SMART WHEELCHAIR FOR DISABLED PERSON

BY

NUTT JATURAT 62011177

A PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF BACHELOR OF ENGINEERING IN BIOMEDICAL ENGINEERING KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG ACADEMIC YEAR 2022

SCHOOL OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

PROJECT CERTIFICATE

Project Title	Smart Wheelchair for Disabled Person
Student Name	Mr.Nutt Jaturat
	Student ID. 62011177
Degree	Bachelor of Engineering in Biomedical Engineering

Project Advisor Signed: _____

(Assoc. Prof. Dr. Chuchart Pintavirooj)

Committee Signed: _____

(Assoc. Prof. Dr. Matthew Paul Gleeson)

Committee Signed: _____

(Asst. Prof. Pimkhuan Hannanta-an)

Committee Signed: _____

(Asst. Prof. Dr. Jamie Alexander O'Reilly)

Committee Signed: _____

(Dr. May Phu Paing)

Head of Department Signed: _____

(Assoc. Prof. Dr. Sarinporn Visitsattapongse)

Project Title	Smart Wheelchair for Disabled Person
Student Name	Mr. Nutt Jaturat
Degree	Bachelor of Engineering in Biomedical Engineering
Project Advisor	Assoc. Prof. Dr. Chuchart Pintavirooj
Academic Years	2022

ABSTRACT

Nowadays, people are affected by various diseases related to the body, including those that affect the lower body, which makes it difficult or impossible for patients to walk. Wheelchairs are the answer to this problem, as they are designed for patients who have difficulty walking or cannot walk. The main goal of a wheelchair is to transport patients who cannot walk to their desired destination either by themselves or with the help of a caregiver. The project objective was to develop an effective solution for patients who are unable to operate the joystick on electronic wheelchairs. The smart wheelchair would be beneficial for patients who live in their room. The smart wheelchair will create a map of the room, and the patient can easily navigate it using only one finger to control it by simply clicking on the pre-created map.

This project developed an electronic wheelchair that could be controlled and driven autonomously using a ROS (Robot Operating System) platform. Additionally, it utilized the SLAM (Simultaneous Localization and Mapping) method, which enables the construction of a map of an unknown environment while also simultaneously localizing the wheelchair that is used for navigation purposes. To assess the accuracy of map creation in an unknown environment, the calibration was performed with and without loading weight to assess the distance and angle of the wheelchair's movement.

The results of this project show that the developed electronic wheelchair successfully builds a map of an unknown environment and utilizes this map for navigation. Users can simply click on their desired destination on the map, and the wheelchair will autonomously navigate towards it. Moreover, the wheelchair can obstacle avoidance, while navigation. Although the smart wheelchair can perform

well, there are some limitations due to the RPlidar. The RPlidar detects everything on the ground in parallel when creating the map and while navigating. Therefore, if any object is located below the RPlidar, it may not be able to detect it.

Keyword: Wheelchair, RPlidar, SLAM, ROS, Navigation, Map

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to Assoc. Prof. Dr. Chuchart Pintavirooj, my project advisor, for his invaluable guidance and support throughout this project. It was his kindness, expertise, and willingness to provide constructive feedback that helped me complete this project successfully.

I would also like to extend my appreciation to the Biomedical Engineering Department of the Faculty of Engineering for providing me with the necessary research equipment and facilities, which were instrumental in completing this project.

Lastly, I would like to express my gratitude to all those who supported and assisted me during this project, including those who may not have been mentioned in this thesis. Their contributions have been invaluable to me.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS/ABBREVIATIONS

Symbols /Abbreviations	Term
ALS	Amyotrophic Lateral Sclerosis
BCE	Before the Common Era
IoT	Internet of Things
PC	Personal Computer
ROS	Robot Operating System
SLAM	Simultaneous Localization and Mapping

CHAPTER 1
INTRODUCTION

This chapter opens with an overview of the statement of the problem. The project objectives and scope of the study are then described.

1.1 Background of the study

A wheelchair is a medical device consisting of a chair with wheels attached. Wheelchairs are designed for patients who have trouble walking or are unable to walk. Wheelchairs are available in a variety of types to address the unique needs of each patient, with manual and powered models being the most popular and ubiquitous. Between the sixth and fifth centuries BCE, a wheelchair was engraved on a stone slate and a child's bed was depicted on a frieze on a Greek vase[1-3]. Powered wheelchairs are among the most popular, particularly for individuals with certain motor impairments. An estimated 1% of the global population requires the use of a wheelchair. According to the 2010 census, there are 3.6 million wheelchair users in the United States, while 49% of institutionalized older adults in Canada use a wheelchair [4]. Depending on the sickness or injury of each patient, there are numerous reasons for patients to need wheelchairs. Who would choose the type of wheelchair, the surroundings, and the caregiver. Our primary focus is on patients who are unable to utilize a manual or self-powered wheelchair, but who can control their wheelchair using a laptop or computer in their living room.

Nowadays, there are many wheelchairs available, but not all of them are comfortable for everyone. For instance, some electronic wheelchairs lack the ability to navigate autonomously or avoid obstacles on their own. Addressing these issues remains a challenge in innovation and development. An example solution involves using ceiling lights as landmarks for self-localization of the wheelchair, along with laser technology for obstacle avoidance [5]. In our project for a wheelchair that provides comfort to patients with weak muscles, we aim to develop a wheelchair capable of building maps in the living room or area where the patient intends to use it.

Additionally, the wheelchair should only require an internet connection to construct maps for navigation purposes.

With the rapid development of relevant SLAM signal (Simultaneous Localization and Mapping) technologies in recent years, path planning, recognition, and simultaneous localization have become feasible [6]. From 2010 to 2016, the slam algorithm will be developed to construct location-based on existing maps without the need for online mapping or creating a map beforehand [7].

This proposal outlines the development of a wheelchair system that enables navigation to a desired destination while ensuring obstacle avoidance. The proposed system will have comprehensive control over the wheelchair's functionality and will be capable of connecting to the internet for remote control via a laptop. One of the key features of this system is its ability to construct maps of unknown environments. Navigating the wheelchair is straightforward: by selecting any location on the retained map, the wheelchair will autonomously navigate to the destination while effectively avoiding obstacles encountered during the journey.

1.2 Objective

1.2.1 To create a smart wheelchair that improves the quality of life for those with mobility impairments through wireless technology.

1.2.2 To create an easily controllable automatic wheelchair for patients.

1.2.3 To create a smart wheelchair that can avoid obstacles during navigation.

1.3 Scope of study

1.3.1 Utilize Raspberry Pi hardware to control the overall system, along with RPlidar for navigation and map creation.

1.3.2 The Raspberry Pi will be connected to and controlled by a laboratory laptop during the operation of the system.

1.3.3 Able to create a map of the environment of interest and can control the robot in the map.

1.3.4 Test and calibrate the correct angular and linear motion with and without a load weight.

1.3.5 Wheelchair is capable of autonomous navigation to reach the destination.

CHAPTER 2
REVIEW OF RELATED LITURATURE AND STUDIES

This chapter explains the definition and functionality of wheelchair. The programming techniques utilizing in this project include Linux, Ubuntu, ROS. In addition. The theory of component and some theory of wheelchair will be reviewed in detail to better understand.

2.1 Wheelchair

Wheelchairs are a type of medical equipment that can assist people who have trouble walking or moving their lower body. As is common knowledge, the wheelchair is represented by the chair that has an attached wheel. There are many different types of wheelchairs, but typically some people prefer to use manual wheelchairs in their daily lives. The typical wheelchair rim measures 53 cm. This wheelchair is very easy to maneuver around objects and is very comfortable to use. Some wheelchair types are designed for particular uses, such as wheelchairs used in sports, which each have personal objectives for achieving player demands. The most crucial characteristic is the flexible acceleration from a standstill. Once more, while the standard wheelchair has a 53 cm rim, 32 cm is typically the average for wheelchair racing [8].

The frame is the wheelchair component that has the highest value and has the greatest impact on how well the wheelchair works. The additional part that is attached to the wheelchair's frame is crucial as well. As shown in figure 2, the essential components are the wheels, axis, caster, leg rest, and armrest [8].

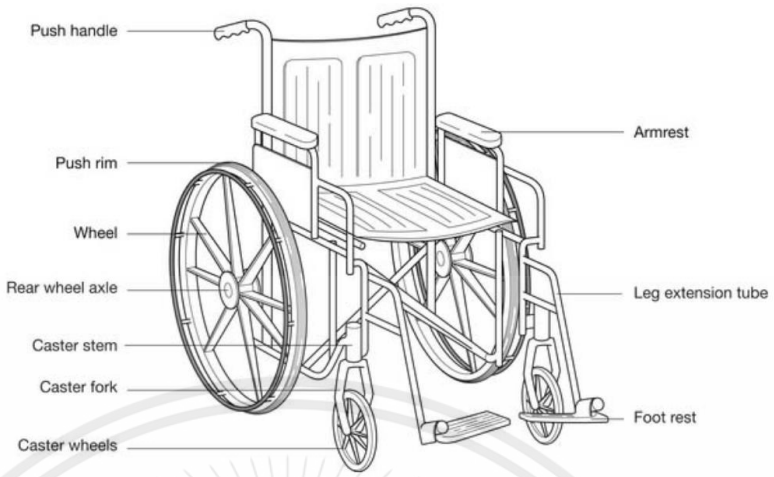

***Figure 1**. Show the component of wheelchair [8]*

The key component of wheelchair is included:

Tires

The wheelchair tires come in two different types: pneumatic and solid rubber. The majority of standard wheelchairs and some lightweight wheelchairs use solid rubber tires. The solid rubber tires are designed for rough terrain and have a high rolling resistance, but they also have low wear rates and require little upkeep. Pneumatic tires are frequently used with ultralight wheelchairs as well as occasionally with lighter wheelchairs. Pneumatic tires have the advantages of a softer ride, reduced rolling resistance, and lighter weight, but they also have high wear rates and expensive maintenance [9].

Wheel

Often, the wheels have spokes or are molded. The size of the wheel depends on the application's goal. The typical command range is between 30 and 66 cm. Low maintenance requirements are a benefit of molded construction, whereas spoked wheels are much heavier and less responsive [9].

Axles

There are two types of rear-wheel axles: fixed and quick-release. Fixed axles are almost always used on standard wheelchairs. Axles can be fixed or quick-release, and quick-release axles are frequently used in ultralight wheelchairs. With fixed axles, a bolt and locknut that must be removed with special tools secures the rear wheel to the frame. A button on the axle's end of the quick-release mechanism makes it simple to remove the tire without using any tools. A wheelchair may need to be disassembled in order to be exported or transported in an automobile. The quick-release axle requires frequency monitoring, whereas the fixed axis requires little maintenance [9].

Caster

The average diameter of a caster ranges from 7.6 to 23.8 cm, with the majority falling between 12.7 and 20.3 cm. The caster wheels can only have mag or solid hub tires, which can be either solid rubber or pneumatic [9].

Leg rests

Leg rests come in a variety of styles, including fixed, swing-away, and elevating. consists of a hanger that attaches to the wheelchair's frame and a footplate that supports the patient's feet. Fixed leg rests are crucial to the frame because they make it lightweight by only having a few parts. Swing-away wheelchairs make wheelchair transfers more comfortable by allowing the regular rests to be removed from the frame. The lower extremities can be positioned at various angles with elevating leg rests. This is beneficial for the physiologic problems [9].

Armrest

There are two types of armrests: fixed-height and adjustable-height. It depends on the wheelchair's objective. When the patient has enough strength to lift his upper body weight with his upper extremity, he can use the armrest to help with the transfer while he isn't pushing the wheelchair [9].

2.2 Force related to wheelchair.

The gravitational force of our planet prevents items from escaping into space on a continuous basis. Therefore, force is the amount of power used to push or pull an item. All objects on the planet are affected by the earth's gravitational pull. If you wish to calculate an object's weight, you must first calculate the force that the earth exerts on it. If an object stays still, the forces are balanced; if they are not, the thing will move. As force is measured in newtons, you must multiply 9.81 to convert from newtons to kilograms [10].

The center of gravity (CG) is crucial when designing a wheelchair. When a patient is in balance, the center of gravity can determine which wheelchair armrest is used. If gravity is a factor, you can control the wheelchair's balance by balancing the patient's weight against that of the wheelchair [10].

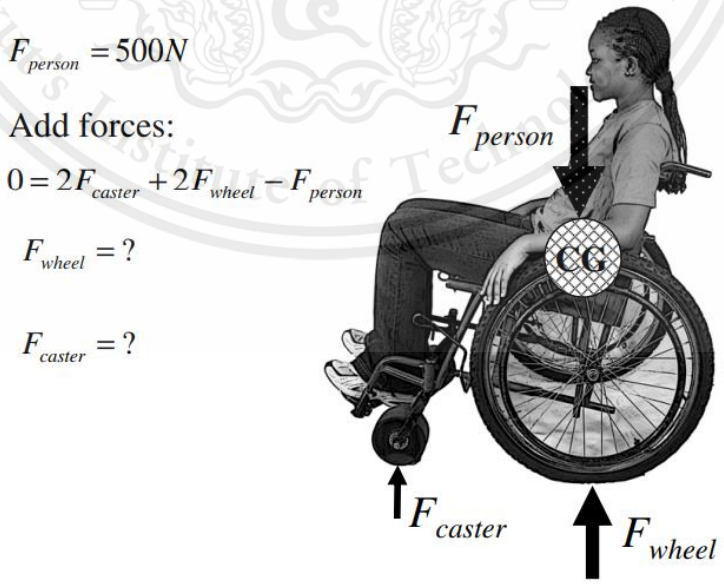

Figure 2. *Free body diagram of patient on wheelchair*[10]

This free body schematic of a patient in a wheelchair is taken from Figure 2. You can determine a patient's weight and the force with which they will push a wheelchair. Since the wheelchair in Figure 2 is not moving, the force that the ground applies to the wheel and casters must be equal [10].

2.3 Electric Wheelchair

The majority of wheelchairs today come in two varieties: manual wheelchairs and electric wheelchairs. Power chairs or auto wheelchairs are other names for electric wheelchairs.

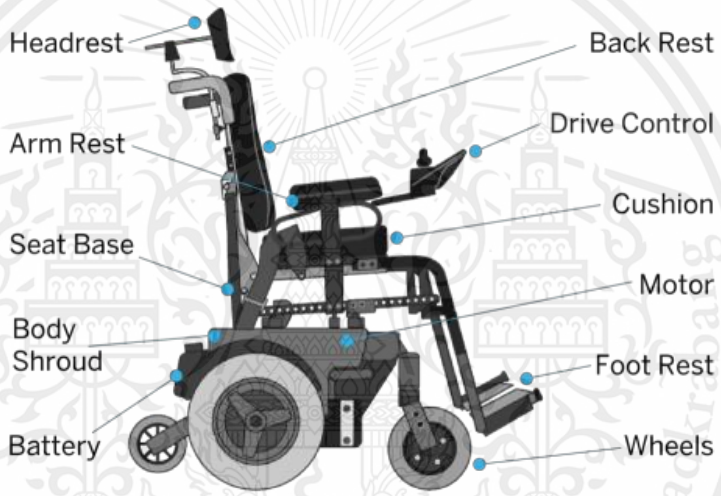

Figure 3. *Component of Electric wheelchair*[11]

Although electric wheelchairs were discovered in the 1900s, demand for them really took off after World War II. Heavy-duty manual folding-frame wheelchairs with a motor, drive belts, and pulleys were used as the basis for the first manufactured electric wheelchairs. The entire system, which is a traditional power wheelchair, was very basic. Joysticks are the controls used to regulate wheelchair movement; they lack programmability. The seating was designed with sling seats and back upholstery because they are very cozy for patients to use. An electric wheelchair has a power base, which includes a motor and battery and is located below the seat. The advantage of this is that the base has high mobility and a seating system that supports good posture. The mechanism used in electric wheelchairs allows for the separation of the wheelchair from the power base. The other electronic system of the wheelchair was also improved while the wheelchair was being converted to a power base wheelchair.

The ability to add a power tilt and recline system and programmable performance settings like forward speed, turning speed, and acceleration are examples of mechanical and electrical advancements [12].

Figure 4. *Joystick of Electrical Wheelchair[13]*

A joystick is a necessary piece of equipment for controlling an electric wheelchair. The joystick shown in Figure 2 controls an electric wheelchair similarly to video game consoles. Electric wheelchair controls have evolved beyond the simple joystick to include usable voluntary movement. For instance, some electric wheelchairs can be moved by moving the head, the breath, the tongue, or the lower extremities [12].

We can divide the drive systems used in electric wheelchairs into two types. The most basic wheelchairs use an indirect first drive that uses pulleys and drive belts. The second is the direct drive system, which power wheelchairs use in conjunction with a gearbox. Modern electric wheelchairs typically employ a power base with a direct system. To be able to afford using the electric wheelchair It takes two 12 Volt batteries. The electric wheelchair should also use other batteries, such as wet cell, gel, or absorbent glass mat types. The electric wheelchair's battery should typically be recharged because changing the battery after each use may not be suitable for the user [12].

The location of the drive wheels allows electric wheelchairs to be categorized into three categories: front-wheel drive, mid- or center-wheel drive, and rear-wheel drive. Rear-wheel drive electric wheelchairs are typically preferred because they are

simple to operate, look similar to manual wheelchairs, and are versatile in daily life. The advantage of center-wheel drive is that it has the greatest degree of maneuverability [12].

2.3.1 Operation of electric wheelchairs

It is a common misconception that the motor is attached to the wheels, powered by a set of batteries, and that the joystick is used to control each component. It starts with the power base to gain a more detailed understanding of the electric wheelchair. Drive wheels make up a power base, which can be positioned in any way depending on the model and the creator's goals. Whether an electric wheelchair has front- or rear-wheel drive depends on where the drive is located. However, front-wheel drive vehicles can turn more easily than rear-wheel drive vehicles [14].

The seating arrangement, which can change based on the brand, model, goal, style, and individual preferences. Wheelchairs are designed for people with limited mobility, and when designing seating, comfort for these people is of utmost importance. The seat is unique to each model and can be altered to suit each person's requirements. Some models may advise them to make things more comfortable for themselves, but they might charge more. The electronic comes next, like the joystick or electronic controller for a motorized wheelchair. The joystick was created with the goal of being easy to use in confined spaces and requiring the least amount of physical effort [14].

2.3.2 Additional Electric Wheelchair Accessories

The patient's comfort and satisfaction can be increased by adding assessors to an electric wheelchair. Electric wheelchairs ought to be made for everyday use, and the right accessories can maximize mobility and independence for each individual [14].

2.4 Phyton

Phyton is a very popular programming language amongst programmers, and it is frequently used to create websites and automate software tasks. Phyton is a general-purpose programming language that can be used to create a variety of programs. This is the benefit of phyton to beginner-programming. Phyton is derived from other

computer programming languages that are simple to use and comprehend. According to RedMonk's analysis, Phython is the second most popular programming language in 2021 [15].

Python is frequently used for constructing websites and applications, automating tasks, performing data analysis, and visualizing data. Python has been adopted by many non-programmers, including accountants and scientists, for a variety of daily chores, such as arranging finances, because it is very simple to learn [15]. Python is an object-oriented, high-level language of programming with dynamic semantics that is interpretable. Because there is no compilation stage, the cycle of edit-test-debug is extremely quick. It is simple to debug Python applications, as a defect or invalid input will never result in a segmentation fault. The Python interpreter and comprehensive standard library are freely accessible for all major platforms in source or binary form [16].y

2.5 Robot operation system (ROS)

ROS (Robot Operating System) navigation is a powerful tool for robot motion planning and control. ROS navigation provides a modular and flexible architecture that allows users to easily build and customize navigation systems for different types of robots. The navigation stack in ROS includes components such as localization, mapping, and path planning, which work together to enable robots to navigate autonomously in complex environments [17].

Although it is feasible to connect ROS with real-time programming, OS is not a real-time framework. The PR2 ether CAT system, used by the Willow Garage PR2 robot, carries ROS signals into and out of a real-time operation. The Orocos Real-time Toolkit and ROS are seamlessly integrated [18].

The goal of ROS framework:

- Thin: Ros was designed to be as lean as possible. Because code produced for ROS can be used with different robot software frameworks, we won't wrap your main(). This explains why integrating ROS with robot software frameworks is simple. OpenRAVE, Orocos, and Player have all been integrated with ROS already [18].

- ROS-agnostic libraries: The desired development approach is to create libraries that are independent of ROS and have clear, useful interfaces [18].
- Language independence: ROS framework are easy to set up in any modern programming language. ROS are already implemented in Phyton, C++, and Lisp, and we have experimental libraries in Java and Lua [18].
- Easy testing: It is simple to set up and take down test fixtures using the ROS unit/integration test framework named rostest [18].
- Scaling: Both huge runtime systems and large development processes are suitable for ROS [18].

In contrast to other platforms with comparable ambitions, ROS' primary purpose is to make code reuse in robotics research and development easier. A distributed process framework called ROS enables runtime loose connectivity and the execution of distinct program designs. These operations are organized into stacks and packages that are simple to divide and distribute. Additionally, ROS enables code aggregation, which speeds up the distribution of labor. All of these features can be integrated with ROS infrastructure tools [18].

2.5.1 Navigation

A conceptually streamlined model of navigation uses velocity commands to relay data from distance measurements made by sensors back to the base. It's difficult to navigate a robot. Navigational needs The robot needs to be ROS-enabled, possess a TF schematic, and post pusher data using the appropriate ROS message type. Additionally, the navigation must be set up to reflect the shape of the robot. High performance is required to aid in this process [19].

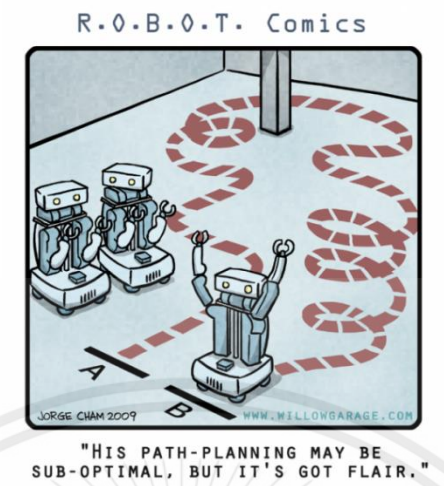

Figure 5. *Example of robot navigation[19]*

The three mains hardware requirement that restrict its

- The mobile base can be controlled by delivering the required velocity directives in the form of: x velocity, y velocity, and theta velocity. It must have both differential and holonomic wheeled robots [19].
- The mobile base must be equipped with a planar laser. This laser is put to work in the mapping and positioning processes [19].
- Since the Navigation Stack was designed for a square robot, its optimal performance is limited to square robots of all sizes. However, huge, rectangular robots may have had trouble navigating the confines of the area [19].

2.5.2 move_base

The move base package offers tools for carrying out actions that give the target in the outside world. A node is used to gain access to a function. By supporting any global planner according to the nav core: Base Global Planner as defined in the package nav core, the move base node links a global and local planner to navigate a task. To carry out navigation tasks, the node move base also stores two costmaps, one for the global planner and the other for a local planner [20].

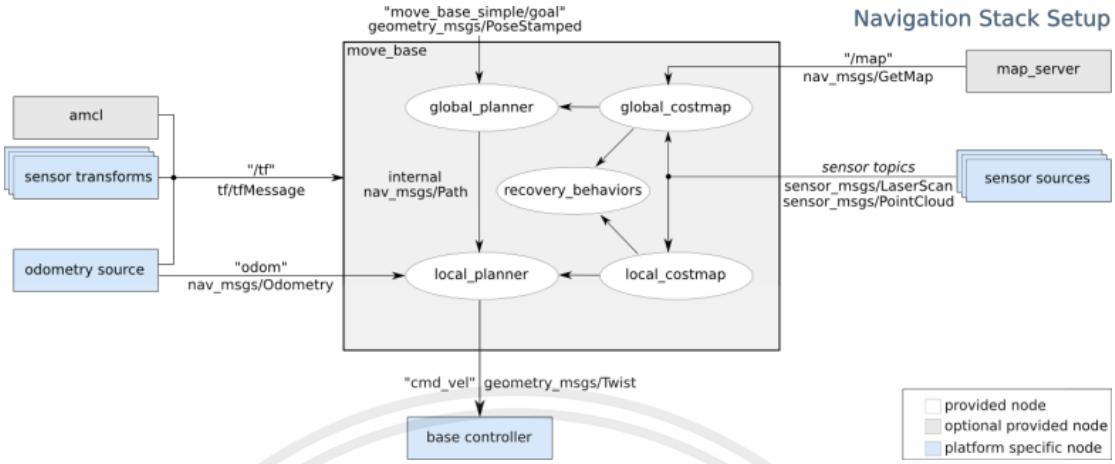

Figure 6. *Show Navigation Stack Setup [20]*

In order to configure, run, and communicate with the navigationstak on a robot, the move base node is preparing a ROS interface. The motion is seen at a high level in Figure 5 and is interacting with other elements. The blure are dependent on the robot platform; the gray nodes are a choice but must be offered for all systems, while white nodes must also be provided [20].

2.5.3 Gmapping

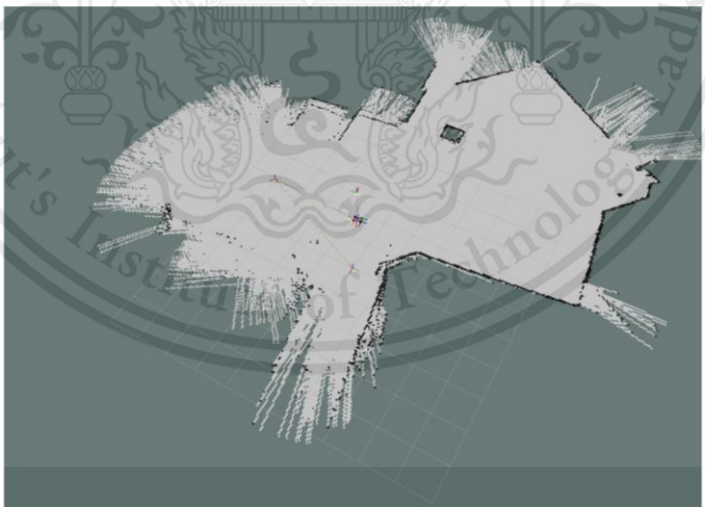

Figure 7. *Show the example of Gmapping*

A ROS wrapper for OpenSlam's Gmapping can be found in this package. Simultaneous Localization and Mapping (SLAM) using lasers is offered by the gmapping package as the ROS node slam gmapping. With the use of slam gmapping,

laser and posture data gathered by a mobile robot can be used to produce a 2 -D occupancy grid map [21].

2.5.4 Teleop

When you press the letter I on your keyboard, Teleop sends the order to the robot's motor, causing it to move forward as seen in Figure 6. Teleop functions like a joystick. As with the arrow, you can control your robot by pressing any other button. You can change the linear speed by pressing W to speed things up or X to slow things down. Additionally, you can vary the angular speed by pressing e to speed up and c to slow down. The greatest component to determine whether your motor will operate properly is the Teleop [22].

```
Reading from the keyboard  and Publishing to Twist!
---------------------------
Moving around:
   u    i    o
   j    k    l
   m    ,    .

q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%
anything else : stop

CTRL-C to quit
```

***Figure 8**. Show the command controlled of Teleop [22]*

2.6 Simultaneous Localization and Mapping (SLAM)

Simultaneous localization and mapping (SLAM) First, we must determine whether it is possible for a mobile robot to be placed in an unknown location and environment while simultaneously creating the most accurate map and determining its location. This is the issue with SLAM. In the preceding ten years, SLAM has been solved. SLAM has been formulated and solved as a theoretical issue in a variety of ways. SLAM are widely used in indoor robots as well as outdoor, underwater, and aerial systems. At the level of theory and concept, SLAM is a solved problem. However, the greatest challenge remains in practically implementing and utilizing perceptually rich maps as part of a SLAM algorithm [23].

For robots to navigate, a map is necessary; similarly, when we need to go somewhere unknown, we also require a map. However, the robot cannot constantly rely on GPS, particularly when operating indoors or outdoors, where the accuracy of GPS is insufficient for safe navigation. Instead, they use SLAM, also known as simultaneous localization and mapping, to discover and map their environment. SLAM, generate their own maps as they travel. To generate the location and environment utilizing sensor data collected from the objects surrounding the robot to construct maps.

2.6.1 Sensor Data Alignment

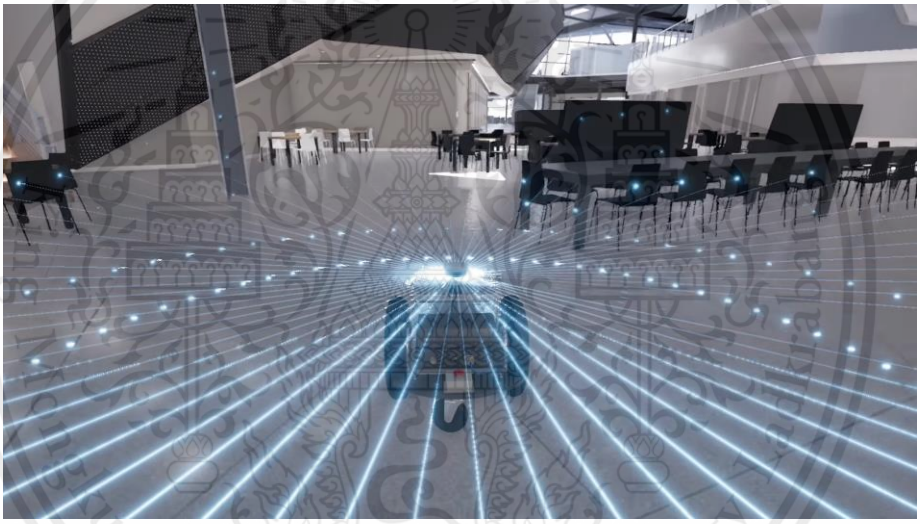

Figure 9. *Show the example of sensor data alignment [24]*

Robots will continuously collect sensor data about their surroundings in split seconds. The camera will correct 90 images per second. And Lidar images, which are utilized for accurate range measurement, are captured 20 times per second [24].

2.6.2 Motion Estimation

It is the best section of a robot's wheels for measuring its distance traveled. Inertial measurement units are also used to measure velocity and acceleration in order to track the position of a robot. In fusion, all sensors are taken into account to produce a more accurate approximation of how a robot is moving [24].

2.7 Linux

Linux is an open-source operating system (OS) that is used to control the hardware and resources of a computer system, including its CPU, memory, and storage. Linux serves as the interface between software and hardware, allowing us to connect all software to the physical resources that support our work [25].

Initially, Linux was developed on a PC using an Intel x86 chipset (32bit). Next, it is developed such that it may be used on all platforms. Because Android has such a large smartphone market, Linux has become a very popular operating system [25].

Linux is like a car engine: it can run on its own, but to be a complete vehicle, it also needs a gearbox, axles, and wheels. If the engine isn't operating properly, other features will also malfunction [25].

2.7.1 Ubuntu

On a PC or virtual private server, you can run Ubuntu, a well-liked Linux-based operating system. The British company Canonical launched Ubuntu in 2004. For convenience and accessibility, Ubuntu is built on the Debian operating system. which are challenging to install and access [26].

The Linux kernel—the main operating system—is the foundation of the Linux operating system family. It makes it possible for hardware and software components to communicate. Ubuntu is a Debian-based Linux distribution. It is appropriate for internet of things (IoT) gadgets, servers, workstations, and cloud computing. The primary distinction between Ubuntu and Linux is that the latter is a Linux distribution, whereas Linux is a family of operating systems built on Unix [26].

2.8 Smart wheelchair's material

2.8.1 Raspberry Pi

Raspberry Pi are comparable to computers, however they differ from them in size. A monitor, a television, a mouse, or a keyboard are just a few examples of input and output hardware that can be utilized with the raspberry pi due to its tiny size and ease of portability. Although the raspberry pi is inexpensive, it may easily transform the setup into a fully functional PC. The Raspberry Pi Zero, Raspberry Pi 1,

Raspberry Pi 2 B, Raspberry Pi3, Raspberry Pi 4B, and Raspberry Pi 400 are the current models that are available. There are numerous Raspberry Pi, and every Raspberry Pi has a unique set of skills and advantages. In this undertaking, we use the Raspberry 4B [27].

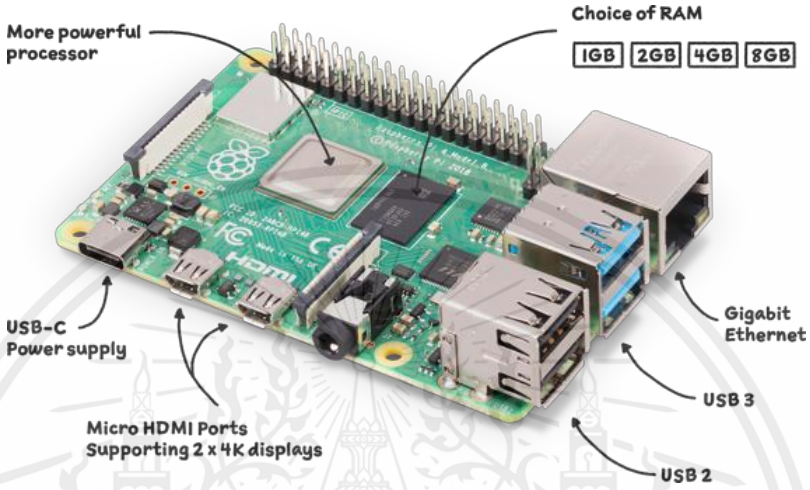

Figure 10. *Raspberry Pi 4B [27]*

The Raspberry Pi 4B was released in 2019 and it is an improvement over its predecessors. It includes a faster 1.5 GHz processor and 2.0 and 3.0 USB ports in addition to memory capacity that has been increased from 2GB to 8GB. With more RAM to please programmers, Raspberry Pi 4 are appropriate for all use cases [27].

2.9.1.1 Features of Raspberry Pi

i. Central Precising Units (CPU)

The Raspberry Pi features a central processing unit, much like any other computer. It serves as the brain of the computer and executes commands utilizing logical and mathematical operations. On its boards, Raspberry Pi uses processors from the ARM11 family [27].

ii. HDMI port

The High-Definition Multimedia Interface (HDMI) connection on the Raspberry Pi gives users the ability to connect to another device and display the visual output. The Raspberry Pi's HDMI cable, which comes in versions 1.3 and 1.4, links the device to an HDTV [27]

iii. Graphic Processing Unit (GPU)

Graphic Processing Unit, or GPU, is another component of the Raspberry Pi board. Its main objective is to increase the calculating speed for images [27].

iv. Memory (RAM)

The key component of a computer's processing system is random access memory. It is the location where current information is kept for easy access. The first Raspberry Pi model has 256MB of RAM. The size was steadily and dramatically improved over time by developers. The capacity of different Raspberry Pi models vary. The Raspberry Pi 4 has 8GB of RAM, making it the model with the highest capacity at the moment [27].

v. Ethernet port

One connectivity hardware component included on the Raspberry Pi B model is the Ethernet connector. Without using any software, the Raspberry Pi's Ethernet connection enables internet connectivity. Additionally, a Raspberry Pi can be directly operated by connecting to a laptop or computer [27].

vi. SD card slot

Every standard computer, as we all know, needs memory to store some form of device. While Raspberry Pi also includes it, it differs from PCs in that it does not include a hard drive or memory card. The Secure Digital card or SD card slot of the Raspberry Pi requires the user to first insert the software or hardware from their computer before using the Raspberry Pi [27].

vii. General Purpose Input and Output (GPIO)

The 40 pins on the Raspberry Pi are used for various applications that we need to add into the device. The Raspberry Pi's GPIO pins are used to connect to other electronic circuits and interact with them. Depending on how the user has programmed them, they can read and manipulate the electric signals from other boards [27].

viii. LEDs

The LED is showing the status of Raspberry Pi

- PWR (Red): This only serves to signal the state of the electricity. Only after the unit is shut off or unplugged from the power source does the red light that indicates it is on go out.
- ACT (Green): This flashes to show any activity on the SD card.
- LNK (Orange): When active Ethernet connectivity has been established, the LNK LED emits an orange glow.
- 100 (Orange): When the data rate reaches 100Mbps while using an Ethernet connection, this light turns on.
- FDX (orange): FDX light also appears when connecting through Ethernet. It demonstrates a full-duplex connection.

ix. USB ports

The Raspberry Pi's primary design concept is its USB ports. They support the keyboard, mouse, etc. There are numerous devices, including mega boards and other sensors, that must be connected to the Raspberry Pi utilizing the USB connection [27]

x. Power source

A 5V micro-USB power cord is generally used with the Raspberry Pi's power source connection. Any Raspberry Pi's electricity usage is influenced by its intended function and the number of connected auxiliary hardware devices [27].

2.9.1.2 Used of Raspberry Pi

Raspberry Pi was developed with education in mind. Raspberry Pi can be used in a variety of ways, including controlling Internet of Things robots and enabling media usage. In this project, the Raspberry Pi was utilized as the computer to drive the motor of an autonomous wheelchair. Using a laptop, we add Ubuntu Mate on the SD card, and the Raspberry Pi runs it from there. connect to a laptop using the Raspberry Pi's IP address to operate it [27].

2.9.1.3 Different versions of raspberry pi

There are different versions of raspberry pi available and their features are as follows in Table 1:

Table 1. Feature of Raspberry Pi [28]

Feature	Raspberry Pi model B+	Raspberry Pi 2 Model B	Raspberry Pi 3 Model B	Raspberry Pi Zero
Soc	BCM2835	BCM2836	BCM2837	BCM2835
CPU	ARM11	Quad Cortex A7	Quad Cortex A53	ARM 11
Operating Freq.	700 MHz	900 MHz	1.2 GHz	1 GHz
RAM	512 MB SDRAM	1 GB SDRAM	1 GB SDRAM	512 MB SDRAM
GPU	259 MHz Videocore IV	250 MHz Videocore IV	400 MHz Videocore IV	250 MHz Videocore IV
Storage	Micro-SD	Micro-SD	Micro-SD	Micro-SD
Ethernet	Yes	Yes	Yes	No
Wireless	Wifi and Bluetooth	No	No	No

2.8.2 Rplidar

Rplidar is a low-cost sensor that may be used for simultaneous localization and mapping by indoor robots (SLAM). Rplidar has the ability to scan an area up to 8 meters in all directions. utilizing the high-speed image processing engine. Additionally, Rplidar can create mapping around itself, which is frequently used in ROS platforms [29].

Figure 11. *RPLIDA2 [29]*

In this project, a Rplidar was set up with a wheelchair to scan the area around it and build a map of the patient's living space. After mapping the patient's living environment, we used the Rplidar to sean of objects to avoid as the wheelchair traveled to its goal.

2.8.3 Encoder

An encoder is a sensor that records feedback. A feedback signal that can be used to calculate location, count, speed, or direction is sent after the encoder converts the motion into an electrical signal. This data can be used by a control device to transmit a command for a specific task [30].

Various types of encoders are used to create signals, but the most common ones are mechanical, magnetic, resistive, and optical. Use optical signal generation frequently. Encoder will give input throughout the optical sensing process via interruption of light [30].

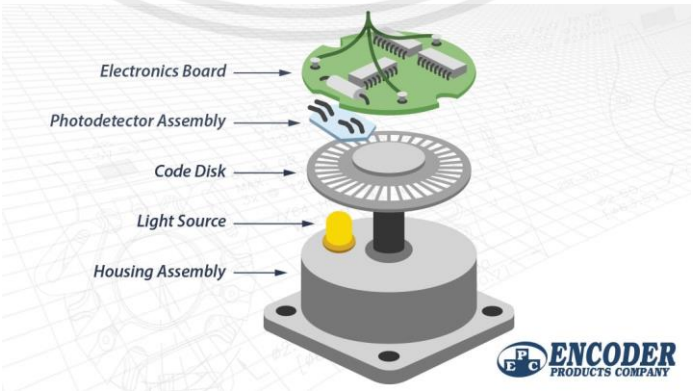

Figure 12. *Outline of Encoder [30]*

The fundamental design of an incremental rotary encoder made with optical technology is shown in Figure 3. The Code Disk, which has an opaque line pattern, is traversed by an LED beam of light (much like the spokes on a bike wheel). The opaque lines on the Code Disk on the encoder shaft cause the light beam from the LED to be broken before being detected by the Photodetector Assembly. This generates a pulse signal: light = on; no light = off. The desired function is produced by the signal once it is conveyed to the counter or controller [30].

2.8.4 MPU 6050

MPU-6050 It is a gadget that tags motion in six axes, is designed to be inexpensive, low-power, and high-performing.

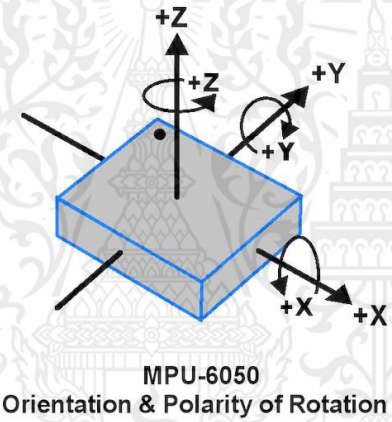

***Figure 13**. Shows axis of MPU-6050 [31]*

Using three accelerometers and a three-axis gyroscope, the MPU-6059 Micro Electro-Mechanical System (MEMS) assists the MPU-6050 in measuring velocity, orientation, acceleration, displacement, and other motion. Additionally, the MPU-6050 includes a DMP, or Digital Motion Processor, which aids with difficult computation resolution. The MPU-6050 will convert 16-bit analog data to digital, which allows it to simultaneously capture three dimensions of motion. Due to the MPU-6050's well-known characteristics and simplicity of usage, microcontrollers like Arduino frequently use it [31].

2.8.5 Motor drive

Motor drive is used to control the motor of the wheel; it will receive instructions from the sending device to change the direction of the wheel as well.

CHAPTER 3
METHODOLOGY

3.1 Introduction

This chapter is describes the methodology in detail, consisting of the overall process of this project, the function to testing. First the chapter describes the working step of each process (section 3.2). Next is the overall of material used in this project consists of hardware. Include et up of Raspberry Pi of the highest value of this project to controlled all of component.

3.2 Action plan

	2022															
	Aug				Sep				Oct				Nov			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Examine the way the intended uses equipment operates as well as related research.																
Setup the ubuntu in Raspberry Pi																
Setup all coding in Raspberry Pi																
Testing RPlidar																
Testing teleop																
Calibration Angular, linear by using teleop																
Get the map and testing Navigation																
Testing by loading weight																
	2023															
	Jan				Feb				Mar				Apr			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4

Adjust the navigation speed to enable it to navigate when carrying a heavier load.															
Improve a specific setup, such as the location of the IMU, RPlidar, etc.															
Improve the wheelchair to achieve better results in navigation.															
Collect the map of interest again and test navigation without load weight and with load weight.															

3.3 Material

In this project consists of two parts; hardware part which is the wheelchair and control system, and software part which is control the wheelchair. The system to avoid the implement when wheelchair going to destination by using laser.

3.3.1 Hardware equipment

- Wheelchair
  - Automatic Wheelchair
  - Rplidar A2
  - Motor
  - Encoder
  - Battery 24V
- Control system
  - Ubuntu 20
  - Linux 20
  - Remote control

3.3.2 Software component

- Raspberry Pi4 8GB RAM
- MPU 6050
- Arduino mega board 2560
- Motor drive 24v

3.4 Design a smart wheelchair

Figure 14. *Show the design of smart wheelchair*

The intelligent wheelchair is designed for the patient, so we will maintain the original condition of the wheelchair, replace the new motor with an encoded motor, and mount the Lidar on the patient's head. The control center of a smart wheelchair is located underneath the patient's seat, as shown in figure 14.

3.5 Design of circuit of control system

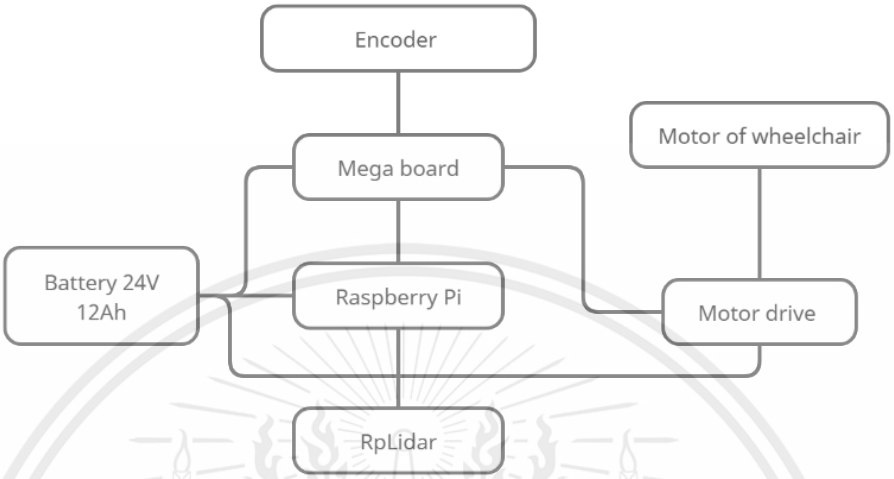

Figure 15. *Circuit of control system*

In the control system, Raspberry Pi serves as the central controller and receives all sensor and laser data. Figure 15 depicts how the battery will be connected to the Raspberry Pi, motor drive, and mega board. Motor drive is attached to the wheelchair's motor to allow for rotation and connection to the mega board. When the wheel rotation encoder receives data, it is sent to the mega board to regulate the wheelchair's rotation. In addition, Raspberry Pi is connected to the Lidar in order to receive data when the Lidar detects the surrounding area around the wheelchair.

3.6 Software Process of control system

3.6.1 Raspberry Pi setup

The Raspberry Pi must be configured, and a fresh SD card must be used to install Ubuntu mate on the SD card. Then, we use Raspberry Pi to read the SD card to Ubuntu mate utilizing mate. When we are able to utilize Ubuntu, we must configure Wi-Fi so that we can operate it remotely by installing ssh-server and nettles. Initially, we used Putty with the right IP address from the Raspberry Pi to configure xrdp via remote control (xrdp setup for we can remote control easily not using putty). For remote control, you must link your Raspberry Pi and your computer to the same Wi-

Fi network. Afterwards, we must install ROS Neotic, Package, Hello robot, HL-340, and Platform.io.

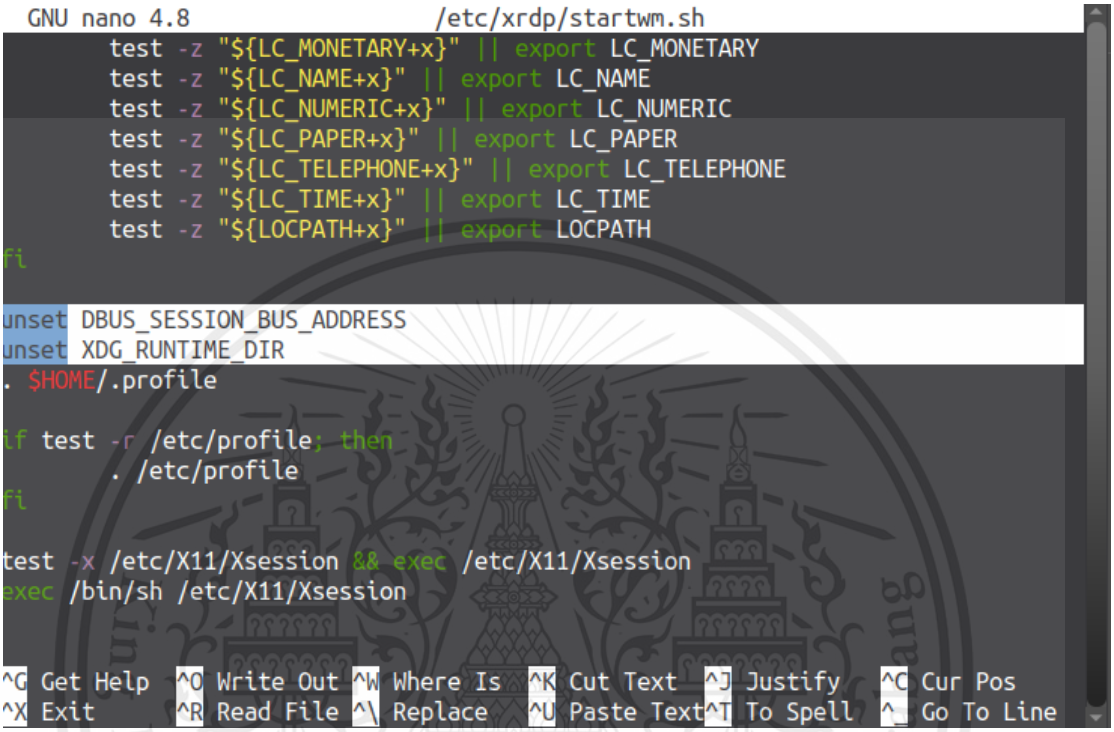

Figure 16. *Setup of XRDP*

3.6.2 Equipment testing

First, you must create a file for the check sensor at encoder, which is known as bringup. When examining the encoder, ensure you have the values of the left and right wheels. The encoder's value will increase when the wheel rotates forward and decrease when it rotates backward. Lidar must detect the entire environment surrounding the wheelchair.

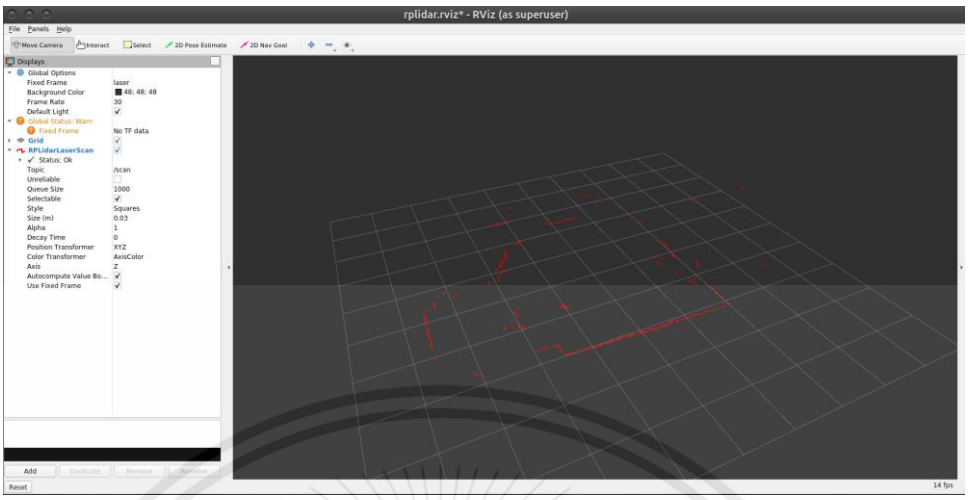

Figure 17. *Lidar testing*

Figure 17 shows that the red line on the space is the object surrounding the Lidar. When Lidar is moved, the shape of the red line will change in real time..

In order to examine the motor, we must first open bringup to see the encoder's value and then open teleop to control the motor. From Figure 18, the encoder terminal is on the left and the teleop terminal is on the right. When we move the teleop ahead by pressing (I) on the terminal, the motor will advance and the encoder will decrease.

Figure 18. *Terminal of encoder and teleop*

Next, configure the size of the wheel by setting the counter per revolution, the diameter of the wheel, and the distance between the left and right wheels. We will measure everything and then configure in linobaseconfig.

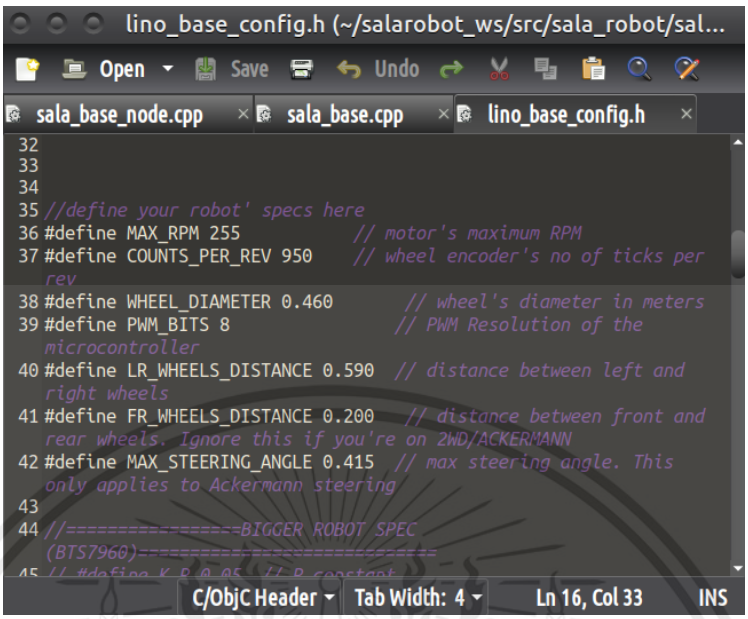

Figure 19. *Files of lino_base_config*

Set the counter per rotation, the diameter of the wheel, and the distance between the left and right wheels before configuring the size of the wheel. Everything will be measured and then configured using linobaseconfig.

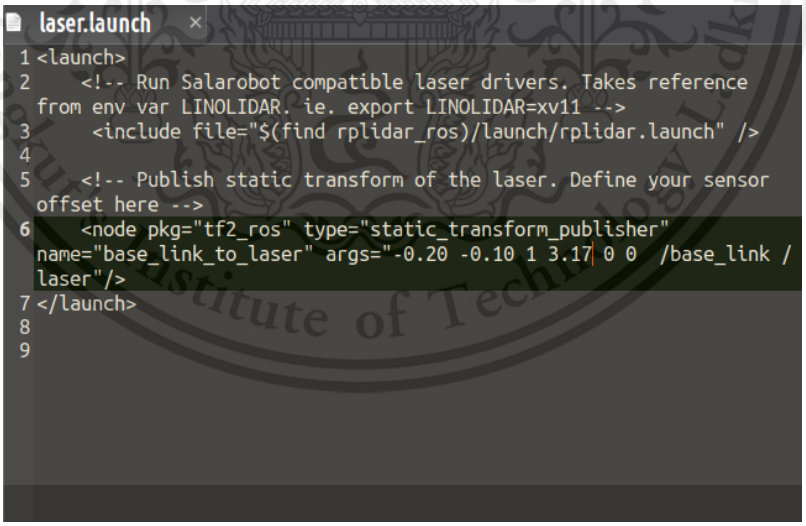

Figure 20. *Files of laser.launch*

Establish imu position as Lidar by measuring from the center of the wheelchair in the imu launch files

```
imu.launch
 8    <!-- Filter and fuse raw imu data -->
 9    <node pkg="imu_filter_madgwick" type="imu_filter_node"
  name="imu_filter_madgwick" output="screen" respawn="false" >
10        <param name="fixed_frame" value="base_footprint" />
11        <param name="use_mag" value="true" />
12        <param name="publish_tf" value="false" />
13        <param name="use_magnetic_field_msg" value="true" />
14        <param name="world_frame" value="enu" />
15        <param name="orientation_stddev" value="0.05" />
16        <param name="angular_scale" value="3.0" />
17    </node>
18
19
20    <!-- Publish static transform from base_footprint to imu_link
  -->
21    <node pkg="tf2_ros" type="static_transform_publisher"
  name="base_footprint_to_imu_link" args="0.04 0 -0.015 0 0 0  /
  base_footprint /imu_link"/>
22 </launch>
Plain Text   Tab Width: 4   Ln 21, Col 118   INS
```

Figure 21. *Files of imu.launch*

3.6.3 Calibration

Before using anything that is important to patients or has an effect on people, we must first test it in this manner. Wheelchair linear calibration and angular calibration are performed. In linear calibration, based on Figure 22's rqt for linear calibration at the top, we can choose how the meter is calibrated and the speed during calibration.

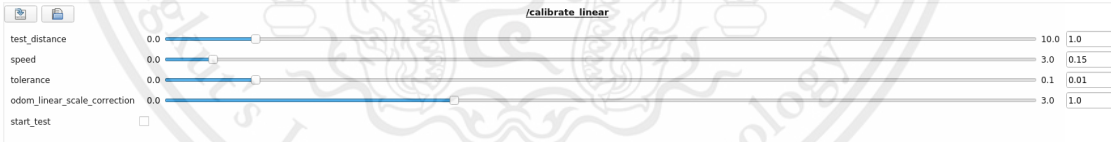

Figure 22. *Show the setting of linear calibration.*

In angular calibration, we set the wheelchair's rotation angle between 1-360 degrees, as depicted in figure 23.

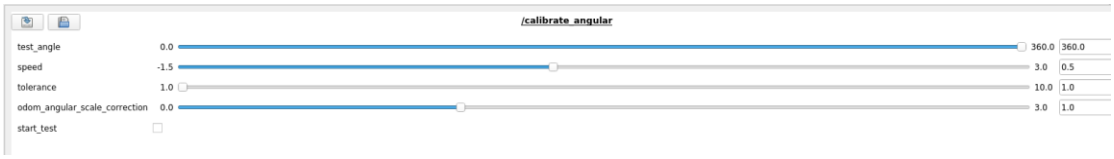

Figure 23. *Show the setting of linear calibration.*

3.6.4 Gmapping

Gmapping is used to generate the map that requires correction using a Lidar encoder and teleop. First, we must bring up and open gmapping. Then, we must use teleop to move a wheelchair around the living room of our target, and as we move the wheelchair, lidar will detect the surrounding surroundings. The mapping will appear as depicted in figure 24.

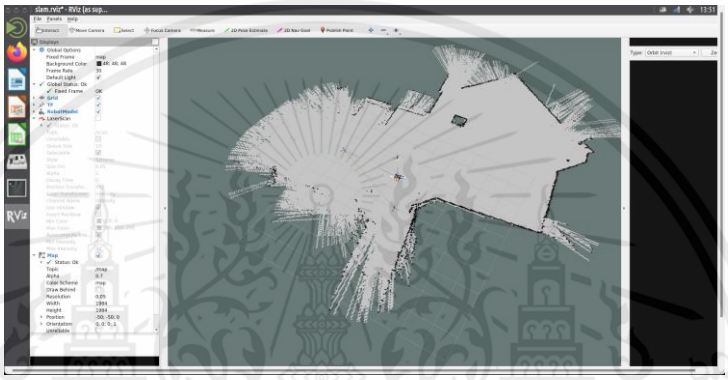

Figure 24. Show the map that we corrected.

3.6.5 Navigation

In the navigation section, we use the Gmapping map, so as shown in Figure 25, you will see the wheelchair model and Lidar will detect the environment in real time. When you need automatic navigation, simply select the destination and click anywhere on the map; the wheelchair will automatically navigate and avoid obstacles along the way.

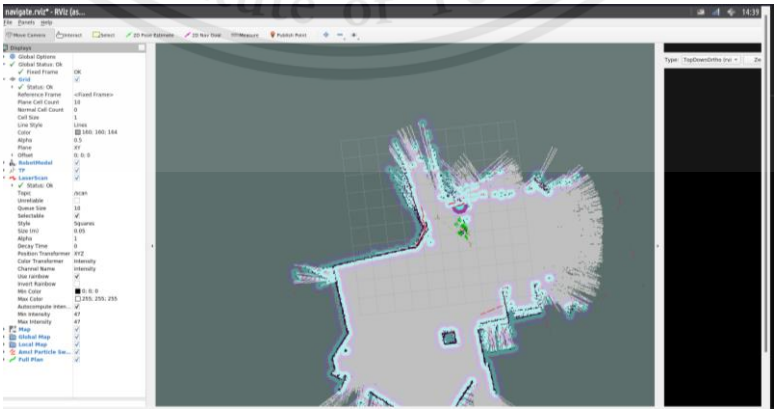

Figure 25. *Show the map when navigation part.*

CHAPTER 4
EXPERIMENTAL RESULTS

4.1 Introduction

The purpose of this project is to build a smart wheelchair for patients who have difficulty walking or are unable to walk to their destination. We are using IoT to control all of our wheelchairs, which we can manage through a Raspberry Pi that controls all of the motors. Our wheelchair needs to be tested for accuracy and consistency. We are using SLAM (Simultaneous Localization and Mapping) to create a map of the patient's living room. The wheelchair must have high accuracy with regards to distance and its sensors (RPLidar) in order to avoid obstacles while navigating.

4.2 Result and control system

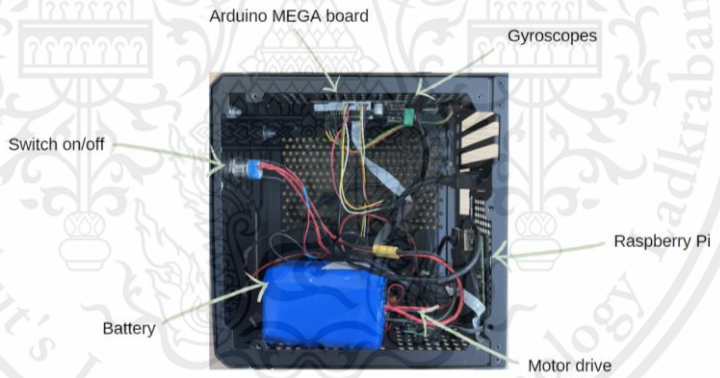

Figure 26. *Circuit box of control system*

The main component we should focus on is the Raspberry Pi, which is the key component of our project. We are using Ubuntu to manage and control the Raspberry Pi, and we have remote control access to our laptop, making it easy to control, change, and add libraries. Next, we connected the other components including the RPlidar, mega board, motor drive, and gyro. Once all the components were connected, we tested the motors first to ensure they were rotating correctly. We used an encoder to collect data on the motor's rotation, which was displayed on the laptop screen (Figure 4.2). After testing the motors, we focused on the RPlidar (Figure 4.3).

Using RVIZ, we could see the red laser of the RPlidar, which detected the environment around it.

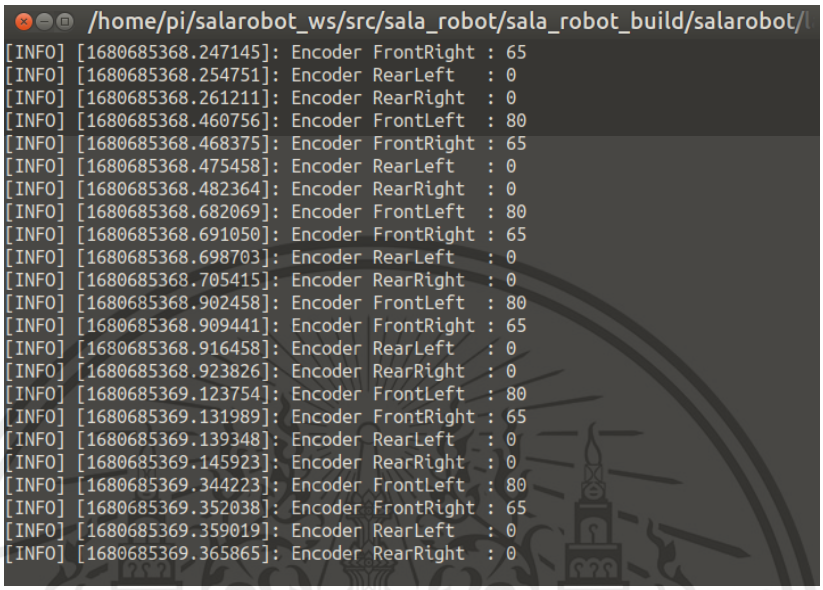

Figure 27. *The amount of value while motor turning.*

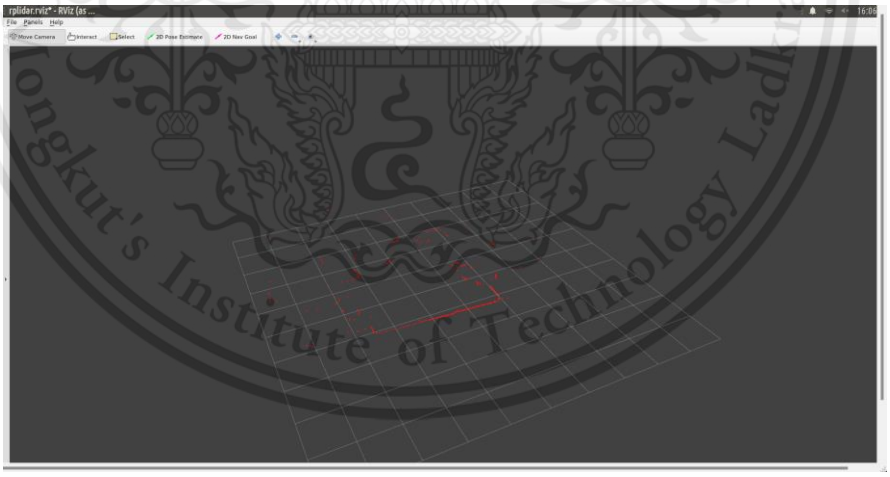

Figure 28. *RVIZ the show the laser of RPlidar that detect environment around the wheelchair.*

Next, we will create a model of the wheelchair that includes the width and height of the wheelchair, as well as the position of the MPU6050 to ensure the correct TF (transform) of the wheelchair. You will see the model of my wheelchair in Figure 4.4.

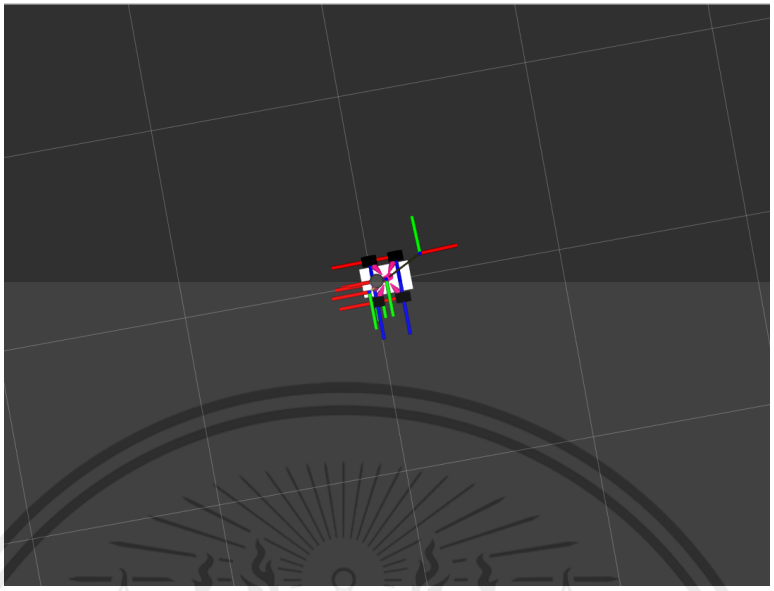

Figure 29. *Model of our wheelchair in rviz*

4.3 Calibration test

So as I said before our wheelchair will need to linear testing and angular testing before we using in collect map. In the (Table 2) will show the result of angular testing and in (Table 3) will show the result of linear testing.

4.3.1 Angular test

In Angular testing, the results from Table 2 are obtained from Figure 4.5 in the testing process. As I mentioned before in Section 3.6.3 on Calibration, we use rqt for calibration.

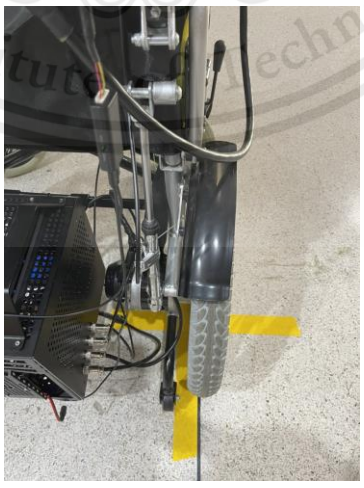

Figure 30*. Notice angular calibration of wheelchair.*

Table 2. Calibration testing of wheelchair in angular

Attempt	45°	90°	180°	360°
1	47°± 5°	92°± 5°	180°± 5°	357°± 5°
2	45°± 5°	97°± 5°	180°± 5°	360°± 5°
3	43°± 5°	87°± 5°	180°± 5°	357°± 5°
4	45°± 5°	90°± 5°	180°± 5°	363°± 5°
5	45°± 5°	92°± 5°	180°± 5°	357°± 5°
6	45°± 5°	92°± 5°	183°± 5°	363°± 5°
7	45°± 5°	90°± 5°	180°± 5°	360°± 5°
8	43°± 5°	87°± 5°	182°± 5°	363°± 5°
9	45°± 5°	90°± 5°	180°± 5°	375°± 5°
10	47°± 5°	90°± 5°	180°± 5°	360°± 5°
SD	1.265	2.722	1.024	5.064
Average	45°	90.7°	180.5°	361.5°

As you will notice from the values in Table 2, the setup should be working well right now. However, certain factors can affect its performance, such as the surface we are using, which has different friction. Therefore, if we change the surface, we will need to set it up again.

4.3.2 Linear test

In linear testing, the results from Table 3 are obtained from Figure 4.6. As mentioned earlier in Section 3.6.3 on calibration, we use rqt for calibration.

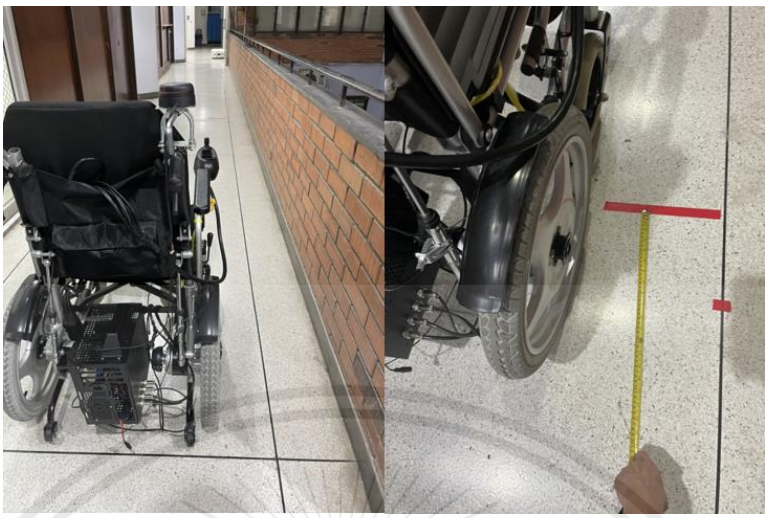

Figure 31. *Notice the linear distance*

Table 3. Calibration testing of wheelchair in linear distance

Attempt	1 m	2 m	5 m	10 m
1	0.99 m	2 m	4.5 m	8.7 m
2	0.98 m	1.95m	4.55m	9 m
3	0.96 m	2 m	4.6 m	8.8 m
4	1.04 m	2 m	4.5 m	9 m
5	1 m	1.95m	4.5 m	8.8 m
6	1 m	2 m	4.5 m	9.1 m
7	1.05 m	2 m	4.55m	9.1 m
8	1,03 m	2 m	4.55 m	8.9 m
9	1.05 m	1.95 m	4.65m	9 m
10	1 m	2 m	4.55 m	9 m
SD	0.029	0.023	0.047	0.128
Average	1.01 m	1.985 m	4.545 m	8.94 m

The result from the test linear calibration from table.3, the result in short distance have much more accuracy than long distance. In long distance.

4.4 Calibration test with loading weight

In this calibration, we load weight onto the wheelchair as shown in Figure 32 to test its accuracy and determine if it can function while carrying weight. We place three large water bottles on the blue section bottle of Figure 32, with a weight of 19 kg, and six kilograms in each of the other bottles in their respective sections.

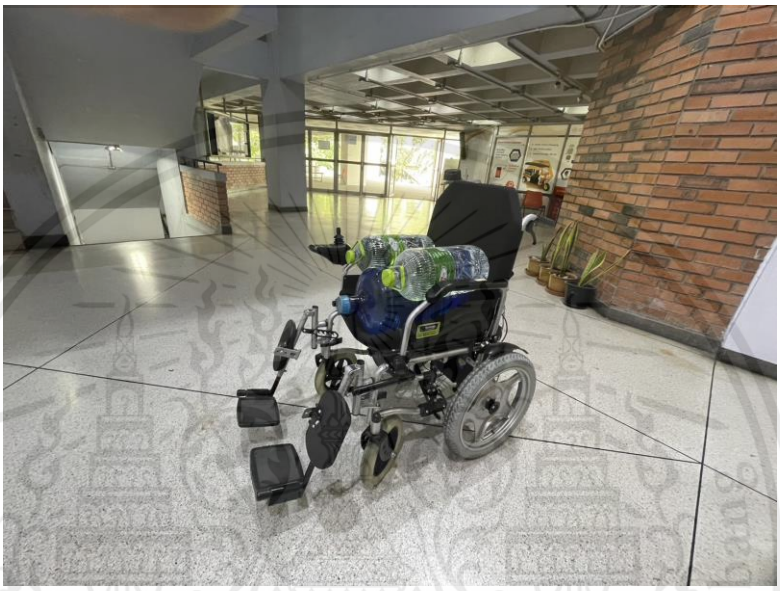

Figure 32. *Wheelchair with loading weight.*

4.4.1 Angular test with loading weight

In this weight-loading test, we follow the same procedure as a normal test, but with the addition of weight. We observe in figure 33 and record the corresponding values in Table 4.

Figure 33. *Notice angular calibration of wheelchair with loading weight.*

Table 4. Calibration testing of wheelchair in angular with loading weight.

Attempt	45°	90°	180°	360°
1	45° ± 5°	90°± 5°	180°± 5°	340°± 5°
2	45°± 5°	93°± 5°	175°± 5°	340°± 5°
3	45°± 5°	93°± 5°	180°± 5°	340°± 5°
4	45°± 5°	93°± 5°	180°± 5°	345°± 5°
5	46°± 5°	90°± 5°	180°± 5°	345°± 5°
6	45°± 5°	93°± 5°	180°± 5°	345°± 5°
7	45°± 5°	90°± 5°	180°± 5°	345°± 5°
8	46°± 5°	90°± 5°	177°± 5°	340°± 5°
9	46°± 5°	90°± 5°	177°± 5°	340°± 5°
10	45°± 5°	90°± 5°	177°± 5°	340°± 5°
SD	0.4°64	1.47	1.8	2.45
Average	45.3°	91.2°	178.6°	342°

4.4.2 Linear test with loading weight

In this test, we load weight into the wheelchair and try to calibrate it linearly to determine if it is working properly. We can observe in figure 34

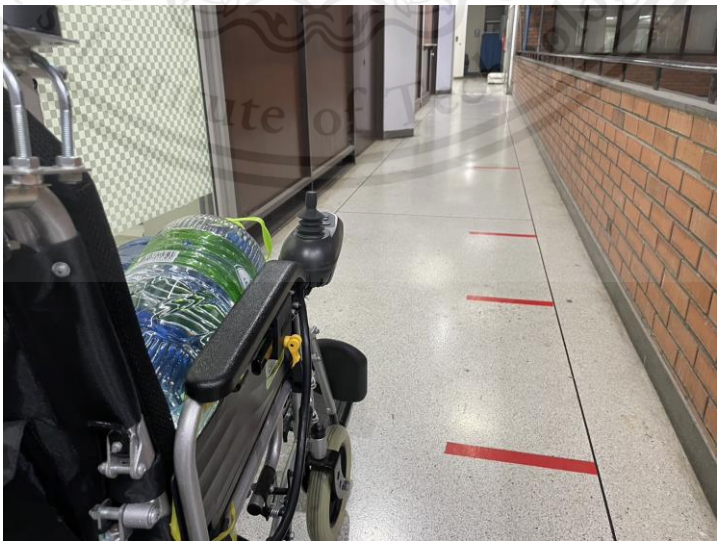

Figure 34. *Notice linear calibration of wheelchair with loading weight.*

Table 5. Calibration testing of wheelchair in linear with loading weight.

Attempt	1 m	2 m	5 m	10 m
1	1.03 m	1.93 m	4.72 m	9.4 m
2	1.07 m	1.92 m	4.7 m	9.54 m
3	1.05 m	1.88 m	4.75 m	9.45 m
4	0.95 m	1.95 m	4.8 m	9.48 m
5	1 m	1.95 m	4.77 m	9.43 m
6	1 m	1.9 m	4.73 m	9.45 m
7	1.03 m	1.92 m	4.73 m	9.47 m
8	0.95 m	1.89 m	4.78 m	9.4 m
9	0.93 m	1.95 m	4.82 m	9.44 m
10	1 m	1.95 m	4.8 m	9.48 m
SD	0.0436	0.0235	0.038	0.04
Average	1.001 m	1.924 m	4.76 m	9.454 m

4.5 Navigate test

In the navigation test, I evaluate the accuracy of the wheelchair's ability to navigate to a destination in Rviz by observing the wheelchiar model. The primary objective of this test is to click on a specific location on our map, after which the wheelchair should automatically navigate to the destination while avoiding obstacles along the way.

4.5.1 Navigate test without loading weight

In this navigation test without weight, we evaluate the wheelchair's ability to navigate to a destination autonomously and avoid obstacles. As shown in Figure 35, the map and wheelchair model are displayed, and the purple arrow indicates the destination we

have set for the wheelchair.

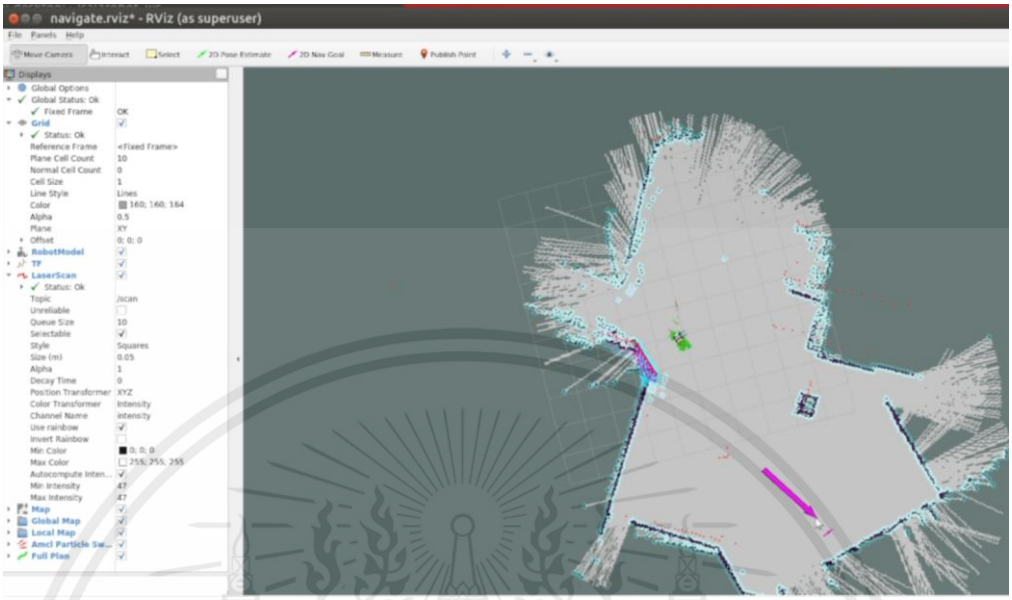

Figure 35. *Show the beginning and destination of navigate in RVIZ.*

After we let the wheelchair reach its destination, you will see the result in figure 36. You will notice that the model of our wheelchair stops at the point we have decided.

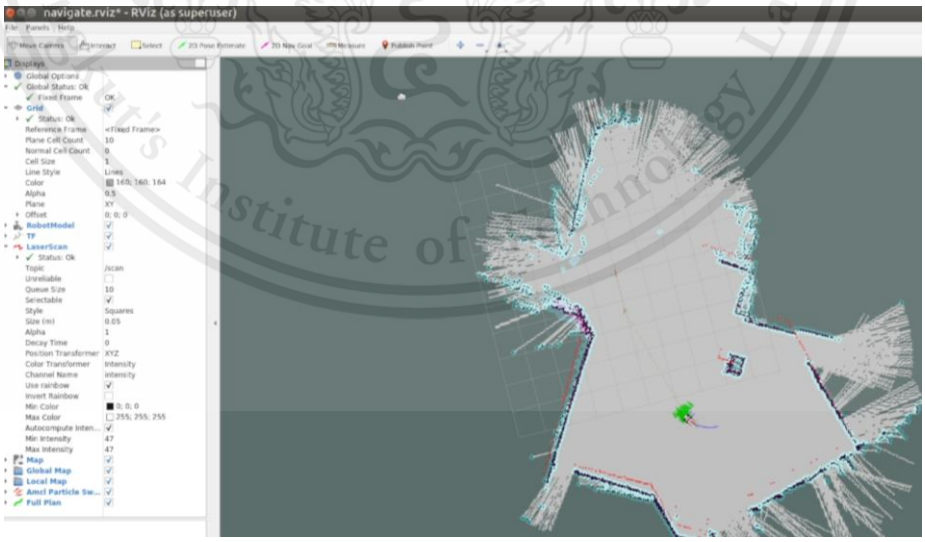

Figure 36. *Show us a model of our wheelchair when we arrive at our destination.*

The next thing we should test is whether our wheelchair will avoid obstacles on its way to the destination or not, as shown in figure 37. The starting point is

indicated by the wheelchiar model, and the destination is where the wheelchair is heading.

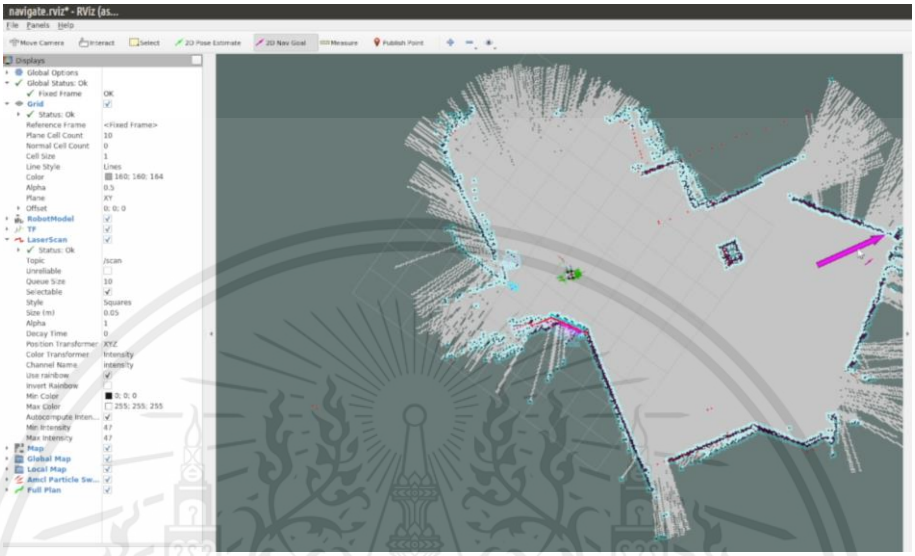

Figure 37. *Show the starting point and destination in RVIZ while testing the implementation of obstacle avoidance.*

Figure 38 shows the pole that our wheelchair avoided before reaching its destination. You can see in the model how our RPlidar detected the pole and caused the wheelchair to move in a curved path to avoid it.

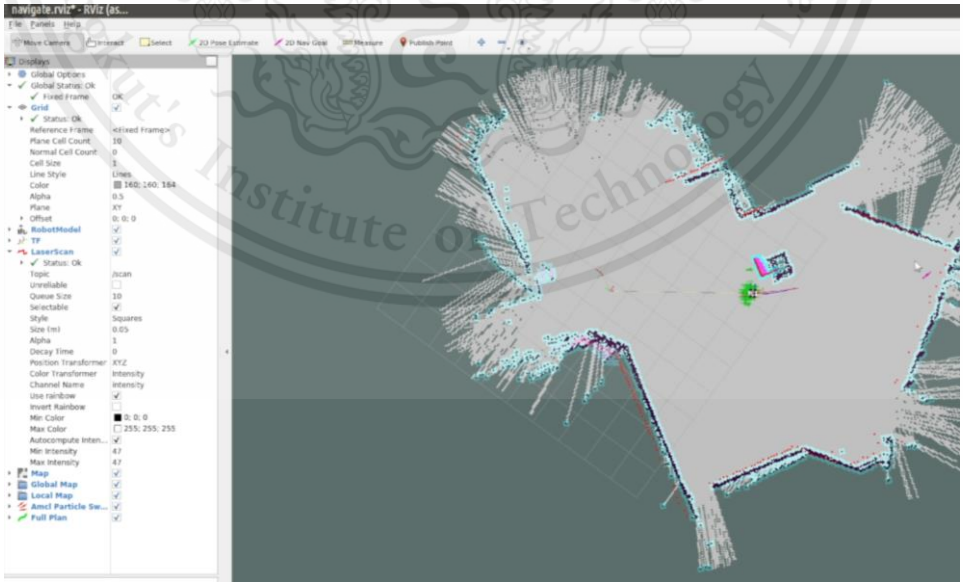

Figure 38. *Show the model of our wheelchair detecting a pole and avoiding it.*

So, from this figure 39, you can see the destination that the wheelchiar avoided before arriving here.

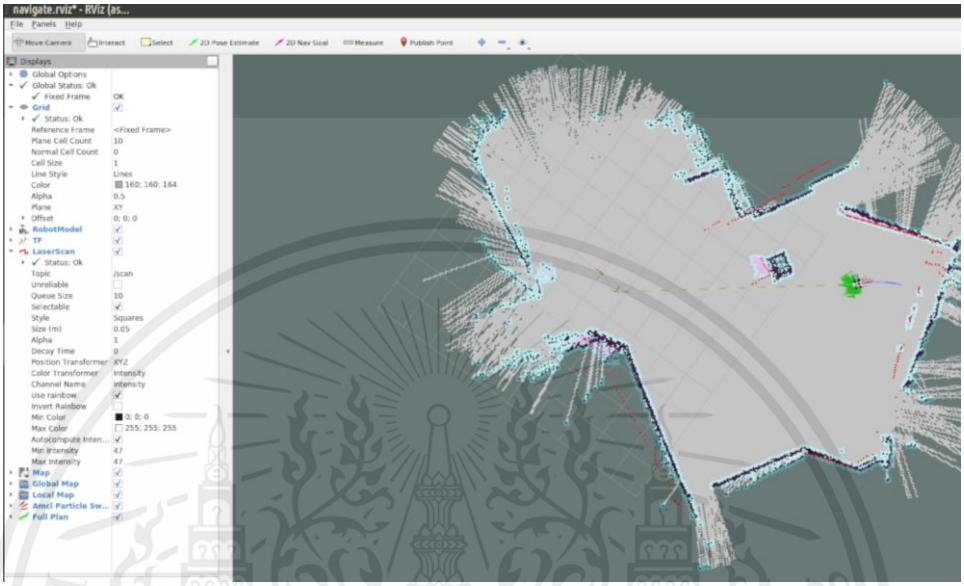

***Figure 39**. Show us a model of our wheelchair when we arrive at our destination.*

Figure 36-39 depicts the picture's border, which consists of a perfect and small line projecting outward from the edge. The perfect map was constructed from a solid wall, whereas the line projecting outward was constructed from a mirror wall.

4.5.2 Navigate test with loading weight

In this navigation test with a loading weight of 40, our wheelchair in the figure represents the destination, and the wheelchair model represents the starting point, indicated by my mouse arrow.

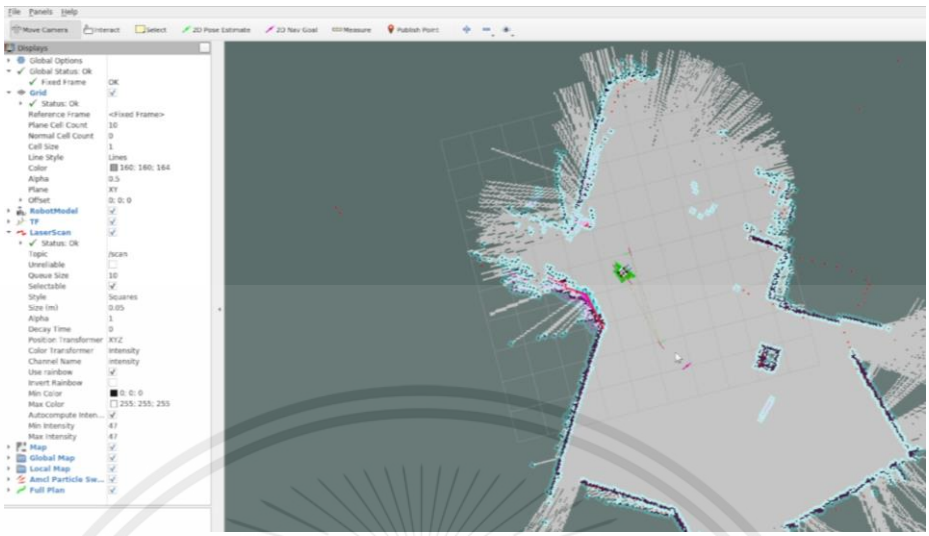

Figure 40. *Show the starting point and destination in RVIZ with loading weight.*

In Figure 41, the wheelchair has arrived at its destination, but it navigated much slower than without a loading weight. Nevertheless, it was still able to reach the destination.

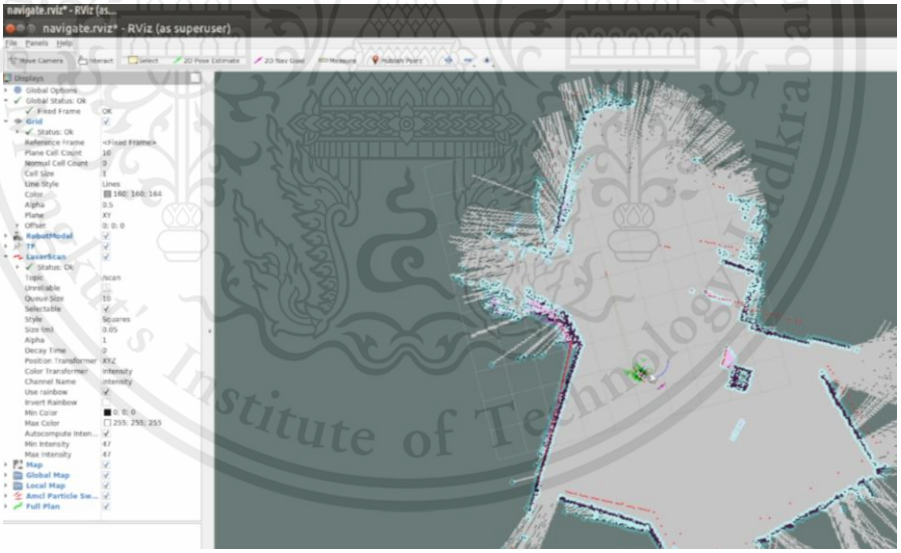

Figure 41. *Show us a model of our wheelchair when we arrive at our destination with loading weight.*

During the final stage of navigation, the implementation to avoid obstacles was still ineffective due to the added weight. In RVIZ, our wheelchair attempted to avoid an obstacle, but it moved slowly and took a long time to avoid it, resulting in the wheelchair crashing into a pole before it could successfully avoid the obstacle.

CHAPTER 5
CONCLUSION

5.1 Introduction

In this chapter, we will discuss and summarize the work completed in each section from Chapter 4, including the design, control setup, and navigation of the wheelchair. This will provide a better understanding of the development of the smart wheelchair.

5.2 Discussion

The key to this project is to develop a wheelchair for disabled persons who have weak muscles and have difficulty controlling other organs, but can still move a finger to control our wheelchair via a touchpad on a notebook. The main goal that we have planned is that our wheelchair must be able to create a living room map for the patient and navigate to the destination by itself, while also avoiding obstacles.

The biggest problem in our project is the software setup because it needs to load numerous libraries and test every device. During the calibration step, we fix the accuracy of linear and angular measurements for short distances. However, it does not have any effect on long distances, so we have trained it multiple times. In our wheelchiar, when we create the map by using RPlidar, map that we create are look similar to the real map that we interest. But when we navigate using our RPlidar, which is used to detect the environment around the wheelchair, it sometimes detects incorrectly and less accurately. We have tried to check and change it many times to achieve the highest accuracy.

We have also faced some problems with the motor and battery. In the manual, we use teleop to control or calibrate the wheelchair, and it doesn't have any problem because we can change and fix it quickly. However, initially, we faced problems with the navigation as the wheelchair did not move. We set it up many times, and each time we tested it, it took a long time because we needed to set up many things. If it was set too high, the wheelchair would not move, and if it was set too low, the wheelchair would not move either. We had to test it until it worked properly.

Furthermore, we use remote monitoring to connect to Raspberry Pi for control, but the problem we encountered was that we needed to use the same internet. During

the navigation test, when my wheelchair moved far from the area of the internet, the connection between the laptop and Raspberry Pi was lost. Moreover, when we needed to change the area of usage for the smart wheelchair, we had to connect manually with Raspberry Pi and obtain the IP address before using it to connect. During this process, we also had to remove every device connected to the control box.

Lastly, another issue we faced was that Raspberry Pi can only connect to Wi-Fi or hotspot that comes from only an Android device.

5.3 Conclusion

This project has achieved its primary design objectives: (1) creating a smart wheelchair using IoT, (2) developing an automatic wheelchair that is easy for patients to control, and (3) developing a wheelchair that can avoid obstacles while navigating. Our project utilized ROS and SLAM, which are primary functions of ROS that provide the necessary tools and libraries for navigation. While ROS and SLAM are typically used in small robots, we employed them in our wheelchair to test their performance. Our project is important because our wheelchair can create maps and avoid obstacles while navigating, which is a success. As a project in the Department of Biomedical Engineering, we aimed to create a medical device that would improve the lives of patients, particularly those with ALS (Amyotrophic Lateral Sclerosis) who have difficulty controlling wheelchairs due to weak muscles. In future research, we aim to improve the accuracy of our wheelchair while navigating, as using only RPlidar may result in detection errors for objects below the laser. Additionally, we plan to improve the map creation process for better navigation accuracy.

However, there is still a need for more studies to further improve the smart wheelchair for disabled individuals. With more research into developing wheelchair it may be possible to turn this technology into a commercially viable product in the future. The smart wheelchair can also be beneficial for patients with other conditions who are unable to control a traditional automatic wheelchair. If the wheelchair is fully successful and has the highest accuracy when working with patients, it can greatly improve the quality of life for these individuals.

REFERENCES

[1] F. books. "Who Invented the Wheelchair?" https://www.mentalfloss.com/article/20768/who-invented-wheelchair (accessed 11 Oct 2022).

[2] J. Flaherty. "Putting the 'Whee!' Back in 'Wheelchair'." https://www.wired.com/2012/05/wheelchair/ (accessed 11 Oct 2022).

[3] Wheelchair-information. "History of Wheelchairs." http://www.wheelchair-information.com/history-of-wheelchairs.html (accessed 11 Oct 2022).

[4] M. Kutbi, "An Egocentric Computer Vision-Based Robotic Wheelchair," Ph.D., Stevens Institute of Technology, Ann Arbor, 2018.

[5] H. Wang, C.-U. Kang, T. Ishimatsu, and T. Ochiai, "Auto-navigation of a wheelchair," *Artificial Life and Robotics,* vol. 1, pp. 141-146, 1997.

[6] J. Sun, X. Yu, X. Cao, X. Kong, P. Gao, and H. Luo, "SLAM Based Indoor Autonomous Navigation System For Electric Wheelchair," in *2022 7th International Conference on Automation, Control and Robotics Engineering (CACRE)*, 2022: IEEE, pp. 269-274.

[7] N. Ragot, R. Khemmar, A. Pokala, R. Rossi, and J.-Y. Ertaud, "Benchmark of visual slam algorithms: Orb-slam2 vs rtab-map," in *2019 Eighth International Conference on Emerging Security Technologies (EST)*, 2019: IEEE, pp. 1-6.

[8] Y. Vanlandewijck, D. Theisen, and D. Daly, "Wheelchair propulsion biomechanics," *Sports medicine,* vol. 31, no. 5, pp. 339-367, 2001.

[9] C. P. DiGiovine. "wheelchair." https://www.britannica.com/topic/wheelchair (accessed 25 Oct, 2022).

[10] R. H. Amos Winter, "Mechanical Principles of Wheelchair Design," D. o. M. E. M. I. o. Technology Ed.

[11] U. S. D. O. TRANSPORTATION. "Basic Components." https://airconsumer.dot.gov/guide/mod2/WheelchairPBC.html (accessed 26 Oct, 2022).

[12] C. P. DiGiovine. "electric wheelchair." (accessed 26 Oct, 2022).

[13] Wsrsloution.com. "Joystick Repairs on Power Wheelchairs." https://wsrsolutions.com/joystick-repairs-on-power-wheelchairs/ (accessed 26 Oct, 2022).

[14] Hoveround. "Electric Wheelchair Parts: 5 Basic Components." https://www.hoveround.com/articles/electric-wheelchair-parts (accessed 27 Oct 2022).

[15] Coursera. "What Is Python Used For? A Beginner's Guide." https://www.coursera.org/articles/what-is-python-used-for-a-beginners-guide-to-using-python (accessed 27 Oct, 2022).

[16] Phyton. "What is Python? Executive Summary." https://www.python.org/doc/essays/blurb/ (accessed 27 Oct 2022).

[17] S. Zhao and S.-H. Hwang, "ROS-Based Autonomous Navigation Robot Platform with Stepping Motor," *Sensors,* vol. 23, no. 7, p. 3648, 2023. [Online]. Available: https://www.mdpi.com/1424-8220/23/7/3648.

[18] AmandaDattalo. "ROS Introduction." http://wiki.ros.org/ROS/Introduction (accessed 28 Oct, 2022).

[19] FabriceLarribe. "navigation." http://wiki.ros.org/navigation?distro=noetic (accessed 29 Oct, 2022).

[20] Nickfragale. "move_base." http://wiki.ros.org/move_base?distro=noetic (accessed 29 Oct, 2022).

[21] GvdHoorn. "gmapping." http://wiki.ros.org/gmapping (accessed 29 Oct, 2022).

[22] MikePurvis. "teleop_twist_keyboard." http://wiki.ros.org/teleop_twist_keyboard (accessed 29 Oct, 2022).

[23] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part I," *IEEE robotics & automation magazine,* vol. 13, no. 2, pp. 99-110, 2006.

[24] S. MARTIN. "What Is Simultaneous Localization and Mapping?" https://blogs.nvidia.com/blog/2019/07/25/what-is-simultaneous-localization-and-mapping-nvidia-jetson-isaac-sdk/ (accessed 29 Oct, 2022).

[25] Saixiii. "What is linux." https://saixiii.com/what-is-linux/ (accessed 29 Oct, 2022).

[26] N. G. "What Is Ubuntu? A Quick Beginner's Guide." https://www.hostinger.com/tutorials/what-is-ubuntu#What_Is_Ubuntu (accessed 29 Oct 2022).

[27] C. BasuMallick. "What Is Raspberry Pi? Models, Features, and Uses." https://www.spiceworks.com/tech/networking/articles/what-is-raspberry-pi/ (accessed 30 Oct, 2022).

[28] Electronicwing. "Raspberry Pi Introduction." https://www.electronicwings.com/raspberry-pi/raspberry-pi-introduction (accessed 30 Oct, 2022).

[29] DFRobot. "RPLIDAR Tutorial Review." https://www.dfrobot.com/blog-1464.html (accessed 30 Oct, 2022).

[30] Encoder. "What Is an Encoder?" https://www.encoder.com/article-what-is-an-encoder (accessed 30 Oct, 2022).

[31] CiferTech. "What Is MPU6050?" https://create.arduino.cc/projecthub/CiferTech/what-is-mpu6050-b3b178 (accessed 30 Oct, 2022).