An-Najah National University Faculty of Engineering

Electrical Engineering Department

Solar Powered Autonomous Battery Electric Vehicle Model

By:

Eng. (Ibrahim Turkman)

Eng. (Yazan Bajawi)

Supervisor

Dr. Raed Jaber

Presented in Partial fulfillment of the requirements for Bachelor's degree in ''Electrical Engineering ''

The 7th Of June 2023

Acknowledgment

We would like to convey our heartfelt gratitude to our parents and our families who gave us tremendous support during our educational career, and without them, we wouldn't be here on the verge of graduation. To them, we say from the deeps of our hearts" thank you".

We would also like to thank our adviser and our doctor, Read Jaber, that completion of the project would not be possible without his help and insights.

Ibrahim Turkman

Yazan Bajawi

Contents

Table of figures

ABSTACT

*In this project we made an autonomies hybrid BEV (*Battery Electric Vehicle) *that has two microcontrollers one as a master and the other one as a slave device, the master device is considered the brain of the system that monitor the road, do image processing take final decisions and send commands to the slave which in turn drive the motors to move in the desired direction and speed.*

A camera fixed at the top of the car is used to send continues video stream to master device, which in turn process the stream and examine the status of track and direct the car to move smoothly along the track

In order to programed the master device and monitor the whole operations done by it we used A VNC player installed at laptop and connected to the master wirelessly.

The main power source for the BEV is Lithium batteries and a 1000 ma power bank which is a part of an integrated power system that work in harmonic with other systems in the BEV.

1 CHAPTER 1. INTRODUCTION

1.1 OVERVIEW

The world consistently moves toward renewable energy sources and get rid off traditional energy sources along with the seek for automation and comfortable, and there's great efforts are done to accomplish these goals as soon as possible, as we sensed the importance of these goals, we decided to support the efforts in our project,

Our project consists of three parts: the first is an electric vehicle that uses four electric motors instead of petrol engine, in this part we eliminate the emissions that come out of petrol vehicles, the second part is an integrated solar power system that converts solar power into DC current which supply a lithium battery that provide the vehicle with electric power. Least but not last a self-driving system which monitor the track, and the vehicle status then take the right decisions without any human interference.

And as we are existing in a transit state between two generation of vehicles, we decided to provide the vehicle with another system to charge the batteries from grid electricity.

1.2 EXISTING PROBLEMS

If fully electric vehicles are to become mainstream and grab a sizeable share of the car parc, there are six challenges to overcome: Customer acceptance, charging infrastructure, chip shortages, battery shortages, reliance on rare earth materials, and the ability to have multiple owners.

The list was put together by Lang Marketing in a recent Aftermarket iReport.

"In expanding their VIO share, EVs face six major challenges, ranging from manufacturing capacity and infrastructure issues to consumer needs," it stated. "While some of these challenges are probably fleeting, others will persist for the balance of the decade and, perhaps, beyond."

Tesla currently dominates the market, making up 80 per cent of electric vehicle sales in the U.S. With about 160 new nameplates on the way, other automakers will need to gain the trust of consumers, not just Tesla enthusiasts.

Infrastructure to allow drivers to charge their vehicles away from home will be another hurdle. While most EV buyers will have a charging station at home, greater availability will allow for a broader customer base. Think of those who live in apartments or don't have a driveway for their vehicle.

Shortages of two key materials will also present challenges. Computer chips are already an issue, as seen in the current new vehicle segment. That is expected to linger for a few more years, Lang noted.

"EVs are semiconductor dependent and continued chip shortages could curtail EV production over the next year or two," the report added.

However, more significant is the lack of battery manufacturing capacity. Lang observed that experts have indicated that battery shortages will reduce the global production of EVs by more than 20 million between 2022-2029.

Keep an eye on rare earth metals, Lang said. These are metals that are difficult to find in large quantities to make mining possible, and they're critical for manufacturing the types of electric motors most found in current EV designs.

Finally, a key question is how EVs will pass through different owners. Most people don't buy new vehicles. So can EVs live long enough to be passed down to a second or third owner?

"ICE-powered vehicles have proven to be very durable and easy to repair," the report said. "It remains to be seen if electric vehicles will be able to fill this same market need across the broad range of secondary buyers in the U.S."

Todd Campau, global aftermarket solutions associate director with IHS Markit raised a similar question during his AAPEX 2021 presentation *5 Trends in 5 Minutes*.

He found that from 2011 to 2020, 79 per cent of EVs sold during this time are still with the original owner. That's a vastly different story from gas vehicles where half have changed hands.

1.3 AIMS AND OBJECTIVES

The major goal of this project is to produce a prototype for an autonomies BEV that uses solar power as a major power source for the vehicle.

We also aim to make a compatible self-driving system that walk with the artificial intelligence revolution the world has these days, to achieve comfortable and safety on roads by eliminating the major Coase for accidents that is humans.

we chose this project because we believe that the t world is developing rapidly and within ten years, cars will not remain as they are.

We hope to find a sponsor that adapts our ideas to make a real Eco-friendly autonomies vehicle that would make a breakthrough in BEVs market.

1.4 SCOPE OF THE WORK

1.4.1 GENERAL OVERVIEW

- A chassis moving with all its contents to get a robot vehicle sub-system
- The costumer sends the command over the wireless cloud using mobile to the micro controller.

- Raspberry pi will receive commands and send them to the slave device (Arduino) that in turns will move dc motors to move in the specified direction.
- For the power supply we used lithium batteries that will be charged by solar system fixed on the VEB and by grid electricity.

1.4.2 HARDWARE OVERVIEW

- Raspberry pi
- Arduino uno
- Solar Panel
- TP4056 lion battery charging module
- TPS61088 boost step up
- Camera module is the new thing added to a perfect project to let you see where your robot vehicle is going.
- DC motors main parts for the vehicle to get motion on the surface through a motor driver for speed and direction control.

- Lithium battery to feed all circuits and motors.

1.4.3 SOFTWARE OVERVIEW

This project can execute commands via raspberry pi and Arduino as they are issued from the pc via Wi-Fi.

These two pieces are a small computer with C++ programming language and Arduino language so Arduino is object-oriented programming language that means the real-time tasks for the sequential reading and sending of each communication with camera stream and for motors controls.

The raspberry Pi can be controlled by a laptop over a Wi-Fi communication the by remote desktop we can access it and program it by C++ in GenY

While arduino can be opened from a laptop and controlled with some software like ARDUINO-1.8.16 that allows access to it from a laptop.

Also, there are software overview explanations found later in this report to get access to the final robot vehicle project.

1.5 IMPORTANCE OF THE WORK

The increasing interest in renewable and sustainable energy systems has prompted many companies operating in the field of car manufacturing and research institutes to work on developing solar-powered cars as a sustainable means of transportation, as some estimates indicated that the value of the solar car market is expected to reach $ 689 million by 2027. Therefore, many car companies are stepping up their efforts to take advantage of this growing market.

On autonomy, the swissinfo.ch headline read: By 2040, nearly one in five cars in Switzerland should be fully automated, with everyone on board able to close their eyes and relax (at least in theory). However, attitudes toward transportation, and in particular shared transportation, will have environmental impacts and will increase the demand for energy resources.

Here we enclose a recent report for the year 2023 by journalist Jenny Kozak, who works for the BBC .

2 CHAPTER2. Theoretical Background and Previous Work

The inspiration of this project came from technology, economy and environmental visions. in our project we have studied as much as we can of projects that established about BEV's and solar powered BEV's since first solar powered vehicle ever made by William Cobb to the newest versions of tesla that It is noteworthy that it's an automotive BEV's that doesn't use renewable energy sources.

And we think its inspiring to mansion cobbs vehicle so we will quote from an article to History.com Editors : " On August 31, 1955, William G. Cobb of the General Motors Corp. (GM) demonstrates his 15-inch-long "Sun mobile," the world's first solar-powered automobile, at the General Motors Powe Rama auto show held in Chicago, Illinois.

Cobb's Sun mobile introduced, however briefly, the field of photovoltaics–the process by which the sun's rays are converted into electricity when exposed to certain surfaces–into the gasoline-drenched automotive industry. When sunlight hit 12 photoelectric cells made of selenium (a nonmetal substance with conducting properties) built into the Sun mobile, an electric current was produced that in turn powered a tiny motor. The motor turned the vehicle's driveshaft, which was connected to its rear axle by a pulley. Visitors to the month-long, $7 million Powe Rama marveled at some 250 free exhibits spread over 1 million square feet of space on the shores of Lake Michigan. In addition to Cobb's futuristic mini-automobile, Powe Rama visitors were treated to an impressive display of GM's diesel-fueled empire, from oil wells and cotton gins to submarines and other military equipment."

CHAPTER 3. METHODOLOGY

2.1 SOFTWARE IMPLEMENTATION

First, we will list the software's we used in our project with a brief explanation about each one then, we will defenestrate the coding process step by step.

2.1.1 Geany

It’s a lightweight text editor that support over fifty programming languages so it can be used in anything related to coding.

Geany is a robust IDE that compatible with most common programming languages such as python, CPP, Java, etc.

2.1.2 Open CV

OpenCV (Open-Source Computer Vision Library) is huge open-source library for computer vision, image processing, and machine learning and now it plays a major role in real-time operation which is essential for nowadays systems. It enable us to process images and videos to identify objects, faces, or even handwriting of a human.

2.1.3 Wiring pi Library

Wiring pi library is a library that governs the communication via serial port between the raspberry pi and the Arduino

2.1.4 Raspicam Library

Raspicam library is a library that enable user to write codes to run raspicam and capture images and videos along with the ability to make a continuous video stream.

2.1.5 C++ language

C++ is a high programming language that gives the user a control over the hardware, its considered to be the most common programming language in the world. By using it user can build high performance applications. Its worthy to mention that CPP has emanating from old C programming language.

2.2 Coding process

In our codding process we followed divide and conquer algorithm that is a strategy of solving a large problem by breaking the problem into smaller sub-problems. solving the sub-problems, and. combining them to get the desired result.

2.2.1 Flashing Operating System

The first step in coding process was to flash an operating system on SD card then insert it in the raspberry to create an environment where we can download programs and libraries for coding process.

For raspberry we flashed a Raspbian which is a free operating system based on Debian optimized for the Raspberry Pi hardware. It set basic programs and utilities that make Raspberry Pi run. In addition, Raspbian gives more than a pure OS, it provides more than 35,000 packages, pre-compiled software bundled in a nice format for easy installation on your Raspberry Pi.

2.2.2 Download VNC player and IP scanner

After flashing Raspbian, we downloaded tow important software on a personal computer so we can remotely communicate with raspberry. IP scanner to detect the IP used by the raspberry to connect to a local Wi-Fi network, the same network where personal computer is connected, then we managed to reach raspberry via VNC viewer .and from this point the real work begins.

2.2.3 Install Open CV

Now we have to install Open CV and its libraries , in this step we will use command prompt for the installation process , at the beginning we updated and upgraded the raspberry pi the following cods :

```
sudo apt-get upgrade
```

```
sudo apt-get update
```

then we installed C make package config to build open CV that we are going to download and we used this code to download the package:

```
sudo apt-get install build-essential cmake pkg-config
```

then we cloned open CV from Github and build it so we can use its libraries in Geany by the following codes:

```
git clone https://github.com/opencv/opencv.git

git clone https://github.com/opencv/opencv_contrib.git

#Create build directory

mkdir build

cd build

make -j4

sudo make install
```

then we installed Raspicam library

installing raspicam:

```
git clone https://github.com/cedricve/raspicam.git

cd raspicam

mkdir build

cd build

cmake ..

make

sudo make install

sudo ldconfig
```

at the end of the day, we call all these installed libraries to Geany so we can finally start real coding.

2.2.4 Geany

Here at the beginning we just call the installed libraries and we detrmined the programming language just by saving the programming file with a (.pp) suffix .

2.2.5 Raspicam Initialization

We wrote the following code to initiate the camera so we can begin image processing and do the control program :

```
 #include <opencv2/opencv.hpp>

#include <raspicam_cv.h>

#include <iostream>


using namespace std;

using namespace cv;

using namespace raspicam;


Mat frame;


 void Setup ( int argc,char **argv, RaspiCam_Cv &Camera )

  {

    Camera.set ( CAP_PROP_FRAME_WIDTH,  ( "-w",argc,argv,400 ) );

    Camera.set ( CAP_PROP_FRAME_HEIGHT,  ( "-h",argc,argv,240 ) );

    Camera.set ( CAP_PROP_BRIGHTNESS, ( "-br",argc,argv,50 ) );

    Camera.set ( CAP_PROP_CONTRAST ,( "-co",argc,argv,50 ) );

    Camera.set ( CAP_PROP_SATURATION,  ( "-sa",argc,argv,50 ) );

    Camera.set ( CAP_PROP_GAIN,  ( "-g",argc,argv ,50 ) );

    Camera.set ( CAP_PROP_FPS,  ( "-fps",argc,argv,100));


}
```

```
int main(int argc,char **argv)
{

      RaspiCam_Cv Camera;
          Setup(argc, argv, Camera);
          cout<<"Connecting to camera"<<endl;
          if (!Camera.open())
          {

          cout<<"Failed to Connect"<<endl;
    }

    cout<<"Camera Id = "<<Camera.getId()<<endl;

       while(1)
   {
   auto start = std::chrono::system_clock::now();
   Camera.grab();
   Camera.retrieve( frame);
   auto end = std::chrono::system_clock::now();


   std::chrono::duration<double> elapsed_seconds = end-start;

   float t = elapsed_seconds.count();
   int FPS = 1/t;
```

```
cout<<"FPS = "<<FPS<<endl;
imshow("orignal", frame);

waitKey(1);
      }
  return 0;
 }
```

2.2.6 Create a region of interest

Region Of Interest (ROI) is the part of frame that we are going to make image processing operations on it. we made region of interest by drawing lines between a cerin point in the frame and we used the following code to accomplish the mission:

```
void perspectv()
{
   line(Fram, sourc [0],sourc[1], Scalar(0,0,255), 2);
   line(Fram, sourc [1], sourc [3], Scalar(0,0,255), 2);
   line(Fram,Source[3], sourc [2], Scalar(0,0,255), 2);
   line(Fram, sourc [2], sourc [0], Scalar(0,0,255), 2);
}
```

Where frame is Mat variable, we declared as a global variable at the head of the page and Source is an array where we saved the points that we drowned the region of interest depending on it.

For scalar it defines the size and cooler of drowned line (Red,Green,Blue).

Figure 0-1- region of interest

2.2.7 Perspective transformation

After we created our region of interest now we are going to make a new frame from the original one but the new one will have what we can call a bird eye view , so we used the following code :

```
Matrix = getPerspectiveTransform(sourc,destin);

   warpPerspective(Fram,framper, Matrix , Size(400,240));
```

at the first one we used a function called getPerspectiveTransform is

a transformation that maps a quadrilateral region from one plane to another plane. where we combined to images.

The getPerspectiveTransform function takes two sets of four points as input. The first set of points represents the coordinates of the quadrilateral in the source frame, and the second set of points represents the corresponding coordinates in the destination image, and we declared an array that contains the second set of points under the name of destination at the head of file.

The function returns a transformation matrix that represents the perspective transform between the two sets of points.

Then we used warpPerspective function to apply the perspective transform to a frame.

By providing the original frame, destination frame, transformation matrix, and the desired size of the output frame, the warpPerspective function applies the perspective transform and generates the transformed image.

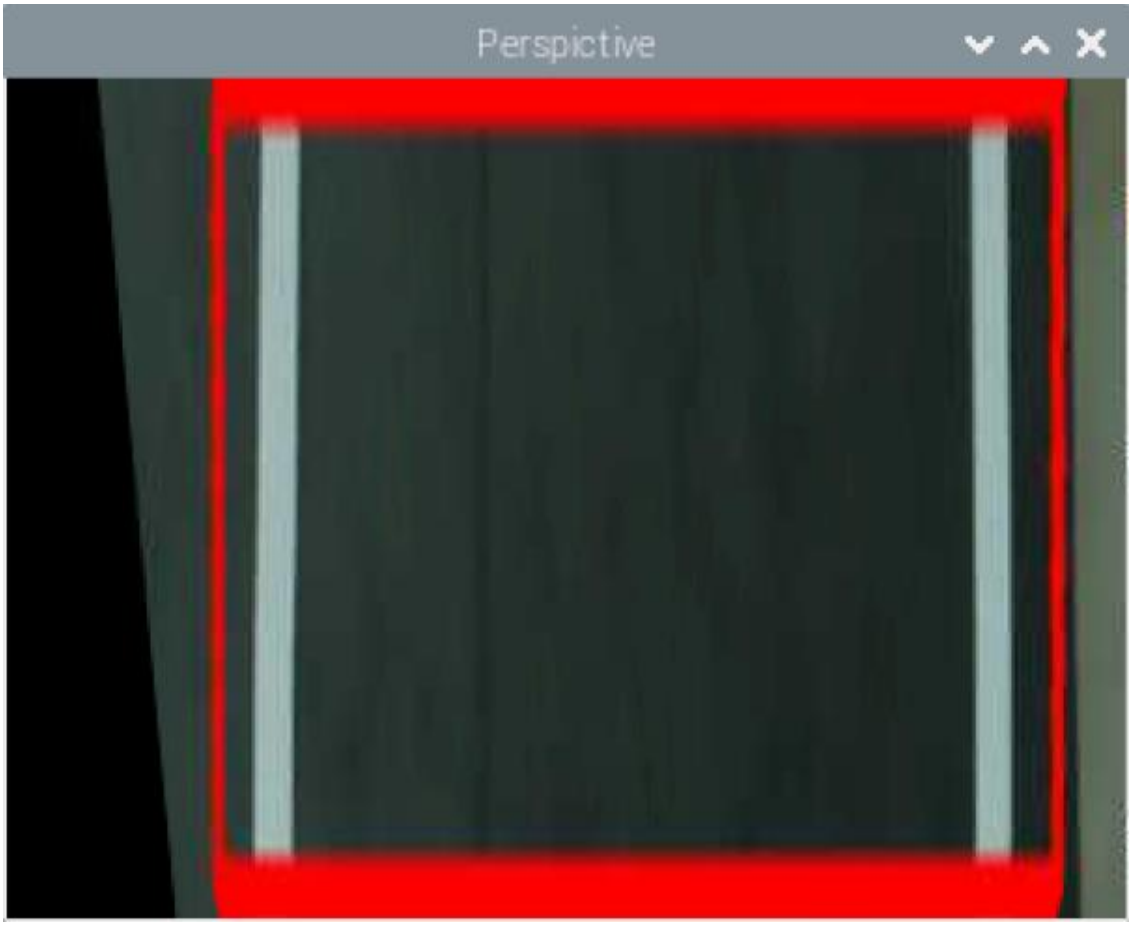


Figure 0-2-perspective transformation

2.2.8 Threshold and canny edge function

In this function we converted the RGB original frame to a grey scale new frame

in gray scale the blur white is considered to be 255 and the deep black is 0 , so in this function we want just to colors to be shown in the new frame so we created a range to what we will consider as white and what considered as deep black , this code put the first block in locating the track lanes.

Then we used canny edge detection which is an algorithm is for detecting edges of the white track lanes, finally we merged the tow fames (framthresh and framedge) and we made a new frame called final and here the code :

```
void thrshold()
{   cvtColor(FramePer,frameGray,COLOR_RGB2GRAY);
    inRange(Framgray,200,255,framThresh);
    Canny(Framgray,framedg, 100, 500, 3, false);
    add(framthresh, framdge, framfinal);
    cvtColor(framfinal,framfinal,COLOR_GRAY2RGB);
    cvtColor(framefinal,framefinalDuplicate,COLOR_RGB2BGR);
}
```

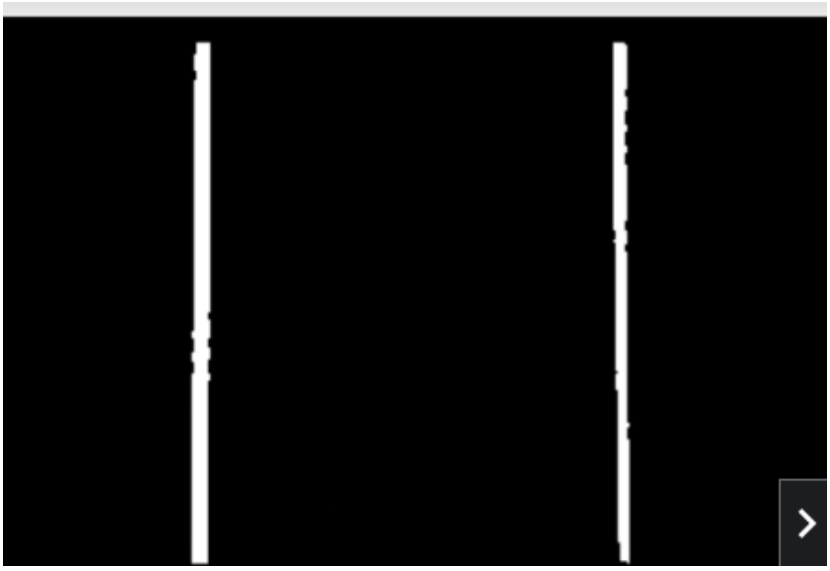


Figure 0-3-canny edge1

Figure 0-4- canny edge2

2.2.9 Track lanes positioning

Here we actually positioned the track lanes by the following method

First, we declared a dynamic array with a size equal to the width of the frame then we created a small region of inters which it is a rectangular that have 1 pixel width and then we passed the rectangular over the frame 400 times, and if the area covered with the rectangular is white it will store a value, so where ever the track planes in the region oo interest they will be located, now we push the values of small RIO into the dynamic array.

Then we divided the array into tow strips left and right, so we can Finally use an iterator which is a pointer to locate the position of maximum intensity in the dynamic array and this crossbones the track lane positions., left and right lanes

After we found the positions of the track lines, we draw a line over the positions of lanes

Code:

```
void Histo ()
{
   Hstogramlan.resize(400);
   Hstogramlan.clear();
   for (int i=0; i<400; i++)
   {
      ReOFIlan= FramefinalDuplicate (Rect(i,140,1,100));
      divide(255,ReOfIlan,ReOfIlan);
      HistogramLane.push_back((int)(sum(ReOfIlan)[0]));
   }
}
void lanfindr()
{
   vector<int>:: iterator Leftptr;
   Leftppinter= max_element(Hstogramlan.begin(), Hstogramlan.begin() + 150);
   LeftlanePosition= distance(histogramlan.begin(),leftpointer);

    vector<int>:: iterator rightpointer;
   rightpointer= max_element(histogramLane.begin()+ 250 , HstogramLane.end());
   rightlanposition= distance(Histogramlane.begin(),rightpointer);

   line(framfinal,Point2f(LeftLanePosition,0),Point2f(LeftLanePosition,240),
Scalar(0,255,0), 2);
   line(framfinal,Point2f(rightlaneposition,0),Point2f(rightlanePosition,240),
Scalar(0,255,0), 2);
}
```

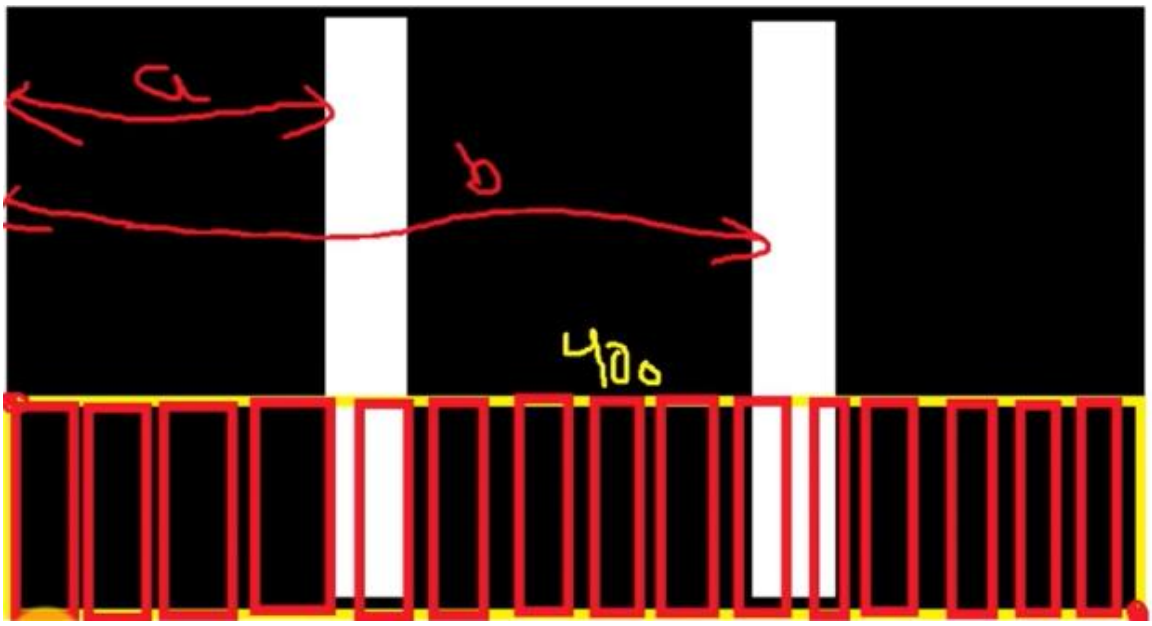


Figure 0-5 lanes positioning

2.2.10 Positioning lane and frame centers

Finally, we positioned the lane centers by a simple mathematic operation because we are already located the track lanes, then we found the frame center.

least but not last we calibrated the to lines to stand above each other when the car is really at the center of the track, then we defined a variable called offset which calculate the difference between the positions of lane center and frame center when the car diverges from the center of the track either to left or right, here the code:

```
void lancentr ()
{
   lanCntr = (rightlanePosition-leftlaneposition)/2 + leftlaneposition;
   framCntr = 176;
   line(Framfinal, Point2f(lanCntr,0),Point2f(lanCntr,240), Scalar(0,255,0),3);
   line(Framfinal, Point2f(framCntr,0),Point2f(framCntr,240), Scalar(255,0,0),3);

   offset= lanCntr - framCntr;
```

Figure 0-6-positioning lane and frame centers

2.2.11 Slave device programming (Arduino UNO)

In this part we are going to control the speed and direction of rotation of each dc motor connected to the Arduino.

We opened a serial port at 9600 rates so raspberry can communicate with Arduino

And then we created several functions to control the movement of vehicle either to stop or to move forward, move left or right, slightly or intensively according to the order it receive from raspberry it will perform a certain function and direct the motion of the vehicle , and this is the slave device code :

```
#include <AFMotor.h>

AF_DCMotor motor1(1);  //define motor 1 as m1
AF_DCMotor motor2(2);  //define motor 2 as m2
AF_DCMotor motor3(3);  //define motor 3 as m3
AF_DCMotor motor4(4);  //define motor 4 as m4

char feedback;

void setup() {
  Serial.begin(9600);

  motor1.setSpeed(0);
  motor2.setSpeed(0);
  motor3.setSpeed(0);
  motor4.setSpeed(0);
}

void backward() {
  motor1.run(FORWARD);
  motor1.setSpeed(255);
  motor2.run(FORWARD);
  motor2.setSpeed(255);
  motor3.run(FORWARD);
  motor3.setSpeed(255);
  motor4.run(FORWARD);
  motor4.setSpeed(255);
}

void forward() {
  motor1.run(BACKWARD);
  motor1.setSpeed(255);
  motor2.run(BACKWARD);
  motor2.setSpeed(255);
  motor3.run(BACKWARD);
```

```
 motor3.setSpeed(255);
  motor4.run(BACKWARD);
  motor4.setSpeed(255);
}


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

void right1 ()
{
  motor1.run(BACKWARD);
  motor1.setSpeed(255);
  motor2.run(BACKWARD);
  motor2.setSpeed(255);
  motor3.run(BACKWARD);
  motor3.setSpeed(120);
  motor4.run(BACKWARD);
  motor4.setSpeed(120);

}

void right2 ()
{
  motor1.run(BACKWARD);
  motor1.setSpeed(255);
  motor2.run(BACKWARD);
  motor2.setSpeed(255);
  motor3.run(BACKWARD);
  motor3.setSpeed(50);
  motor4.run(BACKWARD);
  motor4.setSpeed(50);

}

void right3 ()
{
  motor1.run(BACKWARD);
  motor1.setSpeed(255);
  motor2.run(BACKWARD);
  motor2.setSpeed(255);
```

```
  motor3.run(BACKWARD);
  motor3.setSpeed(0);
  motor4.run(BACKWARD);
  motor4.setSpeed(0);

}
void left1() {
  motor1.run(BACKWARD);
  motor1.setSpeed(120);
  motor2.run(BACKWARD);
  motor2.setSpeed(120);
  motor3.run(BACKWARD);
  motor3.setSpeed(255);
  motor4.run(BACKWARD);
  motor4.setSpeed(255);
}
void left2() {
  motor1.run(BACKWARD);
  motor1.setSpeed(50);
  motor2.run(BACKWARD);
  motor2.setSpeed(50);
  motor3.run(BACKWARD);
  motor3.setSpeed(255);
  motor4.run(BACKWARD);
 motor4.setSpeed(255);
}
void left3() {
  motor1.run(BACKWARD);
  motor1.setSpeed(0);
  motor2.run(BACKWARD);
  motor2.setSpeed(0);
  motor3.run(BACKWARD);
  motor3.setSpeed(255);
  motor4.run(BACKWARD);
  motor4.setSpeed(255);
}

void loop() {

if (Serial.available()) {
    // Read data from serial buffer
     feedback = Serial.read();
```

```
  }

if(feedback=='1')
{
forward();
}

else if(feedback=='2')
{
  right1();
}

else if(feedback=='3')
{
  right2();
}

else if(feedback=='4')
{
  right3();
}

else if(feedback=='5')
{
  left1();
}

else if(feedback=='6')
{
  left2();
}

else if(feedback=='7')
{
  left3();
}

else if(feedback=='8')
{
  backward();
}

else if(feedback=='9')
{
  stop();
```

```
}


}
```

2.2.12 Master Device communication

Finally after we have finished the programming of slave device and also we finished image process operation at the master device , now we are going get use of the image process operations and control the motion of the vehicle by making an if nested loop depending on the value of integer variable offset , each value of offset will send an order to the slave device which in turn will drive motors with the required parameters by the master so the vehicle will run over the track autonomously and keep itself at the middle of the track, full code will be listed in the appendix

2.3 HARDWARE DESIGN

2.3.1 Hardware schematics (1)

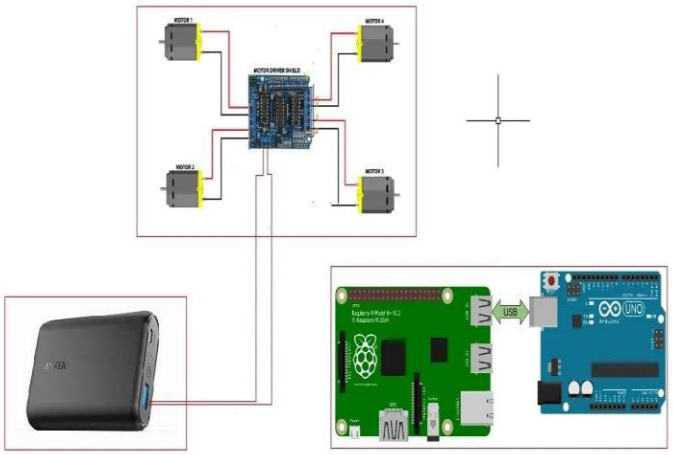
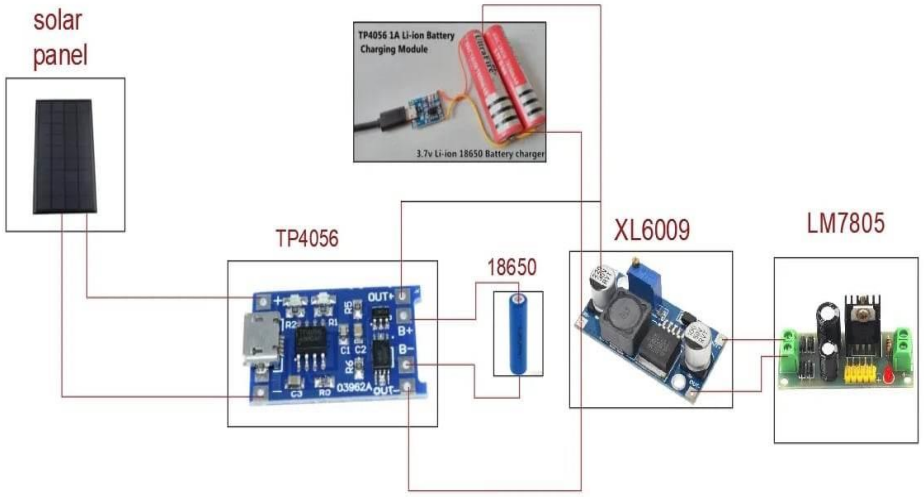


Figure 0-7-hardware 1

2.3.2 Hardware schematics (2)



After a long search, we found that there is a piece that replaces the two pieces XL6009 and LM7805 so we replaced them with another piece TPS61088 of high efficiency

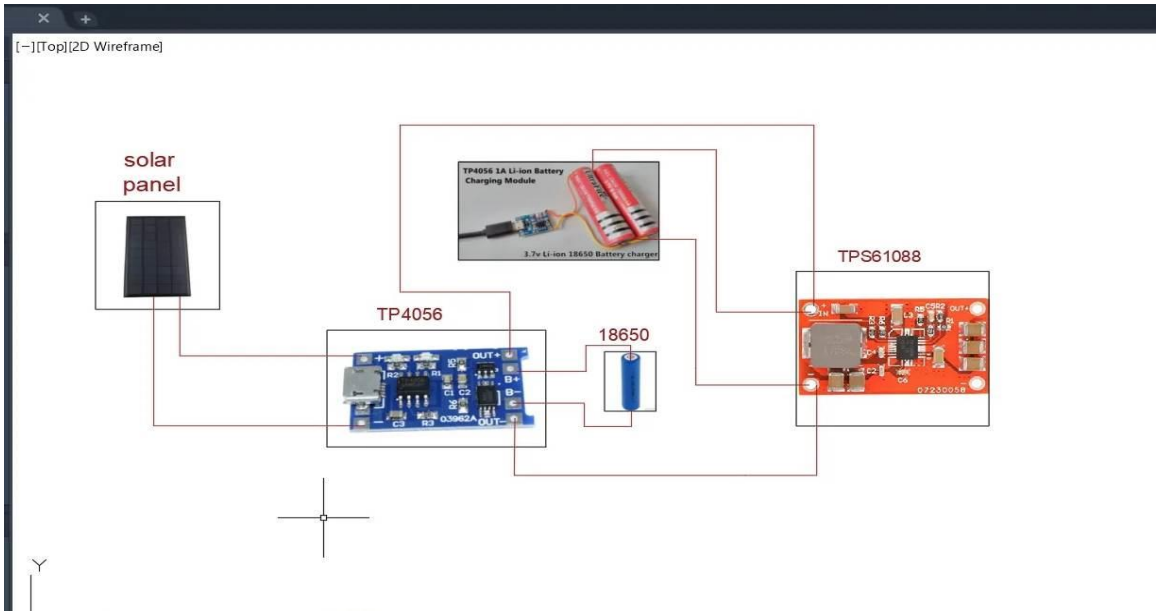


Figure 0-8-hardware 2

2.3.3 Hardware schematics (3)

Here is a picture from the field when we prepared the circuit and explained how the output is given 5.1 volt by a digital voltmeter.

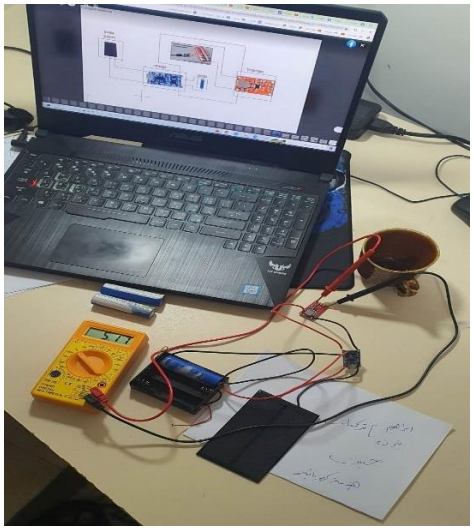


Figure 0-9-hardware 3

2.4 Electric elements used

2.4.1 Panel solar cell:

A mini solar panel, also known as a small-scale solar panel or portable solar panel, is a compact and lightweight photovoltaic (PV) device designed to convert sunlight into electricity. These panels are typically smaller in size compared to standard solar panels used in residential or commercial applications or factories. Mini solar panels are commonly used for charging small electronic devices, powering outdoor lights, or providing energy for camping and hiking trips.

The piece size 69*110mm give output(voltage)5v and output (current)250mili amper.

Figure 0-10-panel solar

2.4.2 TP4056 lion battery charging module

The TP4056 is a popular integrated circuit (IC) used for charging single-cell lithium-ion or lithium-polymer batteries. It is commonly used in small electronic devices, such as power banks, portable speakers, and DIY electronics projects.

The piece contains an entrance charge cable (mini USP) and an entrance input (positive and negative) and two port to connect battery and receive two ports (output).

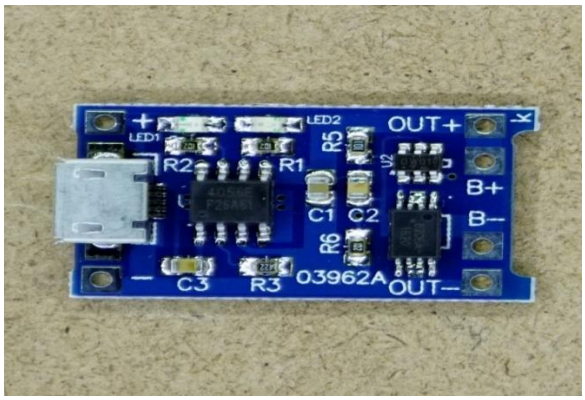


Figure 0-11-TP4056

2.4.3 L293D motor shield for arduino

The L293D is a popular integrated circuit (IC) commonly used for driving DC motors and controlling other inductive loads in various electronic projects. It provides a convenient way to interface microcontrollers or other control systems with motors, allowing you to control their speed and direction.

It contains entrances for motors, entrances for power, and a button to re-engage or rest the piece and contains analog pins to reception analog signal.

This piece can be connected to the Arduino and the way to connect it is by applying it directly to the Arduino as in the picture

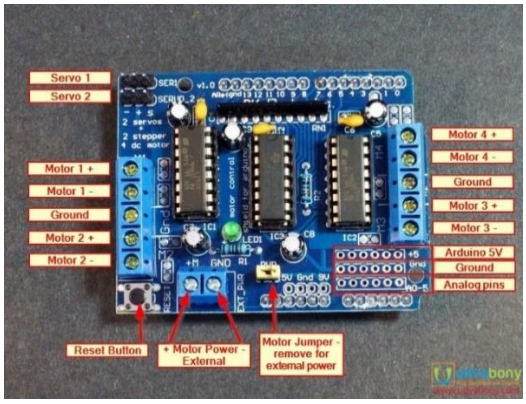


Figure 0-12-L293D

2.4.4 TPS61088 Boost Step Up Power From 2.7 To 5V 3A Supply Module

The TPS61088 is a high-efficiency, synchronous boost converter manufactured by Texas Instruments. It is designed to provide a regulated output voltage from a low-input voltage source, such as a single-cell lithium-ion (Li-ion) or lithium-polymer (Li-Po) battery. The TPS61088 offers several features that make it suitable for portable and battery-powered applications.

The piece can provide an adjustable voltage range from 1.8v to 5.5v and this voltage can be adjusted by using external resistors

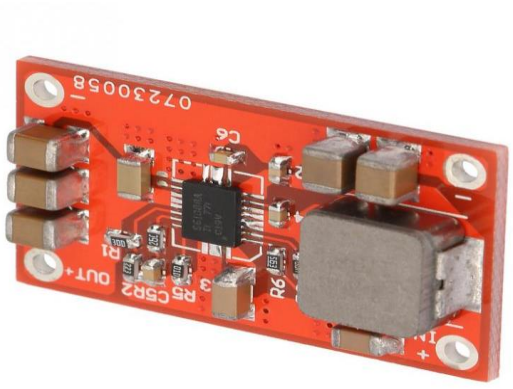


Figure 0-13-TPS61088

2.4.5 18650-lithium battery

he 18650-lithium battery is a common rechargeable battery format that has become widely used in various electronic devices.

the battery has a nominal voltage from 3.6v to 3.7v, And when fully charged the voltage can reach 4.2v and when discharging it can reach 3v.

With regard to the current carried by the battery, it ranges from 1500mA to 3500mA

Figure 0-14-18650-lithium

2.4.6 Raspberry Pi 4 Computer Model B

The Raspberry Pi 4 Model B is a popular single-board computer (SBC) developed by the Raspberry Pi Foundation. It is the fourth generation in the Raspberry Pi series and offers significant improvements over its predecessors in terms of performance and capabilities. It offers ground-breaking increases in processor speed, multimedia performance, memory, and connectivity compared to the prior-generation Raspberry Pi 3 Model B+, while retaining backwards compatibility and similar power consumption.

Overall, the Raspberry Pi 4 Model B is a versatile and powerful SBC suitable for a wide range of projects, including home automation, media centres, retro gaming consoles, robotics, and more. Its improved performance, enhanced connectivity, and expanded memory options make it a popular choice among hobbyists, educators, and professionals alike.

The recommended power supply for the Raspberry Pi 4 is 5V/3A (3000mA) to ensure sufficient power for stable operation. Additionally, the Raspberry Pi 4 Model B itself can consume a maximum of 1.2A of current for its own operation.

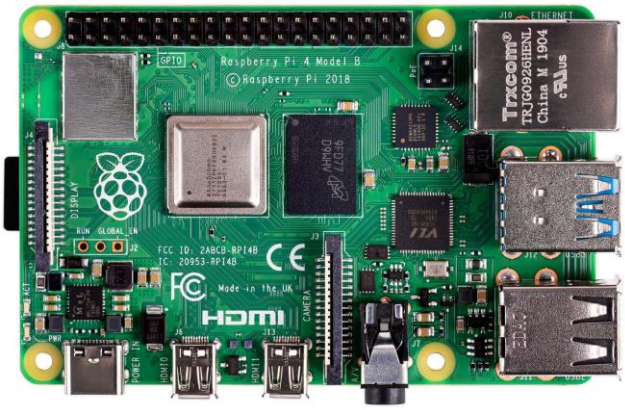


Figure 0-15-raspberry pi

2.4.7 Arduino Uno R3

The Arduino Uno is a popular microcontroller board that is widely used for various electronics projects one of its most important advantages is that it is an open source any student can access the data sheet and installation. It is based on the ATmega328P microcontroller and comes with built-in digital input/output pins, analog inputs, and other features that make it easy to interface with external components and sensors.

The Arduino Uno is powered by the ATmega328P microcontroller, which operates at a clock speed of 16 MHz It has 32KB of flash memory for storing the program, 2KB of SRAM, and 1KB of EEPROM for data storage.

Figure 0-16-Arduino uno

2.4.8 DC GEARED MOTOR

DC motors, also known as direct current motors, are devices that convert electrical energy into mechanical energy. They operate based on the principles of electromagnetism. These motors are widely used in various applications, including industrial machinery, automotive systems, robotics, and appliances.

The speed of a DC motor can be controlled by varying the applied voltage or by using electronic speed controllers (ESC) for brushless DC motors. By adjusting the voltage it usually used h_bridgr we used piece L293Dor controlling the switching pattern of the ESC, the motor's speed and direction can be precisely regulated.

Figure 0 17–dc motor

2.4.9 Car Chassis

In our project we used light weight reinforced chassis to reduce the weight of the vehicle and in the same time can handle and support the operation of vehicle.

Figure 03-3-6-1 DC motor Driver

Figure 0-18-car chassis

2.4.10 Raspberry Pi Camera Module

The Pi camera module is a portable light weight camera that supports Raspberry Pi. It communicates with Pi using the MIPI camera serial interface protocol. It is normally used in image processing, machine learning or in surveillance projects

Figure 0-19-camera module

3 CHAPTER 4. CONSTRAINTS, STANDARDS AND EARLIER COURSEWORK

3.1 CONSTRAINS

- On economy: the budget limitation was the biggest economical constrain, it was pretty hard to obtain the needed amount of money which was pretty big because our market is restricted to two names of suppliers.
- On society: this project counted by many as a way to reach deeply in people's daily life.
- On Manufacturability: the design implemented physically but not as it meant to be due to lack of hardware components.

3.2 STANDARDS

Mainly IEEE standards used in Palestinian projects, due to the lack of information in studying about standards and protocols, and if we wanted to move to industrial form in this design we will use this protocols: At first, IEEE 802.15.4s-2018 will be used, This standard defines a protocol and procedures for Definitions of MAC related functions to enable spectrum resource management are addressed in this amendment to IEEE Std

802.15.4™. It specifies the following: – Spectrum resource measurements and network performance metrics, such as packet error ratio, delay, etc. – Information elements and data structures to capture these measurements, – Procedures for collecting and exchanging spectrum resource measurement information with higher layers or other devices. [1]

IEEE 802.15.4 is a standard which specifies the physical layer and media access control for low-rate wireless personal area networks (LR-WPANs). It is maintained by the IEEE 802.15 working group. It is the basis for the ZigBee, ISA100.11a, Wireless HART, and MI WI specifications, each of which further extends the standard by developing the upper layers which are not defined in IEEE 802.15.4. Alternatively, it can be used with 6LoWPAN and standard Internet protocols to build a wireless embedded Internet.[2]

IEEE P1828 - "Standard for Systems with Virtual Components" [3]

3.3 EARLIER COURSEWORK

During our educational career, we have taken different courses, however, as we are on the verge of graduation, we can tell that some of these courses were helpful and essential while others not at all.

As our study is in English, we were able to make a decent report, and the English courses helped us to improve our capabilities and the ability to read and review external courses and communicate with foreign colleges.

We applied Electronics courses we took in small scale in our project by dealing with motors and drivers.

C language we learned in the university as other programming languages we learned on our own has helped us to write the code for our project.

in the future which can put our project in other level

4 CHAPTER 5. Result, Conclusions and Recommendation

4.1 Result:

We designed a simplified model of an electric car that works on solar energy, which is environmentally friendly, and we used self-driving cars, refers to the ability of a vehicle to operate and navigate without direct human input. It involves the use of advanced technologies such as cameras, and image processing techniques to perceive the environment, make decisions, and control the vehicle's movements. and we used dc motors to drive the vehicle, and we controlled these motors by raspberry pi and Arduino .

4.2 TOTAL CONCLUSION

In this workout the self-learning courses and courses which are unfamiliar to electrical engineering students (until now) had been applied, critical thinking methodology had been used to gain the optimum design within market and budget capabilities. Which revealed that we are hungry to new technologies and to adapt these technologies with each course we studied in our education interval.

The concept of car self-automation, often referred to as autonomous driving or self-driving cars, has been an area of significant technological development and research in recent years. While there have been significant advancements in this field, achieving full autonomy in cars is a complex and ongoing process that involves various technological, regulatory, and societal considerations automation.

Solar panels are used to convert sunlight into electricity, and they play a crucial role in the generation of clean and renewable energy.

Solar power is a renewable energy source that does not deplete natural resources like fossil fuels. By harnessing the sun's energy, we can generate electricity without contributing to greenhouse gas emissions or climate change.

Solar panels produce electricity without emitting pollutants or greenhouse gases. They have a much lower carbon footprint compared to traditional fossil fuel-based power generation. Using solar energy helps to reduce air pollution, combat climate change, and promote a more sustainable future.

The related sciences must be linked to each other in order to reach an integrated and developed model with the development of science.

4.3 SECURITY

This project actually is not fully secured one, since it's online on a free server with simple protection, without a cyber-security protocol, hacking the system and controlling it would be too easy for any one nearly.

n the beginning, we do not need a security system, but in the future, when we build our company and develop our product, we will think of a security system for our codes, programming, and circuits.

4.4 WHERE SUCH A PROJECT CAN BE USED

n the beginning, we will use the idea of this project for our own company, which we will build together, because the idea of our project is applied to every car that walks on wheels, then the world today needs self-driving because the world is going to be comfortable, and cars today have evolved to be environmentally friendly.

This project can be used for teaching in a private institute or industrial schools for industry students. It will be a quantum leap for them, and it will be a practical experience for them to keep pace with development and technology.

4.5 RECOMMENDATIONS:

I. We recommend using designed passive chips to imply the system with the lower possible cost, we suggest using a chip for each independent sub-system to achieve the final integrated system.
II. We recommend using more powerful cameras to cover all the road aspects and this can give the ability to jump the self-driving to another level
III. We recommend upgrade the code by implement a neural network.
IV. We recommend to make a main hub that collects data from clients who use our codes for future updates.
V. We recommend adding sensors to the BEV to monitor it and the conditions around it and it can improve the safety terms of the vehicle.

5 CHAPTER 6. REFERENCES

https://standards.ieee.org/ieee/802.15.4s/5970/ [1]

https://en.wikipedia.org/wiki/IEEE_802.15.4 [2]

https://www.postscapes.com/internet-of-things-protocols/ [3]

Addressing the range anxiety of battery electric vehicles with charging en route by Prabuddha Chakraborty, Robert Parker, Tamzidul Hoque, Jonathan Cruz, Lili Du, Shuo Wang & Swarup Bhunia . [4]

https://www.history.com/this-day-in-history/william-cobb-demonstrates-first-solar-powered-car [5]

Appendix:

```
#include <opencv2/opencv.hpp>
#include <raspicam_cv.h>
#include <iostream>
#include <chrono>
#include <ctime>
#include <wiringPi.h>
#include <wiringSerial.h>
#include <unistd.h>

using namespace std;
using namespace cv;
using namespace raspicam;

Mat Fram , Mtrx, framprs, framgry, framthrsh, framdg, framFinal,    framFinaldupl;
Mat ReOfInlan;
int leftlanposition, rightlaneposition, framcntr, lancntr, offst;

RaspiCam_Cv Camera;
```

```
stringstream ss;
vector<int> HstoLan;

Point2f sorc[] = {Point2f(35,170) ,Point2f(282,170) ,Point2f(15,200) ,Point2f(305,200) };
Point2f Dest[] = {Point2f(60,0) ,Point2f(300,0) ,Point2f(60,232) ,Point2f(300,232) };
int serial;
char A;
 void Setup ( int argc,char **argv, RaspiCam_Cv &Camera )
  {
    Camera.set ( CAP_PROP_FRAME_WIDTH,  ( "-w",argc,argv,400 ) );
    Camera.set ( CAP_PROP_FRAME_HEIGHT,  ( "-h",argc,argv,240 ) );
    Camera.set ( CAP_PROP_BRIGHTNESS, ( "-br",argc,argv,50 ) );
    Camera.set ( CAP_PROP_CONTRAST ,( "-co",argc,argv,50 ) );
    Camera.set ( CAP_PROP_SATURATION,  ( "-sa",argc,argv,50 ) );
    Camera.set ( CAP_PROP_GAIN,  ( "-g",argc,argv ,50 ) );
    Camera.set ( CAP_PROP_FPS,  ( "-fps",argc,argv,15));

}

void perspctv()
{
    line(Fram, sorc [0], sorc [1], Scalar(0,0,255), 2);
    line(Fram, sorc [1], sorc [3], Scalar(0,0,255), 2);
    line(Fram, sorc [3], sorc [2], Scalar(0,0,255), 2);
    line(Fram, sorc [2], sorc [0], Scalar(0,0,255), 2);
```

```
    Matrix = getPerspectiveTransform(sorc,Dest);
    warpPerspective(Fram,framprs, Matrix , Size(400,240));
}

void thrshld()
{
   cvtColor(framprs, framgry,COLOR_RGB2GRAY);
    inRange(framgry,200,255,framthrsh);
    Canny(framgry, framdg, 100, 500, 3, false);
    add(framthrsh, framdg, framFinal);
    cvtColor(frameFinal, framFinal,COLOR_GRAY2RGB);
    cvtColor(framFinal,framfinalDuplicate,COLOR_RGB2BGR);
}

void capture()
{
    Camera.grab();
    Camera.retrieve(Fram);

}
void Hsto()
{
    HstoLan.resize(400);
    HstoLan.clear();
    for (int i=0; i<400; i++)
    {
```

```
        ReOfInlan = framFinal Duplicate (Rect(i,140,1,100));
        divide(255, ReOfInlan, ReOfInlan);
        HstoLan.push_back((int)(sum(ReOfInlan)[0]));
    }
}
void lanfindr()
{
    vector<int>:: iterator leftpointeer;
    leftpointeer = max_element(HstoLan.begin(), HstoLan.begin() + 150);
    leftlanePosition= distance(HstoLan.begin(),leftpointeer);

     vector<int>:: iterator rightpointeer;
    rightpointeer= max_element(HstoLan.begin()+ 250 , HstoLan.end());
    rightlanPosition= distance(HstoLan.begin(),rightpointeer);

    line(framFinal,Point2f(leftlanePosition,0),Point2f(leftlanePosition,240),
Scalar(0,255,0), 2);
    line(framFinal,Point2f(rightlanPosition,0),Point2f(rightlanPosition,240),
Scalar(0,255,0), 2);
}

void lancntr ()
{
    lanCntr = (rightlanPosition - leftlanePosition)/2 + leftlanePosition;
    framCntr = 176;
    line(framFinal, Point2f(lanCntr,0),Point2f(lanCntr,240), Scalar(0,255,0),3);
    line(framFinal, Point2f(framCntr,0),Point2f(framCntr,240), Scalar(255,0,0),3);
```

```
    offset= lanCntr - framCntr;
}

int main(int argc,char **argv)
{

      serial = serialOpen("/dev/ttyUSB0", 9600);   // Open serial port with rate equal to arduino
  if (serial < 0) {
    std::cerr << "Error opening serial port." << std::endl;
    return 1;
  }

  if (wiringPiSetup() == -1) {  // Initialize WiringPi library
    std::cerr << "Unable to initialize WiringPi." << std::endl;
    return 1;
  }


          Setup(argc, argv, Camera);
          cout<<"Connecting to camera"<<endl;
          if (!Camera.open())
          {

          cout<<"Failed to Connect"<<endl;
     }
```

```
 cout<<"Camera Id = "<<Camera.getId()<<endl;



   while(1)
   {

   auto start = std::chrono::system_clock::now();

   capture();
   perspctv ();
   thrshld ();
   Hsto ();
   lanfindr ();
   lancntr ();
   if (offset ==0)
   {

   serialPutchar(serial, '1');  // Send data to Arduino
   std::cout << "Sent data: forward" << std::endl;
     waitKey(1);

}

  else if (offset >0 & offset <10)
   {
```

```
    serialPutchar(serial, '2');  // Send data to Arduino
    std::cout << "Sent data: right1" << std::endl;
      waitKey(1);

}
    else if (offset >=10 & offset <20)
    {

    serialPutchar(serial, '3');  // Send data to Arduino
    std::cout << "Sent data: right2" << std::endl;
      waitKey(1);

}
    else if (offset >20 )
    {

    serialPutchar(serial, '4');  // Send data to Arduino
    std::cout << "Sent data: right3" << std::endl;
      waitKey(1);

}
   else if (offset <0 & offset >-10)
    {

    serialPutchar(serial, '5');  // Send data to Arduino
    std::cout << "Sent data: left1" << std::endl;
```

```
        waitKey(1);


}
    else if (offset <=-10 & offset >-20)
    {

    serialPutchar(serial, '6');  // Send data to Arduino
    std::cout << "Sent data: left2" << std::endl;
      waitKey(1);

}
    else if (offset <-20)
    {

    serialPutchar(serial, '7');  // Send data to Arduino
    std::cout << "Sent data: left3" << std::endl;
      waitKey(1);

}
    ss.str(" ");
    ss.clear();
    ss<<"offset = "<<offset;
    putText(Fram, ss.str(), Point2f(1,50),0,1, Scalar(0,0,255),2);

    namedWindow("Orgnal",WINDOW_KEEPRATIO);
    moveWindow("Orgnal ", 0 , 20 );
```

```
    //resizeWindow("Orgnal ", 640 , 480 );
    imshow("Orgnal ", Fram);

     namedWindow("Prspctiv",WINDOW_KEEPRATIO);
    moveWindow("Prspctiv ", 330 , 20 );
    //resizeWindow("Prspctiv ", 640 , 480 );
    imshow("Prspctiv ", frampers);

    namedWindow("Finall",WINDOW_KEEPRATIO);
    moveWindow("Finall ", 330 , 200 );
    //resizeWindow("Gray", 640 , 480 );
    imshow("final", Finall);

    waitKey(1);
    auto end = std::chrono::system_clock::now();
    std::chrono::duration<double> elapsed_seconds = end-start;

    float t = elapsed_seconds.count();
    int FPS = 1/t;
    cout<<"FPS = "<<FPS<<endl;



    }

    return 0;
}
```