



While you wait

Sign in for points by following the link or scanning the QR code:

<http://bit.ly/2y1YUwh>



Robotathon Tech Talk 2: Algorithms and Control!

October 4 & 5, 2017



Intro

So you've learned how to interface your microcontroller with your sensors. Now you have to do something with that data. But what?

We will learn how to turn feedback from sensors into robot motion.



Controller

A controller is a device configured in such a way that it regulates the behavior of another device or system.

What are some examples of a controller?

Goal: We want to configure your TI launchpad in such a way that it regulates the behavior of your robot.



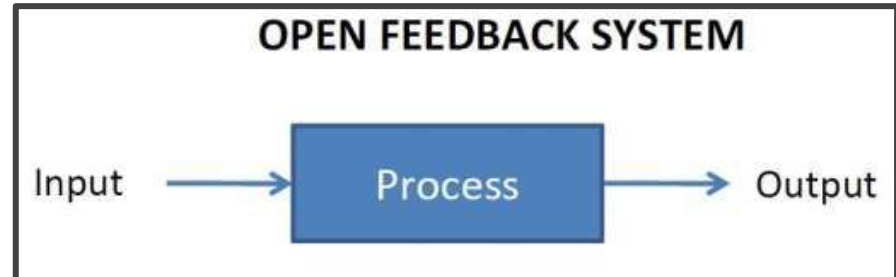
Open Loop Control Systems

An open loop system only cares about the input and what it's told to do.

It doesn't adjust for outside disturbances.

e.g. a toaster. The toaster doesn't know when the toast is done. It just heats up for as long as you tell it.

What happens if you put a piece of toasted toast back inside the toaster?





Closed Loop Control Systems

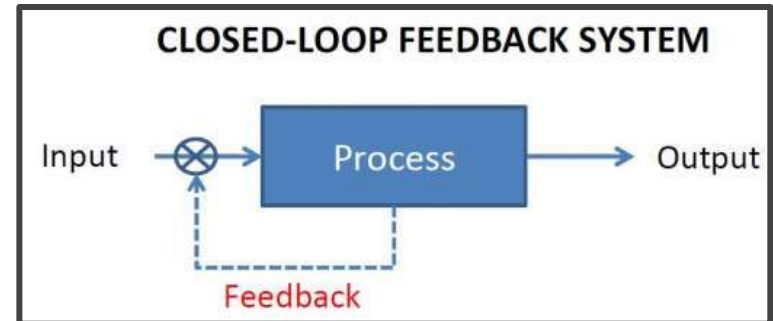
This system takes feedback from the output and accounts for error.

e.g. an air-conditioner: It adjusts it's output based on room temperature

What happens if you put an AC in a room colder than it's setting?

What is the feedback mechanism?

What is the actuator?





Error and Feedback

Error = actual value - desired value

Helps identify how far your system is deviating from the desired outcome

Feed the error back into your controller, so that it can adjust the behavior of the system

This allows for you to fix the error and get closer to the desired value



Proportional Control

Easy to implement

$$\text{output} = (K_p * \text{error}) + p_0;$$

K_p is the proportional constant

p_0 is output with 0 error

Error is (actual value - desired value)

Key Question:

Should K_p be positive or negative? Why?



Shortcomings of Proportional Control

Can't address steady state error.

What is steady state error? E.g. ship with a crosswind, can result in oscillation with large K_p .

How to address this?

Proportional and Integral control!



PI (Proportional-Integral) Control

Adds an integral term to your proportional controller. This allows your controller to take the sum of all past error into account.

$$\text{output} = (K_p * \text{error}) + (K_i * \text{allError}) + p0;$$
$$\text{allError} = \text{allError} + \text{error};$$

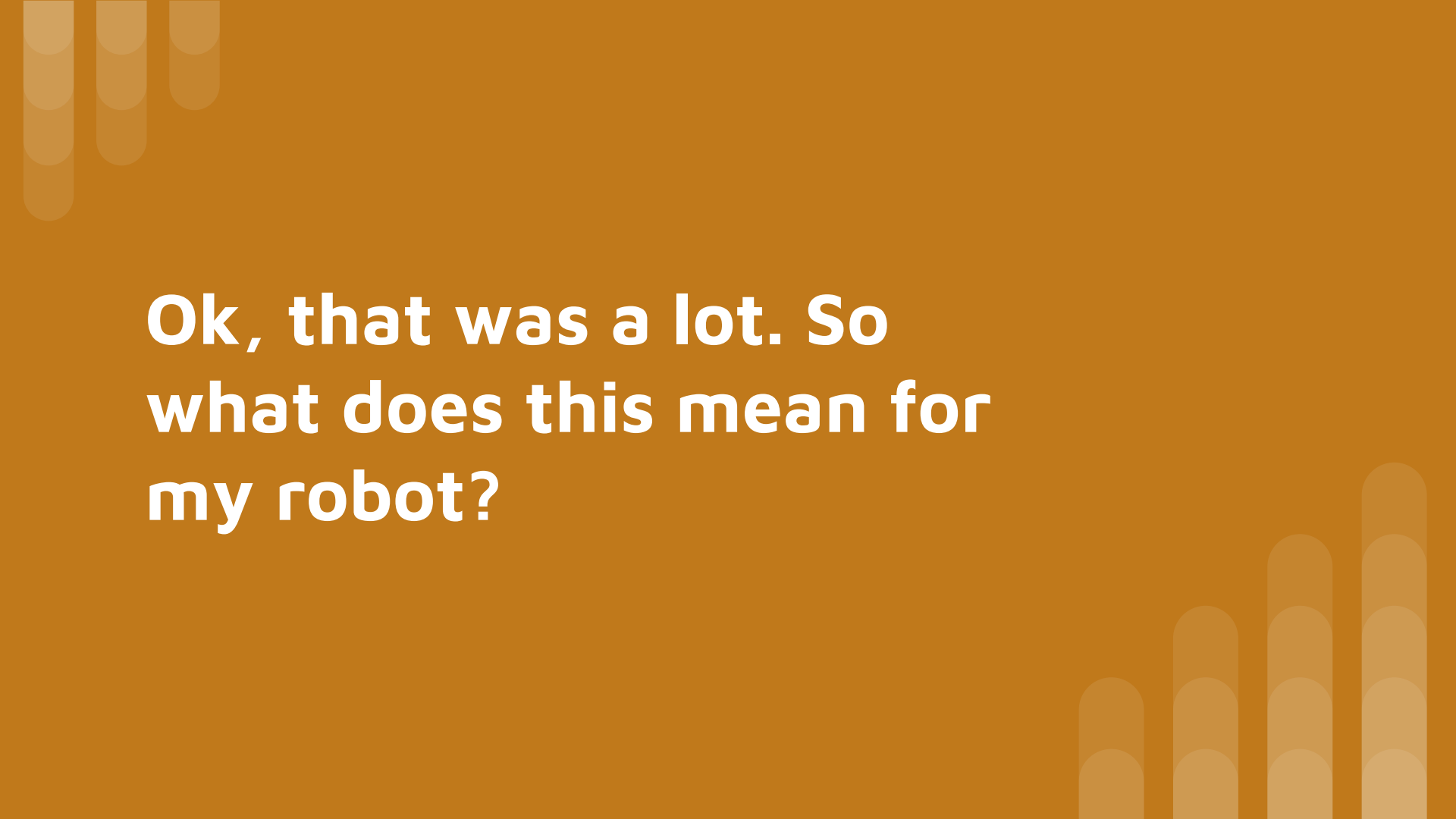
K_i is the integral term constant



PID (Proportional-Integral-Derivative) Control

Adding a derivative term allows your controller to respond to sudden changes in the system's state.

If you are curious about how to implement this, please ask me after!



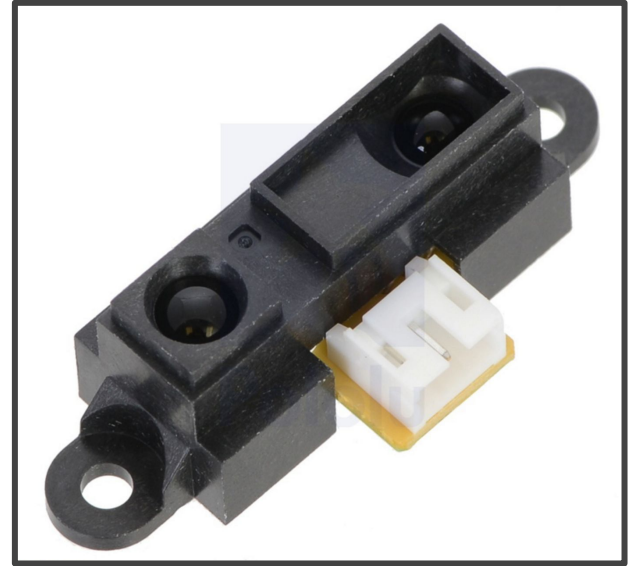
**Ok, that was a lot. So
what does this mean for
my robot?**

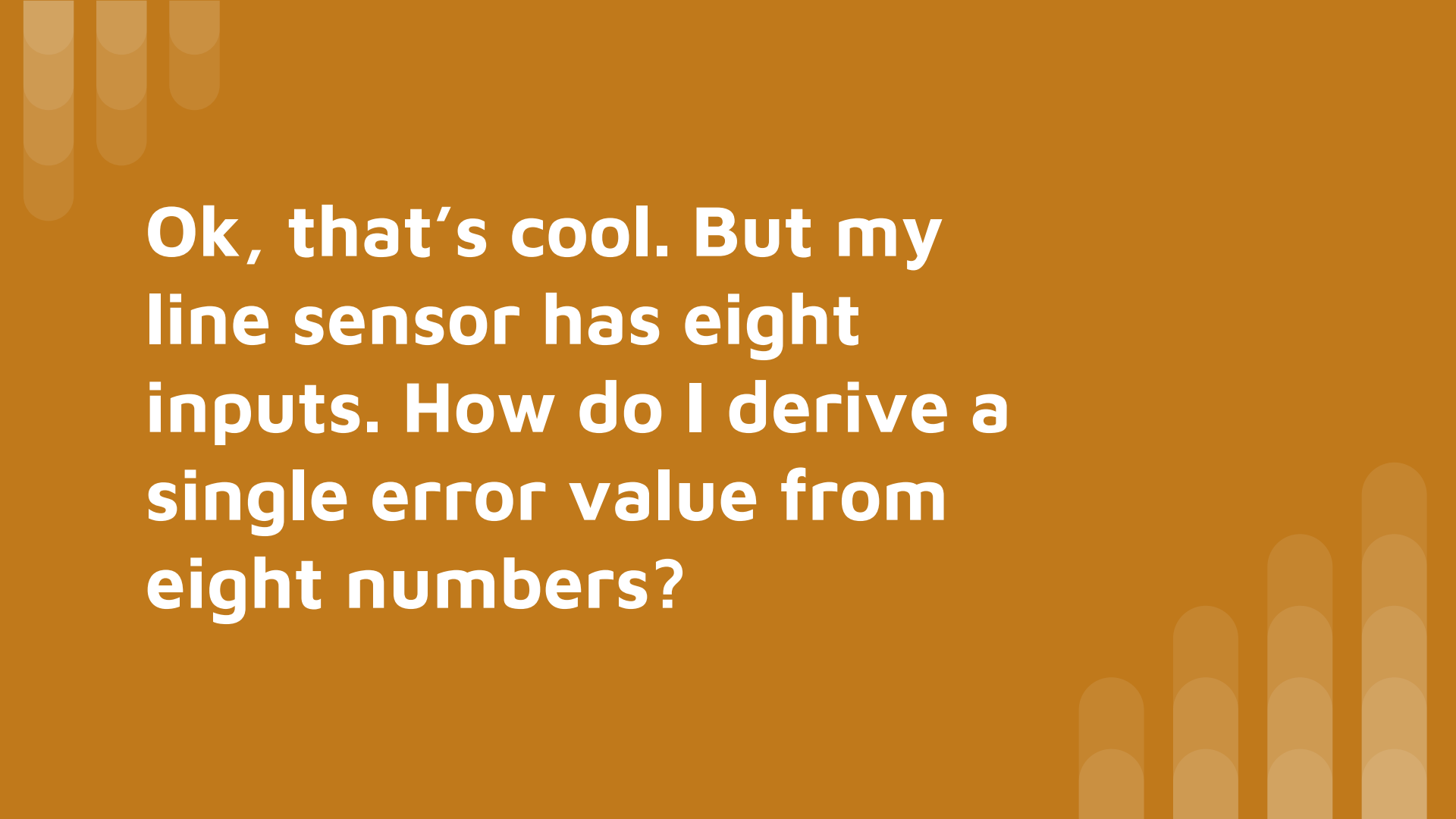
How to derive error from your distance sensor

Take readings at two distances

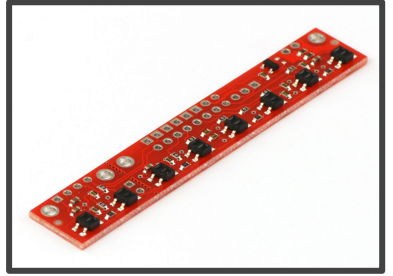
Calculate constant to convert values to cm

Error = desired distance from object - reading from distance sensor





Ok, that's cool. But my line sensor has eight inputs. How do I derive a single error value from eight numbers?



Your distance sensor reads eight values.

Store these 8 values in an array

Determine if each one is reading black or white

Multiply each element in the array by a weight array

Sum elements of the output. A negative value says you are too far in one direction, a positive value says that you are too far in the other direction.

Index	0	1	2	3	4	5	6	7
Reading from Sensor	0.12	0.15	0.08	.96	.88	0.13	0.15	0.02
White = 1 Black = 0	0	0	0	1	1	0	0	0
Weight array	-4	-3	-2	-1	1	2	3	4
Multiply b&w by weight	0	0	0	-1	1	0	0	0
Sum: 0								



Index	0	1	2	3	4	5	6	7
White = 1 Black = 0	0	0	0	0	1	1	0	0
Weight array	-4	-3	-2	-1	1	2	3	4
Multiply b&w by weight	0	0	0	0	1	2	0	0
Sum: 3								

Index	0	1	2	3	4	5	6	7
White = 1 Black = 0	0	0	1	1	1	0	0	0
Weight array	-4	-3	-2	-1	1	2	3	4
Multiply b&w by weight	0	0	0	0	1	2	0	0
Sum: -2								



Program Flow

```
while(1)
{
    Get value from sensor;
    Calculate error;

    // yourAlgorithm takes error as input
    output = yourAlgorithm(error);

    motors(output);
}

int yourAlgorithm(int error){
    int output;
    P or PI controller;
    Return output;
}
```



How to optimize your algorithms

Trial and error

Know how each proportional term affects the behaviour of your robot