KARNATAK LAW SOCIETY'S

GOGTE INSTITUTE OF TECHNOLOGY

UDYAMBAG BELAGAVI -590008

KARNATAKA, INDIA.

A Course Project Report on

Omnidirectional robot

Submitted for the requirements of 3rd semester B.E. in CSE

for "**PYTHON PROGRAMMING (18CSL46)**"

Submitted by

NAME	USN
SHIVANI BANKE	2GI20CS140
SHRADHA MALLIKARJUN PATIL	2GI20CS144
SRUSHTI B MUDENNAVAR	2GI20CS158
YASH HEREKAR	2GI20CS184

Dr. Manjula Ramannavar

Asst. Prof., Dept. of CSE

Academic Year 2021 - 2022 (Even semester)

2021-2022

KARNATAK LAW SOCIETY'S

GOGTE INSTITUTE OF TECHNOLOGY

UDYAMBAG BELAGAVI -590008

KARNATAKA, INDIA.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Course Project work titled **"OMNIDIRECTIONAL ROBOT"** carried out by **Student SHIVANI BANKE, SHRADHA MALLIKARJUN PATIL SRUSHTI B MUDENNAVAR AND YASH HEREKAR** bearing **USNs: 2GI20CS140, 2GI20CS144, 2GI20CS158 AND 2GI20CS184** for **PYTHON PROGRAMMING (18CSL46) Integrated Course** is submitted in partial fulfilment of the requirements for 4th semester B.E. in **COMPUTER SCIENCE AND ENGINEERING,** Visvesvaraya Technological University, Belagavi. It is certified that all corrections/ suggestions indicated have been incorporated in the report. The course project report has been approved as it satisfies the academic requirements prescribed for the said degree.

Date: Signature of guide

Place: Belagavi Dr. Manjula Ramannavar

Asst., Prof.,

Dept of CSE KLS Gogte Institute Of Technology, Belagavi

KARNATAK LAW SOCIETY'S

GOGTE INSTITUTE OF TECHNOLOGY

UDYAMBAG BELAGAVI -590008

KARNATAKA, INDIA.

Academic Year 2021 - 2022 (Even semester)

Semester: IV

Course: Python Programming (18CSL46) Integrated Course

Rubrics for evaluation of Course Project

Student Name :

Student USN :

Student Branch :

S.No	Project Component	Max. Marks	Marks Earned
1	Relevance of the project and its objectives	02	
2	Tools/Framework used	01	
3	Methodology / Design	02	
4	Implementation and Results	03	
5	Project Report	02	
	Total	**10**	

ACKNOWLEDGMENTS

We would want to convey our heartfelt appreciation and gratitude to our professor Dr. Manjula Ramannavar for providing us with a chance to work on this wonderful project which also aided us in conducting an extensive study and also learning a bunch of new topics.

We are very grateful to Manjula ma’am, as she has constantly guided us and supervised us throughout this course project and helped us improve our project as well as our skills.

Secondly, we would like to thank HOD, and our principal for providing us this opportunity in our curriculum, to work as a team, hence this activity turned out to be a great team building project,

Lastly, we would like to thank our friends, for their assistance in the completion of the project, in a short period of time.

As, this project was useful in terms of expanding our knowledge and abilities.

Thank you to everyone who has helped.

ABSTRACT

This paper deals with programming an omnidirectional robot in python and interfacing it with a GUI designed in Tkinter. An omnidirectional robot is a four wheel robot which has 2 degrees of freedom and 7 degree of movement. It has a special wheels called mecanum wheels which enable it to manoeuver like a crab.

TABLE OF CONTENTS

OMNIDIRECTIONAL ROBOT

1. PROBLEM STATEMENT

Develop a python program to control a robot and also develop a GUI to interface with it

2. OBJECTIVE AND SCOPE OF THE PROJECT

- To program raspberry pi pico using micro-python.
- To design and implement a GUI application using Tkinter.
- To Write functions such as:
  - **front:** Moves the robot forward in y axis.
  - **back:** Moves the robot backward in the y axis.
  - **left:** Turns the robot in the left direction.
  - **right:** Turns the robot in the right direction.
  - **leftShift:** Slides the robot to left in the x axis.
  - **rightShift:** Slides the robot to right in the x axis.
  - **printInfo:** Prints system name and embedded operating system name
  - **changeSpeed:** Updates the global speed variable ranging from 1-10 speeds
  - **stop:** Stop all the motors
- To write GUI functions similar to the the one the robot has.

3. TOOLS/FRAMEWORK USED

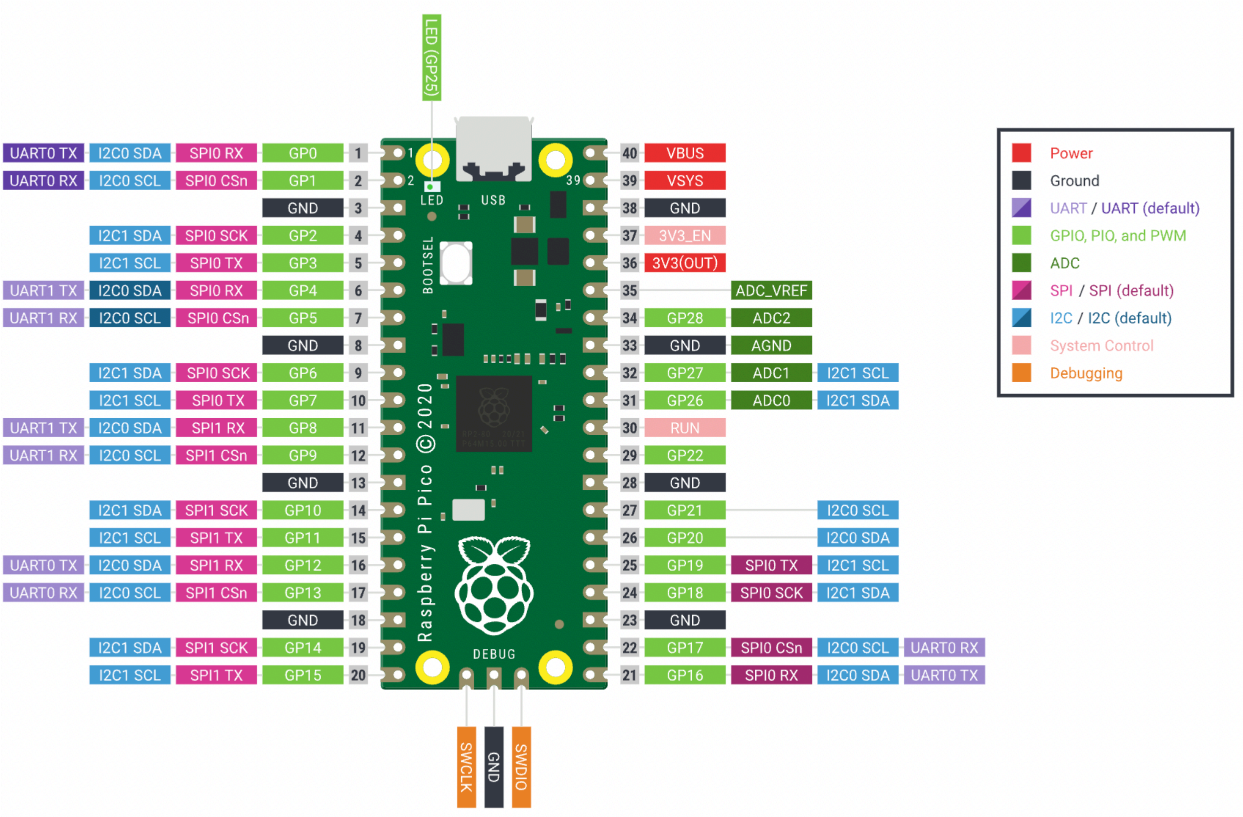
Name	**Quantity**	***Description***
Raspberry pi pico RP2040 microcontroller	**1**	*control the motors and sensors*
12v DC Geared Motors	**4**	*300 rpm*
Dual tb6612fng	**2**	*H bridge for motor control*
HM-10 BLE	**1**	*Bluetooth 4.0 CC2541 wireless module*
Lipo Battery	**1**	*Power supply 2200 mah*
Mecanum wheels	**4**	*Omnidirectional wheels*

RASPBERRY PI PICO MICROCONTROLLER

Specifications of raspberry pi Pico

Raspberry Pi Pico is a low-cost, high-performance microcontroller board with flexible digital interfaces. Key features include:

- RP2040 microcontroller chip designed by Raspberry Pi in the United Kingdom
- Dual-core Arm Cortex M0+ processor, flexible clock running up to 133 MHz
- 264kB of SRAM, and 2MB of on-board flash memory
- USB 1.1 with device and host support
- Low-power sleep and dormant modes
- Drag-and-drop programming using mass storage over USB
- 26 × multi-function GPIO pins
- 2 × SPI, 2 × I2C, 2 × UART, 3 × 12-bit ADC, 16 × controllable PWM channels
- Accurate clock and timer on-chip
- Temperature sensor
- Accelerated floating-point libraries on-chip
- 8 × Programmable I/O (PIO) state machines for custom peripheral support



GPIO Pinout

Dual TB6612FNG motor driver

Specifications of Dual tb6612fng motor driver

The TB6612FNG motor driver can control up to two DC motors at a constant current of 1.2A (3.2A peak). Two input signals (IN1 and IN2) can be used to control the motor in one of four function modes - CW, CCW, short-brake, and stop. The two motor outputs (A and B) can be separately controlled, the speed of each motor is controlled via a PWM input signal with a frequency up to 100kHz. The STBY pin should be pulled high to take the motor out of standby mode. Some of the features are:

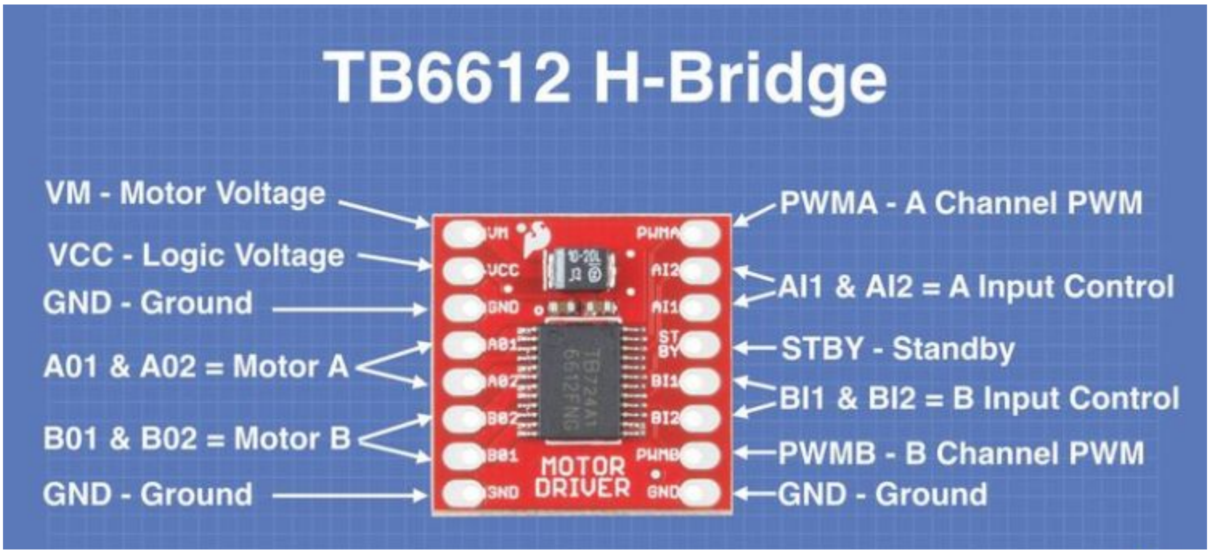
• Power supply voltage: VM = 15 V(Max)

• Output current: OUT = 1.2 A(average) and 3.2 A (peak)

• Standby (Power save) system

• CW/CCW/short brake/stop function modes

• Built-in thermal shutdown circuit and low voltage detecting circuit



HM-10 BLE Bluetooth 4.0 CC2540 wireless module

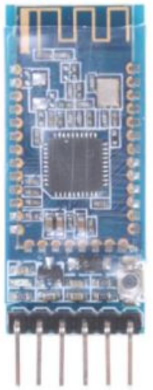
Specifications of HM-10 Bluetooth module:

The HM-10 is a readily available Bluetooth 4.0 module used for establishing wireless data communication. The module is designed by using the Texas Instruments CC2540 or CC2541 Bluetooth low energy (BLE) System on Chip (SoC).

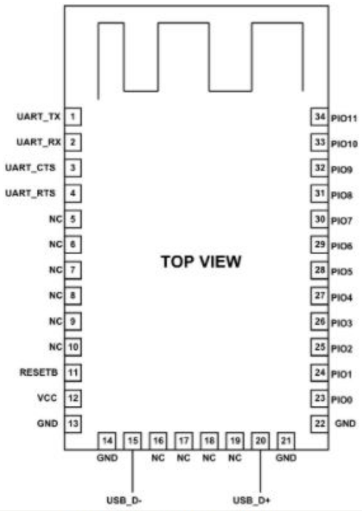
Features:

- BT Version: Bluetooth Specification V4.0 BLE
- Working frequency: 2.4GHz ISM band
- Modulation method: GFSK(Gaussian Frequency Shift Keying)

- RF Power: -23dbm, -6dbm, 0dbm, 6dbm
- Speed: Asynchronous: 2-6K Bytes Synchronous: 2-6K Bytes
- Security: Authentication and encryption
- Service: 0xFFE0 (Modifiable use AT+UUID command)
- Characteristic: 0xFFE1 (Modifiable use AT+UUID command)
- Characteristic: Notify and Write (Modifiable use AT+UU UID command)
- Power: +2.5V~3.3VDC 50mA
- Power: Active state 8.5mA; Sleep state 50~200uA
- Working temperature: –20 ~ +95 Centigrade
- Size: HM-10 27mm x 13mm x 2.2 mm



HM-10 Bluetooth Module

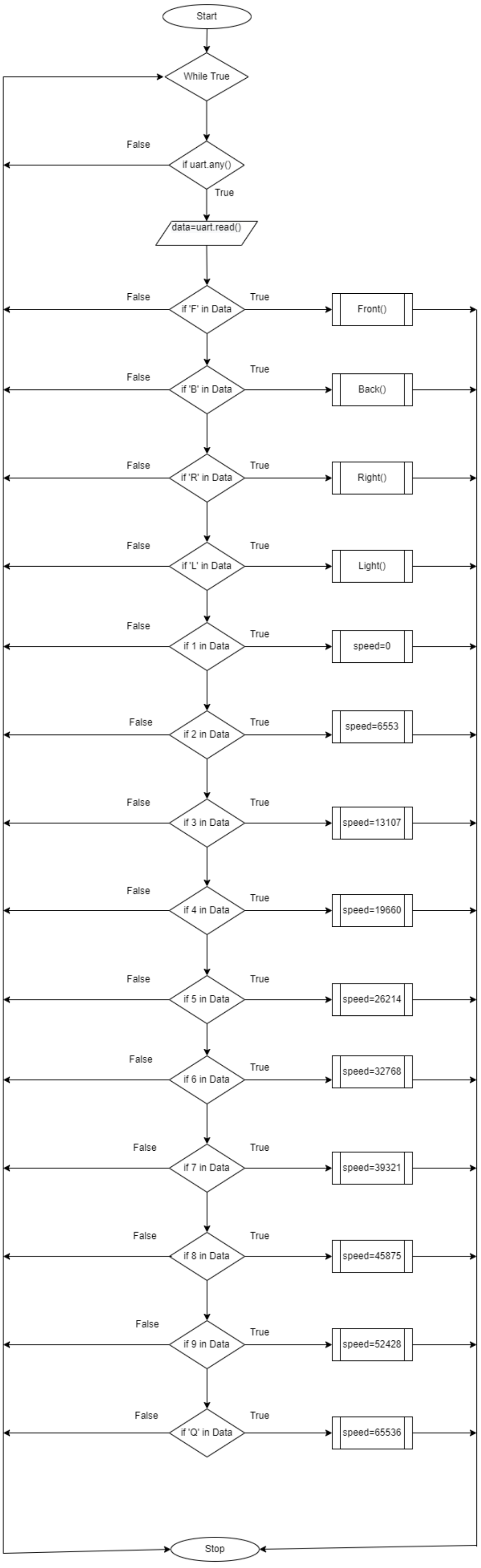


HM-10 Bluetooth Module Pinout

4. METHODOLOGY/DESIGN

FUNCTIONS	WHEEL 1	WHEEL 2	WHEEL 3	WHEEL 4
FRONT	↑	↑	↑	↑
BACK	↓	↓	↓	↓
LEFTTURN	↓	↓	↑	↑
RIGHTTURN	↑	↑	↓	↓
LEFTSHIFT	↓	↑	↑	↓
RIGHTSHIFT	↑	↓	↓	↑
DIAGONALLEFTFRONT	↑	-	-	↑
DIAGONALRIGHTFRONT	-	↑	↑	-
DIAGONALLEFTBACK	-	↓	↓	-
DIAGONALRIGHTBACK	↓	-	-	↓
STOP	-	-	-	-

FLOWCHART

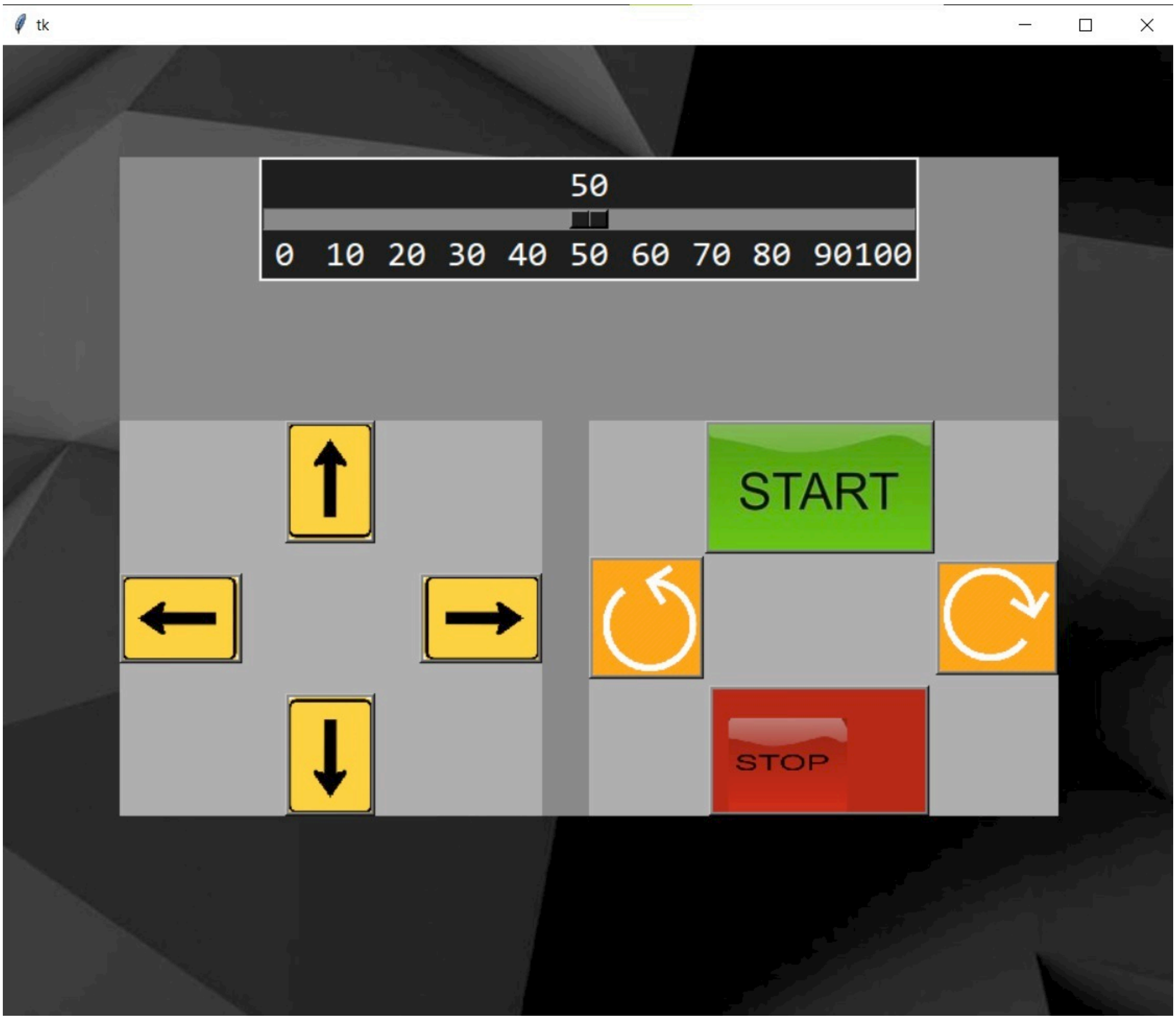

Start
While True
False
if uart.any()
True
data=uart.read()
False
if 'F' in Data
True
Front()
False
if 'B' in Data
True
Back()
False
if 'R' in Data
True
Right()
False
if 'L' in Data
True
Light()
False
if 1 in Data
True
speed=0
False
if 2 in Data
True
speed=6553
False
if 3 in Data
True
speed=13107
False
if 4 in Data
True
speed=19660
False
if 5 in Data
True
speed=26214
False
if 6 in Data
True
speed=32768
False
if 7 in Data
True
speed=39321
False
if 8 in Data
True
speed=45875
False
if 9 in Data
True
speed=52428
False
if 'Q' in Data
True
speed=65536
Stop

5. CODE

```
#https://github.com/1337encrypted/Raspberry-pi-pico/blob/main/pythonbot.py
from machine import Pin, PWM, UART
from utime import sleep
import uos

#create the uart
#uart = UART(id,baudrate,tx,rx)
uart1 = UART(0, baudrate=9600, tx=Pin(0), rx=Pin(1))

#defining pins
led = Pin(25,Pin.OUT)

AIN1 = Pin(2,Pin.OUT)
AIN2 = Pin(3, Pin.OUT)
PWMA1 = PWM(Pin(4))

BIN1 = Pin(5,Pin.OUT)
BIN2 = Pin(6, Pin.OUT)
PWMB1 = PWM(Pin(7))

AIN3 = Pin(8,Pin.OUT)
AIN4 = Pin(9, Pin.OUT)
PWMA2 = PWM(Pin(10))

BIN3 = Pin(11,Pin.OUT)
BIN4 = Pin(12, Pin.OUT)
PWMB2 = PWM(Pin(13))

STBY1 = Pin(14, Pin.OUT)
STBY2 = Pin(15, Pin.OUT)

PWMA1.freq(1000)
PWMB1.freq(1000)
PWMA2.freq(1000)
PWMB2.freq(1000)

speed = 65536

def printinfo():      #Print system name
    print("-"*50)
    print("picoTerm>")
    print(uos.uname())
    led.value(1)

def front():
    AIN1.value(1)
    AIN2.value(0)
    BIN1.value(1)
    BIN2.value(0)
    AIN3.value(1)
    AIN4.value(0)
    BIN3.value(1)
    BIN4.value(0)
    #Applying voltage to the motor using pulse width modulation (PWM)
    PWMA1.duty_u16(speed)
    PWMB1.duty_u16(speed)
    PWMA2.duty_u16(speed)
    PWMB2.duty_u16(speed)

def back():
    AIN1.value(0)
    AIN2.value(1)
    BIN1.value(0)
    BIN2.value(1)
    AIN3.value(0)
    AIN4.value(1)
    BIN3.value(0)
    BIN4.value(1)
    #Applying voltage to the motor using pulse width modulation (PWM)
    PWMA1.duty_u16(speed)
    PWMB1.duty_u16(speed)
    PWMA2.duty_u16(speed)
    PWMB2.duty_u16(speed)

def left():
    AIN1.value(0)
    AIN2.value(1)
    BIN1.value(0)
    BIN2.value(1)
    AIN3.value(1)
    AIN4.value(0)
    BIN3.value(1)
    BIN4.value(0)
    #Applying voltage to the motor using pulse width modulation (PWM)
    PWMA1.duty_u16(speed)
    PWMB1.duty_u16(speed)
    PWMA2.duty_u16(speed)
    PWMB2.duty_u16(speed)

def right():
    AIN1.value(1)
    AIN2.value(0)
    BIN1.value(1)
    BIN2.value(0)
    AIN3.value(0)
    AIN4.value(1)
    BIN3.value(0)
    BIN4.value(1)
    #Applying voltage to the motor using pulse width modulation (PWM)
    PWMA1.duty_u16(speed)
    PWMB1.duty_u16(speed)
    PWMA2.duty_u16(speed)
    PWMB2.duty_u16(speed)

def leftShift():
    AIN1.value(0)
    AIN2.value(1)
    BIN1.value(1)
    BIN2.value(0)
    AIN3.value(1)
    AIN4.value(0)
    BIN3.value(0)
    BIN4.value(1)
    #Applying voltage to the motor using pulse width modulation (PWM)
    PWMA1.duty_u16(speed)
    PWMB1.duty_u16(speed)
    PWMA2.duty_u16(speed)
    PWMB2.duty_u16(speed)

def rightShift():
    AIN1.value(1)
    AIN2.value(0)
    BIN1.value(0)
    BIN2.value(1)
    AIN3.value(0)
    AIN4.value(1)
    BIN3.value(1)
    BIN4.value(0)
    #Applying voltage to the motor using pulse width modulation (PWM)
    PWMA1.duty_u16(speed)
    PWMB1.duty_u16(speed)
    PWMA2.duty_u16(speed)
    PWMB2.duty_u16(speed)

def stop():
    AIN1.value(0)
    AIN2.value(0)
    BIN1.value(0)
    BIN2.value(0)
    AIN3.value(0)
    AIN4.value(0)
    BIN3.value(0)
    BIN4.value(0)
    led.value(0)
    #Applying voltage to the motor using pulse width modulation (PWM)
    PWMA1.duty_u16(0)
    PWMB1.duty_u16(0)
    PWMA2.duty_u16(0)
    PWMB2.duty_u16(0)
    print("Stop: 0")

def main():
    printinfo()                #printing the board details
    global speed
    while True:
        if uart1.any():         #Checking if data available
            data = uart1.read()#Getting data
            #data = str(data)
            #stop()              #Stop
            #print(data)
            if('F' in data):   #Forward
                front()
                print("Front:",speed)
            elif('B' in data): #Backward
                back()
```

```python
                print("Back: ",speed)
            elif('R' in data): #Turn Right
                right()
                print("Right: ",speed)
            elif('L' in data): #Turn Left
                left()
                print("Left: ",speed)
            elif('I' in data): #Right shift
                rightShift()
                print("Right Shift: ",speed)
            elif('G' in data): #Left shift
                leftShift()
                print("Left Shift: ",speed)
            elif('0' in data): #speed = 115
                speed = 0
            elif('1' in data): #speed = 130
                speed = 6553
            elif('2' in data): #speed = 143
                speed = 13107
            elif('3' in data): #speed = 157
                speed = 19660
            elif('4' in data): #speed = 170
                speed = 26214
            elif('5' in data): #speed = 185
                speed = 32768
            elif('6' in data): #speed = 200
                speed = 39321
            elif('7' in data): #speed = 213
                speed = 45875
            elif('8' in data): #speed = 227
                speed = 52428
            elif('9' in data): #speed = 240
                speed = 58982
            elif('q' in data): #speed = 255
                speed = 65536
            else:
                stop()    #Stop

if __name__=='__main__':
    main()
```

6. SCREENSHOTS



7. APPLICATION AND FUTURE ENHANCEMENT

- Industrial applications - single task robots. Pick and place, etc
- Military applications - autonomous unmanned vehicles.
- Space exploration - Curiosity rover on Mars, navigates the terrain on mars and sends feedback back to earth.
- Healthcare - The da Vinci Surgical Robot, performs surgeries.

8. CONCLUSION

During the course of the project we learnt how to program a microcontroller using python and also design a GUI and connect to the robot using the GUI interface.

9. REFERENCES

- https://projects.raspberrypi.org/en/projects/getting-started-with-the-pico
- https://docs.micropython.org/en/latest/rp2/quickref.html#uart-serial-bus
- https://datasheets.raspberrypi.com/pico/pico-datasheet.pdf
- https://datasheets.raspberrypi.com/pico/Pico-R3-A4-Pinout.pdf
- https://github.com/1337encrypted/Raspberry-pi-pico/blob/main/pythonbot.py