Discount Robotics - 421D - 2024 VEX VR Online Challenge

Names of Team Members: Senuka, Alex T., Michael, Ella, Alex B., Alex V., Rian, Prithvi

Team Number: 421D

Location of Team: Cincinnati, Ohio, USA

Description: In this VEX VR project, we've engineered an autonomous program that strategically navigates, intakes, and scores balls using modular functions like preload1, triball sequences, and grabPushAndScore. Central to our approach is the innovative use of Optical and Inertial Sensors, ensuring precise ball handling by monitoring brightness levels for intake and release actions. Our matchloadSequence demonstrates advanced programming with a looping mechanism that adapts travel distances and headings to optimize scoring efficiency. This project has significantly improved our proficiency in utilizing sensors, implementing modular code, and applying strategic programming to complex robotics challenges like this one.

Code:

```python
# Required imports and VEXcode VR robot configuration
import math
import random
from vexcode_vrc import *
from vexcode_vrc.events import get_Task_func

# Brain initialization for VEXcode VR
brain = Brain()

# Drivetrain and other component initialization
drivetrain = Drivetrain("drivetrain", 0)
arm_motor = Motor("ArmMotor", 3)
rotation = Rotation("Rotation", 7)
intake_motor = Motor("IntakeMotor", 8)
optical = Optical("Optical", 11)
```

```
# Setup function to configure initial motor velocities
def setup():
    drivetrain.set_drive_velocity(100, PERCENT)
    drivetrain.set_turn_velocity(100, PERCENT)
    intake_motor.set_velocity(100, PERCENT)
    arm_motor.set_velocity(100, PERCENT)


Brightness_Intake_Threshold = 5  # Constant for optical sensor threshold

# Function to intake a Triball by spinning the intake motor forward and use of optical
sensor
def IntakeTriball():
    intake_motor.spin(FORWARD)

    # Wait until a ball is detected based on brightness
    while optical.brightness() < Brightness_Intake_Threshold:
        wait(5, MSEC)  # Small delay to prevent tight looping

    intake_motor.stop()

# Function to drop a Triball by spinning the intake motor in reverse and use of
optical sensor
def DropTriball():
    intake_motor.spin(REVERSE)

    # Wait until the ball is dropped based on brightness
    while optical.brightness() > Brightness_Intake_Threshold:
        wait(5, MSEC)  # Small delay to prevent tight looping

    intake_motor.stop()

# Sequence of actions to release a preloaded ball
def preload1():
    arm_motor.spin(FORWARD) # Initial arm motor spin to position
    wait(0.23, SECONDS)
    drivetrain.drive_for(FORWARD, 1200, MM)
    drivetrain.turn_to_heading(45, DEGREES)
    DropTriball()

# Sequence of actions to intake and score a second preload ball
```

```python
def preload2():
   drivetrain.turn_to_heading(0, DEGREES)
   drivetrain.drive_for(FORWARD, 150, MM)
   IntakeTriball()
   drivetrain.drive_for(FORWARD, 550, MM)
   drivetrain.turn_to_heading(90, DEGREES)
   DropTriball()

# Function to intake and score the first triball
def triball1():
   drivetrain.turn_to_heading(0, DEGREES)
   drivetrain.drive_for(FORWARD, 80, MM)
   drivetrain.turn_to_heading(270, DEGREES)
   IntakeTriball()
   drivetrain.drive_for(FORWARD, 1700, MM)
   DropTriball()

# Function to intake and score the second triball
def triball2():
   drivetrain.turn_to_heading(60, DEGREES)
   IntakeTriball()
   drivetrain.drive_for(FORWARD, 820, MM)
   drivetrain.drive_for(REVERSE, 820, MM)
   drivetrain.turn_to_heading(270, DEGREES)
   DropTriball()

# Function to intake and score the third triball
def triball3():
   drivetrain.turn_to_heading(0, DEGREES)
   drivetrain.drive_for(REVERSE, 550, MM)
   drivetrain.turn_to_heading(90, DEGREES)
   IntakeTriball()
   drivetrain.drive_for(FORWARD, 650, MM)
   drivetrain.drive_for(REVERSE, 200, MM)
   drivetrain.turn_to_heading(300, DEGREES)
   DropTriball()

# Function to intake and score the fourth triball
def triball4():
   drivetrain.turn_to_heading(90, DEGREES)
   IntakeTriball()
```

```
    drivetrain.drive_for(FORWARD, 800, MM)
    drivetrain.drive_for(REVERSE, 1200, MM)
    drivetrain.turn_to_heading(285, DEGREES)
    DropTriball()

# Function to intake and score the fifth triball
def triball5():
    drivetrain.turn_to_heading(123, DEGREES)
    IntakeTriball()
    drivetrain.drive_for(FORWARD, 775, MM)
    drivetrain.turn_to_heading(270, DEGREES)
    drivetrain.drive_for(FORWARD, 300, MM)
    DropTriball()

# Sequence to perform match loads involving driving and reversing distances
def matchloadSequence():
    # Initialize variables for the for loop within matchloadSequence function
    forward_distance = 1450
    reverse_distance = 1550
    loop_heading = 270

    drivetrain.drive_for(REVERSE, 800, MM)

    # Loop to execute match load actions four times
    for i in range(4):
        drivetrain.turn_to_heading(135, DEGREES)
        IntakeTriball()
        drivetrain.drive_for(FORWARD, forward_distance, MM)

        # Increments forward and reverse distances by 100 to account for loop
discrepancies
        forward_distance += 100
        drivetrain.drive_for(REVERSE, reverse_distance, MM)
        reverse_distance += 100

        drivetrain.turn_to_heading(loop_heading, DEGREES)
        loop_heading += 7 # Increments heading to prevent balls blocking each other
        intake_motor.spin(REVERSE)
        drivetrain.drive_for(FORWARD, 585, MM)
        drivetrain.drive_for(REVERSE, 585, MM)
```

```python
# Sequence to grab a ball in the corner, push another ball, and score
def grabPushAndScore():
    drivetrain.turn_to_heading(135, DEGREES)
    IntakeTriball()
    drivetrain.drive_for(FORWARD, 1950, MM)
    drivetrain.turn_to_heading(270, DEGREES)
    drivetrain.turn_to_heading(270, DEGREES)
    drivetrain.drive_for(FORWARD, 2400, MM)
    drivetrain.turn_to_heading(325, DEGREES)
    drivetrain.drive_for(FORWARD, 400, MM)
    DropTriball()

# Function to score the last ball
def lastBall():
    drivetrain.turn_to_heading(215, DEGREES)
    IntakeTriball()
    drivetrain.drive_for(FORWARD, 450, MM)
    drivetrain.drive_for(REVERSE, 450, MM)
    drivetrain.turn_to_heading(330, DEGREES)
    DropTriball()

# Main function where the robot's autonomous actions are executed
def main():

    # Functions are called in order
    setup()
    preload1()
    preload2()
    triball1()
    triball2()
    triball3()
    triball4()
    triball5()
    matchloadSequence()
    grabPushAndScore()
    lastBall()

vr_thread(main) # Main function calling
```