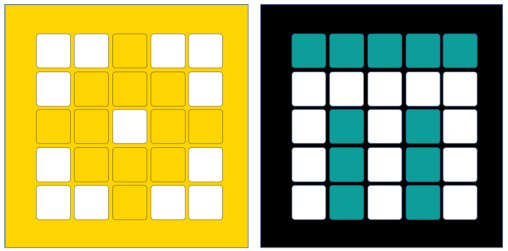
PRIME LESSONS

By the Makers of EV3Lessons



GYRO MOVE STRAIGHT

BY SANJAY AND ARVIND SESHAN

This lesson uses SPIKE 3 software

LESSON OBJECTIVES

Learn to apply proportional control to get your robot to move straight

Learn to apply proportional control to the Gyro sensor move at a particular angle

You must go through the Proportional Line Follower Lesson before you complete this lesson

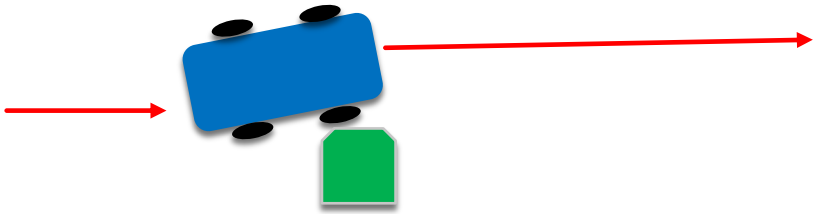
You must also complete the Turning With Gyro Lesson

WHAT IS GYRO MOVE STRAIGHT?

Imagine that you want to drive for 200 cm straight

As you travel, your robot gets bumped by something

A gyro move straight program helps the robot correct itself back to straight, but offset by how much it was bumped



HOW IT WORKS

A proportional line follower and a gyro move straight code share similar properties

To write a gyro move straight program, you must first think about what the error is and what the correction needs to be

Application	Objective	Error	Correction
Gyro Straight	Make the robot at a constant heading/angle	How far you are from that heading/angle	Turn sharper based on how far you are from that angle
Line Follower	Stay on the edge of the line	How far are our light readings from those at line edge (current_light – target_light)	Turn sharper based on distance from line

PSEUDOCODE

Set motor pair

Reset your yaw value to 0

In a loop, compute the error and apply the correction

- Part 1: Compute Error (How far from target angle)
  - To move straight: Target yaw angle=0 (Note: Assuming a horizontal hub placement, we must look at the yaw direction for the angle offset. This may be different for your setup)
  - Distance from target angle is just current yaw reading
- Part 2: Compute a Correction that is proportional to the error
  - Multiply the Error from Part 1 by a constant (that you must experiment and discover for your robot)
- Plug the value from Part 2 into a move block with each motor adjusted proportionally

Exit loop as required by changing loop block

NOTE: If your robot is built in a way that the Front of the hub is not the Front of the Robot, you may have to make some adjustments. See discussion guide item 2.

The code was tested on Drive Base 1.

SOLUTION: GYRO MOVE STRAIGHT

```
from hub import port, motion_sensor
import runloop, motor_pair

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    # Reset the yaw angle and wait for it to stabilize
    motion_sensor.reset_yaw(0)
    await runloop.until(motion_sensor.stable)
    while True:
        # compute the error in degrees. See Turning with Gyro for explanation.
        error = motion_sensor.tilt_angles()[0] * -0.1
        # correction is an integer which is the negative of the error
        correction = int(error * -2)
        # apply steering to correct the error
        motor_pair.move(motor_pair.PAIR_1, correction, velocity=200)

runloop.run(main())
```

DISCUSSION GUIDE

1. Compare the proportional line follower code with the proportional move straight code. What similarities and differences do you see?

   Ans. The code is almost the same. The one difference is how the error is calculated. The error is calculated using the gyro sensor. The correction is identical.

2. What if you wanted to travel at a particular angle (not just straight)? How would the code look different?

   Ans. In Part 1 of the solution code, there is no subtraction block because we were just subtracting "0" since our target heading is moving straight. You would have to subtract your current angle from the target angle if you wanted to move at some other angle.

Target angle = 5 degrees

```
error = motion_sensor.tilt_angles()[0] * -0.1 - 5
correction = int(error * -2)
```

CREDITS

This lesson was created by Sanjay Seshan and Arvind Seshan for Prime Lessons

Additional contributions by FLL Share & Learn community members.

More lessons are available at www.primelessons.org