Gyro Move Straight

By Sanjay and Arvind Seshan

ADVANCED EV3 PROGRAMMING LESSON
Lesson Objectives

- Learn what proportional control means and why to use it
- 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
- Prerequisites: Math Blocks, Data Wires, Proportional Control, Gyro Sensor lessons
Tips For Success

- You must go through the Proportional Control Lesson and the Proportional Line Follower Lesson before you complete this lesson.
- You must also complete the three Gyro Lessons (Introduction to Gyro Sensor, Gyro Turns and Gyro Revisited).

© 2019 EV3Lessons.com, Last edit 1/28/2019
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

<table>
<thead>
<tr>
<th>Application</th>
<th>Objective</th>
<th>Error</th>
<th>Correction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gyro Straight</td>
<td>Make the robot at a constant heading/angle</td>
<td>How far you are from that heading/angle</td>
<td>Turn sharper based on how far you are from that angle</td>
</tr>
<tr>
<td>Line Follower</td>
<td>Stay on the edge of the line</td>
<td>How far are our light readings from those at line edge (current_light – target_light)</td>
<td>Turn sharper based on distance from line</td>
</tr>
</tbody>
</table>
Recalibrate your gyro sensor (if you have not done so already elsewhere in your code) or reset your gyro (yellow gyro block set to reset mode) so that the value starts at “0” and there is no drift.

In a loop, compute the error and apply the correction:
- **Part 1: Compute Error (How far from target angle)**
  - To move straight → Target gyro angle=0
  - Distance from target angle is just current gyro 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 Steering Block

Exit loop as required.
Solution: Gyro Move Straight

Every proportional program must have 2 parts: Part 1 computes the error (in this case, how far you are from straight) and Part 2 computes a correction that is proportional to the error (in this case how much to turn). You can use proportional control with other sensors as well. It works really well!

Part 1: Compute the Error
- Our goal is to go straight (gyro sensor = 0). Therefore, our current gyro reading tells us how far you are from straight and is our error.
- Your robot is usually not more than 30 degrees from straight - so that is a reasonable worst-case error.

Part 2: Computes and Apply the Correction
- We multiply the Error from Part 1 by 0.7 to determine the turn value.
- We picked 0.7 so that when we have the worst case error of 30 or 30, the steering in the Move Block above will be 21 or -21 which is a gradual turn.
- You can adjust this value to make your move straight meet your needs.
- Note you may need to make this negative 0.7 if your motors are connected differently (or swap the motors in the move steering block).

Note: You don't need to use a Constant Block with a data wire. We just did that so it would be more obvious that we multiplied by a constant of our choice.
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.
This tutorial was created by Sanjay Seshan and Arvind Seshan

More lessons at www.ev3lessons.com

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.