(https://www.edx.org)

UTAustinX: UT.6.01x Embedded Systems - Shape the World

KarenWest (/dashboard) ▼

Courseware (/courses/UTAustinX/UT.6.01x/1T2014/courseware) Course Info (/courses/UTAustinX/UT.6.01x/1T2014/info)

Discussion (/courses/UTAustinX/UT.6.01x/1T2014/discussion/forum) Progress (/courses/UTAustinX/UT.6.01x/1T2014/progress)

Questions (/courses/UTAustinX/UT.6.01x/1T2014/a3da417940af4ec49a9c02b3eae3460b/)

Syllabus (/courses/UTAustinX/UT.6.01x/1T2014/a827a8b3cc204927b6efaa49580170d1/)

Embedded Systems Community (/courses/UTAustinX/UT.6.01x/1T2014/e3df91316c544d3e8e21944fde3ed46c/)

We begin the design of the robot car with the basic approach. The robot uses **sensors** to learn about its environment. There will be two IR distance sensors. If we are in the middle of the road, then both sensors will report the same distance to their respective walls.

VIDEO 14.6A. IR SENSOR FOR ROBOT CAR

Help

0:00 / 12:48 1.0x

DR. RAMESH YERRABALLI: So we've looked at a transducer which

was a potentiometer that gave us a voltage value

from a corresponding variable resistance.

Jon, can we look at a different device that

has some significance that we can then use to control something?

DR. JONATHAN VALVANO: Yeah, let's look at the sharp sensor.

The sharp sensor is an infrared distance sensor.

That means it takes a distance to an object

and it will convert it to a voltage.

This means if we connect it up to PE2 in our microcontroller

then our software has access.

Because it's an A to D, we can get a number which is between 0 and 4,095 which represents the distance.

In order to understand this, we need to look

at the voltage created as a function of distance of this sensor.

And this is a very unusual and nonlinear sensor that has this sort of behavior.

It has two phases here.

This first phase where it's really close,

and that's less than 100 millimeters, it behaves in this weird way.

But after 100 millimeters it behaves in a very simple way

such that the distance is approximately equal to some constant

over the voltage.

And so I can use this relationship to write software such that my distance

variable-- this is now in software, in millimeters--

is equal to the sum calibration constant.

I did calibrate it.

And so the result I got was 241,814 divided by this ADC value--

this number that comes from ADC.

So what we have is a software conversion here

which converts the number 0 to 4,095 which is measured by the a

to d into a distance in millimeters which works from about 100

millimeters out to 700 millimeters.

DR. RAMESH YERRABALLI: So just to recall,

the voltage to distance graph here is non-linear.

Whereas for our potentiometer the voltage

to distance-- if you think of the potential meter

as measuring distance-- then that would be a linear graph.

DR. JONATHAN VALVANO: That's correct.

And the software doesn't mind dealing with non-linearity

because we can just compensate for it in this equation.

But what do we use it for?

DR. RAMESH YERRABALLI: Well, we've

been promising

that we will build a robot car that can avoid colliding with walls by sensing.

So let's put this to work.

DR. JONATHAN VALVANO: All right.

So the next step in our control system is to look at the blocks.

The parameter we want to control is called a state variable.

In our case it will be where we are in the road.

We're then going to use transducers to estimate what we know

and learn where we are in the road.

And then we're going to use a control equation, which in our case

will be software.

And then use these outputs to drive actuators that affect the real world.

So this is our world, which is our robot on the road.

And we're going to estimate where we are on the road--

DR. RAMESH YERRABALLI: By using a transducer that is our infrared sensor.

DR. JONATHAN VALVANO: And the A to D converter.

DR. RAMESH YERRABALLI: And we perform some logic

which is the control equation.

This is the intelligence part of our code.

DR. JONATHAN VALVANO: And we're actually going

to drive 2 pulse with modulated outputs for the left and right motors.

And you notice that a control system has a loop, a feedback loop.

And again, the desire is to drive the state variable to a certain set point.

All right, let's put the pieces together.

Let's look at more detail of the state variable.

Here's our road.

And we're going to have a wall and another wall.

And we want to avoid the wall.

And we're going to have our robot car here, which

is moving in this direction.

And we're going to have 2 sensors on the front of the robot.

And we're going to measure the distance between the robot and left wall

and the robot and the right wall.

So we'll call this Dleft, that's the distance to the left wall,

because we're going this way.

And this is the distance to the right wall.

And how do you know if we're in the middle of the road?

DR. RAMESH YERRABALLI: I guess Dleft should be equal to Dright.

DR. JONATHAN VALVANO: Yeah, so what we're going to see

is, we're going to say we are going to look at our error.

In other words, we are going to be unhappy

if the distance to the left wall is different--

so we'll calculate the difference to the right wall.

And so the goal state is to drive the error to 0.

Because if this state variable error is 0 we're in the middle of the road.

So you remember our robot had 2 wheels.

And so what we have is PWM outputs for both wheels.

And so for each PWM outputs-- for instance

we have the right wheel high over the right wheel high plus right wheel low.

And this controls the power to the right motor.

In this example we're going to fix this to 40%.

So the right wheel is going to spin away at 40%.

The interesting part will be on the left wheel.

On the left wheel we're going to take the left high and let it change.

And again, the pulse with modulated signal for the left wheel

is the left high-- the time in which the left output is high

plus the time in which the left is low.

This pulse width modulation here, this

duty cycle, is going to vary.

We're going to let it vary somewhere between 30% and 50%.

So as this varies between 30% or 50% it's

going to be going straight-- if its 40, 40.

40, 30 will turn one way.

40, 50 will turn the other way.

So this parameter here is the one I'm going to adjust in my control loop.

So let's look at the software flow chart.

This is a real time problem so we're going

to use SysTick interrupts to run our control loop.

So every periodic time, at about 100 times per second,

we're going to run this control loop.

We're going to measure the distance to the right and the distance to the left.

We're going to calculate the error, which

we saw last time, which was the difference

between the left and the right.

And now comes the fun part.

What do we do with it?

What I'm going to do with it is I'm going to actually look at the left high

and I'm going to adjust it.

Either spin it faster or slower, depending upon the error.

And I'm going to do this in a typical control way by using a number.

It's called the gain of this control system.

And I'm going to take the error and multiply it times 200

and subtract it off.

So if I'm too far to left, it'll turn right.

If I'm too far to the right, I'll turn left.

Notice what happens here if the error is equal to 0.

What happens if the error-- if I'm in the middle of the road, what

does my control system do?

DR. RAMESH YERRABALLI: So if the error is equal to 0, the left high stays put.

It doesn't change.

DR. JONATHAN VALVANO: So if we're happy, we'll stay happy.

DR. RAMESH YERRABALLI: And remember that we've

kept the right high at a constant.

So that doesn't change either.

So now we don't change either of them, we stay the course.

DR. JONATHAN VALVANO: So if we're in the middle,

we'll just keep being in the middle.

There's one more couple of problems we have

to deal with is that we have to make sure that it doesn't go below 30%.

And so if the duty cycle goes below 30%, we will set the duty cycle at 30%

so it doesn't completely stop or try to spin backwards.

If it's OK then we will continue.

So if we look at the duty cycle here of the left wheel, if it is too slow

we'll fix it at too slow.

Similarly, if it's too fast, we'll look at the duty cycle.

If it is greater than 50% then we will set the duty cycle at 50%.

And these two comparisons will force the range here to between 30% and 50%.

We do have to calculate the other side.

The left low has to be set so that this number here is a constant.

And that constant is 80,000 minus the left high.

And then we return.

So these are the steps of my control loop.

We sense our inputs.

We decide if we're were happy.

We adjust the actual actuator output so that we become more happy.

In other words, if we're too far to the left, we go right.

If we're too far to the right, we go left.

And this is a gain.

In other words, how fast do we turn?

In a control system we want to make sure we don't go crazy.

So we'll check and make sure we have a minimum and a maximum value that

makes sense for our actuator.

And then we perform the output.

And zoom, our robot is spinning.

The goal is to drive a robot car autonomously down a road. Autonomous driving is a difficult problem, and we have greatly simplified it and will use this simple problem to illustrate the components of a control system. Every control system has real-world parameters that it wishes to control. These parameters are called **state variables**. In our system we wish to drive down the middle of the road, so our state variables will be the distance to the left side of the road and the distance to the right side of the road as illustrated in Figure 14.7. When we are in the middle of the road these two distances will be equal. So, let's define *Error* as:

$$Error = D_{left} - D_{right}$$

If *Error* is zero we are in the middle of the road, so the controller will attempt to drive the *Error* parameter to zero.

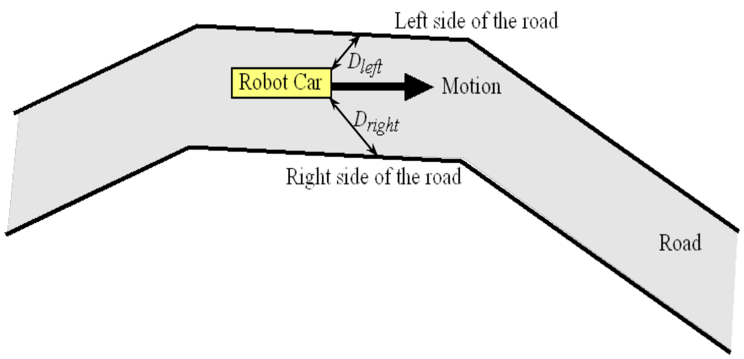

Figure 14.7. Physical layout of the autonomous robot as is drives down the road.

We will need sensors and a data acquisition system to measure D_{left} and D_{right}. In order to simplify the problem we will place pieces of wood to create walls along both sides of the road, and make the road the same width at all places along the track. The Sharp GP2Y0A21YK0F infrared object detector can measure distance (http://www.sharpsma.com (http://www.sharpsma.com/)) from the robot to the wood. This sensor creates a continuous analog voltage between 0 and +3V that depends inversely on distance to object, see Figure 14.6. We will avoid the 0 to 10 cm range where the sensor has the nonmonotonic behavior. We will use two ADC channels (PE4 and PE5) to convert the two analog voltages to digital numbers. Let **`Left`** and **`Right`** be the ADC digital samples measured from the two sensors. We can assume distance is linearly related to 1/voltage, we can implement software functions to calculate distance in mm as a function of the ADC sample (0 to 4095). The 241814 constant was found empirically, which means we collected data comparing actual distance to measured ADC values.

Dleft = 241814/**Left**

Dright = 241814/**Right**

Figure 14.8 shows the accuracy of this data acquisition system, where the estimated distance, using the above equation, is plotted versus the true distance.

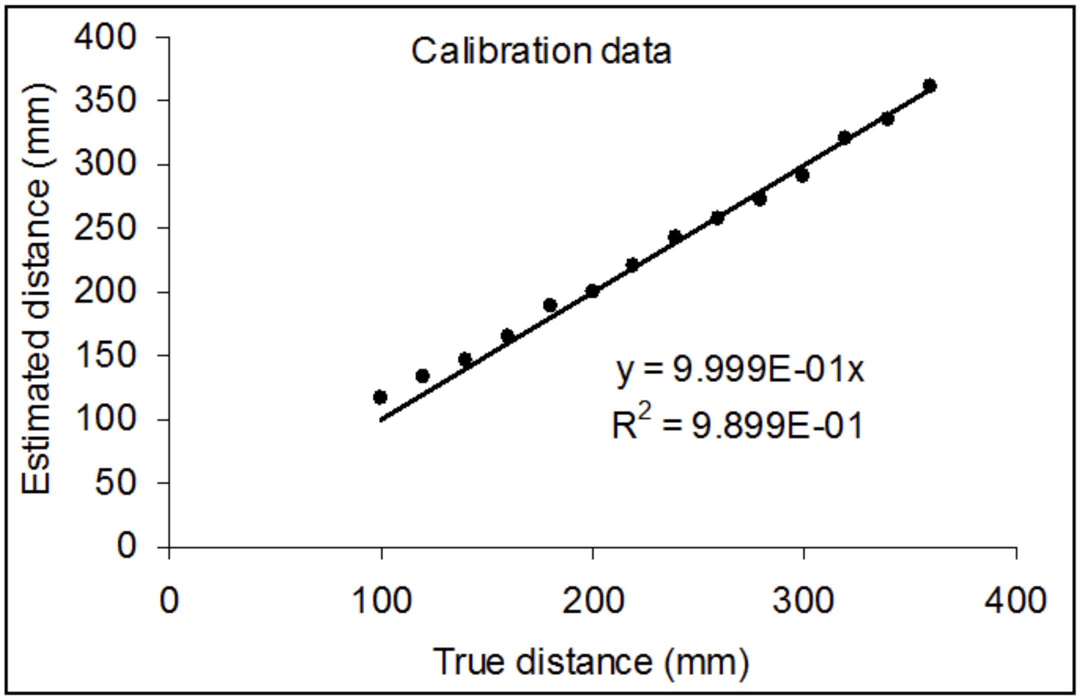

Figure 14.8. Measurement accuracy of the Sharp GP2Y0A21YK0F distance sensor used to measure distance to wall.

Help

About (https://www.edx.org/about-us) Jobs (https://www.edx.org/jobs)
Press (https://www.edx.org/press) FAQ (https://www.edx.org/student-faq)
Contact (https://www.edx.org/contact)

EdX is a non-profit created by founding partners Harvard and MIT whose mission is to bring the best of higher education to students of all ages anywhere in the world, wherever there is Internet access. EdX's free online MOOCs are interactive and subjects include computer science, public health, and artificial intelligence.

(http://www.meetup.com/edX-Global-Community/)
(http://www.facebook.com/EdxOnline)
(https://twitter.com/edXOnline)
(https://plus.google.com/108235383044095082735/posts)

(http://youtube.com/user/edxonline)
© 2014 edX, some rights reserved.

Terms of Service and Honor Code -
Privacy Policy (https://www.edx.org/edx-privacy-policy)

Help