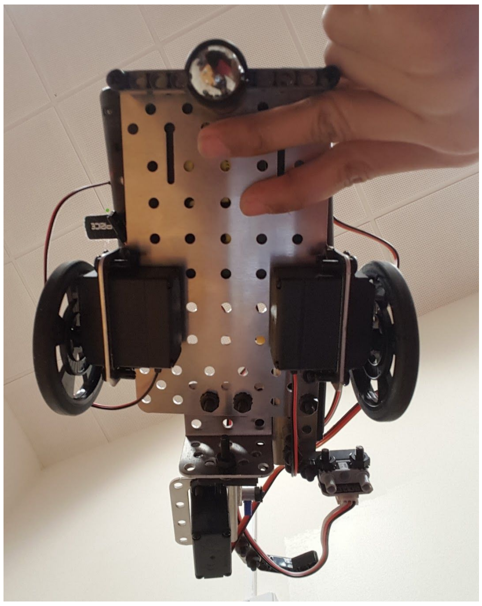
Thermonuclear Falcons
18-0288

Mechanical Design Period 2

Drivetrain



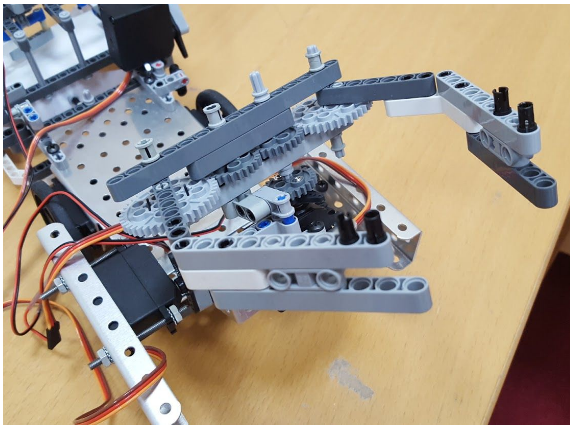
We are making use of a simple drive-direct system. The motors are directly attached to the wheels and are mounted under the chassis. A ball-bearing is mounted on the other side of the chassis, which acts like a counterweight to keep the robot in balance; this creates smoother movement by ensuring the wheels move flat across the board and the chassis is parallel to the surface of the gameboard.

Another drivetrain we could have used is a mecanum drivetrain, which is agile and fairly easy to design and build. Rollers are attached to the circumference, allowing omni-directional movement. However, it does not have enough potential for a high pushing force, is challenging to program, and requires extra gearboxes. The wheels are also expensive and are not part of the Botball kit; we could have made our own mecanum wheels, but we decided that none of the pieces would be suitable as rollers. Hence, we decided to make use of a drive-direct system since we have all the proper materials for it.

A drive-direct system allows our robot to swivel easily and required less materials than to build a mecanum drivetrain. The drivetrain we are using also has a simpler design and was easier to build.

In addition, we have coded a digital compass to assist with the orientation and direction of our robot on the gameboard, which is connected to the movement of the wheels.

Effector



The effector that we are using on Demobot is a claw that collects the five groups of blue poms situated on the blue line, along with stowing them in the water reclamation units (3 inch couplers). A servo acts like a pivot at the base of the claw, allowing the claw to tilt down to pick up the poms and then tilt up and adjust its angle to put the poms in the couplers. Additionally, a gear system that is attached to a servo in the center of the claw allows the pincers to open and close to pick up the poms. We decided a gear system would be the most optimal way to determine the movement of the pincers because of its transmission of torque energy and its ability to control the speed of the movement of the pincers.

There were more possibilities for the effector, however. We could have used the IGUS chain to make a loop that can expand and shrink around the poms. The loop would have to have a tight squeeze around the blue poms, be lifted up and drop the poms into the couplers. We decided not to use this mechanism because firstly, there would be a high chance of the poms falling out of the loop since there is nothing to support the poms at the bottom when the chain loop is being lifted up. Secondly, the loop's accuracy of capturing the poms might be off as a result of it being flimsy; it might not accurately place the poms in the coupler either. Thirdly, lifting the chain and getting it to expand and constrict would require a very complex design, which is more difficult to program and calibrate.

Meanwhile, our claw has a fairly simple design that is easy to build and program. It is also an efficient and effective way of transporting the poms and makes good use of the servos and gears in our kit. Hence, these were the reasons we chose a claw over a grabbing loop mechanism.

Sensor Mount

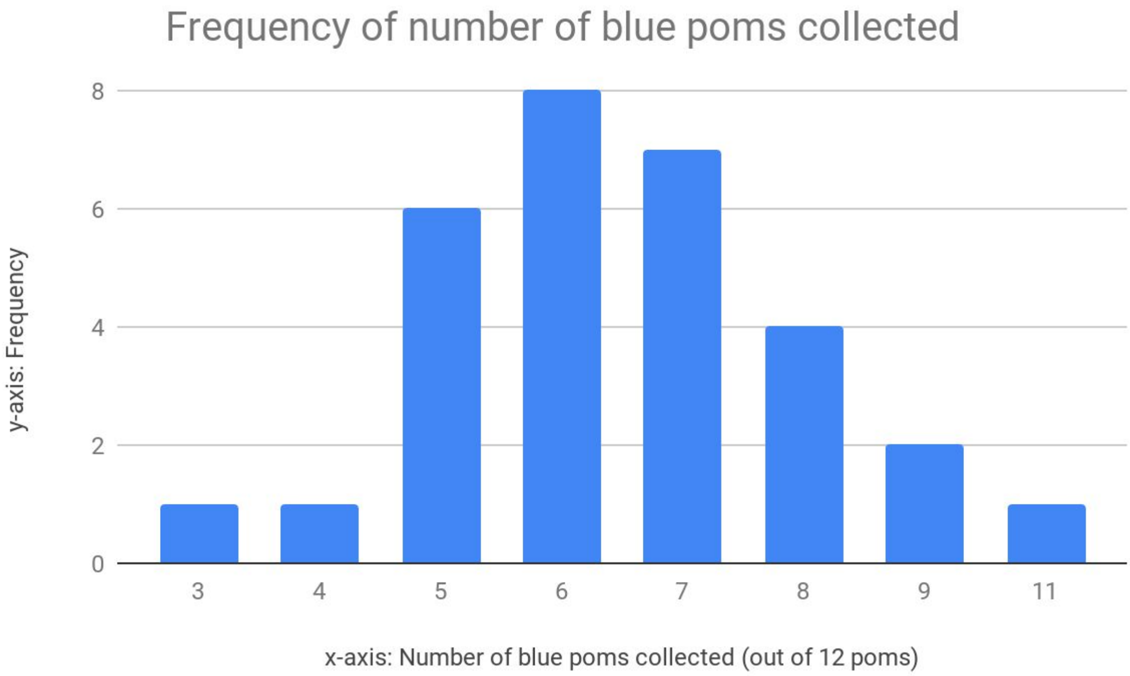
The sensor has a shielding so that extra light does not affect the sensor when using wait_for_light. The sensor is connected so that it can swivel in three mutually perpendicular directions. This will allow the robot to sense from any orientation.

Another possible design was mounting the sensor to a metal piece, which would have kept it in place as opposed to having a swivelling mechanism. Mounting the sensor in one orientation would keep it solidly in place. Yet, we decided to use Lego, as it is more customizable and simpler to use. Furthermore, the Lego allowed us to create an omnidirectional system, which is better than having the sensor fixed in one place because it allows the robot to sense from any direction and have better functionality for how it is mounted on the robot.

As mentioned before, we are also using a digital compass to sense direction. The compass is embedded in the code, so it is not externally visible. Yet, it is still a sensor that is mounted *inside* the robot which detects and fixes errors in the robot's orientation on the board. By doing so, it ensures the robot drives in a straight and smooth line, has accurate turns and can be easily corrected regarding its position if it has a skewed angle.

Data

Graph



Data Collection
The programmers wrote the code for the claw mechanism on the Demobot that collects the blue poms and sorts them in the couplers. The blue poms were arranged in five groups of three and were aligned on the blue line beside the skyscrapers. We ran the code and conducted 30 trials. We observed and used Google Sheets to digitally record the number of poms collected and put into the water reclamation units/ couplers. From the data, we calculated the number of poms put into the couplers each time and generated the above frequency graph for the number of collected and sorted poms.

Data Evaluation

Number of blue poms	Frequency
3	1
4	1
5	6
6	8
7	7
8	4
9	2
11	1

Data Summary	
Average	6.5
Median	6
Mode	6
Standard Deviation	1.607

The data directly relates to the number of poms collected and sorted into the couplers by Demobot's claw; this was to see the consistency of the building and coding aspect of the claw, as well as to see how many poms the claw could collect. The column chart above shows the frequency of the number of poms collected (e.g. how many times 11 poms were collected). From the data analysis, the mode is 6 poms, which means that the Demobot frequently collected 6 out of 12 poms. The average of

6.5 shows that a majority of the time, we were able to collect between around 4 and 8 poms; meanwhile, the large standard deviation of 1.607 indicates that our robot is somewhat inconsistent. This tells us that we need to work on improving both the accuracy and precision of the Demobot and get it to collect a higher number of poms more consistently, as we are aiming for a standard deviation of 1 or below.

Our range of 4-8 poms is a moderate amount, but not substantial enough to get us the maximum number of points. Hence, we can conclude that our collecting mechanism is flawed due to the fact that it collects less poms than it is designed for, as well as its inability to consistently collect the same number of poms.

The data highlights the flaws of the claw system and indicates we must make improvements to enhance the functionality of this mechanism. We have now gained more insight into how we can gain more points in the game as a result of conducting this experiment. We will have to resolve this issue by adjusting the code of the robot to accumulate and sort the poms more reliably and consistently. Furthermore, the small height of the claws may be another reason why the robot wasn't as successful as we hoped it would be; the height of the claw pincers wasn't large enough to support the number of poms. Perhaps making the pincer claws sturdier and longer in height would allow more poms to be carried.

Modified System

We realized that the claw's initial pincers were too small in length, so we decided to extend them by adding more lego pieces; this allowed the claw to pick up more blue poms at once. Furthermore, the small gaps between the lego pieces created a better grab of the poms while ensuring that the poms didn't get stuck between them. The bent lego pieces also contribute to a better grasp of the poms.

We could continue modifying the design by reallocating the lego pieces within the claw until we get the most optimal design that collects as many blue poms as possible. Another further modification we could include is the addition of teeth-like pieces at the bottom of the claws so that the chance of the poms falling out from the bottom is less likely when the claw is being lifted up. The teeth-like pieces would also pick up the poms from the bottom, which would increase the consistency of collecting a lot of poms at once.