Hardware Design Document

Project: DPM Final Design Project
Task: Describe the current and previous hardware implementations, as well as explaining important design decisions.

Document Version Number: 6.0
Date: November 28^{th} 2019
Author: Usama Barlas

Edit History:
Usama Barlas - 2019/10/25 - Initial formatting & layout
Usama Barlas - 2019/10/26 - Filled in Basics of Required Hardware Design
Usama Barlas - 2019/10/26 - Filled in Initial Design Process & Testing
Brendan Marks - 2019/10/26 - Filled in Alternative Design
Usama Barlas - 2019/10/27 - Filled in Alternative Design
Aurelia Haas - 2019/10/27 - Filled in Alternative Design
Usama Barlas - 2019/11/03 - Filled in Adopted Mechanical Design
Usama Barlas - 2019/11/08 - Filled in Adopted Mechanical Design *V2*
Usama Barlas - 2019/11/09 - Filled in Adopted Mechanical Design *V2*
Usama Barlas - 2019/11/10 - Filled in Adopted Mechanical Design *V2*
Pierre Robert-Michon - 2019/11/17 - Filled Post-Beta Design Evolution
Usama Barlas - 2019/11/23 - Filled in Final Hardware Design
Usama Barlas - 2019/11/26 - Filled in Final Hardware Design
Aurelia Haas - 2019/11/28 - Revision of the entire document

1.0 TABLE OF CONTENTS

2.0 BASICS OF REQUIRED HARDWARE DESIGN

The first steps in the basic hardware process is to create a mobile robot. This is done by creating a chassis, essentially mounting the EV3 brick onto this structure, adding a wheelbase and motors. Once this is done, some testing is carried out to make sure the robot's trajectory is straight indicating no faults or discrepancies in the motors' working. In the case where there are any, they must be noted and accounted for in software.

Next comes adding sensors which make the robot aware of its surroundings - essential for any autonomous vehicle. An ultrasonic sensor is added to the front of the robot. The ultrasonic sensor is a digital sensor that can measure the distance to an object in front of it. It does this by sending out high frequency sound waves and measuring how long it takes the sound to reflect back to the sensor. Distance to an object can be measured in either inches or centimetres. This allows us to program the robot to stop a certain distance from a wall or an object.

A light sensor, also known as a color sensor, is also attached to the robot. This sensor can detect the colour or intensity of any light that passes through its lens. This sensor can be used in three different modes: Color Mode, Reflected Light Intensity Mode, and Ambient Light Intensity Mode. In Color Mode, the color sensor recognizes seven colours – black, blue, green, yellow, red, white, and brown – plus no colour. This ability to differentiate between colours means the robot can be programmed to navigate a course with some sort of markings. For example, the robot can be programmed to move around on a white surface until a black line is detected. The sample rate of the color sensor is 1 kHz. The light sensor is mounted to the back of the robot, as shown in Fig 2.0.1. This is so that as the robot follows its course, the robot can identify the markings and calibrate its odometer, and correct its angles. This is needed in order for it to navigate without errors. Furthermore, it needs to be placed away from the center of rotation, so that it can detect the grid line positions relative to the robot.

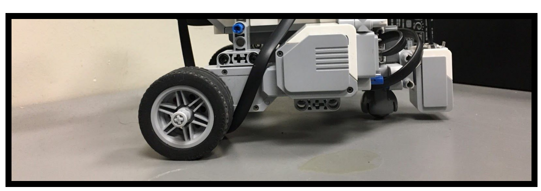

Fig 2.0.1

3.0 INITIAL DESIGN PROCESS AND TESTING

For assembling the first and most basic version of a projectile launcher, we had multiple design alterations. At first, our launcher arm was not long enough and so the ping pong ball was not launched to a sufficient distance. We also noticed the ping pong ball was not stable at rest so we had to make a cradle-like holder to make sure it remained in place. For our first hardware stage, shown in Fig 3.0.1, we realized after multiple launches that the overall design was not stable. The robot would flinch significantly after each launch due to the force. Moreover, the launch mechanism (including the motor) were placed slightly off centre from the base of the robot. The second stage, shown in Fig 3.0.2, was modified so that the entire structure was more stable, and this was done by strengthening the base by adding more legos connected to the bottom portion of the robot, and also moving the mechanism closer to the centre of gravity. This provided it with more stability.

Once again after multiple tests, we found this design to be inefficient in some ways. There were 2 reasons for this. Firstly, our launcher cradle would make the ball launch at a low angle due to its upward angled nature (as seen in Fig 3.0.2). To fix this, we simply made a launchpad that was parallel to the arm, but added legos ahead and behind the pad so that the ball could be held in place until launched. This allowed it a more free angle. The second flaw we realized was that there was simply not enough power in one motor to launch it to the required distance. For this, we installed a second motor and attached the arm to both of these. To provide the launch with more agility, we also attached some rubber bands and this provided it some more edge in terms of acceleration. These changes brought us to another design stage, shown in Fig 3.0.3.

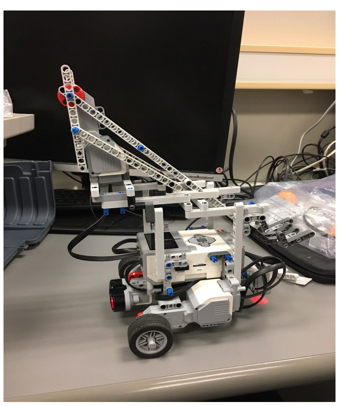

Fig 3.0.1 Left view

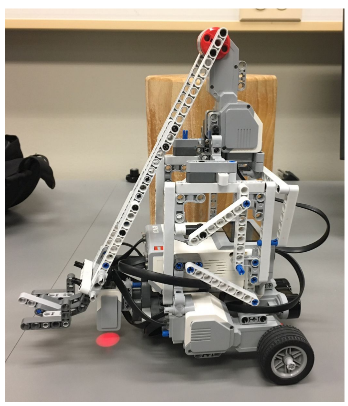

Fig 3.0.2 Right view

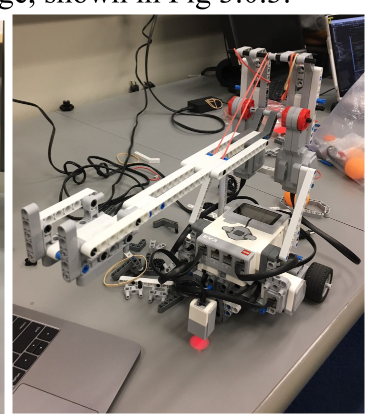

Fig 3.0.3 Back view

After these stages, we will now adopt one of the three mechanical design proposals as advised, will carry out the testing phase and evaluate which one of those suits the best.

4.0 MECHANICAL DESIGN ALTERNATIVES

4.1 DESIGN #1 - ARM LAUNCHER

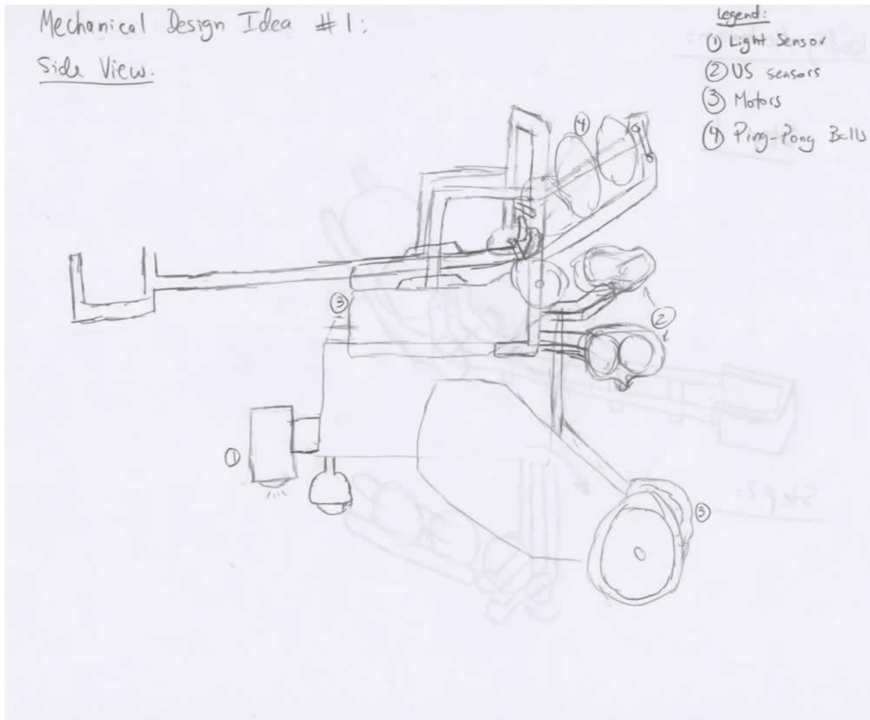

Figure 4.1.1: Alternative Design #1 Overview

The first alternative design utilizes a light sensor placed on its rear end for usage in localization and odometry correction. It also has two US sensors placed at 45 degree angles off the front of it for dual sided obstacle detection. The design uses 2 motors to pull itself, and another two to power the launching mechanism. All motors are placed horizontally to ensure the robot maintains a low profile.

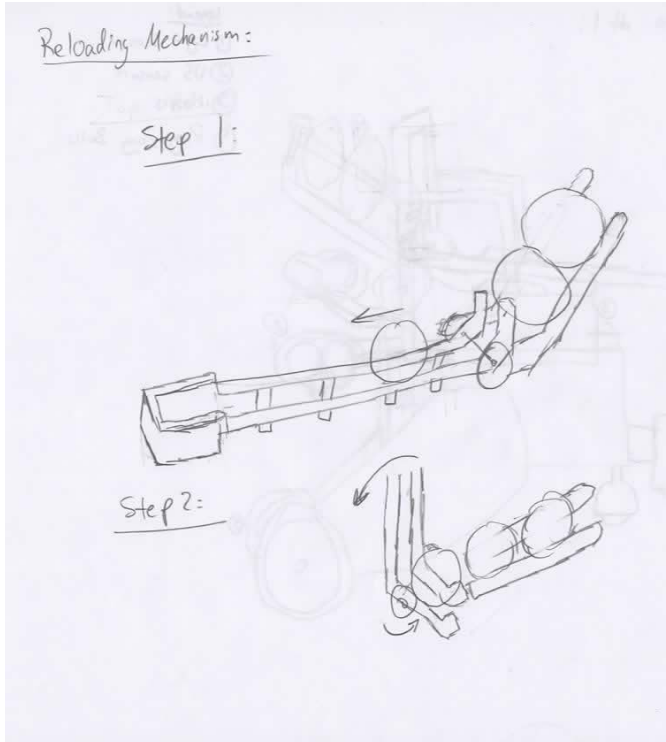

Figure 4.1.2: Alternative Design #1 Reloading Mechanism

This design also features a reloading mechanism that uses the launcher's motion to mechanically pick up another ball at the end of each launch. This is achieved by placing the balls on a ramp and utilizing an angled claw at the beginning of the launch arm. This way, as a ball is launched, the claw's hook passes below the rails allowing a ball to fall into its grasp, and then when it returns to the ready-to-launch position, the claw blocks further balls from falling towards the basket. This ball then slides along the arm until it finally falls into the basket, where it is then ready to be launched.

Pros:

- Low-profile design to easily pass through tunnels.
- Reloading mechanism that doesn't require a motor.
- Multiple US sensors to allow for better obstacle avoidance.

Cons:

- Requires software changes to utilize the extra sensors which weren't present in previous software designs.
- May be difficult to achieve desired launch range due to low height of launch.

4.2 DESIGN #2 - SPRING LAUNCHER

The second alternative design utilizes the same base chassis and sensor placements as the first one. The design uses a motor attached to spring to essentially ‘load’ the launcher as the spring compresses and stores energy. When released, it launches the ball.

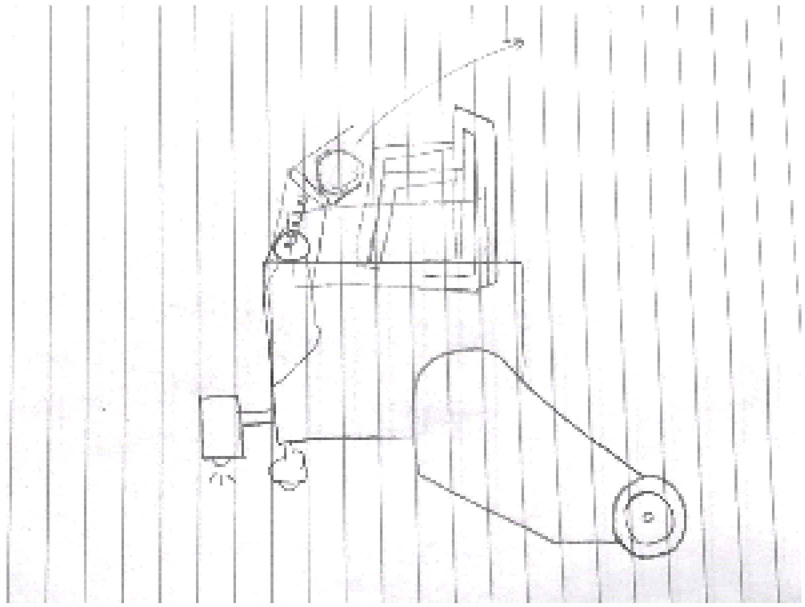

Figure 4.2.1: Alternative Design #2 Overview

Pros:

- Low-profile design to easily pass through tunnels.
- Does not have a long protruding arm and therefore less likely to hit surrounding obstacles
- May be able to launch a ball further due to stored elastic potential energy

Cons:

- May be difficult to design a loading mechanism, which may require a motor
- External part (spring) must be acquired
- Spring may wear out/lose its elasticity over time

4.3 DESIGN #3 - WHEELS LAUNCHER

The third alternative utilizes the same base chassis and sensor placements as the previous ones. However, here the launcher is different. Here, the way the ball is launched is by using 2 wheels connected to 2 motors. While rotating at a certain speed, the wheels will give strength to the ball to reach its destination. This design is also composed by a reloading design.

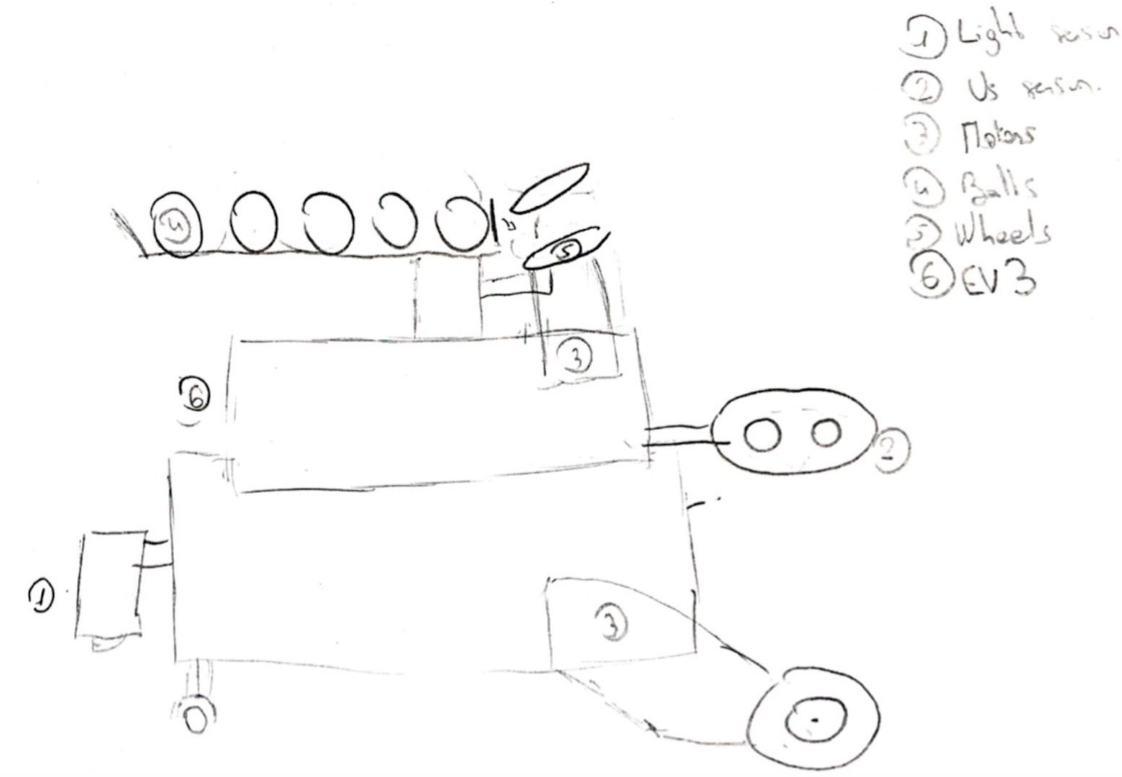

Figure 4.3.1: Alternative Design #3 Overview

Pros:

- Low-profile design to easily pass through tunnels.
- Does not have a long protruding arm and therefore less likely to hit surrounding obstacles
- Loading system integrated

Cons:

- Not necessarily precise
- May not launch the ball as far as asked
- May be complicated to calculate the right angle for both wheels

5.0 METHODOLOGIES

Create an initial robot design, trying to utilize the tools as well as possible.
- Evaluate our design for its capability to meet the requirements adequately and efficiently.
- Try to find possible faults in our initial design for future improvements.
- Design tests to ensure each individual part of the design will be able to accomplish its objective.

6.0 TOOLS

For the construction of the mechanical system, the tools available include 3 full Lejos EV3 building kits. This means that there could be up to 3 EV3 controllers at our disposal for the final project. Some external tools that could be useful to expand our knowledge would be the TA assigned to our group, as well as the professors who could answer any questions we may have. There is also documentation about the Lejos EV3 functionalities online, among many other potentially useful resources.

7.0 ADOPTED MECHANICAL DESIGN

7.1 EVOLUTION OF DESIGN, JUSTIFICATION, FEATURES

After considering the constraints and the engineering challenge we had to tackle, our team initially thought it was best to *not* further use the Arm Launcher mechanism and pursue another design whereby a motor attached to the base would retract a rubber band, which would build tension, and upon release, this would simply push or nudge the ball launching it to the desired position. This was similar to Design #2 as shown in the *Mechanical Design Alternatives* section. Figure 7.1.1 shows this design currently in its initial stages. We quickly realized this design would not be ideal, since with only the motor attached, we had very little height clearance left, and so the elastic mechanism would cause it to be higher than the tunnels.

There are multiple ways to tackle the problem of the height constraint of the robot. One such solution was to place the brick vertically, freeing up space for the motor to be placed closer to the base and so lower. Another solution, which we implemented, was to move the motor structure further backward, so that it could be placed lower (closer to the level of the brick itself). This meant we had to create a much more solid structure to support the motor. This is shown in Figure 7.1.2 and Figure 7.13. When tested, we observed the structure to be slightly flimsy, and so further reinforced it by adding legos from the bottom of the structure to the base of the brick.

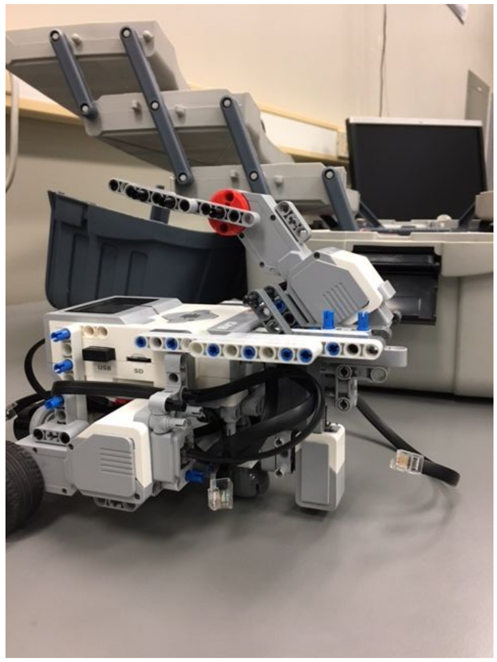

Figure 7.1.1

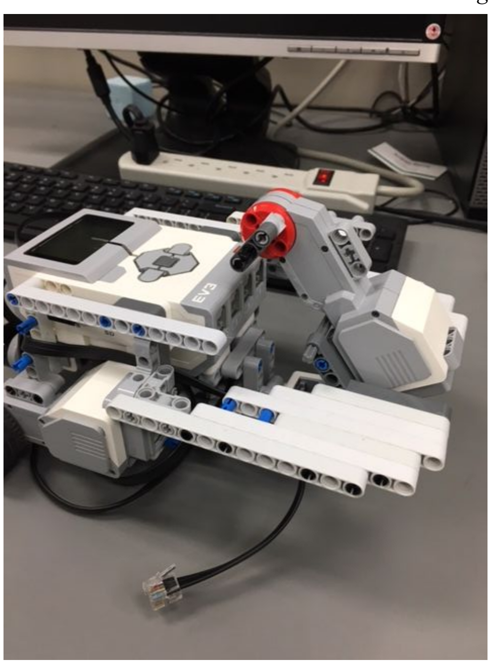

Figure 7.1.2

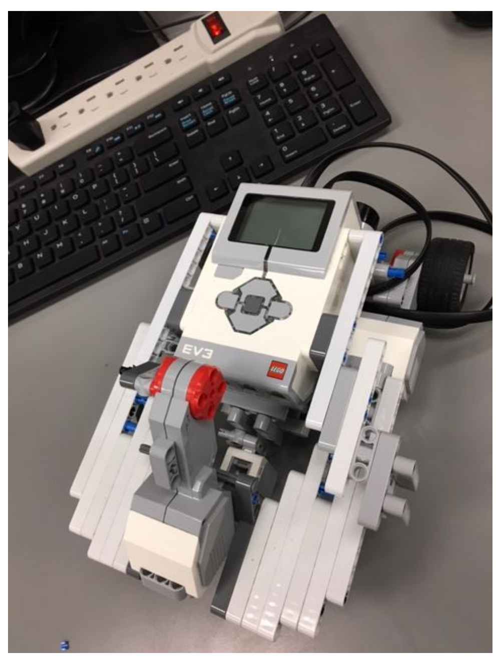

Figure 7.1.3

After some discussions within our group regarding how we would proceed to execute the elastic mechanism launcher, we realized it would be much more complicated than initially thought, to simply make the mechanism extremely consistent and solid. This was because for it to work, lego pieces should essentially be able to pivot using the elastic rubber band, which would make for a very unstable structure.

At this stage, we decided to work towards the arm launcher, and modify our existing design to achieve the required launch distance.

Previously, our arm launcher design had the motors placed on top of the brick. Once again, this would not be able to go through the tunnel and so the total height of the robot had to be modified. For this, we placed the motors adjacent to the brick's sides, as shown in Figure 7.1.4 and 7.1.5.

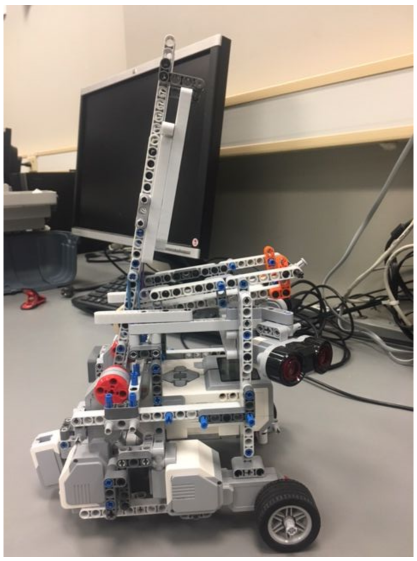

Figure 7.1.4

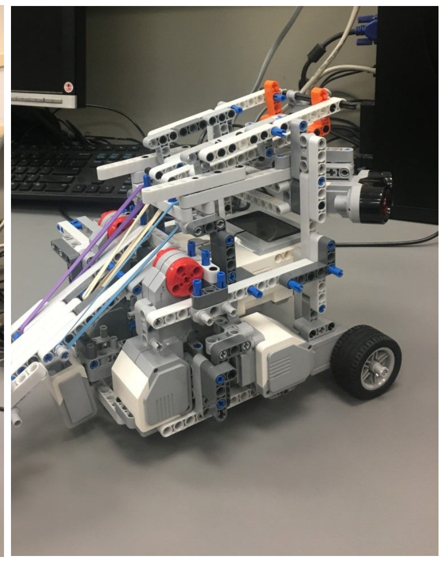

Figure 7.1.5

Because the motors are now further from the centre of gravity of the robot, this meant it had to be extremely stable, and so we strengthened it by connecting the rear portion of the robot to the brick and the front.

For this challenge, we also require obstacle avoidance & efficient navigation and so as opposed to our previous design iterations, we would need two ultrasonic sensors instead of one. These were faced at 45° angles to be able to cover the maximum field of view. This is shown in Figure 7.1.6 below.

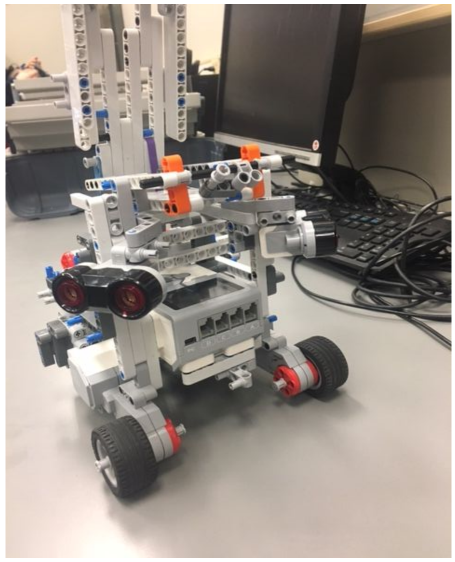

Figure 7.1.6

Another design feature we must implement is a reloading mechanism for the ping pong balls. The robot needs to be able to autonomously load balls into its launching mechanism without human interference. This is extremely tricky because the launcher must not be disturbed with this mechanism in place. For the base of our loading mechanism, we implemented one design shown in Figure 7.1.7. This design does *not* make use of a motor. It uses the launcher's motion to autonomously drop ping pong balls into the launcher using the elastic mechanism.

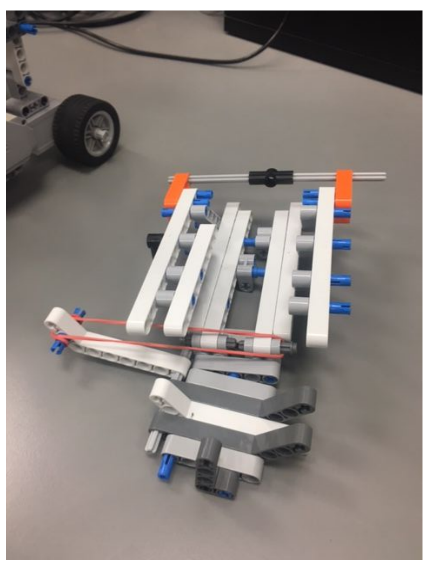

Figure 7.1.7

This mechanism proved to be difficult to implement without a motor because there is no restriction to the ping pong balls from entering the launcher. That is, there needs to be some way to allow the ping pong balls one by one for each launch, and not all at once. For this, we will have to make use of a motor, namely the **EV3 Medium Motor**, which makes use of a rotation mechanism.

Our currently adopted mechanical design is shown in Figure 7.1.8, which features a launcher and a reloading mechanism. This design is similar to the one proposed in Design #1 shown in the *Mechanical Design Alternatives* section. The loading mechanism is currently placed towards the front of the robot, and the entire structure is within the height constraints.

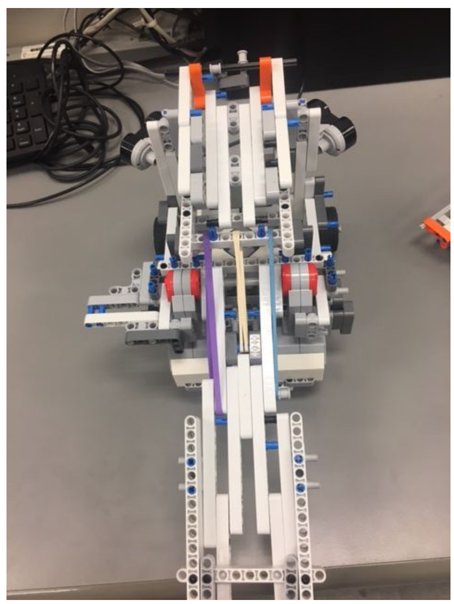

Figure 7.1.8

8.0 ADOPTED MECHANICAL DESIGN V2

8.1 EVOLUTION OF DESIGN, JUSTIFICATION, FEATURES

After the completion of our aforementioned hardware design, we proceeded to carry out some testing, but also attempted to complete the reloading mechanism design. We found that with that design, we would not be able to use the Medium Motor unless we used only one motor for the arm launcher. This would make the launches much weaker, and so we decided to try to implement a different type of reloading mechanism. However after multiple attempts and arriving to impractical results, we decided to not proceed with this design.

For these reasons, and for the information we gathered in the design stages, we decided to modify our design to an **elastic launcher,** which is now our **Adopted Mechanical Design (V2)**. As mentioned before, we initially thought it would be difficult to make this very design stable, but through some processes, we achieved that goal.

The working of this design uses the strength of the motor combined with elastic energy, to "launch" a lego piece upwards which ends up hitting the ball, with much greater launch force one would be able to harness from a standard arm launcher. A view of the design is shown in Figure 8.1.1. As we see, there are two elastic rubber bands building up tension as the lego piece is pulled downwards by **two** motors. When the pivot reaches its maximum rotational point, it automatically releases, which causes the rubber bands to release, and the piece to move upwards rapidly. The motors keep running forward, and we do not keep them in float. This causes even more force to be applied.

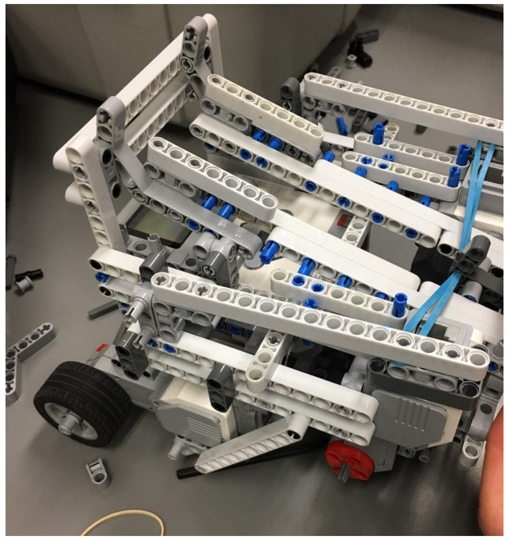

Figure 8.1.1

One issue we quickly faced was crumpling of the structure as the piece was pulled down. Since the piece is held by bands which are connected to the sides of the robot, the base would essentially be pushed down which made the whole robot quite unstable. To fix this issue, we tried to force the motors to try to remain in the same position, which held the rest of the structure together. We did this by restricting their motion, by connecting the top and sides of the motors to the front end of the robot. This is shown in Figures 8.1.2 and 8.1.3.

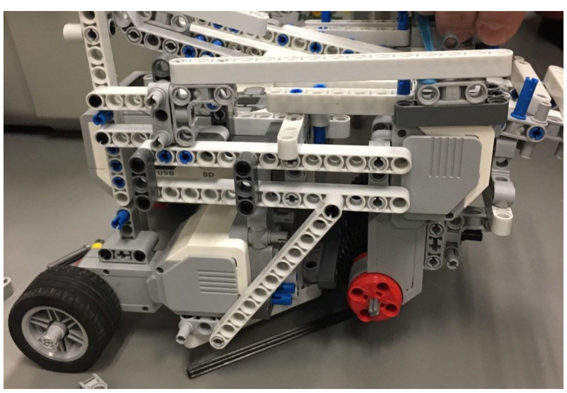

Figure 8.1.2

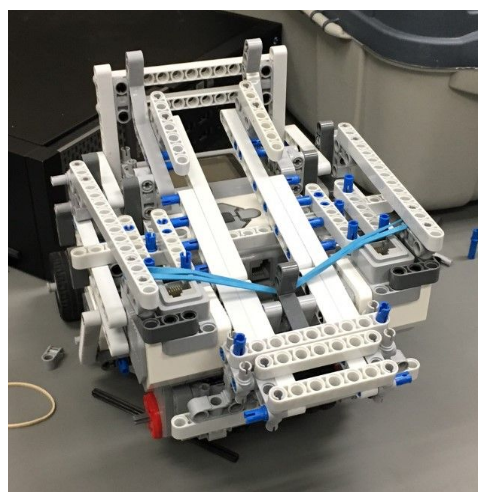

Figure 8.1.3

Furthermore, it is evident now that the centre of mass of the robot is towards the back end. This meant it was extremely sensitive to movements in terms of toppling over to one side. At this stage, we had only one marble roller on the robot, which was placed close to the center as shown in Figure 8.1.4. This needed to be improved, so we modified

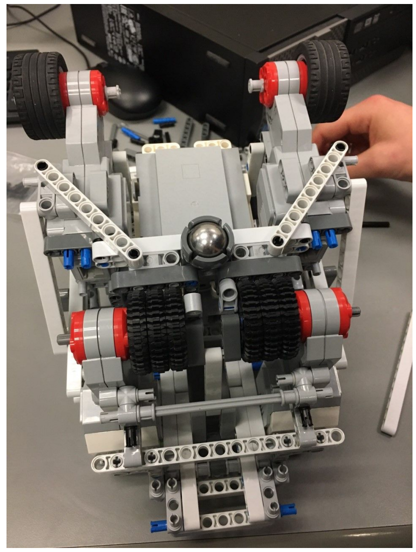

Figure 8.1.4

the base as shown in Figure 8.1.5. Here we make use of 2 marble rollers placed towards the rear. This makes the structure much more stable as the weight is now distributed to 4 points.

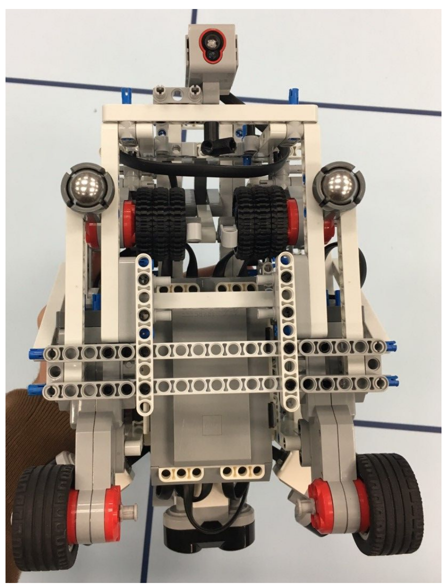

Figure 8.1.5

To depict the **reloading mechanism**, I have included Figures 8.1.6 and 8.1.7. This is a very efficient loading mechanism because it uses the principle of gravity rather than any sort of automated or motorized mechanism. The 'rack' which holds the balls is at an inclined enough for the balls to move downwards by themselves. As one ball is launched, the next one simply falls into place. A key detail is to allow enough time for the ball to fully settle before it is launched.

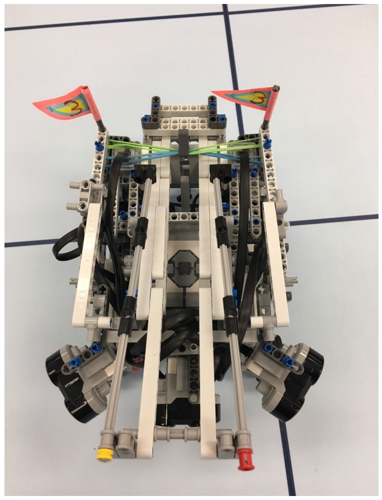

Figure 8.1.6

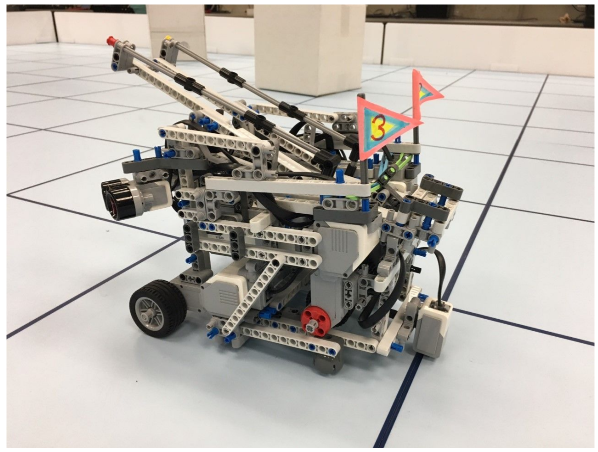

Figure 8.1.7

Another feature of this design is that the rails allow the balls to stay put, which secures them in situations where they may experience force. This could occur when the robot is turning or executes a launch. With these rails, the balls do not slide out of position.

Our robot is making use of 4 sensors in total. 3 ultrasonic sensors are placed facing to the left, straight and to the right. These are placed at the front end of the robot as shown in Figure 8.1.8. The fourth sensor is the light sensor placed at the back end of the robot, far from the centre of the base, for localization. This is shown in Figure 8.1.9.

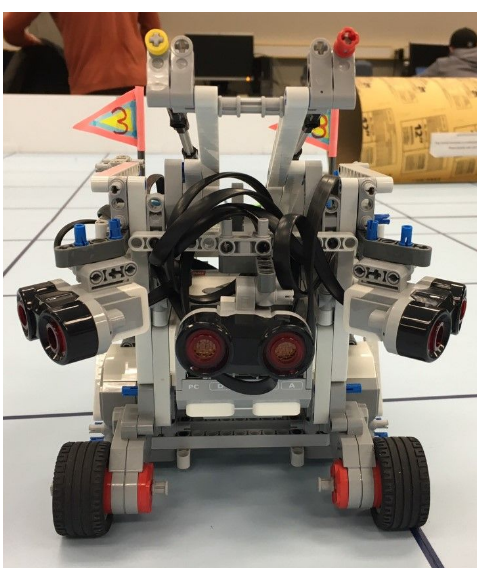

Figure 8.1.8

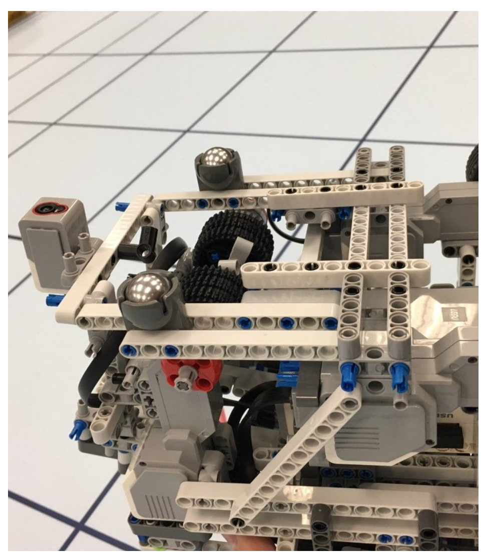

Figure 8.1.9

This section contained the evolution and features of our design, and the justification of our decisions in reaching to our adopted mechanical design for this project.

8.2 EVOLUTION OF DESIGN FOLLOWING BETA DEMO

The beta demo highlighted a few flaws in the hardware design. The primary flaw is the use of only one light sensor on the back of the robot. This is a problem for a few reasons. First of all, it makes the robot's form factor very long, which makes it difficult to localize and manoeuvre in tight spaces. Additionally, having only one light sensor is risky as we have no backup in case we do not detect a line. Finally, it is very restricting for the software, because it makes it a lot more difficult and imprecise for implementing odometry correction. This was our main problem for the Beta Demo, which we performed without odometry correction, ultimately leading up to errors in our displacements and heading, which made us miss the tunnel entry by a couple degrees.

The decision to only use one light sensor was motivated by the hardware constraint of having only 4 sensor ports on the EV3 brick. We wanted to have the widest possible field of view without having to implement a rotating motor on the US sensor, so we chose to add three sensors pointed -45, 0 and 45 degrees to the front of the robot. This meant that we only had one port left for the light sensors. However, moving forward from the Beta Demo, we have deemed it more important to implement odometry correction, which means we have added a light sensor on each lateral side of the motor, left and right next to the marbles (as can be seen in Figure 8.2.1 and Figure 8.2.2), and have removed the back light sensor (shown in Figure 8.2.1) along with two of the front US sensors (leaving the center one, shown in Figure 8.2.3). These modifications will hopefully allow us to be more precise in our displacements and heading calculations.

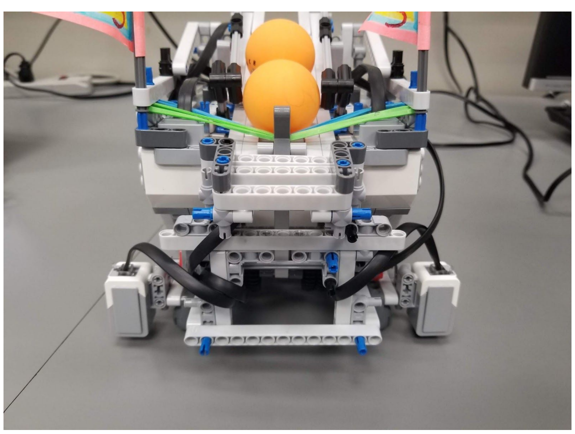

Figure 8.2.1

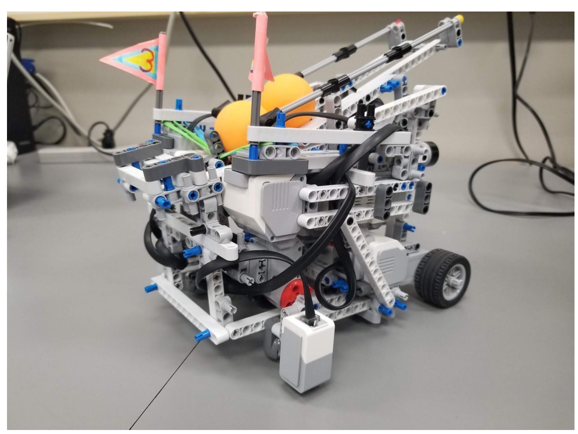

Figure 8.2.2

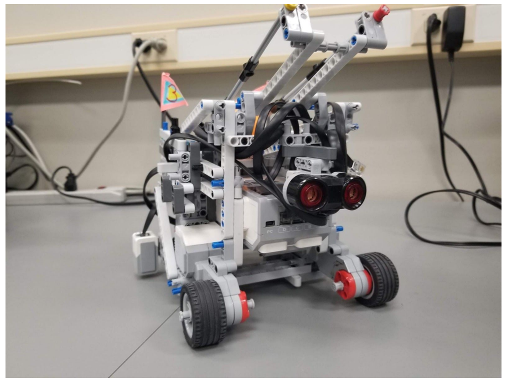

Figure 8.2.3

9.0 FINAL HARDWARE DESIGN

This section provides an overview and recap of our Final Hardware Design for this project.

After a multitude of design alternatives brought by different decisions as shown in this document, and a great amount of testing (outlined in the Testing Document), we arrived to our Final Hardware Design shown in Figure 9.0.1. The LEGO® Digital Designer model is shown in Figure 9.0.2.

Figure 9.0.1

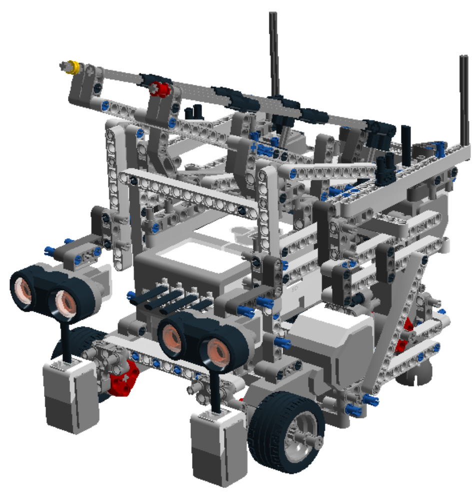

Figure 9.0.2

Our robot features 4 Large Motors (2 for driving the robot, and 2 for the launch mechanism), 2 Light Sensors (for navigation & odometry correction), and 2 Ultrasonic Sensor (for localization & obstacle avoidance). As shown and explained in section 8.1, we also make use of a reloading mechanism, and also a launch mechanism. The only extra material we used include 2 rubber bands, which provide elastic energy for a powerful launch (explained later).

Figure 9.0.3 shows a visual of our reloading mechanism.

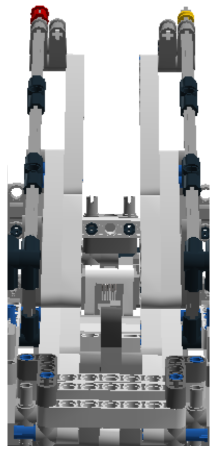

Figure 9.0.3

This mechanism uses the principle of gravity simply to load balls one by one, as each one is launched. We chose this design as it does not require any sort of motors, or a mechanism whereby the launch mechanism itself 'activates' the reloading mechanism, which seemed to be quite common. We realized that could be unreliable at times. As seen, it makes use of two side rails to secure the balls in place. An important factor is for the balls to be able to settle before each launch.

Figure 9.0.4 shows a visual of our launching mechanism.

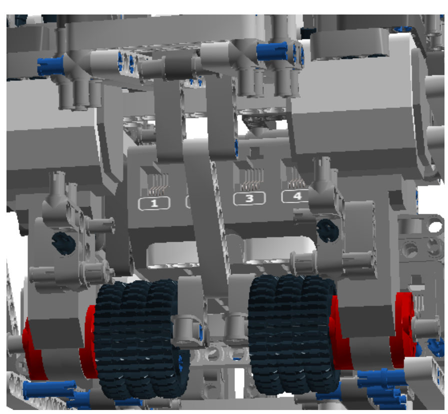

Figure 9.0.4

The working of this mechanism uses the strength motors combined with elastic energy, to "launch" a lego piece upwards which ends up hitting the ball, with much greater launch force one would be able to harness from a standard arm launcher. There are two elastic rubber bands building up tension as the lego piece is pulled downwards by two motors. When the pivot reaches its maximum rotational point, it automatically releases, which causes the rubber bands to release, and the piece to move upwards rapidly. The motors keep running forward, and we do not keep them in float. This causes even more force to be applied. The rubber bands may lose their elasticity over time, and so need to be replaced after a while.

Figure 9.0.4 shows a visual of the robot's side view. The purpose of this is to illustrate the measures taken to increase stability & robustness.

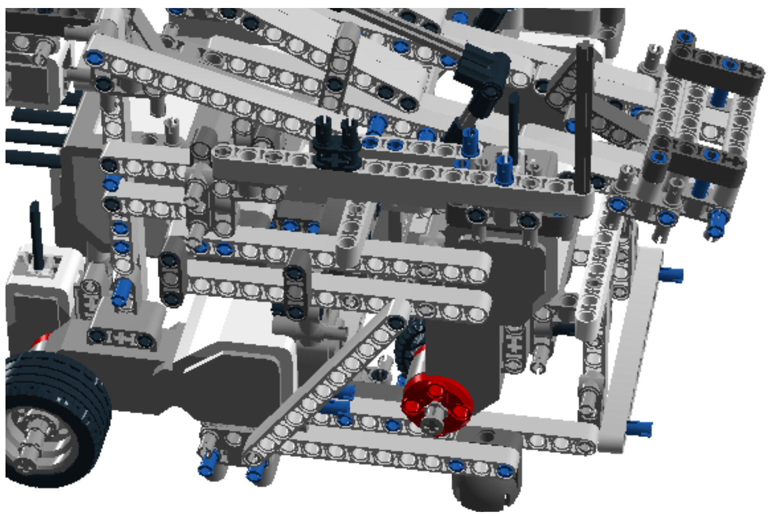

Figure 9.0.4

The motors add a significant amount of weight to the back. To hold up this weight, and for it to be stable, it is not sufficient to simply attach the motors to a structure in the rear. As seen in the Figure, there are multiple parts connecting the rear to the front, and this ensures the motors stay in place. They are also held in place from the bottom.

Figure 9.0.5 shows a visual of the robot's front view. The purpose of this is to illustrate the sensors & their placements.

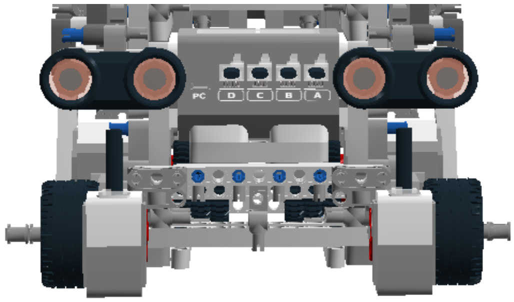

Figure 9.0.5

The light sensors are used for odometry correction which is critical for an accurate trajectory. We have opted to use 2 ultrasonic sensors, so that obstacle avoidance is efficient.

As shown in Figure 9.0.6, the placement of the lights sensors is rather ahead of the motors and not between them. This is so that as the robot traverses its course, it does not make false odometry corrections. That is, the placement is wide enough so that lines are able to be detected consistently.

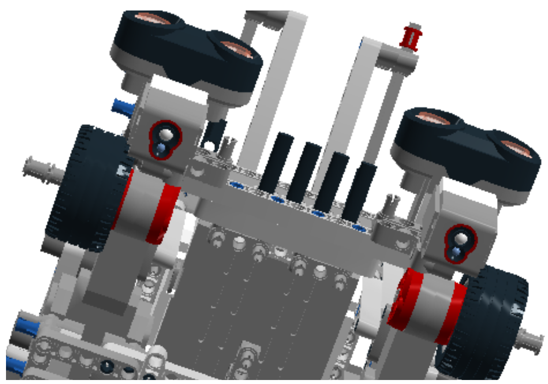

Figure 9.0.6

10.0 GLOSSARY OF TERMS

All terms used should be easily understood by anyone with a basis in electrical, computer or software engineering, so no definitions will be needed.