Team # 14126	Team Name: Fibonacci	Control Award

Autonomous Objectives:
We programmed a total of 4 different autonomous' depending on the state of our robot and specific match strategy. Our **first** autonomous delivers a total of 4 cones onto the high junction (in a 1+3 fashion) and then parks according to our custom signal sleeve. Our **second** autonomous delivers the preloaded cone onto the mid junction and 5 additional cones onto a low junction and then parks. Our **third** autonomous delivers only the preloaded cone onto the high junction and then parks. Our **fourth** autonomous simply parks.

Sensors Used:
Our robot utilizes 1 limit switch, 1 IR color sensor, 2 distance sensors, 2 odometry encoders, 4 drive encoders, a gyro implemented in the REV IMU, 2 lift encoders, 1 range sensor, and a camera using Vuforia.

Key Algorithms:
Our autonomous program uses *5 drive functions*. The functions allow us to individualize inputs such as speed, distance, yaw angle, etc. The function will store the user input data into variables and use it within the function to be executed. All of our drive functions have an input that will employ logic depending on which side of the field we are running on.
Functions:

- <u>Our Functions:</u>
  - GyroDriveENC: to drive forward and backward accurately according to the set path. Utilizes 2 odometry encoders, 4 drive encoders, and gyro.
  - GyroDriveStack: to move forward until the distance sensor sees a cone and then plunging, grabbing, and lifting to the necessary height to deposit the cone. Utilizes a distance sensor, IR color sensor, lift encoders, and drive encoders.
  - GyroStrafeENC: to move side to side accurately using the odometry encoders and gyro.
  - GyroSpin: to spin accurately according to desired direction using the gyro.
  - liftENC: to move the arm up and down safely. Utilizes amperage, limit switch, and lift encoders

Pole Detection - Additionally, we use a range sensor on the back of our robot to detect poles and drive based on that. This way of driving solves our problems with field discrepancies as well as faulty encoder values.
Amperage - We had the problem of our arm overextending. To fix this we implemented an upper and lower limit based on the amps being fed into our lift motors. This fixed any overextension we were experiencing.

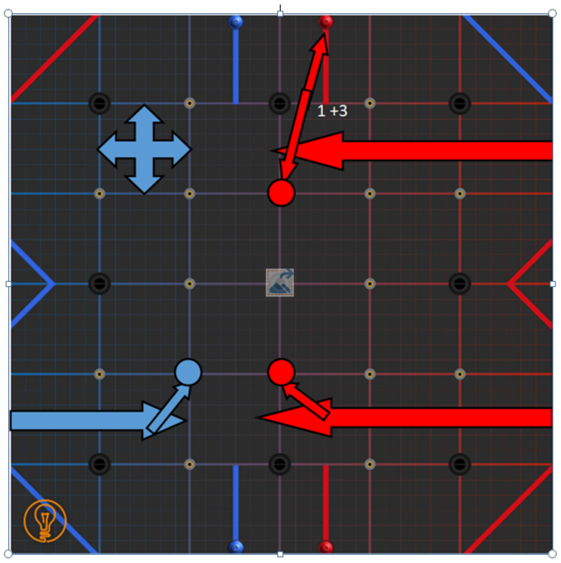
Driver Controlled Enhancements:
In our TeleOp, we have a total of 3 drivers controlled enhancements:

- **Yoink mode** uses *TeleOp automation* to accurately pick up a cone with the single press of a button. Yoink mode uses 2 distance sensors and 1 IR color sensor
- **Arm reset** uses 1 distance sensor and a limit switch to reset the arm encoder values during TeleOp
- **Moving a plow** up and down automatically to square cones and prevent damage.
- **9 Preset buttons** programmed to go to specific heights using encoder values. 4 for delivering onto junctions and 5 for picking up from the stack.

Engineering Portfolio References:

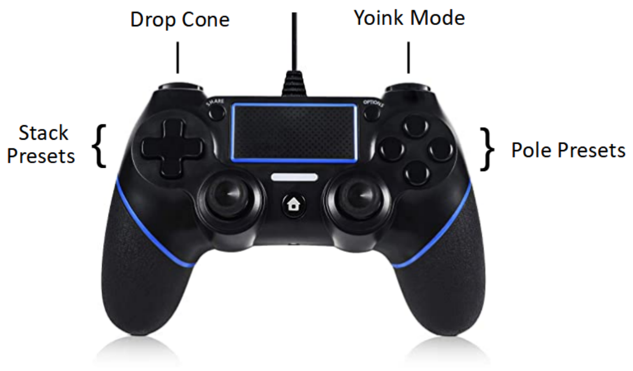
- Design page 8-11.
- Control page 13-15.

Autonomous Program Diagrams

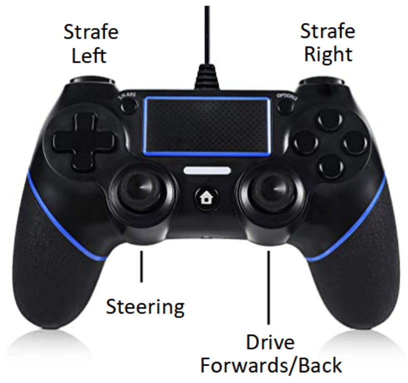


Controller Maps

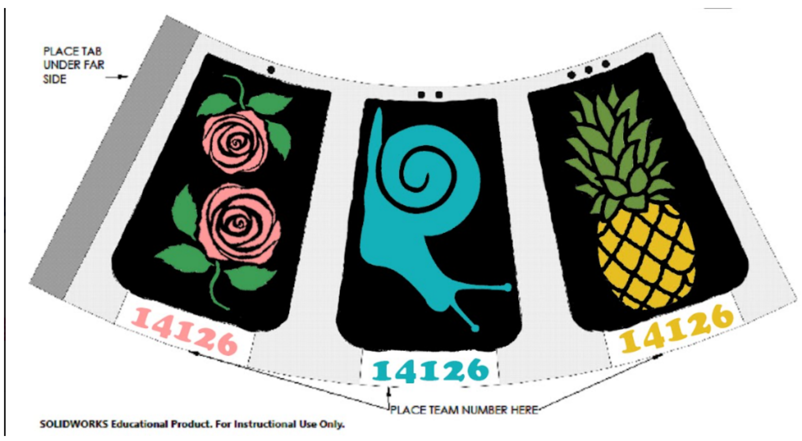
Player 1



Player 2



Custom Sleeve


PLACE TAB UNDER FAR SIDE
14126
14126
14126
PLACE TEAM NUMBER HERE
SOLIDWORKS Educational Product. For Instructional Use Only.