University of Applied Sciences HSHL - Lippstadt

SS2023 - ELE - Prototyping & Systems Engineering

Apollo

Bhavesh - 2210013

Ammar Imran Khan - 2210022

Kashif Raza - 2210021

Nader Yahia - 2210026

Prof. Dr. Stefan Henkler

Contents:

1. Introduction

This document will outline the required specifications for the implementation of **Apollo**, which is a autonomous car designed to autonomously drive around a track layout.
In section 1 of the document, the requirements will be outlined in detail.

1.1 Purpose

This document is essential to make sure every requirement needed to realise the system is clearly outlined and conveyed to the relevant audience reading this document. Many errors can be rectified through means of proper communication, reducing overall costs by a significant amount.

The intended audiences are:

- Software Developers coding the system
- Engineers working with the hardware
- Project Managers

1.2 Scope

This segment will outline technical requirements expected from the Autonomous car named, ‘Apollo’.

- The car needs to be able to detect black-coloured stripes to be able to effectively map out a desired path towards the destination.

- Avoid any intrusions or obstacles without sustaining any damage or putting itself in harm's way. Red-coloured obstacles are to be removed from the track, whilst Blue-coloured obstacles are to be avoided by navigating around them. This process will be realised through a colour detection system, including an additional fail-safe sensor.

- Be able to control its own speed by constantly reading data from the various sensors, to ensure it is able to cruise at max speed and slow down during cornering.

2) In case data from one sensor is being read outside of expected parameters, the backup sensors activate and an error report is sent to the user about a function failing, it should not lose any capabilities upon malfunction of a sensor.

1.3 Components

This segment will outline the materials, electronic devices and any other necessary component required to realise the prototype model. Any information regarding technical details will also be mentioned, including: Memory, Dimensions and Computing Power Each category will be separated based on where the component generally fits.

Materials:
1) Plywood for Main body of the vehicle.
2) PLA for any additional bodywork.

Electronics:
The following sensors will be used:
1) KY-022 Infra-red Sensor - 2x
2) SG90 Mini Servo - 1x
3) Ultrasonic Distance Sensor, HC-SR04 - 1x
4) Conrad 11.1 V 3200 mAh 20C Eco-Line Lipo battery

Controllers:
1) Arduino Uno
2) L293D motor Driver 2*wheel - 1x

1.4 Definitions, acronyms, and abbreviations.

Any uncommon vocabulary or jargon (determined by us) will be listed here to clarify any confusion the reader of this document might encounter.

- Autonomous = Remote Controlled - This implies that this system can be fully controlled using a radio signal via a ‘remote’, operated by the user of the system.

1.5 References

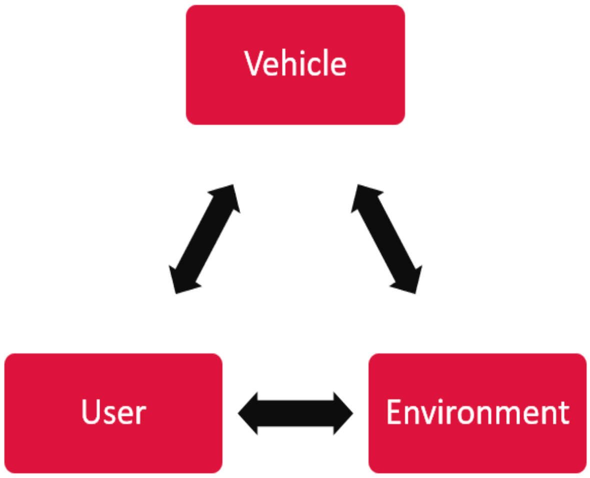
[1] IEEE Software Requirement Specification Std. 830-1998

2. Overall Description

This section will elaborate on how the overall system is supposed to operate, diagrams will be elaborated and provide perspective into the inner workings of the system. Any weakness or constraints within the system are also detailed.

2.1 Product Perspective

A basic outline of the system will be shown in this segment, how it's meant to operate under the most abstract of conditions.



A feedback system created between the 3 main systems that provide input/output. The user will have to occasionally command the autonomous vehicle, whilst it also receives information from the environment.

This diagram abstractly defines the information flow of the system, keeping in mind that more complex analogies will need to be developed to help us realise the system better.

2.2 Product Functions

SysML and UML diagrams will be discussed within this segment.
Any relevant diagram will be referred to as "Fig.X", this is important in order to visualise the system before starting to implement it.

Use Case Diagram:
Refer to Appendix A

The Use Case diagram provides a general outline of how the user can interact with the system.

The user can start the system and initiate special commands like ‘parking’ to engage code specific to that functionality of the vehicular system.

The on-board controller autonomously makes decisions for the vehicle, by continuously exchanging and receiving data from the sensors, it’s able to make decisions on its own that will ensure that the system operates in a predictable manner and causes the most minimal, ideally, no damage.

The environment also provides components that are vital to contributing to the overall sensor data.

Activity Diagram:
Refer to Appendix B

This activity diagram showcases how the system will operate under the most abstract conditions. It is meant to convey a general form of operation, whilst being open to any future updates to the requirements within the system.

We start at the Autonomous car by receiving information from the remote, this allows us to implement any commands necessary for the vehicle to operate, the commands being: Accelerate, brake, park, cruise control.

At the fork node, the responses to different environmental inputs are mapped, decision nodes are implemented to showcase autonomous decision making, completely handled within the controller in the car.

Near the end of operation of the Autonomous system, an error report will be generated in order to outline what improvements would need to be implemented or reworked.

Basing our abstract model upon this diagram helps to clarify what kind of a system needs to be realised under the most basic conditions.

Sequence Diagram:
Refer to Appendix C

The diagram provided depicts the fundamental elements of an autonomous car, including the sensor, controller, and actuator.

The sensor is responsible for detecting the line on the road and sending this information to the controller, which subsequently calculates the steering angle required and sends a command to the actuator.

The actuator then adjusts the steering angle of the car accordingly.

Notably, the sensor can detect turns on the road and control the speed of the car, it can also detect obstacles on the road, and the controller can adjust brake commands accordingly, similarly the colour sensor will adjust the brake commands according to the detected colour. This diagram represents a simplified version of the interactions between these objects and systems in an autonomous car.

Block Diagram:
Refer to Appendix D

This diagram outlines the Autonomous car project by detailing a system designed to control a small car using a microcontroller and various sensors.

The system is capable of following a line, avoiding obstacles, controlling speed, and mapping out a drive path. The input block receives data from the user, line sensors, and ultrasonic sensors.

The controller block processes the input data and sends signals to the motor driver block, which controls the speed and direction of the motor. A battery and power supply provide the necessary energy to run the system. Safety features, such as an emergency stop button and idiot-proof regulations, are also included.

Additionally, a mapping and visualisation block is used to create a map of the car's drive path, allowing for future analysis and improvement of the system's performance.

3D Part Diagrams:

The main body of the vehicle will be presented within this segment, however the rest of the 3D parts and any other relevant design will be presented within the appendix at the end as pictures.
Refer to Appendix E - J

2.3 User Characteristics

What kind of users do we expect for our system? The following users have been identified:

User Group 1:

<u>Hospitals</u>
Apollo is designed for the transportation of equipment in the medical context, the user is meant to give it simple commands and the vehicle manages the more tedious tasks like accelerating, obstacle avoidance by itself, hospitals will be able to integrate an electronic system like this one into their existing communication networks.

User Group 2:

<u>Small-Scale Businesses</u>
With the possibility to transport medical equipment, doesn't restrict its core functionality to that function only, with a few small modular modifications to the compartment, it can be specialised for tasks within a small business chain wanting to transport equipment to their customers or other locations.

2.4 Constraints

I. Must use the Arduino UNO development board.
II. Material consumption of 3d printing materials should be not more than 1kg.
III. Max dimensions should be as stated in the coordinate axes:
   x < 30cm
   y < 20cm
   z < 20cm

IV. Maximum wall thickness should be less than 2mm.

2.5 Assumptions & Dependencies

I. The implemented functions should be operational according to the specification of Arduino UNO, since the code will be dependent on the available memory.
II. Assumed that the prototype system is being used in conjunction with black lanes.
III. The weight of the luggage used by the system should not exceed the supported weight of the wheels.

3. Overall Description

This section lists the requirements for the realisation of the entire system, including the functionalities that will be required.

3.1 Interface Requirements

<u>IR1: Accessibility</u>
Description: The data from the operation of the system should be conveyed to the user in a meaningful way, they are able to easily tell what functions are able to do what. If desired, live updates of data should be able to be accessed by the user.

Rationale: Data is vital to correcting any errors or malfunctions within a system, being able to interpret and present that data in a user-friendly manner is a big part of if a system is easy to maintain.

3.2 Functional Requirements

<u>FR1:</u>
Description: The vehicle should autonomously detect obstacles and have protocols in place to appropriately deal with them by either requesting removal of the obstacle or driving around them in a safe manner.

Rationale: Human error has alot of potential to occur within hard time restrictions, especially in the case where a vehicle might collide with an obstacle at high speed, an appropriate response should be designed for this problem.

FR2:
Description: The vehicle can adjust its own speed according to input from the environment, such as any obstacle detection.

Rationale: In order to fully realise the autonomous part of the vehicle, it's important that it is able to control the power it outputs to the wheels.

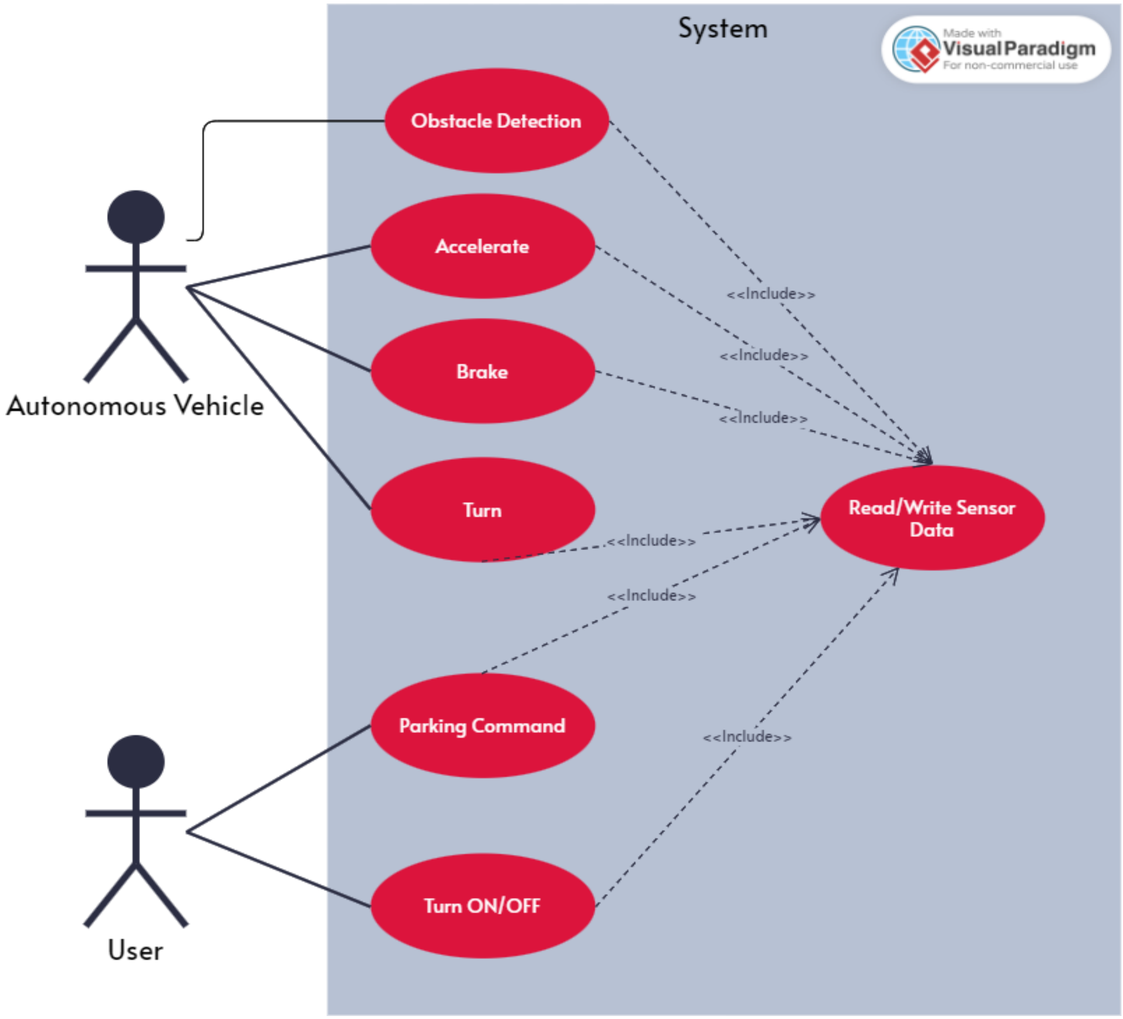
FR3:
Description: The vehicle can map out a lane to drive upon, it is able to read information from the black lanes and follow the line without diverting from the path.

Rationale: The ability to consistently follow the lane is an important aspect of the autonomous delivery system, to ensure any uncertainties can be traced along a fixed path.

3.3 Non-Functional Requirements

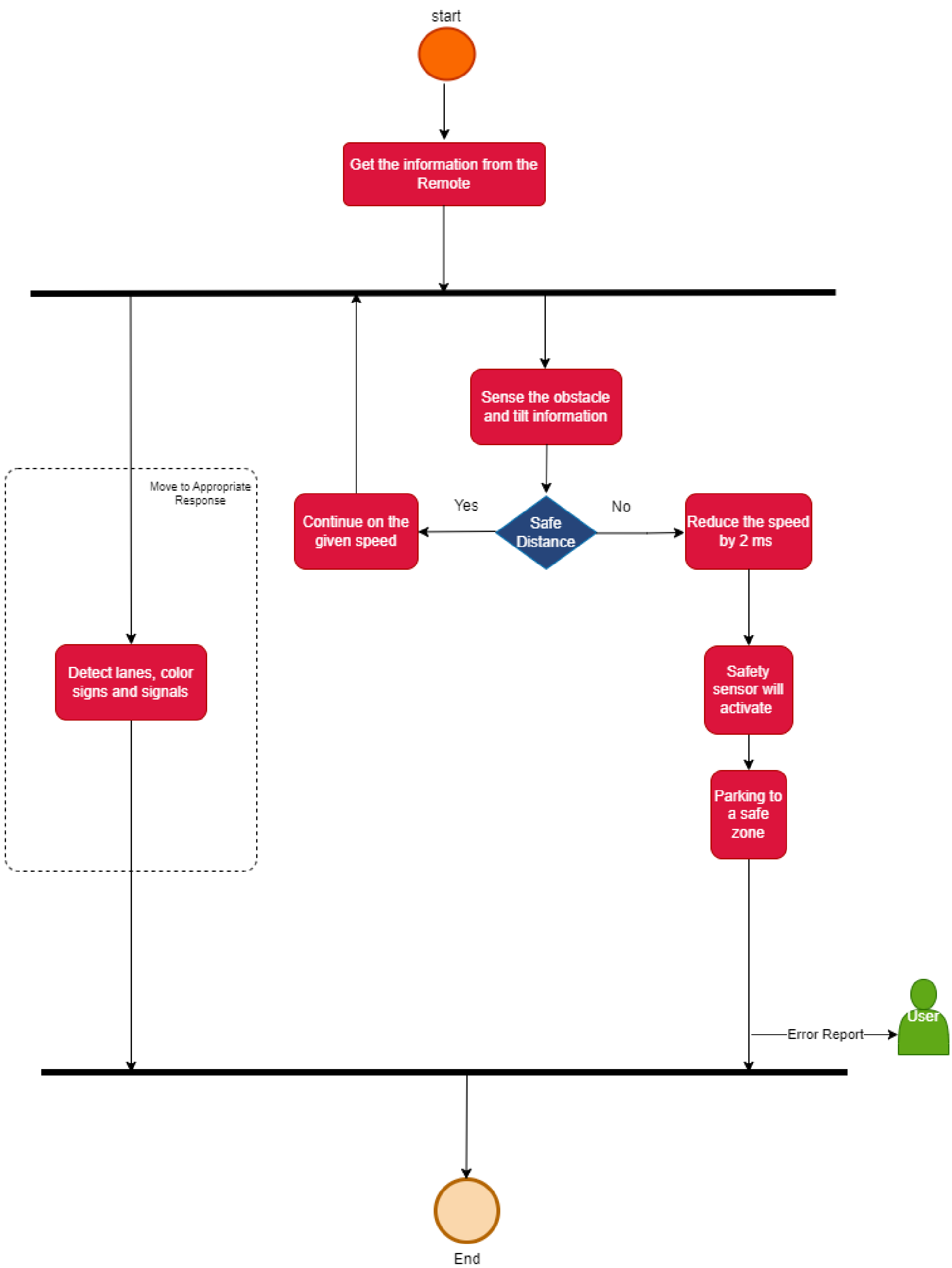
NFR1:
Description: The vehicle design should have a vehicle compartment that is accordingly to the coordinate axes dimensions:
x = 15cm
y = 10cm
z = 10cm
Rationale: A sufficient compartment to support the transport of medical equipment, primary function of Apollo.

4. Appendix

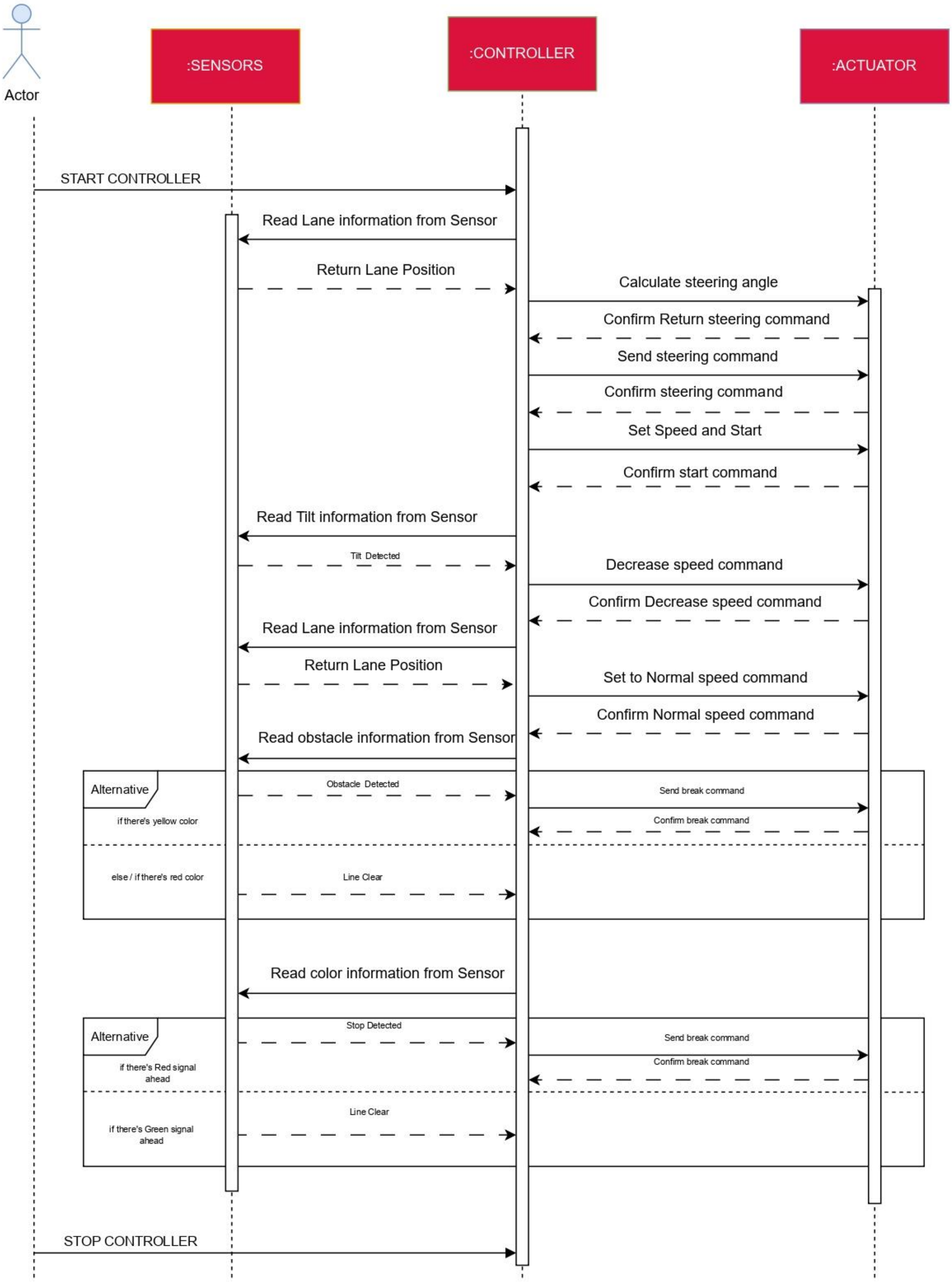


System
Made with
Visual Paradigm
For non-commercial use
Obstacle Detection
Accelerate
Brake
Turn
Parking Command
Turn ON/OFF
Read/Write Sensor Data
<<Include>>
<<Include>>
<<Include>>
<<Include>>
<<Include>>
<<Include>>
Autonomous Vehicle
User

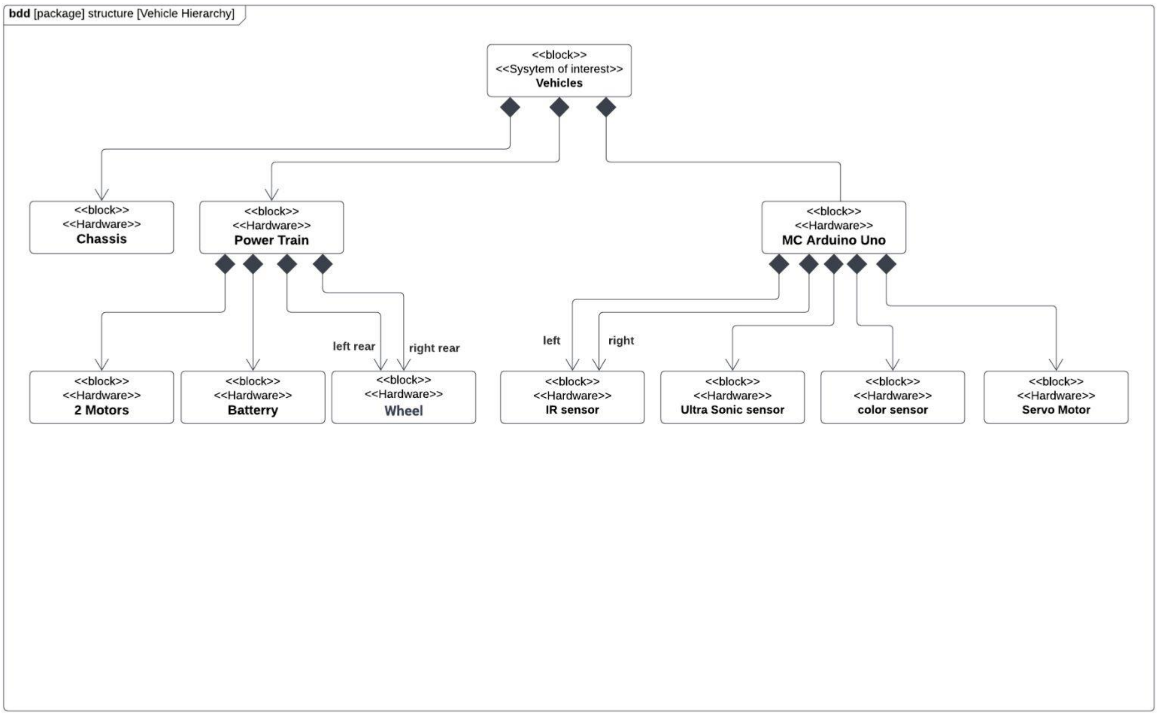
A.


start
Get the information from the Remote
Sense the obstacle and tilt information
Move to Appropriate Response
Continue on the given speed
Yes
Safe Distance
No
Reduce the speed by 2 ms
Detect lanes, color signs and signals
Safety sensor will activate
Parking to a safe zone
Error Report
User
End

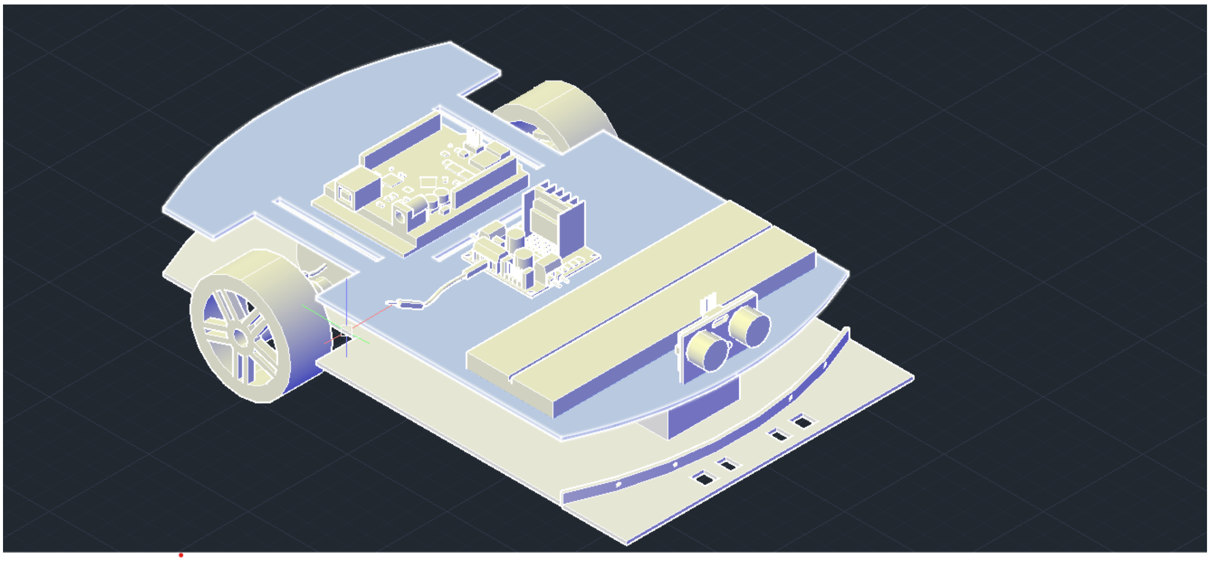
B.


Actor
:SENSORS
:CONTROLLER
:ACTUATOR
START CONTROLLER
Read Lane information from Sensor
Return Lane Position
Calculate steering angle
Confirm Return steering command
Send steering command
Confirm steering command
Set Speed and Start
Confirm start command
Read Tilt information from Sensor
Tilt Detected
Decrease speed command
Confirm Decrease speed command
Read Lane information from Sensor
Return Lane Position
Set to Normal speed command
Confirm Normal speed command
Read obstacle information from Sensor
Alternative
Obstacle Detected
Send break command
if there's yellow color
Confirm break command
else / if there's red color
Line Clear
Read color information from Sensor
Alternative
Stop Detected
Send break command
if there's Red signal ahead
Confirm break command
Line Clear
if there's Green signal ahead
STOP CONTROLLER

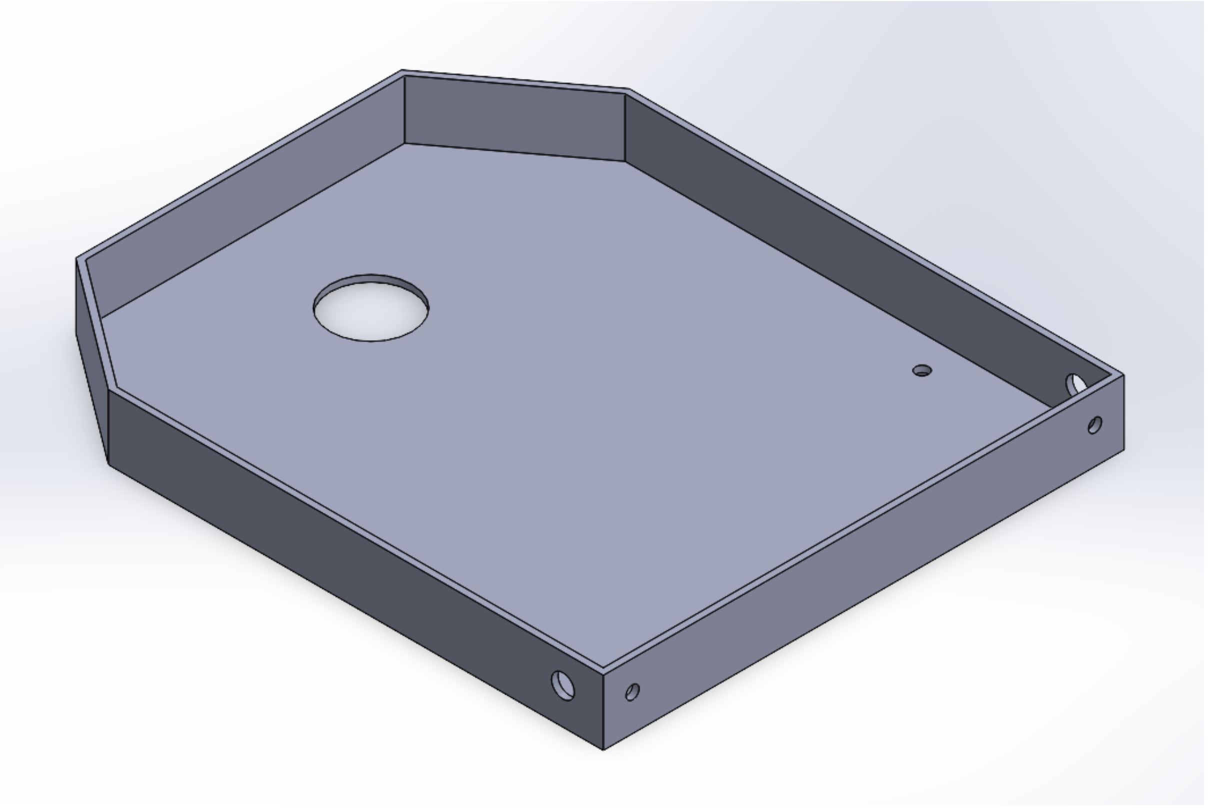
C.


bdd [package] structure [Vehicle Hierarchy]
<<block>>
<<Sysytem of interest>>
Vehicles
<<block>>
<<Hardware>>
Chassis
<<block>>
<<Hardware>>
Power Train
<<block>>
<<Hardware>>
MC Arduino Uno
left rear
right rear
left
right
<<block>>
<<Hardware>>
2 Motors
<<block>>
<<Hardware>>
Batterry
<<block>>
<<Hardware>>
Wheel
<<block>>
<<Hardware>>
IR sensor
<<block>>
<<Hardware>>
Ultra Sonic sensor
<<block>>
<<Hardware>>
color sensor
<<block>>
<<Hardware>>
Servo Motor

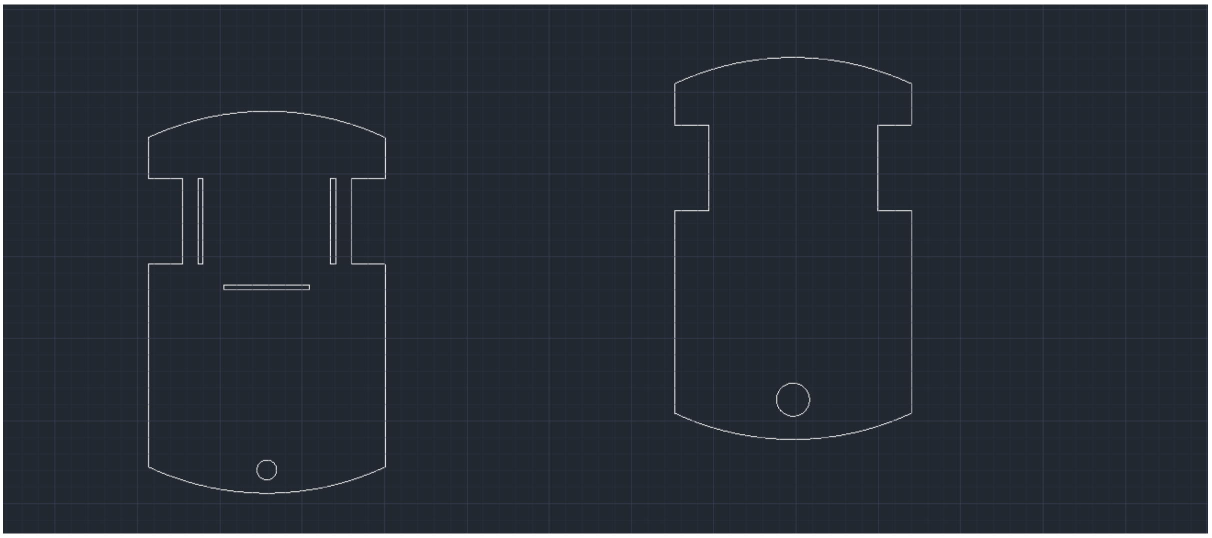
D.



E.

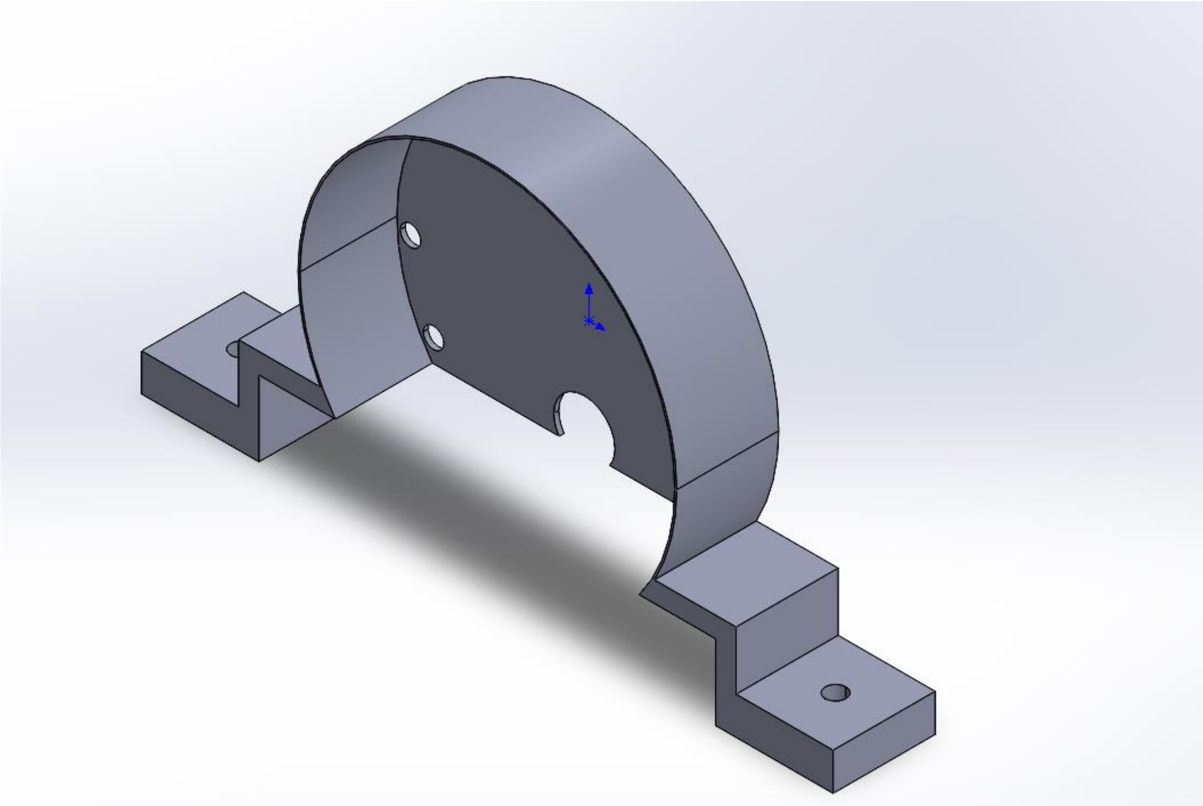


F.

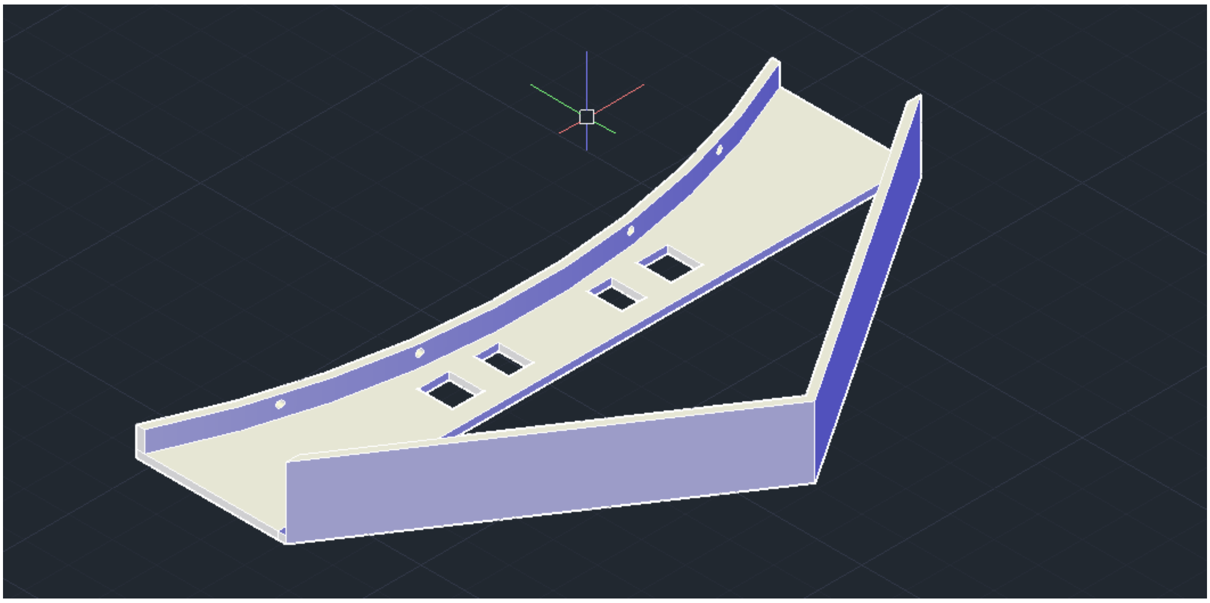


G.

H.



I.



J.