EEE3099S Milestone 3

Group Number	5
Date	03/10/2019
Ricky Jericevich	JRCRIC001
Mehta, Ronak	MHTRON001
Aronson, Jamie	ARNJAM004
Martins, Danilo	MRTDAN014

0 WORK DISTRIBUTION

Table 1 Work distribution for Milestone 3

Task Description	**Person(s) of Responsibility**
Report writeup	ARNJAM004, JRCRIC001
Report editing	ARNJAM004, JRCRIC001, MRTDAN014, MHTRON001
System testing	ARNJAM004, MHTRON001, JRCRIC001
Mechanical Assembly	JRCRIC001
Mechanical Alterations	MHTRON001, JRCRIC001, ARNJAM004
Veroboard design	MHTRON001, ARNJAM004
Maze mapping algorithm design	MRTDAN014
Shortest path algorithm implementation	MRTDAN014
STM functionality code	JRCRIC001
Motor gearbox modification	MRTDAN014

1 INTRODUCTION

1.1 INTRODUCTION

The task was to design, build and test a system against a set of specifications. The purpose of the project was to provide insight to understand the intricacies of the design of systems, including sub-system design and testing. The system that was to be developed was an autonomous multi-wheeled vehicle that would participate in a line maze race where the maze was unknown.

The race was composed of two parts. The first part involved the robot tracking a line and learning a maze. The robot should have then completed the maze again using the shortest path possible. In the second part, the robot was required to participate in a timed event where it would race through the maze on the shortest path that the robot learnt from Part 1. The race must start at the push of an easily accessible button and end once the robot had located itself on the black circular patch. Limited time was given for both parts.

1.2 AIM AND OBJECTIVE

The aim of the project is to design an autonomous multi-wheeled robotic platform capable of starting at the push of an easily accessible button, following a black line (track), solving a line maze, finding and learning the shortest path from the start to the end of the maze. The robot is further required to indicate that it has found the shortest path by stopping and blinking an LED. Once it has learnt the shortest path from the start and end of the maze, the robot is required to take part in a timed event where it must race through the line maze on the shortest path (black line) it has learnt. The robot is required to start the race at the push of an easily accessible button and end the race by stopping and locating itself on the black circular path with diameter of 150 mm.

1.3 DESIGN REQUIREMENTS

1.3.1 Specifications

For part 1, the robot must

- Start at the push of an easily accessible button
- Sense and follow a line
- Detect the end of the maze, which was a black circular patch of diameter 150 mm
- Find the shortest path to the end of the maze
- Stop and blink an LED on completion

For part 2, it must

- Start at the push of an easily accessible button
- Race through the maze using the shortest path found in part 1
- Stop the end of the maze, which was a black circular patch of diameter 150 mm

Other specifications include a maximum volume of 150 mm x 150 mm 100 mm, and being self- powered and autonomous

1.3.2 Additional Requirements

The following extra requirements were decided upon in order to allow for problem prevention/detection.

Modularity

The robot must be easily separated and/or assembled for it to be flexible to changes. More specifically, the chassis of the car and the motors must be easily separable in case a problem occurs with them. For the circuitry, each sensor must be made on its own piece of Veroboard so that it is easier to identify the faulty sensor and/or the faulty component of the sensor. The microprocessor must also be placed on its own piece of Veroboard in such a way that it is easily swappable in case it breaks due to a short circuit or a current overload. The same applies for the motor driver and voltage regulators.

Sensor placement

The sensors must be placed in a way that allows for the tracking of the black line during straight paths, as well as for the detection of corners, junctions, U-turns and the circular finish track. Additionally, they should detect the line successfully from 5mm to 30mm above the line.

Stability

The robot must not wobble in any way during operation and must not tip over during turnings i.e. it should have a stable base.

Simplicity

The robot's mechanical design should be aimed at achieving the core functionality of being able to detect and follow the black line, and the circuit design must be easily understandable so that the debugging of any errors is simplified.

Cost

The required components for building the robot should be readily available and at a relatively reasonable cost, with an upper limit of R600.

Shape

The robot should be designed to distribute the weight evenly on the chassis. It should be able to optimize space availability of the components.

Weight

The robot must have a relatively low weight to allow for ease of movement, maximum speed and low energy consumption.

2 Sub-system Description

2.1 Introduction

The process of completing this project was split up into 3 main phases: mechanical design, sensor design and algorithm implementation.

The mechanical design phase entailed a physical design for the robot such as the shape of the chassis as well as the placement of the various components such as the sensors, motors/wheels and the battery and microprocessor. In the sensor design process, line sensors were designed and built to the specifications. With regards to the algorithm implementation, all the processes should have come together to produce a working model based on the design specifications.

2.2 Mechanical

During this process, three potential designs were modelled and evaluated. This evaluation included comparisons of the robustness of each different design, how well it would complete the task of detecting the line based on sensor placement, as well as where the various components would fit on the structure. A process of designing the specific parts of the robot in order to have them laser-cut out of Perspex was also undertaken. This was done using CAD tools. In order to most effectively compete in the race aspect of this project, modifications were also made to the gearbox of the motor to increase its rotational speed.

2.3 Electrical

The first part of the electrical sub-system involved developing a circuit for the sensor. The second part involved deciding on appropriate components to build a functional sensor and acquiring the components. A list of components was submitted to RS components and was ordered. Once this was completed, a prototype circuit was built on Veroboard and tested to ensure it works and meets the requirements. Once this was complete, the remaining sensors were built. There was also need for various other circuits to be developed such as a motor driver circuit for converting low current signal from STM to high current (around 200mA) to feed into the motors, STM32F0 circuit and the debugger circuit to mount the STM and its debugger and a voltage regulating circuit which would produce regulated 3V3 and 6V output voltages for powering up the sensor circuit & STM debugger and the motor driver circuit respectively.

2.4 Systemic (Algorithmic)

In this process, there were three distinct phases. The first part was the configuration or initialization of the STM32F0's various pins and functions, such as the GPIO ports, PWM and Timers, in order to receive the sensor's signals and drive the motors. It was decided that using interrupts for the sensing of lines was too instantaneous and would result in errors when detecting turns, so the timer was used to poll the sensors every few milliseconds to reduce the chance of turning incorrectly. Next, algorithms were developed to process the sensor signals so that the robot car turns when it needs to at junctions and dead ends and detects the end of the maze, as well as drive the motors forward or in reverse. On completion of this, a maze solving algorithm was to be developed whereby the robot explores the entirety of the maze, recording information about every node (junction) it comes across. Once the maze has been completely explored and the robot was on the black circular patch, an LED should flash. The next phase of this design process was implementing an algorithm that determines the shortest path that the robot can take in order to complete the same maze [3].

3 LOGICAL TESTING OF SUBSYSTEM AND SYSTEM LEVEL TESTING

3.1 SUBSYSTEM TESTING PROCEDURE

3.1.1 Mechanical

After the mechanical structure was assembled, it was placed under significantly more load (weight) than was required. The sturdiness and stability of the structure was then evaluated. The motors were also tested by placing a voltage over them and measuring the speed of rotation. Both tests were then conducted in conjunction with each other to evaluate the overall performance of the mechanical system.

3.1.2 Electrical

The sensor circuit was supplied with a sweeping voltage of 7V - 12V to the input of the circuit and checked to see that the output was between 2.8V - 3.3V when the sensor detects a black line and less than 500 mV when no black line is detected.

The sensor was placed 5mm away from a black line and checked to see that the correct output voltage was obtained. The sensor was then put 30mm away from the black line and checked to see that the correct output was obtained.

The sensor was also tested in different lighting conditions (i.e. a torch light was shone at the sensor) and checked to see that the correct output voltage was given for each case.

The Voltage regulator circuit consisted of two parts. One was to output a regulated 3V3 for powering up the sensor circuit & the STM debugger circuit and the other one was to output a regulated 6V for powering the motor driver circuit. The Voltage regulator circuit was given an unregulated input voltage ranging from 7V – 9V (depicting the voltage from the battery) and the outputs from both the regulating circuits were checked using a multimeter.

The STM board and debugger circuit were fused into one Veroboard and an input of 3V3 was fed into this circuit from the voltage regulator. Pins were allocated for push buttons, sensors and motor drivers from the STM32F0 microcontroller. The STM chip and the debugger were then plugged in and the USB cable was connected from the laptop to the debugger to ensure connectivity.

The motor driver circuit was supplied with a regulated input of 6V from the voltage regulator. The four input pins of the driver were also the GPIO pins on the STM and the four output pins from the driver were connected to the motors (two pins per motor). The output of the motor driver was connected to a multimeter to measure the output voltage and current to be fed into the motors. It was also checked physically to see if the shaft of the motors rotated when connected with driver output and supplied with an input voltage.

3.1.3 Systemic (Algorithmic)

After initialising the GPIO input pins for the sensors, buttons were used to simulate the behaviour of a sensor detecting a black line to check if the code was receiving the signals properly.

Once this was working satisfactorily, the PWM function on the STM was set up. After setting the PWM to have a specific duty cycle, an oscilloscope was used to check that each PWM pin was outputting a square wave with the corresponding duty cycle. This process was repeated once more with a different duty cycle to ensure it works properly.

Thereafter, the timer was set up to have a delay of around 10 ms. This delay was tested by switching an LED on when the timer interrupt triggers (after 10 ms) and then switching it off on the next timer interrupt event.

It was then time to code all the sensors and motor operation. This was done using the fully constructed car with all the circuitry and mechanical parts. The three main parts of the code for driving the car were the error correction interrupt handler, the timer delay interrupt handler and the function for turning left and right.

The two error correction sensors, unlike the other sensors, were operated using interrupts because instantaneous correction of the error was desired. This interrupt was initially tested using buttons to check if the interrupt triggers when the button is pressed.

The timer delay interrupt was already tested using the LED as stated above. The function for turning left and right was tested by checking if the wheels turn in the correct direction when the appropriate sensor detects a black line.

The algorithm for finding the shortest path was developed concurrent to all the code development. Several different virtual model mazes were created, and the sensor inputs were replaced with text inputs, as the maze mapping and shortest path code were created for a command line[2]. Each maze having unique features and different levels of complexity, e.g. no loops, one loop or two loops. The algorithm was then set to map the entire maze by means of an adjacency matrix [5] and output the list with the shortest path to the final position.

3.2 SUBSYSTEM TESTING RESULTS

3.2.1 Mechanical

It was found that the integrity and stability of the structure was satisfactory when placed under a heavy load. The speed of the motors was also increased from 20rpm to 56rpm, thereby significantly increasing the speed of the robot. Both factors tested in conjunction with one another resulted in the robot maintaining its stability while travelling at a fast speed.

When testing how the wheels were spinning on the gears, it was found that one of the motors was faulty. When the motor was placed at a certain angle, the gears would become unmeshed and the shaft would not spin. It was also found that the speed of rotation of one of the motors was less than the other.

[2] The text inputs would then be replaced with inputs from the sensors.

3.2.2 Electrical

When supplied with a voltage sweeping from 7-12V, the sensor circuit outputted a High (around 3.1V) in every instance where a black line was detected. When no line was detected, the circuit's output was around 200mV (Low)

When the sensor was placed 5mm away from the black line, the correct output of around 3.1V was given. An acceptable output of around 3.1V was also given for the sensor detecting a line from 30mm away.

Different lighting conditions had no effect on the sensor. It was able to output the correct 3.1V in every instance of lighting.

The voltage regulator gave a regulated 3V3 and 6V every time when an input voltage was swept from 7V to 9V.

The STM and debugger circuit initially had an error in establishing a connection with the laptop when a USB was plugged in, but after debugging the debugger, it was clear that there was a missing ground connection to the STM. Once this was modified, the STM and debugger circuit worked perfectly fine and the debugger flashed when the USB was plugged in.

The motor driver worked fine from the get-go when a regulated 6V input was fed into it. The output pins of the drivers measured 5V8, 0.2A which would be fed to the motors. The shaft of the motors also rotated when the driver output pins were connected to it.

3.2.3 Systemic (Algorithmic)

The GPIO pins, PWM and Timer was working just as intended. The GPIO pins were shown to be working properly by checking to see if they were receiving the sensor's signals by using both a voltmeter and the debugging function of the STM. The PWM operation was tested using the oscilloscope and initially wasn't working. After rechecking the code, it was noticed that the incorrect output pins were used and after fixing this, the PWM was showing square waves with the correct duty cycle on the oscilloscope, so it as working perfectly. For the timer's delay, the LED was initially constantly on, showing that the 10 ms delay wasn't working properly. It was concluded that the values in the prescaler and auto reload register were incorrect. For some reason the calculations to achieve a 10 ms delay weren't working properly, so after many adjustments to the register values an approximate delay of 10 ms was achieved: The LED switched on after about 10 ms and off after about 10 ms.

When testing the maze mapping algorithm using a virtual maze[3], it was successful in exploring the entirety of each maze and outputting a correct and complete path to the circular black patch. The algorithm considered the loops and did not get stuck in an infinite loop. The shortest path algorithm was then successful in determining the fastest path for the robot to take in all cases.

3.3 SYSTEM TESTING PROCEDURE

Each time a part of the code was completed it was tested to ensure that it works properly before another part of code was written.

[3] A different mazes were drawn on paper, and the code was tested by prompting the user to input the sensor signals (left sensor high? Yes/No). These prompts would then be replaced by actual sensor inputs.

First, the code for the error correction was written for when the car goes skew off the black line. Initially, the error correction behaviour was coded for only the left side of the car. Once this was working satisfactorily, the code was extended to the other side. In order to test this, the robot was set on an intentionally skew course and was allowed to try and correct itself.

The turning behaviour of the car was then tested for when it reached a junction. As with the error correction, the code for sensing a turn and turning was written one side at a time to ensure that it works properly. To test this, the car was positioned to travel along a fashioned black line that had one turn, being the one that was to be tested. This was done with the car driving and where the robot was held by hand to avoid external complications.

Lastly, the detection of the black circle patch was coded and tested as well as the code for blinking a LED upon detecting the circle. The robot was placed along a black line with a circle at the end and allowed to drive over it.

The robot was then tested for a dead end. A straight line was created with nothing at the end of it and the robot could travel along it.

Next, the priorities of the robot were tested. The robot was given multiple potential turns and tested to see if it took the right one according to the preferred sequence i.e. left, forward, right, back.

The group was not able to test the maze mapping and shortest path codes on the actual robot, due to poor time management and fear that it would give rise to other problems. As a result, hardcoding the maze was necessary, and was tested by checking if the robot made the required turns to reach the end of the maze.

3.4 SYSTEM TESTING RESULTS

The operation for error correction needed many adjustments to make it work. The biggest problem faced when getting the error correction working was the fact that after a few hours of testing, one of the sensors had swapped polarity for no apparent reason: when over a black line its output was low instead of high. Figuring out that this was the problem took many hours as it was initially thought that the code and GPIO pins weren't working properly. After this problem was identified, the code was adjusted, and the error correction eventually worked as intended.

When sensing left and right turns, the code worked just as intended and required only a few adjustments to get the wheels turning left/right properly. Detecting the black circular patch also only required a few minor adjustments to get it working properly.

The biggest challenge was the code for detecting a U-turn. The initial code for it didn't work as there were many different cases where it would detect a U-turn when there was no U-turn. Many different algorithms were written for it but there was always at least one case that could result in U-turn when there was none. This was the main reason why the decision to abort the maze exploring and shortest path algorithms was made, as the robot was unable to perform U-turns properly in time for the demo. It was decided to hardcode the instructions so that the robot would never meet a dead end (U-turn) and so that it would reach the end of the maze.

The hardcode refused to work with the error correction code for some unknown reason, so the error correction was disabled. Thereafter, the robot was able to reach the end of the test maze. It was noticed that the robot was going skew due to the complications in the motor not being fixed (one motor was spinning faster than the other), so it often needed manual realignment.

The robot was unsuccessful navigating through the hardcoded path due to an error in the hardcode, as it did not find the black circle. A number of manual realignments were required over the long straights in the final maze.

4 COMMENTARY ON THE MECHANICAL DESIGN

Towards the end of the project, it was found that the motors were turning too quickly (at 140rpm) and not delivering enough torque. This was causing the robot to get stuck when turning. In order to combat this, an extra gear was placed into the gear configuration which resulted in a lower speed (56rpm) and higher torque. After doing this, the one motor's gears weren't meshing properly which resulted in it turning slightly slower than the other motor. Many attempts at fixing it were made, to no avail. This caused the robot to drive skew in the final demo.

When the wheels were laser cut, it was found that there was a taper to the part of the wheel that was supposed to fit around the axle. This presented some difficulty as the wheels had to be forced to fit on the axle.

The group did not come up with a clever way of securing the wheels in place which resulted in them spinning in a skew direction.

It was also found that once the robot was fully assembled, the wires from the sensors were dragging on the ground while the robot was driving. This caused the robot to drive skew. In order to correct this, new, larger wheels were designed and fitted onto the robot.

Initially, the sensor orientation was planned to be vertical but while soldering the sensor circuits, the IR LED's position was not accounted for, and thus it led to the orientation being horizontal. In order to save time and effort, no unsoldering was conducted and instead the sensor design placement was altered which resulted in two of the sensors being placed right underneath the motors. The sensors worked efficiently regardless of this error in orientation. This sensor placement however made disassembling the robot extremely challenging and time consuming. The group was forced to disassemble and reassemble numerous times. That time could have been better spent elsewhere had the sensors been placed smarter.

As a result, new and larger wheels had to be laser cut, so that the sensors could fit under the motors.

Overall, the chassis had a good design which allowed for modularity and the adjustment of wheel position based on other needs.

5 COMMENTARY ON THE ELECTRICAL DESIGN

The sensors that were used on the robot were built proficiently and reliably. Each individual sensor worked properly and accomplished its task. Throughout the whole testing process, none of the sensors broke, however one did swap polarity which caused some confusion. It wasn't clear why the polarity was swapped even after hours of debugging and hence the code was altered in such a way as to accommodate for this change.

When testing the two error correcting sensors, it was found that because they were so close together, the IR light that was being emitted from one LED was triggering the IR sensor on the adjacent Veroboard. This caused the error correcting function to not work. This was solved by adjusting the height of the sensors so that there was some vertical distance between them.

The orientation of the front most sensor (responsible for straight black line tracking) had to be changed from a horizontal orientation to a vertical one to ensure that the sensor would be as close to the ground as possible for optimal sensing. This change in orientation also enabled the robot to fit well inside the length restriction of 150mm.

The Voltage regulator circuit, STM, debugger circuit and the motor driver board all worked proficiently and were able to maintain that standard throughout the entire build and testing process. The circuits that were placed on the robot took up minimal space and allowed for a sense of freedom when placing components. The placement of the header pins was, however, overlooked which resulted in suboptimal cable management.

The key lesson learnt from the post-electrical design work is that one needs to be flexible enough to changes and be able to adapt and figure out alternatives if the recommended procedure does not give the desired results.

6 Commentary on the Systemic (Algorithmic) Design

The process of completing the algorithmic sub-system was underestimated in both time and difficulty. For these reasons, the process was incomplete. It is also noted that every time there was a small error in any of the other sub-systems, progress on the algorithms had to be halted and, in some instances, backtracked due to unforeseen failure in the sensors or mechanical design.

The debugging function that was supposed to be implemented gave an error continuously and, for this reason, finding errors was more challenging as the debugging feature was thus unusable.

Some confusion occurred when the robot was presented with error correction as well as a turn. Using interrupts for this caused a bug, so the error correction was not implemented using interrupts

In an attempt to keep the code as simple as possible, the robot lost some functionality and more errors occurred with the actual operation. It is also noted that some of the code would result in false triggers due to insufficient complexity.

No navigational flowcharts were created to assist with the logic. This made it more difficult to come up with the logic for various scenarios that the car would encounter and how to program around them.

Due to lack of time, the maze mapping and shortest path algorithm were unable to be integrated, even though they were ready. The final maze was then hardcoded, but due to rushing and lack of time, the robot would still not follow the path to the end of the maze.

7 CONCLUSION

Based on the project demo, it can be concluded that the robot was unable to perform as per the requirements. The robot was unable to track a black line correctly due to the error correction function being disabled so that the hardcode could work properly. The robot had to be adjusted by hand periodically so that it would remain on course. The robot was also unable to solve the maze and find the shortest path due to the time constraints. Instead, the known maze was hardcoded, but even with the hardcode, due to time constraints and an error in the code, the robot was still unable to follow the correct path to the end of the maze. The robot was, however, able to detect the final circle of the maze and blink an LED when placed on top of the circle. The detection of the circle was delayed, however. The robot was, however, able to proficiently execute left and right turns, while remaining on course.

Even though the mechanical and electrical sub-systems were complete and functioning correctly by the end of the project, due to poor time management and underestimation of the algorithmic sub-system, an incomplete project was shown for the demo.

The robot was not completed in full, however, many other valuable lessons were learned. The group worked efficiently and coherently as a team to produce a final product and, given an extra day, would have had a complete robot.

For future attempts, it would be vital to manage time efficiently and have ample time for the algorithmic sub-system designing and debugging. It would also be advisable to ensure that the individual parts in the mechanical and electrical sub-systems operate efficiently and, if not, then the design should be flexible enough to changes ensuring that not much time is wasted in debugging these sub-systems.

8 REFERENCES

[1] "RS | world-leading distributor of electronic, industrial and maintenance, repair and operation products," [Online]. Available: https://za.rs-online.com/web/.

[2] Saddam, "Arduino Line Follower Robot Code and Circuit Diagram," [Online]. Available: https://circuitdigest.com/microcontroller-projects/line-follower-robot-using-arduino.

[3] GeeksforGeeks, [Online]. Available: HTTPS://WWW.GEEKSFORGEEKS.ORG/DIJKSTRAS-SHORTEST-PATH-ALGORITHM-GREEDY-ALGO-7/.

[4] "IEEEXplore," [Online]. Available: HTTPS://IEEEXPLORE.IEEE.ORG/DOCUMENT/6997314.

[5] GeeksforGeeks, [Online]. Available: https://www.geeksforgeeks.org/mathematics-graph-theory-basics-set-1/

9 Appendix

9.1 Test Results:

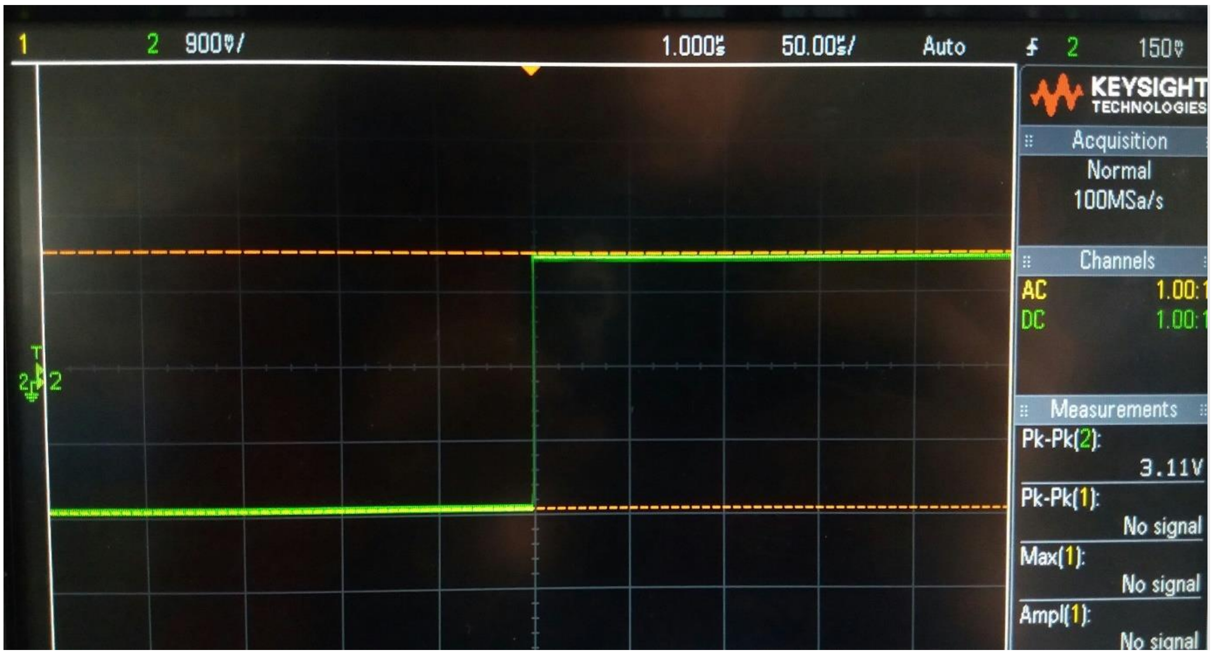

Figure 1: Sensor output reading of 3.1V when black line detected

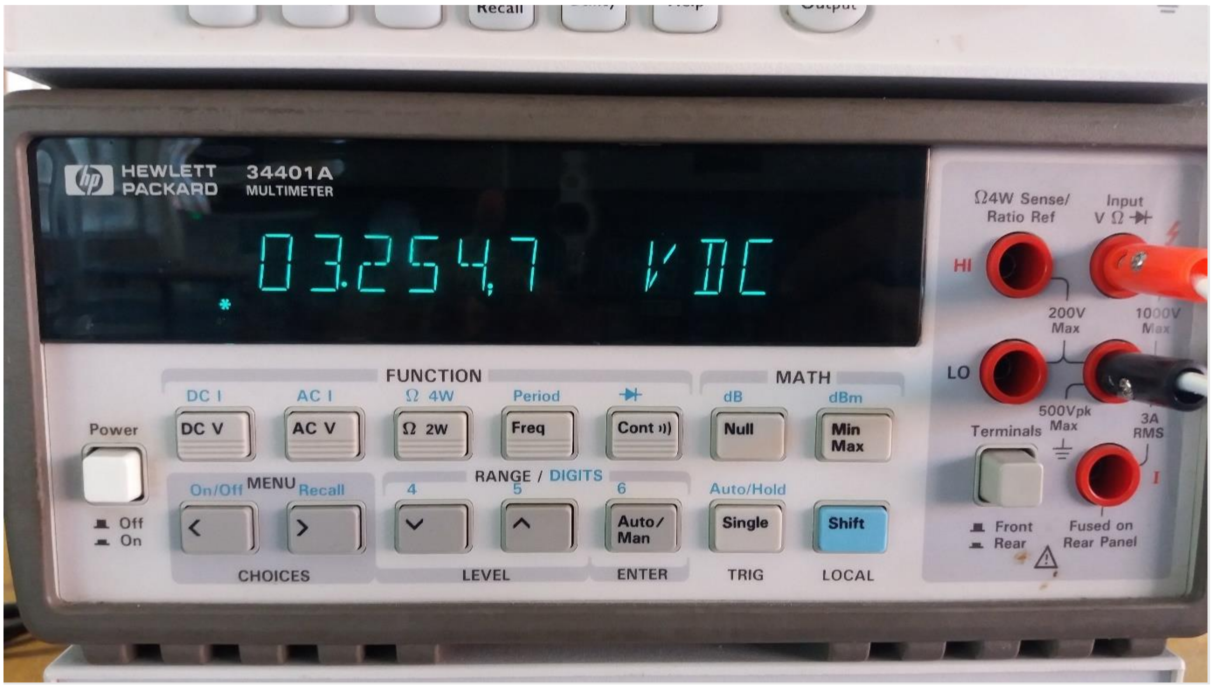

Figure 2: Regulated output from Voltage regulator (3.3V regulated)

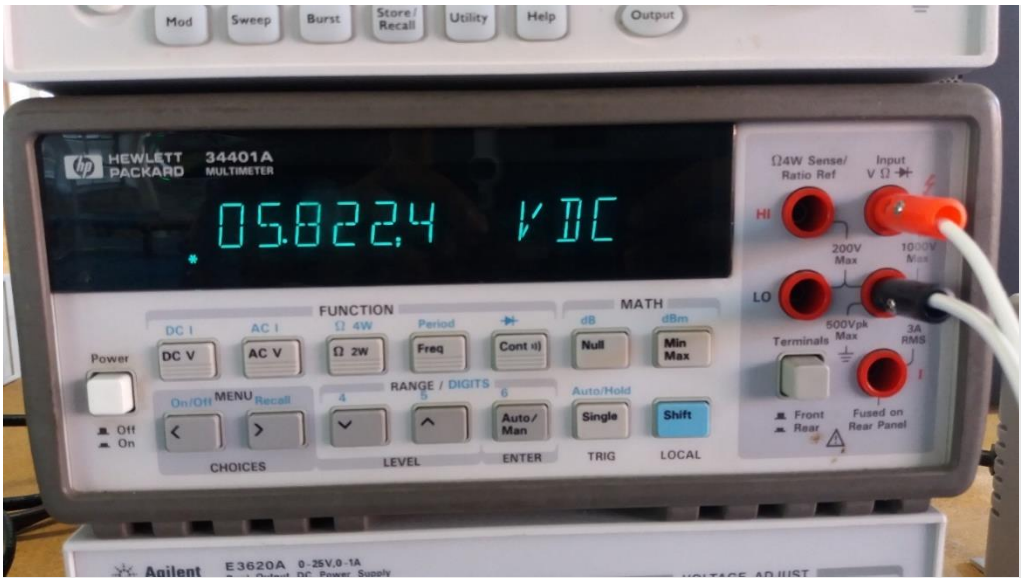

Figure 3: Regulated output from Voltage regulator (6V regulated)

9.2 Final mechanical drawings

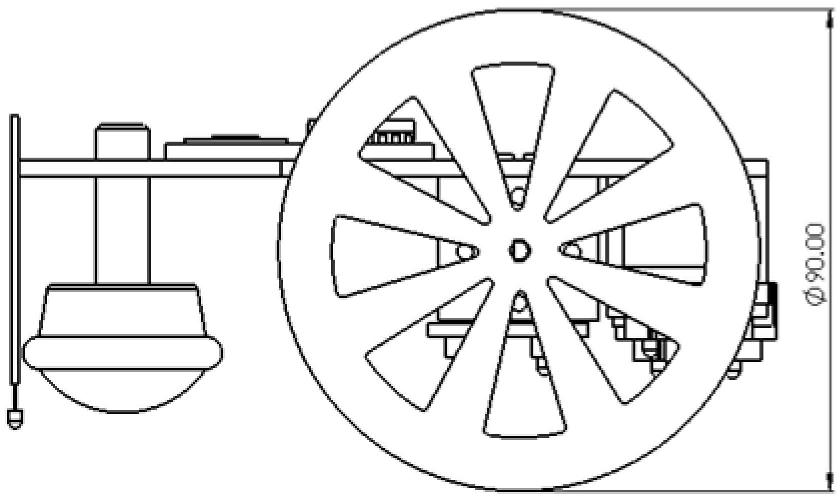

Figure 4: Side view of the robot, showing height

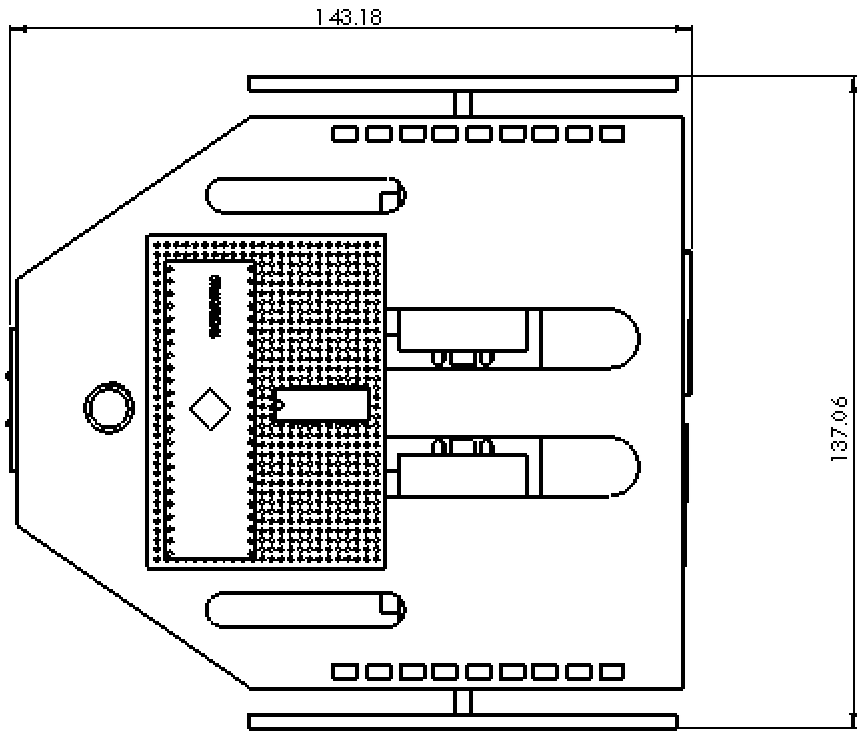

Figure 5: Top view of the robot, showing width and length

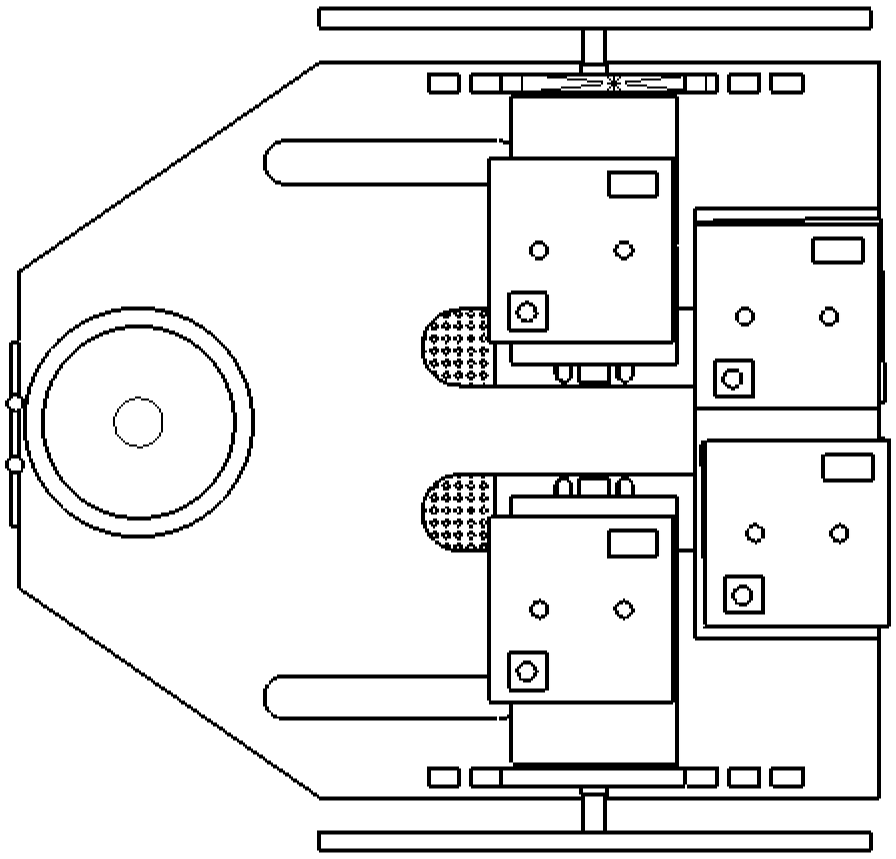

Figure 7: Bottom view of the robot

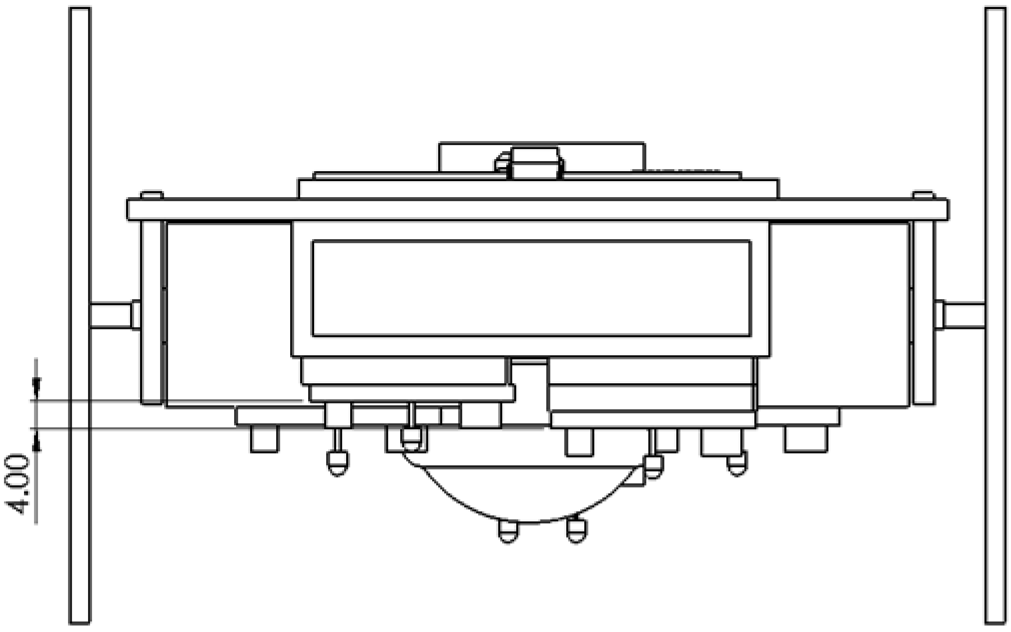

Figure 6: Rear view of the robot, showing height difference between the error correcting sensors

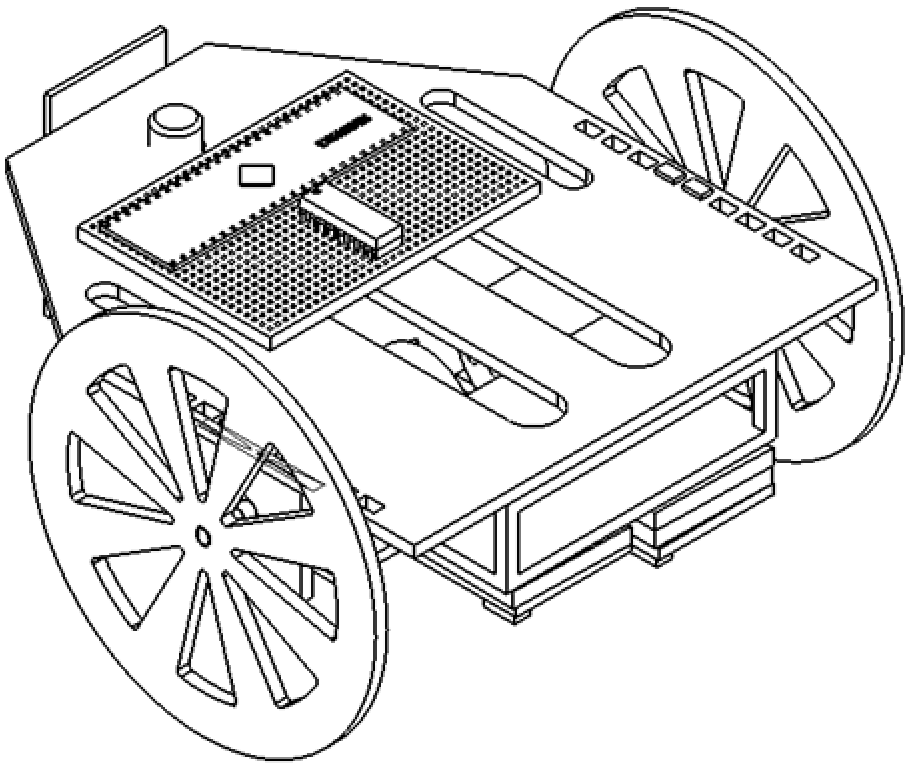

Figure 8: Isometric view of the robot from the rear side, showing that the error correcting sensors are unlevelled.

9.3 Schematics of final design

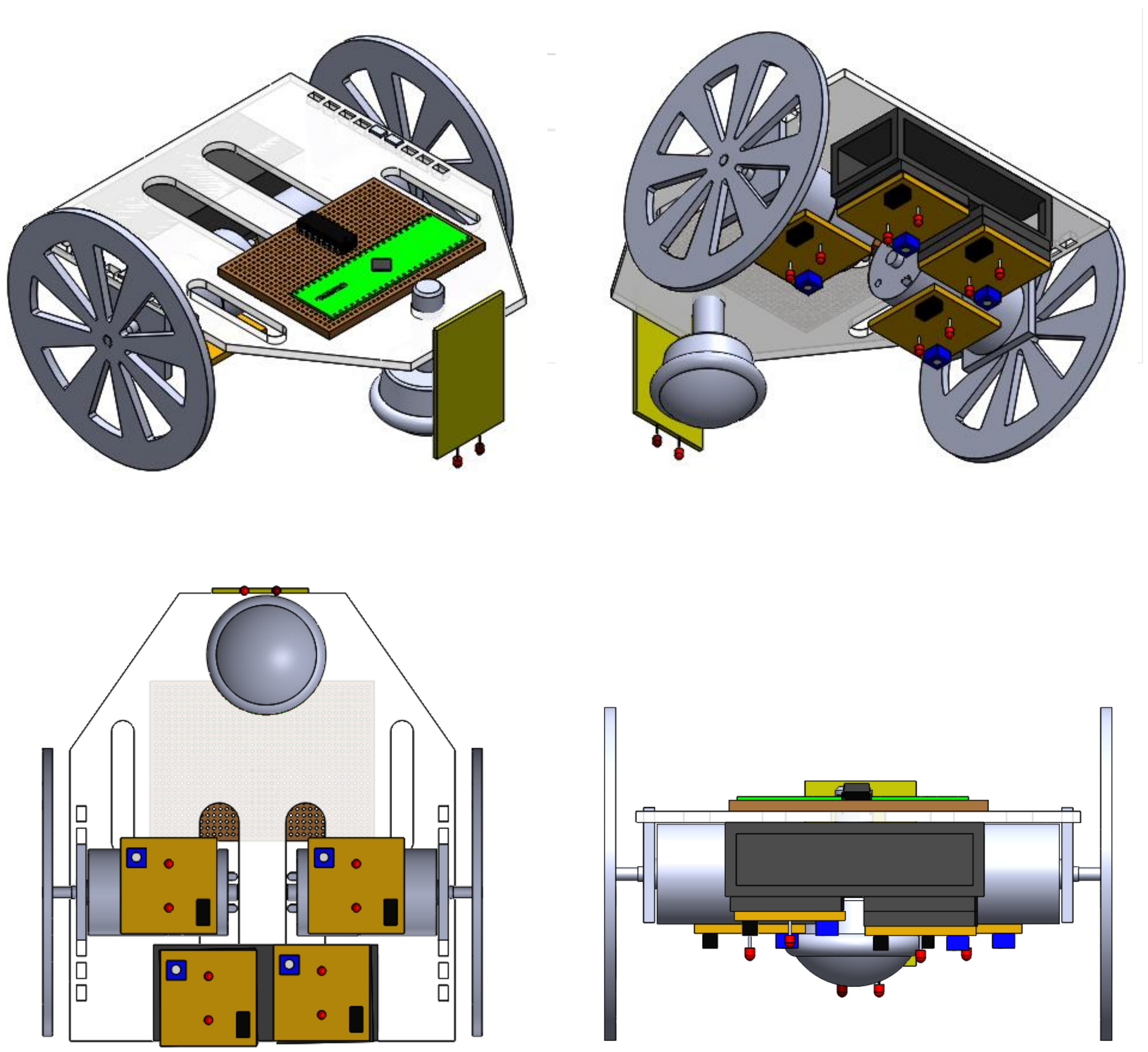

Figure 9: View of the robot from different angles

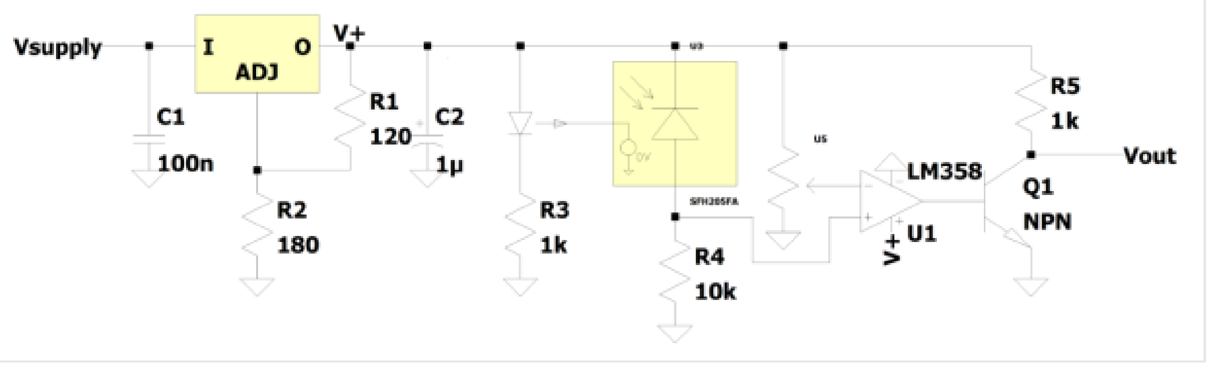

Figure 10: Final schematic of the sensor