DRIVERLESS AUTONOMOUS CAR

MANTECH 6RM3: ROBOT MECHANICS AND MECHATRONICS

Somak Mukherjee (Student No – 400232759)
Email: mukhes8@mcmaster.ca
Professor: Dr. Timber Yuen

Abstract**—**This study implements an Arduino obstacle avoidance car including Bluetooth and Wi-fi control and Robotic gripper. The car can be used as Image & Video processing with Neural Network and Deep Learning concept with Python + OpenCV. The L298N is used as the driving circuit, the PWM output from the Arduino is used for speed control, and the infrared sensor is used to emit infrared rays at a certain frequency to detect obstacles, and all the detection signals are sent to the Arduino. The Arduino acts as a control core to change the speed and steering of the car to achieve automatic obstacle avoidance. The Raspberry Pi 3 and Pi cam is used as operating system for Image and Video Processing.

CONTENTS

1. INTRODUCTION / OBJECTIVE

Whether we are already living in the “Automation” is a matter of debate, but it won’t be for much longer. The current wave of technologies is transforming the business landscape, with Artificial Intelligence, Machine Learning, Neural Network, Deep Learning, Image processing and Robotics advancements allowing companies to automate an increasing number of routine tasks once performed by humans.

In scientific exploration and emergency rescue, it is often necessary to detect some dangers or areas that humans cannot directly reach, which needs to be done by robots. Automated obstacle avoidance when robots travel in complex terrain is an essential and essential function. Therefore, the development of automatic obstacle avoidance systems has emerged. The automatic obstacle avoidance car was developed based on this system. It can be used as a regional exploration robot and an emergency rescue robot’s motion system, allowing the robot to automatically avoid obstacles while traveling.

This study mainly realizes the intelligent mode of the Arduino obstacle avoidance car. The designed theoretical scheme, analysis method, characteristics and innovations can have certain reference significance for the design of automatic semi-automatic robots such as automatic transport robots, mining exploration robots and household automatic cleaning robots. At the same time, this automatic drawing car can be used as a development object of household cleaning robots and realizes economic benefits and forms commercial value when smart homes gradually become a trend.

The most attractive prospect of this automatic drawing path car is the environment that can be used to explore dangerous areas. In the event of fire or poor geological environment, such an automatic drawing path car design can reflect its role. It was able to map out a map of the time and find obstacles for rescuers to search. At this time, the automatic drawing path car can come in handy. By using a camera on it, we can do a lot of work from our own laptop and machine that can’t be done instead of people.

2. APPROACH & METHODS

At present, the cars on the market can be distinguished according to their functions as follows:

[1] Intelligent obstacle avoidance car: When obstacles are encountered, obstacles can be automatically avoided.
[2] Wi-Fi control car & Bluetooth Control Car: The car can control via wi-fi and Bluetooth module.
[3] Robotic Gripper: Through the Robotic gripper Robotic car can be useful in industrial field.
[4] Image & Video Processing: With implementation of Neural Network & Deep Learning concept

This study mainly realized an Arduino intelligent obstacle avoidance car and provided the function of automatically drawing maps. The design method is explained below.

A. Design of The Car

Fig. 1 shows the major core hardware components for the Robotics car.

Table 1 shows the parts list of the Arduino obstacle avoidance car. We prepared some hardware according to the needs of the car design, and formed a basic obstacle avoidance car, which used the Arduino UNO as the main control board and used the infrared sensor as the obstacle avoidance detection.

The Arduino control board used has the following features:

(1) Using a low-cost microprocessor controller (AVR series controller), it can be powered by USB interface, no external power supply, or external 9V DC input.

(2) It can be easily connected to sensors, a wide range of electronic components (e.g. infrared, ultrasonic, etc.).

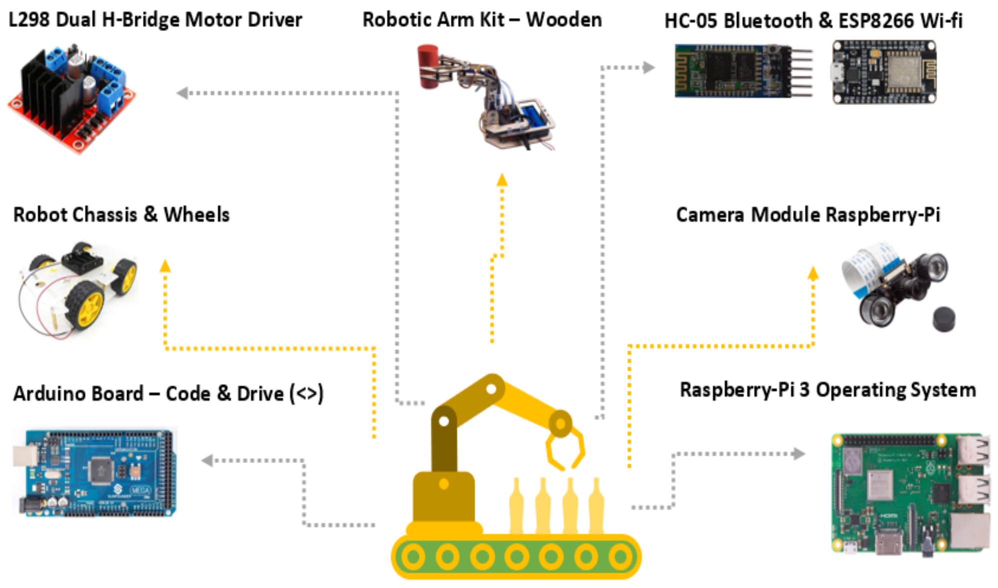
L298 Dual H-Bridge Motor Driver
Robotic Arm Kit – Wooden
HC-05 Bluetooth & ESP8266 Wi-fi
Robot Chassis & Wheels
Camera Module Raspberry-Pi
Arduino Board – Code & Drive (<>)
Raspberry-Pi 3 Operating System

Fig 1. *Core Hardware Components*

(3) Arduino allows us to quickly integrate with Unity to make interactive works. Arduino can use existing electronic components such as switches or sensors or other controller components.

(4) USB interface, no external power supply required. There is also a 9V DC power input.

(7) In terms of applications, you can combine other development software to get out more interesting things.

TABLE 1

Item Name	Quantity
ESP8266 Development	1
Board - Wi-Fi Module	1
Li-ion Battery 1400mAh 3.7V	Several
120pcs Multicolored Jumper Wires Ribbon	Several
Arduino UNO R3 Board	2
Bracket for Ultrasonic Ranging Module	1
Robot Car Chassis	2
Micro Servo Motor with PCA9685 16 Channel 12 Bit PWM	2
L298N Dual H-Bridge DC Stepper Motor Drive Controller	2
Robotic Arm Kit DIY 4-Axis	1
HC-05 Bluetooth Serial Pass	1
Raspberry Pi 3 B+	1
Camera Module for Raspberry Pi	1
Gear motor	8
Screws & other parts (Such as glue, tape etc.)	Several

B. Circuit Diagram

One of the important steps of design the robotic car is circuit design. Fig. 2 and Fig. 3 shows the circuit diagram for Wi-Fi & Bluetooth Control Car and Obstacle Avoidance Car respectively.

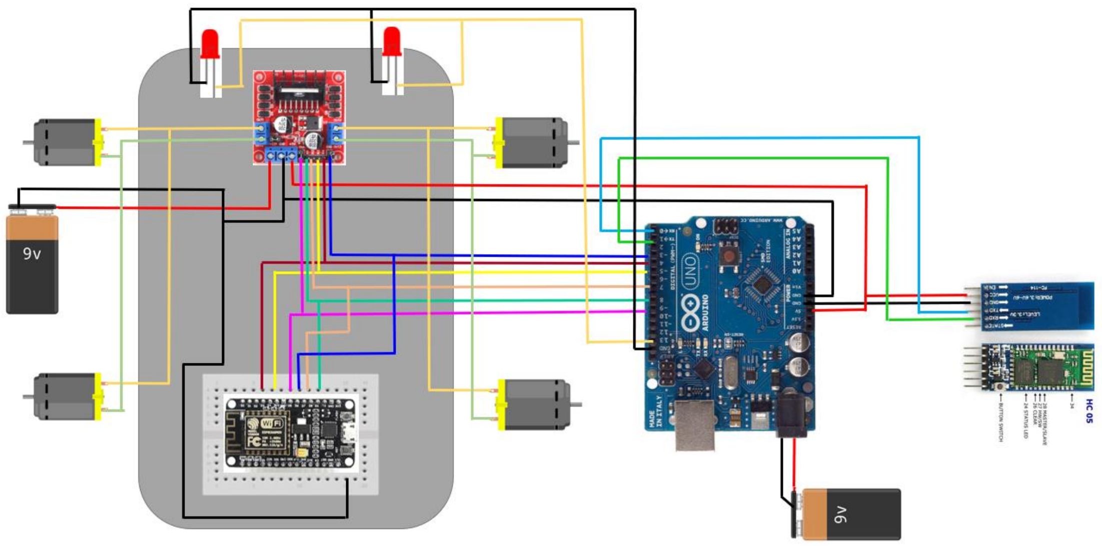

***Fig 2**. Circuit Diagram for Wi-Fi & Bluetooth Control Car*

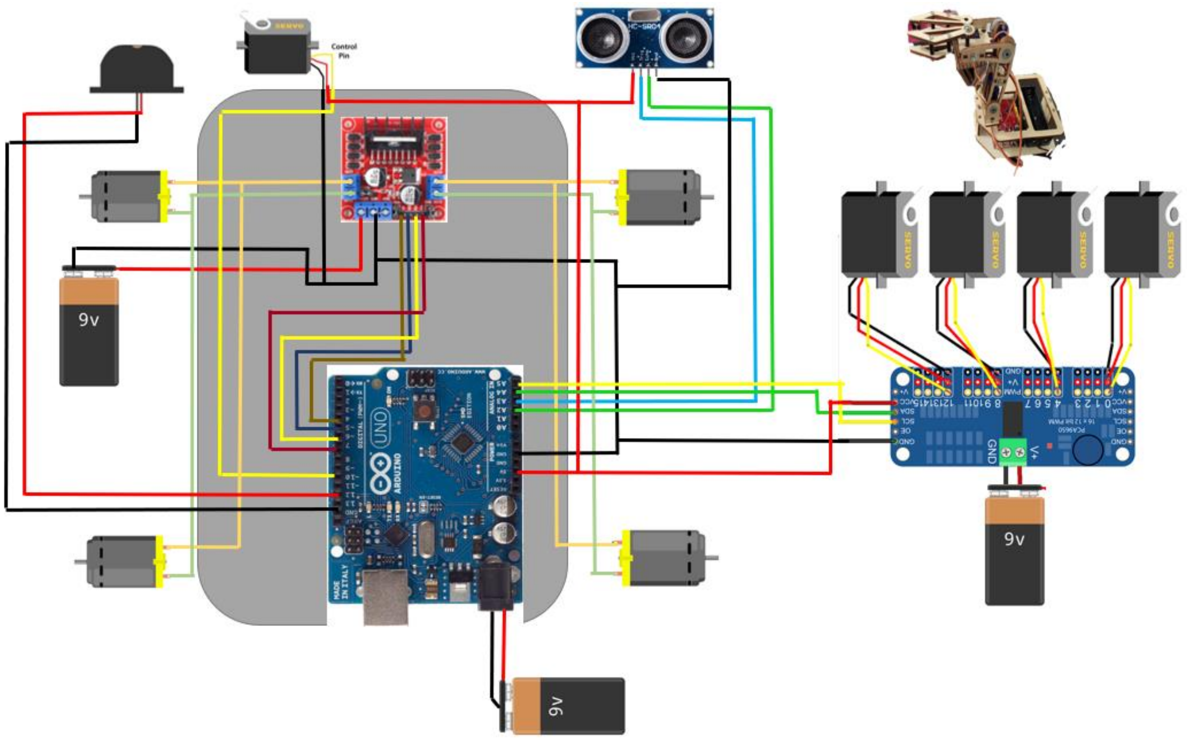

***Fig 3**. Circuit Diagram for Obstacle Avoidance & Robotic Gripper*

C. *Implementation*

The implementation of obstacle avoidance strategy for robot involves the writing and compilation of program using Arduino software. Arduino is a popular programmable board used to create projects.

1. *Intelligent obstacle avoidance car*

Open source code circuit design, program development interface free download, can also be modified according to your needs. Through the infrared obstacle avoidance sensor, it can realize automatic following or obstacle avoidance function. It can be used as a regional exploration robot and an emergency rescue robot's motion system, allowing the robot to automatically avoid obstacles while traveling.

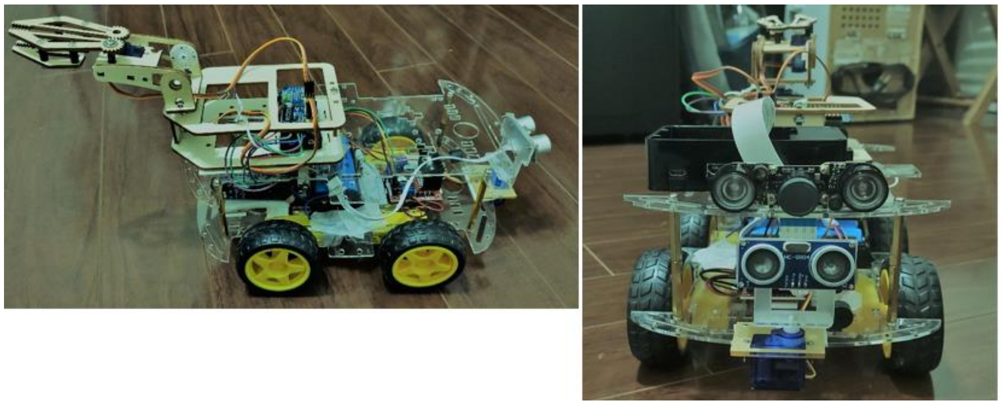

Fig. 4 *Side view & Top view of Obstacle Avoidance Car & Robot Gripper*

2. *Wi-Fi control car & Bluetooth Control Car*

The wireless car is controlled via smartphone which is connected to the Bluetooth module and a Wi-Fi module. Now a days due to advancement in technology various newly designed smart homes make use of Wi-Fi enabled robot for various applications. Robot car controlled by Wi-Fi will make our work much easier as we can make the robot do any work, we need by just a single movement on our mobile phone on computer.

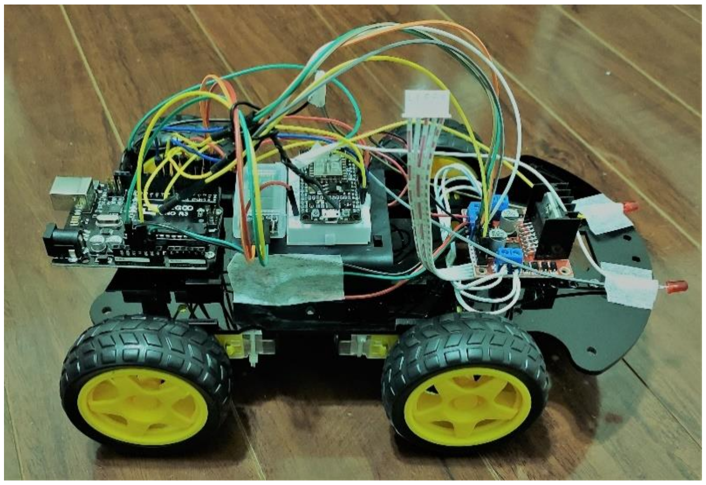

Fig. 5 *Side view of Wi-Fi & Bluetooth Control Car*

3. *Robotic Gripper*

A gripper is a device which enables an object to be manipulated. Just like a hand, a gripper enables holding, tightening, handling and releasing of an object. Industrial robots for tasks normally carried out by human hands has led to the need for more effective handling equipment, especially pretension tools. Grippers are mainly applicable for pick and place a work piece or a component. Load carrying capacity will differs depends upon the nature of the application of the robot.

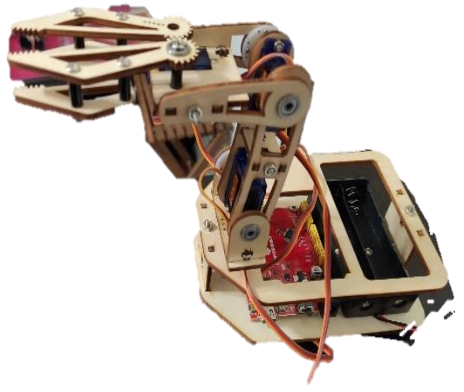

Fig. 6 *Side view of Robotic Gripper*

4. *Image & Video Processing*

The Raspberry Pi is a very cheap computer that runs Linux, but it also provides a set of GPIO (general purpose input/output) pins that allow you to control electronic components. It comes with Pi-Camera module with python interpreter. Pi camera can capture directly to any object which supports Python's buffer protocol (including NumPy). OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. It was developed so that real-time analytics of images and recognition can be done for assorted applications.

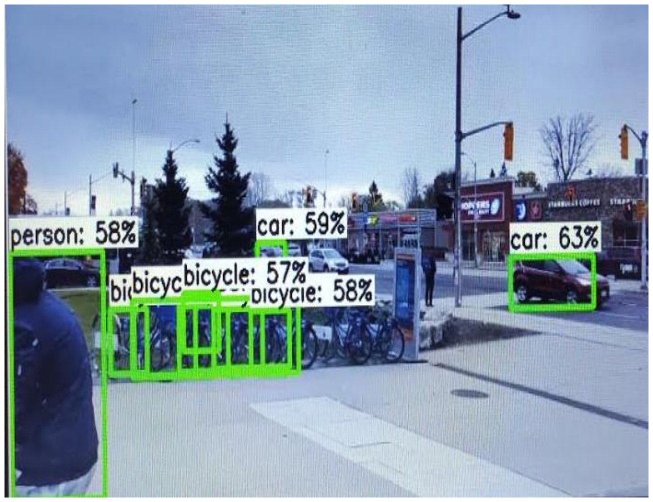

Fig. 7 *Image & Video Processing*

3. ANALYSIS & TESTING

A. Analysis

The hardware implementation phase is divided into six components. The components are:

1. Arduino UNO R3

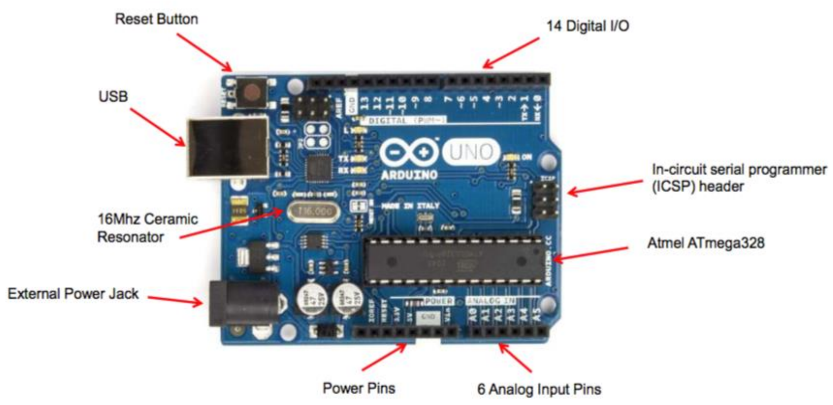

Fig. 8 *Arduino UNO R3 board*

Arduino Uno Overview – Digital Input / Output

- SPI PWM = TX / RX
- 14 Digital I/O
- 5V logic level
- Sink/source 20mA per pin
- Universal Asynchronous Serial Receiver /
- Transmitter (UART)
    - Pin 0 – Receive (RX)
    - Pin 1 – Transmit (TX)
    - MCU INT pins brought out to pins 2, 3
- 8-bit PWM
    - Pins 3, 5, 6, 9, 10, 11
- Serial Peripheral Interface (SPI)
    - Pins 10, 11, 12, 13

Arduino Uno Overview – Analog

- Analog Inputs (A0 to A5)
- Sampled input:
    - 0 to 5V
    - 10 bits resolution
- AREF pin can be used to adjust
- upper limit of input range
- Two-wire Interface COM (TWI)
- Pin A4 – SDA pin (data)
- Pin A5 – SCL pin (clock)

2. HC-SR04 Ultrasonic Sensor

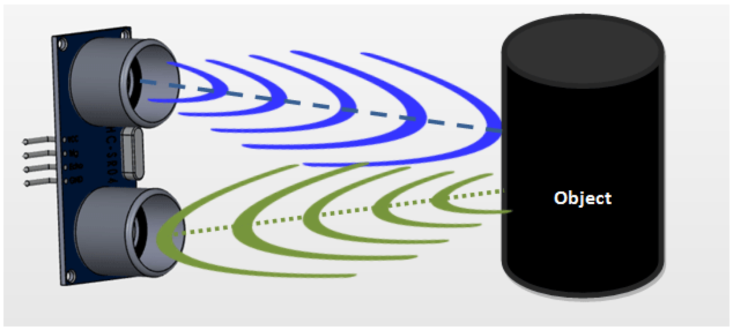

Fig. 9 *Diagram of the basic ultrasonic sensor operation*

An Ultrasonic sensor is a device that can measure the distance to an object by using sound waves. It measures distance by sending out a sound wave at a specific frequency and listening for that sound wave to bounce back. By recording the elapsed time between the sound wave being generated and the sound wave bouncing back, it is possible to calculate the distance between the sonar sensor and the object.

$$distance = \frac{speed\ of\ sound\ \times time\ taken}{2}$$

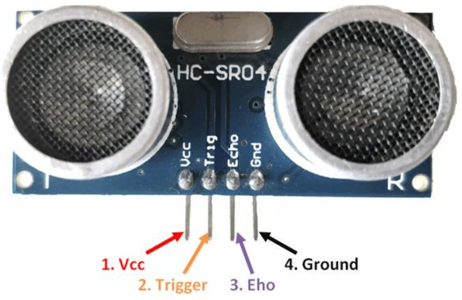

Fig. 10 *Ultrasonic Sensor Pins*

3. L298D Motor Driver Shield – Features

- Up to 4 bi-directional DC motors with individual 8-bit speed selection (so, about 0.5% resolution)
- Up to 2 stepper motors (unipolar or bipolar) with single coil, double coil, interleaved or micro-stepping.
- 4 H-Bridges: L293D chipset provides 0.6A per bridge (1.2A peak) with thermal shutdown protection, 4.5V to12V
- Big terminal block connectors to easily hook up wires (10-22AWG) and power Arduino reset button brought up top

4. SG-90 Servo Motor

A servo motor is an electrical device which can push or rotate an object with great precision. If we want to rotate and object at some specific angles or distance, then we use servo motor. It is just made up of simple motor which run through servo mechanism. If motor is used is DC powered then it is called DC servo motor, and if it is AC powered motor then it is called AC servo motor. We can get a very high torque servo motor in a small and light weight packages. Doe to these features they are being used in many applications like toy car, RC helicopters and planes, Robotics, Machine etc. The position of a servo motor is decided by electrical pulse and its circuitry is placed beside the motor.

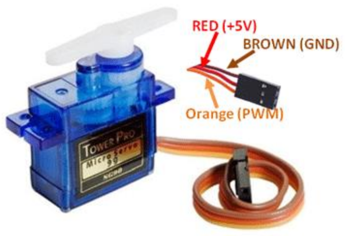

Fig. 11 *SG-90 Servo Motor*

Servo Mechanism
It consists of three parts:
- Controlled device
- Output sensor
- Feedback system

It is a closed loop system where it uses positive feedback system to control motion and final position of the shaft. Here the device is controlled by a feedback signal generated by comparing output signal and reference input signal.

5. DC motor

A Direct Current (DC) motor is a rotating electrical device that converts direct current, of electrical energy, into mechanical energy. An Inductor (coil) inside the DC motor produces a magnetic field that creates rotary motion as DC voltage is applied to its terminal. Inside the motor is an iron shaft, wrapped in a coil of wire. This shaft contains two fixed, North and South, magnets on both sides which causes both a repulsive and attractive force, in turn producing torque.

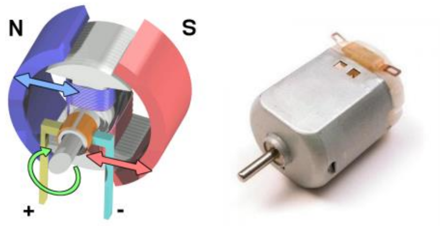

Fig. 12 *SG-90 Servo Motor*

These types of motors are powered by direct current (DC).
- Brushed Motors
- Brushless Motors
- Stepper Motors
- Servo Motors

6. **Raspberry Pi**
The Raspberry Pi is a very cheap computer that runs Linux, but it also provides a set of GPIO (general purpose input/output) pins that allow you to control electronic components. It comes with Pi-Camera module with python interpreter. Pi camera can capture directly to any object which supports Python's buffer protocol (including NumPy). There are several developers and applications that are leveraging the Raspberry Pi for automation.

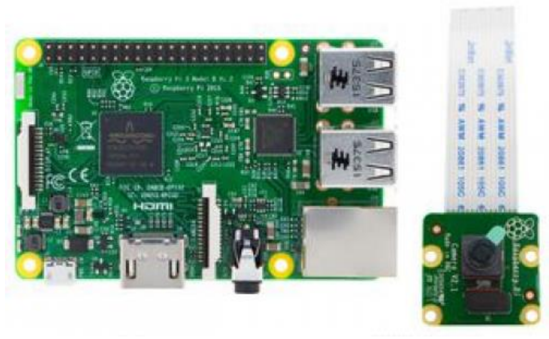

Fig. 13 *Raspberry Pi with Pi Cam*

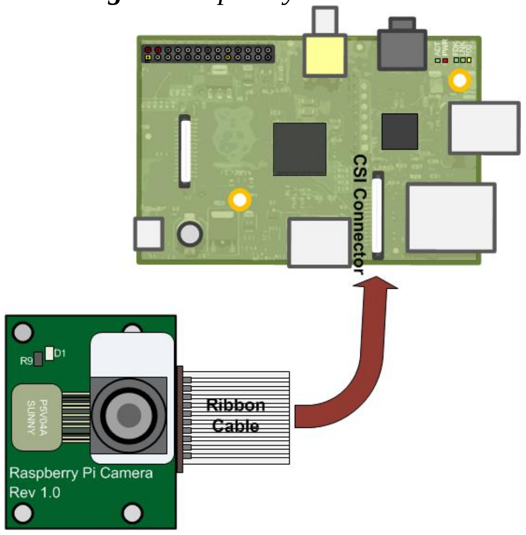

Fig. 14 *Raspberry Pi with Pi Cam Connection*

B. Testing

Arduino IDE writes C/C++ program to complete the movement and control of the Arduino obstacle avoidance car, robot gripper and wi-fi & Bluetooth control car. The sonar system is used in HC-SR04 ultrasonic sensor to determine distance to an object like bats do. It offers excellent non-contact range detection from about 2 cm to 400 cm or 1'' to 13 feet. Its operation is not affected by sunlight or black material. The ultrasonic sensor emits the short and high frequency signal. If they detect any object, then they reflect back echo signal which taken as input to the sensor through Echo pin.

Firstly, we initialize Trigger and Echo pin as low and push the robot in forward direction. when obstacle is detected Echo pin will give input as high to micro-controller. pulseIn() function is used for calculating the time of distance from the obstacle. Every time the function waits for pin to go high and starts timing, then timing will be stopped when pin go to low. It returns the pulse length in microseconds or when complete pulse was not received within the timeout it returns 0. The timing has been determined means it gives length of the pulse and will show errors in shorter pulses. Pulses from 10microseconds to 3 minutes in length are taken into consideration.

After determining the time, it converts into a distance. If the distance of object is moderate then speed of robot gets reduced and will take left turn, if obstacle is present in left side then it will take right turn. If the distance of object is short, then speed of robot gets reduced and will turn in backward direction and then can go in left or right direction.

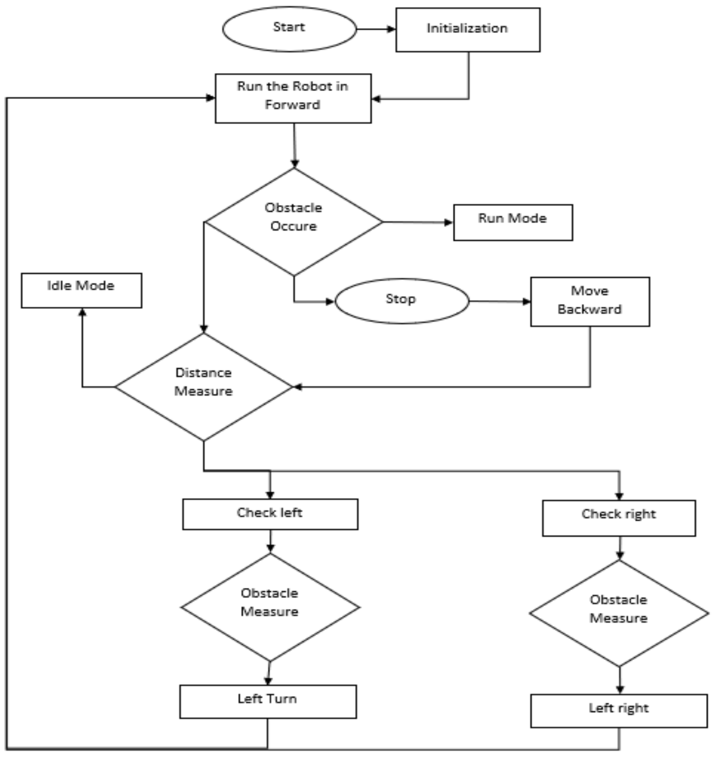

Fig. 15 *Flow chart of Obstacle Avoidance car Programming*

Following is the Obstacle Avoidance code:

```
#include <Servo.h>        // Include Servo Library
#include <NewPing.h>      // Include Newping Library
#include <Wire.h>
#include <Adafruit_PWMServoDriver.h>
#define MIN_PULSE_WIDTH       125
#define MAX_PULSE_WIDTH       575
#define FREQUENCY             50
Adafruit_PWMServoDriver pwm = Adafruit_PWMServoDriver();
// L298N Control Pins
const int LeftMotorForward = 4;
const int LeftMotorBackward = 5;
const int RightMotorForward = 6;
const int RightMotorBackward = 7;
#define TRIGGER_PIN  A1  // Arduino pin tied to trigger pin on the ultrasonic sensor.
#define ECHO_PIN     A2  // Arduino pin tied to echo pin on the ultrasonic sensor.
#define MAX_DISTANCE 250 // Maximum distance we want to ping for (in centimeters).
Maximum sensor distance is rated at 250cm.
Servo servo_motor;  // Servo's name
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE); // NewPing setup of pins and
maximum distance.
```

```
boolean goesForward = false;
int distance = 100;
// our servo # counter
uint8_t servonum = 0;

int buzzer_pin = 12;
void setup()
{
  // Set L298N Control Pins as Output
  pinMode(RightMotorForward, OUTPUT);
  pinMode(LeftMotorForward, OUTPUT);
  pinMode(LeftMotorBackward, OUTPUT);
  pinMode(RightMotorBackward, OUTPUT);
  servo_motor.attach(10);   // Attachs the servo on pin 9 to servo object.
  servo_motor.write(115);   // Set at 115 degrees.
  delay(2000);              // Wait for 2s.
  distance = readPing();    // Get Ping Distance.
  delay(100);               // Wait for 100ms.
  distance = readPing();
  delay(100);
  distance = readPing();
  delay(100);
  distance = readPing();
  delay(100);
  pinMode (buzzer_pin, OUTPUT);
  Serial.begin(9600);
  pwm.begin();
  pwm.setPWMFreq(60);
}
void loop()
{
  int distanceRight = 0;
  int distanceLeft = 0;
  delay(50);
  if (distance <= 25)
  {
    moveStop();
    delay(300);
    digitalWrite (buzzer_pin, HIGH);
    moveBackward();
    delay(400);
    digitalWrite (buzzer_pin, LOW);
    moveStop();
    delay(300);
    MoveServoAB();
    distanceRight = lookRight();
    delay(300);
    distanceLeft = lookLeft();
    delay(300);
    if (distanceRight >= distanceLeft)
    {
      turnRight();
      delay(300);
      moveStop();
    }
    else
    {
      turnLeft();
      delay(300);
      moveStop();
    }
  }
  else
  {
    moveForward();
  }
    distance = readPing();
}
int lookRight()     // Look Right Function for Servo Motor
{
  servo_motor.write(50);
  delay(500);
  int distance = readPing();
  delay(100);
  servo_motor.write(115);
  return distance;
}
int lookLeft()      // Look Left Function for Servo Motor
{
  servo_motor.write(180);
  delay(500);
  int distance = readPing();
  delay(100);
  servo_motor.write(115);
  return distance;
}
int readPing()      // Read Ping Function for Ultrasonic Sensor.
{
  delay(100);
  int cm = sonar.ping_cm();
  if (cm==0)
  {
    cm=250;
  }
  return cm;
}
void moveStop()       // Move Stop Function for Motor Driver.
{
  digitalWrite(RightMotorForward, LOW);
  digitalWrite(RightMotorBackward, LOW);
  digitalWrite(LeftMotorForward, LOW);
  digitalWrite(LeftMotorBackward, LOW);
}
void moveForward()    // Move Forward Function for Motor Driver.
{
    digitalWrite(RightMotorForward, HIGH);
    digitalWrite(RightMotorBackward, LOW);
    digitalWrite(LeftMotorForward, HIGH);
    digitalWrite(LeftMotorBackward, LOW);
}
void moveBackward()   // Move Backward Function for Motor Driver.
{
  digitalWrite(RightMotorForward, LOW);
  digitalWrite(RightMotorBackward, HIGH);
  digitalWrite(LeftMotorForward, LOW);
  digitalWrite(LeftMotorBackward, HIGH);
}
void turnRight()      // Turn Right Function for Motor Driver.
{
  digitalWrite(RightMotorForward, LOW);
  digitalWrite(RightMotorBackward, HIGH);
  digitalWrite(LeftMotorForward, HIGH);
  digitalWrite(LeftMotorBackward, LOW);
}
void turnLeft()       // Turn Left Function for Motor Driver.
{
  digitalWrite(RightMotorForward, HIGH);
  digitalWrite(RightMotorBackward, LOW);
  digitalWrite(LeftMotorForward, LOW);
  digitalWrite(LeftMotorBackward, HIGH);
}
void MoveServoAB() {
    pwm.setPWM(0, 0, 150 );
  delay(500);
    pwm.setPWM(0, 0, 370 );
  delay(500);
    pwm.setPWM(3, 0, 400 );
  delay(500);
    pwm.setPWM(3, 0, 575 );
  delay(500);
    pwm.setPWM(0, 0, 575 );
  delay(500);
    pwm.setPWM(0, 0, 370 );
  delay(500);
}
```

Python is an interpreted, high-level, general-purpose programming language. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Reading and writing images are essential to any computer vision project. Machines see and process everything using numbers, including images and text

A detection model has two files associated with it: a detect.tflite file (which is the model itself) and a labelmap.txt file (which provides a label map for the model). Google provides a sample quantized SSDLite-MobileNet-v2 object detection model which is trained off the MSCOCO dataset and converted to run on TensorFlow Lite. It can detect and identify 80 different common objects, such as people, cars, cups, etc. You can also use a custom object detection model by yourself. The TensorFlow Object Detection API requires several additional Python packages, specific additions to the PATH and PYTHONPATH variables, and a few extra setup commands to get everything set up to run or train an object detection model. YOLO apply a single neural network to the full image. This network divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities.

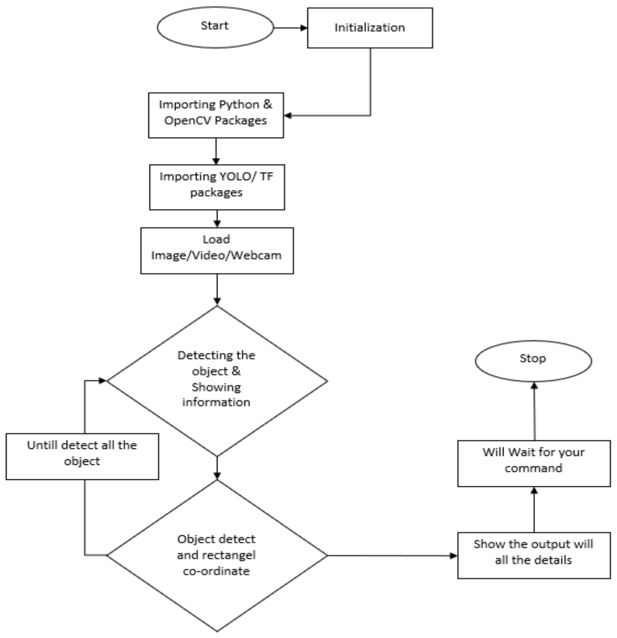
Start
Initialization
Importing Python & OpenCV Packages
Importing YOLO/ TF packages
Load Image/Video/Webcam
Detecting the object & Showing information
Untill detect all the object
Object detect and rectangel co-ordinate
Show the output will all the details
Will Wait for your command
Stop

Fig. 16 *Flow chart of Object detection Programming*

Following is the Object Detection Code (Python + OpenCV):

```python
import cv2
import numpy as np
# Load Yolo
net = cv2.dnn.readNet("yolov3.weights", "yolov3.cfg")
classes = []
with open("coco.names", "r") as f:
    classes = [line.strip() for line in f.readlines()]
layer_names = net.getLayerNames()
output_layers = [layer_names[i[0] - 1] for i in net.getUnconnectedOutLayers()]
colors = np.random.uniform(0, 255, size=(len(classes), 3))
# Loading image
img = cv2.imread("IMG7.jpg")
img = cv2.resize(img, None, fx=0.4, fy=0.4)
print(img.shape)
height, width, channels = img.shape
# Detecting objects
blob = cv2.dnn.blobFromImage(img, 0.00392, (416, 416), (0, 0, 0), True, crop=False)
net.setInput(blob)
outs = net.forward(output_layers)
# Showing informations on the screen
class_ids = []
confidences = []
boxes = []
for out in outs:
    for detection in out:
        scores = detection[5:]
        class_id = np.argmax(scores)
        confidence = scores[class_id]
        if confidence > 0.5:
            # Object detected
            center_x = int(detection[0] * width)
            center_y = int(detection[1] * height)
            w = int(detection[2] * width)
            h = int(detection[3] * height)
            # Rectangle coordinates
            x = int(center_x - w / 2)
            y = int(center_y - h / 2)
            boxes.append([x, y, w, h])
            confidences.append(float(confidence))
            class_ids.append(class_id)
indexes = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.4)
print(indexes)
font = cv2.FONT_HERSHEY_PLAIN
for i in range(len(boxes)):
    if i in indexes:
        x, y, w, h = boxes[i]
        label = str(classes[class_ids[i]])
        color = colors[i]
        cv2.rectangle(img, (x, y), (x + w, y + h), color, 2)
        cv2.putText(img, label, (x, y + 30), font, 3, color, 3)
cv2.imshow("Image", img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

4. RESULT & CONCLUSION

A. RESULT

This study uses Arduino IDE 1.8.0 and writes C/C++ program to complete the movement and control of the Arduino obstacle avoidance car. In order to detect obstacles, we have sensors on the front of the car body to sense the front obstacles. The Arduino obstacle avoidance Wi-Fi module, Bluetooth module and the L298N are also placed in the front part of the car, with the Arduino microcontroller in the middle, the battery installed at the rear of the car, and the DC geared motor placed under the car body on both sides. And in one of the DC geared motors is equipped with a distance measuring module, the overall realization of the Arduino obstacle avoidance car shown in Fig. 4.

This Bluetooth module interface has the terminal: where 1 is for turn the light on, 0 is to turn the light off, 2, 3, 4, 5, 6 is for moving forward, moving backward, moving right, moving left and stop the car respectively. Also, the Wi-fi module interface has five buttons for moving the car. After the user presses the button to connect the car, the car will be connected to the WI-FI to follow the user command.

Deep learning is an artificial intelligence function that imitates the workings of the human brain in processing data and creating patterns for use in decision making. Darknet is an open source neural network framework written in C and CUDA. It is fast, easy to install, and supports CPU and GPU computation. YOLOv3 is extremely fast and accurate. Moreover, you can easily tradeoff between speed and accuracy simply by changing the size of the model, no retraining required. We used to YOLOv3 model to detect the object, video and webcam processing, which was quite successful, but we need to better continuance machine to get the perfect result.

B. DIFFICULTIES

Although the concept and design of the project seemed perfect, there were some problems faced while actual implementation:

1. While working on obstacle avoidance car we faced the problem of ultrasonic sensor due to wrong connection. Earlier it was not able to detect the object and failure to do that the car was crashing.
2. Working with Robotic gripper, we faced issues due to angel mismatch. This happened because initially we set one angle to all the servo and due to this robot gripper was unable to move to exact position as we wanted and that damaged one party of the gripper body. So, we had to fix the angel one by one setting up of the servo angle.
3. During working on wi-fi and Bluetooth we have faced issued like not able to connect to Bluetooth, not able to connect to wi-fi.
4. On object detection, we had to find the large dataset to detect the object. Also due to low configuration of laptop our detection model was not able to perform initially. So, we had to work will another open source model which is easier to install in laptop

C. RECOMENDATION

1. If the current project is interfaced with a camera (e.g. a Webcam) robot can be driven beyond line-of-sight & range becomes practically unlimited as networks have a very large range.
2. Use as a fire fighting robot: By adding temperature sensor, water tank and making some changes in programming we can use this robot as firefighting robot.
3. We can extend this project with wireless technology by IR (or) RF (or) ZIGBEE.
4. We can use the DTMF receiver by using the mobile phone.

D. CONCLUSION

The intelligent car is a miniaturized robot that can perform specific tasks through intelligent means. It has the advantages of low production cost, simple circuit structure and convenient program debugging. Thanks to its strong practicality, the intelligent car is popular among robot lovers and factories. It can be applied to home cleaning robots and exploration of dangerous areas. In this study, the Arduino is used as the control core of the obstacle avoidance car, and image and video processing which can be useful in many aspects.

E. FUTURE STEPS

AI, Deep Learning and machine learning will have a transformative impact on industrial robots. While these technologies are still in their infancy, they will continue to push the boundaries of what's possible with industrial robotic automation over the next few decades.

5. REFERENCE

[1] Z. Wu, L. Liu, Y. An, J. Wu, and H. Shao, "The intelligent robot arm based on sense of sight," in *Intelligent Control and Automation (WCICA), 2016 12th World Congress on*, 2016, pp. 1229-1233: IEEE.

[2] Pandey, S. Pandey, and D. Parhi, "Mobile robot navigation and obstacle avoidance techniques: A review," *Int Rob Auto J*, vol. 2, no. 3, p. 00022, 2017.

[3] Grokhotkov, "ESP8266 arduino core documentation," *ESP8266*, 2017.

[4] L. Hu, Y. Miao, G. Wu, M. M. Hassan, and I. Humar, "iRobot-Factory: An intelligent robot factory based on cognitive manufacturing and edge computing," *Future Generation Computer Systems*, vol. 90, pp. 569-577, 2019.

[5] Y. ZHU, J. CHEN, B. ZHANG, and Y. ZHOU, "Design of Intelligent Control System for Mobile Robot Based on Arduino," *Electronic Science and Technology*, vol. 5, p. 038, 2017.

[6] A. C. Bento, "IoT: NodeMCU 12e X Arduino Uno, Results of an experimental and comparative survey," *International Journal*, vol. 6, no. 1, 2018.