# MY ROVEROID

Project Guide: Prof. John Pradeep Darsy

Project Id:17WSEC278

BY:

Student Coordinator: Manjari Bisht-17BCE7075

Members: Rohit Bhargav Peesa-17BEC7024

Manoj Kumar Pydimarri-17BEV7012

Rutvi Boda-17BCD7018

Sai Pranay Reddy- 17BCD7021

**CONTENTS**

1. Hardware Requirements
2. Software Requirements
3. Tools
4. Circuit diagrams
5. Specifications
6. Future Scope
7. References

**Hardware Requirements:**

1. Arduino uno
2. Raspberry PI-3B
3. Motor Drivers-L298N\*2
4. Gyro sensor-GY521
5. Servo motors SG-90\*4
6. Potentiometer\*4
7. Breadboard
8. Jumper wires
9. Motor Wheels

10.Camera module-V2.1 8MP

**Software Requirements:**

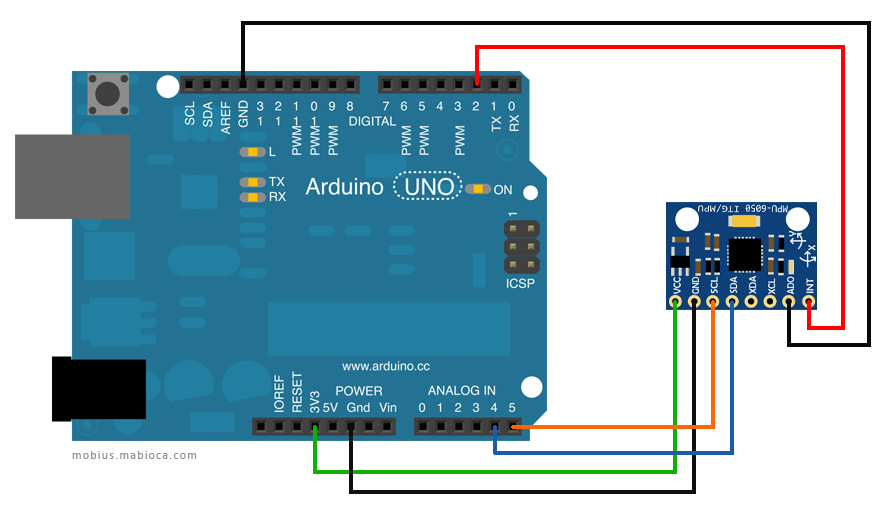
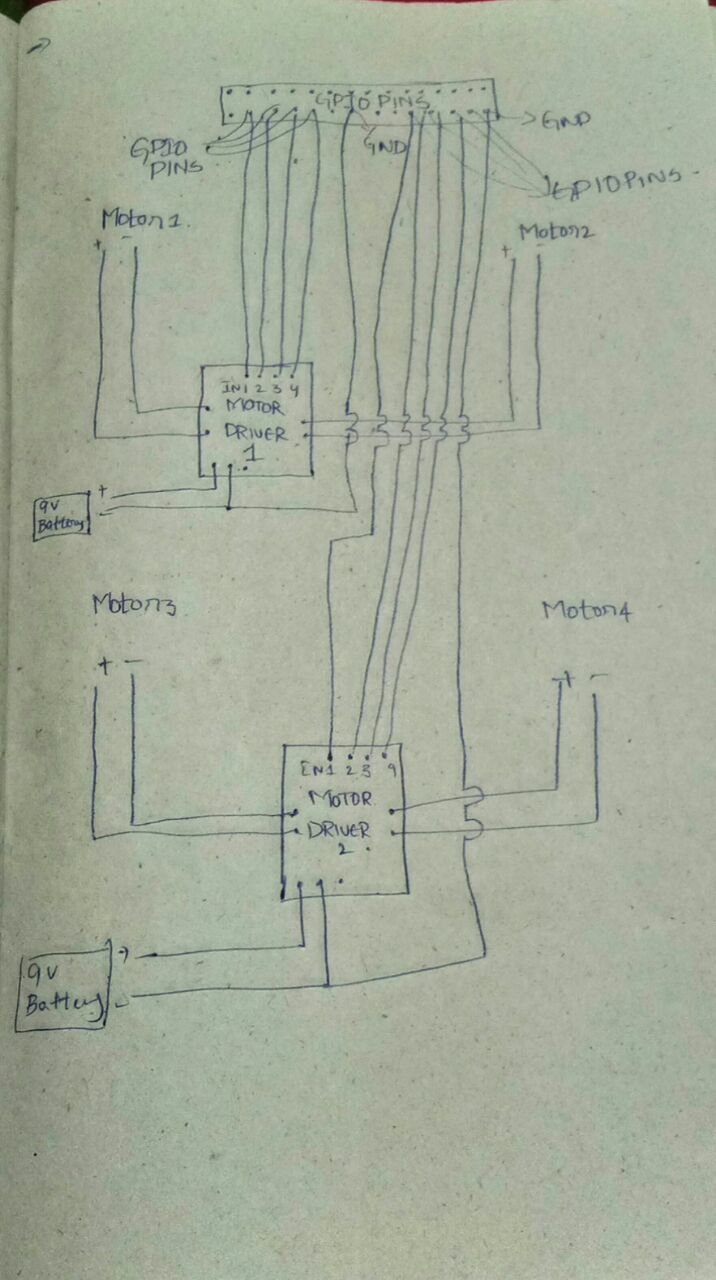
1. Arduino IDE
2. Raspbian Jessie
3. Tensorflow
4. OpenCV
5. Python
6. MIT APP inventor
7. Libraries:

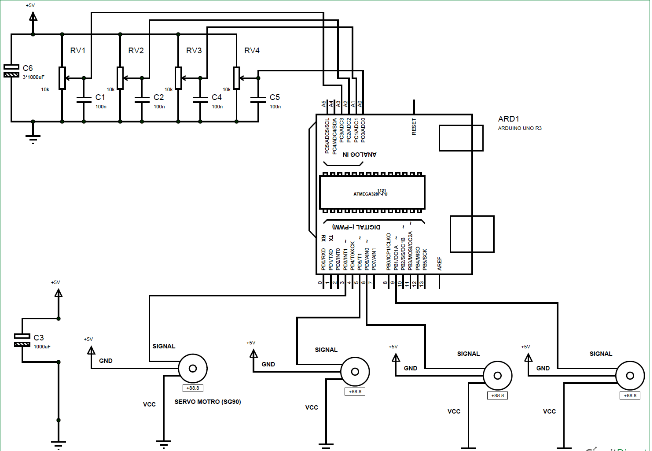
* Smbus
* Mpu6050
* Imutils
* Matplotlib
* Lxml
* Scipy
* Jupyter

Tools:

* Soldering iron
* Glue gun

***Circuit diagrams:***



**Arduino specifications:**

|  |  |
| --- | --- |
| Microcontroller | [ATmega328P](http://www.atmel.com/Images/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf) |
| Operating Voltage | 5V |
| Input Voltage (recommended) | 7-12V |
| Input Voltage (limit) | 6-20V |
| Digital I/O Pins | 14 (of which 6 provide PWM output) |
| PWM Digital I/O Pins | 6 |
| Analog Input Pins | 6 |
| DC Current per I/O Pin | 20 mA |
| DC Current for 3.3V Pin | 50 mA |
| Flash Memory | 32 KB (ATmega328P) of which 0.5 KB used by bootloader |
| SRAM | 2 KB (ATmega328P) |

**Raspberry PI specifications:**

The Raspberry Pi 3 is the third-generation Raspberry Pi. It replaced the Raspberry Pi 2 Model B in February 2016.

* Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
* 1GB RAM
* BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board
* 40-pin extended GPIO
* 4 USB 2 ports
* 4 Pole stereo output and composite video port
* Full size HDMI
* CSI camera port for connecting a Raspberry Pi camera
* DSI display port for connecting a Raspberry Pi touchscreen display
* Micro SD port for loading your operating system and storing data
* Upgraded switched Micro USB power source up to 2.5A

**GY-521 specifications:**

Color: Blue- Material: PCB + Plastic + copper- Chip: MPU-6050- Power supply: 3~5V- Communication mode: standard IIC communication protocol- Chip built-in 16bit AD converter, 16bit data output- Gyroscopes range: +/- 250 500 1000 2000 degree/sec- Acceleration range: +/- 2g, +/- 4g, +/- 8g, +/- 16g- Immersion Gold plating PCB, machine welding process to ensure quality- Pin pitch: 2.54mm- Great for DIY projects- Packing list:- 1 x Module- 2 x Pins

**IDEA BEHIND THE PROJECT**

The main motto of develop a roveroid capable of supporting various kinds of assignments of detecting various objects and informing user whether they are of our requirement and to pick them or not. It should also be automated and controlled from distance.the project is to

Summary:

This is a project in which we have made Rover. It can move forward and backward, left and right. Here we would be controlling motor and 4 wheels using Gyro sensor. Later we have written codes for our requirements and input them onto Raspberry PI and Arduino using respective methodologies. We need to connect PI to hotspot of android phone, we can see live in any one devices connected to hotspot. We can also view live transmission generated by typing IP address in search of internet explorer or chrome. We have added a camera to it so that it can be used to view places where rover is going using onboard Bluetooth of PI and cellular device. We have attached robotic arm to base of rover whose servos would be controlled by potentiometers. The movements of the robotic arm are controlled by the help of servo motors. Servo motors are being used here as it has a precise control over its velocity position and acceleration. Bluetooth has been used as a medium of connection between the raspberry pi and the arduino. We have also added image processing ability to it but only problem here is if we need imaghe processing we need to stop the live transmission. The app developed is capable of showing the live video as is shot by the camera. The app can also be used to control the movements of the rover.

***FUTURE SCOPE:***

* We can add GPS module to find location, using GPS we can set a location and send rover to specific location.
* We can add temperature, humidity and pressure then make it like small mobile weather station.
* We can not only recognise present requirement but we can add various requirements like face recognition, using touch sensor we can make it encrypted.
* We can control whole project using Alexa as voice based system.
* We can add various sensors and satisfy various outputs to use it for various requirements
* We can make it automated completely.
* We can make use of on broad Bluetooth and use gyro from distance also.

***Codes:***

Robotic Arm-

#include<Servo.h>

Servo servobase;

Servo servoelbow;

Servo servowrist;

Servo servoclaw;

int potmeter1=A1;

int potmeter2=A2;

int potmeter3=A3;

int potmeter4=A4;

int ValPotm1;

int ValPotm2;

int ValPotm3;

int ValPotm4;

void setup()

{

servobase.attach(3);

servo2.attach(5);

servo3.attach(6);

servo4.attach(9);

}

void loop()

{

ValPotm1=analogRead(potmeter1);

ValPotm1=map(ValPotm1,0,1023,0,180);

servobase.write(ValPotm1);

delay(15);

ValPotm2=analogRead(potmeter2);

ValPotm2=map(ValPotm2,0,1023,0,180);

servoelbow.write(ValPotm2);

delay(15);

ValPotm3=analogRead(potmeter3);

ValPotm3=map(ValPotm3,0,1023,0,180);

servowrist.write(ValPotm3);

delay(15);

ValPotm4=analogRead(potmeter4);

ValPotm4=map(ValPotm4,0,1023,0,180);

servoclaw.write(ValPotm4);

delay(15);

}

Rover:

from mpu6050 import mpu6050 #Importing mph module and time function and GPIO modules

import time

import RPi.GPIO as GPIO

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

#GPIO config to BCM mode

GPIO.setup(6, GPIO.OUT)

#Assigning Output pins

GPIO.setup(13, GPIO.OUT)

GPIO.setup(19, GPIO.OUT)

GPIO.setup(26, GPIO.OUT)

sensor = mpu6050(0x69)

#assigning address as 69

while True:

accelerometer\_data = sensor.get\_accel\_data()

#Calling the function to measure the sensor data

print accelerometer\_data

x=accelerometer\_data['x']

#Assigning the sensor axis values to different variables

y=accelerometer\_data['y']

# for further processing

z=accelerometer\_data['z']

print x

print y

print z

#STOP

if (z>10.5):

#Threshold value for Stop and other positions like below

D0=GPIO.output(6, GPIO.LOW)

D1=GPIO.output(13, GPIO.LOW)

D2=GPIO.output(19, GPIO.LOW)

D3=GPIO.output(26, GPIO.LOW)

print 'STOP'

#forward

elif (x<-2):

D0=GPIO.output(6, GPIO.HIGH)

D1=GPIO.output(13, GPIO.LOW)

#From the configuration table you can match the values for each motion

D2=GPIO.output(19, GPIO.HIGH)

D3=GPIO.output(26, GPIO.LOW)

print 'FORWARD'

time.sleep(0.5)

#backward

elif (x>3):

D0=GPIO.output(6, GPIO.LOW)

D1=GPIO.output(13, GPIO.HIGH)

D2=GPIO.output(19, GPIO.LOW)

D3=GPIO.output(26, GPIO.HIGH)

print 'BACKWARD'

time.sleep(0.5)

# giving delay for the motor to react

#RIGHT

elif (y>0):

D0=GPIO.output(6, GPIO.LOW)

D1=GPIO.output(13, GPIO.HIGH)

D2=GPIO.output(19, GPIO.HIGH)

D3=GPIO.output(26, GPIO.LOW)

print 'RIGHT'

time.sleep(0.5)

#LEFT

elif (y<0):

D0=GPIO.output(6, GPIO.HIGH)

D1=GPIO.output(13, GPIO.LOW)

D2=GPIO.output(19, GPIO.LOW)

D3=GPIO.output(26, GPIO.HIGH)

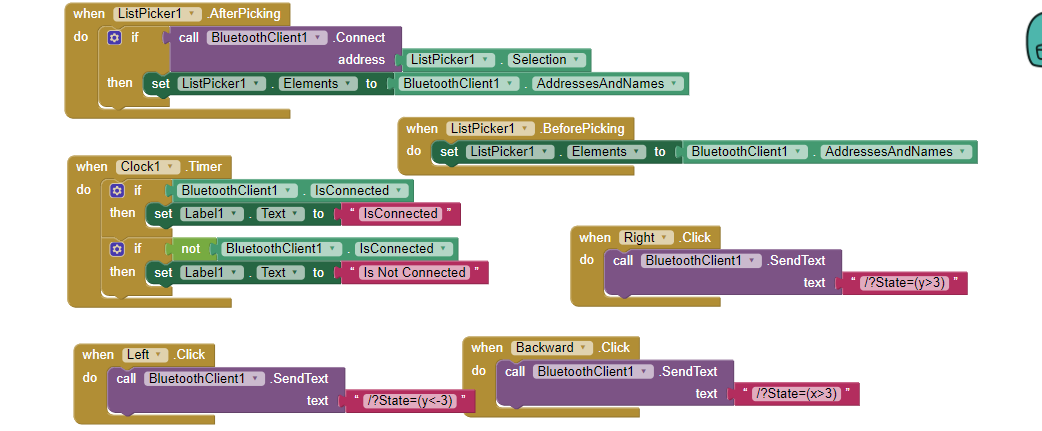
print 'LEFT'

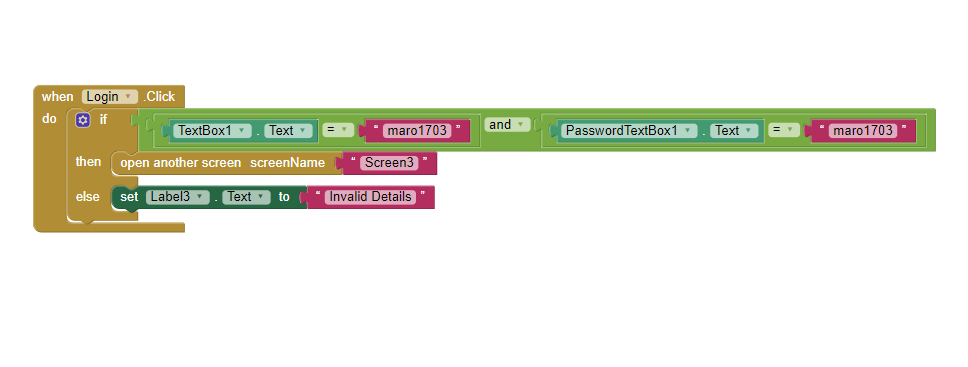
time.sleep(.5)

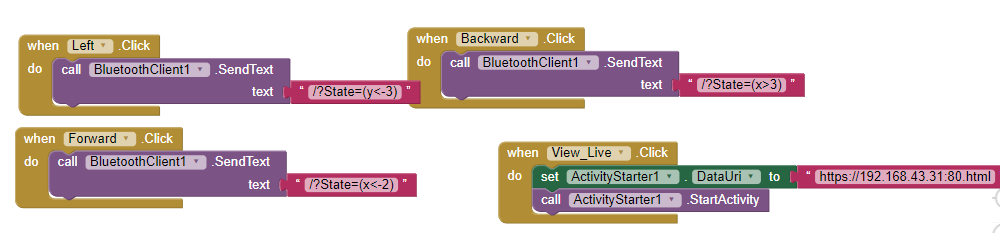
time.sleep(1)

#Giving delay for measurement

App-







**References:**

www.tensorflow.org

www.hackster.io

www.opencv.org

www.github.com

www.circuitdigest.com