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
On our team, we discuss “proportional” as a game.

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 (use masking tape to draw a line on the floor).

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 does proportional mean?

The robot moves proportionally – moving more or less based on how far the robot is from the target distance.

For a line follower, the robot may make a sharper turn if it is further away from the line.

Proportional Control can be more accurate and faster.

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.
The Pseudocode for every proportional control program consists of two stages:

- Computing an error $\rightarrow$ how far is the robot from a target
- Making a correction $\rightarrow$ 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.
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).
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
Challenge

To learn how to use proportional control, create a Dog 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)

<table>
<thead>
<tr>
<th>Application</th>
<th>Objective</th>
<th>Error</th>
<th>Correction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dog Follower</td>
<td>Get to a target distance from wall</td>
<td>How many inches from target location (current_distance – target_distance)</td>
<td>Move faster based on distance</td>
</tr>
</tbody>
</table>
Solution: Ultrasonic Dog Follower

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

**Part 1: Compute the Error**
- Error is Current Distance - Target
- We have chosen 15cm as the target.

**Part 2: Computes and Apply the Correction**
- We multiply the Error from Part 1 by 10 to determine the speed
- We picked 5 to create a reasonable range of speeds for our robot

Example:
- Sensor Reading = 10cm
- Error = 3cm
- error*5 = (3*5) = 15
- 15 power is a good speed to be use at 3cm from the target for our robot.
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
This tutorial was created by Sanjay Seshan and Arvind Seshan

More lessons at www.ev3lessons.com