Minor Project (EC-455) Report

On

Gesture Incepted Wireless Car

Submitted in the partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in

Electronics and Communications Engineering

(2018-2022)

UNIVERSITY SCHOOL OF INFORMATION & COMMUNICATION TECHNOLOGY

GURU GOBIND SINGH INDRAPRASTHA UNIVERSITY, DELHI

Under the guidance of MR. PARIJAT MATHUR

418 RAJESHWAR

420 Prakhar mishra

008 Ankit kumar

424 Divyansh singh

Declaration

I, RAJESHWAR DHANKAR student of B.Tech. (Electronics and Communication Engineering) hereby declare that the project work done on “GESTURE INCEPTED WIRELESS CAR” submitted to University School of Information, Communication and Technology, Guru Gobind Singh Indraprastha University, Dwarka, Delhi in partial fulfillment of the requirement for the award of degree of Bachelor of Technology comprises of my original work and has not been submitted anywhere else for any other degree to the best of my knowledge.

RAJESHWAR
41816412818
USICT, GGSIPU

Certificate

This is to certify that the Minor Project (EC-455) entitled "GESTURE INCEPTED WIRELESS CAR" submitted by RAJESHWAR, Roll No. 41816412818 in partial fulfillment for the requirement of the award of the degree of Bachelor of Technology in Electronics and Communication Engineering at USICT, GGSIPU, Delhi has been carried out under my supervision and guidance.

Place : Dwarka
Date : 04/01/2022

Mr. PARIJAT MATHUR
Assistant Professor
USICT, GGSIPU

Acknowledgement

I want to express my gratitude to Dean, USICT for providing resources required for this project work. No volume of words is enough to express my gratitude towards my guide Mr. PARIJAT MATHUR sir who has been very concerned and has aided for all the materials essential for the preparation of this report.

He has helped me to explore this vast topic in an organized manner and provided me with all the ideas on how to work towards a research- oriented venture. I am thankful to him for the motivation and inspiration that triggered me for the project work.

ABSTRACT

The Internet of things (IoT) describes physical objects & their sets that are embedded with sensors, processing ability, software, and other technologies, and that connect and exchange data with other devices and systems over the Internet or other communications networks.

In keeping with the ever increasing acceptance of Home IoT, Hand Gestures are demonstrated here to actuate a control system, which is a car. The car is controlled wirelessly using RF modules and the prime gesture recognition component APDS9960.

Wireless actuation has been enhanced with a dual application of controlling the car with gestures fed into a camera and recognised with computer vision (OpenCV).

Index

1. Gesture Controlled Car
    a. Introduction
    b. Components
    c. Schematic
    d. Working

2. Controller
    a. Introduction
    b. Components
    c. Schematic
    d. Code

3. Computer Vision Application
    a. Introduction
    b. Methodology
    c. Code

4. Compiled Project
    a. Circuit
    b. Running the project

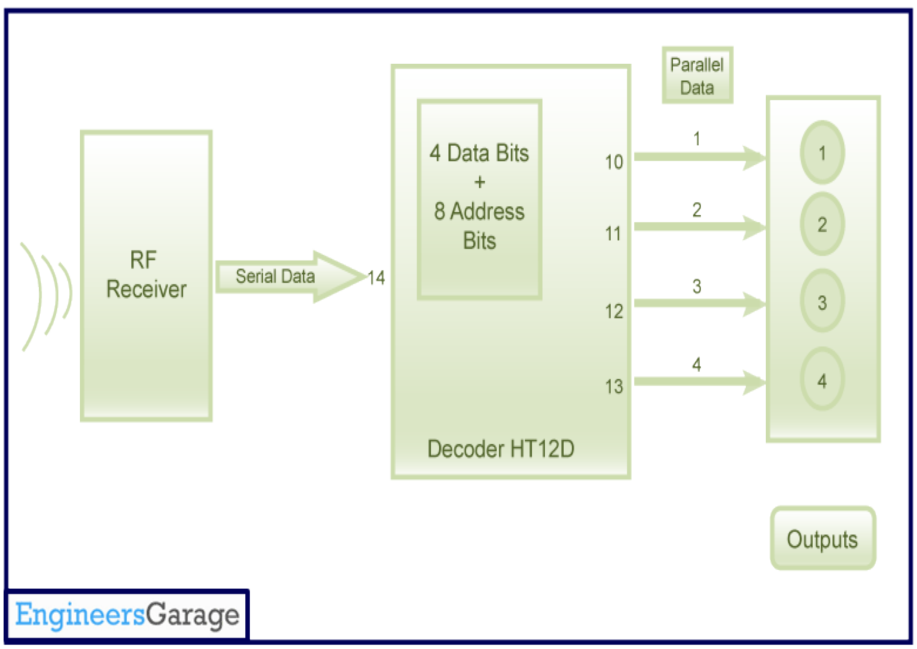
5. Conclusion

6. References

Gesture Controlled Car

Introduction

2 bits of data is transmitted consisting of 8 address bits and 4 data bits. The signal is received at the receiver's end which is then fed into the decoder IC. If address bits get matched, the decoder converts it into parallel data and the corresponding data bits get lowered which could be then used to drive the LEDs or any device in consideration. The outputs from this system can be used to drive relays or other electronic devices that can operate on receiving a logic high.



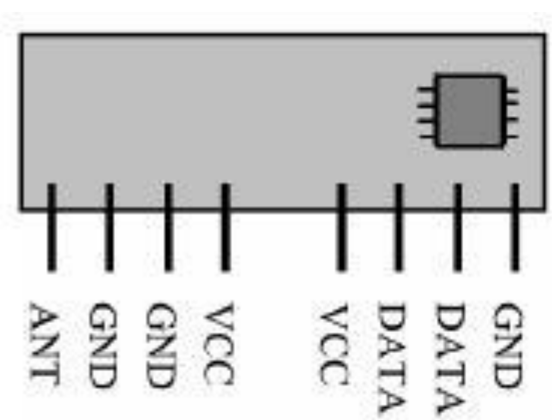
The car is controlled via the input encoded serial communication from the transmitter. Signals are encoded by the encoder with predefined numbers for **FORWARD, BACKWARD, LEFT, RIGHT and STOP** signals.
The signal when received by the RF module is fed into the decoder which turns it from analog to digital (4 output bits).
The signal is then given to the motor driver via PINS D0, D1, D2, D3 into the two motor drivers.

Components Used

RF Receiver Module:

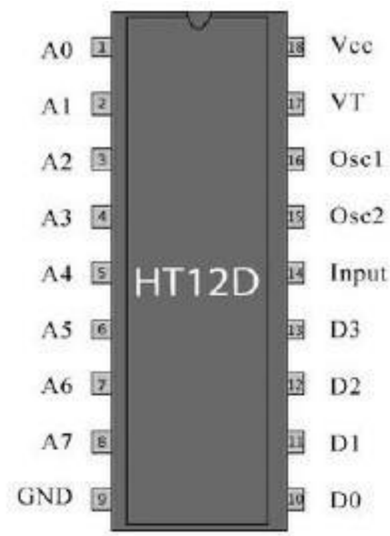
The RF module, as the name suggests, operates at Radio Frequency. The corresponding frequency range varies between 30 kHz & 300 GHz. In this RF system, the digital data is represented as variations in the amplitude of the carrier wave. This kind of modulation is known as Amplitude Shift Keying (ASK).

Transmission through RF is better than IR (infrared) because of many reasons. Firstly, signals through RF can travel through larger distances making it suitable for long range applications. Also, while IR mostly operates in line-of-sight mode, RF signals can travel even when there is an obstruction between transmitter & receiver. Next, RF transmission is more strong and reliable than IR transmission. RF communication uses a specific frequency unlike IR signals which are affected by other IR emitting sources. This RF module comprises an RF Transmitter and an RF Receiver. The transmitter/receiver (Tx/Rx) pair operates at a frequency of 434 MHz. An RF transmitter receives serial data and transmits it wirelessly through RF through its antenna connected at pin4. The transmission occurs at the rate of 1Kbps - 10Kbps.The transmitted data is received by an RF receiver operating at the same frequency as that of the transmitter.



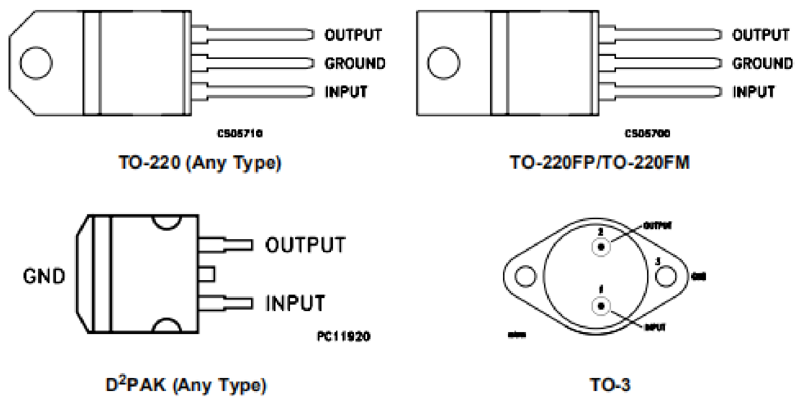
HT12D Decoder:

The primary function of HT12D is to decode the 12-bit that is received by the input pin. Since the IC comes with an in-built Oscillator it is very easy to make this IC work. The IC should be powered by 5V (pin 18) and the ground pin (pin 9) is grounded. For decoding a data the IC will require an oscillator, luckily this IC has one in-built. We just have to connect the OSC1 and OSC2 (pin 15 & 16) through a 470K resistor to invoke it. The 4-bit data that is received can be obtained on pins AD0 to AD1 and an address of 8-bit has to be set using the pins A0 to A7. It is very important that your Decoder should have the same address as that of the encoder. A basic circuit diagram for the HT12D IC is shown below.



L7805CV Voltage Regulator:

The L78 series of fixed voltage regulators are designed with thermal overload protection that shuts down the circuit when subjected to an excessive power overload condition, internal short-circuit protection that limits the maximum current the circuit will pass, and output transistor safe-area compensation that reduces the output short circuit current as the voltage across the pass transistor is increased. In many low current applications, compensation capacitors are not required.

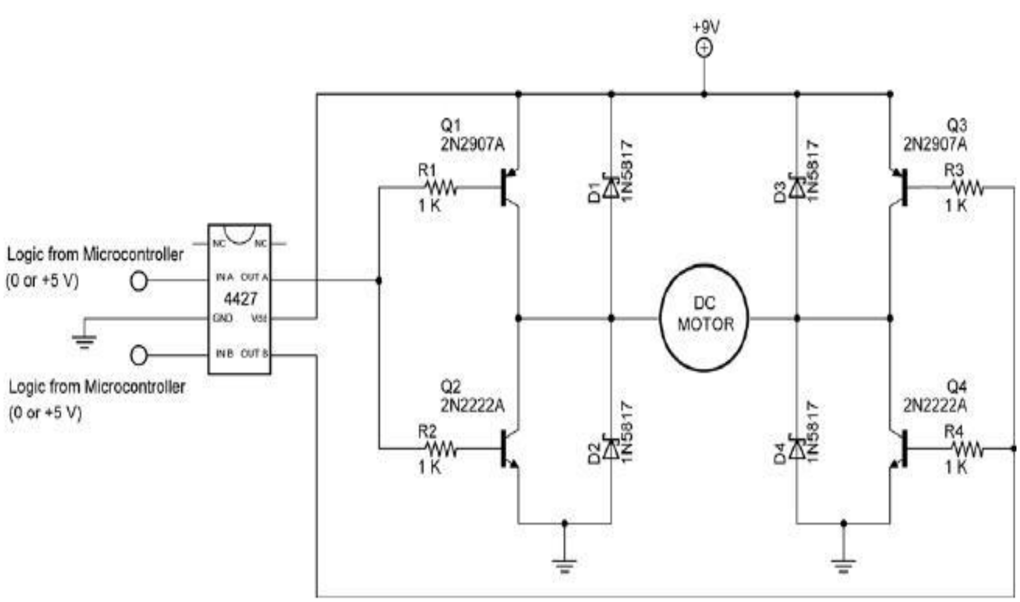


Motor Driver:

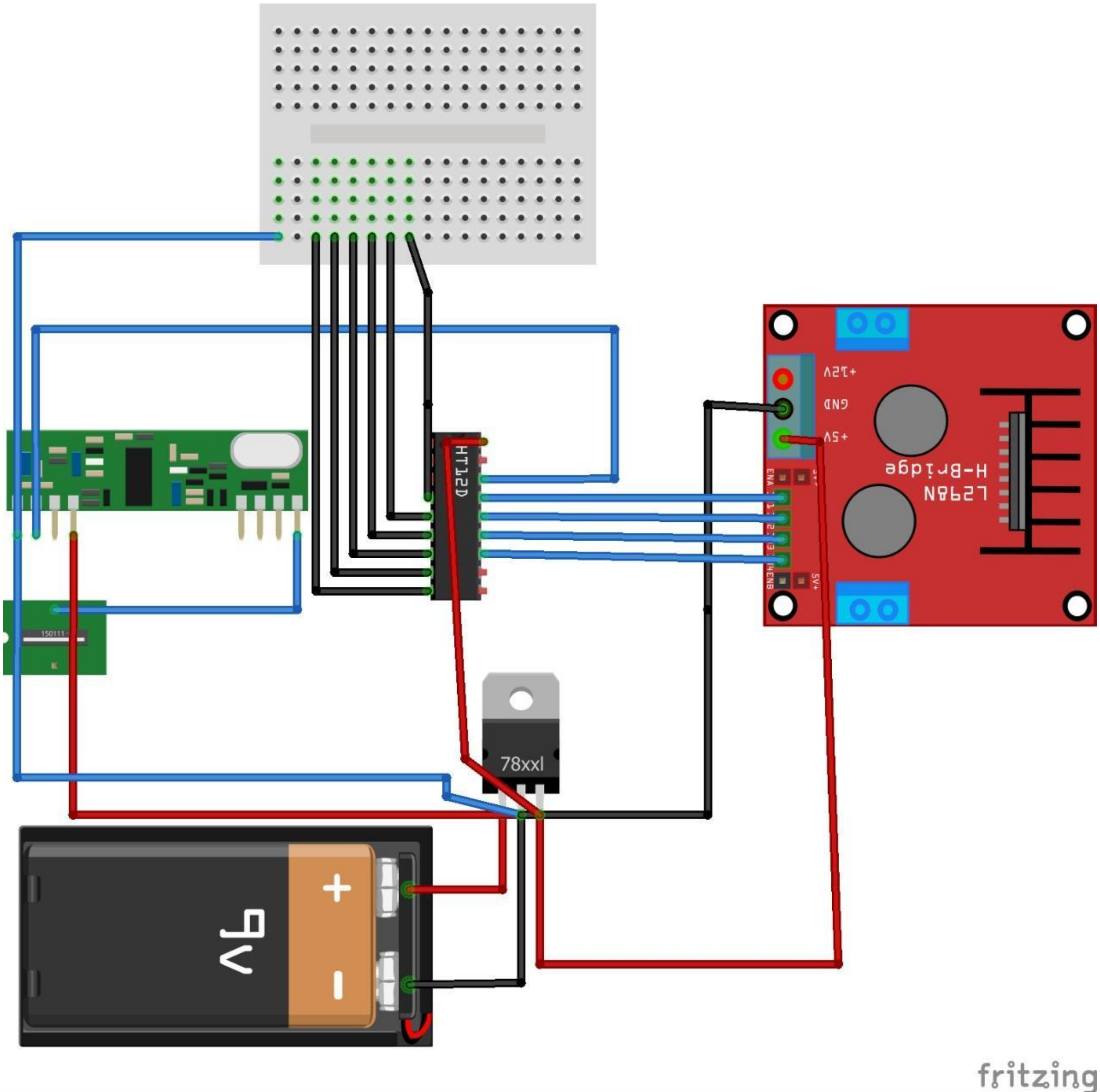
The L78 series of fixed voltage regulators are designed with thermal overload protection that shuts down the circuit when subjected to an excessive power overload condition, internal short-circuit protection that limits the maximum current the circuit will pass, and output transistor safe-area compensation that reduces the output short circuit current as the voltage across the pass transistor is increased. In many low current applications, compensation capacitors are not required.

DC Motors:

When kept in a magnetic field, a current-carrying conductor gains torque and develops a tendency to move. In short, when electric fields and magnetic fields interact, a mechanical force arises. This is the principle on which the DC motors work.



Schematic:



Working

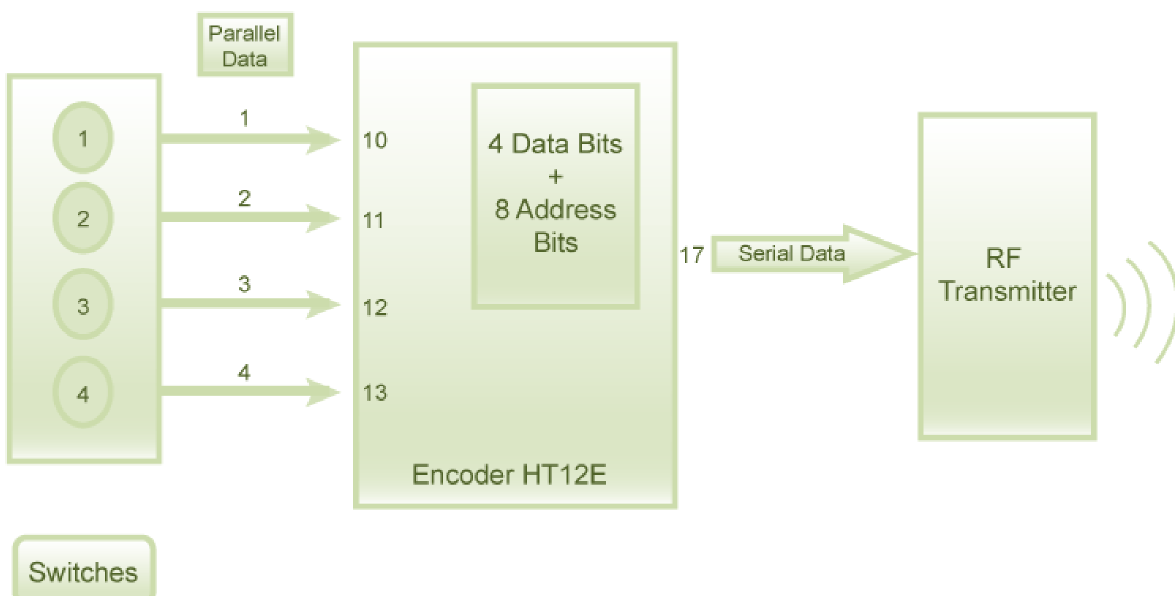
The CAR is controlled via the input encoded serial communication form the transmitter. Signals are encoded by the encoder with predefined signals for FORWARD, BACKWARD, LEFT, RIGHT and STOP signals. After receiving the input signal , the signal is fed into the decoder and the serial bit stream is decoded. The signal is then given to the motor driver via PIN D0 , D1,D2,D3 into the two motor drivers.

If the two motors both move forward/backwards the car moves forward/backwards. If the motors rotate in different directions, the car takes a turn. The 4 bit signals for these movements were predefined and mapped to different gestures through the code.

Controller

Introduction

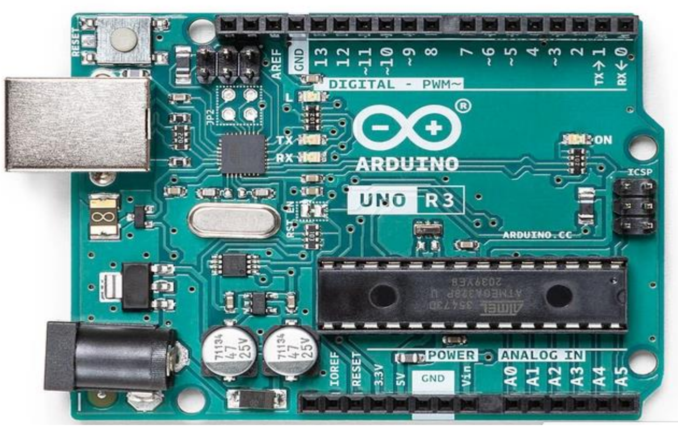
The Transmitter module comprises Arduino microcontroller ,decoder , RF Transmitter chip and Gesture sensor(APDS9960). In this section the analog input from the sensor is fed into the Arduino PIN SDA and SCL for data transfer(SCL for the Clock synchronisation) and Arduino chip ATMEGA328 interprets the analog signal with the help of requisite library function. ATMEGA328 chip is then coded for 4 bit output signal from PIN D7,D8,D9,D10 and these outputs account for directional motion of the car. A 4 bit signal is then used as input to decoder and converted serial data is transferred via ASK modulated RF signal.



Components

Arduino UNO board:

Arduino UNO is a very valuable addition in electronics that consists of a USB interface, 14 digital I/O pins(of which 6 Pins are used for PWM), 6 analog pins and an Atmega328 microcontroller. It also supports 3 communication protocols named Serial, I2C and SPI protocol This board contains a USB interface i.e. A USB cable is used to connect the board with the computer and Arduino IDE (Integrated Development Environment) software is used to program the board. The unit comes with 32KB flash memory that is used to store the number of instructions while the SRAM is 2KB and EEPROM is 1KB.



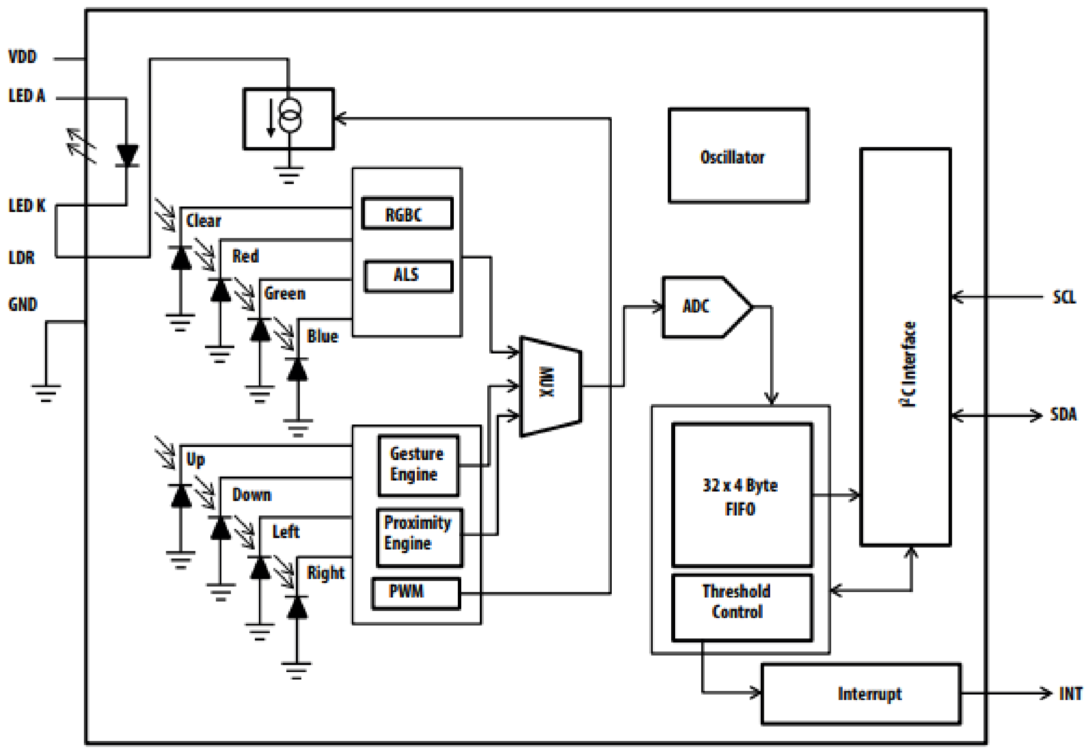
Gesture Sensor APDS9960:

The Broadcom APDS-9960 is a digital RGB, ambient light, proximity and gesture sensor device in a single 8-pin package. The device has an I2C compatible interface providing red, green, blue, clear (RGBC), proximity and gesture sensing with IR LED. The RGB

and ambient light sensing feature detects light intensity under various lighting conditions and through various attenuation materials including darkened glass. In addition, the integrated UV-IR blocking filter enables accurate ambient light and correlated color temperature sensing.

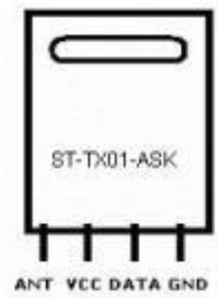
The proximity and gesture feature is factory-trimmed and calibrated to 100mm proximity detection distance without requiring customer calibrations. Gesture detection utilizes four directional photodiodes, integrated with a visible blocking filter, to accurately sense simple UP-DOWN-RIGHT-LEFT gestures or more complex gestures. The addition of micro-optics lenses within the module provides high efficient transmission and reception of infrared energy. An internal state machine allows the device to be put into a low power state between RGB, proximity and gesture measurements providing very low power consumption.

Functional Block Diagram



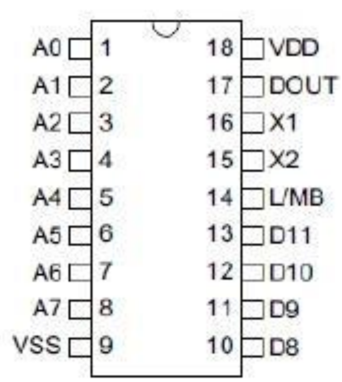
Encoder:

HT12E is an encoder integrated circuit of 212 series of encoders. They are paired with 212 series of decoders for use in remote control system applications. It is mainly used in interfacing RF and infrared circuits. The chosen pair of encoder/decoder should have the same number of addresses and data format. Simply put, HT12E converts the parallel inputs into serial output. It encodes the 12 bit parallel data into serial for transmission through an RF transmitter. These 12 bits are divided into 8 address bits and 4 data bits. HT12E has a transmission enable pin which is active low. When a trigger signal is received on the TE pin, the programmed addresses/data are transmitted together with the header bits via an RF or an infrared transmission medium. HT12E begins a 4-word transmission cycle upon receipt of a transmission enable. This cycle is repeated as long as TE is kept low. As soon as TE returns to high, the encoder output completes its final cycle and then stops.

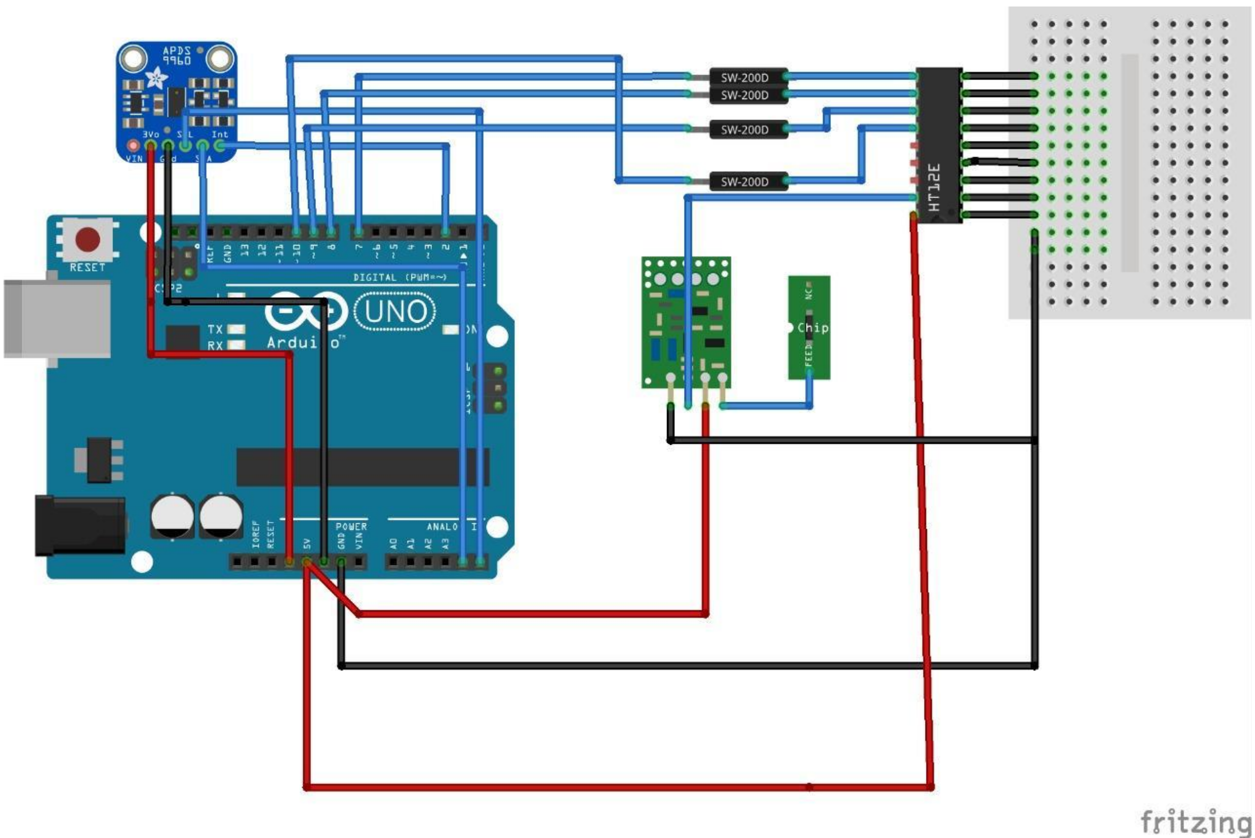


RF module (Transmitter):

This radio frequency (RF) transmission system employs Amplitude Shift Keying (ASK) with a transmitter/receiver (Tx/Rx) pair operating at 434 MHz. The transmitter module takes serial input and transmits these signals through RF. The transmitted signals are received by the receiver module placed away from the source of transmission. The system allows one way communication between two nodes, namely, transmission and reception. The RF module has been used in conjunction with a set of four channel encoder/decoder ICs. Here HT12E & HT12D have been used as encoder and decoder respectively.



Schematic:



Code: (Objective - C)

```
// 3.3V          VCC              Power
// GND           GND              Ground
// A4            SDA              I2C Data
// A5            SCL              I2C Clock
```

```
// 2             INT                 Interrupt

#include <Wire.h>
#include <SparkFun_APDS9960.h>

// Pins
#define APDS9960_INT    2 // Needs to be an interrupt
pin

// Constants

// Global Variables
SparkFun_APDS9960 apds = SparkFun_APDS9960();
int isr_flag = 0;
int P7 = 7;
int P8 = 8;
int P9 = 9;
int P10 = 10;

void setup() {

  // Set interrupt pin as input and other pins as
output
  pinMode(APDS9960_INT, INPUT);
```

```
  pinMode(P7, OUTPUT);

  pinMode(P8, OUTPUT);

  pinMode(P9, OUTPUT);

  pinMode(P10, OUTPUT);


  // Initialize Serial port

  Serial.begin(9600);

  Serial.println();

  Serial.println(F("--------------------------------
"));

  Serial.println(F("SparkFun APDS-9960 -
GestureTest"));

  Serial.println(F("--------------------------------
"));


  // Initialize interrupt service routine

  attachInterrupt(0, interruptRoutine, FALLING);


 // Initialize APDS-9960 (configure I2C and initial
//values)

  if ( apds.init() ) {

    Serial.println(F("APDS-9960 initialization
complete"));

  } else {
```

```
    Serial.println(F("Something went wrong during APDS-9960 init!"));
  }

  // Start running the APDS-9960 gesture sensor engine
  if ( apds.enableGestureSensor(true) ) {
    Serial.println(F("Gesture sensor is now running"));
  } else {
    Serial.println(F("Something went wrong during gesture sensor init!"));
  }
}

void loop() {
  if( isr_flag == 1 ) {
    detachInterrupt(0);
    handleGesture();
    isr_flag = 0;
    attachInterrupt(0, interruptRoutine, FALLING);
  }
}

void interruptRoutine() {
  isr_flag = 1;
```

```
}

void handleGesture() {
    if ( apds.isGestureAvailable() ) {
    switch ( apds.readGesture() ) {
      case DIR_UP:{
        Serial.println("UP");
        digitalWrite(P7, LOW);
        digitalWrite(P8, HIGH);
        digitalWrite(P9, HIGH);
        digitalWrite(P10, LOW);
        break;
      }
      case DIR_DOWN:{
        Serial.println("DOWN");
        digitalWrite(P7, HIGH);
        digitalWrite(P8, LOW);
        digitalWrite(P9, LOW);
        digitalWrite(P10, HIGH);
        break;
      }
      case DIR_LEFT:{
        Serial.println("LEFT");
```

```
  digitalWrite(P7, LOW);
  digitalWrite(P8, LOW);
  digitalWrite(P9, HIGH);
  digitalWrite(P10, HIGH);
  break;
}
case DIR_RIGHT:{
  Serial.println("RIGHT");
  digitalWrite(P7, HIGH);
  digitalWrite(P8, HIGH);
  digitalWrite(P9, LOW);
  digitalWrite(P10, LOW);
  break;
}
case DIR_NEAR:{
  Serial.println("NEAR");
  digitalWrite(P7, HIGH);
  digitalWrite(P8, HIGH);
  digitalWrite(P9, HIGH);
  digitalWrite(P10, HIGH);
  break;
}
case DIR_FAR:{
```

```
        Serial.println("FAR");
        digitalWrite(P7, HIGH);
        digitalWrite(P8, HIGH);
        digitalWrite(P9, HIGH);
        digitalWrite(P10, HIGH);
        break;
      }
      default:{
        Serial.println("NONE");
        digitalWrite(P7, HIGH);
        digitalWrite(P8, HIGH);
        digitalWrite(P9, HIGH);
        digitalWrite(P10, HIGH);
      }
    }
  }
```

Computer Vision Application

Introduction

Computer vision is a field of artificial intelligence (AI) that enables computers and systems to derive meaningful information from digital images, videos and other visual inputs — and take actions or make recommendations based on that information.

One of the driving factors behind the growth of computer vision is the amount of data we generate today that is then used to train and make computer vision better.

One such open source library is ***OpenCV,*** made by google. It is a huge open-source library for computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When integrated with various libraries, such as NumPy, python is capable of processing the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features. When OpenCV was designed the main focus was real-time applications for computational efficiency. All things are written in optimized C/C++ to take advantage of multi-core processing.

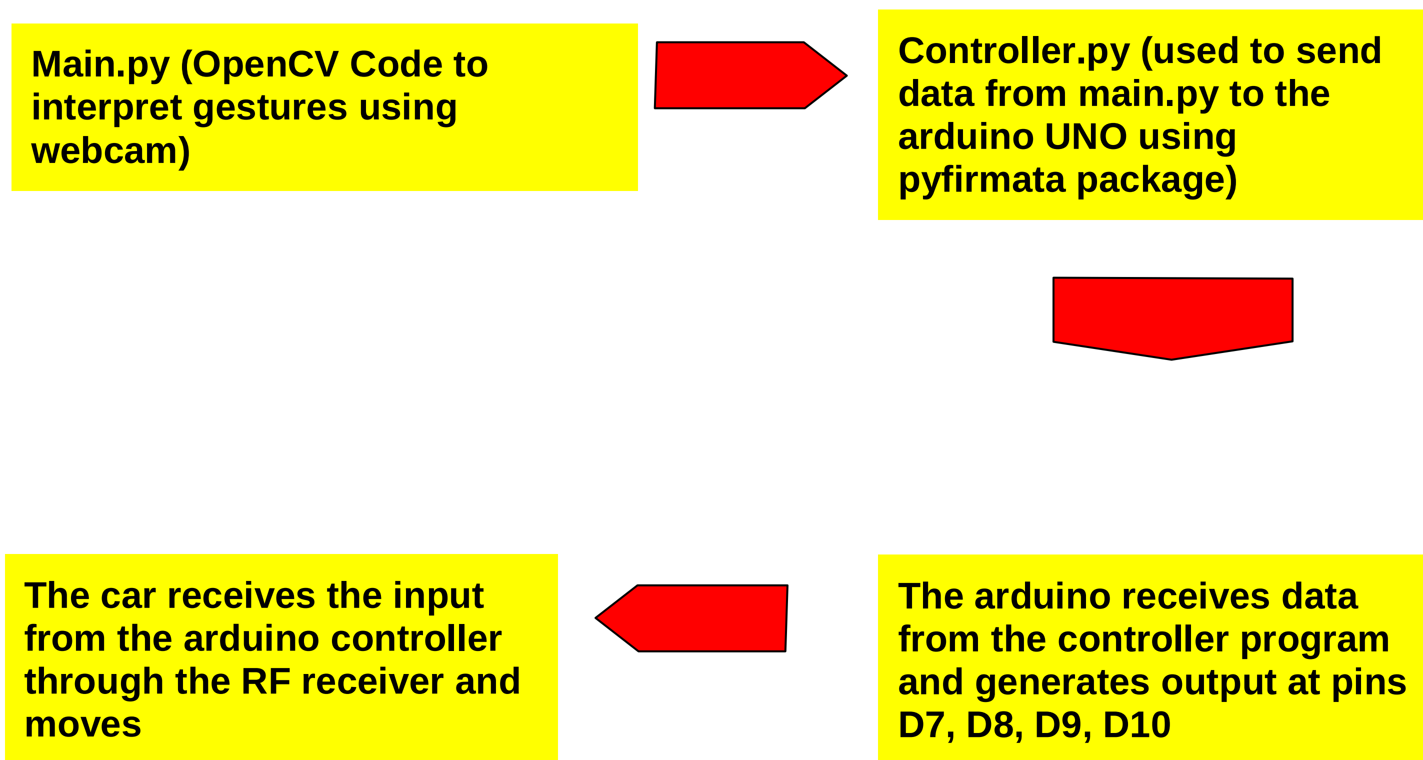
The *Hand Detection and Tracking* module of OpenCV is what we have used specifically in this project, to advance our gesture controlled functionality. It is capable of recognising human hands, as a result of being pre-trained with around 30000 images of hands.

Methodology

Using OpenCV's hand detection capabilities, we can detect a human hand with the help of 21 different data points on different locations of the hand, which at the programming level are objects containing XY coordinates and RGB values for each individual data point.

Using the relative positions of these data points, it is possible to differentiate the number of fingers being held up by the hand. The project utilises this functionality to create 5 different gestures each mapped to a certain motion the car goes through.

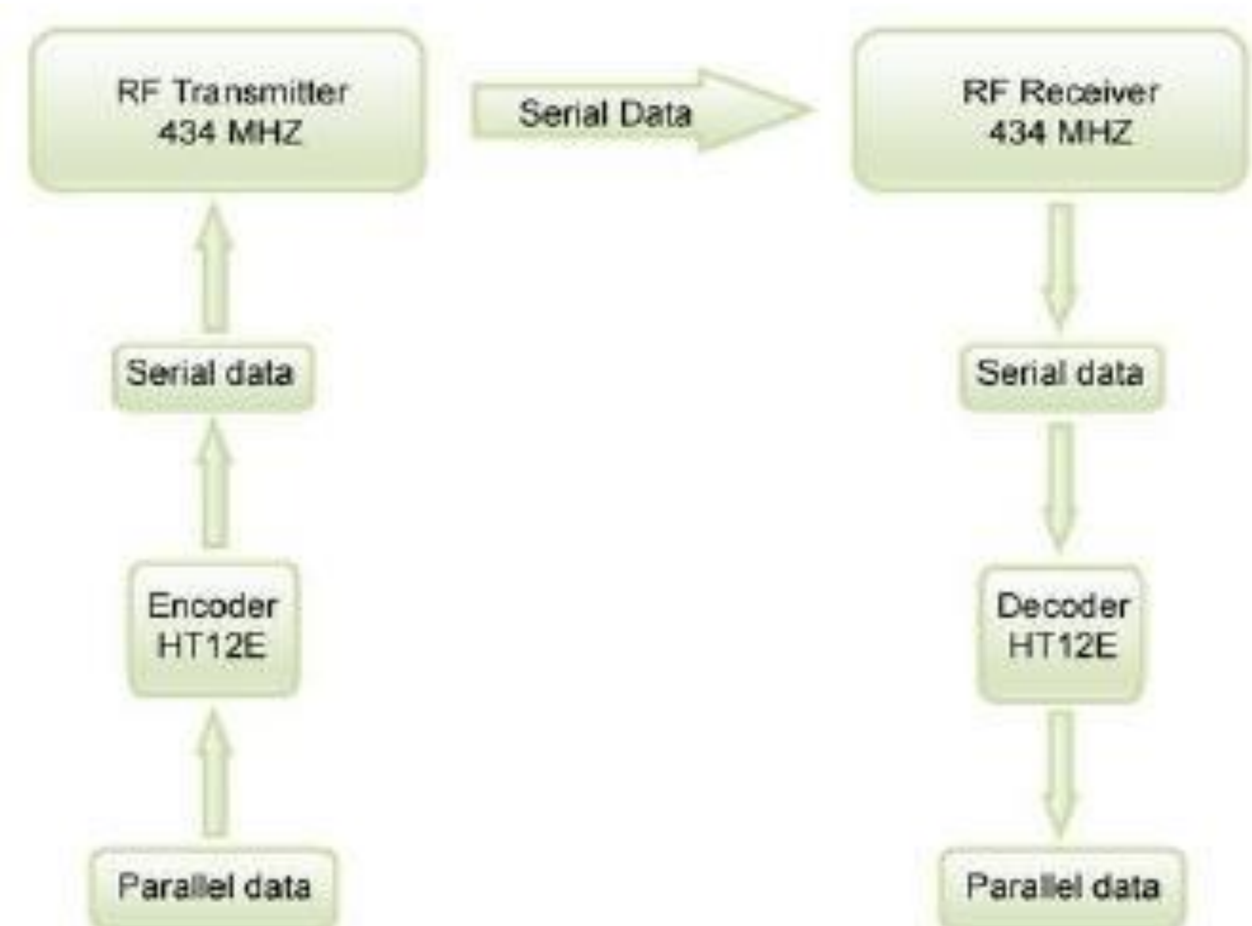
Computer Vision Application Architecture:



Code : https://github.com/prakhartp

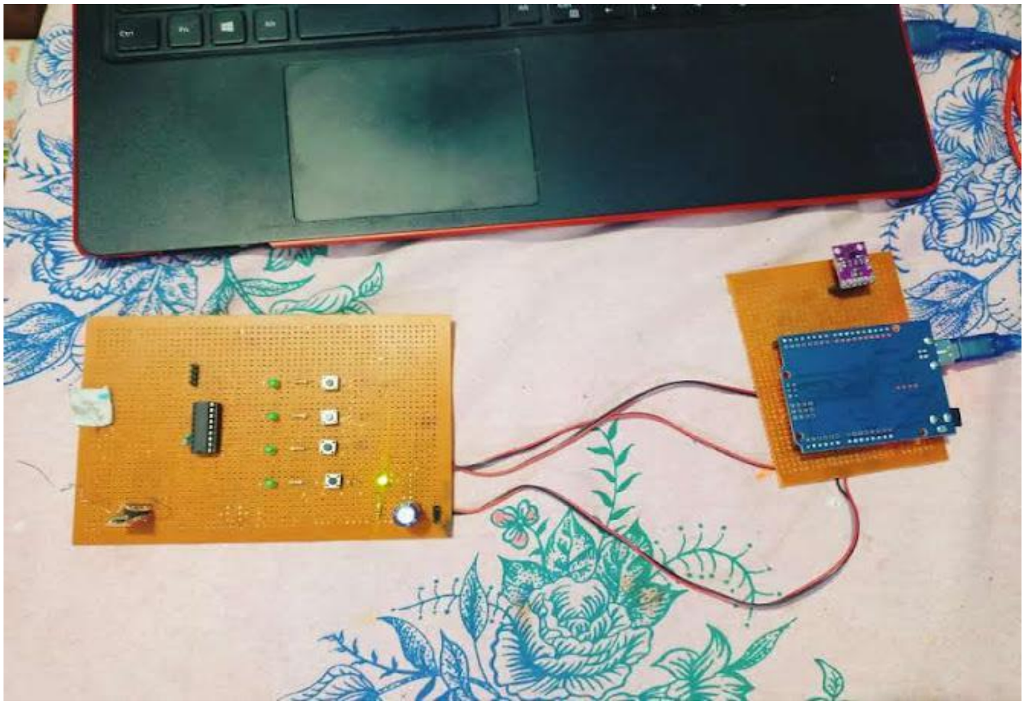
Compiled Project

The final project is compiled with the gesture car and transmitter connected through the RF ASK communication with transmitter and receiver module. The project is also enhanced with Computer Vision functionality.

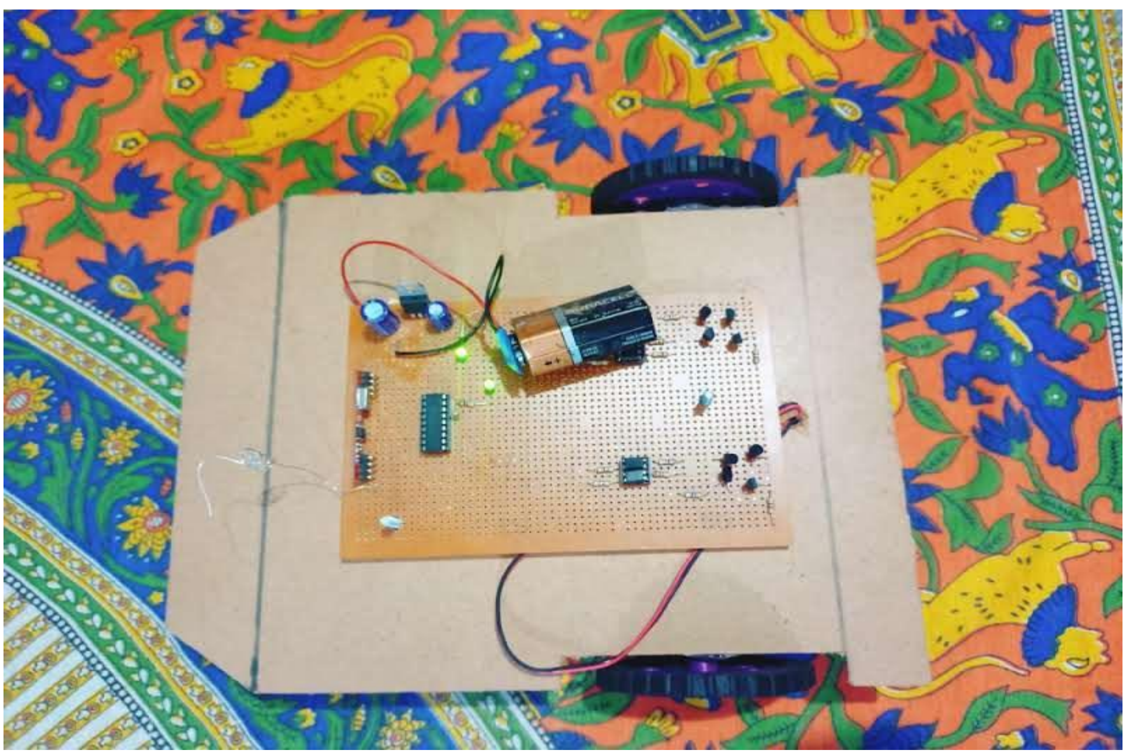


The gesture is firstly recognized from the APDS9960 and its analog input is sent to encoder IC which converts 4 bit output signal to serial data and transfers it through RF signal. The encoded serial is received at the receiver module of the RF module and its data is sent to decoder IC. Serial data is converted back to 4 bit original output and the motor driver is run in the motor driver circuit.

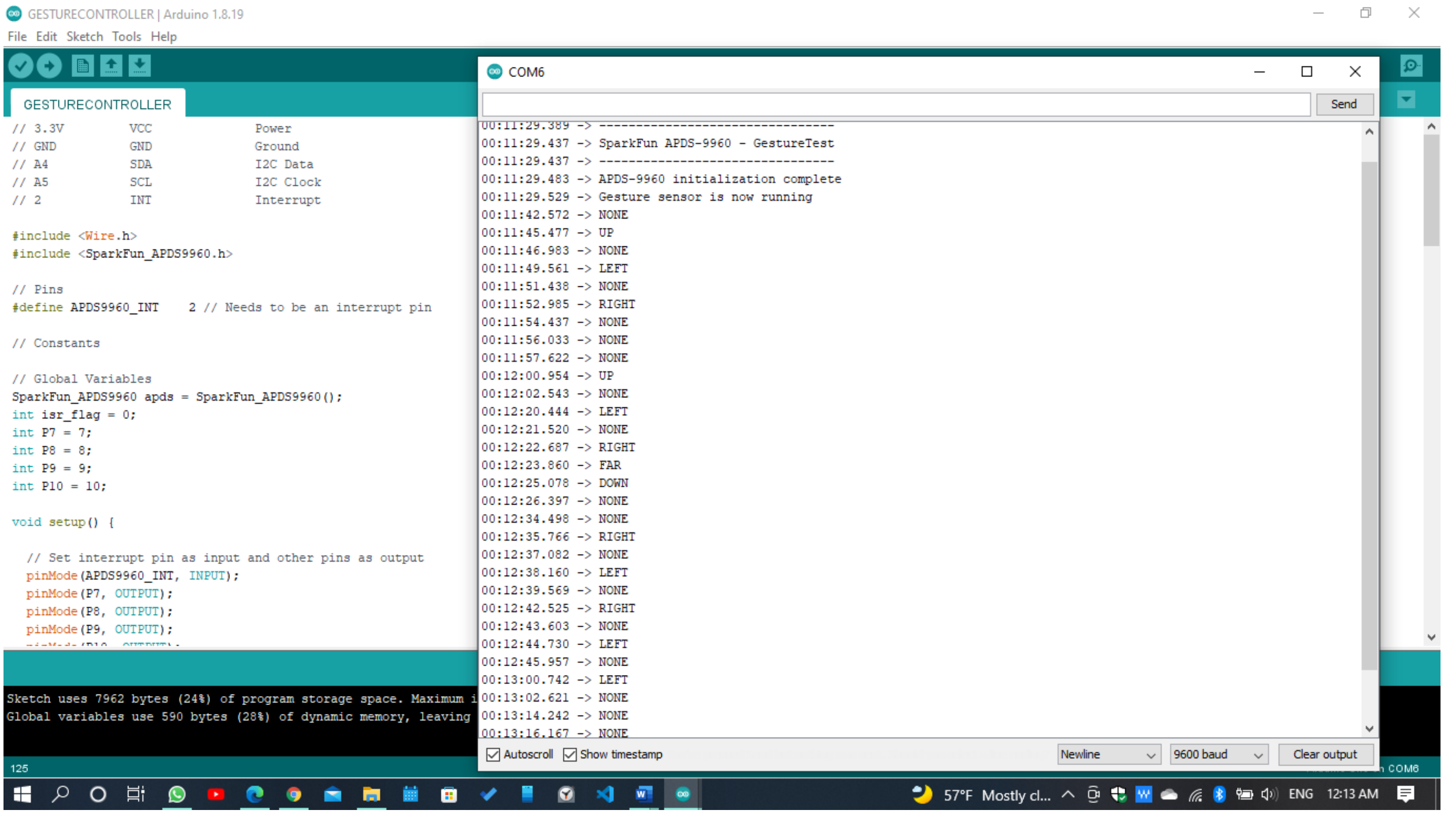
Controller Module:



Car:



Running the project



The above Image shows how the various gestures are logged and displayed in the arduino IDE Serial Monitor window.

Representing this data in a tabular form:

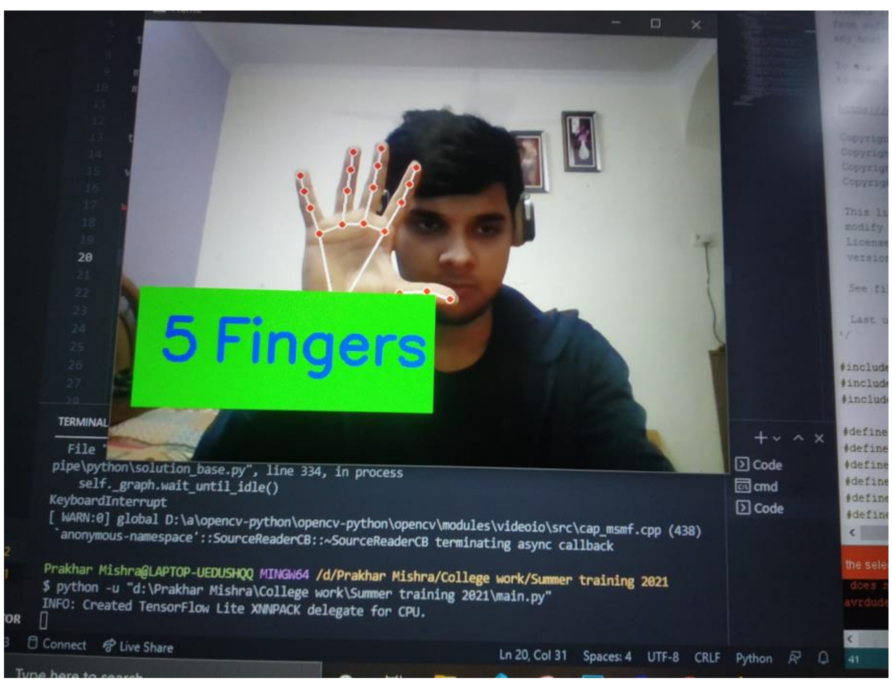
SERIAL NO.	GESTURE	OUTPUT	RESULT
1	FORWARD	UP	SUCCESS
2	LEFT	LEFT	SUCCESS
3	RIGHT	RIGHT	SUCCESS
4	BACKWARD	ERROR	FAILURE
5	STOP	NONE	SUCCESS
6	FORWARD	UP	SUCCESS

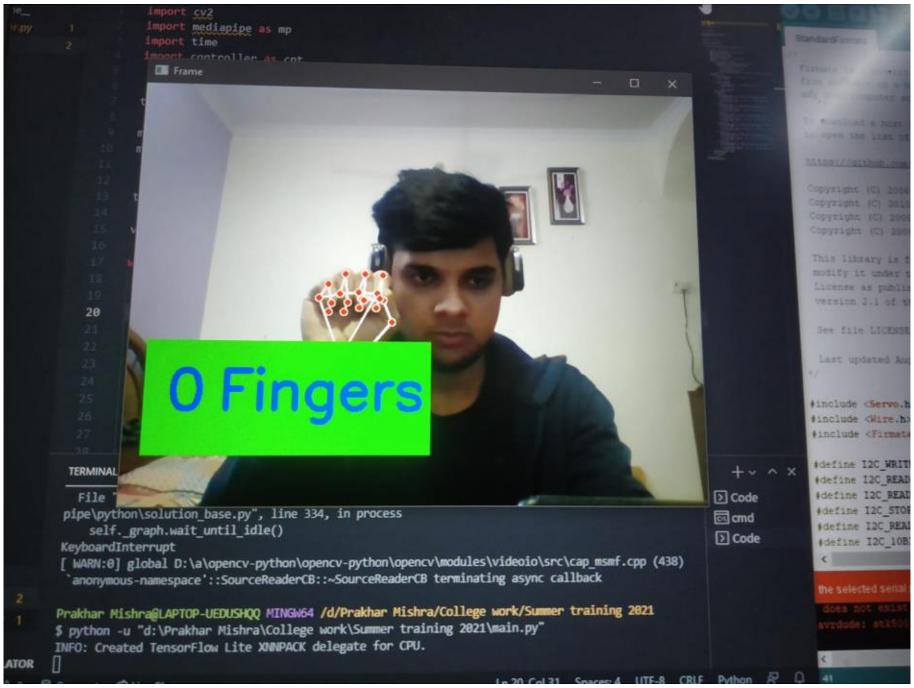
7	LEFT	LEFT	SUCCESS
8	RIGHT	RIGHT	SUCCESS
9	BACKWARD	DOWN	SUCCESS
10	STOP	NONE	SUCCESS
11	FORWARD	UP	SUCCESS
12	LEFT	LEFT	SUCCESS
13	RIGHT	RIGHT	SUCCESS
14	BACKWARD	ERROR	FAILURE
15	STOP	NONE	SUCCESS

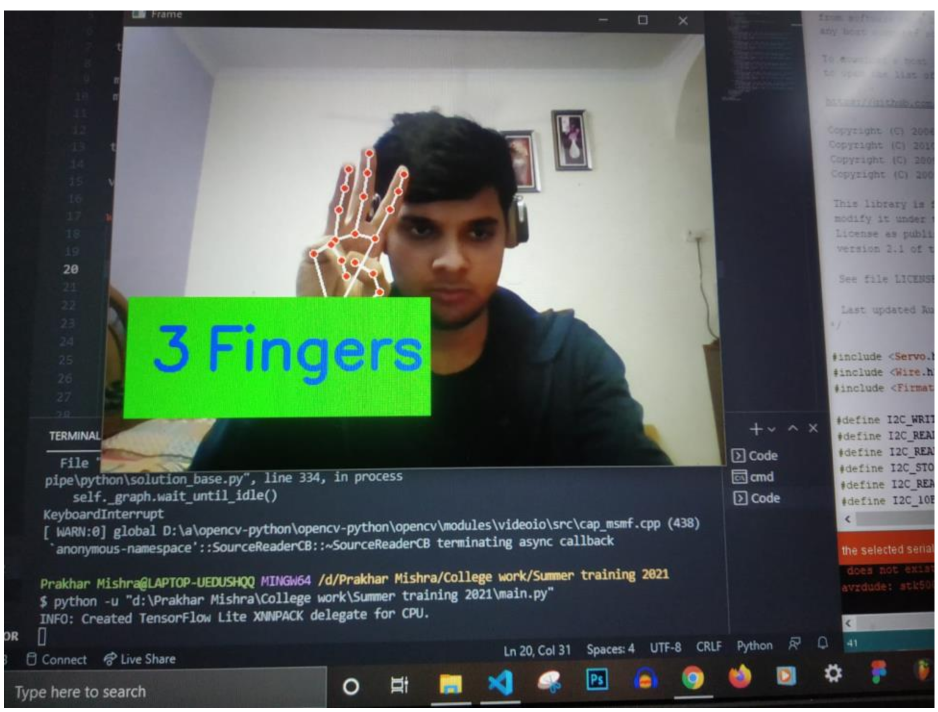
Percentage Error:

Error = 2*15/100 = 13.3%

Gesture recognition using OpenCV (Pictorial Representation):




import cv2
import mediapipe as mp
import time
Frame
0 Fingers
TERMINAL
pipe\python\solution_base.py", line 334, in process
self._graph.wait_until_idle()
KeyboardInterrupt
[WARN:0] global D:\a\opencv-python\opencv-python\opencv\modules\videoio\src\cap_msmf.cpp (438)
`anonymous-namespace'::SourceReaderCB::~SourceReaderCB terminating async callback
Prakhar Mishra@LAPTOP-UEDUSHQQ MINGW64 /d/Prakhar Mishra/College work/Summer training 2021
$ python -u "d:\Prakhar Mishra\College work\Summer training 2021\main.py"
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.


3 Fingers
TERMINAL
pipe\python\solution_base.py", line 334, in process
self._graph.wait_until_idle()
KeyboardInterrupt
[WARN:0] global D:\a\opencv-python\opencv-python\opencv\modules\videoio\src\cap_msmf.cpp (438)
`anonymous-namespace'::SourceReaderCB::~SourceReaderCB terminating async callback
Prakhar Mishra@LAPTOP-UEDUSHQQ MINGW64 /d/Prakhar Mishra/College work/Summer training 2021
$ python -u "d:\Prakhar Mishra\College work\Summer training 2021\main.py"
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
Ln 20, Col 31
Spaces: 4
UTF-8
CRLF
Python
Type here to search

Conclusion

The wireless gesture controlled car is a reliable and user driven way to communicate with the drive control of a car & consists of affordable hardware which can be interfaced with any microcontroller for data processing .

This model is very useful for easy learning of cars & moving robots and wireless control and also useful for physically handicapped users so that if this system is integrated it can elegantly supplement the end user.

Test runs revealed minor gesture recognition error which can be improved with additional capabilities like machine learning and by shifting to a superior processor albeit at the cost of increased system price.

The work can be upgraded to accommodate any moving robot with 3 axes control.

References

1. https://cdn.sparkfun.com/assets/learn_tutorials/3/2/1/Avago-APDS-9960-datasheet.pdf
2. https://supplyframe.componentsearchengine.com/viewer/3D.php?partID=179428
3. https://en.wikipedia.org/wiki/Arduino_Uno#:~:text=The%20Arduino%20Uno%20is%20an,and%20developed%20by%20Arduino.cc.&text=The%20board%20has%2014%20digital,a%20type%20B%20USB%20cable.
4. https://2020.robotix.in/tutorial/auto/motor_driver/
5. https://docs.opencv.org/4.x/d9/df8/tutorial_root.html
6. https://roboticsbackend.com/control-arduino-with-python-and-pyfirmata-from-raspberry-pi/