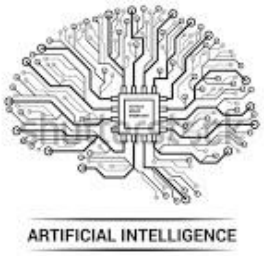


SHYAKA Aimé

Work:

- **Human following robot with ARDUINO(C++)**
- **Human following Robot in PYTHON**

Content

- Introduction and Work description
- CIRCUIT Diagram and Algorithm of Robot
- Hardware requirements
- Software requirements
- CODES(C/C++)
- CODES(PYTHON)
- Procedures
- conclusion
- references

- **Introduction and work description**

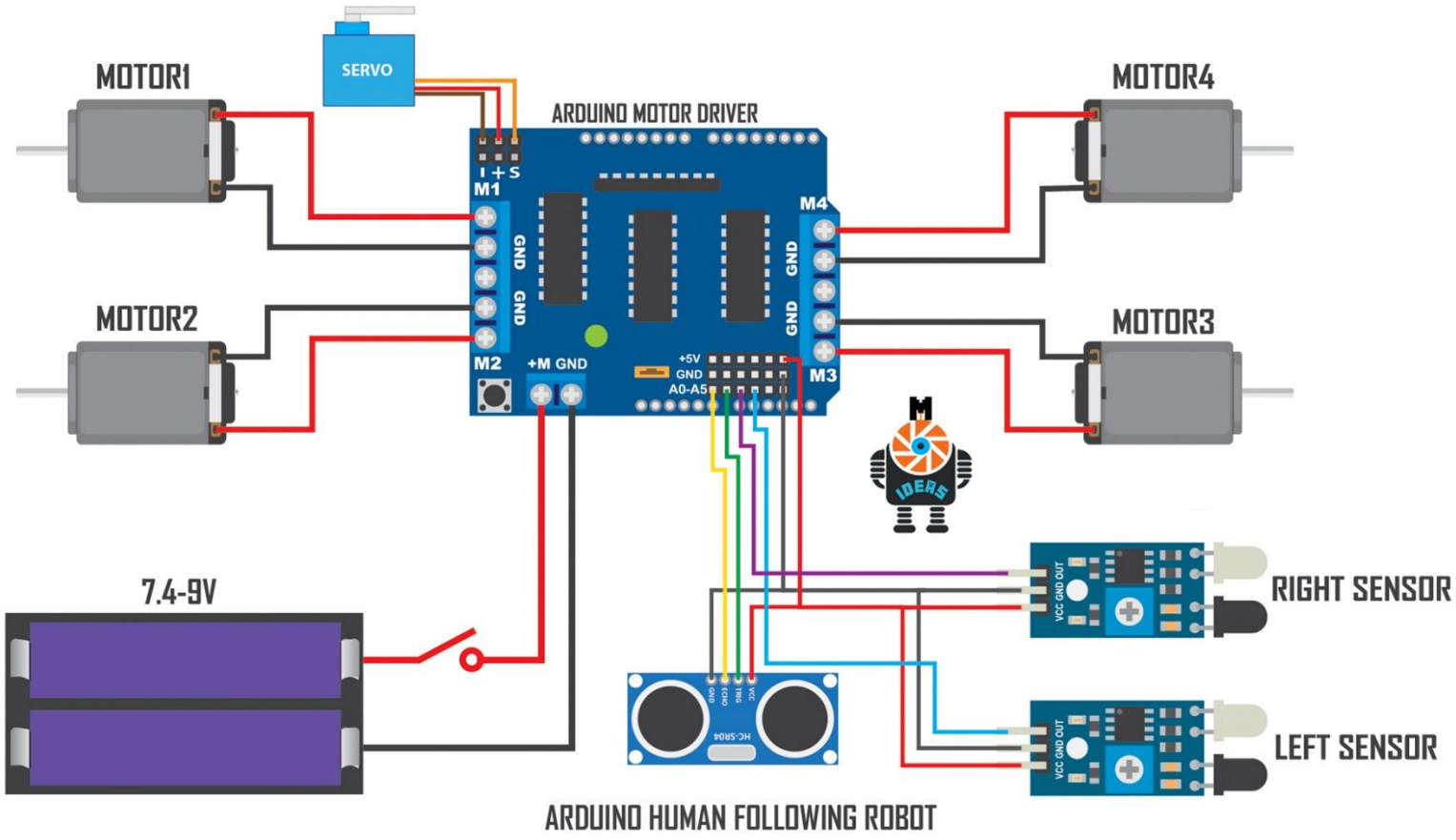
Human following robot is robot that uses 2 IR sensors and an ultrasonic sensor. IR sensors used to follow the human or object ,ultrasonic sensor is used to move back the robot. And it is programmed in C++ (also can be coded in PYTHON)

• Introduction and work description

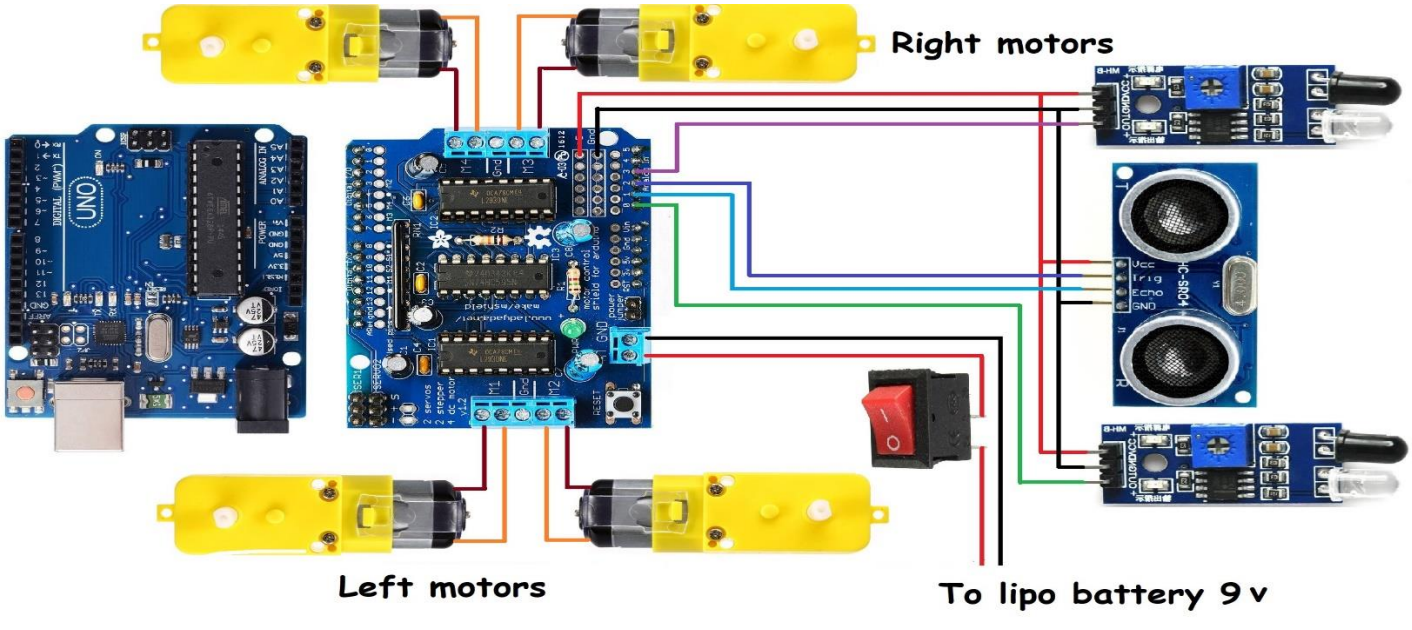
Main requirements and their role

1. **Arduino Uno:** The digital and analog input/output pins equipped in the board can be interfaced to various expansion boards and other circuits. Serial communication interface is a feature in this board, including USB which will be used to load the programs from computer.
2. **Voltage regulator:** It is used to maintain a constant voltage level and also used for regulating AC or DC voltages. A voltage regulator contain negative feed-forward design or it may also contain negative feedback control loops.
3. **RF Module:** The RF (radio frequency) module will have 2 units which acts as the transmitter and receiver. It receives the signals from the user and controls the actuation of the robot accordingly.
4. **Ultrasonic Sensor:** It is capable of sensing motion of the human and therefore it is also called as a motion sensor. Whenever a human pass through this sensor it will automatically sense the motion through IR radiation and send the data to the microcontroller.

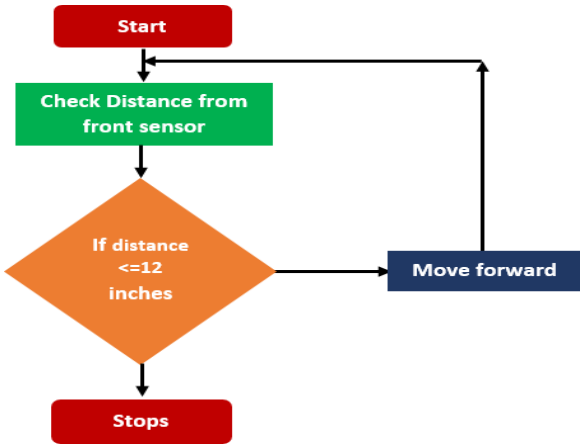
•Circuit Diagram and Algorithm of Robot



•Circuit Diagram and Algorithm of Robot

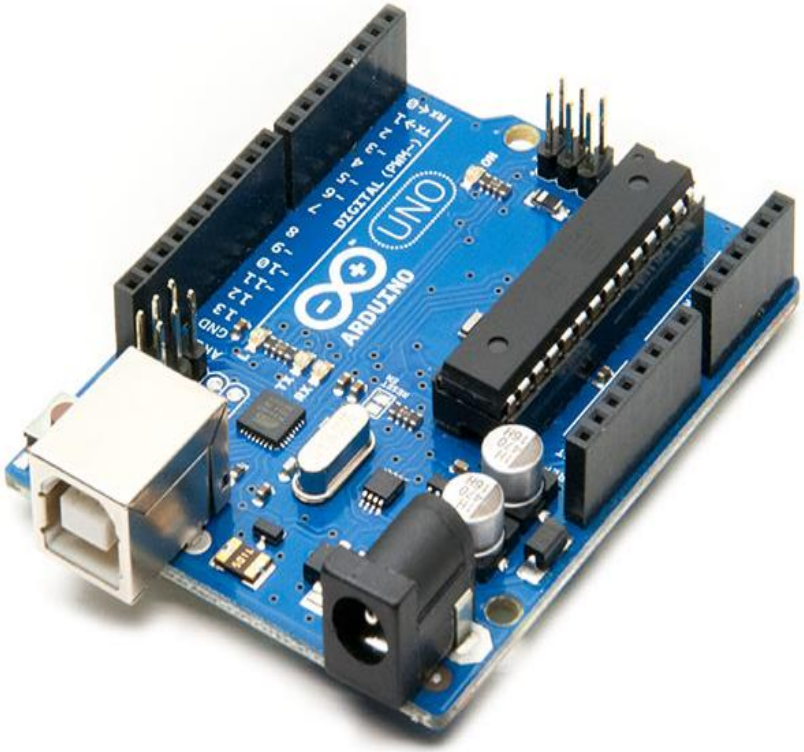


Algorithm of Robot



•Hardware requirements

- Microcontroller board – Arduino Uno



- Wheels & TT gear motor(4x)

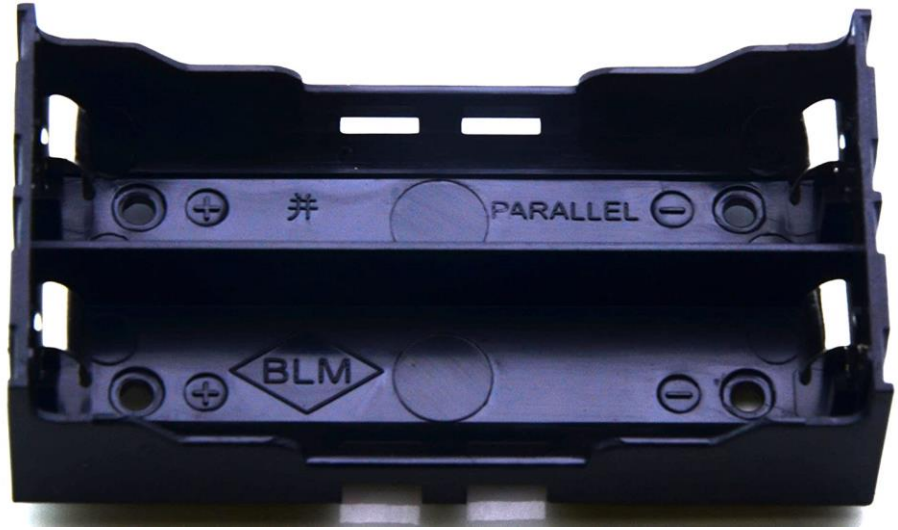
- Servo motor

- Ultrasonic sensor

- Infrared sensor(2x)

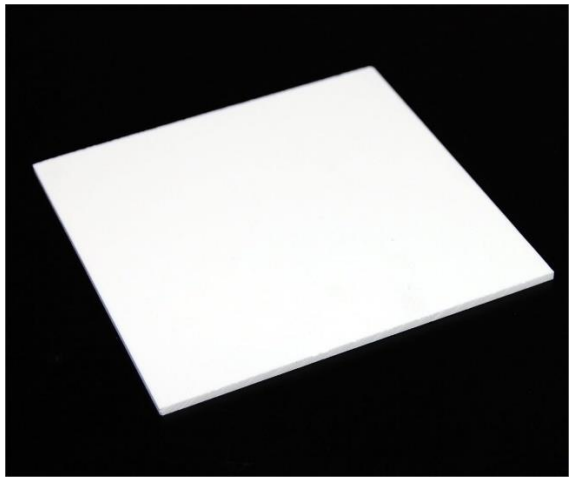
- 18650 Li-on battery(2x)

- 18650 Li-on battery holder



- Male and Female jumper wires

- Acrylic sheet & glue

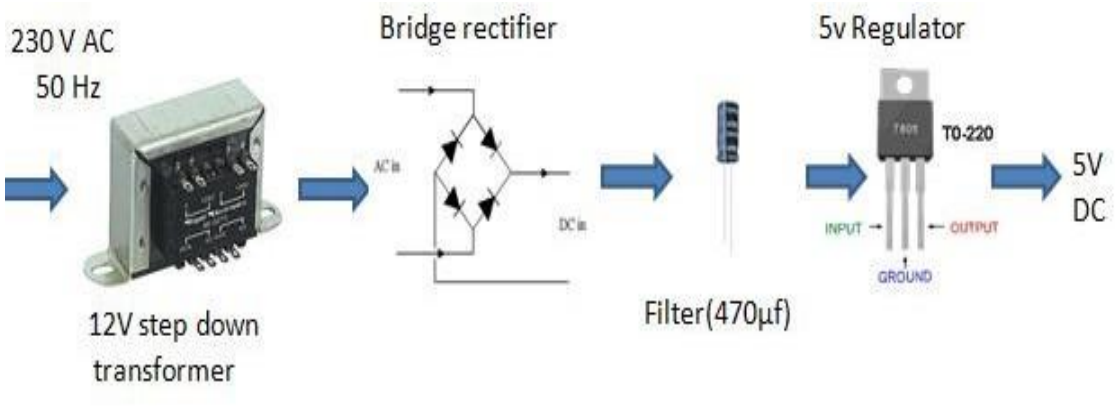


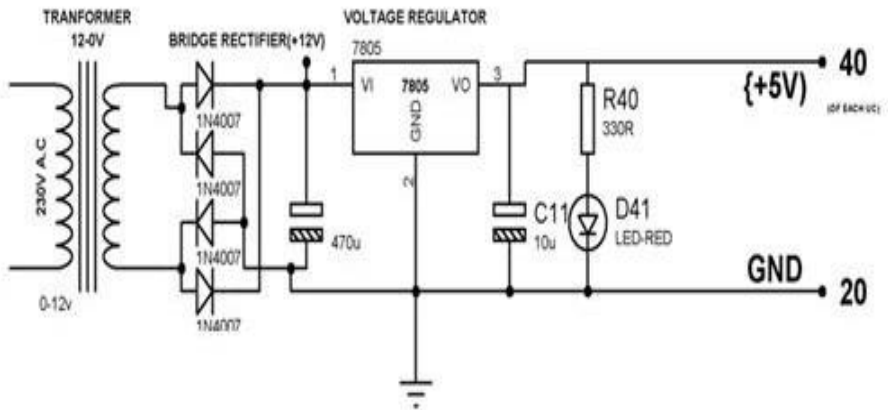
- DC motor

- Converts direct current electrical power into mechanical power
- The very basic construction of a dc motor contains a current carrying armature which is connected to the supply end

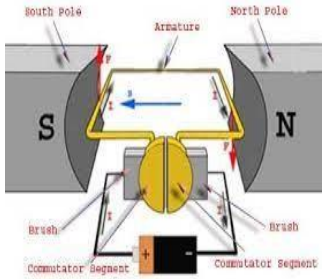
DC Motor construction and power supply

Power supply





DC motor construction



•Software requirements

- Arduino IDE
- Programming languages used

  Embedded (C/C++) on

  Arduino or PYTHON

•Codes (C++)

```
// by SHYAKA Aime
//Arduino Human Following Robot
// You have to Install the AFMotor and NewPing library Before Uploading the sketch//
// To install the libraries ( first download the AF Motor driver, NewPing and Servo
Library zip file //
// then Go to Skecth >> Include Library >> Add .Zip Library >> Select The downloaded zip
file >> Done) //
#include<NewPing.h>
#include<Servo.h>
#include<AFMotor.h>
```

```
#define RIGHT A2
#define LEFT A3
#define TRIGGER_PIN A1
#define ECHO_PIN A0
#define MAX_DISTANCE 100


NewPing sonar(TRIGGER_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;

int pos =0;

void setup() {
  // setup code:
  Serial.begin(9600);
myservo.attach(10);
{
for(pos = 90; pos <= 180; pos += 1){
  myservo.write(pos);
  delay(15);
} for(pos = 180; pos >= 0; pos-= 1) {
  myservo.write(pos);
  delay(15);
}for(pos = 0; pos<=90; pos += 1) {
  myservo.write(pos);
  delay(15);
}
```

```
}
pinMode(RIGHT, INPUT);
pinMode(LEFT, INPUT);
}

void loop() {
  // main code here, to run repeatedly:
  delay(50);
 unsigned int distance = sonar.ping_cm();
Serial.print("distance");
Serial.println(distance);

int Right_Value = digitalRead(RIGHT);
int Left_Value = digitalRead(LEFT);

Serial.print("RIGHT");
Serial.println(Right_Value);
Serial.print("LEFT");
Serial.println(Left_Value);

if((Right_Value==1) && (distance>=10 && distance<=30)&&(Left_Value==1)){
  Motor1.setSpeed(120);
  Motor1.run(FORWARD);
  Motor2.setSpeed(120);
  Motor2.run(FORWARD);
  Motor3.setSpeed(120);
  Motor3.run(FORWARD);
  Motor4.setSpeed(120);
  Motor4.run(FORWARD);
}else if((Right_Value==0) && (Left_Value==1)) {
  Motor1.setSpeed(200);
  Motor1.run(FORWARD);
  Motor2.setSpeed(200);
  Motor2.run(FORWARD);
```

```
  Motor3.setSpeed(100);
  Motor3.run(BACKWARD);
  Motor4.setSpeed(100);
  Motor4.run(BACKWARD);
}else if((Right_Value==1)&&(Left_Value==0)) {
  Motor1.setSpeed(100);
  Motor1.run(BACKWARD);
  Motor2.setSpeed(100);
  Motor2.run(BACKWARD);
  Motor3.setSpeed(200);
  Motor3.run(FORWARD);
  Motor4.setSpeed(200);
  Motor4.run(FORWARD);
}else if((Right_Value==1)&&(Left_Value==1)) {
  Motor1.setSpeed(0);
  Motor1.run(RELEASE);
  Motor2.setSpeed(0);
  Motor2.run(RELEASE);
  Motor3.setSpeed(0);
  Motor3.run(RELEASE);
  Motor4.setSpeed(0);
  Motor4.run(RELEASE);
}else if(distance > 1 && distance < 10) {
  Motor1.setSpeed(0);
  Motor1.run(RELEASE);
  Motor2.setSpeed(0);
  Motor2.run(RELEASE);
  Motor3.setSpeed(0);
  Motor3.run(RELEASE);
  Motor4.setSpeed(0);
  Motor4.run(RELEASE);
   }
 }
```

•Codes(PYTHON)

```
# -*- coding: utf-8 -*-
"""
-------------------------------------------------------------------------------
OpenCV Human face tracker combined with arduino powered bot to
follow humans.
         @authors:
Yash Chandak    Ankit Dhall
TODO:
convert frame specific values to percentages
-------------------------------------------------------------------------------
"""

import numpy as np
import sys
import time

"""
PySerial library required for arduino connection
OpenCV library requierd for face tracking
"""

import serial
import cv2

"""
Arduino connected at port No. COM28,
Confirm and change this value accordingly from control panel
Baud Rate = 9600
```

```
"""

arduino = serial.Serial('COM28', 9600)
time.sleep(2) # waiting the initialization...
print("initialised")

#gets the direction for Arduino serial
def direction(bound, initArea=40000):
    """
    Direction control Index:
    '<' , '>' are the frame check bits for serial communication
    Numbers represent the direction to be moved as per their position on numpad
    1: Back Left
    2: Back
    3: Back right
    4: Left
    5: Stay still
    6: Right
    7: Front Left
    8: Forward
    9: Forward right
    """

    #anchor the centre position of the image
    center=(320, 240)
    #current rectangle center
    curr = (bound[0] + bound[2]/2, bound[1]+bound[3]/2)
    out=0
    flag=0
    fb = 0 #0-stay 1-fwd 2-bwd
    lr = 0 #0-stay 1-left 2-right

    #if the object is coming closer i.e. it's size is increasing then move bwd
    if bound[2]*bound[3] > (initArea+5000) or bound[1]<50 :
```

```
        fb = 2
    #if the object os moving away i.e. it's size is decreasing then move towards it
    elif bound[2]*bound[3] < (initArea-5000) or (bound[1]+bound[3])>430 :
        fb = 1
    else :
        fb = 0

    #move right
    if curr[0] > (center[0] + 100):
        lr = 2
    #move left
    elif curr[0] < (center[0] - 100):
        lr = 1
    else:
        lr = 0

    if lr == 0 and fb == 0:
        out = 5
        print "stay"
    elif lr == 0 and fb == 1:
        out =8
        print "fwd"
    elif lr == 0 and fb == 2:
        out = 2
        print "back"
    elif lr == 1 and fb == 0:
        out = 4
        print "left"
    elif lr == 1 and fb == 1:
        out = 7
        print "fwd left"
    elif lr == 1 and fb == 2:
        out = 1
        print "left back"
```

```
    elif lr == 2 and fb == 0:
        out = 6
        print "right"
    elif lr == 2 and fb == 1:
        out = 9
        print "fwd right"
    elif lr == 2 and fb == 2:
        out = 3
        print "bwd right"
    else :
        out = 5
        print "Stay Still"

    #Write the encoded direction value on the serial communication line
    print out
    arduino.write('<')
    arduino.write(str(out))
    arduino.write('>')

def detectAndDisplay(frame):
    #use OpenCV HAAR face detetcion algorithm to detect faces
    faces = cascade.detectMultiScale(frame, scaleFactor=1.1, minNeighbors=3,
                                     minSize=(30, 30),maxSize=(500,500),
                                     flags=cv2.cv.CV_HAAR_SCALE_IMAGE)

    #if any face is detected then process else continue searching
    if(len(faces)!=0):
        #If number of faces in the image is more than 1
        #Then choose the one with maximum size
        max_area=-1
        i=0
        for (x,y,w,h) in faces:
            if w*h > max_area:
                max_area=w*h
```

```
                pos=i
            i=i+1

        RECT=faces[pos]
        #Mark the face being tracked on the image display
        cv2.rectangle(frame, (RECT[0], RECT[1]), (RECT[0]+RECT[2], RECT[1]+RECT[3]), (0,
255, 0), 2)
        #draw_str(frame, (RECT[0], RECT[3]+16), 'x: %.2f y: %.2f size: %.2f' % (RECT[2]-
RECT[0])/2 % (RECT[3]-RECT[1])/2 % RECT[2]*RECT[3])

        #Put the text details about the ROI on imdisplay
        cv2.putText(frame, `RECT[0] + RECT[2]/2`+'  '+`RECT[1]+RECT[3]/2`+'
'+`RECT[2]*RECT[3]`, (RECT[0],RECT[1]+RECT[3]),  cv2.FONT_HERSHEY_SIMPLEX , 1,
(0,0,255));

        #compute direction for the arduino bot to be moved.
        direction(RECT)

    else:
        print 'Search...'
        arduino.write('<')
        arduino.write(str(5))
        arduino.write('>')

    cv2.imshow('frame',frame)


cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
#cascade = cv2.CascadeClassifier('haarcascade_frontalface_alt.xml')


cap = cv2.VideoCapture(1)
cap.grab()
ret, frame = cap.retrieve()
```

```
cv2.namedWindow('frame')

#Run the tracker in infinite loop
while(1):
    #grab the frames from web camera
    ret, frame = cap.retrieve()
    if ret ==0:
        print "frame not loaded"
    if ret==True:

        #Resize the frame for faster computation
        #cv2.resize(frame,(240,320))

        #Process the frame and pass data to arduino
        detectAndDisplay(frame)

        #cv2.imshow('input',frame)

        #press ESC to exit program
        ch = cv2.waitKey(1)
        if ch==27:
            break

#Free up memory on exit
cap.release()
cv2.destroyAllWindows()
arduino.close()
```

Note: Even we have provided source codes of C++ and PYTHON in this work we will only use C++ , because our microcontroller is ARDUINO-UNO

•Procedures

After looking on hardware requirements, software requirements and source codes of Robot, now we are going to look on procedures to make it.

1.Gather components like Arduino, Dc geared motors with wheels, 2 IR sensors, ultrasonic sensor, servo motor, cardboard, l293d motor driver shield, battery 9 v, switch, Jumper wires etc.

2. Take cardboard and place all 4 motors with wheels on it to make a car.

3. Place IR sensors on the car.

4. Stick servo motor on the car.

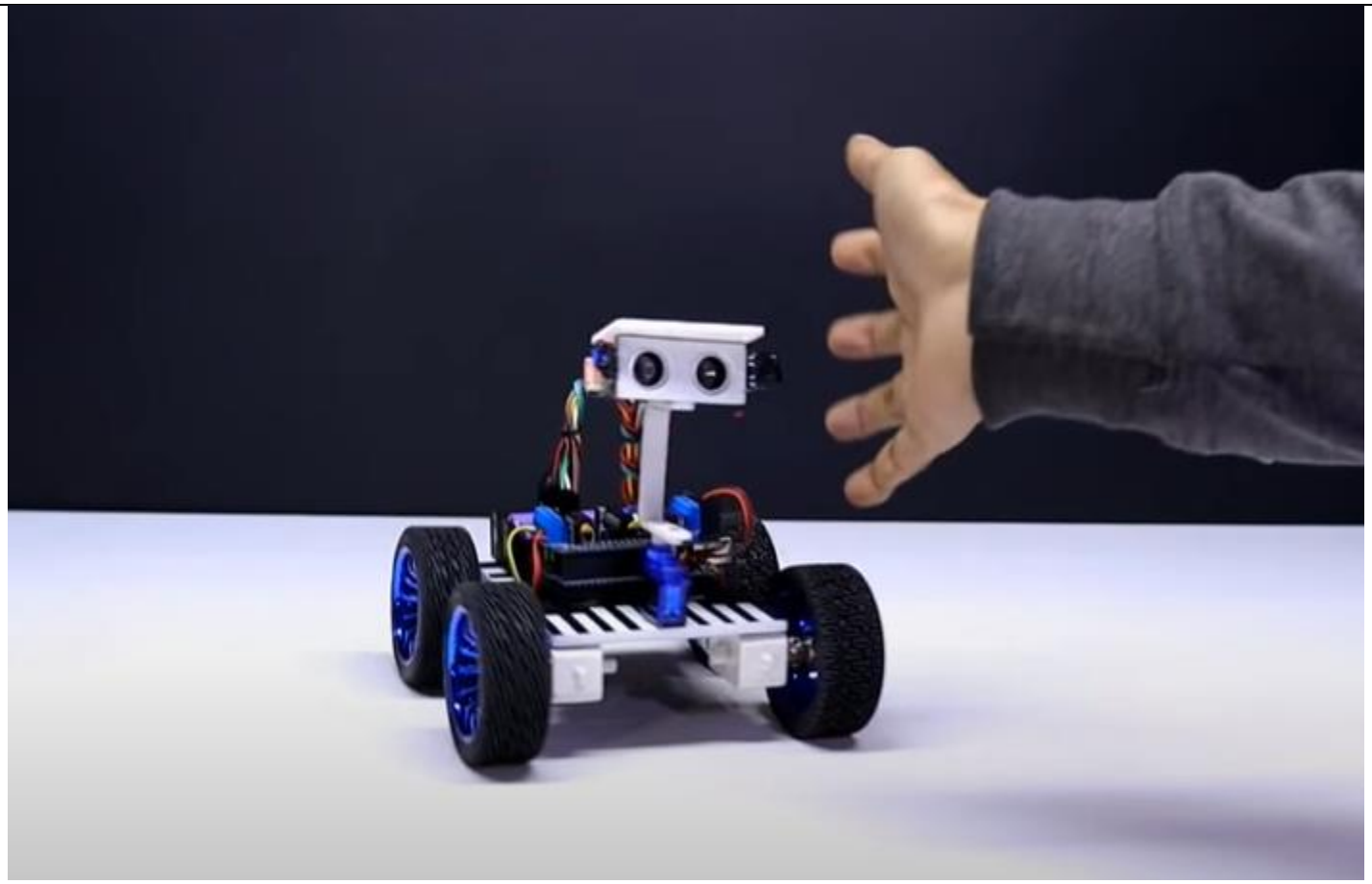
5. Place ultrasonic sensor in box and add handle to it after that connect it on servo motor.

6. Connect Arduino and l293d motor shield on car.

7. Do connections as shown in circuit diagram.

8. Upload the code and connect switch with battery to l293d motor driver shield.

Adjust the sensitivity of IR sensors.



Final result

Advantages of Robot

- **This robot can be used in industries or searching purpose**

•References

- https://www.arduino.cc/
- https://MITadmissions.org/blogs/entry/do-you-want-to-build-a-robot/
- **Video tutorial:** https://www.youtube.com/watch?v=yAV5aZ0unag
- **Hardware tools:** https://www.amazon.com/
- **Source codes used in this work:**

  https://github.com/SHYAKA-Aime/arduino-foll-humman-robot

Organized by SHYAKA Aime

https://aime-blog.netlify.app/