



A project report on

**Development of Mobile Robot for Inspecting of Environment
Condition in Hazardous Terrains**

submitted in partial fulfillment of the requirements for the degree of

B. Tech

In

Electronics and Telecommunication Engineering

By

SOUMYA JYOTI MONDAL
UTKARSH PANDEY

2004099
2004104

Under the guidance of

Prof. Supervisor :- Prof. Suprava Patnaik and Prof. Amit Bakshi

School of Electronics Engineering
KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
(Deemed to be University)
BHUBANESWAR

APRIL 2023

CERTIFICATE

This is to certify that the project report entitled “**Development of Mobile Robot for Inspecting of Environment Condition in Hazardous Terrains**” submitted by

SOUMYA JYOTI MONDAL
UTKARSH PANDEY

2004099
2004104

in partial fulfillment of the requirements for the award of the **Degree of Bachelor of Technology in Electronics and Telecommunication Engineering** is a bonafide record of the work carried out under my guidance and supervision at School of Electronics Engineering, KIIT (Deemed to be University).

Signature of Supervisor

Prof. Suprava Patnaik

School of Electronics Engineering
KIIT (Deemed to be University)

Signature of Supervisor

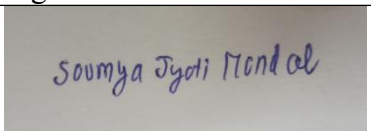
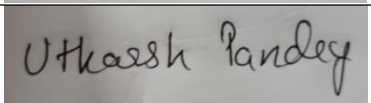
Prof. Amit Bakshi

School of Electronics Engineering
KIIT (Deemed to be University)

ACKNOWLEDGEMENTS

We are incredibly grateful to our supervisors, Prof. Suprava Patnaik and Prof. Amit Bakshi , for providing us with outstanding supervision during the course of our project work. He has been a huge influence to us with his kindness, commitment, effort, and attention to detail. Sincere gratitude to you, sir, for your unending kindness and tolerance towards us. We would especially like to thank him for all of the assistance in carefully and patiently editing all of our manuscripts.

We also want to express our gratitude to the project coordinators and the associate dean and professor of the school of electronics engineering, Dr. (Mrs.) Sarita Nanda, as well as to the associate dean and professor of the school, Dr. (Mrs.) Suprava Patnaik. We appreciate their support and suggestions all throughout the entire project work in the sixth semester of our undergraduate course.

Roll Number	Name	Signature
2004099	SOUMYA JYOTI MONDAL	
2004104	UTKARSH PANDEY	

Date: 07/11/2023

ABSTRACT

Mobile robotics has emerged as a rapidly growing field in recent years, thanks to advances in sensor and actuator technologies, machine learning algorithms, and computational hardware. Today's mobile robots are capable of performing a wide range of tasks in various environments, from manufacturing floors to hospital wards, from outdoor fields to outer space.

One of the key features of modern mobile robotics is their ability to navigate autonomously in dynamic and uncertain environments. This is achieved through the integration of multiple sensors such as cameras, lidars, and sonars, and the use of algorithms that enable the robots to interpret sensory data and make decisions in real-time. Furthermore, modern mobile robots can communicate with other devices, enabling them to collaborate with humans and other robots to achieve complex tasks.

Another important aspect of modern mobile robotics is their ability to learn from experience. Machine learning algorithms enable mobile robots to adapt to changing environments, recognize objects, and make predictions based on patterns in data. This enables robots to perform tasks more efficiently and effectively, and to improve their performance over time.

This project describes the design and implementation of a model mobile robot for the purpose of inspection of environmental condition by going in a hazardous terrain. It is an unmanned vehicle capable of autonomous driving by avoiding the obstacles in the terrain, it senses the environment using sensor system, and computer vision algorithms for better understanding of the environment and live streaming the video feed on to web page using an onboard computer as a server. The robot uses a LAN communication protocol to extend its communication range by forming a chain interconnected WiFi modules where the starting and ending point being the onboard computer in the robot and the computer of the user.

Keywords-robot, computer vision, autonomous, server, sensor system, LAN.

TABLE OF CONTENTS

Acknowledgement

Abstract

Table of Contents

CHAPTER 1: INTRODUCTION	1
1.1 Motivation	1
1.2 Background Studies /Literature Survey	2
1.3 Objectives	3
CHAPTER 2: METHODOLOGY	4
2.1 Applied Techniques and Tools	4
2.2 Technical Specifications	
CHAPTER 3: EXPERIMENTATION AND TESTS	13
3.1 Experimental Setup/Design	13
3.2 Mathematical Modeling, Circuits etc.	15
3.3 Prototype Testing/Simulations	
	:
CHAPTER 4: CHALLENGES, CONSTRAINTS AND STANDARDS	22
4.1 Challenges and Remedy	22
4.2 Design Constraints	23
4.3 Alternatives and Trade-offs	24
CHAPTER 5: RESULT ANALYSIS AND DISCUSSION	26
5.1 Results Obtained	26
5.2 Analysis and Discussion	28
CHAPTER 6: CONCLUSIVE REMARKS	30
6.1 Project Planning, Progress and Management	30
6.2 Conclusion	34
6.3 Further Plan of Action / Future Scope	35

REFERENCE

APPENDIX A: GANTT CHART

APPENDIX B: PROJECT SUMMARY

APPENDIX C: CODE

CHAPTER 1

INTRODUCTION

1.1 Motivation

In earlier times, before technology became advanced, the conventional method of testing the safety of an unexplored region was to send animals or humans with some precautions but still that was not enough of an effective method as lives were lost either on sight or later due to inefficient testing of the environment. The modern day solution to this problem is to send autonomous mobile robots which are both more efficient and also does not pose any danger to human lives, thus conducting a safe and effective testing of a terrain which was not ventured before.

1.2 Background Studies /Literature Survey

Intelligent mobile robots that can move independently were laid out in the real world around 100 years ago during the second world war after advancements in computer science. Since then, mobile robot research has transformed robotics and information engineering. For example, robots were crucial in military applications, especially in teleoperations, when they emerged during the second world war era. Furthermore, after the implementation of artificial intelligence (AI) in robotics, they became autonomous or more intelligent. Currently, mobile robots have been implemented in many applications like defense, security, freight, pattern recognition, medical treatment, mail delivery, infrastructure inspection and developments, passenger travel, and many more because they are more intelligent nowadays with artificial intelligence technology.

1.3 Objectives:

The main Objectives for the project are as follows:-

1. Exploring the selected hazardous terrain .
2. Collect data of the environment using sensor array.
3. Send back collected data real-time for analysis and visualization.

Overall, the objectives of these mobile robots are to enhance safety, convenience, and accessibility for humans, making the exploration experience more accessible and safer for everyone.

CHAPTER 2 METHODOLOGY

2.1 Applied Techniques and Tools

The objective of the mobile robot is to go in the hazardous terrain to collect data on its environmental conditions and send back the data to the user at the base station. The autonomous driving system is made using micro-controller (Arduino) programmed in embedded C interfaced with ultrasonic sensor (HR04) mounted over a servo motor with 180 degrees of rotation to get proximity of obstructing structures in the environment for a collision free self driving and a 4DOF manipulator to collect samples. The microprocessor or onboard computer (Raspberry Pi 4) is interfaced with temperature & humidity sensor (DHT11), gas sensor (MQ2) and camera module (Pi V2.1), the DHT11 and MQ2 are programmed in python3 and the data is acquired by the 2 sensors using the raspberry PI 4 and the data is visualized using graphical techniques using python 3 with matplotlib and seaborn libraries, for live streaming raspberry pi 4 is used as a processor to stream the video feed on web page using Flask library by browsing the URL in the search engine, further object detection, segmentation and classification is done on the live feed using OpenCV, Pixellib and YOLO v6. In the event an unknown object or entity is observed by the human eye on the live stream which is not present in the dataset then image augmentation technique using augmentations library in python3 is used to create various forms of the object or entity in different in various conditions and image resolution is up scaled ESR-GAN to create a new data set to make this new item detectable by the computer vision model.

2.2 Technical Specifications

A 5V 3A power source is used to power the Raspberry Pi, it is interfaced with Pi V2 camera module and sensor array (Gas, Temperature, Humidity Sensor). It is used as a server for the LAN communication between the itself and the user's computer forming a chain of interconnected WiFi router.

A 5V 3A power source via switch is used to power the Arduino, for the autonomous mobile system it is interfaced with 4BO motors for mobility and ultrasonic sensor for detection of surrounding obstacles mounted over a servo motor to give ultrasonic sensor a 180 degree of view.

The Pi V2 camera is enabled and live streaming is done on the web-page by URL using the library Flask also for recording OpenCV is used. The received video feed from the Pi V2 camera is fed into the object classification, detection and segmentation algorithm (library used: OpenCV, Pixellib, YOLO) for object identification and marking as per the data set used.

2.3 Design Approach

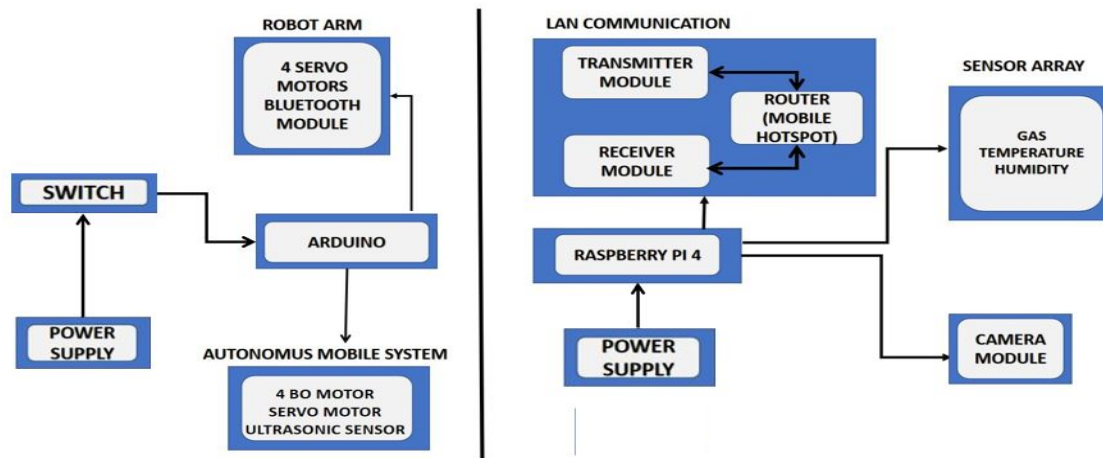


Fig 2.1 : Block diagram of the mobile bot

Fig 2.1 shows the block diagram of the Mobile Bot. Here the mobile bot comprises of 4 main modules.

- **Module 1 : LAN Communication** - It comprises of three parts, First part is the Transmitter module that's attached to the PC and helps to send command to the raspberry pi. The next one is the router (mobile hotspot) that helps to establish the communication between the PC and the Raspberry pi. The third part is the receiver module it helps the raspberry pi to receive the commands sent by the PC.
- **Module 2 : Sensor Array** - It has four sensors connected to the raspberry pi, they are camera, gas sensor, Temperature sensor and ultrasonic sensor. The camera is used for live stream and computer vision application, here we used the raspberry pi camera module 2. For gas sensor, temperature and humidity sensor we used MQ 2 and DHT11 sensors are being used respectively. The ultrasonic sensor is being used for driving the robot.
- **Module 3 : Autonomous mobile base** - It comprises of 4 BO motor to drive the robot and 1 ultrasonic sensor and servo motor to detect the object, also a 4DOF manipulator to collect samples. Both are driven with the help of Arduino UNO.

- **Module 4 : Data analysis and Visualization-** The acquired data from the environment sensing sensor array is plotted and visualized for better understanding of the terrain over a period of time. The visual data received is used in three parts, first is any unknown object or entity is detected it is augmented multiple times to create sampled and added to the data set for future use, second the visual data is used for object classification-detection-segmentation operation, third is Enhanced Super-Resolution Generative Adversarial Networks (ESRGANs)- generally GANs train two neural networks: the discriminator and the generator, simultaneously. The generator is to create fake images while the discriminator judges them as real or fake. SRGANs used this idea in the domain of image super-resolution. The generator produces super-resolution images, while the discriminator judges them as real and fake. Building on the foundation led by SRGANs, the ESRGAN's main aim is to introduce model modification such that the training is efficient and less complex. In the ESRGAN the generator has been Enhanced for better performance. The core ideology behind ESRGAN was not only to enhance the results but also to make the process far more efficient. The Block diagram of ESRGANs is attached below

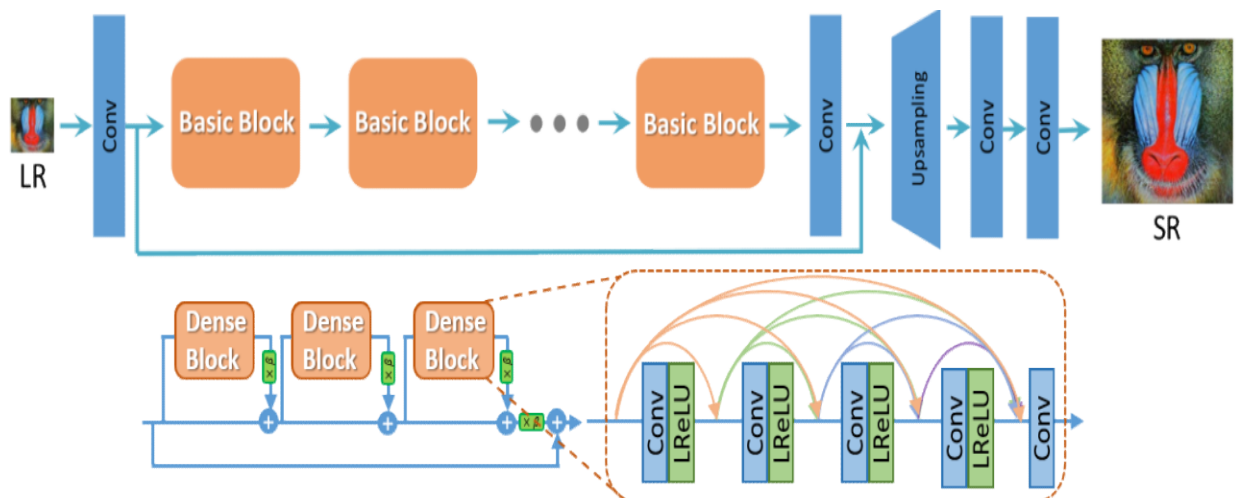


Fig 2.2 : ESRGAN block diagram

CH

CHAPTER 3: EXPERIMENTATION AND TESTS

3.1 Experimental Setup.

Fig 3.1, Fig 3.2, Fig 3.3, Fig 3.4 are pictures of the robot and manipulator. Both have an Arduino interfaced that drives the manipulator and bot using Ultrasonic sensor and BO motors and Raspberry placed above it that control all its sensor (except Ultrasonic sensor) the camera feed is transmitted over WiFi network to the PC where that is being recorded to implement computer vision in it.

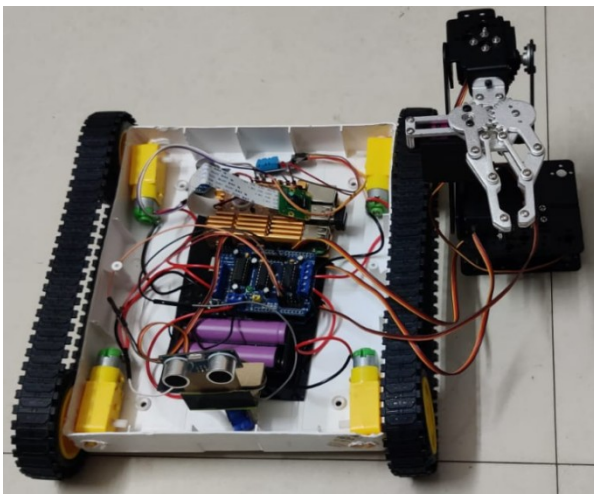


Fig 3.1: Expanded Setup with robotic arm



Fig 3.2 DOF manipulator for sample collection



Fig 3.3 :Side View of the Mobile Bot

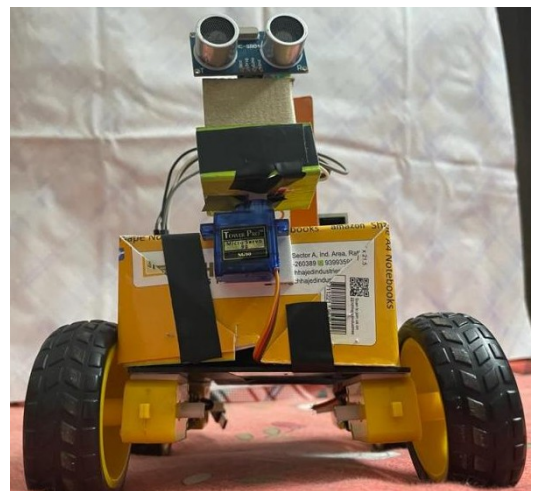


Fig 3.4 :Front View of the Mobile Bot

3.2 I Modeling, Circuits etc.

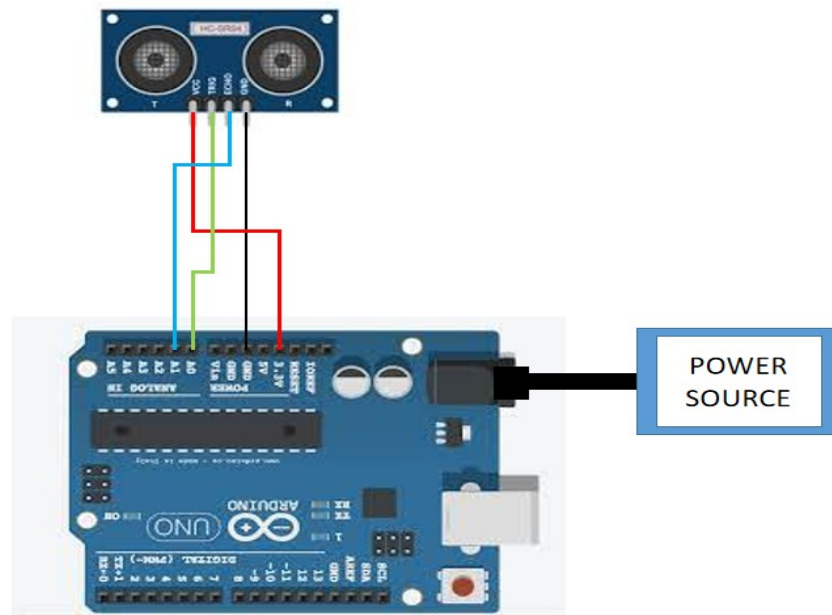


Fig3.5: Circuit Diagram of Microcontroller(Arduino)

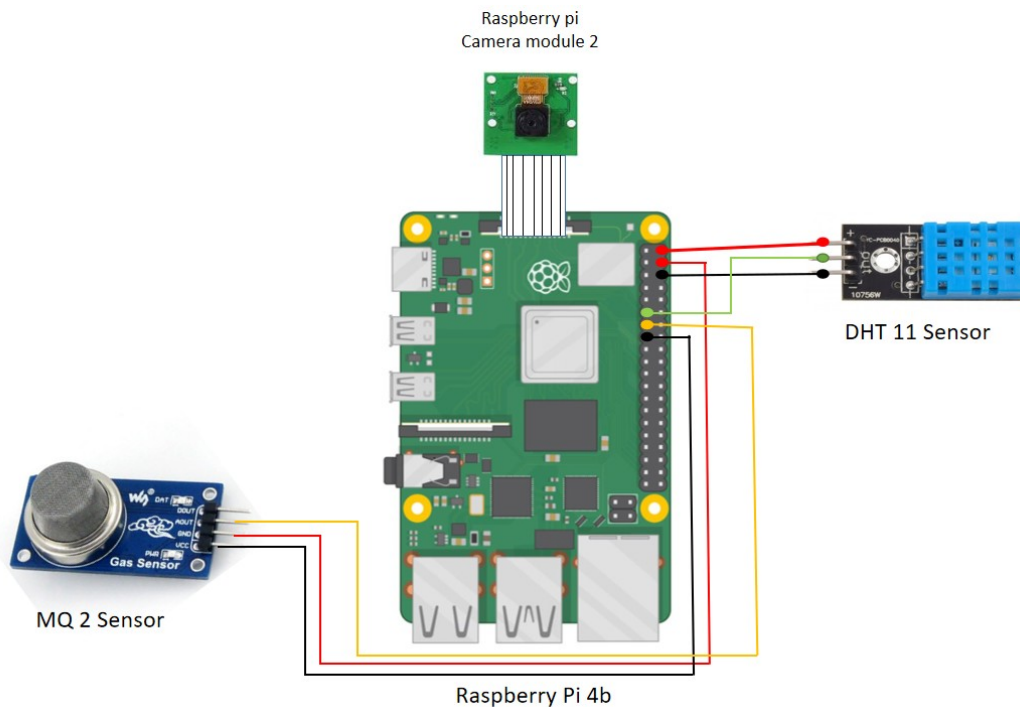


Fig3.6: Circuit diagram of Microprocessor(Raspberry Pi)

3.3 Prototype Testing/Simulations:

The autonomous robot vehicle automatically finds another way upon encountering an obstacle. A manipulator having 4DOF receives a signal from the transmitter (remote) through a Bluetooth module interfaced to it by bread board and Arduino. The sensors are actively collecting data from the environment and displaying the values on the Raspberry desktop.

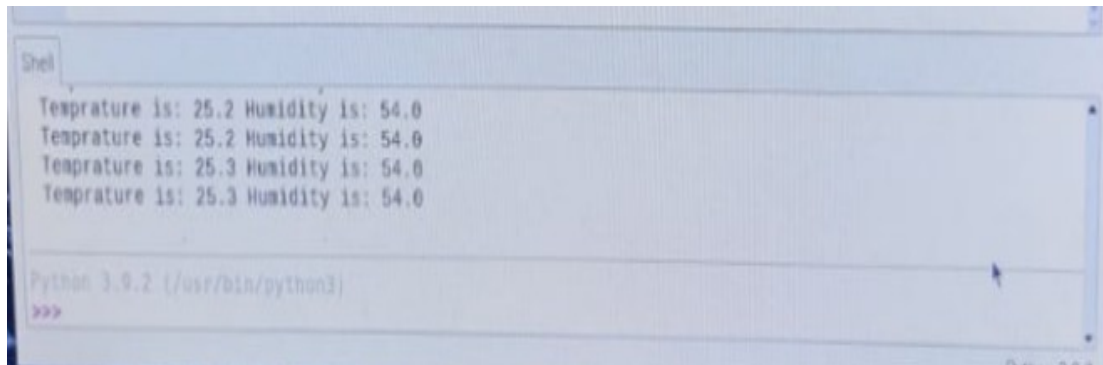


Fig 3.7: DHT-11 sensor output

In fig 3.7 the live temperature and humidity reading of DHT-11 sensors from the surrounding environment of robot is being presented on the output shell. The data is visualized for better understanding and decision making.

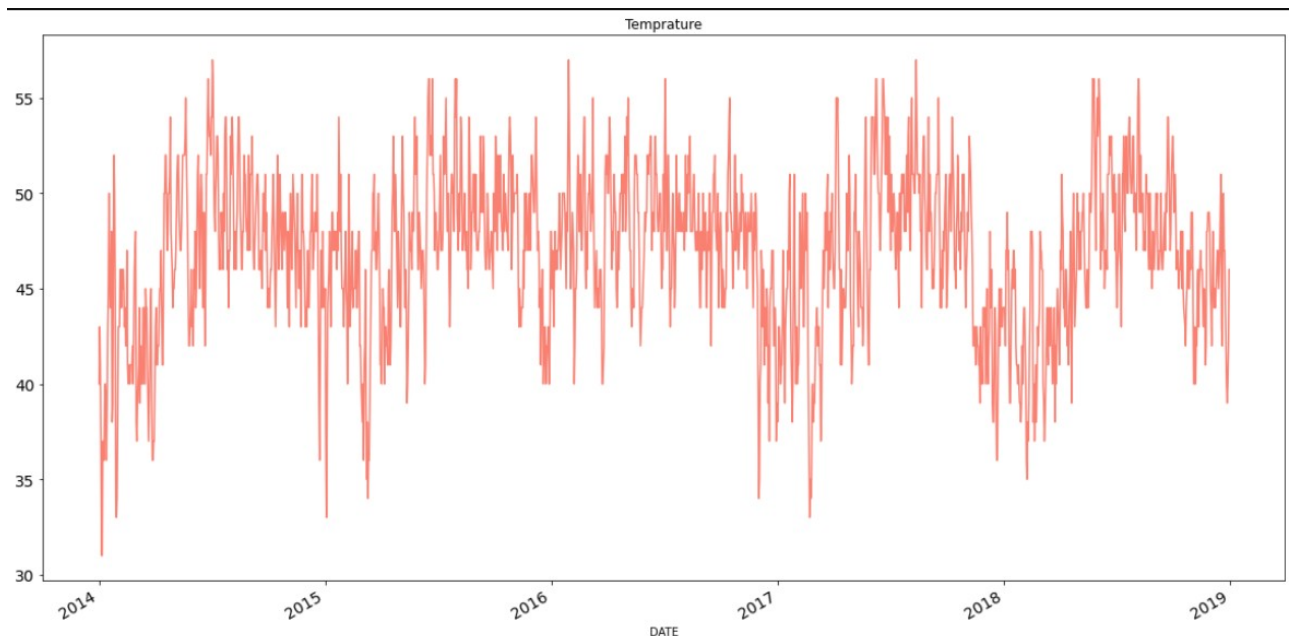




Fig 3.8: MQ-2 gas sensor output

In fig 3.8 the detection of gases of MQ-2 sensors from the surrounding environment of robot is being presented on the output shell.

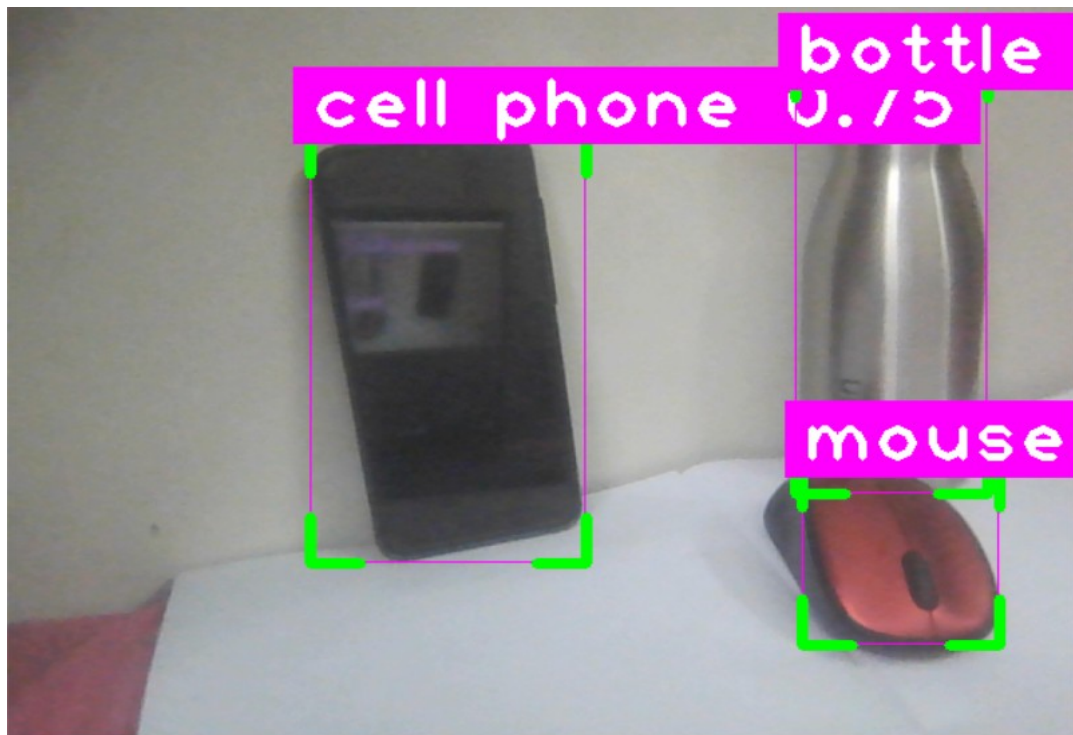


Fig 3.9: Object detection by pi camera module

In the fig 3.9 the received live streamed video is fed into the computer vision model which is doing the object detection and classification as per the given data set, and the next figure shows object classification and segmentation.

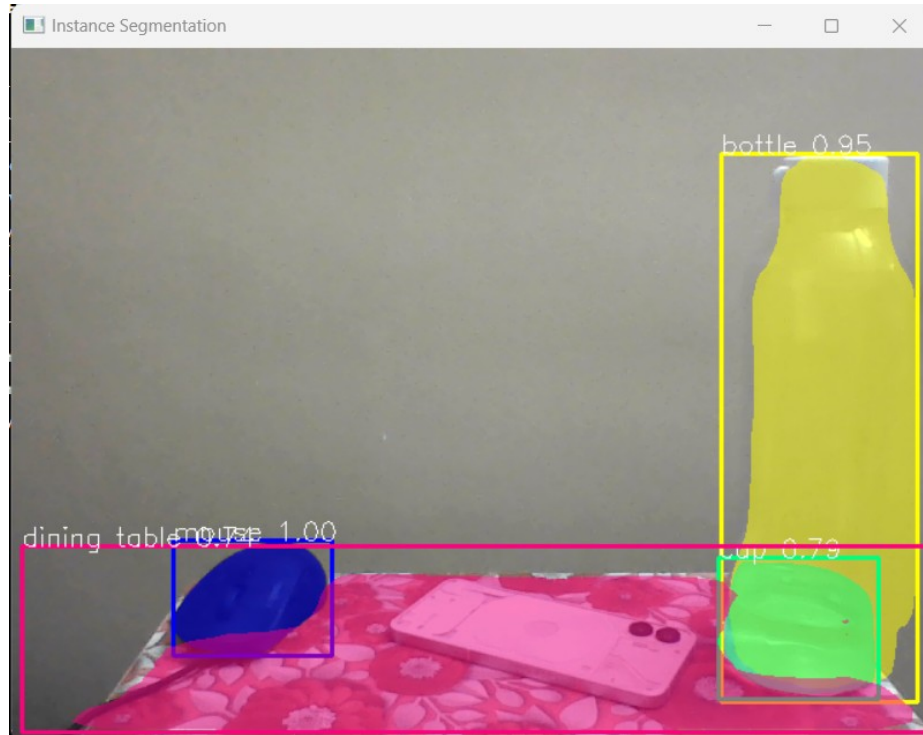


Fig 3.10: Object segmentation Output

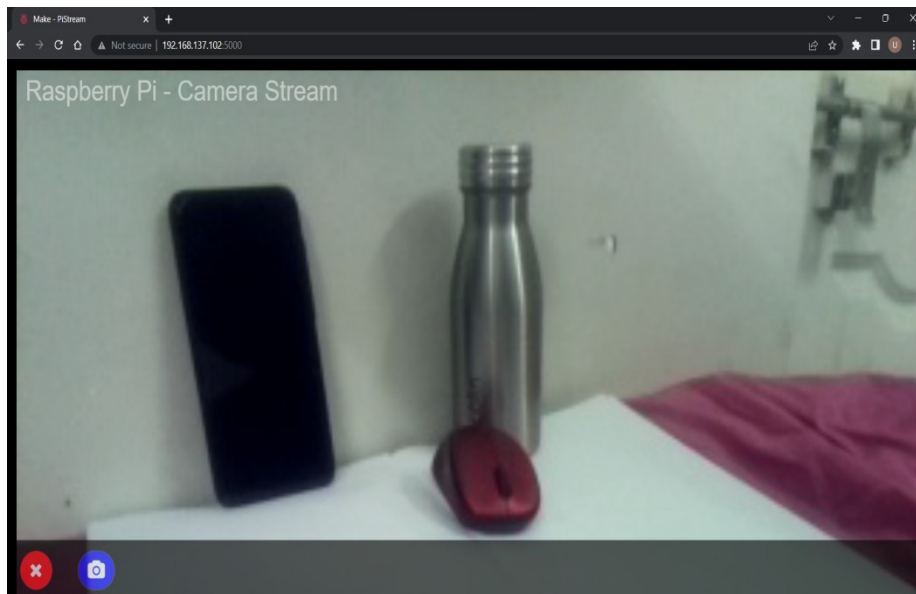


Fig 3.11: URL live steaming on web page

In fig 3.11 the Pi camera is enabled for recording and live streaming it on the given URL. The live stream is also used for the purpose of computer vision. If any unknown object or entity (let's say an unknown plant species) is observed by the human eye then that object's image is augmented multiple times and upscaled using ESR-GAN and then added to the data set to start detecting the unknown object or entity.



Fig 3.12:Original image sample

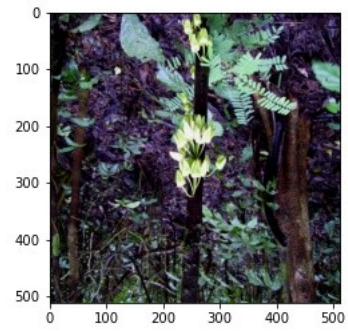


Fig3.13:Augmented image



Original(500x480)



ESRGANs output

Fig 3.14: ESRGAN input and output

CHAPTER 4: CHALLENGES AND REMEDY

4.1 Challenges Faced

We started off with the codes for all the sensors, we firstly did the coding for the camera module for the object detection functionality which was the most critical part as the output of the object detection was not accurate all the time. Secondly, we did the coding for the individual sensors with some trial and error. Then in 4DOF manipulator we had to test and calibrate the individual servos before assembling the complete system, and lastly, we faced a challenge in sourcing of parts at right price to meet our best possible needs.

We got errors in codes so we ran them multiple times to get the most optimal code for the smooth operation of our sensors. We used strong database in our ML code for object detection like 'coco_names' to make our model more efficient in object detection. To prevent malfunction in our manipulator we tested each servo stage wise before connecting them all together, then we did further calibration for smooth operation of the manipulator. Lastly, regarding the sourcing of parts for mobile robots, we contacted a Karnataka-based vendor who designed the required chassis for us at an efficient price which was most important for us from the project budget point of view.

4.2 Design Constraints

In this project our design is constrained by functionality of the Raspberry pi 4 which has 4 GB RAM and 32GB storage which is not sufficient for performing on object detection properly. Regular code are made for PC with fixed extensions where as same algorithm is applied in embedded system the previous extensions are not supported and the required extensions are not readily available processing power

4.3 Alternatives and Trade-offs

We can use a 3D printed body instead of a pre developed chassis to do better space management and make the robot more compact. We can use a better sensor array for more precise readings. Instead of regular wheels we can use caterpillar track for better movement in uneven terrains.

CHAPTER 5: RESULT AND DISCUSSION

5.1 Results Obtained

We tested all the sensors individually with our microprocessor Raspberry Pi 4 and after some calibration they are working efficiently. Lastly, after doing some adjustment and calibration our manipulator is functioning efficiently .Our code for sensors, actuators and overall manipulator is running so now we need to assemble all the components in the robot body and do test run.Our software functionalities(object classificatio-detection-segmentation, data visualization,live streaming,recording, augmentation and GAN) are working efficiently with good precision.

5.2 Analysis and Discussion

We had some failed attempts in our object segmentation and detection system but finally it was successfully functioning . our augmentation model had less precision and produced less diverse sample but that was fixed with augmentation function parameter adjustment. Our GAN and and data visualization model ran successfully.

CHAPTER 6: CONCLUSIVE REMARKS

6.1 Conclusion

A mobile autonomous robot that can avoid obstacles and choose the optimum path for exploration has been developed. It may be deployed in an uncharted terrain. It also has a camera module installed so that it can provide us real-time data via a URL and conduct object classification-detection-segmentation using the same live stream. It contains a variety of sensors implanted to provide information about the environment, such as temperature, humidity, and the presence of any dangerous gases, if observed any unknown object then it can be added to the data set by the image augmentation and GAN image upscaling. This robot is able to be put in the exploration sector, making exploration easier and safer for more people.

6.2 PLANNING AND PROJECT MANAGEMENT

Table 6.1 Showing details about project planning and management

Activity	Starting week	Number of weeks
Literature review	1 st week of July	4
Finalising problem	1 st week of August	1
Individual software component initial experimentation	2 nd week of August	2
Software system complete experimentation	1 st week of September	2
Software-hardware system complete development	3 rd week of September	2
Final testing of software-hardware components	4 th week of October	2

References

1. Dmitry Topolsky , Irina Topolskaya and Iuliia Plaksina, “Development of a Mobile Robot for Mine Exploration,” by , 2021.
2. Jong-Hoon Kim, “Design Concept and Motion Planning of a Single-Moduled Autonomous Pipeline Exploration Robot,” ,2022.
3. S.-G. Roh and H. R. Choi, “Differential-drive in-pipe robot for moving inside urban gas pipelines,” *Robotics, IEEE Transactions on*, vol. 21, no. 1, pp. 1–17, Feb. 2005.
4. J.-H. Kim, G. Sharma, and S. S. Iyengar, “FAMPER: A fully autonomous mobile robot for pipeline exploration,” in *IEEE-ICIT*, 2010.
5. Sirinart Tangruamsub; Manabu Tsuboyama, “Mobile robot vision-based navigation using self-organizing and incremental neural networks,” ,2009.

Appendix A: Gantt Chart:

Tasks	Months				
	July	August	September	October	November
Planning					
Research					
Design					
Implementation					
Flow up					

2Appendix B: Project Summary

Project Title	Development of Mobile Robot for Inspecting of Environment Condition in Hazardous Terrains
Team Members	Soumyo Jyoti mondal, Utkarsh Pandey
Supervisors	Prof. Suprava Patnaik and Prof. A. Bakshi
Semester / Year	VII / IV year
Project Abstract	<p>This paper describes the design and implementation of a model mobile robot for the purpose of inspection of environmental condition by going in a hazardous terrain. It is a unmanned vehicle capable of autonomous driving by avoiding the obstacles in the terrain,it senses the environment using sensor system ,a 4 DOF manipulator(robot arm)for collecting samples and computer vision algorithms for better understanding of the environment and live streaming the video feed on to web page using a onboard computer as a server. The robot uses a LAN communication protocol to extend its communication range by forming a chain interconnected WiFi modules where the starting and ending point being the onboard computer in the robot and the computer of the user. The screenshot of live stream is taken and those screenshot are used all enhancement program like image Augmentation,Image segmentation and ESRGAN.</p> <p>Keywords: mobile robot,computer, vision, manipulator, server, WiFi module, LAN, Image Augmentation, Image Segmentation, ESRGAN.</p>
List at least two significant realistic design constraints that are applied to your project.	The two primary constraints of this project are the hardware constraints that is chassis structure and component space management . The image input in ESRGAN should be small size or it will not work.
Briefly explain two significant trade-offs considered in your design, including options	<ul style="list-style-type: none"> • 3D modelled and printed body or chassis . • Higher precision sensor array.

considered and the solution chosen	<ul style="list-style-type: none"> • SLAM using ROS to implemetation for Map construction and localization
Describe the computing aspects, if any , of your project. Specifically identifying hardware-software trade-offs, interfaces, and/or interactions	The embedded C language codes have been written for implementing the autonomous driving system using Arduino. This software is compatible with Arduino. We are also using python3 for writing codes in raspberry pie and for object detection.

Appendix C: CODE

Flask camera code:

```
#Modified by smartbuilds.io
#Date: 27.09.20
#Desc: This web application serves a motion JPEG stream
# main.py
# import the necessary packages
from flask import Flask, render_template, Response, request, send_from_directory
from camera import VideoCamera
import os

pi_camera = VideoCamera(flip=False) # flip pi camera if upside down.

# App Globals (do not edit)
app = Flask(__name__)

@app.route('/')
def index():
    return render_template('index.html') #you can customize index.html here

def gen(camera):
    #get camera frame
    while True:
        frame = camera.get_frame()
        yield (b'--frame\r\n'
               b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n\r\n')

@app.route('/video_feed')
def video_feed():
    return Response(gen(pi_camera),
                    mimetype='multipart/x-mixed-replace; boundary=frame')

# Take a photo when pressing camera button
@app.route('/picture')
def take_picture():
    pi_camera.take_picture()
    return "None"
```

```
if __name__ == '__main__':
```

```
    app.run(host='0.0.0.0', debug=False)
```

DHT-11 sensor code:

```
import RPi.GPIO as GPIO
import dht11
import time
GPIO.setmode(GPIO.BCM)
myDHT=dht11.DHT11(pin=17)
try:
    while True:
        result=myDHT.read()
        if result.is_valid():
            print('Temperature is:',result.temperature,'Humidity is:',result.humidity)
            time.sleep(.2)
except KeyboardInterrupt:
    GPIO.cleanup()
    print('GPIO good to go')
```

MQ-2 gas sensor:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

# Define GPIO pins for the sensor and LED
sensor_pin = 27
led_pin = 25

# Set up the GPIO pins
GPIO.setup(sensor_pin, GPIO.IN)
GPIO.setup(led_pin, GPIO.OUT)

while True:
    # Read the sensor value
    sensor_value = GPIO.input(sensor_pin)

    # Turn on the LED if gas is detected
    if sensor_value == 1:
```

```

        GPIO.output(led_pin, GPIO.HIGH)
        print("Gas detected!")
    else:
        GPIO.output(led_pin, GPIO.LOW)
        print("No gas detected.")

# Wait for a short time before reading the sensor again
time.sleep(0.1)

```

OBJECT DETECTION:

```

from ultralytics import YOLO
import cv2
import cvzone
import math
import numpy as np

model = YOLO('yolov8n.pt')
classNames = ["person", "bicycle", "car", "motorbike", "aeroplane", "bus", "train", "truck", "boat",
              "traffic light", "fire hydrant", "stop sign", "parking meter", "bench", "bird", "cat",
              "dog", "horse", "sheep", "cow", "elephant", "bear", "zebra", "giraffe", "backpack", "umbrella",
              "handbag", "tie", "suitcase", "frisbee", "skis", "snowboard", "sports ball", "kite", "baseball bat",
              "baseball glove", "skateboard", "surfboard", "tennis racket", "bottle", "wine glass", "cup",
              "fork", "knife", "spoon", "bowl", "banana", "apple", "sandwich", "orange", "broccoli",
              "carrot", "hot dog", "pizza", "donut", "cake", "chair", "sofa", "pottedplant", "bed",
              "diningtable", "toilet", "tvmonitor", "laptop", "mouse", "remote", "keyboard", "cell phone",
              "microwave", "oven", "toaster", "sink", "refrigerator", "book", "clock", "vase", "scissors",
              "teddy bear", "hair drier", "toothbrush"
]

#cap = cv2.VideoCapture("http://192.168.218.102:8080/video")
#cap = cv2.VideoCapture("C:\\Users\\avikd\\Documents\\Python programming\\Untitled Folder\\Object-
Detection-101\\Videos\\bikes.mp4")
cap = cv2.VideoCapture(0)
cap.set(3, 640)
cap.set(4, 480)

while True:
    success, img = cap.read()
    results = model(img, stream = True)
    for r in results:
        boxes = r.boxes
        for box in boxes:
            x1,y1,x2,y2 = box.xyxy[0]
            x1,y1,x2,y2 = int(x1), int(y1),int (x2), int(y2)

            w,h = x2-x1,y2-y1

```



```

cvzone.cornerRect(img, (x1,y1,w,h))

#Confidence
conf = math.ceil((box.conf[0]*100))/100

#class name
cls = int(box.cls[0])
cvzone.putTextRect(img,f'{classNames[cls]} {conf}',(max(0, x1),max(50,y1)))

cv2.imshow("Image", img)
if cv2.waitKey(1) == ord("q"):
    break;

cap.release()
cv2.destroyAllWindows()

```

OBSTACLE AVOIDING:

```

#include <AFMotor.h>
#include <NewPing.h>
#include <Servo.h>

#define TRIG_PIN A0
#define ECHO_PIN A1
#define MAX_DISTANCE 200
#define MAX_SPEED 255 // sets speed of DC motors //190->90
#define MAX_SPEED_OFFSET 20

NewPing sonar(TRIG_PIN, ECHO_PIN, MAX_DISTANCE);

AF_DCMotor motor1(1, MOTOR12_1KHZ);
AF_DCMotor motor2(2, MOTOR12_1KHZ);
AF_DCMotor motor3(3, MOTOR34_1KHZ);
AF_DCMotor motor4(4, MOTOR34_1KHZ);
Servo myservo;

boolean goesForward=false;
int distance = 100;
int speedSet = 0;

void setup() {

```

```

myservo.attach(10);
myservo.write(115);
delay(2000);
distance = readPing();
delay(100);
distance = readPing();
delay(100);
distance = readPing();
delay(100);
distance = readPing();
delay(100);
}

void loop() {
  int distanceR = 0;
  int distanceL = 0;
  delay(40);

  if(distance<=20)//15->20
  {
    moveStop();
    delay(100);
    moveBackward();
    delay(300);
    moveStop();
    delay(200);
    distanceR = lookRight();
    delay(200);
    distanceL = lookLeft();
    delay(200);

    if(distanceR>=distanceL)
    {
      turnRight();
      moveStop();
    }else
    {
      turnLeft();
      moveStop();
    }
  }else
  {
    moveForward();
  }
}

```

```

distance = readPing();
}

int lookRight()
{
    myservo.write(50);
    delay(500);
    int distance = readPing();
    delay(100);
    myservo.write(115);
    return distance;
}

int lookLeft()
{
    myservo.write(170);
    delay(500);
    int distance = readPing();
    delay(100);
    myservo.write(115);
    return distance;
    delay(100);
}

int readPing() {
    delay(70);
    int cm = sonar.ping_cm();
    if(cm==0)
    {
        cm = 250;
    }
    return cm;
}

void moveStop() {
    motor1.run(RELEASE);
    motor2.run(RELEASE);
    motor3.run(RELEASE);
    motor4.run(RELEASE);
}

void moveForward() {

    if(!goesForward)

```

```

{
  goesForward=true;
  motor1.run(FORWARD);
  motor2.run(FORWARD);
  motor3.run(FORWARD);
  motor4.run(FORWARD);
  for (speedSet = 0; speedSet < MAX_SPEED; speedSet +=2) // slowly bring the speed up to avoid
loading down the batteries too quickly
  {
    motor1.setSpeed(speedSet);
    motor2.setSpeed(speedSet);
    motor3.setSpeed(speedSet);
    motor4.setSpeed(speedSet);
    delay(5);
  }
}

```

```

void moveBackward() {
  goesForward=false;
  motor1.run(BACKWARD);
  motor2.run(BACKWARD);
  motor3.run(BACKWARD);
  motor4.run(BACKWARD);
  for (speedSet = 0; speedSet < MAX_SPEED; speedSet +=2) // slowly bring the speed up to avoid
loading down the batteries too quickly
  {
    motor1.setSpeed(speedSet);
    motor2.setSpeed(speedSet);
    motor3.setSpeed(speedSet);
    motor4.setSpeed(speedSet);
    delay(5);
  }
}

```

```

void turnRight() {
  motor1.run(FORWARD);
  motor2.run(FORWARD);
  motor3.run(BACKWARD);
  motor4.run(BACKWARD);
  delay(1000);
  motor1.run(FORWARD);
  motor2.run(FORWARD);
  motor3.run(FORWARD);
}

```

```

    motor4.run(FORWARD);
}

void turnLeft() {
    motor1.run(BACKWARD);
    motor2.run(BACKWARD);
    motor3.run(FORWARD);
    motor4.run(FORWARD);
    delay(1000);
    motor1.run(FORWARD);
    motor2.run(FORWARD);
    motor3.run(FORWARD);
    motor4.run(FORWARD);
}

```

Bluetooth controlled robot arm

```

#include <Servo.h>
#define SERVO_BASE      2
#define SERVO_SHOULDER  3
#define SERVO_ELBOW     4
#define SERVO_GRIPPER   5

Servo myservo_1; // create servo object to control a servo
Servo myservo_2;
Servo myservo_3;
Servo myservo_4;

unsigned char Data_String[25],Data_Index = 0,New_Data_Rec_Flag = 0;
unsigned int Received_Servo_Value[4],Final_Servo_Val[4];
unsigned char Index_i = 0,Index_j = 0,Counter_to_Refresh = 0;

void setup() {
    Serial.begin(9600);
    myservo_1.attach(SERVO_BASE);    // attaches the servo on pin 2 to the servo object
    myservo_2.attach(SERVO_SHOULDER); // attaches the servo on pin 3 to the servo object
    myservo_3.attach(SERVO_ELBOW);   // attaches the servo on pin 4 to the servo object
    myservo_4.attach(SERVO_GRIPPER); // attaches the servo on pin 5 to the servo object

    myservo_1.write(90);
    delay(200);
    myservo_2.write(90);
    delay(200);
    myservo_3.write(90);

```

```

delay(200);
myservo_4.write(90);
delay(200);

Received_Servo_Value[0] = 90; // Default values
Received_Servo_Value[1] = 90;
Received_Servo_Value[2] = 90;
Received_Servo_Value[3] = 45;

Final_Servo_Val[0] = 90; // Default values
Final_Servo_Val[1] = 90;
Final_Servo_Val[2] = 90;
Final_Servo_Val[3] = 45;
}

void loop()
{
  if(New_Data_Rec_Flag==0)
  {
    if (Serial.available()) // check whether bluetooth data is available
    {
      // read incoming serial data:
      char inChar = Serial.read(); // read bluetooth data one by one
      //Serial.print(inChar);
      if(inChar==0x0A) // End character of bluetooth data
      {
        Data_String[Data_Index] = inChar;
        Data_Index = 0;
        New_Data_Rec_Flag = 1;
      }
    }
    else
    {
      if(inChar!=0x2C) // removing ascii value, except for 0x2C which is ascii for ,
      {
        inChar = inChar - 0x30; // removing Ascii value of 0, so that we can get exact value
      }
      Data_String[Data_Index] = inChar;
      Data_Index++;
    }
  }
}

if(New_Data_Rec_Flag==1)
{
  Received_Servo_Value[0] = 0;
  Received_Servo_Value[1] = 0;
  Received_Servo_Value[2] = 0;
  Received_Servo_Value[3] = 0;
}

```

```

for(Index_i = 0, Index_j = 0;;)
{
    if(Data_String[Index_j]==0x2C)
    {
        Index_j++;
        Index_i++;
        Serial.print("A ");
    }
    else if(Data_String[Index_j]==0x0A)
    {
        New_Data_Rec_Flag = 0;
        Serial.print("B ");
        break;
    }
    else
    {
        Received_Servo_Value[Index_i] = Received_Servo_Value[Index_i] * 10 + Data_String[Index_j];
        Index_j++;
        Serial.print("C ");
    }
}
Serial.print(Received_Servo_Value[0]);
Serial.print(" ");
Serial.print(Received_Servo_Value[1]);
Serial.print(" ");
Serial.print(Received_Servo_Value[2]);
Serial.print(" ");
Serial.println(2*Received_Servo_Value[3]);
}

Counter_to_Refresh++;
delay(1);
if(Counter_to_Refresh >= 10) // delay of 10 msec = 1 msec * 10, this will allow smooth movement of servos
{
    Counter_to_Refresh = 0;
    if(Received_Servo_Value[0] != Final_Servo_Val[0])
    {
        if(Received_Servo_Value[0] > Final_Servo_Val[0])
        {
            Final_Servo_Val[0]++;
        }

        if(Received_Servo_Value[0] < Final_Servo_Val[0])
        {
            Final_Servo_Val[0]--;
        }
        myservo_1.write(180 - Final_Servo_Val[0]); // adjuted as per app
    }
}

```

```

    }

    if(Received_Servo_Value[1]!=Final_Servo_Val[1])
    {
        if(Received_Servo_Value[1]>Final_Servo_Val[1])
        {
            Final_Servo_Val[1]++;
        }

        if(Received_Servo_Value[1]<Final_Servo_Val[1])
        {
            Final_Servo_Val[1]--;
        }
        myservo_2.write(Final_Servo_Val[1]);
    }

    if(Received_Servo_Value[2]!=Final_Servo_Val[2])
    {
        if(Received_Servo_Value[2]>Final_Servo_Val[2])
        {
            Final_Servo_Val[2]++;
        }

        if(Received_Servo_Value[2]<Final_Servo_Val[2])
        {
            Final_Servo_Val[2]--;
        }
        myservo_3.write(Final_Servo_Val[2]);
    }

    if(Received_Servo_Value[3]!=Final_Servo_Val[3])
    {
        if(Received_Servo_Value[3]>Final_Servo_Val[3])
        {
            Final_Servo_Val[3]++;
        }

        if(Received_Servo_Value[3]<Final_Servo_Val[3])
        {
            Final_Servo_Val[3]--;
        }
        myservo_4.write(180 - (2*Final_Servo_Val[3])); // adjuted as per app
    }
}
}

```