MASSACHVSETTS INSTITVTE OF TECHNOLOGY
Department of Electrical Engineering and Computer Science
6.081—Introduction to EECS I
Spring Semester, 2007

Work for Week 5

Issued: Tuesday, March 6

This handout contains

- Software Lab for March 6th
- Prelab exercises due Thursday March 8th before lab
- Post-lab exercises due Tuesday March 13th in Lecture.

Dynamic Feedback Control

In many of the labs so far, you developed progressively more sophisticated approaches for measuring the robot's environment, as well as progressively more intelligent approaches to controlling the robot's actions based on those measurements. In those labs, our primary goal was to have you learn a little about the robot as well as have you investigate some ideas in software engineering. In this lab, we will also be using measurements to control the robot, also known as feedback control, but we will be investigating a situation in which detailed analysis is needed to achieve a desired performance. The problem we are asking you to investigate is how to control the robot so that it drives down the center of a narrow corridor at a constant forward velocity. Such a task is similar to what an automobile driver does when keeping a car on the road. We will suggest a simple feedback scheme to keep the robot in the center of corridor, and ask you to develop and analyze a mathematical model based on difference equations that explains the behavior of that simple feedback system. In next week's lab, you will use more sophisticated mathematical tools to develop improvements to the simple feedback scheme suggested in this lab.

The summary of tasks for this week are:

- Post-lecture software lab on solving second order difference equations
- Pre-lab tutor exercises to practice solving difference equations
- Robot lab on implementing and analyzing a simple feedback scheme
- Post-lab writeup and exercises due Tuesday in lecture.

Tuesday's Software Lab: Solving second-order difference equations

For this lab, you will be writing a python program that solves arbitrary second-order homogeneous difference equations analytically, by computing natural frequencies. Your program should take as inputs the initial values $x[0]$ and $x[1]$ as well as coefficients a_1 and a_2 for the difference equation in the form

$$y[n] = a_1 * y[n-1] + a_2 * y[n-2].$$

Your program should print out the solution to the difference equation and also return a procedure that, when called with n, returns $y[n]$. For example, if the difference equation is

$$y[n] = 2 * y[n-1] - 2 * y[n-2],$$

and the initial conditions are $y[0] = 0$, $y[1] = 1$, your program should print something equivalent to

```
y[n] = (2.77555756156e-017+0.5j)*(1-1j)**n  +  (-2.77555756156e-017-0.5j)*(1+1j)**n
```

and your program should ALSO return a function which can be used to evaluate $y[n]$ for any n.

Getting Started

To save time, you should use our implementation of the polynomial manipulation program from the recent post-lab exercises in your second-order difference equation solver. Please download *importPolynomial.py* and *polynomial.pyc*. Executing the module *importPolynomial.py* in IDLE will import our version of the polynomial manipulation routines implemented in the class Polynomial. You can see the class interface by typing

```
help(Polynomial)
```

at the IDLE command line. Note that the Polynomial class has functions to add, multiply, print and find the roots of polynomials.

Demonstrate to your LA that you understand the Polynomial class. Then, before you write any code, describe your approach to the difference equation solver to your LA.

A note on complex numbers

For this problem, your procedure will sometimes need to compute z^n where z is a complex number. Python understands complex numbers to some extent, but you have to write them in the form $a + bj$, where a is the real part and b is the imaginary part. Note that Python uses the convention that $j = \sqrt{-1}$. This should not surprise you, as entering EECS students, you already appreciate that the variable i must be reserved for electrical current.

So, for example, the Python command

(-4)**0.5

will produce an error, but the Python command

(-4+0j)**0.5

produces (1.2246063538223773e-016+2j). (Why the very small real part?)

Note: use `y**0.5` to compute the square root of a number. The python *sqrt()* function does not understand complex numbers.

Demonstrate that your program works

Use your program to solve the difference equation

$$y[n] = 2 * y[n-1] - 2 * y[n-2],$$

using the initial conditions $y[0] = 0$, $y[1] = 1$. Demonstrate to your LA that the program prints the correct information and also returns the correct procedure.

Use your program to find difference equations with given properties

Use your program to find a difference equation whose solution oscillates rapidly, but the amplitude of the oscillation decays slowly. Use your program to find a difference equation whose solution oscillates rapidly, but the amplitude of the oscillation remains constant.

What to turn in

For this software lab, hand in a brief (few page) lab report along with your post-lab report on March 13th. Also, hand in a copy of your carefully commented python code with an explanation, *written in actual English sentences*, of how and why it works.

Homework in preparation for Thursday's lab

This section includes problems to complete with the online tutor. These are due before lab on Thursday. The examples are to help you solidify your understanding of difference equations.

Read the entire document!

Please read the entire section on the lab before coming to lab.

Exercises with the online tutor

Use the online tutor to complete the problems due Thursday, March 8th.

Thursday's Lab

As we are sure you have come to expect, Thursday's lab will end with a nano-quiz that is based both on tutorial problems due on thursday, and on material covered this week. It should be no great surprise that you will be asked a question about the solution to a difference equation. Please keep in mind that the point of the quizzes is as a diagnostic for us (did we succeed in teaching you?) and to encourage you to do the tutorial problems before coming to lab and participate in lab.

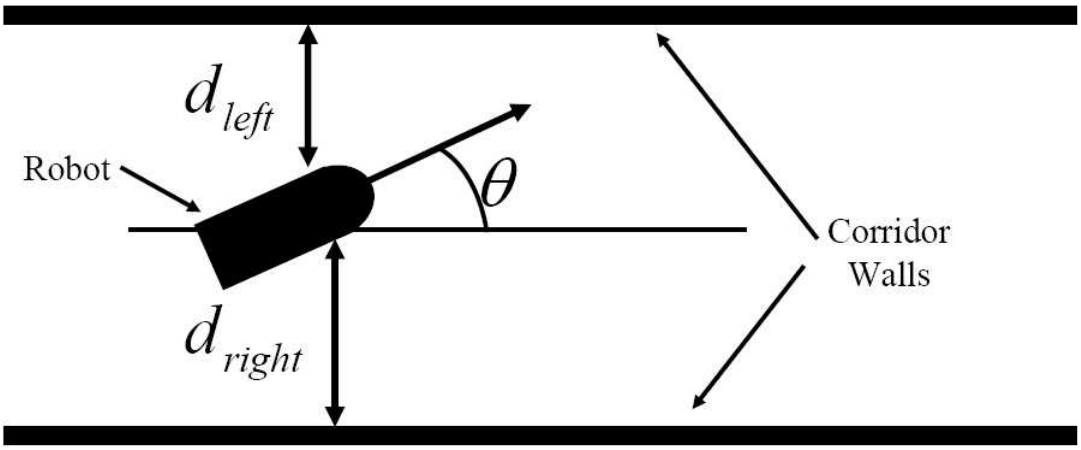

Figure 1: Robot in corridor.

Robot Feedback System

In this lab you will be controlling the robot to drive down a narrow corridor as shown in Figure 1. Notice that in the figure, we have denoted the forward velocity of the robot, V, the distance to the left wall, d_{left}, the distance to the right wall, d_{right}, and the angle the robot is making with respect to the parallel walls, θ.

Consider the problem of trying to steer the robot down the center of the corridor while keeping it moving forward with a constant velocity. For example, if the robot is in the center of the corridor and pointing in a direction parallel to the walls, one could keep the robot moving forward with the command

```
motorOutput(0.1, 0.0),
```

which will keep the robot moving forward 0.1 meters per second. If the robot is too close to the right wall, one could issue the command

```
motorOutput(0.1, 0.1),
```

which will keep the robot moving 0.1 meters per second but will cause the robot to rotate to the left. If the robot is too close to the left wall, one could issue the command

```
motorOutput(0.1, -0.1)
```

which will keep the robot moving 0.1 meters per second but will cause the robot to rotate to the right.

The above suggests a very simple feedback system which keeps the robot moving forward at 0.1 meters per second and tries to keep the robot in the center of the corridor. One could set the forward velocity in the motorOutput() command to 0.1 and set the rate of rotation in the motorOutput() command to be proportional to the distance from the center of the corridor.

Experiment with the above dynamic feedback control scheme by writing a SoaR brain that on every step, sets the robot velocity and updates the robot rotation so that it is proportional to the robot's signed distance from the corridor center ($d_{right} - d_{left}$). In order to make the distance measurement a little less sensitive to robot rotation, we suggest that you determine the distance to the left wall by computing the minimum of the o^{th} and 1^{st} and the distance to the right wall by computing the minimum of the 6^{th} and 7^{st} sensor. Try your brain on your robot in one of our narrow corridors. Experiment with different constants of proportionality (also known as gains) in your feedback system, and try starting the robot both near and far from the center of the corridor.

Checkpoint: 3:00 PM

- Code for the dynamic feedback control brain
- Results on robot motion in the corridor for several gains and initial positions

Difference Equation Derivation

When the SoaR system is communicating with the robot and running your `brain`, it executes the step function many times every second. If you wrote your brain correctly, this means that many times every second, your brain is taking sensor readings to determine the offset from center, and then updating the robot rate of rotation. Suppose we denote $d[n]$ to be the signed displacement from center due to the n^{th} execution of the step function. Then, your feedback control algorithm will be setting the rate of rotation at the n^{th} step to be $K * d[n]$, were K is the feedback gain. Can you derive a second-order difference equation that can be used to solve for $d[n]$?

To help get you started, suppose that at step n, the robot is displaced from center by an amount $d[n]$, and is moving with a velocity V in a direction that makes an angle $\theta[n]$ as shown in Figure 1. In addition, suppose that the step function is executed every δt seconds. Then

$$d[n] = d[n-1] + V\delta t \sin\theta[n-1] \approx d[n-1] + V\delta t\theta[n-1]$$

where the approximation holds if the angle $\theta[n-1]$ is small.

Now think about how to write a difference equation for $\theta[n-1]$, and how to combine the two difference equations in to a single equation for $d[n]$.

Checkpoint: 4:00 PM

- Difference equation model for the robot displacement from center

Validating your model

Determine and execute experiments to validate your model. That is, show that your model predicts the actual robot's behavior. What does the model predict about how the robot will behave as you change the feedback gain? Is the model accurate?

There are many ways to determine the sampling rate being used by your robot. One direct way to measure the sample rate is to use the `time()` command, which you can use if you put the line:

```
from time import *
```

at the top of your brain.

Please keep in mind that if you print something every time the brain executes a step, you will slow down the step function and change the sample rate!!!

Checkpoint: 4:45 PM

- Model validation (show computer plots and robot behavior)

3. Lab Writeup due March 13th

Write a brief description of how you derived your difference equation model for the simple robot dynamic control system and explain how you determined the parameters in the difference equation. In addition, describe what experimental measurements you made to validate your model. Please make sure your lab writeup addresses the following questions or requests:

- How are the difference equation natural frequencies related to your feedback gain? Draw a plot of natural frequency locations in the complex plane for many values of feedback gain. Be sure to denote the location of the unit circle.
- What does your answer to above two question tell you about the solution to the difference equation model of the robot feedback system.
- Show plots of the differential equation model solution for different feedback gains and a nonzero initial offset from the corridor center. Why are you plots consistent with the natural frequency locations?
- What does the model tell you about your ability to find a feedback gain for which the robot will not eventual hit the wall of the corridor?

Concepts covered in this assignment

Here are the important points covered in this assignment:

- Learn about difference equations and how to solve them.
- Learn about the connection between natural frequencies and the solutions of difference equations.
- Learn to reason about systems using difference equations.
- Begin to learn about the issue of stability in dynamic feedback systems.