3304R Robot Documentation

Understanding the Game

Game Objects

Tower Takeover has two game objects. For the purpose of this document, the term 'game objects' refers to both items that the robot interacts with and field elements.

Cubes

Undeniably, the crucial game object in this game is the cube. The cubes are, as the name would suggest, cubes that measure roughly 4 inches on each axis. These cubes are meant to be either stacked or deposited in towers.

Tower

The tower is a game object that the cube can be put in. This allows the stacks' score to be multiplied by (1+x), where x is the number of cubes that have been placed in towers. Note that the bonus is applied only to cubes that have the same color as the cube in the tower. Refer to the game manual for more information.

Scoring

Robot Constraints

In Tower Takeover, each aspect of the game requires different capabilities from the competing robots. As such, it is important to understand exactly what maneuvers the game entails, and what is necessary to accomplish these tasks.

1. **Cube Transport**
   The cubes need to be lifted off the ground in some way and arranged in a stack formation. This means that the bot's specifications must include the following:

   - A way to grasp the cubes for transport. Possible solutions include a claw, or an intake pushing cubes into a storage portion of some sort. In the case of an intake, the storage should be supplemented with quadrilateral barriers to keep the cubes stacked.
   - Quantity is also a key factor. Especially during the autonomous period, the number of cubes that the bot can take in at once will greatly affect the efficiency and potency of the program. However, it is important to note that efficiency and precision are inversely related. A proper analogy would be using a scalpel to amputate, and a hacksaw to perform surgery.
   - Some way to raise the cubes in order to take cubes out of the towers and drop cubes into the towers. This means that in order to be able to effectively play the game, the primary holding mechanism must be adequately able to grip the cubes despite excessive vertical movement, preferably withstanding high speeds.
2. **Tower Interaction**
   The towers are crucial to effectively scoring points. These depositions effectively multiply the

overall score of a team. Although strategy is also key to utilize the towers effectively, there are certain things that the bot must be able to do to effectively use strategies:

- Some sort of elevation mechanism for the cube transport. This is necessary because the bot needs to be able to take cubes out of the towers, but also be able to drop cubes in.
- There are of course constraints on the lift itself. The lift must have linear vertical extension sequence. Any sort of arc-based lift, like attaching a C-Channel to a motor, is out of the question, as the ascension and descension would interfere with the towers themselves, and additionally be cumbersome to operate.
- The lift, as a core concern, should be able to keep the cube transport in front of the drivetrain to some extent. More on that shortly.

3. **Stack Deposition**
   Although it may seem a simple matter, the actual deposition of the stack is quite important. What's the point of making an 11-cube stack within the bot if the bot can't place those very cubes within the proper area? The constraints on the bot with this element are not as drastic as with the other two elements. If the cube transport protrudes past the drivetrain, there should be no issues. This is to ensure that the bot can roll in front of the stack area and deposit the cubes without having to go through the cumbersome process of driving over the barrier, risking creating an invalid stack.

Development

Early Season

Iteration 1:

The Bot:

For our first iteration of our bot, we decided to create a claw-bot. The features of the bot included:

- A 3-level claw, that enable us to grab up to 3 cubes at any given time while still maintaining precision.
- A double-reverse-four-bar lift, that allows us to create stacks ranging up to 11 cubes tall.
- A miniaturized drivetrain that allows for extra clearance towards the front of the bot
- A tank drive, providing an easy-to-use driver experience.

[Insert Picture Here]

We encountered numerous challenges along the way developing our bot, one of the main ones being shrinking the lift while maintaining structural integrity. The final solution ended up being a cross bar connecting the front of the bot to the back, as pictured below.

[Insert Picture Here]

Another area that took some perfection was the gearing for the second extension of the lift. We wanted to move the bars inward slightly to stop the lift from rubbing against itself. After some tinkering around, a final solution was adding some extra gears inward.

[Insert Picture Here]

The Code:

The code itself at this stage is rudimentary. We are using the Purdue Robotics Operating System (PROS) because of its exemplary PID support, which allows our bot to be more accurate in its movements, turns, and direction. Some of the features of this early code are:

- **PID Control:** Although not completely tuned yet, the PID as of now enables our bot to move at roughly 150 RPM while maintaining a constant speed. The main achievement of this iteration of tuning the PID was achieving a stable forward/ backward movement, which it achieved. Future iterations will include Turn and Movement PID to minimize overshoot and increase turn accuracy.
- **Calibration:** We've designed an experimental calibration function to maintain autonomous accuracy despite drift on various fields. The calibration function utilizes an ultrasonic sensor in order to check the distance from the back wall after a test movement, then scales the rest of the bot's movement based on the ratio of the distance it moves comparative to the distance it was supposed to.
- **Kill Switch:** If something goes wrong with the bot and the bot must be effectively disabled, or the bot should be locked into a specific place in the endgame, we've programmed in a kill switch. This kill switch stops all the motors and puts them into the 'hold' brake mode. This effectively renders the bot immovable.

Testing:

Upon testing the bot, there were a few things that we noticed very quickly:

- The lift was too heavy. The amount of high strength gears and c-channels we used was far too much for our motors to handle, which caused the lift to quickly stall.
- The handling of the claw was horrid: It wasn't as accurate as we expected and took too long to make stacks that weren't pre-made on the field.
- The claw didn't have the grip we assumed it would have. The cubes kept slipping out of the claw, despite the anti-slip material we added to the corners of the plexiglass of the claw.
- We forgot to account for inertia when writing the calibration function. Therefore, we would like to refine the equation we use to find the calibration coefficient.

Taking these results into consideration, we began working on our second iteration.

<u>Iteration 2</u>

Note: There were no major changes to the code, so we've elected to not write documentation for it in this iteration.

The Bot:

The bot itself was not scrapped, but we swapped out the claw for an intake. This was done for the following reasons:

- **Higher Efficiency:** In standalone testing, the intake was able to devour 5 cubes in a matter of seconds. This efficiency would greatly enhance our autonomous period.

- **Security:** The rubber rollers on bottom combined with the rectangular barriers going above the intake made it so that the cubes fell out of the intake far less often than with the claw.
- **Capacity:** We added a cross-bar on the top of the intake that was made of two steel 1-wides, then added rails within the c-channel supports also comprised of steel 1-wides. This basically allowed the cubes to go past the top of the intake while having a passively extending method of security.
- **Weight:** Theoretically, the weight of the intake would be much lighter than the weight of our claw, due to the minimalistic design.

Although the lift is still not perfect, we'd like to complete our intake mechanism before redoing the lift.

[Figure 2.1 Early Sketches of Intake Mechanism]

REDACTED

[Figure 2.2 Intake Mechanism Prototype Version 1]

REDACTED

[Figure 2.3 Intake Mechanism v1 Mounted]

REDACTED

[Figure 2.4 Lift Extended with Intake Mechanism]

REDACTED

Testing:

Overall we found that the intake was much faster at actually stacking cubes while in the bot's posession. However we noted a couple of issues with this intake:

- There was little horizontal tolerance – in order for the cube to go in, you had to be perfectly aligned with the cube, which is difficult in the context of a match. We mitigated this issue by adding plexi bumpers that aligned the cube for us, however it was stilll an issue.
- There is a lot of unnecessary motion in lifting the intake and bringing it back down – stacking using this design is a hassle and could be much more efficient.
- We began noticing some issues with the flipout – we had a sliding extensino on top, but that sliding extension would whip outwards when we tried to flip out. This got the intake caught on the inside of our bot.
- The intake actually ended up being heavier than the claw (at least at this point)

Note – We made another 5 or 6 iterations of the claw, however they all functioned the same as this one - using the rubber rotor (thingie) to intake cubes from the sides of the tube. Due to the fact that it was just a weight decrease, we have included an image of each iteration below. Note that each consecutive iteration decreases the amount of steel used and increases the plexiglass.

[Figure 3.1] – Plexiglass tube, aluminum slider

Redacted

[Figure 4.1] – aluminum tube, Plexi extension, Plexi reiforcements

Redacted

[Figure 5.1] – plain aluminum design – no plexi used

Redacted

[Figure 6.1] – plain plexiglass design – aluminum 1x1 L channels used to reinforce the corners and crossbar.

Redacted

[Figure 7.1] – Near final design – Aluminum chassis with plexi reinforcement + aluminum extension

Redacted

[Figure 8.1] – Final design – Aluminum chassis with plexi reinforcement, bevel gears, and aluminum extension

Redacted