

ADVANCED EV3 PROGRAMMING LESSON



Proportional Control

By Sanjay and Arvind Seshan



Lesson Objectives



- Learn what proportional control means and why to use it
- Learn to apply proportional control to different sensors
- Prerequisites: Math Blocks, Color Sensor Calibration, Data Wires

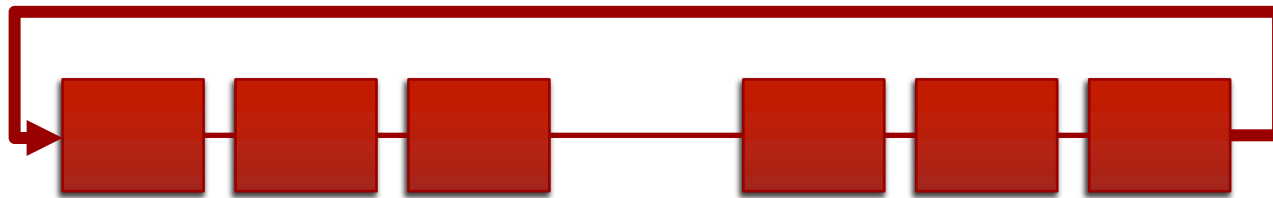
Learn and Discuss Proportional Control

- Let's start with a game
- Imagine that you blindfold one teammate. He or She has to get across the room as quickly as they can and stop exactly on a line drawn on the ground
- The rest of the team has to give the commands.
- When your teammate is far away, the blindfolded person must move fast and take big steps. But as he gets closer to the line, if he keeps running, he will overshoot. So, you have to tell the blindfolded teammate to go slower and take smaller steps.
- You have to program the robot in the same way!



What Proportional Control Looks Like

- The Pseudocode for every proportional control program consists of two stages:
 - Computing an error → how far is the robot from a target
 - Making a correction → make the robot take an action that is proportional to the error (this is why it is called proportional control). You must multiply the error by a scaling factor to determine the correction.



Compute Error

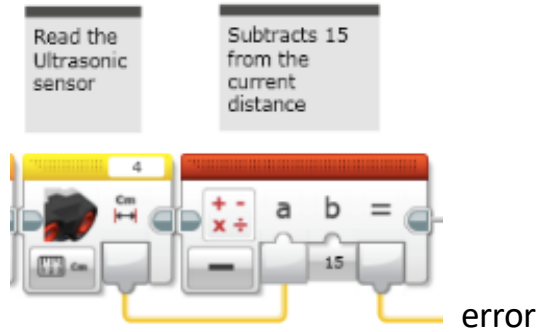
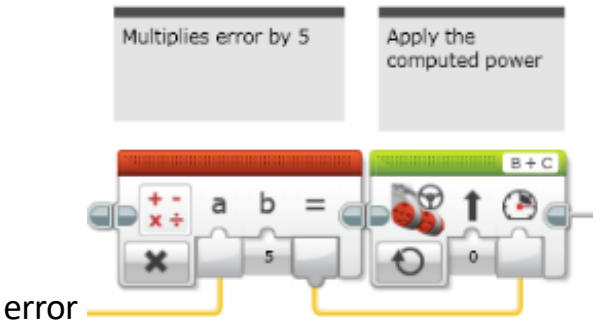
Make Correction

Challenge

- To learn how to use proportional control, create a Robot Follower program
 - Use proportional control with the ultrasonic sensor to get the robot to stay 15cm away from the human at all times (even when the human moves)

Objective	Error	Correction
Get to a target distance from human	How many cm from target location (current_distance – target_distance)	Move faster based on distance

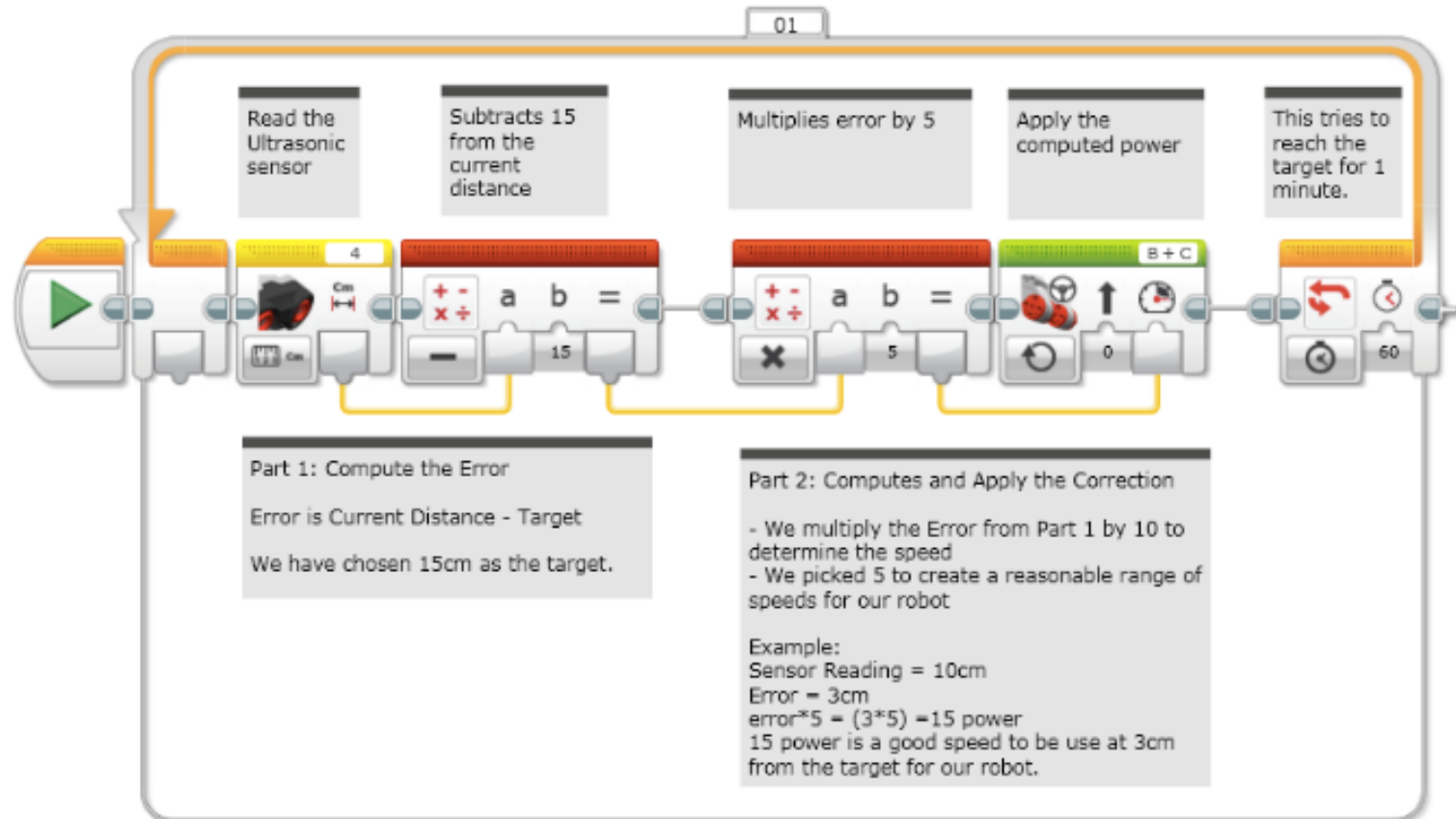
Challenge

Compute Error	
How many cm from target location (current_distance – target_distance)	
Compute/Apply Correction	
Move faster based on distance	

Putting It All Together: Ultrasonic Robot Follower

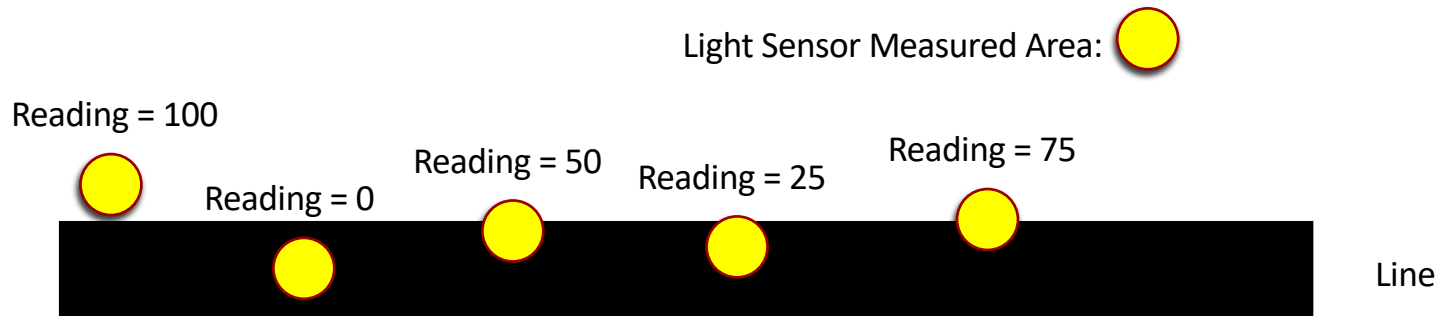
We are trying to make a program that stays 7cm away from a moving object. This program uses proportional control.

This code was written by Droids Robotics



How Far Is the Robot From The Line?

- Reflected light sensor readings show how “dark” the measured area is on average
- Calibrated readings should range from 100 (on just white) to 0 (on just black)



Line Following

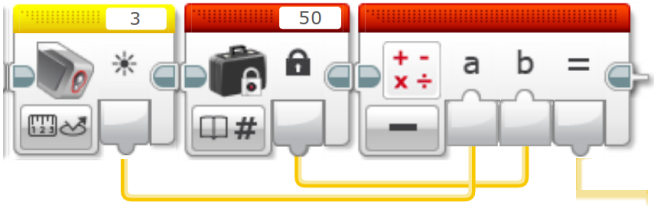
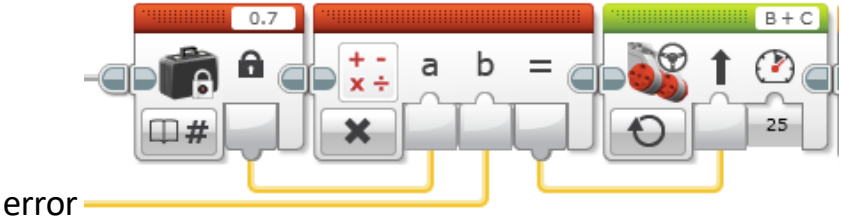
- **Computing an error** → how far is the robot from a target
 - Robots follow the edge of line → target should be a sensor reading of 50
 - Error should indicate how far the sensor's value is from a reading of 50
- **Making a correction** → make the robot take an action that is proportional to the error. You must multiply the error by a scaling factor to determine the correction.
 - To follow a line a robot must turn towards the edge of the line
 - The robot must turn more sharply if it is far from a line
 - How do you do this: You must adjust steering input on move block

How do you make a Proportional Line Follower?

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 (computed in Step 3) to adjust the robot's turn towards the line.

Challenge

Compute Error	 <p>The diagram shows a sequence of three Scratch code blocks connected by yellow lines. The first block is a 'Say' block with the text '3' and a duration of 2 seconds. The second block is a 'Wait' block with a duration of 50 seconds. The third block is a 'Math' block with the operation 'a - b' and an output field labeled 'error'.</p>
Compute/Apply Correction	 <p>The diagram shows a sequence of three Scratch code blocks connected by yellow lines. The first block is a 'Say' block with the text '0.7' and a duration of 2 seconds. The second block is a 'Math' block with the operation 'a * b' and an output field labeled 'error'. The third block is a 'Move' block with a distance of 'B + C' and a speed of 25.</p>

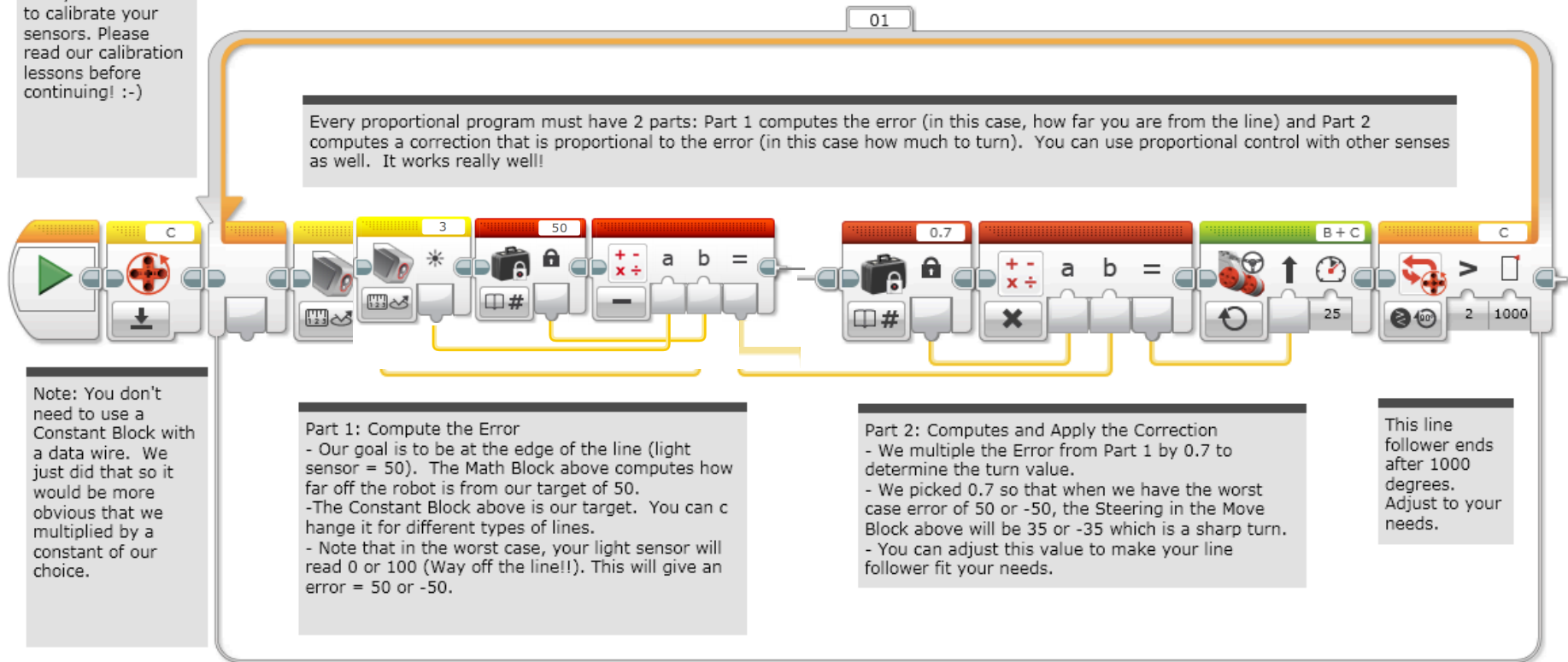
A Proportional Line Follower

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! :-)

We recommend that your team uses a proportional line follower like this one. It will be smoothest of the 4 line followers in this lesson. There are even better line followers (that use PID control), but a line follower that uses the "P" is a great start.

A proportional line follower changes the angle of the turn based on how far from the line the robot is.

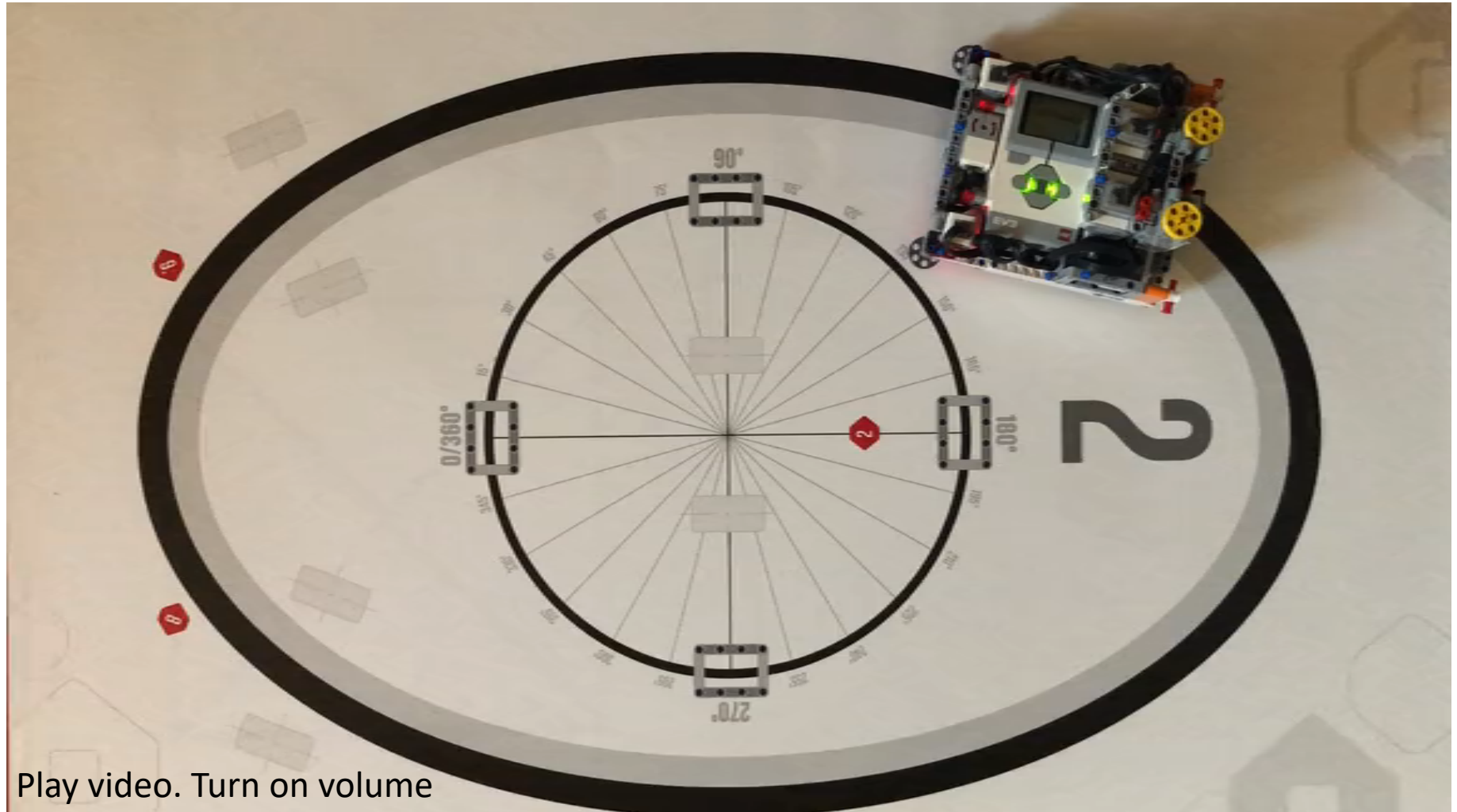
Every proportional program must have 2 parts: Part 1 computes the error (in this case, how far you are from the line) and Part 2 computes a correction that is proportional to the error (in this case how much to turn). You can use proportional control with other senses as well. It works really well!



Key Step: Tuning the Constant

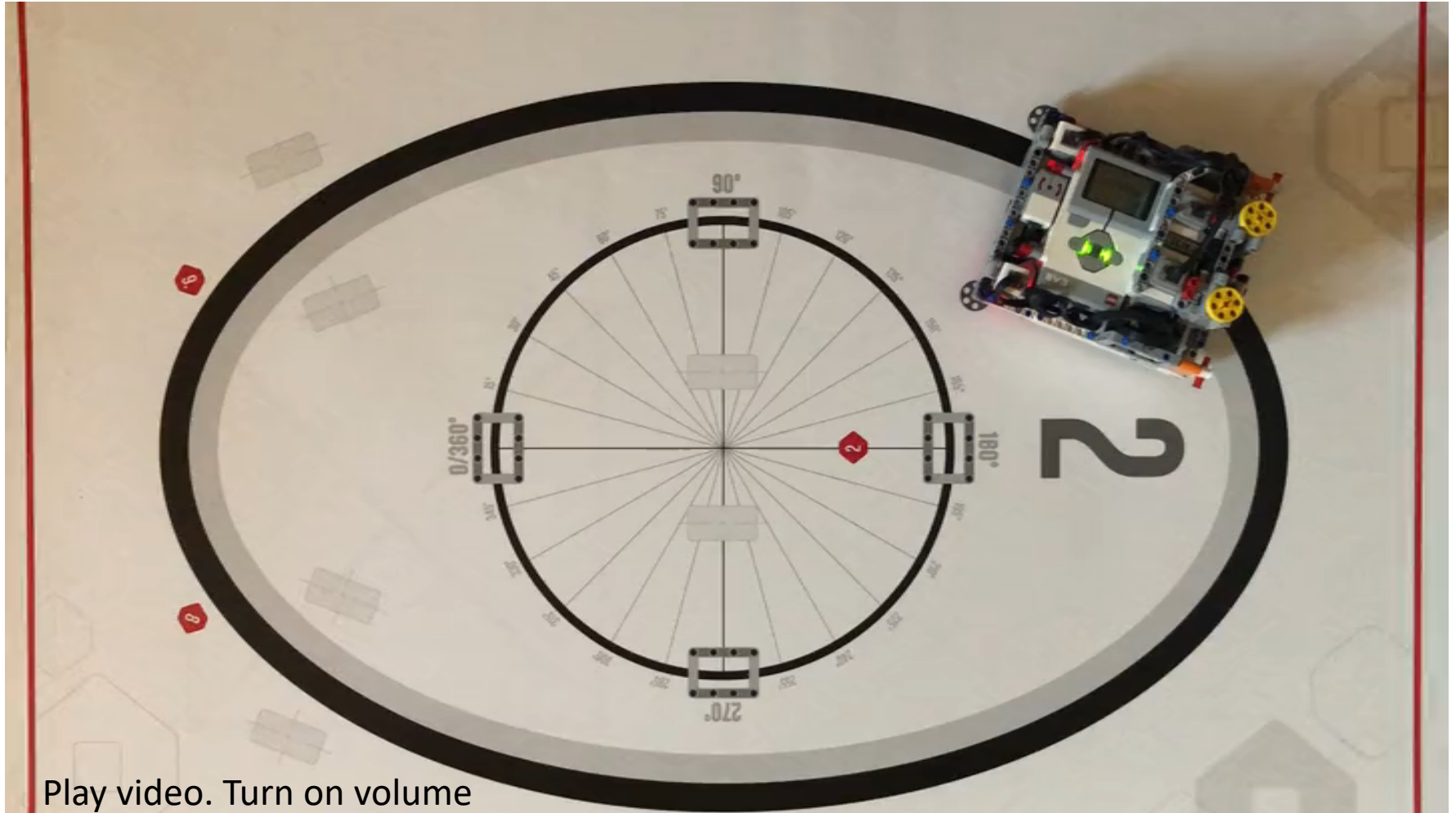
- Note, the 0.7 constant in the previous slide is specific to our robot – you need to tune this value for yourself
- This constant is called the Proportional Constant, or Constant of Proportionality
- The most common way to tune your constant is trial and error.
- This can take time. Here are some tips:
 - Start with your constant as 1.0 adjust by ± 0.5 initially
 - Adjust to a point where the controller is pretty smooth
 - Adjust ± 0.1 for fine tuning

Proportional Control (0.6 Constant)



Play video. Turn on volume

Proportional Control (0.8 Constant)



Play video. Turn on volume

Discussion Guide

1. What does proportional control mean?

Ans. Moving more or less based on how far the robot is from the target distance

2. What do all proportional control code have in common?

Ans. Computing an error and making a correction

Credits

- 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](https://creativecommons.org/licenses/by-nc-sa/4.0/).