ADVANCED EV3 PROGRAMMING LESSON

Using the Gyro Sensor and Dealing with Drift

By Sanjay and Arvind Seshan

Lesson Objectives

1. Learn what the Gyro Sensor does

2. Learn about 2 common problems with using the gyro sensor (drift and lag)

3. Learn what "drift" means

4. Learn how to correct for drift with a gyro "calibration" technique

5. Understand why it is important to consider multiple solutions to a problem such as gyro drift

Prerequisites: Data wires, Loops, Logic & Comparison Blocks

Update: Please read the "Gyro Sensor Revisited" lesson for important updates (8/6/2017) after you complete this lesson.

What is the Gyro Sensor?

- ↗ Gyro sensor detects rotational motion

- ↗ The sensor measures the rate of rotation in degrees per second (rate)

- ↗ It also keeps track of the total rotational angle and therefore lets you measure how far your robot has turned (angle)

- ↗ The accuracy of the sensor is ±3 degrees for 90 degree turn

Gyro Sensor Problems

↗ There are 2 common Gyro issues – drift and lag

    ↗ Drift – readings keep changing even when the robot is still

    ↗ Lag – readings are delayed

↗ In this lesson, we focus on the first problem: drift.

    ↗ We will cover lag in the Gyro Turn lesson

↗ Solution to drift: gyro calibration

    ↗ The source of the drift problem is that the gyro must "learn" what is still.

    ↗ For a color sensor, you have to "teach" the robot what is black and white

    ↗ For your gyro, you need to calibrate the sensor to understand what is "still"

Gyro Calibration to Solve Problem 1: Lag

- The gyro auto-calibrates when the robot is turned on or the gyro wire is connected. If the robot is moving during calibration, the gyro "learns" the wrong value for "still" – this causes drift!

- Unfortunately, there is no gyro calibration block. There a few ways to make the sensor recalibrate.

IMPORTANT NOTES

- The below are critical notes for using the gyro correctly!!!!!

- THE ROBOT MUST BE STILL WHEN YOU RUN ANY OF THESE CALIBRATION PROGRAMS!!!!

- JUST LIKE THE COLOR CALIBRATION, YOU SHOULDN'T RUN THIS EVERY TIME YOU NEED TO READ THE GYRO. YOU SHOULD CALIBRATE IN A SEPARATE PROGRAM JUST BEFORE YOU RUN YOUR PROGRAM OR ONCE AT THE BEGINNING OF YOUR PROGRAM.

Calibration: Strategy 1

The gyro recalibrates when it switches modes. So, a "rate" reading followed by an "angle" reading calibrates the gyro.

Second, add a wait block to give the sensor a bit of time to fully reset. Our measurements show that 0.1 seconds is sufficient.
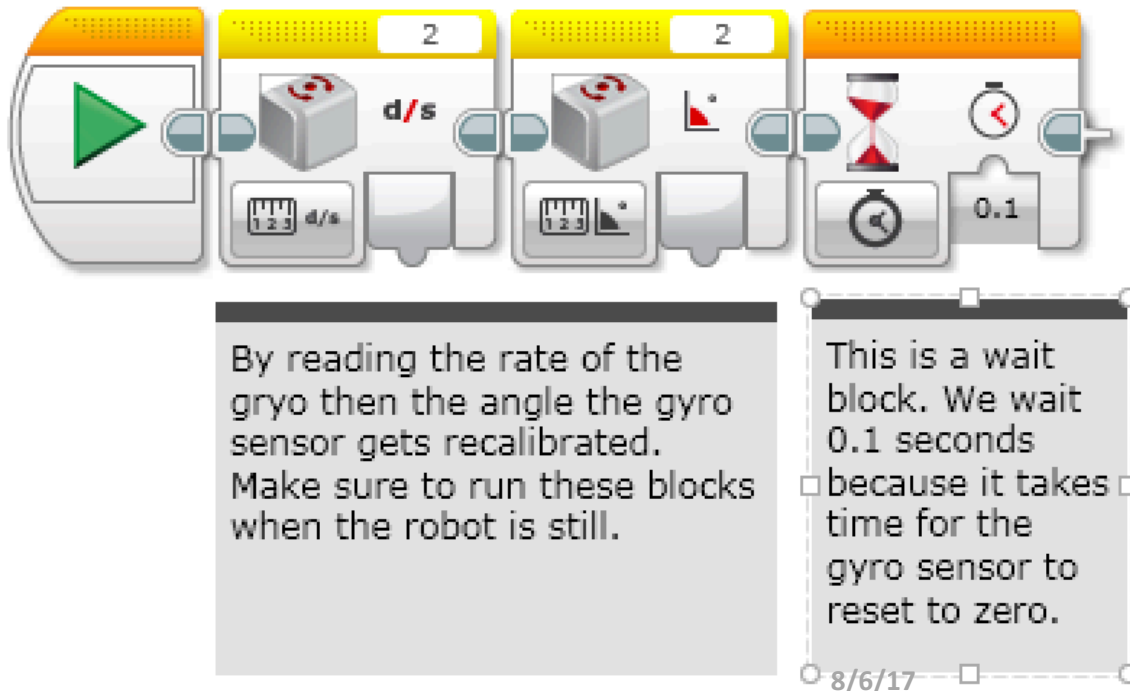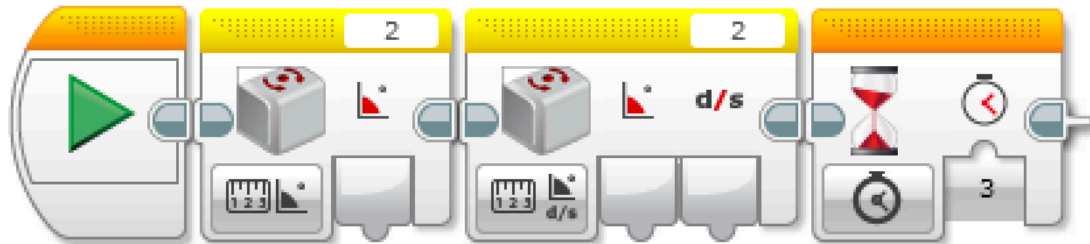

Note that in the rest of your program, you should only use the "angle" modes of the gyro. Using the "rate" or "rate and angle" mode will cause the gyro to recalibrate.

By reading the rate of the gryo then the angle the gyro sensor gets recalibrated. Make sure to run these blocks when the robot is still.

This is a wait block. We wait 0.1 seconds because it takes time for the gyro sensor to reset to zero.

8/6/17

7

Calibration: Strategy 2

This version of the calibration leaves the gyro in rate+angle mode. This is useful if you use the rate output.

The downside of this version is that it takes longer (about 3 seconds). Also, you cannot use gyro reset anymore!

Note that in the rest of your program, you should only use the "rate + angle" modes of the gyro. Using the "angle" or "rate" mode will cause the gyro to recalibrate. Also, ***DO NOT*** use the gyro reset - this forces the gyro into angle mode which will cause a long 3 second recalibration.

By reading the angle of the gryo then the rate+angle the gyro sensor gets recalibrated. Make sure to run these blocks when the robot is still.

This is a wait block. We wait 3 seconds because it takes time for the gyro sensor to reset to zero. This type of reset takes much longer
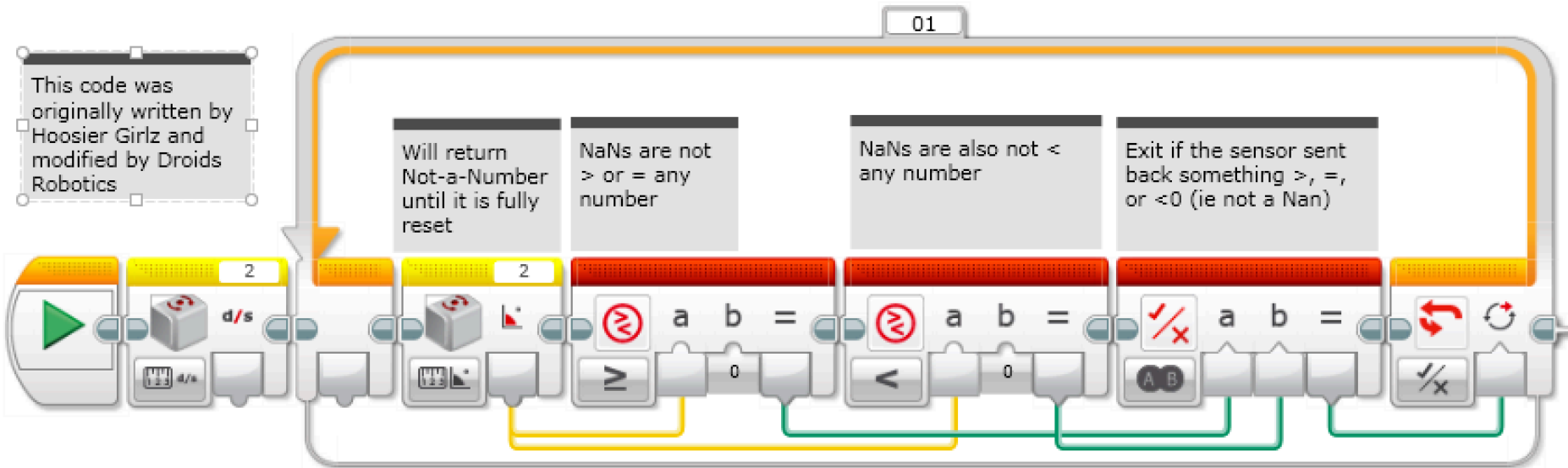
8/6/17

8

Strategy 3: Pseudocode

↗ Having a fixed time wait for the gyro to calibrate may not always work.

↗ The gyro returns "Not a Number" (NaN) until it has actually reset and NaNs are not >, =, or < any number. This is because they are not numbers

↗ The only way you can know when it is fully reset is to make sure you are getting back a real number, instead of a Not-a-Number value

   ↗ STEP 1: Recalibrate the gyro
   ↗ STEP 2: start a loop
   ↗ STEP 3: read angle
   ↗ STEP 4: check angle >= 0
   ↗ STEP 5: check angle < 0
   ↗ STEP 6: OR outputs of steps 4 & 5
   ↗ STEP 7: If the output of step 6 is true, exit loop

↗ At this point, the sensor drift should be gone.
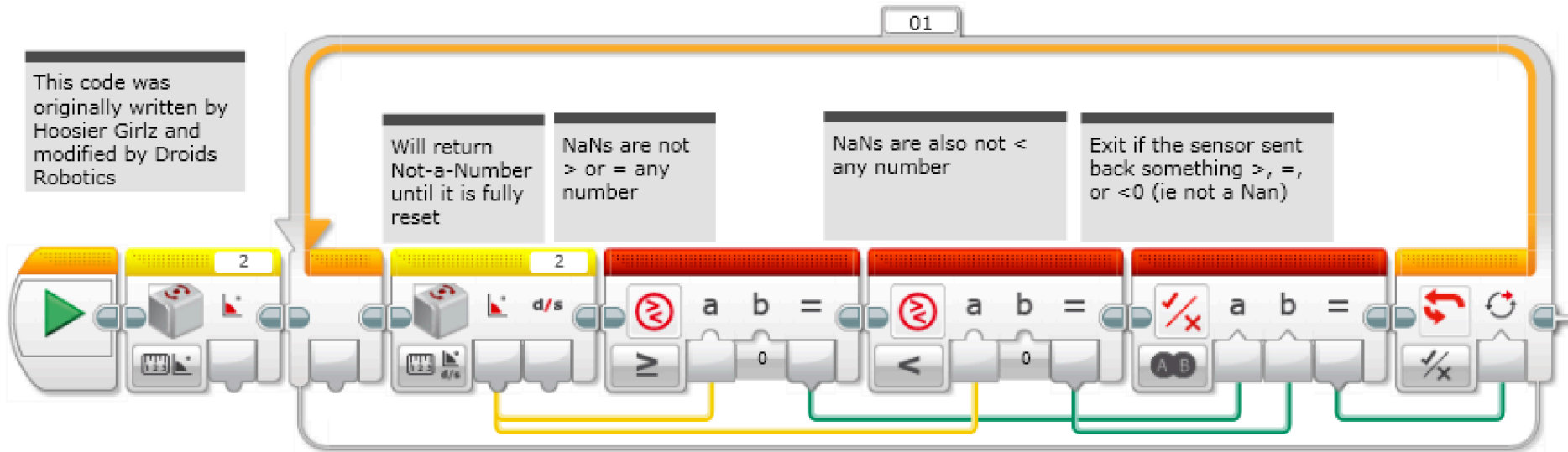
Strategy 3 Solution

This code was originally written by Hoosier Girlz and modified by Droids Robotics

Will return Not-a-Number until it is fully reset

NaNs are not > or = any number

NaNs are also not < any number

Exit if the sensor sent back something >, =, or <0 (ie not a Nan)

This version of the calibration leaves the gyro in angle mode. This is probably the most common way to use the gyro. This code takes about 0.1sec to run.

Note that in the rest of your program, you should only use the "angle" modes of the gyro. Using the "rate" or "rate and angle" mode will cause the gyro to recalibrate.

Strategy 4 Solution

This code was originally written by Hoosier Girlz and modified by Droids Robotics

Will return Not-a-Number until it is fully reset

NaNs are not > or = any number

NaNs are also not < any number

Exit if the sensor sent back something >, =, or <0 (ie not a Nan)

This version of the calibration leaves the gyro in rate+angle mode. This is useful if you use the rate output.

Note that in the rest of your program, you should only use the "rate + angle" modes of the gyro. Using the "angle" or "rate" mode will cause the gyro to recalibrate. Also, ***DO NOT*** use the gyro reset - this forces the gyro into angle mode which will cause a long 3 second recalibration.

Discussion Guide

1. **What are 2 common problems when programming with the gyro?**
   Ans. Gryo drift and Gyro lag

2. **What does Gyro drift mean?**
   Ans. The Gyro readings keep changing even when the robot is still

3. **Can you move your robot when you calibrate your gyro?**
   Ans. No!! Keep the robot still.

4. **Do you need to calibrate your gryo before every move?**
   Ans. No. Once before you run your entire program

5. **Why might it be important to consider multiple solutions to a problem?**
   Ans. In robotics, there are multiple ways to solve a problem and there might be tradeoffs between the solutions (e.g. how long the code takes to run the code, can you use both rate and angle readings?)

Update: Please read the "Gyro Sensor Revisited" lesson for important updates (8/6/2017)

Credits

↗ This tutorial was written by Sanjay Seshan and Arvind Seshan and uses code shared by Hoosier Girlz (http://www.fllhoosiergirlz.com)

↗ More lessons at www.ev3lessons.com