The University of Manchester

Department of Electrical and Electronic Engineering

Embedded Systems Project 2020-21

DESIGN REPORT #2

Title: Technical Characterisation

Group Number: 39

Group members:	ID Number	I confirm that this is the group's own work.
Mohammad Arab	10446769	☒
Ionut Buciuman	10455641	☒
Ahmad Huneiti	10528746	☒
Danial Murshed	10245339	☒
Yuwen Peng	10789777	☒
Liangchao WU	10573442	☒

Tutor: Wuqiang Yang

Date: 10/12/2020

Contents

1. Introduction 1
2. Software 1
3. Line sensor characterisation 5
4. Circuit diagram for proposed line sensors 10
5. Non-line sensors 12
6. Control 13
7. Hardware overview 16
8. Summary 19
9. References 20

1. Introduction

The aim of this report is to develop an understanding of the systems which will control the movement of the line following buggy. The objectives are to identify the proposed sensors by thorough experimentation and analysis of the results, to suggest and to specify the software requirements for the STM32 microcontroller, to research other sensors and control systems, and develop the requisite PCB and chassis designs which will facilitate the achievement of a working solution.

There are a few outcomes from this report in order for it to be considered successful. These will include the following recommendations for the final design of the buggy:

- Choice of the most optimal sensors and sensor array design
- A Prototype of the logic behind the code with a deep understanding of the implemented control system
- A detailed PCB plan and full consideration of the chassis design and manufacture

Achieving these outcomes will allow the group to achieve the aim of the report.

The Software and the Control section are strongly interlinked as the various instructions needed will get the logic based on the control system used for a series of actions. Looking at the project from a physical point of view, Line sensor characterization, Circuit diagrams and Hardware are considered to have an interdependent relationship all under the design aspect. These are only a few of the links between the sections in this report. More connections can be observed in the report itself and due to this co-dependency between the sections, it is crucial that effective communication is adopted throughout the whole process in order to achieve a coherent final piece of work. By understanding how the line sensors and control algorithm works a suitable software architecture was developed to implement the buggy's functions. Different line sensors that will allow the buggy to follow the white line were tested before choosing the most suitable sensor based on the system's requirements. Other non-line sensors were also used to measure some important parameters of the buggy. In order to control speed and direction of the buggy, a control algorithm has to be implemented. In this report, two control algorithms are compared and the most suitable one is chosen. Finally, the chassis needs to be designed in a way to allow the buggy to move through the track and overcome any obstacles.

2. Software

The buggy is placed on a black track which has a white line in the middle in the beginning. When the buggy is turned on, 6 front line sensors begin to detect the white line and start to move along it. There are several requirements including line brake, slope and steps in the track for the buggy to overcome automatically. When it meets a bend it will change direction by changing two wheels' speed to keep following the white line. For the ramp, by detecting the changes in the angle of the track, speed will be controlled to keep a constant speed/torque to go over the whole slope. Finally, when the buggy meets a return point which also the end of the white line. The buggy will stop and can receive a command to do a U-turn to make itself face back and start to detect the white line again and go back along the track to the

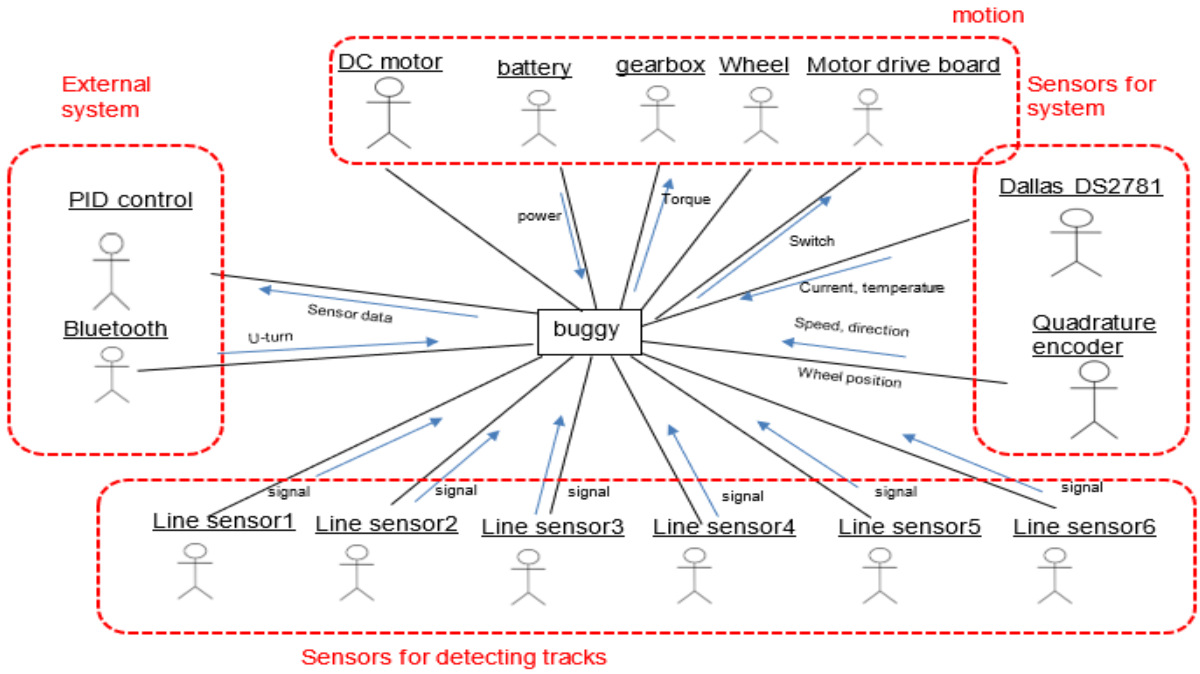
start point.

2.1 Software Constraints

The sample delay present during the implementation has to be considered as a limiting factor. It is important to ensure that this sampling speed is slower than the processing speed and since the Nucleo-F401RE has an 84 MHz clock rate then it is very unlikely for it not to be able to handle such a program. The maximum expected speed of the buggy is 0.958 m/s and the minimum bend radius is 50 mm which means the buggy has almost 0.052 s to change direction. This result in a minimum sampling rate of approximately 18 Hz. This means the Nucleo is capable of processing data according to the buggy's requirements.

For the buggy to function properly 2 PWM, 2 wheel position, 6 LED signals, 6 sensor outputs, and 2 wheel speed variables will need to be initialised in the program. Assuming all variables are initialised as integers which in reality is not the case then 72 bytes of RAM is needed. Of course, this is not an accurate representation and many more variables may be used. The Nucleo-F401RE contains 96 kilobytes of RAM which makes it unlikely that the program will require more space.

2.2 Context Diagram

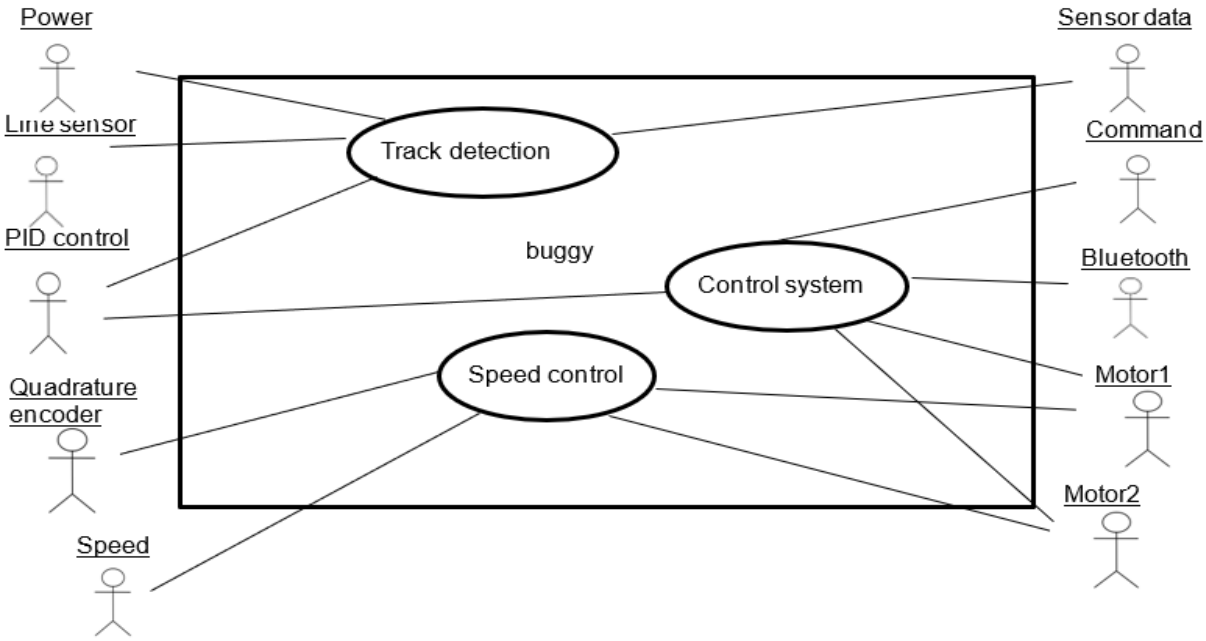


2.3 Table Of Messages

Name	Description	Source	Destination	Arrival Pattern	Interface
Direction	**Forward or backward movement**	**Quadrature encoder**	**System**	**Aperiodic**	**GPIO pins**
Position	**Position of the wheel**	**Quadrature encoder**	**system**	**Aperiodic**	**GPIO pins**
LED signal	**Switching LED on and off**	**System**	**Line-sensor board**	**Periodic**	**GPIO pins with ADC**

U turn	**Makes the buggy turn around**	**Bluetooth**	**System**	**Aperiodic**	**UART**
Voltage	**Measures the battery voltage**	**Dallas DS2781**	**System**	**Aperiodic**	**-**
Sensors output	**Returns data of sensors output**	**Line-sensor board**	**System**	**Continuous**	**GPIO pins with ADC**
PWM	**Pulses sent to motor drive to move the buggy**	**System**	**Motor drive board**	**Periodic**	**GPIO pins with PWM**

2.5 Case Diagram



2.6 Case descriptions

Track detection

Main:

Line sensors detect the position of the white line

Give signals to the control system

Control system gives power to the motor

Alternative flow 1:

If the position of the white line is not desired

Give signals to the control system

Control system gives differential power to the motor to adjust the position over the line

Alternative flow 2:

If the line sensor can't detect the position of the white line

The buggy will stop

Speed control

Main:

Quadrature encoder detects the speed of the buggy

Compared it with the desired speed

Alternative flow 1:

If the speed higher than the desired speed

The motor speed will decrease.

Alternative flow 2:

If the speed lower than the desired speed

The motor speed will increase.

Control system

Main:

The buggy will go along the white line automatically via PID control

Alternative flow 1:

If the buggy receives a command from the Bluetooth

The buggy will do the command

2.7 Object specifications

Track detection

```
Def trackdetection():
While (middle sensor = white line detected)
        Left motor output= right motor output
If (other sensor = white line detected)
        Give signal to the control algorithm
            Change the motor output
If (all sensors = 0)
        Motor output = 0
```

Speed control

```
If (speed > desired speed)
    decrease the motor speed
If (speed < desired speed)
    Increase the motor speed
```

Control system

```
Def controlsystem():
While (Bluetooth command = 0)
        Send sensor data to the control algorithm
If (Bluetooth command = 1)
         Do the command
```

3. Line sensor characterisation

3.1 Experimental method:

To start the experiment, a TCRT5000 sensor was soldered on a small PCB. Four pins were needed to be mapped on an 8-pin DIL socket correctly. The way TCRT5000 is orientated is crucial. The square on the PCB board indicates pin 1. Pins 2 and 7 are used for the emitter. Pins 3 and 6 are used for a phototransistor [2]. The circuit was built using a stripboard, which was cut using a track cutter in order to avoid unwanted short circuits. Moreover, multi-stranded wires were connected to the stripboard where the TCRT5000 sensor was placed on the board and connected to the myDAQ [2]. The connection was done using the wiring diagram with the same wire colours used in the diagram. The 5 V and 0 V were connected to the circuit from the myDAQ board. The myDAQ analogue positive input was connected to the sensors' outputs, while the analogue negative output was connected to 0 V [2]. The circuit was assembled and measurements were recorded. The effect of different types of lights and the vertical and horizontal placement of the sensor were recorded. The horizontal measurements were taken using a test rig while the vertical measurements were taken using a ruler.

3.2 Graphs Analysis

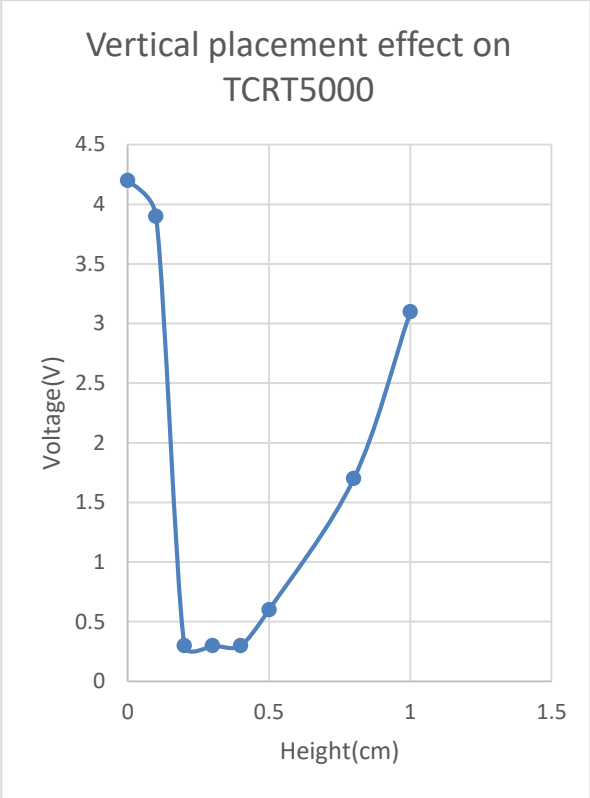


Figure 1: Height experiment for the sensor

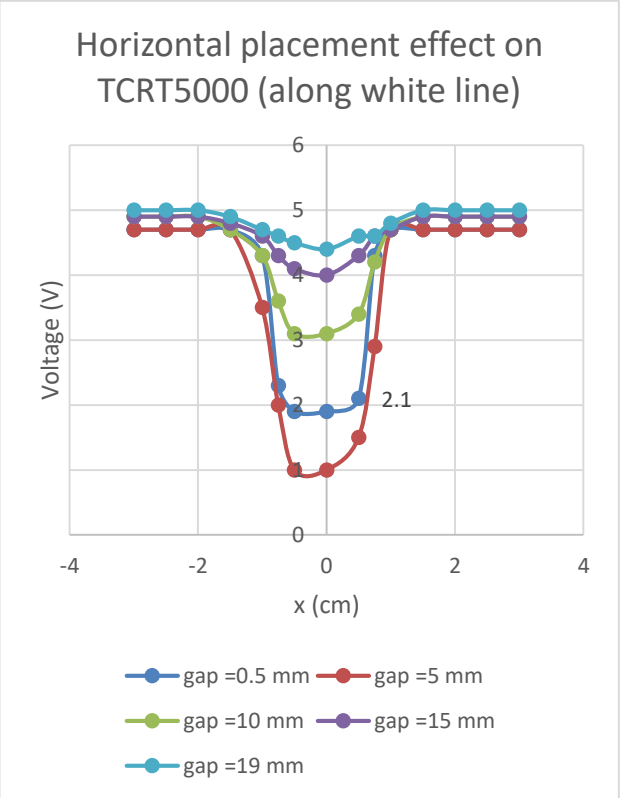


Figure 2: Horizontal effect on TCRT5000 (along white line)

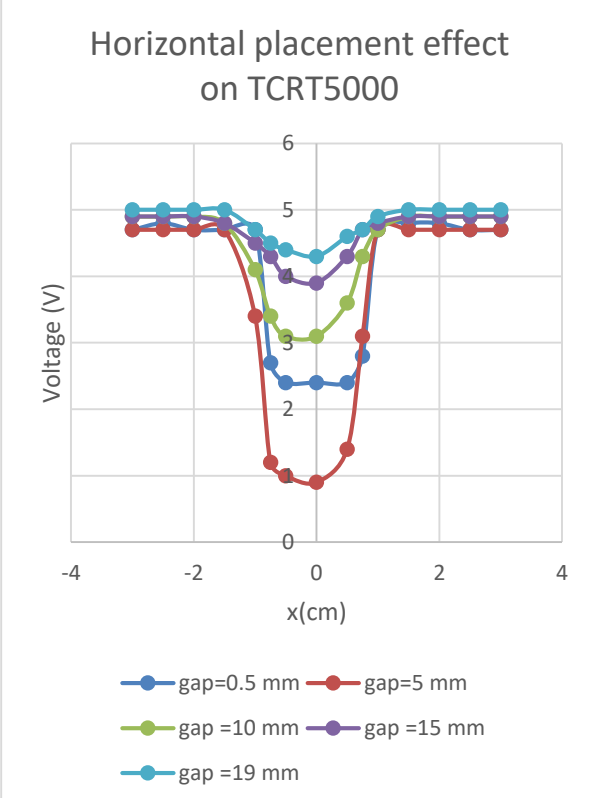


Figure 3: Horizontal effect on TCRT5000 (across the width of the white line)

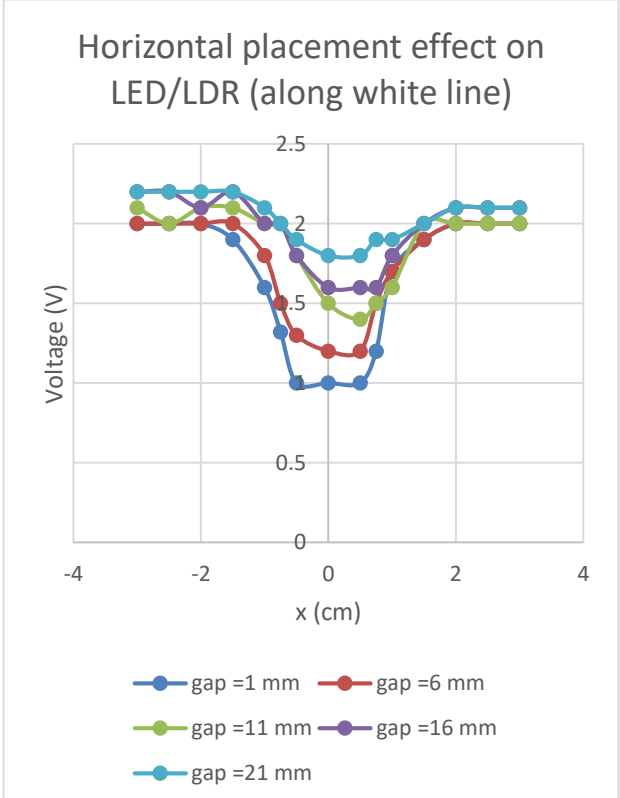


Figure 4: Horizontal effect on LED/LDR (along white line)

The obtained graphs from the different placements (vertical and horizontal) of the sensor give a clear idea about the sensor and LED/LDR sensitivity regarding their placement on the buggy. It can be assumed from **Figure 1** that the TCRT5000 is most sensitive when placed exactly above the white line within a few millimetres from it. Regarding the horizontal placement of the sensor, in **Figure 2** when the measurements are taken along the white line, it was observed that the bigger the gap is the more sensitive the sensor is. Furthermore, when the x-axis is moving towards zero the sensitivity decreases and start to increase when it moves to the positive side. In **Figure 3**, the measurements were taken when the sensor is placed across the width of the white line. It was noticed that the results are approximately the same even with the decreased sensitivity from the negative side to the increase in the positive side. For the last LED/LDR, results shown in **Figure 4** indicate that the LED/LDR will have high sensitivity even when the gap increases. As it can be observed that no big change occurs in terms of the sensitivity at zero except at 1 mm and 6 mm where the value of voltage is small in comparison with other gaps. When the x-axis is moved from the negative side to the positive side a small decrease in sensitivity occurs but it increases again when it reaches the positive side.

3.3 Different line sensors:

There are different line sensors that can be used with different advantages that are important to point out:

1. BPW17N: Several features that are provided by the phototransistor such as having an angle of viewing of $\varphi = \pm 12^{\circ}$. Considered to be good with visible and near-infrared radiation. Can detect a wavelength of 400 – 1100 nm

2. OPE5658: High speed: 25 ns rise time. Wavelength = 850 nm. Wide beam angle. Low forward voltage.

3. TSKS5400S: Peak wavelength: $\lambda_p = 950\ nm$. High reliability, power and radian intensity. Half intensity angle of $\varphi = 30^{\circ}$. High pulse current operation

4. TEKT5400S: High radiant sensitivity. 940 nm emitters matched with a daylight blocking filter. Fast response times. Half sensitivity angle of $\varphi = \pm 37^{\circ}$.

5. TSHA6200, TSHA6201, TSHA6202, TSHA6203: High reliability. Peak wavelength: $\lambda_p = 875\ nm$. Angle of half intensity: $\varphi = \pm 12^{\circ}$. Low forward voltage

6. TCRT5000, TCRT5000L: Daylight blocking filter. Emitter wavelength= 950 nm. Peak operating distance: 2.5 mm

7. 5.0mm Round LED Lamp: Water clean lens. High luminous output. Standard 5 mm round package.

8. SFH 203 P, SFH 203 PFA: Switching time: 5 ns. 5 mm LED plastic package. The (SFH 203 P) is most suitable for a wavelength of 400 nm to 1100 nm while the (SFH 203 PFA) for a wavelength of 800 nm.

9. VT900 SERIES: No specific advantages mentioned in the datasheet; therefore, it is not considered to be suitable.

Sensor:	Type	Dimensions (L x W x H)(mm)	Wavelength (nm)	Daylight filter	φ
BPW17N	Detector	3.3 x 2.4 x 29.8	400 – 1100	-	$\pm 12^{\circ}$
OPE5658	Emitter	5.7 x 5.7 x 32.7	850	-	$\pm 22^{\circ}$
TSKS5400S	Emitter	5 x 2.65 x 5	950	-	30°.
TEKT5400S	detector	5 x 2.65 x 5	940	Contains filter	$\pm 37^{\circ}$
TSHA6200, TSHA6201, TSHA6202, TSHA6203	Emitter	5.8 x 5.8 x 35.9	875		$\pm 12^{\circ}$
TCRT5000, TCRT 5000L	Sensor	10.2 x 5.8 x 7	950	Contains filter	-
OVL-5521	Emitter	5.9 x 5.9 x 32.7	-	-	$\pm 8^{\circ}$
SFH 203 P, SFH 203 PFA	Detector	5.9 x 5.9 x 34	400 -1100 (SFH 203 P), 800(SFH 203 PFA)	-	$\pm 80^{\circ}$
VT900 SERIES	Emitter	-	-	-	-

Wavelengths, angular response, and rejecting unwanted signals such as the direct sunlight [2] were the criteria taken into consideration when pairing emitters and detectors. Both VT900 series and the OVL-5521 are not taken into consideration due to the lack of required information on related datasheets. The TCRT5000 includes a sensor with a built-in detector and emitter, a direct daylight blocking filter, and a wavelength of 950 nm. The TSKS5400S (emitter) and TEKT5400S (detector) form a pair since the wavelengths of the emitter and detector are compatible with the advantage of the TSKS5400S having a day-light filter and a fast response time. The half intensity angle of the detector also matches with the half sensitivity angle of the detector. The OPE5658 (emitter) and the SFH 203 P(detector) also form a pair due to the 850 nm wavelength of the emitter and the 400 nm to 1100 nm range of detection for the detector. The emitter has a high speed of 25 ns rise time and is

good for pulse operation. Another pair to consider is the TSHA6200 (emitter) and the BPW17N (detector).

Based on these considerations and comparisons, it was decided that the TCRT5000 is the most suitable sensor for the buggy since it is pre-built which reduces errors that may arise when trying to pair an emitter and detector. Moreover, it includes a day-light filter.

When the LED is off the sensor is detecting ambient light. When a fluorescent light is used, it is observed that the fluorescent light does not emit on the infrared frequency that the sensor measures [2]. On the other hand, when the incandescent light is used, the incandescent light will emit on the same infrared frequency that the detector measure, therefore the sensor will be sensitive to direct sunlight and works [2]. The TCRT5000 will thus be placed on the bottom part of the chassis in order to block the unwanted direct sunlight that will have severe effect in terms of the sensor following the track fixed white line. In terms of the dark current measurements, the dark current is a small current that is associated with the photosensitive devices [3]. To start measuring the dark current, which is the lowest value that the sensor gives, a large resistance is required with a high input impedance voltage measuring device while blocking the light from the sensors [2].

The implemented PID control theory will deal with the line-break problem in the track. PID control theory stands for Proportional-Integral-Derivative, it is mainly used in order to eliminate the error in the controller with time [4]. The implementation can be with proportional, PI or PID, for the desired buggy PID was suggested as the suitable method in order to control the buggy due to the damping property that the derivation in the PID will provide to the system as shown in **equation (1)** which will allow the buggy to move more smoothly and overcome obstacles such as the line-break [1].

$$\mathrm{u(t)} = \mathrm{K_p e(t)} + \mathrm{K_i} \int_0^{\mathrm{t}} \mathrm{e(t^*)dt^*} + \mathrm{K_d} \frac{\mathrm{de}}{\mathrm{dt}} \quad (1)$$

Where K_d: derivative gain, K_i: integral gain, K_p: proportional gain.

It is important to point out the role of both the proportional gain as shown in **equation (1)** and the integral gain. Regarding the proportional gain, the motors speed difference is proportional to the distance the buggy is away from the line-break (black line). If the proportional gain was small, the buggy should move further away from the line-break before any unwanted correction actions are taken by the controller, therefore large proportional gain will cause the controller to be more active which could lead to the buggy moving out of the line [1].

The integral gain will solve the problem that the large proportional gain causes. Applying the integral as shown in **equation (1)** will fix the error of moving out of the line. Coding is an important part regarding the control theory thus, having constant sampling times when coding is essential in order to eliminate errors [1]. The software and programming part of the construction of the buggy will play the main role in order to overcome the line-break problem, by writing the suitable code for the implementation of the control theory.

3.4 Measurements and errors:

In terms of the possible errors that could have affected the overall results, the vertical experiment did have an error of $\pm 1.5\ mm$ moreover, the accuracy of the scope voltage was found to be $\pm 0.08\ V$. It is also important to consider that the height experiment was conducted using a ruler which also has an error of $\pm 0.05\ mm$.

Overall, the error presented in the height experiment is considered to be normal. While for the horizontal experiment the voltage scope did have the same error as for the previous experiment but, the gap measurement did have an error of $\pm 1\ mm$. Furthermore, an error such as the direct sunlight could be present which will affect the sensor's measurement if the light was detected and emitted. The wrong placement of LEDs with no current limiting resistor could lead to the destruction of the LED. Also, an error that could occur is cutting the stripboard incorrectly which could produce an unwanted short circuit in the circuit that was built for the experiment [2].

4. Circuit diagram for proposed line sensors

In this section, the circuit diagrams that were drawn are shown to provide a graphical representation of the electrical circuits and the connections that will be made to build the buggy.

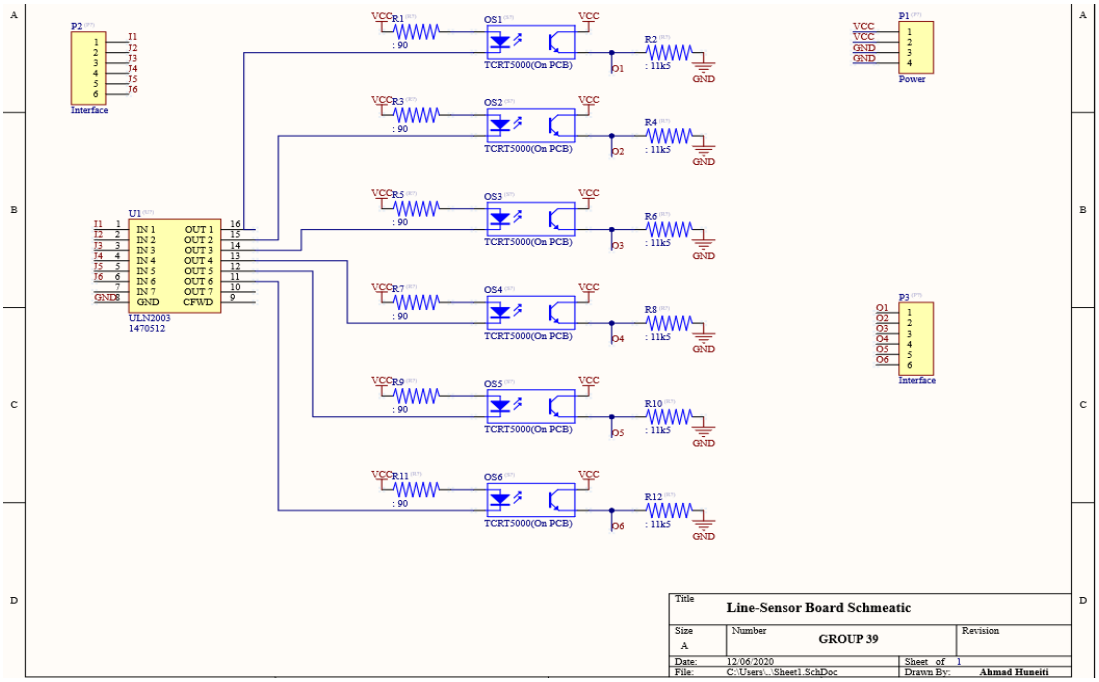


Figure 5: Schematic Diagram of the line-sensor board.

In **Figure 5,** P2 and P3 represent 6-pin headers that allow data to be transmitted between the microcontroller and the line sensors. P2 connects the microcontroller to the inputs of U1 which represents the Darlington buffer. P3 connects the outputs of the sensors to the microcontroller. A resistor value of 90 Ω was chosen for the LED as this lets enough light to be emitted without exceeding the maximum LED current of 60mA. A value of 11k5 Ω was chosen for the transistor.

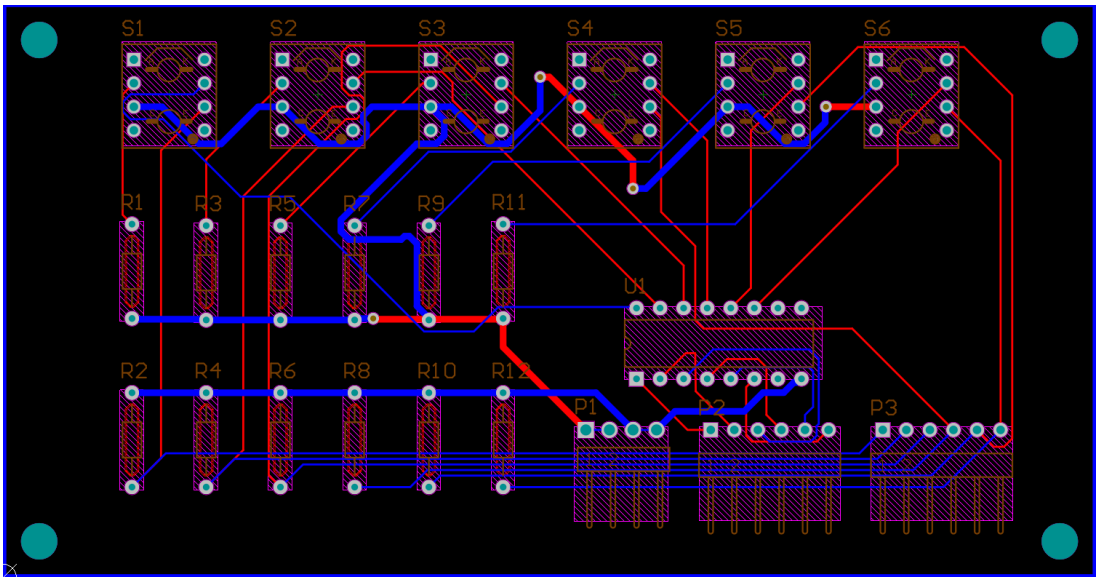


Figure 6: PCB layout of the line-sensor board.

The dimensions of the PCB in **Figure 6** are 118 mm x 62 mm. The distance between the sensors is 16 mm and this specific distance was chosen to make sure that the sensors are able to detect the white line all the time.

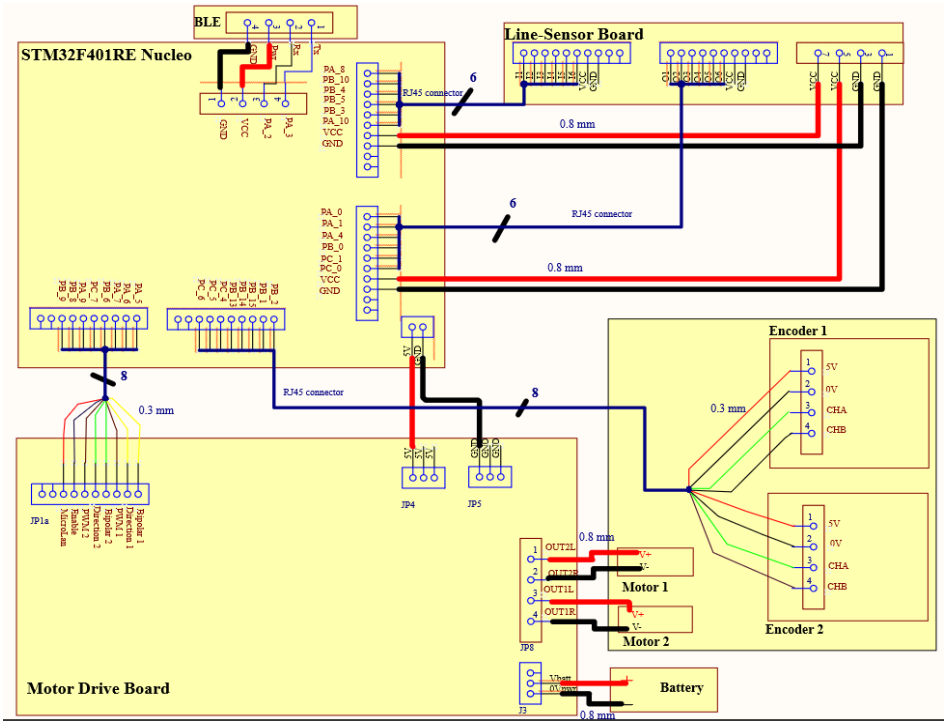


Figure 7: Wiring Diagram

As shown in **Figure 7**, 9 V AA batteries are connected to the drive board that includes a DC-DC converter which provides a 5 V supply to other boards such as the microcontroller, the line-sensor board, and the wheel encoders. The batteries power the motors directly using the motor outputs which should be limited to 10 V. The line-sensor board's inputs and outputs are connected to pins of the microcontroller that support analogue to digital conversion in the case it is needed. As for the motor drive board, its pins are connected to GPIO pins that are able to generate PWM to drive

the motors. The BLE module is connected to UART pins of the microcontroller.

5. Non-line sensors

This section highlights the importance of using sensors to determine some parameters of the buggy such as the speed, direction, and battery level. A Quadrature encoder provides two 90° out-of-phase output channels that produce digital square wave pulses as the buggy starts to move. By comparing the pulses produced by the two channels the direction in which the buggy moves may be figured out [5]. The speed of the buggy may be determined by measuring the time period between two certain events such as a rising edge. In this buggy, a Dallas DS2781 integrated circuit is used to give a precise voltage, temperature, and current measurement of the battery [1]. This is done by connecting a single wire to any GPIO pin on the microcontroller.

The gearbox provided contains an AEAT-601B-F06 hall effect quadrature encoder which will be connected to the STM32F401RE microcontroller to allow data acquisition. The microcontroller can measure the encoder by connecting it to the GPIO pins of the microcontroller and producing interrupts when certain events occur. For example, to determine the direction of the buggy an interrupt may be produced when a rising edge occurs on one of the channels. At this point, the two channels may be compared and depending on whether the encoder is rotating clockwise or anticlockwise either one of the channels will lead the other [6]. The speed may be measured by starting a timer for a certain period of time to be able to calculate the encoder tick rate. Then, by using TIM2 and TIM5 on the microcontroller the counters automatically count the number of pulses generated by one of the channels by using X2 encoding. The speed of the buggy may be calculated using the following equations [1]:

$$\text{Encoder Tick Rate} = \frac{(\text{Current Encoder Ticks} - \text{Previous Encoder Ticks})}{\text{Sample Time}} \quad (2)$$

$$\text{Wheel Velocity} = \text{Encoder Tick Rate x f}(\text{wheel diameter}, \text{gear ratio}, \text{CPR}) \quad (3)$$

Where f is a function of three variables and CPR is the number of counts per revolution. The gear ratio and wheel diameter are known.

$$\text{Translational velocity} = \frac{\text{Wheel Velocity Left} + \text{Wheel Velocity Right}}{2} \quad (4)$$

$$\text{Angular Velocity} = \frac{\text{Wheel Velocity Left} - \text{Wheel Velocity Right}}{\text{Distance between wheels}} \quad (5)$$

In addition to the sensors mentioned above, a Bluetooth will also be added to allow data to be exchanged. An HM-10 BLE module will be used since it is easy to use and can be connected to the Nucleo-board using the UART serial port. Another impressive feature of the HM-10 is that it is cheap and consumes a very low amount of power considering its function [7]. Its operating voltage is between 2.0 V and 3.6 V which means that it must be mounted to a breakout board to make it 5 V compatible. After the connection is set up the module communication between it and the microcontroller is done using simple commands that have already been set up in pre-written libraries. The BLE module will be used to generate a command that will make the buggy turn around at a certain point in the track.

6. Control

The reflection coefficients of the white line and the dark part of the track are different. Depending on the light intensity of the reflected light, the six sensors will be used to calculate the distance of the vehicle to the centre of the white line to measure the position of the line. In other words, the line will be detected by receiving voltage signals from the line sensors.

When the buggy is not moving above the white line, some sensors will measure the distance and send a signal to the PID controller, and then the signal will be sent to the two motors, the speed of the buggy will then be adjusted to move along the white line. The speed of the two wheels is calculated as follows:

$$E = \frac{\sum_{i=0}^{6} v_i^1 * I}{\sum_{i=0}^{6} v_i^1} - 3 \quad (6)$$

$$V_L = -K_P * E + K_D * E' + V_N \quad (7)$$

$$V_R = k_P * E - K_D * E' + V_N \quad (8)$$

$$E' = \frac{(E - E_{-1})}{\Delta_T} \quad (9)$$

Where $V_N = (V_L+V_R) / 2$ is the forward speed, K_P is proportional gain and K_D is derivative gain.

When the speed of these two wheels is known, then the direction of the buggy to guide the buggy back to the line is also acknowledged accordingly. The relationship of *Θ,* which is the quantity change in direction, is calculated by the following equations:

$$\theta = \frac{v_R - v_L}{I} \quad (10)$$

$$r = \frac{V_L + V_R}{V_R - V_L} * \frac{I}{2} \quad (11)$$

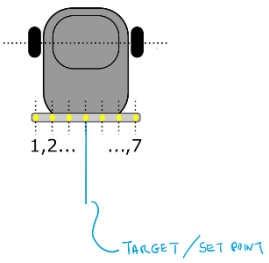


Figure 8: Rear view of buggy

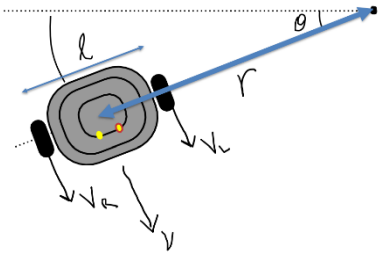


Figure 9: Buggy kinematics

The proportional controller is produced, in which an output value that is proportional to the present error value. The proportional response is adjusted by multiplying the error with a constant K_P, called proportional gain constant [4].

If the system being controlled has a rapid response time, the proportional controller will modulate the output to the controlling device and it will increase the stability of the buggy. The power output is always proportional to the error. The speed of the buggy changes gradually in proportion to the change of the error. Using a proportional control algorithm is more complex than using a bang-bang control because it becomes hard to determine the appropriate value of K_P. Also, a

proportional controller cannot eliminate the residual SP − PV error in processes of compensating, because it needs an error to generate a proportional output.

The benefit of using bang-bang control is that it frequently arises in minimum time and is easier and more convenient to conduct in a system compared to a proportional controller. A drawback that may arise is that depending on the width of the hysteresis gap and inertia in the process, there will be an oscillating error signal around the desired setpoint value. Another drawback is that it may cause the buggy to zigzag across the line resulting in instability and speed variation.

A PID controller consists of three basic coefficients: proportional (K_P), integral (K_I) and derivative (K_D). These gains are varied to achieve an optimal system response. The desired value setpoint (SP) and the measured output process variable (PV) are compared and the result is an error value which is used in calculating proportional, integral, and derivative responses. By summing the three responses it is possible to obtain the output of the controller. The output of the controller is used as an input to the system being controlled and to change some aspects of the system [4].

Proportional (P):

The relationship between the difference in the motors' speeds, with the terms, manipulated variable (u), and the error (e) is given as:

$$u(t) = K_P * e(t) \tag{12}$$

Where t is time and K_P is the proportional gain. Proportional gain is the first quantity that needs to be decided, but it is difficult to determine an appropriate value. Having a proportional gain that is too large can lead to instability of the system, while too little will cause the buggy to respond slowly when tracking the line. If you apply only proportional control, you may find that it doesn't keep the vehicle directly above the line. Instead, the vehicle may stay slightly to one side of the line.

Integral(I):

In a PID controller, the integral is the sum of the instantaneous error over time and gives the accumulated offset that should have been previously corrected. The accumulated error gets multiplied by the integral gain (K_I) and added to the controller output [4]. To solve the problem that only proportional control works, the integral gain needs to be applied to eliminate the steady-state offset. The equation for this controller is:

$$u(t) = K_P * e(t) + K_I * \int_0^t e(\tau)\, d\tau \tag{13}$$

where K_I is the integral gain. The integral term eliminates the residual steady-state error that occurs with a pure proportional controller and accelerates the movement of the process towards setpoint. However, since the integral term responds to accumulated errors from the past, it can cause the present value to overshoot the setpoint value.

Derivative(D):

A derivative term does not consider the error, but the rate of change of error is trying to bring this rate to zero [8]. The equation for a PID controller is as follows:

$$u(t) = K_P * e(t) + K_I * \int_0^t e(t')\, dt' + K_D^* \frac{de}{dt} \tag{14}$$

where K_D is the derivative gain. Derivative action can provide damping to the system, so it should make the vehicle move more smoothly. Furthermore, derivative action predicts system behaviour, thus it improves settling time and stability of the system [8]. However, a significant problem with derivative action is that it can amplify any noise that you have in your sensor measurements.

In general, the actions dictated by the controller are the buggy's performance in a wide range of operating conditions which include:

1. Controlling the speed of the buggy. Using the encoder to read the speed and change to the desired speed value. The PID controller is much better in this case. By detecting the buggy's speed, the PID controller can then keep the speed constant. The PID controller automatically adjusts the voltage to increase torque on the uphill climb and decrease it on the downhill slope. This process is accomplished by changing the duty cycle of PWM.
2. Controlling the direction of the buggy. In details, the sensors detect the line location and the controller determines if the buggy needs to change direction. This makes the PID controller an optimal choice. After detecting the location of the buggy by the sensors, the PID controller will reduce the overshooting and decrease oscillations around the setpoint. When the width of the pulse is changed, the average voltage of the signal changes and the motor drive board manages to turn right or left. As a result, the buggy can follow the white line in the correct forward direction.

The designed layout of the buggy implements 6 sensors which indicate the distance of the vehicle to the centre of the white line. These sensors are fixed at the front of the chassis board in a line. This kind of design enables the buggy to look for the line more quickly and more conveniently. The reference value and the measured output are compared and the result is an error value which is used in calculating PID controller responses. The error value for each sensor will be uploaded to the controller and it will be processed by the algorithm code. The output of the PID controller is used as an input to the system for controlling and changing the direction in which the buggy is moving and the movement along the white line.

It is recommended that the sensors are spaced less than the width of the white line which means that, when the line is exactly between two sensors, they both react and the controller can get twice the accuracy with more precise signals. This will influence the frequency of the PID algorithm execution by making it more efficient.

The TCRT5000 works by transmitting infrared light from the LED and registering any reflected light on its phototransistor. This alters the flow of current between its emitter and collector according to the level of light it receives. The analogue pin A0 provides a continuous reading in the form of varying voltage, the higher the voltage the more infrared light is being received. The digital pin on the other hand is either high (on) or low (off) [9]. When the board is powered and not enough infrared light is received the digital pin will be high, and the LED is lit, which indicate the sensors detecting the white line. When the trigger level set by the potentiometer is passed, the digital pin is then set to low and the LED will be in the out state which shows that the sensor detects the black line. The sensors send signals to the PID algorithm controller as an element in the microcontroller and then the PID algorithm will decide whether the buggy will change direction. Furthermore, a quadrature encoder provides output channels that produce digital square wave pulses. By comparing the pulses, the direction in which the buggy moves may be figured out using the PID algorithm.

The PID control algorithm will read analogue data which will produce more reliable results. As a result, the sensors are connected to GPIO pins of the microcontroller that will support analogue to digital conversion.

There 6 sensors, which are all of type TCRT5000, attached to the line sensor board which is placed at the front of the buggy. TCRT5000 consists of high transmission power of infrared photodiode and high sensitivity photoelectric transistor. The output signal is processed by the Schmitt circuit, which becomes more stable and reliable. By placing the sensors in a line and at the front of the buggy, the sensors will detect any change and send data to the control algorithm which will in turn respond accordingly. The sensors are arranged in a way to allow a spacing of 16 mm between them since the distance between every two sensors should be less than the width of the line which is about 17 mm. The sensors are connected to GPIO pins of the microcontroller that support analogue to digital conversion. It allows the PID control algorithm to read analogue data which will produce reliable results since PID is more accurate when analogue data is used.

Implementing the algorithm can be done with a microcontroller, which is flexible. It is required to pick an appropriate sampling time: 1/10 to 1/100 of settling time which should be considerably precise, ideally within 1% accuracy. The accuracy is controlled by using a timer interrupt. It cannot be too fast nor too slow in order to get the right accuracy. Subsequently, interactive commands have to be used to set K_P, K_I, K_D which determine whether it is possible to program an adaptive PID controller. The whole process needs to satisfy the equation as follows:

$$\mathrm{u(t)} = \mathrm{K_P} * \mathrm{e(t)} + \mathrm{K_I 1} * \int_0^{\mathrm{t}} \mathrm{e(t')\,dt'} + \mathrm{K_D^*} \frac{\mathrm{de}}{\mathrm{dt}} \quad (15)$$

And this can also be approximated in discrete time as:

$$\mathrm{u}(t_k) = \mathrm{u}(t_{k-1}) + \mathrm{e}(t_k)[K_P + K_I T_S + K_D T_S] + \mathrm{e}(t_{k-1})[-K_P - 2K_D T_S] + K_D\, T_S \mathrm{e}(t_{k-2}) \quad (16)$$

The line breaks associated with the track are up to 6 mm. When the buggy encounters a line break, the buggy should just ignore it and keep the previous state of motion. The buggy has to be programmed so that if it detects a line break that is greater than 6 mm, which means it has reached the endpoint, it should stop. This could be done by using the quadrature encoders which are able to output the position of the buggy.

Included with the TCRT5000 is a day-light blocking filter which reduces the effect of sunlight on the readings of the sensor. Also, by calculating the different PID parameters in different light conditions, like strong, medium, and soft light a code can be written to change the PID parameters according to different light conditions. This method plans to use the Application Shield, whereby the potentiometer will be used to set the brightness and then the program will change the settings accordingly.

7. Hardware overview

The hardware section from Design Report 2 represents 2D drawings of chassis and extra parts for construction of the buggy. The extra part is a battery support under the chassis and it also has the role of holding the line sensor board. The report

includes 3D stress analysis/deflection for chassis, calculation of deflection and material choice.

For the construction of chassis, the most common materials are Acetyl, Glass-reinforced, Aluminium, and Mild Steel. In the table below are these materials along with some of their properties

Comparison of material properties table [1]

Material Characteristic	Acetyl	Glass-Reinforced Laminate	Aluminium	Mild Steel
Density (Mg/m^3)	1.8	1.9	2.7	7.8
Flexural strength (MPa)	91	255	310	414
Ultimate Tensile Stress (MPa)	67	175	310	414
Young's Modulus (GPa)	2.8	11.5	69	207

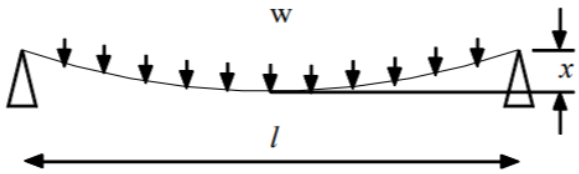


Figure 10: Distributed load [1]

If there is a piece of length = l, with width = b and height = h, and weight (W) is applied along the piece (**Figure 10**), the deflection (x) can be calculated:

$$x = \frac{Wl^4}{384E\frac{bh^2}{12}} \tag{17}$$

Where E is Young's Modulus.

Suppose there are 4 chassis (made of Acetyl, Glass-reinforced, Aluminium and Mild Steel). All with the same length, width and weight. The only difference is the height.

There is the same length and width, so the only difference is height, and it is inverse proportional with density (d).

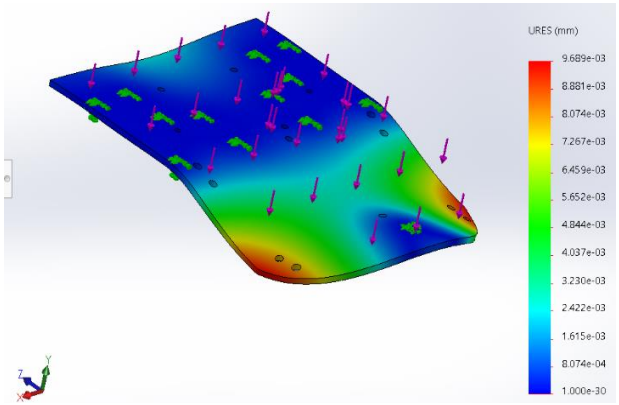
Since 384, 12, W, l and b are constants, the equation is simplified to the following:

$$x = const\frac{d^2}{E} \tag{18}$$

For Acetyl, $x = const\ 1.16$, Glass-reinforced, $x = const\ 0.3139$, Aluminum, $x = const\ 0.106$ and Mild Steel, $x = const\ 0.294$.

In ideal conditions, when the mass is equally distributed, the aluminium deflection is three times lower than Glass reinforced and Mild Steel and over ten times lower than Acetyl. Considering the deflection, the best material for the chassis is aluminium.

It is very easy to make a chassis from acetyl. Recently, 3D printers are very common and it is simple to make complex forms. Glass reinforced, Aluminium and Mild Steel probably need to be cut with advanced machines using high-pressure water jets and it is almost impossible to create 3D pieces, like a battery support. The mass of the buggy will be under 1.5 kg, the stress will be low and the deflection will be negligible. Due to the points above, the ease of manufacture is more important than the deflection, so the best material for the chassis is acetyl.



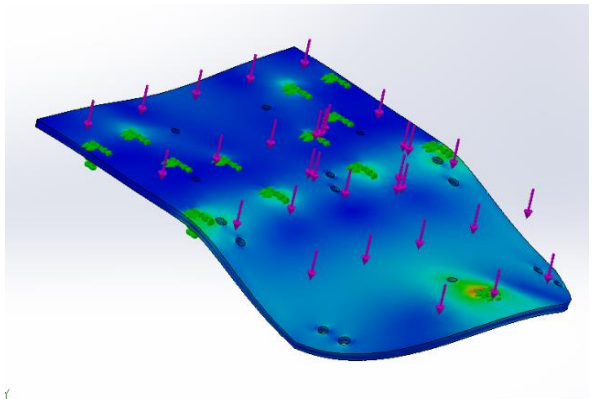


Figure 11: Solidworks deflection

Figure 12: Solidworks stress analysis

Figure 11 and **12** are Solidworks simulations on acetyl chassis when a uniform force of 30 Newton is applied. The maximum deflection from **Figure 11** is 9.6e-3 mm around the corners. The stress analysis shows that most of the pressure is on the front wheel. But considering that the chassis will have the majority of the weight on the back, the deflection and stress in the front of the buggy will not be a problem.

Most of the weight is distributed on the back of the buggy (Motors, gearboxes, wheels). The Battery pack will be under the chassis, between gearboxes and castor wheel. In this way, the centre of gravity will be under the chassis, and the buggy will be stable. In **Figure 12**, the largest stress is on the front wheel. But in reality, the mass is not equally distributed. The stress on the front wheel will be a part of the battery pack, a part of battery support, a part of sensors and the front part of the chassis. The rest will be on the rear wheel. The motors are in the back, so the buggy has rear traction. This is helpful for going up the slope. The striations on the rear wheel will help with traction.

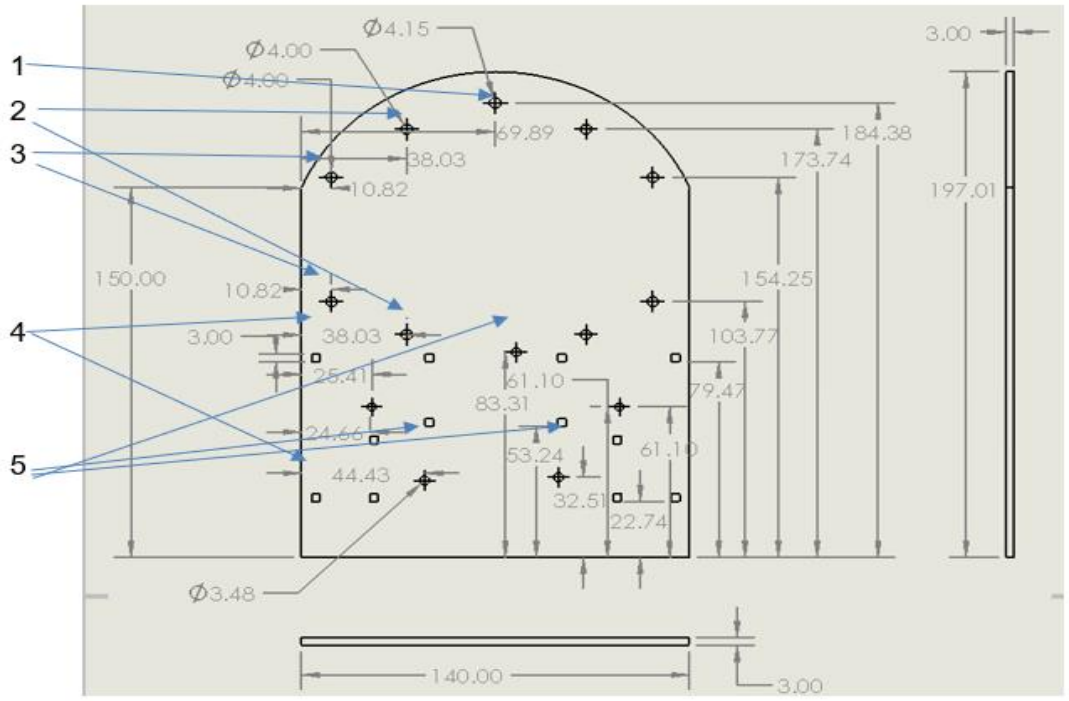


Figure 13 – 2D drawing of chassis

Figure 13 is the drawing of the chassis, with all holes. The dimensions are in mm. The gearbox will fit in the square holes (4), the castor wheel will fit in the front hole (1), the Motor Driver Board will fit in circular holes (3), the battery support will fit in the holes, above and below Motor Driver Board (2) and the Nucleo board will fit in circular holes between gearbox holes (5).

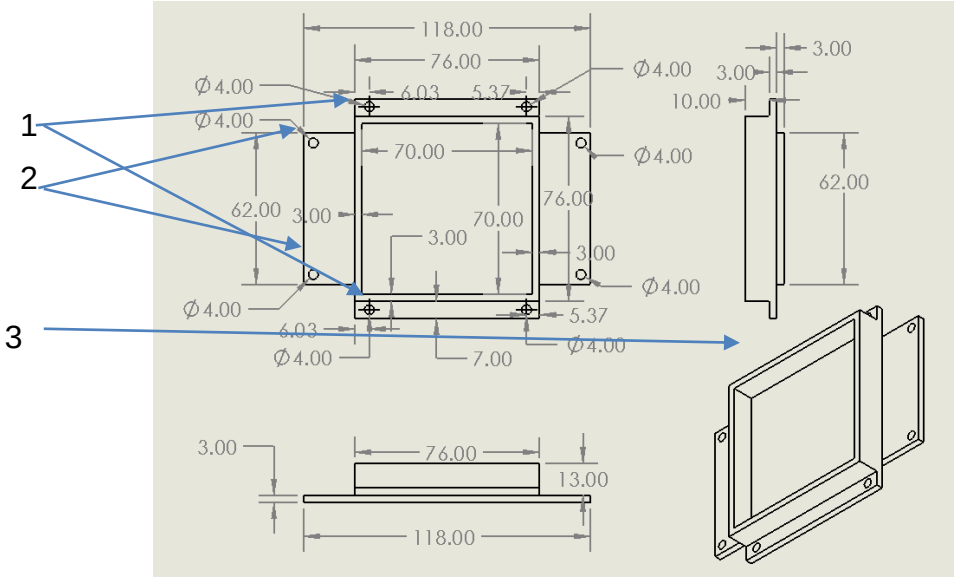


Figure 14 – Battery support drawing

Figure 14 is the support for the battery and line sensors. The piece is fixed to the chassis through 4 long screws (2). The height of the walls (3) is 10 mm. The battery stays inside the walls. The sensor board will fit in lateral holes (1), under the battery support. The height of the sensors from the ground is adjusted with screws.

With all the considerations above, the chassis made from acetyl will be sturdy enough to finish the race and so deflection shouldn't be a problem. The position of the sensor module (between the front wheel and rear wheel) is not the best for fast response, but it is easy to fit all components, without the need to add one more level. In this way, the buggy will be lighter and it will perform better.

8. Summary

The report is focused on the characterization of the line sensors and on the design and control implementation of the buggy. This process resulted in the following outcomes and key findings:

- The selection of the most suitable light sensor is TCRT5000 after tedious comparison with other options. The resistance of the LED of the emitter is 90 Ω and the resistance of the collector is 11k5 Ω. There also had to be a selection and consideration of different non-line sensors which will help the buggy to efficiently execute the different programmed tasks.
- The design of the PCB and showing the connections between the microcontroller and the circuit is also a fundamental outcome from this report. It has been decided that the dimensions for the PCB are 118 mm x 62 mm, which will make it easy to fit on the buggy.
- The chassis has been designed to accommodate the 6 sensors in a line at the front and is able to withstand various stress as it can be seen in Figure 12.

The structure of the buggy is rigid and stable. This has also been achieved by using acetyl which is the most appropriate material for a buggy of this size.

- The algorithm used for this buggy to work is the PID algorithm and in the software section it has been possible to formulate how to tackle the different adversities that the actions required by the buggy might present.

Some of the potential issues that may be encountered are direct sunlight affecting our sensors and breaks in the line that might make the robot stop. To overcome the sunlight, it has been decided to place the sensors very close to the line (the lowest point of the chassis) or to calculate different PID parameters to avoid any sort of interference with the job of the sensors. To resolve the issue of line breaks it has been decided to write an efficient code that is capable of tackling that sort of scenario and differentiate between the end of the track and a tiny gap in the line.

9. References

[1] Marsh, L., Embedded Systems Project Technical Handbook 2020-21. University of Manchester, UK, 2020.

[2]: University of Manchester. (2020). ESP: Sensor Characterisation Lab. [Online]. Available:https://video.manchester.ac.uk/faculties/eb93b3a8b5a268cd92d4a041fcd72231/64278804-9cbf-4d33-81ba-0a9424fb5886

[3]: Merken, P. and Vandersmissen, R., 2016. Dark current and influence of target emissivity. Photonics Imaging Technol.

[4] Jibrail, S.F. and Maharana, R., 2013. PID Control of Line Followers (Doctoral dissertation).

[5] Dynapar.com. 2020. Quadrature Encoders - The Ultimate Guide. [online] Available at: https://www.dynapar.com/Technology/Encoder_Basics/Quadrature_Encoder/ [Accessed 10 December 2020].

[6] Didel.com. 2020. [online] Available at: http://www.didel.com/mot/RomEnco.pdf [Accessed 10 December 2020].

[7] Components101. 2020. HM-10 Bluetooth Module. [online] Available at: https://components101.com/wireless/hm-10-bluetooth-module [Accessed 10 December 2020].

[8] Ctms.engin.umich.edu. 2020. Control Tutorials For MATLAB And Simulink - Introduction: PID Controller Design. [online] Available at: https://ctms.engin.umich.edu/CTMS/index.php?example=Introduction&section=ControlPID [Accessed 10 December 2020].

[9] Instructables.com. 2020. [online] Available at: https://www.instructables.com/TCRT5000-Infrared-Reflective-Sensor-How-It-Works-a/ [Accessed 10 December 2020].