Directions for Autonomous Setup 2023
CENTERSTAGE First Tech Challenge Game
Marist Robotics

Description:

These steps outline the process of setting up Autonomous Op Modes to deliver pixels to the Spike Marks or the Backdrop in the CENTERSTAGE Game. The first part of these directions will guide you through setting up methods to deliver pixels to zones one, two, and three. The second part will integrate Tensor Flow Vision or Touch Sensors to detect the randomization.

Summary of Autonomous Goals for CENTERSTAGE:

Game Flow:

1. Scoring System Chooses a random zone (1, 2, or 3)

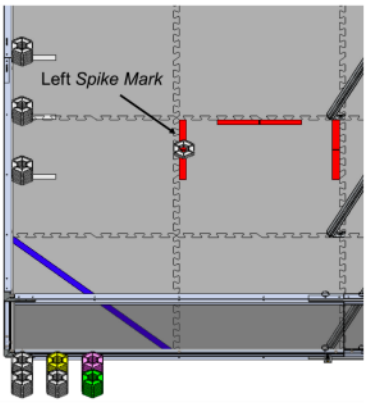
2. White Pixel or Team Prop is placed on Spike Mark in zone 1, 2, or 3

3. Autonomous Starts - Robot uses Vision or Sensors to detect which zone has been selected. (Tensor Flow for Pixel, Tensor Flow or Touch Sensor for Team Prop)

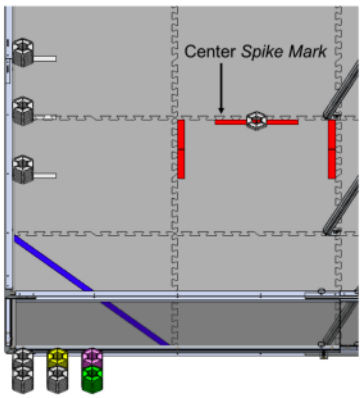
4. Robot places a Purple Pixel on Zone 1, 2, or 3 on the Spike Mark (10 or 20 points)

5. Robot places a yellow pixel on the backdrop Zone 1, 2, or 3 (10 or 20 points)

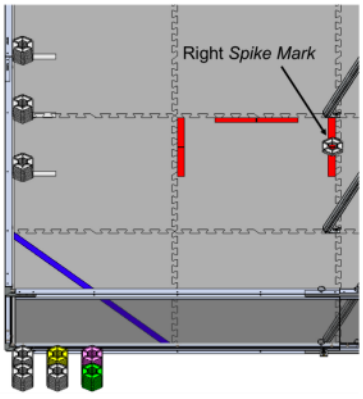
6. Robot parks in backstage area.



Zone 1



Zone 2



Zone 3

Stage 1: Write Autonomous Methods for Robot to place Purple Pixel on Spike Marks for Zone 1, 2, or 3

This process programs the actions for step 4 above. These steps assume a robot with an arm and two servos that hold the pixel in a claw type mechanism.

1. Place the Robot on either Red Left or Blue Right.

2. Turn on Robot and find OnBot Java on the programming computer.

3. Find the Op Mode Auto_Test_2023.java (Below REACH Add Functions Here)

4. In the fields area of Auto_Test_2023.java, define a private integer field called zone: (We will use this later) (Line 16 in sample code below)

```
 7
 8   @Autonomous(name="Auto_Test_2023", group="Training")
 9   //@Disabled
10   public class Auto_Test_2023 extends LinearOpMode {
11
12       /* Declare OpMode members. */
13       MaristBaseRobot2023 robot   = new MaristBaseRobot2023();
14       private ElapsedTime runtime = new ElapsedTime();
15
16       private int zone = 1;
17
```

4. Define three method stubs as shown below. (Lines 61 to 73)

```
59
60       // REACH: Add Functions Here
61       // Methods for Autonmous Zones
62
63       public void zoneOne() {
64
65       }
66
67       public void zoneTwo() {
68
69       }
70
71       public void zoneThree() {
72
73       }
74
```

5. Around line 39 in the runOpMode() method and write the following conditionals to act as a switch: (Lines 39 to 48 below)

```
38
39          // If statements to choose autonomous routine
40          if (zone == 1) {
41              zoneOne();
42          }
43          else if (zone == 2) {
44              zoneTwo();
45          }
46          else {
47              zoneThree();
48          }
49
```

6. Go back to the zoneOne() method. The sequence for placing the purple pixel could be the following:

- Close the grasper on the Purple Pixel
- Delay for about 1 second
- Raise the Arm a short distance
- Drive forward into the Spike Mark Area
- Turn Left about 45 degrees
- Lower the Arm to the floor
- Open the grasper
- Delay for about 1 second
- Raise Arm a short distance
- Drive backward about 6 inches

Commands available to move Servos and Arm are:

```
robot.leftHand.setPosition(position);
// Position is double between 0 and 1 inclusive

robot.rightHand.setPosition(position);
// Position is double between 0 and 1 inclusive
robot.leftArmMotorDeg(power, degrees, wait); //

robot.move(distance, power);

robot.turnLeft(distance, power);

robot.turnRight(distance, power);

delay(seconds);
```

Code sample for zoneOne() is below. (Note that you will have to define your own values for servo positions and arm angle due to robot differences).

```
70
71      public void zoneOne() {
72          // Close the Servos
73          robot.leftHand.setPosition(0.6); // Close
74          robot.rightHand.setPosition(0.3);
75          delay(2);
76
77          // Lift Arm
78          robot.leftArmMotorDeg(0.3, 45, 5);
79
80          // Move and Turn Left, Move Again
81          robot.move(24, 0.3);
82          robot.turnLeft(45, 0.3);
83          robot.move(6, 0.3);
84
85          // Arm Down and Release
86          robot.leftArmMotorDeg(0.3, -40, 5);
87          robot.leftHand.setPosition(0); // Open
88          robot.rightHand.setPosition(0.8);
89          delay(2);
90
91          // Backup
92          robot.move(-6, 0.3);
93      }
94
```

7. Code and test zoneOne() on the field. Try coding a few lines at a time and then testing parts of the zoneOne() autonomous. Once the entire sequence is finished. Test several times and make sure you note where the robot starts.

8. Once zoneOne() is finished, change private int zone to 2 as shown below:

```
11
12        /* Declare OpMode members. */
13        MaristBaseRobot2023 robot   = new MaristBaseRobot2023();
14        private ElapsedTime runtime = new ElapsedTime();
15
16        private int zone = 2;
17
```

9. Now write the routine for zoneTwo() method. It will almost be the same as zoneOne() except the robot will not need to turn.

10. After the zoneTwo() method is finished. Test several times to make sure the purple Pixel is placed properly. Again, zoneTwo() will be very similar to zoneOne() except for the turn.

11. Change the private int zone to 3 and write the code for the zoneThree() method. Test for consistency.