Control Award, Sponsored by Arm, Inc., Submission Form

Team #: 15317	Team Name: Error Code 404

Award Video Submission Link: https://youtu.be/aV5HqIrGXds

Autonomous objectives:

- Determine the number of rings in the starting stack.
- Pick up and transport the wobble pole to the appropriate square on the field.
- Determine the robot's distance to the base of the goal.
- Shoot three rings successfully into the high goal

OR

- Shoot three rings and successfully knock down all three powershots.
- Park the robot on the launch line.

Sensors used:

- 1 frontal webcam is used to run OpenCV during autonomous
- 1 rear webcam is used to run Vuforia during Teleop in order to determine the robot's position, heading, and distance from the goal.
- 4 encoders are used to direct our four driving motors to correct distances.
- 1 encoder is used to maintain a constant motor speed for our shooter flywheel regardless of available voltage.
- 1 touch sensor is placed near the end of our shooter and tracks when rings are launched

Key algorithms:

- The robot calculates the angle that it needs to turn in order to point towards the goal. It does this by using Vuforia to determine its coordinates and its heading. It then moves in the most efficient direction to reach the angle.

- The robot calculates the angle that its shooter must be set to in order to aim accurately by finding its distance to the goal using Vuforia. This algorithm incorporates data sets that we collected.

- The robot calculates the power that its shooter must operate at in order to prevent rings from traveling further than 16 feet when fired. This algorithm incorporated our team's own data from testing. We use an encoder to make sure that our shooter flywheel maintains its speed, regardless of battery power.

- The robot uses OpenCV to determine the number of rings in the starting stack during Autonomous. It does this by analyzing the color of the pixels in a targeted portion of our webcam's feed.

- The robot uses Vuforia to find the robot's heading and coordinates in order to calculate the distance and direction it needs to travel in order to reach a specific point from anywhere on the field. We use this to aim consistently at the powershot during endgame.

Driver controlled enhancements:

- With a push of a button, drivers can:
    - Make the robot turn towards the goal,
    - Set the robot's shooter to the correct angle for shooting in the high goal at its distance, and
    - Set power of the robot's shooter in order to shoot accurately into high goal without exceeding

the 16 feet ring trajectory limit.
    - This can be done from anywhere on the field where the robot's side webcam can recognize a Vuforia Wall Target.

- Our robot can operate a lift using the same motor as our collector. This depends on whether the gear of the lift meshes with the gear on the collector shaft. Our code automatically shifts this gear to the right position, depending on whether the driver is controlling the lift or just the collector using the gamepad.

- Our robot has a small servo that we repurposed to indicate whether our camera is recognizing a Vuforia target. It pivots up to let our drivers know that they can successfully auto-aim and auto-position using Vuforia

- With the push of our button, our robot will:
    - Turn to face directly towards the powershot targets and
    - Move to a specific point on a field to maximize powershot accuracy.
    - Afterward, with an additional button, our drivers can strafe a set amount to aim at the next powershot.
    - This can be done from anywhere on the field where the robot's side webcam can recognize a Vuforia Wall Target.

Engineering notebook references:
To view our Engineering Notebook, please visit our team's website at:
ftc15317.weebly.com/engineering-notebook

- To see more about our driver controlled enhancements, including:
    - Auto-aiming
    - Synchronous state engine features such as auto-positioning
    - Visual indicator for Vuforia recognition
    - Automatic PTO positioning

  please see pages 2-13 of our Engineering Notebook. Please also see the Programming section of our Engineering Portfolio for our testing, data analysis, autonomous program organization, and further explanation of the Teleop features

- To learn more about our Autonomous programs, see pages 14-21 of our Engineering Notebook.

- To see more specifically about the OpenCV state we use to detect the number of rings in the starting stack during the Autonomous program, see pages 22-26 of our Engineering Notebook.

- To view our state for shooting during autonomous, reference pages 27-28 of our Engineering Notebook.

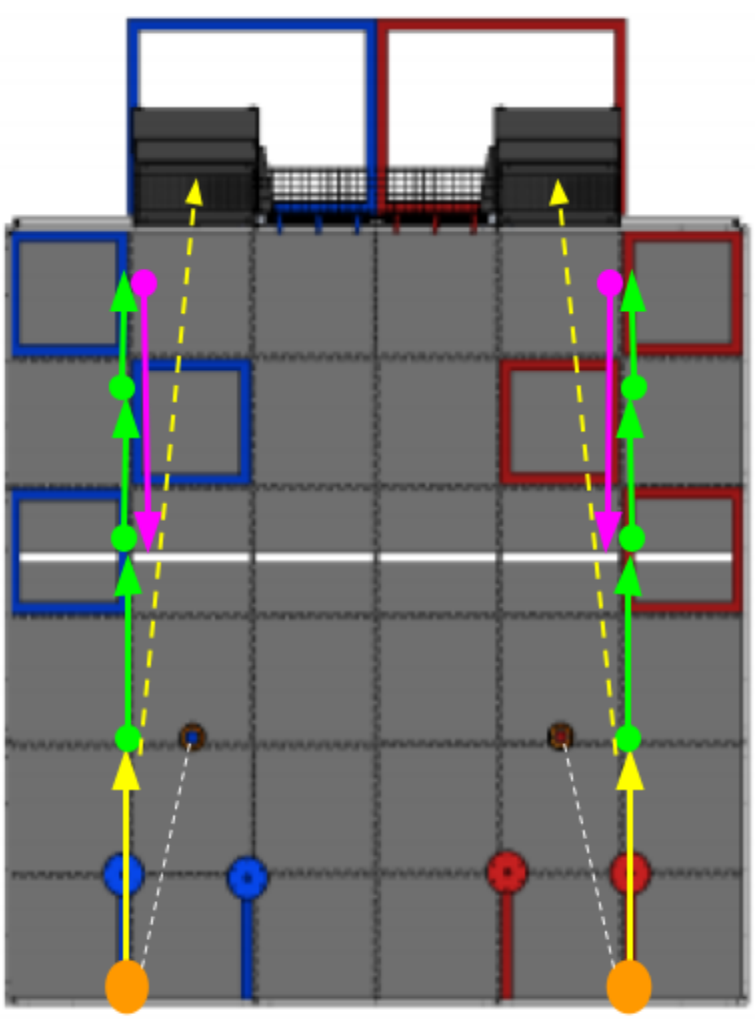
To explore our team's full code for this season, please visit our **GitHub Repository**.

Autonomous program diagrams:

For outer autonomous starting positions:

2. Move forward and shoot three rings into the high goal. Track each ring leaving the shooter using a touch sensor.

1. Use OpenCV with frontal webcam to determine the number of stacked rings. Pick up the wobble pole.



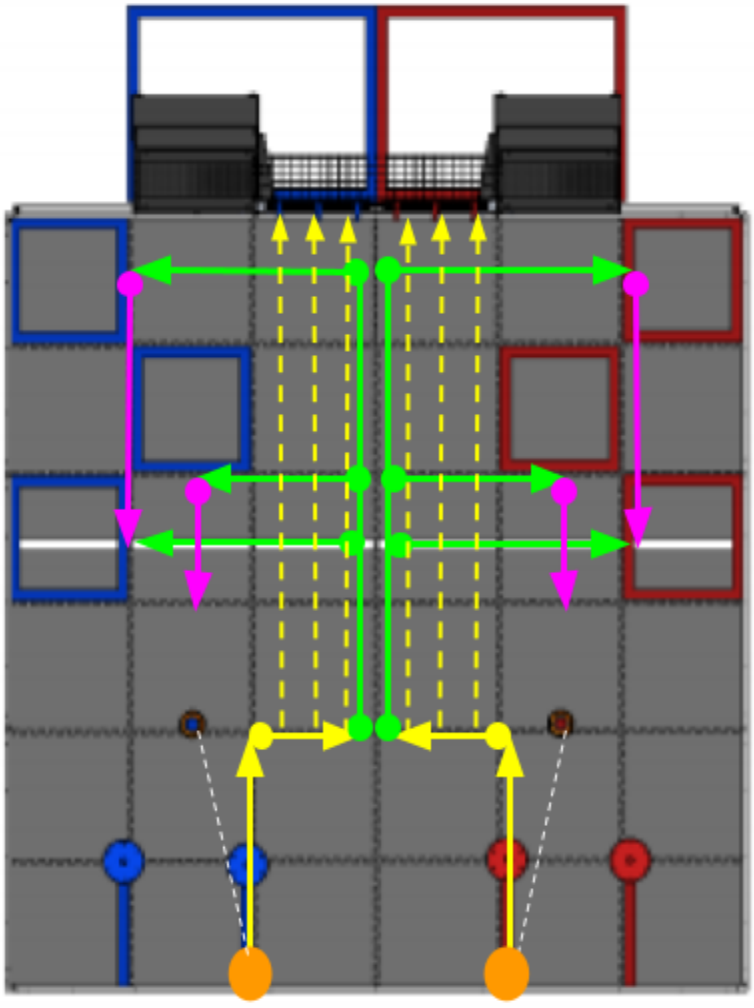
3. Move forward based on the number of rings in the starting stack. Rotate and drop off the wobble pole in the taped box.

4. Park on the launch line by moving backwards using encoder values.

For inner autonomous starting positions:

2. Move forward and shoot each power shot, strafing in between. With a touch sensor, track each ring being shot.

1. Use OpenCV with frontal webcam to determine the number of stacked rings. Pick up the wobble pole.



3. Move forward and turn based on the number of rings in the starting stack. Rotate and drop off the wobble pole in the taped box.

4. Park on the launch line by moving backwards using encoder values.