

MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE
SUMY STATE UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

Master Work

On the topic:

“Smart Robot Control Information Technology”

Head of Department

Student Group IH.M-05аН

Supervisor

Dovbysh A.S.

Uba B.V.

Oleksiienko G.A.

SUMY 2021

MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE
SUMY STATE UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

Approved _____

Head of department Dovbysh A.S.

“ ____ ” _____ 2021

TASK To Master Project

Student of Master's Level, group IH.M-05аН specialty "Informatics" Uba
 Bartholomew Vincent

Topic “Smart Robot Control Information Technology”

Approved by order of the Sumy State University

№ _____ of _____ 2021

Contents Explanatory Note: 1) Overview of projects for building a map of the area; 2) selection of methods and tools for developing the smart car robot; 3) The practical implementation of the smart car robot; 4) Analysis of the results.

Date of Task

“ ____ ” _____ 2021.

Mater's work supervisor

Oleksiienko G.A.

Received task to be performed

Uba B.V.

ABSTRACT

Note: 52 pages, 17 figures, 19 sources literature, 3 app.

Object of study: Smart robot car

Purpose of the work: development of a mapping system prototype

Research methods: implementation of a smart robot using C++ and Arduino IDE. implementation of the server side of building an area map using Telegrams-Bot, Python and PyCharm.

Results: the analysis of literature, methods and tools for development of a mapping system prototype using the Arduino board has been carried out. After getting acquainted with the existing solutions I developed a smart robot. The mapping system prototype use the Pygame for building an area map and the Telegram-bot for show the map.

SMART CAR, MAPPING SYSTEM PROTOTYPE, TELEGRAM-BOT, PYTHON,
PYGAME, ARDUINO, ARDUINO IDE, TERA TERM, PYCHARM,
PYTHONANYWHERE

TABLE OF CONTENTS

INTRODUCTION.....	5
1. INFORMATION REVIEW.....	6
1.1. AUTONOMOUS UV ROBOT WITH SLAM	6
1.2. ROUTE SMART CAR CONTROLLED OVER SMART DEVICE (Wi-Fi)	8
1.3. HUMAN NAVIGATION USING INDOOR MAPPING.....	9
1.4. KALMAN AND ADAPTIVE FILTERS	11
1.5. FORMULATION OF THE PROBLEM.....	12
2. SELECTING PROGRAMMING INSTRUMENTS.....	13
2.1. COMPONENTS DESCRIPTIONS OF USED PARTS	14
2.2. PYGAME	26
2.3. TELEGRAM-BOT. BOTFATHER	27
2.4. TERA TERM SOFTWARE.....	28
3. APPLICATION REALIZATION.....	29
3.1. STRUCTURE DIAGRAMS.....	29
3.2. SMART CONTROL CAR SCHEME	31
3.3. CREATING A TELEGRAM-BOT	33
3.4. SERVER IMPLEMENTATION	35
3.5. CHOICE OF FILTERS	38
3.6. BUILDING AN AREA MAP.....	45
CONCLUSION.....	48
ACKNOWLEDGMENT	49
REFERENCES.....	50
ADDITION	52

INTRODUCTION

Throughout the millennia, knowledge and technology have had a significant influence on civilizations. Without technology, civilizations would have faced several challenges in all sectors of life. Thousands of civilizations have settled on Earth over the centuries, spreading out across the continents; humans have had to travel thousands of miles or come into direct contact with harmful or dangerous environments based on their lines of work; many of them have died along the way due to illness caused by exposure to certain environments or the danger they have to face while encountering some challenges due to a lack of technologies at the time. Because there was no effective technique to save and retain energy for more amazing things, a lot of information and outstanding ideas were lost due to the deaths of a person or a group of people.[1]

Today's communication includes robotics. This is a rapidly rising and fascinating area in today's globe. It's currently the simplest method to keep up with the newest technological advancements. Communication is a component of technological growth, therefore I choose to work in this sector and build stuff that would make human civilization easier in today's world, both for youngsters and grownups, and professionals depending on the application and area used.[1]

The Arduino Mega2560, a smart microcontroller capable of controlling other compatible external components, is the basis for these research. This is a prototype smart automobile, often known as a robot, that scans and maps its surroundings using some of the Arduino external hardware modules. SLAM is used to identify objects in its orbit or path (simultaneous localization and mapping). An infrared sensor is used by the smart car to detect impediments that may be placed in its path for testing purposes or that may appear unexpectedly based on real-time observations.[1]

1. INFORMATION REVIEW

1.1. Autonomous UV Robot With SLAM



Fig 1.1 UV Robot [1]

The smart robot has 4 wiper motors, 2 motor drivers, UVsensor, UV light, PIR sensor for sensing motion. The sensors mounted around it which are used for detecting humans and also for avoiding obstacles in its path. It has a charging station which it uses path planning to identify and goes back to every 5 hours to charge up. The robot is equipped with three lamps that emits powerful UV radiation with a wavelength of 240 nm. Because of its mobility and different sensors equipped on it, it can disinfect a variety of nooks and small alleyways, the UV BOT functionality can also be controlled and monitored through an app. The robot was designed and called the UV BOT by the designer of the car, according to the designer the UV BOT's technology and design allows it to operate without exposing workers to either dangerous UV radiation or infected patient rooms[1].

1.1.1 ROS – ROBOT OPERATING SYSTEM

ROS is a set of libraries, plug-ins and tools that aid software developers in the creation of robot applications. Other features include hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more.

You can control and program your robot in three different ways:

- Computer/mouse/keyboard
- Network Cable
- Robot WiFi Access Point
 - Among this three listed above the most preferred and easiest that was used for project UV BOT was to connect the monitor and keyboard to the Raspberry Pi. The robot was set up as a master while the laptop as an observer machine.
 - They three different robot programming ways have quite a few advantages, disadvantages and reliability.

Advantages:

- Computer/mouse/keyboard - Desktop/Laptop are not needed.
- Network Cable - Small number of steps.
- Robot WiFi Access Point – laptop is only required

Disadvantages:

- Computer/mouse/keyboard – Extra keyboard, mouse, and display are required.
- Network Cable – It requires internet cable, desktop/laptop.
- Robot WiFi Access Point – laptop is only required

Reliability:

- Computer/mouse/keyboard –High.
- Network Cable – Medium.
- Robot WiFi Access Point – low.

Key point to note:

While the developer and the programmer were working on the project UV BOT they discovered that RVIZ won't in VNC (Virtual Network Computing), they also made it well known that Virtual Box or VMware is much preferable to work with ROS remotely.

1.2. Route Smart Car Controlled Over Smart Device (Wi-Fi)

This smart robot is a 2-motor wheel bot that can be controlled through a smart phone or any device capable of sending signals to the Arduino robot. The 2WD Car is controlled by an Arduino board programmed to receive information or an instruction through a receiver incorporated into the Robot design. The inventor's core innovative concept is to control a step motor-based robot by sketching a path on a web-based map and having the robot transport its items to their destination by simply moving your finger on a mobile device. The project's designer also did mention that a simple working prototype robot car was built entirely without the use of feedback sensors. As a result, position error builds up over time. This project could very well inspire someone to create the ideal carrier robot by incorporating feedback sensors to correct position errors. If it's integrated with a routing algorithm like the Dijkstra algorithm, it'll be even more convenient.

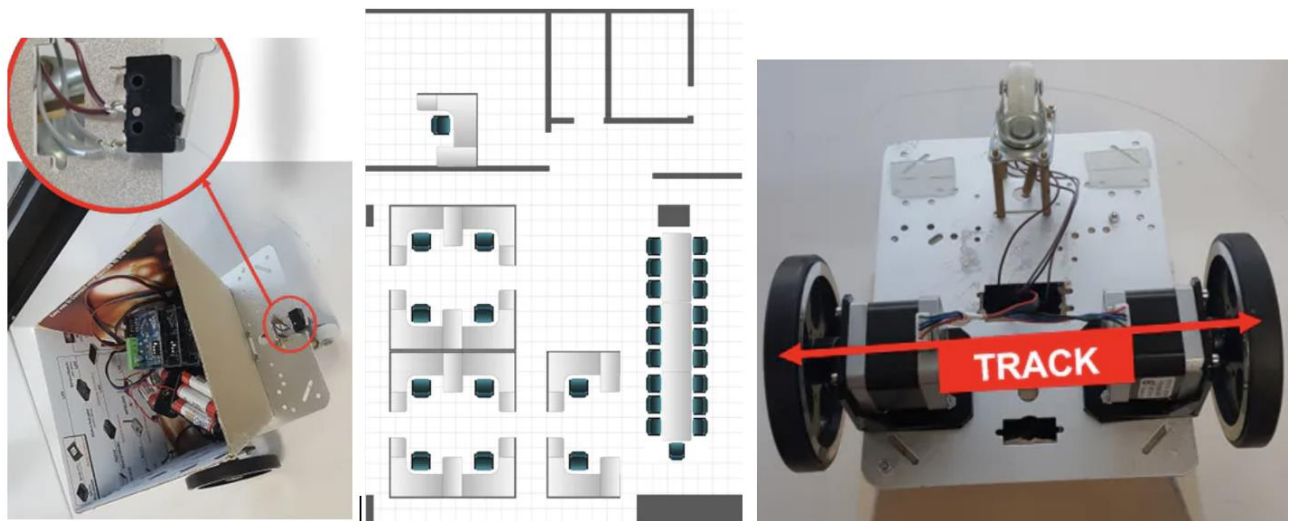


Figure 1.2 Smart car controlled over Bluetooth [3]

The route bot consist of 2 DC motors, PHPoC WiFi shield 2, 2 Stepper motor controllers, OpenBuilds Micro Limit Switch, Arduino Uno and some other unlisted compatible Arduino external components[2].

1.3. Human Navigation Using Indoor Mapping

There are a lot of ways and uses for projects similar to these, this very Human Navigation Indoor Mapping robot project was documented and designed by two authors Christian Roger and SabineTimpf. The goal of their study was to offer a low-cost SLAM implementation that could be used to quickly generate a baseline for navigable maps in indoor, enclosed situations. Non-impaired pedestrians, wheelchair users, and robots are just a few examples of use cases that require various levels of precision when it comes to indoor maps. For their project work, they both chose the non-impaired pedestrians use case as the group with the lowest accuracy requirement which they spent a sum total of 300euros for the construction of their prototype, They chose the lobby of a university building as a test location. For directing the robot's operations and displaying the scans, a program was created. During their test run as they stated, they encountered few errors but wasn't much of a huge problem cause it was not more than 70cm. The study's findings might be applied in a variety of ways, including as the platform for indoor navigation systems in vast, complicated locations like airports or hospitals. It might also be used to map unfamiliar and potentially dangerous interior settings, such as those containing gas or smoke.

This mapping smart robot incorporates a photo interrupt sensor, infrared sensor, laser sensor, ultrasonic sensor, Bluetooth module, servo, magnetometer, and an Arduino uno Rev3 with motor shield with two DC motors. This collection of component modules is really easy to use, and mounting them is similarly simple. It's also quite inexpensive to buy, and it can be found and purchased online from a variety of merchants who offer electronics or Arduino components. This smart automobile is

powered by three infrared sensors that are spaced by roughly 5 millimeters. The components of certain additional hardware aren't specified. The body of the car is also built on a Runt Rover Sprout 2WD chassis [4].

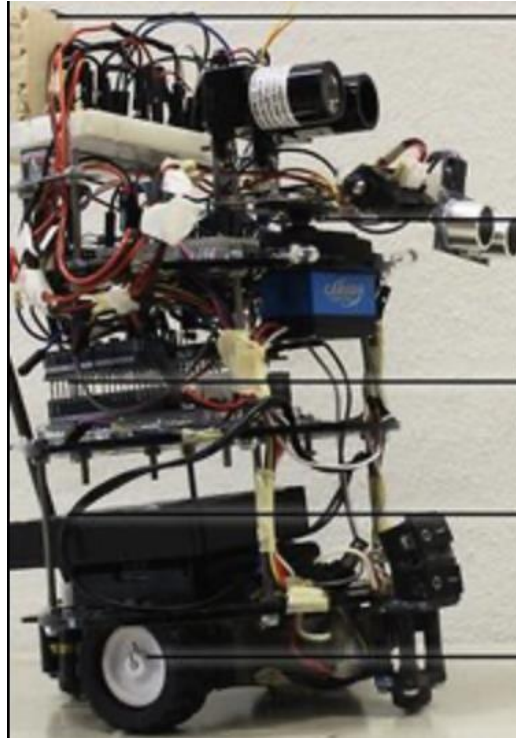


Fig 1.3.1 – Mapping Smart car (side view) [5]

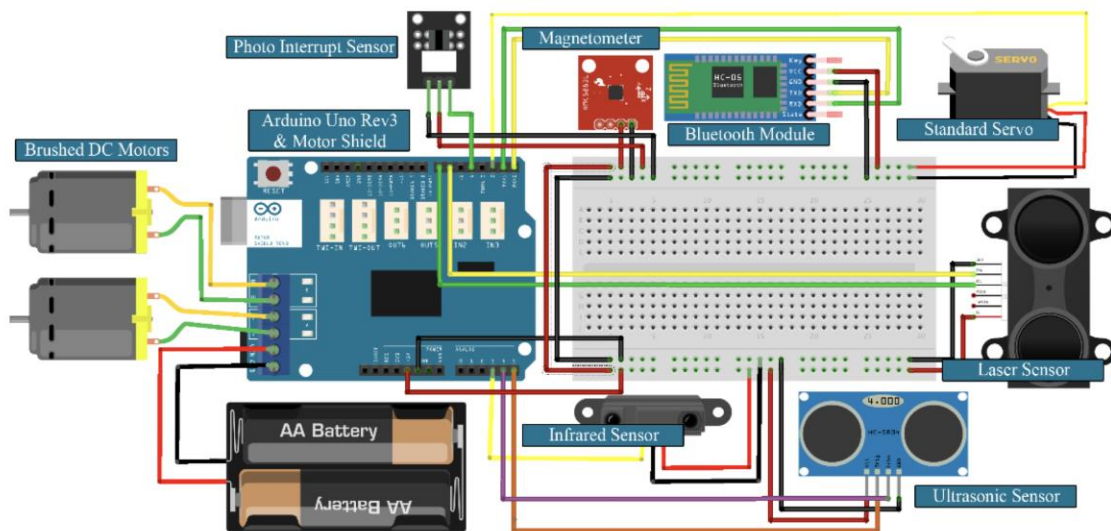


Fig 1.3.2 – Mapping Smart car (Components connections scheme) [5]

1.4. Kalman And Adaptive Filters

ADAPTIVE FILTER

An adaptive filter iteratively simulates the connection between the filter's input and output signals, which is a computational device. The filter values of an adaptive filter are self-adjusted using an adaptive algorithm.

Advantages Of Using Adaptive Filters

- Adaptive filters can be used to eliminate noise with a changing power spectrum which occurs over time, whereas traditional digital filters can't do that.
- While in online mode adaptive filter can handle real-time or online modeling tasks, such as identifying an unknown system.

NOTE: Primarily, Adaptive filters becomes very handy when dealing with real-time and online signal processing applications.

KALMMAN FILTER

A Kalman filter may be used in any situation when you have ambiguous information about a dynamic system and want to make an informed assumption about what the system will do next. Kalman filters are suitable for systems that change often. They have the benefit of being memory efficient, which mean they don't need to retain any record other than the previous state. which makes them very quick, making them ideal for real-time challenges and embedded systems.

Advantages Of Using Kalman Filters

- using the Kalman filter, the steady state error level linked with measurement noise can be greatly reduced.
- The Kalman filter is actually rather simple and straightforward to comprehend.

1.5. Formulation of the problem

The aim of the work is to develop a prototype of a mapping system that will scan the environment and build a map of the area. The prototype will include:

- 1) a data collector based on sensors;
- 2) a data processor that will build a map based on the received data;
- 3) a data transmitter, which will transmit data to the processor and return the map of the area to the client.

To achieve this goal, the following goals were formulated:

- 1) analyze known solutions for building a mapping system prototype;
- 2) choose the appropriate methods and tools to solve the problem;
- 3) design the software and hardware part of the prototype;
- 4) design data transfer between system components;
- 5) develop and test the prototype mapping system.

2. SELECTING PROGRAMMING INSTRUMENTS

USED COMPONENTS For Prototype:

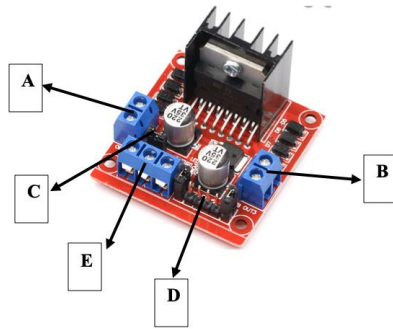
The hardware implementation step is broken down into the following sections.

The elements are as follows:

- IC L298N (Motor Driver)
- 12V stepper motor 28BYJ-48
- Arduino Mega 2560
- ULN2003 stepper motor driver
- Power Pack
- DC Motors on the Left and Right
- DC Motor Wheels
- Electric Switch(ON/OFF)
- M - M and F M jumper wires
- Sharp GP2Y0A21YK0F IR distance sensor (10-80 cm).
- Smart vehicle chassis with 4WD.
- Bluetooth module (HC-06)

2.1. Components Descriptions Of Used Parts

IC L298N (Motor Driver)

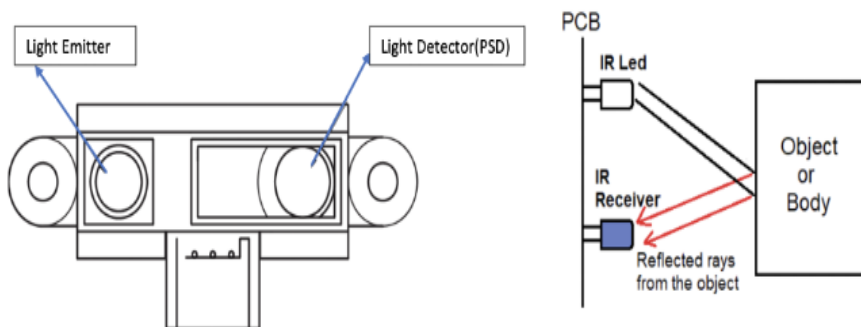


The L298N motor driver module enables users to run two motors with up to 2A apiece in both sides spontaneously and simultaneously. The board could easily be driven with 12V or above, and indeed the L298N motor driver's maximal voltage input is stated to be 46V–50V. The whirling directions of DC motors can be altered by adjusting the input voltage on their respective linking terminals. To change the trajectory and velocity of the DC motor, a merging of an H-Bridge for motor rotation (direction) and PWM for motor velocity control is employed. The motor driver module is made up of two threaded port blocks for the A and B motor wire connections, as well as another terminal block labeled D that has eight pins for various different wire terminal connections. Some of the terminals are used as Ground pins, 5V pins for upwards and reverse rotary motion, and others are utilized for numerous different reasons. The additional pins (ENA and ENB) are being needed to adjust speed. [6] The portion marked E is a main power supply screw port that may be either as an input or output (5V) pin ports, while the section marked C is a 5V EN jumper [1.]

Sharp GP2Y0A21YK0F IR distance sensor



The IR distance sensor is an electronic Arduino device that is widely used by Arduino programmers for object detection. Sharp IR distance sensors are frequently paired together with a microcontroller and microprocessor platforms such as Arduino, Raspberry Pi, and other comparable platforms. The sensor has a total of 3 pins for controlling the sensor, with a 10 cm to 80 cm (4" to 32") measurement range. This infrared sensor is less expensive than the sonar rangefinders, yet it performs far better than other IR options.



A Diagram illustration of how IR distance sensor works

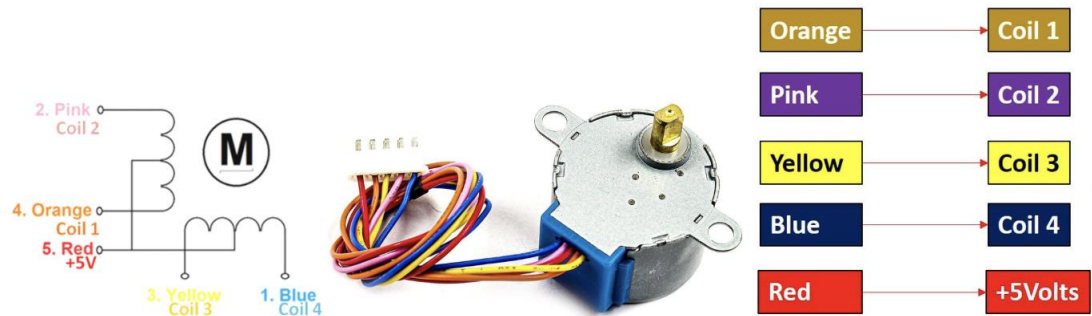
To detect the distance between objects, an IR distance sensor employs a beam of infrared light that reflects off them. The distance is estimated by triangulating the light beam. An IR LED and a light detector, also known as Position Sensing Device, make up the IR sensor. Whenever a light beam is reflected by an object or an obstacle, the reflected beam reaches the light detector, where it forms a "optical spot" on the PSD. The three pins, yellow, black and red,

Yellow – The yellow wire or the V0 pin should be linked to the A0(analog) port on the Arduino board.

Black – The wire should be attached to Arduino on port GND.

Red – This pin port should be connected to 5v on the Arduino uno board[10].

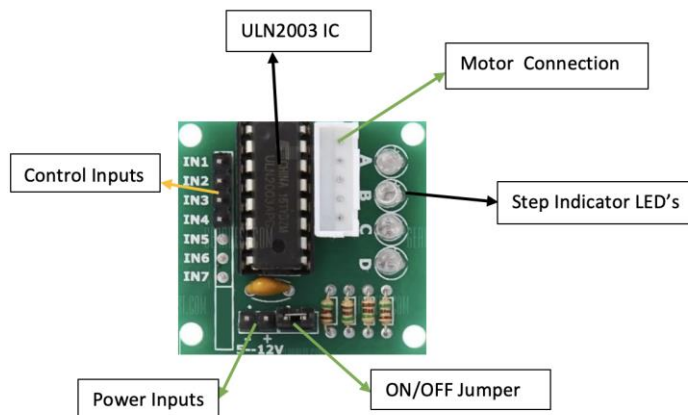
Stepper Motor 12V 28BYJ-48



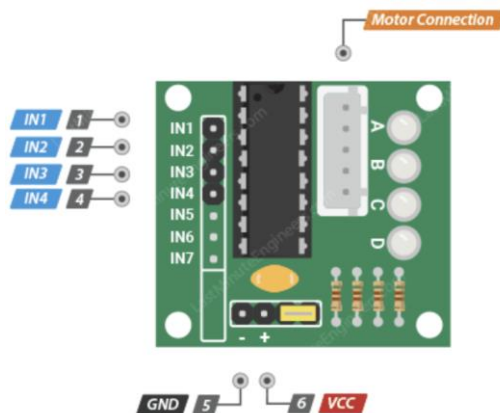
The Stepper motor 12V 28BYJ-48 is a 11.25° degrees, When the 28BYJ-48 motor is in full step mode, each step translates to a rotation of 11.25 degrees, based on the data sheet. That implies each cycle has 32 stages ($360^\circ/11.25^\circ = 32$). This is the most prevalent stepper motor in low-power industrial applications, as well as in popular hobbyist projects. The 28BYJ-48 is a five-coil unipolar stepper motor with a 5V working voltage, The motor's current usage is roughly 240mA. That implies the 28BYJ-48 contains four coil patterns, each of which can be driven with 5 volts. Because of the 5 volt working voltage, any microcontroller, such as the TM4C123 Tiva Launchpad, may simply operate this motor.

As can be seen from the picture description above, the motor has four coils, one of which is connected to +5V (Red) while the others Orange, Pink, Yellow, and Blue are pulled out as wires which supply the coils with sequence logic[9].

ULN2003 stepper motor driver



The board features a connection that properly matches the motor wires, making it very simple to attach the motor to the board. There are four control inputs (IN1-IN4) as well as power supply connections that are designed specially on the board. The ULN2003 driver board is a popular motor driver IC that consists of an arrangement of seven Darlington transistor pairs. Four of the seven pairs are employed on this board, and each pair can drive loads up to 500mA and 50V. On the PCB, four LEDs display activity on the four control input lines, which signify the stepping state, which are labelled from A - D on the board. When stepping, they offer a lovely view because of the activity on them.



NOTE:

- The IN1–IN4 pins control the motor functionalities. by Connecting them to Arduino's digital output pins.

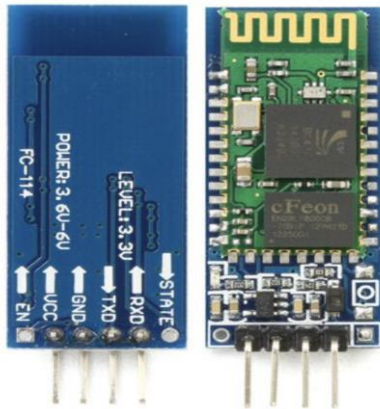
- GND is ground pin or the negative connection pin port to the driver.
- The motor is powered by the VDD pin, which provides 5 volts. It is suggested that you use a 5V external power supply with it. Using the 5V voltage supply from your Arduino board to drive this stepper motor is not a good idea since it consumes and requires much power.
- Motor Connector pin ports, this is where the motor connects to the rest of the system which has a unique ports design than the others. Because the connection is keyed, it can only be used in one direction.

Electric Switch(ON/OFF)



Switches are among the electrical components that may break or halt an electrical circuit flow owing to their various and distinct designs. Whether a switch is ON or OFF is determined by stopping current or diverting it from one conductor to another. The toggle switch utilized in the design of this project is used to control the flow of electricity in the smart car. This switch has two fundamental actions, ON and OFF. When the switch is on, electricity from the battery pack flows to the motor shield, which then passes 5V to the Arduino Mega2560 board. When the switch is off, current from the battery pack can no longer flow. [1].

Bluetooth module (HC-06)



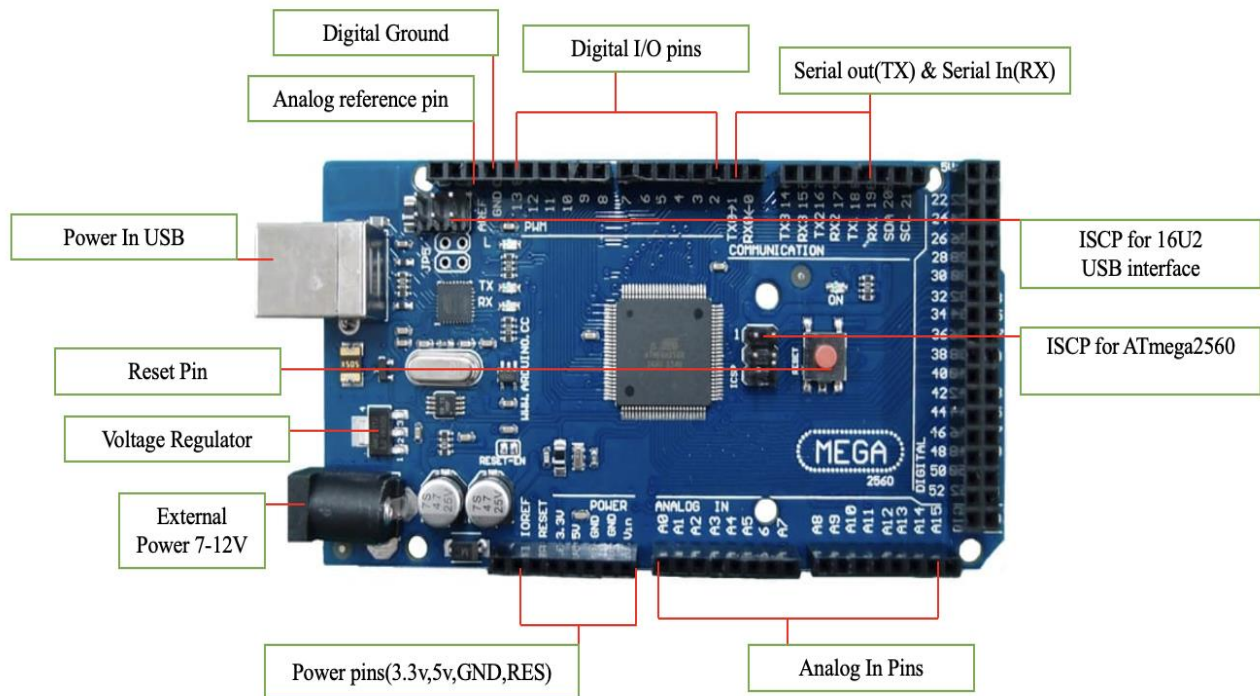
The HC-06 model for Arduino has 4 or 6 pins relying on the manufacturer style. An Average of 3.6V to 6V could be supplied to HC-06 4pins, the ground pin is linked to the Arduino ground, and the RXD and TXD pins are attached to different suitable ports in the Arduino depending on the developer programming approach. [5] The Arduino Bluetooth model HC-06 may be connected to a Bluetooth device such as a phone, a computer, or an iPad. The RXD pin is 3.3V and should be linked to the Arduino Uno and Mega to prevent the model from reaching 5V, which might harm it[1].

DC Motor wheels



In all, the smart car has four wheels comb, two on the left side and two on the right side. Since a wheel cannot function on its own, it need the help of a DC Motor; similarly, the Dc motor cannot propel the smart car without the wheels. The wheels support the full weight of the vehicle and are constructed in such a manner that the entire weight of the car may be pivoted. [1]

Arduino Mega2560



The Arduino mega2560 board is perhaps most certainly, and the most widely utilized among the core parts of an Arduino design stage used by various developers of electronic products centered on Arduino. [5] During the development stage of the design process, the Arduino mega2560 is suitable for diagnosing, debugging software and evaluating hardware pieces of devices. This is due to the existence of several unique features integrated in the board; the Arduino mega2560's characteristics and capabilities are as follows: [1]:

The interface converter, it a special port with allows you to connect the card directly to the USB ports of the computer the port designs can be different depending on it design models.

3.3 and 5 V powerful stabilizers ports or pins, which allows you to power your devices from the board.

The board can also be powered from the computer through different designed special port via USB.[1]

External devices are connected to the computer during the development and debugging phases through connections that are configured to turn on or off based on the application. [1]

The board also contains some LEDs indicators like the RXD, TXD and the ON led indicators, as the name implies it is used to know when the board is switched on for the ON led indicator, and when the TX and RX is in used the RXD and TXD LEDs also indicates when it is being used.

The ATmega328 microcontroller is used in the Arduino Mega, and its design includes:

There are 54 digital input & output ports in total.

There are 16 analog port inputs.

16 MHz clock frequency

USB plugin port is available.

Connector for electricity.

Connector for in-circuit programming.

There's a reset button.

Regulator of voltage.

LEDs are used as indicators .[7]

NOTE: The voltage stabilizer on the Arduino board regulates the amperes of current coming into the board, and when the reset button is hit, the reset button will establish brief contact with the ground pin, forcing the board to reset any code existing on the board. The voltage stabilizer should be included on the Arduino board for good functionality, not so that it can be connected with on our Arduino board, but so that it can actually assist the board manage the voltage flowing in. To avoid fully destroying the Arduino board, it is not recommended to connect a 20V power source to it[1].

Power Pack



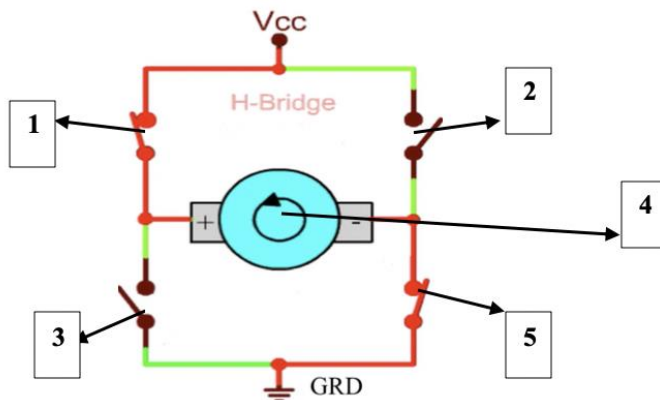
The overall output of the battery pack when inserted with a battery is defined by the cumulative total of the two-battery voltage. This Arduino battery pack employed in the construction of the smart car can only incorporate a maximum of two cells. With a power source ranging from 6 to 20 volts, the Arduino board may work and function properly. If you deliberately supply more than 12 volts to the Arduino board, the power regulator on the board may overheat, resulting in minor or major degradation to the board. During the smart car's development. I realized that a voltage range of 7 to 9 volts is ideal for correctly supplying and distributing accurate electricity to the Arduino board as well as the external components linked to it. Easily plug the "+" end of the batteries to the Arduino microcontroller and the "-" terminal to Arduino ground. The orange or red LED's on the Arduino board should signal that the board is powered on. [1]

DC Motors



By uploading or transmitting a value between 0 and 255 from the motor shield velocity controller ports to an Arduino analog output, you may regulate the motor's speed.

The DC motor may be effectively hooked and attached to a PWM output source of a transistor, which is needed to adjust the velocity of the motor by tweaking the PWM, based on the scale and voltage capability of the DC motor. The voltage going through the DC motor is changed based on the direction you pick for regulating the rotational motion of the motor. An H-Bridge is the best popular approach for implementing this adjustment. The H-bridge system is made up of four switching elements, and the motor must be situated at the center to produce the H-like configuration pattern. When two certain switches are triggered at the same instant, the current flow direction is reversed, allowing users to change the motor's rotational motion by revolving or whirling upward or downward[1].



H-bridge connection graphically represented.

The motor revolves in a certain motion when 1 and 5 are set to HIGH and 2 and 3 are set to LOW, as shown in the H-bridge connection example.

The motor revolves in the reverse way from the first point when 1 and 5 are set to LOW and 2 and 3 are set to HIGH.

The DC motor is represented by component 4 on the diagram.[1]

Arduino IDE



Arduino was initially released in 2005 as a basic programming software (IDE) or tool for Mac and Pc users by Interaction Design Institute Ivrea (IDII) students. Following its launch, Arduino has proved successful in creating, decorate, and create prototypes for the electronics company's development stage. [8] The microcontroller device is free software and planned in such a manner that it can conveniently be integrated with a combination of technologies for handling the effectiveness and reactions of many other systems attached to the board, such as sound systems, screen displays, motors, drivers, stepper motor, led, as well as some other suitable external parts, such as the Bluetooth module used for my prototype smart control technology design.[1]

The most impressive aspect of Arduino is the ease with which people with little or no expertise can create basic scripts and also comprehend and utilize the IDE with little or no difficulty. Because of its features and capabilities, Arduino is one of the most well-known and widely used tools for engineers and inventors in developing and creating interactive items or devices. Massimo Benzi and David Cuartielles established Arduino in 2005, focusing on a wiring framework from 2003[1].

Machine programs may be readily written and uploaded to the Arduino board. Because it is based on an open-source platform. It's important to remember that Arduino

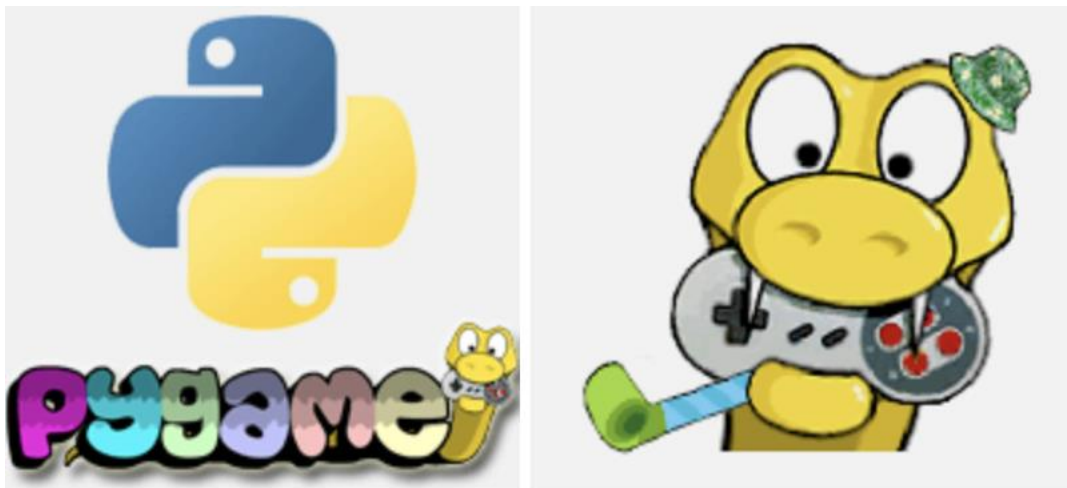
is interoperable with Linux, Mac OS X, and Pc. The Arduino interface is also written in the Java programming language. Because the code execution might be in c, c++, java, or python, programs are compiled into c/c++. There are several limits to the Arduino software coding language, such as storing all of your scripts in the same folder.

Although executing java straight on an Arduino is a little tricky, java could be used to interact with and operate an Arduino via another java-enabled computer using serial port software system. [1]

To integrate the languages C and C++, the Arduino IDE contains certain code structure rules. Only two functions are required by the user: setup() and loop(). Both functions must be contained in the sketch, even if they are not utilized elsewhere in the code. The script on the Arduino IDE is called a sketch. It's crucial to note that while C and C++ are both mainstream scripting language, they are significantly different in terms of the notion or approach of OOP (Object Oriented Programming), in the notion that C++ supports OO features whereas C does not. However, the Arduino IDE/compiler accepts C and C++ as input. C++ is also used to write many of the libraries which are included in Arduino, although it is important to note that C++ is not garbage collected. You must control the memory on a daily basis. [1]

2.2. Pygame

Pygame is among the major fields in python for enormous libraries for Game development. Pygame is a cross-platform collection of Python modules for making video games, which contains a collection of computer graphics and sound libraries for the Python programming language. Pete Shinnars created Pygame to take the role of PySDL, his innovation, the Pygame is well suited to the development of client-side applications that may be packaged as a standalone executable.[11]



Pygame may be installed in a variety of methods, including using the pip tool and the command "py -m pip install -U pygame --user." The alternative option is to install through IDE; to do so, follow the steps below:

- From your file tab navigate to settings,
- From the settings select project interpreter and on the right side click on the "+" icon,
- Type Pygame into the search field after clicking the "+" icon, next hint on the install package button to download and install. [11]

Note: Enter the command "import pygame" to see if pygame was installed entirely and correctly.

2.3. Telegram-bot. BotFather

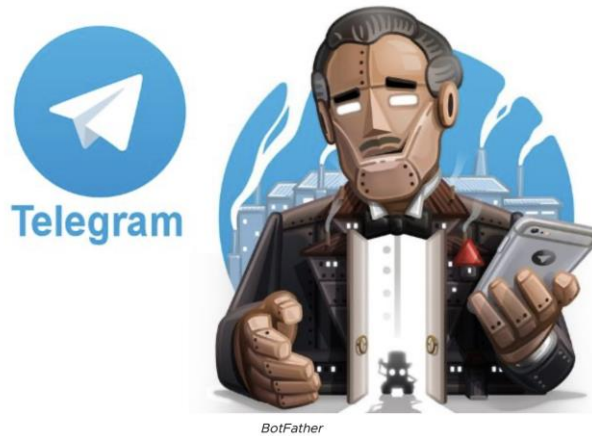


Figure – 2.4 Telegram BotFather

BotFather is the master of all bots cause of Its abilities that allows you to create new bot accounts as well as manage the ones you already have. The bot comes with pre-programmed bot commands that allow users to perform a variety of tasks and commands on it. These are a handful of few of some command line options: `"/deletebot"` from the name, this command line lets users delete newly generated bots, `"/mybot"` lets users edit newly created bots, `"/revoke"` lets users revoke access tokens for newly created telegram custom bot, and `"/newbot"` – this command grant one the access to create and customize his/her bot with a unique name, unique HTTP API token, this token makes it possible for the user to build a python code for the different functionality of choice to permit the user of the newly created telegram bot. Bot can be used for other features like to Receive targeted news and updates, It can add material from other services like Gmailbot, Googlebot, and YouTubebot to Telegram chats. Bots do not have an online status or a last seen timestamp; instead, the interface displays the label 'bot.' Bots are unable to start communication dialog with users.

NOTE: For the project a telegram bot “[@Mappingcarbot](#)” is created for communication and interaction purposes for my smart car.

2.4. Tera Term Software



Figure 2.4 – Tera Team Software

Tera Term which is most of the time also known as TeraTerm, this software application is an open source software which is also free at the same time. this software also mimics variety of computer terminals, ranging from DEC VT100 to DEC VT382. Telnet, SSH 1 & 2, and serial port connections are all supported. It also includes a macro programming language as well as a few other handy plugins.

- The software application connects to the Arduino smart car via Bluetooth connection and wait to receive IR distance reading from the sensor, when the sensor completes its 360° rotation with the help of the stepper the programming saves it as a txt file.
- When the file is saved, the file is sent from the pc to telegram bot, the file is then saved by the sever and is then ready for mapping.
- The user of the telegram bot have to initiate a dialog for the bot to reply or carry on command instructions.

3. APPLICATION REALIZATION

3.1. Structure Diagrams

Diagram Illustration of the used component:

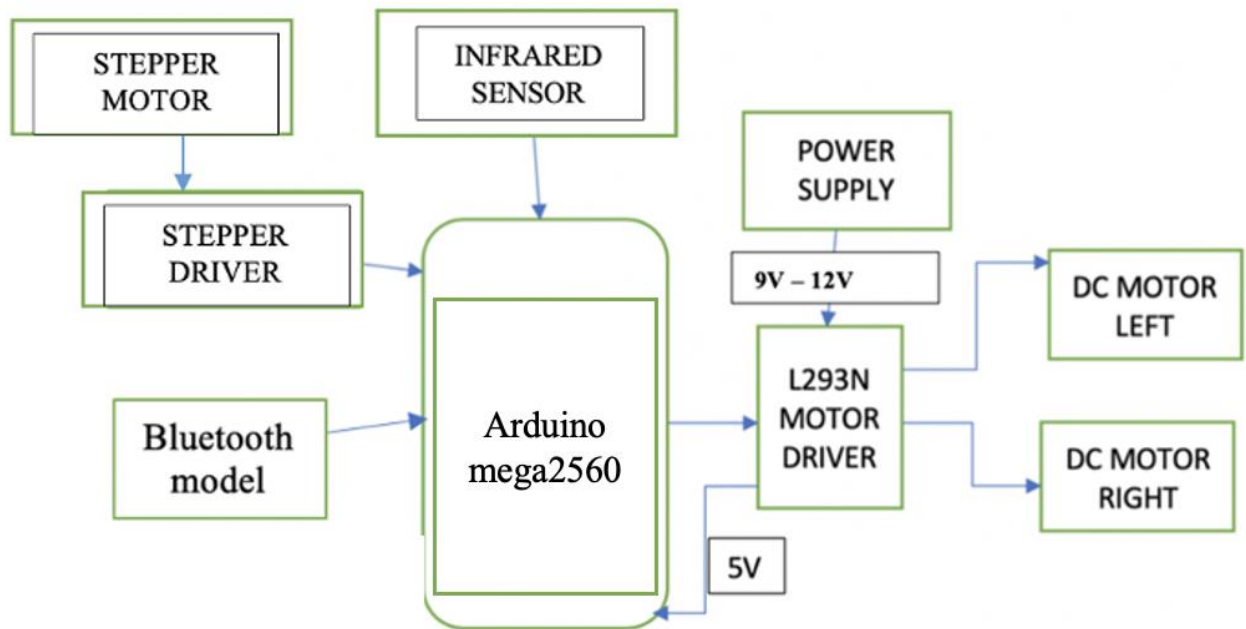


Figure 3.1.1 - Diagram of robot used component

In the diagram above, all the wiring for my robot hardware are depicted and described as mathematical forms for hardware parts, whereas the lines with arrows indicate wiring. The diagram illustration above shows how external power between 9voltage – 12voltage is being distributed from the source into the L293N motor driver, this motor driver then regulated 5voltage into the Arduino Mega2560. The Arduino Mega2560 is then capable of distributing 5v or 3.3v to any external components which are connected to it for proper and stable voltage supply.[1]

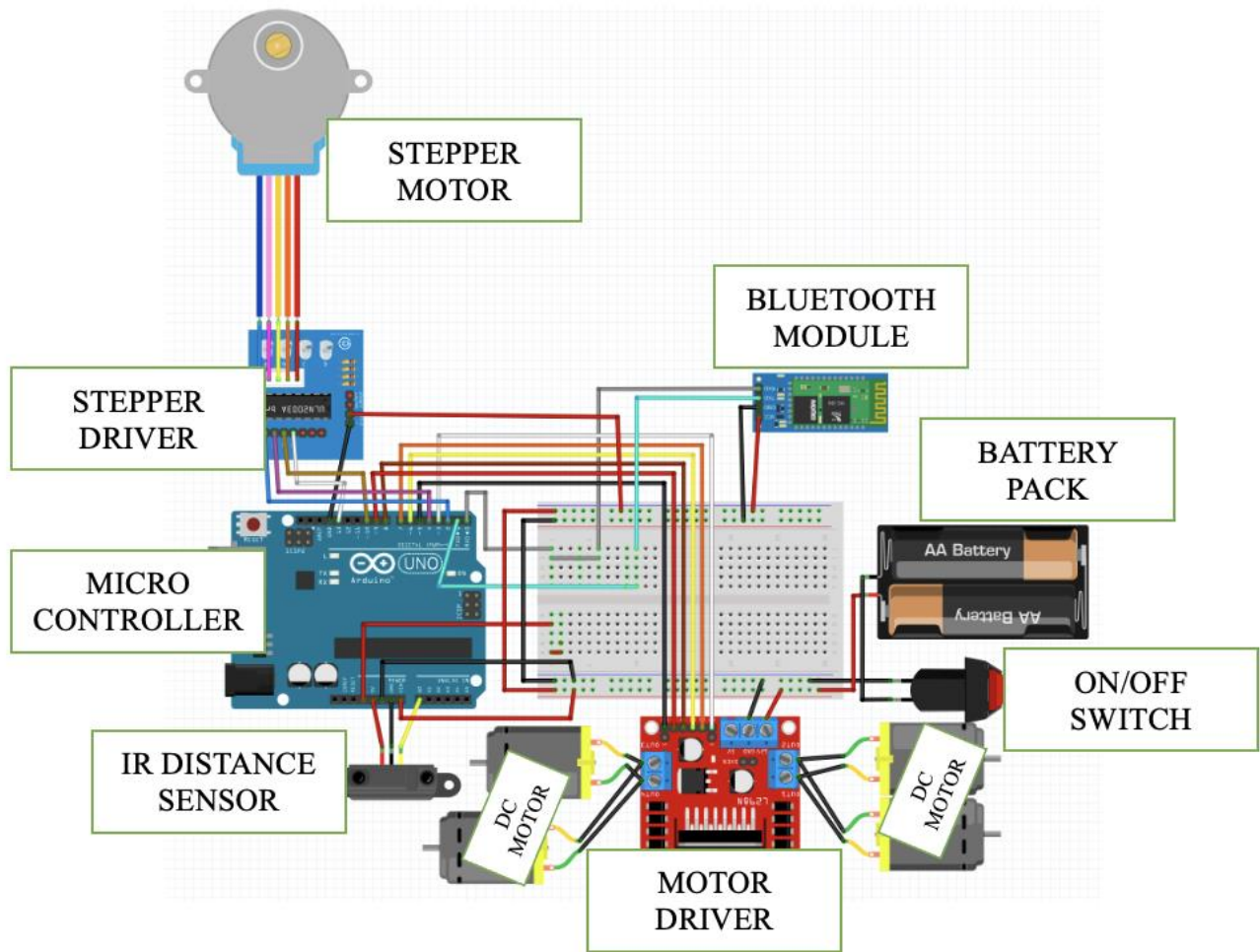


Figure 3.1.2 - Robot Scheme

This diagram above is designed using Fritzing IDE by using the inbuilt component parts on the IDE to represent a well-connected connections of the robot design components and how there interact with one another (power input 8V – 12V voltage goes in to the motor driver). The switch on the diagram is used to stop the entire distribution supply of power to the smart car, while the IR distance sensor is mounted on the stepper motor to enable smooth 360 degrees movement. When the Smart car is switched on from the power switch, the car stays on delay for 20ms and rotate the IR distance sensor(car movement from point A to point B is controller from smart phone) which is mounted on the stepper motor for 360 degrees taking 2159 measurement at it complete rotation(360°), once the rotation is complete the stepper rotates back to starting point.

3.2. Smart Control Car Scheme

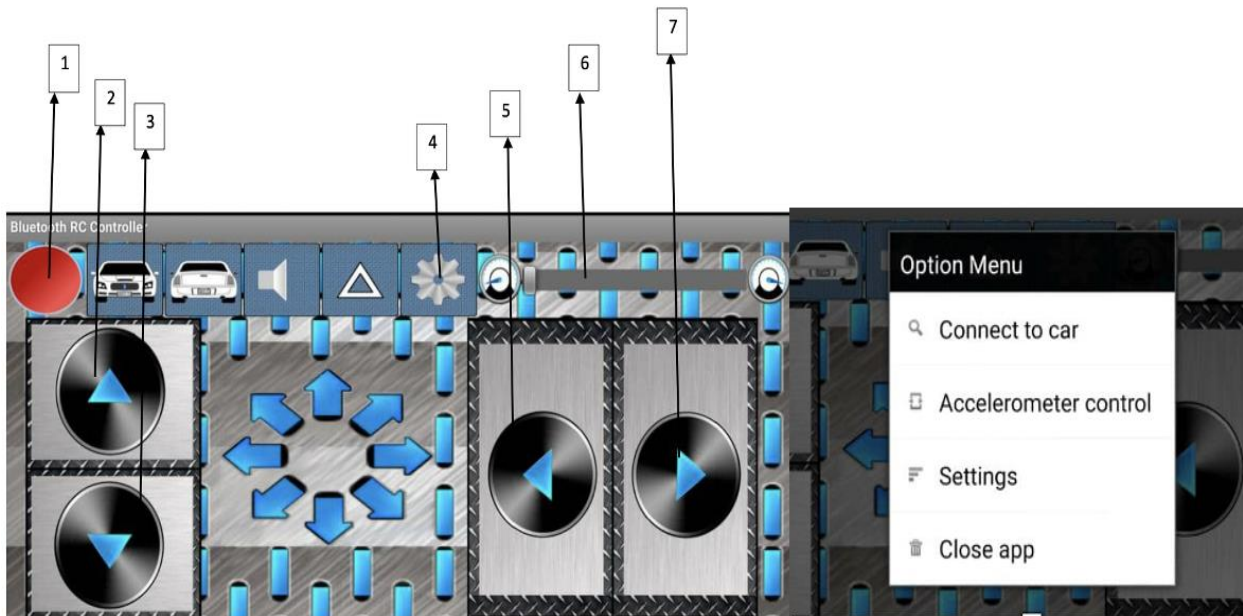


Fig 3.2.1 Rc controller with Bluetooth (android app)

This program is exclusively available for Android users; consequently, the Bluetooth rc controller program for Ios and Os x users is still to be posted to the Appstore.[1]

The marked buttons are used to connect to the smart car or to control its movement.

- The "1" portion is a signal indicator that changes color based on whether the device is connected or not.

GREEN indicates that the device is connected; RED indicates that it is not connected.

- 2 – The control button controls movement in a forward or upward direction.
- 3 – Press this control button to go backwards or down.
- 4 – When you hit this key, a drop menu displays, as seen in Figure 3.2.1.

Connecting the device to your smart car, accelerometer control, app settings, and shutting the app altogether are all available from the drop-down menu on the left.

- 5 – The control regulates the direction of the car's left turn. [1]

- 6 – The control increases the vehicle's acceleration.
- 7 – The right turn direction of the car is controlled by this key. [1]

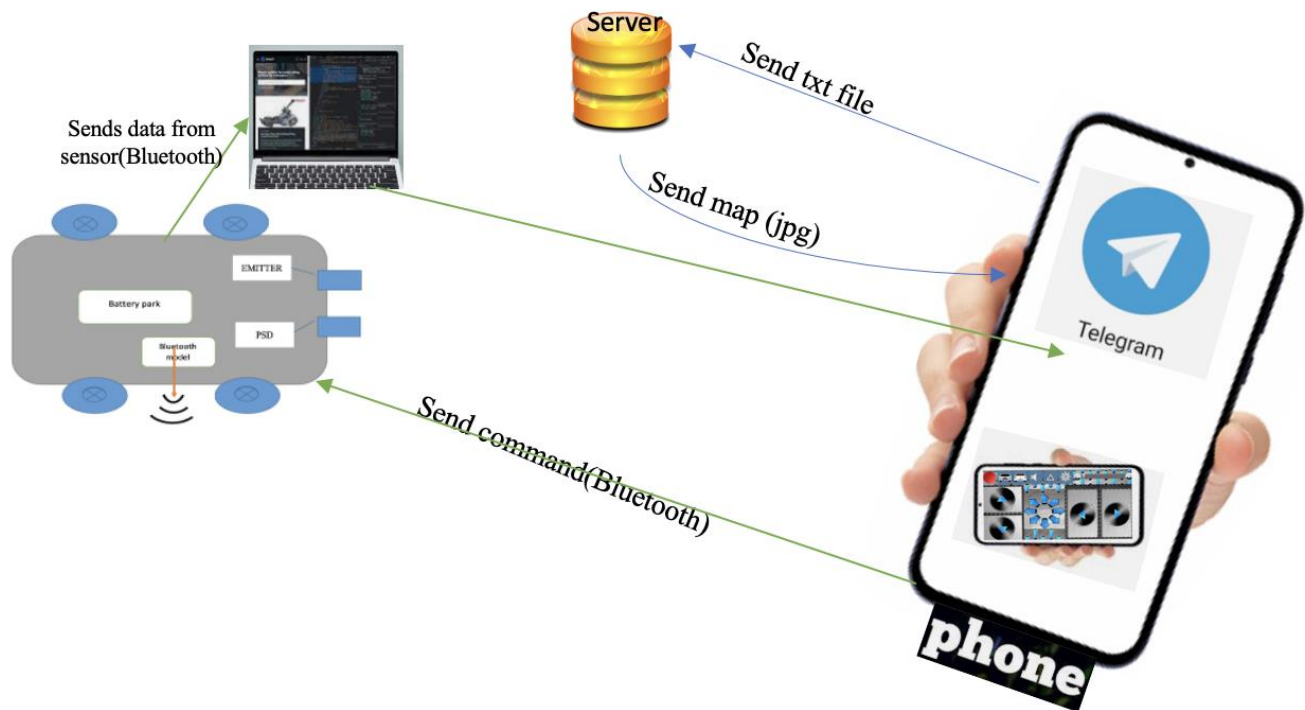


Figure 3.2.2 - Scheme of data transfer between system components

The diagram above is a pictorial full representation of the complete working principle of my smart car. It shows various communication between the smart car, mobile smartphone, telegram, and the server. The Arduino smart car sends txt data to the server through telegram and also it can be controlled from the mobile smartphone using Bluetooth. The IR distance sensor readings is sent to the pc over Bluetooth connection and the readings are saved in txt file format which is then sent to telegram from the pc. The txt file is sent to the telegram bot, the bot saves the txt file and the user inputs a “/map” command for the bot to send the file for analysis. The server reads the file and uses the values on the txt file to plot a map and sends back a plotted map back to the telegram bot in jpg file format.

3.3. Creating a Telegram-bot

To make a new bot, use the “/newbot” command will have to be typed by the user into the BotFather chat cause BotFather will not start a dialog with its user. Before generating an authentication token for your new bot, the BotFather will request the user to input your name and username.

- Your bot's name appears in contact information and other places.
- The Username is a unique name that can be used in links or in chat to mention. When Creating a username is must finish with 'bot,' e.g.'smartcar bot' or 'SmartCarBot.' Usernames that the user creates must be between the range of 5-32 characters long which are also very much case insensitive.
- The token is a string that looks like this: 110xx1543:AAHdqTxxxxxxGWJxfSeofSAsxxxxxLDsaw, and it's needed to authorize the bot and submit queries to the Bot API.

NOTE: Keep your token private and protected; it may be used to operate your bot by anybody.

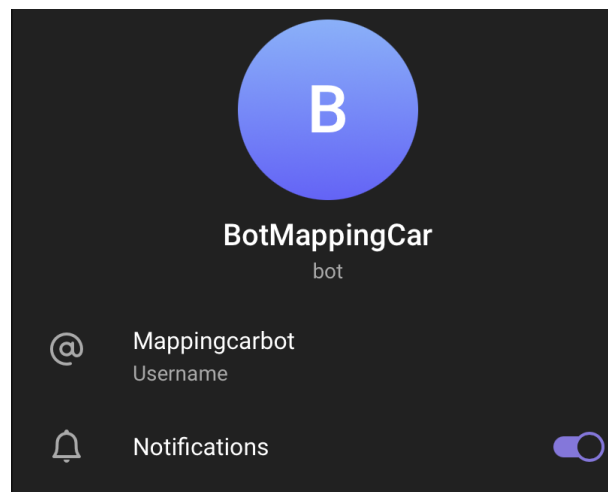


Figure 3.3 Telegram Bot for Smart Car Control

When working with the “@Mappingcarbot” telegram bot created for the smart car control, the list down instructions should be followed accordingly:

- Firstly, a txt file is sent from the IR distance sensor on our Arduino smart car to the telegram bot created, just as seen on figure 3.3 a txt file is being sent to the bot(the file must be in txt format) “AdaptationAngle.txt”. The file is saved and recorded by the bot.
- After the txt file is saved by the bot, the user have to pass in a command to the telegram bot for the bot to initiate mapping using the values on the saved txt file. The user have to send “/map” command message to the telegram bot to draw map. Once the map is complete, the server sends back the plotted map to the telegram in jpg format.

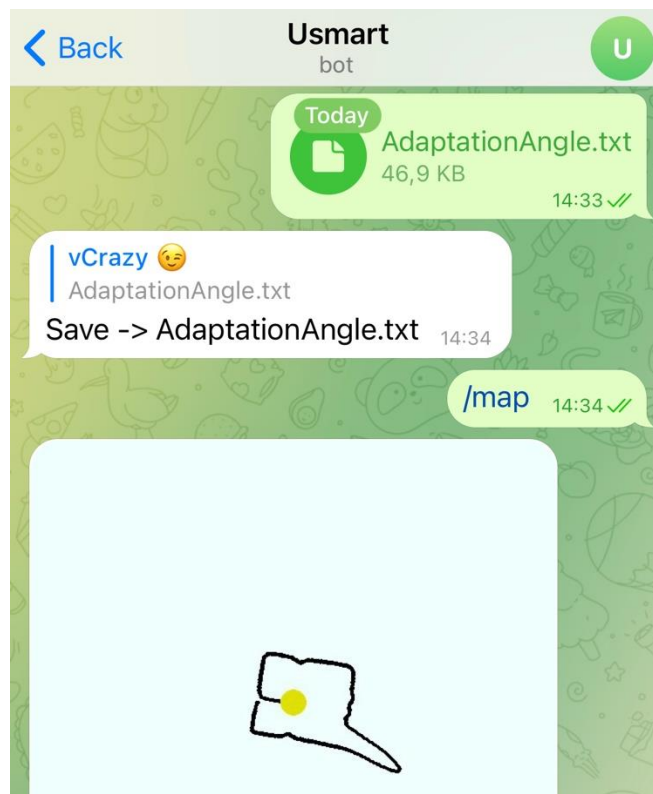


Figure – 3.3 TelegramBot Chat for Smart Car

3.4. Server implementation

Building and working with server I made use of an online platform called the “Pythonanywhere.com”, which is based on the Python programming language for web hosting service and online integrated development environment. It allows you to access server-based Python and Bash command-line interface from within your browser. A code editor with syntax highlighting is also included.

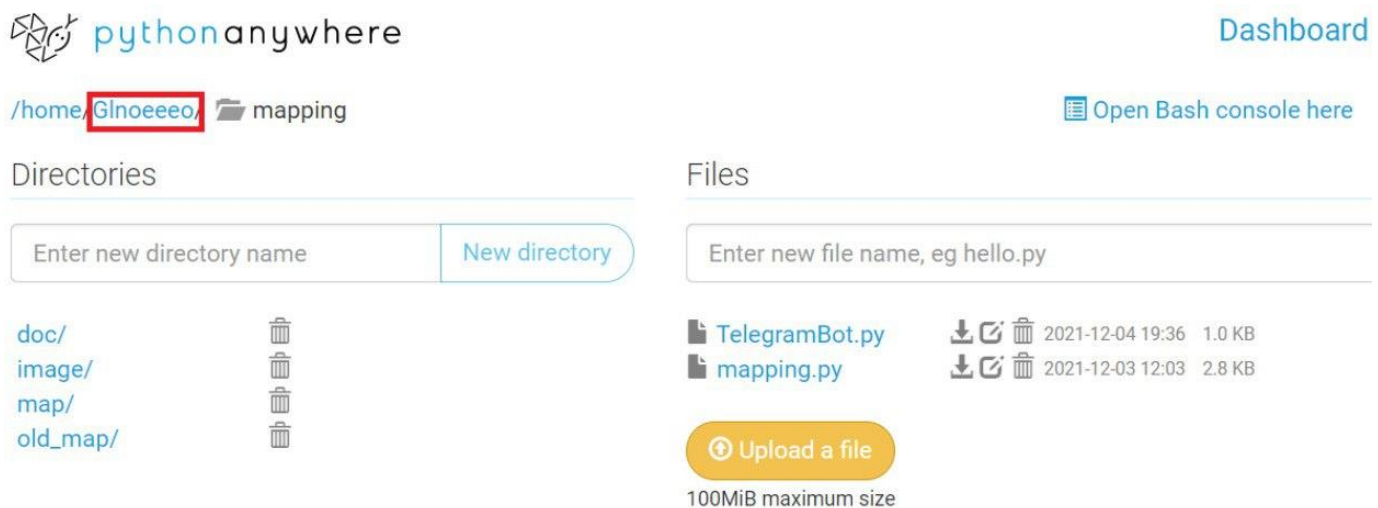


Figure – 3.4.1 - Files Directory Folder on Server(online)

- The Pythonanywhere online server holds two main python code files for the smart car control project which are the TelegramBot file and mapping file, it also holds four sub files in side our main folder created on the server. The four sub files are “/doc” folder, “/image” folder, “/map” folder and “/old_map” folder. On the server folder “Glnoeeeo” we created, all the four sub folder is created inside the main folder. The “TelegramBot.py” file and the “mapping.py” file is also uploaded directly into “Glnoeeeo” Server main folder.

Dashboard

Welcome, [Glnoe](#)CPU Usage: 5% used – 5.94s of 100s. Resets in 9 hours, 22 minutes [More Info](#)File storage: 11% full – 57.6 MB of your 512.0 MB quota [More Info](#)[Upgrade Acc](#)Recent
Consoles

+ 5 -

[Bash console 22478719](#)[Bash console 22479030](#)Recent
Files

+ 5 -

[/home/Glnoeeee/mapping/mapping.py](#)[/home/Glnoeeee/mapping/
TelegramBot.py](#)Recent
Notebooks

+ 5 -

Your account does not support
Jupyter Notebooks. [Upgrade your
account](#) to get access!All
Web apps[Glnoeeee.pythonanywhere.com](#)[Open Web tab](#)

Figure – 3.4.2 – Bash Console on Server (online)

Using the pythonanywhere online server is provides you with free 2 server bash console access for your project work, two bash console is created for our two main python files, the TelegramBot.py and the mapping.py. The TelegramBot.py runs the start-up server for or telegram app for the bot to be functional to perform it task, while the second bash console runs the mapping.py which initiate the mapping code for txt file received from the Arduino IR distance sensor.

[/home/Glnoeeee/mapping/](#) doc[Open Bash console here](#)

Directories

Enter new directory name

[New directory](#)

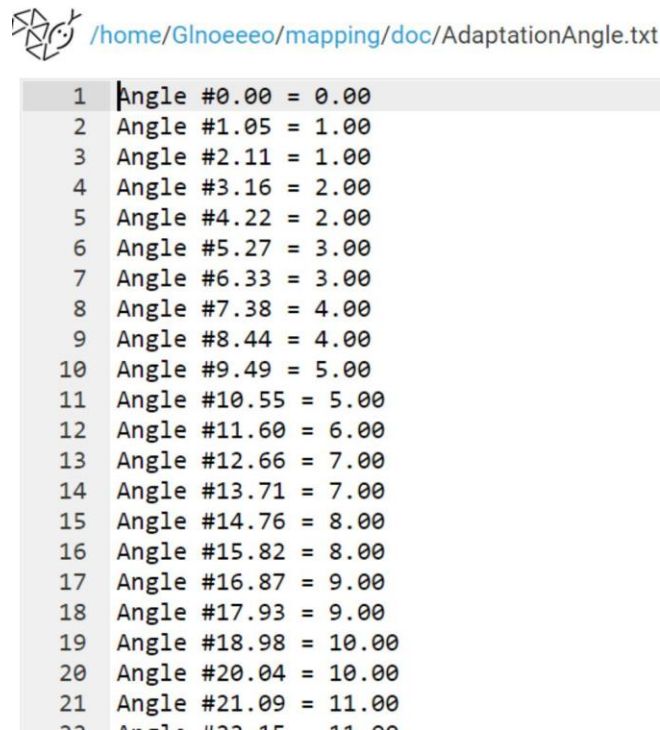
Files

Enter new file name, eg hello.py

01.12.txt		2021-12-07 08:14	236.9 KB
3cvad.txt		2021-12-01 01:11	47.0 KB
AdaptationAngle.txt		2021-12-07 08:13	47.0 KB
AlfaBetaAngles.txt		2021-12-01 01:14	47.0 KB
alfa.txt		2021-11-30 22:22	47.0 KB
calm.txt		2021-12-01 01:01	47.0 KB
leastSquaresAngle.txt		2021-12-01 01:36	47.0 KB
name.txt		2021-12-07 08:14	10 bytes
test2.txt		2021-12-07 08:13	47.3 KB
test23.txt		2021-12-01 10:26	47.3 KB
test3_OK.txt		2021-12-01 01:12	236.7 KB
test45.txt		2021-12-01 10:27	46.4 KB
without.txt		2021-12-01 01:12	47.0 KB
CalmanAngle.txt		2021-12-01 01:16	47.0 KB

Figure – 3.4.3 – Directory for Saved Files on Server

The “/doc” sub folder is located inside our main folder(“Glnoeeee”), this folder holds all the txt files sent to the telegram bot, the telegram bot receives the txt file and saves it to the “/doc” folder created on the pythonanywhere sub folder of “Glnoeeee” main folder on the online server. Figure 3.4.3 is a screenshot of all the sent sample files sent to the telegram bot during our smart car test run.



The screenshot shows a text file named `AdaptationAngle.txt` located at `/home/Glnoeeee/mapping/doc/`. The file contains a list of 21 lines, each representing a measurement. The first column is a line number (1-21), the second column is the word "Angle", the third column is a hash symbol followed by a number, and the fourth column is an equals sign followed by a number. The numbers in the third column increase from 0.00 to 21.09, and the numbers in the fourth column increase from 0.00 to 11.00.

Line	Angle	Distance
1	#0.00	0.00
2	#1.05	1.00
3	#2.11	1.00
4	#3.16	2.00
5	#4.22	2.00
6	#5.27	3.00
7	#6.33	3.00
8	#7.38	4.00
9	#8.44	4.00
10	#9.49	5.00
11	#10.55	5.00
12	#11.60	6.00
13	#12.66	7.00
14	#13.71	7.00
15	#14.76	8.00
16	#15.82	8.00
17	#16.87	9.00
18	#17.93	9.00
19	#18.98	10.00
20	#20.04	10.00
21	#21.09	11.00

Figure – 3.4.4 – txt File From IR sensor

The txt files sent from our Arduino IR distance sensor of the smart car holds two important values for our mapping.py to correctly plot our mapping graph. The values is hold are the angle value and the distance value, from Figure 3.4.4, the fist values are the angle values while the second values are the distance values.

NOTE: The program code to pythonanywhere.com online server is added to the “ADDITITION” section of the project report. The codes for the TelegramBot.py and code for mapping.py, both codes will be included to the section.

3.5. Choice of Filters

Fig. 3.5.1 – This diagram above is the initial setup used during mapping test stages, the robot is being boxed in which represents obstacles or objects in this case. The smart Arduino robot is connected to the PC and the angles with readings(Stepper and IR sensor) is being calculated using the Arduino uno IDE, when it is done, the readings which were recorded is being sent to PyCharm IDE for map plotting.

- five filter samples where used, Alpha Beta, Average, Adaptive, Least mean square, Simple Kalman filters, all this filters listed will be tested individually with my robot to best pick the filter that will work better for the smart robot[14].



Figure 3.5.1 – Environment Mapping Setup

1) WITHOUT FILTER CASE

Figure 3.5.2 from the diagram above, is representation of when filter is not used during reading from the IR sensor for mapping, the corners are not a well-defined in a way in shows the initial diagram of the test environment in Figure 3.5.1.

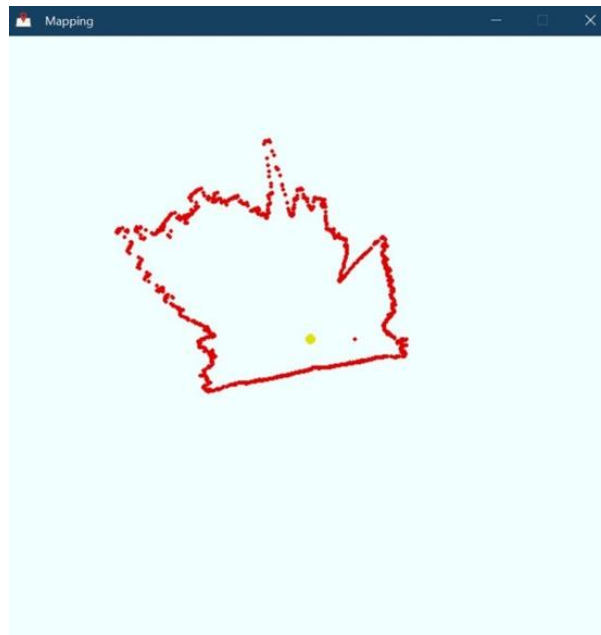


Figure – 3.5.2 Diagram for Without Filter

2) ALPHA BETA FILTER CASE

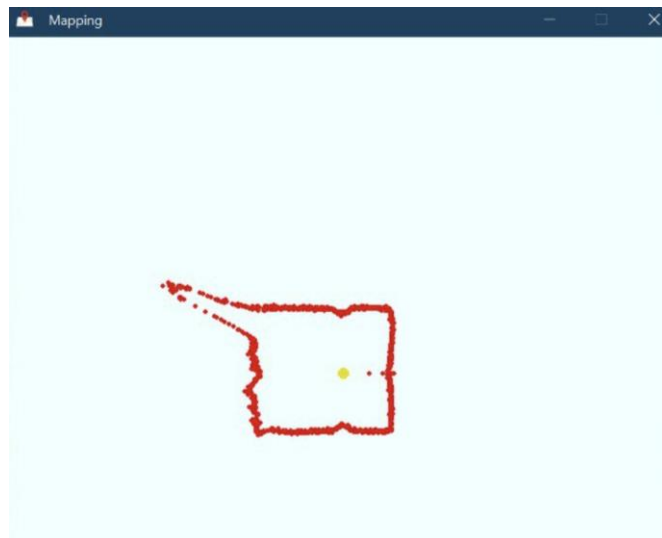


Figure 3.5.3 – Alpha Beta Filter

Code sample for Alpha Beta Filter:

```

float dt = 0.02;
float sigma_process = 3.0;
float sigma_noise = 0.7;

float ABfilter(float newVal) {
    static float xk_1, vk_1, a, b;
    static float xk, vk, rk;
    static float xm;

    float lambda = (float)sigma_process * dt * dt / sigma_noise;
    float r = (4 + lambda - (float)sqrt(8 * lambda + lambda * lambda)) / 4;
    a = (float)1 - r * r;
    b = (float)2 * (2 - a) - 4 * (float)sqrt(1 - a);

    xm = newVal;
    xk = xk_1 + ((float) vk_1 * dt );
    vk = vk_1;
    rk = xm - xk;
    xk += (float)a * rk;
    vk += (float)( b * rk ) / dt;
    xk_1 = xk;
    vk_1 = vk;
    return xk_1;
}

```

A simple kind of observer for estimating, data smoothing, and control applications is when an alpha beta filter comes in handy. It has similarities with Kalman filters and linear state observers, all of which are essential for control theory. Its main benefit is that it does not need a comprehensive system model.

3) AVERAGE FILTER CASE

Code sample for Avarage Filter:

```
const int NUM_READ = 30;

float midArifm() {
    float sum = 0;
    for (int i = 0; i < NUM_READ; i++)
        sum += значение;
    return (sum / NUM_READ);
}
```

The arithmetic average is computed by adding the total of the values and then dividing by the number of them. The first approach works precisely as described in the code diagram above: using a loop, everything is summed into a variable, then divided by the number of measurements.



Figure 3.5.4 – Average Filter

4) ADAPTIVE FILTER CASE



Figure 3.5.5 – Adaptive Filter

Code sample for Adaptive Filter:

```
float expRunningAverageAdaptive(float newVal) {  
    static float filVal = 0;  
    float k;  
  
    if (abs(newVal - filVal) > 1.5) k = 0.9;  
    else k = 0.03;  
  
    filVal += (newVal - filVal) * k;  
    return filVal;  
}
```

The coefficient could be made adaptive such that it responds to rapid changes in the value to function appropriately with abruptly changing signals. When the filtered value is "far", from the real one, for an instance, the coefficient rises dramatically, allowing you to swiftly reduce the "gap" between the two values. If the value is "near," the coefficient is set low to effectively filter noise.

5) LEAST MEAN SQUARE FILTER CASE



Figure 3.5.6 – Least Mean Square Filter

Code sample for Least mean square Filter:

```
float a, b, delta;

void minQuad(int *x_array, int *y_array, int arrSize) {
    int32_t sumX = 0, sumY = 0, sumX2 = 0, sumXY = 0;
    arrSize /= sizeof(int);
    for (int i = 0; i < arrSize; i++) {
        sumX += x_array[i];
        sumY += (long)y_array[i];
        sumX2 += x_array[i] * x_array[i];
        sumXY += (long)y_array[i] * x_array[i];
    }
    a = (long)arrSize * sumXY;
    a = a - (long)sumX * sumY;
    a = (float)a / (arrSize * sumX2 - sumX * sumX);
    b = (float)(sumY - (float)a * sumX) / arrSize;
    delta = a * (x_array[arrSize-1] - x_array[0]);
}
```

The technique of least squares is the next filter that permits users to examine a noisy process and anticipate its behavior.

6) SIMPLE KALMAN FILTER CASE

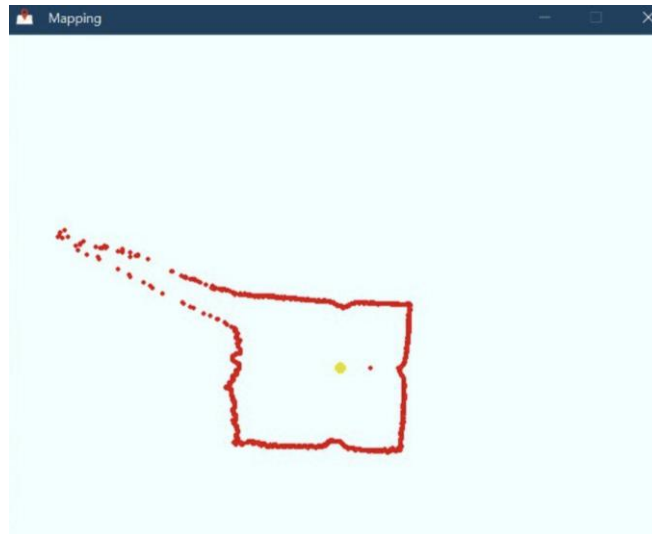


Figure 3.5.7 – Simple Kalman Filter

Code sample for Simple kalman Filter:

```
float _err_measure = 0.8;
float _q = 0.1;

float simpleKalman(float newVal) {
    float _kalman_gain, _current_estimate;
    static float _err_estimate = _err_measure;
    static float _last_estimate;

    _kalman_gain = (float)_err_estimate / (_err_estimate + _err_measure);
    _current_estimate = _last_estimate + (float)_kalman_gain * (newVal - _last_estimate);
    _err_estimate = (1.0 - _kalman_gain) * _err_estimate + fabs(_last_estimate - _current_estimate) * _q;
    _last_estimate = _current_estimate;
    return _current_estimate;
}
```

The filter controls the anticipated measurement noise, the estimate spread (which can be set the same as the measurement noise, but can also adjust itself during the Simple kalman filter operation), and the rate of change of values (0.001-1, this can be altered by choice), are all adjusted by the filter.

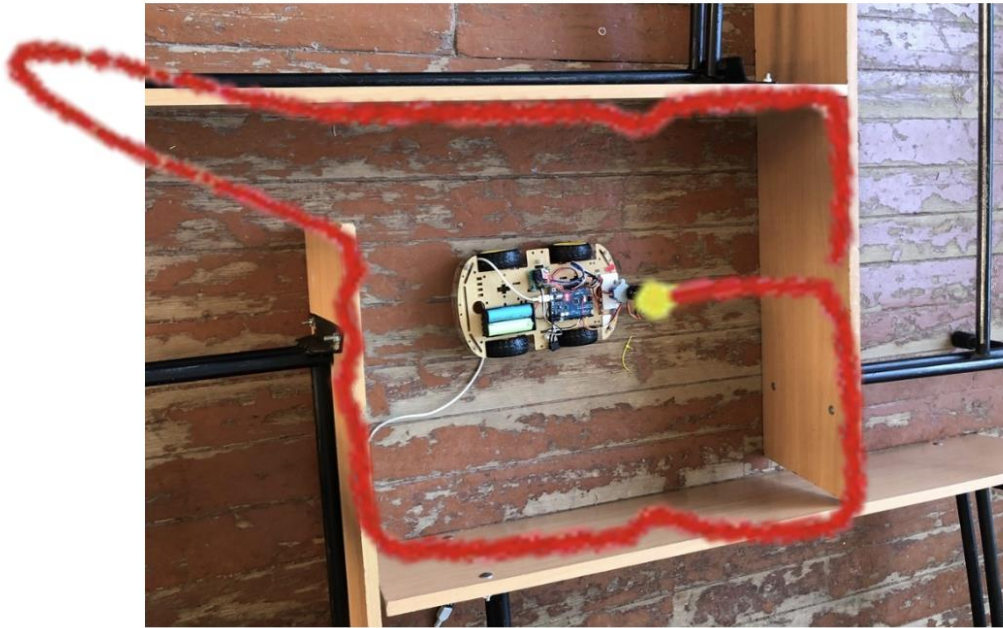


Figure 3.5.8 – Overlapping filter map with real image

For This project the Adaptive filter will be used, from it diagram in Fig 3.5.5 we can see during the test check it completely shows a well-defined picture of the robot and the object surrounding it. Figure 3.5.8 shows the actual instance of when the selected chosen(Adaptive Filter) choice of filter is placed over on the actual environment used for the project test purposes. The Red drawn along the real image of the test environment shown on Figure 3.5.1 is over lapped by Arduino IR distance scanned map which is implemented by using the adaptive filter. The Yellow dot on the picture is the position of the sensor placed Infront of the smart robot.

3.6. BUILDING AN AREA MAP

The smart car from it starting position will take 5 position reading, all the position readings from the map will be overlapped with their actual images respectively. The

Pictures below represent all the positions taken by the smart car during its prototype test drive.

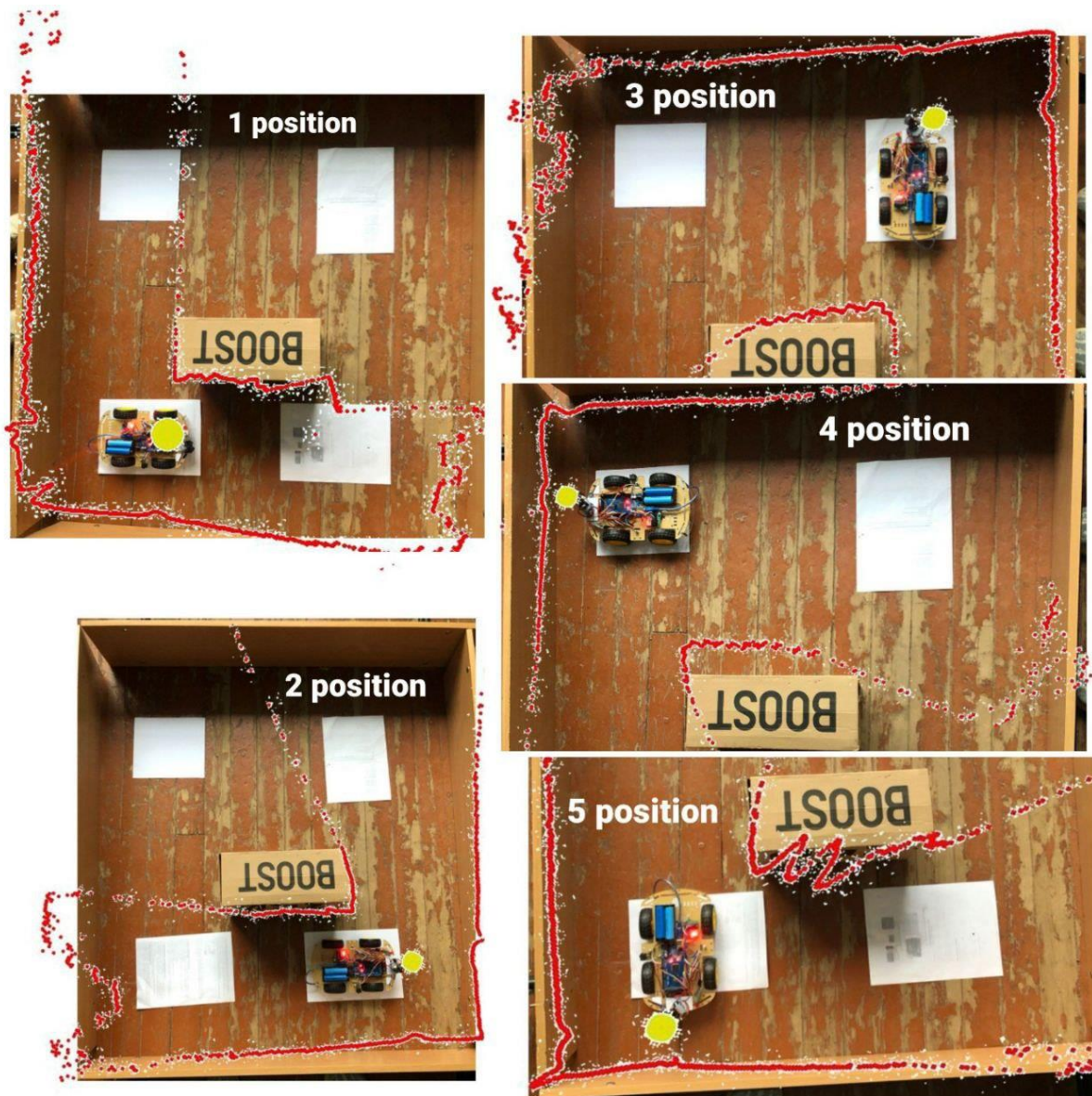


Figure – 3.6.1 Overlapping all position map with real image

- Position 1 – starting position of the smart car.
- Position 2, 3, 4, 5 – movement is made before taking next position for reading,

NOTE: From all the position pictures above, the red outlines the surrounding borders of the test environment while the yellow dots indicate the position of the smart

CONCLUSION

Based on the results of the analysis of existing prototypes for building a terrain map based on Arduino, I identified problems in their functionality and accuracy of construction. Therefore, it was decided to create an alternative mapping system.

After analyzing the methods and tools, I designed a prototype of a mapping system and a system for transferring data between the elements of the system. C++ was used to collect data and control a smart robot. The components of the system interact with each other under the means of Bluetooth, Tera Term, Telegram-bot. To send data to the server and display a map of the area, a telegram bot was developed and launched on the server. A Pygame to build the map was used. The Python codes on a free host pythonanywhere.com was run.

As a result, all the tasks were achieved. A prototype of a mapping system has been developed, which creates a map of the area with good accuracy. In the future, it is possible to make improvements to the system when deciding an algorithm for simultaneous localization and mapping (SLAM), to that already for detailing objects in 3-D space.

ACKNOWLEDGMENT

It's a privilege to appreciate the wonderful family of Sumy State University for their earnest help in maintaining my practical and laboratory abilities in computer science for my master's report.

First and foremost, I want to express my gratitude to my parents for their support, enthusiasm, and financial aid. Without all of this, I would not have been able to complete my school studies up to this point, and I owe a huge debt of gratitude to my entire family.

I'd also want to thank my department's HOD, Mr. Anatoly Stepanovich Dovbysh, the Dean, the deputy dean, and all of the lecturers for having such a positive impact on my life, academic skills, and so on via their teaching and lecturing efforts. My abilities and understanding have greatly improved as a result of the practical classes.

I'd want to express my gratitude to Mrs Galina, my supervisor, for her supervision and direction during my project work and for making it possible. I'd also want to thank her for believing in me and seeing my project through, and a very big thanks to Mr Sasha for his assistance and the time he dedicated for me to successfully complete my project work.

Thanks also to my department and the board of those in charge of Sumy University's foreign relations office, which has assisted in the management of issues relating to the training sector. A paper will not be sufficient to demonstrate their assistance and direction for all of the work they confined in me during my school years.

I'm grateful to have had the good fortune of receiving consistent encouragement, support, and advice from all of the computer science teaching team, which aided me in completing my master's report work effectively. In addition, I'd want to express my heartfelt gratitude to the entire laboratory personnel for their prompt assistance.

I'd want to express my gratitude to all other anonymous staff members or group members who helped me achieve good practice in various ways.

REFERENCES

1. Uba, V. B. "Obstacle Avoiding Smart Robot ," Bachelor's thesis //Sumy State University , 46., 2020.
2. "Autonomous UV Robot With Slam," 2021, [accessed 04-Nov-2021]. [Online]. Available:<https://create.arduino.cc/projecthub/361126/autonomous-uv-robot-with-slam-215203>
3. G. Kampf, D. Todt, S. Pfaender, and E. Steinmann, "Persistence of coronaviruses on inanimate surfaces and their inactivation with biocidal agents," Journal of Hospital Infection, vol. 104, no. 3, pp.246–251, 2020.
4. "Arduino Drawing Route For Office," 2019, [accessed 04-Sept-2021]. [Online]. Available: https://create.arduino.cc/projecthub/iot_lover/arduino-drawing-route-for-car-on-the-office-s-map-cc6d82
5. "Arduino Indoor Mapping With Slam," 2017 - 2020, [accessed 04-Nov-2021]. [Online]. Available: https://www.austriaca.at/0xc1aa5576_0x00390cd2
6. N. Van Doremalen, T. Bushmaker, D. H. Morris, M. G. Holbrook, A. Gamble, B. N. Williamson, A. Tamin, J. L. Harcourt, N. J. Thornburg, S. I. Gerber et al., "Aerosol and surface stability of sars-cov-2 as compared with sars-cov-1," New England journal of medicine, vol. 382, no. 16, pp. 1564–1567, 2020
7. "Arduino Starter Kits," 2017 - 2021, [accessed 24-Nov-2021]. [Online]. Available: <https://www.elecrow.com/download/Starter%20Kit%20for%20Arduino%28user%20manual%29.pdf>
8. "Autonomous Hexapod Robot," 2020, [accessed 4-Dec-2021]. [Online]. Available: <https://www.instructables.com/id/Spider-Pig-Autonomous-Hexapod-Robot/>
9. J. Folkesson, J. Leonard, J. Leederkerken, and R. Williams, "Feature tracking for underwater navigation using sonar," in Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS07), 2007, pp. 3678–3684
10. "Arduino Material Handout," 2020, [accessed 2-Dec-2021]. [Online]. Available: https://dlnmh9ip6v2uc.cloudfront.net/learn/materials/1/Arduino_final_handout.pdf

11. "Arduino 28byj48 Stepper Motor," 2020, [accessed 2-Dec-2021]. [Online]. Available: <https://lastminuteengineers.com/28byj48-stepper-motor-arduino-tutorial/>
12. "Arduino Microcontroller Board," 2020, [accessed 2-Dec-2021]. [Online]. Available: <https://en.wikipedia.org/wiki/Arduino>
13. "Arduino IR Distance Sensor," 2020, [accessed 2-Dec-2021]. [Online]. Available: <https://www.makerguides.com/sharp-gp2y0a21yk0f-ir-distance-sensor-arduino-tutorial/>
14. "Pygame," 2020, [accessed 2-Dec-2021]. [Online]. Available: <https://www.javatpoint.com/pygame>
15. S. Khan, D. Wollherr and M. Buss, "Modeling Laser Intensities For Simultaneous Localization and Mapping," in IEEE Robotics and Automation Letters, vol. 1, no. 2, pp. 692-699, July 2016.
16. Welch, G., Bishop, G. An introduction to the Kalman Filter. University of North Carolina at Chapel Hill. Retrieved from https://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf. Accessed in January 25, 2017.
17. SparkFun Company. 9 degrees of freedom Breakout – MPU 9150. Retrieved from <https://www.sparkfun.com/products/11486>. Accessed in February 5, 2016
18. Horvat, Z. Finding the Mode of an Array. Retrieved from <http://www.codinghelmet.com/?path=exercises/array-mode>. Accessed in March 24, 2016
19. Arain M A, Trincavelli M, Cirillo M, Schaffernicht E, Lilienthal A J. Global Coverage Measurement Planning Strategies for Mobile Robots Equipped with a Remote Gas Sensor. Sensors 2015, 15(3). p. 6845-6871.
20. Eirik Thon. Mapping and navigation for collaborating mobile robots. Master's thesis, Dept. of Engineering Cybernetics, NTNU, 2016.

ADDITION

Arduino code:

```

/*
  @brief
  @date 14.11.2021

*/

#include "GyverFilters.h"
#include <Stepper.h>

#define sharp 0

GKalman testFilter(0.4, 0.5);
GABfilter testFilter2(0.08, 40, 1);
RingAverage<int, 50> fil;
GMedian3<int> testFilter3;

Stepper myStep(32, 2, 10, 4, 13); //32 steps for 1 rotate motor,
//with reducer need 2048

int potpin = 0;           // analog pin used to connect the potentiometer
int val;
float distance;
// variable to read the value from the analog pin

int speed_left_motor = 0;
int speed_right_motor = 0;

```

```

int LeftSpeed = 3;
int RightSpeed = 5;
int RightMoveF = 8;
int RightMoveR = 9;
int LeftMoveF = 6;
int LeftMoveR = 7;

char trip[] = {'K', 'F', 'F', 'S', 'K', 'L', 'S', 'F', 'F', 'S', 'K', 'L', 'S', 'F', 'F', 'S', 'K', 'L', 'S',
'F', 'F', 'S', 'K'} ;

int state; //Signal variable from Bluetooth
int vSpeed = 255; // speed 0-255

void setup() {

    myStep.setSpeed(200);
    Serial.begin(9600);
    Serial.flush();
    Serial.println("f, w");

    pinMode(LeftSpeed , OUTPUT); //Control Speed Of The Left
    pinMode(RightSpeed, OUTPUT); //Control Speed Of The Right

    //Direction for left Side
    pinMode(LeftMoveF, OUTPUT);
    pinMode(LeftMoveR, OUTPUT);

    //Direction for Right Side

```

```
pinMode(RightMoveF, OUTPUT);
pinMode(RightMoveR, OUTPUT);
}
```

```
float dist() {
  // 5V/1024 = 0.00488281
  float volt = analogRead(sharp) * 0.00488281;
  float distance = 29.988 * pow(volt, -1.173);
  // Serial.print("Dist = ");

  //delay(20);
  delay(1); // for video
  return distance;
}
```

```
void meas() {
  for (int i = 0; i <= 2048; i++) {
    myStep.step(1);          // 1 stepps = 0.17578 grad   6 stepps == 1.0546 grad
    float distance ;
    distance = testFilter.filtered(dist());
    Serial.println("Angle #" + String(i * 1.0546) + " = " + String(distance));
    // delay(50);
  }
  myStep.step(-2048);        //Rotate angle to 0 (360 cycle * 6 stepps)
  delay(1000);
}
```

```
// side: 0 - left side;   1 - right side
```

```

// way: 0 - backl 1 - forward
// velocity [0; 255]//ANALOG

void motor(bool side, bool way, int velocity) {
    if (side) {
        speed_right_motor = velocity;
        if (!way) {
            analogWrite(LeftSpeed , velocity); //Setting the engine speed forward
            digitalWrite(RightMoveF, HIGH); //Turn
            digitalWrite(RightMoveR, LOW); //Move Forward
        } else {
            analogWrite(LeftSpeed , velocity); //Set the speed of the engine back
            digitalWrite(RightMoveF, LOW); //turn
            digitalWrite(RightMoveR, HIGH); //Move Forward
        }
    } else {
        speed_left_motor = velocity;
        if (!way) {
            analogWrite(RightSpeed, velocity); //set the engine speed
            digitalWrite(LeftMoveF, LOW); //turn
            digitalWrite(LeftMoveR, HIGH); //Move Forward
        } else {
            analogWrite(RightSpeed, velocity); //set the engine speed
            digitalWrite(LeftMoveF, HIGH); //Turn
            digitalWrite(LeftMoveR, LOW); //Move Forward
        }
    }
}

```

```

//NOTE ---[RED(9-R) *** YELLOW(6-L) ]
void loop() {
  delay(10000);
  for (int i = 0; i < sizeof(trip); i++) {
    switch (trip[i]) {
      case 'K':
        meas(); // measurment
        break;
      case 'F':
        motor(0,1,100); // front
        motor(1,1,100);
        delay(600);
        break;
      case 'S': //stop
        motor(1, 1, 0);
        motor(0,1,0);
        break;
      case 'L':
        motor(1,1,180); // left
        motor(0,0,180);
        delay(650);
        break;
    }
  }

  //Save incoming data from HC-06 to state

```



```

if (Serial.available() > 0) {
    state = Serial.read();

    // Mode Bluetooth
    // 4 speed level for app
    if (state == '0') {
        vSpeed = 0;
    }
    else if (state == '1') {
        vSpeed = 100;
    }
    else if (state == '2') {
        vSpeed = 180;
    }
    else if (state == '3') {
        vSpeed = 200;
    }
    else if (state == '4') {
        vSpeed = 255;
    }

    /***/

    // Forward-> F
    if (state == 'F') {
        analogWrite(LeftSpeed , vSpeed); //Setting the engine speed forward
        digitalWrite(RightMoveF, LOW); //Turn
        digitalWrite(RightMoveR, HIGH); //Move Forward
        analogWrite(RightSpeed , vSpeed); //Setting the engine speed forward
        digitalWrite(LeftMoveF, HIGH); //Turn
    }
}

```

```

    digitalWrite(LeftMoveR, LOW); //Move Forward
}

/*****

//Forward Left -> G.
else if (state == 'G') {
    analogWrite(LeftSpeed , vSpeed); //Setting the engine speed forward
    digitalWrite(RightMoveF, LOW); //Turn
    digitalWrite(RightMoveR, HIGH); //Move Forward
    analogWrite(RightSpeed , 100); //Setting the engine speed forward
    digitalWrite(LeftMoveF, HIGH); //Turn
    digitalWrite(LeftMoveR, LOW); //Move Forward
}

/*****

// Forward Right -> I
else if (state == 'I') {
    analogWrite(LeftSpeed , 100); //Setting the engine speed forward
    digitalWrite(RightMoveF, LOW); //Turn
    digitalWrite(RightMoveR, HIGH); //Move Forward
    analogWrite(RightSpeed , vSpeed); //Setting the engine speed forward
    digitalWrite(LeftMoveF, HIGH); //Turn
    digitalWrite(LeftMoveR, LOW); //Move Forward
}

/*****

//Back -> B
else if (state == 'B') {
    analogWrite(LeftSpeed , vSpeed); //Setting the engine speed forward
    digitalWrite(RightMoveF, HIGH); //Turn
    digitalWrite(RightMoveR, LOW); //Move Forward

```

```

    analogWrite(RightSpeed , vSpeed); //Setting the engine speed forward
    digitalWrite(LeftMoveF, LOW); //Turn
    digitalWrite(LeftMoveR, HIGH); //Move Forward
}

/*****/

// Back Left -> H
else if (state == 'H') {
    analogWrite(LeftSpeed , vSpeed); //Setting the engine speed forward
    digitalWrite(RightMoveF, HIGH); //Turn
    digitalWrite(RightMoveR, LOW); //Move Forward
    analogWrite(RightSpeed , 100); //Setting the engine speed forward
    digitalWrite(LeftMoveF, LOW); //Turn
    digitalWrite(LeftMoveR, HIGH); //Move Forward
}

/*****/

// Back Right -> J
else if (state == 'J') {
    analogWrite(LeftSpeed , 100); //Setting the engine speed forward
    digitalWrite(RightMoveF, HIGH); //Turn
    digitalWrite(RightMoveR, LOW); //Move Forward
    analogWrite(RightSpeed , vSpeed); //Setting the engine speed forward
    digitalWrite(LeftMoveF, LOW); //Turn
    digitalWrite(LeftMoveR, HIGH); //Move Forward
}

/*****/

//Left -> L
else if (state == 'L') {
    analogWrite(RightSpeed , vSpeed); //Setting the engine speed forward

```

```

    analogWrite(LeftSpeed , vSpeed); //Setting the engine speed forward
    digitalWrite(RightMoveF, LOW); //Turn
    digitalWrite(RightMoveR, HIGH); //Move Forward
    digitalWrite(LeftMoveF, LOW); //Turn
    digitalWrite(LeftMoveR, HIGH); //Move Forward

}

/*****
//Right -> R
else if (state == 'R') {
    analogWrite(LeftSpeed , vSpeed); //Setting the engine speed forward
    analogWrite(RightSpeed , vSpeed); //Setting the engine speed forward
    digitalWrite(LeftMoveF, HIGH); //Turn
    digitalWrite(LeftMoveR, LOW); //Move Forward
    digitalWrite(RightMoveF, HIGH); //Turn
    digitalWrite(RightMoveR, LOW); //Move Forward
}

/*****Stop*****/
//Stop -> S
else if (state == 'S') {
    analogWrite(LeftSpeed , 0); //Setting the engine speed forward
    analogWrite(RightSpeed , 0); //Setting the engine speed forward
}
}

}

```

Server Code For TelegramBot & Mapping:

TelegramBot:

```
# -*- coding: utf-8 -*-
import sys
import telebot
reload(sys)
sys.setdefaultencoding('utf-8')
bot =
telebot.TeleBot('2114094831:AAHjjgeGw06jMntTW65ajqW1tixhV_VUOU')
@bot.message_handler(content_types=['document'])
def handle_docs_photo(message):
    try:
        file_info = bot.get_file(message.document.file_id)
        downloaded_file = bot.download_file(file_info.file_path)
        src = '/home/Glnoeeeo/mapping/doc/' + message.document.file_name
        f = open('doc/name.txt', 'w')
        f.write(message.document.file_name + '\n')
        f.close()
        with open(src, 'wb') as new_file:
            new_file.write(downloaded_file)
        bot.reply_to(message, "Save -> " + message.document.file_name)
    except Exception as e:
        bot.reply_to(message, str(e))
@bot.message_handler(commands=['map'])
def send_map(message):
    img = open('map/map.jpg', 'rb')
    bot.send_photo(message.chat.id, img)
bot.polling(none_stop=True, interval=0)
```

Mapping code:

```

import os
os.environ["SDL_VIDEODRIVER"] = "dummy"
import pygame
import math

pygame.init()
screen = pygame.Surface((600, 600))
screen.fill((240, 255, 255))
# Circle
YELLOW = (225, 225, 0)
RED = (225, 0, 0)
def circle(x, y, r, color):
    pygame.draw.circle(screen, color, (x, y), r)
##### start position of car
position_x = [300, 300, 300-100, 300-150, 350]
position_y = [300, 300-100, 300-150, 350, 350]
robot_x = 250
robot_y = 250
def to_x(r, f):
    x = r * math.cos(math.radians(f/6)) / 0.3
    return x
def to_y(r, f):
    y = r * math.sin(math.radians(f/6)) / 0.3
    return y
# Game loop
running = True
a = "
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
    try:
        fl = open('doc/name.txt', 'r')

```

```

last_line = f1.readlines()[-1]

f2 = open('doc/' + last_line[0:-1], 'r')
doc_angle = f2.readlines()
f_r = []
for line in doc_angle:
    first = line.find('=')
    r1 = float(line[first + 1:first + 7])
    first = line.find('#')
    f1 = float(line[first + 1:first + 6])
    f_r.append([r1, f1])
except ValueError:
    error = pygame.image.load('image/error.jpg')
    error.set_colorkey((255, 255, 255))
    screen.blit(error, (0, 0))
b = -1
angle = 540
color = [(0, 0, 0), (0, 255, 0), (0, 0, 255), (255, 255, 0), (255, 0, 0)]
for i in f_r:
    x = to_x(i[0], i[1] - angle) # 360 = 2159
    y = to_y(i[0], i[1] - angle) # 90 = x; x = 540
    if i[1] == 0.0:
        b += 1
        angle += 540
        robot_x = position_x[b]
        robot_y = position_y[b]
        circle(x + robot_x, y + robot_y, 2, color[b])
        circle(robot_x, robot_y, 15+b, YELLOW)
if last_line != a:
    a = last_line
    pygame.image.save(screen, "map/map.jpg")
    # screen.fill((240, 255, 255))
else:
    pygame.image.save(screen, "old_map/map.jpg")
    screen.fill((240, 255, 255))

```