AUTONOMOUS MOBILE ROBOT

-G DURGAPRASAD

CONTENT

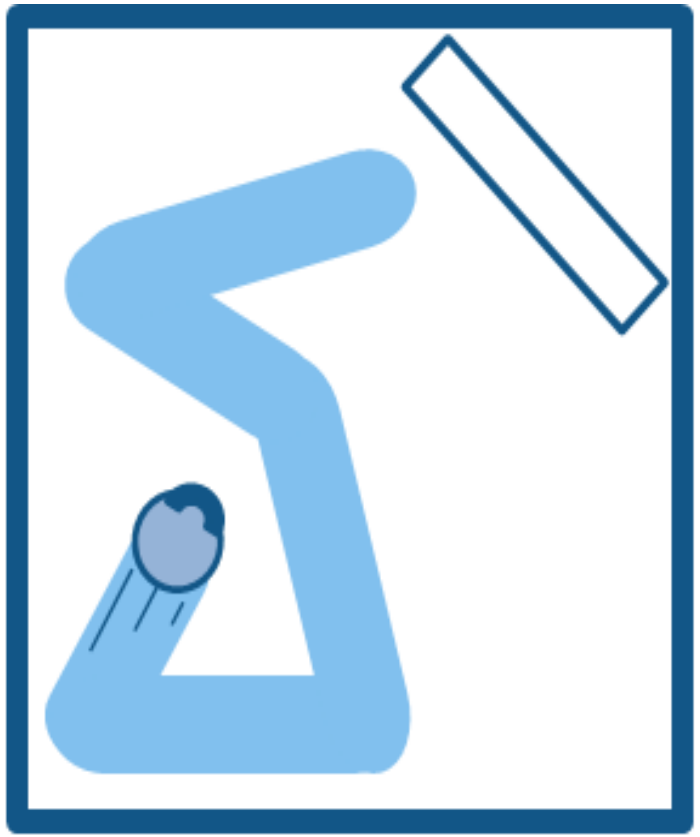
- SLAM
- Visual SLAM
- Methodology
- Raspberry pi connections
- Working
- Future scope

SLAM

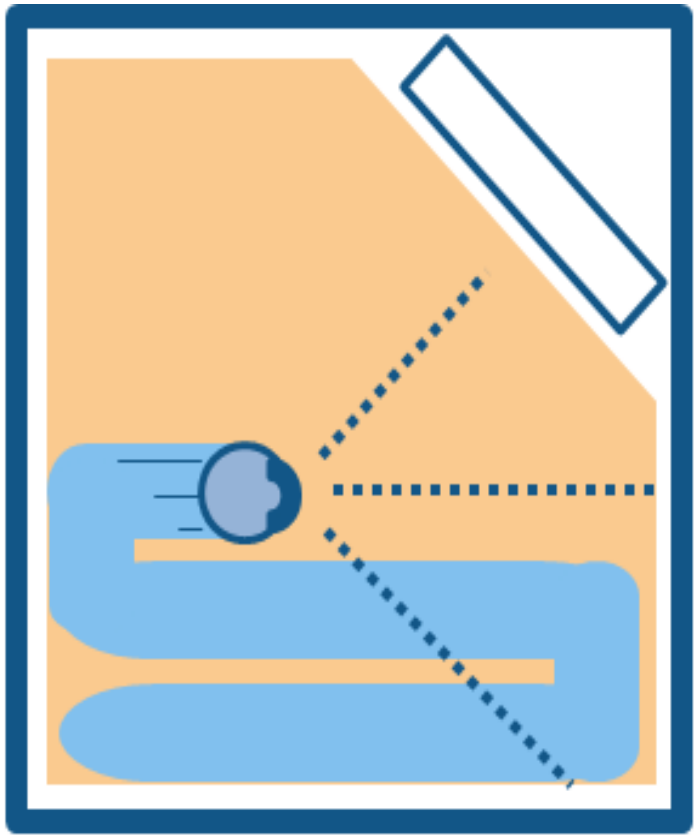
- SLAM stands for Simultaneous Localization And Mapping.
- It refers to the computational problem in autonomous vehicles and robots of constructing or updating a map of an unknown environment while simultaneously keeping track of its own position or location within an environment.
- It is used to over come localization and mapping problem similar to chicken or egg problem.

Example

- **Consider a home robot vacuum without SLAM, it will just move randomly within a room and may not be able to clean the entire floor surface.**
- **In addition, this approach uses excessive power, so the battery will run out more quickly.**
- **On the other hand, robots with a SLAM algorithm can use information such as the number of wheel revolutions and data from cameras and other imaging sensors to determine the amount of movement needed. This is called localization.**
- **The robot can also simultaneously use the camera and other sensors to create a map of the obstacles in its surroundings and avoid cleaning the same area twice. This is called mapping.**



Without SLAM:
Cleaning a room randomly.



With SLAM:
Cleaning while understanding the room's layout.

Visual SLAM

- Visual SLAM (Simultaneous Localization and Mapping) is a technique used in robotics to enable a device, such as a robot or a camera, to simultaneously localize itself within an unknown environment and construct a map of that environment using visual information.
- Unlike traditional SLAM methods that rely on specialized sensors like LIDAR or depth cameras, visual SLAM operates primarily on visual data captured by cameras, making it more lightweight and versatile.

Methodology

1. **Feature Extraction:**
   - The camera captures images of the environment as it moves, providing a stream of visual data.
   - Feature extraction algorithms analyze these images to identify distinctive points, corners, or regions. These features are selected because they are easily identifiable and can be tracked across frames.
   - Features are typically detected based on local intensity gradients, textures, or other characteristics present in the images.

2.Feature Matching:

- After extracting features from consecutive frames, the system matches corresponding features between frames to establish correspondences.
- Matching involves comparing the descriptors of features in one frame with those in another frame to find the best matches.
- Feature matching helps track how these features move in the camera's field of view over time, providing information about the camera's motion relative to its previous position.

3.Motion Estimation (Visual Odometry):

- Using the correspondences between features in consecutive frames, the system estimates the camera's motion (translation and rotation) between frames. This process is known as visual odometry.

- Visual odometry algorithms analyze the motion of tracked features to compute incremental updates to the camera's pose, providing information about how the camera has moved relative to its previous position.
- The estimation of camera motion is crucial for updating the camera's pose and constructing an accurate map of the environment.

4.Map Construction:

- As the camera moves through the environment, it captures images and provides visual input to the SLAM system.
- The system constructs a map of the environment using the tracked features and the estimated camera poses.
- Each frame captured by the camera contributes to the map construction process by adding new features and updating the camera's pose.
- The map consists of keyframes (representative frames) and the 3D positions of the features observed in those frames.

5.Loop Closure Detection:

- Loop closure detection aims to correct for accumulated drift by recognizing when the camera revisits a previously observed location.
- This step involves comparing the current frame with previously visited areas in the map to detect similar visual patterns or features.
- Loop closure detection helps improve map consistency by refining the camera poses and updating the map to ensure global consistency across the entire map.

6.Map Optimization:

- After loop closures are detected, the system optimizes the map and camera poses using bundle adjustment techniques.

- Bundle adjustment refines the map and camera trajectories to minimize errors and ensure global consistency across the entire map.
- This optimization step improves the accuracy of the map and the camera poses, resulting in a more reliable representation of the environment.

7.Re-localization:

- Re-localization allows the system to recover quickly from tracking failures or localization losses by matching the current camera image with features stored in the map.
- By determining the best matches between the current image and the features in the map, the system can accurately estimate the camera's pose and continue the mapping process.

Raspberry pi 3b+

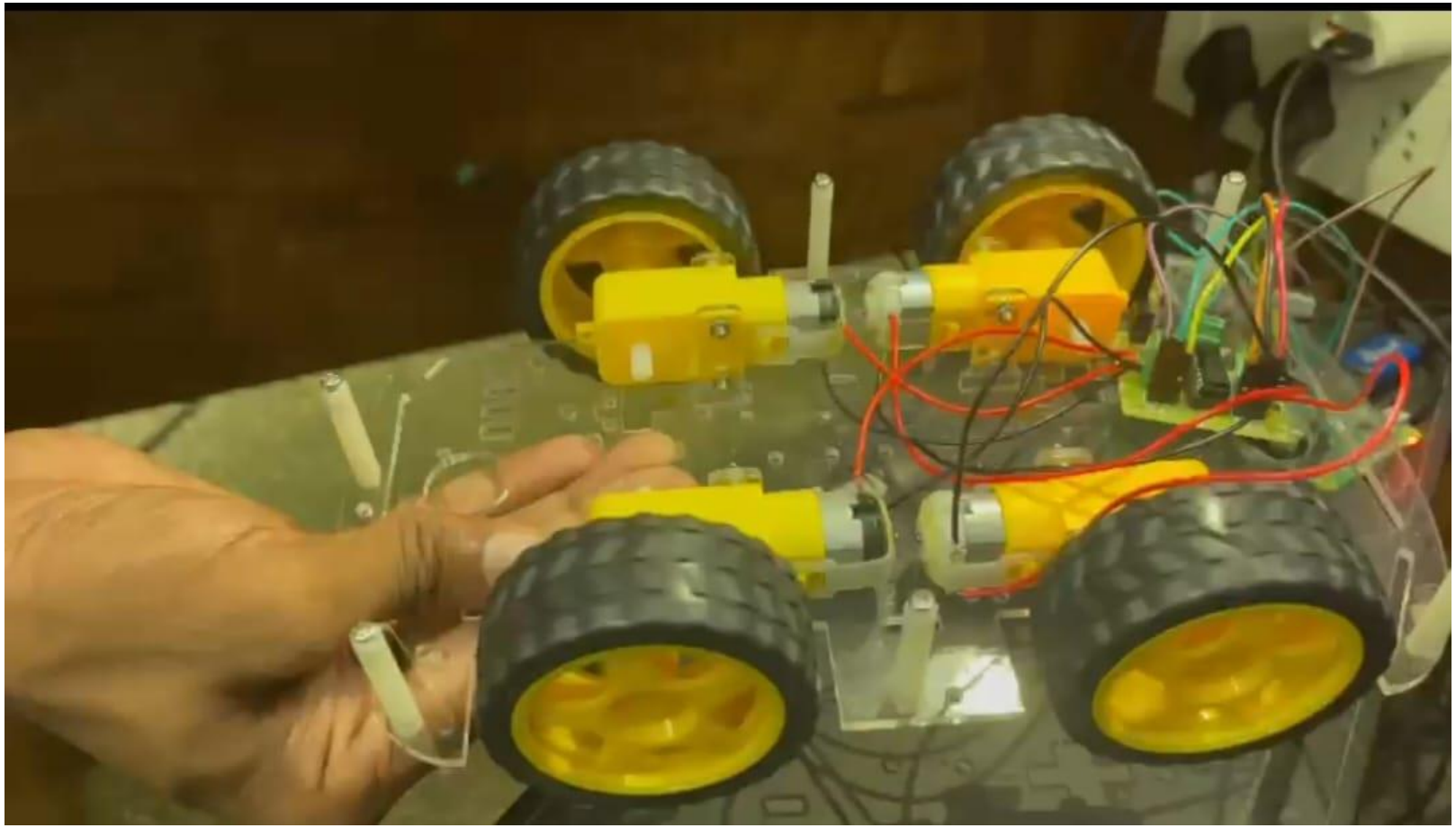
- The Raspberry Pi 3 Model B+ is an enhanced version of the Raspberry Pi 3 Model B, offering improved performance and connectivity features.
- It is powered by a 1.4GHz 64-bit quad-core ARM Cortex-A53 processor and includes 1GB of LPDDR2 SDRAM.
- The key features of the Raspberry Pi 3 Model B+ include dual-band 2.4GHz and 5GHz IEEE 802.11ac wireless LAN (Wi-Fi) for faster wireless networking, Bluetooth 4.2/BLE support, and Gigabit Ethernet connectivity.

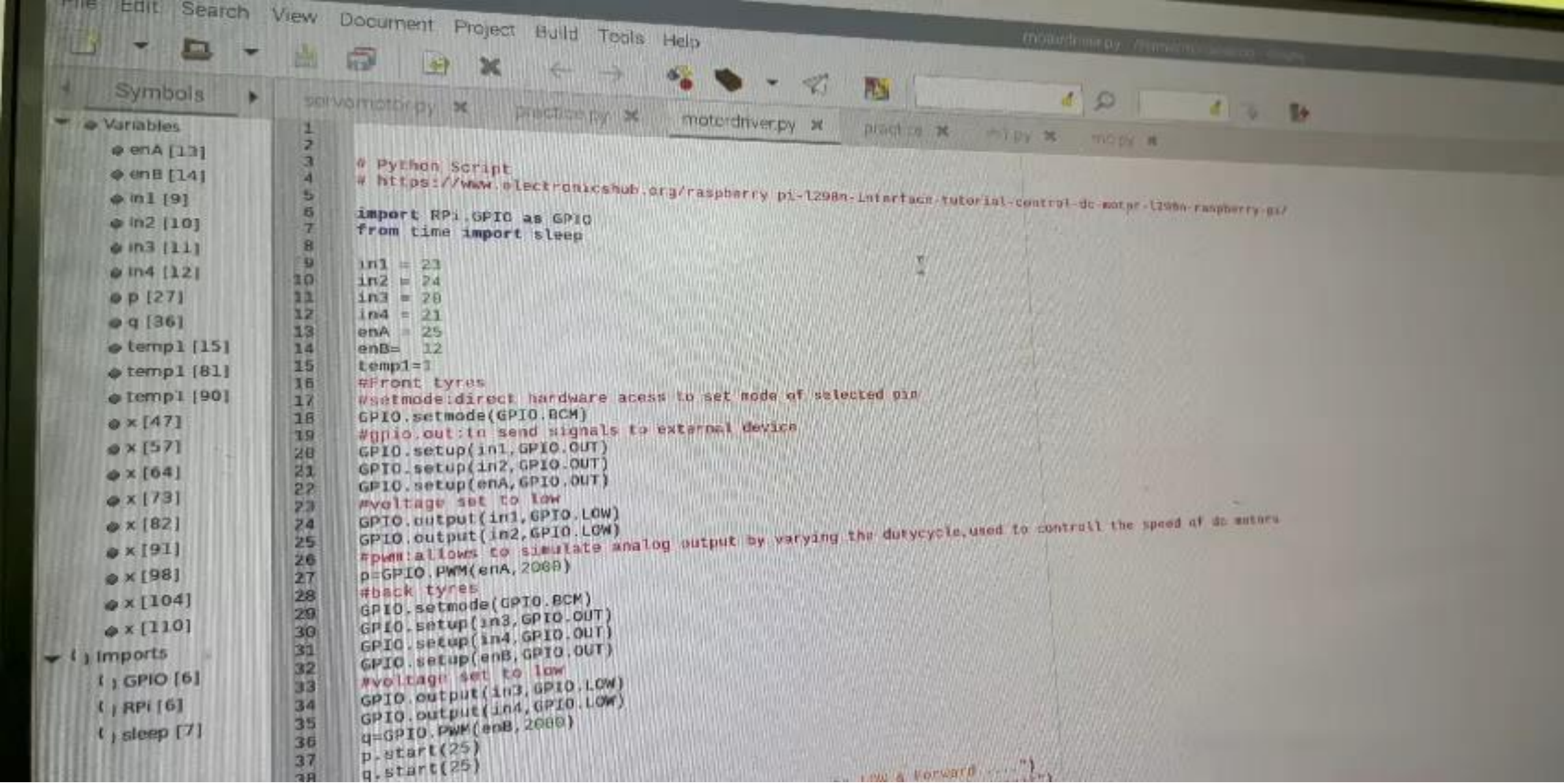
Working

- Connect the motor driver to the Raspberry Pi 3B+ using jumper wires.
- Typically, you'll connect GPIO pins from the Raspberry Pi to the control inputs of the motor driver (e.g., input pins for direction control and PWM signal for speed control).
- Ensure that the motor driver is properly powered according to its specifications
- Connect the DC motor to the output terminals of the motor driver.
- Make sure to observe polarity (positive and negative terminals) to avoid damaging the motor.

- Open a text editor on the Raspberry Pi (e.g., Geany IDE or nano) and write a Python script to control the motor.
- Use the GPIO library (e.g., RPi.GPIO) to interface with the GPIO pins connected to the motor driver.
- Import the necessary libraries to control the motor (e.g., PWM signal for speed control).

Connection of motordriver and motors




File Edit Search View Document Project Build Tools Help
Symbols
Variables
enA [13]
enB [14]
in1 [9]
in2 [10]
in3 [11]
in4 [12]
p [27]
q [36]
temp1 [15]
temp1 [81]
temp1 [90]
x [47]
x [57]
x [64]
x [73]
x [82]
x [91]
x [98]
x [104]
x [110]
imports
GPIO [6]
RPi [6]
sleep [7]
servomotor.py
practice.py
motordriver.py
practice
1 # Python Script
2 # https://www.electronicshub.org/raspberry pi-l298n-interface-tutorial-control-dc-motor-l298n-raspberry-pi/
3 import RPi.GPIO as GPIO
4 from time import sleep
5
6 in1 = 23
7 in2 = 24
8 in3 = 20
9 in4 = 21
10 enA = 25
11 enB= 12
12 temp1=1
13 #Front tyres
14 #setmode:direct hardware acess to set mode of selected pin
15 GPIO.setmode(GPIO.BCM)
16 #gpio.out:to send signals to external device
17 GPIO.setup(in1,GPIO.OUT)
18 GPIO.setup(in2,GPIO.OUT)
19 GPIO.setup(enA,GPIO.OUT)
20 #voltage set to low
21 GPIO.output(in1,GPIO.LOW)
22 GPIO.output(in2,GPIO.LOW)
23 #pwm:allows to simulate analog output by varying the dutycycle,used to controll the speed of dc motors
24 p=GPIO.PWM(enA,2000)
25 #back tyres
26 GPIO.setmode(GPIO.BCM)
27 GPIO.setup(in3,GPIO.OUT)
28 GPIO.setup(in4,GPIO.OUT)
29 GPIO.setup(enB,GPIO.OUT)
30 #voltage set to low
31 GPIO.output(in3,GPIO.LOW)
32 GPIO.output(in4,GPIO.LOW)
33 q=GPIO.PWM(enB,2000)
34 p.start(25)
35 q.start(25)

Future scope

- We will try to add servomotor for turning mechanism.
- We will add camera for collecting surrounding data in form of images.
- Then apply some filters like Kalman, extended Kalman filter on collected image data to generate visual odometry.
- And will try to implement SLAM on visual odometry to generate its map, and location in map and use the turning mechanism if any obstacles in map or path to generate a perfect or beneficial path for a robot

THANK YOU