DUAYENLER Ltd. Şti.

Middle East Technical University

Department of
Electrical and Electronics Engineering

EE493 ENGINEERING DESIGN I

Car Chasing Robot
Conceptual Design Report

Supervisor: Assoc. Prof. Emre Özkan
METU EE / C-112

Project Start: 4/10/2018
Project End: 26/5/2019
Project Budget: $450

Company Name : Duayenler Ltd. Şti.

Members	Title	ID	Phone
Sarper Sertel	Electronics Engineer	2094449	0542 515 6039
Enes Taştan	Hardware Design Engineer	2068989	0543 683 4336
Erdem Tuna	Embedded Systems Engineer	2617419	0535 256 3320
Halil Temurtaş	Control Engineer	2094522	0531 632 2194
İlker Sağlık	Software Engineer	2094423	0541 722 9573

December 26, 2018

Contents

1 Executive Summary

Recently, with the quick development of the computers and internet, a revolutionary change is about happen. The operation speed of electronics have dramatically increased as the sizes have shrank. Thus, developing and integrating new systems into the cars are easier than it has ever been. Along with such advances, as a result, intelligent cars have been around for a few years. The intelligent vehicle technologies enable simpler vehicle operations and better driving safety. Such technologies involve autonomous driving, lane departure assistance, ambient dependent headlight adjustment and other safety related features. To address the needs in the development phases of the aforementioned technologies, DUAYENLER Ltd. Şti. (DUAYENLER) is founded. DUAYENLER aims to be one of the Lodestars in the industry with its innovative approaches.

The company consists of talented engineers from different but closely related fields, namely, computer, electronics and control. This combination is what makes DUAYENLER advantageous in R&D phase. Easy looking technologies involve complex development phases. And complexity can only be resolved by a team work. The members of the company are self-aware and cooperate closely to each other. With all its synergy, DUAYENLER is capable of accomplishing the tasks and overcoming possible problems.

The main technology that DUAYENLER focuses on is autonomous driving. Such technology yields a contemporary and innovative solution for car industry. The development consists of combination of several systems. The sensing system is responsible for understanding and mapping the environment properties such as detecting lane boundaries and possible obstacles. This system is one of the most important elements in the project together with computation system. Not only being able to map surroundings as image is important but also extracting meaningful data out of them and convert them to a useful data for rest of the systems. The computation unit is the core system of the project. Furthermore, the processed data must be output as a physical phenomenon, that is, steering of the wheels. This is realized by the combination of driving and motion subsystems. Since a vehicle is generally not solo, the project consists of a system dedicated to communicate with other vehicles. This is the communication system. As anticipated, the whole system must be assembled on a single body. The structure system realizes such needs in the project.

The aim of this report is to give reader a solid understanding of the project from DUAYENLER's point of view. The duration of the project is expected to last 33 weeks, from the beginning of October 2018 to May 2018. The estimated cost for research and development phase is about $ 450 whereas mass manufacturing would not exceed $200. Along with the vehicle, the customer will be provided with deliverables such as technical manuals, elliptical path, rechargeable battery and the charger. The vehicle will have two (2) years of warranty.

2 Introduction

DUAYENLER is established with the aim of developing autonomous car technologies for near future. To serve that purpose, Car Chasing Project is initiated by the company. The project can be summarized as a vehicle that can autonomously follow a path and detect the other surrounding vehicle as well as communicating them to have a reliable driving environment. With this project, the company aims to accomplish the following objectives:

1. Sensing the environment and other vehicles on the roads
2. Automatic adaptive lane detection
3. Self driving
4. Autonomous wireless communication with surrounding counterparts

A considerable amount of effort and work force has been put on the project to fulfill the required objectives. So far, the team has figured out several important steps towards the realization of the project.To start with, the wireless communication between the vehicles is modeled and implemented. A reliable communication environment is established using Wi-Fi protocol. Currently, the vehicles can communicate with each others by means of associated handshake protocol messages in a race scenario. Secondly, computer vision algorithms are developed and implemented as a solution to lane detection problem. The algorithms are developed based on open source computer vision library OpenCV. To obtain a direction predicting results, color thresholding, edge detection, hough transform algorithms are used respectively. Furthermore, the communication between image processing platform and microcontrollers for motor driving is constructed. It is the essential part of solving the self driving problem. On the mechanical part, different motor&wheel combinations are tested to obtain the best performance. To test the computer vision on board, a prototype vehicle is assembled and necessary equipment is mounted on it. Currently, the team is working on the improvement of computer vision algorithms.

In this report, the company provides technical details about the implemented solutions, other possible solution alternatives with objective comparisons as well as a clear action plan showing the necessary further steps for realization of the project. The emphasis on this report is primarily put on the detailed analysis of proposed solutions, supported with relevant test results in both system and subsystem level. In addition, future plans including new test designs for current solutions as well as for other alternatives, the action plan in case of unexpected outcomes by clearly specifying the responsibilities of each member in the team.

3 Overall Project

The main objective of this project is to design and produce a self driving mini-car that can follow a path with a varying properties with in a 200 dollar budget. The project can be investigated under six main systems and their twelve subsystems. *Figure 1* shows the organization structure of the project.

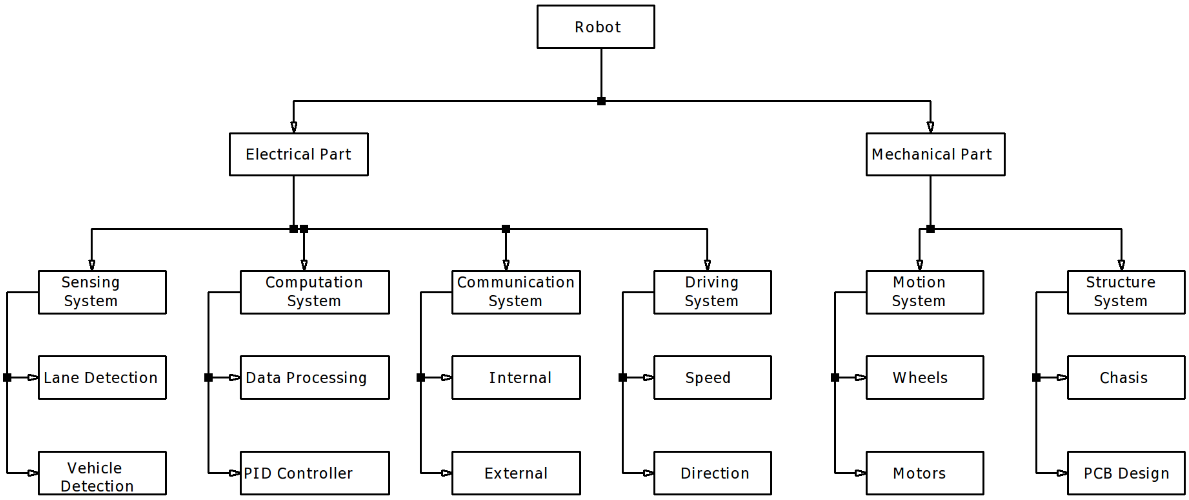


Figure 1: System organization of the project

1. Sensing System

   This system is responsible for interpreting data from the environment. It has two subsystems namely,

   (a) *Lane Detection Subsystem* which is responsible for detecting sides of the path as its name suggests

   (b) *Vehicle Detection Subsystem* which is responsible for detecting opponent vehicle if it is close to the vehicle more than 5 cm

2. Computation System

   This system is responsible for computational works of the vehicle. The system mainly give meaning to data generated by the sensing system. It has two subsystems namely,

   (a) *Data Processing Subsystem* which is responsible for processing the output data of lane detection unit and produce data for PID control unit.

   (b) *PID Controller Subsystem* which is responsible for controlling the motors of the vehicle.

3. Communication System

   This system is responsible for communication inside the vehicle and communication between vehicles. It has two subsystems namely,

    (a) *Internal Communication Subsystem* which is responsible for communication inside the vehicle mainly the communication between Raspberry Pi and Arduino.
    (b) *External Communication Subsystem* which is responsible for the communication of the vehicle with the outside world mainly with the opponents.

4. Driving System

   This system is responsible for the motion of the vehicle. Two parameters that are the direction and the speed of the vehicle is controlled by this unit accordingly to the information coming from controller. It has two subsystems namely,

    (a) *Direction Subsystem* which is responsible for the orientation of the vehicle and keeps the road and the vehicle aligned.
    (b) *Speed Subsystem* which is responsible for the overall speed of the vehicle by adjusting it considering other effects on the vehicle.

5. Structure System

   This system is responsible for mechanical structure of the vehicle. Placement and orientations of both electrical and mechanical components are considered in this system. It has two subsystems namely,

    (a) *Chassis Subsystem* which is responsible for the connections of mechanical components in the vehicle.
    (b) *Printed Circuit Board Subsystem* which is responsible for the placement of electrical components.

6. Motion System

   This system is responsible for maintaining mechanical feasibility of the driving system. It has two subsystems namely,

    (a) *Wheels Subsystem* which is responsible for the design of the wheel structure depending on the required speed-torque parameters.
    (b) *Motors Subsystem* which is responsible for the correct selection of the motor that is suitable to requirements.

General interaction of these subsystems with each other can be investigated at system diagram of the project at *Figure 2*.

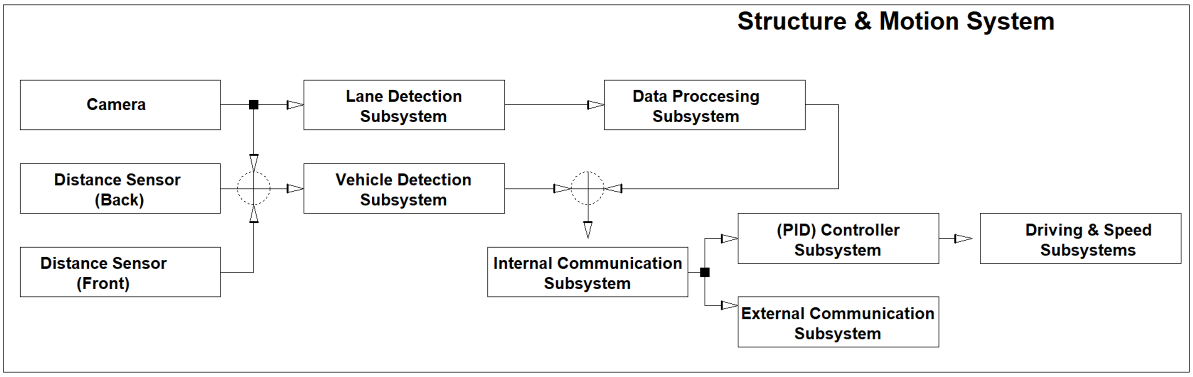


Figure 2: System Diagram of the Overall Project in Subsystem Level

4 Detailed Solutions to Subsystems

V-Model is a very popular tool for system engineers to plan their projects. To ease the project tracking process, the V-Model was constructed by the DUAYENLER. The overall look of the V-Model can be seen at *Figure 3*. This section includes the explanation, requirements, test procedures and test results for the subsystems.

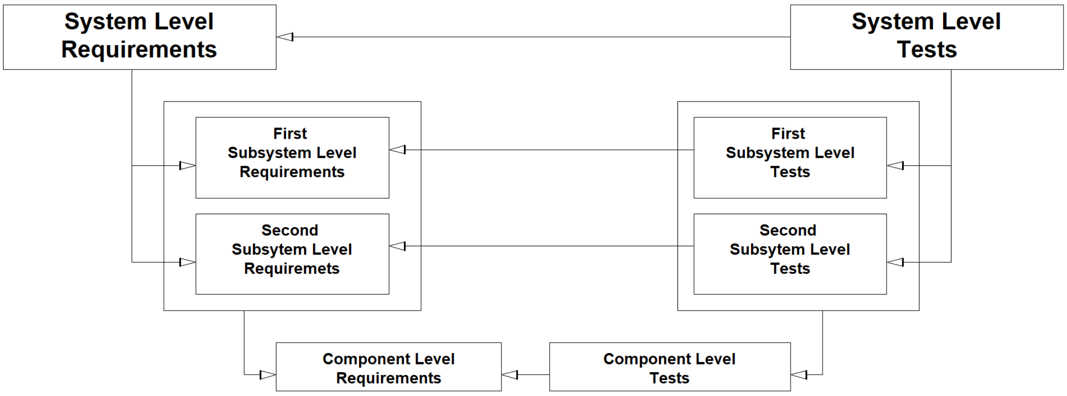


Figure 3: General Structure of the V-Model Utilized in the Project

4.1 Sensing System

Sensing system has two main subsystems which are Lane Detection subsystem and Vehicle Detection subsystem. The requirements of this subsystem are listed below:

- The system should detect the sides of the road.
- The system should not be effected from external disturbances.
- The system should detect the opponent vehicle.

4.1.1 Lane Detection Subsystem

The subsystem basically detects the lane. This subsystem uses OpenCV libraries for processing camera frames. The input is captured from a Raspberry Pi camera that is mounted to the vehicle. The captured frame is firstly, preprocessed by a denoising filter and HSV color filter. The edges in the frame is detected by Canny Edge Detector algorithm. The output of Canny is a binary image filled with ones and zeros. The resulting binary image is processed with Hough Line detector to find pixels that possibly form a line. After the lines are found, the processed frame is sent to Data Processing subsystem. The block diagram of this subsystem is given in *Figure 4*. The requirements of this subsystem are listed below:

- The subsystem should be able to detect only the shades of green color
- The subsystem should be able to detect edges in the camera frame in any light condition
- The subsystem should be able to tell differences between disturbances and lane
- The subsystem should be able to interpret the middle of the lane if both sides are present at the frame

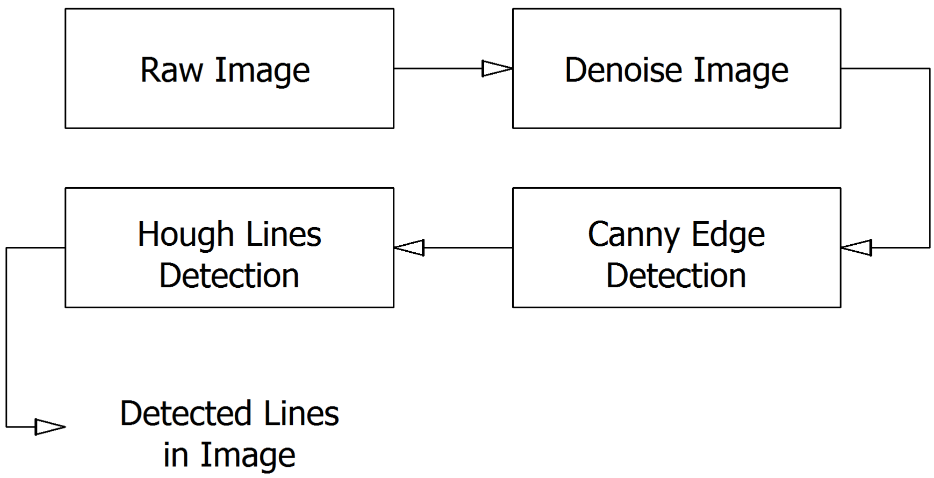


Figure 4: The Proposed Algorithm for the Lane Detection.

4.1.1.1 Main Solution for Lane Detection Subsystem

The first processing on the captured frame is denoising by blurring. The blurring filter is a GaussianBlur of (3x3) matrix with zero variance both in x and y directions. Next, the filtered image is color filtered in HSV color space. The filter range of the HSV filter is adjusted only to include shades green. The lower bound for HSV filter is [H=60, S=120, V=106] and the higher bound is [H=82, S=255, V=235]. Later the color filtered image is processed by Canny Edge Detector. This process eliminates all pixels except those are constituting an edge. Edge pixels are usually formed when there is a transition from one object to other. To find pixels constituting a line, Hough Lines function is used.

Hough Lines function outputs two pixels points on the image that form a line. The output of the function is tens of such line points. The accuracy of the function can be adjusted by playing with the input parameters. The function basically draws all lines that might go through a point in the free space (in polar coordinates) and intersects those lines. If the origin points of the intersected lines are within a specified bound, then the line is defined and it can be expanded further. If found points on a line don't exceed a threshold, that line is discarded. The adjustment of such parameters are done with trial and error approach. An important note is that this function is probabilistic, meaning that even if the input frame is never changed, the output line points would be close to previous point but not the same. An example call for this function together with its parameters is given *Script 1.* The output, line detected frame, is sent to Data Processing subsystem.

```
HoughLinesP(input=img_threshed, output=line,
rho=1, theta=CV_PI / 180, threshold=15,
minLineLength=30, maxLineGap=40);
```

Script 1: Hough Lines Function with its Parameters

4.1.1.2 Alternative Solutions for Lane Detection Subsystem

After the module demo result of the camera-oriented lane detection, former solutions are considered to support camera result and enhance robustness of the tracking. Possible future solutions are the followings:

- Light sensor: The purpose of using this sensor is to adjust color filtering in the lane detection. In the test phase, it is observed that color filter in HSV space had inaccuracies such as detecting colors other than green. Even though HSV mostly eliminates the illumination change in a color, an adaptive threshold setting might be a good option for color filtering to work more robust in extreme low and extreme high lighting conditions. Lighting sensor will have the luminosity of the medium and this data should adaptively set the color filter range.

- Laser sensor: As IR sensor array solution, which is already considered at the beginning of the term, laser approximate sensors can be constructed as array and used to detect edges of the lane. The expected enhancement is that laser sensors are more robust under extreme illumination conditions. To clarify the solution, two laser sensor arrays sense both sides of the lane. According to array's data, vehicle orients itself.

- Color sensor: This approach is similar to laser sensor solution, regarding the sensor array construction. However, color sensor working principle is based on RGB recognition, so in this solution green output of the arrays is focused point. According to green output, path will be detected.

4.1.1.3 Lane Detection Subsystem Tests

The test procedure that will make sure correct functioning of this subsystem are list in the following sections.

- Light Condition Test
    1. Mirror the Raspberry Pi screen into Laptop via VNC
    2. Execute the lane detection algorithm in Raspberry Pi
    3. Change the location of the camera and Pi to conduct test
    4. Observe the results in different locations
    5. If the visible lane sides can be detected without any additional object, the result of the test can be considered as success.
- Visual Disturbance Test
    1. Mirror the Raspberry Pi screen into Laptop via VNC
    2. Execute the lane detection algorithm in Raspberry Pi
    3. Put different objects into lane
    4. Observe the results with different disturbances
    5. If the objects outside of lane is not detected and the objects inside the road only detected only at its border with road, the result of the test can be considered as success.

4.1.1.4 Results of Lane Detection Subsystem Tests

The lane detection tests were conducted for the detection algorithm of the camera. The results were promising. The algorithm sweeps up the surrounding disturbances completely. The sample outputs together with Data Processing subsystem are shown in *Figure 5*.

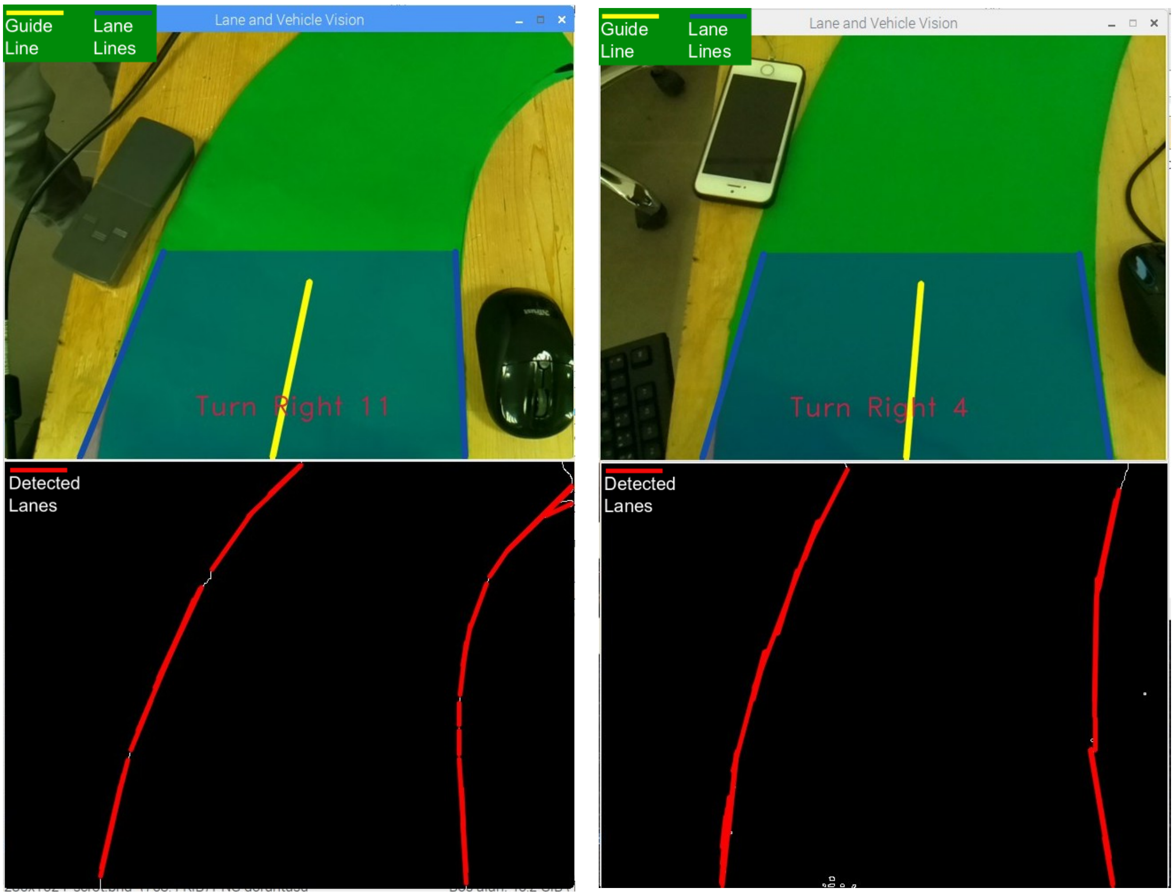
Before making the camera the main solution for the lane detection subsystem. Several tests were conducted using infra-red and color sensors. The tested infrared sensors was TCRT5000 & QRD1114, and the tested color sensor was TCS3200. The tests were conducted under different illumination conditions and at different locations. These tests conducted for these sensors were modified versions of the tests conducted for the tests conducted for the *Vehicle Detection Subsystem*. The test procedures can be examined at *Section 4.1.2.3*. The test result for initial test in which two infrared sensors were tested to see whether they can differentiate the height of road from a fixed height can be seen at *Table 1*. Although the results were promising especially for TCRT5000, the tests conducted at CCC (Cultural and Convention) revealed that the sensor is not able to cope with highly reflective floors under direct sunlight.

Table 1: The Results for the IR Sensor Tests

Sensor Name	Height from base	Output Value for Base	Output Value for Path	Difference (Percentage)
TCRT5000	4	500	900	44 %
TCRT5000	5	690	950	27 %
QRD1114	4	810	960	15.6 %
QRD1114	5	940	970	3 %

4.1.2 Vehicle Detection Subsystem

The subsystem is the first step of safely competing with an opponent in a racing path. By the help of two distance sensors at the back and front of the vehicle the subsystem aims to find whether there is a opponent vehicle nearby.



(a) (b)

Figure 5: Lane Detection Test Results.

4.1.2.1 Main Solution for Vehicle Detection Subsystem

This subsystem uses two time of flight distance sensor which is enhanced IR sensor. One at the back of the vehicle responsible for detecting the chasing opponent and one at the front of the vehicle responsible for detecting the chased opponent. The subsystem produces positive output if the chasing vehicle or chased vehicle is within a range of 5 cm from the vehicle. Since the sensor reading is planned to be performed using Arduino, the required trigger for handshake protocol can be easily send through serial commutation or directly implemented in the Arduino as an alternative. The requirements of this subsystem are listed below:

- The subsystem should detect the opponent to be caught with in a 5 cm
- The subsystem should detect the chasing opponent if it reaches from back with in a 5 cm
- The subsystem should trigger the handshake protocol

4.1.2.2 Alternative Solutions for Vehicle Detection Subsystem

- Ultrasonic sensor: These can be another approach to this problem. They send a sound wave and take back the echo, then give a PWM voltage related to the distance. However, using ultrasonic sensors can cause problems if the opponent also uses ultrasonic sensors due to the interference. Besides, when measurement is angled, the sensor reading fails. The reason why this is not the first option is that the specified path involves many sharp turns.
- Infrared sensor: Using these sensors Infrared sensors are the most trivial solution to measuring distance. However, IR sensors are weak to use in outside, especially sunshine. Although they are so sensitive to sun, they can still be a solution, if the first choice is failed, unexpectedly, this solution can be reconsidered.
- Camera Support: If required the opponent vehicle can be detected using the camera. As determined in the *Standart Committees*, all vehicles will carry a box that is distinguishable from its surroundings. Using the predefined measurements of this rectangle, actual measurement between vehicle can be found. However, due to budget limitations, this solution may not be applicable for detecting the vehicle at the rear.

4.1.2.3 Vehicle Detection Subsystem Tests

- Front Vehicle Detection Test in Closed Environment: This test aims to validate whether the subsystem is capable of detecting the opponent vehicle that is close to the vehicle more than 5 cm from the front. The test procedure is as follows;

  1. Make the connection of the desired sensor and Arduino properly
  2. Hold the sensor at an angle of 90 degree with respect to ground
  3. Place the test object 5 cm in front of the desired
  4. Observe the output of the subsystem
  5. Repeat the step 3 & 4 with different distances
  6. If the output of the subsystem generates logical positive for distances smaller than 5 cm and logical zero for distances greater than five, the test result can be considered as success

- Rear Vehicle Detection Test in Closed Environment: This test aims to validate whether the subsystem is capable of detecting the opponent vehicle that is close to the vehicle more than 5 cm from the rear. The test procedure is as follows;

    1. Repeat the test steps of the *Front Vehicle Detection Test in Closed Environment* with the desired sensor for the desired rear sensor.

- Angled Approach Test: This test aims to validate whether the subsystem is capable of detecting the opponent vehicle that is close to the vehicle more than 5 cm if the vehicles are not perfectly aligned. The test procedure is as follows;

    1. Make the connection of the desired sensor and Arduino properly
    2. Hold the sensor at an angle of 90 degree with respect to ground
    3. Place the test object 5 cm in front of the sensor with 30 degree angle with respect to the sensor
    4. Observe the output of the subsystem
    5. Repeat the step 3 & 4 with different distance and angle values
    6. If the output of the subsystem generates logical positive for distances smaller than 5 cm for all angle values with respect to sensor and logical zero for distances greater than 5 cm, the test result can be considered as success

- Vehicle Detection in Different Sunlight Conditions Test: This test aims to validate whether the subsystem is capable of detecting the opponent vehicle that is close to the vehicle more than 5 cm in different sunlight conditions. The test procedure is as follows;

    1. Repeat the test steps of the *Front Vehicle Detection Test in Closed Environment* in CCC (Cultural and Convention) ground under direct sunlight
    2. Repeat step 1 in CCC (Cultural and Convention) under artificial light, in other words, under no direct sunlight conditions
    3. Repeat steps 1 & 2 for different locations of E Building including Graduation Laboratory
    4. If the output of the subsystem generates logical positive for distances smaller than 5 cm under all light conditions and logical zero for distances greater than 5 cm, the test result can be considered as success

4.1.2.4 Results of Vehicle Detection Subsystem Tests

All the test procedures mentioned at the **Section 4.1.2.3** were applied to the ultrasonic sensor (HC-SR04) and two infrared sensors (TCRT5000 & QRD1114). The ultrasonic sensor were showed very inaccurate result especially in *Angled Approach Test.* The test results can be examined at *Table'2.* Considering the fact that the path itself is elliptical and there would always be an angle between vehicle even though it may be very small for some cases, it was decided that ultrasonic sensors are not a good choice for this subsystem. However, it is always possible that these sensors can be supportive sensors for the subsystem.

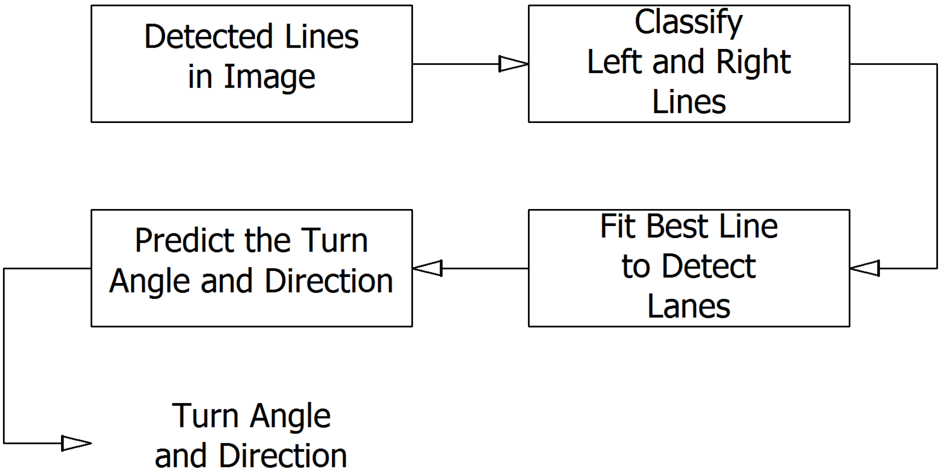


Figure 6: The Proposed Algorithm for Data Processing Subsystem.

Table 2: The Results for the Angled Approach Test for HC-SR04

Actual Distance	The Angle	Measured Distance
3 cm	90	3.15 cm
5 cm	90	5 cm
20 cm	90	20.01 cm
40 cm	90	40.25 cm
5 cm	45	6.78 cm
20 cm	45	28.8 cm
30 cm	45	42.4 cm

Unlike ultrasonic sensors, infrared sensors showed very accurate result inside the closed environments like laboratory under artificial lights. However, the results under direct sunlight especially in CCC were not as good as expected. Thus, it was decided that the main solution should be an enhanced version of infra-red sensors namely the ones utilizing the "time-of-flight" concept. Moreover, laser sensors might be good alternative for these subsystems.

4.2 Computation System

Computation system has two main subsystems which are Data Processing subsystem and PID Controller subsystem. The requirements of this system are listed below:

- The system should be able to produce middle line to follow
- The system should be able to control the robot

4.2.1 Data Processing Subsystem

The data processing subsystem is the main computation unit for the vehicle. The main objective of this subsystem is to give PID controller subsystem a meaningful data from unorganized data coming from lane detection subsystem.

4.2.1.1 Main Solution for Data Processing Subsystem

The input of this system is an edge detected binary image. Next, points of the lines are classified as left or right borders of the lane. The elimination of the wrong points are done concurrently with the classification. Then, filtered points are fitted in two separate lines to create left and right borders of the lane. As the lane borders are found, the next and the last step is to determine the direction of the vehicle. The output of this subsystem is a turning angle and a direction. The whole process is summarized in *Figure 6* as a block diagram. The requirements of this subsystem are listed below.

Data Processing Subsystem Requirements:

- The subsystem should be able to analyze data produced by sensing system
- The subsystem should be able to produce the angle information required by the controller subsystem
- The subsystem should be able to work on Raspberry Pi
- The subsystem should be able to process one frame at most in 100 milliseconds

Then next step is to classify the line points as left or right. The lines are firstly eliminated according to their slopes. If slope of a line is not in invalid slope region of ± 0.005, then it is a valid line. This process is done to get rid of unnecessary low sloped lines. After elimination, line points set must be determined as left or right. At this point, this classification is done according to double checking. The center of the image, that is 320th vertical pixel. If initial and final horizontal points of the image is in the same half, the line belongs to that half. This method works nicely in most regions of the path. However, it is a bit error prone in case of a sharp turning angle or losing one of the lanes . This algorithm will be improved to be more robust.

After finding the all left and right line points, actual left and right lines are constructed by applying the least square method to the both point sets. The result of this method is 4 points (x1,y1,x2,y2) to refer the left lane line and 4 points to refer the right lane line.

The last process on the image is predicting the turn angle and direction. The prediction of direction is done by comparing the slopes of the left and the right lines. The turning is to be made to the side of the lane line having less slope. The angle is determined by computing the angle between the normal line of the current direction and guide line. Guide line is constructed with the average of the middle points of the right and the left lines and current point of the vehicle. Vehicle's current point is assumed to be in the middle of the beginning of the image. Thus, the beginning point of the

```
std::vector<cv::Vec4i> lines // like a 4x4 matrix
double slope_thresh = 0.005; // absolute threshold slope
for (lines_points)
        startP = Point(x1, y1);
        endP = Point(x2, y2);
        line_slope = startP/endP;
        if(abs(line_slope) >= slope_thresh) line_is_valid;
        else line_is_not_valid;

for (lines_is_valid)
        if(x1>320 && x2>320) it_is_right_line;
        else if(x1<320 && x2<320) it_is_left_line;
        else discard_the_line;
```

Script 2: The Algorithm to Classify the Lane Lines as Right or Left

guideline is the current position of the vehicle and the endpoint is the target point to be reached. The output of this processing is shown in *Figure 7*. The turning angle and the direction is sent to PID Controller subsystem.

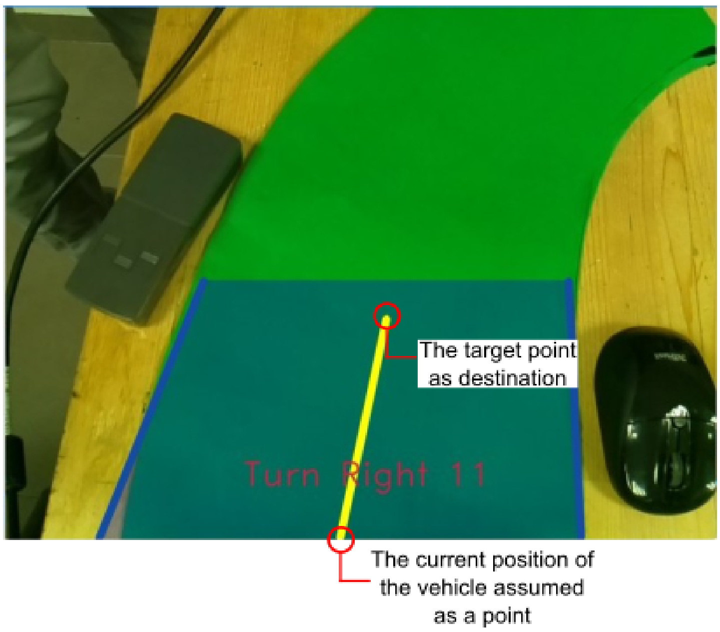


Figure 7: The Prediction of Turn Angle and Direction.

4.2.1.2 Alternative Solutions for Data Processing Subsystem

The proposed algorithm for data processing has some flaws that are already discussed. To overcome those problems, DUAYENLER has some alternative solutions. An improvement might be on finding lane boundaries. This operation is currently realized with Hough Lines function. Another approach would be to scan rows an accurately edge detected image. If a white pixel is encountered, this would imply a lane boundary and the last time a white pixel is encountered would mean te other lane boundary. If the edge detected image contains noise, the solution might yield a wrong result. However, this can be overcomed by setting maximum number of white points that can be detected in a row.

Another improvement could be on positioning on the lane. The proposed algorithm assumes that the vehicle follows the lane always in the center. This assumption is not the case always. The new algorithm will differentiate between the center of the lane and center of the capture image. By averaging coordinate points of both lane boundaries the actual lane center can be found and this point must be in some proximity with respect to the center of the image.

4.2.1.3 Data Processing Subsystem Tests

This test will enable if data processing will generate the desired output, namely turn direction and angle. Test procedure is as follows:

1. Link the output of Lane Detection subsystem to Data Processing subsystem.
2. Asses if the output coincide with physical reality of the path

4.2.1.4 Results of Data Processing Subsystem Tests

The tests are done. The results are positive and reflects the expectations. The turn angle and direction are properly output. A flaw of this subsystem is that both right and the left lane lines must be determined. Otherwise, the prediction does not give stable results. This must be improved together with the Lane Detection subsystem regarding the code algorithm. The sample outputs can be seen in *Figure 4 and Figure 7.*

4.2.2 PID Controller Subsystem

The output of the data processing subsystem does not mean much for the rotating parts of the vehicle. The motors should be driven using some sort of closed loop system. PID controllers are the most used controller in the robotics field. The purpose of the PID controller is basically to eliminate the error from the desired steady state.The requirements of this subsystem are listed below:

- The subsystem should be able to control the motors
- The subsystem should be able to react the external disturbances

4.2.2.1 Main Solution for PID Controller Subsystem

In our case the desired steady state error to be compensated by the controller is angle information coming from the data processing unit. For this purpose discrete time version of a general PID controller will be utilized. General form for discrete time PID controller may be seen below:

$$G_{PID}(z) = K_{bias} + K_p + K_I \frac{1}{1 - z^{-1}} + K_D(1 - z^-1)$$

Using bilateral Tustin transform where

$$z = \frac{1+\bar{s}}{1-\bar{s}}$$

The discrete time controller can be designed as if it is in continues domain.

4.2.2.2 Alternatve Solutions for PID Controller Subsystem

Theoretically very similar approach for designing a controller is to design in a S-domain. For this kind of controllers, general form of the continues time PID controller may be seen below:

$$G_{PID}(s) = K_{bias} + K_p + K_I \frac{1}{s} + K_D s$$

Another alternative method is to design the controller directly in discrete domain without using bilateral Tustin transform.

4.2.2.3 PID Controller Subsystem Tests

- PID Parameters Test for Given Input: The test aims to speed up the process of finding the right PID parameters for the controller. The test procedure is as follows;

  1. Connect the Vehicle Motors to Motor Controller
  2. Connect the Motor Driver to Arduino
  3. Give the angle value that the subsystem should compensate
  4. Give the power to the motors
  5. Observe the behaviour of the vehicle
  6. If the vehicle rotates with an angle given in step 3 without any feedback given, the result of the test can be considered as success.

- Path Tracking Test: The test aims to check the validity of the PID parameters in closed loop feedback system. The test procedure is as follows;

  1. Make the necessary connection between motors Arduino and data processing unit
  2. Place the vehicle to the desired empty path
  3. Observe the behaviour of the vehicle
  4. If the vehicle can follow the path smoothly, the result of the test can be considered as success.

- Tracking a Path with Obstacles Test: The test aims to check the validity of the PID parameters in closed loop feedback system if there is obstacles. The test procedure is as follows;

  1. Make the necessary connection between motors Arduino and data processing unit
  2. Place the vehicle to the desired path with obstacles
  3. Observe the behaviour of the vehicle

4. If the vehicle can follow the path and compensate the steady state errors due to obstacles without showing oscillatory behaviour and in a reasonable time (in less than 2 seconds), the result of the test can be considered as success.

- Path Tracking Test with Physical Disturbances: The test aims to check the validity of the PID parameters in closed loop feedback system if there is a physical disturbance that effect the location of the vehicle. The test procedure is as follows;

1. Make the necessary connection between motors Arduino and data processing unit
2. Place the vehicle to the desired empty path
3. Observe the behaviour of the vehicle
4. If the vehicle can follow the path and compensate the steady state errors due to physical disturbance without showing oscillatory behaviour and in a reasonable time (in less than 2 seconds), the result of the test can be considered as success.

4.2.2.4 Results of PID Controller Subsystem Tests

Due to other limitations, the initial version of the PID controller subsystem was tested only for *PID Parameters Test for Given Input.* The test results were promising for the time being. Other tests are planning to be conducted on the subsystem in the following semester.

4.3 Communication System

Communication system has two main subsystems which are Internal Communication subsystem and External Communication subsystem. The requirements of this system are listed below::

- The subsystem should ensure safe internal communication
- The subsystem should ensure safe external communication

4.3.1 Internal Communication Subsystem

This subsystem covers the communication of the components inside vehicle. Currently, Raspberry Pi and Arduino are two components that requires communication. To prevent the large amount of cable connection, a serial communication protocol is implemented. The requirements of this subsystem are listed below:

- The microcontrollers should be able to communicate with each other via serial communication
- The internal communication speed should be compatible with the processing speed of the lane detection subsystem

```python
import serial
ar=serial.Serial("/dev/ttyUSB0",9600)
```

Script 3: Serial object decleration in Python

```c
#include "./arduinoModule/rs232.h"
int cport_nr = 24; /* /dev/ttyUSB0 */
int bdrate = 9600; /* 9600 baud */
char mode[] = { '8', 'N', '1', 0 }; // 8 data bits, no parity,
    1 stop bit
char str_send[2][BUF_SIZE]; // send data buffer
unsigned char str_recv[BUF_SIZE]; // recv data buffer
RS232_OpenComport(cport_nr, bdrate, mode);
```

Script 4: Serial communication setup in C

4.3.1.1 Main Solution for Internal Communication Subsystem

There are several serial communication protocols that can be used to maintain the connection such as SPI, I2C. However, the first choice is to use USB serial port of the Arduino. Since RPi is practically a computer, it can recognize Arduino as a device using a serial port such as /ttyUSB0 in case of a Linux based OS. When recognized, RPi can send any piece of strings to the Arduino via USB cable. The process of communication is as follows:

1. Arduino should be connected to the Pi.
2. Using Arduino IDE or any other method such as listing serial ports and checking for Arduino and so on, the serial port name should be detected
3. Baud rates of two sides should be the same. 9600 is generally enough but if needed, it can be incremented to satisfy fast communication
4. On Arduino side, `Serial.begin(9600)` command should be executed and serial port should be read repeatedly to capture the incoming data
5. On Pi side, using any language C++ or Python, messages to serial port can be send

There are minor differences when implementing the code in Python, C++ and C. Python is the most practical one:

Python has a library called serial by which any type of data can be send through serial ports. Script 3 is used to declare a serial object. Then using `ar.write("some string \r".encode())`, the string "some string" can be send to Arduino. Note that "\r" carriage return character carries a high importance because it shows that a string is terminated and any other incoming data belongs to the new piece of string.

As alternatives, the implementation on C is also examined. Sample codes to implement the same communication in C is in Script 4. Since it is a low level language, specification of buffer size and other parameters should be done in the code.

On Arduino side, there are also several option that we can read the incoming data. Using `Serial.read()` command is one of the simplest solutions. However, it contains some issues like conversion from string to integer and when to stop. Furthermore, the incoming data should be considered in character basis for the exact control.

4.3.1.2 Alternative Solutions for Internal Communication Subsystem

The communication between Raspberry Pi and Arduino can be realized by the use of other peripheral protocols or libraries.

There is a `SerialCommand.h` library for the Arduino which allows executing a function depending on the incoming string. Using .addCommand(”str”,func) of the library any function can be associated with any string coming from serial port. Moreover, the functions can have argument. For example, let the string ”PWMSET” be execute a function setpwm() but the PWM value is required. If incoming string is of the form ”PWMSET 150”, using .next() function of the library, the value 150 can be read and converted into integer and interpreted as the PWM value to be set.

Another solution is to use I2C or SPI protocols to set communication between Arduino and Raspberry Pi. These options require simple wiring connections between the pins. A point to remark is being cautious to operating voltage levels of pins of Arduino and Raspberry Pi.

Besides aforementioned alternatives, Wi-Fi communication can also be considered between Arduino and Raspberry Pi. This solution requires deployment of a Wi-Fi module on Arduino. Some hardware exists in market such as ESP8266 and EMW3165. This solution would require a Wi-Fi network in the medium which makes the solution less implementable.

A last solution would be to remove Arduino from the system. This would require the handover of operations of Arduino to Raspberry Pi. The resulting system could be considered as a single board solution, that is, all relevant subsystems operate on Raspberry Pi. This option would cause heating problems on Raspberry Pi. A cooling mechanism should also be applicable in this case.

4.3.1.3 Internal Communication Subsystem Tests

This test will enable if communication between Arduino and Raspberry Pi can be realized. Test procedure is as follows:

1. Generate data on Raspberry Pi in a rate that reflects the time consumed of Data Processing subsystem. This will yield a realistic data rate.
2. Send random text data to Arduino.

3. Do the initial integration between Arduino and Raspberry Pi.
4. Send data from Raspberry Pi to Arduino.
5. Increase data speed to the specified data rate.
6. Check the accuracy of the retrieved data.

4.3.1.4 Results of Internal Communication Subsystem Tests

The results of the tests revealed that all steps are successful but the last step. The data send rate is determined to be 25 strings per second. The string length varies between one and three characters. The data is fully received if the rate is slower than 25 strings per second. However, the data loss and improper decoding is observed on the Arduino side. This must corrected by means of coding or switching to an alternative solution.

4.3.2 External Communication Subsystem

This subsystem covers basically the handshake protocol i.e. communication with the other vehicles on the path. The requirements of this subsystem are listed below:

- The subsystem should be able to communicate with the opponent via Wi-Fi protocol

- The subsystem should be able to execute handshake protocol

Communication subsystem enables the robot to communicate with the opponent using the handshake protocol agreed on standard committee. According to the standard committee, Wi-Fi modules must be used to implement handshaking. Since Raspberry Pi was used in the project, there is no need to get a separate Wi-Fi module; the internal Wi-Fi module of the Raspberry Pi was used.

4.3.2.1 Main Solution for External Communication Subsystem

Socket programming is an effective tool to implement client-server communication algorithms. It can be implemented in Python or C++. Our algorithms are written in Python for now, yet it can easily be converted to C++ if the team members decide that it is necessary. The algorithms for client and server sides are slightly different. *Figure 8* shows the functions that are used for client and server sides to create communication between client and server.

Here is the summary of the key functions from socket library:

- socket.socket(): Creates a new socket using the given address family, socket type and protocol number.

- s.bind(address): Binds the socket to the address defined previously.

- s.listen(backlog): Sets up the maximum number of connections that can be made to the socket, which must be at 1 for the project.

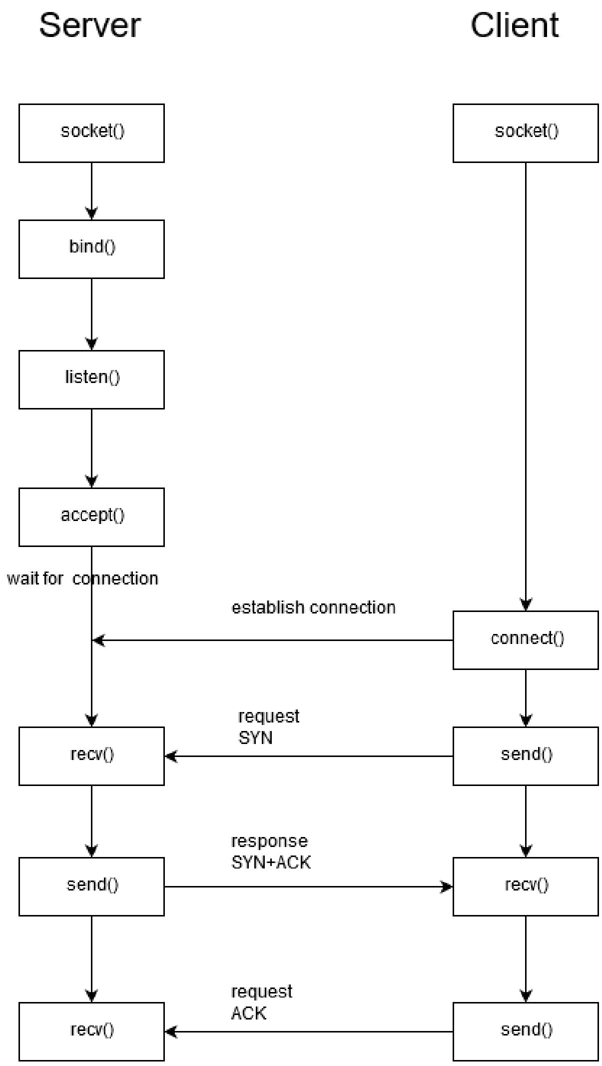


Figure 8: Basic Functions in Python Socket Programming to Implement Handshaking

- s.accept(): Waits until connection arrives, then accept the client connection. Returns the client socket connected to the server as (conn, address) pair, where conn is a new socket object and address is the address bound to this socket
- s.connect(): Provides client to connect to the server
- s.send(): Transmits message to the remote socket.
- s.recv(): Receives message from the remote socket
- socket.close(): closes the socket; i.e., ends the communication with the opponent at the end of the race.

It is stated in the standard committee that each team must be assigned a static IP to communicate with the other robots. Duayenler has the static IP stated as "192.168.1.7" and the ID as “07”. Since Raspberry Pi 3 comes with a built-in wireless adapter, configuring it as a Wi-Fi hotspot is possible. To assign given IP to the robot, Raspberry Pi must be set as an access point from the terminal.

In the algorithm that was implemented for the handshake, in a continuous loop, the front and rear sensors' values are been checked. There are two functions which are for client and server modes, respectively. If the front sensor senses the opponent in 5 cm range, our main code visits the client mode function. If the rear sensor senses the opponent in 5 cm range, server mode function runs. If our robot is in the server mode, the rear sensor value is again checked. The acknowledge message ($< ID >$ 01) or reject message ($< ID >$ 11) is sent according to the sensor value.

4.3.2.2 Alternative Solutions for External Communication Subsystem

Alternative solution for external communication system is implementing handshake protocol using a seperate Wi-Fi module with Arduino.

4.3.2.3 External Communication Subsystem Tests

- Raspberry Pi as Client Test: This test aims to validate whether the Raspberry Pi can successfully fulfill the requirements set by *Handshake protocol* as a client.Test procedure is as follows:

  1. Create a hotspot from the computer
  2. Connect the Raspberry Pi to the hotspot
  3. Modify the client code to be tested according to IP address of the computer
  4. Run the server code from computer
  5. Run the client code from the Raspberry Pi
  6. Try the possible combinations from the terminals of both sides
  7. The test result can be considered as success if both sides respond according to the *Handshake Protocol.*

- Raspberry Pi as Server Test: This test aims to validate whether the Raspberry Pi can successfully fulfill the requirements set by *Handshake protocol* as a client.Test procedure is as follows:

  1. Create a hotspot from Raspberry Pi.
  2. Connect the computer to the hotspot
  3. Modify the client code to be tested according to IP address of the Raspberry Pi.
  4. Run the server code from Raspberry Pi.
  5. Run the client code from the computer.
  6. Try the possible combinations from the terminals of both sides
  7. The test result can be considered as success if both sides respond according to the *Handshake Protocol.*

4.3.2.4 Results of External Communication Subsystem Tests

The first and simplest test has been done on one computer (or raspberry pi) using the same device as client and server, at the same time. To achieve that, the computer's (or raspberry pi's) IP address should be defined in the host section defined in the client mode function. Secondly, the codes were tested on two computers. Thirdly, one raspberry pi and one computer were used for the test. All tests were successful if the server side is connected to the internet and client side is connected to the server via hotspot. The outputs of the tests were given in the *Figure 9* and *Figure 10*.

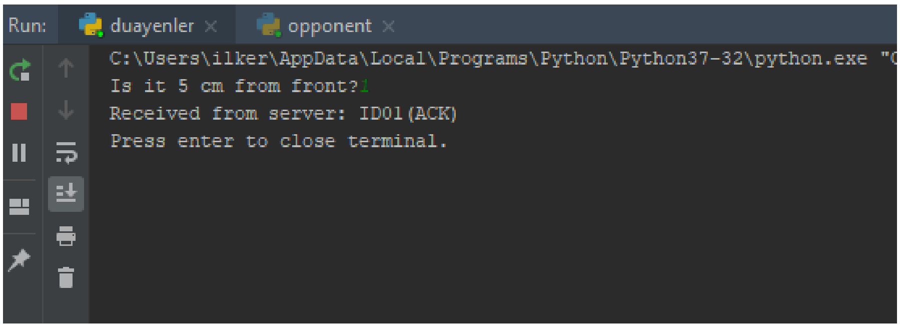


Figure 9: Test Results of Handshaking for Client Side

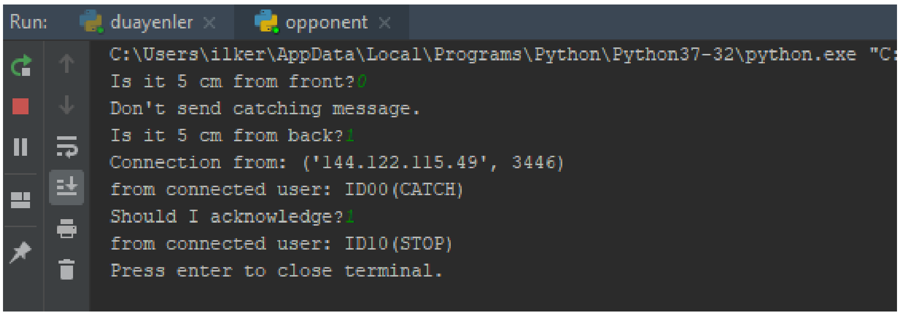


Figure 10: Test Results of Handshaking for Server Side

4.4 Driving System

Driving system has two main subsystems which are Direction subsystem and Speed subsystem. The requirements of this system are listed below.

Driving System Requirements:

- The subsystem should control motion subsystem according to output of the computation system

4.4.1 Direction Subsystem

Direction unit is responsible for the orientation of the vehicle. It stores the last required orientation and the new one coming from the controller. After that, it tries to make the orientation as close as new one. Both data can be represented as vectors.The angle between those two vector is tried to be minimized by the controller. Before moving on to the operation, note that the angle can be used as a measure of the error that the direction unit have. The less the angle the more correctly operates the direction unit. The requirements of this subsystem are listed below:

- The subsystem should drive the motors according to computation system outputs
- The system should ensure that the vehicle follows the lane

4.4.1.1 Main Solution for Direction Subsystem

Depending on the configuration of the wheels, exact control of the vehicle might vary. However, there are certain methods to accomplish orientation. The vehicle will definitely have two wheels or palettes that will be driven by two separate DC motors. That configuration allows differential drive method to orient the vehicle. PWM values of the motors can be adjusted such that the speed difference between them results in a turn as much as desired angle. The exact difference values on the PWM values depends on the specs of the used motors and voltage sources.

Two different H-bridge motor drivers are proposed to be used to drive DC motors: L298N and L293D. Both can drive two motors separately with one IC. However, maximum current rating of the former one is larger being 2A while L293D can supply 0.6A per channel.

4.4.1.2 Alternative Solutions for Direction Subsystem

As in the case of another configuration that involves one or two servo motors to control the directions of the front wheels. This configuration is more robust compared to ball caster utilization. However, there are more motors to control and it requires more complicated differential drive algorithms involving both DC motor differential and servo PWM to orient the front wheels.

4.4.1.3 Direction Subsystem Tests

- Straight Drive Test: This test aims to find the PWM bias between the two motors. The test procedure is as follows:
  1. Make the necessary connections between motors, motor controller and the Arduino
  2. Set the PWM values of the motors equal
  3. Observe the behaviour of the motors
  4. Increase the PWM value of the slower motor until a point the vehicle can go in a straight line.
  5. Record this PWM difference to use in PID controller subsystem

- Circular Drive Test:

    1. Make the necessary connections between motors, motor controller and the Arduino
    2. Desired curvature is decided
    3. According to motion of the vehicle PWMs of the motors are set
    4. PID parameters are set according to this test

4.4.1.4 Results of Direction Subsystem Tests

The test were conducted for the motor pairs used in the Critical Module Demo. The test can be repeated for new motor pairs if needed.

4.4.2 Speed Subsystem

This unit acts as a complementary module for direction unit. It will act as a state machine. In one state, the unit will try to increase the speed of the vehicle by making overall increase in both PWM values of DC motors. The feedback of this system will be the cost function mentioned in driving unit. If that cost exceeds a specified level, unit goes to another state in which the unit will decrease the overall speed to allow direction unit to operate more correctly. In short, this unit tries to compensate the error of the direction unit by changing the overall speed of the vehicle. The requirements of this subsystem are listed below:

- The subsystem should decrease the vehicle speed at the narrow lane

- The subsystem should increase the vehicle speed at the wide lane

- The subsystem should decrease the vehicle speed at the extreme disturbance

4.4.2.1 Main Solution for Speed Subsystem

Currently, there is no implementation for the subsystem because its operation depends on the control feedback loop and the success of the direction unit. Once they are operating and the vehicle is capable of following the lane at, least partially, then the speed subsystem will be implemented on top of the control algorithm.

The best place to implement the state machine is Arduino as motor driving is also controlled by it. Main requirement for this system to operate is a measure of error or success for the direction subsystem. It can be the same as the error input of controller i.e. turning angle or a function of it. The state machine will act depending on the value of the error input. When it is above a critical level, the vehicle will show a steep deceleration to compensate the error of the direction unit. In other cases, It is wise to implement the speed controller in the form of at least PD controller. In other words, the change in the overall speed will also be maintained by a controller whose error input is not necessarily tried to be made zero but rather below a pre-specified level. State machine diagram can be seen in *Figure 11*

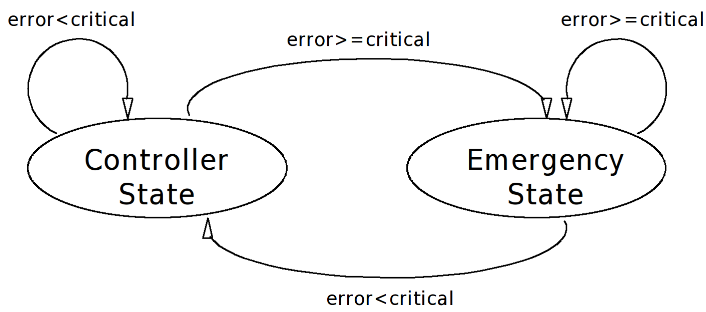


Figure 11: State diagram of speed subsystem

4.4.2.2 Alternative Solutions for Speed Subsystem

There are many ways that the state machine can be implemented. Depending on the test results, the state diagram of the machine might me modified by adding new states. Furthermore, the controller type can be changed or completely removed and replaced by some discrete increase or decease levels to control the overall speed. The changes will be applied depending on the test results.

4.4.2.3 Speed Subsystem Tests

- Determination of the error input: This test aims to find the correct error input for overall speed controllers. The test procedure is as follows:

  1. Make all the necessary connection
  2. Start up the vehicle
  3. Execute lane detection and controller algorithms
  4. Set the error input of the both controller algorithms the same
  5. Observe the behavior
  6. Repeat the same process with a linear function of the input
  7. Observe the success of the tracking algorithm

  By checking in which case the vehicle is able to follow the path with less error, the error input can be decided.

- Determination of the critical error value: This test aims to specify above which error value, the emergency state should be triggered.

  1. Make all the necessary connection
  2. Start up the vehicle
  3. Execute lane detection and controller algorithms
  4. While the vehicle is moving, give disturbance of different types
  5. Record the maximum value of the error encountered during the disturbances.
  6. Find the maximum value

4.4.2.4 Results of Speed Subsystem Tests

No test result is currently available due to mentioned reasons.

4.5 Motion System

Motion system has two main subsystems which are Wheels subsystem and Motors subsystem. The requirements of this system are listed below:

- The system should ensure that the vehicle can drive itself with enough power

4.5.1 Wheels Subsystem

There are possible solution for wheel placement on the chassis, and several wheel types. Some wheels are designed for better gripping on different surfaces. To avoids obstacles on the path, gripping of the wheel is an important concept. Some wheel types are ball caster, toy car wheel and palette. Besides, wheel placement and the wheel number should be combined with the wheel type choice. The requirements of this subsystem are listed below:

- The subsystem should ensure that the wheels can grip lane without slipping in all conditions

4.5.1.1 Main Solution for Wheels Subsystem

The wheel placement combination that will be utilized is 2+1 combination. This combination can be assembled by placing 2 car wheels (with motors) to the back and the one boll caster to the front or vice versa. These configurations provide easy implementation and fairly reliable handling on the path. However, for certain obstacles may significantly disturb vehicles balance in this configuration.

4.5.1.2 Alternative Solutions for Wheels Subsystem

- Palettes: Another combination is palette system. This system is used in real world where robust vehicles are needed. Similarly, this configuration can help to handle obstacle in the path, but it costs for harder implementation and driving.

- 2+2 Wheels Last implementation is 2+2 configuration. In this configuration 2 wheels can be placed at the back and the rest at the front by placing motors to back wheels. To ease turning of the vehicle, front wheels can be controlled with a servo motor as back wheels operate in the differential drive mode. This combination may provide both enhanced grip and reliable operation.

4.5.2 Motor Subsystem

Motors are the one of the important physical components of the project. They represent the whole system's actions. As stated in the project definition, the path should be completed within 20 seconds and total length of the path is approximately 3 meters. Using this specifications, a simple calculation for RPM requirement yields

$$\begin{aligned} v = \frac{3}{20} &= \pi \times D \times \frac{RPM}{60} \\ RPM &= \frac{2.865}{D} \end{aligned} \tag{1}$$

where D is the diameter of the wheels in meter. Currently used wheels has 6.5 cm diameter, which implies that RPM=44. This means that under load average RPM needs to exceed this value. To estimate torque value, it is necessary to predict components weight. When weight of the current components are measured, result is around 1kg, so rated torque of the each motor expected to 3 times larger than this value to obtain well acceleration in loaded condition.

The requirements of this subsystem are the followings:

- The subsystem should ensure that the motors can supply enough torque to accelerate the vehicle
- The subsystem should ensure that the motors can execute driving system outputs without deviation

4.5.2.1 Main Solution for Motors Subsystem

One of the widely used motor type is brushed gearhead DC motors. They are cheap and require enough torque for this competition. They have mainly two section one is the normal DC motor and gear head. Gear is a must to enhance torque performance of the DC motor. In other words, gear balance rpm and torque of the motor.

4.5.2.2 Alternative Solutions for Motors Subsystem

Another option is servo motors. Servo motors are high-torque motors that can turn in a desired angle. Servos can be utilized in the direction of the vehicle on the front wheels in the four-wheel solution. By using this solution, turning radius can be decrease significantly.

4.5.2.3 Motors Subsystem Tests

This unit's test is focusing on two crucial parameters of the motor which are RPM and torque values of the motor. Basic pulley system is used to calculate torque value of the motor. Distance is measured under the condition that at rated voltage and approximated load, to calculate loaded RPM value calculation.

4.5.2.4 Results of Motors Subsystem Tests

The tests are done. The results are negative because motors torque value did not match with the declaration of the supplier. 3kg-cm is the decelerated value, but motors can only produce 750 g-cm. Therefore, this system is failed in torque requirement, and reconsidered in the following period.

- Torque Test:

  1. Fix the motor at horizontal position with respect to ground
  2. Attach an object of one kilogram
  3. Power up the motor
  4. Increase the weight to a point where the motor is not pulling anymore
  5. Record the value and check with expected results
  6. If the result is not comparable with expected values and very low, motor can be considered as broken
  7. Contact the seller for more information

RPM test has not been done since torque value is not supporting test setup.

4.6 Structure System

Structure system has two main subsystems which are Chassis subsystem and Printed Circuit Board subsystem. The requirements of this system are listed below:

- The system should ensure that structure is robust for external effects
- The system should ensure that structure is balanced
- The system should ensure that vehicle has a good appearance

4.6.1 Chassis Subsystem

Main purposes of this subsystem are protection of the critical elements of the robot and holding components together. The most important part of this section is weight distribution. The chassis is supposed to be light and strong because of the competition purposes. However, it should balance the robot to be able to handle turns. The requirements of this subsystem are listed below:

- The subsystem should ensure that the chassis is rigid
- The subsystem should ensure that the chassis have enough space for components
- The subsystem should ensure that the chassis can provide low center of mass

4.6.1.1 Main Solution for Chassis Subsystem

Current chasis structure relies on two pre-designed plexiglass layers. Raspberry Pi and Arduino is placed on the upper layer while motor driver and the battery are on lower one. To keep the center of mass of the vehicle close to the ground, battery is placed as low as possible. The connection of the motor driver and Arduino consists of eight cables two of which are the power lines. The cables are placed in a way that they cause no entanglement with any other parts. The connection between RPi and Arduino is currently accomplished by USB cable.

Since there is not much component on the vehicle, the space on the layers are enough to locate the components. However, placing the camera of RPi has been a great problem. The view angle of the camera turned out to be considerable small than expected. Other several cellphone cameras were tried but they are could not satisfy the requirement that both side of the lane should be visible either. The only solution was to elevate the camera. That is why a camera holder structure is designed and added to the system.

The requirement for the camera holder can be listed as

- It should be integrated to th front of the vehicle
- It should be as rigid as possible to reduce the vibration on the camera
- It should be light weight so that does not effect the center of mass considerably
- It should be adjustable in terms both elevation and camera angle

To satisfy the requirements the holder is built using 4mm plexiglass. The choice satisfies the rigidity and light weight possible. A thinner one would result in less rigidity and increased vibration on the system. The designed structure, whose layout can be seen in *Figure 12*, has the elevation range from 35 cm to 45 cm and a camera angle ranging from 0^o to 45^o. Having manufactured, the camera holder is integrated to the vehicle (*see Figure 13*). After integration, the view of the camera can completely cover the both edges of the path (*see Figure 5*)

Current version of chassis which were used in critical module demo can be seen at *Figure 13*.

4.6.1.2 Alternative Solutions for Chassis Subsystem

There are many improvements to be done on the chassis. When all the components such as vehicle detecting sensors are added, the space will not be enough. The best alternative is to make a custom design chassis structure. By that way, many problems can be resolved. By the time each systems are tested and validated and decided to be used as the actual solution, required space will have been clearly defined. After that point the best placement can be configured and chassis will be designed accordingly to the requirements of that time. Moreover, while designing the chassis, the possibility of implementation of an alternative solution will be considered as it affects the overall layout of the vehicle.

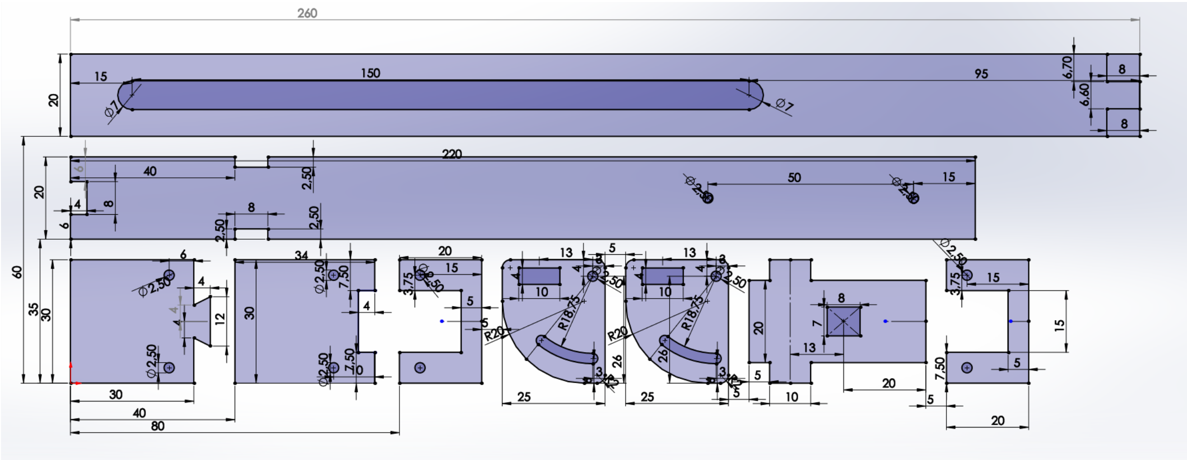


Figure 12: Camera holder design layout

4.6.1.3 Chassis Subsystem Tests

- Inertia test: Aims to deduce a quantifiable measure of stabilization. Test procedure is as follows:

    1. Prepare a straight path
    2. Power up the vehicle
    3. Execute the edge detection and control algorithm
    4. Give different type of disturbances
    5. Observe the deviation from straight line
    6. Repeat the process with different component configurations

Currently no test results available due to lack of control algorithm.

4.6.2 Printed Circuit Board Subsystem

The main role of this part is decreasing connection mess and increase vibration strength of the robot against disturbances. Also, this section increases rigidity of the whole system. The requirements of this subsystem are listed below:

- The subsystem should ensure that all the electronic components are placed on PCB

- The subsystem should ensure that all the connections are firmly secured and robust to vibrations.

This subsystem aims to make all the circuit connections rigid and compact. Currently, there is wire connections between Arduino-Motor driver and Arduino-RPi. However, addition of vehicle detection sensors and other lane detection alternatives will increase the amount of components, hence, wires. In addition, to use the space occupied

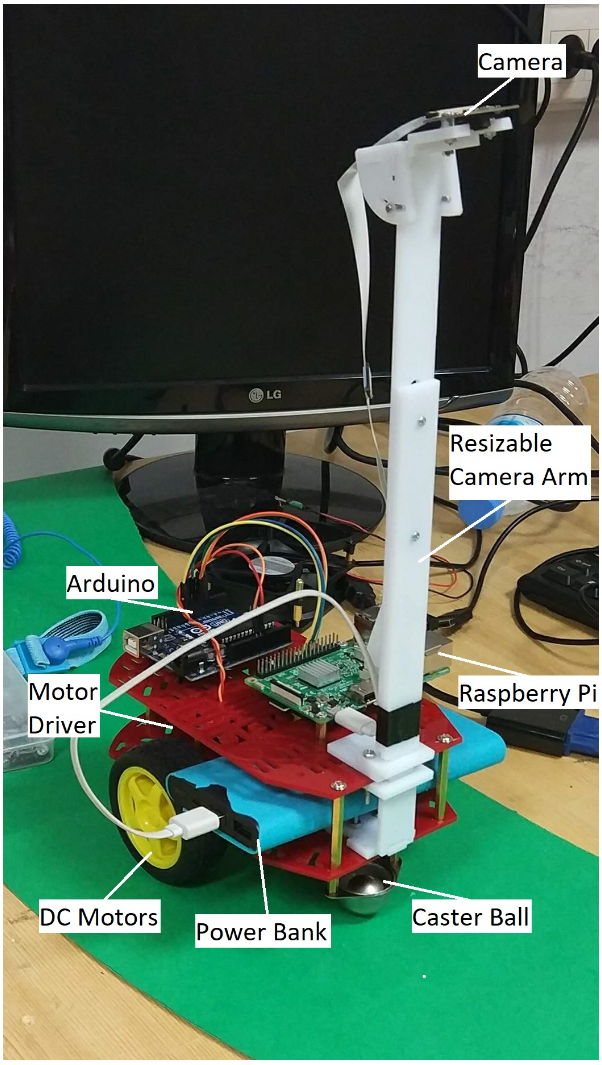


Figure 13: Current version of the chassis

by the Arduino UNO board, Arduino Mini can be used. This also allows to build the circuit board as shield for Arduino Mini. After that any other sensors and connections can be made through PCB. In other words, PCB acts as a breakout board for each item integrated to the system in a more rigid and compact way.

4.6.2.1 Printed Circuit Board Subsystem Tests

Once completed, the board should be tested if it is built properly and correctly.

- Short test: Aims to check all the wanted connections are present. The test procedure is as follows:
  1. Open multimeter for short circuit test
  2. Find the ends of each routing
  3. Check the continuity using multimeter probes

4. Check if there is any unwanted short circuit
5. If exist, eliminate

5 Plans

Each team member is assigned to a subsystem according to their interest and qualification. *Figure 14* summarizes the assignments. Besides, a Gannt Chart is prepared to have an detailed overview of future works and available in *Appendix A*. Future plans

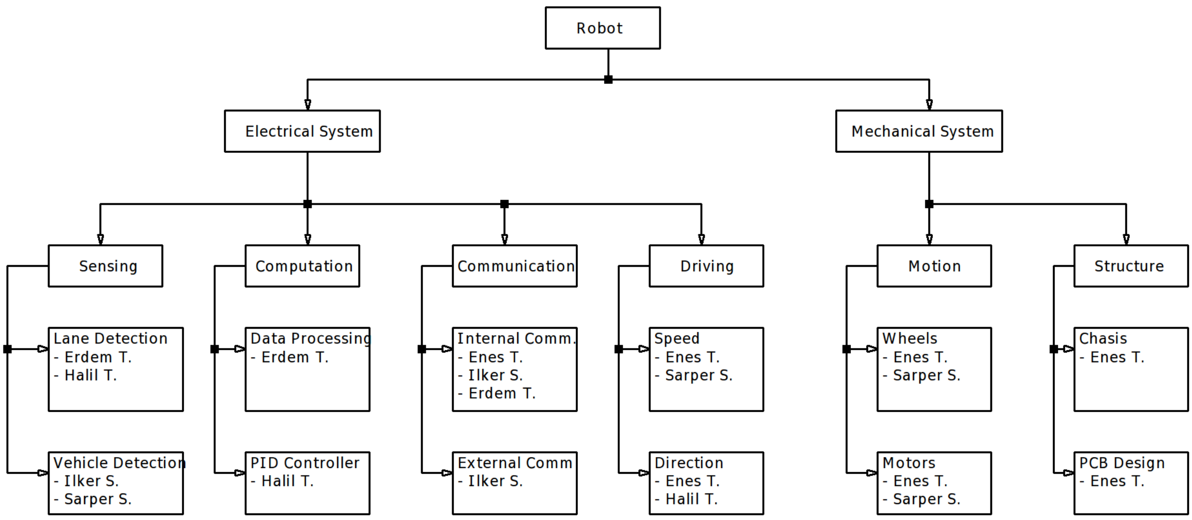


Figure 14: Work assignment is subsystem level

regarding the improvement of the project are listed here in system level.

5.1 Sensing System Plans

Lane detection algorithm is reconstructed and develop by Erdem. The main enhancement is focusing on robustness of the algorithm. The algorithm currently detects the boundaries of the placed obstacles on the path. This might be fixed in Data Processing subsystem. But, alternative solutions that are introduced in *Section 4.1.1.2* might be applied, compared and assessed in terms of robustness and accuracy.

Supportive solutions to lane detection which are color sensor and laser sensor array are going to be studied. More specifically, they are constructed and tested by Sarper during semester holiday. After the results, compatibility of these solutions are going to be discussed.

Sensors for vehicle detection will be ordered and tested before the submission of critical design review report and the most promising one will be selected considering the requirements.

5.2 Computation System Plans

The Data Processing subsystem has some weak points. For instance, it is not able to predict turn angle and direction in the case of missing lane lines. This will be worked out by Erdem. The plan is that the system will be able to predict urn angle and direction even if one lane is detected on the path.

Another point that data processing subsystem has weakness is if an obstacle is present on the path, the boundaries of the obstacle may constitute a line. This is regarded as the part of lane line in the current processing algorithm. This flaw must be fixed by Erdem, since this implementation leads to a wrong prediction on turn angle and direction.

Regarding the PID Controller subsystem plans, the current controller is designed z domain. This applies good in theory but a better approach will be implemented by Halil. This approach is designing the controller in $\overline{s}$ domain by using bilateral Tustin transform. This is continuous approximation of z domain, which may yield more accurate results in the control of the car.

5.3 Communication System Plans

According to the test results, server side must be connected to the internet, and client side must be connected to the server via hotspot to establish communication. However, it is not the desired result; since both sides must be connected to each other in the project. The main plan regarding this subsystem is to solve this problem. Also, the input of the external communication subsystem must be the data coming from the sensors of the vehicle detection subsystem. Since the vehicle detection subsystem was not set up on the robot yet; the input of the external communication subsystem was not the sensors. In the upcoming weeks, the handshaking tests taking the data from vehicle detection subsystem will be conducted.

5.4 Driving System Plans

The direction subsystem will be enhanced by the Enes and Halil following the successful communication between Raspberry Pi and Arduino and initial version of PID controller subsystem is successfully combined with data processing subsystem output.

The speed subsystem will be enhanced following the early outputs of driving subsystem. Different solution approaches will be tried by Sarper and Enes to find perfect speed conditions to finish the path within desired time.

5.5 Structure System Plans

A new chassis is going to be designed considering narrow turn section of the lane and external obstacles. Rigidity and compactness are the essential requirement for the chassis. To achieve these requirements, Enes and Sarper are going to design a chassis. In addition, the front and back recognition plates agreed on the standard committee will be installed.

After all the components are tested and validated to be used in the project, a proper PCB will be designed by Enes. It will provide compactness and robustness to the whole electrical system by significantly reducing excess cabling and possibility of loose connections.

5.6 Estimated Cost of Project

Estimated cost anaysis for the project can be investigated at *Table 3*. The reproducible vehicle is expected to cost under 200 dollar as desired by the project requirements.

Table 3: Estimated Cost Analysis for the Project

Component	Price (in Dollar)
Raspberry Pi	48
Camera	23
Chassis	20
Arduino	5
Motors	22
Wheels	8
Motor Driver	2.5
Powerbank	12
Li-po Battery	15
Distance Sensors	18
Additional Components	10
Additional Payments	15
Total Project	198.5

6 Conclusion

This report consists of a detailed project description and explanation about autonomous vehicle on a predefined path. The solution approaches, alternatives and tests comply with the proof-of-concept idea and prototyping purposes. The further improvements and developments will be on manufacturing level.

In particular, a general overview of the systems are given in V-Model together with the system block diagram. Then, subsystems are presented in a detailed manner by explaining current solution approach, used test steps, obtained test results and possible solution alternatives. The whole system has not been merged into a single body yet. The systems are individually functioning with the limitations. One limitation is caused by Data Processing subsystem, that is, the vehicle must be able to see both lane lines. Another limitation is against obstacles on the path. If the obstacles have long enough edges, they are regarded as the part of lane lines and they contribute to lane line drawing. Such limitations will be worked out extensively to develop a more robust vehicle. Besides, some solutions have worked out very nicely such as external communication with other vehicles, adjustable camera arm, wheels and lane detection algorithm. On the other hand, the budget of the project will be kept below the budget limit and the total work will be reproducible with less than \$ 200. The company believes that the reliability of a business depends on staying within company budget.

DUAYENLER will keep their diligent work to have a more robust and faster vehicle at the end of project. The company members believe that the final work will bring an innovative approach to the design and development of autonomous vehicles with its features and the way that features are implemented.

7 Disclaimer

All information and content contained in this report are provided solely for proof-of-concept. DUAYENLER Ltd. Şti. guarantees that the report and information contained obeys the restrictions and rules ordered by the Standard Commitee.

Halil TEMURTAŞ   Erdem TUNA   Enes TAŞTAN

Sarper SERTEL   İlker SAĞLIK

27 December 2018

A Gantt Chart

		October 1-5	October 8-12	October 15-19	October 22-26	Oct 29 - Nov 02	November 05-09	November 12-16	November 19-23	November 26-30	December 03-07	December 10-14	December 17-21	December 24-28	Dec 31 - Jan 04	January 7-11	January 14-18	January 21-25	January 28-31	February 4-7	February 11-14	February 18-21	February 25-28	March 4-7	March 11-14	March 18-21	March 25-28	April 1-4	April 8-11	April 15-18	April 22-25	Apr 22 - May 05	May 09-12	May 12-15
	T0+	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
	1. Semester																																	
	Holiday																																	
	2. Semester																																	
1	**Concept Development Phase (3 Weeks)**																																	
2	**Assessment of System Requirement Phase (3 Week)**																																	
3	**Conceptual Design and Preliminary Design Phase (4 Weeks)**																																	
4	**Critical Design Phase (17 Weeks)**																																	
5	**Test & Evaluation Phase (20 Weeks)**																																	
6	**Implementation & Finalization Phase (8 Weeks)**																																	
7	**Project Ending (T0+33 Weeks)**																																	
1	**Concept Development Phase**																																	
1.1	**Activities**																																	
1.1.1	Literature Research and Determination and Similar Platform Specifications																																	
1.1.2	Feasibility Works																																	
2	**Assessment of System Requirement Phase**																																	
2.1	**Activities**																																	
2.1.1	Determination of Team Logo and Vision & Mission																																	
2.1.2	Problem Define State for All Projects																																	
2.1.3	Solve Defined Problem State for All Projects																																	
2.1.4	General Component Research																																	
2.2	**Outcomes**																																	
2.2.1	Business Statement Report																																	
3	**Conceptual Design and Preliminary Design Phase**																																	
3.1	**Activities**																																	
3.1.1	Preliminary Electrical System Design																																	
3.1.1.1	Preliminary Sensing Unit Design																																	
3.1.1.2	Preliminary Computational Unit Design																																	
3.1.1.3	Preliminary Driving Unit Design																																	
3.1.2	Preliminary Mechanical System Design																																	
3.1.2.1	Preliminary Motion Unit Design																																	
3.1.2.2	Preliminary Structure Design																																	
3.2	**Outcomes**																																	
3.2.1	Preliminary Report																																	

		October 1-5	October 8-12	October 15-19	October 22-26	Oct 29 - Nov 02	November 05-09	November 12-16	November 19-23	November 26-30	December 03-07	December 10-14	December 17-21	December 24-28	Dec 31 - Jan 04	January 7-11	January 14-18	January 21-25	January 28-31	February 4-7	February 11-14	February 18-21	February 25-28	March 4-7	March 11-14	March 18-21	March 25-28	April 1-4	April 8-11	April 15-18	April 22-25	Apr 22 - May 05	May 09-12	May 12-15
	T0+	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
4	**Critical Design Phase**																																	
4.1	**Subsystem Design Phase**																																	
4.1.1	Sensing System Desing																																	
4.1.1.1	Lane Detection Subsystem Design																																	
4.1.1.2	Vehicle Detection Subsystem Design																																	
4.1.2	Computation System Desing																																	
4.1.2.1	Data Proccesing Subsystem Design																																	
4.1.2.2	PID Controller Subsystem Design																																	
4.1.3	Communication System Design																																	
4.1.3.1	Internal Communication Subsystem Design																																	
4.1.3.2	External Communication Subsystem Design																																	
4.1.4	Driving System Design																																	
4.1.4.1	Direction Subsystem Design																																	
4.1.4.2	Speed Subsystem Design																																	
4.1.5	Structure System Design																																	
4.1.5.1	Chassis Subsystem Design																																	
4.1.5.2	PCB Subsystem Design																																	
4.1.6	Motion System Design																																	
4.1.6.1	Wheels Subsystem Design																																	
4.1.6.2	Motors Subsystem Design																																	
4.2	**Critical Design Outputs**																																	
4.2.1	Standards Report																																	
4.2.2	Module Test Demo																																	
4.2.3	Conceptual Design Report																																	
4.2.4	Presentations																																	
4.4.1	Critical Design Review Report																																	

		October 1-5	October 8-12	October 15-19	October 22-26	Oct 29 - Nov 02	November 05-09	November 12-16	November 19-23	November 26-30	December 03-07	December 10-14	December 17-21	December 24-28	Dec 31 - Jan 04	January 7-11	January 14-18	January 21-25	January 28-31	February 4-7	February 11-14	February 18-21	February 25-28	March 4-7	March 11-14	March 18-21	March 25-28	April 1-4	April 8-11	April 15-18	April 22-25	Apr 22 - May 05	May 09-12	May 12-15
5	**Test & Evalution Phase**																																	
4.1	**Subsystem Test Phase**																																	
4.1.1	Sensing System Testing																																	
4.1.1.1	Lane Detection Subsystem Testing																																	
4.1.1.2	Vehicle Detection Subsystem Testing																																	
4.1.2	Computation System Testing																																	
4.1.2.1	Data Proccesing Subsystem Testing																																	
4.1.2.2	PID Controller Subsystem Testing																																	
4.1.3	Communication System Testing																																	
4.1.3.1	Internal Communication Subsystem Testing																																	
4.1.3.2	External Communication Subsystem Testing																																	
4.1.4	Driving System Testing																																	
4.1.4.1	Direction Subsystem Testing																																	
4.1.4.2	Speed Subsystem Design																																	
4.1.5	Structure System Testing																																	
4.1.5.1	Chassis Subsystem Testing																																	
4.1.5.2	PCB Subsystem Testing																																	
4.1.6	Motion System Testing																																	
4.1.6.1	Wheels Subsystem Testing																																	
4.1.6.2	Motors Subsystem Testing																																	
6	**Finalization Phase**																																	
6.1	**Activities**																																	
6.1.1	To be detailed																																	
6.2	**Outcomes**																																	
6.2.1	Finalized Product																																	
6.2.2	Final Report																																	
6.2.3	Final Demo																																	
7	**Project Ending**																																	