CamJam EduKit Robotics - Obstacle Avoidance

Project Obstacle Avoidance

Description You will learn how to stop your robot from bumping into things.

Equipment Required

For this worksheet you will require:

- Your robot with the ultrasonic sensor attached to the front (see Worksheet 6).

How it Works

In Worksheet 6 you learnt how to detect how far in front of your robot an object was using the ultrasonic sensor. In this worksheet you are going to get the robot to avoid driving into obstacles.

Here is the logic for how the robot will work in this worksheet:

- The robot will drive forwards until it detects that there is an object a few centimetres in front of it.
- The robot will reverse and then turn right.
- The robot will then drive forwards again.

Code

The code will be based on the code you completed at the end of Worksheet 7 copy and add variables for the ultrasonic sensors' GPIO pins to the section where the motor GPIO pins are defined:

```
# Define GPIO pins to use for the distance sensor
pinTrigger = 17
pinEcho = 18
```

And add the distance sensor setup line to the robot setup line:

```
sensor = DistanceSensor(echo=pinEcho, trigger=pinTrigger)
```

Add variables to set how close to an obstacle the robot can go, how long to reverse for and how long to turn right for. You should change these variables to suit your robot.

```
# Distance Variables
hownear = 15.0
reversetime = 0.5
turntime = 0.75
```

You are going to add another function that will return whether the ultrasonic sensor sees an obstacle:

```
# Return True if the ultrasonic sensor sees an obstacle
def isnearobstacle(localhownear):
    distance = sensor.distance * 100

    print("IsNearObstacle: " + str(distance))
    if distance < localhownear:
        return True
    else:
        return False
```

You also need to write a function that will make the robot avoid the obstacle by moving back a little, then turning right.

```
# Move back a little, then turn right
def avoidobstacle():
    # Back off a little
    print("Backwards")
    robot.value = motorbackward
    time.sleep(reversetime)
    robot.stop()

    # Turn right
    print("Right")
    robot.value = motorright
    time.sleep(turntime)
    robot.stop()
```

The code that controls the robot then goes at the end of the code.

```
try:
    # repeat the next indented block forever
    while True:
        robot.value = motorforward
        time.sleep(0.1)
        if isnearobstacle(hownear):
            robot.stop()
            avoidobstacle()

# If you press CTRL+C, cleanup and stop
except KeyboardInterrupt:
    robot.stop()
```

This code runs the commands within the `try:` section until the Ctrl+C keys are pressed.

Within the `try:` is a `while True:`. Since True is always true, the code within that while will run forever – or until Ctrl+C is pressed.

Within the `while True:`, the code makes the robot go forward until it is `HowNear` cm from an obstacle. When it is, the robot reverses and turns right.

Your code should now look like this:

```
# CamJam EduKit 3 - Robotics
# Worksheet 9 - Obstacle Avoidance

import time  # Import the Time library
from gpiozero import CamJamKitRobot, DistanceSensor  # Import the GPIO Zero
Libraries

# Define GPIO pins to use for the distance sensor
pintrigger = 17
pinecho = 18

robot = CamJamKitRobot()
sensor = DistanceSensor(echo=pinecho, trigger=pintrigger)
```

```
# Distance Variables
hownear = 15.0
reversetime = 0.5
turntime = 0.75

# Set the relative speeds of the two motors, between 0.0 and 1.0
leftmotorspeed = 0.5
rightmotorspeed = 0.5

motorforward = (leftmotorspeed, rightmotorspeed)
motorbackward = (-leftmotorspeed, -rightmotorspeed)
motorleft = (leftmotorspeed, 0)
motorright = (0, rightmotorspeed)


# Return True if the ultrasonic sensor sees an obstacle
def isnearobstacle(localhownear):
    distance = sensor.distance * 100

    print("IsNearObstacle: " + str(distance))
    if distance < localhownear:
        return True
    else:
        return False


# Move back a little, then turn right
def avoidobstacle():
    # Back off a little
    print("Backwards")
    robot.value = motorbackward
    time.sleep(reversetime)
    robot.stop()

    # Turn right
    print("Right")
    robot.value = motorright
    time.sleep(turntime)
    robot.stop()


# Your code to control the robot goes below this line
try:
    # repeat the next indented block forever
    while True:
        robot.value = motorforward
        time.sleep(0.1)
        if isnearobstacle(hownear):
            robot.stop()
```

```
            avoidobstacle()

# If you press CTRL+C, cleanup and stop
except KeyboardInterrupt:
    robot.stop()
```

Running the Code

Place your robot on the floor near a wall or other obstacles and run your code.

The robot should then approach the wall until it gets close. It should then reverse and turn right.

Tuning the Robot

As mentioned in previous worksheets, every robot is different. You should therefore experiment with the distance variables (`hownear`, `reversetime`, and `turntime`), as well as the speed of the robot, if necessary (`leftmotorspeed`, `rightmotorspeed`).

Challenge

Instead of just reversing and going right, make your robot turn left, measure the distance to the next object, then turn right and do the same and drive in the direction that has an obstacle furthest away.