**CHAPTER 1: INTRODCUTION**

**1.1 BACKGROUND:**

The remote controlled toy cars were available as early in the mid-1960s produced by Italian and British manufacturers. Each car consists of a transmitter and a receiver. The transmitter is based on a controlling device such as joystick, which sends radio signals to the receiver. The reviver process the radio signals through a chip which controls the motor connected to the wheels. All the components are powered by electrical sources such as lithium batteries, DC batteries. The remote controlled car market is lucrative one with billions of revenues generated from sales. The concept of remotely controlled vehicle is not limited to the toy business, as it has far more important significances in other sectors such as robots, military, space-exploration, mining and much more.

The military uses remote controlled vehicle such as drones to navigate through the war zones, and the remotely controlled robots to defuse the bombs. The space agency uses robots built upon these vehicles to explore the lands of outer space.

The use of remotely controlled vehicles has allowed to explore the dangerous areas without risking the any human lives. Our project is to build a simple remotely controlled car as an step forward to learn about implementation of hardware and software in a single framework.

**1.2 OBJECTIVES:**

1. To build a remote controlled car

2. To control the car with the help of X-BOX Controller.

3. To design a software to process the instructions

4. To detect the objects that come on the path of car and avoid them

**1.3 SIGNIFICANCES:**

The car can be used to explore the hazardous environment without any necessity to send the human manpower on the site. This kind of system can also be expanded into a drone system, which can explore the land aerially. This system can be equipped with a video camera to continuously live stream the image of the area it explores, and also with thermal sensors to detect any sort of radiation anomaly.

**CHAPTER 2: EMBEDDED SYSTEM DEVELOPMENT**

**2.1 Requirements Engineering:**

The following are the requirements of the system

* Input : Movement input from Xbox Controller, ID from RFID
* Output: Motors direction, Livestream video
* Processing: Controller Processing, Video Processing, Distance Processing
* Input Supply: 9V for motors, 5V for Raspberry Pi
* Size: Small Scale
* Operating Environment: Flat terrain surface

**2.2 Hardware-Software Partitioning:**

**2.2.1 Hardware Specifications:**

**1. Raspberry Pi**

Raspberry Pi is used as the brain of the project, processing the video from the stereoscopic camera and live streaming to the smartphone intact in Google Cardboard. It also the controls the wheels of the car by processing the input from the Xbox controller. In addition, the pi is also responsible for processing the movements of the VR set to correspond the movement of the camera mounted in front of the car. The car is provided with two Raspberry Pis for faster processing. Two Pis have divided tasks, such that all the components work smoothly in simultaneously manner. Both Raspberry Pi are powered by 5V battery.

**2. Webcam**

A webcam is avideo camera that connects to a [computer](https://simple.wikipedia.org/wiki/Computer), and can let people see each other over the [Internet](https://simple.wikipedia.org/wiki/Internet) or without the internet. Most people that have webcams use them with an [instant messenger](https://simple.wikipedia.org/wiki/Instant_messenger) to see each other at the same time. Webcams can also be used for recording [videos](https://simple.wikipedia.org/wiki/Video) and [video blogs](https://simple.wikipedia.org/wiki/Blog). The webcam can be part of a computer, [mobile phone](https://simple.wikipedia.org/wiki/Mobile_phone) or it can be an independent device.

**3. XBOX Controller**

A wireless XBOX controller is used to control the directions and movements of the car. A chip XBOX receiver is connected to the Raspberry Pi, to connect the controller to the system. When the buttons and joystick are used, controller sends signals to the receiver, which in its part sends received signal to Pi for processing.

**4. H-BRIDGE**

L298N H-bridge Dual Motor Controller Module 2A with Raspberry Pi. This allows you to control the speed and direction of two DC motors, or control one bipolar stepper motor with ease. The L298N H-bridge module can be used with motors that have a voltage of between 5 and 35V DC.

**5. Motor**

We have used two 6V 250 Rpm Plastic Gear motor, wheel and a free moving wheel at the front to construct the moving part for the car.

**6. Ultrasonic sensor**

As the name indicates, ultrasonic sensors measure distance by using ultrasonic waves. The sensor head emits an ultrasonic wave and receives the wave reflected back from the target. Ultrasonic Sensors measure the distance to the target by measuring the time between the emission and reception.

**7. Power Sources**

For providing power to our hardware, we used two different power sources. We used a 5V 1mAh power bank to power the Raspberry Pi and 6V battery for H-BRIDGE.

**8. Car Chassis and Body Parts**

Two car chassis plate was used along with fasteners and zip ties to assemble the body for the car.

**2.2.2 Software Specifications:**

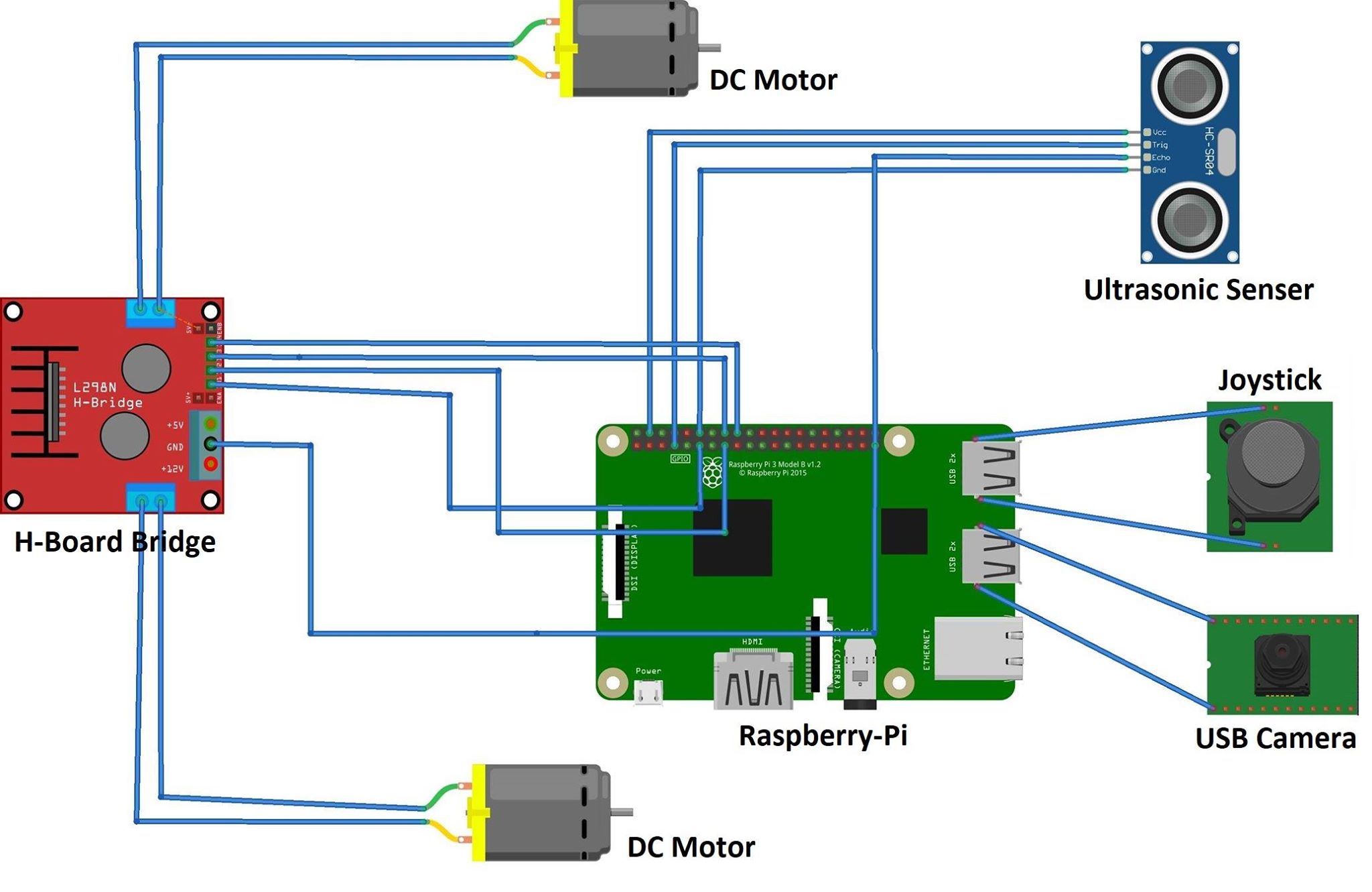
Operating System: Linux Mint, Ubuntu

Programming Language: Python

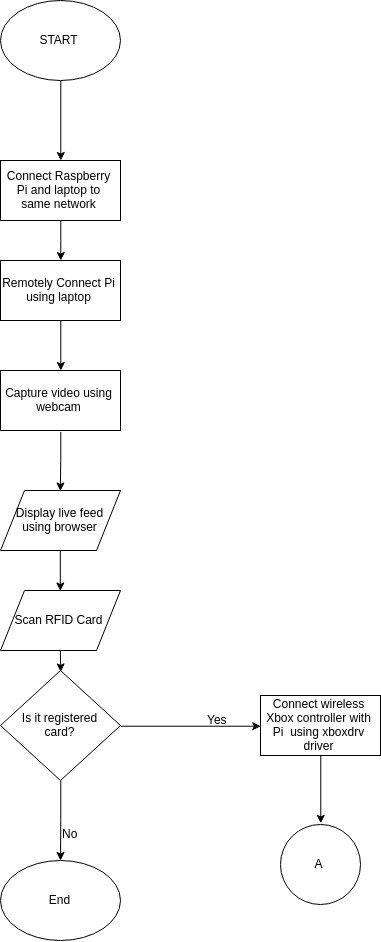
Development tools: Atom, Pycharm

Library Used: Xbox Gamepad User space Driver for Linux(Xboxdrv)

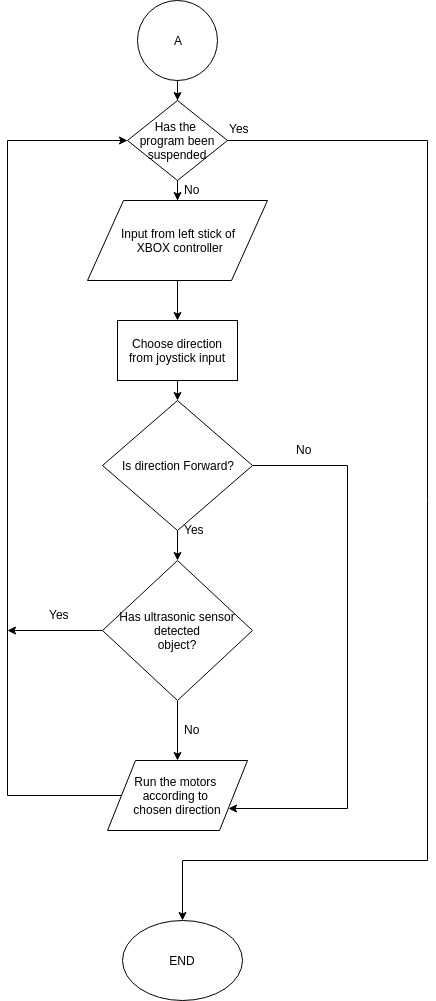
**2.3 System Design**



**Fig: System Design**

****

**Fig. FlowChart for System Initialization**

****

**Fig. Flowchart for Motor Control**

**2.4. Implementation and Testing**

**2.4.1 Hardware Implementation**

The different hardware components of the system were assembled according to the schematic diagram.

**2.4.2 Software Implementation**

The source code of the system was written by Python for all hardware components. Each component has its own set of code, run simultaneously to control the whole car.

* Live feed module

The live feed is streamed over the network port by Pi command

* Ultrasonic Module

This module is separately written by Python, to employ the connected GPIO fot the sensor operation

Source Code:

*import RPi.GPIO as GPIO;*

*importime;*

*def dist():*

*GPIO.setmode(GPIO.BCM);*

*TRIG = 04;*

*ECHO = 18;*

*GPIO.setup(TRIG,GPIO.OUT);*

*GPIO.output(TRIG,0); #set low*

*GPIO.setup(ECHO,GPIO.IN);*

*#print("Starting Measurement");*

*time.sleep(0.001);*

*GPIO.output(TRIG,1);*

*time.sleep(0.00001); #set trigger high and wait for 10 microsecond*

*GPIO.output(TRIG,0); #and set trigger high*

*#now we have to listen to the input pin*

*while GPIO.input(ECHO) == 0:*

*pass #do nothing*

*start = time.time(); #time.time() gives current time in sec*

*while GPIO.input(ECHO) == 1:*

*pass*

*stop = time.time();*

*distance = (stop-start)\*170\*100;*

*GPIO.cleanup();*

*return distance;*

* RFID

The module to scan and recognize the id card was also exclusively written in Python. It works with default module of RFID Scanner

Source Code:

*import MFRC522*

*import RPi.GPIO as GPIO*

*class SimpleMFRC522:*

*def \_\_init\_\_(self):*

*self.READER = MFRC522.MFRC522()*

*def read(self):*

*status,TagType= self.READER.MFRC522\_Request*

*(self.READER.PICC\_REQIDL)*

*if status != self.READER.MI\_OK:*

*return None*

*status, uid = self.READER.MFRC522\_Anticoll()*

*if status != self.READER.MI\_OK:*

*return None*

*id = self.uid\_to\_num(uid)*

*return id*

*def uid\_to\_num(self, uid):*

*n = 0*

*for i in range(0, 5):*

*n = n \* 256 + uid[i]*

*return n.*

* Xbox Controller

The module for Xbox support for Linux was written by Steven Jacbos. The project implemented the module to take input from the user. The input is processed by the different module given below, which includes processing ultrasonic sensor and RFID as well..

Source Code:

*import xbox*

*import sys*

*import time*

*import math*

*import RPi.GPIO as GPIO*

*import ultrasonic*

*import SimpleMFRC522*

*def initial():*

*GPIO.setmode(GPIO.BCM);*

*GPIO.setup(17, GPIO.OUT);*

*GPIO.setup(22, GPIO.OUT);*

*GPIO.setup(23, GPIO.OUT);*

*GPIO.setup(24, GPIO.OUT);*

*def move\_wheel(IN1, IN2, IN3, IN4):*

*GPIO.output(17, IN1);*

*GPIO.output(22, IN2);*

*GPIO.output(23, IN3);*

*GPIO.output(24, IN4);*

*def joypad(x,y):*

*angle = 0.0*

*if x >= 0.0 and y > 0.0:*

*# first quadrant*

*angle = math.degrees(math.atan(y/x)) if x!=0.0 else 90.0*

*elif x < 0.0 and y >= 0.0:*

*# second quadrant*

*angle = math.degrees(math.atan(y/x))*

*angle += 180.0*

*elif x < 0.0 and y < 0.0:*

*# third quadrant*

*angle = math.degrees(math.atan(y/x))*

*angle += 180.0*

*elif x >= 0.0 and y < 0.0:*

*# third quadrant*

*angle = math.degrees(math.atan(y/x)) if x!=0.0 else -90.0*

*angle += 360.0*

*return angle*

*reader = SimpleMFRC522.SimpleMFRC522() #object*

*id = reader.read()*

*print("Scan your RFID Card...")*

*while not id:*

*id = reader.read()*

*if id == 115173304638:*

*print(" ")*

*print("Welcome Bibash Shrestha")*

*print(" ")*

*joy = xbox.Joystick()*

*while not joy.Back():*

*if joy.connected():*

*print ("Connected "),*

*else:*

*print("Disconnected"),*

*angle = joypad(joy.leftX(), joy.leftY())*

*if (60 <= angle < 120):*

*distance = ultrasonic.dist()*

*if distance < 20:*

*pass*

*print("Object Ahead"),*

*print("Distance: "+ str(distance)),*

*else:*

*initial()*

*move\_wheel(1,0,1,0) #forward*

*time.sleep(0.1)*

*elif (120 <= angle < 180):*

*initial()*

*move\_wheel(0,0,1,0) #forward\_left*

*time.sleep(0.001)*

*elif (0 < angle < 60):*

*initial()*

*move\_wheel(1,0,0,0) #forward\_right*

*time.sleep(0.001)*

*elif (240 <= angle < 300):*

*initial()*

*move\_wheel(0,1,0,1) #backward*

*time.sleep(0.001)*

*elif (300 <= angle < 360):*

*initial()*

*move\_wheel(0,1,0,0) #backward\_left*

*time.sleep(0.001)*

*elif (180 < angle < 240):*

*initial()*

*move\_wheel(0,0,0,1) #backward\_right*

*time.sleep(0.001)*

*else:*

*print(" "),*

*print(" "),*

*initial()*

*move\_wheel(0,0,0,0)*

*GPIO.cleanup()*

*# Move cursor back to start of line*

*print chr(13),*

*joy.close()*

*else:*

*print("Acess Denied")*

**2.4.3 Testing**

The testing conducted was to consider the implementation details while testing the system, you give the test inputs in such a way that each and every line of code is tested.

**CHAPTER 3: COST ESTIMATION**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.N. | Components | Rate | Quantity | Cost (in Rs) |
| 1. | Raspberry Pi | 6300 | 1 | 6300 |
| 2. | Car Chasis | 100 | 2 | 200 |
| 3. | Motor | 150 | 2 | 300 |
| 4. | Wheel | 100 | 2 | 200 |
| 5. | Power Bank | 3500 | 1 | 3500 |
| 6. | Xbox Controller | 5000 | 1 | 5000 |
| 7. | USB Camera | 800 | 1 | 800 |
|  |  |  | Total | 16,300 |

**CHAPTER 4: RESULTS AND CONCLUSION**

The overall development of the system is a success. The project has accomplished our main objective of building a car from scratch using different hardware and software components. The controlling of the car using the Xbox controller has been successful and we have also been able to add RFID security to the system. The integration of the Raspberry Pi in development of the system has been insightful, as been the development of the own source code for the various hardware components of the system.

In conclusion, the team has been satisfied with the obtained results and look forward to developing the system into more efficient one.

**4.1 Limitations**

* Artificial intelligence (AI) is yet to be implemented.
* The model is trained to only stationary object.
* The processing power of Raspberry Pi is not enough**.**

4**.2 Future Enhancement**

In order to implement AI in our car, we need a lot of processing power plus supply voltage. We can train our model to overcome various obstacles and situations. We can also improve image processing power by exchanging webcam with the 360 degree camera. If we get the sufficient resources, we will be able to add more additional features in our model.

**References:**

* Freddy, K. ( 2016, May 6) Google Cardboard Xbox 360 controlled Raspberry Pi VR Robot Tank. Retrieved from<https://www.youtube.com/watch?v=z9gSZwVMybc>
* Freddy, K. ( 2017, September 17) Google Cardboard Xbox 360 controlled Raspberry Pi VR Robot Tank. Retrieved from https://www.eeweb.com/featured-projects/google-cardboard-xbox-360-controlled-raspberry-pi-vr-robot-tank
* O’Hanlon, Martin (2014, October 17) Raspberry Pi, Xbox 360 controller, Python. Retrieved from<http://www.stuffaboutcode.com/2014/10/raspberry-pi-xbox-360-controller-python.html>
* (2017, April 29) UV4L WebRTC has now landed on the Raspberry Pi Zero and Zero W models!. Retrieved from<http://www.linux-projects.org/2017/04/29/uv4l-webrtc-for-pizero/>