DID NOT DETECT WHEN TURNING RIGHT

Based on the tests the robot have run 10 delivery to the destination "A1" and out of the 10 tests, 2 run has fail the reason that have been observed is the robot is stuck when turning.

Table 7 A2 Destination Table

Destination Room 2(A2)	Result
Test 1	PASS
Test 2	PASS
Test 3	FAIL – LINE SENSOR GLITCHING
Test 4	FAIL – STUCK WHEN TURNING
Test 5	PASS
Test 6	PASS
Test 7	PASS
Test 8	FAIL
Test 9	PASS
Test 10	PASS

Based on the tests the robot have run 10 delivery to the destination "A2" and out of the 10 tests, 2 run has fail the reason that have been observed is the line sensor is glithching, resulting stuck when turning and the robot is stuck when turning.

Table 8 B1 Destination Table

Destination Room 3 (B1)	Result
Test 1	PASS
Test 2	PASS
Test 3	PASS
Test 4	FAIL – STUCK WHEN TURNING
Test 5	PASS
Test 6	FAIL – STUCK WHEN TURNING
Test 7	FAIL – STUCK WHEN TURNING
Test 8	PASS
Test 9	FAIL - SENSOR DID NOT DETECT WHEN TURNING RIGHT
Test 10	FAIL – SENSOR DID NOT DETECT WHEN TURNING RIGHT

Based on the tests the robot have run 10 delivery to the destination “B1” and out of the 10 tests, 5 run has fail the reason that have been observed is the robot is stuck when turning and the sensor did not detect when turning right.

Table 9 B2 Destination Table

Destination Room 4 (B2)	Result
Test 1	PASS
Test 2	PASS
Test 3	PASS
Test 4	PASS
Test 5	FAIL – STUCK WHEN TURNING
Test 6	FAIL – STUCK WHEN TURNING
Test 7	PASS
Test 8	PASS
Test 9	PASS
Test 10	FAIL – STUCK WHEN TURNING

Based on the tests the robot have run 10 delivery to the destination “B2” and out of the 10 tests, 3 run has fail the reason that have been observed is the robot is stuck when turning.

By observing the data, we are able to compute the accuracy by calculating the Delivery Success Rate and the Error rate, as below:

Table 10 Result Analysis Table

Destination	Result		Total
	Fail	Pass	
ROOM 1 (A1)	2	8	10
ROOM 2 (A2)	2	8	10
ROOM 3 (B1)	5	5	10
ROOM 4 (B2)	3	7	10
Total	12	28	40

Delivery Success Rate :

$$Delivery\ Success\ Rate = \left(\frac{Succesful\ Deliveries}{Total\ Deliveries}\right) x\ 100\%$$

$$Delivery\ Succes\ Rate = \left(\frac{28}{40}\right) x\ 100\%$$

$$Delivery\ Succes\ Rate = 70\%$$

Error Rate :

$$Error\ Rate = (\frac{Failed\ Deliveries}{Total\ Deliveries}) x\ 100\%$$

$$Error\ Rate = (12/\ 40)\ x\ 100\%$$

$$Error\ Rate = 30\%$$

As a result for the computing the data above, we are able to get the Delivery Success Rate that is 70% out of the total deliveries and 30% for the fail deliveries. For the conclusion of the outcome the Smart Package delivery robot has an accuracy of 70% of accuracy for delivering packages to the destination.

4.4.1 Analysis Result for Package Delivery Efficiency

After running test for the Smart Package Delivery Robot, it has run a total of 40 test run to 4 different location that is A1, A2, B1 and B2 and as a result the total fail test is 12.

In each failed run we have observed the reason or possible causes for the fail and can be concluded as below:

Table 11 Pareto Chart Table

Fail Reason	Number of times	Cummulative Frequency
Stuck When Turning	8	66.67%
Sensor Did not Detect When Turning	3	91.67%
Line Sensor Glitching	1	100.00%
Total	12	

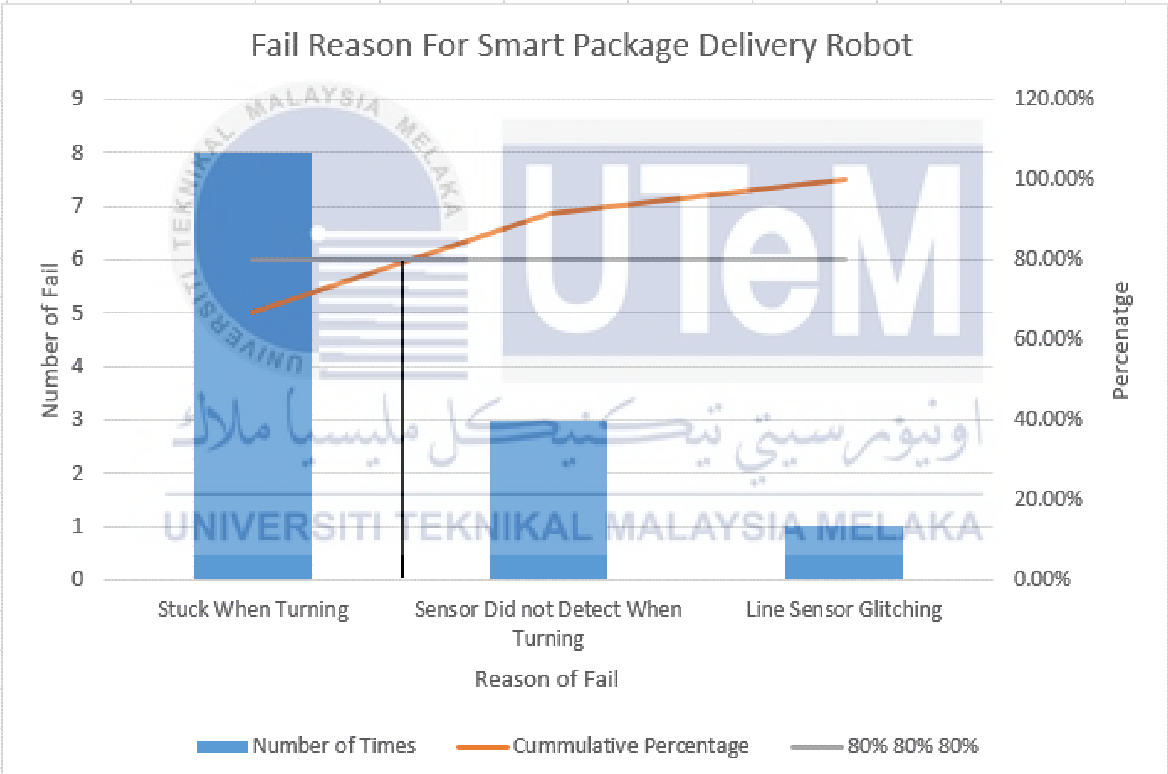


Figure 106 Pareto Chart for Analysis 1

As a summary of the analysis on the fail reason for the smart package delivery robot, the effeciency of the system will improve greatly if we further investigate on the main problem of the robot navigation system, in this case the line following system that is the problem stuck when turning. Based on the pareto chart above, the package delivery robot should focus on tackling the highest fail reason in order to improve its overall effeciency.

4.5 Package Delivery with Passcode Verification System

A passcode is used in a package delivery with passcode verification system as a safe way to confirm and approve the delivery of a package to the intended recipient. When a recipient may not be there in person when the delivery occurs or when an extra degree of security is needed, this mechanism is frequently put into place.

In this project, TAC verification is used as a verification method for the receipent to receive their package securely. TAC code is used so that the code generated is random and cannot be intercept by others rather than the receipent. When the sender input into the system the telephone number of the receiver, TAC code will be sent with SIM800L GSM through message app in the phone.

In this system, the main components are the microcontroller Arduino Mega, ESP32 , SIM800L GSM and the solenoid lock. Arduino Mega will first create the TAC code when sender input the destination and telephone number of the receiver, the TAC code will then send to the ESP32 that will communicate with the SIM800L GSM that will send message to the receiver phone number.

```
StaticJsonDocument<200> doc;

 doc["TAC"] = TAC;
 doc["noPhone"] = new_noPhone ;
 doc["location"] = location;

Serial.println(new_noPhone);
Serial.println(TAC);
Serial.println(location);
serializeJson(doc, Serial3);
```

```
if (mySerial.available())
{
  StaticJsonDocument<1024> doc;

  DeserializationError err = deserializeJson(doc, mySerial);

  doc.isNull();

  if (err == DeserializationError::Ok )
  {

    TAC = doc["TAC"].as<String>();
    noPhone = doc["noPhone"].as<String>();
    location = doc["location"].as<String>();


  }
  else
  {
    Serial.print("deserializeJson() returned ");
    Serial.println(err.c_str());

    while (mySerial.available() > 0)
      mySerial.read();
  }
```

Figure 107 ArduinoJson Communication Coding

For the software side of the system, ArduinoJson is used for communication of the two microcontroller boards. The sender Arduino Mega will send the TAC, Phone Number and Destination to the receiver ESP32. When ESP32 receive data from the Arduino Mega, it will then open up communication with the SIM800L GSM and send the TAC code to the phone number received.


```
last_noPhone.toCharArray(PhoneNoChar,60);
last_TAC.toCharArray(tacChar,60);

if(noPhone != "" && Sent == 0){
  sim8001.sendSMS(PhoneNoChar ,tacChar );
  Sent = 1;
}
```

Figure 108 GSM800L Send Message Coding

When the receipent have received the TAC code and the smart package delivery robot has arrived to its destination, the receipent will have to input the code to the robot.

Figure 110 TAC Input Correct and Incorrect LCD Display

When the TAC code is inserted and the user has pressed "#" to confirm, the system will then run through an if-else statement to check if the TAC code is correct or not. If it is correct, the LCD will display "Done! Press #" to signify that the code entered is correct. Otherwise if the TAC code is wrong it will display "Try Again! Press *" to indicate that the TAC code is entered wrong.

```
//ORDER COMPLETE
if (keypad_status == 3 && TAC_pass == TAC){
  //order_complete = 1;
  digitalWrite(RELAY_PIN, LOW);//UNLOCK
  lcd.setCursor(0,2);
  lcd.print("Done! Press #");

}
if (keypad_status == 3 && TAC_pass != TAC){
  TAC_pass = 0;
  TAC_String = "";
  lcd.setCursor(0,2);
  lcd.print("Try Again! Press * ");
}
```

Figure 109 Package System Lock & Unlock

Next, when the TAC code is entered correct, the solenoid lock will open and the storage of the Smart Package Delivery Robot will be accessible for the receipent to receive their package. After that, when package has succesfully received, the solenoid lock will lock the storage and the robot will return on its way back to main position.

Figure 111 Solenoid Lock & Unlock

The lock and unlock of the solenoid if the TAC entered is correct is located in the main loop of the program in the microcontroller, in the if-else statement when “”TAC_pass == TAC’’ the solenoid will unlock using “digitalWrite(RELAY_PIN, HIGH).”


```
if(keypad_status > 3 && TAC_pass == TAC ){
    digitalWrite(RELAY_PIN, HIGH);//LOCK
    order_complete = 1;
  }
```

Figure 112 Coding for unlocking Solenoid Lock

4.5.1 Passcode Verification System Analysis

An analysis of a system intended to use passcodes to confirm the identification of users or entities is referred to as a passcode verification system analysis. The purpose of this analysis is to evaluate the passcode verification system's advantages, disadvantages, and general effectiveness.

In this project, the aim of the analysis is to determine the accuracy of verifying the TAC code and the solenoid lock unlocking the storage for the package to be received. Therefore, the test is to receive and input the correct TAC code and observing the result if the solenoid lock will unlock the storage. The scope of the test is to set the robot to deliver to one destination and to run the test repeatedly without reseting the system. The data is shown as below when the robot is delivering to destination "A1" and input the TAC code using the keypad into the system.

Table 12 Table Passcode Verification

TAC Verification	Result
Test 1	PASS
Test 2	PASS
Test 3	PASS
Test 4	PASS
Test 5	FAIL – NO LINE CONNECTION
Test 6	FAIL – NO LINE CONNECTION
Test 7	PASS
Test 8	PASS
Test 9	FAIL – SYSTEM FAIL TO SEND
Test 10	PASS

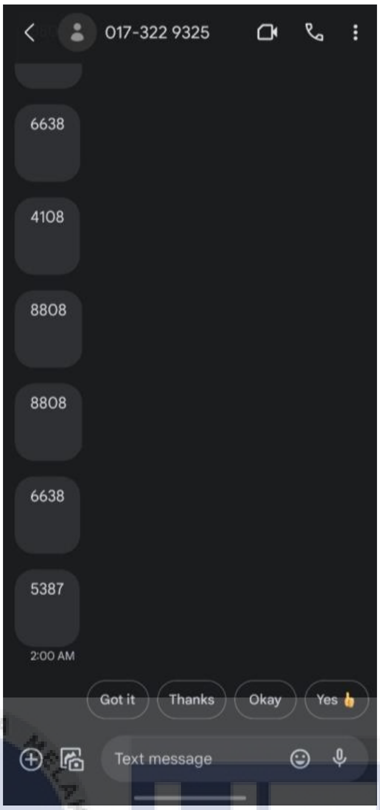


Figure 113 TAC Verification message

From the data above, the result of passcode verification of when running 10 test, the result is out of 10 test 3 run is fail. Based on the observation of the fail run, the core problem that causes it to fail is the TAC code is not sending to the receiver thus, not being able to input the correct TAC code into the system. This problem maybe because of any bug in the system or there are no line connection to the sim-card.

For the Passcode verification system accuracy, using the data we can calculate the percentage of its accuracy such as :

$$Passcode\ Verification\ Accuracy$$
$$= (Successful\ Verifications / Total\ Verifications)\ x\ 100\%$$
$$= \left(\frac{7}{10}\right) x\ 100\% = 70\%$$

Based on the calculated data the accuracy of the passcode verification lock is 70% of the overall 10 test run. Here we can conclude that the passcode verification is above the define percentage and is considered accurate.

4.6 Summary

In conclusion, data accuracy analysis and its effectiveness is crucial since it offers useful data, assists with accurate decision-making, and enhances the general effectiveness and success of the overall system. In this project, the acquired data helps to get more insight on the problems of each system, for example in the package delivery system, most of the problem is cause by turning of the robot. The line sensor of the package delivery robot is not able to detect the line after the turning of the robot and causes the robot to be stucked.

By this data we can also conclude that the overall accuracy of the smart package robot to accurately deliver package to each destination. Moreover, the data from the passcode verification system also has provided significant data that shows the problem with the system. TAC code is either not sent by the robot because of the bug in the system or the line connectivity of the sim-card in not available. Therefore, will cause the passcode verification system test run to fail. To conclude, after all the data we compiled and calculated the accuracy percentage for the package delivery system is 70% with 30% error rate and the accuracy percentage for the passcode verification system is the same 70% accuracy.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Introduction

We have examined the importance of our smart package delivery system in this paper, evaluating its precision, effectiveness, and overall impact on the delivery environment. Let's now connect all the pieces from our investigation and consider the broader implications of our conclusions. The conclusion serves as a summary as well as a point of entry into the helpful suggestions and insights that will direct the course of our project moving forward and add to the continuing discussion on dependable and effective package delivery methods.

5.2 Future Works

In considering how our smart package delivery project will develop in the future, a number of important areas stand out as needing more investigation and improvement. Future research could explore improved route optimization algorithms and use sophisticated machine learning models to dynamically modify delivery routes in response to real-time variables like traffic patterns, weather, and road closures.

Moving forward, a significant improvement in the course of our smart package delivery project is switching from Arduino to a dedicated microprocessor architecture. This is a strategic choice motivated by the requirement for a more customized and scalable approach to meet the changing needs of our system. As opposed to Arduino, a microprocessor offers more freedom and control over hardware configurations and real-time decision-making processes. By using a microprocessor, we can better tailor the system's capabilities to meet our needs, maximizing performance and responsiveness.

In the future, cloud computing integration into our system architecture will be a crucial development in the path of our smart package delivery project. This innovative strategy is motivated by the understanding of the revolutionary potential that cloud connectivity offers in terms of scalability, data accessibility, and real-time responsiveness. The smart package delivery system can securely store and centralize large volumes of data, enabling full analytics and insights, by connecting to the cloud.

5.3 Potential for Commercialization

The increasing demand for innovative logistics solutions in the growing e-commerce and supply chain sectors presents an important commercialization potential for the smart package delivery concept. This system covers an essential niche through a strong emphasis on accuracy, productivity, and cutting-edge technological integration. It provides a solution that can keep up with the growing demands of a market that is changing quickly. The smart package delivery system's cutting-edge features—such as the addition of passcode verification and a line-following system—present a unique selling point that sets it apart from competing products. Additionally, the project's flexibility—especially in light of the use of scalable technology like microprocessors and cloud connectivity—puts it in a position to act as an adaptable option for companies of various sizes.

5.4 Conclusion

With the help of Arduino ATMEGA2560, ESP 32, Blynk Application OTP Code number, a practical solution is offered to create the intelligent system for university students, lecturer and staff to receive their package and parcel safely with the implement of line following robot which offers faster process using advanced technology on Internet Of things. Future goals for this technology include offering a system and services for package delivery in universities. The user will benefit from the innovative robot service technology developed by this system.

This leads to the project's stated goal of implementing "Smart Package Delivery System using Line Following Robot with Passcode Verification ". This project was able to fulfil the first objective regarding the capability to design and developed the Smart Package Delivery System using Line Following Robot with Passcode Verification by using Proteus. Implementation of Blynk Application, manage to provide the lecturer to receive a OTP Code number for security purpose and delivered system. It resulted in the development of contemporary software and IOT technologies, which may be enhanced to become more sophisticated and practical.

In this proposed system, the method will be done using Arduino Mega that is interfaced with ESP32. Arduino Mega deliver all the message to other component. There are three objectives accomplished in this proposal to create a system in such a sense. This project is successfully developing a complete design Smart of Package Delivery System using Line Following Robot with Passcode Verification .The programming coding successfully compile and gain result to connect it to the hardware as the project functioned well. Last objectives that successfully achieved is on the integration between the hardware of this project and the Internet of things which is ready to be access.

REFERENCES

[1] M. Agarwal and A. R. Kumar, "IOT based Smart Delivery Box."

[2] L. B. A and A. Professor, "SMART AI BASED DELIVERY ROBOT." [Online]. Available: www.irjmets.com

[3] W. Tu, T. Zhao, B. Zhou, J. Jiang, J. Xia, and Q. Li, "OCD: Online Crowdsourced Delivery for On-Demand Food," *IEEE Internet Things J*, vol. 7, no. 8, pp. 6842–6854, Aug. 2020, doi: 10.1109/JIOT.2019.2930984.\

[4] I.T.S. Engineering College. Computer Science & Engineering Department, Institute of Electrical and Electronics Engineers. Uttar Pradesh Section, and Institute of Electrical and Electronics Engineers, *2015 International Conference on Computer and Computational Sciences (ICCCS) : 27-29 January 2015.*

[5] E. Kuantama, A. Brian, L. Lukas, and P. B. Mardjoko, "Simple delivery robot system based on line mapping method", [Online]. Available: https://www.researchgate.net/publication/283650680

[6] F. Zhang and L. Chen, "OTP_SAM: DHCP security authentication model based on OTP."

[7] Manipal University Jaipur. School of Computing and Information Technology and Institute of Electrical and Electronics Engineers, *2019 2nd International Conference on Intelligent Communication and Computational Techniques (ICCT) : Manipal University, Jaipur, Sep. 28-29, 2019*

[8] M. H. Hersyah, D. Yolanda, and H. Sitohang, "Multiple laboratory authentication system design using fingerprints sensor and keypad based on microcontroller," in *2020*

International Conference on Information Technology Systems and Innovation, ICITSI 2020 - Proceedings, Institute of Electrical and Electronics Engineers Inc., Oct. 2020, pp. 14–19. doi: 10.1109/ICITSI50517.2020.9264969.

[9] A. S. J. Ali *et al.*, "Towards IoT-based Notification System for Agriculture Electric Fence," in *2020 IEEE Student Conference on Research and Development, SCOReD 2020*, Institute of Electrical and Electronics Engineers Inc., Sep. 2020, pp. 269–273. doi: 10.1109/SCOReD50371.2020.9250958.

[10] A. A. Abdulkareem Alawsi, B. H. Jasim, and S. M. Raafat, "Design and implementation of a global smart box for quadcopter delivery system," in *Proceedings of the 3rd International Conference on Intelligent Sustainable Systems, ICISS 2020*, Institute of Electrical and Electronics Engineers Inc., Dec. 2020, pp. 1408–1415. doi: 10.1109/ICISS49785.2020.9316012.

[11] M. Agarwal and A. R. Kumar, "IOT based Smart Delivery Box."

[12] R. Safin, R. Lavrenov, K. H. Hsia, E. Maslak, N. Schiefermeier-Mach, and E. Magid, "Modelling a TurtleBot3 Based Delivery System for a Smart Hospital in Gazebo," in *SIBCON 2021 - International Siberian Conference on Control and Communications*, Institute of Electrical and Electronics Engineers Inc., May 2021. doi: 10.1109/SIBCON50419.2021.9438875.

[13] S. S. N, S. R. V, S. M. V, and U. Student, "Smart Box Delivery System For Faster And Safer Delivery," 2021. [Online]. Available: www.ijcrt.org

[14] S. Eeshwaroju, P. Jakkula, and I. Abdellatif, "An IoT based three-dimensional dynamic drone delivery (3D4) system," in *Proceedings - 2020 IEEE Cloud Summit, Cloud Summit 2020*, Institute of Electrical and Electronics Engineers Inc., Oct. 2020, pp. 119–123. doi: 10.1109/IEEECloudSummit48914.2020.00024.

[15] S. Dustdar, “Web-Scale Workflow Building Energy Management Systems,” 2013. [Online]. Available: www.computer.org/internet/

[16] W. Tu, T. Zhao, B. Zhou, J. Jiang, J. Xia, and Q. Li, “OCD: Online Crowdsourced Delivery for On-Demand Food,” *IEEE Internet Things J*, vol. 7, no. 8, pp. 6842–6854, Aug. 2020, doi: 10.1109/JIOT.2019.2930984.

[17] Institute of Electrical and Electronics Engineers, *2019 8th International Conference on Modeling Simulation and Applied Optimization (ICMSAO).*

[18] Institute of Electrical and Electronics Engineers, *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS) : 6-7 Jan. 2017.*

[19] Institute of Electrical and Electronics Engineers, *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS) : 6-7 Jan. 2017.*

[20] “Design Implementation of High-Performance Line following robot”.

[21] International Islamic University Chittagong, Institute of Electrical and Electronics Engineers. Bangladesh Section, and Institute of Electrical and Electronics Engineers, *2018 International Conference on Innovations in Science, Engineering and Technology : ICISET 2018 : International Islamic University Chittagong, Chittagong, Bangladesh : 27-28 October 2018.*

[22] N. Maniha Abdul Ghani, F. Naim, and T. Piow Yon, “Two Wheels Balancing Robot with Line Following Capability.”

[23] A. Ma’arif, A. A. Nuryono, and Iswanto, “Vision-Based Line following Robot in Webots,” in *Proceeding - 1st FORTEI-International Conference on Electrical Engineering, FORTEI-ICEE 2020*, Institute of Electrical and Electronics Engineers Inc., Sep. 2020, pp. 24–28. doi: 10.1109/FORTEI-ICEE50915.2020.9249943.

[24] A. Ma'arif, A. A. Nuryono, and Iswanto, "Vision-Based Line following Robot in Webots," in *Proceeding - 1st FORTEI-International Conference on Electrical Engineering, FORTEI-ICEE 2020*, Institute of Electrical and Electronics Engineers Inc., Sep. 2020, pp. 24–28. doi: 10.1109/FORTEI-ICEE50915.2020.9249943.

[25] SCAD College of Engineering and Technology and Institute of Electrical and Electronics Engineers, *Proceedings of the International Conference on Trends in Electronics and Informatics (ICOEI 2019) : 23-25, April 2019*.

APPENDIX

```
//KEYPAD
#include <Keypad.h>

const byte ROWS = 4; //four rows
const byte COLS = 4; //three columns
char keys[ROWS][COLS] = {
  {'1','2','3' , 'A'},
  {'4','5','6' , 'B'},
  {'7','8','9', 'C' },
  {'*','0','#' , 'D'}
};
byte rowPins[ROWS] = {34, 35, 36, 37};
byte colPins[COLS] = {30, 31, 32 , 33};

String location = "";
String noPhone = "";
String TAC_String = "";
int TAC_pass = 0;
int keypad_status=0;
int order_complete = 0;
int go_dest = 0;

Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );

//LINE FOLLOWING SENSOR
#define IN1 53
#define IN2 52
#define IN3 51
#define IN4 50
#define IN5 40
#define CAL 3
#define AN A1

int right, midright,mid,midleft,left;

int turnright = 0;
int turnleft = 0;
int turnaround = 0;
int rotate = 0;

//TAC
int TAC = 0 ;
int done_tac = 0;

//
```

```
#define buzzer_pin 2

//RELAY
#define RELAY_PIN 10

//MOTOR DRIVER
#include "CytronMotorDriver.h"

CytronMD motor1(PWM_PWM, 5, 6);   // PWM 1A = Pin 5, PWM 1B = Pin 6.
CytronMD motor2(PWM_PWM, 7, 8);  // PWM 2A = Pin 7, PWM 2B = Pin 8.

#define MOTOR_TURN_SPEED 160

//PID
float Kp = 0.3; //related to the proportional control term;
            //change the value by trial-and-error (ex: 0.35).
float Ki = 0; //related to the integral control term;
            //change the value by trial-and-error (ex: 0.0008).
float Kd = 0.5 ; //related to the derivative control term;
            //change the value by trial-and-error (ex: 0.6).
int P;
int I;
int D;

int lastError = 0;
boolean onoff = false;

const uint8_t maxspeeda = 180; //180
const uint8_t maxspeedb = 180; //180
const uint8_t basespeeda = 140;//140
const uint8_t basespeedb = 140;//140


//COUNTING
int Count = 0 ;
int done = 0;
int finish = 0;
unsigned long previousMillis = 0;
unsigned long previousMillis1 = 0;

//LCD
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

// Set the LCD address to 0x27 for a 16 chars and 2 line display
LiquidCrystal_I2C lcd(0x27, 20, 4);

//JSON
#include <ArduinoJson.h>
```

```
void setup() {
  lcd.begin();
  lcd.backlight();

  Serial.begin(9600);
  Serial3.begin(4800);

  pinMode(IN1,INPUT);
  pinMode(IN2,INPUT);
  pinMode(IN3,INPUT);
  pinMode(IN4,INPUT);
  pinMode(IN5,INPUT);
  pinMode(AN, INPUT);

  pinMode(buzzer_pin, OUTPUT);

  randomSeed(analogRead(3));

  pinMode(RELAY_PIN,OUTPUT);
  digitalWrite(RELAY_PIN, LOW);//UNLOCK

}

void loop() {
  keypad.addEventListener(keypadEvent);
  readKeypad();

  right = digitalRead(IN1);
  midright = digitalRead(IN2);
  mid = digitalRead(IN3);
  midleft = digitalRead(IN4);
  left = digitalRead(IN5);
/*
  Serial.print(right);
  Serial.print(" ");
  Serial.print(midright);
  Serial.print(" ");
  Serial.print(mid);
  Serial.print(" ");
  Serial.print(midleft);
  Serial.print(" ");
  Serial.println(left);
*/

  lcd.setCursor(0,0);
  lcd.print("Press '#' for OK");
  digitalWrite(buzzer_pin,LOW);


  if(turnright == 0 && turnleft == 0 && go_dest == 1){
```

```
digitalWrite(RELAY_PIN, HIGH);//UNLOCK

if(location == "A1"){
//dest A
dest_A1();
}
else if(location == "B1"){
//dest B1
dest_B1();
}
else if(location == "A2"){
//dest A2
dest_A2();
}
else if(location == "B2"){
//dest A2
dest_B2();
}

//ORDER COMPLETE
if (keypad_status == 3 && TAC_pass == TAC){
  //order_complete = 1;
  digitalWrite(RELAY_PIN, LOW);//UNLOCK
  lcd.setCursor(0,2);
  lcd.print("Done! Press #");


}
if (keypad_status == 3 && TAC_pass != TAC){
  TAC_pass = 0;
  TAC_String = "";
  lcd.setCursor(0,2);
  lcd.print("Try Again! Press * ");
}
if(keypad_status > 3 && TAC_pass == TAC ){
    digitalWrite(RELAY_PIN, HIGH);//LOCK
    order_complete = 1;
  }
  digitalWrite(buzzer_pin, LOW);
//follow_line();
//STOP LINE
  if(right == 1 && midright == 1 && mid == 1 && midleft == 1 && left == 1){
    digitalWrite(buzzer_pin,HIGH);
  //unsigned long currentMillis = millis();
  //if(currentMillis - previousMillis >= 500){
  //previousMillis = currentMillis;
  delay(100);
  motor1.setSpeed(0);
  motor2.setSpeed(0);
```

```
            Count += 1;
            //}

            }

          //STOP
            else if(right == 0 && midright == 0 && mid == 0 && midleft == 0 && left ==
0){
            digitalWrite(buzzer_pin,HIGH);
            motor1.setSpeed(0);
            motor2.setSpeed(0);

            }
            else if(right == 1 && midright == 0 && mid == 0 && midleft == 0 && left ==
0){

            motor1.setSpeed(-180);   // Motor 1 runs forward at full speed.
            motor2.setSpeed(180);  // Motor 2 runs backward at full speed.

            }
            else if(right == 0 && midright == 0 && mid == 0 && midleft == 0 && left ==
1){

            motor1.setSpeed(180);   // Motor 1 runs forward at full speed.
            motor2.setSpeed(-180);  // Motor 2 runs backward at full speed

            }
            else{
              PID();

            }


  }//turn right
  else{

  if(keypad_status == 2){
  lcd.setCursor(0,3);
  lcd.print(location);
  lcd.setCursor(3,3);
  lcd.print(noPhone);

  sendJSON();
  go_dest = 1;
  }

  motor1.setSpeed(0);
  motor2.setSpeed(0);
  }
```

```
  //TURN RIGHT
  turn_right();
  //TURN LEFT
  turn_left();

  turn_around();

}

void turn_around(){

  if(turnaround == 1){

  motor1.setSpeed(-180);   // Motor 1 runs forward at full speed.
  motor2.setSpeed(170);  // Motor 2 runs backward at full speed.
  if(right == 1 && midright == 0 && mid == 0 && midleft == 0 && left == 0){
    turnaround = 0 ;
    done += 1;
  }
  }
}

void turn_right(){

  if(turnright == 1){
  motor1.setSpeed(-155);   // Motor 1 runs forward at full speed.
  motor2.setSpeed(170);  // Motor 2 runs backward at full speed.
  if(right == 1 && midright == 0 && mid == 0 && midleft == 0 && left == 0){
    turnright = 0 ;
    done += 1;
  }
  }
}

void turn_left(){

  if(turnleft == 1){

  motor1.setSpeed(170);   // Motor 1 runs forward at full speed.
  motor2.setSpeed(-145);  // Motor 2 runs backward at full speed.

  if(right == 0 && midright == 0 && mid == 0 && midleft == 0 && left == 1){
    turnleft = 0 ;
    done += 1;
  }
  }

}
```

```
///////////////////////////////////////////////////PID/////////////////////////////////////////////////////////

void forward_brake(int posa, int posb) {
  //set the appropriate values for aphase and bphase so that the robot goes straight
  motor1.setSpeed(posb);   // Motor 1 runs forward at full speed.
  motor2.setSpeed(posa);  // Motor 2 runs backward at full speed.


}

void PID(){

  uint16_t position = analogRead(AN);

//set point = 500

  int error = 500 - position;

  P = error;
  I = I + error;
  D = error - lastError;
  lastError = error;

  int motorspeed = P*Kp + I*Ki + D*Kd;

  //Serial.println(motorspeed);
  int motorspeeda = basespeeda + motorspeed;
  int motorspeedb = basespeedb - motorspeed;

  if (motorspeeda > maxspeeda) {
    motorspeeda = maxspeeda;
  }
  if (motorspeedb > maxspeedb) {
    motorspeedb = maxspeedb;
  }
  if (motorspeeda < 0) {
    motorspeeda = 0;
  }
  if (motorspeedb < 0) {
    motorspeedb = 0;
  }


  forward_brake(motorspeeda, motorspeedb);

}
////////////////////////////////////////FOLLOW LINE////////////////////////////////////
/*
void follow_line(){
```

```
  if(right == 0 && midright == 0 && mid == 0 && midleft == 0 && left == 0){
    STOP();
  }
  //STRAIGHT
  else if(right == 0 && midright == 0 && mid == 1 && midleft == 0 && left == 0){
    STRAIGHT();
  }
  else if(right == 0 && midright == 1 && mid == 1 && midleft == 1 && left == 0){
    STRAIGHT();
  }
  else if(right == 0 && midright == 1 && mid == 1 && midleft == 0 && left == 0){
    STRAIGHT();
  }
    else if(right == 0 && midright == 0 && mid == 1 && midleft == 1 && left == 0){
    STRAIGHT();
  }

  //SLIGHT LEFT
  else if(right == 0 && midright == 1 && mid == 0 && midleft == 0 && left == 0){
    SLGHT_LEFT();
  }
  //SLIGHT RIGHT

  else if(right == 0 && midright == 0 && mid == 0 && midleft == 1 && left == 0){
    SLGHT_RIGHT();
  }
  //LEFT
  else if(right == 1 && midright == 1 && mid == 0 && midleft == 0 && left == 0){
    LEFT();
  }
  else if(right == 1 && midright == 0 && mid == 0 && midleft == 0 && left == 0){
    LEFT();
  }

  //RIGHT
  else if(right == 0 && midright == 0 && mid == 0 && midleft == 0 && left == 1){
    RIGHT();
  }
  else if(right == 0 && midright == 0 && mid == 0 && midleft == 1 && left == 1){
    RIGHT();
  }

}

void STOP(){
  digitalWrite(IN_1, LOW);
  digitalWrite(IN_2, LOW);
  digitalWrite(IN_3, LOW);
  digitalWrite(IN_4, LOW);
  analogWrite(ENA, 0);
```

```
  analogWrite(ENB, 0);
}
void STRAIGHT(){
  digitalWrite(IN_1, HIGH);
  digitalWrite(IN_2, LOW);
  digitalWrite(IN_3, HIGH);
  digitalWrite(IN_4, LOW);
  analogWrite(ENA, 120);
  analogWrite(ENB, 120);
}
void SLGHT_LEFT(){
  digitalWrite(IN_1, HIGH);
  digitalWrite(IN_2, LOW);
  digitalWrite(IN_3, LOW);
  digitalWrite(IN_4, HIGH);
  analogWrite(ENA, 130);
  analogWrite(ENB,100);
}
void SLGHT_RIGHT(){
  digitalWrite(IN_1, LOW);
  digitalWrite(IN_2, HIGH);
  digitalWrite(IN_3, HIGH);
  digitalWrite(IN_4, LOW);
  analogWrite(ENA, 100);
  analogWrite(ENB,130);
}
void LEFT(){
  digitalWrite(IN_1, HIGH);
  digitalWrite(IN_2, LOW);
  digitalWrite(IN_3, LOW);
  digitalWrite(IN_4, HIGH);
  analogWrite(ENA, 160);
  analogWrite(ENB, 130);
}
void RIGHT(){
  digitalWrite(IN_1, LOW);
  digitalWrite(IN_2, HIGH);
  digitalWrite(IN_3, HIGH);
  digitalWrite(IN_4, LOW);
  analogWrite(ENA, 130);
  analogWrite(ENB,160);
}
*/
////////////////////////////////////////////////////////////DESTINATION///////////////////////////////////////////////////
/////////
//DESTINATION A1
void dest_A1(){

  digitalWrite(RELAY_PIN, HIGH);//LOCK
  if(Count == 0){
```

```
    if(done == 0){
    turnaround = 1;
    lcd.clear();
    }
  }
  else if(Count == 3){

    if(done == 1){
    turnright = 1;
    }
  }
  else if(Count == 4){

    if(done == 2){
    if(order_complete == 1){

    turnaround = 1;
    }

    }
  }
  else if(Count == 5){
    if(done == 3){
      turnleft = 1;
    }
  }
  else if(Count == 8){
    lcd.clear();
    Count = 0;
    location = "";
    noPhone = "";
    done = 0;
    go_dest = 0;
    order_complete = 0;
    keypad_status= 0 ;
    TAC_String = "";
    TAC_pass = 0;
    TAC = 0;
    done_tac = 0;
    digitalWrite(RELAY_PIN,LOW);//UNLOCK
  }

}

//DESTINATION B1
void dest_B1(){

  digitalWrite(RELAY_PIN, HIGH);//LOCK
  if(Count == 0){
    if(done == 0){
```

```
    turnaround = 1;
    lcd.clear();
    }
  }
  else if(Count == 3){

    if(done == 1){
    turnleft = 1;
    }
  }
  else if(Count == 4){

    if(done == 2){
    if(order_complete == 1){

    turnaround = 1;
    }

    }
  }
  else if(Count == 5){
    if(done == 3){
      turnright = 1;
    }
  }
  else if(Count == 8){
    lcd.clear();
    Count = 0;
    location = "";
    noPhone = "";
    done = 0;
    go_dest = 0;
    order_complete = 0;
    keypad_status= 0 ;
    TAC_String = "";
    TAC_pass = 0;
    TAC = 0;
    done_tac = 0;
    digitalWrite(RELAY_PIN,LOW);//UNLOCK
  }

}

//DESTINATION A2
void dest_A2(){

  digitalWrite(RELAY_PIN, HIGH);//LOCK
  if(Count == 0){
    if(done == 0){
    turnaround = 1;
```

```
  lcd.clear();
  }
 }
 else if(Count == 4){

  if(done == 1){
  turnright = 1;
  }
 }
 else if(Count == 5){

  if(done == 2){
  if(order_complete == 1){

  turnaround = 1;
  }

  }
 }
 else if(Count == 6){
  if(done == 3){
   turnleft = 1;
  }
 }
 else if(Count == 10){
  lcd.clear();
  Count = 0;
  location = "";
  noPhone = "";
  done = 0;
  go_dest = 0;
  order_complete = 0;
  keypad_status= 0 ;
  TAC_String = "";
  TAC_pass = 0;
  TAC = 0;
  done_tac = 0;
  digitalWrite(RELAY_PIN,LOW);//UNLOCK
 }

}

//DESTINATION B2
void dest_B2(){

 digitalWrite(RELAY_PIN, HIGH);//LOCK
 if(Count == 0){
  if(done == 0){
  turnaround = 1;
  lcd.clear();
```

```
    }
  }
  else if(Count == 4){

    if(done == 1){
    turnleft = 1;
    }
  }
  else if(Count == 5){

    if(done == 2){
    if(order_complete == 1){

    turnaround = 1;
    }

    }
  }
  else if(Count == 6){
    if(done == 3){
      turnright = 1;
    }
  }
  else if(Count == 10){
    lcd.clear();
    Count = 0;
    location = "";
    noPhone = "";
    done = 0;
    go_dest = 0;
    order_complete = 0;
    keypad_status= 0 ;
    TAC_String = "";
    TAC_pass = 0;
    TAC = 0;
    done_tac = 0;
    digitalWrite(RELAY_PIN,LOW);//UNLOCK
  }

}

/////////////////////////////////////////////////////KEYPAD//////////////////////////////////////////////////

void readKeypad(){
char key = keypad.getKey();



if(keypad_status == 0 )
```

```
{

 lcd.setCursor(0,1);
 lcd.print("Destination:");

 lcd.setCursor(13,1);
  switch(key){
    case NO_KEY:
    break;

    case '0':
    location = location + "0";
    lcd.print(location);
    break;

    case '1':
    location = location + "1";
    lcd.print(location);
    break;

    case '2':
    location = location + "2";
    lcd.print(location);
    break;

    case '3':
    location = location + "3";
    lcd.print(location);
    break;

    case '4':
    location = location + "4";
    lcd.print(location);
    break;

    case '5':
    location = location + "5";
    lcd.print(location);
    break;

    case '6':
    location = location + "6";
    lcd.print(location);
    break;

    case '7':
    location = location + "7";
    lcd.print(location);
    break;
```

```
        case '8':
        location = location + "8";
        lcd.print(location);
        break;

        case '9':
        location = location + "9";
        lcd.print(location);
        break;

        case 'A':
        location = location + "A";
        lcd.print(location);
        break;

        case 'B':
        location = location + "B";
        lcd.print(location);
        break;

        case 'C':
        location = location + "C";
        lcd.print(location);
        break;

        case 'D':
        location = location + "D";
        lcd.print(location);
        break;

        case '#':
          break;

        case '*':
          location = "";
          noPhone = "";
          lcd.clear();
      }
  }

  if(keypad_status == 1 ){

      lcd.setCursor(0,2);
      lcd.print("No.Phone:");

      lcd.setCursor(9,2);
       switch(key){
        case NO_KEY:
        break;
```

```
case '0':
noPhone = noPhone + "0";
lcd.print(noPhone);
break;

case '1':
noPhone = noPhone + "1";
lcd.print(noPhone);
break;

case '2':
noPhone = noPhone + "2";
lcd.print(noPhone);
break;

case '3':
noPhone = noPhone + "3";
lcd.print(noPhone);
break;

case '4':
noPhone = noPhone + "4";
lcd.print(noPhone);
break;

case '5':
noPhone = noPhone + "5";
lcd.print(noPhone);
break;

case '6':
noPhone = noPhone + "6";
lcd.print(noPhone);
break;

case '7':
noPhone = noPhone + "7";
lcd.print(noPhone);
break;

case '8':
noPhone = noPhone + "8";
lcd.print(noPhone);
break;

case '9':
noPhone = noPhone + "9";
lcd.print(noPhone);
break;
```

```
      case 'A':
      noPhone = noPhone + "A";
      lcd.print(noPhone);
      break;

      case 'B':
      noPhone = noPhone + "B";
      lcd.print(noPhone);
      break;

      case 'C':
      noPhone = noPhone + "C";
      lcd.print(noPhone);
      break;

      case 'D':
      noPhone = noPhone + "D";
      lcd.print(noPhone);
      break;

      case '#':
        break;

      case '*':
        noPhone = "";
        location = "";
        keypad_status = 0;
        lcd.clear();
    }
  }
  if(keypad_status >= 2 ){

    lcd.setCursor(0,1);
    lcd.print("TAC : ");
    lcd.setCursor(6,1);

    switch(key){
      case NO_KEY:
      break;

      case '0':
      TAC_String = TAC_String + "0";
      lcd.print(TAC_String);
      break;

      case '1':
      TAC_String = TAC_String + "1";
      lcd.print(TAC_String);
      break;
```

```
case '2':
TAC_String = TAC_String + "2";
lcd.print(TAC_String);
break;

case '3':
TAC_String = TAC_String + "3";
lcd.print(TAC_String);
break;

case '4':
TAC_String = TAC_String + "4";
lcd.print(TAC_String);
break;

case '5':
TAC_String = TAC_String + "5";
lcd.print(TAC_String);
break;

case '6':
TAC_String = TAC_String + "6";
lcd.print(TAC_String);
break;

case '7':
TAC_String = TAC_String + "7";
lcd.print(TAC_String);
break;

case '8':
TAC_String = TAC_String + "8";
lcd.print(TAC_String);
break;

case '9':
TAC_String = TAC_String + "9";
lcd.print(TAC_String);
break;

case 'A':
TAC_String = TAC_String + "A";
lcd.print(TAC_String);
break;

case 'B':
TAC_String = TAC_String + "B";
lcd.print(TAC_String);
break;
```

```
      case 'C':
      TAC_String = TAC_String + "C";
      lcd.print(TAC_String);
      break;

      case 'D':
      TAC_String = TAC_String + "D";
      lcd.print(TAC_String);
      break;

      case '#':
      TAC_pass =  TAC_String.toInt();
        break;

      case '*':
        TAC_String = "";
        keypad_status = 2;
        lcd.clear();
    }


}
}

void keypadEvent(KeypadEvent key){
    switch (keypad.getState()){
    case PRESSED:
        if (key == '#') {
        keypad_status += 1;
        }
        break;


}
}


//////////////////////////JSON SEND ///////////////////////////////////////////////

void sendJSON(){
  String new_noPhone = "+6" + noPhone;


  if(done_tac == 0){
  TAC = random(1000,9999);
  done_tac = 1;
  }


    StaticJsonDocument<200> doc;
```

```
    doc["TAC"] = TAC;
    doc["noPhone"] = new_noPhone ;
    doc["location"] = location;

   Serial.println(new_noPhone);
   Serial.println(TAC);
   Serial.println(location);
  serializeJson(doc, Serial3);
}
```