Control Award Sponsored by Arm Submission Form

****Please turn in this sheet during your judge interview along with your engineering portfolio****

Team # 14212	Team Name: Metrobotics

Autonomous objectives: The robot can detect the position of our custom shipping element and deliver the starting freight to the corresponding level on the alliance shipping hub. After the robot delivers the freight, it then parks completely within either the warehouse or storage unit depending on our starting position.

Sensors used: We used an external webcam for object detection. We attached it to the front of our robot in order to detect the position of the shipping element at the start of autonomous. We also used the built-in encoders in the motors to receive and track the number of ticks the robot is at currently and will move to. This provides a more precise autonomous movement.

Key algorithms: We used OpenCV as our object detection algorithm. Our OpenCV algorithm detects our shipping element by looking for a specific color (green in our case) and determines the position using a pipeline made for creating and determining points on a rectangle. Once the shipping element is in view of the rectangle created and the appropriate color is detected, the algorithm determines the position of the shipping element by calculating whether it is within a certain range of x-coordinate values on the rectangle. This is done by using if-else statements to differentiate between the different randomizations.

Driver controlled enhancements: In order to make our Tele-Op more efficient during the driver-controlled period, we used several finite state machines to automate several algorithms.

Our arm state machine control algorithm allows us to move the linear slide to a set position for each level while the driver maintains complete control over the robot. When the driver determines that the robot is in the correct position to deliver the freight, a button is pressed to autonomously drop the freight, and reset the servo and linear slide. To account for any errors, we implemented a fail-safe button that returns the linear slide and servo to a neutral position.

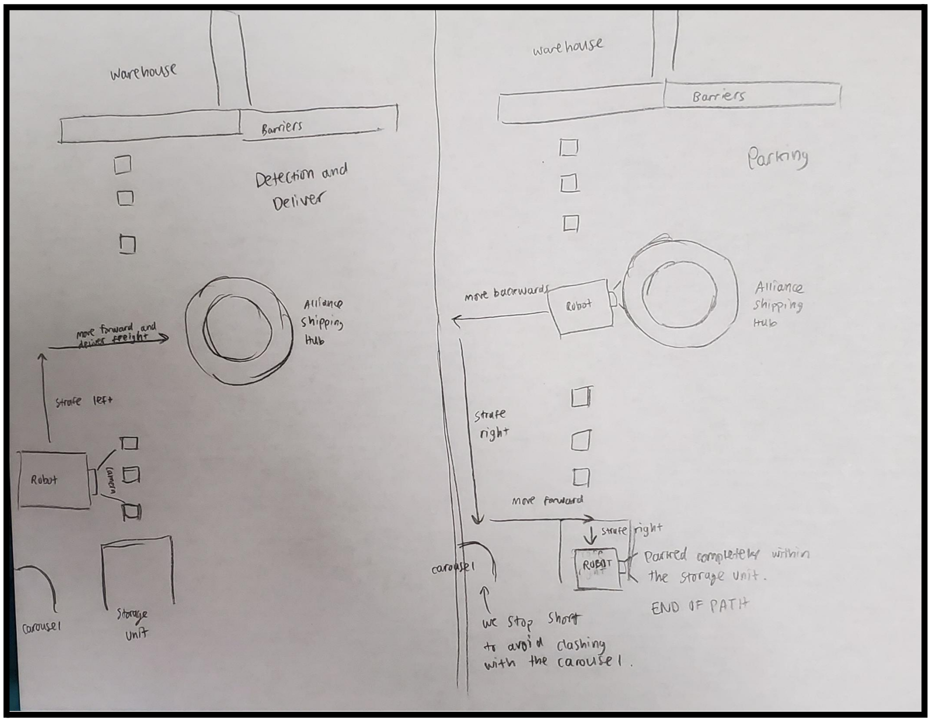
An intake state machine control algorithm was implemented to automate the intake process. By pressing a button, the intake motor can either intake freight or out-take external freight to avoid the multiple-freight penalty by autonomously reversing the intake motor and rotating a servo to lift the intake. This was done to allow the driver to focus more on driving towards the freight than managing the intake system.

We also implemented a state machine for our drive train to shift between a slow and fast mode. By using the right and left bumpers on our controller, the driver can change the speed of the wheel motors while driving. This was added so our driver can quickly move around the field, but also make finer adjustments during the intake and the freight-deposition process.

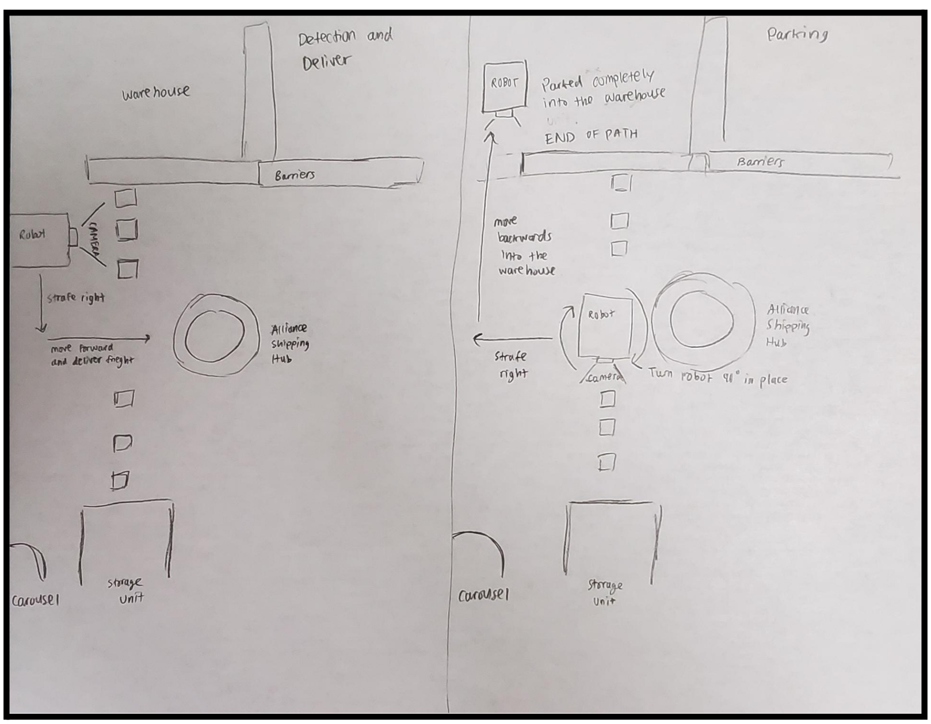
Engineering portfolio references: Pages **11** and **12** of our Engineering Portfolio details the design process and goes more in depth on our systems and algorithms.

Autonomous Program Diagrams:

General Right Side Path -



General Left Side Path -



These are the main two paths that are slightly altered depending on the team alliance (red or blue) and left or right position. We also incorporated a sleep statement at the beginning to temporarily pause our autonomous mode in case we may clash with our teammate.