Self-Driving Car Using Stereo-Vision And Anomaly Detection

Mahmoud Ahmed , Nada Ashraf , Omar Ismail , Sarah Fouad
Lobna Shaheen , Alaa Hamdy

November 2019

1 Introduction

1.1 Scope of this document

A Self-Driving Car is aimed to improve and maintain safety conditions on roads while driving and decrease the rate of road accidents. Most Accidents are caused by a human error. A Self-Driving Car is purely analytical that it acts like a smart computer, there won't be neither emotions nor distraction involved while driving because computers are faster and smarter to decide the right action than our mind. A Self-Driving Car is not only about maintaining safety for humans, but it is also beneficial in maintaining the car safety as it alerts you before hitting any road anomalies such as speed bumps and potholes.

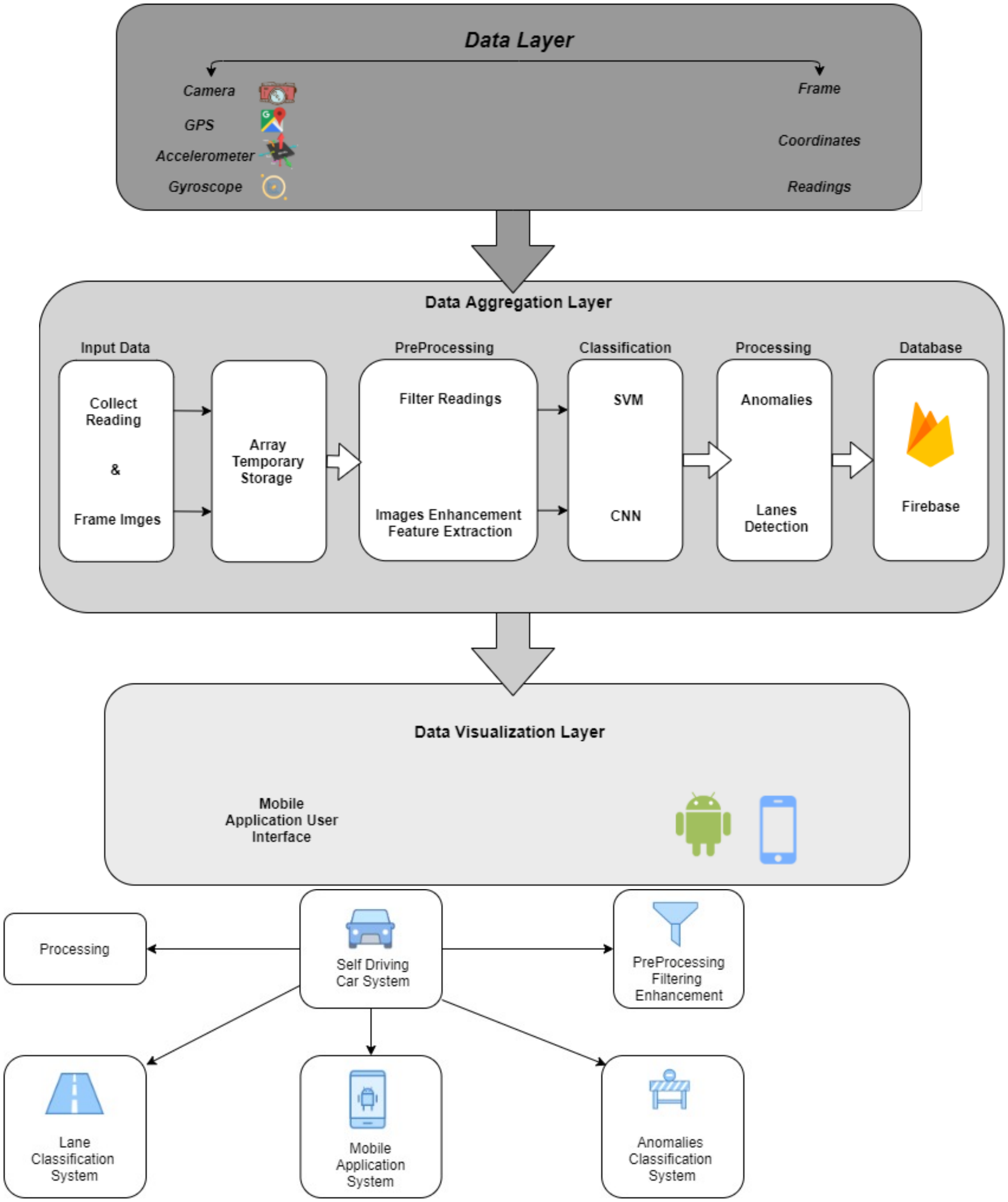
1.2 Purpose of this document

The main purpose of this software requirement specification document is to clarify and illustrate our project (Self-Driving Car) system requirements which are mainly about implementing a sensor based self-driving car that is able to make its own decisions while driving and detecting road anomalies. Our project is considered to be a device (car) that contains a variety of sensors such as Gyroscope, Accelerometer, GPS sensor and a dual camera to perceive the surroundings areas, lanes and objects. These sensors also identify appropriate navigation paths and take the lead in driving during the whole ride. We also provide a fulfilled description about each stage and explain in details the algorithm and classifier used during the development process.

1.3 Overview

Our System consists of three main stages Processing, Pre-processing and Output. The first stage consists of Stereo-Vision Cameras, Ultra-Sonic Sensor and the Cloud. In this stage the Stereo-Vision and Ultra-Sonic are responsible for capturing, detecting and collecting the data set (speed pumps, potholes, etc..)

and save its location then upload it on the Cloud. The Stereo-Vision Cameras (dual cameras) also used to help the car to know the depth map between the vehicles in front of it. The second stage consists of Accelerometer, GPS and Gyroscope. The Accelerometer is responsible for measuring the acceleration of the car while driving which will help us when alerting the driver to slow down his speed. The Gyroscope is responsible for measuring or maintaining orientation and angular velocity (x, y, z) which will help us when collecting data about the road anomalies. The GPS is used to highlight the road anomaly detected and save its location to alert the vehicles that will pass on this location later. Finally, the last stage is the output which contains the motor and a buzzer. Our whole system is controlled by A Raspberry-Pi 3 which runs by Support Vector Machine algorithm that classify the readings of the road conditions from the sensors and a CNN algorithm that is also used to classify the images of the road lanes. The Block diagram is shown in figure (1). also the last has our mobile application that represents the reports of the self-driving car and the live state of its action to our user.


Data Layer
Camera
GPS
Accelerometer
Gyroscope
Frame
Coordinates
Readings
Data Aggregation Layer
Input Data
Collect Reading
&
Frame Imges
Array Temporary Storage
PreProcessing
Filter Readings
Images Enhancement Feature Extraction
Classification
SVM
CNN
Processing
Anomalies
Lanes Detection
Database
Firebase
Data Visualization Layer
Mobile Application User Interface
Processing
Self Driving Car System
PreProcessing Filtering Enhancement
Lane Classification System
Mobile Application System
Anomalies Classification System

1.4 Business context

Our proposed automated system is implemented to reduce the rate of accidents and keep you safe while driving because a future full self-driving car will be a safer one. In addition, our system is also designed to maintain the safety of the car from bad roads and unseen road anomalies. Our project target is to be implemented with the least possible cost that it can be put in each single car and save millions of lives. By that we maintain the safety conditions on roads and reduce the accidents rate as possible.

2 General Description

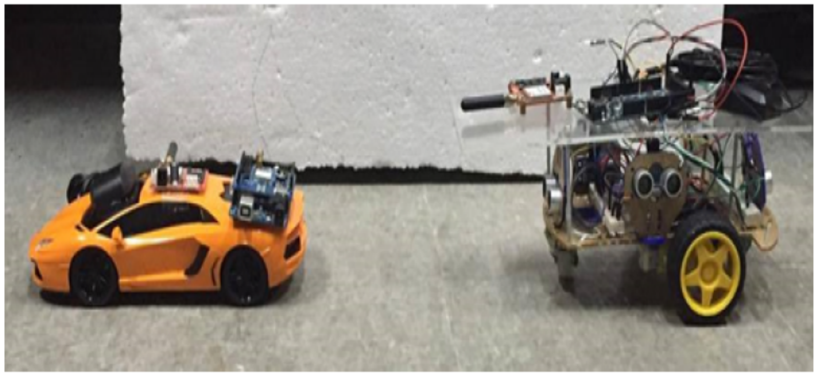
2.1 Product functions

- The Car will detect the distance between any obstacles in front of it.
- The Car will detect speed bumps.
- The Car will detect potholes.
- The Car will save road anomalies coordinates
- Alert to the driver when the car passes on previously detected anomaly.
- The Car slows down or changes lane before any detected anomaly.

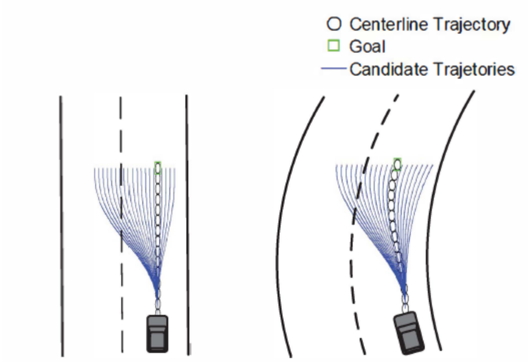
2.2 Similar system information

Towards self-driving car using convolutional neural network and road lane detector[1] system discusses the evolution in computer vision and how it has grown that now it is easy to develop a self-driving car using deep learning. YOLO (You Only Look Once) which is a real-time CNN method that was used to implement this project. This method is implemented to detect other objects from images. They also used a road lane detector which detect road track from the video frame. After collecting this information, they are passed to the controller which integrate both of the data (objects and lanes) to help the self-driving car to make it 's owns right decision-making process. ***Self-driving***

and driver relaxing vehicle[2] There are two Aims of this embedded car, the first one is automated driving during traffic jam, the second one is (to make dynamic destination) following another car that is familiar with its destination by continuously receiving its direction/location. So in this paper vehicles are smart enough to take intelligent decisions in as little time as possible and vehicles can determine the distance from another vehicle/obstacles. Also in this project there are many sensors used for example: ultrasonic sensors are used to avoid obstacles, GPRS module to gets the route and move in this path and most importantly mobile robot is used in this project to be as a small demo for vehicle and to put all the hardware components on it as in figure (1). All these things are made only to make the driver more relaxed while driving his own vehicle.



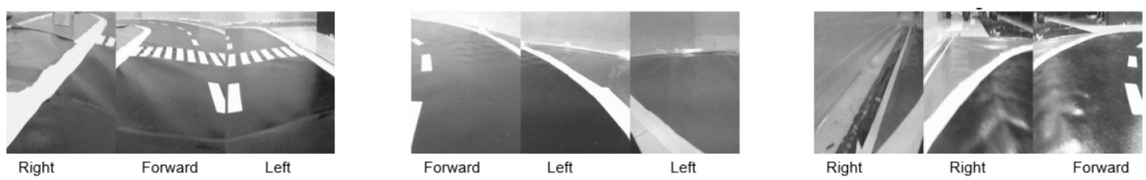
A model based path planning algorithm for self-driving cars in dynamic environment [3] proposed a new algorithm called path planning. This algorithm used to determine the shortest trajectory in dynamic environment while there are complex multi obstacles in this path. Also this algorithm is generating online trajectory. In case of nearby collision, the self-driving car take the right decision to avoid this collision by taking any other suitable lane.



(a) Straight road scenario (b) Curve scenario

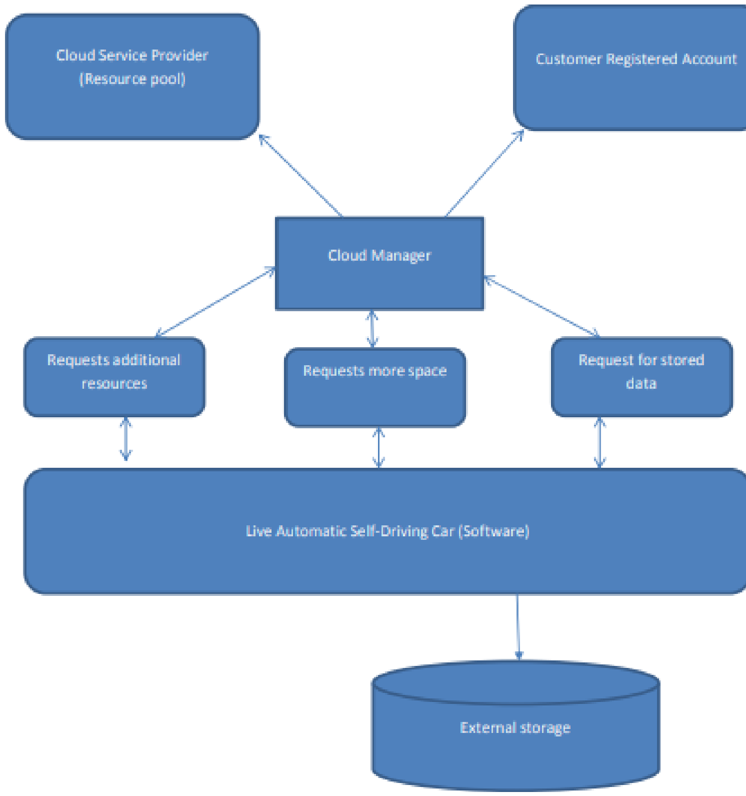
Real-Time Self-Driving Car Navigation Using Deep Neural Network system [4] analysis the greatest technology and artificial intelligence in the future and how humans will benefit from it. They implemented a self-driving car prototype using deep neural network/CNN on a raspberry Pi 3 model B. they had the ability to make their own data set and collect these data and information. They made an oval artificial path outdoor with an 8 shaped traffic signs and using the camera they took images as data. The steps were to train these data and then move on to the test experiment. According to the output or final result the self-driving car prototype makes it 's own decision and take the right action to move forward/backward/stop etc.

Self-Driving Cars Using CNN and Q-Learning[5] Paper is mainly focusing on the learning process of the self-driving car. They prepared a specific road environment for the self-driving car and will extract road features and other real world obstacles to train the car using deep learning algorithms and let the car car learn everything that is related to this whole state by itself from zero to pro. Also the model will extract features by itself from the state and learn on the important useful features only. The autonomous car is based of a small raspberry pi and a software programs using python based library to control the car over a network or locally. They have used two methodologies to practice the self-driving car. The first methodology is “Supervised learning” where the car was driven on a road/environment and collected more than three thousands data-points. Then a CNN model was trained based on this and achieved 73 percent test and 89 percent train accuracy. The second methodology is “Reinforcement learning” where the car was trained for three different road signs which are Stop sign, No Left sign, and Traffic lights using Deep Q-Network (DQN) with an existing CNN model. Also these mention road signs were detected in the environment using Open CV cascade classifiers.



(a) Forward (b) Left (c) Right

Cloud Based Self Driving Cars [6] Cloud Based Self Driving Cars their aim is to reduce the data storage which the autonomous car needs to as the self-driving car generate 1 GB per second their idea is to reduce the data storage which the autonomous car needs to by not downloading any useless data on the cloud they filter the needed data and upload it to the cloud. The problem is that when any data uploaded to the cloud they will need time, effort and more money to buy extra storage, hence this system upload the code to the cloud according to the user that will use the system. The cloud contains traffic rules, maps and signal information, there are to constrains in this system the first thing that must be found is good network connection, the second thing is available storage in the cloud server to upload and retrieve data from it. This car consists of 4 radars, 1 camera, wheel encoder and GPS. Also there is external storage which is very important when travelling from any location to another given that the road may be free from the Internet so this external storage is the only savior in this



situation.

2.3 User characteristics

DRIVER:

- The autonomous car is an embedded system and the driver won't have a huge interaction with our system but at least he/she will start and stop the self driving mode.
- Must have basic knowledge of driving and using our vehicle.

2.4 User problem statement

Implementing a self-driving car which detects road anomalies and save its co-ordinates to alert the driver when he/she passed there at any other time also it measures distance between vehicles and obstacles using stereo vision to take intelligent action with them.

2.5 User objectives

By using our autonomous vehicle,

- The driver will have accurate detection of any road anomalies and the vehicle will take intelligent reaction with any of these anomalies
- The driver will be relaxed and safe in this vehicle as it will avoid any clash with any other car or obstacle
- Decrease the rate of accidents by improving its alarm system
- The driver can intervene at any time to take the full control of this vehicle

2.6 General constraints

- MPU, Accelerometer and gyroscope must be will fixed to avoid false readings.
- Stereopi (2 cameras) must be putted to have clear view so that it can take clear images
- Good internet connection to save the data continuously in real time

3 Functional Requirements

3.1 Road Lane Classification Function

Functional Requirements	DetectRoadLane
Input	EnhancedImage
Description	A function to classify and detect the road lanes in the roads
Priority	High
Risks	Cameras interruption
Constrains	Bad quality and deleted road lanes
Dependency	Depends on StartSelfDriving , EnhanceRoadLane
Output	Detects the roadlane that the car follows

3.2 Anomalies Classification Function

Functional Requirements	DetectAnomalies
Input	FilteredReadings
Description	A function to classify and detect road anomalies
Priority	High
Risks	Sensors Movement can cause issues
Constrains	N/A
Dependency	Depends on StartSelfDriving , FilterSensorsData
Output	Detects roadanomalies that the car will behave based on them

3.3 Start Self Driving Function

Functional Requirements	StartSelfDriving
Input	N/A
Description	A function to boot up the raspberry pie with all equipped cameras and sensors to perform their functionality
Priority	High
Risks	Car Motors Fails / Battery Gets empty
Constrains	N/A
Dependency	Depends on DetectRoadLane function
Output	Car moves forward to start the self-driving mode

3.4 Change Car Lane To Left Function

Functional Requirements	MoveToLeftLane
Input	N/A
Description	A function to change the car lane to the left lane
Priority	High
Risks	Incoming Fast car from behind / Car moves next to our car
Constrains	N/A
Dependency	Depends on StartSelfDriving , DistanceMeasure , AnomaliesDetection
Output	Changes car to the left lane

3.5 Change Car Lane To Right Function

Functional Requirements	MoveToRightLane
Input	N/A
Description	A function to change the car lane to the right lane
Priority	High
Risks	Incoming Fast car from behind / Car moves next to our car
Constrains	N/A
Dependency	Depends on StartSelfDriving , DistanceMeasure , AnomaliesDetection
Output	Changes car to the right lane

3.6 Measure DepthMap Function

Functional Requirements	MeasureDepthMap
Input	EnhancedImages
Description	A function that measures the distance between our vehicle and the vehicles/objects in front using stereovision
Priority	High
Risks	Cameras Interruption
Constrains	Dust or dirt on the camera
Dependency	Depends on StartSelfDriving
Output	Grayscale video frames that defines the distances between our vehicle and objects/vehicles infront

3.7 Slow Car Down Function

Functional Requirements	SlowCarDown
Input	DeaccRatio , MinSpeed
Description	A function that slows down the car based on certain conditions
Priority	Mid
Risks	N/A
Constrains	N/A
Dependency	Depends on StartSelfDriving , MeasureDepthMap, AnomaliesDetection
Output	Slows down car speed to the Minimum Speed defined with the deceleration ratio defined.

3.8 Stop car Function

Functional Requirements	StopCar
Input	N/A
Description	A function that stops the car after slowing it down based on certain conditions
Priority	Mid
Risks	N/A
Constrains	N/A
Dependency	Depends on StartSelfDriving , MeasureDepthMap, SlowCarDown
Output	Stops the car movement

3.9 Road Lane Image Enhancement Function

Functional Requirements	EnhanceRoadLane
Input	RoadLaneImage
Description	A function that enhances and prepares the road lane image to be classifies
Priority	High
Risks	N/A
Constrains	Deleted or not clear road lanes
Dependency	Depends on StartSelfDriving
Output	Enhanced road lane image that has only the road lane

3.10 Filter Sensors Reading Function

Functional Requirements	FilterSensorData
Input	SensorsData
Description	A function that filters the sensors reading to avoid noises
Priority	Low
Risks	N/A
Constrains	N/A
Dependency	Depends on StartSelfDriving
Output	Filtered sensors data to be used for classification

3.11 Upload Current car Location Coordinates Function

Functional Requirements	UploadCurrentLocation
Input	N/A
Description	A function that uploads the coordinates of the current car to the cloud server.
Priority	High
Risks	Network connection failure
Constrains	Network connection failure will cause problems uploading the coordinates
Dependency	Depends on StartSelfDriving function
Output	Uploads the location of the car to the cloud server

3.12 Move Car Forward Function

Functional Requirements	MoveCarForward
Input	AccRatio , MaxSpeed
Description	A function that moves the car based on the given Acceleration ratio and maximum speed.
Priority	High
Risks	N/A
Constrains	N/A
Dependency	Depends on StartSelfDriving , MeasureDepthMap functions
Output	Moves the car forward to the maximum speed defined with the acceleration ratio defined

3.13 Android Application Retrieve anomalies Function

Functional Requirements	RetrieveAnomalies
Input	N/A
Description	function to retrieve data of anomalies from server
Priority	Mid
Risks	N/A
Constrains	Bad Network connection
Dependency	N/A
Output	Data Retrieved to the mobile application

3.14 Android Application Retrieve distance Function

Functional Requirements	RetrieveDistance
Input	N/A
Description	function to retrieve distance of front cars from server
Priority	Mid
Risks	N/A
Constrains	Bad Network connection
Dependency	N/A
Output	Retrieved Distance from front cars to the mobile application

3.15 Android Application View anomalies Function

Functional Requirements	View Anomalies On Screen
Input	N/A
Description	function to view data of road anomalies from server
Priority	Mid
Risks	N/A
Constrains	N/A
Dependency	Depends on RetrieveAnomalies Function
Output	View Anomalies data on mobile application screen

3.16 Android Application View Distance from Front cars Function

Functional Requirements	ViewDistance
Input	N/A
Description	Function to view data of distance from cars from server
Priority	Mid
Risks	N/A
Constrains	N/A
Dependency	N/A
Output	View Anomalies data on mobile application screen.

3.17 Get Video Frame Function

Functional Requirements	getFrame
Input	N/A
Description	function to retrieve each video frame
Priority	High
Risks	Camera disconnect may cause problems
Constrains	N/A
Dependency	Depends on StartSelfDriving
Output	An image from the video frames

3.18 Get Readings From Sensors

Functional Requirements	getReadings
Input	N/A
Description	function to read data from Accelormeter and Gyro-Scope sensors
Priority	Mid
Risks	N/A
Constrains	N/A
Dependency	Depends on StartSelfDriving
Output	Accelormeter and Gyro-Scope data

3.19 Get Car Current Location

Functional Requirements	getCurrentLocation
Input	N/A
Description	function to retrieve current location of car
Priority	Mid
Risks	N/A
Constrains	Car must be facing the sky directly
Dependency	Depends on StartSelfDriving
Output	Longitude and Latitude for the car current location.

3.20 Set Destination Function

Functional Requirements	SetDestination
Input	Lng,Lat
Description	function to set the location where the car will head to.
Priority	Low
Risks	N/A
Constrains	N/A
Dependency	N/A
Output	N/A

3.21 AlertDriver Function

Functional Requirements	AlertDriver
Input	N/A
Description	function to alert the driver when car is about to make sudden decision.
Priority	Mid
Risks	No Network Connection
Constrains	Bad network connection might cause failure of sending the alert.
Dependency	MeasureDepthMap , MoveToRightLane , MoveToLeftLane
Output	N/A

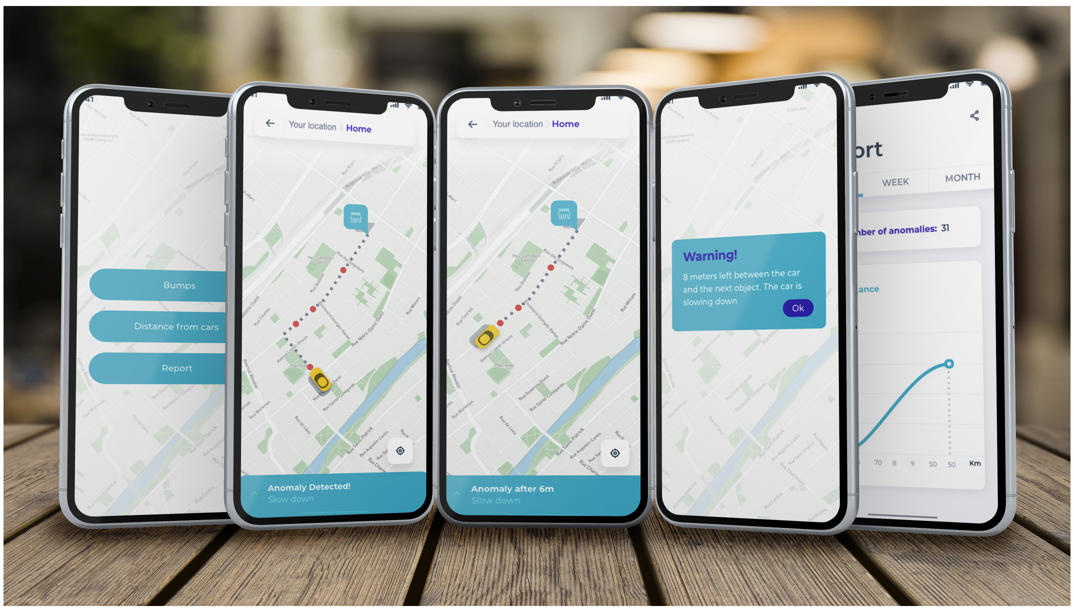
3.22 CD Anomalies To Upload Query Function

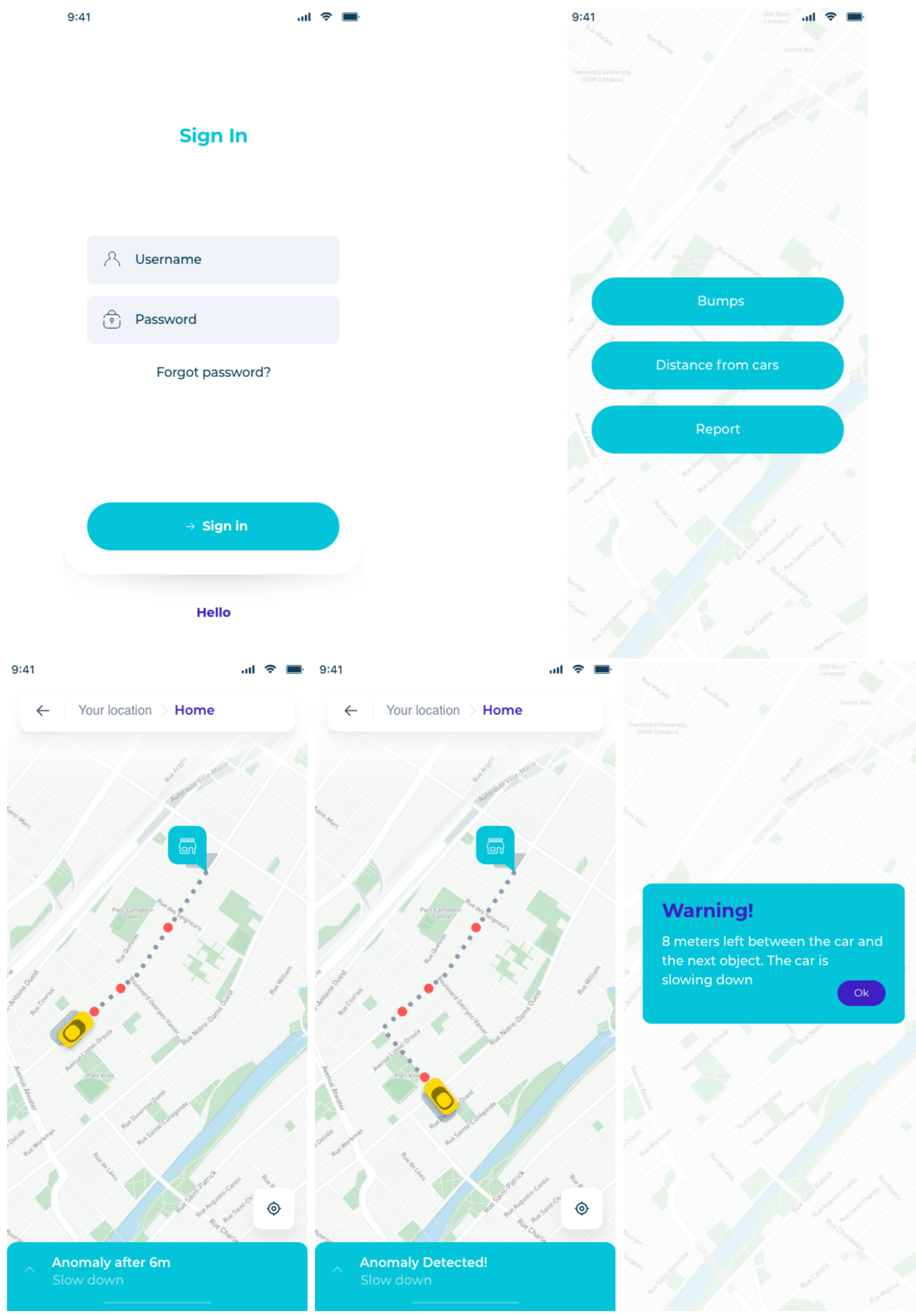
Functional Requirements	AddAnomalyToUploadQuery, RemoveAnomalyFromUploadQuery
Input	N/A
Description	two functions to add and remove the anomalies readings to/from server upload query.
Priority	High
Risks	No Network Connection
Constrains	Bad network connection might cause failure of uploading/removing the data.
Dependency	Depends on StartSelfDriving , DetectAnomaly function
Output	Upload and remove the anomalies data to/from server

4 Interface requirements

4.1 User Interface

Our system user interface is considered to be an android application which is very usable and clear. The system alerts the user when detecting any road anomaly, the system alerts the user before hitting the road anomaly and finally notify the user before slowing down.




9:41
Sign In
Username
Password
Forgot password?
Sign in
Hello
9:41
Bumps
Distance from cars
Report
9:41
Your location
Home
Anomaly after 6m
Slow down
9:41
Your location
Home
Anomaly Detected!
Slow down
Warning!
8 meters left between the car and the next object. The car is slowing down
Ok

4.1.1 CLI

N/A

4.1.2 API

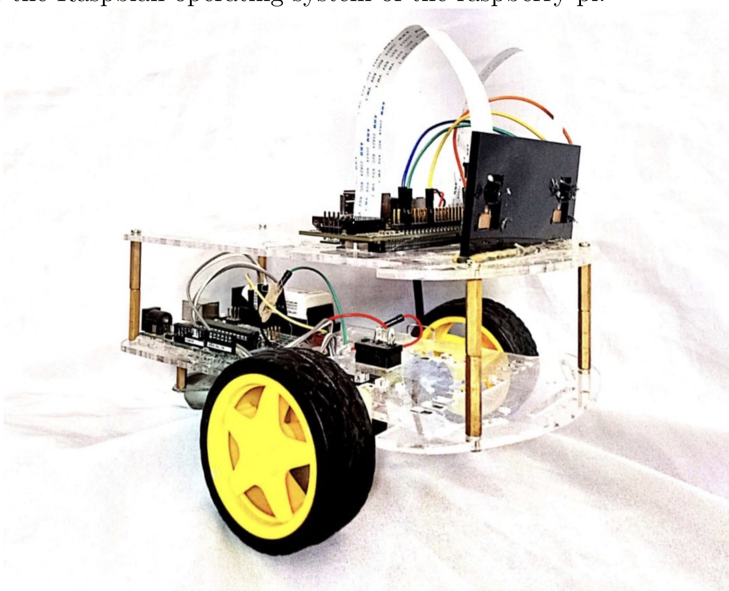
Our system uses google maps API to detect the current location of the detected anomaly and store it in the fire base server.

4.1.3 Diagnostics or ROM

N/A.

4.2 Hardware interfaces

The car consists of two motors, L298N H-Bridge which is a motor driver that allows a full control of the two car motors at the same time, and Arduino UNO that simulate the car movement. The main component of the system is the raspberry pi that is connected to the MPU unit, Wi-Fi dongle, and stereo vision cameras. By all of these hardware components connected together, the system is able to accurately recognize its whole surrounding environment. As shown in figure **??**, The two motors are being controlled by L298N motor driver based on the received signals from Arduino that is powered by an external nine-volt battery, and Arduino sends these signals based on the serials received from the raspberry pi. These serials are being transmitted on a serial path that is connected between Arduino and raspberry pi. Moreover, the algorithms take their input either as video frames or data readings from the cameras and sensors connected with the raspberry pi. The SD card is also a part of stereo pi as it has the Raspbian operating system of the raspberry pi.



4.3 Communications interfaces

N/A

4.4 Software interfaces

N/A

5 Performance requirements

This system calculates distance using stereo vision so it require a dual camera that captures videos then calculate the distance in each frame although we detect the road lane and this require extra processes to be performed on the video so that the system will be able to take a decision to move forward, stop, turn right or left ,....etc. However system detects bumps and holes then saves its location to avoid them later on and this feature require reading sensors data ,doing some operations to detect anomalies and a memory to save the location of the anomaly.Moreover, this system should be integrated with another system that simulate car system especially that part which simulate the car motion and driving behavior.So that Stereo Pi was the best solution for this case as Stereo Pi contains CM3+ Compute Module which provides Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.2GHz, 1GB LPDDR2

SDRAM, 8GB/16GB/32GB eMMC Flash memory, or a Lite variant without eMMC Flash memory.Also StereoPi allows users to attach two Camera Modules to their Raspberry Pi Compute Module — it's a great tool for building stereoscopic cameras, 360^{o} monitors, and virtual reality rigs. (CM3+). This newest version of RaspberryPi flexible board for industrial applications offers over ten times the ARM performance, twice the RAM capacity, and up to eight times the Flash capacity of the original Compute Module. StereoPi is an open source stereoscopic camera based on Raspberry Pi. It can capture, save, livestream, and process real-time stereoscopic video and images. StereoPi opens up countless possibilities in robotics, AR/VR, computer vision, drone instrumentation, panoramic video, and more. For the storage part, it is dynamic storage as microSD card slot is used for the operating system of the raspberry pi and for the running program it self. (accessed by Raspberry Pi CM3/3+ Lite)

6 Design constraints

6.1 Standard constraints

N/A.

6.2 Hardware limitations

- Our Prototype car must have 3 or more +9v Batteries.
- Our car must have a Raspberry pie 3+ so it can handle the amount of process it makes.
- Raspberry pie must have at least a memory card of 8 Gigabytes.

6.3 Others as appropriate

- Our car must be always connected to network as long as the self-driving mode is on to upload the anomalies data to the server.
- The two cameras (Stereo Pi) on the car must not move and be hold tightly so it don't affect the road lane classification process badly.
- The MPU (Accelerometer and Gyroscope) must be hold tightly on the car without moving so it don't affect the road anomalies classification process badly.

7 Other non-functional attributes

7.1 Binary Compatibility

Our system must run on noobs operating system to run the raspberry pi ,also the noobs operating system must have Arduino IDE and Python IDE.

7.2 Performance and speed

The autonomous car must be fast in everything such as: save the coordinates of any anomalies in real time on cloud, speed of alerting the driver before any saved anomalies, speed of processing many frames of road video in one second to take the right decision, speed of read and processing sensor's data from hardware. Our system must have no delays because any small delay can lead to losing someone's life so the system must be accurate to millisecond.

7.3 reliability

Any driver can rely on our autonomous car as the sensors reading will be very clear and accurate so everything about decision making will be a piece of cake with the help of the artificial intelligence and with high speed so the driver will be relaxed will using the car. but other wise if the car isn't connected to the internet the car won't be able to retrieve or save the anomalies location.

7.4 safety

Make the driving as safe as possible and it is useful for disabled people we can't drive for themselves so our system will help them go where ever they want by an easily way and without need for uber or taxi or anyone.

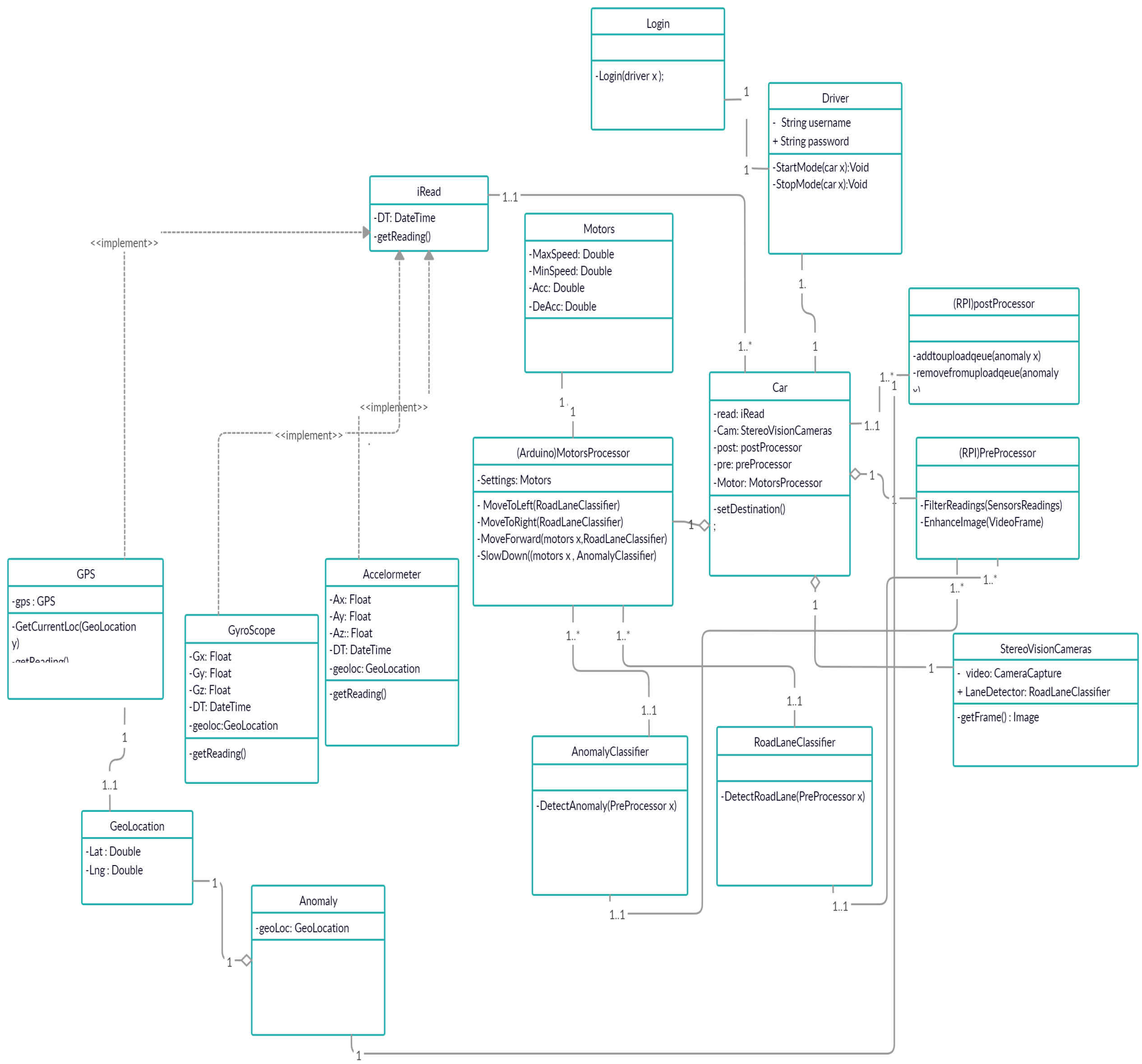
7.5 Re-usability

As in our car there are pre-processing for images and hardware readings, so anyone who is interested in embedded systems can re-use our pre-processing component.

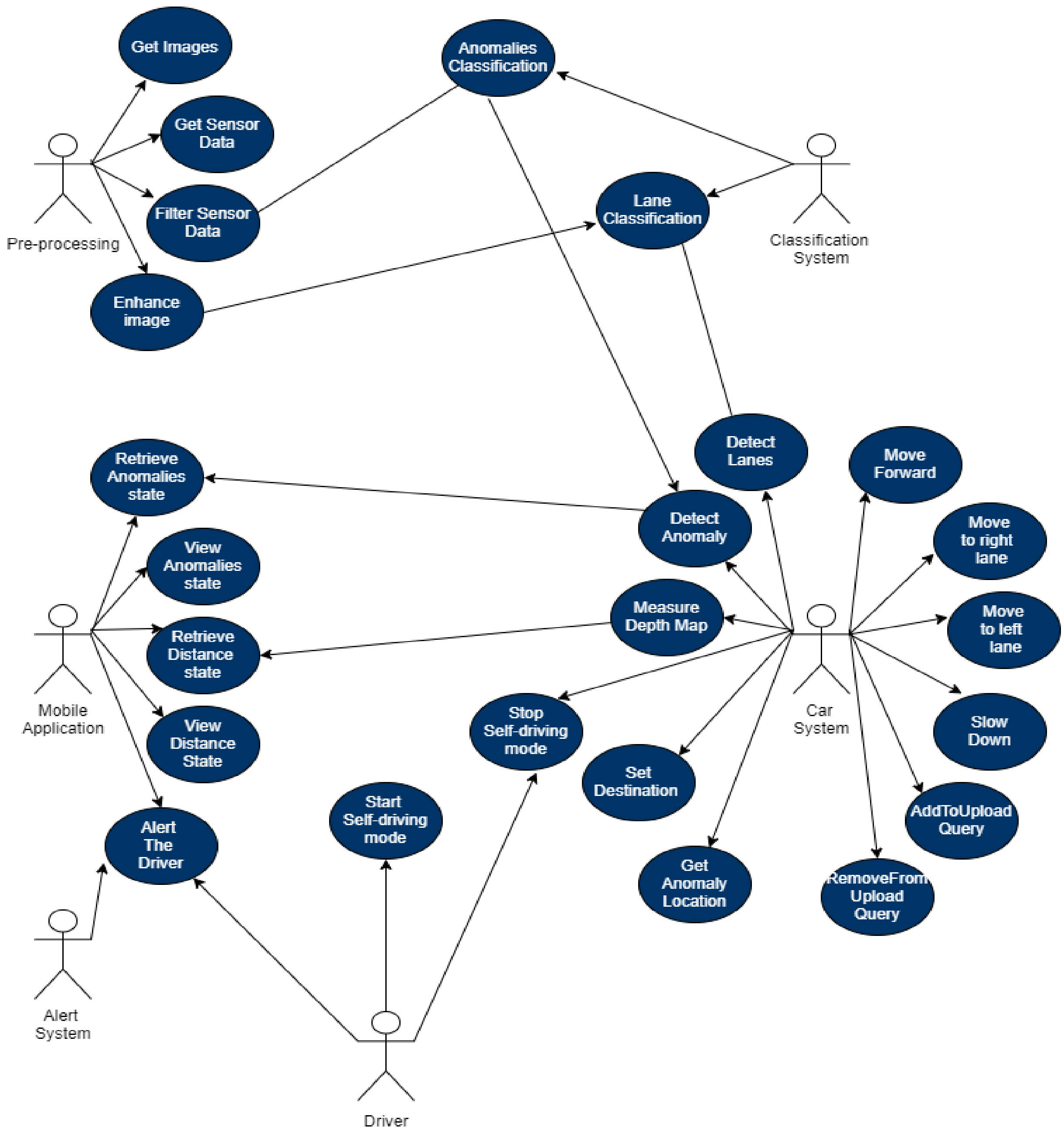
7.6 Extensibility

our car will be integrated with many web services like google maps api to update the positions of any anomalies.

8 Preliminary Object-Oriented domain analysis



9 Operational scenarios



9.1 Car Scenarios

1. Scenario 1
   - Pre-processing System takes images and enhance them.
   - Classification system takes the enhanced images and starts to classify the exact lane dimensions.
   - After classification the car system will have clear recognition of the lane and the car will by it's lane and take the right decision based on this info.
2. Scenario 2
   - Pre-processing System takes sensor readings and filter them.
   - Classification system takes the filtered readings and starts to classify the road anomalies.
   - After classification the car system will save the location of that anomaly.
3. Scenario 3
   - According to the pre-processing and classification system the car will have the total decision to change the lane .
4. Scenario 4
   - According to the pre-processing and classification system the car will have the total decision to increase/decrease its speed before any bump or hole.

9.2 Driver Scenarios

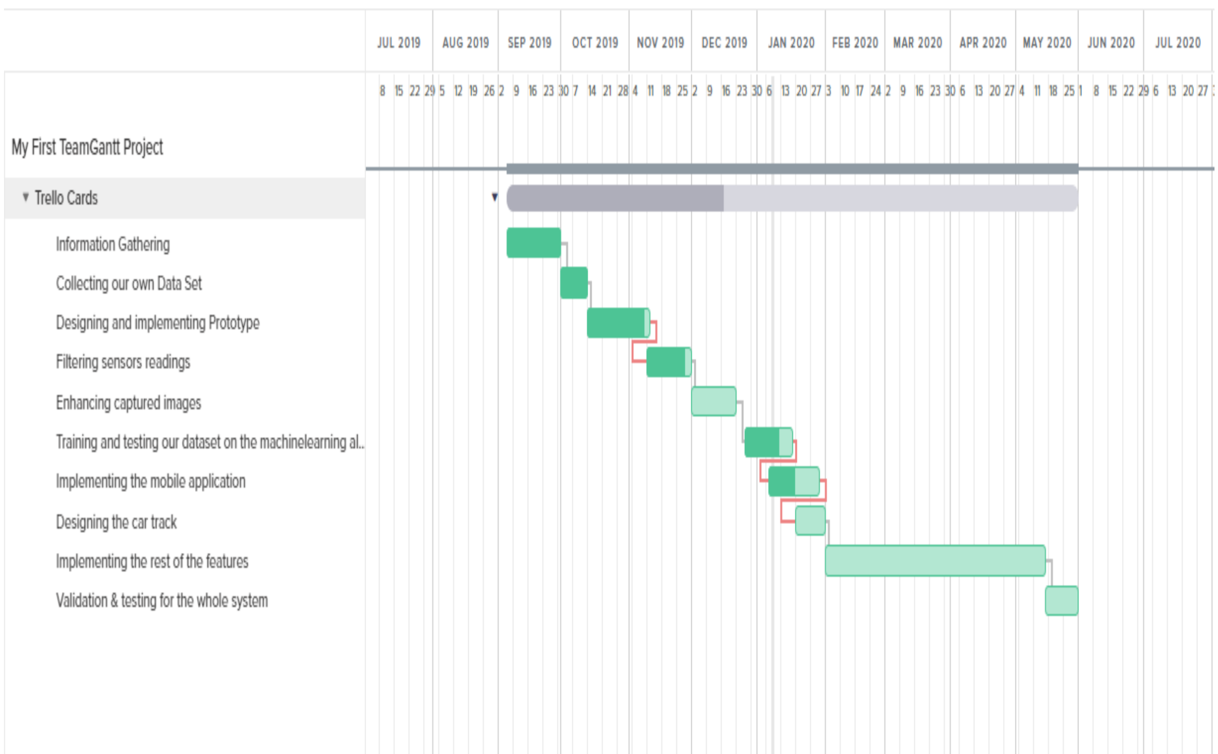
1. Scenario 1
   - if there is any thing abnormal(anomalies, another car) the driver will Receive the notification from the application.
2. Scenario 2
   - Driver can stop the self-driving mood.

9.3 Mobile App Scenarios

1. Scenario 1
   - Retrieve the anomalies state from the car.
   - view this anomalies to alert the driver that there is a bump or holes in front of him.

2. Scenario 2
    - Retrieve the distance between the car and any object.
    - view this depth map between any object and our autonomous car to alert the driver.
3. Scenario 3
    - if there is any thing abnormal(anomalies, another car) the app will send notification to the driver to alert him and to make him safe from any sudden action.

10 Preliminary schedule analysis



11 Preliminary budget analysis

Name	Quantity	Cost	Payment
StereoPi	x1	2,800.00	2,800.00
Arduino UNO	x2	145	290
Battery (9v)	x2	35	70
ChargeableBattery (9v)	x1	130	130
Battery connector	x3	10	30
Battery Charger	x1	35	35
2WD Robot Car	x1	175	175
L298N Motor Driver	x1	80	80
MPU-6050	x1	95	95
Bluetooth HC-50(ZS-040)	x1	125	125
Ultrasonic Sensor	x1	48	48
Wires	x2	22	44
lcd	x1	65	65
Total Payment			**3,987.00**

12 Appendices

12.1 Definitions

- Ultra-Sonic Sensor: A type of sensors which generate high frequency sound waves and evaluates the echo which is received back by the sensor.
- Gyroscope: is a device that can measure and maintain the orientation and angular velocity of an object
- Raspberry-Pi 3: A series of small single-board computers
- Accelerometer: A device that measures proper acceleration.
- Stereo-Vision dual cameras: A type of camera with two or more lenses with a separate image sensor or film frame for each lens.

12.2 Abbreviations

- GPS: Global Positioning System.
- SVM: Support Vector Machine is a model with associated machine learning algorithms that analyze data used for classification and regression analysis.
- CNN: convolutional neural network is a class of deep neural networks.
- MPU: Micro-Processing Unit

12.3 Collected material

N/A

13 References

[1] Nugraha, Brilian amp; Su, Shun-Feng amp; Fahmizal,. (2017). Towards self-driving car using convolutional neural network and road lane detector. 65-69. 10.1109/ICACOMIT.2017.8253388.

[2] Q. Memon, M. Ahmed, S. Ali, A. R. Memon and W. Shah, quot;Self-driving and driver relaxing vehicle,quot; 2016 2nd International Conference on Robotics and Artificial Intelligence (ICRAI), Rawalpindi, 2016, pp. 170-174. doi: 10.1109/ICRAI.2016.7791248.

[3] Chaocheng Li, Jun Wang, Xiaonian Wang and Yihuan Zhang, quot;A model based path planning algorithm for self-driving cars in dynamic environment,quot; 2015 Chinese Automation Congress (CAC), Wuhan, 2015, pp. 1123-1128. doi: 10.1109/CAC.2015.7382666

[4] T. Do, M. Duong, Q. Dang and M. Le, quot;Real-Time Self-Driving Car Navigation Using Deep Neural Network,quot; 2018 4th International Conference on Green Technology and Sustainable Development (GTSD), Ho Chi Minh City, 2018, pp. 7-12.

[5] Ali Chishti, Syed , Riaz, Sana , BilalZaib, Muhammad , Nauman, Mohammad. (2018). Self-Driving Cars Using CNN and Q-Learning. 1-7. 10.1109/INMIC.2018.8595684.

[6]Yeshodara, Naveen , Nagojappa, Namratha Kishore, Nikhitha. (2014). Cloud Based Self Driving Cars. 1-7. 10.1109/CCEM.2014.7015485.