**BHARTI VIDYAPEETH DEEMED UNIVERSITY**

**AMPLIFY MINDWARE**



MINOR PROJECT-2

**A**

**MINOR-PROJECT REPORT**

**ON**

**RC-CAR**

**SUBJECT: IOT(Rc-Car)**

**SUBMITTED BY**

ANKIT SUTHAR

(BSIT/17/015)

Date:

Project Completion Certificate

This is to certify that the project report entitled as UBER TRIP ANALYSIS.

Submitted by:

Ankit Suthar

BSIT/17/015

Academic Year: 2018-2019

This bonafied work is carried out by the student under the supervision of MR. Bhaskar Anand and it is submitted towards the partial fulfillment of the requirement of BVDU-Amplify- DITM, Pune.

Internal Guide Course Coordinator Head of Department

Prof. Prashant Hinduja

Seal/Stamp Of College Principal

Place: Pune (BVDU-Amplify- DITM, Pune)

**ACKNOWLEDGEMENT**

In the accomplishment of this project successfully, many people have best owned upon me their blessing and the heart pledged support, this time I am utilizing to thank all people who have been concerned with this project.

Primarily I would like to thank god for being able to complete this project with success. Then I would like thank my teachers Mr. Bhaskar Anand and Mrs. Chanda for giving me the idea of this excellent topic and helping me through out the project. I would also like to thank my elder brother, whose valuable guidance has been the ones that helped me patch this project and make it full proof success. His suggestions and his instructions have served as the major contributor towards the completion of this project.

I greatly acknowledge all the support provided by my friends who helped me to accomplish this project, their valuable suggestions have been very helpful in various phases of the completion of the project. Last but not the least I would like to thank my family members for encouraging and supporting me throughout this Journey.

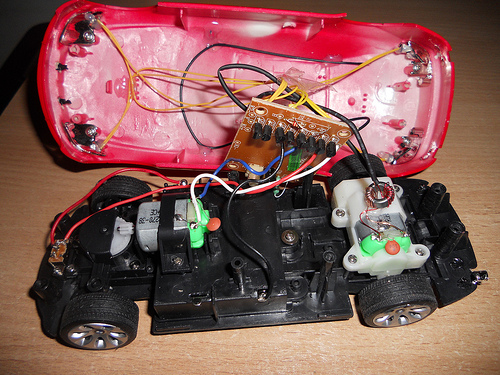
**INDEX**

Topic RC-CAR

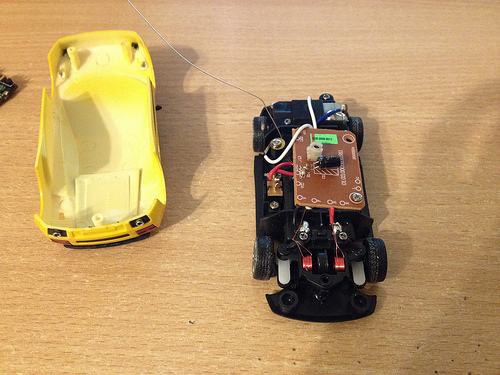
|  |  |  |
| --- | --- | --- |
|  | Contents  [1.SYSTEM ANALYSIS 5](#_Toc5800396)  [2.INTRODUCTION: 7](#_Toc5800397)  [2.1 Overview 7](#_Toc5800398)  [2.2 EXISTING SYSTEM 8](#_Toc5800399)  [**2.3** PROBLEMS WITH EXISTING SYSTEM 9](#_Toc5800400)  [**2.4** WORK PROPOSAL 9](#_Toc5800401)  [*3.SCOPE OF WORK* 10](#_Toc5800402)  [**3.1 OBJECTIVES** 10](#_Toc5800403)  [**3.2 TECNOLOGY AND ASSOCIATED PLATFORMS** 10](#_Toc5800404)  [*4.FEASIBILITY ANALYSIS* 11](#_Toc5800405)  [4.1 Objects 11](#_Toc5800406)  [**4.2 TYPES OF FEASIBILITY** 11](#_Toc5800407)  [*5.System Architecture/Design* 11](#_Toc5800408)  [**5.1 Introduction/model:-** 11](#_Toc5800409)  [**5.2 Diagram:** 12](#_Toc5800410)  [**5.3 UML/ERD/Process Flow/DFD’s** 16](#_Toc5800411)  [*6.Implementation* 18](#_Toc5800412)  [**6.1 Theoretical Implementation:->** 18](#_Toc5800413)  [**6.2 Actual work of contribution:-** 18](#_Toc5800414)  [**6.3 Screen shot/live system/Piece of code/Wireframes** 19](#_Toc5800415)  [7.Testing 26](#_Toc5800416)  [8.Conclusion (Key learnings) 28](#_Toc5800417)  [*9.Future scope* 30](#_Toc5800418)  [*10.References* 31](#_Toc5800419) |  |

## 1.SYSTEM ANALYSIS

The simple RC car usually contains one brushed motor for rear wheels driving and one bushed motor for its steering. The driving is limited to the full speed backward and full speed forward. The steering is limited as well, the motor can turn wheels to the left or to the right at the maximum angle, and there are no intermediate driving positions besides centre value. Also then the wheels are turned to some side the motor is stalled, as it keeps turning to the desired position but some gears prevent it from going in that direction – thus motor stalls and gets very warm.



In other cars, steering is controlled using magnets – each front wheel is connected to the magnet which attracts the centre knob of the steering mechanism then the voltage is applied to the steering magnet, so if we want to make a left turn, we apply voltage to the left magnet, and if we want to make a right turn, we apply voltage to the right magnet.



Some cars also has built-in lights, what’s strange that those lights are not controllable, but just light up than you go forward or backwards.

And finally the car contains some control board that accepts commands from the remote control pad and drives motors, lights, and steering magnets. This board consists of a receiver module and a h-bridge for driving motors (usually just a couple of transistors or FETs are used to make a h-bridge instead of a h-bridge chip) and some transistors for controlling steering magnets or lights.

So let’s build our own controlling board with additional features:

* Controllable by a smartphone instead of a remote controller.
* Several cars can be controlled by the same smartphone (not at once).
* Adjustable driving speed.
* Individually controllable lights (if available).
* Adjustable steering (if possible depending on steering mechanism setup).

## 2.INTRODUCTION:

### 2.1 Overview

The central piece of hardware that controls all the others is a Raspberry Pi 2 Model B+. The original aim of the Raspberry Pi project (by the UK-based Raspberry Pi Foundation) was to create a small, credit card-sized computer to help teaching children the basics of computer science. The module is well equipped with different kinds of input/output ports and it performs quite well in tasks that are related to this thesis project.

The Raspberry Pi model 2 B+ has the following features:

• 4 USB ports

• 40 GPIO pins

• Full HDMI port

• Ethernet port

• Combined 3.5mm audio jack and composite video

• Camera interface (CSI)

• Display interface (DSI)

• Micro SD card slot

• VideoCore IV 3D graphics core (1) Hardware capabilities vary by models. The Raspberry Pi Foundation developed a few Linux distributions for the Raspberry Pi. It was designed so that the end-user would use Python as the main scripting language, however several others, including C, C++, Java, Ruby, Perl, Smalltalk are also supported.

In this project, we're using python scripts run on a Raspberry Pi to set GPIO outputs to an L293D motor controller IC and run a DC motor in either direction at any speed.

First things first; a Raspberry Pi is an open-source credit card sized computer with 40 open GPIO pins. GPIO stands for "General Purpose Input/Output", which means these pins can either send electrical signals to drive hardware or receive them and read sensor data. We're using them as outputs, to send signals to the L293D IC Chip, which is just a chip used to control DC motors. Nothing special.

Python is a computer programming language, comparable to Javascript or C++. We'll be using very simple python commands, and no prior computer programming knowledge will be necessary.

The way we'll control the speed of the motor is by using a python module called PWM. That stands for Pulse Width Modulation. What PWM means is just controlling the amount of time a voltage is on by flipping between high and low for a set amount of time. The amount of time the voltage is high is called the 'duty' or 'duty cycle', and whatever percentage that is will be the percentage of power the motor runs on.

To see how PWM looks as an output, refer to the diagram above.

The L293D motor IC uses two pins referred to as inputs to sense the desired direction of the output, and another pin called Enable to sense On/Off. So, in our code, with the Enable pin On, if we want the motor to spin forward, we'll set input 1 to 'True' or 'HIGH', and input 2 to 'False' or 'LOW'. And if we want it to spin backwards, we'll set input 1 to 'False" or 'LOW' and input 2 to 'True' or "HIGH'. If both inputs are True or both are False, the motor will not run.

That's how we'll control the direction, but what about speed? We talked about PWM right? So we'll just Pulse Width Modulate both inputs, right? We could, but it would be too complicated. Since the IC has an Enable pin that controls its On/Off state, we can leave both inputs set to run and just modulate the Enable pin, and the IC will only put out power according to the duty we set in the Enable pin. That way we keep the code simpler, and less things can go wrong.

So, on the Raspberry Pi, we'll be using 3 GPIO output pins on the GPIO board, one 3.3V power supply, and one grounding pin. 2 of the output pins will be for the inputs on the IC, and one will be for the Enable.

### 2.2 EXISTING SYSTEM

In the Existing System, car is not so powerful to operate on every type of terrain. The motor power is not efficient to drive the car.

It is unable to work in all meteorological project and unable to gather all information accurately and precisely

### **2.3 PROBLEMS WITH EXISTING SYSTEM**

* Lack of security of data
* Motor power
* Battery
* Driver motor is not powerful

### **2.4 WORK PROPOSAL**

PROPOSED SYSTEM

The aim of proposed system is to develop a system of improved facilities. The proposed system can overcome all the limitations of the existing system.

ADVANTAGES OF THE PROPOSED SYSTEM

The system is very simple and easy to implement. The system requires very low system resources and the system will work in almost all configurations. It has got following features:

* Security of data
* Reduce the damage of the machines
* Better service
* User friendliness and interactive
* Minimum the required
* android based system

## *3.SCOPE OF WORK*

### **3.1 OBJECTIVES**

This project has 3 stages for its completion:

* To reduce time for the organization
* To increase efficiency and accuracy of the system
* To reduce pressure on the labour and reliving man power from repetitive and dull job
* To make retrieval of information faster
* To make the system more feasible
* To reduce large amount of paper work
* To make the system more reliable to avoid any ambiguity

To make the system

### **3.2 TECNOLOGY AND ASSOCIATED PLATFORMS**

HARDWARE REQUIREMENTS

Processor : 1.4GHz 64-bit quad-core ARMv8 CPU.

RAM : 1 GB

SDcard : 16 GB

Wi-Fi : 802.11n dongle to connect to Pi remotely

SOFTWARE REQUIREMENTS

Operating System : Raspbian OS

Back end : Python 3

## *4.FEASIBILITY ANALYSIS*

### 4.1 Objects

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spend on it. Feasibility study lets the developer foresee the future of the project and the usefulness.

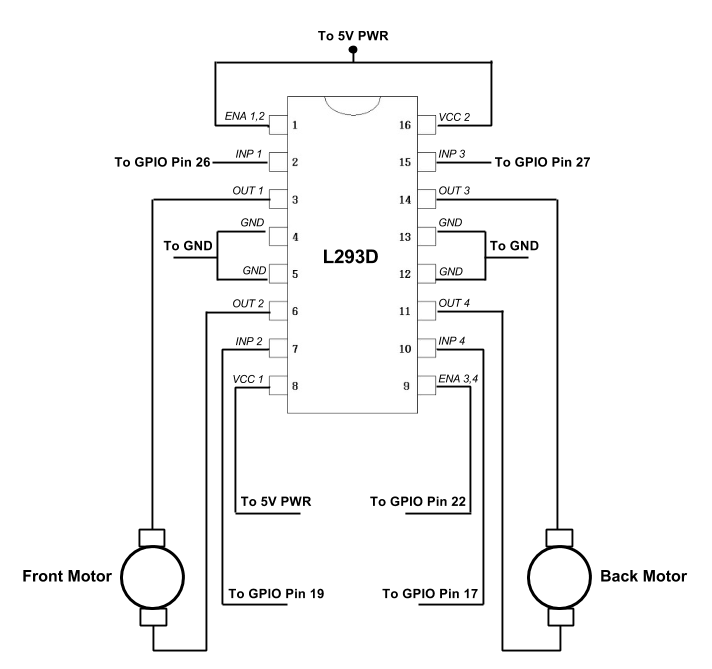
**4.2 TYPES OF FEASIBILITY :**

1. **TECHNICAL FEASIBILITY** : The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, desing.
2. **ECONOMIC FEASIBILITY** : The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest.
3. **BEHAVIROL FEASIBILTY** : The behavioral aspects are considered carefully and conclude that the project is behaviourally feasible.

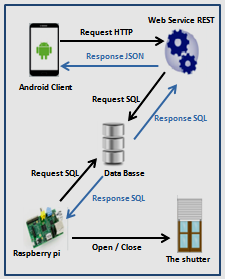
## *5.System Architecture/Design*

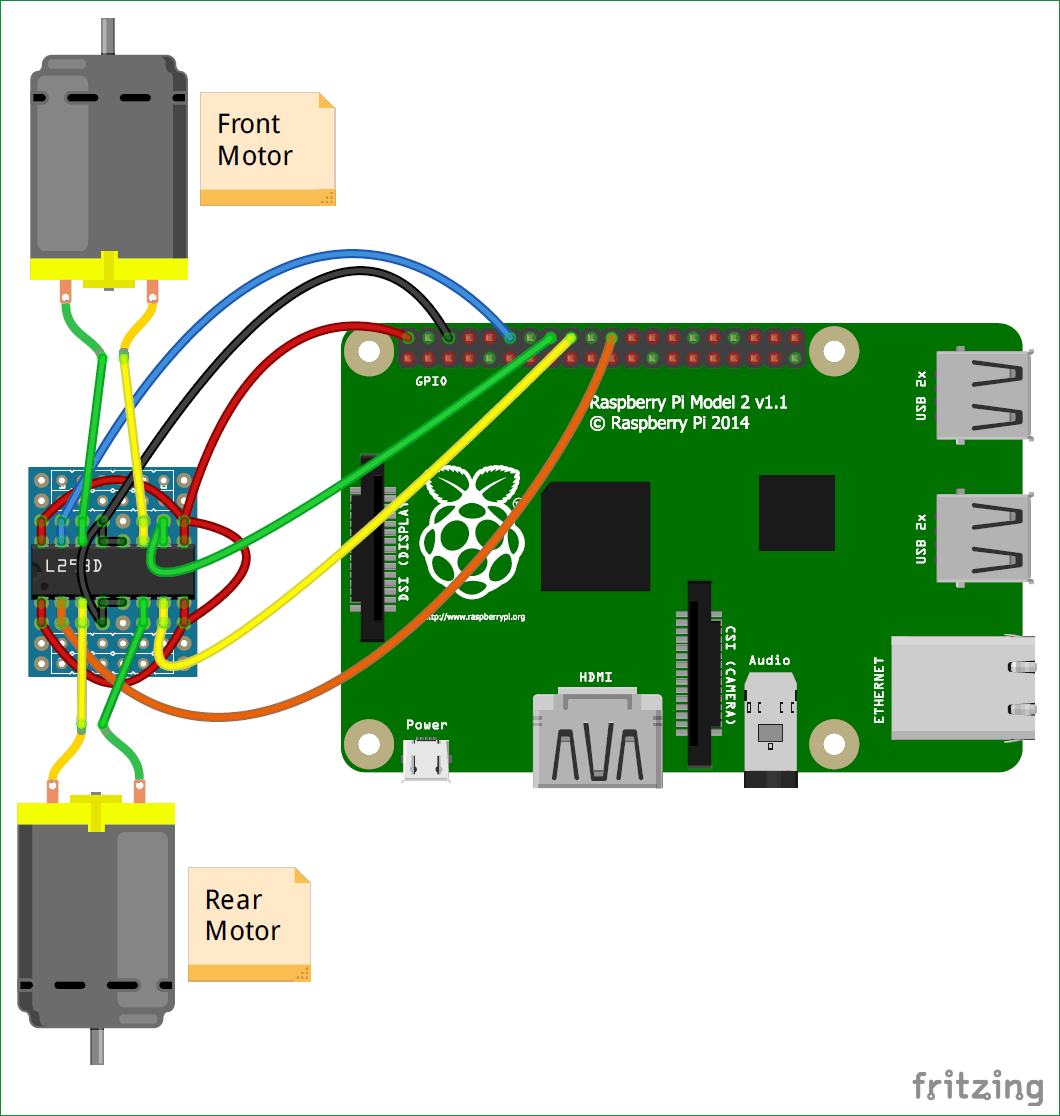
|  |  |
| --- | --- |
|  | **5.1 Introduction/model:-** |

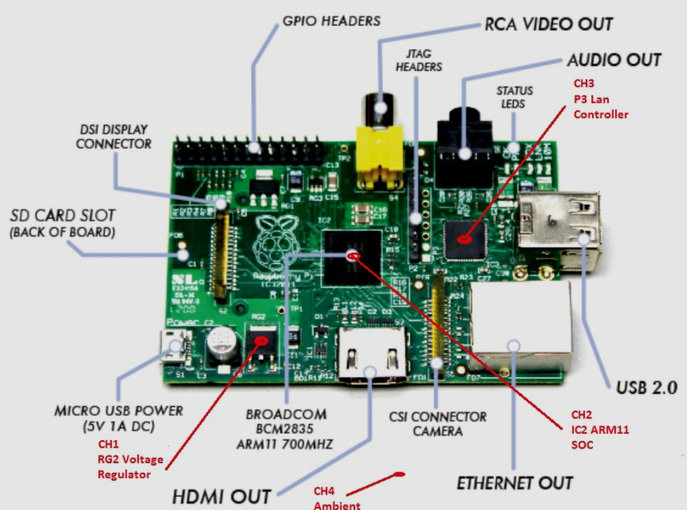
Our initial approach was to use the Raspberry Pi as a radio transmitter by figuring out the frequencies for the RC car using [**pi-rc**](about:blank) and replace the remote control. We adopted this approach, and things were progressing well until we developed slightly more complicated circuits. The car would be too fast around the turns, and the natural approach to take would be to reduce the speed around the corners. We would have had to make modifications to control the DC motors of the car or get a geared remote controlled car. We went ahead with the first approach!

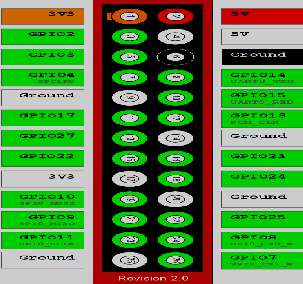
We will be referring the DC motor controlling the left/right direction as the front motor and the motor controlling the forward/reverse direction as the back motor. An L293D Motor Driver IC will be used to control the motors.

**5.2 Diagram:**->

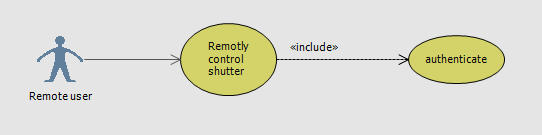


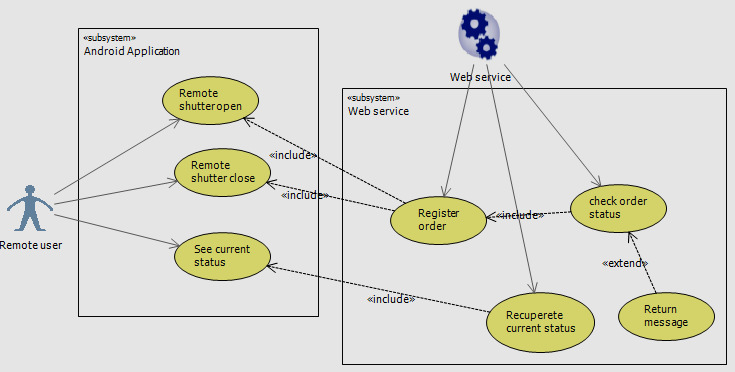


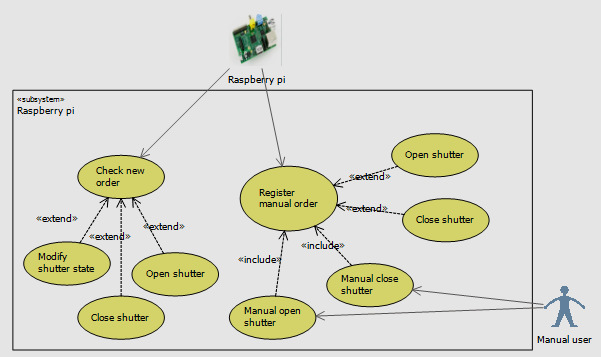
****

****

### **5.3 UML/ERD/Process Flow/DFD’s**



******

******

## *6.Implementation*

### **6.1 Theoretical Implementation:->**

The 2 wheels of the chassis are connected to 2 separate motors. The motor driver IC L293D is capable of driving 2 motors simultaneously. The rotation of the wheels is synchronized on the basis of the sides i.e. the left front and left back wheels rotate in sync and right front and right backwheels rotate in sync. Thus the pair of motors on each side is given the samedigital input from L293D at any moment. This helps the car in forward, backward movements when both side wheels rotate in same direction with same speed. The car turns when the left side wheels rotate in opposite direction to those in right [22]. The chassis has two shelves over the wheels separated by 2 inch approx. The IC is fixed on the lower shelf with the help of two 0.5 inch screws. It is permanently connected to the motor wires and necessary jumper wires are drawn from L293D to connect to Raspberry Pi []. The rest of the space on the lower shelf is taken by 8 AA batteries which provide the power to run the motors. To control the motor connected to pin 3 (O1), pin 6 (O2), the pins used are pin 1, pin 2 and pin 7 which are connected to the GPIOs of Raspberry pi via jumper wires[].

|  |  |  |
| --- | --- | --- |
| **Pin 1** | **Pin 2** | **Function** |
| High | **L**ow | Clockwise |
|  |  |  |

### **6.2 Actual work of contribution:-**

The work could be enhanced by improving the algorithm by adding machine learning to it. The present algorithm performs the operations on all the frames. It is accurate but its efficiency could be further enhanced if it starts learning by itself and avoid unnecessary calculations of the regions which are already known or familiar. Once the car starts travelling on the roads, it determines the obstacles (mainly static) on the way and note their characteristic features. An XML file is generated every time the car travels. It stores the following information:

• The distance between the two nodes.

• The number of roads diverging from a particular node.

• The number of speed breakers and other static obstacles on the road joining two nodes.

• The distance of speed breakers and other static obstacles from a specific node.

• Height and the width of an obstacle. The information stored in the XML helps the car understand and remember the path being followed for the next time.

### **6.3 Screen shot/live system/Piece of code/Wireframes**

**MainActivity->**

public class MainActivity extends AppCompatActivity {  
  
 Button up, down, left, right;  
  
 @SuppressLint("ClickableViewAccessibility")  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_main);  
 up = (Button) findViewById(R.id.forward);  
 down = (Button) findViewById(R.id.backward);  
 left = (Button) findViewById(R.id.left);  
 right = (Button) findViewById(R.id.right);  
  
 up.setOnTouchListener(new RepeatListener(50, 50, new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 sendRequest("forward");  
 }  
 }));  
 down.setOnTouchListener(new RepeatListener(50, 50, new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 sendRequest("backward");  
 }  
 }));  
  
 left.setOnTouchListener(new RepeatListener(50, 50, new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 sendRequest("left");  
 }  
 }));  
  
 right.setOnTouchListener(new RepeatListener(50, 50, new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 sendRequest("right");  
 }  
 }));  
 }  
  
 @SuppressLint("StaticFieldLeak")

private void sendRequest(String command) {  
 new AsyncTask<String,Void,Void>(){  
 @Override  
 protected Void doInBackground(String... strings) {  
 try {  
 URL = new URL("http://192.168.43.186:5000/"+strings[0]);  
 HttpURLConnection conn = (HttpURLConnection) url.openConnection();  
 if(conn.getResponseCode() == HttpURLConnection.HTTP\_OK){  
 Log.e("Sent","Signal");  
 }  
 } catch (MalformedURLException e) {  
 e.printStackTrace();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 return null;  
 }  
 }.execute(command);  
  
}

}

**Main\_activity.xml->**

<Linear Layout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerInParent="true"  
 android:gravity="center"  
 android:orientation="vertical">  
  
 <Button  
 android:id="@+id/forward"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:gravity="center"  
 android:text="Forward" />  
  
 <LinearLayout  
 android:layout\_width="match\_parent”

android:gravity="center"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal">  
  
 <Button  
 android:id="@+id/left"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Left" />  
  
 <Button  
 android:id="@+id/right"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Right" />  
  
  
 </LinearLayout>  
  
 <Button  
 android:id="@+id/backward"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="Backward" />  
 </LinearLayout>  
</RelativeLayout>

**Android Manifest->**

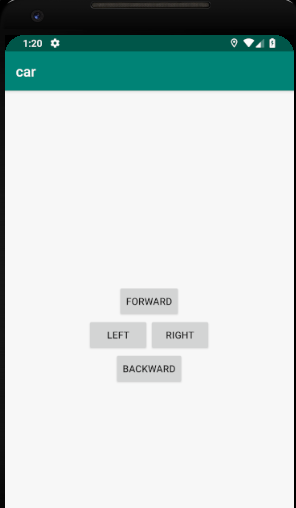
<uses-permission android: name="android.permission.INTERNET"/>

**RepeatListener->**

public class RepeatListener implements OnTouchListener {  
  
 private Handler = new Handler();  
  
 private int initialInterval;  
 private final int normalInterval;  
 private final View.OnClickListener clickListener;  
 private View touchedView;  
  
 private Runnable handlerRunnable = new Runnable() {  
 @Override  
 public void run() {  
 if(touchedView.isEnabled()) {  
 handler.postDelayed(this, normalInterval);  
 clickListener.onClick(touchedView);  
 } else {  
 // if the view was disabled by the clickListener, remove the callback  
 handler.removeCallbacks(handlerRunnable);  
 touchedView.setPressed(false);  
 touchedView = null;  
 }  
 }  
 };  
  
 */\*\*  
 \* @param initialInterval The interval after first click event  
 \* @param normalInterval The interval after second and subsequent click  
 \* events  
 \* @param clickListener The OnClickListener, that will be called  
 \* periodically  
 \*/* public RepeatListener(int initialInterval, int normalInterval,  
 View.OnClickListener clickListener) {  
 if (clickListener == null)  
 throw new IllegalArgumentException("null runnable");  
 if (initialInterval < 0 || normalInterval < 0)  
 throw new IllegalArgumentException("negative interval");  
  
 this.initialInterval = initialInterval;  
 this.normalInterval = normalInterval;  
 this.clickListener = clickListener;  
 }  
  
 public boolean onTouch(View view, MotionEvent motionEvent) {  
 switch (motionEvent.getAction()) {  
 case MotionEvent.ACTION\_DOWN:  
 handler.removeCallbacks(handlerRunnable);  
 handler.postDelayed(handlerRunnable, initialInterval);  
 touchedView = view;  
 touchedView.setPressed(true);  
 clickListener.onClick(view);  
 return true;  
 case MotionEvent.ACTION\_UP:  
 case MotionEvent.ACTION\_CANCEL:  
 handler.removeCallbacks(handlerRunnable);  
 touchedView.setPressed(false);  
 touchedView = null;  
 return true;  
 }  
  
 return false;  
 }  
  
}

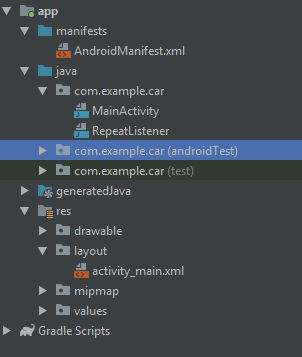
## *7.Testing*

***App test->***

****

## Test types and location->

Located at module-name/src/test/java/.

******

Alternatively, you can create a generic Java file in the appropriate test source set as follows:

1. In the **Project** window on the left, click the drop-down menu and select the **Project** view.
2. Expand the appropriate module folder and the nested **src** folder. To add a local unit test, expand the **test**folder and the nested **java** folder; to add an instrumented test, expand the **androidTest** folder and the nested **java** folder.
3. Right-click on the Java package directory and select **New > Java Class**.
4. Name the file and then click **OK**.

Also be sure you specify the test library dependencies in your app module's build.gradle file:

dependencies {  
    // Required for local unit tests (JUnit 4 framework)  
    testImplementation 'junit:junit:4.12'  
  
    // Required for instrumented tests  
    androidTestImplementation 'com.android.support:support-annotations:24.0.0'  
    androidTestImplementation 'com.android.support.test:runner:0.5'  
}

## 8.Conclusion (Key learnings)

When I started this project, I had a quite brief idea how what would be implemented. The main main for me was to use and practice my skills in using those technologies involved. I have learned a lot about working with Java and Android. I have also gained some experience in telecommunication. It is quite certain that I did not manage to do everything perfectly, I have still developed a lot. Fortunately, these mistakes were not deadly, but they rather had a constructive effect. All in all, I am proud of my work and all the efforts that I invested into it.

The key points where this project could be mainly improved:

1. The communication protocol between the RPi and the client platform could be more platform independent. At this point data is (de)serialized into Java objects. One nicer solution could http/soap messages.

2. The server prorgram language is Java. Java is a platform idependent language per se. However, not all microcontrollers support running a Java Runtime Environment. That being said it would not be possible to run the same code eg. on an raspberry for example.

3. Running the client application is also quite strictly bound to Android. It was written directly with Android Studio, which means that this code is hard to port into other platforms, such as Apple iOS, or Windows (Either with PC or Mobile). A solution to this could have been to implement the application using a platform independent developing environment. Xamarin could be a way to go.

4. Of course, the existing functionalities could be improved by more sensors and functionalities, such as an infrared lamp for night vision, or by security mechanisms, such as to automatically stop when an object is too close ahead, horn. Only fantasy defines the limit.

5. A better user interface is also optional, for example introducing a turn-bytilting, joystick/wheel support, or even more, a virtual reality support. Although it would be quite an expensive hardware for a thesis project at this point, it would probably increase the user experience a lot.

6. The usage of such hardware that allows a smoother acceleration (a Pulsewidth modulation).

. 8. It might also be a huge improvement to control the car via the Internet. In that way the car would not have to be required to be in the same network.

## *9.Future scope*

Better and better batteries for electric. I think now that performance has kinda tapered off, more will try to couple scale looks with that performance. Might be another hit out there like SCT's.   
  
As for Nitro, I see it going into the fourstroke era before dying off. Wont be long and ANYTHING two stroke will be banned in Cali and europe, so it's only a matter of time before viable alternatives are found if nitro will survive.  
  
Better and better electronics for 1/8th scale may take the classes to new levels. It's already booming here and the electronics out now are far from reliable.

## *10.References*

1. T. Ball, “Train your own opencv haar classifier,” 2013. [Online]. Available: [http://coding-robin.de/2013/07/22/train-your-own-opencv-haar-classifier.html](about:blank)

2. DARPA, “The darpa grand challenge,” 2014. [Online]. Available: [https://www.darpa.mil/news-events/2014-03-13](about:blank)

3. A. Deshpande, “The 9 deep learning papers you need to know about,” 2016. [Online]. Available: [https://adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html](about:blank)

4. L. Earnest, “Stanford cart,” 2012. [Online]. Available: [https://web.stanford.edu/~learnest/cart.htm](about:blank)

5. M. J. Focazio, “Self-driving cars could account for 75% of all vehicles on the roads by 2040,” 2012. [Online]. Available: [http://co.water.usgs.gov/trace/pubs/wrir-99-4279/](about:blank)

6. N. GHABOOSI, “Visualizing outputs cnn model training phase,” 2017. [Online]. Available: [http://www.ir.com/blog/visualizing-outputs-cnn-model-training-phase](about:blank)

7. S. Guennouni, A. Ahaitouf, and A. Mansouri, “A comparative study of multiple object detection using haar-like feature selection and local binary patterns in several platforms,” Modelling and Simulation in Engineering, vol. 2015, p. 17, 2015.

8. S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, “Detection of traffic signs in real-world images: The german traffic sign detection benchmark,” in Neural Networks (IJCNN), The 2013 International Joint Conference on. IEEE, 2013, pp