

SMART UNDERGROUND CYLINDRICAL PARKING SYSTEM

By

Yong Tze Liang

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

**BACHELOR OF INFORMATION TECHNOLOGY (HONOURS) COMPUTER
ENGINEERING**

Faculty of Information and Communication Technology
(Kampar Campus)

JAN 2022

REPORT STATUS DECLARATION FORM

Title: Smart Underground Cylindrical Parking System

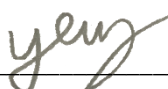
Academic Session: 2022

I YONG TZE LIANG
(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,


(Author's signature)

TEOH
(Supervisor's signature)

Address:
191, Jalan Dahlia,
Taman Dahlia, 09000,
Kulim, Kedah

Teoh Shen Khang
Supervisor's name

Date: 18/4/2022

Date: 21 April 2022

FACULTY/INSTITUTE* OF INFORMATION AND COMMUNICATION TECHNOLOGY

UNIVERSITI TUNKU ABDUL RAHMAN

Date: 18/04/2022

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that Yong Tze Liang (ID No: 18ACB02524) has completed this final year project/ dissertation/ thesis* entitled “Smart Underground Cylindrical Parking System” under the supervision of Mr Teoh Shen Khang (Supervisor) from the Department of Computer and Communication Technology, Faculty/Institute* of Information and Communication Technology, and _____ (Co-Supervisor)* from the Department of _____, Faculty/Institute* of _____.

I understand that University will upload softcopy of my final year project / dissertation/ thesis* in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.


Yours truly,



(YONG TZE LIANG)

DECLARATION OF ORIGINALITY

I declare that this report entitled “**Smart Underground Cylindrical Parking System**” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : 

Name : Yong Tze Liang

Date : 18/4/2022

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisor, Teoh Shen Khang who has given me this opportunity to come up with my own project and helping me throughout the questions I had. It is my first step into project design, robotics and also gaining more familiarity using the Raspberry pi. A million thanks to you.

I must also thank my parents for their support, and continuous encouragement throughout the course, and my friends who helped me through every step of the way.

ABSTRACT

This project is a robotic and IoT project made to help solve some problems related to parking. It will be an alternative choice for parking structures which can bring more benefits. As normal parking not only wastes space but also causes traffic congestion and is non-environmentally friendly. The article goes through more of the disadvantages of parking lots, and also going through some of the inventions that have been made. This project aims to use automation to solve these problems, at the same time bringing automation to the mainstream. We will be making a small-scaled prototype for demonstration purposes. From the design, RPi is chosen to be the brain of the system as it has several GPIO pins for the signals to control the components, it also is able to connect to a screen through HDMI port and is able to send signals to other systems. The motors used are stepper motors as it has high precision, which the system needs in order to not damage anything. The system will be using Geany as the programming software, and Python will be used as the programming language. The system will be able to retrieve data from the file, get input from user, compare data, save data, and control the other components. By receiving input from the user on whether to store or to retrieve, the system will do the respective processes before passing the data to the motor control process, which will calculate the angle and level for the motors in order to get to the specified slot. The entire structure will be made out of wood, use around 5 motors and RPi to control it. The process will be a loop, waiting for users to use the system while checking and updating the status when the system is used. The challenges face by this project will be the cost, tm and work that will have to be put in to actually implement in the real world. But in the end, there are still many improvements that can still be made to the system, which already makes it stand out to those systems that cannot.

TABLE OF CONTENTS

TITLE PAGE	i
REPORT STATUS DECLARATION FORM	ii
SUBMISSION OF FINAL YEAR PROJECT FORM	iii
DECLARATION OF ORIGINALITY	iv
ACKNOWLEDGEMENTS	v
ABSTRACT	vi
TABLE OF CONTENTS	vii
LIST OF FIGURES	x
LIST OF TABLES	xii
LIST OF SYMBOLS	xiii
CHAPTER 1 INTRODUCTION	1
1.1 Problem Statement and Motivation	1
1.2 Project Scope and Direction	3
1.3 Project Objectives	3
1.4 Contributions	4
1.5 Background Information	4
1.6 Project Timeline	6
1.7 Report Organization	7
CHAPTER 2 LITERATURE REVIEW	8
2.1 Car Parking Problem in Urban Areas, Causes and Solutions	8
2.2 Previous Inventions	8
2.2.1 Parkmatic Automated Carousel	9
2.2.2 Volkswagen’s Car Towers at Autostadt in Wolfsburg, Germany	11
2.2.3 IoT based Smart Parking System	13
2.2.4 Automated Parking Facility ECO Cycle	15
2.3 Summary	16

CHAPTER 3 SYSTEM METHODOLOGY/APPROACH	18
3.1 System Design Diagram	18
3.1.1 System Architecture Diagram	20
3.1.2 Use Case Diagram and Description	20
3.1.3 Activity Diagram	24
CHAPTER 4 SYSTEM DESIGN	26
4.1 System Overview	26
4.2 General Work Procedures	29
4.3 Tools	31
4.3.1 Hardware in Use	31
4.3.2 Software in Use	32
4.3.3 Programming Language in Use	32
4.4 System and Tools Design	34
4.5 Flowchart	37
CHAPTER 5 SYSTEM IMPLEMENTATION	40
5.1 Hardware Setup	40
5.1.1 Raspberry Pi 3B+ to DRV8825	40
5.1.2 Stepper Motor/Lead Screw	41
5.1.3 Raspberry Pi Zero W to L298N to Actuator	42
5.2 Software Setup	44
5.2.1 Raspberry Pi 3B+	45
5.2.1.1 File Handling and LED	45
5.2.1.2 Hardware Declaration (DRV8825)	46
5.2.1.3 Movement	47
5.2.1.4 MQTT	48
5.2.1.5 GUI	49
5.2.2 Raspberry Pi Zero W	50
5.2.2.1 Hardware Declaration (L298N)	50
5.2.2.2 MQTT and Movement	54
5.3 Final Assembly	52

5.4	Implementation Issues and Challenges	55
CHAPTER 6 SYSTEM EVALUATION AND DISCUSSION		56
6.1	System Testing and Performance Metrics	56
6.2	Test Setup and Results	61
6.3	Objective Evaluation	68
CHAPTER 7 CONCLUSION		69
7.1	Conclusion	69
7.2	Recommendation	70
REFERENCE		71
APPENDIX		
	WEEKLY LOG	A1-A7
	POSTER	P1-P2
	PLAGIARISM CHECK RESULT	T1-T4
	FYP2 CHECKLIST	C1

LIST OF FIGURES

Figure Number	Title	Page
Figure 1.1a	Full Parking.	2
Figure 1.1b	Empty Parking.	2
Figure 1.6a	Project Timeline (FYP1)	6
Figure 1.6b	Project Timeline (FYP2)	6
Figure 2.2.1	Car Carousel.	10
Figure 2.2.2	Car Tower.	12
Figure 2.2.3	Car Parking Sensor.	14
Figure 2.2.4	ECO Cycle.	16
Figure 2.3	Concept of system used in public.	17
Figure 3.1.1	System Architecture Diagram	20
Figure 3.1.2	Use Case Diagram	20
Figure 3.1.3.1	Store and Retrieve Activity Diagram	24
Figure 3.1.3.2	Motion Activity Diagram	25
Figure 4.1.1	Rough Sketch of System Proposed.	27
Figure 4.1.2	Concept of moving platform using lead screw and motor.	27
Figure 4.2.1	Previous Hardware Implementation of System from FYP1.	28
Figure 4.2.2	New Hardware Implementation of System	30
Figure 4.4.1	Sample Connection	34
Figure 4.4.2	Stepping Mode of DRV8825	34
Figure 4.4.3	Sample Connection (RPi Zero W)	36
Figure 4.5.1	Flowchart	37
Figure 4.5.2	Movement Function Flowchart	38
Figure 4.5.3	Visualisation of Actuator and Platform	39
Figure 5.1.1a	Wiring of the RPi 3B+	40
Figure 5.1.1b	Wiring of the RPi 3B+	41
Figure 5.1.2	Motors connected to lead screw with coupling shaft	42

Figure 5.1.3a	Wiring of RPi Zero W	43
Figure 5.1.3b	Wiring of RPi Zero W	43
Figure 5.1.3c	Wiring of RPi Zero W	44
Figure 5.2.1.1	Code Snippet(File Handling and LED)	45
Figure 5.2.1.2	Code Snippet(Hardware Declaration)	46
Figure 5.2.1.3	Code Snippet(Movement)	47
Figure 5.2.1.4	Code Snippet(RPi 3B+ MQTT)	48
Figure 5.2.1.5a	Code Snippet(GUI)	49
Figure 5.2.1.5b	Code Snippet(GUI)	49
Figure 5.2.2.1	Code Snippet(Hardware Declaration)	50
Figure 5.2.2.2	Code Snippet(RPi Zero W MQTT and Movement)	51
Figure 5.3.1	Final Prototype	52
Figure 5.3.2	Base of Final Prototype	53
Figure 5.3.3	Top and Moving Platform of Final Prototype	54

LIST OF TABLES

Table Number	Title	Page
Table 2.2.1	Strength and Weaknesses of Car Carousel	11
Table 2.2.2	Strength and Weaknesses of Car Tower	13
Table 2.2.3	Strength and Weaknesses of IoT based Smart Parking System	15
Table 2.2.4	Strength and Weaknesses of ECO Cycle	16
Table 3.1.2.1	Use Case Description (Store)	21
Table 3.1.2.2	Use Case Description (Retrieve)	22
Table 3.1.2.3	Use Case Description (Motion)	23
Table 6.1.1	Test Phase 1	56
Table 6.1.2	Test Phase 2	57
Table 6.1.3	Test Phase 3	58
Table 6.1.4	Test Phase 4	58
Table 6.1.5	Test Phase 5	59
Table 6.2.1	Test Setup 1	61
Table 6.2.2	Test Setup 2	62
Table 6.2.3	Test Setup 3	64

LIST OF ABBREVIATIONS

<i>RPi</i>	Raspberry Pi
<i>Vref</i>	Voltage Reference
<i>GUI</i>	Graphical User Interface
<i>GPIO</i>	General Purpose Input/Output

CHAPTER 1

Introduction

As automation has started to become more and more mainstream in this day and age, investors and inventors have been looking into ways to implement automation into our daily lives, from voice activated home assistance to self-driving cars, these inventions are revolutionizing our world. There are also inventions made to help us by solving our daily life problems. Since IoT and robotic systems are mainly used in automation, anyone can come up with an idea to solve various problems if they put in some time to research.

1.1 Problem Statement and Motivation

Less than 30% of the earth's surface is covered by land, which includes cities, forests, and mountains. With the world walking into a technologically advanced future, more and more space has been consumed for the sake of humanity's progression. As human population has been steadily increasing, it just made the issue with waste of space more severe. Research have shown that the average number of vehicles per household is around 1.88, and that around 1.4 billion cars are active as of 2020. Which has caused the problem we will be discussing today, problems risen due to parking. When driving around, there are usually 2 types of situations that we can notice. Either we see a large amount of space being used for parking slots but is almost never full, or we see too less parking slots, leading to tons of problems such as double parking or causing disruption in traffic flow. Besides, no one can deny that when a parking space is not occupied, it is essentially wasting space. Whereas when the parking slot is full, cars would be going around the parking space looking for an empty slot, polluting the air and sometimes causing traffic congestions. Even so, this is just a part of the disadvantages of parking, more will be discussed later.



Figure 1.1a Full Parking

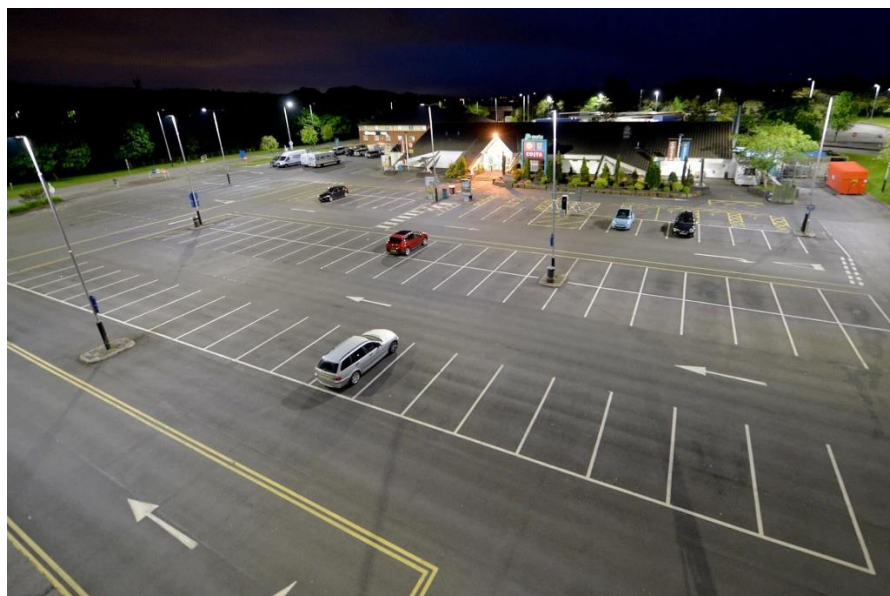


Figure 1.1b Empty Parking

To simplify, parking lots has been the cause of space wasted for quite some time. Although there are parking structures, building basement(underground type) or podium(above ground type) or even individual structure(multi-story garage type), can solve some of these problems, it is only suitable for use by huge companies and malls, and not for other smaller companies or public places. Therefore, on-street parking and off-street parking are usually opted for in these situations. Therefore, we are looking to resolve issues for those that are opting for off-street parking (Supermarket parking slots) or on-street parking (Public parking slots) that may cause other issues such as traffic congestions.

1.2 Project Scope and Direction

The project provides an alternative parking system that would be more beneficial to the environment and space saving. The user will be able to use the system easily and be assured of the safety the system provides. To use the system the user will park their car on the designated spot and go to the interface to set a password. The slot number and password will be saved into the system and the mechanical components will then start and move the parked car to its designated spot underground. The entire process will be automated, reducing car emission and congestion, also giving more security to the vehicles as they are stored underground. To retrieve the vehicle, users will just go to the interface and key in the slot number and password, the system will then go and obtain the vehicle from the assigned spot if the password matches to the one stored in the system.

The direction of this project is to become an alternative parking system to those that already exist. The system will also bring automation into the daily lives of people around the world while at the same time providing advantages from the existing systems.

1.3 Project Objectives

The project aims to design an embedded software for controlling all the mechanical components. The entire system will be controlled by a Raspberry Pi 3 B+, the motors will be connected to a driver which will be connected to the RPi. The RPi will also handle the file handling and interface.

Besides, this project aims to design a graphic user interface for human interaction. As the project will be opened to use for everyone, the interface should be made to be visually pleasing and simple to understand, no matter the user wishes to store or retrieve their vehicle.

This project also aims to implement a small-scale model for verification purpose. Since the project is meant to be a parking system, a small model will be made for demonstration purposes.

For this phase, the project will aim to design a simple user interface and embedded system to control the rotation and y-axis movement. The project will also start working on the base of the small-scale model which includes the motors controlling the rotation and y-axis movement system.

1.4 Contributions

The contribution of this project is to solve the problems brought by parking spaces. If the system were to be implemented in the future, it would be able to help people around the world solve problems as it would reduce pollution, traffic congestion and space wastage. By reducing traffic congestions, roads can become safer with less road rages and accidents happening overall. Whereas with the space saved, we can have better utilization of those parts of land such as plantations, which would further help with the pollution.

On top of that, with a user-friendly interface, anyone will be able to use the system. This way it will eliminate the idea of only the rich can use automation. Automation should be available for everyone, and by solving these problems for everyone, it would help inspire more inventions to help solve other problems. By having this system, everyone will be able to experience the benefits of an automated parking system.

1.5 Background information

As mentioned before, with automation on the rise, many sectors have started to implement automation. Other than reducing workloads for users, automation can also improve our daily lives in different ways. Automation is also much more secure and can help reduce waste of space. GIKEN, a Japanese company, has created an automated bicycle parking facility called Eco Cycle. It required minimal space aboveground as the whole facility was underground, and also eliminated risk of theft and exposure to weather. It was able to save such a huge amount of space that would have been used for bicycle parking slots, leading to better utilization of land aboveground. The invention also helped Japan promote use of the bicycle and prevent disorderly parking. From this invention, we were able to see the advantages it has brought to the “parking” section.

As we know, human population has been steadily increasing, and based on research, the average numbers of vehicles per household is around 1.88. It may not seem like much but there are about 7.9 billion people on this world as of the time of writing this. Therefore, no one can deny that parking has also become a problem because of this. When driving around, there are usually 2 types of situations that we can notice. Either we see a large amount of space being used for parking slots but is almost never full, or we see too less parking slots, leading to tons of problems such as double parking or

CHAPTER 1

causing disruption in traffic flow. Other than that, a huge plot of land has been made into a parking lot that might never be full, essentially wasting space that could have been used for other purposes, such as planting trees. As of the problem a lot of cities face, lack of greenery.

With an underground parking system, it would certainly provide more space aboveground, as well as let us utilize the most out of an area. Being automated also mean that the users, drivers in this case, would not need to physically park their cars and worry about the problem of double parking. Similar to the perks provided by Eco Cycle, one's vehicle would also be safe from theft and bad weather, other than providing a solution to irresponsible parking. This project would not try to replace the existing parking structures such as those in huge buildings or malls, but to provide a solution to places that have to work with a smaller area such as parking slots of public environments and supermarkets.

Therefore, to make an automated system, we will always work with robotics and IoT systems. To further make it clear, robotics is a system that is using machines to perform certain tasks that is traditionally done by human beings. It involves design, construction, operation, and the use of mechanical parts to do certain actions. Some of these systems are automated while some require human input to operate. As for IoT, it describes the network of things that are embedded with sensors, software and other components. These components are used for the purpose of connecting and exchanging data with systems over the internet.

As the main brain of the system is a RPi, we will introduce the microprocessor. The Raspberry Pi is a microprocessor that is used by a lot of inventors that requires a processor for their project. It can act as a computer, with coding apps preinstalled on it. By programming the intended program onto the RPi, the system will run the program, receiving input from users or sensors, and outputting the signals based on the programme on it. One can also use it as a basic computer as it is able to do what a normal computer can do. There are several versions of RPi, with different processing power and size, users are able to use choose the most suitable microprocessor for their system. For example, using model Pi zero when there's no need of huge processing power and a smaller size is preferred, while using Model 3 B+ as it has larger processing power and can handle more processes.

CHAPTER 1

For the motors used, stepper motor will be preferred. This is because in our system we will need a motor with high precision, which can be achieved by using the stepper motor. Stepper motors have multiple “toothed” electromagnets arranged around the central rotor. The central rotors also are “toothed”, this way there’s more control over the rotor. There are two sets of wires, when trying to make the shaft turn, one electromagnet is powered, which will magnetically attract the gear’s teeth on the rotor. When the teeth are aligned with the first electromagnet, they are slightly offset from the next electromagnet. Then the next electromagnet is powered while the first is not, with the constant power alteration between the two electromagnets, the rotor will slowly rotate teeth by teeth. This is why it is able to achieve precise angles, which leads it to become the usual motor used in a 3D printer.

1.6 Project Timeline

TASK	Project Week													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Revised Proposal	█													
Define Project Objective and Scope	█													
Collection of Data		█												
Analysis for literature review		█												
Determine system design			█											
Define technologies involved			█											
Planning system architecture			█											
Testing the Instruments				█										
Building the base				█	█	█	█	█	█	█	█	█	█	█
Hardware				█	█	█	█	█	█	█	█	█	█	█
Software					█	█	█	█	█	█	█	█	█	█
Refining the prototype								█	█	█	█	█	█	█
Documentation										█	█	█	█	█
Report with supervisor current progress			█					█					█	
Presentation of FYP 1														█

Figure 1.6a Project Timeline (FYP1)

TASK	Project Week													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Testing the instruments	█													
Build moving platform		█	█	█	█	█	█	█	█	█	█	█	█	█
Hardware		█	█	█	█	█	█	█	█	█	█	█	█	█
Software				█	█	█	█	█	█	█	█	█	█	█
Combining base and moving platform							█	█	█	█	█	█	█	█
Hardware							█	█	█	█	█	█	█	█
Software							█	█	█	█	█	█	█	█
Testing the system				█	█	█	█	█	█	█	█	█	█	█
Building the outer structure				█	█	█	█	█	█	█	█	█	█	█
Fixing error of system (if faced)				█	█	█	█	█	█	█	█	█	█	█
Refining the prototype									█	█	█	█	█	█
Documentation										█	█	█	█	█
Report with supervisor current progress			█					█					█	
Presentation of FYP 2														█

Figure 1.6b Project Timeline (FYP2)

1.7 Report Organization

This report is organized into 7 Chapters: Chapter 1 Introduction, Chapter 2 Literature Review, Chapter 3 Proposed Methodology/Approach, Chapter 4 System Design, Chapter 5 System Implementation, Chapter 6 System Evaluation and Discussion, Chapter 7 Conclusion.

Chapter 1 includes a brief introduction, problem statement and motivation, project scope and direction, project objectives, contribution, background information, project timeline and report organization. Chapter 2 is the literature review carried out on articles on car parking and several existing Car Parking systems using IoT. A summary of the literature review is also included. Next, chapter 3 will explain the methodology of the system and show the system architecture diagram, use case diagram and description, and activity diagram. Chapter 4 will be the System Design, which will include system overview, general work procedures, tools (hardware, software, programming language...), circuit and tools design, and flowchart. Chapter 5 will include the how the hardware is setup, how the software is setup, final assembly and the issues and challenges faced during implementation. Chapter 6 will include the system testing and performance metrics, testing setup and result, and objective evaluation. As for Chapter 7, it will contain the conclusion and recommendation.

CHAPTER 2

Literature Reviews

2.1 Car Parking Problem in Urban Areas, Causes and Solutions

As of reviewing this paper, it further fortifies that car parking has caused a lot of problems, especially in urban areas. As stated by Hossam in the research, imbalance between parking supply and demand can be taken as the initial reason for the parking problems. He stated that people prefer to own cars as cars offer an unmatched combination of speed, autonomy, and privacy[1], which is true. Every household would prefer to travel in a car than on public transport, as they would be able to have a more private space for themselves, while the later would have to share the space with other strangers. It is also stated that parking problems are often overlooked in urban planning and transportation.

A few causes from the paper that could be used for this project would be, most of the old and historical cities have been planned with narrow streets, as at the time they used not cars, but horses. Therefore, when we wish to preserve the history of the buildings, these streets cannot be altered, causing them to exceed their planned capacities when all kind of vehicles move and park on these streets[1]. Other than that, it was stated that there is always a miscalculation of parking demand in new cities and new planned urban areas. Leading to either shortage or excessive parking spaces. It is also noted that there is a tendency of parking space provided are curb-parking and parking levels on street level, not parking structures with several floors to absorb the increasing number or cars. These causes can be used to relate to the causes of the problem we have right now, and it has also fortified the fact that it would be almost impossible to increase the street size to implement more parking spaces.

The research even has a few more problems caused by both having shortage or having excessive parking spaces. If there's a shortage of parking, drivers would usually circle around the vicinity looking for empty spaces, this phenomenon is called cruising. Cruising causes streets to become overcrowded as they have already arrived but are looking for spaces to park their car. It also consumes more fuel and emits extra pollution. On the other hand, having a large and excessive parking lots affect the

CHAPTER 2

environment indirectly. It was said that dark pavement of parking lots promotes water quality degradation, raises air temperature, and consumes land. Parking lots are considered to be the most environmentally harmful type of land use as well.

This paper has also given a clear concept of the forms of parking that has been provided, also showing innovative alternatives made by others to replace or disguise parking structures. The solution Hossam proposes is to view parking spaces as real estates and charge those who park there. The main objective was to sway people from driving as they would have to pay a fee just to park. By reducing the number of cars on the streets, it would indirectly decrease the parking problems.

The problem with the proposed solution would be that if it was actually decided that people would have to pay for parking, having a low fee would not be enough to sway people, but charging more would definitely cause an outrage by the people. Even if it does work, there would still be a huge area used for parking, and as stated before, parking lots are considered the least glamorous and most environmentally harmful type of land use.

2.2 Previous Inventions

2.2.1 Parkmatic Automated Carousel

Made by Parkmatic[2], the carousel was the solution to maximise the number of car parking in the least amount of space. It could hold up to either 14 SUV-sized or 16 Sedan-sized vehicles in the space of only 2 [3]. To simplify, it was a carousel for cars. Cars would enter from the floor level and the pallet will rotate the carousel, moving the vehicle up. When the owner wishes to get their car, they would just have to key in their parking space number and the carousel will move the vehicle down either clockwise or counterclockwise, depending on whichever is closer. The design also suggested the use of a turntable in front of the carousel, so that when the vehicle has successfully exited the carousel, the turntable could rotate the vehicle to the desired direction.



Figure 2.2.1 Car Carousel

The implementation of a carousel does help in maximising the number of car parking in the least amount of space. It also makes the vehicles safe from theft. But it does not actually protect the vehicles from the harsh weather, unless built indoors, which does contradict what we wanted, a public parking slot. The build is also aboveground, which can prove to be unstable at times. If there were harsh winds or minor tremors, the platforms on the carousel might sway and cause damage to the cars as well. Besides that, building aboveground would still use unnecessary space.

A way around the problems mentioned above would be to build it underground, with the entrance and exit ground level. It might seem like a small change, but the difference it makes is quite rewarding. For instance, the cars are now safe from harsh

CHAPTER 2

conditions, rain or sunlight won't be able to get through. If a minor tremor were to happen, being underground would be better as it provides more stability than aboveground. As taught to us, when earthquakes or tremors happen, get to the ground floor as soon as possible.

It should be noted though that this design does solve quite a bunch of the problems that were faced, such as reducing pollution by cars moving around looking for parking spaces. Other than that, since it is rectangular in shape, it would also be easier to implement into existing places. It also showed that it could mount a screen onto it to produce income by commercial as well. It certainly does prove to be a good solution to the problem faced but improvements can definitely still be made.

Strengths	Weaknesses
<ul style="list-style-type: none">• Easily constructed and deconstructed• Space saving• Easy to operate• Reduce pollution	<ul style="list-style-type: none">• It is built aboveground• Does not provide protection against harsh weathers

Table 2.2.1 Strengths and Weaknesses of Car Carousel

2.2.2 Volkswagen's Car Towers at Autostadt in Wolfsburg, Germany

There are 2 gleaming car parking towers of glass and galvanized steel which stores cars at Autostadt. Each of the towers are 60 meters tall and can house up to 400 cars[4]. These towers are connected to the Volkswagen factory by a 700-meter underground tunnel. The finished cars are transported via a conveyer belt to one of the two towers' basement to be moved into their position. There is a central beam where mechanical arms would move along and rotate to move the vehicles in and out of their bays.

When a customer makes a car purchase, the car is picked from the "silo" and transported out to the customer without having driven a single meter. The mechanical arms move with a speed of 2 meters per second. This was a major project, with over 400 architects working on designing the entire site and spending roughly 476 Million USD to build it.



Figure 2.2.2 Car Tower

Different from the carousel, this design was not made to solve any problems. It was made to be an attraction. Right now, it is used to store Volkswagen cars and has roughly 2 million visitors yearly. Though the design wasn't to solve parking problems, it has a similar design to how Eco Cycle works. The building is built aboveground, which as mentioned in the carousel, could be unstable but since it was built as a building in mind, the structures and stabilizers were made to withstand every harsh conditions. As it stands, it would still be better if it was underground, opening up the whole space aboveground to be used.

Looking at this design, the reason to holding back implementing it to solve our problems would be that it is a huge project. Building something this big will take a lot of time and preparation, and as the design shows, a huge chunk of space has been wasted in the middle. As each level can store up to more than 10 cars, arranging the cars in a circular manner would produce 2 types of circles, an outer circle, and an inner circle.

The space outside of the outer circle would be considered the free space, where it is not a part of the building, and can be used to build other stuffs. The space in between the outer circle and the inner circle would be used to house the cars, as intended. The

space in the inner circle however would only be used by the mechanical arm, which when taken into consideration, is wasting a whole lot of space. As the entire area will not be used to do anything or else it may disrupt the movement of the mechanical arms. This shows that if we design our project with cars per level in mind, we could end up wasting more space than saving them. If we were to design it by cars per level, another way should be thought of to minimize the waste of space.

Another downside to this design is that there are only two mechanical arms operating, with each responsible of half of the bays. With 400 bays available to hold cars, and only two mechanical arms, it will take a long time for it to actually fill up. Cars would have to line up to get placed into the slots, which would cause emission of pollution. It also makes it harder for those who are short on time. Therefore, from this design, a better one could be made to resolve the problems we face.

Strengths	Weaknesses
<ul style="list-style-type: none"> • Holds a lot of cars • Stable • Reduce pollution 	<ul style="list-style-type: none"> • Built as an attraction • Whole system too large • Hard to be implemented into the public • Only two mechanical arms operating on the giant system

Table 2.2.2 Strengths and Weaknesses of Car Tower

2.2.3 IoT based Smart Parking System

This paper shows a more widely used aspect with further improvements. The idea is to have sensors on each parking lot that would detect whether a vehicle was parked. The system consists of parking sensors, a processing unit, the Cloud, and a mobile application[5]. When a car is parked in the spot, the sensors would send signals to the processing unit, which in this case is a Raspberry pi. The processing unit acts as an intermediate between the sensors and the cloud. The data would be updated to the cloud which stores the records related to the parking area. Using the mobile application, one could immediately see where the available parking spaces are and would not waste time circling around the perimeter looking for parking spaces.

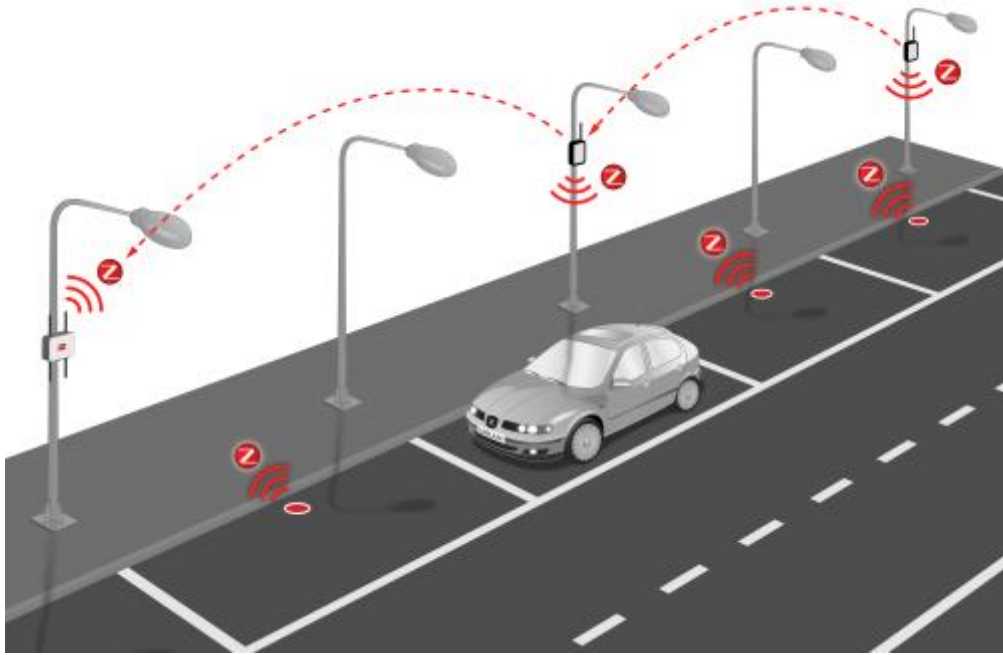


Figure 2.2.3 Car Parking Sensor

It was also designed that by using the app, one could book a parking slot in advance so that they will have a space to park when they arrive. It would then ask how long the car will be parked to make it available for others after. Payment is made after and when the car has been parked, the app would ask to confirm the car parking occupancy. It was also designed that if one does not confirm their occupancy, whether because they parked in the wrong space or the space they booked was parked by another car, an alarm would go off, notifying the authorities. If the driver overshoots its parking time, the app will also prompt a notification to the driver to either extend its parking time or leave the parking in a certain amount of time. If the driver fails to do so, notify the authorities.

Part of this system can be seen used in a lot of malls parking structures. Although it does not have an app associated, it does have panels that show which routes would lead to more available spaces. The idea was to reduce the emission of pollution in such a huge parking structure. But that is its own disadvantage, it was designed for a parking structure or a large off-road parking, not exactly preferable for public use. The only real advantage is that it can provide a more systematic approach to parking where drivers can book a parking space ahead of time. If it were to be implemented in the public space, every driver would have to download the app and use it, and those

who didn't download the app would be in a tough spot. Therefore, this solution to the parking problem isn't actually suitable, it just provides convenience in certain cases.

Strengths	Weaknesses
<ul style="list-style-type: none"> • Easy to implement and use • Reduce pollution 	<ul style="list-style-type: none"> • Waste space • Not exactly environmentally friendly

Table 2.2.3 Strengths and Weaknesses of IoT based Smart Parking System

2.2.4 Automated Parking Facility ECO Cycle

Now we will look at the invention that inspired this project the most, the ECO Cycle. It has a height of 12.4 meters and 8.36 meters in diameter. There are two versions of it, an underground version, and an aboveground version, both able to hold 204 bicycles each[6]. It would only take an average of 13 seconds for the owners to retrieve their bike. Users would have a card that is used to scan as an identification, When the card is scanned, the platform would move to the respective level and retrieve the bike. It should also be noted that there are specification limits to the bikes that can be parked in the system.

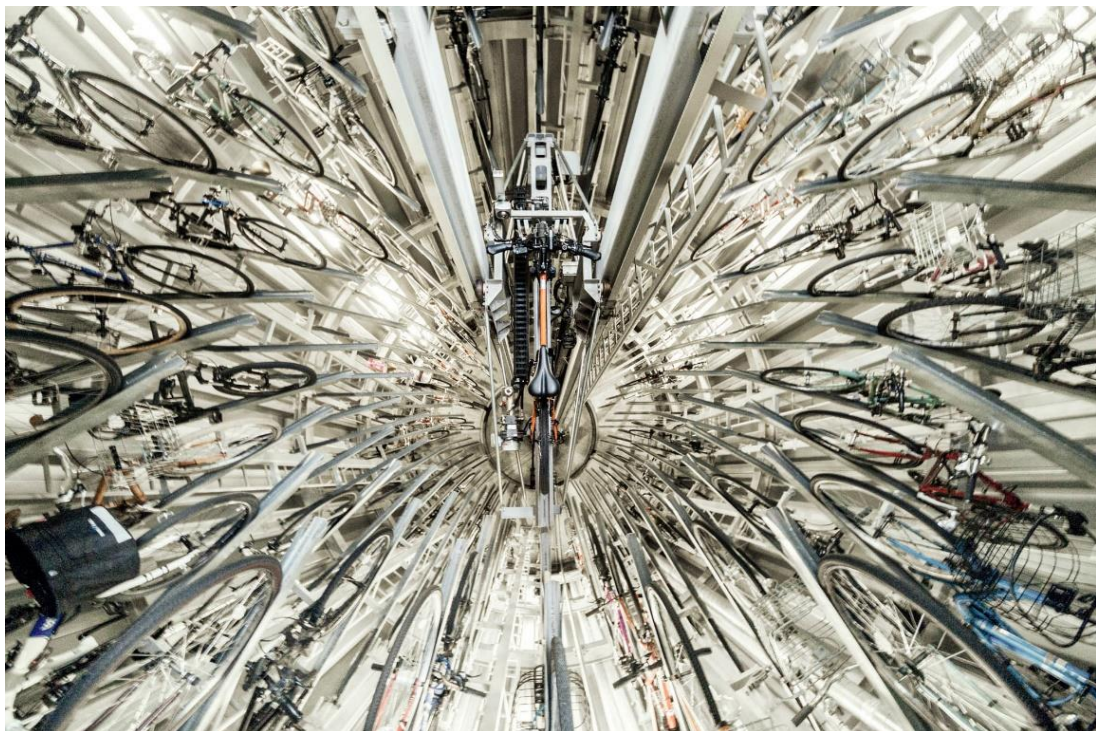


Figure 2.2.4 ECO Cycle

CHAPTER 2

The system designed can withstand earthquake seismic intensities of up to 6 as well as provide water resistance[7]. This invention has a huge impact on space saving as in reality it only needed a small space to build the entrance booth and it could hold up to 204 bikes underground, freeing up the space aboveground[8]. It also solved irresponsible bike parking in Japan. With its design, it could be constructed or deconstructed quickly for relocation.

Strengths	Weaknesses
<ul style="list-style-type: none">• Easily constructed and deconstructed• Space saving• Easy to operate• Reduce pollution	<ul style="list-style-type: none">• Not built with cars in mind

Table 2.2.4 Strengths and Weaknesses of ECO Cycle

2.3 Summary

After doing the literature review, we see that parking lots are not only wasting space, causing traffic congestion, it is also environmentally harmful. As stated by Hossam, parking lots would raise the land temperature because of the use of black pavements for the ground. Black absorbs the most heat in all the colours, therefore making the temperature higher on parking lots. Not to mention parking lots are designed with quantity in mind, there would be no trees or greenery to help cool down the area as they are seen as obstacles that would reduce the count of parking spaces. This being said, the way to solve this would be to move the parking underground, this way the space above could still be used to plant trees or be used for other causes, making the space more glamourize able.

The following problems can be solved together. As having either excessive or insufficient parking lots would lead to their own problems, we would also look into the lead causes of these problems. As both of these problems are happening to these specific types of parking, on-road parking, and off-road parking, we can determine that the problems would most likely be caused by inefficient space usage. Both of these types of parking are on ground level, and as we know if we can't expand horizontally, we should think of another way to expand, which in this case we could

expand vertically. Reducing the area used by expanding underground, we would be able to make use of the minimum amount of space and hold the maximum number of cars. This way even if we have excessive parking, it will not waste as much space, and we won't have the problem of having insufficient parking spaces, indirectly removing traffic congestion caused by it.

Building the structure underground would also solve the last problem. By making it automated, we can remove the threat of theft, at the same time protecting the cars from harsh weather. Other than that, being automated also means it would be less chances to cause an accident due to double parking or carelessness.

The drawing below shows a concept of the solution being implemented on on-road parking.

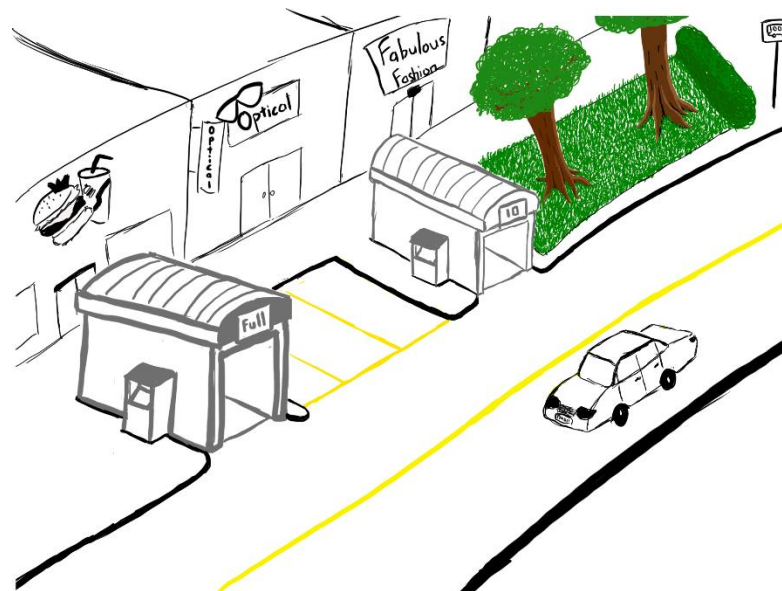


Figure 2.3 Concept of system used in public

As seen, the whole plot of land originally used for parking can be used to plant trees or bushes to have greenery even in town. There would be a few original on-road parking for those who will only park for a while and would not like to use the smart parking system.

CHAPTER 3

PROPOSED METHODOLOGY/APPROACH

As mentioned before, car parking has been a thorn that has been pricking at our world for a long time. This project aims to solve that, by designing embedded software that will control the mechanical components of an automated parking system. In this project we will build a small-scaled model of the system as a prototype.

The methodology used in this project will be the prototype methodology, since we are to develop a small-scaled prototype, test it and rework into a working model. Since prototyping methodology consists of 4 phases, with the first phase being planning phase.

In the planning phase, starts the discussion and planning of the project. Several proposals and ideas were shared and discussed, taking a lot of related topics into consideration. Which led to the title and the idea being finalized.

The following is the analysis phase, where the objective, problem statement, and literature review are discussed. In this phase we start to narrow down what we will do in the project by deciding on the problem statement and objective. After the two have been decided, literature review is carried out by looking at past inventions that try to solve the parking issue. The literature review is also done with some emphasis on comparing these past inventions, while slowly forming a suitable design in the head.

The next phase is the designing phase. In this phase the design of the prototype and system will be noted down. As a rough idea has been formed in the analysis phase, we use that idea and start to refine it. The most time will be spent in this phase as we are to design the system and decide on the components that will be used. So that we can do some research on the components to know how it functions and how to control them. A lot will be taken into consideration when designing the system. After finalizing the design, we will start to implement it.

CHAPTER 3

The last would be the implementation phase. For the implementation phase we will divide them into 2 sub-phases, which are Software phase and Hardware phase. We separate them so that when a problem arises in any of the phases, we can change the design of that specific phase, without having to change everything of the design. To do so we must define a clear input and output of each phase, to make each phase standalone. This is further explained and broken down in Chapter 4 and Chapter 5. After all the sub-phases are completed, we will combine them into the final prototype model.

Of course, there will always be some difficulties when creating something, and this project was the same. The design of the project has gone through multiple renditions as the project has been progressing as there has been a few barricades when implementing them. But all these barricades have been solved by making improvements to the system.

3.1 System Design Diagram

3.1.1 System Architecture Diagram

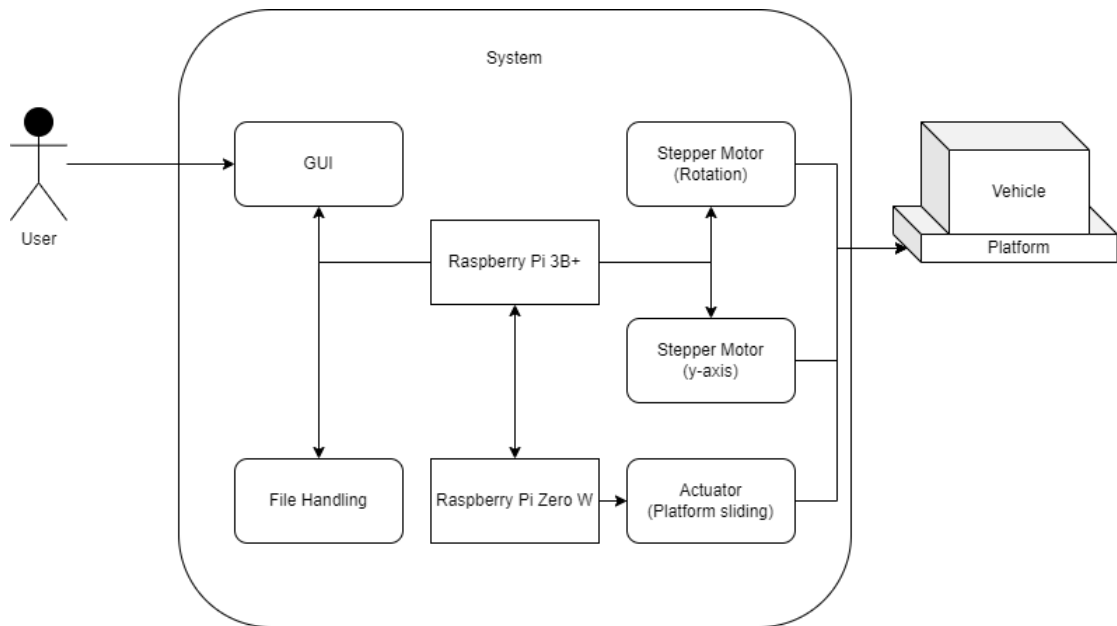


Figure 3.1.1 System Architecture Diagram

3.1.2 Use Case Diagram and Description

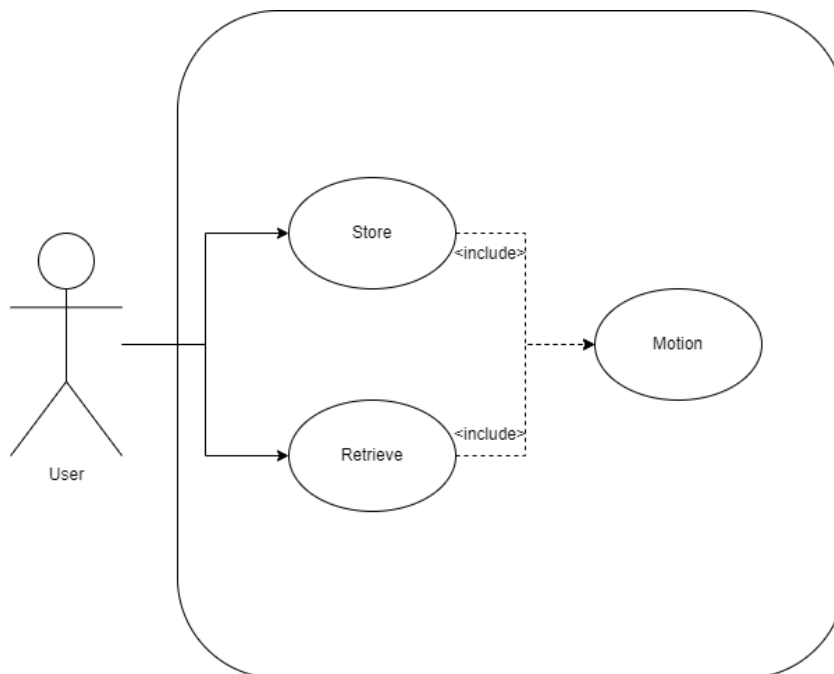


Figure 3.1.2 Use Case Diagram

Store

Use Case Name: Store	Importance Level: High
Primary Actor: User	Use Case Type: Essential
Stakeholders and Interests: User – wants to store their vehicle into the system	
Brief Description: This use case describes the process of user using the system to store their vehicle.	
Trigger: User chooses the store option on the GUI Type: Internal	
Relationships: Association: User Include: Motion Extend: - Generalization: -	
Normal Flow of Events: <ol style="list-style-type: none"> 1. The system displays a choice to store or retrieve 2. User chooses store and is prompted a window showing the current slot number. 3. User keys in the desired password and press store 4. The system saves the password and then looks for the next empty slot. 5. The system sends the current slot and next slot to Motion. 	
SubFlows: - System detects no other empty slot, will stop other users from choosing store.	
Alternate/Exceptional Flows: - User presses close on the store window, window will just close. - User keys in 0 as password which is not allowed, will show prompt to try again.	

Table 3.1.2.1 Use Case Description (Store)

Retrieve

Use Case Name: Retrieve	Importance Level: High
Primary Actor: User	Use Case Type: Essential
Stakeholders and Interests: User – wants to retrieve their vehicle from the system	
Brief Description: This use case describes the process of user using the system to retrieve their vehicle.	
Trigger: User chooses the retrieve option on the GUI Type: Internal	
Relationships: Association: User Include: Motion Extend: - Generalization: -	
Normal Flow of Events: <ol style="list-style-type: none"> 1. The system displays a choice to store or retrieve 2. User chooses to retrieve and is prompted a window asking user to key in slot number and password. 3. User keys in the slot number and password and press retrieve. 4. The system checks the respective slot number and checks the saved password and the one keyed in by user. 5. When the password is correct, the system sets the slot number as next slot. 6. The system sends the current slot and next slot to Motion. 	
SubFlows: - System detects no other empty slot, will stop other users from choosing store.	
Alternate/Exceptional Flows: - User presses close on the store window, window will just close. - User keys in 0 as password which is not allowed, will show prompt to try again.	

Table 3.1.2.2 Use Case Description (Retrieve)

Motion

Use Case Name: Motion	Importance Level: High
Primary Actor: System	Use Case Type: Essential
Stakeholders and Interests: System – Obtains current slot number and next slot number	
Brief Description: This use case describes the process of movement of the system after getting the required data.	
Trigger: The system obtains current slot number and next slot number Type: Internal and External	
Relationships: Association: - Include: Store, Retrieve Extend: - Generalization: -	
Normal Flow of Events: <ol style="list-style-type: none"> 1. The system receives the current slot number and next slot number. 2. The system will calculate the angle and level of the current slot and the next slot. 3. It then calculates the movement between the two slots by checking the difference between them so that the movement will be fluid. 4. The system then moves according to the calculated motion. 5. It is then open to the next user afterwards. 	
SubFlows: -	
Alternate/Exceptional Flows: -	

Table 3.1.2.3 Use Case Description (Motion)

3.1.3 Activity Diagram

Store and Retrieve

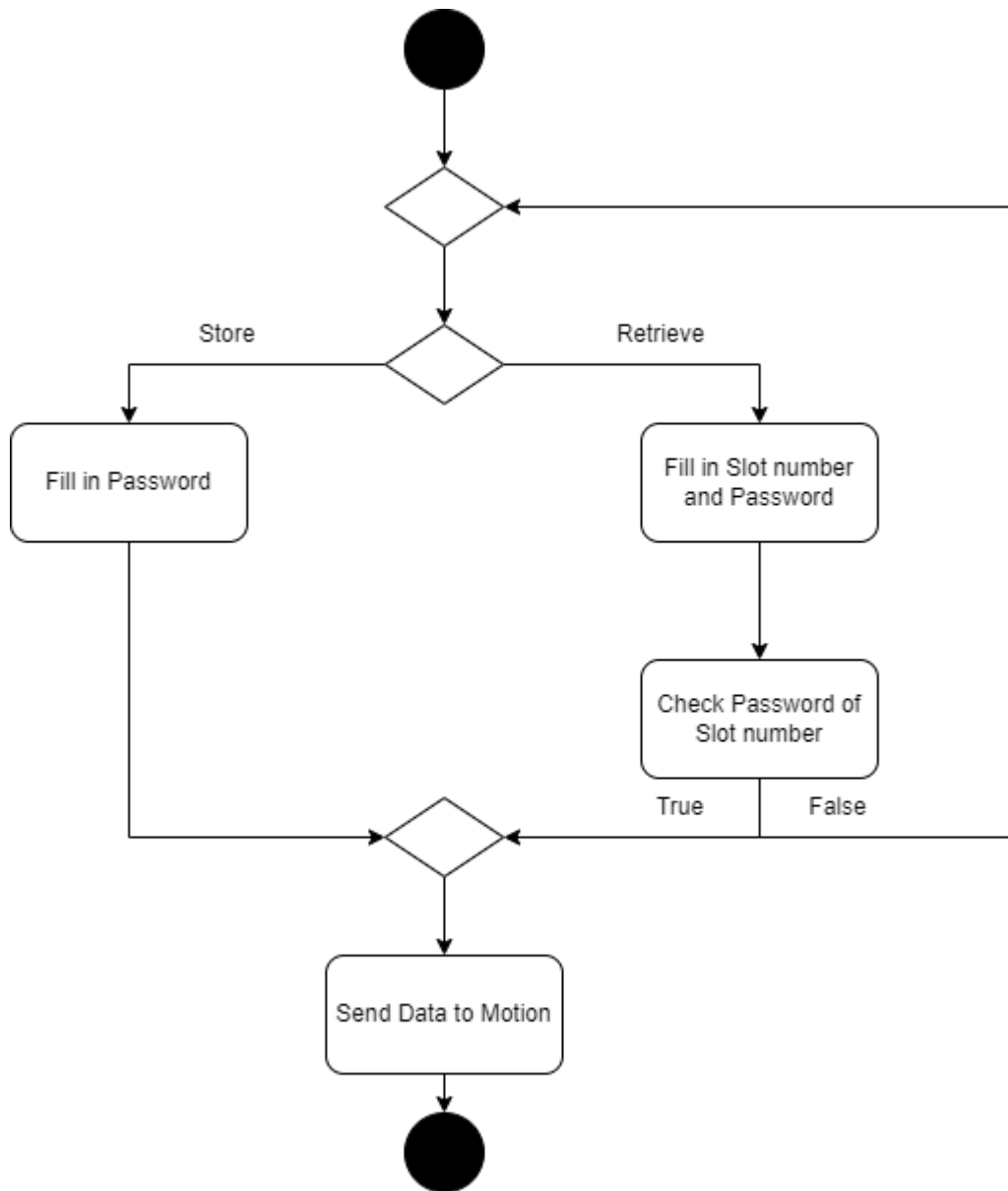


Figure 3.1.3.1 Store and Retrieve Activity Diagram

Motion

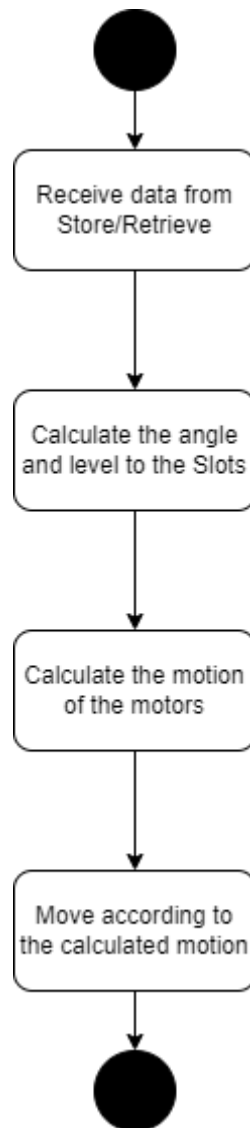


Figure 3.1.3.2 Motion Activity Diagram

Chapter 4

System Design

4.1 System Overview

In the beginning, the user will park their car at the designated spot, which is as if they are parking at a parking slot normally. Then the user will get out of their vehicle, lock it and go to a screen provided near the system, which will show the user the current slot number, and let the user set a password for that slot. After saving the password, the system will then move the vehicle to its designated slot while taking a new platform up for the next user. As for when the user wants to retrieve their vehicle, they will go to the same screen to enter their slot number and password. The system will compare the password with the respective slot, only when the password is the same, it will put the empty platform back to the original slot and retrieve the vehicle from the specified slot.

The entire system will have 5 levels with 4 slots on each level. One stepper motor will be responsible for the rotation/angle of the system, which is set to 0, 90, 180 and 270 degrees. The y-axis movement will be controlled by two other stepper motors. The motors will be connected to a lead screw respectively, which will allow the platform to move up and down depending on the rotations of the motors. There will also be motors that will slide the platform into its respective slots. This whole part of the prototype is called the moving platform as it moves the platforms that will be placeholders for the vehicles.

The moving part will be in the middle of the structure, which is where the slots that will hold the vehicles will be. This structure is constructed to be stable enough to hold the vehicles. In the small-scale prototype, we will only construct parts of the structure and the entire moving part, so that when demonstrating we can show more clearly how the system works.

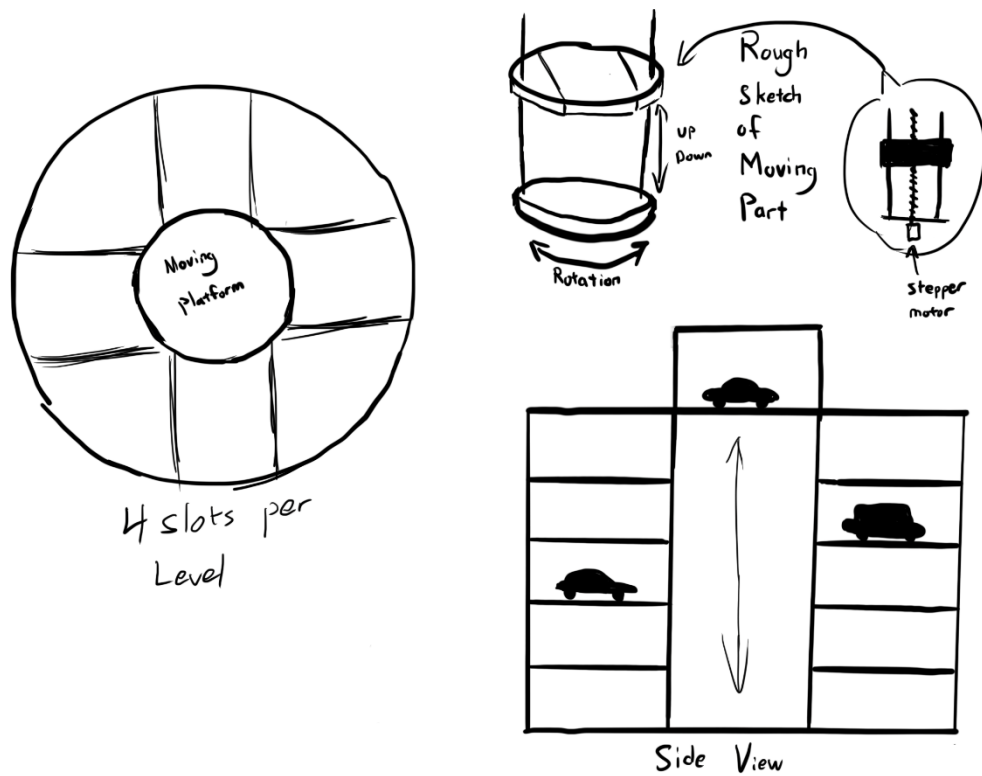


Figure 4.1.1 Rough Sketch of System Proposed

As to further elaborate on how the y-axis will work, the concept used is the same as how a 3D printer does. With a fixed lead screw that will rotate, as long as there is another point of support for the platform that is connected to the lead screw, the platform will be able to slide along when the lead screw rotates. Which can be shown through the pictures below.

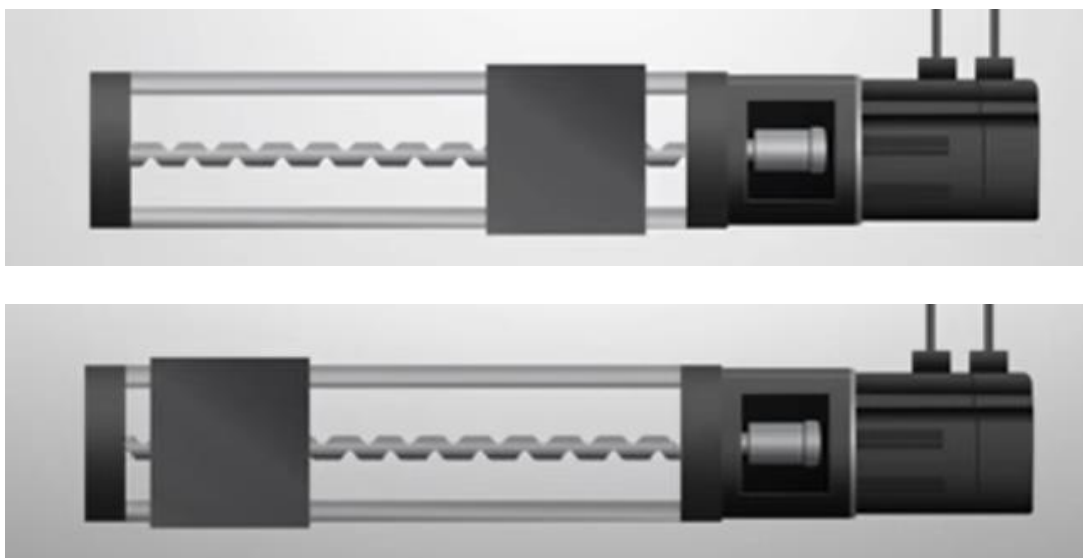


Figure 4.1.2 Concept of moving platform using lead screw and motor

CHAPTER 4

The prototype will be using 2 lead screws, one on each end. Therefore, each screw should be able to act as the point of support for the other lead screw. Further support can be added if needed.

As the system heavily relies on the angles and the number of rotations of the motor to determine the position of the moving platform, the movement of the motors are crucial. When data is entered into the system, after comparing the password, a calculation will be run through the system to determine how much each motor will spin in order for them to reach the designated spot.

4.2 General Work Procedures

First, by looking into the system, it is found that the system can be separated into parts, which we can categorize based on the movement control. After having an idea of how the system should work, we can start to draw out a few renditions of a simple implementation block diagram for the system.

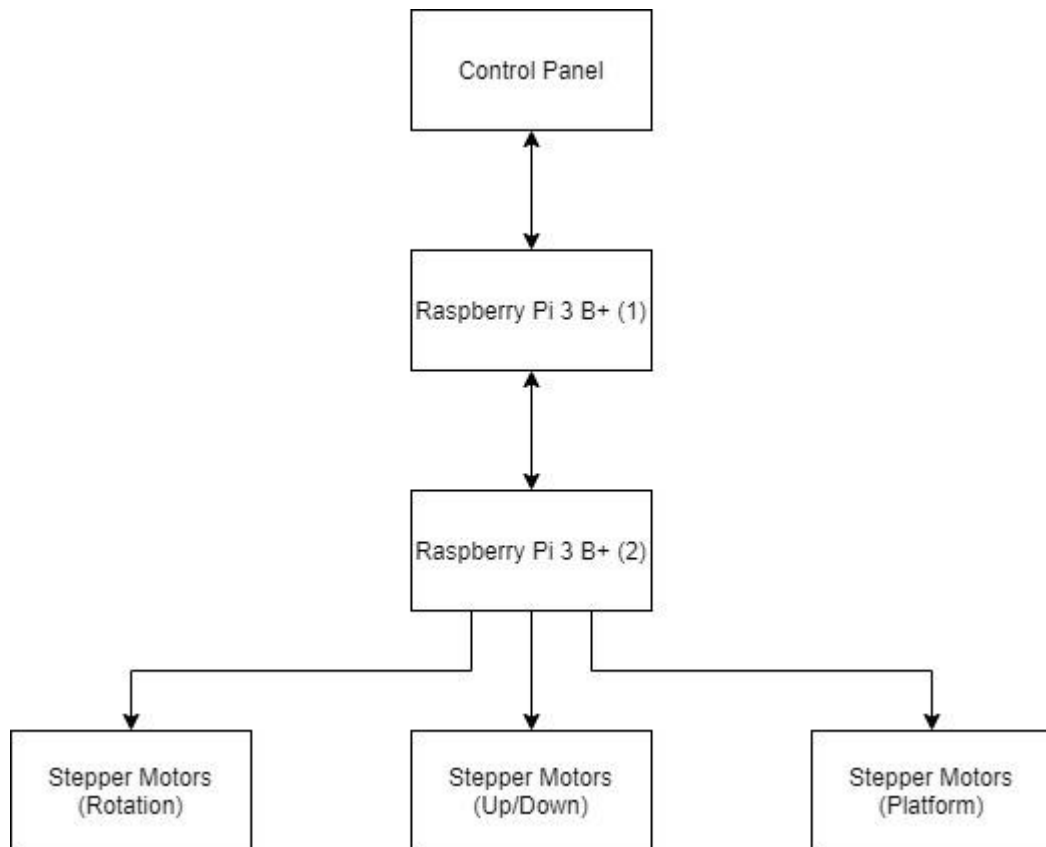


Figure 4.2.1 Previous Hardware Implementation of System from FYPI

From this rendition, we are using 2 different RPis. One to act as the main brain of the system, which handles the User Interface, file handling and communicates with the second RPi with the inputs for each movement. As stated, the second RPi will be responsible of all the movements. This means that the User will never be able to directly access the motors. There are some perks to this design as it provides a safer environment for the system, as there is like a barrier between the user and the motors responsible for the movement.

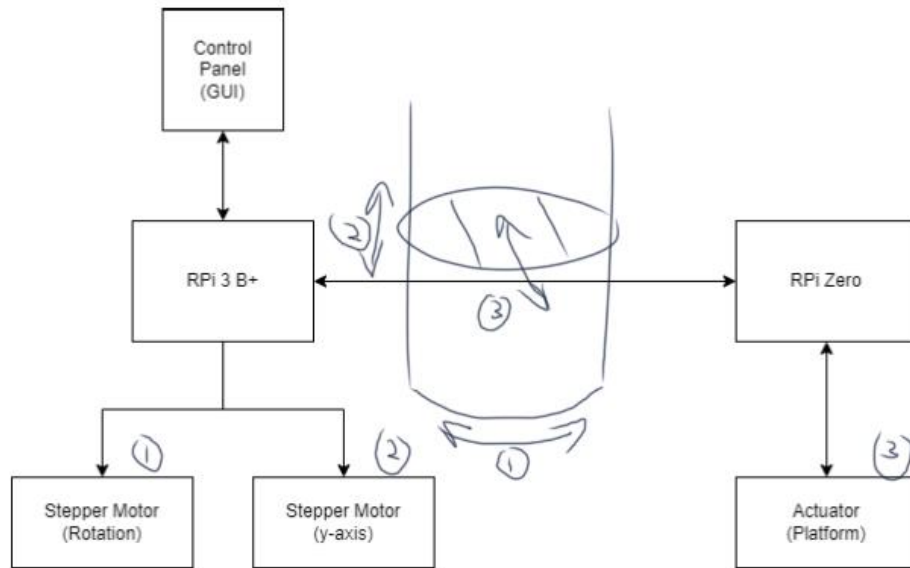


Figure 4.2.2 New Simple Hardware Implementation of System

From this newer rendition of the system, it provides a much cleaner system. The difference from this rendition from the previous is how we separate the workload. When we think from the perspective of security, the first rendition will be better, but the problem that arises is with the wiring. Since only one RPi is controlling all the movement functions. The wiring system must be cleaned up or else wire entanglement is bound to happen with all the rotations and y-axis movements. As for the newer rendition of the design, we separate the movement functions into two sets, one which is responsible for the rotation and y-axis movement, and the other only responsible for the sliding movement.

This means that we can reduce the wiring connected to the moving part of the system to merely one wire which will provide the power. There is a possibility of making it completely wire free as well by attaching a power cell which will be charged when the system is idle. But we are not doing that in this project. With this we can have a better understanding on how we will build the system or even further improve it (Using three RPis instead of 2) if we had a higher budget.

After that, we can start with deciding on the tools that we will use. The tools that we have decided are listed in a subchapter below. As we are using the newer rendition of the implementation block diagram, to describe more on how it works, it can be summarised to as follow. Our system will have two parts, a base which

would be used to control the rotation and y-axis (Up/Down) motion, and a small base(moving platform) that is used to retrieve or store the platforms. As the small base will be moving most of the time, the idea is to keep it as light as possible to not add additional strain to the base motors, therefore we wish to reduce wiring and size of the components. This will be why we chose RPi 3 B+ as the main base RPi and a RPi Zero W as the small base RPi.

RPi 3 B+ will be responsible of the user interface, file handling and control over rotation and y-axis. While RPi Zero W will only be used to control the sliding movement to retrieve or store.

4.3 Tools

4.3.1 Hardware in Use

- Raspberry Pi 3 Model B+

The main hardware involved in this project is a Raspberry Pi 3 B+, which acts as the brain of the system. It is an ARM based single board computer running on Linux Operating System. Basically, it is a microcomputer, able to surf the internet and run programs. The RPi has a 64-bit quad core processor running at 1.4GHz, a dual-band 2.4GHz and 5GHz wireless LAN, Bluetooth 4.2/BLE, fast Ethernet, and a Micro SD format card support[9]. It also is able to send signals to and receive from other RPis.

- Raspberry Pi Zero W

A smaller version of Raspberry Pi which has one micro usb power input, one micro usb for external hardware, and one micro-HDMI port. The W in its name stands for wireless. It is able to do everything a regular RPi can do since it has the same number of pins and ports, however it is weaker in terms of performance. Since the small size, it only has a BCM 2835 SOC Processor and 512MB RAM[10]. However, it is capable enough for this prototype.

- DRV8825 Stepper Motor Drivers

This is a cheap driver that is used to control the stepper motors[11]. A stepper motor driver is able to reduce the audible noise and achieve precise, reliable control of the

stepper motor. By receiving signals from the RPi, it will then send signals to the stepper motors. The controls include rotation angle, rotation speed and torque.

- Stepper Motors

Stepper motors are used to achieve precise control over the system[12]. All the movements of the system will be done with these motors as we want to have precise numbers on the angle, height, and when the system moves the platform to its respective spots, this is all in order to ensure the vehicles won't get damaged or fall during the transition from slot to slot.

- L298N Motor Driver

The L298N Motor Driver is an integrated monolithic circuit that can be used to drive inductive loads[13], which in our case is to be used to control the linear actuator. It is able to control two DC motors or one stepper motor depending on the assigned use. This motor driver was chosen because there were uncertainties when buying the components as there were many ideas for the platform movement. It was either using stepper motors to reduce the variety of hardware used or to use an actuator. L298N offered the choice to be able to control both.

- Linear Actuator

A linear actuator is actually a DC motor but instead of rotational motion, it converts the motion into linear motion[14]. These motors are usually used in electric beds in hospitals, electric wheelchairs, and also solar panel trackers. There are various types, some with a slower speed but higher thrust, or some with longer extension length but weaker thrust. They also come in different sizes based on user need.

4.3.2 Software in Use

- Geany

Geany is the programming software used in the project as it is pre-installed in the RPi. As we will be coding using Python as the programming language, and tests are always run to ensure the motors are working fine, Geany is chosen as it is a small and lightweight integrated development environment, providing small and fast IDE.

4.3.3 Programming Language in Use

- Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. It is simple and easy to learn as its syntax emphasizes

CHAPTER 4

readability. Python also supports modules and packages, encouraging program modularity and code reuse. Not to mention, RPi comes with software that allows users to code in Python. It is especially advantageous to use Python when one wishes to do object oriented, and GUI programming productively, as it does not have a compilation step.

4.4 Circuits and Tools Design

- Raspberry Pi 3 B+ (Rotation Motion and Y-axis Motion Control)

For the connection between the RPi to the motor drivers to the motor, they are connected this way as an example.

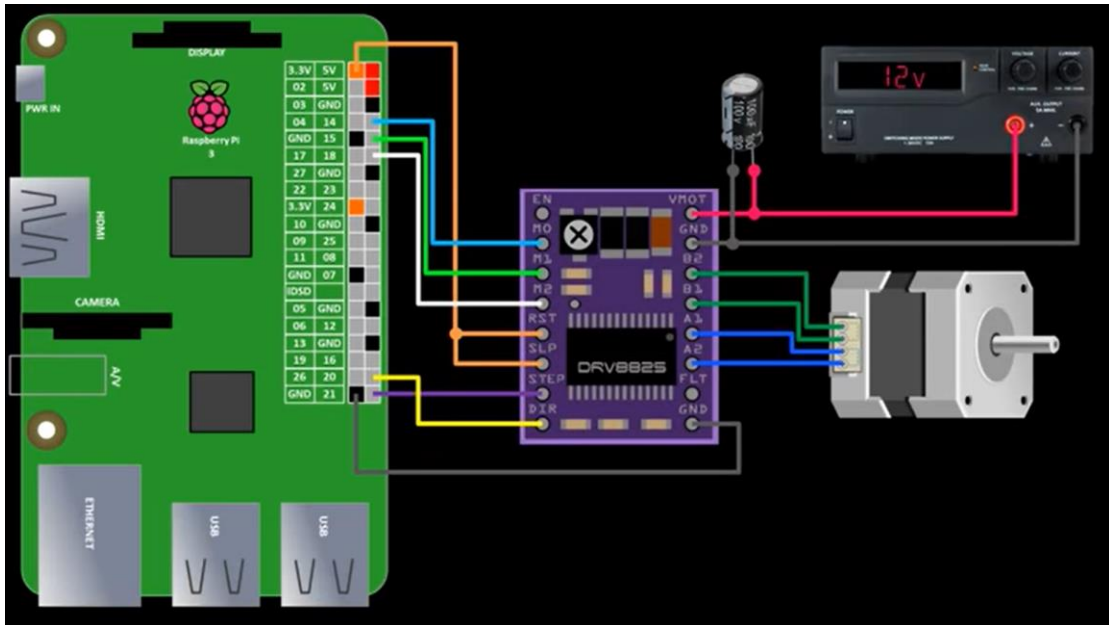


Figure 4.4.1 Sample connection (RPi 3B+)

Pin 14, 15, 18 from the RPi is connected to M0, M1, M2 of DRV8825, these three pins are used to set the stepping mode of the motor, ranging from full step to 32 steps. By setting the stepping mode, it influences the speed of the motors and also the torque of the motor. The outputs on the RPi Pin 14, 15, and 18 are set to the respective outputs for each mode as shown below.

Mode0	Mode1	Mode2	Step Mode
0	0	0	Full step (2-phase excitation)
1	0	0	1/2 step (1-2 phase excitation)
0	1	0	1/4 step (W1-2 phase excitation)
1	1	0	8 microsteps/step
0	0	1	16 microsteps/step
1	0	1	32 microsteps/step

Figure 4.4.2 Step Mode of DRV8825

The reset(RST) and sleep(SLP) are connected to 3.3V as they are to be set to high. This is because their default low states are to reset and sleep. The step pin on DRV8825 will be connected to pin 21 and direction pin(DIR) will be connected to

pin 20 of the RPi. Any GPIO pins on the RPi can be used since they will be specified in the code. The Motor Positive pin (VMOT) is connected to the positive terminal of the power supply while the following Ground pin (GND) is connected to the negative terminal. A capacitor is placed across the motor supply as close to the driver as the DRV8825 is susceptible to destructive voltage spikes. As for the B2, B1, A1, A2 pins, they are connected to the motor. The motors coils should be determined so that one coil will be on A1, A2 and the other on B1, B2. Otherwise, the motors won't be able to function. The last ground pin will be connected to the ground of the RPi. The ground pins can also be connected together to make a common ground for the system. Since we are using more than one motors, some RPi output signals from GPIO pins such as Pin 14, 15, 18 can be kept the same for all the drivers. Other connections like the direction and step can also be used for some motors that have the same function, for example the 2 motors that control the y-axis movement.

- Raspberry Pi Zero W

To control the sliding movement of the platform during placing back the current slot and retrieving the next slot, a Raspberry Pi Zero W is used because of its smaller size. For this part of the system, it will be built onto the moving platform so the lighter the components the better. The connections of this part of the system are almost the same as the ones with DRV8825 stepper motor driver, but since we will be using a Linear Actuator, we will use L298N as the motor driver. The Linear actuator is responsible for the sliding of the platform from the system to its designated spot or vice versa. It will extend and retract based on what the system needs it to do. Since the linear actuator actually behaves as such of a dc motor, the connection from the Raspberry Pi will only require In1, In2 and Ena.

There is no need for another wire for speed or direction as we can directly affect it by changing the output for In1 and In2. As mentioned before, the Linear Actuator behaves like a DC motor, therefore the L298N allows control over the flow of power through the motor, when In1 is set high and In2 is set low, the linear motor will be set to extend while when In2 is set high and In1 low, it is set to retract. The Ena will be used to control whether the linear actuator will receive the power, so if it is set low, there will be no movement. As we are using a linear actuator, the Ena

can be set high always as the actuator will actually stop itself when it reaches the maximum extension and maximum retraction. The sample connectivity can be seen shown below.

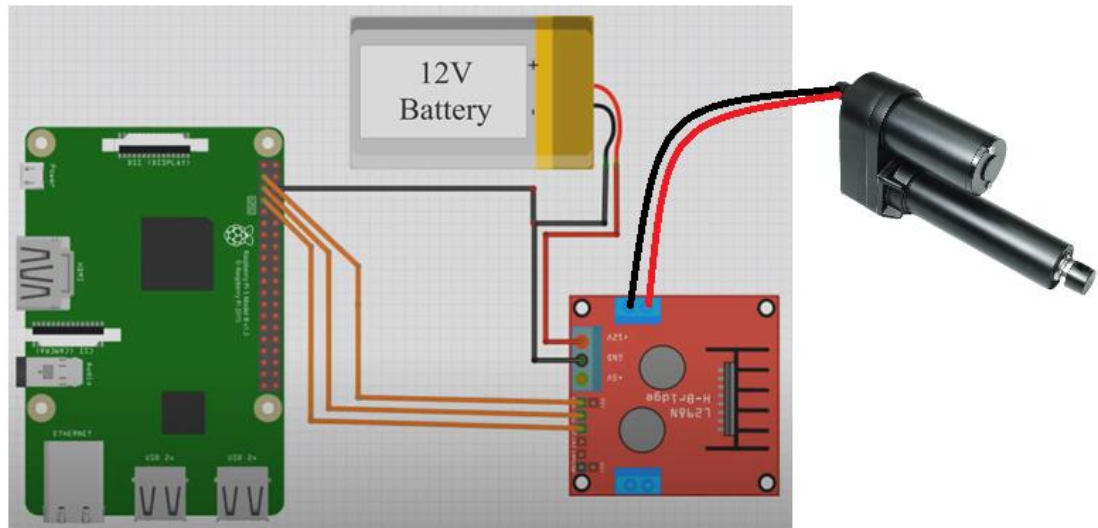


Figure 4.4.3 Sample connection (RPi Zero W)

The communication between the two Rpis, Raspberry Pi 3 B+ and Raspberry Pi Zero W, will be done using MQTT. When RPi 3 B+ gets the data, it calculates the movement for the angles and levels of the current slot and next slot, and since the movement from RPi Zero W is always the same, it is programmed to do the same motion when called. The flow becomes such that after calculating the angle and level, the stepper motors controlled by RPi 3 B+ will move first to the current slot position, then it will send a signal to RPi Zero W so that it will slide the platform into its designated slot. RPi Zero W will then send a signal back to RPi 3 B+ to let it know it has slid the platform in, then the stepper motors controlling the y-axis will lower a bit before sending another signal back to RPi Zero W to let it retract. The communication is then repeated after the retraction, so that RPi 3 B+ will start the motion towards the next platform. The communication between both RPis will be the same with the extension and retraction, only that the y-axis will rise a little instead before the retraction. After obtaining the next platform, the system will return to the top, completing the motion.

- GUI and File Handling

A simple GUI is built for prototyping purposes, it just has the basic buttons to store, retrieve, provide windows for each choice, and get the input data. As for file

handling, it is to read data from and write data to a file. This is used to act as a backup save file of the parking slots. The file will contain the slot number, condition, and password of all the parking slots. So, when the system is boot up, all these data will be read into the system, and after each set of movements, the data will be updated from the system into the file. This way, even if when the system is shut down, it will retain all the data when it is booted up again. An LED is also place in to indicate when the system is full.

All the basic coding will be explained in Chapter 5 Software Setup.

4.5 Flowchart

The software will be made to match the functions in the flowchart below.

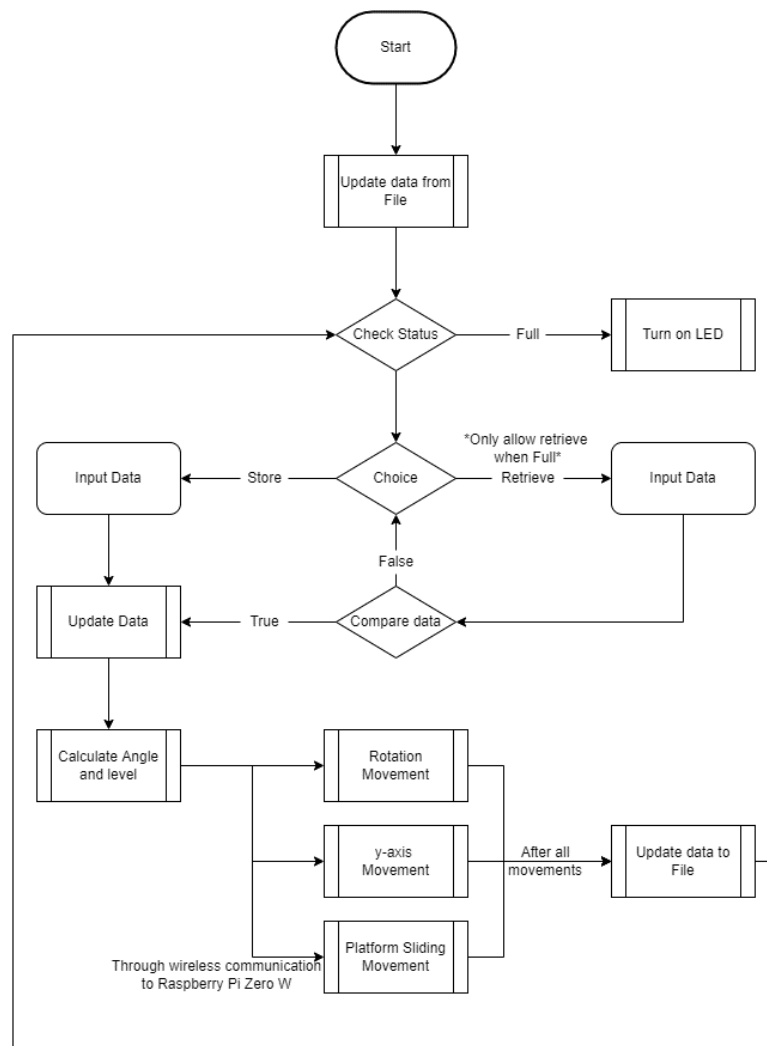


Figure 4.5.1 Flowchart

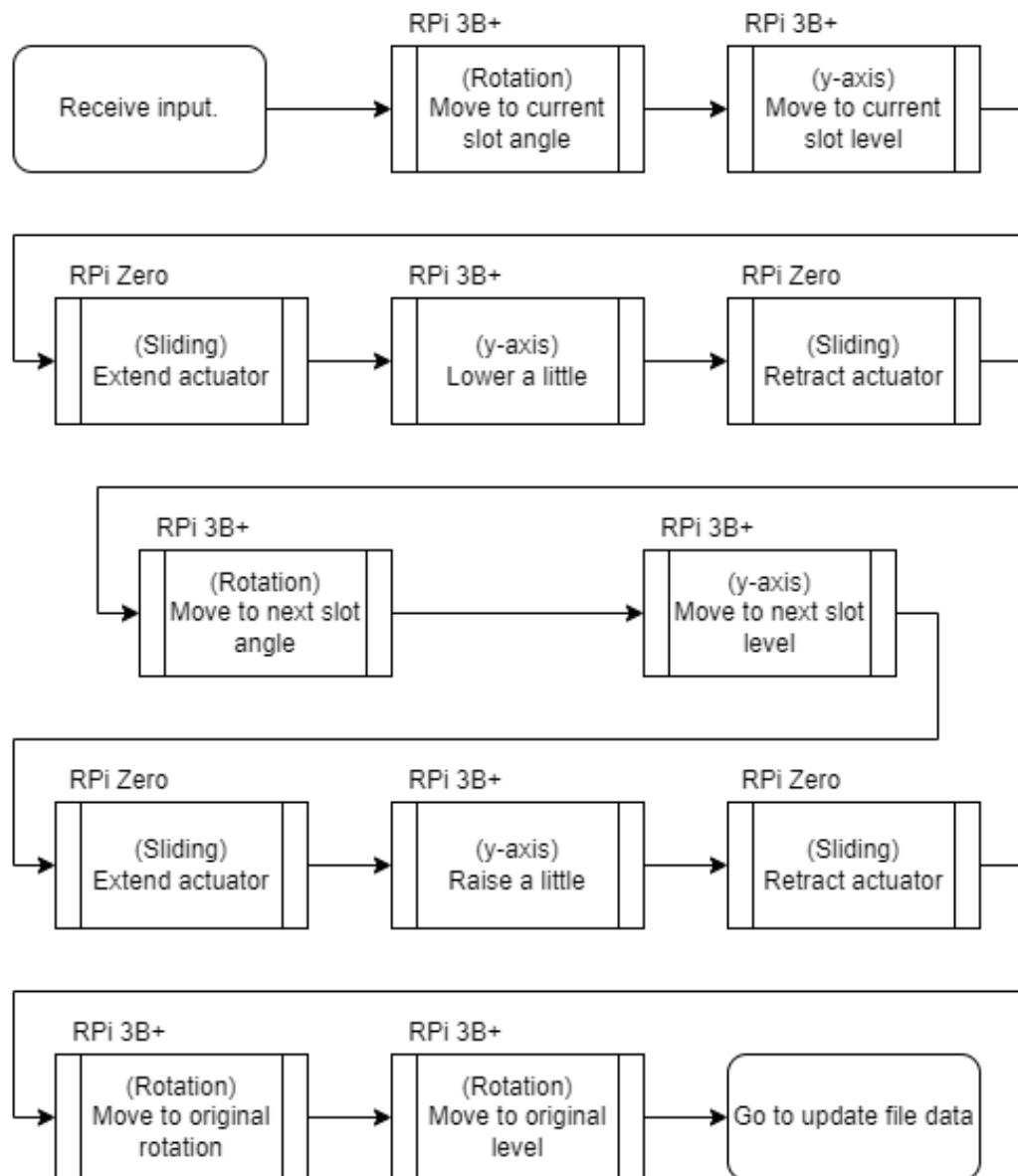


Figure 4.5.2 Movement Function Flowchart

From the flowchart above, we can understand clearer on how the movement is done. When the Angle and Level data has been calculated, it starts by placing the current platform back to its original slot. First it will rotate to the correct angle, then lower to the correct level. A signal is sent from RPi 3B+ to RPi Zero W after the previous movement, to let it know it should now extend the actuator. The extension of actuator will also slide the platform into its respective slot, as the concept is explained below.

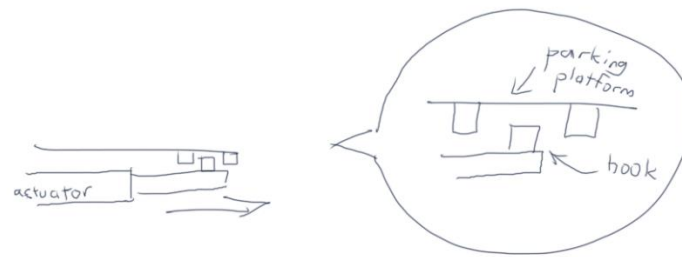


Figure 4.5.3 Visualisation of Actuator and Platform

The platform will then lower a little to “unhook” the platform from the actuator before another signal is sent to retract it. Then it will rotate to the new angle and move higher or lower based on what the next platform level is. Do take note that at this point, the position moving platform will be situated lower than the actual platform after the level movement. This is because we never reverted the lower movement when we unhooked. It then extends the actuator and raises the platform to “hook” into the new platform before retracting, pulling the new platform into the moving platform. This whole process will be clearer when we explain on the prototype built.

Chapter 5

System Implementation

5.1 Hardware Setup

5.1.1 Raspberry Pi 3 B+ to DRV8825 drivers

The wiring and connections were done for three motors, which is responsible for the base movement, which is the rotation and y-axis movement. The connections are based on what's stated in chapter 4, with the output from pin 14, 15, 16 connected to all the drivers as they will all be using the same stepping mode. The power supply is also connected in parallel to all the motor drivers so that each motor will receive enough power.

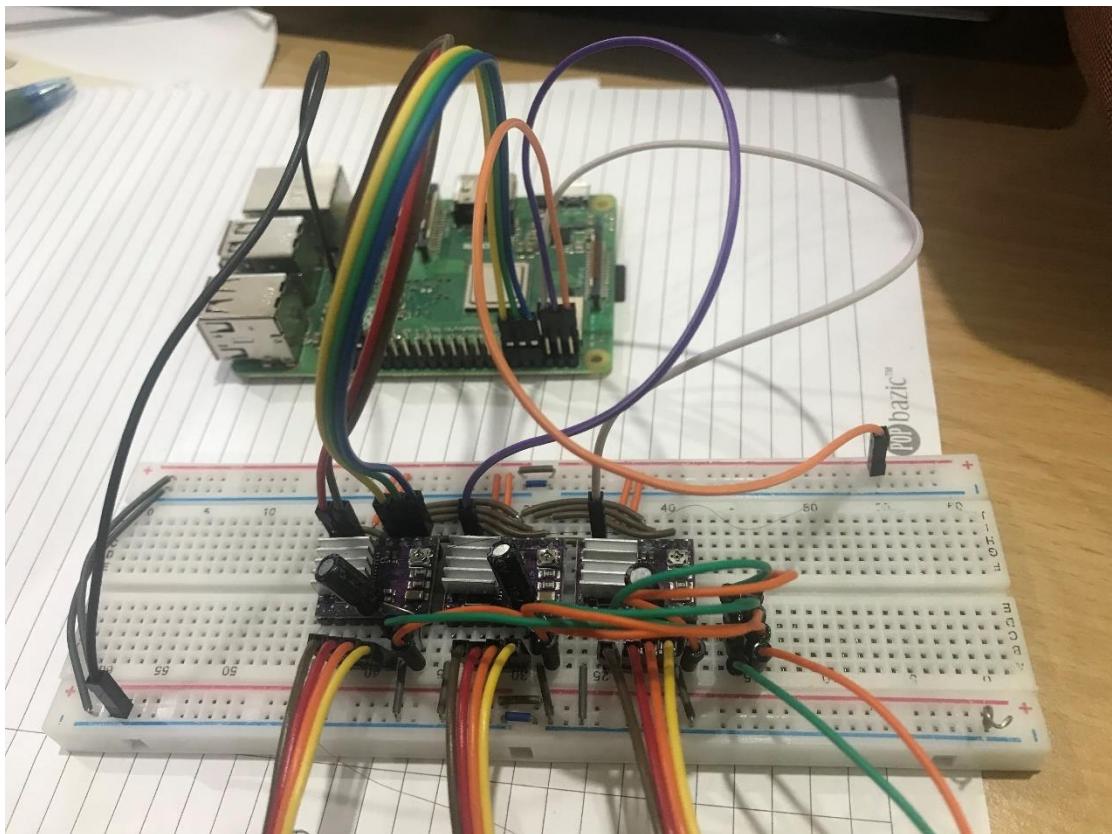


Figure 5.1.1a Wiring of the RPi 3 B+

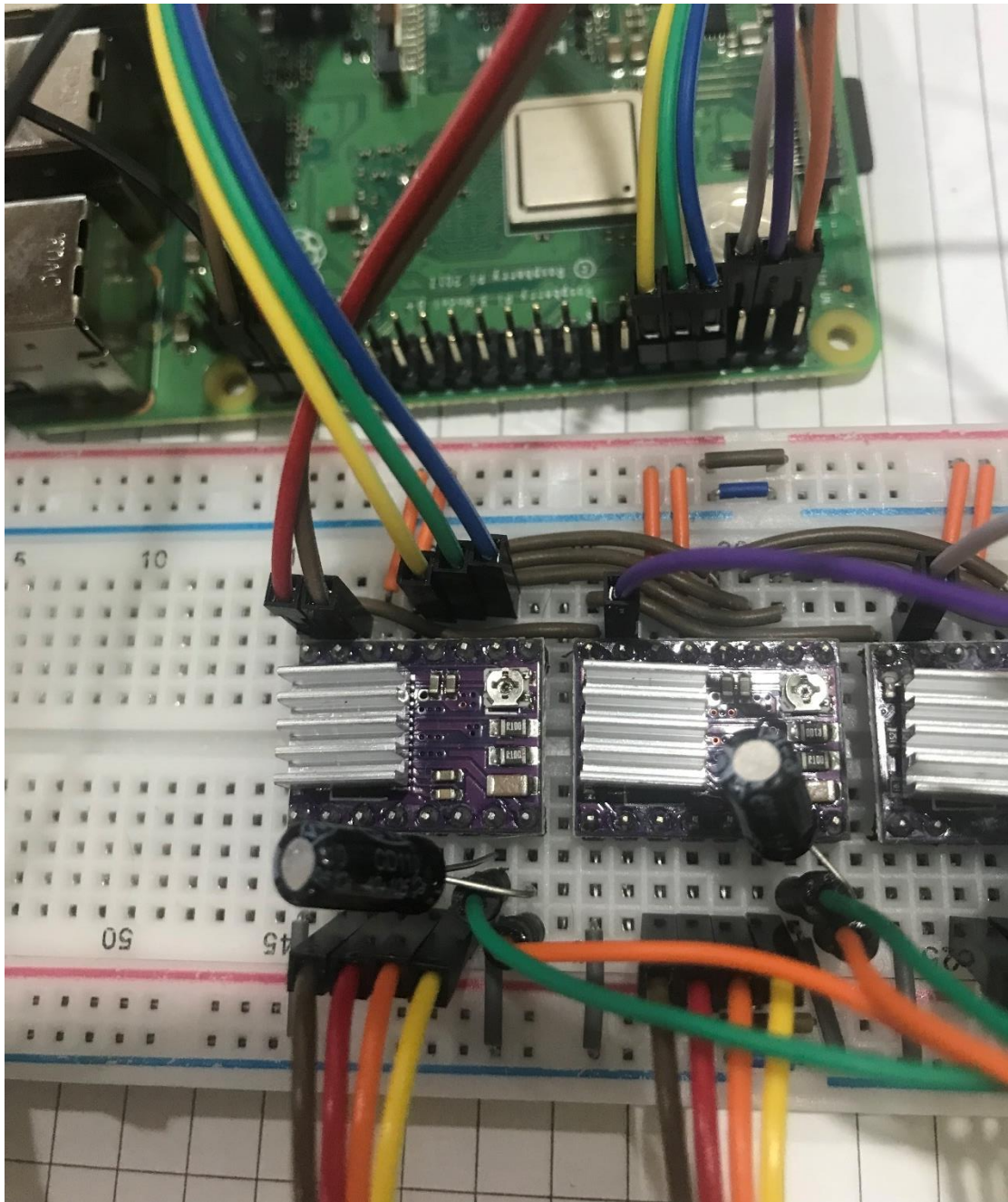


Figure 5.1.1b Wiring of the RPi 3B+

5.1.2 Stepper Motor/Lead Screw

Two of the Stepper motors will be connected to lead screws using a flexible coupling shaft. All the while with the lead screw held in place using mounted ball bearings to smooth out the rotation. On each of the lead screw will be a screw nut that will then be used to mount the moving platform.



Figure 5.1.2 Motors connected to lead screw with coupling shaft

5.1.3 Raspberry Pi Zero W to L298N to Actuator

By setting up RPi Zero W based on as described in Chapter 4, it will then be connected to the linear actuator. As for RPi Zero, we will only use it to control the actuator, we can skip the setup through a breadboard and directly connect everything to the L298N and Linear Actuator. From Chapter 4, we know that L298N will only require 3 inputs to control the actuator. For this case we use GPIO13, GPIO19 and GPIO26 as Ena, In1 and In2. The actuator is also connected to the correct side as the inputs on the LN298 driver.

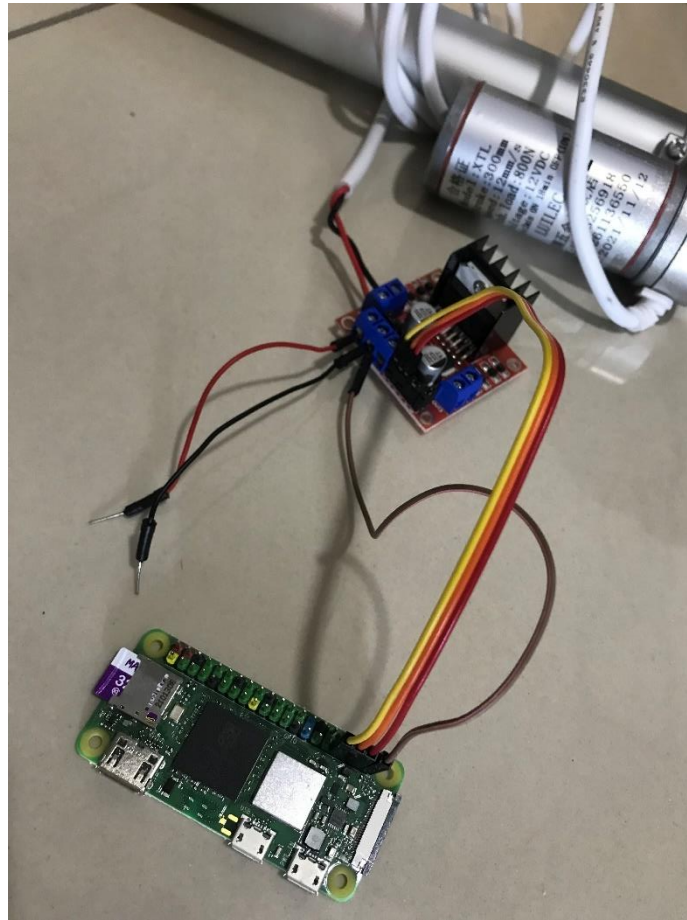


Figure 5.1.3a Wiring of RPi Zero W



Figure 5.1.3b Wiring of RPi Zero W

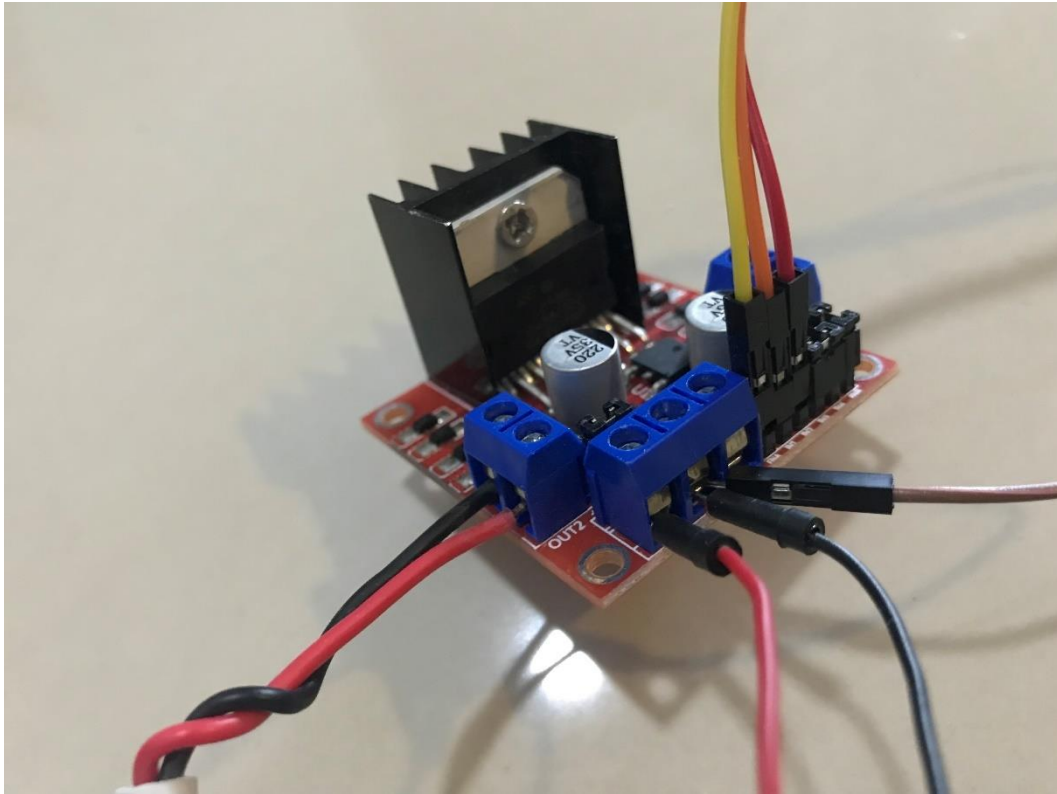


Figure 5.1.3c Wiring of RPi Zero W

5.2 Software Setup

First a simple code was created to test the motors and actuators. Then, a simple GUI is designed as a substitute for prototyping purposes. The system was able to save data in the right space and compare them correctly. When moving into file handling, it took a bit of time to understand how the files are saved, but after a few days of tinkering, the system was working as intended, with the data saved and retrieved successfully. Then it went to programming the control of the motors into the system, as we have tested the motors before, the codes were the same. The only thing needed to be added were the calculations that will be made to command the motors, formulas were used to set 5 levels with 4 angles at each level, representing 4 slots at each level. These numbers are calculated when the input is keyed into the system, determining which level and angle to go to first and then which level and angle to go to next before returning an empty platform/platform with vehicle back to the top.

The codes will be explained below briefly with provided code snippets of the actual code.

5.2.1 Raspberry Pi 3B+

5.2.1.1 File Handling and LED

```

#File Handling and LED
slotnumber = []
slotcon = [] #0=unoccupied, 1=occupied, 2=Slot on platform
password = []
count = 0
LED = 26
Full = 1
Ava = 0
occupied = 0
GPIO.setup(LED, GPIO.OUT)
f = open("parkingslot.txt", "r")
for x in f:
    count = count+1
    if count == 1:
        slotnumber.append(x)
    elif count == 2:
        slotcon.append(x)
        if slotcon(x) == 2:
            occupied += 1
    elif count == 3:
        password.append(x)
        count = 0
f.close()
if occupied == 20:
    GPIO.output(LED, Full)
else GPIO.output(LED, Ava)

def save_and_check():
    occupied = 0
    f = open("parkingslot.txt", "w")
    for a in range(len(slotnumber)):
        f.write(slotnumber[a])
        f.write(slotcon[a])
        f.write(password[a])
        if slotcon[a] == "1\n":
            occupied+=1
    f.close()
    if occupied == 20:
        GPIO.output(LED, Full)
    else GPIO.output(LED, Ava)

```

Figure 5.2.1.1 Code Snippet(File Handling and LED)

This is the codes set up and definition for file handling and LED. First, we declare the variables and set GPIO 26 as output. Then we will get the data from parkingslot.txt and check if all the slots are filled. If they are all occupied, the LED will light up, if not the LED will turn off. Then we define a function that will be done to save the data into the file and check if the slots are full. This function will be called after all the movement functions are done.

5.2.1.2 Hardware Declaration (DRV8825)

```

#Hardware
DIR = 21 # Direction GPIO Pin
STEP1 = 20 # Step GPIO Pin
STEP2 = 2
CW = 1 # Clockwise
CCW = 0 # Counterclockwise
SPR = 200 # Steps per revolution (360/1.8)

GPIO.setup(DIR, GPIO.OUT)
GPIO.setup(STEP1, GPIO.OUT)#rotation
GPIO.setup(STEP2, GPIO.OUT)#Y-axis
GPIO.output(DIR, CW)

MODE = (14,15,18)
GPIO.setup(MODE, GPIO.OUT)
RESOLUTION = { 'Full':(0,0,0),
               'Half':(1,0,0),
               '1/4':(0,1,0),
               '1/8':(1,1,0),
               '1/16':(0,0,1),
               '1/32':(1,0,1)}
GPIO.output(MODE, RESOLUTION['Half'])

step_count1 = SPR * 2 * 5 #based on resolution, if 1/4 then *4 and so on
                        #since we are using lead screw,we need to determine how many
                        #revolutions for platform to move form one end to the other,
                        #then * that number to step count(Only for testing purpose)

step_count2 = SPR * 2
delay = .0208/2 #also based on resolution, if 1/4 then /4 and so on

```

Figure 5.2.1.2 Code Snippet(Hardware Declaration)

These are the setup and declaration for the Hardware, specifically for the inputs of DRV8825. GPIO pin 21, 20, 2 are set as the output for DIR(direction), STEP1 and STEP2(Steps for GPIO Pin). There are two steps because one is for y-axis while the other is for rotation. GPIO pins 14, 15 and 18 will be used to set the Mode, which has been explain in Chapter 4. We will also set step_count, which is the number of steps to make a full revolution. step_count1(for y-axis) will be with x5 as we have set five revolutions for each level.

5.2.1.3 Movement

```

#Movement
def basicmovement(step, angle, direction):
    GPIO.output(DIR, direction)
    for x in range(int(angle)):
        GPIO.output(step, GPIO.HIGH)
        sleep(delay)
        GPIO.output(step, GPIO.LOW)
        sleep(delay)
    sleep(.5)

def movement(option, action):
    global turnangle2, returnangle, turnlevel2, returnlevel, angleDIR, levelDIR
    currentslot = 0
    for y in range(len(slotnumber)):
        if slotcon[y]=="2\n":
            currentslot = y
            if action == "store":
                slotcon[y] = "1\n"
            elif action == "retrieve":
                slotcon[y] = "0\n"

    levelcur = int((currentslot)/4)+1 #determine the level #unsure yet
    anglecur = int(currentslot%4) #split into 4 angles
    turnangle1 = step_count2/4*anglecur
    turnlevel1 = step_count1*levelcur

    newslot = option
    newslotnum = slotnumber[option]
    levelnew = int(option/4)+1 #determine the level #unsure yet
    anglenew = int(option%4) #split into 4 angles

    slotcon[option]='2\n'
    password[option]='0\n'

    if anglecur>=anglenew:
        angledif = anglecur - anglenew
        angleDIR = CCW
    else:
        angledif = anglenew - anglecur
        angleDIR = CW

    if levelcur>=levelnew:
        leveledif = levelcur - levelnew
        levelDIR = CW
    else:
        leveledif = levelnew - levelcur
        levelDIR = CCW

    turnangle2 = step_count2/4*angledif
    returnangle = step_count2/4*anglenew

    turnlevel2 = step_count1*leveledif
    returnlevel = step_count1*levelnew

```

Figure 5.2.1.3 Code Snippet(Movement)

With the declaration from before, we will use it to define a basic movement function which will be used by all the other movement functions. `basicmovement()` will take in step, angle and direction. step being the motor (STEP1 or STEP2), angle being the rotation input which will be calculated, and direction being clockwise or counterclockwise.

As we can see, the movement function will be called when user wants to store or retrieve their vehicle, here is where the calculations are done. This is done by receiving the next slot number, which in retrieving case is the slot number user key in, and in

store case is the next available empty slot. We will also take the input action, to know what to do with the current slot. The calculation is done by comparing the angle and the level of current slot and the next slot. After calculation, the `basicmovement()` function will be called with their respective inputs.

5.2.1.4 MQTT

```
def on_connect(client, userdata, flags, rc):
    print("rc: "+str(rc))
    if rc==0:
        print("Connection OK")
    else:
        print("Connection Not OK")

def on_publish(client, userdata, result):
    print("Message Published\n")

def on_subscribe(client, obj, mid, granted_qos):
    print("Subscribed with QOS: " + str(granted_qos))

def on_message(client, userdata, msg):
    msg.payload = msg.payload.decode("utf-8")
    print("Topic: "+msg.topic+"\n Message: "+str(msg.payload))
    if msg.payload == '1':
        pub("2")
    elif msg.payload == '2':
        movement2()
    elif msg.payload == '3':
        pub("4")
    elif msg.payload == '4':
        movement3()

c1 = mqtt.Client('Rpi3B+')
c1.on_connect=on_connect
c1.on_publish=on_publish
c1.on_subscribe=on_subscribe
c1.on_message=on_message
c1.connect(broker,port)
c1.subscribe(topic2,0)
c1.loop_start()

def pub(x): #publish
    msg = x
    #yaxis
    if msg == '1': #extend
        c1.publish(topic1,msg)
    elif msg == '2': #Down Retract
        basicmovement(STEP2, (SPR*2), CCW)
        print ("lower to retract")
        c1.publish(topic1,msg)
    elif msg == '3': #extend
        c1.publish(topic1,msg)
    elif msg == '4': #Up Retract
        basicmovement(STEP2, (SPR*2), CW)
        print ("rise to retract")
        c1.publish(topic1,msg)
```

Figure 5.2.1.4 Code Snippet(RPi3B+ MQTT)

Since RPi 3B+ will be communicating with RPi Zero W through MQTT, this will be the code. We have to define the function and declare the settings. We will also declare what to do when we receive a reply and what to publish.

As we can see from the code above, there will be different messages published and received as there will be communication between the two RPis throughout all the motion movements.

5.2.1.5 GUI

```

#GUI
window = Tk()
window.title("Smart Underground Parking System")
screen_width = window.winfo_screenwidth()
screen_height = window.winfo_screenheight()
app_width = 500
app_height = 400
window.geometry(f"{app_width}x{app_height}+{int(screen_width/2 - app_width/2)}+{int(screen_height/2 - app_height/2)}")
lbl1 = Label(window, text = "Welcome to the S.U.P.S")
lbl1.place(relx=0.5, rely=0.05, anchor=CENTER)
lbl2 = Label(window, text = "What would you like to do?")
lbl2.place(relx=0.5, rely=0.15, anchor=CENTER)

def SuccessWindow():
    global pop
    pop = Toplevel(window)
    pop_width = 400
    pop_height = 200
    pop.geometry(f"{pop_width}x{pop_height}+{int(screen_width/2 - pop_width/2)}+{int(screen_height/2 - pop_height/2)}")
    pop.config(bg="green")
    pop_label = Label(pop, text="SUCCESS", bg = "green", fg="white", font=("helvetica", 24))
    pop_label.pack(pady=10)
    pop.after(5000, lambda:pop.destroy())

def FailWindow():
    global pop
    pop = Toplevel(window)
    pop_width = 400
    pop_height = 200
    pop.geometry(f"{pop_width}x{pop_height}+{int(screen_width/2 - pop_width/2)}+{int(screen_height/2 - pop_height/2)}")
    pop.config(bg="red")
    pop_label = Label(pop, text="Please Try Again", bg = "red", fg="white", font=("helvetica", 24))
    pop_label.pack(pady=10)
    pop.after(5000, lambda:pop.destroy())

def CompleteWindow():
    global pop
    pop = Toplevel(window)
    pop_width = 400
    pop_height = 200
    pop.geometry(f"{pop_width}x{pop_height}+{int(screen_width/2 - pop_width/2)}+{int(screen_height/2 - pop_height/2)}")
    pop.config(bg="green")
    pop_label = Label(pop, text="COMPLETE", bg = "green", fg="white", font=("helvetica", 24))
    pop_label.pack(pady=10)
    pop.after(5000, lambda:pop.destroy())

def parkPopupwindow():
    for i in range(len(slotnumber)):
        if slotcon[i]=="2\n":
            currentslot=i
    setpassword = simpledialog.askstring("Slot: " + slotnumber[currentslot][0:2], "Please Set Your Password")
    password[currentslot] = setpassword + "\n" #need to add if statement
    for i in range(len(slotnumber)):
        if slotcon[i]=="0\n":
            movement(i, "store")
            break

```

Figure 5.2.1.5a Code Snippet(GUI)

```

def rchoice(option):
    inputSlot = inSlot.get() + "\n"
    inputPass = inPass.get() + "\n"
    pop.destroy()
    if option == "yes":
        for i in range(len(slotnumber)):
            if slotnumber[i]==inputSlot and password[i]==inputPass and slotcon[i]=="1\n":
                movement(i, "retrieve")
                print ("Success")
                SuccessWindow()
                break
        else:
            if i == len(slotnumber)-1:
                print ("Fail")
                FailWindow()
    else:
        print("-") #do nothing

def retrievebutton():
    global pop
    pop = Toplevel(window)
    pop.title("Retrieve")
    pop_width = 400
    pop_height = 200
    pop.geometry(f"{pop_width}x{pop_height}+{int(screen_width/2 - pop_width/2)}+{int(screen_height/2 - pop_height/2)}")
    pop.config(bg="green")

    pop_label = Label(pop, text="Please enter your Slot Number and Password", bg = "green", fg="white", font=("helvetica", 12))
    pop_label.pack(pady=10)

    my_frame = Frame(pop)
    my_frame.pack(pady=5)

    global inSlot
    global inPass
    inSlot = Entry(my_frame)
    inSlot.grid(row=0, column=1)
    inPass = Entry(my_frame)
    inPass.grid(row=1, column=1)
    check = Button(my_frame, text="Enter", command=lambda:rchoice("yes"))
    check.grid(row=2, column=0)
    cancel = Button(my_frame, text="Cancel", command=lambda:rchoice("no"))
    cancel.grid(row=2, column=2)

    btn2 = Button(window, text="Retrieve", fg='red', font=("Helvetica", 32), command=retrievebutton)
    btn2.place(relx=0.75, rely=0.45, anchor=CENTER)

def on_closing():
    if messagebox.askokcancel("Quit", "Do you want to quit"):
        window.destroy()
window.protocol("WM_DELETE_WINDOW", on_closing)

```

Figure 5.2.1.5b Code Snippet(GUI)

As seen, we have defined a lot of pop-up windows, with different messages, as well as the park and retrieve windows. From the code above we can see the other process retrieving a vehicle has to go through (Checking the password), before calling the movement function.

5.2.2 Raspberry Pi Zero W

5.2.2.1 Hardware Declaration (L298N)

```
class Actuator():
    def __init__(self, Ena, In1, In2):
        self.Ena = Ena
        self.In1 = In1
        self.In2 = In2
        GPIO.setup(self.Ena, GPIO.OUT)
        GPIO.setup(self.In1, GPIO.OUT)
        GPIO.setup(self.In2, GPIO.OUT)
        self.pwm = GPIO.PWM(self.Ena, 100)
        self.pwm.start(10)

    def moveF(self, x=100, t=2):
        GPIO.output(self.In1, GPIO.LOW)
        GPIO.output(self.In2, GPIO.HIGH)
        self.pwm.ChangeDutyCycle(x)
        sleep(t)

    def moveB(self, x=100, t=2):
        GPIO.output(self.In1, GPIO.HIGH)
        GPIO.output(self.In2, GPIO.LOW)
        self.pwm.ChangeDutyCycle(x)
        sleep(t)

    def stop(self, t=0):
        self.pwm.ChangeDutyCycle(0)
        sleep(t)

actuator1 = Actuator(13, 19, 26)
```

Figure 5.2.2.1 Code Snippet(Hardware Declaration)

For RPi Zero W, it only works as the actuator control. So, in this case, an actuator class is created. This is different from DRV8825 on RPi 3B+ because for DRV8825 there's a few inputs that can be shared. For L298N, we create the class so that if there were to require a few different actuators, it can be quickly defined. In the class, the move forward function, move backward function and stop function are also defined.

5.2.2.2 MQTT and Movement

```

def on_connect(client, userdata, flags, rc):
    print("rc: " + str(rc))
    if rc==0:
        print("Connection OK")
    else:
        print("Connection Not OK")

def on_publish(client, userdata, result):
    print("Message Published\n")
    pass

def on_subscribe(client, obj, mid, granted_qos):
    print("Subscribed with QOS: " + str(granted_qos))

def on_message(client, userdata, msg):
    msg.payload = msg.payload.decode("utf-8")
    print("Topic: "+msg.topic+"\n Message:"+str(msg.payload))
    print("Movement called")
    if msg.payload == '1':
        #Movement Function
        actuator1.moveF(100,30)
        c1.publish(topic2,msg.payload)
    elif msg.payload == '2':
        actuator1.moveB(100,30)
        c1.publish(topic2,msg.payload)
    elif msg.payload == '3':
        actuator1.moveF(100,30)
        c1.publish(topic2,msg.payload)
    elif msg.payload == '4':
        actuator1.moveB(100,30)
        c1.publish(topic2,msg.payload)

c1 = mqtt.Client("RpiZero")
c1.on_connect = on_connect
c1.on_publish = on_publish
c1.on_subscribe = on_subscribe
c1.on_message = on_message
c1.connect(broker,port)
c1.subscribe(topic1,0)
c1.loop_start()

```

Figure 5.2.2.2 Code Snippet(MQTT and Movement)

As this part does not require any other functions, we just combine the movement into the MQTT definition. When it receives a message, it will carry out the movement based on the received message. Before publishing a reply.

5.3 Final Assembly

With the software and hardware done, the next step is to combine them into a working prototype. A wooden shell/model as been created to house the motors actuators.



Figure 5.3.1 Final Prototype

From the image above, the top part of the system contains the moving platform, and the lower part is the base that contains motors for rotation and y-axis, as shown in the image below. The Lead Screw is screwed onto the side planks to lock them into place and also provide support.



Figure 5.3.2 Base of Final Prototype

As the motors are risen up into the air, if the two wooden planks aren't there, instead of spinning the lead screw, the motors will be the one spinning. So, to solve this, two wooden planks placed next to the motors that control the y-axis to stop the motors from turning. They also provide some help to the balancing of the system.

Between the two motors for y-axis, is the motor for the rotation, which also carries the entire weight of the model on it. There would be better ways to complete this task, by providing a larger and more stable base, however it was the only way that came into mind at the time.



Figure 5.3.3 Top and moving platform of Final Prototype

For this part of the system, it shows the top, with the moving platform under it. The RPi Zero W and the actuator would be situated under the moving platform. From the image, it is shown a simulation of the platform being slide in or out. To do so, as explained in Chapter 4, a hole is cut out into the moving platform, and a position to hook the platform with the actuator is also made.

5.4 Implementation Issues and Challenges

Creating the final prototype wasn't without its issues and challenges. To start talking about the challenges, we will first bring up the most common error that will usually happen, which is human error. There have been some misunderstandings when purchasing the components and tools necessary. Because of this, the size of the actuator bought for this project was larger than expected and therefore could not be fit onto the prototype. Therefore, in the final product does not contain the actuator on the moving platform, but when demonstrating or explaining, the actuator will be shown and explained in how it would work.

With human error out of the way, we will move to the next challenge faced. Which was lack of tools. As the prototype envisioned was to be in a cylindrical shape made of wood, it was not possible without an actual sawblade. So, the prototype was made as an octagonal shape instead of cylindrical. Although this is an issue, it should be noted that the actual system would actually be in a cylindrical shape instead of octagonal. It is only octagonal for prototyping purposes.

One major challenge faced when testing was with the MQTT communication during motion. As seen from the codes above, we call the movement function only when there is input, however because the function ends with it sending a message to RPi Zero W, it counts the process as complete and will then let users use the system immediately after. Several attempts to solve this was done but to no avail. That was why a complete pop up was created, which will pop up after the entire process is complete. This should be a temporary solve until a better way can be found. Until then, users will have to wait for the complete window pop up if the system was just used or else the processes will crash with each other. It should also be noted that issue will only happen if the second process is retrieved.

Chapter 6

System Evaluation and Discussion

6.1 System Testing and Performance Metrics

For this project, the tests are carried out in between of the implementation stage. By doing so, the end product can be created without too much redesign as the system will be worked upon immediately when there's an error during test phase. Therefore, the test phase has been divided into 5 parts.

Phase 1 – GUI and File Handling (RPi 3B+)	
Description: <ul style="list-style-type: none"> - GUI is designed with all the basic requirements. - File handling is implemented - When user stores a vehicle using the GUI, data will be saved into the file - When user retrieves a vehicle, the data is updated into the file 	
Test Case	Expected Result
Store	<ul style="list-style-type: none"> - Condition of current slot is set to (1) and password is set to input password. - Next available slot (0) becomes current slot (2)
Retrieve	<ul style="list-style-type: none"> - Input slot number and password is checked, only if true with data saved, then will update. - Current Slot condition is set to (0) - Input slot number condition becomes (2), and password is set to 0

Table 6.1.1 Test Phase 1

This is the first test carried out when GUI is done, and file handling is implemented. The purpose is to see if the file handling is working accordingly as data is already stored internally, so the data saved into the file will also be the data used by the system.

Phase 2 – Stepper Motor (RPI 3B+)	
Description: <ul style="list-style-type: none"> - Wiring of RPi 3B+ to DRV8825, and DRV8825 to Stepper motor are set up. - Coding has been written to calculate and carry out the movements. - When user uses the system, system will calculate the optimal path (How many revolutions for the motors). - After calculating, the system will send signals to move the motors accordingly. 	
Test Case	Expected Result
Store	<ul style="list-style-type: none"> - Motors will rotate accordingly to the position of current slot. - Motors will rotate to the position of next available slot. - Motors will move rotate back to the position on the top.
Retrieve	<ul style="list-style-type: none"> - Motors will rotate accordingly to the position of current slot. - Motors will rotate to the position of the input slot. - Motors will move rotate back to the position of the top.

Table 6.1.2 Test Phase 2

In this phase, it is to test out if the motors are working accordingly. As this is only testing the motors, only the rotations can be observed. For example, when Current slot is 01 and user wants to retrieve slot 05, the y-axis motors will rotate 5 counterclockwise revolutions to get to slot 01. The rotation motor will rotate 90 degrees clockwise, followed by another 5 counterclockwise revolutions by y-axis motors to get to slot 05. And lastly, rotation motor will rotate 90 degrees counterclockwise, followed by 10 clockwise revolutions by the y-axis motor to get back to the top. There will be several of these tests carried out to test out several different slots.

Phase 3 – Actuator (RPi Zero W)	
Description:	
<ul style="list-style-type: none"> - Wiring of RPi Zero W to L298N, and L298N to Actuator is set up. - A simple coding is written to test out the movements of the Actuator. - No input as the function is directly coded in to extend or to retract. 	
Test Case	Expected Result
MoveF (Extend)	- Actuator Extends.
MoveB (Retract)	- Actuator Retracts

Table 6.1.3 Test Phase 3

The test here is only to test out the coding and the actuator. So before the actual implementation, there's not much testing since there will only be two motions, extend fully and retract fully.

Phase 4 – MQTT (RPi 3B+ and RPi Zero W)	
Description:	
<ul style="list-style-type: none"> - MQTT is set up on both RPis. - Messages to be published from both sides are defined. - No input or output, only to test out the communication between the two RPis. 	
Test Case	Expected Result
Run the file	<ul style="list-style-type: none"> - RPi 3B+ publishes a message to RPi Zero - After receiving message, RPi Zero gives a reply. - RPi 3B+ publishes another message when it receives the reply from RPi Zero. - Back and Forth communication will carry out 3 more times before ending.

Table 6.1.4 Test Phase 4

This test is to check on the communication between the RPis.

Phase 5 – Final Product	
Description: - Everything from the first 4 phases is combined.	
Test Case	Expected Result
Store	<ul style="list-style-type: none"> - Stepper motor rotates to current position. - RPi 3B+ sends a MQTT message to RPi Zero. - RPi Zero fully extends the actuator, and then sends a reply signal to RPi 3B+. - RPi 3B+ lowers the platform by one y-axis revolution after receiving the reply - RPi 3B+ sends next MQTT message to RPi Zero. - RPi Zero retracts the actuator and sends next reply. - Stepper Motor moves to according to the calculated movement to the next available slot. (Ends with one revolution lower than the actual slot) Sends the third message to RPi Zero. - RPi Zero extends the actuator and sends next reply. - RPi 3B+ rises the platform by one y-axis revolution after receiving the reply - RPi 3B+ sends last MQTT message to RPi Zero. - RPi Zero retracts the actuator and sends last reply. - Stepper Motor then rotates back to the position on top. - Condition of current slot is set to (1) and password is set to input password. - Next available slot (0) becomes current slot (2) - Data is updated to file.
Retrieve	<ul style="list-style-type: none"> - Checks the input slot number and password, if true, continue. - Stepper motor rotates to current slot position. - RPi 3B+ sends a MQTT message to RPi Zero. - RPi Zero fully extends the actuator, and then sends a reply signal to RPi 3B+.

	<ul style="list-style-type: none"> - RPi 3B+ lowers the platform by one y-axis revolution after receiving the reply - RPi 3B+ sends next MQTT message to RPi Zero. - RPi Zero retracts the actuator and sends next reply. - Stepper Motor moves to according to the calculated movement to the Input slot. (Ends with one revolution lower than the actual slot) Sends the third message to RPi Zero. - RPi Zero extends the actuator and sends next reply. - RPi 3B+ rises the platform by one y-axis revolution after receiving the reply - RPi 3B+ sends last MQTT message to RPi Zero. - RPi Zero retracts the actuator and sends last reply. - Stepper Motor then rotates back to the position on top. - Current Slot condition is set to (0) - Input slot number condition becomes (2), and password is set to 0 - Data is updated to file.
--	---

Table 6.1.5 Test Phase 5

For this phase of the test, it can be carried out before putting everything into the model and also after assembling the final prototype.

6.2 Test Setup and Results

As there are 5 different test phases, there's also several test setup and results. Only 3 setups will be done as test phase 3 and test phase 4 does not require any inputs, there's not really anything that could be tested. Test phase 3 and test phase 4 were run as a normal file and the output was as expected. So test setup 3 will just skip to represent test phase 5.

Test Setup 1		
Test Case	Expected Result	Actual Result
User Stores with current position of 03 and password is 193547.	Condition of 03 changes from (2) to (1). Password is saved to 03. Next available slot, 04, condition changes from (0) to (2). Data updated to file.	(Viewed from text file) 03 condition changes from (2) to (1) 03 Password changes from 0 to 193547 04 condition changes from (0) to (2)
User Retrieves from slot 03 with password 185246.	Checks the password for 03, 185246 is not the same as 193547. Pop out Fail Window.	Fail window is popped out.
User Retrieves from slot 03 with password 193547.	Checks the password for 03, matches. Current slot, 04, condition changes from (2) to (0). Condition of 03, changes from (0) to (2). Password is saved to 0. Data updated to file.	(Viewed from text file) 04 condition changes from (2) to (0) 03 Password changes from 193547 to 0 03 condition changes from (1) to (2)

Table 6.2.1 Test Setup 1

Test Setup 2		
Test Case	Expected Result	Actual Result
User Stores with current position of 03 and password is 193547.	<p>Stepper motor moves to position 03. (Rotation 180 degrees clockwise, y-axis 5 revolutions counterclockwise)</p> <p>Stepper motor then moves to position next available slot, 04. (Rotation 90 degrees clockwise)</p> <p>Returns to original position. (Rotation 270 degrees counterclockwise, y-axis 5 revolutions clockwise)</p>	<p>Rotation motor rotate 180 degrees clockwise.</p> <p>Y-axis motors rotate 5 revolutions counterclockwise.</p> <p>Rotation motor rotates 90 degrees clockwise.</p> <p>Rotation motor rotates 270 degrees counterclockwise</p> <p>Y-axis motors rotate 5 revolutions clockwise</p>
User Stores with current position of 04 and password 123456	<p>Stepper motor moves to position 04. (Rotation 270 degrees clockwise, y-axis 5 revolutions counterclockwise)</p> <p>Stepper motor then moves to position next available slot, 05. (Rotation 270 degrees counterclockwise, y-axis 5 revolutions counterclockwise)</p> <p>Returns to original position. (y-axis 10 revolutions clockwise)</p>	<p>Rotation motor rotates 270 degrees clockwise</p> <p>Y-axis motors rotate 5 revolutions counterclockwise</p> <p>Rotation motor rotates 270 degrees counterclockwise</p> <p>Y-axis motors rotate 5 revolutions counterclockwise</p> <p>Y-axis motors rotate 10 revolutions clockwise</p>
User Retrieves slot 20 with password 123123	<p>Checks the password for 20, matches.</p> <p>Stepper motor moves to current slot position, 05. (Y-axis 10 revolutions counterclockwise)</p>	<p>Y-axis motors rotate 10 revolutions counterclockwise</p> <p>Rotation motor rotates 270 degrees clockwise</p>

	<p>Stepper motor then moves to input position, 20. (Rotation 270 degrees clockwise, y-axis 15 revolutions counterclockwise) Returns to original position. (Rotation 270 degrees counterclockwise, y-axis 25 revolutions clockwise)</p>	<p>Y-axis motors rotate 15 revolutions counterclockwise Rotation motor rotates 270 degrees counterclockwise Y-axis motors rotate 25 revolutions clockwise</p>
<p>User Retrieves slot 04 with password 123456</p>	<p>Checks the password for 04, matches. Stepper motor moves to current slot position, 20. (Rotation 270 degrees clockwise, Y-axis 25 revolutions counterclockwise) Stepper motor then moves to input position, 04. (Y-axis 20 revolutions clockwise) Returns to original position. (Rotation 270 degrees counterclockwise, y-axis 5 revolutions clockwise)</p>	<p>Rotation motor rotates 270 degrees clockwise Y-axis motors rotate 25 revolutions counterclockwise Y-axis motors rotate 20 revolutions clockwise Rotation motor rotates 270 degrees counterclockwise Y-axis motors rotate 5 revolutions clockwise</p>

Table 6.2.2 Test Setup 2

Test Setup 3		
Test Case	Expected Result	Actual Result
User Stores with current position of 03 and password is 193547.	<p>Stepper motor moves to position 03. (Rotation 180 degrees clockwise, y-axis 5 revolutions counterclockwise)</p> <p>Sends signal to RPi Zero to extend Actuator.</p> <p>Lower the platform by 1 revolution.</p> <p>Sends signal to RPi Zero to retract Actuator.</p> <p>Stepper motor then moves to position next available slot, 04. (Rotation 90 degrees clockwise)</p> <p>Sends signal to RPi Zero to extend Actuator.</p> <p>Rise the platform by 1 revolution.</p> <p>Sends signal to RPi Zero to retract Actuator</p> <p>Returns to original position. (Rotation 270 degrees counterclockwise, y-axis 5 revolutions clockwise)</p> <p>Condition of 03 changes from (2) to (1).</p> <p>Password is saved to 03.</p> <p>Next available slot, 04, condition changes from (0) to (2).</p> <p>Data updated to file.</p>	<p>Rotation motor rotate 180 degrees clockwise.</p> <p>Y-axis motors rotate 5 revolutions counterclockwise.</p> <p>Actuator extends.</p> <p>Y-axis motors rotate 1 revolution counterclockwise</p> <p>Actuator retracts</p> <p>Rotation motor rotates 90 degrees clockwise.</p> <p>Actuator extends.</p> <p>Y-axis motors rotate 1 revolution clockwise</p> <p>Actuator retracts</p> <p>Rotation motor rotates 270 degrees counterclockwise</p> <p>Y-axis motors rotate 5 revolutions clockwise</p> <p>(Viewed from text file)</p> <p>03 condition changes from (2) to (1)</p> <p>03 Password changes from 0 to 193547</p> <p>04 condition changes from (0) to (2)</p>

<p>User Stores with current position of 04 and password 123456</p>	<p>Stepper motor moves to position 04. (Rotation 270 degrees clockwise, y-axis 5 revolutions counterclockwise)</p> <p>Sends signal to RPi Zero to extend Actuator.</p> <p>Lower the platform by 1 revolution.</p> <p>Sends signal to RPi Zero to retract Actuator.</p> <p>Stepper motor then moves to position next available slot, 05. (Rotation 270 degrees counterclockwise, y-axis 5 revolutions counterclockwise)</p> <p>Sends signal to RPi Zero to extend Actuator.</p> <p>Rise the platform by 1 revolution.</p> <p>Sends signal to RPi Zero to retract Actuator.</p> <p>Returns to original position. (y-axis 10 revolutions clockwise)</p> <p>Condition of 04 changes from (2) to (1).</p> <p>Password is saved to 04.</p> <p>Next available slot, 05, condition changes from (0) to (2).</p> <p>Data updated to file.</p>	<p>Rotation motor rotates 270 degrees clockwise</p> <p>Y-axis motors rotate 5 revolutions counterclockwise</p> <p>Actuator extends.</p> <p>Y-axis motors rotate 1 revolution counterclockwise</p> <p>Actuator retracts</p> <p>Rotation motor rotates 270 degrees counterclockwise</p> <p>Y-axis motors rotate 5 revolutions counterclockwise</p> <p>Actuator extends.</p> <p>Y-axis motors rotate 1 revolution clockwise</p> <p>Actuator retracts</p> <p>Y-axis motors rotate 10 revolutions clockwise</p> <p>(Viewed from text file)</p> <p>04 condition changes from (2) to (1)</p> <p>04 Password changes from 0 to 123456</p> <p>05 condition changes from (0) to (2)</p>
--	--	---

<p>User Retrieves slot 20 with password 123123</p>	<p>Checks the password for 20, matches.</p> <p>Stepper motor moves to current slot position, 05. (Y-axis 10 revolutions counterclockwise)</p> <p>Sends signal to RPi Zero to extend Actuator.</p> <p>Lower the platform by 1 revolution.</p> <p>Sends signal to RPi Zero to retract Actuator.</p> <p>Stepper motor then moves to input position, 20. (Rotation 270 degrees clockwise, y-axis 15 revolutions counterclockwise)</p> <p>Sends signal to RPi Zero to extend Actuator.</p> <p>Rise the platform by 1 revolution.</p> <p>Sends signal to RPi Zero to retract Actuator.</p> <p>Returns to original position. (Rotation 270 degrees counterclockwise, y-axis 25 revolutions clockwise)</p> <p>Condition of 05 changes from (2) to (0).</p> <p>Input slot, 20, condition changes from (1) to (2).</p> <p>Password of 20 set to 0.</p> <p>Data updated to file.</p>	<p>Y-axis motors rotate 10 revolutions counterclockwise</p> <p>Actuator extends.</p> <p>Y-axis motors rotate 1 revolution counterclockwise</p> <p>Actuator retracts</p> <p>Rotation motor rotates 270 degrees clockwise</p> <p>Y-axis motors rotate 15 revolutions counterclockwise</p> <p>Actuator extends.</p> <p>Y-axis motors rotate 1 revolution clockwise</p> <p>Actuator retracts</p> <p>Rotation motor rotates 270 degrees counterclockwise</p> <p>Y-axis motors rotate 25 revolutions clockwise</p> <p>(Viewed from text file)</p> <p>05 condition changes from (2) to (0)</p> <p>20 condition changes from (1) to (2)</p> <p>20 Password changes to 0</p>
<p>User Retrieves slot 04 with password 654321</p>	<p>Checks the password for 04, does not match.</p> <p>Pop out fail window.</p>	<p>Fail window popped up.</p>

<p>User Retrieves slot 04 with password 123456</p>	<p>Checks the password for 04, matches.</p> <p>Stepper motor moves to current slot position, 20. (Rotation 270 degrees clockwise, Y-axis 25 revolutions counterclockwise)</p> <p>Sends signal to RPi Zero to extend Actuator.</p> <p>Lower the platform by 1 revolution.</p> <p>Sends signal to RPi Zero to retract Actuator.</p> <p>Stepper motor then moves to input position, 04. (Y-axis 20 revolutions clockwise)</p> <p>Sends signal to RPi Zero to extend Actuator.</p> <p>Rise the platform by 1 revolution.</p> <p>Sends signal to RPi Zero to retract Actuator.</p> <p>Returns to original position. (Rotation 270 degrees counterclockwise, y-axis 5 revolutions clockwise)</p> <p>Condition of 20 changes from (2) to (0).</p> <p>Input slot, 04, condition changes from (1) to (2).</p> <p>Password of 04 set to 0.</p> <p>Data updated to file.</p>	<p>Rotation motor rotates 270 degrees clockwise</p> <p>Y-axis motors rotate 25 revolutions counterclockwise</p> <p>Actuator extends.</p> <p>Y-axis motors rotate 1 revolution counterclockwise</p> <p>Actuator retracts</p> <p>Y-axis motors rotate 20 revolutions clockwise</p> <p>Actuator extends.</p> <p>Y-axis motors rotate 1 revolution clockwise</p> <p>Actuator retracts</p> <p>Rotation motor rotates 270 degrees counterclockwise</p> <p>Y-axis motors rotate 5 revolutions clockwise</p> <p>(Viewed from text file)</p> <p>20 condition changes from (2) to (0)</p> <p>04 condition changes from (1) to (2)</p> <p>04 Password changes to 0</p>
--	---	---

Table 6.2.3 Test Setup 3

6.3 Objective Evaluation

To recall, the objectives for this project are to design an embedded software for controlling all the mechanical components, to design a graphic user interface for human interaction and to implement a small-scaled model for verification purposes.

From test setup 1 (test phase 1), the objective to design a graphic user interface for human interaction has been achieved. As it was the first step of the system by making a user interface together with the file handling system, it became the core interface that users will interact with. The interface was built to have a clear and direct indication, so users will be able to use it even if its their first time accessing the system.

For test setup 2 (Test phase 2), the objective of creating an embedded software for controlling all the mechanical components have been started. As the test setup was only to test three stepper motors with reliance on the input to get the output, it was quite clear that the system has achieved this. The system was able to get an input of password/ slot number + password and controlled the motors to rotate to certain lengths. Therefore, a part of the mechanical movements has been successfully controlled by the embedded software, leaving only the actuator to be connected, which was done so in the following test setup.

Test setup 3 was a combination of all test phase, as test phases 3 and 4 did not have a specific test setup since there wasn't many different variants of expected results, do be assured that they were tested briefly and have passed the test before test setup 3 was done. With that said, in test setup 3, the objective of designing an embedded software for controlling all mechanical components has been achieved fully since it does implement the control over the actuator as well.

The objective to implement a small-scaled model for verification purposes has also been achieved. This was the test setup that was used before putting everything into the wooden model and also after combining all of them into the prototype. Since the inputs, the outputs would be the same, only that the results on the prototype would be the platform moving based on the rotations of the motors.

After completing the testing. It can be safe to state that all three of the objectives in this system has been completed.

Chapter 7

Conclusion

In conclusion, this project is to apply automation as a solve to some of our daily problems. The purpose of the project is to bring automation into our daily lives and at the same time solve some of the problems faced. The project scope of the project is to provide an alternative parking system that would be more beneficial to the environment and space saving. The problem statement is that normal parking space has brought upon lots of disadvantages all the while wasting space. Although several automation systems have been invented, they are not that suitable to be implemented in public spaces, therefore most of the problems persist, such as traffic congestion and non-environmentally friendly.

This project will be a better solution that is suitable to store cars, opening up more space and also reducing the risk of traffic congestion. As mentioned before, by moving the system underground, it opens so much space above ground. As for the normal underground parking system, the advantage this system has over it is that it's much more safer and there will be less vehicle emissions. Safer in terms of the cars will be below ground, reducing the risk of burglars trying to steal the cars or break in to grab valuables. Besides, with only one exit and a password system, there would only require one surveillance camera for each system to ensure the safety of around 20 vehicles, whereas if 20 vehicles are parked normally in a parking lot, multiple surveillance cameras will be needed.

Although the time and cost needed to implement the system will be more than normal parking lots, the benefit it brings to the world is still undeniable. Therefore, even if we slowly implement the system to public, there will be low improvement and the people will also start to get used to automation. There's still a lot of improvements that can be added to the system, with mechanical doors and led screens to show how much space is left, we believe the system will become the best choice among the competitors.

Recommendation

There is no doubt that this system can still be improved. Just looking back at the report, the two renditions of design already provide different perks, and if we were to combine the two designs and use 3 RPi's, it would definitely be a safer and better system. There are countless improvements that still could be done, using more stabilization and we are just at the beginning of the system.

Taking the idea from before by using 3 RPi's, it reduces so much stress on the main RPi as right now the main RPi has to handle a lot of the work, from GUI to file handling to motor control. All these works can be split better among the RPi used. Having more RPi definitely would help reduce the strain on the processors but since this project was aiming to not use such high cost, it can be said two RPi's would suffice.

Other than this, we can also improve the storing system. For this project we use the internal file system of RPi. But the system could have used a cloud system to store all of this, this way not only the RPi will have the data, but there would also be a record on the cloud system of which vehicle was stored or retrieved at a certain time. Which would provide additional security.

Other than additional improvements, there are also challenges that will be faced when trying to implement the system into the real world. Bringing back the discussion on cost. As the design is built to be used by the public in mind, it could be a challenge for the officials to agree to invest in a project such as this. Although the system can be made to be monetized by requiring the users to pay, it might deter the users from using the system. Therefore, a plan to make it a better deal must be made to actually implement the system into the real world.

Another challenge faced would be the work needed to be put into building the system, as the same with any other constructions. Since the system is an underground system, the work needed to be put into this will be greater than just creating normal parking lots. There would be a need to shut down the part of the road just to dig down and build the system, whereas one would just have to measure and draw the lines to make a parking space beside a road.

REFERENCES

Bibliography

[1]

H. Ibrahim, "Car Parking Problem in Urban Areas, Causes and Solutions," *papers.ssrn.com*, Nov. 25, 2017.
https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3163473 (accessed Mar. 15, 2021).

[2]

"Carousel," *Parkmatic*. <https://www.parkmatic.com/automated/carousel/> (accessed Mar. 15, 2021).

[3]

"Carousel Brochure." Accessed: Mar. 15, 2021. [Online]. Available:
<https://www.parkmatic.com/wp-content/uploads/2020/08/Parkmatic-Flyer-Carousel.pdf>

[4]

T. English, "Volkswagen's Car Towers at Autostadt in Wolfsburg, Germany," *interestingengineering.com*, Apr. 08, 2020.
<https://interestingengineering.com/volkswagens-car-towers-at-autostadt-in-wolfsburg-germany> (accessed Mar. 15, 2021).

[5]

Abhirup Khanna and Rishi Anand, "IoT based Smart Parking System," *ResearchGate*, Jan. 22, 2016.
https://www.researchgate.net/publication/303842610_IoT_based_Smart_Parking_System (accessed Mar. 16, 2021).

[6]

"ECO Cycle™," *GIKEN LTD*. <https://www.giken.com/en/products/automated-parking-facilities/eco-cycle/> (accessed Mar. 16, 2021).

[7]

"Specifications of Eco Cycle," *GIKEN LTD*.
<https://www.giken.com/en/products/automated-parking-facilities/eco-cycle/specifications-of-eco-cycle/> (accessed Mar. 16, 2021).

[8]

"Automated Parking Facility ECO Cycle Brochure." Accessed: Mar. 16, 2021. [Online]. Available: https://www.giken.com/en/wp-content/uploads/developments_ecocycle.pdf

References

[9]

R. P. (Trading) Ltd, "Buy a Raspberry Pi 3 Model B+," *Raspberry Pi*.
<https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/> (accessed Mar. 18, 2021).

[10]

R. P. Ltd, "Raspberry Pi Zero 2 W," *Raspberry Pi*.
<https://www.raspberrypi.com/products/raspberry-pi-zero-2-w/> (accessed Feb. 15, 2022).

[11]

"Pololu - DRV8825 Stepper Motor Driver Carrier, High Current," *www.pololu.com*.
<https://www.pololu.com/product/2133> (accessed Mar. 19, 2021).

[12]

B. Earl, "All About Stepper Motors," *Adafruit Learning System*, May 05, 2014.
<https://learn.adafruit.com/all-about-stepper-motors> (accessed Mar. 19, 2021).

[13]

"Raspberry Pi L298N Interface Tutorial | Control a DC Motor with L298N and Raspberry Pi," *Electronics Hub*, Feb. 09, 2018.
<https://www.electronicshub.org/raspberry-pi-l298n-interface-tutorial-control-dc-motor-l298n-raspberry-pi/> (accessed Feb. 17, 2022).

[14]

"12V Linear Actuator 300mm Stroke," *Cytron Technologies Malaysia*.
<https://my.cytron.io/p-12v-linear-actuator-300mm-stroke> (accessed Feb. 23, 2022).

APPENDIX

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: 3	Study week no.: 1
Student Name & ID: Yong Tze Liang 18ACB02524	
Supervisor: Mr Teoh Shen Khang	
Project Title: Smart Underground Cylindrical Parking System	

1. WORK DONE

The tools completed during FYP 1 was tested to ensure it's working.
 Knowledge was refreshed since there was a few months gap between FYP1 and FYP2

2. WORK TO BE DONE

Start to do research on the build the moving platform
 Work on the coding on RPi 3 B+ while waiting for RPi Zero W to arrive.
 Start some coding for RPi Zero which implements L298N on the RPi 3B+ so that it can be moved to RPi Zero quite quickly.

3. PROBLEMS ENCOUNTERED

None

4. SELF EVALUATION OF THE PROGRESS

At this week, decision of using a second RPi was made and chose RPi Zero W.
 Since there was a few months gap, with a fresh mind, found some flaws in the code written before.
 Not a lot happened as this was more of a decision-making week.



 Supervisor's signature



 Student's signature

APPENDIX**FINAL YEAR PROJECT WEEKLY REPORT***(Project II)*

Trimester, Year: 3	Study week no.: 3
Student Name & ID: Yong Tze Liang 18ACB02524	
Supervisor: Mr Teoh Shen Khang	
Project Title: Smart Underground Cylindrical Parking System	

1. WORK DONE

Done some research on L298N to understand how it works.
Reworked some of the coding on RPi 3B+ since some flaws were found two weeks prior.

2. WORK TO BE DONE

Fully test out the Stepper Motors at campus with power supply.
Try using L298N with one of the stepper motors using RPi 3B+.

3. PROBLEMS ENCOUNTERED

None

4. SELF EVALUATION OF THE PROGRESS

Didn't have much progress as Week 2 was Chinese New Year. Will pick up the pace once get back to campus since there is a power supply to test out motors not. Before they were tested with batteries which were drained quickly.



 Supervisor's signature



 Student's signature

APPENDIX**FINAL YEAR PROJECT WEEKLY REPORT***(Project II)*

Trimester, Year: 3	Study week no.: 5
Student Name & ID: Yong Tze Liang 18ACB02524	
Supervisor: Mr Teoh Shen Khang	
Project Title: Smart Underground Cylindrical Parking System	

1. WORK DONE

Tested out the stepper motors with the file handling fully, with the improved system.
Tried out using L298N with stepper motor.

2. WORK TO BE DONE

Research on the Moving Platform design.
Move some coding from RPi 3B+ to RPi Zero W.

3. PROBLEMS ENCOUNTERED

Testing L298N using stepper motor was faced with failure. Had to do some other research on other tools to use instead of stepper motors.

4. SELF EVALUATION OF THE PROGRESS

As testing of L298N with stepper motor was faced with failure, this is caused by lack of knowledge and understanding. Could try to solve this by doing more research but after thinking about it, decided that maybe stepper motor isn't a good choice for the moving platform. This will hinder a bit of the expected timeline.

Teoh

Supervisor's signature

Yong

Student's signature

APPENDIX**FINAL YEAR PROJECT WEEKLY REPORT***(Project II)*

Trimester, Year: 3	Study week no.: 7
Student Name & ID: Yong Tze Liang 18ACB02524	
Supervisor: Mr Teoh Shen Khang	
Project Title: Smart Underground Cylindrical Parking System	

1. WORK DONE

RPi Zero has arrived and has been set up.
 After researching, decided to use an actuator for the moving platform.
 Started with woodwork to create the model as was waiting for Actuator.

2. WORK TO BE DONE

Build the moving platform.
 Test the communication between two RPis.
 Do testing and documentation.

3. PROBLEMS ENCOUNTERED

Had a little hard time with the woodwork.

4. SELF EVALUATION OF THE PROGRESS

Didn't encounter too many problems. Waiting for the linear actuator to arrive. Woodwork was the most challenging and time-consuming part.

TEOH

Supervisor's signature

yong

Student's signature

APPENDIX**FINAL YEAR PROJECT WEEKLY REPORT***(Project II)*

Trimester, Year: 3	Study week no.: 9
Student Name & ID: Yong Tze Liang 18ACB02524	
Supervisor: Mr Teoh Shen Khang	
Project Title: Smart Underground Cylindrical Parking System	

1. WORK DONE

Worked on the wooden model.
 Actuator arrived and was implemented to RPi Zero.
 Tests were ran to control the movement of actuator and they were a success.

2. WORK TO BE DONE

Start implementing MQTT into both the RPis.
 Test the movement motion of both RPis when connected to MQTT.

3. PROBLEMS ENCOUNTERED

None

4. SELF EVALUATION OF THE PROGRESS

Not much happened, mostly waiting for the components to arrive while refining the report. Did research on the components so that I am familiar with them.



 Supervisor's signature



 Student's signature

APPENDIX

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: 3	Study week no.: 11
Student Name & ID: Yong Tze Liang 18ACB02524	
Supervisor: Mr Teoh Shen Khang	
Project Title: Smart Underground Cylindrical Parking System	

1. WORK DONE

Tested the RPis communication and reworked parts of the code
Continued working on the wooden model.

2. WORK TO BE DONE

Complete the wooden model.
Combine everything into the final prototype.
Test out the complete system.
Work on the Report

3. PROBLEMS ENCOUNTERED

Had a few problems when combining both RPis using MQTT. Most of them solved.

4. SELF EVALUATION OF THE PROGRESS

As mentioned, there were a few problems encountered regarding the MQTT when combined with all of the movement functions. Most of them were solved with some research and some had to be a temporary solve. Wooden model was still hard since there's not a lot of tools to be used. But the final prototype is slowly coming together.



 Supervisor's signature



 Student's signature

APPENDIX**FINAL YEAR PROJECT WEEKLY REPORT***(Project II)*

Trimester, Year: 3	Study week no.: 13
Student Name & ID: Yong Tze Liang 18ACB02524	
Supervisor: Mr Teoh Shen Khang	
Project Title: Smart Underground Cylindrical Parking System	

1. WORK DONE

Completed the final prototype.
 Changed a lot of the report.
 Added some quality-of-life improvements.

2. WORK TO BE DONE

Finalize the report and the Project.
 Present on the Project.

3. PROBLEMS ENCOUNTERED

None

4. SELF EVALUATION OF THE PROGRESS

Marked the end of the FYP, prepared for the presentation next week.

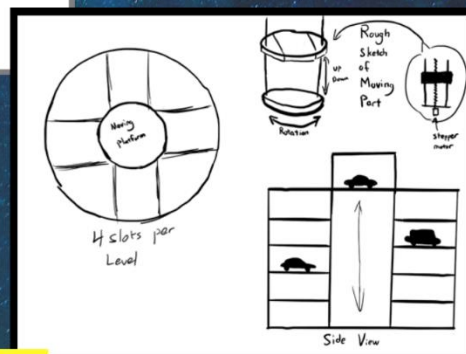
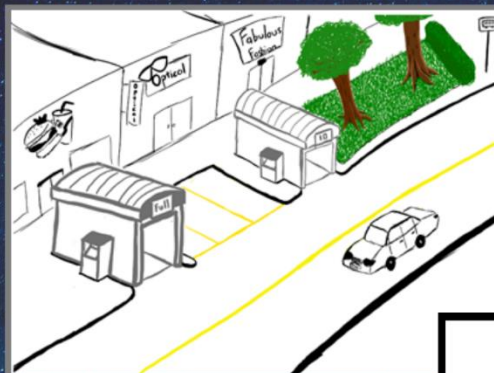
 Supervisor's signature

 Student's signature

Smart underground cylindrical parking system

Yong Tze Liang

This project combines both robotic and IoT to create an automative system that will aim to solve problems brought by traitional parking, such as wasting space, polluting the air and causing traffic congestion. The system aims to become a better alternative that can bring benfits and introduce automation into the daily lives of everyone.



How it works

By using the Raspberry Pi as the barim of the system, users will be presented a choice to store or reetrieve vehicles. By choosing store, users will then set a password, and the system will store the vchile and password. When user wants to retrieve their vehicle, they enter the slot number and password, allowing the system to check the information entered with the data saved, only allowing the user to obtain the vehicle when the passwor is the same. This provides a security measure, letting the users feel safer.

Why Underground?

As the world is always looking to ways to expand, soon we will be bound to be left with no land to expand. When we move the system underground, it opens up such a large area for usage. Other than that, being underground provides more security and benefits the environment more as well.

System Design

The design was made after observing the other inventions. To better fit the use and purpose of the system, it was designed to be a smaller size to be a better fit in the public, also poviding better stability.

PLAGIARISM CHECK RESULT

FYP2 Report Turnitin 1

ORIGINALITY REPORT

5 %	4 %	1 %	4 %
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	eprints.utar.edu.my Internet Source	1 %
2	Submitted to CSU, San Jose State University Student Paper	1 %
3	www.papercamp.com Internet Source	1 %
4	interestingengineering.com Internet Source	<1 %
5	Submitted to Universiti Teknologi Malaysia Student Paper	<1 %
6	Submitted to Universiti Tunku Abdul Rahman Student Paper	<1 %
7	www.coursehero.com Internet Source	<1 %
8	Submitted to University of Wales, Bangor Student Paper	<1 %
9	Submitted to University Of Tasmania Student Paper	<1 %

PLAGIARISM CHECK RESULT

10	www.industrytap.com Internet Source	<1 %
11	Submitted to University of Nottingham Student Paper	<1 %
12	Submitted to The National College for High Speed Rail Student Paper	<1 %
13	Submitted to 7034 Student Paper	<1 %
14	Bengtsson, T., S. Kumar, R.-J. Ubar, A. Jutman, and Z. Peng. "Test methods for crosstalk-induced delay and glitch faults in network-on-chip interconnects implementing asynchronous communication protocols", IET Computers & Digital Techniques, 2008. Publication	<1 %
15	Submitted to University of Sydney Student Paper	<1 %
16	Submitted to American University of the Middle East Student Paper	<1 %
17	www.cs.mun.ca Internet Source	<1 %
18	engineeringprojectsguide.blogspot.com Internet Source	<1 %

PLAGIARISM CHECK RESULT

19	Submitted to Nottingham Trent University Student Paper	<1 %
20	Submitted to University of Birmingham Student Paper	<1 %
21	eduzaurus.com Internet Source	<1 %
22	mbd.ase.ro Internet Source	<1 %
23	Tae Kyoon Kim, Won Sohn, Seok Ho Noh. "The design and implementation of TSBB for KOREASAT D-DBS (digital direct broadcasting satellite) system", IEEE Transactions on Consumer Electronics, 1997 Publication	<1 %
24	www.info.omikk.bme.hu Internet Source	<1 %

Exclude quotes On

Exclude matches < 8 words

Exclude bibliography On

PLAGIARISM CHECK RESULT

Form Title: Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1 of 1



FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	Yong Tze Liang
ID Number(s)	18ACB02524
Programme / Course	Computer Engineering
Title of Final Year Project	Smart Underground Cylindrical Parking System

Similarity	Supervisor's Comments (Compulsory if parameters of originality exceed the limits approved by UTAR)
Overall similarity index: <u>5</u> % Similarity by source Internet Sources: <u>4</u> % Publications: <u>1</u> % Student Papers: <u>4</u> %	
Number of individual sources listed of more than 3% similarity: <u>0</u>	
Parameters of originality required, and limits approved by UTAR are as Follows: (i) Overall similarity index is 20% and below, and (ii) Matching of individual sources listed must be less than 3% each, and (iii) Matching texts in continuous block must not exceed 8 words <i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.</i>	

Note: Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

7E07

 Signature of Supervisor
 Name: Teoh Shen Khang
 Date: 21 April 2022

 Signature of Co-Supervisor
 Name: _____
 Date: _____



UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

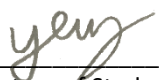
CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	18ACB02524
Student Name	Yong Tze Liang
Supervisor Name	Mr Teoh Shen Kang

TICK (√)	DOCUMENT ITEMS
	Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
	Front Plastic Cover (for hardcopy)
√	Title Page
√	Signed Report Status Declaration Form
√	Signed FYP Thesis Submission Form
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
√	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
√	Appendices (if applicable)
√	Weekly Log
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
√	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

***Include this form (checklist) in the thesis (Bind together as the last page)**

I, the author, have checked and confirmed all the items listed in the table are included in my report.



 (Signature of Student)
 Date: