Android based WI-FI controlled robot for Store Management

Project report submitted in partial fulfillment
of the requirements for the degree of

Bachelor of Technology
in
Mechanical-Mechatronics Engineering

by

Ashish Sharma - 20ume013
Ronak Surana - 20uec105

Under Guidance of
Dr. Atul Mishra

Department of Mechanical-Mechatronics Engineering
The LNM Institute of Information Technology, Jaipur

November 2023

The LNM Institute of Information Technology
Jaipur, India

CERTIFICATE

This is to certify that the project entitled Android based WI-FI controlled robot using Raspberry Pi for Store Management, submitted by Ronak Surana (20uec105), Ashish Sharma (20ume013) in partial fulfillment of the requirement of degree in Bachelor of Technology (B. Tech), is a bonafide record of work carried out by them at the Department of Mechanical-Mechatronics Engineering, The LNM Institute of Information Technology, Jaipur, (Rajasthan) India, during the academic session 2023-2024 under my supervision and guidance and the same has not been submitted elsewhere for award of any other degree. In my/our opinion, this report is of standard required for the award of the degree of Bachelor of Technology (B. Tech).

Date

Adviser: Prof. Atul Mishra

Acknowledgments

I would like to express my sincere gratitude towards **Dr. Atul Mishra** our project guide, for their patient guidance, enthusiastic encouragement, and useful critiques on this research work. Without his invaluable guidance, this work would never have been successful .

I would also like to thank **Mr. Udayveer Singh** for helping us by providing detailed knowledge on equipment's and machinery which we need in this project and all the time being ready to resolve our queries.

We would also to express our gratitude towards our batchmates for being an unconditional source of motivation and support and helping us in whatever form they can.

We would like to thank God Almighty and our parents for giving us the strength, knowledge, ability, and opportunity to undertake this project and to persevere and complete it satisfactorily. Without their blessings, this achievement would not have been possible.

Abstract

In today's times, robotic technology has come a long way from not just getting constrained to big industries, but also the small businesses using them to made their work easier, less time-consuming, and required less labour.
With the advancement of technology, the cost of robots has been reduced, and there Functionalities have increased, making them more accessible. Especially in developing countries, as the lifestyle of people is getting better Day by day, they are becoming more conscious about health and food types. This has led to the development of more and more shopping complexes. The shopping complexes due to the market demand are getting forced to put more and more variety of products. Checking the availability of products multiple times a day manually in big stores can be very time-consuming, and the chances of errors are huge, so there is a need to find a solution to this problem.
Through our work, we are trying to develop a robot that will be equipped with a camera. that will show pictures of the products placed on the shelves. The robot may carry a shaft to adjust the camera up and down according to the height of the shelf. The picture can be processed either an on-board or remote server using algorithms. We are on board here. The robot will identify various products, detect incidents like out-of-stock (OOS) and misplaced items.
As a shopping complex is a crowded place, an autonomous robot may create a problem for customers, so it needs to be controlled by someone sitting at the control station. For this We have developed a mobile app that can help control the robot left, right, and forward. backward, stop, and control speed up and down.
As we identify the product, we will update the quantity in the database with some additional information like which shelf it is placed upon.

Contents

List of Figures

Chapter 1

Introduction

1.1 The Area of Work

In this project we are trying to develop a android based wifi controlled robot for store management which has a rpi camera enabled with the help of which we are doing image processing. The image processed is used to maintain database of store and hence we are maintaining the data present in store for better data analysis.

1.2 Tools Used

1.2.1 Raspberry Pi

Raspberry Pi is a small lost cost computer that was originally designed to teach computer science to students. It is about the size of a credit card and we can connect a monitor, keyboard, and mouse to form a single computer. It is capable of running OS like Windows, ubuntu and has its own operating system ROS.We can install OS in a memory card and then insert that card in the slot given on the RPI board. To make it run just install Putty and VNC Viewer on your PC. Connect your PC and RPI with LAN cable. Now type raspberrypi.local in putty and press enter. Now type username, press enter, type password and then press enter. Now open VNC viewer and type raspberrypi.local in the search bar and press enter. Now your RPI desktop will open. See the pictures at the bottom for the steps.
RPI has 40 pins.There functionality are:

- 3.3V (on 2 pins)
- 5V (on 2 pins)
- Ground (on 8 pins)

- General purpose input and output
- PWM (pulse width modulation)
- I2C
- I2S
- SPI
- Serial

We can connect these pins to modules through jumper wires.[3]

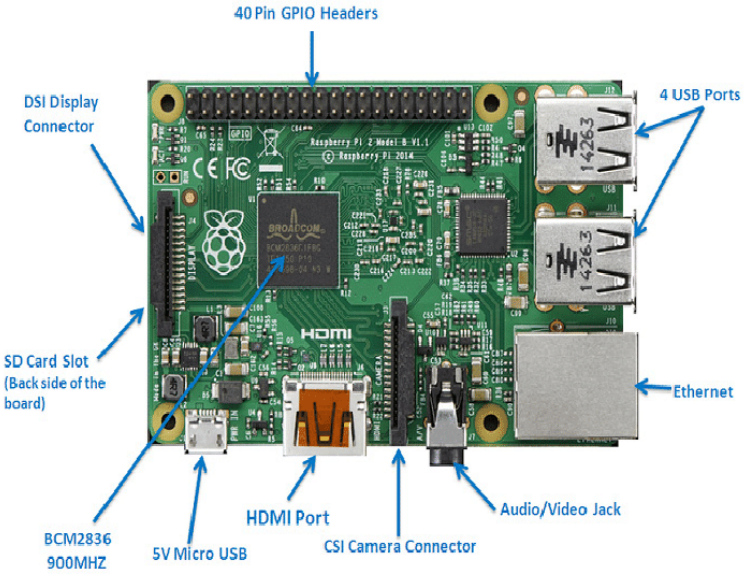


FIGURE 1.1: Raspberry Pi Board (Source Google Images)

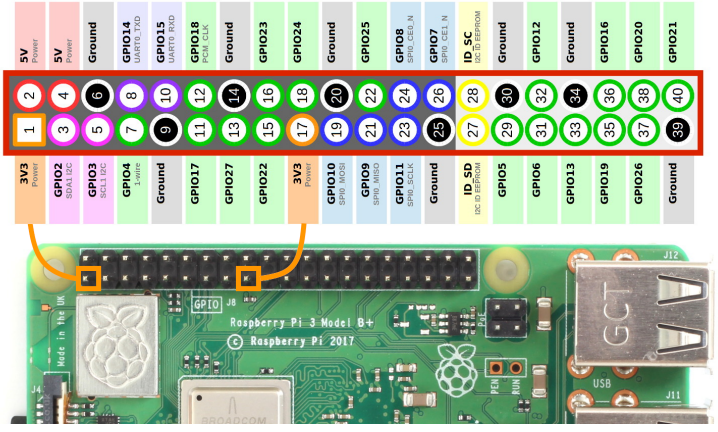


FIGURE 1.2: Raspberry Pi GPIO pin layout (Source Google Images)

1.2.2 L298N motor driver

A 12V battery is used to power the motor driver. We can connect two dc motors to it. We connect it to RPI using jumper wires. Its pins configuration is as follows:

- En1 enable for Motor1 connect to GPIO pin.
- A1 motor1 input 1 connect to GPIO pin.
- B1 motor1 input 2 connect to GPIO pin.
- En2 enable for Motor2 connect to GPIO pin.
- A2 motor2 input 1 connect to GPIO pin.
- B2 motor2 input 2 connect to GPIO pin.
- 5V input-5V pin from RPI.
- 12 V+ and 12V- pin to battery.
- GND to ground pin in RPI.
- Out1 and Out2 connect it to Motor1.
- Out3 and Out4 connect it to Motor2.

Provide input1=0 and input2=0 motor stops rotating.
Provide input1=0 and input2=1 motor rotates clockwise.
Provide input1=1 and input2=0 motor rotates anti-clockwise.

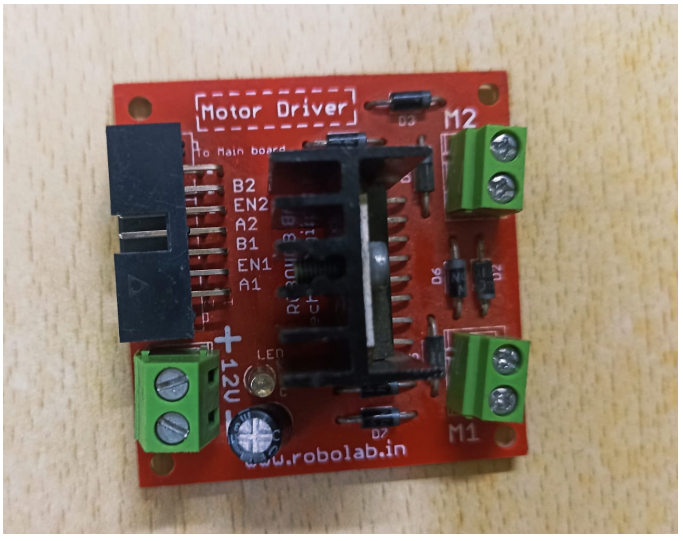


FIGURE 1.3: L298 Motor Driver

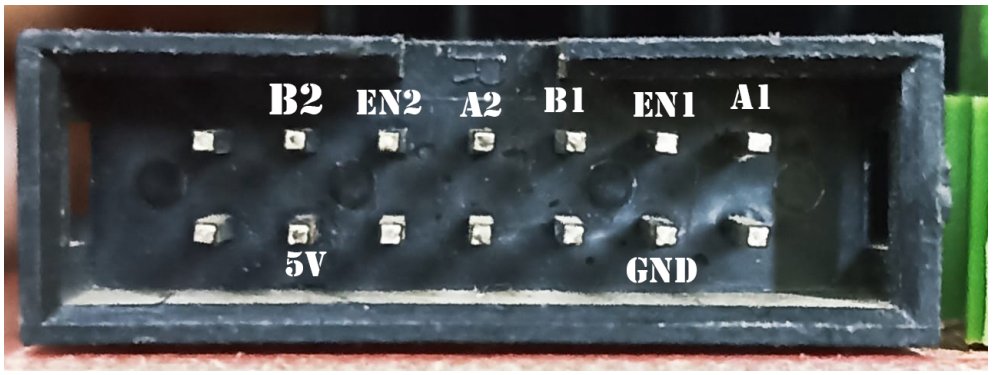


FIGURE 1.4: L298 Motor Driver Pin

1.2.3 DC Motor

DC Motor is used to move the robot. It needs 5V to run. To run it connects to Motor driver.

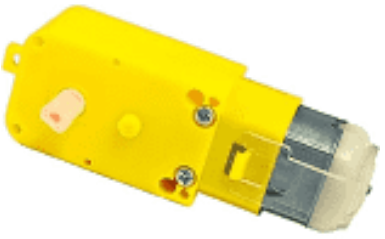


FIGURE 1.5: L298 Motor Driver Pin (Source Google Images)

1.2.4 RPI Camera

This camera is used to capture images. It is inserted into the slot provided in RPI.

FIGURE 1.6: Raspberry Pi Camera (Source Google Images)

1.2.5 Python and its Libraries

We will code this project in Python programming language. It is preferred because has a very rich library. In this project, we have used many but the most important is Open CV and which will be used for image processing purposes[4].

1.2.6 Databases

We have used the Firebase Realtime database which is cloud hosted database. Data is stored in JSON format and synchronized in realtime to every connected client.

1.2.7 Android Studio

We have developed the Android app using android studio software. It uses Java and Kotlin to develop an application. We have made an app using Java programming language. Then we generated an APK file which we install on a smartphone to control the robot.

1.3 Problem Addressed

There are various solution available in the market to address out-of-Stock(OOS) situation. Classifying them:

- The existing solutions are either in-accurate,expensive or just hypothetical.
- Existing robots available in the market are either social robots and not much inventory management purpose. Probable reason might be connectivity,heights of shelves in store and count products placed at back of each other in a shelf.

We took the following actions to address the aforementioned problem:

- To resolve connectivity we eliminated the wifi module and use a system such that Android App sends instructions to firebase realtime datanase and RPI reads the data from database.
- To Solve the height issue we inserted a arm which moves up and down adjusting according to the height of shelf.
- Regarding product count,the camera click photos of product and compares it with existing image in database and increase that product count in database.

1.4 Existing System

Currently, there are three popular methods for product availability check:
1. Manual collection and data entry
2. QR Codes
3. RFID
4. Weighted Smart Shelves
5. Wifi/bluetooth operated robots

1.4.1 Manual collection and data entry

In this, the staff of the store takes data of the products placed on the shelf. They note which product is available or not on paper or tablet. Then they provide the data to the data entry operator at the store. The entry operator enters the data in software like Excel, tableau, etc. Then the team sees what products to order from the market or not.
The Disadvantages are:
This system is good for small stores but for large stores which have fast stock-out and less labour it creates a problem for them.

1.4.2 QR Codes

QR Codes are specialized for of bar codes used to transfer data. The QR code links to website which collects feedback from the customer regarding product availability or misplacement.
The Disadvantages are:
This method is too customer centeric and customer instead of giving store manager the information may prefer to buy product from other store.

1.4.3 RFID

RFID (Radio-Frequency Identification) is a crucial technology for real-time tracking in the supply chain. It involves tags (with unique IDs), readers, and software. Tags on products store information and respond to RFID readers' signals, allowing for efficient inventory management. RFID offers advantages like improved accuracy, better inventory management, and prevention of errors.
The Disadvantages are:
The challenges include cost, simultaneous scanning of multiple tags, and potential privacy issues post-purchase.

1.4.4 Weighted Smart Shelves

This system involves weighted-sensing mat integrated with RFID Technology and ZigBee transceivers for inventory management on store shelves. The mats detects changes in weight on store shelves and can be adjusted based on sensitivity based on the type of product. The sensors measure the weight,enabling the back-end software to calculate product quantity. They also provides location based on the pressure detection. If system detects restocking,it send alert messages such as text or emails.
The Disadvantages are:
Installation cost is very high.
Sensitivity to weight - Very Light Weight products cannot be detected.
Do not provide information like which product is placed and product is placed at its correct position or not.

1.4.5 Wifi controlled Robots

The app is such that it connects to a robot using Bluetooth or Wifi. For it, we add a wifi module (ESP8266) to our raspberry pi. The robot functions quite well and it is in use for a long time but only within a certain range or we can say the range signals of the wifi module can reach. It also has another issue of disruptions. When someone comes between the phone and the robot it doesn't function properly. Especially the mirror comes in between it stops functioning completely.

Chapter 2

Literature Review

2.1 Introduction

In Harvard Business Review[5] by Ben Forgan on topic ”what robots can do for retail” he clearly explained the importance of robots in retail stores.
Here in this paper he in begin clearly stated that the real use of robots in retail is to track products placed on shelf and customer buying pattern which can increase efficiency and accuracy. He also cited a example of Auchan Retail Portugal,a large European grocery retailer is launching a robot for supermarkets which will capture photos of every shelf and aisle which will further help deriving insights about Out-of-stock merchandise and pricing.
Further in the paper the writer assumes a hypothetical scenario where the robots are deployed in store and in the store the supply of sugar-free peanut butter is diminishing twice the rate regular peanut butter. The robot automatically orders the product that is going diminishing without human intevention. If human do this insight it will take time. It is exactly similar to stock market where the specialized algorithms are used to predict prices of stock which can lead to profits in near future. Currently robots are used for assembly line operations like delivery of heavy items that mostly behind the scenes.
In an article published in Forbes[6], it was stated that roughly 150,000 robots will be present in-store by 2025.Many experts believe that due to the spread of the COVID-19 pandemic, contactless service behaviour could speed up this process.

2.2 Robots available in Market

Marty by Badger Technologies,Tally by Simbe Robotics,Alphabot by Walmart,Millie by Woolworths, and SmartSight EMA50 are some of the well-known robots that solve OOS situations and product misplacement issues. They use computer vision for recognition, an app for controlling robots, and also see results through this application. Through these results, we see

that shopping stores in developed countries have started adopting the strategy of smart inventory management and making more inventions to reduce errors during their functioning.[7]

2.3 Insights from Retail Stores

Retailers allocate 5% of their sales on logistics on average. At the shop level, inventory handling (38%) and holding (7%), account for the majority of these expenditures.For retailers to be profitable, better in-store inventory management is therefore crucial.

The following are the responses that customers have when an OOS condition arises: 31% purchase the necessary item, but they do it from a different retailer or online. 26% purchase a different brand. 19% of consumers continue to purchase the same brand but in a different colour, size, or taste. 15% purchase the item later. 9% don't make any purchases.

9% of consumers who make no purchases result in a EUR 4 billion loss. Given the abundance of options currently open to him, the consumer would be compelled to permanently go to a new retailer if the OOS scenario recurs. The weekly revenue adds up to EUR 150,000 in lost revenue when a "loyal" family abandons a particular retailer. Moreover, 20% of all OOS cases are unresolved after three days. Retailer profitability and performance are negatively impacted by this. A 3% increase in OSA for manufacturers corresponds to a 1% increase in sales, whereas a 2% increase in OSA for retailers corresponds to a 1% gain in sales. In order to keep customers, retail establishments must guarantee high OSA. [1][8]

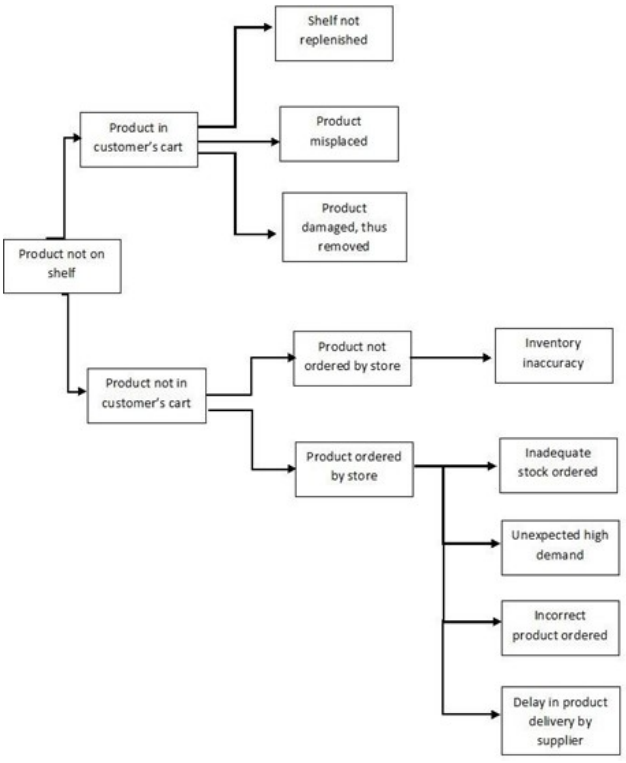


FIGURE 2.1: Reasons for OOS[1]

2.4 Various suggestions on Scanning Product

The robot must be able to recognise different products, locate them using a planogram, and recognise events like out-of-stock conditions and lost objects. Many approaches have been done to address this issue. For example, Zimmerman[9] can read a product's barcode from a shelf photograph. After that, it segments the shelf picture to match the retrieved image after retrieving the product image from a database. In the absence of a match, the out-of-stock flag is set. On the other hand, Gokturk[10] computes occupancy in a confined compartment using triangulation techniques using a camera and various illumination sources. Additionally, it advises employing a stereo-vision system or depth sensors to determine occupancy. Other patents, like those discuss general systems that can recognise items, provide planograms, recognise instances when supply is running low, and report % occupancy of products. The Hofman patent [11] offers an image-based system that can identify items, provide a count, recognise OOS conditions, and provide product layout. They employ a variety of properties, like SURF and color, to identify items and OCR to read text from logos. They don't offer any performance data for their process, nor do they offer any information on how they go about counting products.

2.5 Implemented Solution

The Solution to problems of Stock tracking to avoid Out-of-Stock (OOS) situation and product misplacement issue lies in designing a robot which can move around the store and scan the shelves for products and compare with the existing products images and if similar image found then update there count in Database.The robot should be controlled by an Application and we should be able to view the camera visuals.

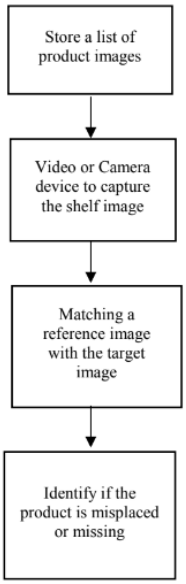


FIGURE 2.2: Implemented Solution

2.5.1 Input reference image and target Image

The reference image is the image already take by user with which we compare and target image is the image captured by camera device. The captured image is first converted to greyscale and cropped appropriately. Cropping is done to detect multiple instances of a reference image in the target image.

2.5.2 Extract Features

Now we will extract features from input image. For this we will use SURF Algorithm which generates a set of feature pairs between reference and target image.
The main objective of using SURF Algorithm is it has powerful attributes which are scale invarience,lighting invarience,contrast invarience etc.
Getting More Deeper in this algorithm:
i)Integral Image
An image is made by Pixels.

$$I\sum(x,y) = \sum_{i=0}^{i\leq x}\sum_{j=0}^{j\leq y} I(i,j)$$

ii) Intersect Point Detection(Fast-hessian Detector)
The SURF Algorithm determines the site of interset using a blob detector based on the Hessian. The Hessian Matrix H(x,sigma) in p at scale sigma, given a point P=(x,y) in an image I, is defined as follows:

$$\mathcal{H}(\mathrm{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathrm{x}, \sigma) & L_{xy}(\mathrm{x}, \sigma) \\ L_{xy}(\mathrm{x}, \sigma) & L_{yy}(\mathrm{x}, \sigma) \end{bmatrix}$$

iii) Orientation Assignment:
Two first-order Haar wavelets are used to convolve the image. A vector in a two-dimensional space represents the filter responses at specific sampling points surrounding the keypoint. All vectors within its range are added up using a rotating window of 60, and the orientation is determined by the longest vector that results.
iv)Descriptor Generation:
In the SURF Algorithm all the points in image are assigned same weight which can be found using:

$$W_P = \frac{\text{Number of Detected Products w.r.t Point P}}{\text{Number of Training Images in Target Image}}$$

2.5.3 Matching

The matching of reference and target images is done with the help of descriptors. The 2 methods for doing this are:
1)Characteristics points and descriptors of target and reference image are obtained. Then compare images using these descriptors.
2) In another method we only obtain characteristic points of reference image with the descriptor. Then compare this descriptor with points on target image which is believed to be same.

”matchfeature” function returns indices of matching features in two images.

2.5.4 Determining presence of reference image in target image

If the reference image is detected in target image a box is drawn over the detected product and the count of that product is updated in database.This process is done in iterative manner all the reference images is detected in target Image.

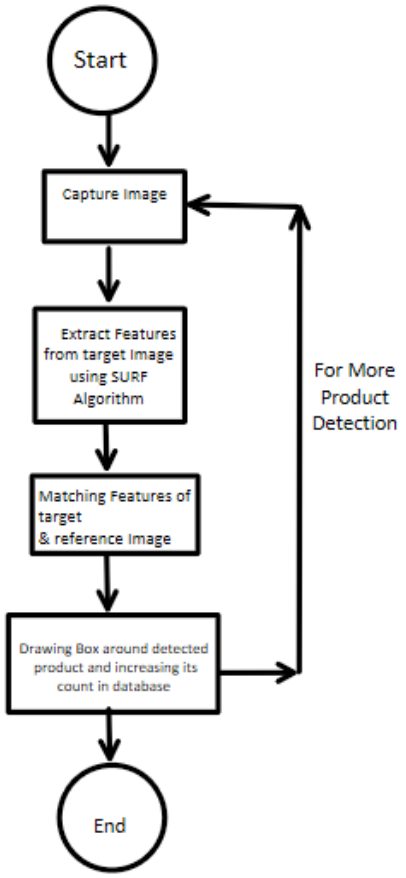


FIGURE 2.3: Flowchart of Image recognition

2.6 Robot Development

2.6.1 Types of Wheeled Robot

- Robots with One Wheel
  This type of robot is unstable. A spherical Robot can be consider a single wheeled robot.
- Robots with Two Wheels
  Two-wheel robots are divided into two types: bicycle-type and inverted-pendulum-type. A bicycle-type robot has a front and rear wheel, reducing robot width but not maintaining pose. Inverted-pendulum-type robots have two wheels and static stability, but

require dynamic balancing control. Pendulum-type robots can climb stairs but require constant control effort. Both types have their advantages and disadvantages.

- Robots with Three Wheels
  Robot with 3 wheels are highly stable and has simple structure.It is the most widely structure. Here we will discuss about only Two-Wheel Differential-Drive Robot. A two-wheel differential-drive robot is a popular design with two active fixed wheels and one passive caster wheel. It has advantages such as a simple mechanical structure, low fabrication cost, zero turning radius, and easy calibration of systematic errors. However, it faces challenges in moving irregular surfaces and only offers bidirectional movement. Its types are given in figure:

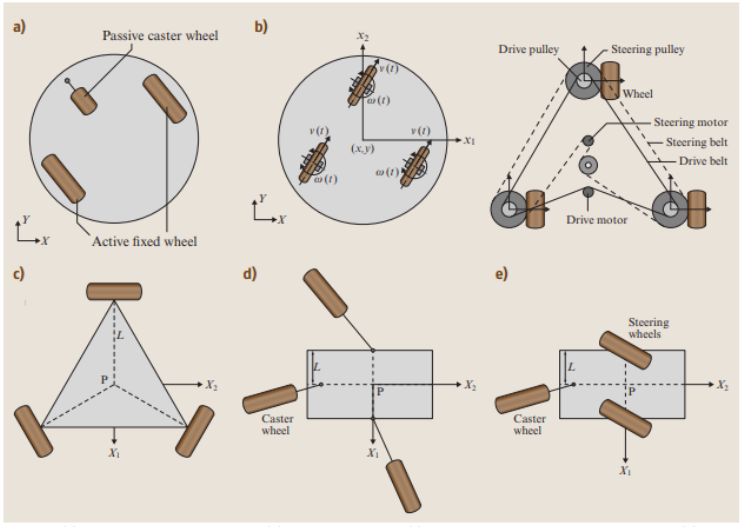


Fig. 17.7 (a) Two-wheel differential drive, (b) synchronous drive, (c) omnimobile robot with Swedish wheels, (d) omnimobile robot with active caster wheels, (e) omnidirectional robot with active steerable wheels

FIGURE 2.4: Sources:[2]

- Robots with Four Wheels
  The focus is on a car-like structure of four-wheel robots, with front two wheels synchronously steered to maintain a constant rotation center. This type of robot is stable during high-speed motion but requires a complicated steering mechanism, making parking motion control difficult in cluttered environments.
  In robot developed further in this project we have used two-wheel differential-drive robot.

2.6.2 Types of Wheels

The most commonly used wheels in robots are:Standard Wheels,Caster and Ball Wheels,Mecanum Wheels and Omni Wheels.
We have used 1 Caster wheels because it is good for indoor design,Easily orient in any direction and provides balance to robot.

We have used standard wheels because they offer two degrees of freedom, can travel forward or reverse, and can be used as drive or idler wheels. They come in various mounting arrangements, with a wide selection of tire treads, and can be used indoors or outdoors.

2.6.3 Differential wheeled robot

The Kinematics behind Differential wheeled robot is well explained in text [12]

Chapter 3

Proposed Work

3.1 To open RPI interface follow the steps:

1. We first installed putty in your machine.
2. Now in hostname(or IP address) type ”raspberrypi.local”.Press Enter.
3. Now type the username ”rpi”,press enter,type password ”rpi” and then press enter again.
4. Now install VNC viewer in your machine.
5. In VNC server address or search pane type ”raspberrypi.local” and press enter.
6. Now the window of RPI will pop up.

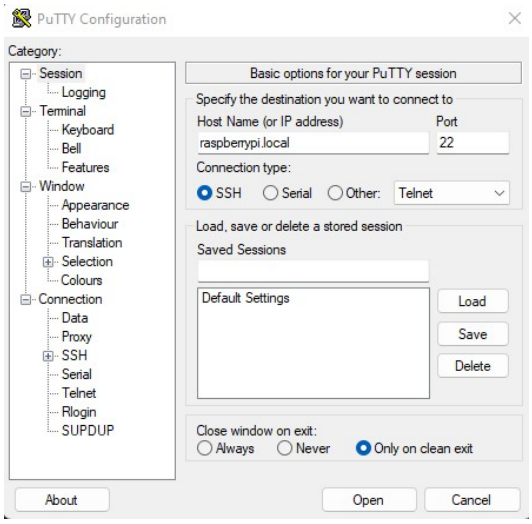


FIGURE 3.1: Putty

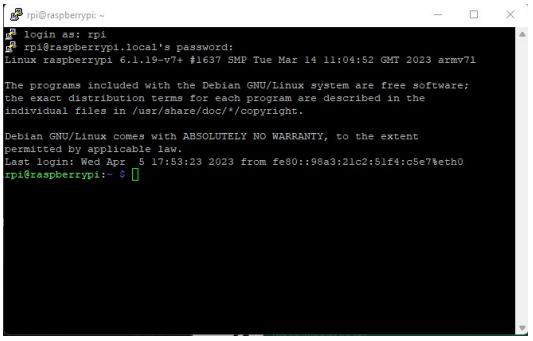


FIGURE 3.2: Putty Command Line

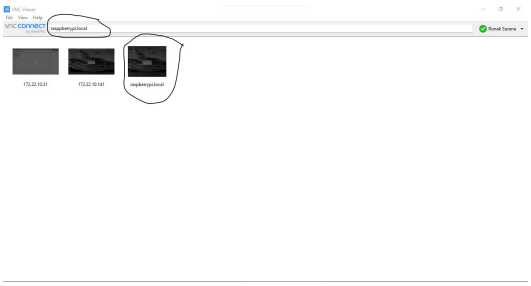


FIGURE 3.3: VNC Viewer

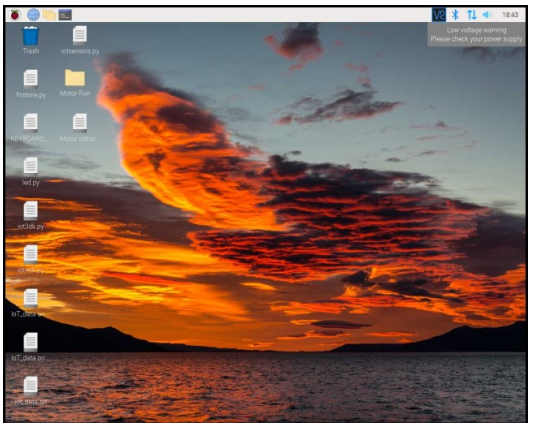


FIGURE 3.4: RPI Desktop

3.2 Code writing:

1. To write code in RPI. First install any python IDE.In case you are using text editor remember to save file with extension ".py".

2. In our case we have used thonny IDE.To run file in this IDE the is a run button.

3. If you want to import any library open rpi terminal and type commands which you will find in documentation of that library.

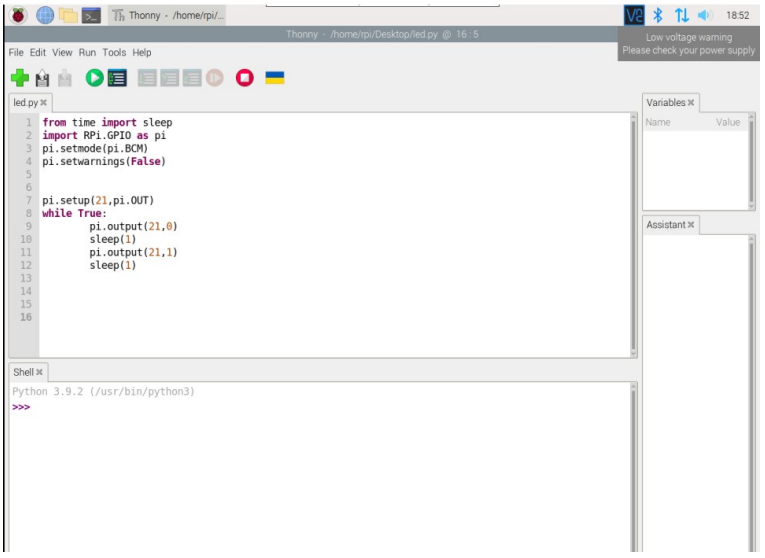
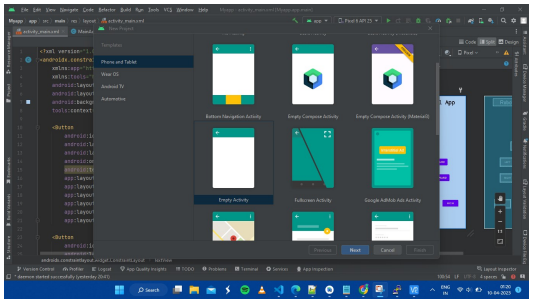
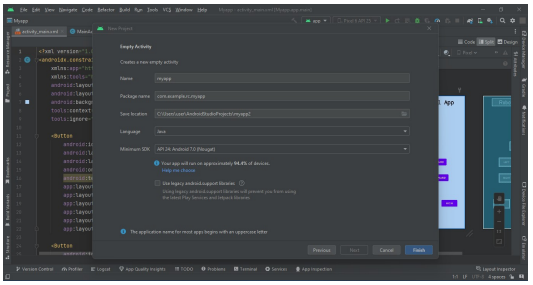


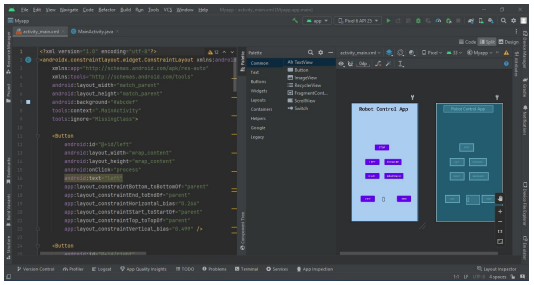
FIGURE 3.5: IDE for writing and Running Code

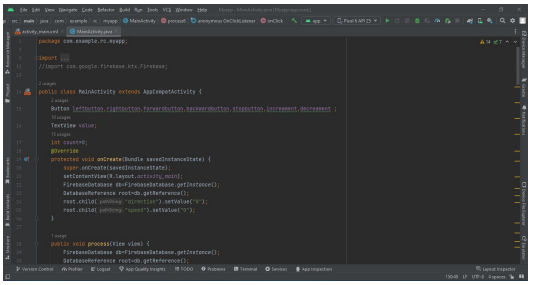
3.3 Developing the Mobile App:

1. Install android studio.
2. Make a New Project. Click on Empty Activity. Give a Name to your Application.Specify package name,location where you want to save your file.We have give ”myapp” name to our app.
3. There are two languages in which we can develop an android app-Java and Kotlin. We have used java to develop an app.
4. Specify Minimum SDK that is the device in which you are running your app should have min this android version.We have specified ”Android 7”.
5. Press Finish.Now open only two files activitymain.xml and MainActivity.java.
6. In activitymain.xml,we specify the frontend of that app i.e. how to app looks.In this in the right pane we drag and drop the objects that we want in the app.And in the coding pane of this file we write what text should be want in the button or what text we want to display etc. Also we write if somone click on button which function it should trigger.
7. In our app in we have added seven buttons and in ’android:text=”name”’ typed left,right,forward,backward,high and low. To make the these button function we add a line ’android:onclick=”functionName”’. When somebody clicks on button it triggers the function that we wrote in MainActivity.java.
8. In MainActivity.java we write the function and it is basically the brain that you are giving to app.









3.4 Making and connecting database to android app and get data in RPI:

1. Follow the steps given in the documentation(it keeps updating).
2. To get the firebase data into the rpi write the code.
3. The fields can be created manually or gets created automatically also.
4. In our case we have added two fields one speed and another direction.
5. when someone presses left in app value of direction in db becomes ”1”,right ”2”,forward ”3” and backward becomes ”4”.
6. when someone press high in app speed gets increamented by 10 and when someone press low in app speed gets decreamented by 10.
7. when someone presses stop both speed and direction becomes 0.

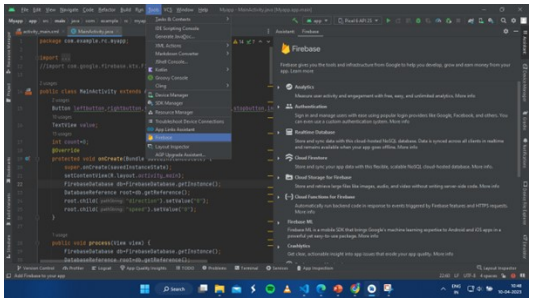


FIGURE 3.6: click on tools and then firebase and then realtime database

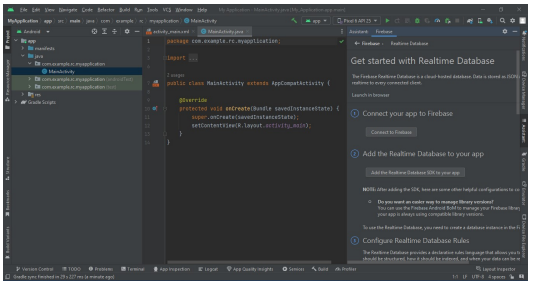


FIGURE 3.7: click on connect to firebase

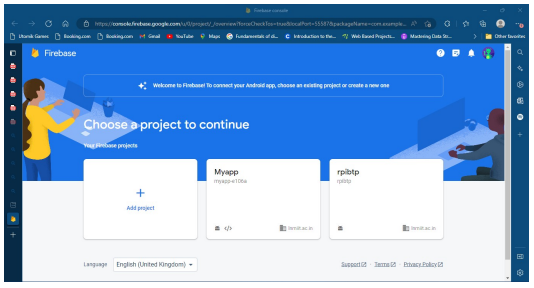


FIGURE 3.8: click Add project to create project

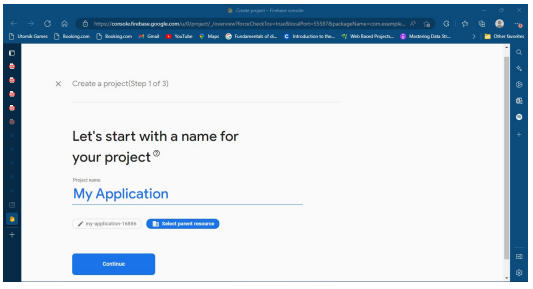


FIGURE 3.9: Give name to project

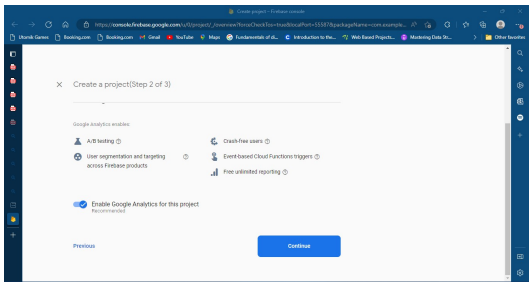


FIGURE 3.10: click on continue

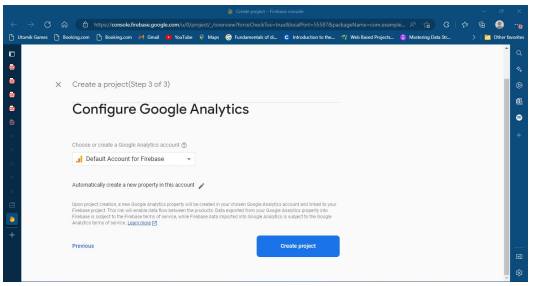


FIGURE 3.11: click create project

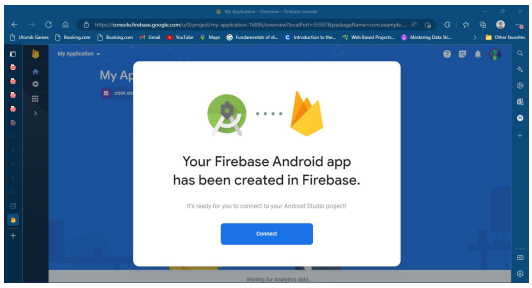


FIGURE 3.12: click connected to connect fb to android studio

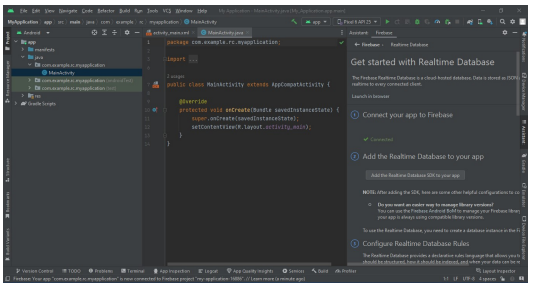


FIGURE 3.13: Add realtime database to app

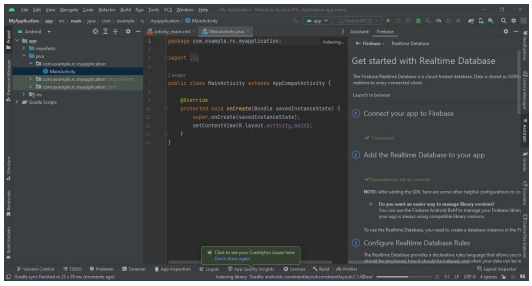


FIGURE 3.14: connected as well as added realtime db to our app

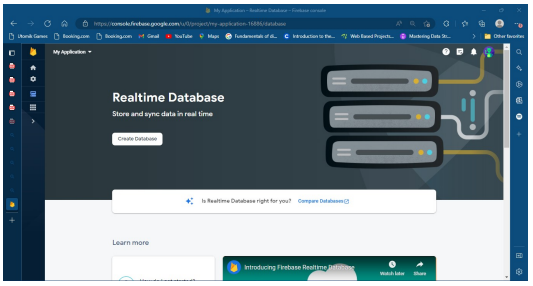


FIGURE 3.15: creating a realtime database

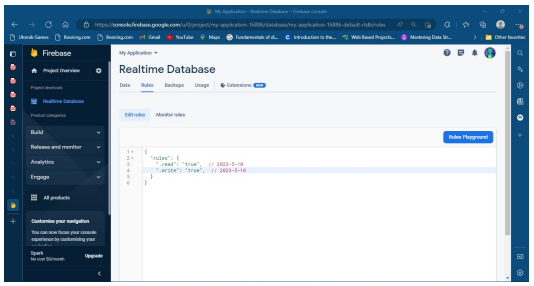


FIGURE 3.16: changes we have to made in rules

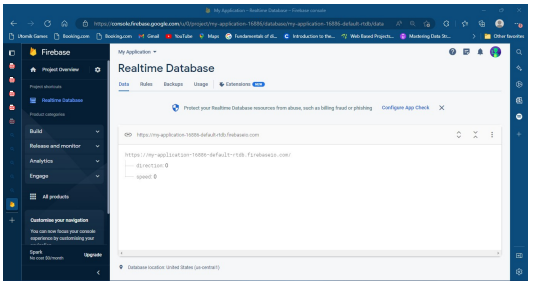


FIGURE 3.17: Realtime database

3.5 Running the robot:

1. Make the connections according to the figure.
2. Now in terminal of RPI write the program to automatically run the code and connect to the particular wifi network automatically.
3. Now as you connect power to RPI the code will run and you can control the robot.

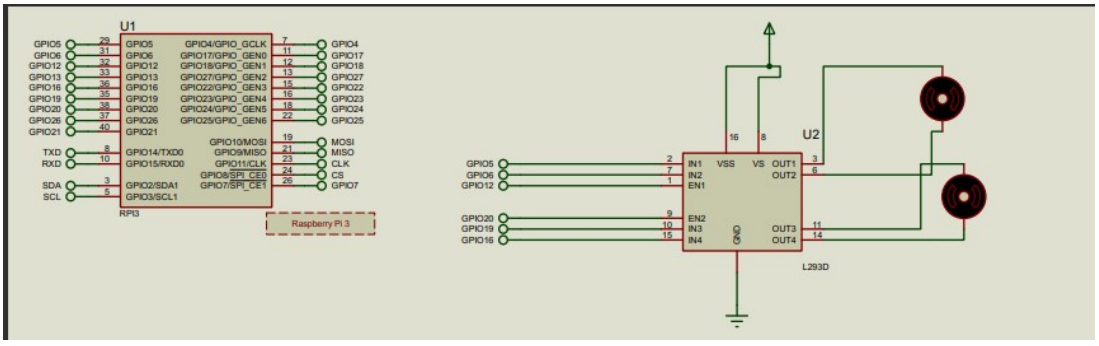


FIGURE 3.18: Connection of RPI with dc Motors in Proteus

3.6 Connecting the Camera:

1. Connect the camera to RPI in the place provided on the board.
2. Now add the program to click photos at a particular time interval and then further process that picture.

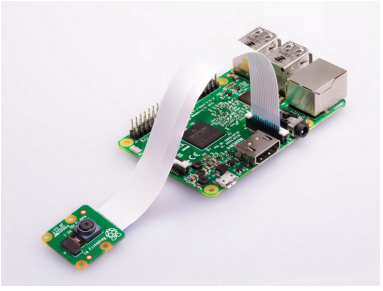


FIGURE 3.19: RPI camera connected to RPI(Source Google Images)

3.7 Codes to Develop Android App

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://
    schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#abcdef"
    tools:context=".MainActivity"
    tools:ignore="MissingClass">

    <!-- Buttons for Robot Control -->
    <Button
        android:id="@+id/left"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="process"
        android:text="left"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.266"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.499" />

    <!-- Repeat similar Button elements for other directions -->

    <!-- TextView to display a value -->
    <TextView
```

```
        android:id="@+id/value"
        android:layout_width="80dp"
        android:layout_height="60dp"
        android:text="0"
        android:textSize="40sp"
        android:layout_marginLeft="30dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.525"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.847" />

    <!-- Buttons for setting motor speed -->
    <Button
        android:id="@+id/low"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="process5"
        android:text="Low"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.179"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.833" />

    <!-- Repeat similar Button elements for other speed controls -->

    <!-- TextView to display the app title -->
    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Robot Control App"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.449"
        android:textSize="30dp"
        android:textStyle="bold"
        android:textColor="@color/black"
        android:textAlignment="center"
        android:fontFamily="monospace"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.022" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

LISTING 3.1: MainActivity.java file

```
package com.example.rc.myapp;
```

```
2
3  import androidx.appcompat.app.AppCompatActivity;
4  import android.os.Bundle;
5  import android.view.View;
6  import android.widget.Button;
7  import android.widget.TextView;
8  import com.google.firebase.database.DatabaseReference;
9  import com.google.firebase.database.FirebaseDatabase;
10
11 public class MainActivity extends AppCompatActivity {
12     Button leftbutton, rightbutton, forwardbutton, backwardbutton,
       stopbutton, increament, decreament;
13     TextView value;
14     int count = 0;
15
16     @Override
17     protected void onCreate(Bundle savedInstanceState) {
18         super.onCreate(savedInstanceState);
19         setContentView(R.layout.activity_main);
20         FirebaseDatabase db = FirebaseDatabase.getInstance();
21         DatabaseReference root = db.getReference();
22         root.child("direction").setValue("0");
23         root.child("speed").setValue("0");
24     }
25
26     // Implement onClick methods for each button
27     // ...
28
29     // Process method for the "Stop" button
30     public void process4(View view) {
31         FirebaseDatabase db = FirebaseDatabase.getInstance();
32         DatabaseReference root = db.getReference();
33         value = findViewById(R.id.value);
34         stopbutton = findViewById(R.id.stop);
35         stopbutton.setOnClickListener(new View.OnClickListener() {
36             @Override
37             public void onClick(View view) {
38                 root.child("direction").setValue("0");
39                 root.child("speed").setValue(0);
40                 value.setText("" + 0);
41             }
42         });
43     }
44
45     // Process method for the "Low" button
46     public void process5(View view) {
47         FirebaseDatabase db = FirebaseDatabase.getInstance();
48         DatabaseReference root = db.getReference();
49         decreament = findViewById(R.id.low);
50         value = findViewById(R.id.value);
51         decreament.setOnClickListener(new View.OnClickListener() {
52             @Override
```

```
            public void onClick(View view) {
                // Implementation for decreasing speed
                // ...
            }
        });
    }

    // Process method for the "High" button
    public void process6(View view) {
        FirebaseDatabase db = FirebaseDatabase.getInstance();
        DatabaseReference root = db.getReference();
        increament = findViewById(R.id.high);
        value = findViewById(R.id.value);
        increament.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                // Implementation for increasing speed
                // ...
            }
        });
    }
}
```

LISTING 3.2: activity_main.xml file

3.8 Robot moving wheel code

```
import RPi.GPIO as GPIO
from time import sleep
import firebase_admin
from firebase_admin import credentials, db

# Initialize the Firebase Admin SDK with your service account key
cred = credentials.Certificate('/home/rpi/Desktop/Motor_Run/testfile/myapp-
    e106a-firebase-adminsdk-gl9uq-a62990e6c3.json')
firebase_admin.initialize_app(cred)
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
In1, In2 = 3, 5
In3, In4 = 11, 13


GPIO.setup(In1, GPIO.OUT)
GPIO.setup(In2, GPIO.OUT)

GPIO.setup(In3, GPIO.OUT)
GPIO.setup(In4, GPIO.OUT)


```

```
pwm1 = GPIO.PWM(In1, 100)
pwm1.start(0)

pwm2 = GPIO.PWM(In2, 100)
pwm2.start(0)

pwm3 = GPIO.PWM(In3, 100)
pwm3.start(0)

pwm4 = GPIO.PWM(In4, 100)
pwm4.start(0)

ref = db.reference('/')
speed = ref.child("speed").get()
direction = ref.child("direction").get()


while True:
    speed = ref.child("speed").get()
    direction = ref.child("direction").get()
    if direction == 0 or speed == 0:  # not running initially
        pwm1.ChangeDutyCycle(0)
        pwm2.ChangeDutyCycle(0)
        pwm3.ChangeDutyCycle(0)
        pwm4.ChangeDutyCycle(0)
    else:
        if direction == "1":  # left
            pwm1.ChangeDutyCycle(speed)
            pwm2.ChangeDutyCycle(0)
            pwm3.ChangeDutyCycle(0)
            pwm4.ChangeDutyCycle(0)
        elif direction == "2":  # right
            pwm1.ChangeDutyCycle(0)
            pwm2.ChangeDutyCycle(0)
            pwm3.ChangeDutyCycle(speed)
            pwm4.ChangeDutyCycle(0)
        elif direction == "3":  # Forward
            pwm1.ChangeDutyCycle(speed)
            pwm2.ChangeDutyCycle(0)
            pwm3.ChangeDutyCycle(speed)
            pwm4.ChangeDutyCycle(0)
        elif direction == "4":  # backward
            pwm1.ChangeDutyCycle(0)
            pwm2.ChangeDutyCycle(speed)
            pwm3.ChangeDutyCycle(0)
            pwm4.ChangeDutyCycle(speed)
        else:
            break
GPIO.cleanup()
```

LISTING 3.3: Motor Control with Firebase Integration

3.9 Image Capturing and Processing

```
import cv2
import numpy as np
import RPi.GPIO as GPIO
import firebase_admin
from firebase_admin import credentials, db
import picamera

# Initialize Firebase app
cred = credentials.Certificate('/home/rpi/Desktop/Motor_Run/testfile/myapp-e106a-firebase-adminsdk-gl9uq-a62990e6c3.json')
firebase_admin.initialize_app(cred)

# Create a reference to the root of your database
ref = db.reference('/')

# Load the target image (product in a different color box)
camera=picamera.PiCamera()
camera.capture('/home/rpi/Desktop/target_image.png')
target_image = cv2.imread('/home/rpi/Desktop/target_image.png', cv2.IMREAD_GRAYSCALE)

dicto={"Pic_1":"Rin","Pic_2":"Ariel","Pic_3":"Tide"}

for i in range(1,4):
    # Load the template image (one of the products in a different color box)
    pic_i_value = "Pic_"+str(i)
    template_image = cv2.imread(f'/home/rpi/Desktop/{pic_i_value}.png', cv2.IMREAD_GRAYSCALE)
    # Initialize the ORB detector
    orb = cv2.ORB_create()
    # Find keypoints and descriptors in the template and target images
    kp1, des1 = orb.detectAndCompute(template_image, None)
    kp2, des2 = orb.detectAndCompute(target_image, None)
    # Check if descriptors are empty
    if des1 is None or des2 is None:
        print("No descriptors found in one of the images.")
    else:
        # Convert descriptors to CV_32F if needed
        if des1.dtype != np.float32:
            des1 = des1.astype(np.float32)
        if des2.dtype != np.float32:
            des2 = des2.astype(np.float32)
        # Create a BFMatcher (Brute Force Matcher) and perform matching
        bf = cv2.BFMatcher()
        matches = bf.knnMatch(des1, des2, k=2)
        # Apply ratio test to select good matches
        good_matches = [m for m, n in matches if m.distance < 0.7 * n.distance]
```

```
        for m, n in matches:
            if m.distance < 0.75 * n.distance:
                good_matches.append(m)
        if len(good_matches)>=4:
            # Create a copy of the target image to draw contours on
            target_with_contours = target_image.copy()
            # Extract coordinates of keypoints from good matches
            src_pts = np.float32([kp1[m.queryIdx].pt for m in good_matches
]).reshape(-1, 1, 2)
            dst_pts = np.float32([kp2[m.trainIdx].pt for m in good_matches
]).reshape(-1, 1, 2)
            # Calculate the homography matrix
            M, _ = cv2.findHomography(src_pts, dst_pts, cv2.RANSAC, 5.0)
            # Get the dimensions of the template image
            h, w = template_image.shape
            # Define the four corners of the template image
            template_corners = np.float32([[0, 0], [w, 0], [w, h], [0, h]])
.reshape(-1, 1, 2)
            # Transform the corners to the perspective of the target image
            transformed_corners = cv2.perspectiveTransform(template_corners
, M)
            # Draw a contour around the detected object
            cv2.polylines(target_with_contours, [np.int32(
transformed_corners)], True, 255, 3, cv2.LINE_AA)
            # Create a resizable window and display the matched image with
contours
            # cv2.namedWindow('Matched Image with Contours', cv2.
WINDOW_NORMAL)
            # cv2.resizeWindow('Matched Image with Contours', 800, 600)  #
Replace with your desired width and height
            # cv2.imshow('Matched Image with Contours',
target_with_contours)
            base=ref.child("Products").get()
            x=base[dicto[pic_i_value]]+1
            #print(x,dicto[pic_i_value])
            result1 = ref.child("Products").update({dicto[pic_i_value]: x})
            cv2.waitKey(0)
            cv2.destroyAllWindows()
        else:
            print("Not enough good matches. Skipping update.")
```

LISTING 3.4: Object Recognition and Firebase Update

3.10 Auto connect to wifi and run code

Auto connect to wifi:

1) Type the command in putty: sudo nano/etc/wpa_supplicant/wpa_supplicant.conf

2) Then next type: network={

ssid=”Test Wifi Network”
psk=”SecretPassWord”
}
3) Press ctrl+X followed by Y and press enter.
4) Then type sudo reboot.
5) To check that rpi is connected to wifi use command: ifconfig wlan0

Auto run code on startup:
1) Open putty.Login into rpi.
2) Navigate to directory where file is located with command: cd ”location”
3) Look at the program with the command: nano ”filename”
4) Run the program with the command: sudo python ”filename”

Chapter 4

Simulation and Results

4.1 Android App UI

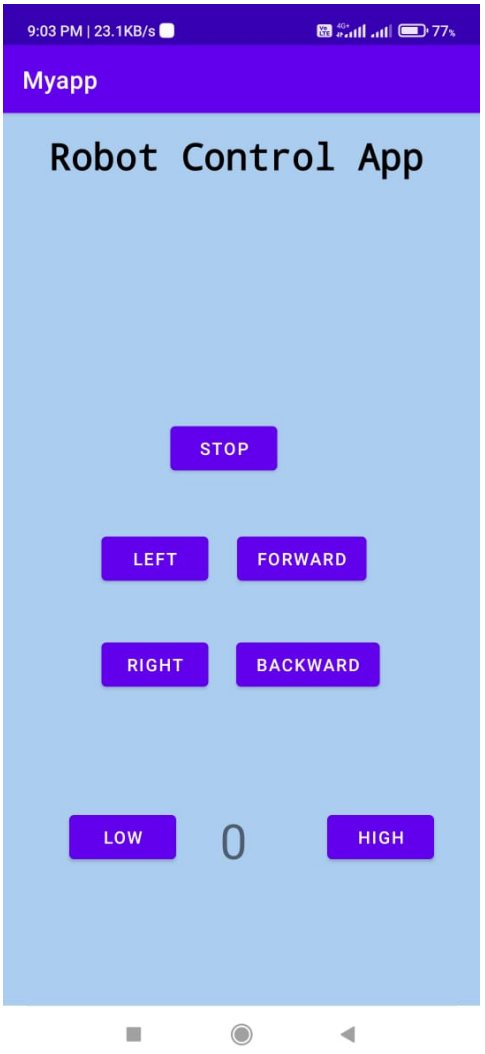


FIGURE 4.1: Android App

4.2 Image Recognition

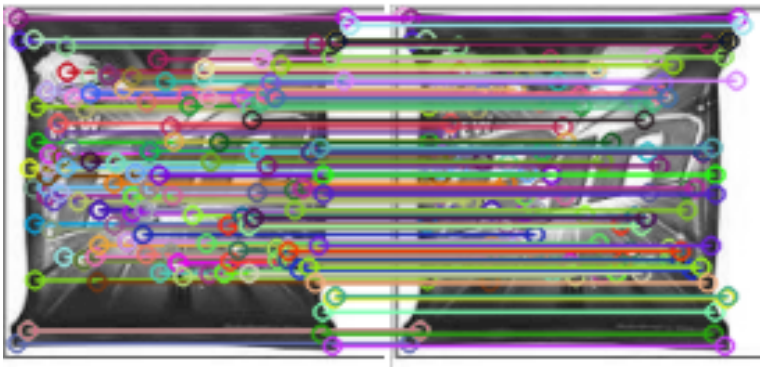


FIGURE 4.2: Reference image and Target Image Matching

4.3 Database

FIGURE 4.3: Initial Database

FIGURE 4.4: Final after Product Count Increase

4.4 Robot Model

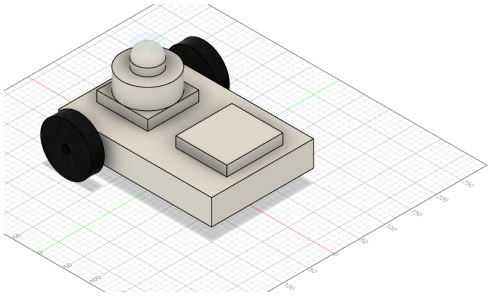


FIGURE 4.5: Robot Simulation Top View

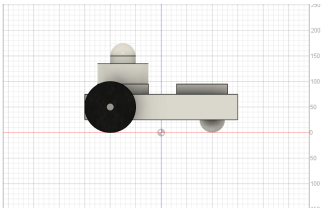
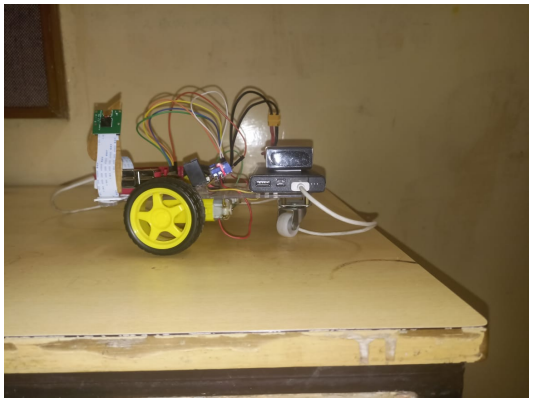
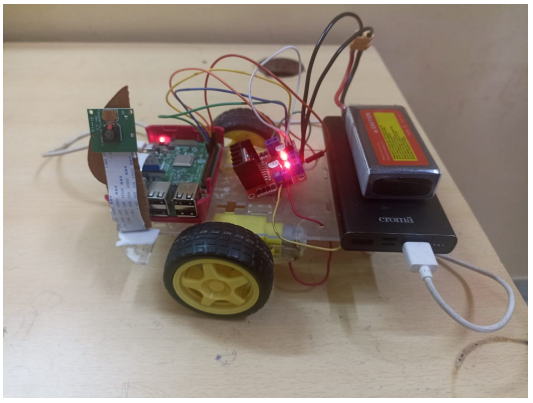


FIGURE 4.6: Robot Simulation Side View





Chapter 5

Conclusions and Future Work

5.1 Conclusion

In the world of Artificial Intelligence and 5G Fast Internet the lifestyle of people are changing. Every Industry is adopting new technologies to keep pace with time. Shopping style of people is also changing. From online shopping people have build trust on buying goods but when it comes to buying grocery items the people still trust going to nearby stores. The inventory management have always been a problem for grocery stores. The solution that we saw in above discussion that are already existing in market have some flaws. Through this project we tried to resolve existing problems in the store management. The android app controlled robot moves wirelessly with the support of raspberry-pie. It is having a feature through which we can control the speed of the robot. Raspberry Pi-powered robot is programmed to move around a shopping store and use camera to capture images of shelves in store. The robot use the input images and apply some algorithms on it. The images of the available products in store are already stored in the database. The algorithms compares these images with input image and if found the robot will transmit this data to a central database, allowing managers to monitor stock levels and make informed decisions about restocking.

5.2 Future Work

In future work can be on:

- Developing More accurate algorithms for product recognition from capture image.
- Using Depth Cameras to count product placed at back of each other.
- Can be combined up with robotic arm to pick product and attach bucket for collection.

- An AI based product can also be developed which collects feedback from customer and also guide them in their shopping.

Bibliography

[1] R. Moorthy, S. Behera, and S. Verma, "On-shelf availability in retailing," vol. 115, pp. 47–51, 04 2015.

[2] W. C. Guy Campion, "Handbook-robotics-ch-17,"

[3] Raspberry Pi.[Online].https://www.raspberrypi.com/documentation/.

[4] Open CV.[Online].https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html.

[5] B. Forgan, "What robots can do for retail."

[6] B. Morgan, "The 3 best in-store robots and why they work."

[7] K. Matthews, "5 robots now in grocery stores show the future of retail."

[8] R. Moorthy, S. Behera, S. Verma, S. Bhargave, and P. Ramanathan, "Applying image processing for detecting on-shelf availability and product positioning in retail stores," pp. 451–457, 08 2015.

[9] T. Zimmerman, "System and method for performing inventory using a mobile inventory robot," Mar. 27 2008,uS Patent App. 11/534,162. [Online]. Available: https://www.google.com/patents/US20080077511.

[10] S. B. Gokturk and A. Rafii, "Occupancy detection and measurement system and method," Oct. 2 2003,uS Patent App. 10/678,998.

[11] Y. Hofman and M. Rotenberg, "System and method for identifying retail products and determining retail product arrangements," June 20 2012, uS Patent App. 13/528,189.

[12] Wikipedia, "Differential wheeled robot."