Machine Preliminary Progress Report

McGill Engineering Games 2024

Submission date: December 1st, 2023

Table of contents

1. Overall Solution Presentation

Overview of our solution

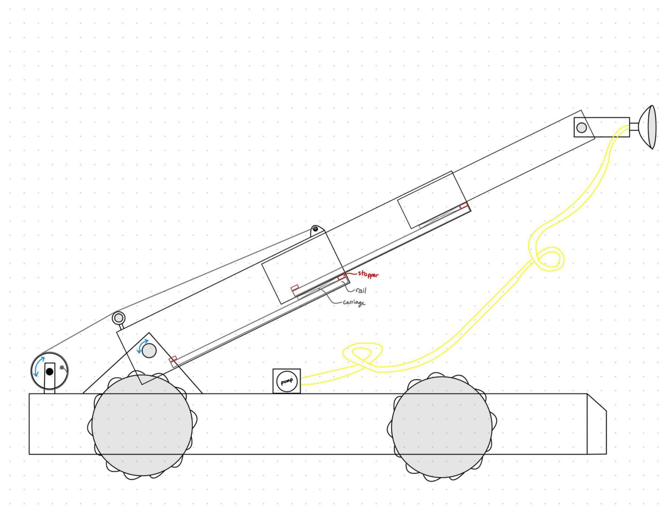

Figure 1: Overview diagram of the robotic solution

Our robotic solution will consist of a rectangular aluminum base surrounded by four mecanum wheels and surmounted by a metal arm equipped with a vacuum pump which will be used to grab and drop balls via a suction cup. Each mecanum wheel will be individually powered by a 12V DC motor. The use of mecanum wheels means our robotic solution will be highly agile being able to maneuver effectively through the competition zone. Moreover, since mecanum wheels permit stationary rational movement, having the arm rotate with respect to the chassis will not be necessary. Finally, the arm will consist of a two member telescoping arm to which the suction cup will be mounted. Such a design will also allow our robotic solution to be able to pass under the launch platform.

1.1 Movement system

The movement system of the robotic solution consists of four mecanum wheels (one to each corner), with each wheel being driven by its own 12V DC motor. All four mecanum wheels will be attached to the rectangular aluminum chassis.

1.2 Grabbing/Dropping system

The grabbing and dropping tasks will be carried out by a telescoping robotic arm to which a suction cup will be attached at the end. The suction force will be supplied by a small pump mounted on the chassis. An endless screw system will allow the arm to extend and a high-torque stepper motor will allow the arm to raise and lower.

1.3 Control system

The control system of the robotic solution is composed of a controller, a single-board computer, some software, and motor drivers. The single-board computer runs the software program and constantly polls the buttons of the controller such that the logic associated with each button can be triggered when the button is pushed. The hardware components like the different motors and the vacuum will be plugged into the computer's GPIO pins set through the motor drivers, which will be used as an abstraction layer to drive the mecanum wheels as well.

2. Subsystem Presentation

2.1 Movement system

2.1.1 Mecanum wheels subsystem

The chosen wheel setup for the robotic solution consists of four independently driven mecanum wheels. Each mecanum wheel consists of one large, powered wheel with a series of unpowered rollers attached to it, each set at a 45º angle. The wheels can be controlled independently. This allows the robotic solution to travel directly in any direction (a-c), rotate on its center (e), rotate on the central point of an axle (f), or move around a bend (d) *(movement visuals in Figure 2 (appendix))*. We have chosen this wheel type and configuration because it allows for superior movement control, with an increased number of movement options. This will allow us to have more maneuverability under difficult conditions, such as navigating around silos or steering under the launch platform.

2.2 Grabbing/Dropping subsystem

2.2.1 Telescopic arm subsystem

First, due to the use of mecanum wheels, having the arm being able to rotate with respect to the chassis will not be necessary. Therefore, the arm will only be able to pivot uniaxially on the chassis to raise and lower itself, similarly to a crane. This rotation of the arm will be made possible through a gear-driven system. First, the gear drive will be powered by a NEMA 23 stepper motor. Moreover, in order to assure sufficient torque, the stepper motor would be connected to the arm through a gear drive system with a gear ratio of 4. In fact, a gear ratio of 4 will be sufficient given the chosen stepper motor. Although the system must supply sufficient torque, the arm must be able to raise and lower relatively quickly due to the 5-minute time restraint. At the moment, we are also considering another option to use a pulley system instead of gears in order to raise and lower the arm. However, in both cases, the same stepper motor would be used.

As previously mentioned, our arm will be telescopic. This telescopic motion will be powered by a pass-through stepper motor and an endless screw. A pass-through stepper motor, as opposed to a traditional stepper motor, rotates a small threaded section within the motor's body. This rotation motion of the threaded section within the motor means that an endless screw can be fed into the stepper motor and moved uniaxially. Having one end of the screw connected to a section of the arm and the other fed into the pass-through stepper motor means that motion of the stepper motor leads to uniaxial movement of this section of the arm. Moreover, after analyzing the required length our telescoping arm must achieve in order to attain the desired silos according to our selected competition strategy, it was determined that a total length of 70 cm was required to reach our targeted silos. Therefore, considering the base's dimensions, it was deemed that two 35 cm sections would be sufficient. Moreover, the telescoping action would be facilitated by a rail and carriage system.

2.2.2 Vacuum subsystem

As previously mentioned, after many options were considered, it was determined that the optimal method of carrying the balls in a secure, fast and effective method, was through a suction cup powered by a small vacuum pump. Moreover, the suction cup would be attached to its own separate actuator, allowing the suction cup's angle to be adjusted depending on whether it had to pick up a ball in the mining station or drop it in a silo. Different suction cups have been purchased and will be tested. Finally, the system would be powered by a small DC vacuum pump mounted on the chassis in order to reduce the weight present on the arm. Both the arm and the suction cup would be connected via vacuum tubes.

2.3 Control subsystem

2.3.1 Controller subsystem

The controller chosen for this robotic solution is a wireless Logitech controller. We have chosen this controller mainly because that is what we had used in previous years, meaning our code is already configured to connect to it and poll its commands. The simplicity of this controller is another criterion which guided our decision, since it has several different types of buttons and commands which can be used either to provide a one-time input or a value from a range in order to gradually increase or decrease a value, e.g., the speed or angle of a motor. There is also a "mode" button which can be used to change the entire configuration of the buttons, allowing for more flexibility in the control system. We are currently investigating having a first button configuration when moving the robot, then another configuration for grabbing and dropping balls so that we can use some of the same buttons for both. Finally, the fact that the controller can be connected directly in a Raspberry Pi 3 through one of the USB ports and communicate via Bluetooth makes it very easy to use with our wireless solution. The controller is powered by two AA batteries.

2.3.2 Computer subsystem

The computer driving our system is a Raspberry Pi 3 single-board computer. The Pi runs the main software program while it receives the input commands from the controller. The hardware components like the vacuum, servos, and motors will be connected to the computer via the onboard GPIO pins set. We have chosen this single-board computer since it is what most of our control system team had the most experience with, and since we had a few Raspberry Pis already to reuse from last year, along with some configuration and code that we knew would work with it. The Raspberry Pi is currently being powered by a micro-USB cable, but it will eventually be powered by 3.7V lithium batteries along with the motor driver. The Raspberry Pi runs a Linux-based operating system and is equipped with a 16 GB micro-SD card for storage.

2.3.3 Software subsystem

The control system software is stored in memory and is written mainly in the Python programming language. We have chosen this language since most of our team had experience with using it for robotics projects, and since we already had some code we could reuse for the configuration of the I/O pins and for the controller. When the script first starts, it reads and parses an XML file containing the robot's hardware configuration, that is, an exhaustive list of all the components along with the pin number they are connected to. Then, the program enters the main loop, where it constantly polls the controller for input. When a button is pressed on the controller, the event that was raised triggers a predefined action which sends the right input to the corresponding GPIO pin of the Raspberry Pi, e.g., making a motor turn to a specific speed. Figure 2 illustrates the flow of the inputs in the control system.

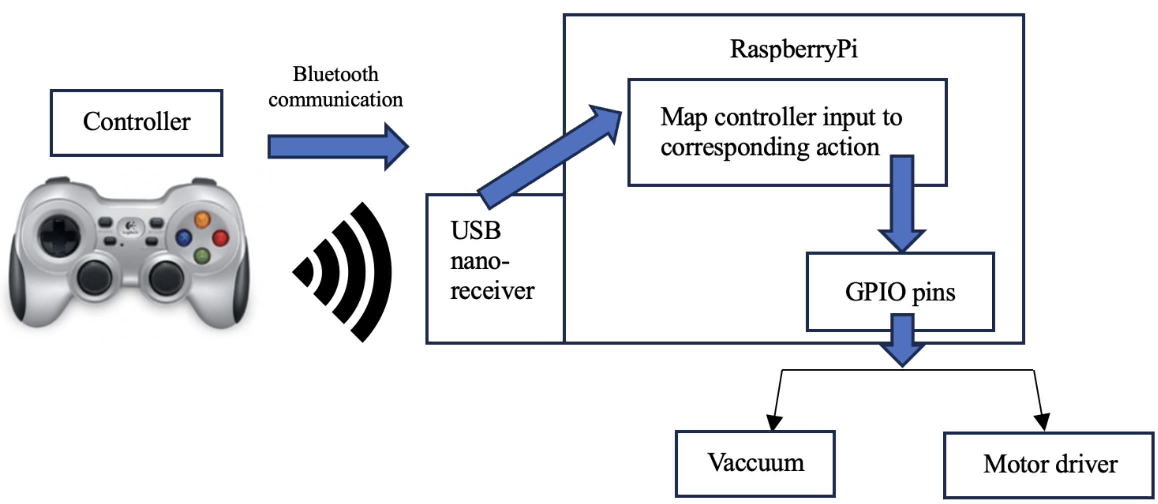

Figure 2: Diagram of the input flow in the control system

For more simplicity in the implementation, we have decided that the robot will only be able to move in the four 90-degrees directions, i.e., "forward", "backward", "left", and "right", and that it can rotate on itself to grab/drop balls. In the controller used in this competition, the directions of the robot will be controlled by the left joystick. A joystick input of ± 45 degrees with regards to the target direction will result in that target direction, e.g., an input between 45 degrees and 135 degrees will make the robot go forward. The rotation of the robot will be controlled by the bumpers, where the right bumper makes the robot rotate clockwise on itself and the left bumper makes the robot rotate counterclockwise on itself. The arm will be controlled by the right joystick. The top direction will make the arm rise, the down direction one will make the arm lower, the right direction will elongate the arm, and the left direction will retract the arm. When button A is pushed, it will activate the vacuum to grab the ball, and pushing button A again will turn off the vacuum and let the ball drop. Figure 4 (see Appendix) illustrates controller mappings. The autonomous part of the trials will also be triggered by a button; we are planning on using button Y for this purpose.

2.3.4 Motor driver subsystem

To control the motors, the motor drivers L298 will be used. These drivers allow the Raspberry Pi to interface with the stepper motors and the DC motors without damaging the computer, since the Raspberry Pi is not suited for this. Instead, the motors are powered by 3S lipo batteries through the L298 drivers. The DC motors we use are suited for 12V, and 3S lipo batteries offer a voltage between 11.1-12.6 V, which is what we need. Then, with the GPIO pins of the Raspberry Pi, a PWM signal is sent to the appropriate pin on the driver, so that the speed can be controlled. The vacuum of the arm will be powered similarly, either through a driver or not, we are still investigating for now. To ensure that the cables and components are protected in case of a current higher than specifications, we will be adding a fuse next to the battery packs such that current will not reach other components if it is too high (see Figure 5 in the Appendix for the full circuit diagram).

3. Considered Strategies and Expected Results

3.1 Strategy for the Individual Trial

When doing the analysis of which silos were the more advantageous, three points were considered.

First, the number of points that could be gained from the silos is not the same everywhere. We noticed that the silo which is worth the most points is the one on the launch platform, with 40 points per mineral, as opposed to the silos in the transformation and analysis zones, respectively worth 20 and 10 points per mineral. The ones in the center of the field are clearly more advantageous than the ones in the analysis zones, which is why we have decided to target these silos first.

Secondly, since the field is rather large, it is important to consider the distance that our robot will need to cross at each step. That way, the only analysis zone silos that can be filled are the ones on the opposite side of our mining station. This means that we need to cross a long distance to drop and refill the balls. With only five minutes, a lot of time would be wasted on movement. Therefore, we have decided to prioritize the silos in the center of the map, since we do not need to cross the gravel or cross under the platform in order to reach them.

Lastly, each silo presents its own difficulty. For the highest silos, an orange cap is placed on top which limits the angle and the direction used by the solution to fill it. Also, since it is not planned for the solution to cross the gravel, our arm needs to reach from a long distance to drop the mineral inside the silos placed in the transformation zones. The same thing can be observed for the rocket placed on the liftoff platform because of its size and height. Considering the factors presented above, we have decided that the smaller silos would be better to target because of the points they can award us, despite their difficulty of being hard to reach, for which we will have to build a longer arm.

For the autonomous phase, we have determined that it would be best to avoid movements requiring precision such as grabbing and dropping minerals, since we cannot guarantee that the motors will always move exactly to the position required, and a lack of precision in this position could easily lead to failure of grabbing or dropping a mineral, which would result in time loss. The autonomous phase being only 30 seconds, we thought it would be best to maximize that time by having the robot do large, simple movements which do not require much precision. Therefore, our current strategy is to simply move in a straight line along the side wall towards the ramp, then going along the ramps and platform and, finally, reaching the mining station by the other side wall. Since the field is rather large, this simple movement would probably take a few seconds even if we were controlling the robot, so this time will be saved. If there is still time left after this phase (meaning we did not take back control of the robot yet), the robot could pick up minerals and drop them in the nearest facing silo in a simple back and forth movement, which would result in placing up to 7 minerals. This requires more precision, however, so we would need to make sure the robot is well calibrated and does not go too fast in order not to drop minerals and have them roll away for us to pick up again.

At the moment, we have no plans of attempting to climb on the platform at the end of the trial, since we believe that it would take a long time to succeed - and specific hardware which we have not planned to build - and we think this time would be better spent filling more silos.

3.2 Strategy for the Duels

The duels strategy resembles that of the individual trials, considering similar factors. These include points, distance from the mining station and silo difficulty. Newly added factors consist of planning around the opposing team's movements as well as control zones–a team must have the last mineral in each deposit site in the zone to control it.

For the autonomous section, our strategy is to use the starting ball and other minerals to fill the towers in the transformation zone closest to our mining station. The initial movements will follow that of the individual trail's strategy, going towards and along the ramps to get to the ramp facing the mining location and the targeted tower (closest to the wall). Once the mineral is deposited, we then go back and forth from the mining station until the tower is full or the thirty seconds run out. Should the tower be full, the robot will then move to the following tower in the transformation zone. Throughout the autonomous period, we will use game elements to help keep our robot aligned. For example, using a wall to go in a straight line.

Once we enter the driver controlled period, we have four phases. The first consists of immediately filling the rocket tower as it is worth the most points per mineral and control zone, and can be done without having to traverse from one analysis zone to another. Once filled, we move onto the second phase where the driver will attempt to fill towers located in the transformation zone nearest to the mining zone, allowing us to minimize travel time to and from the towers. As soon as the towers in the transformation zone are filled, the driver will evaluate if there is still space within the opposite transformation zone and fill them correspondingly. Moving onto the third phase, this part of the strategy is optional and shall only be completed until the fourth phase must begin. It consists of filling as many of the analysis zone towers as possible. These towers are left for last as they are worth the least and are located farthest from the mining zone. There are also no control points attributed to them. Finally, in the fourth phase the driver will attempt to control the following zones in decreasing priority: Rocket, closest transformation and farthest transformation. This order is based on points and distance from the mining location. This phase will begin approximately once minute before the end of the duels, but may vary depending on the towers that are not under our control.

3.3 Expected Results and Test Score

For the individual phase, by filling the rocket and the transformation silo the closest to our mining station we can have a minimum of 4x40 + 6x20 = 280 points. If we fill the whole center, the maximum we could have is 280 + 2x4x20 + 6x20 = 560 points. This is the point potential of the silos in the center of the field. In the case we fill all our analysis silos, an added 2x10x10 + 6x10 = 260 could be gained, but it is less likely.

For the duels, we expect to be able to gain control of as many zones as we can and put as many minerals as we can in the center silos. It is hard to calculate the score in the duels because we do not know the other teams' strategies. If we assume all the center silos contain half our minerals, we could have a total of 2x3x20 + 2x2x20 + 2x40 = 280 points.

4. Risk Management Process

4.1 Risks that influenced our design

Using a conventional wheelbase, there was a chance the robot would get stuck (probability: 50%, severity: 3). We opted for mecanum wheel to remove this risk entirely. The mecanum wheels allow us to move in any direction without the need to rotate.

Using a typical claw mechanism to grab and hold the minerals gives a high chance for the mineral to slip when being grabbed (probability: 60%, severity: 1). So, we opted for a suction cup design that would be built with the mineral dimensions in mind for optimal suction and adhesion.

4.2 Current risks

Risk	Probability (%)	Impact (1-10 scale)	Mitigation	Contingency
Suction cup drops mineral	40	2	Testing and calibrating suction cups beforehand	Only for the control phase. If it stays close, pick the same one up. Otherwise, get a new one.
Infinite screw gets stuck	5	3	Testing and lubing of the infinite screw	Change how minerals are picked up
Getting stuck in gravel	20	5	Design has long arms so we don't have to get close to gravel.	Only necessary for the autonomous phase. Wait for the controlled phase to end.
Controller disconnects	5	8	Make sure the controller batteries are	Bring a second controller (use it in a subsequent run)

			charged, test it beforehand	
Catching on fire	1	10	Use of 2 drivers (instead of 1), use cables with lower gauge, use a fuse to protect the circuit from high current	NA

4.3 Overall risk assessment

Using the results explained in the previous sections, we may quantify the overall risk of our proposed solution, ranging from 0 being the best case and 10 being the worst.

$$Risk = \sum probability \times severity = 40\% \times 2 + 5\% \times 3 + 20\% \times 5 + 5\% \times 8 + 1\% \times 10 = 2.45$$

5. Current CAD assembly

Following are presented pictures of our latest CAD assembly. The base of the robotic solution, including its drivetrain, has already been manufactured and assembled. Parts shown in gray are those who remain to be manufactured or printed. Some significant features which are missing from this CAD assembly are first and foremost, all the controllers, batteries and other electronic components which will be present on the base of the robot. Moreover, this assembly is also lacking the chain gear system which will be powering the arm, the rails which will be facilitating the telescoping motion, and finally the pivoting suction cup at the end of the arm.

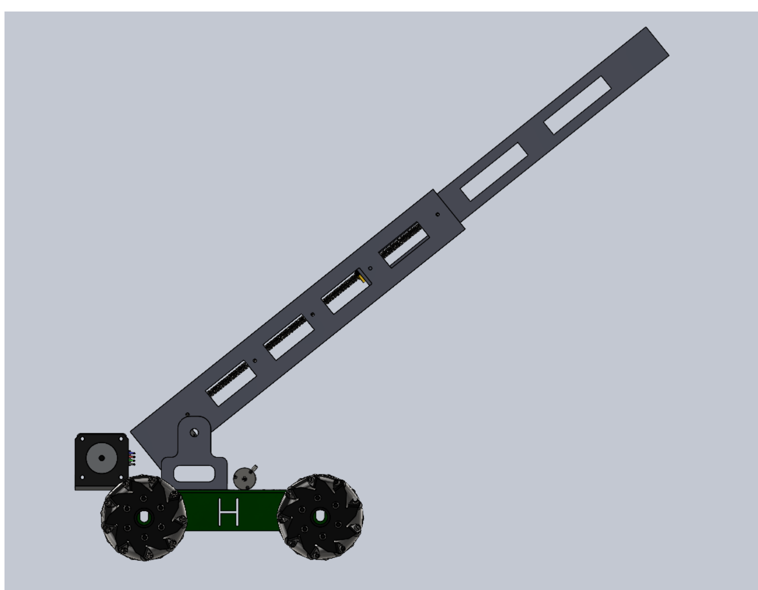

Fig. 3 Side view of the robotic solution

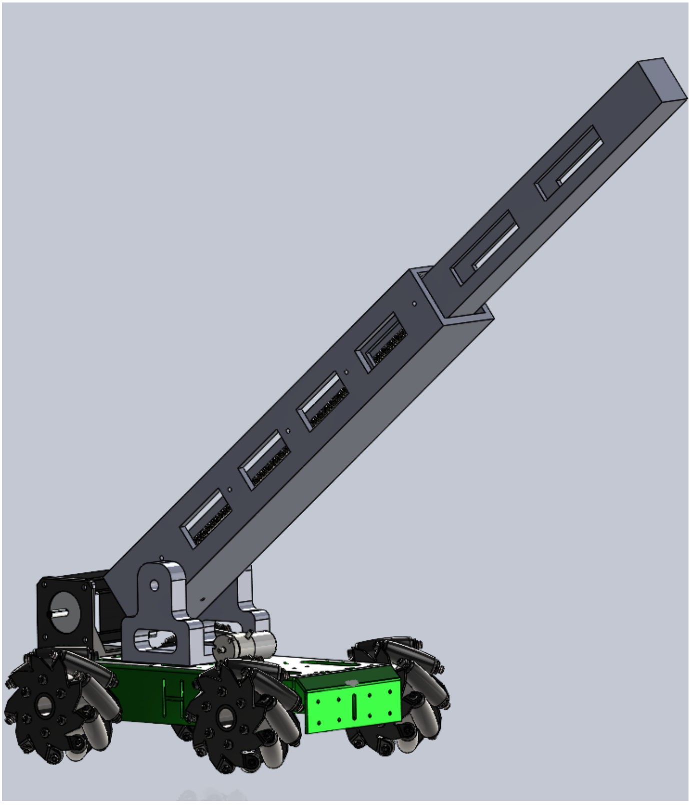


Fig. 4 Diagonal views of the robotic solution

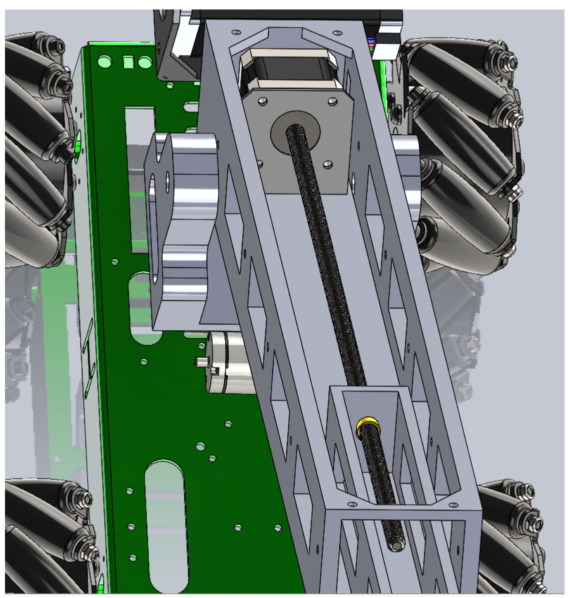

Fig. 5 Close up view of the screw driven telescoping mechanism

6. Appendix

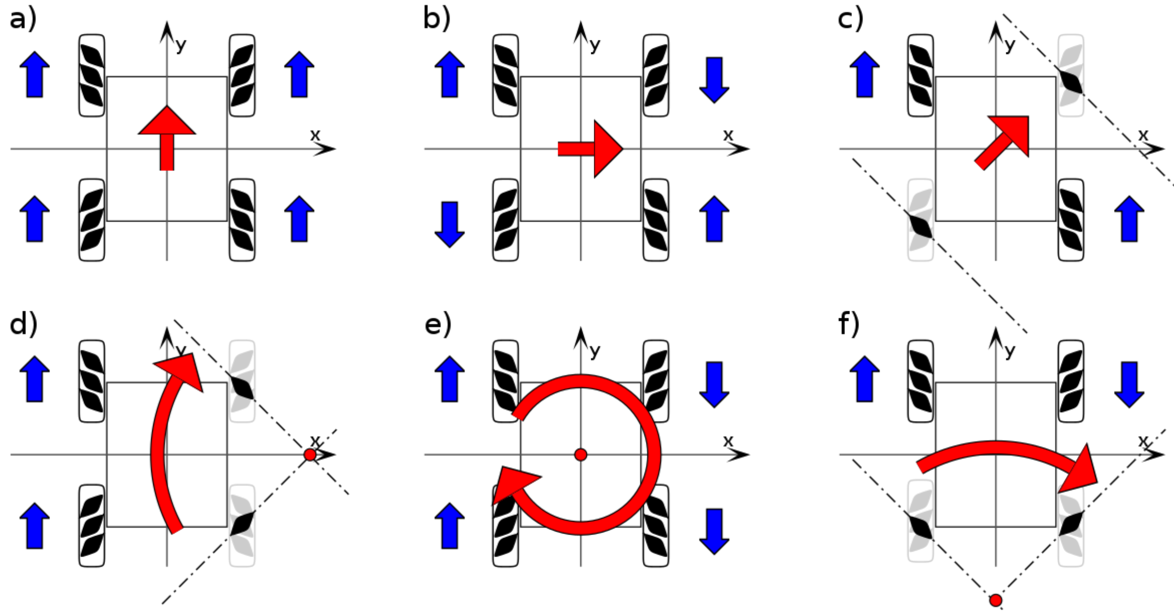

Figure 6: Diagram of movement options with mecanum wheels. Blue is the rotation of indicated rollers. Red is the direction of solution's movement.

Figure 7: Layout of button functionality on the controller

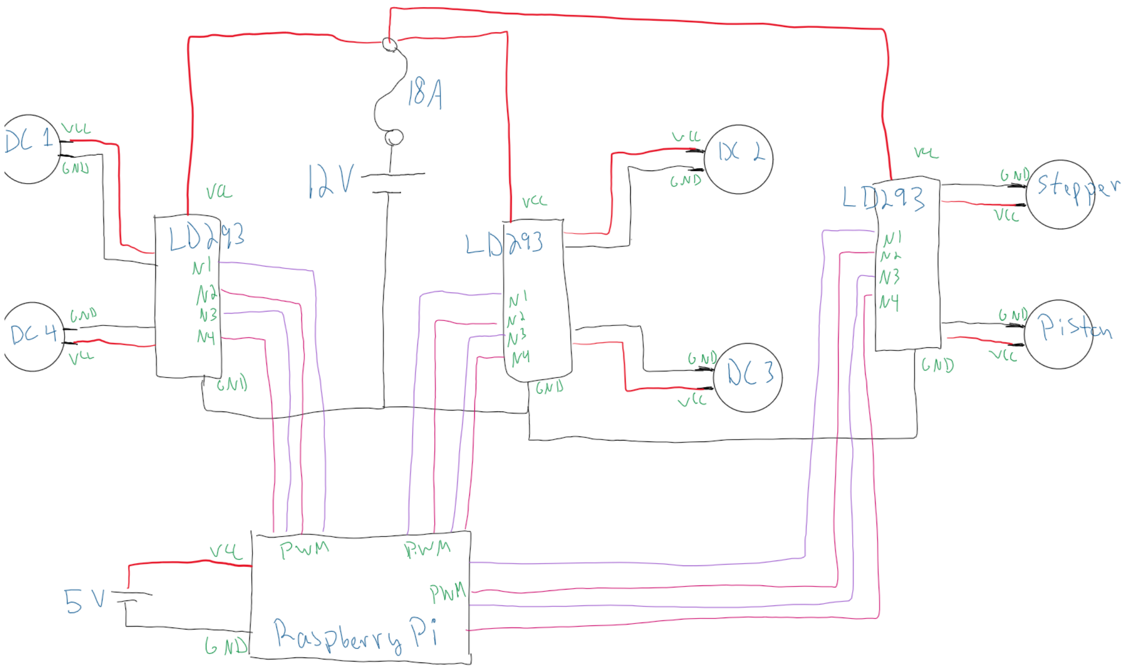

Figure 8: Circuit diagram of the system. The six motors are driven by the LD293 drivers, which are controlled by the PWM pins of the Raspberry Pi. A fuse makes sure that the current does not exceed 18 A.