ADVANCED EV3 PROGRAMMING LESSON

Line Followers: Basic to PID

By Sanjay and Arvind Seshan

Lesson Objectives

↗ Evaluate and compare different line followers

↗ Prerequisites: Complete all Line Follower lessons on EV3Lessons.com, Calibration

↗ Videos will not play in PDF

Which Program Works Best for Which Situation?

Simple Line Follower
- Most basic line follower
- Wiggles a lot due to sharp turns
- Good for rookie teams → need to know loops and switches

Smooth Line Follower
- Almost the same as simple
- Turns are less sharp
- Has trouble on sharp curves
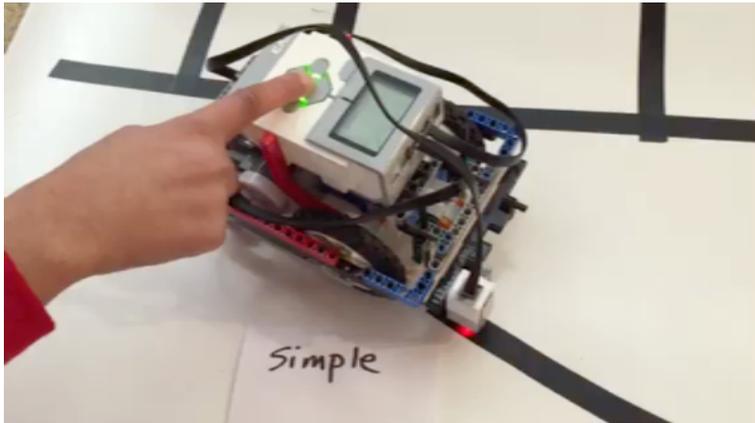- Good for rookie teams → need to know loops and switches

3-Stage Follower
- Best for straight lines
- Droids do not recommend this. Just learn the proportional line follower.
- Need to know nested switches

Proportional Follower
- Uses the "P" in PID
- Makes proportional turns
- Works well on both straight and curved lines
- Good for intermediate to advanced teams → need to know math blocks and data wires
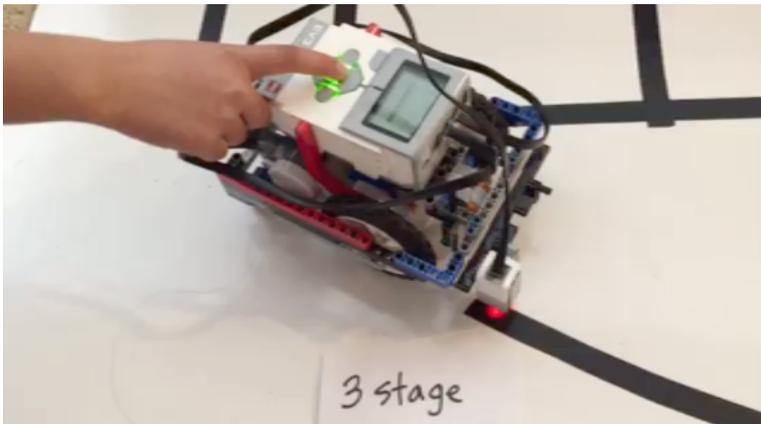
Watch the videos on the next 2 slides to see all four.
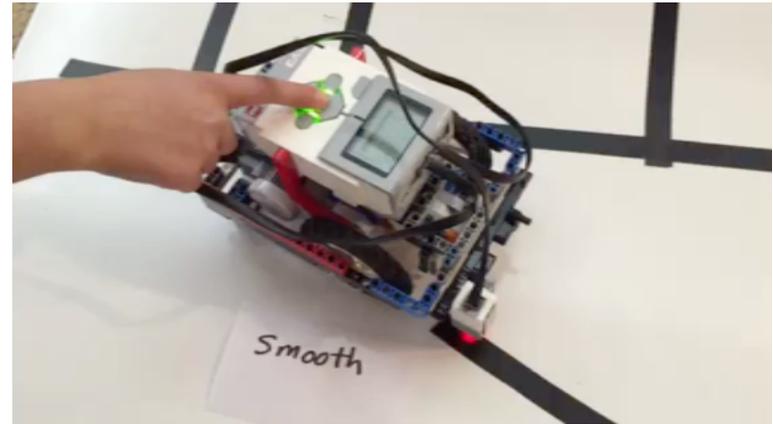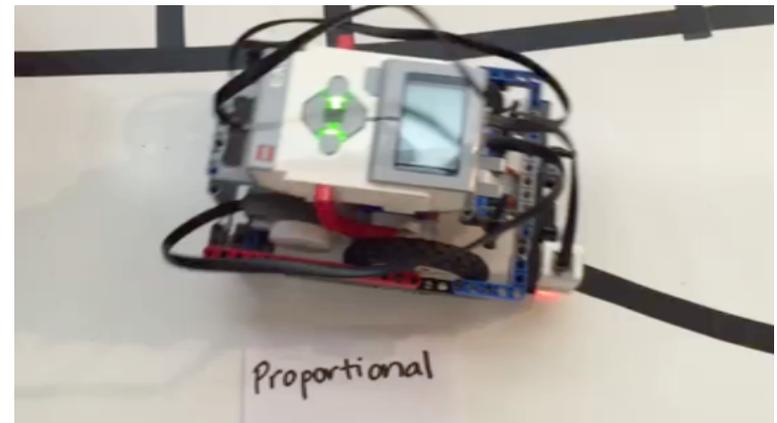
Curved Line: Watch Videos

Simple Line Follower

Simple

Smooth Line Follower

Smooth

3-Stage Follower

3 stage
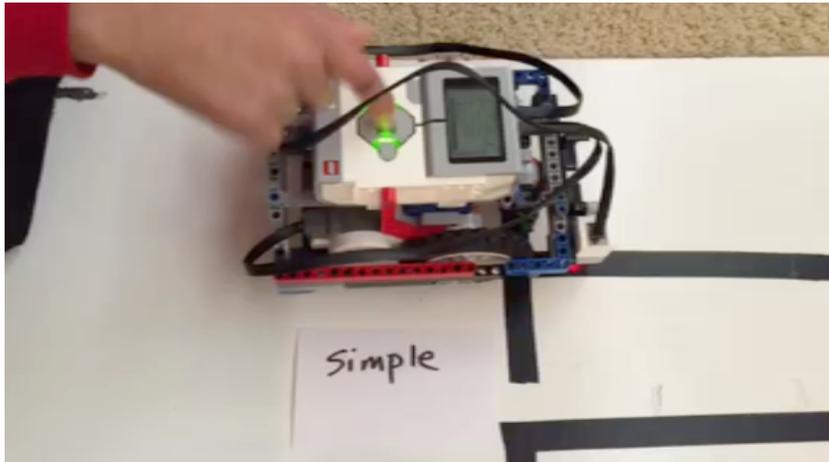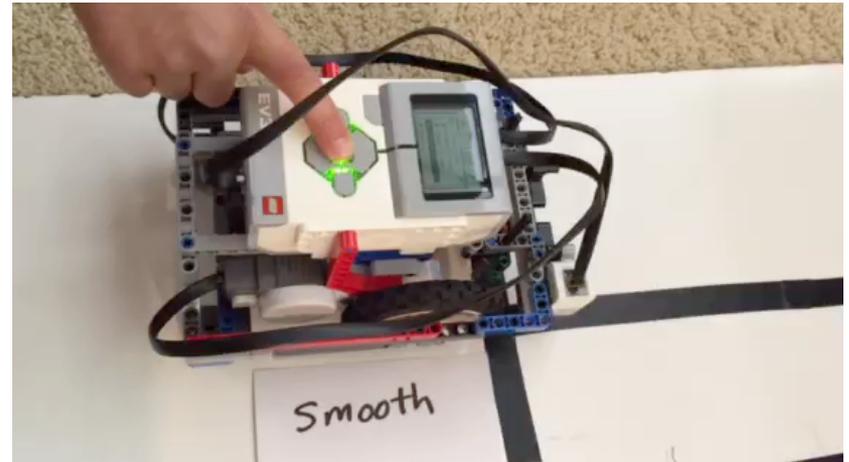
Proportional Follower

Proportional

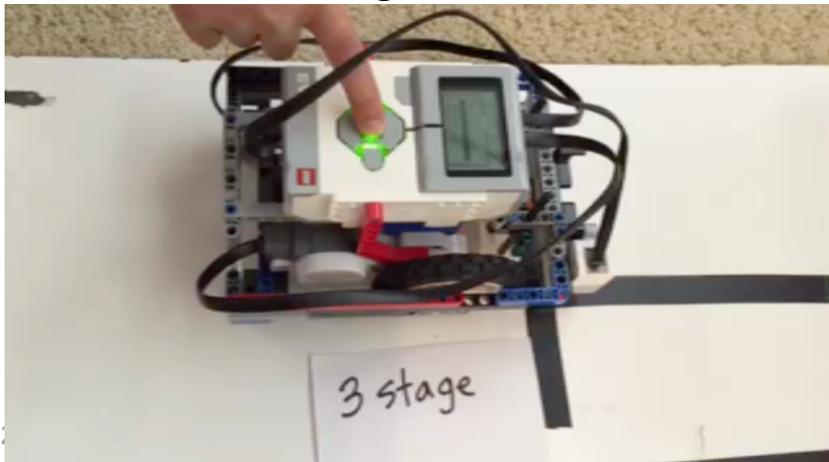# Straight Line: Watch Videos

Simple Line Follower

Smooth Line Follower

3-Stage Follower

Proportional Follower
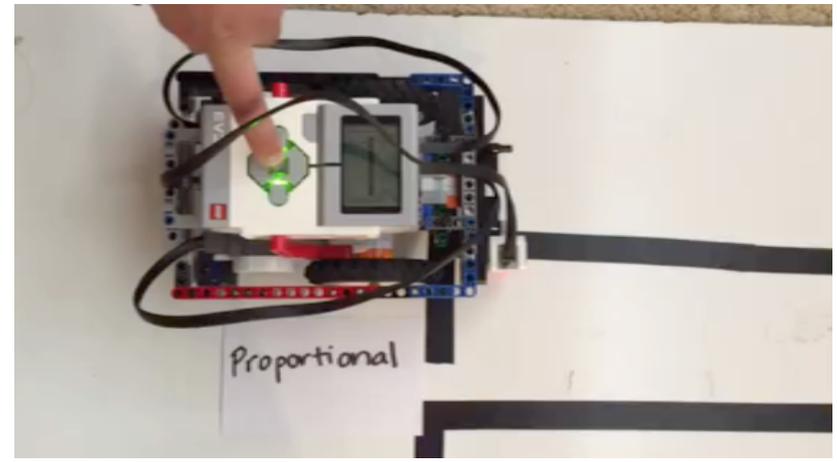
Watch Videos

Proportional Follower
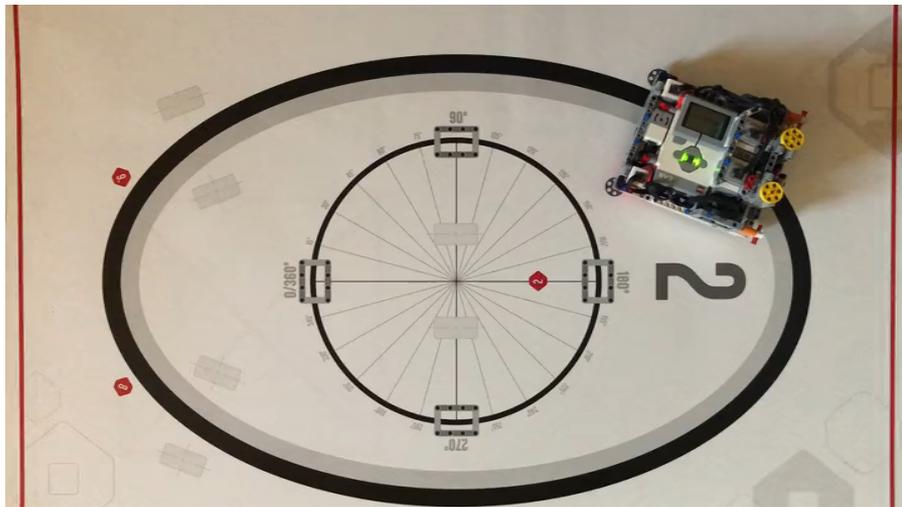
PID Follower
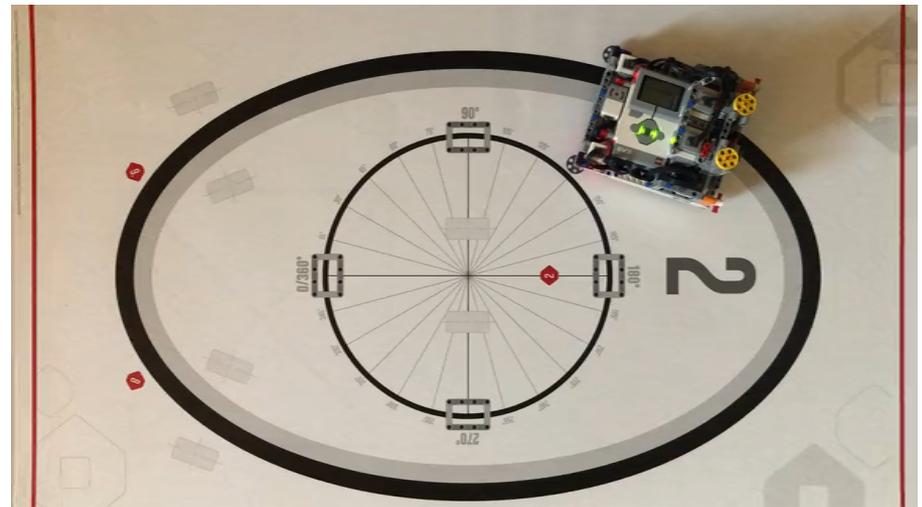
3 Line Follower Challenges

↗ Challenge 1: Can you write a simple line follower? Hint: Review Beginner: Basic Line Follower lesson

↗ Challenge 2: Can you write a smoother line follower? Hint: Change how sharp the turns are in a simple line follower.

↗ Challenge 3: Can you write a three-stage line follower where the robot moves different 3 different ways (left, right or straight) based on the reading from the color sensor?

A Note About Our Solutions

↗ CALIBRATE:

    ↗ The programs use the EV3 Color Sensor in Light Sensor mode

    ↗ You will have to calibrate your sensors.

    ↗ Please refer to Intermediate: Color Sensor Calibration Lesson

↗ PORTS:

    ↗ The Color Sensor is connected to Port 3.

    ↗ Please change this for your robot.

↗ WHICH SIDE OF THE LINE:

    ↗ Please take note of which side of the line the code is written for

Simple Line Follower

Simple Line Follower: The goal of this program is to create a very simple line following programming to follow the left side of a line. This is the most commonly taught program.

Note: This program uses the Color Sensors in Light Mode. This means that you will have to calibrate your sensors. Please read our calibration lessons before continuing! :-)

The switch checks if the robot is reading white or black.

If the robot reads white, turn right sharply.

If the robot reads black, turn left sharply.

This will run forever unless you press "OFF"!

You probably want to line follow only for a certain distance. You click on the infinite symbol on the loop to change when you want to exit the loop.
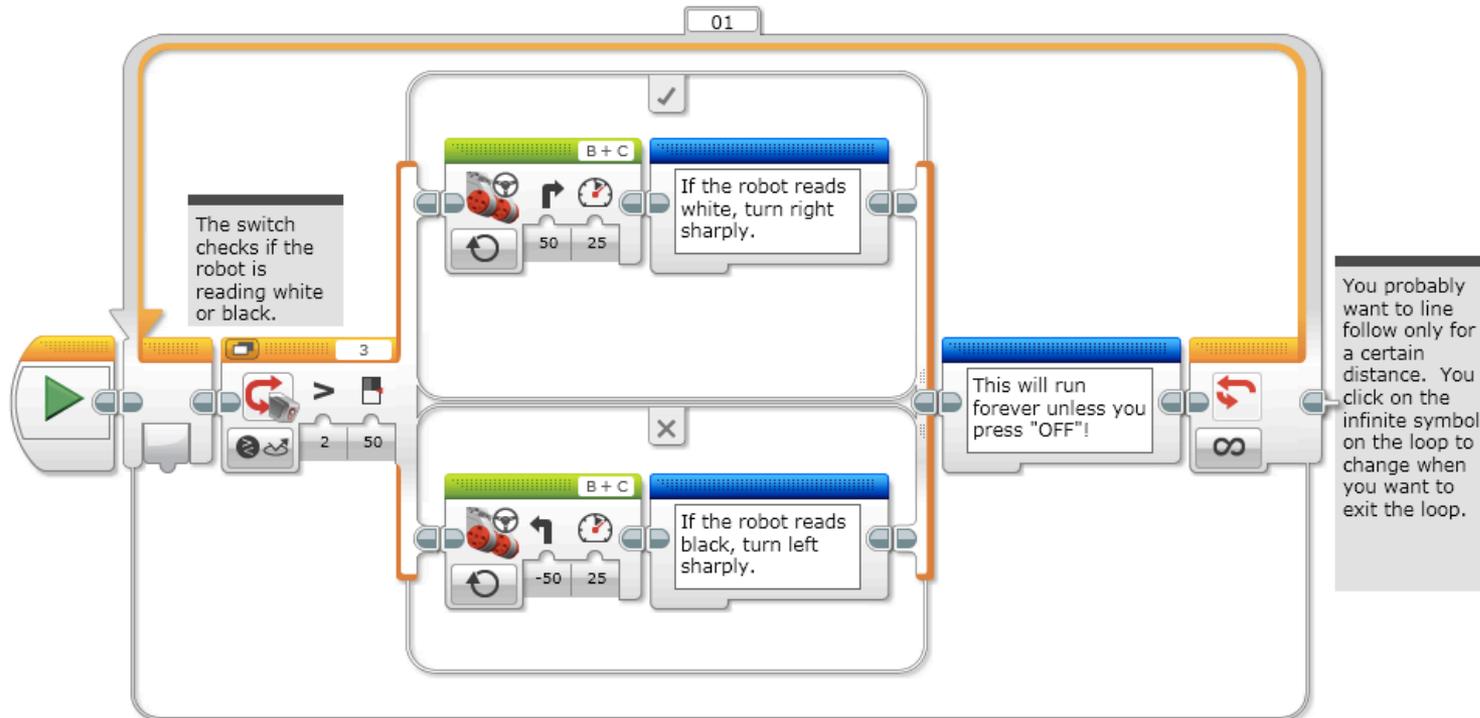
Smooth Line Follower

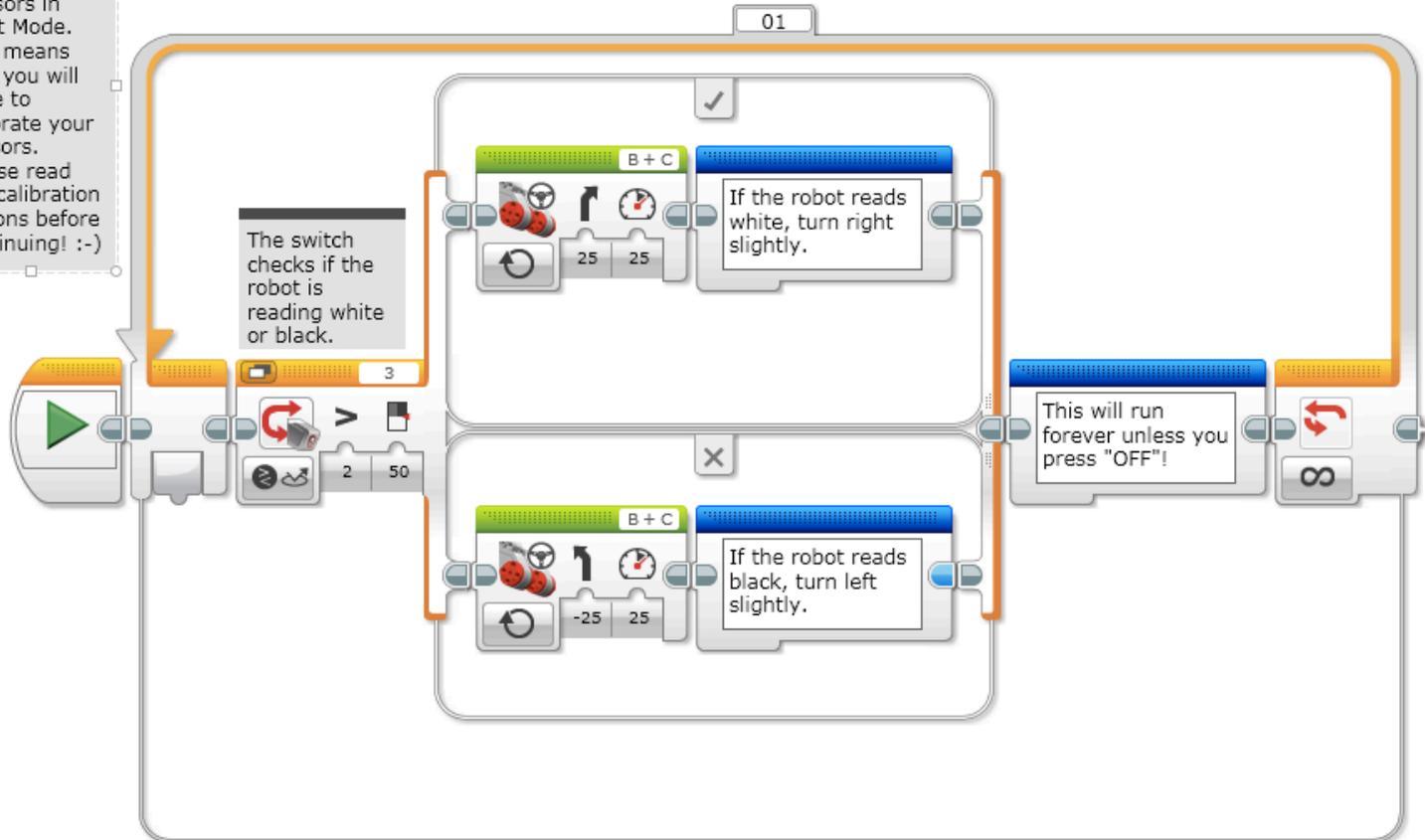Note: This program uses the Color Sensors in Light Mode. This means that you will have to calibrate your sensors. Please read our calibration lessons before continuing! :-)

Smooth Line Follower: The goal of this program is to create a simple line follower, but smoother than the first. This line follower will be smoother because it makes less sharp turns. The only difference between the Simple and the Smooth is the angle of the turns.

The switch checks if the robot is reading white or black.

If the robot reads white, turn right slightly.

If the robot reads black, turn left slightly.

This will run forever unless you press "OFF"!

You probably want to line follow only for a certain distance. You click on the infinite symbol on the loop to change when you want to exit the loop.

Three-Stage Line Follower

Note: We present this line follower because many teams talk about a multi-stage line follower and want to know how to write one. Our team recommends that you avoid this program and learn to make a proportional line follower!

Note: This program uses the Color Sensors in Light Mode. This means that you will have to calibrate your sensors. Please read our calibration lessons before continuing! :-)
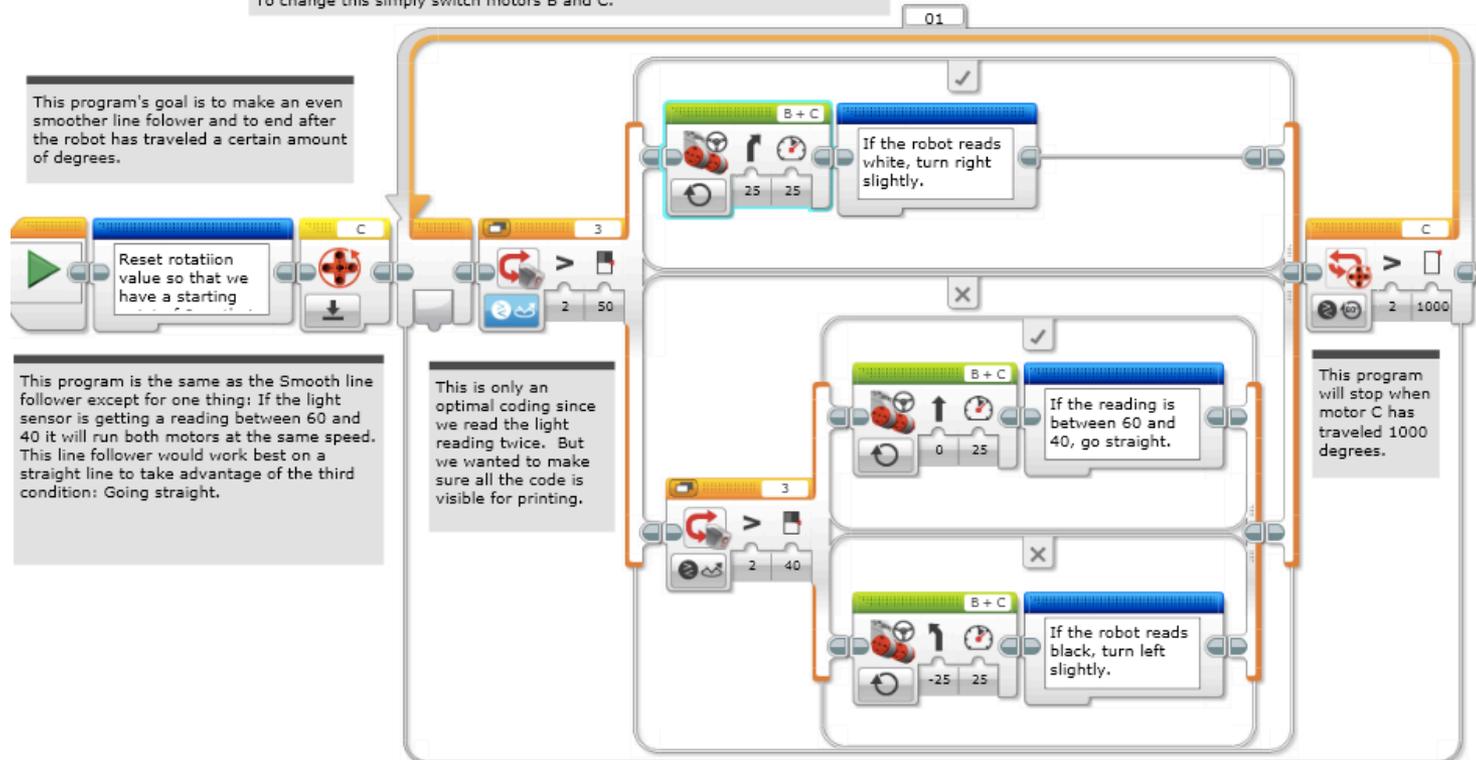
Note 1: If B is your right motor, this program will follow the left side of the line.
Note 2: If B is your left motor, this program will follow the right side of the line.
To change this simply switch motors B and C.

This program's goal is to make an even smoother line folower and to end after the robot has traveled a certain amount of degrees.

Reset rotatiion value so that we have a starting

If the robot reads white, turn right slightly.

If the reading is between 60 and 40, go straight.

If the robot reads black, turn left slightly.

This program will stop when motor C has traveled 1000 degrees.

This program is the same as the Smooth line follower except for one thing: If the light sensor is getting a reading between 60 and 40 it will run both motors at the same speed. This line follower would work best on a straight line to take advantage of the third condition: Going straight.

This is only an optimal coding since we read the light reading twice. But we wanted to make sure all the code is visible for printing.

Proportional Pseudocode

Can you write a proportional line follower that changes the angle of the turn depending on how far away from the line the robot is?

Pseudocode:

1. Reset the Rotation sensor (Only required for line following for a total distance)

2. Compute the error = Distance from line = (Light sensor reading – Target Reading)

3. Scale the error to determine a correction amount. Adjust your scaling factor to make you robot follow the line more smoothly.

4. Use the Correction value (computer in Step 3) to adjust the robot's turn towards the line.

Proportional Line Follower

Note: This program uses the color sensor in reflected light mode. You will need to calibrate your color sensor. If you do not know how to calibrate, please refer to our Calibration lesson.

Please refer to Proportional Control Lesson for more details
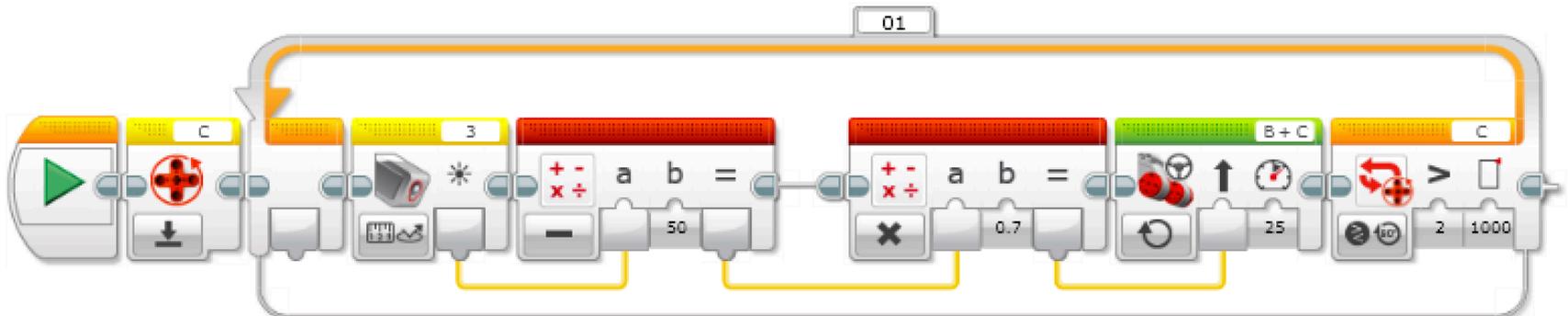
Reset the rotation sensor

Part 1: Compute the Error
Our goal is to stay at the edge of the line (light sensor = 50)

Part 2: Apply the correction
The error in part 1 is multiplied by a Constant of Proportionality (0.7). This will be different for each robot/application. See slides 9-11 to learn how to tune this number.

This line follower ends after 1000 degrees. Change this to suit your needs.

PID Pseudocode

1. Take a new light sensor reading

2. Compute the "error"

3. Scale error to determine contribution to steering update (proportional control)

4. Use error to update integral (sum of all past errors)

5. Scale integral to determine contribution to steering update (integral control)

6. Use error to update derivative (difference from last error)

7. Scale derivative to determine contribution to steering update (derivative control)

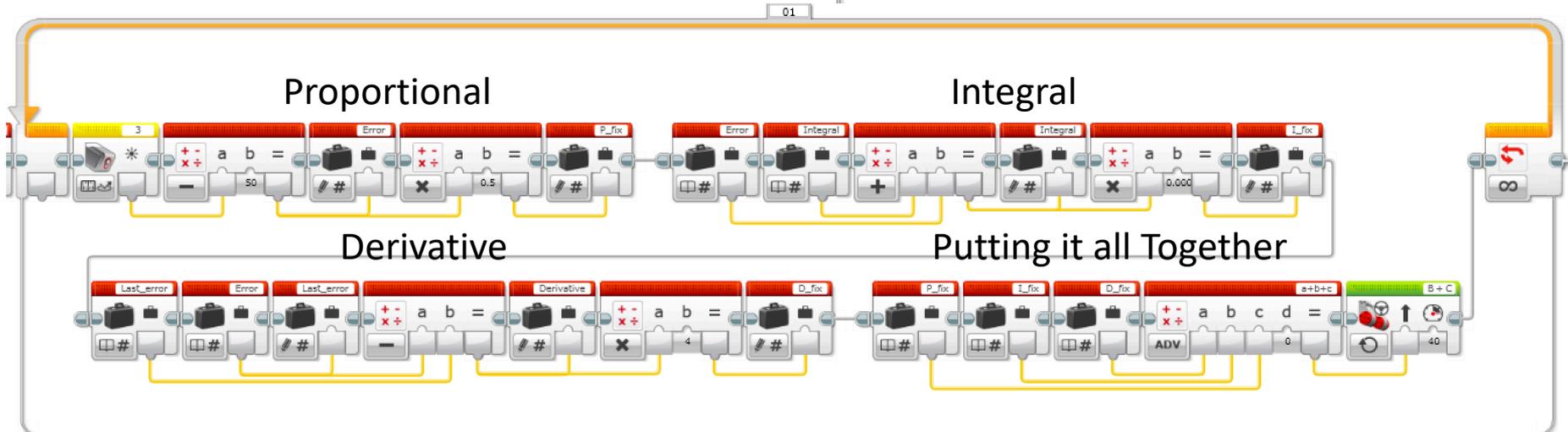8. Combine P, I, and D feedback and steer robot

PID Code

Set up the variables for the last error and integral before the loop and initialize to 0 because they are read before being written.
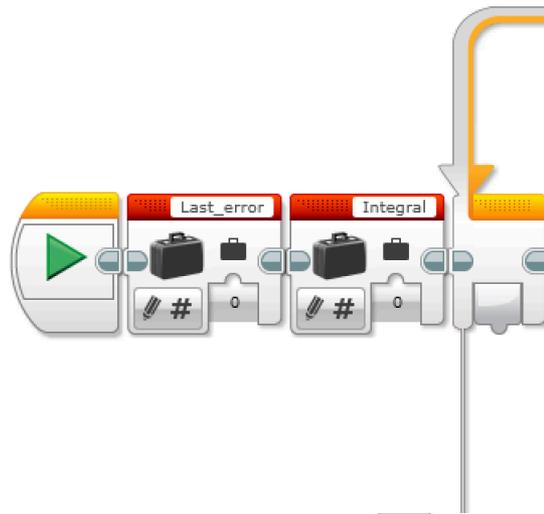
Refer to PID lesson for more details

Code has been split for readability. Continues below.

Last_error   Integral

01

Proportional

Integral

Derivative

Putting it all Together

Evaluating Line followers

Proportional

↗ Uses the "P" in PID

↗ Makes proportional turns

↗ Works well on both straight and curved lines

↗ Good for intermediate to advanced teams → need to know math blocks and data wires

PID

↗ It is better than proportional control on a very curved line, as the robot adapts to the curviness

↗ However, for FIRST LEGO League, which mostly has straight lines, proportional control can be sufficient

Credits

↗ This tutorial was created by Sanjay Seshan and Arvind Seshan

↗ More lessons at www.ev3lessons.com