Vedad Bassari
Steven Man

Final Report - Roborats Competition

I) Introduction and Objectives

The roborats competition is based on a straightforward principle - design a robot that moves around a designated competition area and gathers "cheese," or foam blocks. The robot should be able to do this more effectively than its opponent to proceed in the competition. However, several complicating factors make the contest more challenging. Firstly, the blocks are valued differently based on their position on the board (Table 1, Figure 1). Importantly, some of the blocks, known as sky cheese, are kept above the board using magnets. These blocks are weighted more heavily, making them difficult to obtain but worthwhile objectives.

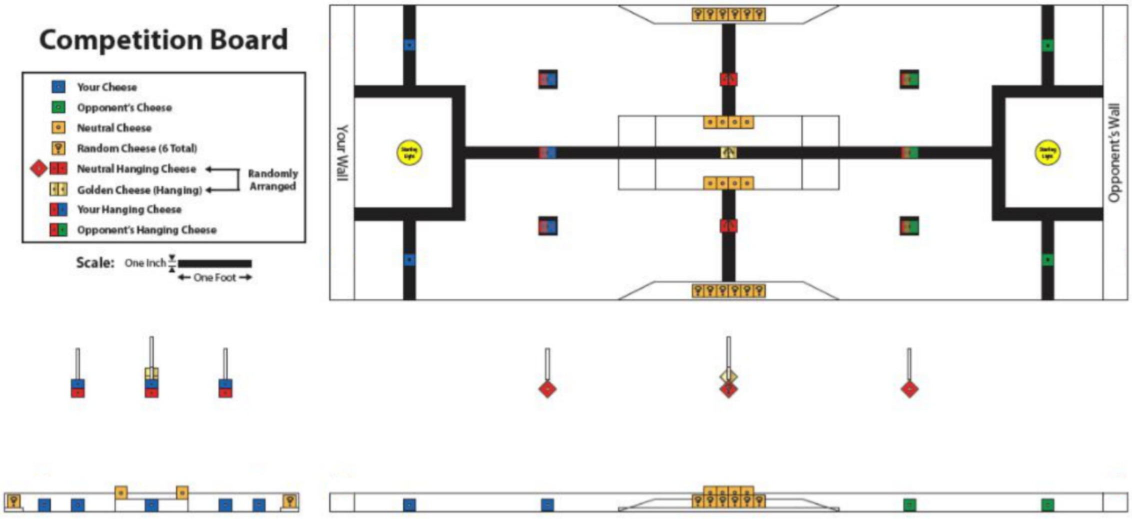


Figure 1 - Competitions board setup.

	Home	Neutral/Random	Opponent
Ground Cheese	2	3	4
Sky Cheese	4	5	6

Table 1 - Cheese value assignments.

Yet another complicating factor is the wall-cheese multiplier, which guarantees that any cheese placed on the friendly wall will be worth four times its original value. Finally, the sky cheese, whose location is selected at random among the center sky-positions, is worth 80 points if placed on the wall. The totality of this setup creates ample room for varying strategies between the competitors.

While the robots are free to explore the design space set up by these rules, there are some notable design constraints. Firstly, the vehicle must fit within a cubic-foot box before competing. Additionally, the chassis must be made from lego pieces, and sensors and motors should be approved before use. Lastly, the robots must be ready for their runs at specified times, so robustness and longevity (especially in collision with other robots) are critical qualities for successful bots.

II) Available Components

Chassis: The chassis was composed entirely of lego pieces. Importantly, technique pieces were made available in large quantities. Their use enabled us to maintain good structural properties while reducing the weight of the vehicle.

Arduino Hardware and Software: To control the robot, an arduino uno was used. We used the arduino development environment to interface with the microprocessor.

Arduino Shields: Two arduino shields were used to augment the microcontroller. Firstly, the adafruit motor shield V2.3 was used to command the DC and servo motors. Additionally, a custom shield was used to implement the remaining components of the circuit.

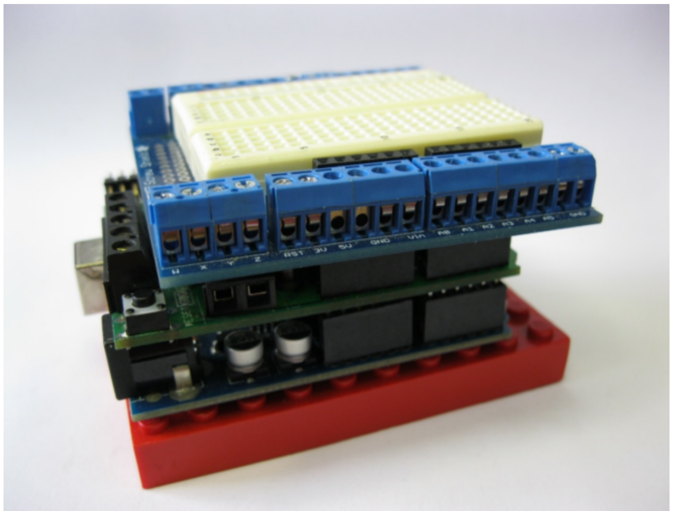


Figure 2 - Arduino and arduino shields stack.

Sensors and Actuators:

1) DC motors: Two DC motors were used to move the chassis around the board.
2) Servo motors: Two servo motors were used to actuate the gripping mechanism. The first servo was used to lift the servo tower (incorporated to meet the size constraint), and the second servo directly actuated the gripper.

3) Range sensors: Infrared short-range sensors were used to follow the board walls. The input from these sensors was used as the input to the wall-following control scheme which will be detailed in the next section.
4) Reflectance sensors: Infrared reflectance sensors were implemented to follow the dark lines that are featured on the board. The information from these sensors was also important in identifying the robot's position with respect to the golden cheese.
5) Photoresistor: We used a photoresistor to start the robot with the board lights.
6) Micro-switches: A number of microswitches were used in the device, with two primary objectives. Firstly, microswitches were used to interface with the robot in order to select the mode of operation. Two additional micro-switches were used as the emergency stop switch and the collision detection method.
7) Break beam sensors: We created rudimentary encoders using two break beam sensors, each accounting for the number of rotations of one of the wheels. The encoder data was essential in stopping at appropriate distances.

Figure 3 - Range sensors, lego pieces, and DC motors as used in the robot.

Power Supply: Two different power supplies were used in order to operate the robot remotely. First, a 9V battery was used to power the arduino and the sensors attached to it. Secondly, we incorporated a battery shelf (totaling 7.5V) to power the motor shield. The increased number of batteries supported the substantial current draw of the DC motors.

III) Design Description

Robot Strategy:
From the beginning of the project, a quick overview of the possible strategies suggested that the highest point-totals could only be achieved using sky cheese. Specifically, placing the golden cheese on the back wall would yield a score of 80, resulting in a likely victory through mastery of a single action item. This, as well as the stimulating

nature of the gripping problem it entailed, encouraged us to design our robot to target the golden cheese. After retrieving the cheese and placing it on the friendly wall, the robot would simply sit in front of the cheese to protect it. As further discussed in the lessons learned section, this approach proved to be extremely risky, as any mishap or unanticipated disturbance during the competition could result in no points being collected by the robot. This encouraged us to plan a fail-safe approach for the robot, but the rigid time constraints of the competition prevented us from implementing a secondary scoring mechanism.

Chassis Design: Speedy motion was required for the robot to reach the golden cheese early in the competition. Subsequently, major emphasis was placed on minimizing the chassis size and reducing the gear ratio to the minimum value that provided sufficient climbing torque. To converge onto the optimal gear ratio, we carried-out a set of ad hoc tests with different chassis configurations. We ultimately chose to use a gear ratio of 37.5, providing a good balance between the speed and climbing ability. The wheel size selection was dominated by the combination of wheel availability and chassis geometry. Finally, the motors were continuously operated above half of their top speed to maximize available power.

Gripper Design:

The gripper consists of two main mechanisms: a lifting mechanism that erects the gripper at the beginning of the match, and a gripping mechanism that actuates the gripper. The former was implemented with the specific goal of meeting the overall size requirement, and its implementation was relatively straightforward. The second mechanism was more challenging to design, as the gripping servo needed to be kept close to the pivot point of the lifting mechanism to minimize the load on the lifting servo. To accomplish this, the output of the gripping servo was delivered to the gripper gears using an extended axle with an attached worm-gear. To prevent axial motion of the worm-gear, several spacers were included to constrain its motion.

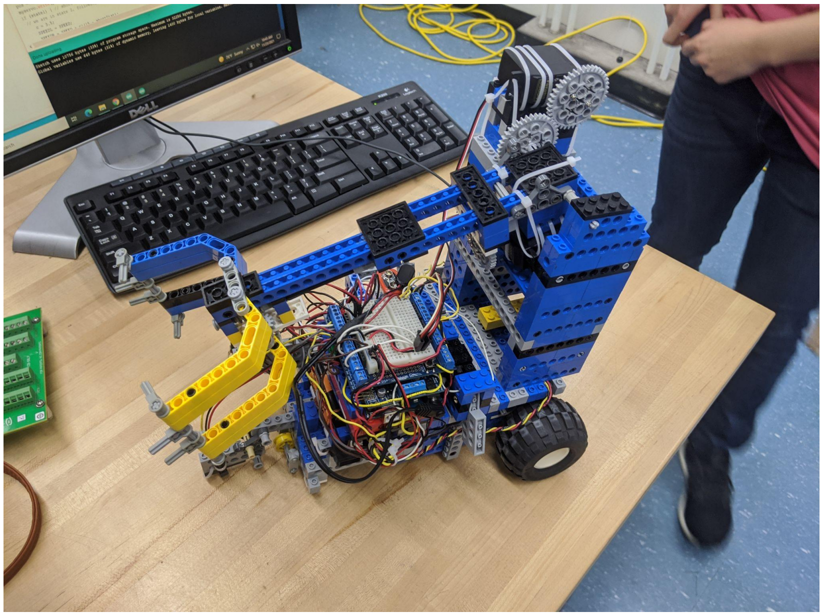


Figure 4 - finished chassis and gripper assembly

Motion-Planning, Sensors, and Control:

Three different motion schemes were prepared for the robot to match the three possible locations of the golden cheese.

a) Center motion: This mode was activated if the golden cheese was placed at the center position. The robot followed the edge of the line at the center of the board using a reflectance sensor, changing motor powers proportionally as the reflectance sensor detected regions of high and low reflection. The stopping criterion in this mode was a specific encoder count, identified empirically. The return path was an open-loop command, simply moving the robot back until a collision with the back-wall was detected. At this point, the cheese would be dropped off.
   The control problem during forward motion was straightforward, as the simple proportional controller was able to keep the robot centered. However, controlling backwards motion with a trailing sensor proved to be substantially more difficult, which resulted in the open loop programming described above.
b) Side motion (left and right): This mode was activated if the golden cheese was placed at either the left or right positions. While the details of the motion depended on the exact side, the overall implementation was identical. Throughout this mode, the robot maintained a set distance from the board walls using the range sensors and a PID controller (Arduino PID Library). The controller was tuned periodically to ensure appropriate responses. The stopping criterion for this motion-scheme was a black line that marked the center of the board - this marker was identified using the reflectance sensor.
   Similar to the discussion above, the task of wall-following during reverse motion proved to be a major challenge. As such, this mode also featured an open-loop reverse motion phase.

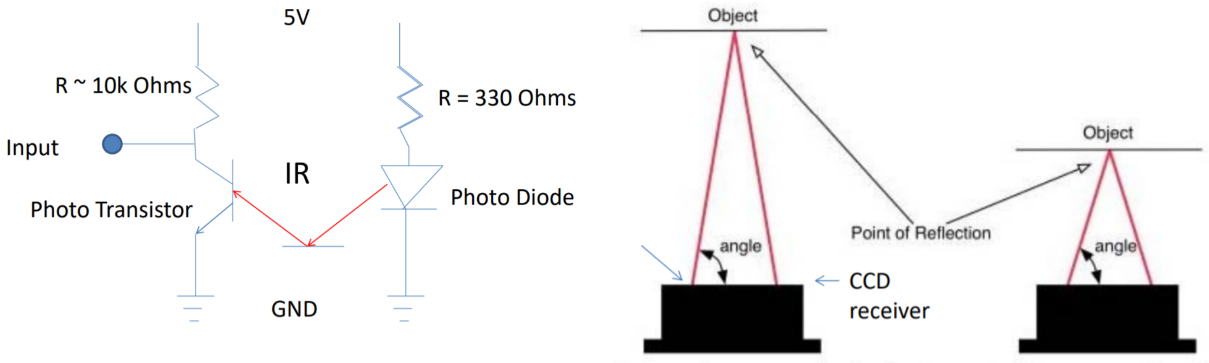


Figure 5 - Working principle of reflectance and range sensors, respectively, as used to control robot motion.

IV) Competition Performance

During the competition, we were able to score 80 points in one of our matches against the opponent. Unfortunately, we lost our first and third match leading to our elimination. Relative to other designs, our robot is more dependent on every component working flawlessly (sensors, motors, and batteries) since not following the wall precisely or stopping at the right distance would not give us any points while other designs are constantly moving around the board to gather ground and sky cheeses until the end. The performance of our robot is not as reliable as the design that swept sky cheeses into a basket since our results are either 80 points or 0.

V) Lessons Learned

We would tell next year's students to focus on solving one problem at a time and make sure they have one set of functions complete before moving on to the next because integration would become nearly impossible if we do not know for certain that every component works flawlessly on its own. The main changes we made after the trial run were to reduce crane weight and improve crane structural strength (the chassis was too complex to reduce) and we left the gear-train the same since power was at an adequate level for ramp climbing and planar maneuvers.

If we're competing again, we would improve the overall structural strength of the chassis and incorporate more closed-loop control (to avoid incoming traffic). I would leave the sensors used and mode-selection buttons the same since they've worked pretty well for our runs.

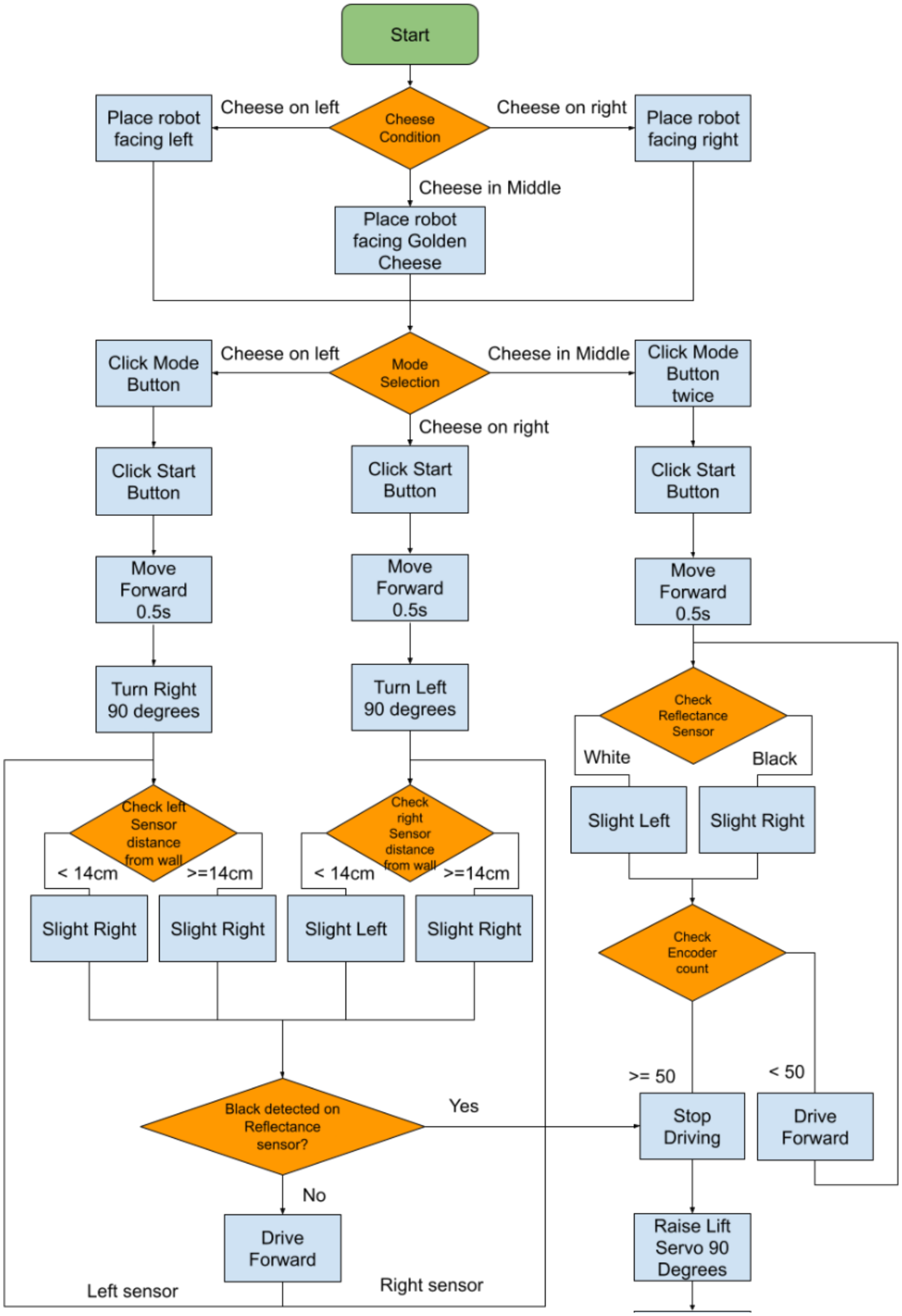
Our design process started with defining the problem: we want the golden cheese. We then designed the chassis and mechanism required for gripping and dropping off the golden cheese. After testing the required hardware we programmed iteratively to achieve the correct motion sequence to secure the golden cheese. Next time, we will perform a unit test on each component before hardware assembly so we understand how well the different components will function independently.

Vedad Bassari
Steven Man

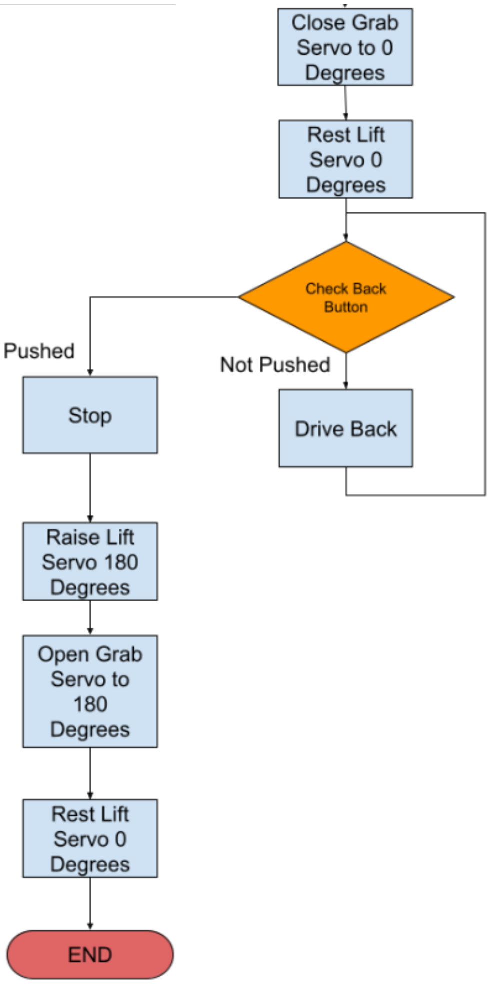
VI) Appendix

Flow chart attached below.
Arduino code submitted separately as a .ino file.

Vedad Bassari
Steven Man



Vedad Bassari
Steven Man


Close Grab Servo to 0 Degrees
Rest Lift Servo 0 Degrees
Check Back Button
Pushed
Not Pushed
Stop
Drive Back
Raise Lift Servo 180 Degrees
Open Grab Servo to 180 Degrees
Rest Lift Servo 0 Degrees
END