MINISTRY OF EDUCATION AND TRAINING

CAPSTONE PROJECT DOCUMENT

Members:	Luyen Bao Anh Pham Minh Hoang Le Xuan Huong Phung Duc Luat Do Cao Phong Dang Ngoc Tu
Supervisor:	Mr. Hoang Xuan Son
Project Code:	MEx

Hanoi, August 24th 2017

- CONTENTS

I. INTRODUCTION

1.1 Purpose

This document is created as the introduction for project MEx – our Capstone Project at FPT University. In this document, we will descript the overview of some existing systems, the initial idea for our project, a brief description about our expected system and some potential risks, critical assumptions, constrains. Moreover, this document also shows opportunities what it offers for users.

1.2 Acronyms and Definitions

Acronym & Abbreviation	Definition
MEx	Mini Explorer
FU	FPT University
VR	Virtual Reality
Remote Controller	The devices include steering wheel, pedals, gear stitch, used for driving simulator
PiCar	The model of automobile, with the Raspberry Pi inside used in this project

Table 1-1: Definitions and Acronyms

1.3 The People

1.3.1 Supervisor

	Full name	Phone	Email	Title
Supervisor	Hoàng Xuân Sơn	0936232008	SonhHX@fe.edu.vn	

Table 1-2: Supervisor's information

1.3.2 Team member

No	Full name	StudentID	Phone	Email	Role
1	Luyện Bảo Anh	SE03747	01672788452	anhlbse03747@fpt.edu.vn	Team Leader
2	Phạm Minh Hoàng	SE03769	0904882411	hoangpmse03769@fpt.edu.vn	Team Member
3	Lê Xuân Hướng	SE03388	01649132648	huonglxse03388@fpt.edu.vn	Team Member

4	Phùng Đức Luật	SE03164	01656885023	luatpdse03164@fpt.edu.vn	Team Member
5	Đỗ Cao Phong	SE03196	0979064208	phongdcse03196@fpt.edu.vn	Team Member
6	Đặng Ngọc Tú	SE03591	0868463132	tudnse03591@fpt.edu.vn	Team Member

Table 1-3: Team member information

1.4 Project information

- Project name: Mini Explorer System
- Project code: MEx
- Project group name: MEx Team
- Product type: Embedded System
- Timeline: From May 8th to August 26th, 2017

1.5 The idea

Nowadays, the rapid development of technology has a strong impact on the life of human beings. Along the rapid expansion of economic, the improvement of living standard, the demand of people about a comfort, safe and convenience life, car is going to one of main means of transportation. But in Vietnam, the prices of cars are still too high for people to own one. Therefore, they have to take driving courses so that they can practice in the real car at relatively high prices. Not to mention, during the practice, can cause accidents for the user when they are not proficient yet.

MEx is the idea of first-person view - driving simulation system through virtual reality (VR) technology. The user controls an automobile model by wheel controller, pedals as in the real car. The camera will be set up and provide first-person view to a virtual reality lens to observe all vehicle movements.

The system simulates the whole process of driving a car so that user can learn how to drive easily at home rather than going to the driving courses.

1.6 Proposal of system

1.6.1 The scope

The scope of MEx is a prototype of control device. It includes both hardware and software. Finally, product must be satisfied some below specification.

- Streaming video with following feature:
  - Frame rate: 24 frames/second.
  - Delay: Depend on network rate.
  - Resolution: 640x480
- Controllers and models must have minimum functions such as running (forward, backward), steering, braking.

- Working on the terrain is relatively flat, not too rough, not too complex, no waves or interference obstructions.
- Tracking the motion of the head to provide the most sensible viewing angle

1.6.2 Existing system

Recently, there are various products like MEx with good functions, attractive design interface. Below are some of these products:

1.6.2.1 FPW Driving - Drive an RC Car with First person view

FPW Driving is a prototype of Paul Yan – Arduino Team member. With an old PS2 wheel controller, two Arduinos, a mini FPV camera, and a headset as a standalone monitor, he made a system to control a RC car with first-person view very smoothly. The RC car–which is equipped with a Micro–interfaces with the wheel using a Uno and a PS2 Shield. Both Arduinos communicate via a pair of nRF24L01 modules

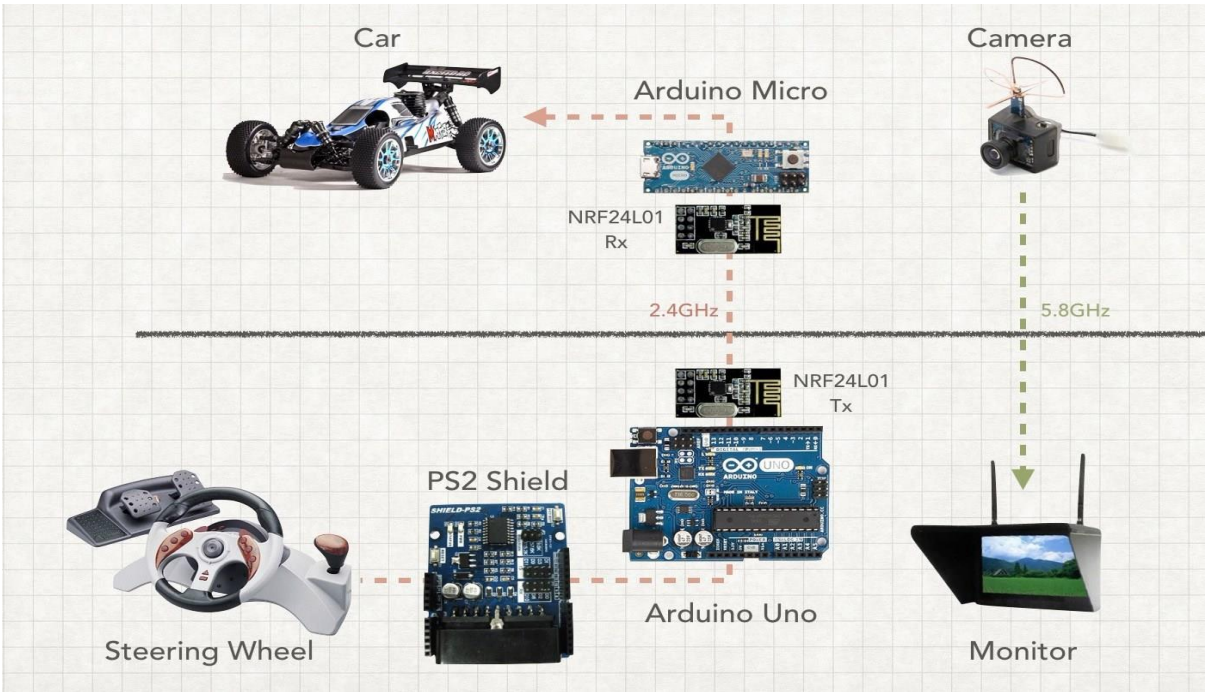


Figure 1-1: FPW-Driving Diagram

Without tracking the movement of the head, this system can only provide the front sight from straight view of the car. It will become difficult for users when they want to look to the right or left. This is the biggest shortcoming of Yan's FPW Driving.

1.6.2.2 RACEROOM with Oculus Rift

Raceroom is the one of the most typical driving simulation games. This game offers players a system of more than 20 arcades and more than 60 vehicles to enter the race. With Oculus Rift (VR lens from Microsoft), playing Raceroom will be more fun and realistic with live sound and various gameplay modes.

Player can skip racing mode, and practice driving in a virtual environment. With Driving Force from Logitech or steering wheel from other suppliers, Raceroom would be suitable for those who want to practice driving.

Figure 1-2: Screen from Playing Raceroom with Oculus Rift

As a product that has been commercialized, Raceroom is being sold on Steam for around $ 20. Along with the expensive equipment that comes with the Oculus glass, Logitech's controllers make the price of this kit close to $2000. It is very expensive for drive learner.

1.7 Benefit from project

1.7.1 For team members

- Have more experiences in working in project, project management.
- Have more knowledge about Arduino, Raspberry Pi, Android and mechanical.
- Improve skill about communicate with team members and how to work in team more effective.

1.7.2 For community

- Have a new feeling of driving.
- Learn or get more driving practice anytime.
- Explore the distant area without the need for actual movement.

1.8 Critical assumption and constraints

1.8.1 Critical assumption

- Training: Developers can self-training Arduino and Raspberry Pi Programming with Python in 3 Weeks.
- Human resources: Assume that all members in team have a good healthy to work

1.8.2 Critical constraints

- Time & Deadline: We must complete task on time. We have 14 weeks for working, each member works 4 hours/day and 5 days/week. We do not have more time for us to complete developing and deliver application to teachers. Besides, we have to submit report documents before deadline to teacher can review.
- Quality: The products must be run well
- Process: We have to follow the software processing of FPT Software
- Human resources: There are 6 members in our team, each member have to study 4 subjects at school.

1.9 Potential risks

After studying about this project, we find out some problem that we may be encountered:

- Under-estimate scope and time or miss deadline because lack of experience in group working, managing and controlling work.
- Equipment got broken because of careless or accident.
- Human resources: Team member cannot complete their works because of health reasons, key member leave team or un-cooperating on team.
- Change requirements: Requirement changed when some functions cannot be completed or some technologies is not suitable.

II. PROJECT MANAGEMENT PLAN

2.1 Definition Problem

The Introduction is clearly specified reason why MEx project was chose to develop. It is an overview concept about MEx system and be discussed some main function of existing system.

You now have the knowledge of the system's scope. This document will present project planning to get the target. All the tasks and time to implement, the resource of the system, and the risk maybe meet during development.

2.1.1 Name of this Capstone Project

This Capstone project named Mini Explorer System, abbreviated as MEx.

2.1.2 Boundaries of the System

2.1.2.1 Boundaries of the System

The system under development of this Capstone Project will include:

- The controller has the task of sending the request via wireless, saving control information, controlling device.
- Wireless is an information bridge between the controller and car.
- A central circuit board in the car has responsible for data exchange with the gateway through Arduino to transmit, receive and process information from user.
- User manual, Test Document
- Design circuit broad, Design Document
- Source code Android App and Arduino

2.1.2.2 Development Environment

Below is the list of hardware and software requirements needed for development environment:

Hardware requirements:

Develop:

- Arduino/WeMos
- Raspberry Pi
- Sensor, servo motor, resistors, capacitors, wire…
- Personal Computers with 4 Gigabytes of RAM or more

Test:

- Personal Computers

Software requirements:

- Operating System: Windows 8.1, 10 Pro – 64bit
- Design software: Proteus 7.8
- IDEs: Android Studio and SDK tools, JDK 7, Arduino IDE, Python IDLE.
- Document: Microsoft Office 2016, Microsoft Project 2016

2.2 Project organization

2.2.1 System Process Model

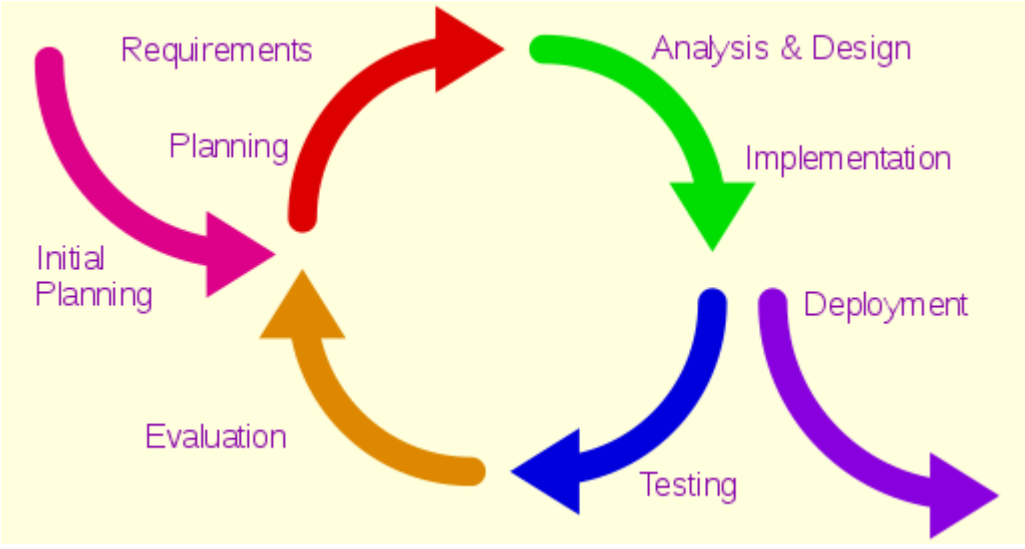


Figure 2-1: Iterative and Incremental Software Process Model

This figure above describes the information and products flow lifecycle process model. MEx project uses the Iterative and Incremental Software Process Model.

Iterative and Incremental Software Process Model is a method of software development that is model around a gradual increase in feature additions and a cyclical release and upgrade pattern.

The Iterative and Incremental Software Process Model is most use when the scope of the project is big, the major requirements were defined clearly, some more detail will be added in time, and for the newbie group in software development.

By using this software process model, we break down the developing system task into series of smaller tasks are completed separately, evaluated, and subsequently re-worked until the system's performance adequately. In addition, the iterative model is easier than other models when the issues were discovery. They are feedback to the team, and solution found while the project is still in development.

2.2.2 Roles and Responsibilities

2.2.2.1 Organization and Structure

No	Name	Role	Responsibilities
1	Hoàng Xuân Sơn	Supervisor	- Approving and supporting process to run project. - Suggesting solution when the project meets issue.
2	Luyện Bảo Anh	Project Manager	- Select methodology to identify risk. - Set common rule to all members in project to avoid risk. - Approve solution to resolve issues.
3	Lê Xuân Hướng	Technical Leader	- Investigate technical issues. - Review source codes.
4	Phùng Đức Luật	QA Leader	- Keeping all of member on process and follow common rule. - Reviewing quality of resolved issues.
5	Đỗ Cao Phong	Software Dev	- Follow process of project. - Follow common rule of project. - Develop android product
6	Phạm Minh Hoàng	Hardware Dev	- Follow process of project. - Follow common rule of project. - Develop Arduino product and remote controller
7	Đặng Ngọc Tú	Test Leader	- Involve to test product -Responsible for test execution, test set-up, test run, evaluation of test run and error recovery

Table 2-1: Project Structure

2.2.2.2 Project Team Member

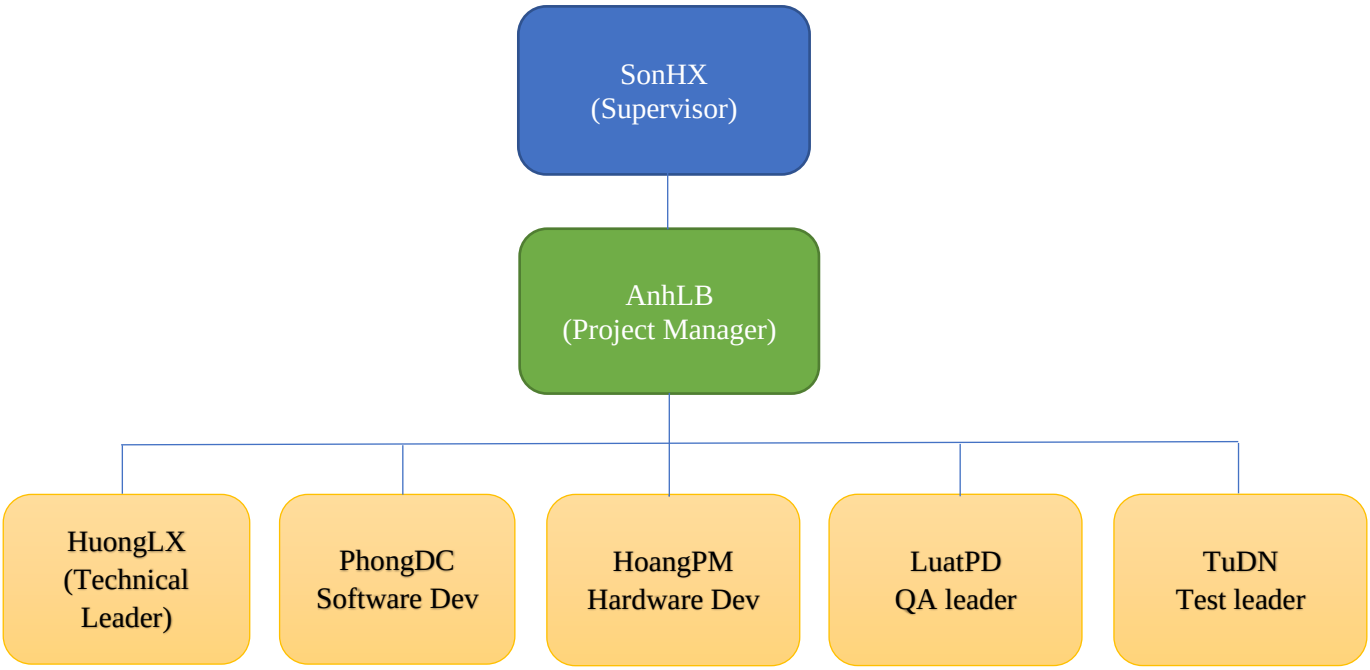


Figure 2-2: Project Team Member

2.2.3 Tools and Techniques

- Programing languages: Java, Python 3.5, Arduino
- Process Model: Iterative and Incremental Software Process Model.
- IDEs: Android Studio, Arduino IDE, Python IDLE
- Design tool: Fritzing, Photoshop, Proteus
- Other:
    - Revision control: Gitkraken &Tortoise GIT
    - Office tools: MS Office 2016

2.3 Project management plan

ID		Task Mode	Task Name	Duration	Start	Finish	Resource Names
1			**Capstone Project Plan**	**80 days**	**Mon 5/8/17**	**Fri 8/25/17**	
2			**Phase 1: Create Function required for Car**	**45 days**	**Mon 5/8/17**	**Fri 7/7/17**	
3	✓		**1.1. Initiating**	**3 days**	**Mon 5/8/17**	**Wed 5/10/17**	
4	✓		1.1.1. Generate Project Ideas	2 days	Mon 5/8/17	Tue 5/9/17	MEx Team
5	✓		1.1.2. Ideas Review	1 day	Tue 5/9/17	Tue 5/9/17	MEx Team
6	✓		1.1.3. Meeting for closing final topic and set group working rules	1 day	Wed 5/10/17	Wed 5/10/17	MEx Team and Supervisor
7	✓		**1.2. Planing and preparing**	**11 days**	**Fri 5/12/17**	**Fri 5/26/17**	
8	✓		1.2.1. Requirement analysis and Planning	2 days	Fri 5/12/17	Mon 5/15/17	MEx Team
9	✓		1.2.2. Tools and devices analysis and choosir	1 day	Mon 5/15/17	Mon 5/15/17	MEx Team
10	✓		1.2.3. Download and intall softwares	2 days	Tue 5/16/17	Wed 5/17/17	MEx Team
11	✓		1.2.4. Order Pi-kit and Wifi sheild from over sea	8 days	Wed 5/17/17	Fri 5/26/17	HoangPM,TuDN
12	✓		**1.2.5. Project Integration Management**	**3 days**	**Fri 5/12/17**	**Tue 5/16/17**	
13	✓		Develop Project management plan	3 days	Fri 5/12/17	Tue 5/16/17	AnhLB
14	✓		1.2.6. Setup project environment	3 days	Fri 5/12/17	Tue 5/16/17	MEx Team
15	✓		**1.2.7. Project Scope Management**	**3 days**	**Fri 5/12/17**	**Tue 5/16/17**	
16	✓		1.2.7.1. Define project scope	2 days	Fri 5/12/17	Mon 5/15/17	MEx Team
17	✓		1.2.7.2. Collect requirements	3 days	Fri 5/12/17	Tue 5/16/17	MEx Team
18	✓		**1.2.8. Project Time Management**	**5 days**	**Tue 5/16/17**	**Sun 5/21/17**	
19	✓		1.2.8.1. Build work breakdown	5 days	Tue 5/16/17	Sun 5/21/17	AnhLB
20	✓		1.2.8.2. Develop resources plan	5 days	Tue 5/16/17	Sun 5/21/17	HoangPM,HuongLX
21	✓		**1.2.9. Project Quality Management**	**4 days**	**Fri 5/19/17**	**Wed 5/24/17**	
22	✓		1.2.9.1. Create quality criteria	2 days	Fri 5/19/17	Sun 5/21/17	AnhLB,LuatPD
23	✓		1.2.9.2. Create quality management plan	4 days	Sun 5/21/17	Wed 5/24/17	LuatPD
24	✓		**1.2.10. Project Risk Management**	**10 days**	**Mon 5/15/17**	**Fri 5/26/17**	
25	✓		1.2.10.1. Identify risks	8 days	Mon 5/15/17	Wed 5/24/17	LuatPD,TuDN
26	✓		1.2.10.2. Plan risk mamagement	3 days	Wed 5/24/17	Fri 5/26/17	LuatPD,TuDN
27	✓		**1.3. Technical understanding**	**11 days**	**Tue 5/16/17**	**Tue 5/30/17**	
28	✓		1.3.1. Studying Android and Java developme	11 days	Tue 5/16/17	Tue 5/30/17	MEx Team
29	✓		1.3.2. Study about MEx technology	6 days	Tue 5/16/17	Tue 5/23/17	AnhLB
30	✓		1.3.3. Study Wifi communication and module Wifi Sheild	9 days	Tue 5/16/17	Fri 5/26/17	HoangPM,PhongDC
31	✓		1.3.4. Study about Arduino, Pi	9 days	Tue 5/16/17	Fri 5/26/17	LuatPD,HuongLX
32	✓		1.3.5. Study about Circuit of hardware	9 days	Tue 5/16/17	Fri 5/26/17	TuDN,HuongLX,LuatPD
33	✓		**1.4. Design**	**6 days**	**Mon 5/29/17**	**Mon 6/5/17**	
34	✓		**1.4.1. Design Car model for MEx System**	**6 days**	**Mon 5/29/17**	**Mon 6/5/17**	
35	✓		1.4.1.1. Create a Car with 3 servo	6 days	Mon 5/29/17	Mon 6/5/17	HuongLX
36	✓		1.4.2. Design Controller for MEx system	3 days	Thu 6/1/17	Mon 6/5/17	AnhLB
37	✓		1.4.3. Embedded and Android Software Archtecture design	3 days	Thu 6/1/17	Mon 6/5/17	PhongDC,HoangPM
38	✓		1.4.4. Embedded and Android Software Detailed design	3 days	Thu 6/1/17	Mon 6/5/17	TuDN,LuatPD,PhongDC
39	✓		**1.5. Implementation**	**18 days**	**Thu 6/1/17**	**Mon 6/26/17**	
40	✓		1.5.1. Assemble Components	6 days	Mon 6/5/17	Mon 6/12/17	AnhLB
41	✓		1.5.2. Code for wifi communication between Android app and Car (arduino)	6 days	Mon 6/12/17	Mon 6/19/17	PhongDC,HoangPM

ID	Task Mode	Task Name	Duration	Start	Finish	Resource Names
42		1.5.3. Code for Car model embedded software module	18 days	Thu 6/1/17	Mon 6/26/17	HuongLX,HoangPM
43		1.5.4. Code for Android application	17 days	Thu 6/1/17	Fri 6/23/17	PhongDC,TuDN
44		1.5.5. Make move (forward, backward, turnleft, turnright) controlled by Android	10 days	Mon 6/12/17	Fri 6/23/17	AnhLB,HuongLX,LuatPD
45		**1.6. Testing**	**11 days**	**Fri 6/23/17**	**Fri 7/7/17**	
46		1.6.1. Design test cases	4 days	Fri 6/23/17	Wed 6/28/17	TuDN,LuatPD
47		1.6.6. System test	8 days	Wed 6/28/17	Fri 7/7/17	MEx Team
48		**Phase 2: Additional Function and Controller**	**30 days**	**Mon 7/10/17**	**Fri 8/18/17**	
49		**2.1. Planning and preparing**	**4 days**	**Mon 7/10/17**	**Thu 7/13/17**	
50		2.1.1. Study python and Rasberry Pi	4 days	Mon 7/10/17	Thu 7/13/17	AnhLB,HuongLX
51		**2.2. Design**	**7 days**	**Thu 7/13/17**	**Fri 7/21/17**	
52		2.2.1. Design for Hardware	6 days	Thu 7/13/17	Thu 7/20/17	MEx Team
53		2.2.2. Design for embedded software	4 days	Mon 7/17/17	Thu 7/20/17	HoangPM,HuongLX,Luat
54		2.2.3. Update Android Software Archtecture design	5 days	Mon 7/17/17	Fri 7/21/17	PhongDC,TuDN
55		**2.3. Implementation**	25 days	Mon 7/10/17	Fri 8/11/17	
56		2.3.1. Assemble components	13 days	Wed 7/19/17	Fri 8/4/17	AnhLB,HuongLX,PhongD
57		2.3.2. Update Embedded software	13 days	Wed 7/19/17	Fri 8/4/17	HoangPM,HuongLX
58		2.3.3. Code for Raspi, communication between Car(Pi) and controller	9 days	Wed 7/19/17	Mon 7/31/17	AnhLB,HuongLX
59		2.3.4. Add camera function and GearVR	7 days	Sun 7/23/17	Mon 7/31/17	PhonngDC,TuDN
60		2.3.5. Update Android application with more function	13 days	Wed 7/19/17	Fri 8/4/17	LuatPD,PhongDC,TuDN
61		2.3.6. Complete Car Model	11 days	Sun 7/30/17	Fri 8/11/17	HuongLX,TuDN,AnhLB
62		2.3.7 Run in fact	2 days	Mon 7/10/17	Tue 7/11/17	MEx Team
63		**2.4. Testing**	**10 days**	**Mon 8/7/17**	**Fri 8/18/17**	
64		2.4.1. Design Test cases	3 days	Mon 8/7/17	Wed 8/9/17	LuatPD,TuDN
65		2.4.3. System test	7 days	Thu 8/10/17	Fri 8/18/17	MEx Team
66		**2.5. Perform quality assurance**	**4 days**	**Tue 8/15/17**	**Fri 8/18/17**	
67		2.5.1. Review coding	3 days	Tue 8/15/17	Thu 8/17/17	AnhLB,LuatPD
68		2.5.2. Integration test	4 days	Tue 8/15/17	Fri 8/18/17	LuatPD,PhongDC
69		2.5.3. Acquire project team	4 days	Tue 8/15/17	Fri 8/18/17	AnhLB
70		**Prepare for the Thesis Defend**	**5 days**	**Sun 8/20/17**	**Fri 8/25/17**	
71		1. Project review	2 days	Sun 8/20/17	Mon 8/21/17	MEx Team,Supervisor
72		2. Print documents	1 day	Mon 8/21/17	Mon 8/21/17	TuDN
73		3. Prepare for the final presentation	3 days	Wed 8/23/17	Fri 8/25/17	MEx Team
74		**Document**	**72 days**	**Thu 5/11/17**	**Sat 8/19/17**	
75		1. Report 1: Introduction	6 days	Thu 5/11/17	Thu 5/18/17	TuDN,AnhLB
76		2. Report 2: Project management plan	6 days	Mon 5/22/17	Mon 5/29/17	TuDN
77		3. Report 3: System requirements specification	16 days	Mon 5/29/17	Mon 6/19/17	LuatPD
78		4. Report 4: System design description	10 days	Mon 6/26/17	Fri 7/7/17	TuDN,LuatPD
79		5. Report 5: System implementation & test	21 days	Mon 7/10/17	Mon 8/7/17	AnhLB,LuatPD
80		6. Report 6: System User's Manual	11 days	Mon 8/7/17	Sat 8/19/17	TuDN,LuatPD
81		7. Slide	7 days	Fri 8/11/17	Sat 8/19/17	AnhLB,TuDN
82		8. Final report	7 days	Fri 8/11/17	Sat 8/19/17	TuDN,AnhLB

Figure 2-3: Project Management Plan

Refer to [MEx_ProjectPlan.mpp]

2.3.1 Human Resource

Human resource:

- Team member
- Supervisor

Non – human resource:

- Equipment: Personal Computers, Arduino, Raspberry PI
- Building: FPT University, Thach Hoa, Thach That, Hanoi
- Building: FPT University's Library, Thach Hoa, Thach That, Hanoi

2.3.2 Meeting Minutes

An example of group's meeting minute during the time executed project:

Meeting/Project Name:	MEx		
Date of Meeting:	15/052017	***Time: (Type)***	2 hours
Facilitator:	AnhLB	***Location:***	Library of FPT University
Note Taker:	TuDN		
Kick off and create Project Charter.			
2. Attendance			

Name	***Roles***	***E-mail***	***Phone***
Luyện Bảo Anh	Project Manager	AnhlbSE03747@fpt.edu.vn	01672788452
Lê Xuân Hướng	Technical Leader	HuongLXSE03388@fpt.edu.vn	01649132648
Đỗ Cao Phong	Developer	PhongDCSE03196@fpt.edu.vn	0979064208
Phạm Minh Hoàng	Developer	HoangPMSE03769@fpt.edu.vn	0904882411
Phùng Đức Luật	QA Leader	LuatPDSE03164@fpt.edu.vn	01656885023
Đặng Ngọc Tú	Test Leader	TuDNSE03591@fpt.edu.vn	0868463132

3. Content:
1. Kick off meeting. 2. Identify Goals and Objectives. 3. Specify roles and responsibilities. 4. Estimate project budget. 5. Identify main project success criteria. 6. Develop Project Charter. 7. Assign mission for each member. 8. Set up time for next meeting.

Table 2-3: Meeting minutes template

2.3.3 Risk Management Plan

No	Name	Probability	Prevention	Correction	Impact
1	**Miscommunication**	Medium	After a meeting, one group member creates an interview Report. Every participant or absence person should get a copy of this report. Team members should not hesitate to ask questions if they are unclear.	When it becomes clear that miscommunication is causing problem, the team members are gathered in a meeting to clear thing up.	High
2	**Design Error**	High	The design should be reviewed very critically. Team leader	When errors in the design are noticed by PM or team leader	Medium

			should be consulted frequency on his opinion about the feasibility and the correctness of certain design decisions.	should be consulted to help correct the design errors as soon as possible. All the work, that depends on the faulty design, should be halted until the error is corrected.	
3	**Hardware Failure**	Low	Check all of hardware before buying. Make sure the current and volt of this hardware before using.	Creating a list of store that is selling this hardware. Checking it exist if having plan goes to buy.	High
4	**Illness or absence of team member**	Medium	Team members should warn their team leader timely before a planned period of absence.	By ensuring that knowledge is shared between team members, work can be taken over quickly by someone else if a person gets ill.	Medium
5	**Requirement change**	Medium	Carefully brainstorm system's features among team members. Regularly hold meeting to define and discuss all the features of systems. Design system carefully. Analyze all the	Team meetings with supervisor to determine whether new feature should be implemented or not. Team leaders create implementation plan for implemented	High

			possible cases to minimize the change	features and sent to team members.	
6	**Time shortage**	High	Project manager should create more spare time and calculate plus 20% buffer time.	Lacking time is the fatal problem, can run project to failure. PM should analysis and has change on the next phase.	High

Table 2-4: Risk Management Plan

2.3.4 Communication Plan

2.3.4.1 Communication between team members

Weekly meeting schedule: By using Iterative and Incremental Process Model, MEx Project System will be divided into a series of small tasks, each task will be assigned to team members by Technical Leader and depend on difficulty, and Technical Leader will be assigned deadlines for each task. We will have a meeting every Thursday, Friday and Monday to report

The progress of whole team's tasks. Any member who doesn't finish his/her task (without reasonable explanation), will be fined. If there is any issue, we will discuss and find solution together. If it is too difficult and can't be solved by ourselves, we will ask our supervisor for advises.

- Unscheduled meeting: If someone has an important problem want to be solved immediately, we will have a meeting for discussion.
- Communication channel: Our main communication channels are face-to-face meeting, email, Facebook, Skype. However, we sometimes can make a phone call or instant message if someone has problem.

2.3.4.2 Communication with Supervisor

- *Face-to-face* meeting: Weekly on every Thursday afternoons to make sure that supervisor can keep tracking of the team's progress.
- *E*-mail: Gmail is the fastest way to get device and document checking from supervisor.
- *Mobile phone:* is used to get time and place arranged for the meeting every weekly.

2.4 Project Directory

Main folder	Sub-folder	Purpose
Document	Design	Diagrams and schematics.
	Reports	Store deliverables of reports
	Plan	Schedule and task list for member
	Slide	Presentation slideshow and resource
Source	Android	Store android app source code
	Raspi	Store Python code for Raspberry Pi
	Arduino	Firmware source of Remote Controller
Users	Each team members own a folder	Team member's working area
Reference		Store reference needed in project

Table 2-5: Project Directory

III. REQUIREMENTS

3.1. User Requirements Specification

3.1.1. User requirement

- User use smartphone (or remote controller) connect to car's raspberry pi through WIFI
- User can use smartphone or remote controller to control the movement of the car and the camera
- User can use smartphone to switch control between smartphone or remote controller
- In remote controller mode, user can use smartphone to see the camera view

3.1.2. Car requirement

- Right direction under the control of user
- Operation of the car must be stable and safe
- Easy to receive signal from user

3.1.3. Android application requirement

- Well design with minimum cost and maximum quality
- Has long durability in period of time
- Be simple and user-friendly

3.1.4. Remote Controller requirement

- Auto connect to system
- Low-cost hardware module

3.2. System Requirement Specification

3.2.1 Interface requirement

3.2.1.1 User Interface

User interface (UI) of mobile application must be design base on Flat UI Design. In this project, we only develop mobile application on Android. We ensure that the navigation options in the application will be similar; and all error occurring and exception handling will be catch and display for user with friendly messages.

3.2.1.2 Hardware interface

- **Smartphone devices**: the application only runs on smart phone which support Android 4.4+; the component helps user to control the car and also display the "view" from the car's camera
- **Arduino WEMOS**: Use to control the car in remote controller, the signal will be transfer to Raspberry PI through WIFI
- **Raspberry PI**: Receive control signals from remote controller (or Smartphone device) and control the operation motors/Servo; also transfer video data from camera to smartphone device

3.2.1.3 Software interface

Figure 3-1: Controller screen

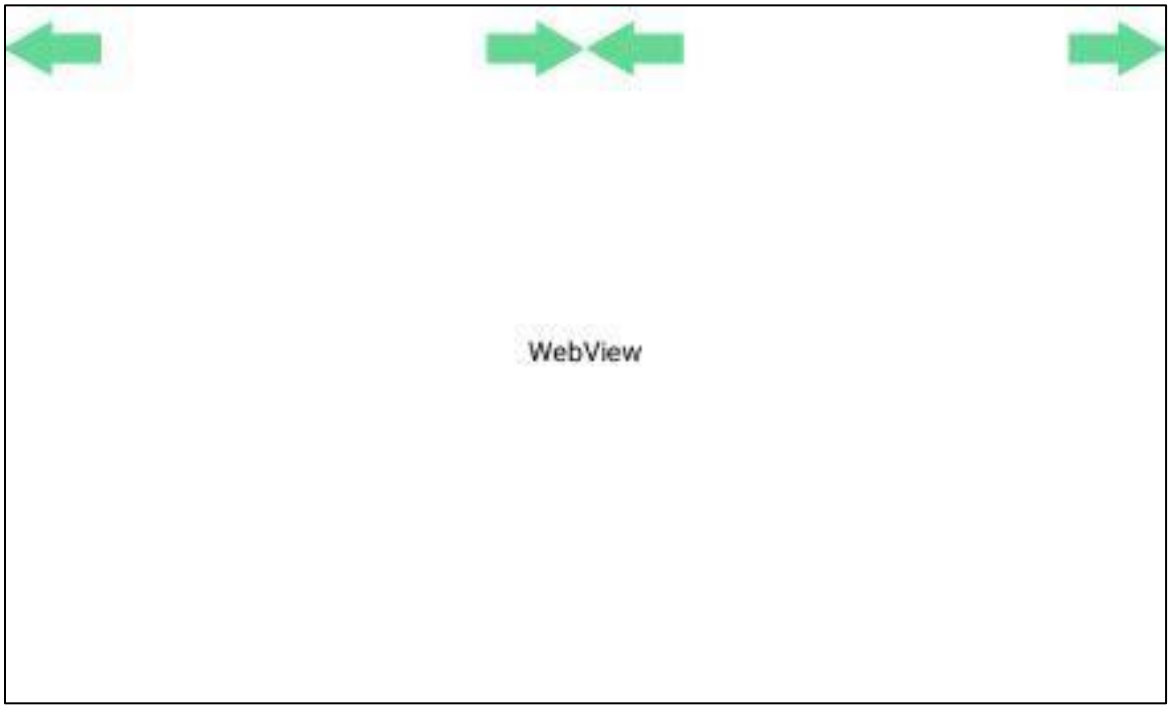
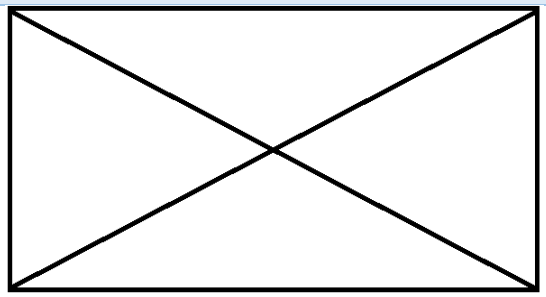
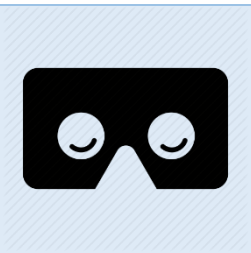
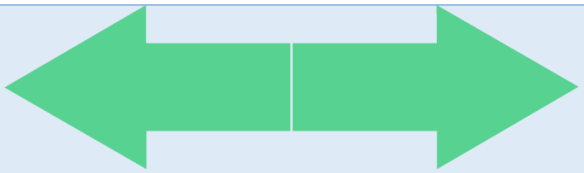


Figure 3-2: VR mode screen

This table below will show detail function of all component of **Android application**:

No.	Component	Function	Illustrate Image	Description
1	Steering Wheel	Control movement of the car		User can turn left/right to Control the car turn left/right
2	Pedal button	Increase the car speed		User can press the button to increase car speed
3	Break button	Decrease the car speed		User can press the button to decrease the car speed
4	Gear switcher	Switch run mode		User can pull the switcher down to change the car mode. P – parking D – Forward R – Backward

5	Light signal	Light signal		User can press an arrow to control car light signal. Press 1 time to turn on, then press again/or the other button to turn off (or change the light)
6	Streaming screen	Stream all image info get from camera module in the car		Users watch this to know all info that robot's camera is streaming
7	VR mode button	Enter the VR mode		Press the button to enter the VR mode
8	Disconnect button	Disconnec t to the car system		Press the button to disconnect
9	Light signal	Display the light signal status		Receive data from system, when the light signal left/right turn on, the arrow will change into the brighter color

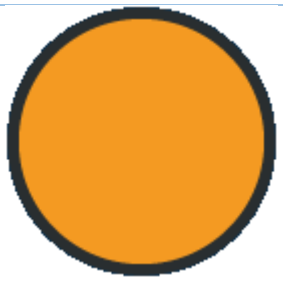
10	Buzzer	Turn on the sound signal		Hold the "sound signal" button to turn on then release to turn off the sound signal

Table 3-1: Functions Detail

3.3. System Features

3.3.1 General use Case Diagram

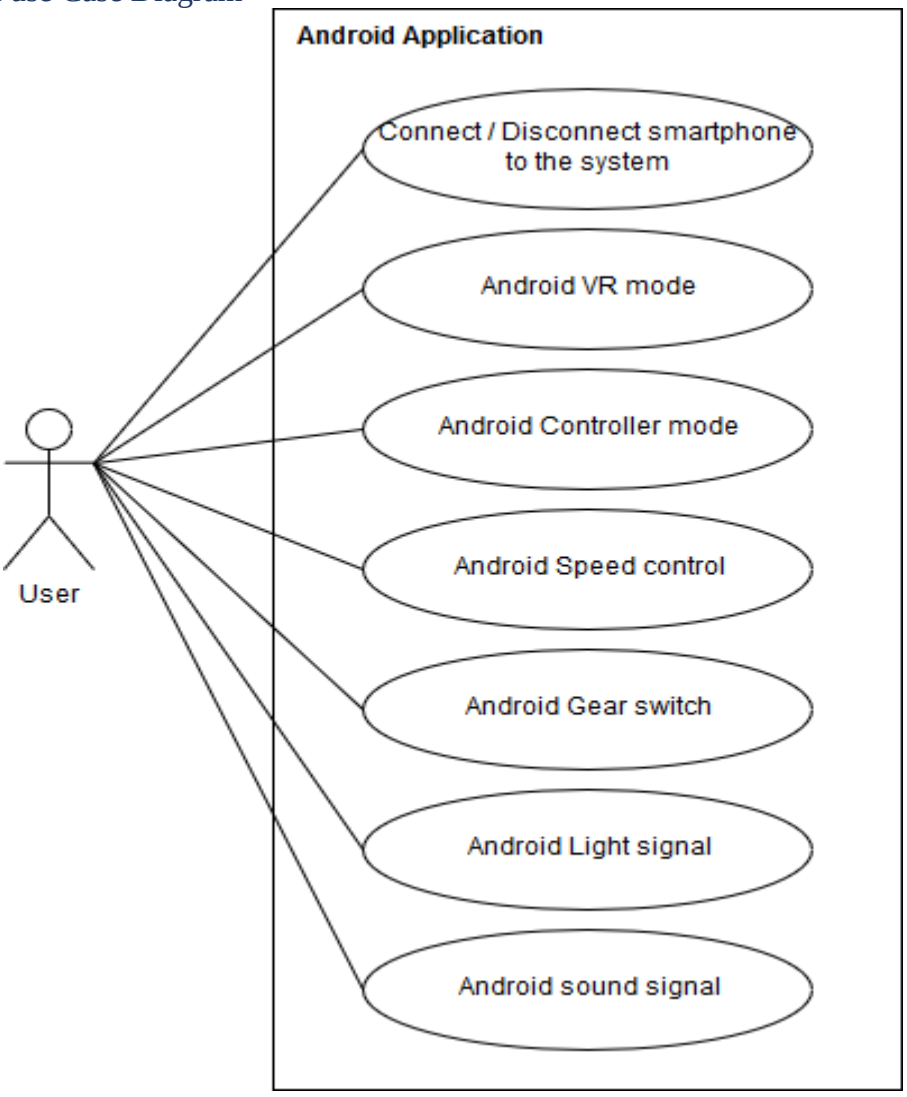


Figure 3-1: Use case (Android) diagram

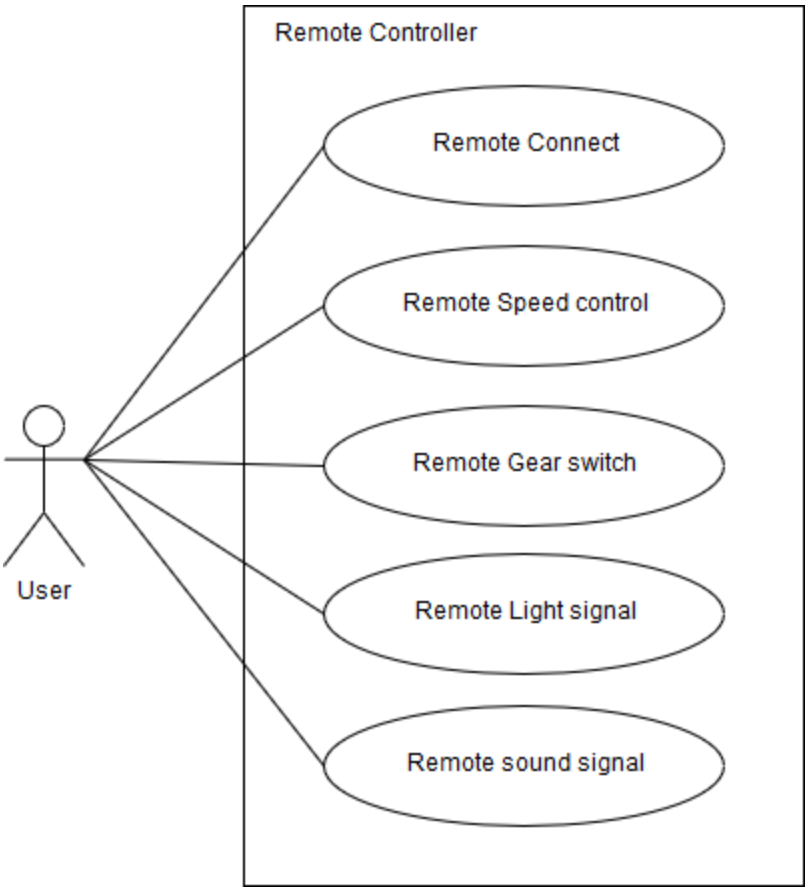


Figure 3-2: Use case (Remote controller) diagram

3.3.2 Functional requirement

3.3.2.1 Connect smartphone

USE CASE 1 SPECIFICATION			
Use-case No.	UC001	**Use-case version**	1.0
Use-case Name	Connect smartphone		
Author	PhongDC		
Date		**Priority**	High
Actor: User			

Description: User use smartphone to connect to system though WIFI
Goal: Successfully connected to the car system
Triggers:
- Connect to the car WIFI
- Start Android Application
- Press the "Connect device" button

Pre-condition:
- Android application is start successful
- Car WIFI is turn on

Post-condition:
- Disable the Remote controller.

Main Success Scenario:

No	Actor	Action
1	User	Connect to car's WIFI
2	User	Press the "Connect device" button
3	Car System	Check the connection: • Return "OK" if success • Return "NO" if fail to connect • Return "LIMIT" if any device has connected.
4	Car System	Transfer signal to android device
5	Android	• If success: move to "controller mode" screen • If fail: display message "Could not connect to server, please check the connection". • If limit: display message "Over capacity. Close the previous connection or reset server".

Alternative Scenario: None

Exceptions: N/A

3.3.2.2 Disconnect Smartphone

USE CASE 2 SPECIFICATION			
Use-case No.	UC002	**Use-case version**	1.0
Use-case Name	Disconnect smartphone		
Author	PhongDC		
Date		**Priority**	High

Actor: User
Description: User use smartphone to disconnect to the car system
Goal: Successfully disconnected to the car system
Triggers:
- Press “disconnect device” button

Pre-condition:
- Android application is connected successful with the car system

Post-condition:
- After disconnect user can connect again.

Main Success Scenario:

No	Actor	Action
1	User	Press “Disconnect device” button
2	Android System	Display disconnects successful message
3	Android System	Move to “Main” Screen

Alternative Scenario: None

Exceptions: N/A

Note and issues:

3.3.2.3. Enter the VR mode

USE CASE 3 SPECIFICATION			
Use-case No.	UC003	**Use-case version**	1.0
Use-case Name	Android VR mode		
Author	PhongDC		
Date		**Priority**	High

Actor: User
Description: Allow User to enter VR mode to view the camera in the VR mode and control the car by Remote controller
Goal: Successful enter the VR mode
Triggers:
- Press on “VR mode” button

Pre-condition:
- The Android application is connected successful with the car system
- Remote controller is turned on

Post-condition:
- Allow remote controller to control the car
- User can turn back to “Controller mode”

Main Success Scenario:

No	Actor	Action
1	User	Press on “VR mode” button
2	Android System	Send signal to Car system
3	Android System	Display the “VR mode" screen

Alternative Scenario: None

Exceptions: N/A

Note and issues:

3.3.2.4. Enter Controller mode

USE CASE 4 SPECIFICATION			
Use-case No.	UC004	**Use-case version**	1.0
Use-case Name	Controller mode		
Author	PhongDC		
Date		**Priority**	High

Actor: User
Description: Allow User to enter Controller mode to control the car by Android application
Goal: Successful enter the Controller mode
Triggers:
- Press on “back” button on the Android device

Pre-condition:
- Successful enter the “VR mode”

Post-condition:
- The remote controller is disable.
- User can turn back to the “VR mode”

Main Success Scenario:

No	Actor	Action
1	User	Press on “back” button
2	Android System	Send signal to the car system
3	Car System	Disable the “Remote controller”
4	Android System	Display the “Controller mode” screen

Alternative Scenario: None

Exceptions: N/A
Note and issues:

3.3.2.5 Android Speed control

USE CASE 5 SPECIFICATION			
Use-case No.	UC005	**Use-case version**	1.0
Use-case Name	Android speed control		
Author	PhongDC		
Date		**Priority**	High

Actor: User
Description: Use Android application to control the car movement
Goal: Control movement of the car
Triggers:
- Turn the steering wheel to left or right that user want the car to turn in that direction.
- Press the "Pedal" button to increase the movement speed of the car
- Press the "break" button to decrease the movement speed of the car

Pre-condition:
- Android application is connected successful with the car system
- Enter the "Controller mode"
- Gear switch isn't in "parking mode"

Post-condition:
- N/A

Main Success Scenario:

No	Actor	Action
1	User	Turn the steering wheel to left or right and also press the "Pedal" button to increase the movement speed or press the "break" button to decrease the movement
2	System	Send a signal to car's system to control the movement of the car

Alternative Scenario: None

Exceptions: N/A

Note and issues:

3.3.2.6 Android Gear Switch

USE CASE 6 SPECIFICATION			
Use-case No.	UC006	**Use-case version**	1.0
Use-case Name	Android Gear switch		
Author	PhongDC		
Date		**Priority**	High

Actor: User
Description: User use smartphone to control car run mode (Run forward / backward)
Goal: Control run mode of the car
Triggers:
- Drag the "Gear Switcher" button to switch between "parking", "run forward" and "run backward" mode

Pre-condition:
- Android application is connected successful with the car system
- Enter "Controller" mode

Post-condition:
- N/A

Main Success Scenario:

No	Actor	Action
1	User	Drag the "Switcher" lever.
2	Android System	Send a signal to car system to switch run mode - "0": parking - "1": run forward - "2": run backward

Alternative Scenario: None

Exceptions: N/A

Note and issues:

3.3.2.7 Android light signal

USE CASE 7 SPECIFICATION			
Use-case No.	UC007	**Use-case version**	1.0

Use-case Name	Android light signal		
Author	PhongDC		
Date		**Priority**	High

Actor: User
Description: User use smartphone to turn on/off the light signal
Goal: successful to turn on/off the light signal
Triggers:
- Press the "light signal left" or "light signal right" arrow button

Pre-condition:
- Android application is connected successful with the car system
- Enter "Controller" mode

Post-condition:
- N/A

Main Success Scenario:

No	Actor	Action
1	User	Press the "light signal left" or "light signal right" arrow button
2	Android System	Transfer signal to car system - "1": light signal left - "2": light signal right
3	Car system	Receive the signal then turn on the light.
4	User	Press the arrow button again to turn off or tab the other arrow button to change the light signal.
5	Android system	Transfer signal to car system - "0": turn off - "1": light signal left - "2": light signal right

Alternative Scenario: None

Exceptions: N/A

Note and issues:

3.3.2.8 Android Sound signal

USE CASE 8 SPECIFICATION			
Use-case No.	UC008	**Use-case version**	1.0
Use-case Name	Android Sound signal		

Author	PhongDC		
Date		**Priority**	High

Actor: User
Description: Allow use to turn on the sound signal
Goal: Successful to turn on the sound signal
Triggers:
- Hold the "Sound" button

Pre-condition:
- Android application is connected successful with the car system
- Enter "Controller" mode

Post-condition:
- N/A

Main Success Scenario:

No	Actor	Action
1	User	Hold the "sound" button to turn on the sound then release the button to turn off.
2	Android System	Transfer signal to the car system - "1": turn on - "0": turn off

Alternative Scenario: None

Exceptions: N/A

Note and issues:

3.3.2.9 Remote Connect

USE CASE 9 SPECIFICATION			
Use-case No.	UC009	**Use-case version**	1.0
Use-case Name	Remote connect		
Author	HoangPM		
Date		**Priority**	High

Actor: User
Description: When user turn on the "Remote controller", it'll auto connect to the car system
Goal: Successful connect to the car system
Triggers:
- Turn on the "Remote controller"

Pre-condition:
- N/A

Post-condition:
- N/A

Main Success Scenario:

No	Actor	Action
1	User	Turn on the "Remote controller"
2	Remote System	Transfer signal to the car system
3	Car system	Receive signal. Check if any device has connected. - Return "OK": Successful to connect - Return "NO": Fail to connect - Return "LIMIT": if any device has connected
4	Car system	If connect successful. System check the controller mode. - Return "True" if Android device has not connected or user use the "VR mode" in the android app. - Return "Fail" if user use the "Controller mode".

Alternative Scenario: None

Exceptions: N/A

Note and issues:

3.3.2.10 Remote Speed control

USE CASE 10 SPECIFICATION			
Use-case No.	UC010	**Use-case version**	1.0
Use-case Name	Remote Speed control		
Author	HoangPM		
Date		**Priority**	High

Actor: User
Description: Use Remote controller to control the car movement
Goal: Control movement of the car
Triggers:
- Turn the steering wheel to left or right that user want the car to turn in that direction.

- Push the “Pedal” button to increase the movement speed of the car
- Push the “break” button to decrease the movement speed of the car

Pre-condition:
- Remote Controller is connected successful with the car system
- Remote controller has allowed to control.
- Remote Gear switch lever isn’t in “parking mode”

Post-condition:
- N/A

Main Success Scenario:

No	Actor	Action
1	User	Turn the steering wheel to left (or right) and also tab on the “Pedal” button to increase the movement speed or tab the “break” button to decrease the movement
2	Remote System	Send a signal to car’s system to control the movement of the car

Alternative Scenario: None

Exceptions: N/A

Note and issues:

3.3.2.11 Remote Gear Switch

USE CASE 11 SPECIFICATION			
Use-case No.	UC011	**Use-case version**	1.0
Use-case Name	Remote Gear switch		
Author	HoangPM		
Date		**Priority**	High

Actor: User
Description: User use smartphone to control car run mode (Run forward / backward)
Goal: Control run mode of the car
Triggers:
- Pull “Gear Switcher” Lever to switch between “parking”, “run forward” and “run backward” mode

Pre-condition:
- Remote controller is connected successful with the car system
- Remote controller has allowed to control

Post-condition:
- N/A

Main Success Scenario:

No	Actor	Action
1	User	Pull the “Switcher” lever.
2	Remote System	Send a signal to car system to switch run mode - “0”: parking - “1”: run forward - “2”: run backward

Alternative Scenario: None

Exceptions: N/A

Note and issues:

3.3.2.12 Remote Light Signal

USE CASE 12 SPECIFICATION			
Use-case No.	UC012	**Use-case version**	1.0
Use-case Name	Remote Light Signal		
Author	HoangPM		
Date		**Priority**	High

Actor: User
Description: User use Remote controller to turn on/off the light signal
Goal: successful to turn on/off the light signal
Triggers:
- Tab the “light signal left” or “light signal right” button

Pre-condition:
- Remote controller is connected successful with the car system
- Remote controller has allowed to control

Post-condition:
- N/A

Main Success Scenario:

No	Actor	Action
1	User	Tab the “light signal left” or “light signal right” arrow button
2	Remote System	Transfer signal to car system - “1”: light signal left - “2”: light signal right
3	Car system	Receive the signal then turn on the light.
4	User	Tab the button again to turn off or tab the other button to change the light signal.

5	Remote system	Transfer signal to car system - "0": turn off - "1": light signal left - "2": light signal right

Alternative Scenario: None

Exceptions: N/A

Note and issues:

3.3.2.13 Remote Sound Signal

USE CASE 13 SPECIFICATION			
Use-case No.	UC013	**Use-case version**	1.0
Use-case Name	Remote Sound Signal		
Author	TuDN		
Date		**Priority**	High

Actor: User
Description: Allow use to turn on the sound signal
Goal: Successful to turn on the sound signal
Triggers:
- Hold the "Sound" button

Pre-condition:
- Remote controller is connected successful with the car system
- Remote controller has allowed to control

Post-condition:
- N/A

Main Success Scenario:

No	Actor	Action
1	User	Hold the "sound" button to turn on the sound then release the button to turn off.
2	Remote System	Transfer signal to the car system - "1": turn on - "0": turn off

Alternative Scenario: None

Exceptions: N/A

Note and issues:

3.2.3 Non-Functional Requirement

3.2.3.1 Reliability

- Mobile application and Remote Controller must work correctly with car system, have no conflict between devices.
- Only 1 android device and 1 Arduino can connect to the car system at the same time
- Car's angle, speed, run mode must be the same as the display on the mobile

3.2.3.2 Availability

- The mobile application and Remote controller are easy to connect to system to transform data.
- Power of components like Remote controller, Car System, Camera and smart phone is stable and available

3.2.3.3 Maintainability

- Microcontroller and all components can be replaced easily
- Implementation code must be follow coding standard, clearly commented for maintaining and enhancing system in the future

3.2.3.4 Performance

- System has to response one command in less than 100ms

3.3. Infrastructure and Tools

3.3.1 Hardware

No	Item
1	Raspberry Pi 3
2	Arduino WEMOS D1 R2
3	Arduino UNO R3
4	Speed-reducer Motor
5	Servo MG996R
6	PS1/PS2 motor controller
7	Webcam Logitech C170
8	Pin Cell 3.7
9	Frame mica
10	Module L9110s
11	Module XL6009

3.3.2 Software and tool

- Android Studio
- Python IDLE 3.5
- Arduino IDE
- Proteus 8
- Fritzing

IV. RESEARCH AND STUDY

4.1 Hardware study

4.1.1. Raspberry PI 3

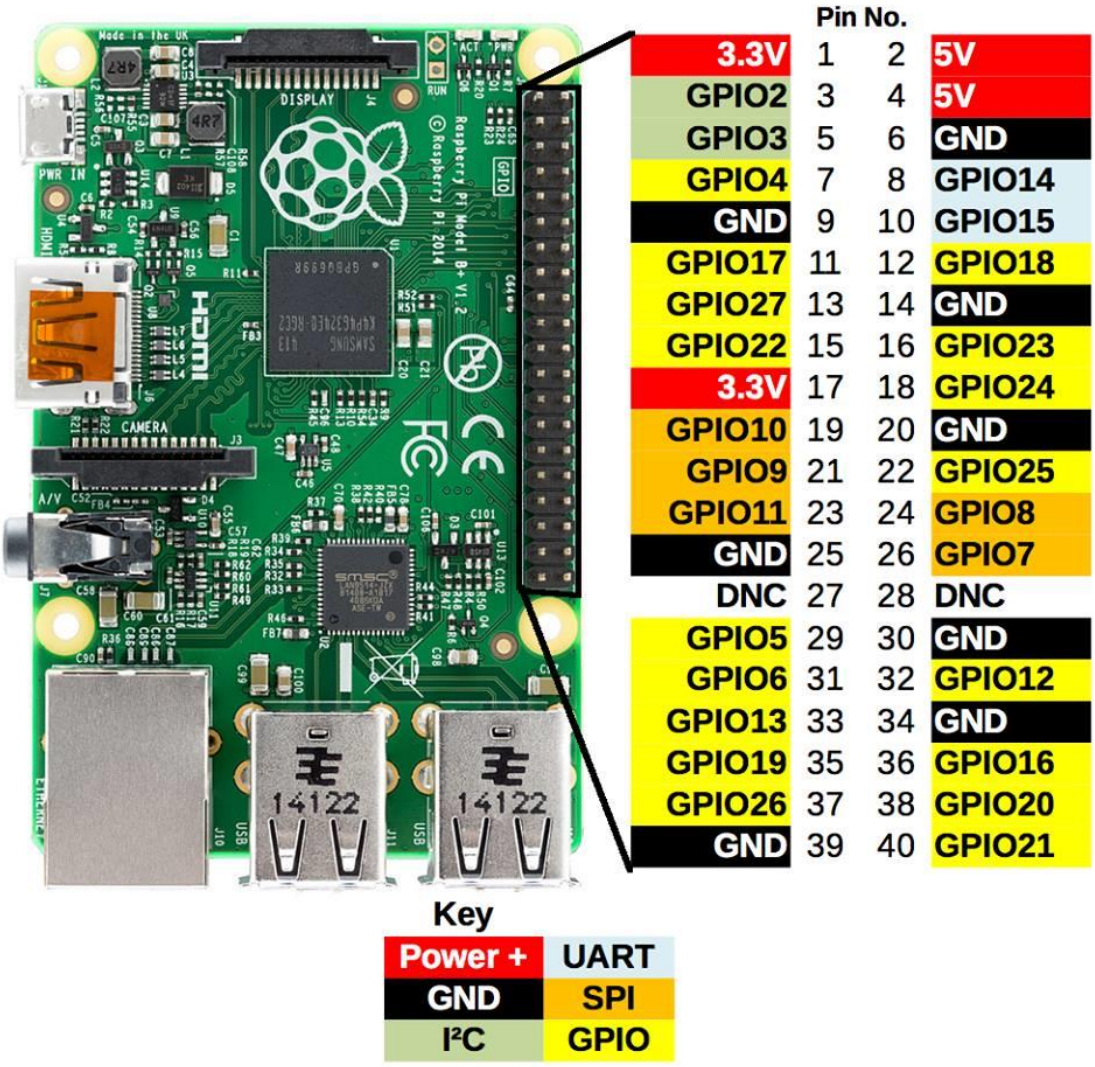


Figure 4-1: Raspberry PI

The Raspberry Pi 3 Model B is the third generation Raspberry Pi. This powerful credit-card sized single board computer can be used for many applications and supersedes the original Raspberry Pi Model B+ and Raspberry Pi 2 Model B. Whilst maintaining the popular board format the Raspberry Pi 3 Model B brings us a more powerful processer, 10x faster than the first generation Raspberry Pi. Additionally, it adds wireless LAN & Bluetooth connectivity making it the ideal solution for powerful connected designs. Connecting it to a computer with a USB cable or battery to get started simply.

Processor	Broadcom BCM2837 SoC with a 1.2 GHz 64-bit quad-core ARM Cortex-A53 processor, with 512 KB shared L2 cache
GPU	Broadcom VideoCore IV @ 250 MHz OpenGL ES 2.0 MPEG-2 and VC-1
Memory	1GB
Operating System	Boots from Micro SD card, running a version of the Raspbian operating system
Network	10/100 Mbit/s Ethernet, 802.11n wireless, Bluetooth 4.1
Memory Card Slot	Push/Pull micro SDIO
USB 2.0 port	4 (via the on-board 5-port USB hub)
GPIO Connector	40-pin 2.54 mm (100 mil) expansion header: 2x20 strip Providing 27 GPIO pins as well as +3.3 V, +5 V and GND supply lines

Table 4-1: Raspberry PI 3 Specifications

4.1.2. Arduino Uno

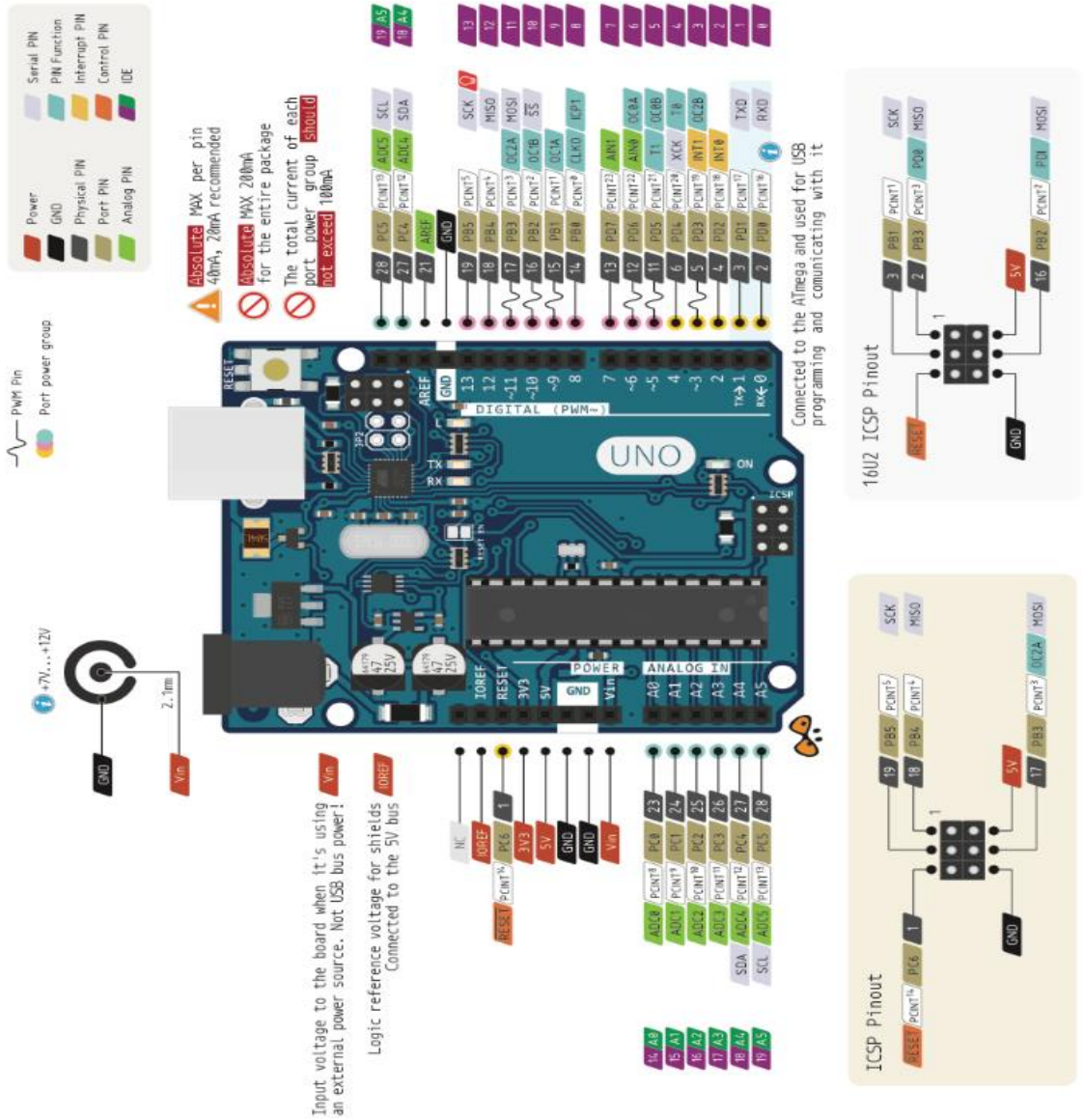


Figure 4-2: Arduino Uno

The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started.

Microcontroller	ATmega328
Architecture	AVR
Operating Voltage	5V
Flash memory	32KB of which 0.5 KB used by Boot-loader
SRAM	2KB
Clock Speed	16 MHZ
Analog I/O Pins	6
EEPROM	1 KB
DC Current per I/O pins	40 mA on I/O pins; 50 mA on 3.3V pin
Input Voltage	7 – 12 V
Digital I/O pins	20
PWM Output	6
PCB Size	53.4 x 68.6 mm
Weight	25g

Table 4-2: Arduino Uno specifications

4.1.3. Arduino Wemos-D1R2

Figure 4-3: Wemos-D1R2

Wemos-D1R2 is an ESP8266-12 based WIFI enabled microprocessor unit on an Arduino-UNO footprint

Microcontroller	ESP8266EX
Operating Voltage	3.3V
Input Voltage Range	9V to 24V
Output	5V at 1A max
Digital I/O pins	11
Analog input pins	1
Flash Memory	4MB
Board Dimensions	68.6mm x 53.4mm
Weight	25g

Table 4-3: Wemos-D1R2 specifications

4.1.4. L298N – Motor module

The Motor Shield is based on the L298, which is a dual full-bridge driver designed to drive inductive loads such as relays, solenoids, DC and stepping motors. It lets us drive two DC motors, controlling the speed and direction of each one independently.

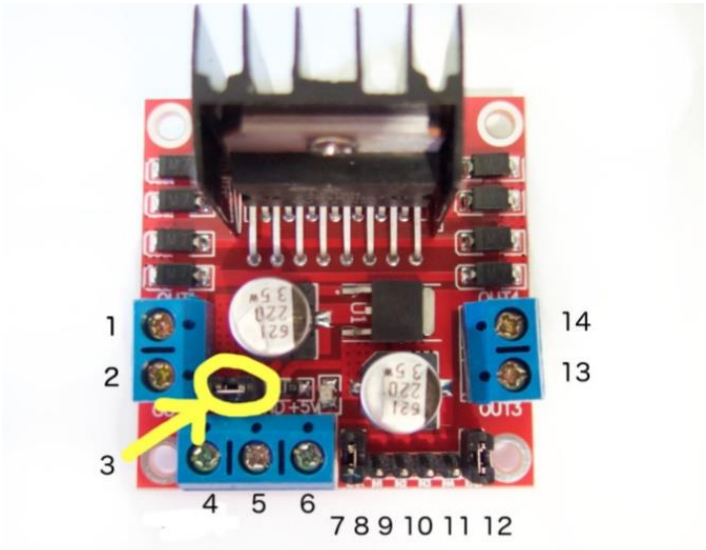


Figure 4-4: L298N pin diagrams

1	DC motor 1 "+" or stepper motor A+
2	DC motor 1 "-" or stepper motor A
3	12V jumper - remove this if using a supply voltage greater than 12V DC. This enables power to the onboard 5V regulator
4	Connect your motor supply voltage here, maximum of 35V DC. Remove 12V jumper if >12V DC
5	GND
6	5V output if 12V jumper in place, ideal for powering your Arduino
7	DC motor 1 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control.
8	IN1
9	IN2
10	IN3
11	IN4
12	DC motor 2 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control.
13	DC motor 2 "+" or stepper motor B+
14	DC motor 2 "-" or stepper motor B

Table 4-4: L298N Pin-out

- **Specification**
  - Operating Voltage: 4V ~ 35V
  - Motor controller L298N, Drives 2 DC motors or 1 stepper motor
  - Max current: 2A per channel or 4A max
  - Free running stop and break function

- Summary Chip: ST L298N
- Logic power supply: 5V
- Max power: 25w
- Weight: 35g
- Size: 55mm x 60mm x 30mm
- Storage temperature: -25ºC ~ +135ºC

4.1.5. Module XL6009

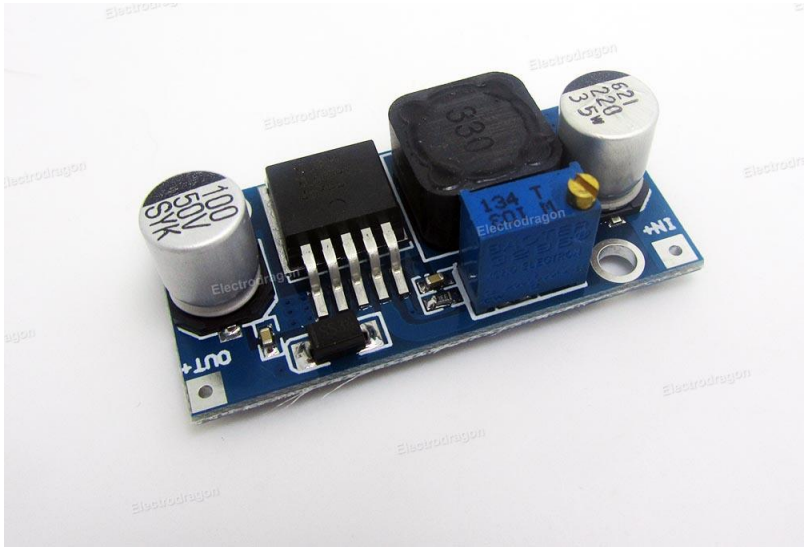


Figure 4-5: Module XL6009E1

The module uses the second generation of high-frequency switching technology XL6009E1 core chip performance than the first-generation technology. XL6009 boost module has superior performance than LM2577 based modules.

Input voltage	3.2V ~ 32V
Output Voltage	4V ~ 38V
Modules Type	Non-isolated step-up (Boost)
Rectification	Non-synchronous rectification
Rated output current	3A
Peak Output Current	4A
Module size	43mm x 21mm x 14mm
Input mode	IN + input positive level, IN-input negative
Conversion efficiency	Up to 94%

Table 5-5: XL6009 specifications

4.1.6. Charging circuit TP4056

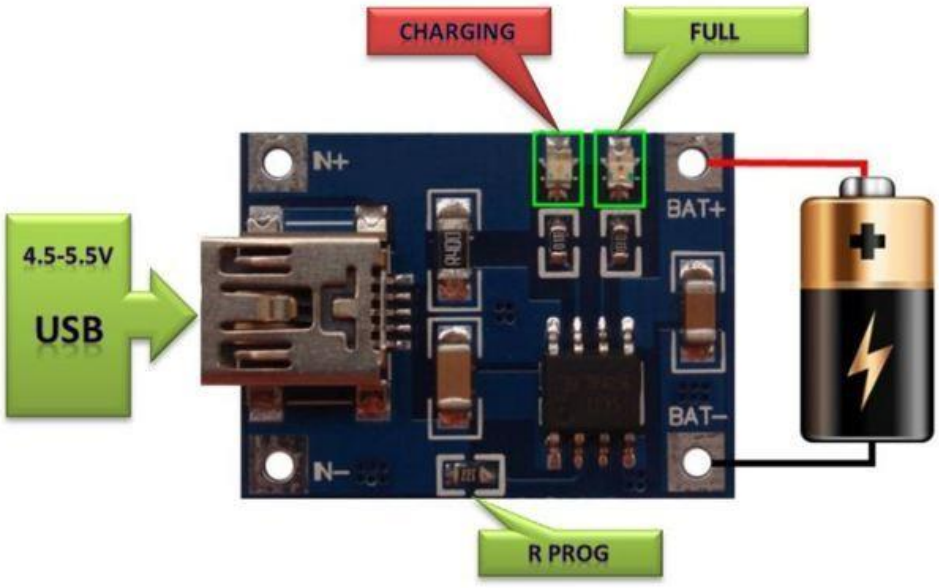


Figure 4-6: Charging circuit

This tiny module is perfect for charging single cell 3.7V 1Ah or higher LIPO cells such as 16550s that don't have their own protection circuit. Based around the TP4056 charger IC and DW01 battery protection IC this module will offer 1A charge current then cut off when finished. Furthermore, when the battery voltage drop below 2.4V the protection IC will switch the load off to protect the cell from running at too low of a voltage – and also protects again over voltage and reverse the polarity connection.

Input voltage	4.5 ~ 5.5V
Full charge voltage	4.2V
Operating ambient temperature range	-10ºC ~ +85ºC
Charging Current	1A (by default)

Table 4-6: Specifications

4.1.7. Gear speed-reducer motor

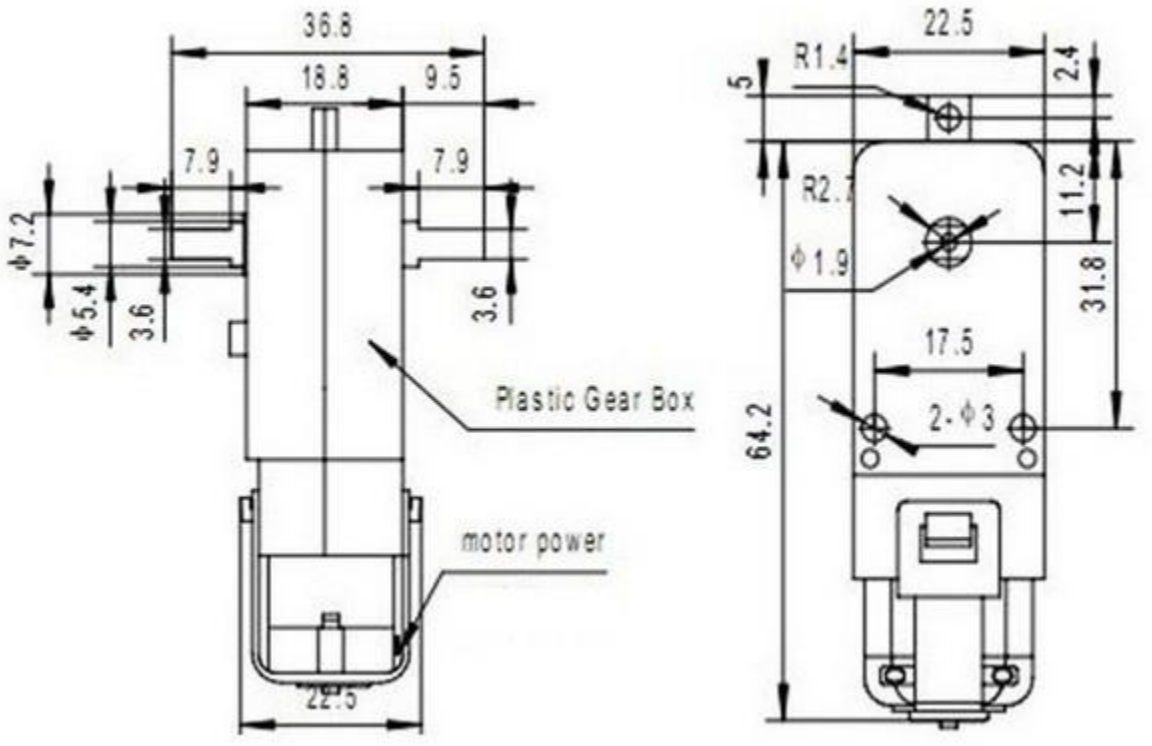


Figure 4-7: Gear speed-reducer motor

- **Specifications:**
  - **Input voltage: 3~9V**
  - **Input current: 150 ~ 200mA**
  - **Transfer rate: 1/40**

V	Current Consumption	Rounds per second
3V	≤200mA	200 ± 10%
6V	≤150mA	90±10%

Table 4-7: Details

4.1.8. Servo MG996R

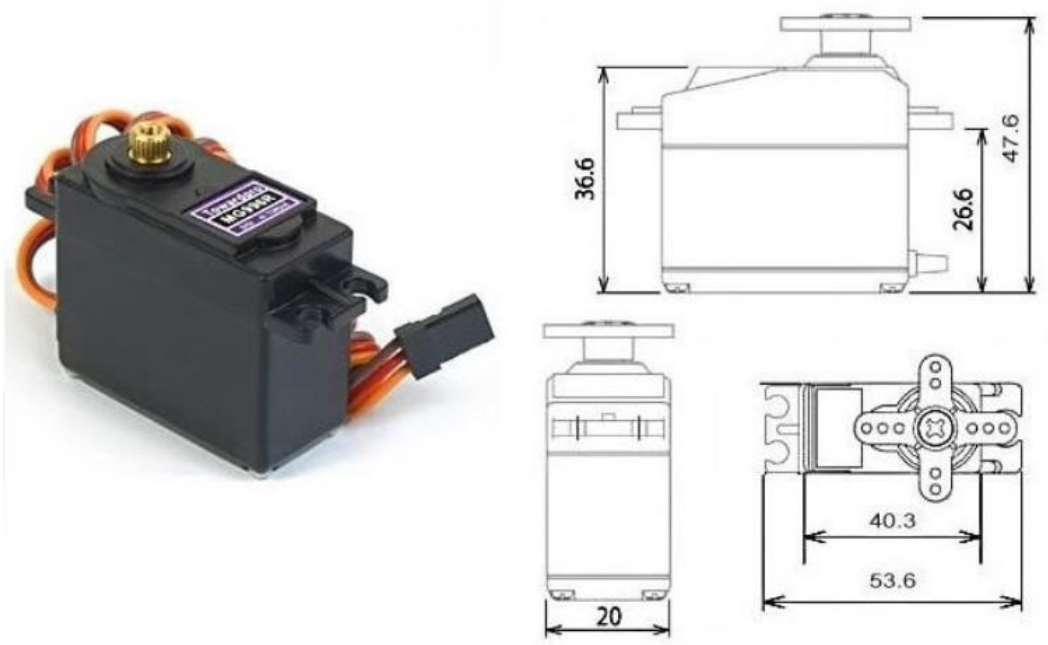


Figure 4-8: MG9960R

This High-Torque MG9660R digital servo features metal gearing resulting in extra high 10kg stalling torque in a tiny package.

Weight	55g
Dimension	40.7 x 19.7 x 42.9mm approx.
Stall torque	9.4 kg/cm (4.8V), 11 kg/cm (6V)
Operating speed	0.17 sec/60º (4.8V), 0.14 sec/60º (6V)
Operating Voltage	4.8V ~ 7.2V
Running current	500mA ~ 900mA (6V)
Stall current	2.5A (6V)
Deal band width	5µ
Temperature range	0ºC ~ 55ºC

Table 4-8: MG9960R specifications

4.1.10. Camera Logitech C170

Figure 4-10: Logitech C170

- **Specifications**
  - Supports VGA 640 x 480 Video Calling
  - XVGA Video Capture Up to 1024 x 768
  - Take 5MP photos
  - Logitech Fluid Crystal Technology
  - Built-in microphone with Noise Reduction
  - Universal Clip for Laptops & Monitors
  - Hi-Speed USB 2.0 Certified
  -

4.1.11. Module DC-DC Step-Down Voltage converter – MP1584

This DC/DC Step-Down voltage converter is based on MP1584, it converts input voltage between 4.5V and 28V into a smaller voltage between 0.8V and 18V, capable of driving a 3A load with excellent line and load regulation.

Input Voltage	4.5V ~ 28V
Output Voltage	0.8V ~ 18V
Continuous Output Current	3A Max
Peek Output Current	4A
Max. Efficiency	92%
Switching Frequency	100kHz to 1.5MHZ
Dimensions	22mm x 17mm x 4mm

Table 4-10: Specifications

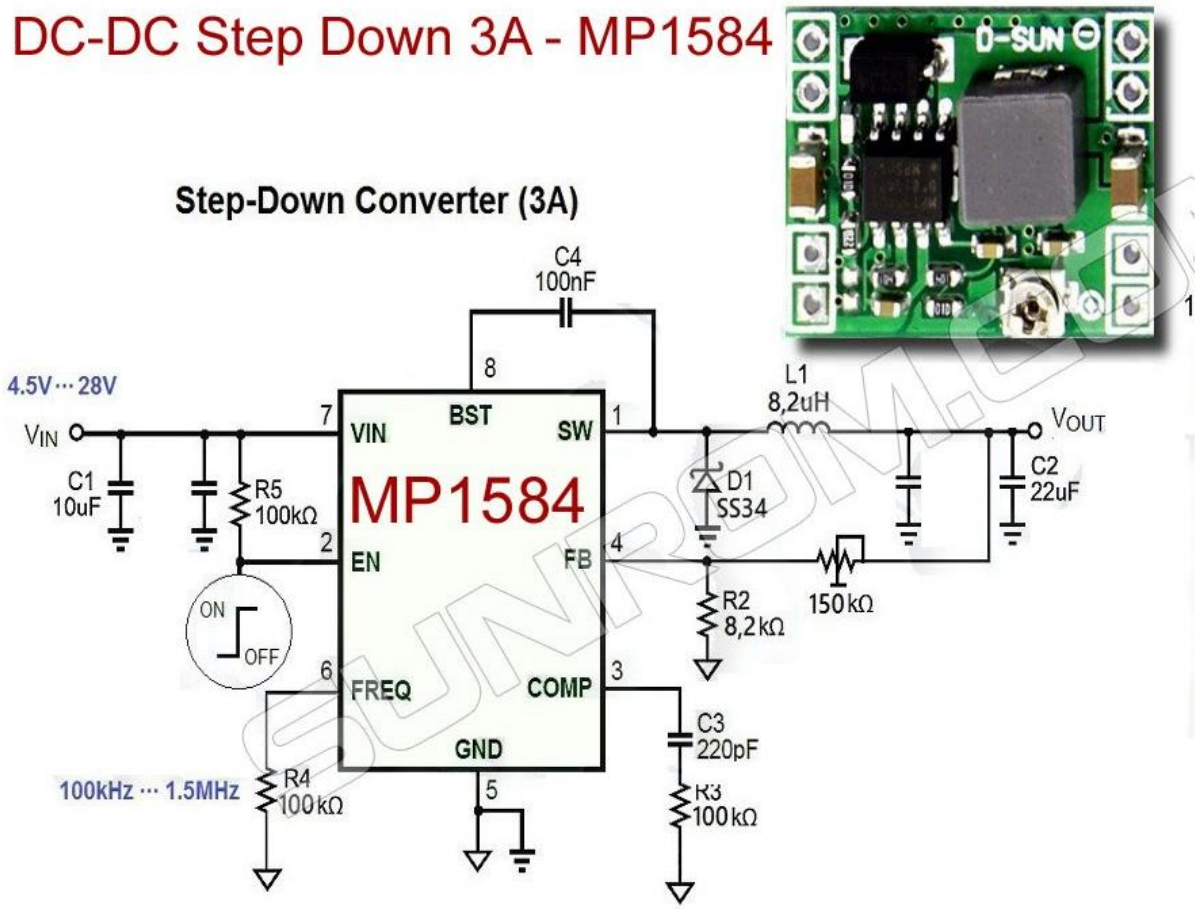
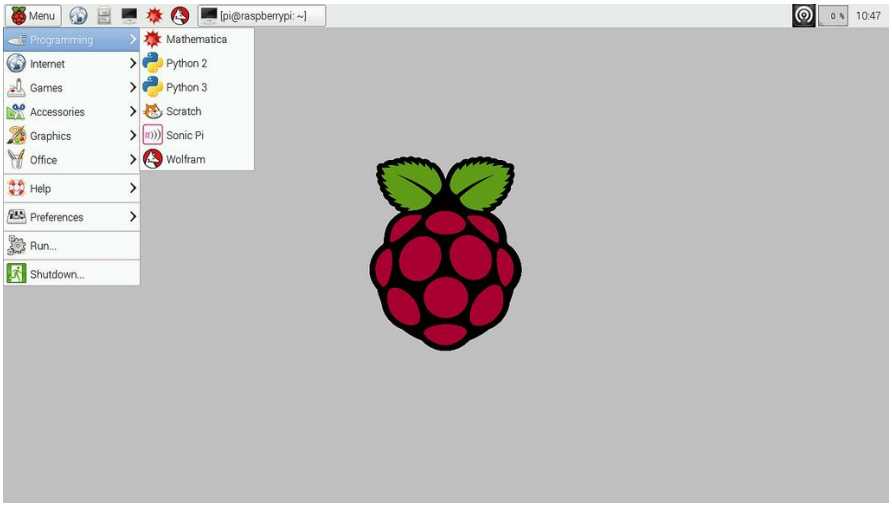


Figure 4-11: DC-DC Step-Down Converter

4.2. Software Study

4.2.1. Raspbian



Raspbian has been the standard Raspberry Pi operating system, more commonly referred to as a "Distro", since the Pi arrived in 2012 and we have seen it grow over time into the capable Distro that we use today. Raspbian is a fork of the massively popular Debian distribution and it is jointly maintained by the Raspberry Pi Foundation and the community.

We choose this Operation system because: Being based on Debian, Raspbian comes with the APT (Advanced Packaging Tool) as it's package manager, which is used to install software from the vast Raspbian repositories, but Raspbian also comes with raspi-config, a menu based tool that simplifies the act of managing Raspberry Pi configurations such as setting up an SSH, overclocking and enabling the official Raspberry Pi camera.

We now have a simplified and refined interface that groups applications and configuration tools into clearer categories. As Raspbian is the default distro for Raspberry Pi it is also the distro that sees the most improvement and innovations examples of this are the RPi.GPIO library that enables Python to talk to the GPIO (General Purpose Input Output) pins.

- **Feasibility of this choose:**
  - Easy-to-use interface
  - Large supportive community
  - Stability for taking first step with the Raspberry PI

4.2.2. Python

Python is an object-oriented, high-level, interpreted programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Raspberry Pi Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

- **Feasibility of this choose**
  - Easy to learn syntax
  - Reduces the cost of program maintenance
  - Encourages program modularity and code reuse
  - There is no compilation step, the edit-test-debug cycle is incredibly fast

V. SYSTEM DESIGN DESCRIPTION (SDD)

5.1. Architecture design

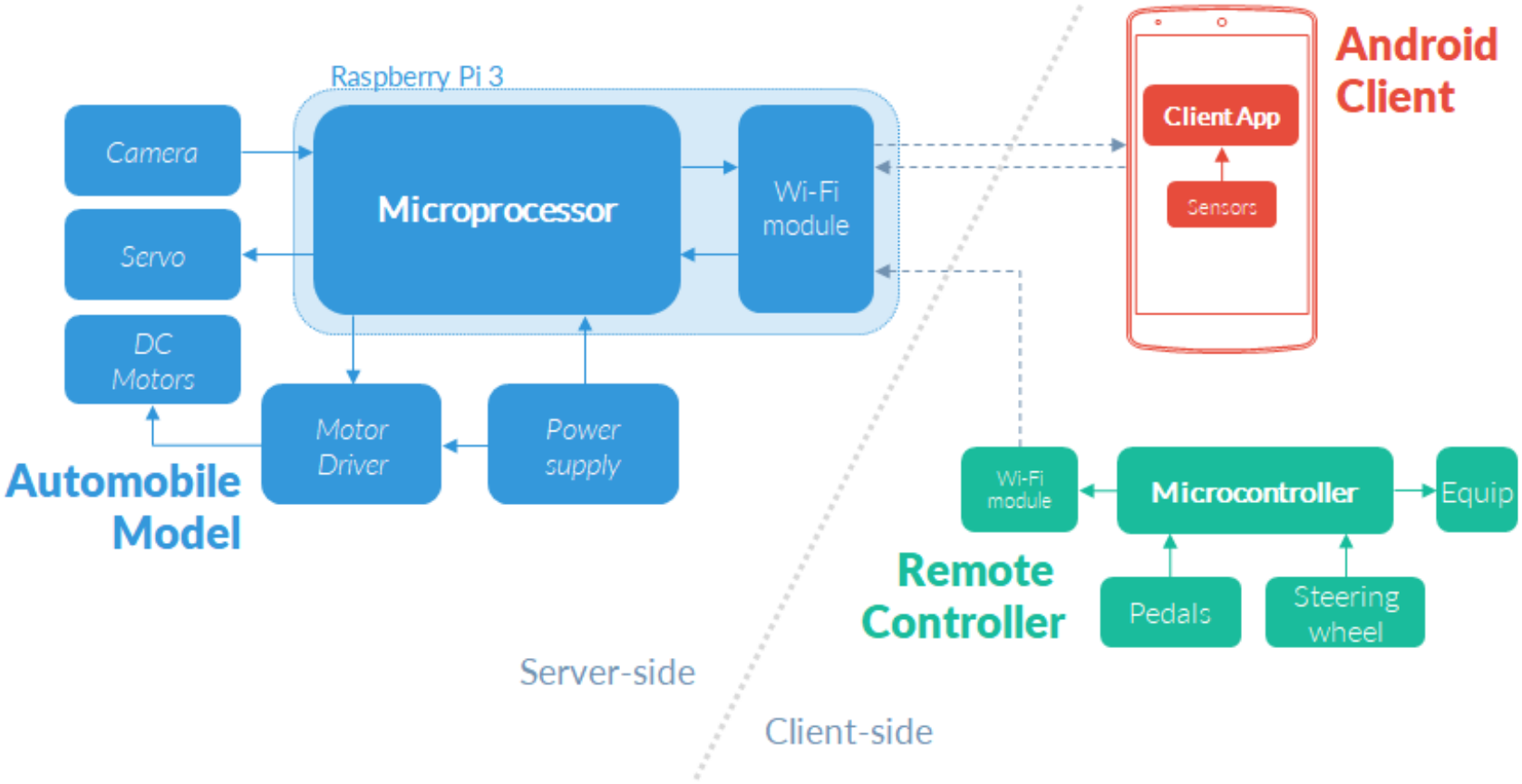


Figure 5-1: Architecture design

5.2. Sequence diagram

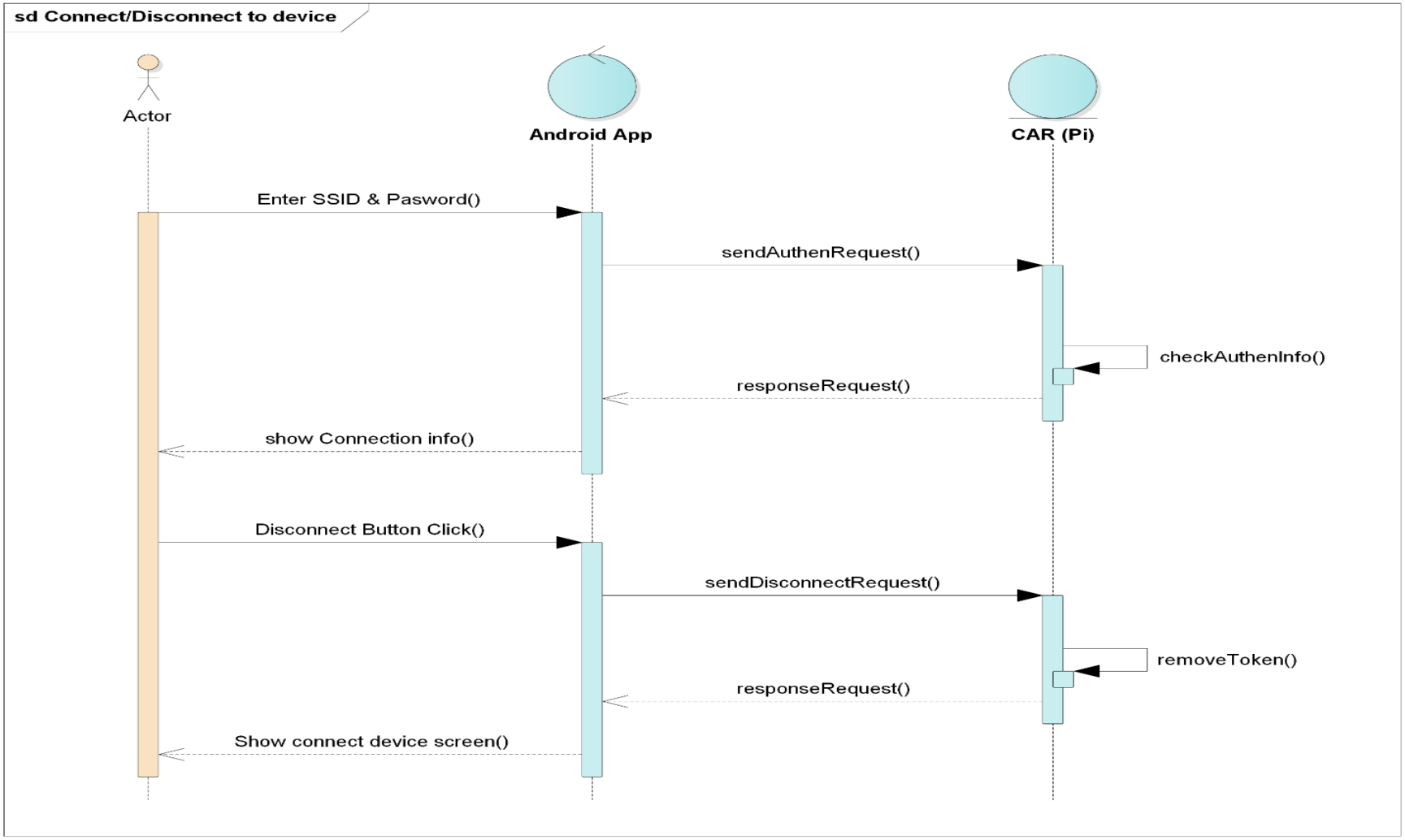


Figure 5-1: Connect/Disconnect to device

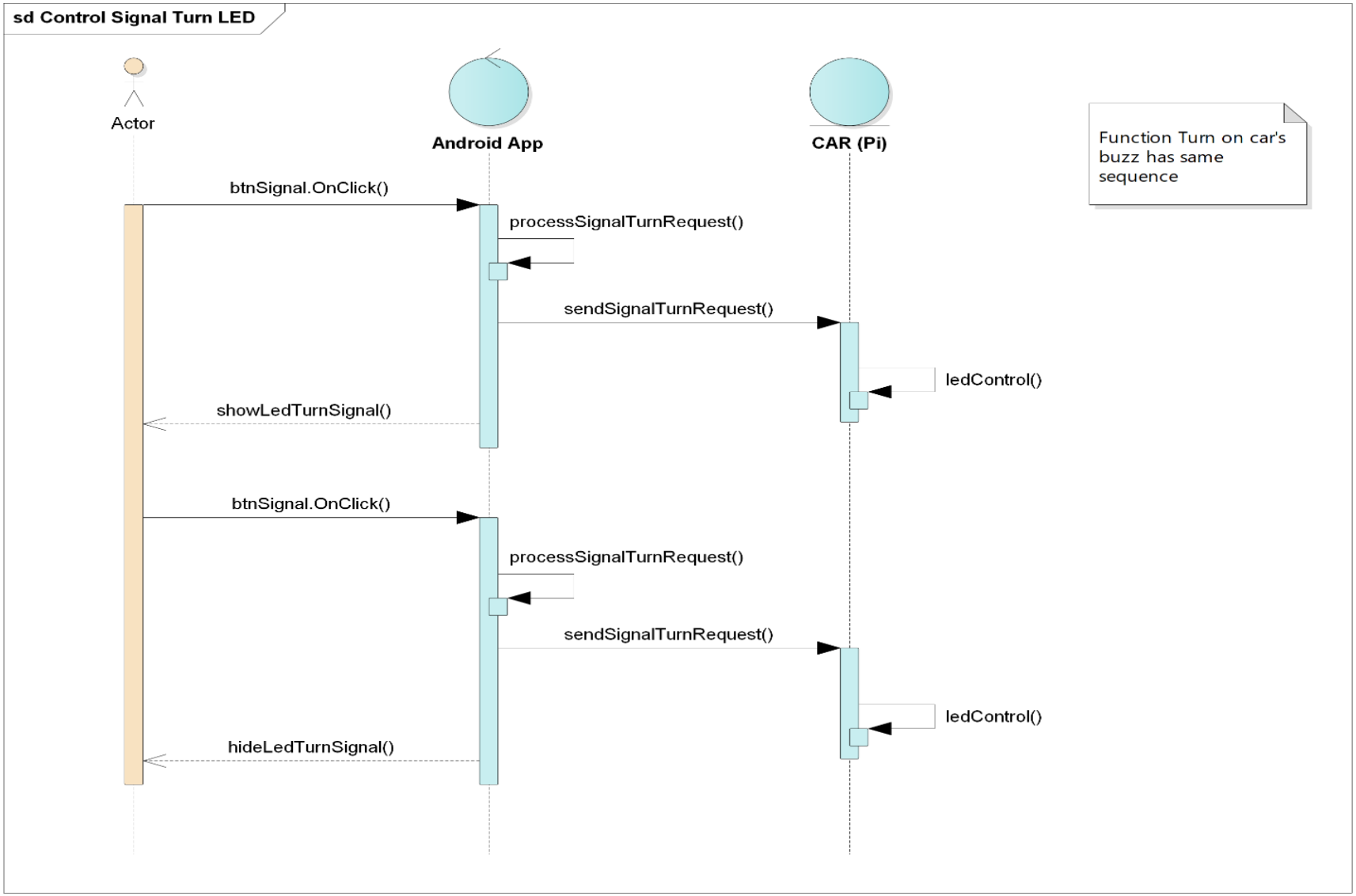


Figure 5-2: Control signal turn LED

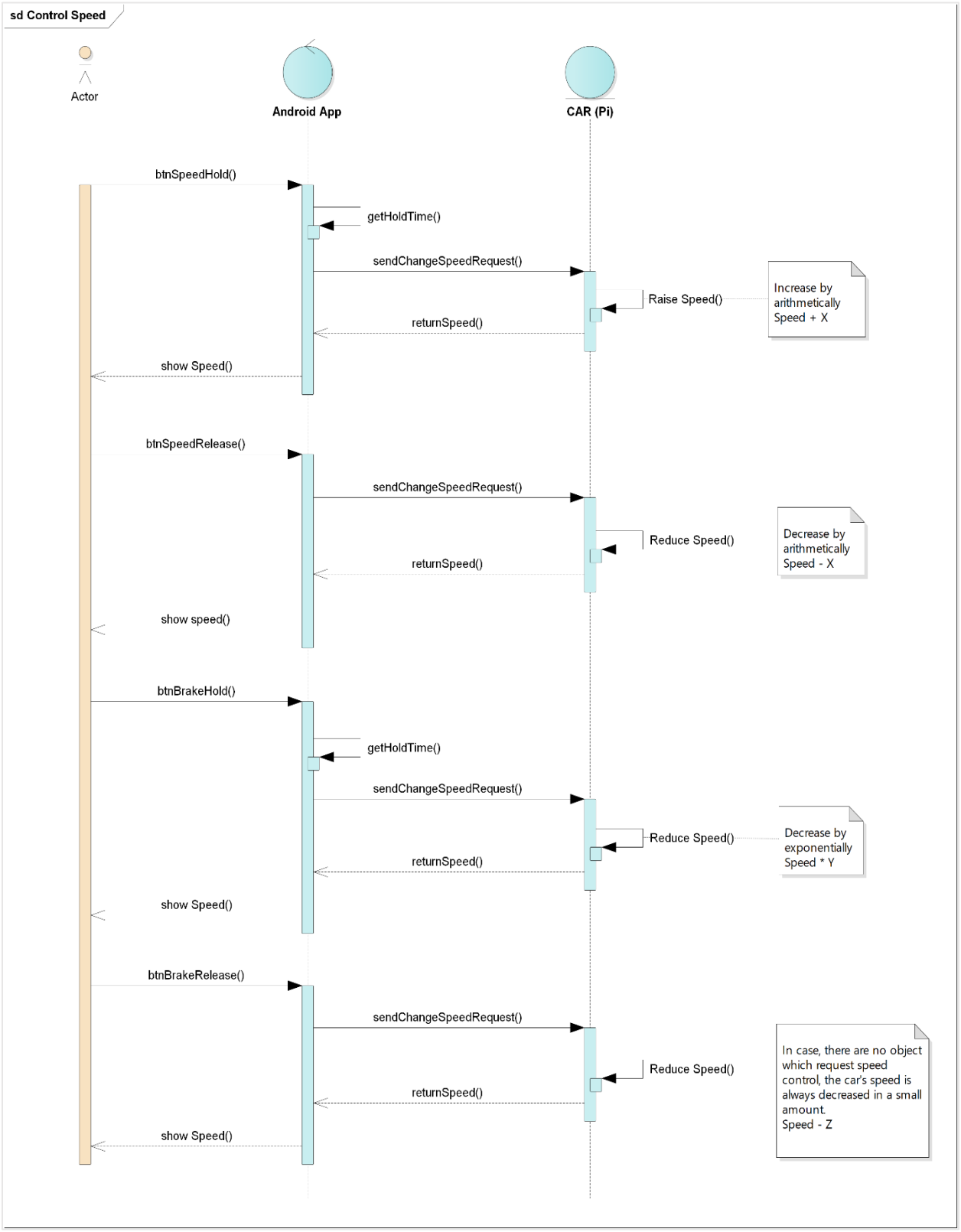


Figure 5-3: Control Speed

5.2. Flowchart

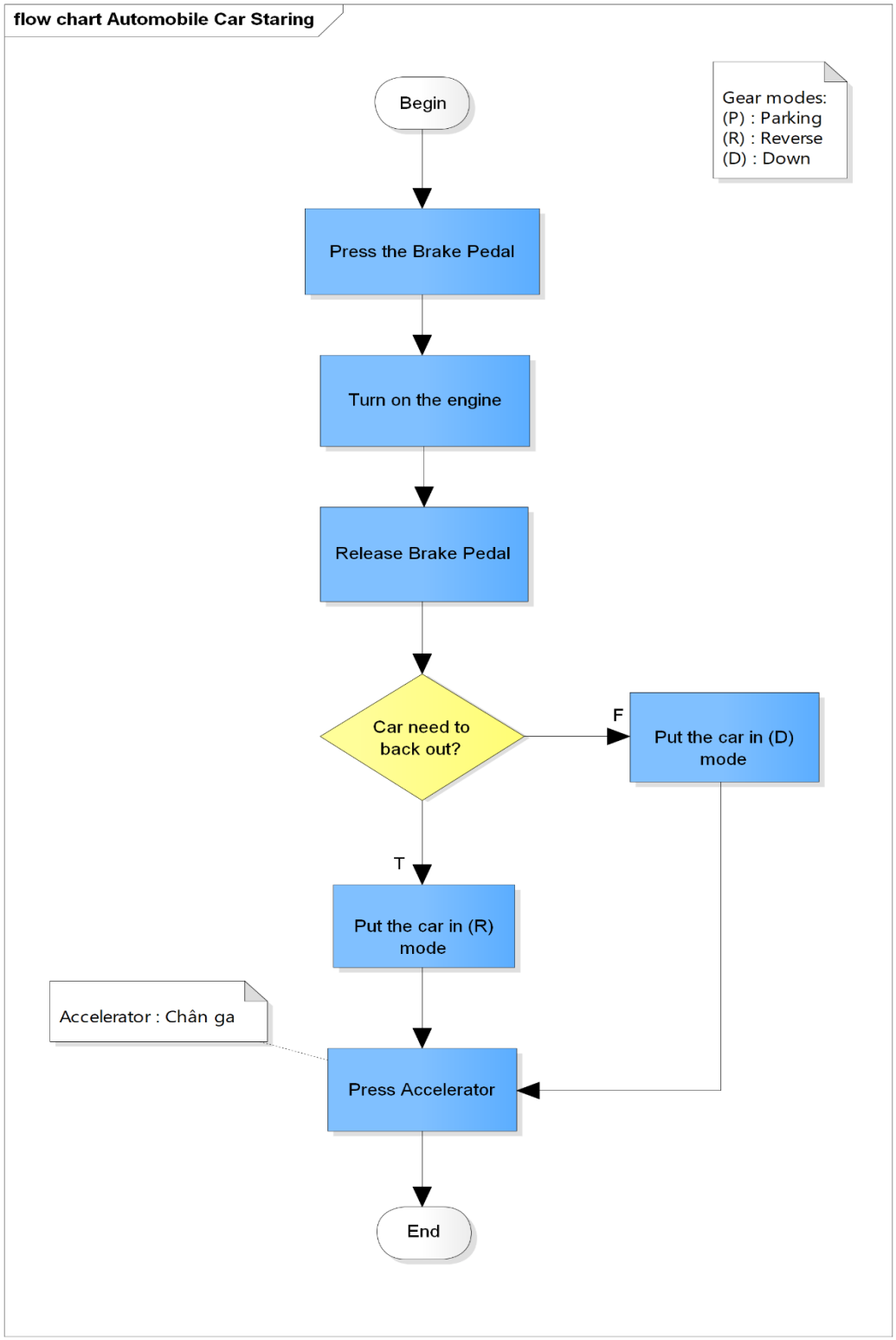


Figure 5-4: Automobile car starting

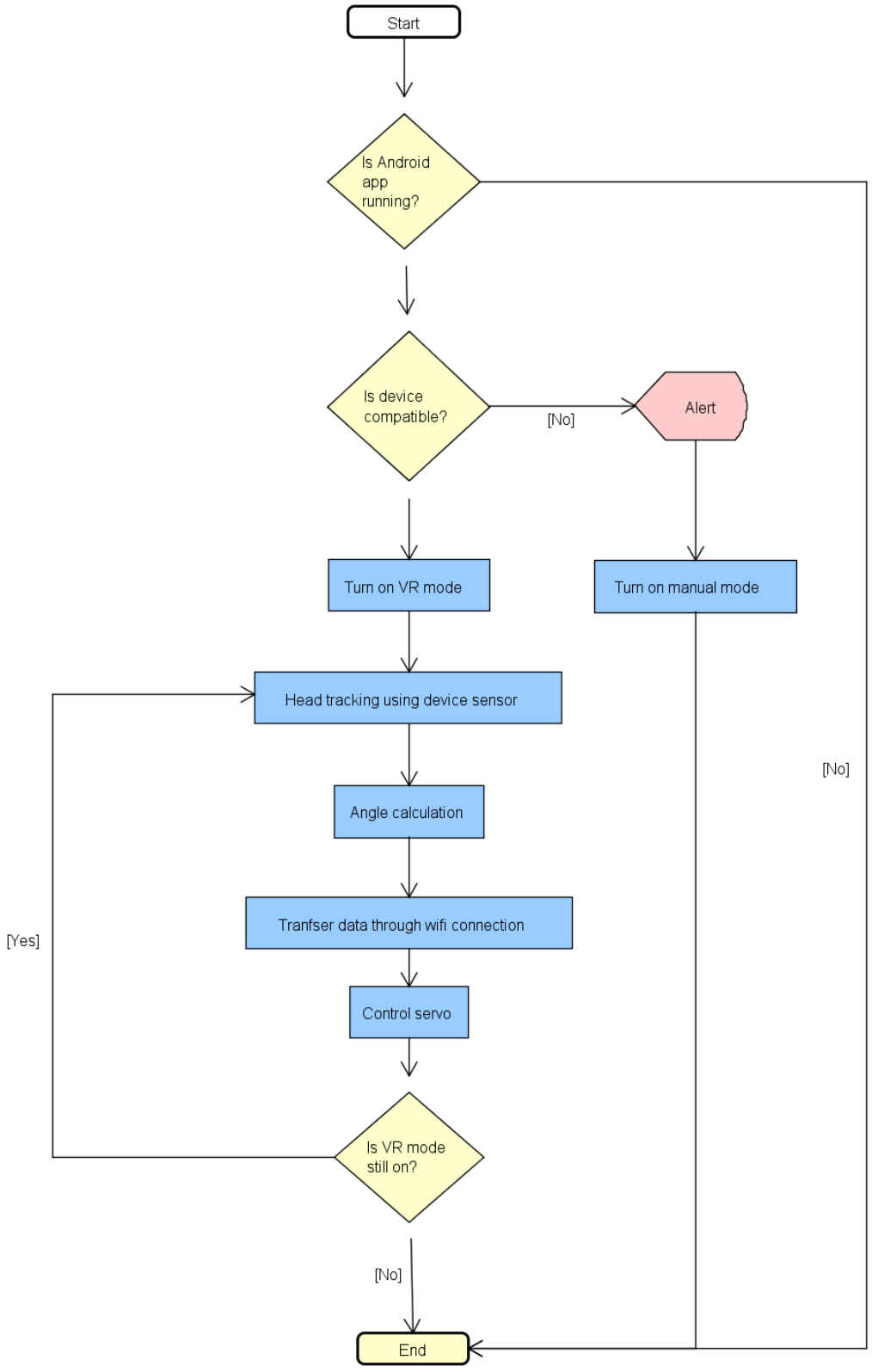


Figure 5-6: Handling Sensor

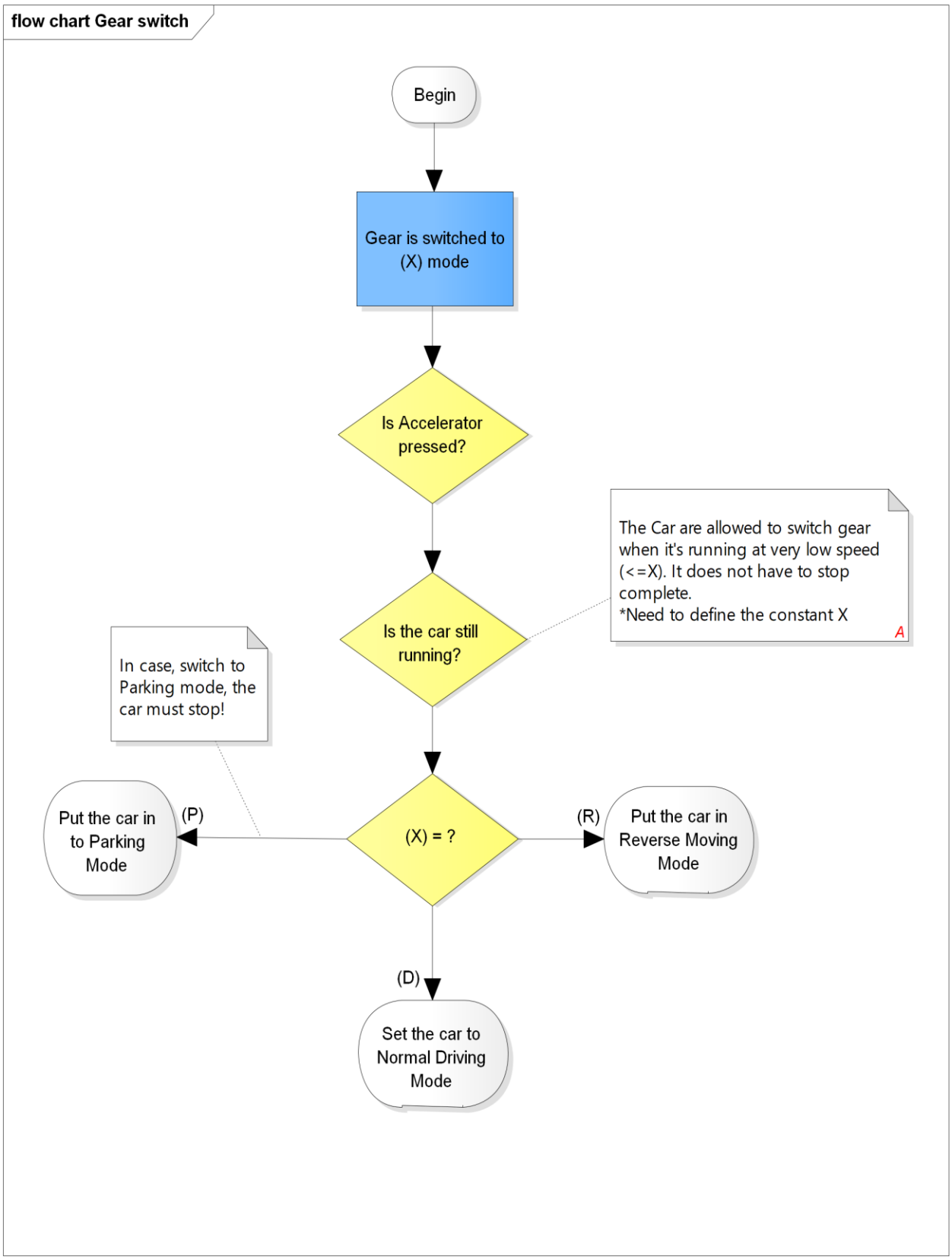


Figure 5-7: Gear Switcher

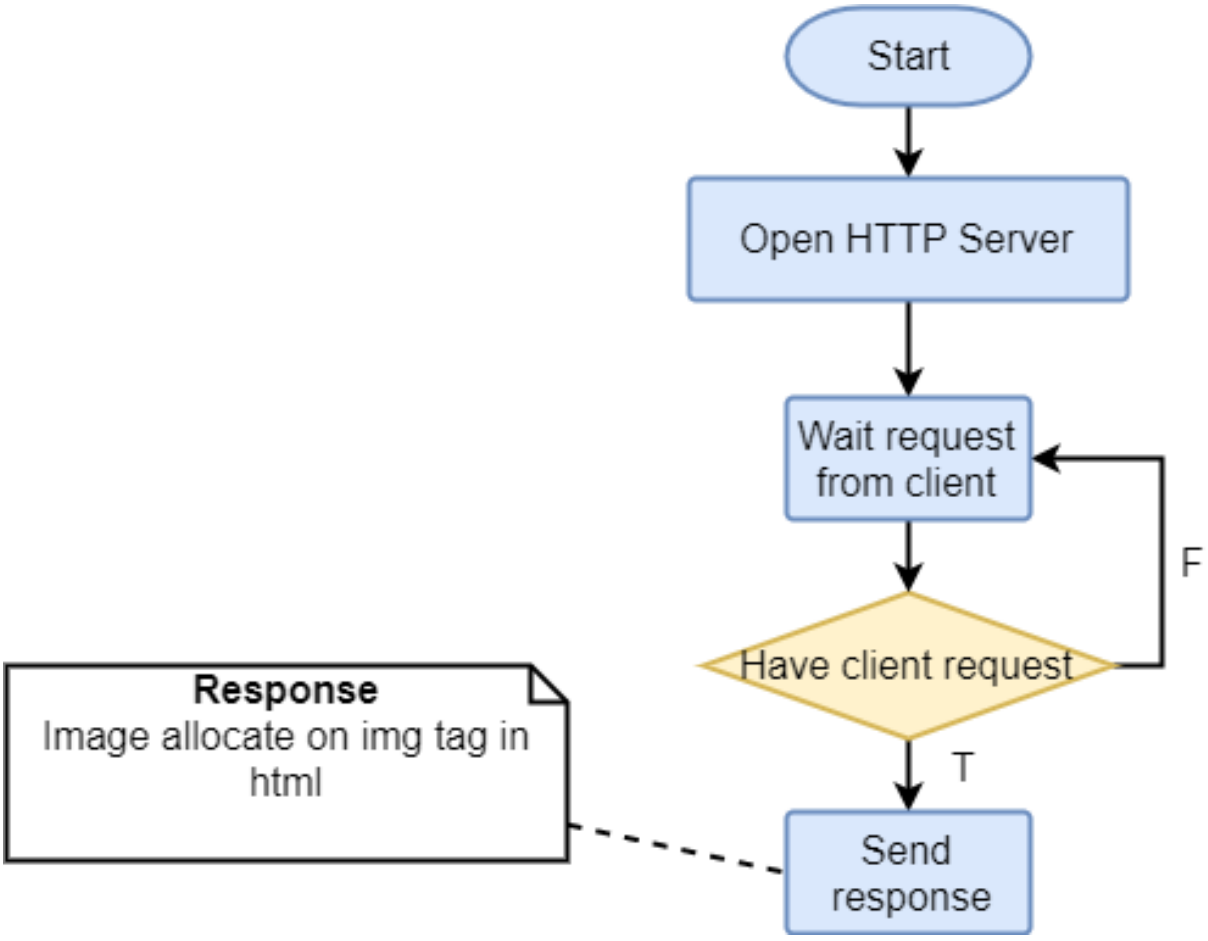


Figure 5-8: Stream Video

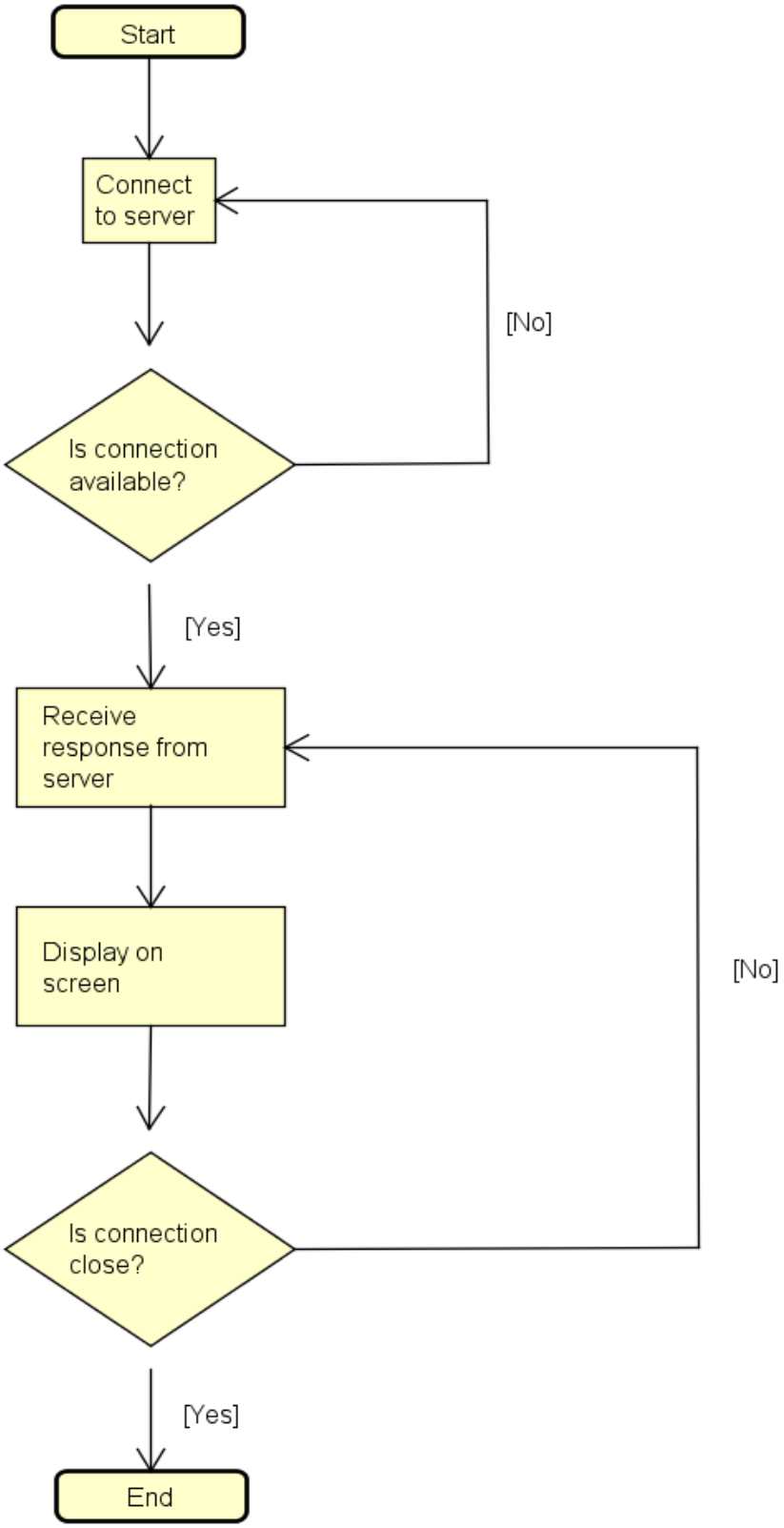


Figure 5-9: Android Steam work flow

VI. IMPLEMENTATION & TESTING

6.1. Analysis and selection of tools, devices

6.1.1. Motor controller

We used 2 motor combined with L9110s module to control car movement. The module will control 2 motor with customize speed.

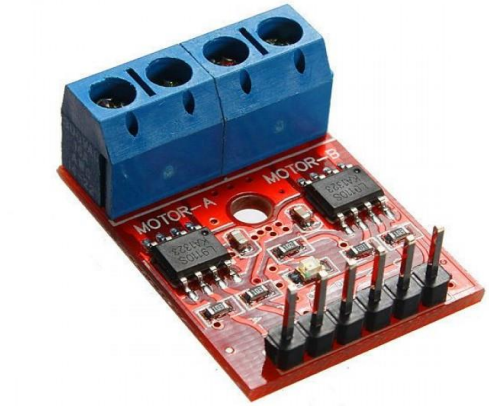


Figure 6-1: Module L9110s

Figure 6-1: Motor

When the car go forward or backward, two motor will turned the same direction and speed.

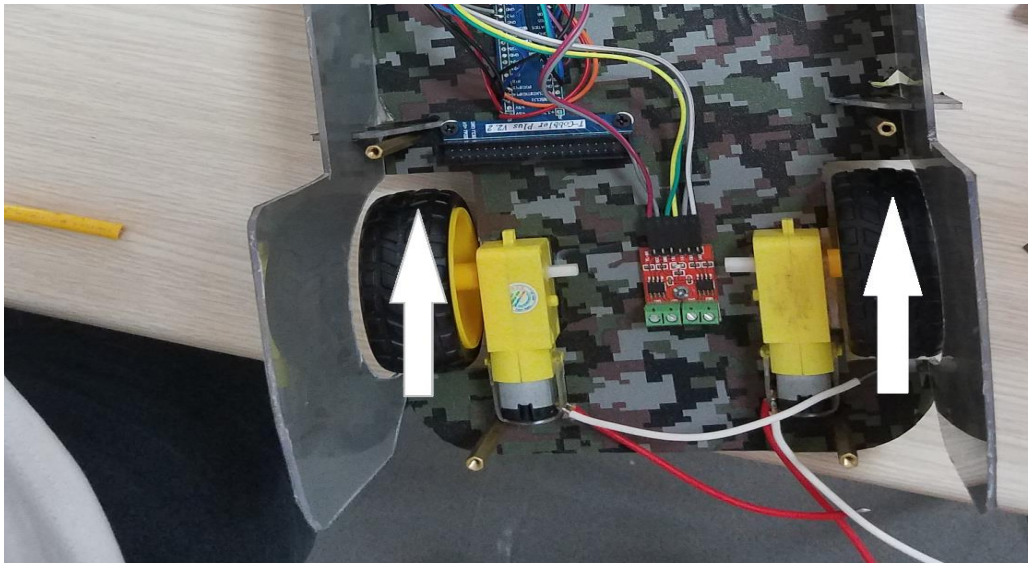


Figure 6-3: Run forward

6.2.2. Servo car

We used servo (MG996R) to control the car. When the car turn left (or right), the servo will turn into the opposite side (right or left).We also set the swivel angle of servo between 45 to 135

Figure 6-4: Servo MG996R

Figure 6-5: Turn left

6.2.3. Servo camera

We used servo (MG996R) to control the camera (Webcam Logitech C170). The servo combine with the Aluminum shelf to hold the camera. When driver turn Android device to left (or right), servo will turn in the same direction.

Figure 6-6: Servo camera

6.2.4. Remote Controller

We use Arduino Uno to process data combine with WEMOS to transfer data to Raspberry server through WIFI; we also use mechanical parts of Panther-Lord (Car racing game control) to control the car in Remote Controller mode. The Remote include 3 parts:

- Steering: wheel, light and sound signal button
- Transmission: Gear lever, hand-break
- Pedals: throttle pedal, break-pedal

Figure 6-7: Remote controller

6.2.5. Supply power for car system

We use LGDB118650 pin cell, a rechargeable battery that can be use and recharge many time. We combine LGDB118650 with module XL6009 to convert voltage into 20V that suitable and powerful enough for the car system; and also use module DC-DC step-down converter MP1584 to convert voltage into 5V - suitable for Raspberry PI. Besides, we use Charging circuit TP4056 to help user recharge the pin cell in the easiest way.

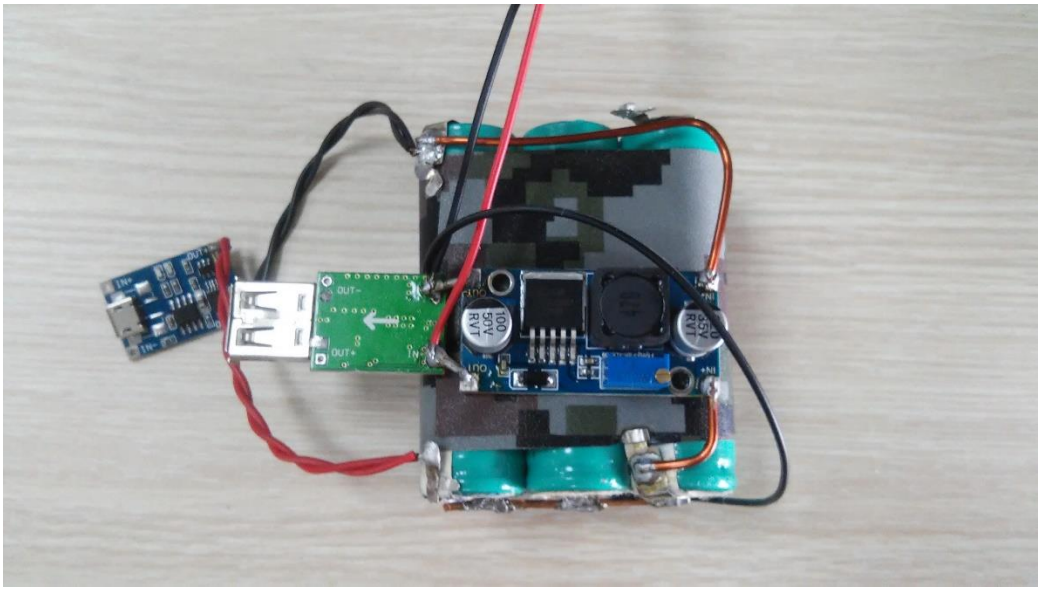


Figure 6-8: Power supply for car system

6.3. Hardware design

6.3.1. Raspberry PI connector

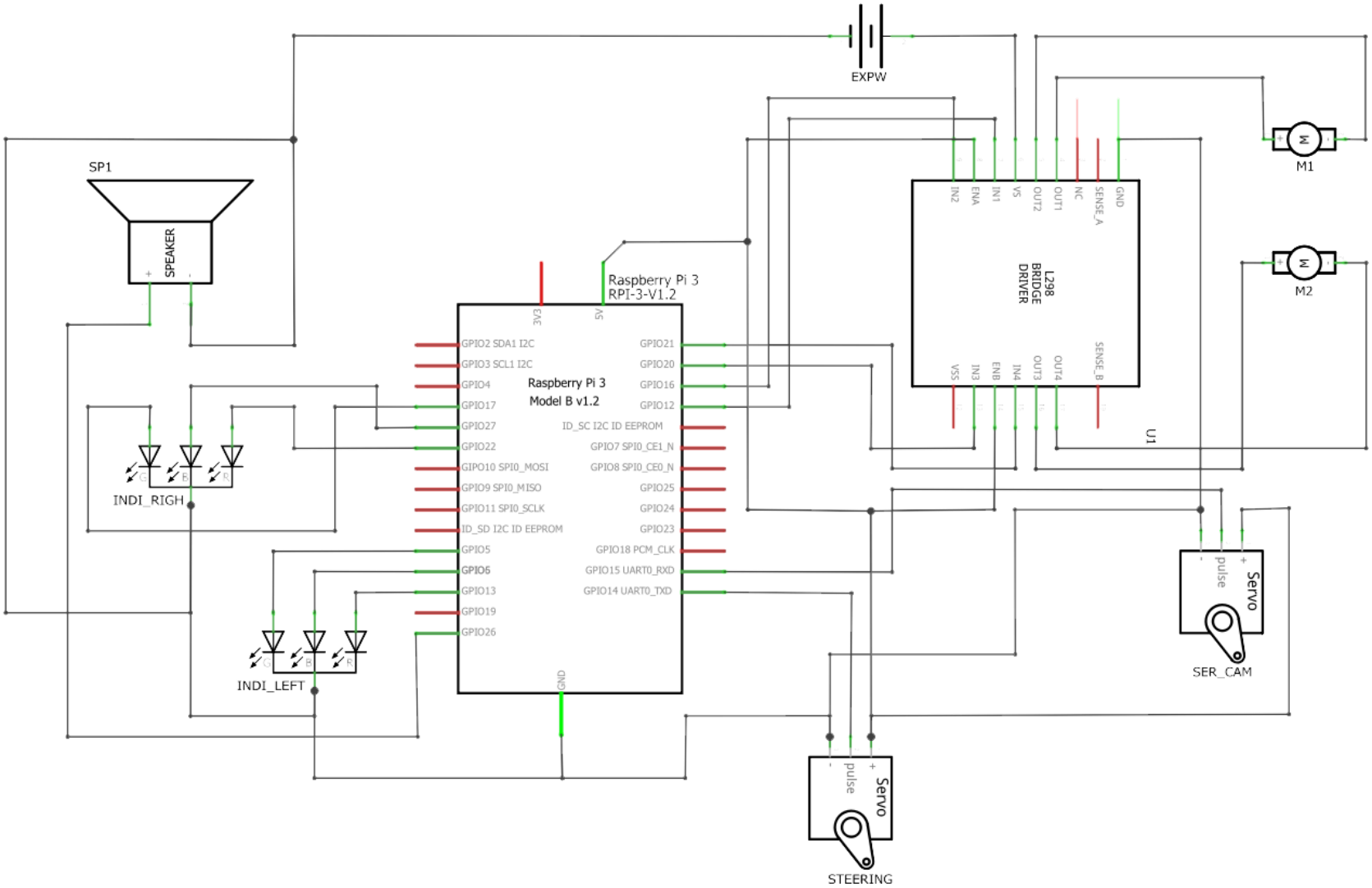


Figure 6-9: Raspberry PI connector schematic

6.3.2. Remote controller

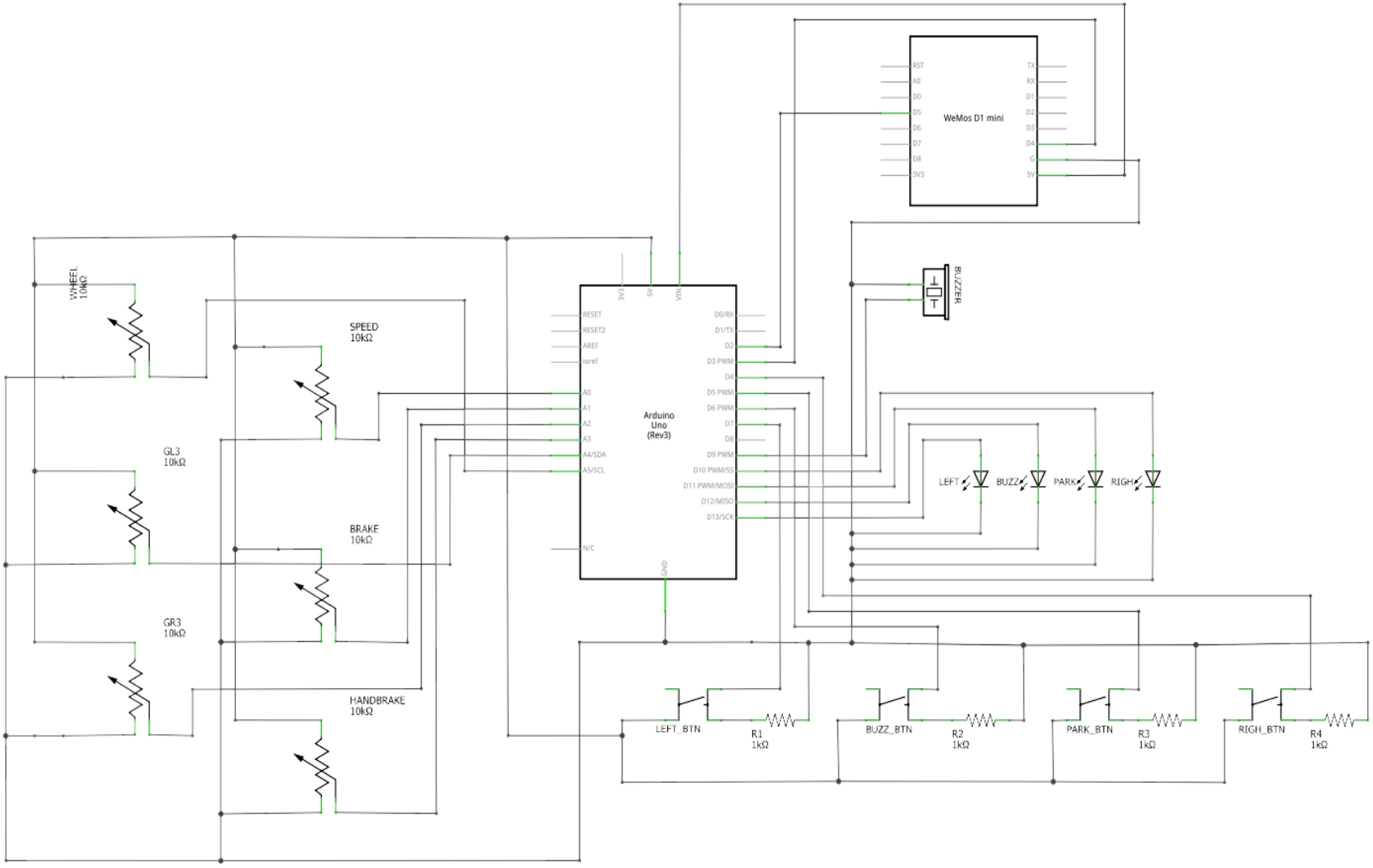


Figure 6-10: Remote Controller schematic

6.4. Software design

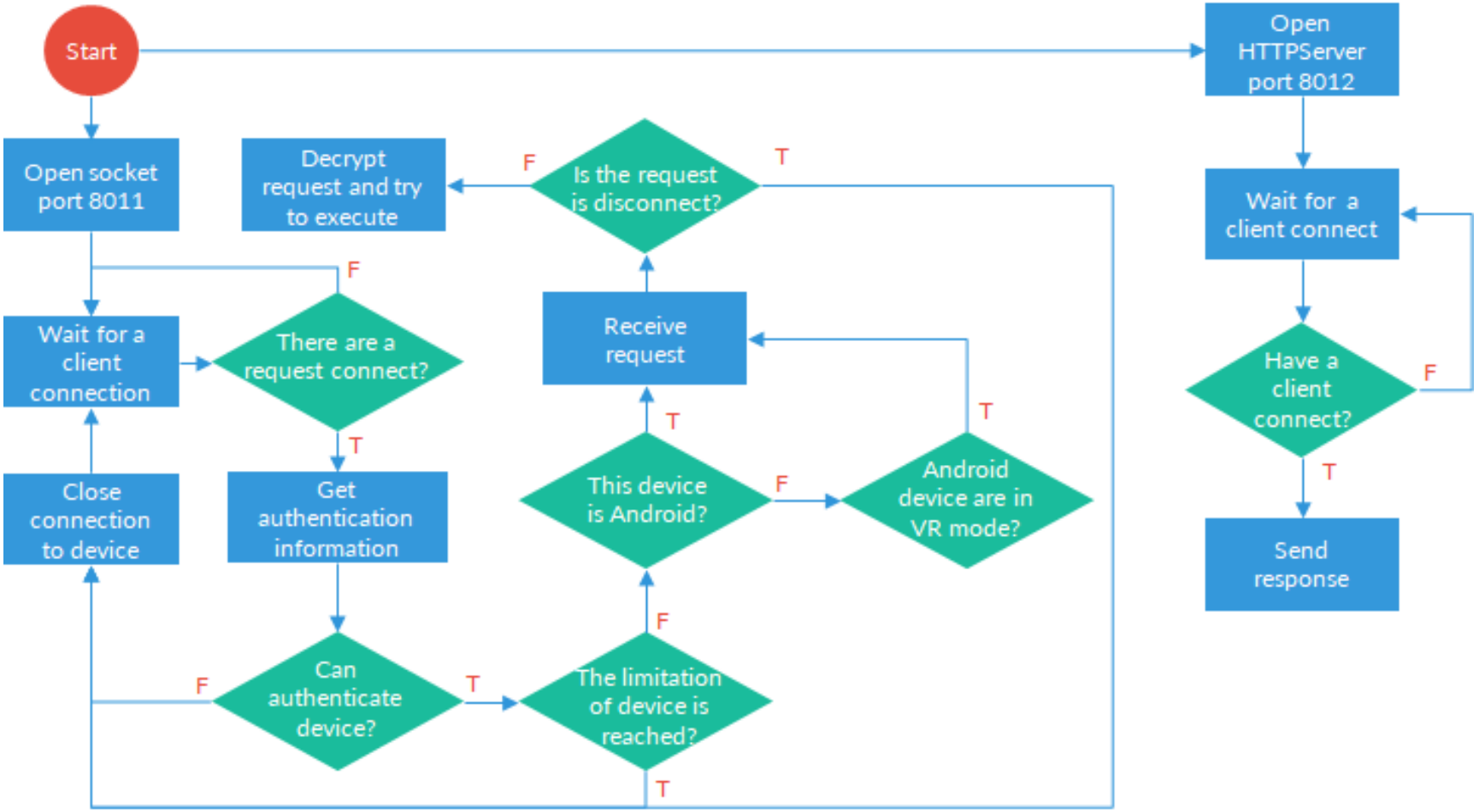


Figure 6-11: Software design

6.5. Mechanical Design

The Car design is based on Audi A1 Sport-back dimension (with ratio: 1/10).

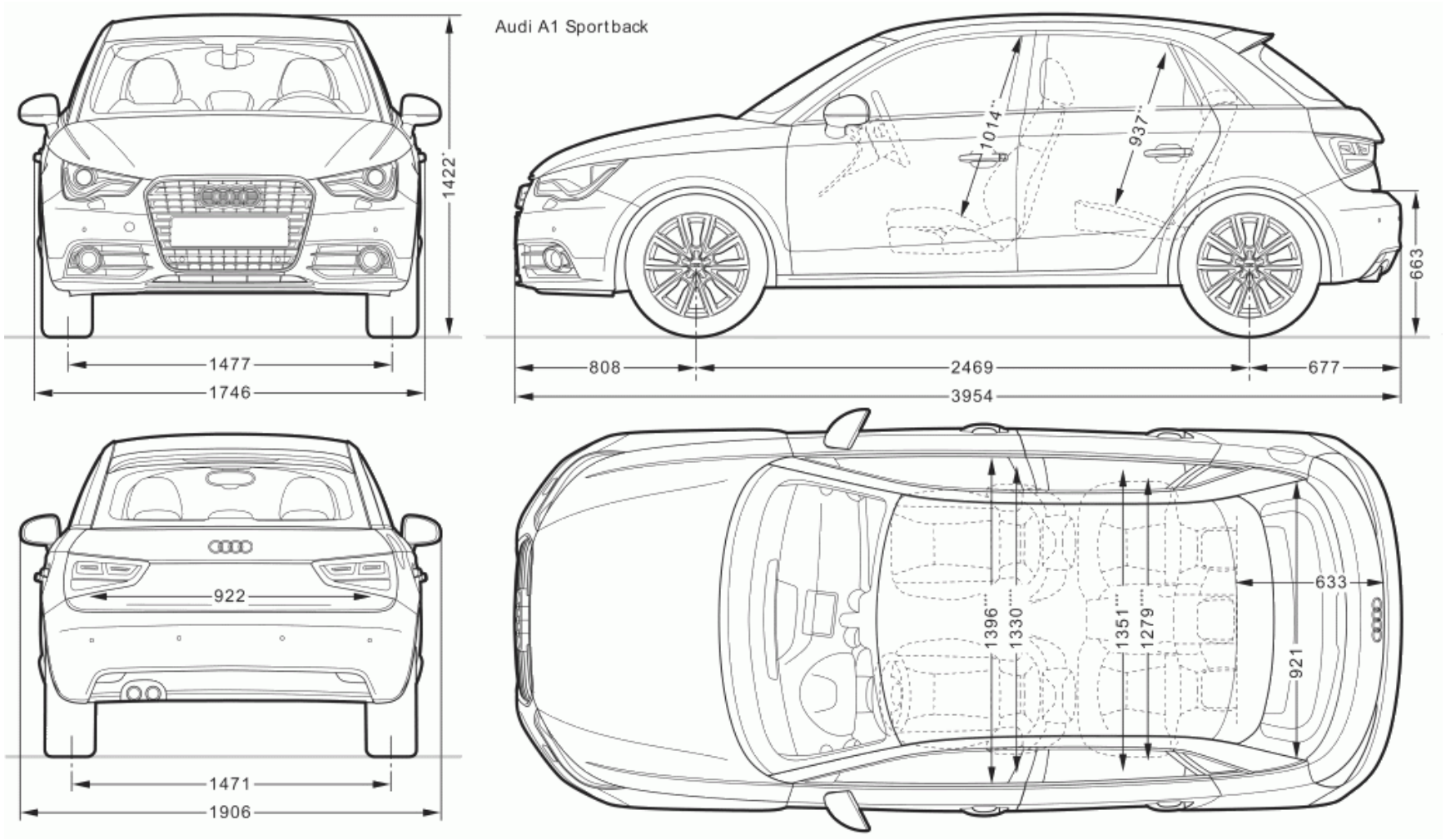


Figure 6-12: Audi A-1

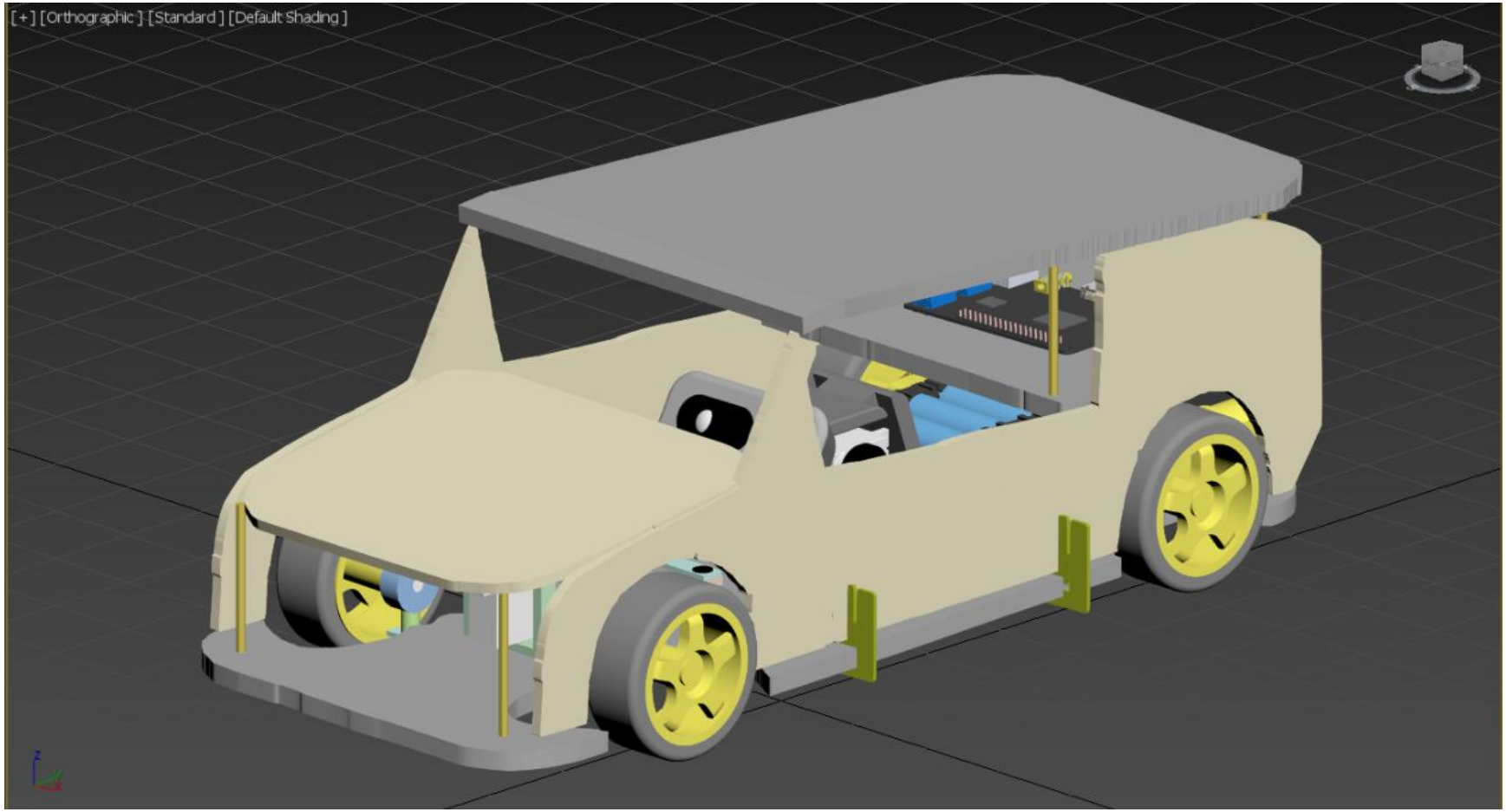


Figure 6-13: Over view

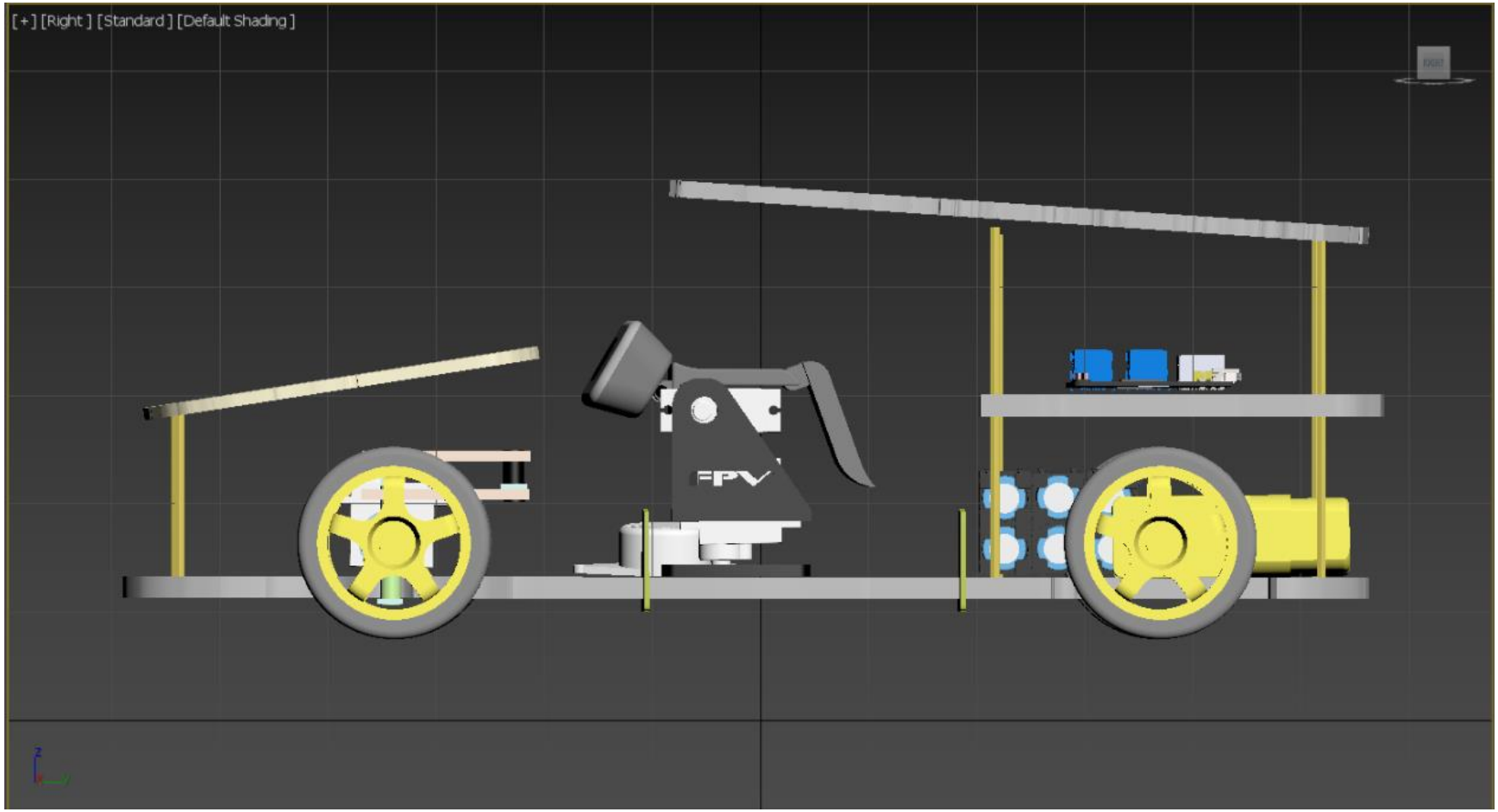


Figure 6-14: Right wire-frame

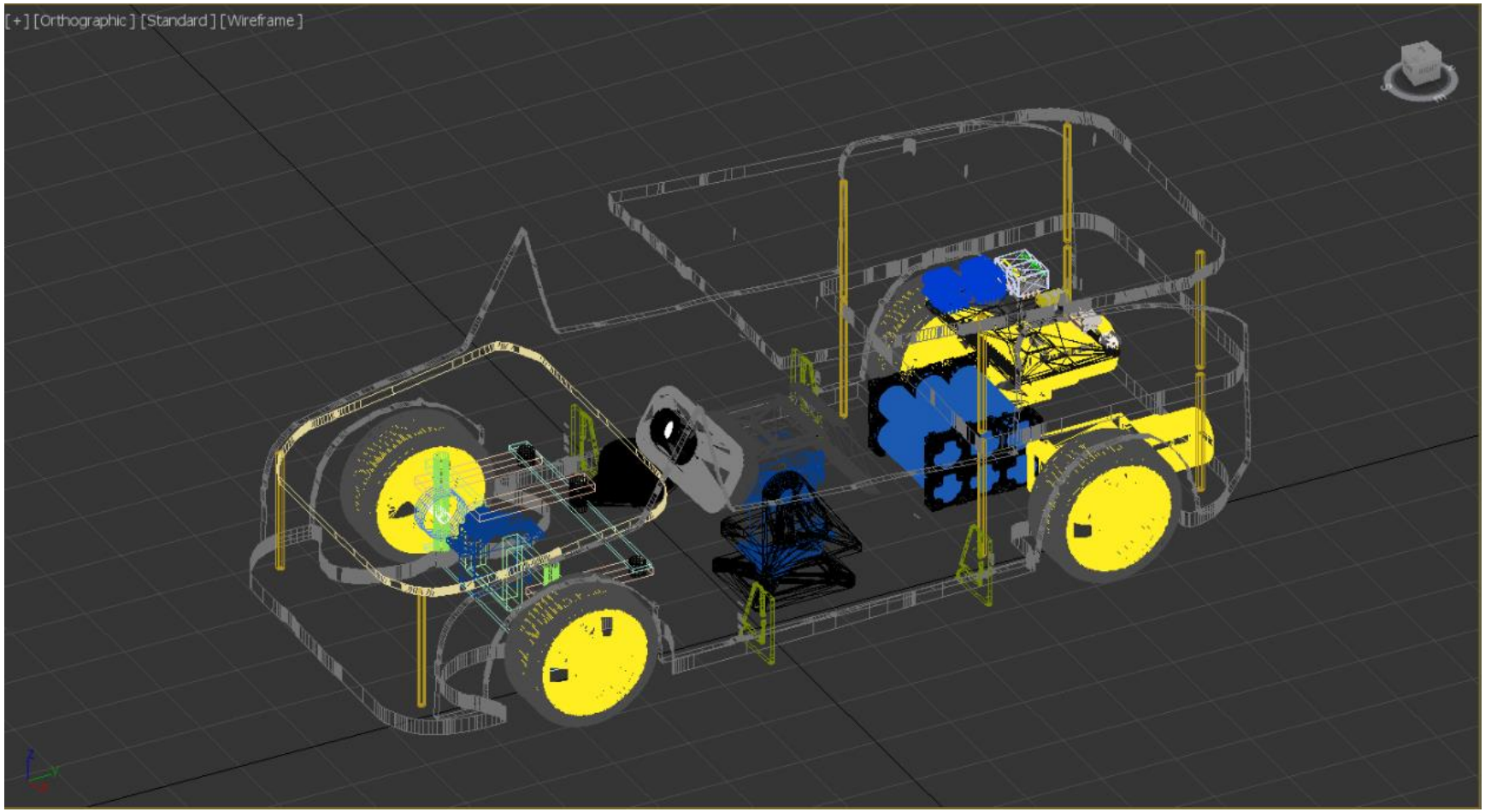


Figure 6-15: Side wire-frame

6.6. Testing

6.6.1. Purpose

The primary purpose of this report is to detect software failures so that detects may be discovered and correct to ensure that our project is thoroughly tested and resulting in a successful implementation project.

6.6.2. Test approach

- Unit testing
- Integration testing
- System testing
- Acceptance testing

6.6.3. Test environment:

Resource	Name and type
Personal Laptop	
Android smartphone	
WIFI	
Window 10	Operating system
Android	Android 4.4, 5.1, 7.0
Android Development tool	Android Studio
Arduino IDE	Arduino 1.8.1

6.6.4. Test Case

6.6.4.1. Software

No.	ID	Test case Description	Test Case Procedure	Expected Output	Inter-Test Case Dependence	Status	Date	Note
1	Android_1	Open [Mini Explorer] main screen	Touch on [Mini Explorer] application icon	Open [Mini Explorer] main screen: screen is displayed with the following information -Text: Project name - 1 connect button		Pass	14/08/2017	

2	Android_2	Showing [WIFI is turned off] dialog after press on [connect] button (If WIFI is turned off)	Touch on [Connect] button	Show alert dialog which contain a text: “WIFI is turned off”	[Mini Explorer] screen is displayed; WIFI of android device is off or not connect with Car WIFI	Pass	14/08/2017	
3	Android_3	Open [Controller] screen	Touch on [Connect] button	Open [Controller] screen: screen is displayed with the following information -1 [Disconnect] button -5 control buttons -2 buttons change speed -1 [VR mode] button -1 web view	[Mini Explorer] screen is displayed; WIFI of android device is on and connected with the Car WIFI	Pass	14/08/2017	
4	Android_4	Send control car data	Touch on [Gear switcher]/[pedal]/[break]/[light signal]/[sound signal]/[steering wheel] button	Send signal to car system and show message on Raspberry PI in order: “Speed”:0, ”angle”:0, ”mode”:0, ”light”:0, ”buzzer”:0	[Controller] screen is displayed; WIFI of android device is on and connected with the Car WIFI	Pass	14/08/2017	
5	Android_5	Open [VR mode] screen	Touch on [VR mode] button	Open [VR mode] screen: screen is displayed with the	[Controller] screen is displayed; WIFI of android device is on and	Pass	14/08/2017	

				following information -1 Web views that display Camera image info	connected with the Car WIFI			
6	Android_6	Send control camera module data	Turn android device to left or right	Send signal to car system and show message on Raspberry PI	[VR mode] screen is displayed; WIFI of android device is on and connected with the Car WIFI	Pass	14/08/2017	

6.6.4.2. Hardware

No.	ID	Test Case Description	Test Case Procedure	Expected Output	Inter-Test Case Dependence	Status	Date	Note
1	Car_1	Change run mode testing	Pull the [Gear switcher]	Raspberry receive correct message and change the car run mode	-[Controller] screen is displayed; WIFI of android device is on and connected with the Car WIFI	Pass	14/08/2017	
2	Car_2	Car turn left testing	Turn the [steering wheel] to left	Raspberry receive correct message and the car turn left	-[Controller] screen is displayed; WIFI of android device is on and connected with the Car WIFI	Pass	14/08/2017	
3	Car_3	Car turn right testing	Turn the [steering wheel] to left	Raspberry receive correct message and the car turn right	-[Controller] screen is displayed; WIFI of android device is on and connected	Pass	14/08/2017	

					with the Car WIFI			
4	Car_4	Car increase speed testing	Push the [Pedal]	Raspberry receive correct message and the car increase speed	-[Controller] screen is displayed; WIFI of android device is on and connected with the Car WIFI	Pass	14/10/2017	
5	Car_5	Car decrease speed testing	Push the [Break]	Raspberry receive correct message and the car decrease speed	-[Controller] screen is displayed; WIFI of android device is on and connected with the Car WIFI	Pass	14/08/2017	
6	Car_6	Sound signal testing	Press on the [Sound signal]	Raspberry receive correct message and turn on the sound signal	-[Controller] screen is displayed; WIFI of android device is on and connected with the Car WIFI	Pass	14/08/2017	
7	Car_7	Turn on light signal testing	Press on the [Light signal left]/ [Light signal right]	Raspberry receive correct message and turn on the light signal left/right	-[Controller] screen is displayed; WIFI of android device is on and connected with the Car WIFI	Pass	14/08/2017	
8	Car_8	Turn off light signal testing	Press on the [Light signal left]/[Light	Raspberry receive correct message	-[Controller] screen is displayed;	Pass	14/08/2017	

			signal right] again	and turn off the light signal left/right	WIFI of android device is on and connected with the Car WIFI - [Light signal left]/ [Light signal right] has turned on			
9	Car_9	Turn the [camera module] to left	Turn android device to left	Raspberry receive correct message and turn the [camera module] to left	-[Controller] screen is displayed; WIFI of android device is on and connected with the Car WIFI - Camera module has turned on	Pass	14/08/2017	
10	Car_10	Turn the [camera module] to right	Turn android device to right	Raspberry receive correct message and turn the [camera module] to right	-[Controller] screen is displayed; WIFI of android device is on and connected with the Car WIFI - Camera module has turned on	Pass	14/08/2017	

VII. RESULTS

7.1. Limitation of system

We find out the system may have these following limitations:

- Power supply for system is unstable
- Video Streaming delay
- Some JSON signal are missing when transfer from WEMOS to Server
- No casing for product

7.2. Solution for reducing limitations

To reduce some risk and limitation, we will implement a number of following activities:

- Use Step-up power module XL6009 to increase the power supply
- Reduce frame per second
- Increase response time when transfer JSON
- Manufacture casing for product

7.3. Difficulties

The difficulties in the implementation process project:

- Team is lack of knowledge about android's rotation vector, steaming video.
- New in Raspbian.
- The team must self-learn about new tool and new programing languages.
- Team member are ill and unavailable at critical time.
- Hardware component are damaged due to electric power.

7.4. Lesson learned

- Learn and use some useful software.
- Know programing Python.
- Planning task, create Schedule.
- Working in group, find problem and resolve problem.

VIII. SYSTEM USER'S MANUAL

8.1 Setting up hardware

- Press the "power" button on car
- Supply power for remote controller.
- Remote Controller will be auto connected with the car system.

8.2 Android Application

The first, download and install the "Mini explorer" app on android device (Recommendation: Android 4.4+).

8.2.1 Connect WIFI

- Open WIFI setting on your smart phone
- Choose "RazzPi" WIFI
- Enter password (The WIFI SSID and password will be provided in user manual. Default: "123@123a")
- Choose "Connect"

8.2.2 Connect Car system

- Open "Mini explorer" app.
- Press "Connect" button. After success, direct move to "Controller" screen.

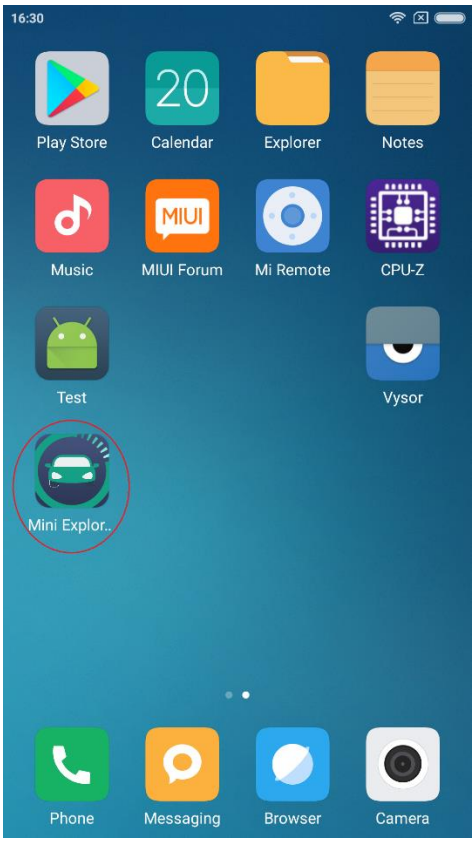


Figure 8-1: Open "Mini explorer" app

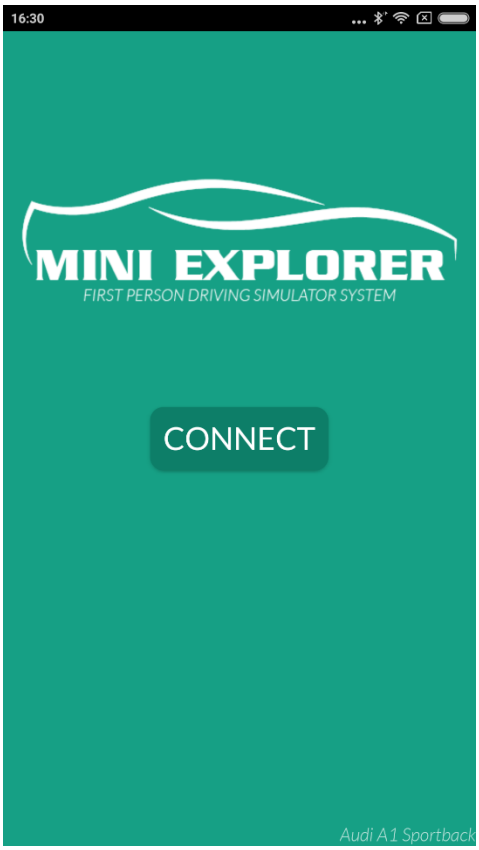


Figure 8-2: Choose "Connect" button on the main screen

Figure 8-3: Successful

8.2.3 Control car movement

- Pull the “gear switcher” down to change the run mode (Default mode: parking)
- Turn the “Steering wheel” to left (or right) that you want the car move in that direction.
- Press the “Pedal” button to increase the car speed
- Press the “Break” button to decrease the car speed
- Press the “light signal” arrow button to turn on the light signal
- Hold the “Buzzer” button to turn on the sound signal
- Press the “Disconnect” button to disconnect from car server

Figure 8-4: Controller Screen

8.2.4 VR mode

Push the “VR” button to enter the VR mode, after that put your android device into the VR (Virtual reality) headsets. Now, your android device will only display the camera view and you can control your car by using the “remote controller”. You can also turn the camera to left (or right) by turn your head to the direction that you want.

Figure 8-5: VR mode

IX. REFERENCES

- http://www.pyimagesearch.com/2016/04/18/install-guide-raspberry-pi-3-raspbian-jessie-opencv-3/ - Install guide: Raspberry PI 3 + Raspbian Jessie + OpenCV3
- https://www.raspberrypi.org/documentation/configuration/wireless/access-point.md - Setting up a Raspberry P as an access point in a standalone network
- https://www.raspberrypi.org/documentation/linux/usage/rc-local.md
- http://www.chioka.in/python-live-video-streaming-example/
- http://opencv.org/
- https://electronics.stackexchange.com/questions/99434/arduino-vs-microprocessor-vs-microcontroller
- https://developer.android.com/guide/topics/sensors/sensors_overview.html
- https://github.com/nisrulz/sensey
- https://github.com/arduino/Arduino/tree/master/hardware/arduino/avr/libraries/SoftwareSerial
- https://www.draw.io/