Chapter 1

Three Mode Pole Placement Controller

1.1 Introduction

During the fall semester of 2003, I designed a pole placement controller to place the first two lightly damped modes of SAMII at pole locations with higher damping, but with the same natural frequencies. This first attempt at a pole placement controller made the third mode unstable.

At the end of last semester, I curve fit a model that included the first three modes of the flexible base to the bode data. This curve fitting was done to enable a pole placement controller design that placed the first three modes of the flexible base. Obviously, there will always be unmodeled modes in the system. It was my hope that placing the third mode would place the unmodeled modes above the bandwidth of the hydraulic actuators.

This section details the design and implementation of the three mode pole placement controller/observer including a derivation of the state-space model from the transfer function model.

1.2 State-Space Model Derivation

A controllable canonical state-space model is derived from a transfer function representation of the system. The results of this derivation are output to an m-file that implements this model numerically with the coefficients from curve fitting done last semester. This model is verified in section 1.3 by overlaying bode plots generated with the state-space model with those generated with the transfer function model and from experimental data.

1.2.1 Transfer Function Manipulation

The transfer funciton for θ/d can be written as

$$\frac{\theta}{d} = \frac{{\omega_d}^2 \left(s^2 + 2\zeta_2\omega_2 s + {\omega_2}^2\right)\tau}{s{\omega_2}^2 \left(s^2 + 2\zeta_d\omega_d s + {\omega_d}^2\right)(s+\tau)} \tag{1.1}$$

The transfer function for $\ddot{x}/\theta$ can be written as

$$\frac{\ddot{x}}{\theta} = \frac{s^4 B_1}{s^2 + 2\zeta_1\omega_1 s + {\omega_1}^2} + \frac{s^4 B_2}{s^2 + 2\zeta_2\omega_2 s + {\omega_2}^2} + \frac{s^4 B_3}{s^2 + 2\zeta_3\omega_3 s + {\omega_3}^2} \tag{1.2}$$

multiplying equation 1.2 by equation 1.1 allows the numerator of the transfer function between $\ddot{x}/d$ to be written as

$$\begin{aligned} N_x = &\left(s^2 + 2\zeta_3\omega_3 s + {\omega_3}^2\right)\left(s^2 + 2\zeta_2\omega_2 s + {\omega_2}^2\right) s^4 \tau {\omega_d}^2 B_1 \\ &+ \left(s^2 + 2\zeta_3\omega_3 s + {\omega_3}^2\right)\left(s^2 + 2\zeta_1\omega_1 s + {\omega_1}^2\right) s^4 \tau {\omega_d}^2 B_2 \\ &+ \left(s^2 + 2\zeta_2\omega_2 s + {\omega_2}^2\right)\left(s^2 + 2\zeta_1\omega_1 s + {\omega_1}^2\right) s^4 \tau {\omega_d}^2 B_3 \end{aligned} \tag{1.3}$$

The denominator of this transfer function can be written as

$$D = \left(s^2 + 2\zeta_3\omega_3 s + {\omega_3}^2\right)\left(s^2 + 2\zeta_1\omega_1 s + {\omega_1}^2\right) s{\omega_2}^2 \left(s^2 + 2\zeta_d\omega_d s + {\omega_d}^2\right)(s + \tau) \tag{1.4}$$

For the sake of the state-space representation, we will use D as the common denominator for the tranfer functions The numerator and denominator of the transfer function θ/d would need to be mulplied by the term

$$\frac{D}{D_\theta} = \left(s^2 + 2\zeta_3\omega_3 s + {\omega_3}^2\right)\left(s^2 + 2\zeta_1\omega_1 s + {\omega_1}^2\right) \tag{1.5}$$

Expanding the denominator gives

$$\begin{aligned} D = &s^8 {\omega_2}^2 \\ &+ \left(2\zeta_1\omega_1{\omega_2}^2 + {\omega_2}^2\tau + 2{\omega_2}^2\zeta_d\omega_d + 2{\omega_2}^2\zeta_3\omega_3\right) s^7 \\ &+ \left(4{\omega_2}^2\zeta_3\omega_3\zeta_d\omega_d + 2{\omega_2}^2\zeta_d\omega_d\tau + {\omega_1}^2{\omega_2}^2 + 2{\omega_2}^2\zeta_1\omega_1\tau + 4{\omega_2}^2\zeta_1\omega_1\zeta_d\omega_d + {\omega_2}^2{\omega_d}^2 + 2{\omega_2}^2\zeta_3\omega_3\tau + 4{\omega_2}^2\zeta_1\omega_1\zeta_3\omega_3 + \omega\right. \\ &+ \left(2{\omega_2}^2{\omega_3}^2\zeta_d\omega_d + 4{\omega_2}^2\zeta_1\omega_1\zeta_3\omega_3\tau + {\omega_2}^2\tau{\omega_d}^2 + 2{\omega_2}^2\zeta_1\omega_1{\omega_d}^2 + 2{\omega_2}^2\zeta_3\omega_3{\omega_d}^2 + 2{\omega_2}^2{\omega_1}^2\zeta_d\omega_d + 4{\omega_2}^2\zeta_1\omega_1\zeta_d\omega_d\tau + \omega\right. \\ &+ \left(2{\omega_2}^2{\omega_3}^2\zeta_d\omega_d\tau + 4{\omega_2}^2\zeta_1\omega_1{\omega_3}^2\zeta_d\omega_d + {\omega_2}^2{\omega_1}^2{\omega_3}^2 + {\omega_2}^2{\omega_1}^2{\omega_d}^2 + 2{\omega_2}^2\zeta_3\omega_3{\omega_d}^2\tau + 2{\omega_2}^2\zeta_1\omega_1{\omega_3}^2\tau + 2{\omega_2}^2{\omega_1}^2\zeta_d\omega_d\tau\right. \\ &+ \left({\omega_2}^2{\omega_1}^2{\omega_d}^2\tau + 4{\omega_2}^2\zeta_1\omega_1{\omega_3}^2\zeta_d\omega_d\tau + 4{\omega_2}^2{\omega_1}^2\zeta_3\omega_3\zeta_d\omega_d\tau + {\omega_2}^2{\omega_3}^2{\omega_d}^2\tau + 2{\omega_2}^2{\omega_1}^2{\omega_3}^2\zeta_d\omega_d + 2{\omega_2}^2\zeta_1\omega_1{\omega_3}^2{\omega_d}^2 + 4\right. \\ &+ \left(2{\omega_2}^2{\omega_1}^2{\omega_3}^2\zeta_d\omega_d\tau + {\omega_2}^2{\omega_1}^2{\omega_3}^2{\omega_d}^2 + 2{\omega_2}^2{\omega_1}^2\zeta_3\omega_3{\omega_d}^2\tau + 2{\omega_2}^2\zeta_1\omega_1{\omega_3}^2{\omega_d}^2\tau\right) s^2 \\ &+ s{\omega_2}^2{\omega_1}^2{\omega_3}^2{\omega_d}^2\tau \end{aligned}$$

Expanding the numerator for θ/d gives

$$\begin{aligned} N_\theta = &\tau{\omega_d}^2 s^6 \\ &+ \left(2\tau{\omega_d}^2\zeta_2\omega_2 + 2\tau{\omega_d}^2\zeta_3\omega_3 + 2\tau{\omega_d}^2\zeta_1\omega_1\right) s^5 \\ &+ \left({\omega_2}^2\tau{\omega_d}^2 + 4\tau{\omega_d}^2\zeta_2\omega_2\zeta_1\omega_1 + 4\tau{\omega_d}^2\zeta_1\omega_1\zeta_3\omega_3 + \tau{\omega_d}^2{\omega_1}^2 + 4\tau{\omega_d}^2\zeta_2\omega_2\zeta_3\omega_3 + \tau{\omega_d}^2{\omega_3}^2\right) s^4 \\ &+ \left(2\tau{\omega_d}^2\zeta_2\omega_2{\omega_1}^2 + 2{\omega_2}^2\zeta_1\omega_1{\omega_d}^2\tau + 2{\omega_2}^2\zeta_3\omega_3{\omega_d}^2\tau + 2\tau{\omega_d}^2\zeta_2\omega_2{\omega_3}^2 + 8\tau{\omega_d}^2\zeta_2\omega_2\zeta_3\omega_3\zeta_1\omega_1 + 2\tau{\omega_d}^2\zeta_1\omega_1{\omega_3}^2 + 2\tau\omega\right. \\ &+ \left(4\tau{\omega_d}^2\zeta_2\omega_2{\omega_3}^2\zeta_1\omega_1 + {\omega_2}^2{\omega_3}^2{\omega_d}^2\tau + 4\tau{\omega_d}^2\zeta_2\omega_2\zeta_3\omega_3{\omega_1}^2 + 4{\omega_2}^2\zeta_1\omega_1\zeta_3\omega_3{\omega_d}^2\tau + {\omega_2}^2{\omega_1}^2{\omega_d}^2\tau + {\omega_1}^2{\omega_3}^2{\omega_d}^2\tau\right) s^2 \\ &+ \left(2{\omega_2}^2\zeta_1\omega_1{\omega_3}^2{\omega_d}^2\tau + 2{\omega_2}^2{\omega_1}^2\zeta_3\omega_3{\omega_d}^2\tau + 2\tau{\omega_d}^2\zeta_2\omega_2{\omega_3}^2{\omega_1}^2\right) s \\ &+ {\omega_2}^2{\omega_1}^2{\omega_3}^2{\omega_d}^2\tau \end{aligned}$$

Expanding the numerator for $\ddot{x}/d$ gives

$$\begin{aligned}N_x = &\left(\tau{\omega_d}^2 B_1 + \tau{\omega_d}^2 B_2 + \tau{\omega_d}^2 B_3\right) s^8 \\ &+ \left(\left(2\zeta_3\omega_3 + 2\zeta_1\omega_1\right)\tau{\omega_d}^2 B_2 + \left(2\zeta_2\omega_2 + 2\zeta_1\omega_1\right)\tau{\omega_d}^2 B_3 + \left(2\zeta_3\omega_3 + 2\zeta_2\omega_2\right)\tau{\omega_d}^2 B_1\right) s^7 \\ &+ \left(\left({\omega_2}^2 + 4\zeta_1\omega_1\zeta_2\omega_2 + {\omega_1}^2\right)\tau{\omega_d}^2 B_3 + \left({\omega_3}^2 + 4\zeta_1\omega_1\zeta_3\omega_3 + {\omega_1}^2\right)\tau{\omega_d}^2 B_2 + \left({\omega_3}^2 + 4\zeta_2\omega_2\zeta_3\omega_3 + {\omega_2}^2\right)\tau{\omega_d}^2 B_1\right) s^6 \\ &+ \left(\left(2\zeta_1\omega_1{\omega_2}^2 + 2{\omega_1}^2\zeta_2\omega_2\right)\tau{\omega_d}^2 B_3 + \left(2\zeta_2\omega_2{\omega_3}^2 + 2{\omega_2}^2\zeta_3\omega_3\right)\tau{\omega_d}^2 B_1 + \left(2\zeta_1\omega_1{\omega_3}^2 + 2{\omega_1}^2\zeta_3\omega_3\right)\tau{\omega_d}^2 B_2\right) s^5 \\ &+ \left({\omega_2}^2{\omega_3}^2{\omega_d}^2\tau B_1 + {\omega_1}^2{\omega_3}^2{\omega_d}^2\tau B_2 + {\omega_2}^2{\omega_1}^2{\omega_d}^2\tau B_3\right) s^4 \end{aligned} \tag{1.8}$$

1.2.2 Controllable Cannonical Form

For a system with the transfer function

$$\frac{y}{u} = \frac{b_n s^n + b_{n-1}s^{n-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0} \tag{1.9}$$

the controllable cannonical realization would be

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \\ -a_0 & -a_1 & \cdots & -a_{n-1} \end{bmatrix} \tag{1.10}$$

$$\mathbf{B} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \tag{1.11}$$

$$\mathbf{C} = \begin{bmatrix} b_0 - b_n a_0 & b_1 - b_n a_1 & \cdots & b_{n-1} - b_n a_{n-1} \end{bmatrix} \tag{1.12}$$

$$\mathbf{D} = b_n \tag{1.13}$$

For this system $n = 8$ and the coefficients of the denominator polynomial are

$$a_0 = 0 \tag{1.14}$$

$$a_1 = {\omega_2}^2{\omega_1}^2{\omega_3}^2{\omega_d}^2\tau \tag{1.15}$$

$$a_2 = 2{\omega_2}^2{\omega_1}^2{\omega_3}^2\zeta_d\omega_d\tau + {\omega_2}^2{\omega_1}^2{\omega_3}^2{\omega_d}^2 + 2{\omega_2}^2{\omega_1}^2\zeta_3\omega_3{\omega_d}^2\tau + 2{\omega_2}^2\zeta_1\omega_1{\omega_3}^2{\omega_d}^2\tau \tag{1.16}$$

$$a_3 = {\omega_2}^2{\omega_1}^2{\omega_d}^2\tau + 4{\omega_2}^2\zeta_1\omega_1{\omega_3}^2\zeta_d\omega_d\tau + 4{\omega_2}^2{\omega_1}^2\zeta_3\omega_3\zeta_d\omega_d\tau + {\omega_2}^2{\omega_3}^2{\omega_d}^2\tau + 2{\omega_2}^2{\omega_1}^2{\omega_3}^2\zeta_d\omega_d + 2{\omega_2}^2\zeta_1\omega_1{\omega_3}^2{\omega_d}^2 + 4{\omega_2}^2\zeta_1\omega_1 \tag{1.17}$$

$$a_4 = 2{\omega_2}^2{\omega_3}^2\zeta_d\omega_d\tau + 4{\omega_2}^2\zeta_1\omega_1{\omega_3}^2\zeta_d\omega_d + {\omega_2}^2{\omega_1}^2{\omega_3}^2 + {\omega_2}^2{\omega_1}^2{\omega_d}^2 + 2{\omega_2}^2\zeta_3\omega_3{\omega_d}^2\tau + 2{\omega_2}^2\zeta_1\omega_1{\omega_3}^2\tau + 2{\omega_2}^2{\omega_1}^2\zeta_d\omega_d\tau + 2{\omega_2}^2\omega \tag{1.18}$$

$$a_5 = 2{\omega_2}^2{\omega_3}^2\zeta_d\omega_d + 4{\omega_2}^2\zeta_1\omega_1\zeta_3\omega_3\tau + {\omega_2}^2\tau{\omega_d}^2 + 2{\omega_2}^2\zeta_1\omega_1{\omega_d}^2 + 2{\omega_2}^2\zeta_3\omega_3{\omega_d}^2 + 2{\omega_2}^2{\omega_1}^2\zeta_d\omega_d + 4{\omega_2}^2\zeta_1\omega_1\zeta_d\omega_d\tau + {\omega_2}^2{\omega_1}^2\tau + 4 \tag{1.19}$$

$$a_6 = 4{\omega_2}^2\zeta_3\omega_3\zeta_d\omega_d+2{\omega_2}^2\zeta_d\omega_d\tau+{\omega_1}^2{\omega_2}^2+2{\omega_2}^2\zeta_1\omega_1\tau+4{\omega_2}^2\zeta_1\omega_1\zeta_d\omega_d+{\omega_2}^2{\omega_d}^2+2{\omega_2}^2\zeta_3\omega_3\tau+4{\omega_2}^2\zeta_1\omega_1\zeta_3\omega_3+{\omega_2}^2{\omega_3}^2 \tag{1.20}$$

$$a_7 = 2\zeta_1\omega_1{\omega_2}^2 + {\omega_2}^2\tau + 2{\omega_2}^2\zeta_d\omega_d + 2{\omega_2}^2\zeta_3\omega_3 \tag{1.21}$$

$$a_8 = {\omega_2}^2 \tag{1.22}$$

because $a_8 \neq 1$ all of the coefficients (a_n and b_n) must be divided by a_6 before being plugged into the matix representation.

With θ as the output, the coefficients of the numerator polynomial are

$$b_0 = {\omega_2}^2{\omega_1}^2{\omega_3}^2{\omega_d}^2\tau \tag{1.23}$$

$$b_1 = 2{\omega_2}^2\zeta_1\omega_1{\omega_3}^2{\omega_d}^2\tau + 2{\omega_2}^2{\omega_1}^2\zeta_3\omega_3{\omega_d}^2\tau + 2\tau{\omega_d}^2\zeta_2\omega_2{\omega_3}^2{\omega_1}^2 \tag{1.24}$$

$$b_2 = 4\tau{\omega_d}^2\zeta_2\omega_2{\omega_3}^2\zeta_1\omega_1 + {\omega_2}^2{\omega_3}^2{\omega_d}^2\tau + 4\tau{\omega_d}^2\zeta_2\omega_2\zeta_3\omega_3{\omega_1}^2 + 4{\omega_2}^2\zeta_1\omega_1\zeta_3\omega_3{\omega_d}^2\tau + {\omega_2}^2{\omega_1}^2{\omega_d}^2\tau + {\omega_1}^2{\omega_3}^2{\omega_d}^2\tau \tag{1.25}$$

$$b_3 = 2\tau{\omega_d}^2\zeta_2\omega_2{\omega_1}^2+2{\omega_2}^2\zeta_1\omega_1{\omega_d}^2\tau+2{\omega_2}^2\zeta_3\omega_3{\omega_d}^2\tau+2\tau{\omega_d}^2\zeta_2\omega_2{\omega_3}^2+8\tau{\omega_d}^2\zeta_2\omega_2\zeta_3\omega_3\zeta_1\omega_1+2\tau{\omega_d}^2\zeta_1\omega_1{\omega_3}^2+2\tau{\omega_d}^2{\omega_1}^2\zeta_3\omega \tag{1.26}$$

$$b_4 = {\omega_2}^2\tau{\omega_d}^2 + 4\tau{\omega_d}^2\zeta_2\omega_2\zeta_1\omega_1 + 4\tau{\omega_d}^2\zeta_1\omega_1\zeta_3\omega_3 + \tau{\omega_d}^2{\omega_1}^2 + 4\tau{\omega_d}^2\zeta_2\omega_2\zeta_3\omega_3 + \tau{\omega_d}^2{\omega_3}^2 \tag{1.27}$$

$$b_5 = 2\tau{\omega_d}^2\zeta_2\omega_2 + 2\tau{\omega_d}^2\zeta_3\omega_3 + 2\tau{\omega_d}^2\zeta_1\omega_1 \tag{1.28}$$

$$b_6 = \tau{\omega_d}^2 \tag{1.29}$$

With $\ddot{x}$ as the output, the coefficients of the numerator polynomial are

$$b_0 = 0 \tag{1.30}$$

$$b_1 = 0 \tag{1.31}$$

$$b_2 = 0 \tag{1.32}$$

$$b_3 = 0 \tag{1.33}$$

$$b_4 = {\omega_2}^2{\omega_3}^2{\omega_d}^2\tau B_1 + {\omega_1}^2{\omega_3}^2{\omega_d}^2\tau B_2 + {\omega_2}^2{\omega_1}^2{\omega_d}^2\tau B_3 \tag{1.34}$$

$$b_5 = 2\tau{\omega_d}^2B_3\zeta_1\omega_1{\omega_2}^2+2\tau{\omega_d}^2B_3{\omega_1}^2\zeta_2\omega_2+2\tau{\omega_d}^2B_1\zeta_2\omega_2{\omega_3}^2+2\tau{\omega_d}^2B_1{\omega_2}^2\zeta_3\omega_3+2\tau{\omega_d}^2B_2\zeta_1\omega_1{\omega_3}^2+2\tau{\omega_d}^2B_2{\omega_1}^2\zeta_3\omega_3 \tag{1.35}$$

$$b_6 = \tau{\omega_d}^2B_3{\omega_2}^2+4\tau{\omega_d}^2B_3\zeta_1\omega_1\zeta_2\omega_2+\tau{\omega_d}^2B_3{\omega_1}^2+\tau{\omega_d}^2B_2{\omega_3}^2+4\tau{\omega_d}^2B_2\zeta_1\omega_1\zeta_3\omega_3+\tau{\omega_d}^2B_2{\omega_1}^2+\tau{\omega_d}^2B_1{\omega_3}^2+4\tau{\omega_d}^2B_1\zeta \tag{1.36}$$

$$b_7 = 2\tau{\omega_d}^2B_2\zeta_3\omega_3 + 2\tau{\omega_d}^2B_2\zeta_1\omega_1 + 2\tau{\omega_d}^2B_3\zeta_2\omega_2 + 2\tau{\omega_d}^2B_3\zeta_1\omega_1 + 2\tau{\omega_d}^2B_1\zeta_3\omega_3 + 2\tau{\omega_d}^2B_1\zeta_2\omega_2 \tag{1.37}$$

$$b_8 = \tau{\omega_d}^2B_1 + \tau{\omega_d}^2B_2 + \tau{\omega_d}^2B_3 \tag{1.38}$$

Finding the transfer function θ/d from the matrices according to

$$\frac{\theta}{d} = \mathbf{C}\left(s\mathbf{I} - \mathbf{A}\right)^{-1}\mathbf{B} + D \tag{1.39}$$

gives

$$\frac{\theta}{d} = \frac{\left(s^2 + 2\zeta_2\omega_2 s + {\omega_2}^2\right)\tau{\omega_d}^2}{s{\omega_2}^2\left(s+\tau\right)\left(s^2 + 2s\zeta_d\omega_d + {\omega_d}^2\right)} \tag{1.40}$$

giving us back what we started with and proving that the state-space representation is correct. Similarly the numerator of the transfer function $\ddot{x}/d$ from the matrices is

$$N_x = s^4\left(s^2+2\zeta_2\omega_2 s+{\omega_2}^2\right)\left(s^2+2s\zeta_3\omega_3+{\omega_3}^2\right){\omega_d}^2\tau B_1+s^4\left(s^2+2s\zeta_3\omega_3+{\omega_3}^2\right)\left(s^2+2s\zeta_1\omega_1+{\omega_1}^2\right){\omega_d}^2\tau B_2+s^4\left(\tag{1.41}$$

which is exactly what we started with in equation 1.3. and the denominator can be written as

$$D = s{\omega_2}^2\left(s+\tau\right)\left(s^2+2s\zeta_d\omega_d+{\omega_d}^2\right)\left(s^2+2s\zeta_3\omega_3+{\omega_3}^2\right)\left(s^2+2s\zeta_1\omega_1+{\omega_1}^2\right) \tag{1.42}$$

The state space matrices are given by

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & -{\omega_1}^2{\omega_3}^2{\omega_d}^2\tau & \frac{-2{\omega_2}^2{\omega_1}^2{\omega_3}^2\zeta_d\omega_d\tau-{\omega_2}^2{\omega_1}^2{\omega_3}^2{\omega_d}^2-2{\omega_2}^2{\omega_1}^2\zeta_3\omega_3{\omega_d}^2\tau-2{\omega_2}^2\zeta_1\omega_1{\omega_3}^2{\omega_d}^2\tau}{{\omega_2}^2} \quad \frac{-{\omega_2}^2{\omega_1}^2{\omega_d}^2\tau-4{\omega_2}^2\zeta_1\omega_1{\omega_3}^2\zeta_d\omega_d\tau-4{\omega_2}^2 \end{bmatrix} \tag{1.43}$$

$$\mathbf{C} = \begin{bmatrix} {\omega_1}^2{\omega_3}^2{\omega_d}^2\tau & \frac{2{\omega_2}^2\zeta_1\omega_1{\omega_3}^2{\omega_d}^2\tau+2{\omega_2}^2{\omega_1}^2\zeta_3\omega_3{\omega_d}^2\tau+2\tau{\omega_d}^2\zeta_2\omega_2{\omega_3}^2{\omega_1}^2}{{\omega_2}^2} & \frac{4\tau{\omega_d}^2\zeta_2\omega_2{\omega_3}^2\zeta_1\omega_1+{\omega_2}^2{\omega_3}^2{\omega_d}^2\tau+4\tau{\omega_d}^2\zeta_2\omega_2\zeta_3\omega_3{\omega_1}^2+4{\omega_2}^2\zeta_1 \\ 0 & -\frac{\left(\tau{\omega_d}^2B_1+\tau{\omega_d}^2B_2+\tau{\omega_d}^2B_3\right){\omega_1}^2{\omega_3}^2{\omega_d}^2\tau}{{\omega_2}^2} & -\frac{\left(\tau{\omega_d}^2B_1+\tau{\omega_d}^2B_2+\tau{\omega_d}^2B_3\right)\left(2{\omega_2}^2{\omega_1}^2{\omega_3}^2\zeta_d\omega_d\tau+{\omega_2}^2{\omega_1}^2{\omega_3}^2\omega_ \\ & & {\omega_2}^4 \end{bmatrix} \tag{1.44}$$

$$\mathbf{D} = \begin{bmatrix} 0 \\ \frac{\tau{\omega_d}^2B_1+\tau{\omega_d}^2B_2+\tau{\omega_d}^2B_3}{{\omega_2}^2} \end{bmatrix} \tag{1.45}$$

This state space system representation is output to the m-file `ccfaccelwnums3modes.m`.

1.3 State-Space Model Verification

Figure 1.1 overlays bode plots for the hydraulic actuator from the transfer function and state-space models with experimental data. Figure 1.2 does the same for the flexible base model. The close agreement between the transfer function and state-space curves gives me a high degree of confidence that the state-space derivation was done correctly. The reasonable agreement between the bode plots generated with the state-space model and the experimental data gives me reasonable confidence that the model will accurately represent the physical system.

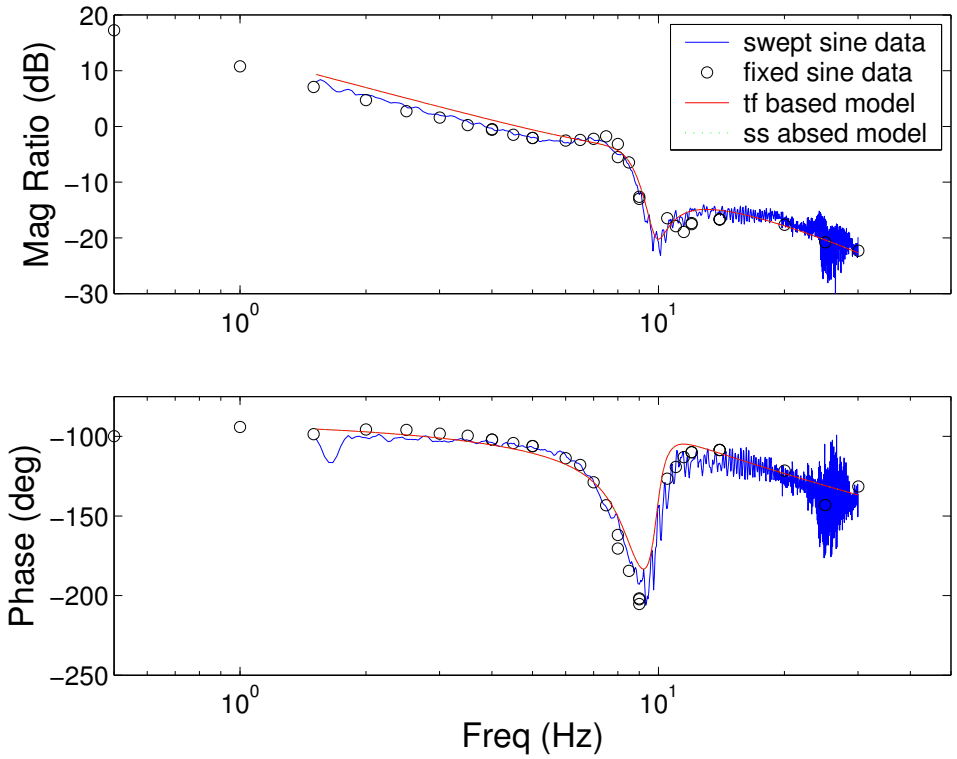

Figure 1.1: Comparison of bode plots from transefer function and state-space based models and experimental data for the hydraulic actuator. The input is the voltage into the servo-valve of joint 2. The output is joint 2 angular position. The parameters for the models are from a curve fit with a phase weighting of 0.1.

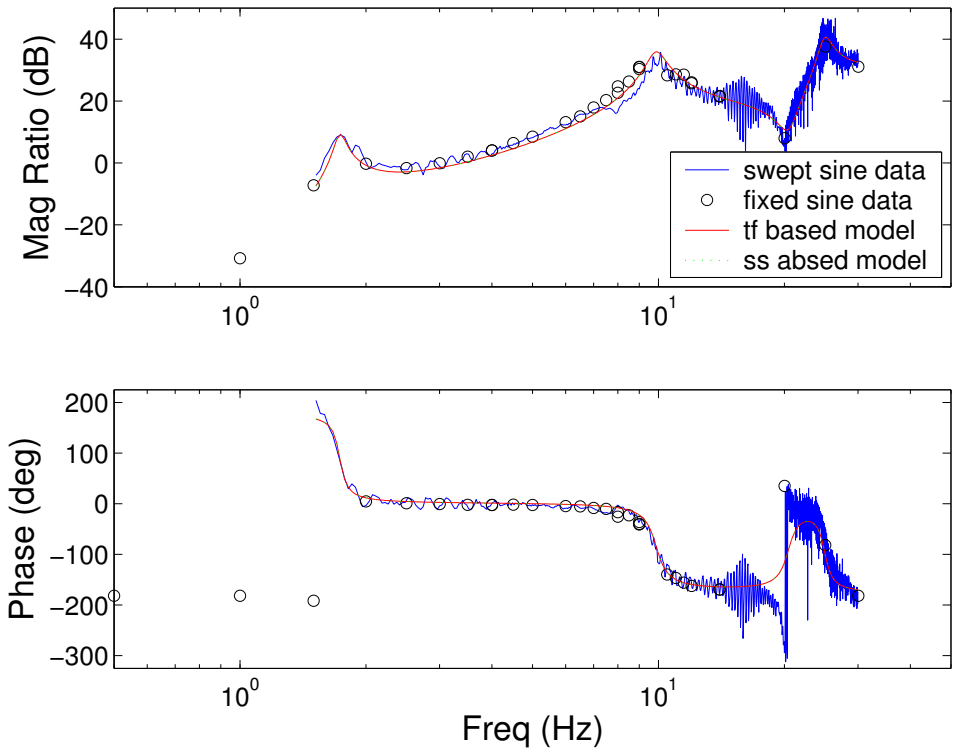

Figure 1.2: Comparison of bode plots from transfer function and state-space based models and experimental data for the flexible base. The input is joint 2 angular position and the output is base acceleration. The parameters for the models are from a curve fit with a phase weighting of 0.1.

1.4 Digital Controller/Observer Design

This file documents the design of a digital state feedback observer controller for SAMII. The open loop pole locations for SAMII operating around a nominal configuration of $(-90°, 90°, 90°, 0°, 0°, 0°)$ are

$$p_{ol} = \begin{bmatrix} 0 \\ -158.9 \\ -5.275 + 156.7i \\ -5.275 - 156.7i \\ -7.729 + 54.74i \\ -7.729 - 54.74i \\ -0.3245 + 10.88i \\ -0.3245 - 10.88i \end{bmatrix} \tag{1.46}$$

The poles for a system having unity θ feedback and no vibration suppression are

$$p_{\theta fb} = \begin{bmatrix} -5.275 + 156.7i \\ -5.275 - 156.7i \\ -128.3 \\ -4.214 + 52.84i \\ -4.214 - 52.84i \\ -37.56 \\ -0.3245 + 10.88i \\ -0.3245 - 10.88i \end{bmatrix} \tag{1.47}$$

The desired pole locations for the state feedback system being designed are

$$p_{des} = \begin{bmatrix} -110.9 - 110.9i \\ -110.9 + 110.9i \\ -128.3 \\ -37.48 - 37.48i \\ -37.48 + 37.48i \\ -37.56 \\ -7.698 - 7.698i \\ -7.698 + 7.698i \end{bmatrix} \tag{1.48}$$

Figure 1.3 plots the real vs. imaginary parts of these poles.

Figure 1.4 plots the real vs. imaginary parts of the digital poles.

Figure 1.5 shows a bode plot for θ/v for the closed loop state feedback system with the desired pole locations.

Figure 1.6 shows a bode plot for observer system designed by pole placement.

Figure 1.7 shows the step response of the state feedback controller observer without any noise.

Figure 1.8 shows the step response of the state feedback controller observer system from a simulation in Simulink with accelerometer noise.

While the acceleration signals from Figures 1.7 and 1.8 look promising, this controller made the thhird mode of the flexible base unstable when implemented experimentally. This third mode instability lead me to refit a wider frequency range of my bode data to include the third mode in my models and to begin redesigning a new controller similar to this one but considering the third mode.

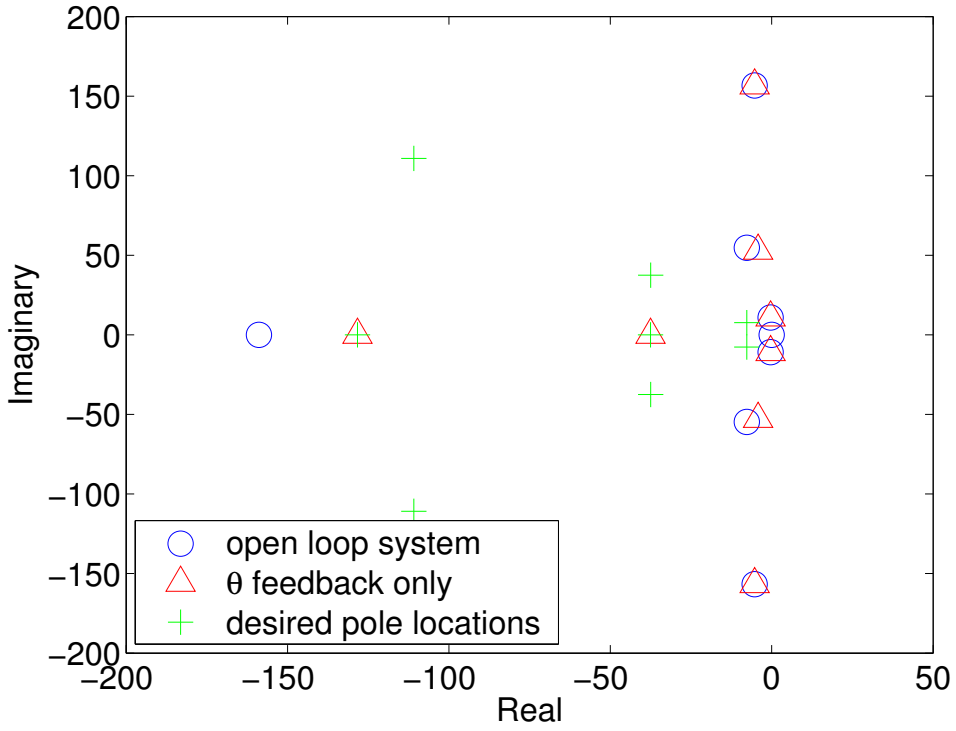

Figure 1.3: Pole locations for the open loop system, a controller for SAMII that has only θ feedback (i.e. no vibration suppression), and the desired pole locations for a full state feedback system.

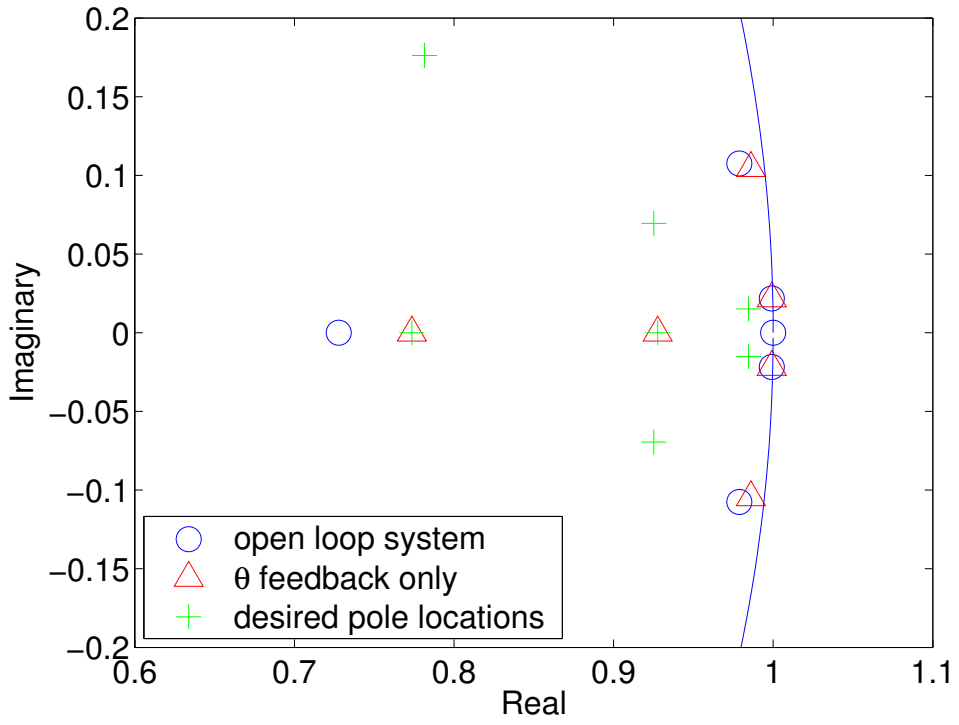

Figure 1.4: Digital pole locations for the open loop system, a controller for SAMII that has only θ feedback (i.e. no vibration suppression), and the desired pole locations for a full state feedback system.

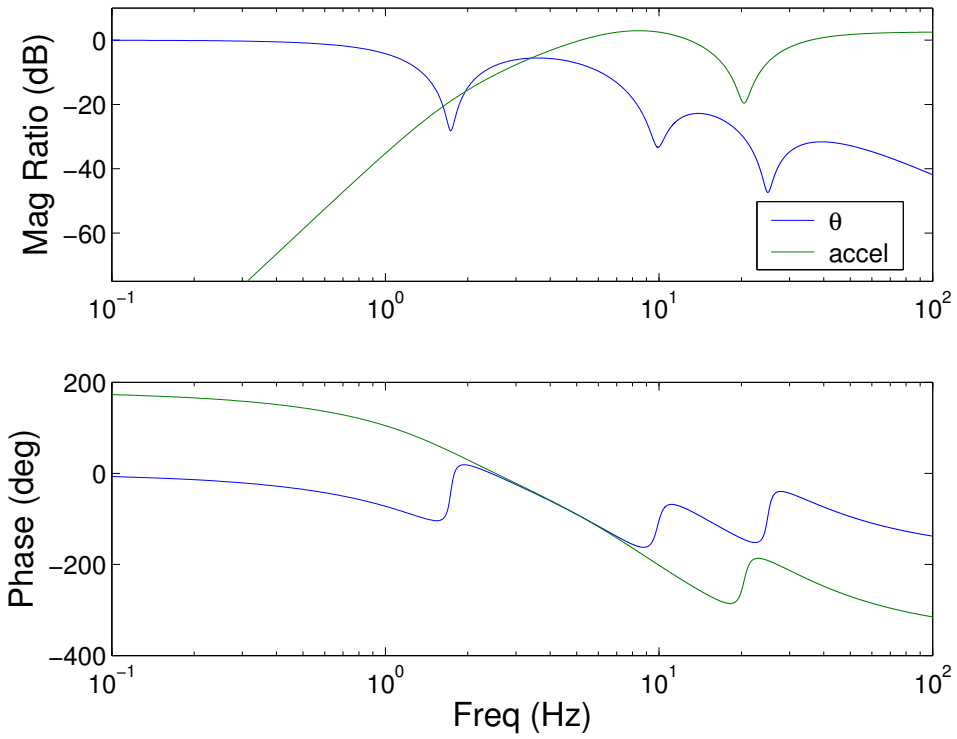

Figure 1.5: Bode plots for θ/v and $\ddot{x}/\theta$ for the SAMII control system with the desired closed loop poles.

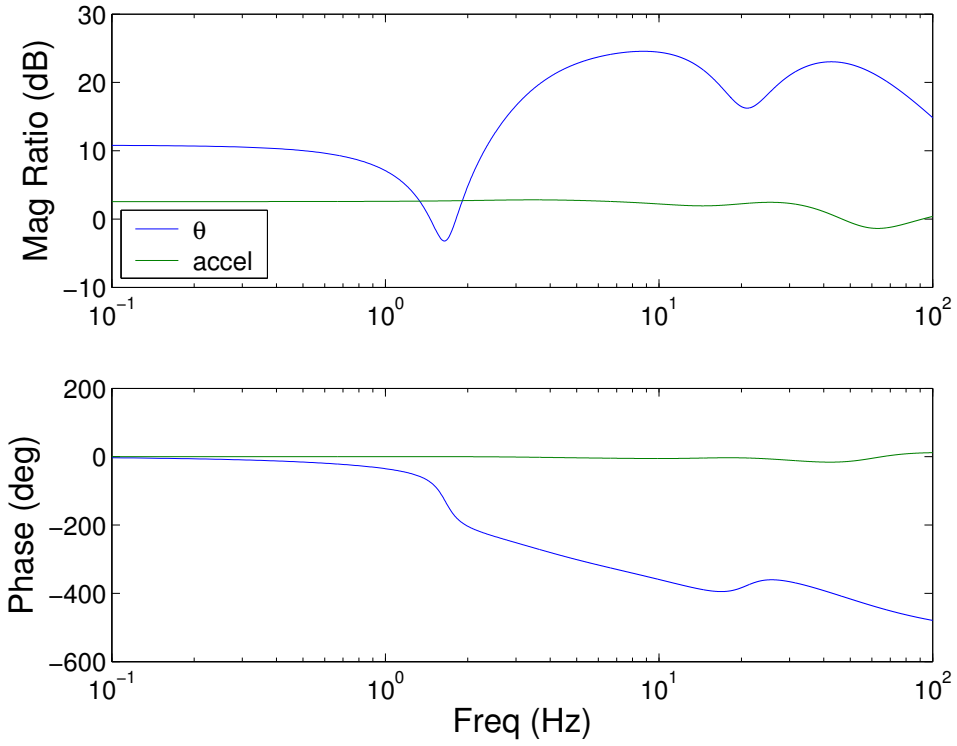

Figure 1.6: Bode plots for an observer system designed by pole placement.

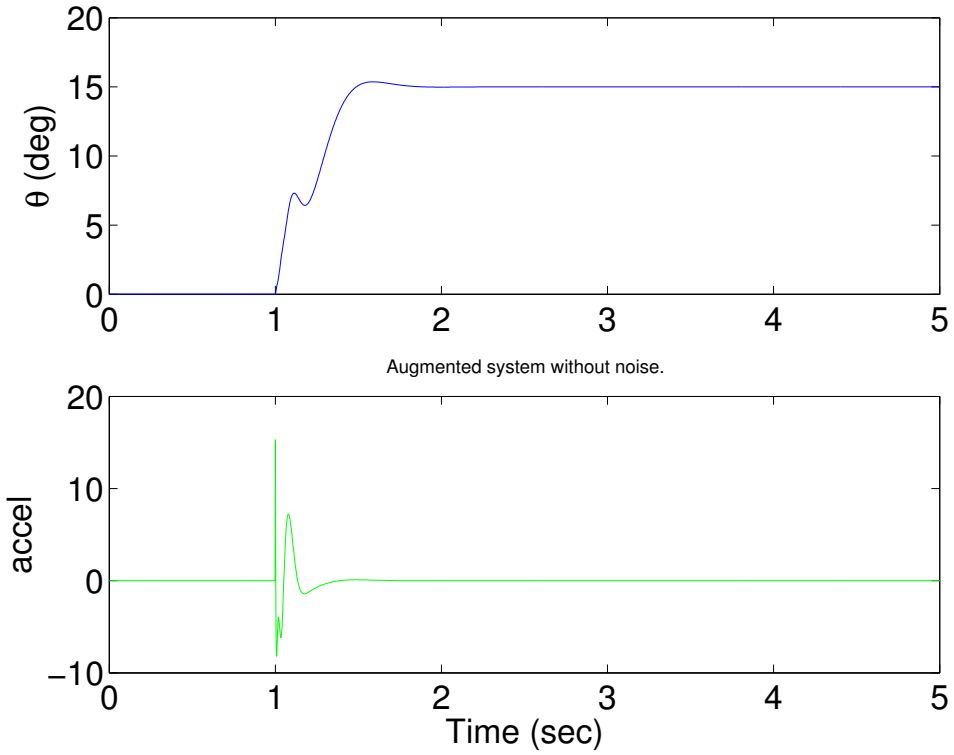

Figure 1.7: Step response for the state feedback controller observer system with no noise.

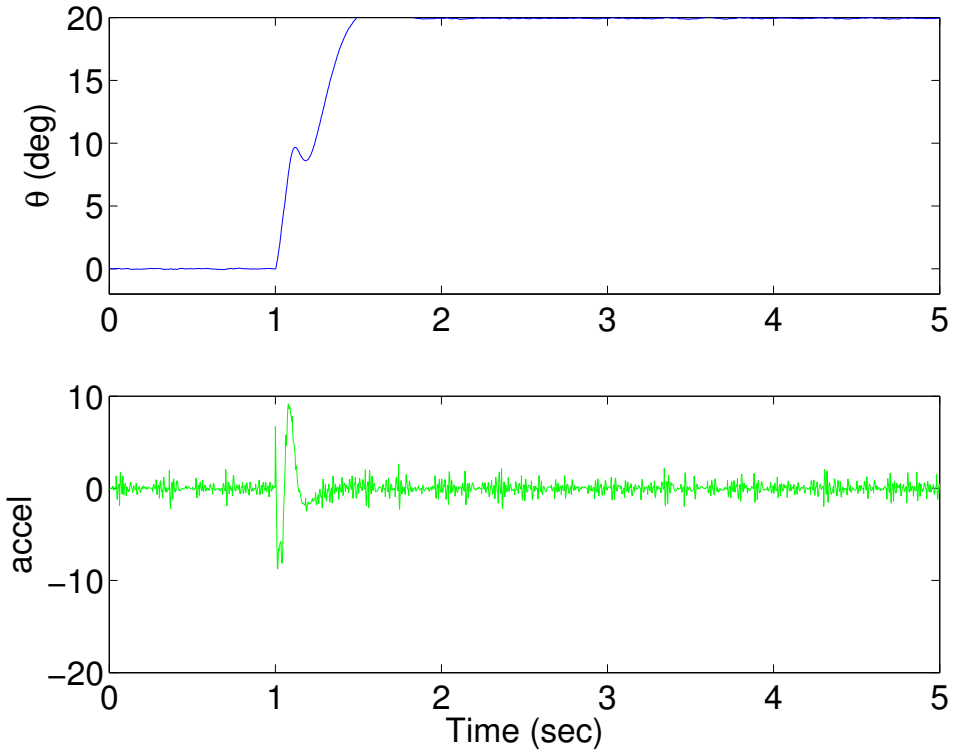

Figure 1.8: Step response for the state feedback controller observer system from a Simulink simulation with accelerometer noise.

1.5 Implementation: Dither/Noise Problem

The controller seemed to perform fairly well in simulation and it seemed the time had come to implement it experimentally. The real world was not as kind. Enabling accelerometer feedback with this controller resulted in the valve oscillating back and forth at a fairly high frequency. While I could hear the valve moving, the arm just started to drift. I recreated this response by sending a dither signal to the valve.

Figure 1.9 shows the effect on θ_2 of switching on the state-space controller and then switching on acceleration feedback. (When acceleration feedback is turned off, a constant value of 0 is used to replace the measured acceleration value). The state-space controller can be switched on without much problem. However, when accelerometer feedback is switched on, noise sensitivity causes a serious problem. Note that I am able to recreate the drifting of the joint angle by sending a 200Hz voltage to joint 2. Simply setting the voltage to a 0 volt DC value does not cause the drift. Figure 1.12 shows that the DC value of the input voltage when the acceleration feedback has been switched on is approximately zero, but the voltage is switching fairly rapidly from the positive to the negative saturation limit. This seems to indicate that the joint is being dithered by the input voltage.

Figure 1.10 shows the effect on a_2 of switching on the state-space controller and then switching on acceleration feedback.

Figure 1.11 shows the effect on v_2 of switching on the state-space controller and then switching on acceleration feedback.

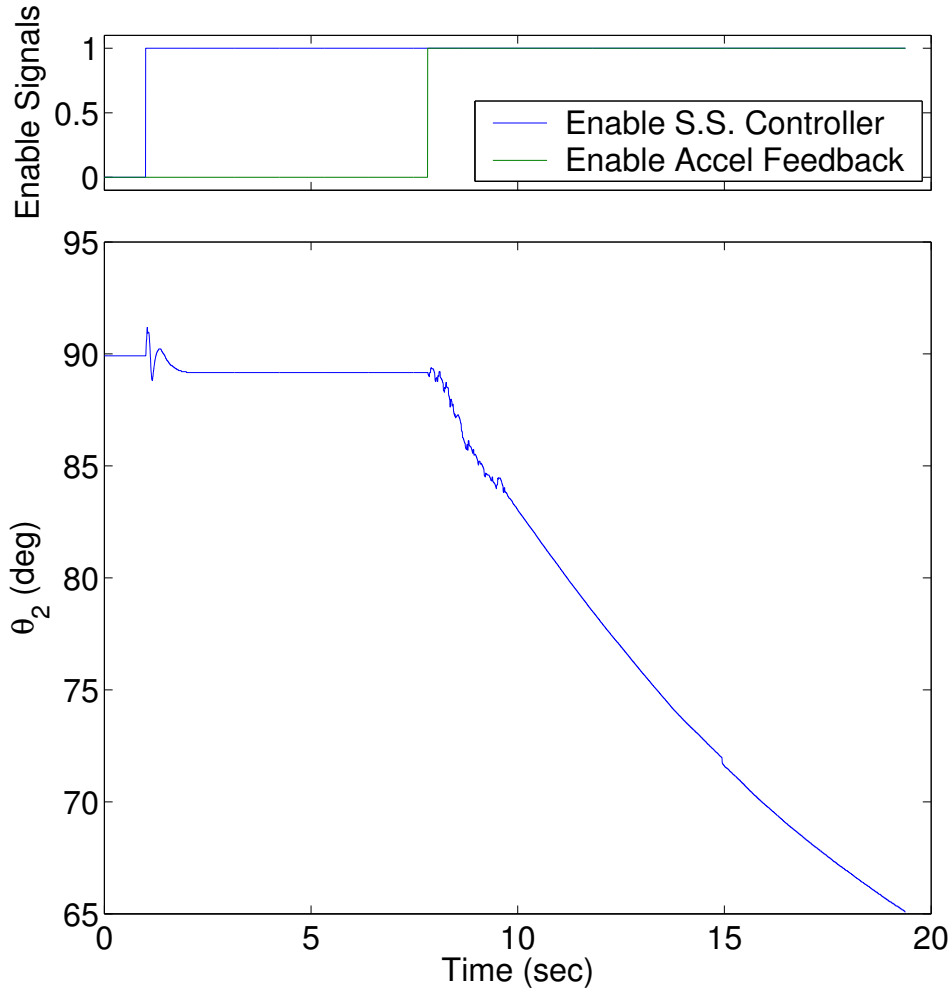

Figure 1.9: The effect on θ_2 of switching on the state-space controller and then switching on acceleration feedback.

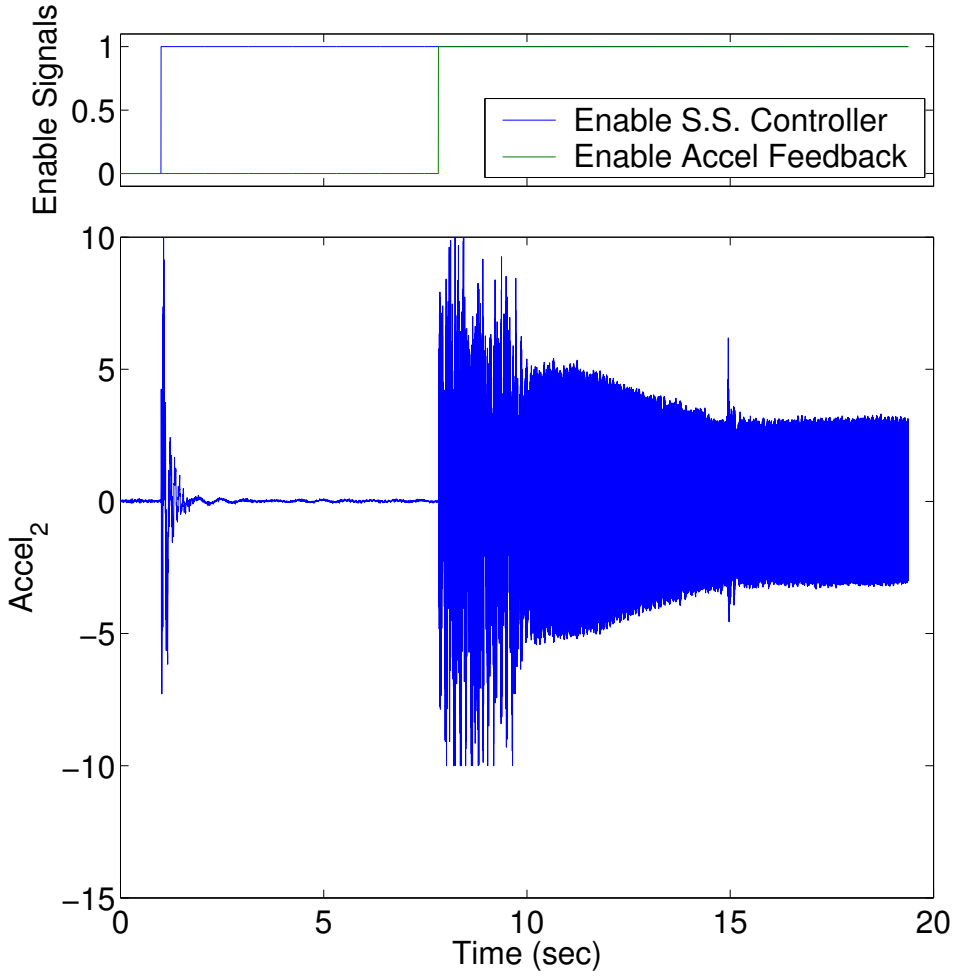

Figure 1.10: The effect on a_2 of switching on the state-space controller and then switching on acceleration feedback.

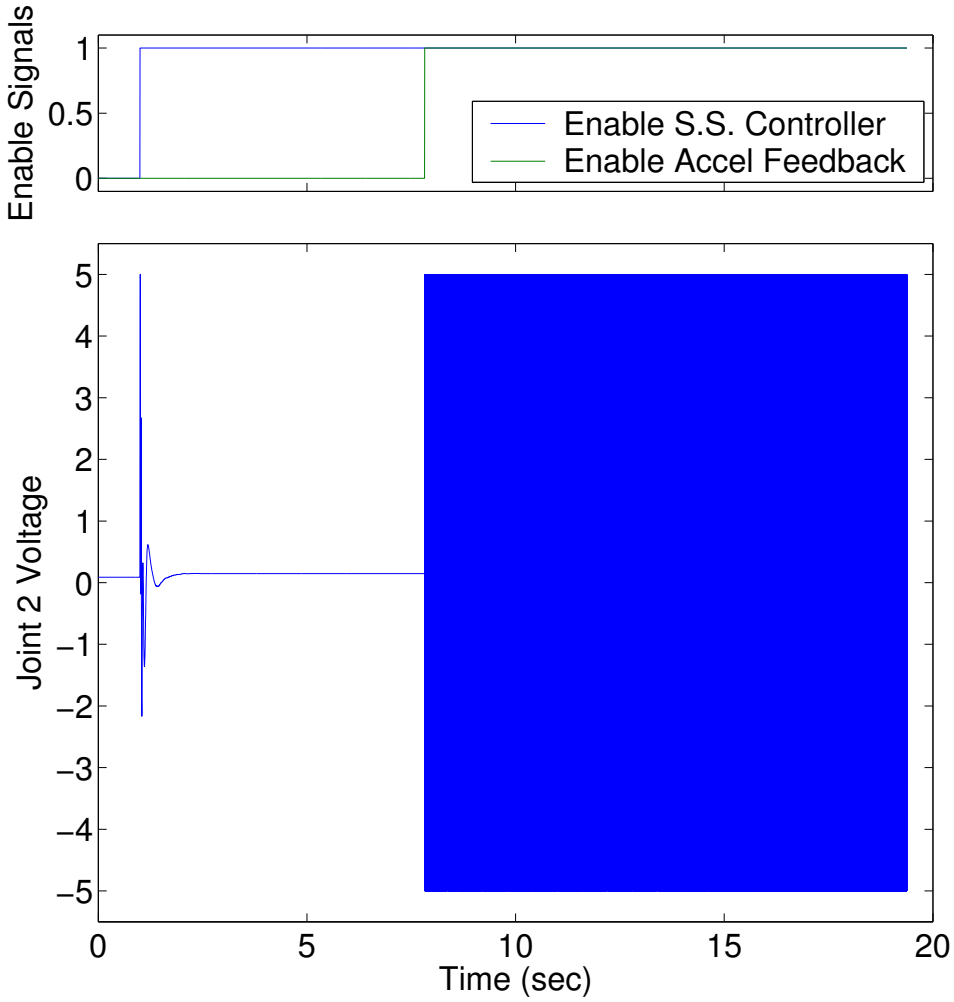

Figure 1.11: The effect on v_2 of switching on the state-space controller and then switching on acceleration feedback.

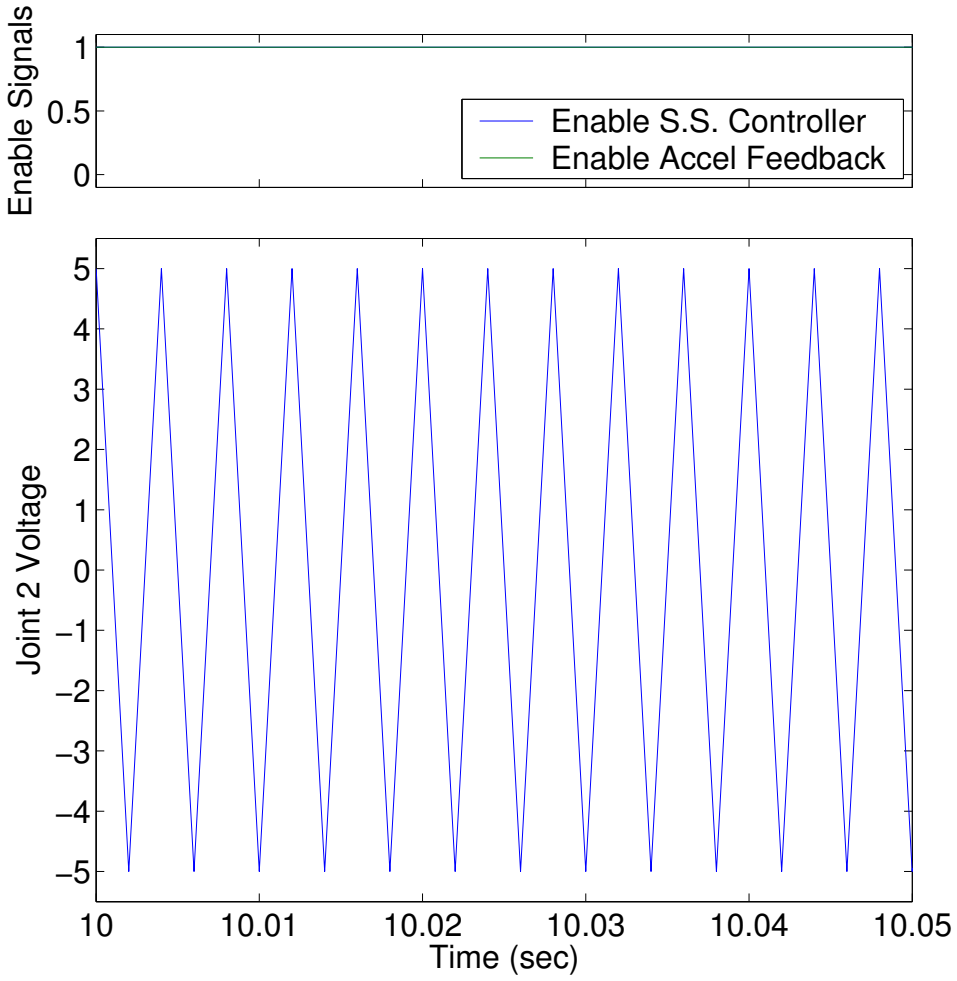

Figure 1.12: Zooming in on Figure 1.11.

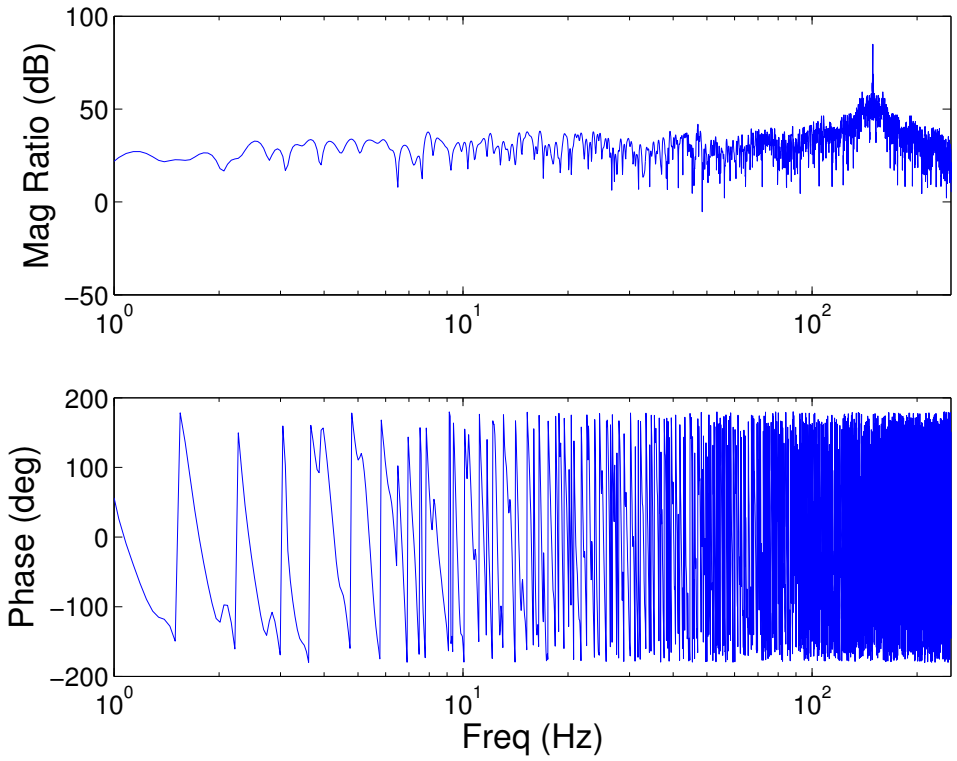

Figure 1.13: Bode plot of the voltage sent to the joint 2 actuator.

A.1 Matlab Files Used

A.1.1 Main Matlab File

The main Matlab file is `digital_design_01_09_04.m` This file designs a digital pole placement controller/observer to place the first three poles of SAMII. The pole placement is done by leaving the magnitude of the pole (the undamped natural frequency) unchanged, but changing the damping ratio to a specified value.

This function calls `loadparams_ss_01_09_04_s4_m3` to define the state-space matrices A, B, C, and D based on coefficents from curve fitting bode data.

This function outputs a text file that can be cut and pasted into a intialization block of a masked subsystem in Simulink for easy implementation of this controller in simulation or experimentally using real-time workshop.

A.1.2 Additional Matlab Files

`ccfaccel3modes.m`

This file starts with the transfer functions for the hydraulic actuator and base acceleration including the first three modes of the flexible base. From there it derives a controllable canonical state space representation of the system. This file outputs the results of its derivation to a LaTeX file for easy readability. The output file is `ccfaccel3modes.tex`.

The transfer functions used in this derivation include an s^4 term in the numerators of the transfer functions between the base accleration and the theta input (angular position) (i.e. $\ddot{x}/\theta$).

It also creates a Matlab m-file that defines the state space matrices in terms of the variables used in this derivation. Editting this file so that it begins with numerically defining each of these variables (i.e. w1=2*pi*10), gives an m-file that has the properly defined state space representation of the system. The output m-file generated by this file is `ccfaccelwnums3modes.m` and the editted version is `ccfaccelwnums3modes_editted.m` which actually evalutes the lines of the files `bodefit_11_14_03_pw=0_1.txt` where the parameters are all defined.

`ccfaccelwnums3modes.m`

This file was created by the m-file `ccfaccel3modes.m` and it contains a CCF state-space model for SAMII based on SISO transfer functions about a nominal operating point. To make this function useful, it must be editted so that the coefficents it uses are numerically defined before they are used.

`ccfaccelwnums3modes_editted.m`

This appears to be one of two main files in this directory, the other being `digital_design_01_09_04.m`. This file loads curve fit parameters and then defines the state-space model based on those parameters. It calls `compbodeplots_01_09_04.m` to compare bode plots from transfer function and state-space models with one another and with experimental data.

The parameters used by this file are stored in the file `bodefit_11_14_03_pw=0_1.txt`.

clpolelocs.m

```
A=varargin{1};
B=varargin{2};
C=varargin{3};
deszeta=varargin{4};

  varargout{1}=eigcl;
```

This function finds the desired closed loop poles of a state feedback controller for SAMII by first finding the poles with unity theta feedback and then moving any complex poles to the same natural frequency but with the specified damping ratio.

compbodeplots_01_09_04.m

This file generates the bode plots from transfer function and state-space models of the system. This file is called by `ccfaccelwnums3modes_editted.m`. This function calls `samiimodel`.

digital_design_01_09_04.m

This file designs a digital pole placement controller/observer to place the first three poles of SAMII. The pole placement is done by leaving the magnitude of the pole (the undamped natural frequency) unchanged, but changing the damping ratio to a specified value.

This function calls `loadparams_ss_01_09_04_s4_m3` to define the state-space matrices A, B, C, and D based on coefficents from curve fitting bode data.

This function outputs a text file that can be cut and pasted into a intialization block of a masked subsystem in Simulink for easy implementation of this controller in simulation or experimentally using real-time workshop.

load_data_s4_m3_01_09_03.m

This file loads experimental swept and fixed sine data. The swept sine data is loaded into global variables. This file is called by `ccfaccelwnums3modes_editted.m`.

loadparams_ss_01_09_04_s4_m3.m

This function loads system parameters from curve fit data and then defines state-space matrices based on those parameters. It also defines a transfer function model based on the same parameters for comparison purposes. The outputs of this function as A,B,C, and D matrices and transfer functions for the hydraulic actuators and flexible base.

This function is very similar to `ccfaccelwnums3modes_editted.m` except that it is implemented as a function so that all of the parameters defined to create the matrices do not clutter up the workspace.

This function is called by `digital_design_01_09_04.m` so that it will have a system model to design the controller around.

matrix_ssobs.mat

samiimodel.m

This function is called by `compbodeplots_01_09_04.m`. This function takes a vector of coeffiecients as an input and outputs the bode magnitudes and phases for both the actuator and flexible base models.

```
coeffsin=varargin{1};

varargout{1}=act_fit_mag;
varargout{2}=base_fit_mag;
varargout{3}=act_fit_ph;
varargout{4}=base_fit_ph;
```

sim_w_obs_R13_1_sat_fixed_011504.mdl

ssbodedata.m

This function generates bode plot data for a state-space model.

A.1.3 Verbatim Matlab Files

`ssJan04_fullstory_verb_apndx.pdf`

Chapter 2

Encoder Trouble Shooting

SAMII's joint 2 encoder has intermittently seemed to read incorrectly. The most obvious sign of this problem has been sending joint 2 to 0° and having the arm be visibly 5-10° above horizontal.

In an attempt to determine whether this is a hardware or software problem (i.e. is the encoder not sending Quanser the correct electrical signals, or is some problem with the real-time kernel causing Quanser to miss some counts of the encoder), I hooked the encoder up to an oscilloscope. The oscilloscope data was then saved to an ascii file over an ethernet connection and the data was analyzed in Matlab.

2.1 Verification of Oscilloscope Measurement

This document compares the encoder measurement of the Quanser/Wincon software with that from an oscilloscope (the oscilloscope measurement was processed using Matlab code to convert it to degrees from the two digital signals). During this testing, the Quanser reading appeared to be correct, i.e. with joint 2 reading 0° the link appeared to be truly horizontal.

The idea behind doing this was to have a way to externally verify the encoder reading. If the reading in Quanser appears to be off, the oscilloscope and the Matlab data processing can be used to verify whether or not the encoder signals that are reaching the Quanser board is correct. If the oscilloscope data shows that the correct signals are reaching the board, then the problem is in the software or the Quanser board. If he oscilloscope and the Quanser reading are the same, but both seem off compared to the actual physical configuration of the joint, then the problem is in the hardware somewhere (a loose wire or something).

Since the Quanser signal seemed to be reading correctly when this data was collected, this data only verifies that the oscilloscope was able to sample fast enough and didn't miss any counts.

Figure 2.1 shows the joint response $\theta_2(t)$ when a commanded step input is given to move the joint from 90° to 0°. It takes SAMII a little over 1 second to complete this motion. Figures 2.2-2.4 zoom in on various portions of Figure 2.1, showing that the agreement between Quanser and the oscilloscope reading is quite good.

2.2 Problem Captured with Oscilloscope

Fortunately, I was able to capture this intermittent problem with the oscilloscope before too long. Figures 2.5-2.16 show data from tests when the encoder reading appeared to be a problem.

One way that the problem manifested itself was an apparent flat spot in the graph of angle versus time under a constant voltage input to the valve (Figure 2.5, for example). A constant voltage should give a constant velocity, so that this graph should be a straight line. Figures 2.5, 2.9, and 2.13 all show a glitch around 35°. The

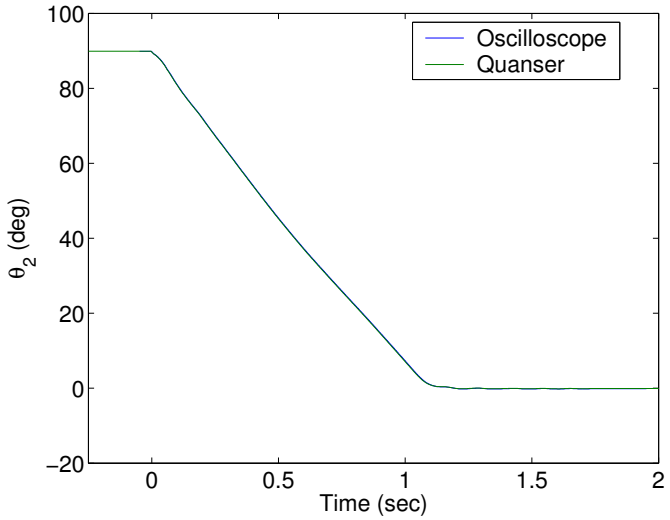

Figure 2.1: Verifying the Quanser encoder reading for joint 2 using an oscilloscope.

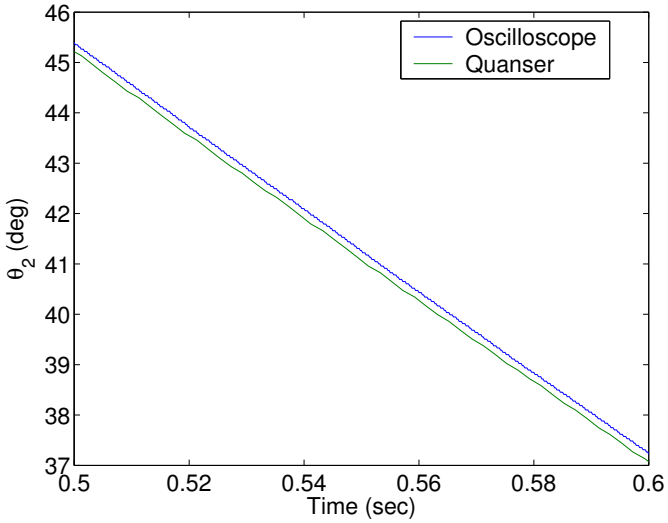

Figure 2.3: Verifying the Quanser encoder reading for joint 2 using an oscilloscope. (Zooming in on the middle of Figure 2.1).

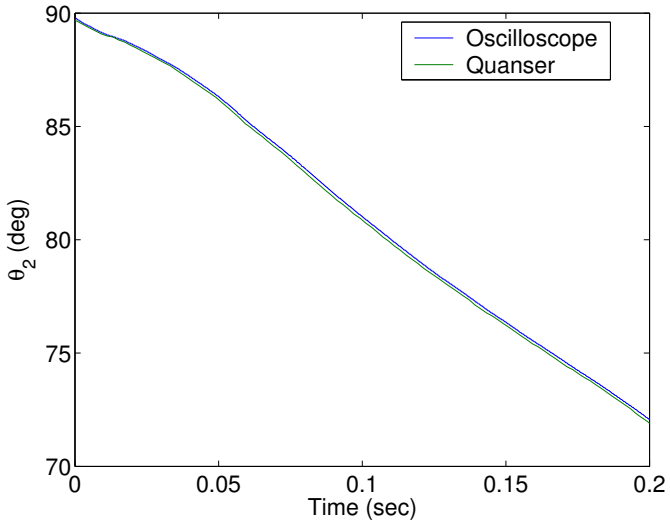

Figure 2.2: Verifying the Quanser encoder reading for joint 2 using an oscilloscope. (Zooming in on the beginning of Figure 2.1).

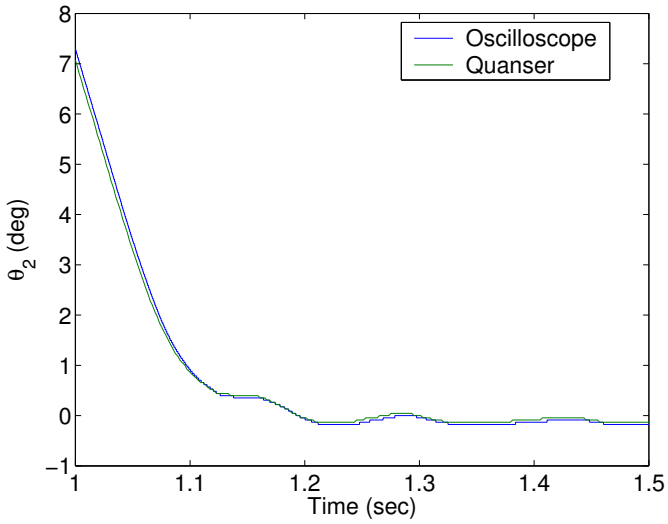

Figure 2.4: Verifying the Quanser encoder reading for joint 2 using an oscilloscope. (Zooming in on the end of Figure 2.1).

repeatability of this error seems to indicate that it is a hardware rather than a software problem.

2.2.1 Test 1

Figure 2.5 shows the angle vs. time response for the first test preformed on 01/28/02. The angle is calculated based on encoder signals captured directly by the oscilloscope. The response should be essentially a straight line. There is a flat spot in this line around 35°. Figure 2.6 zooms in on the portion of Figure 2.5 with the flat spot.

Figure 2.7 shows the digital encoder signals for the same time as in Figure 2.5. The flat spot is where there appears to be a blank spot in Figure 2.7 around 0.7 seconds. Figure 2.8 zooms in on this blank portion of Figure 2.7. Figure 2.8 would seem to indicate that the arm just stop moving for a few hundredths of a second. This did not appear to be the case watching the movement (it appears to move with constant velocity to the naked eye). But, I don't know if I would be able to see such a glitch without the aid of high speed video. Fortunately, the results in sections 2.2.2 and 2.2.3 are more conclusive.

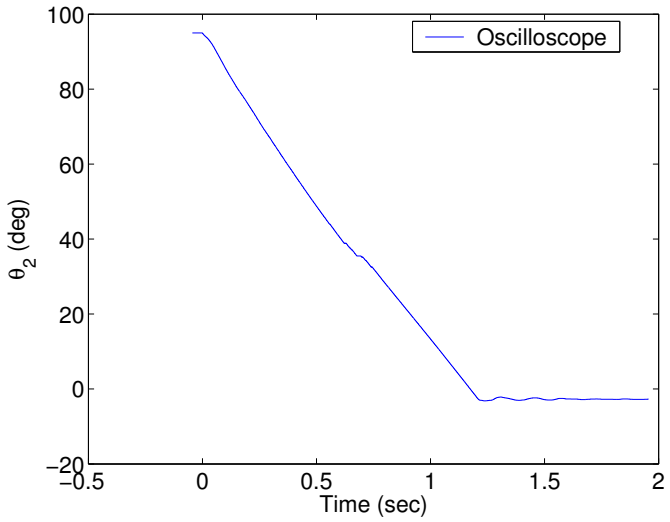

Figure 2.5: Encoder readings with trouble-shooting wire connected directly to the encoder. Test #1. Sampling frequency of 5kHz. The +5V is externally supplied and the encoder is not connected to Quanser at all.

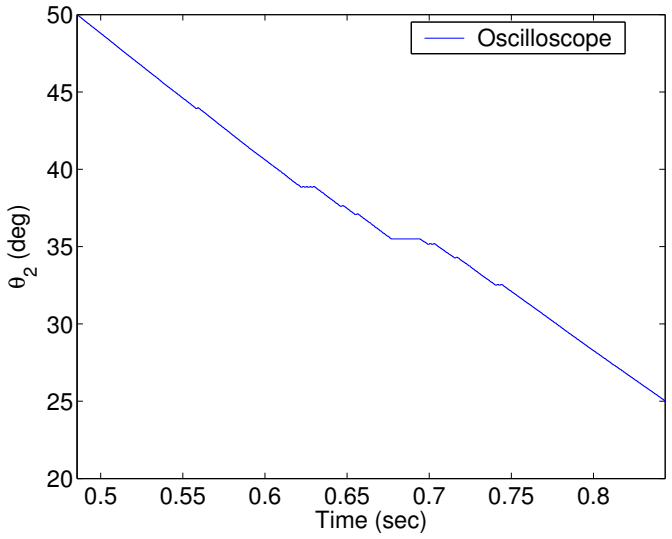

Figure 2.6: Encoder readings with trouble-shooting wire connected directly to the encoder. Test #1 . (Zooming in on Figure 2.5.)

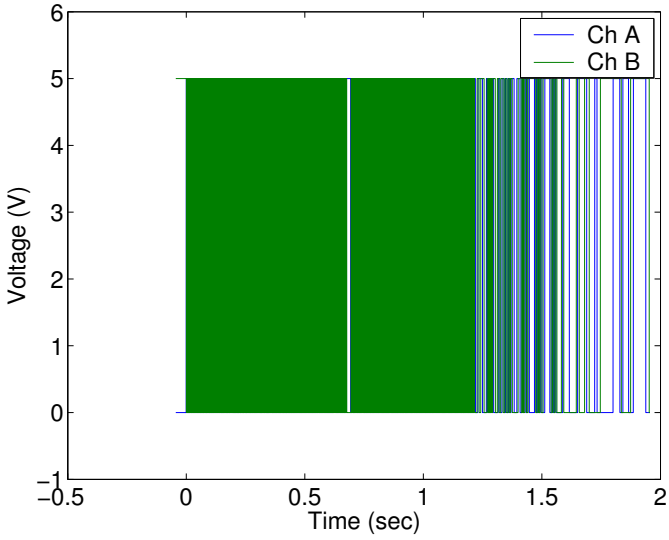

Figure 2.7: Encoder signals after software processing for noise. Test #1. Sampling frequency of 5kHz.

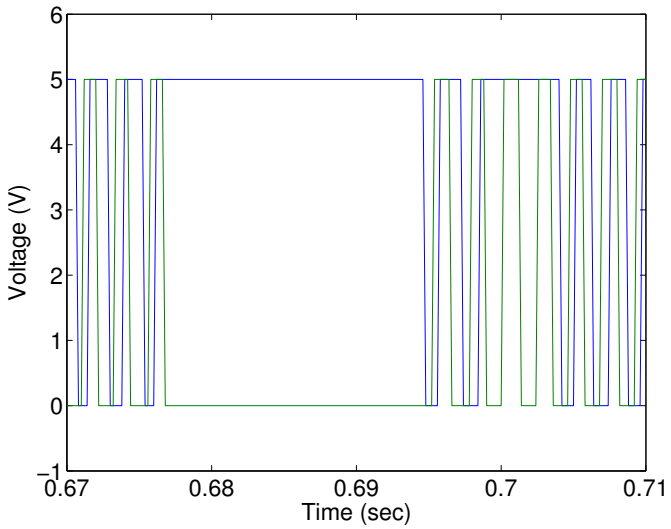

Figure 2.8: Encoder signals after software processing for noise. Test #1. (Zooming in on Figure 2.7.)

2.2.2 Test 2

Figure 2.9 shows the angle vs. time response for the second test preformed on 01/28/02. Just as in test 1, there is a flat spot in this line around 35°. Figure 2.10 zooms in on the portion of Figure 2.9 with the flat spot.

Figure 2.11 shows the digital encoder signals for the same time as in Figure 2.10. Around 0.37 seconds the encoder seems to do something strange, similar to what was seen in Figure 2.7. Figure 2.12 zooms in on this portion of Figure 2.11.

There is one key difference between Figure 2.12 and what was seen before in Figure 2.8. Instead of both encoder channels stopping for a short time (as in Figure 2.8), channel B reads several counts while channel A holds constant. The only way that this could actually reflect the true motion of system is if the arm where to vibrate back and forth through 1 count, so that only channel B was triggered. The encoder resolution is roughly 0.0439° (8192 counts per revolution or 360/8192°). It seems highly unlikely that this oscillation in joint 2 angle actually happened. This would seem to indicate that something is wrong with the sensor.

The results from Test 4 (section 2.2.3), seem to show slightly more clearly that this is in fact a sensor issue.

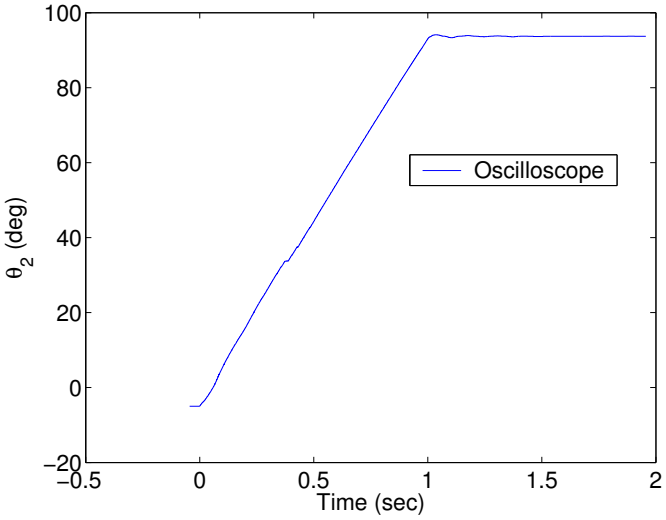

Figure 2.9: Encoder readings with trouble-shooting wire connected directly to the encoder. Test #2. Sampling frequency of 5kHz. The +5V is externally supplied and the encoder is not connected to Quanser at all.

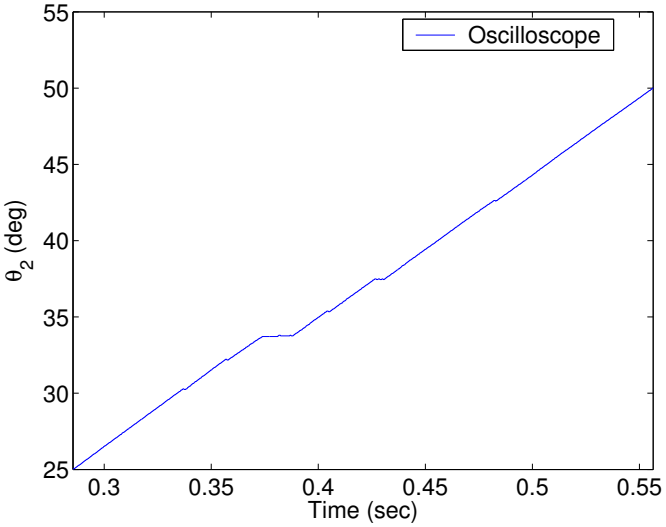

Figure 2.10: Encoder readings with trouble-shooting wire connected directly to the encoder. Test #2 . (Zooming in on Figure 2.9.)

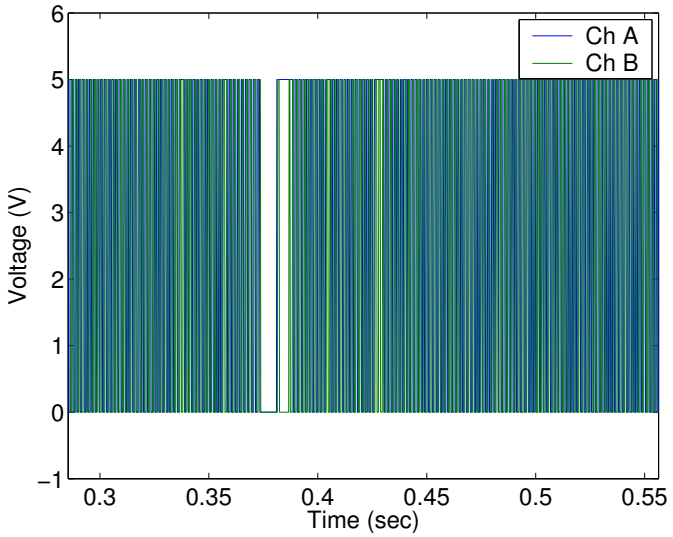

Figure 2.11: Encoder signals after software processing for noise. Test #2. Sampling frequency of 5kHz.

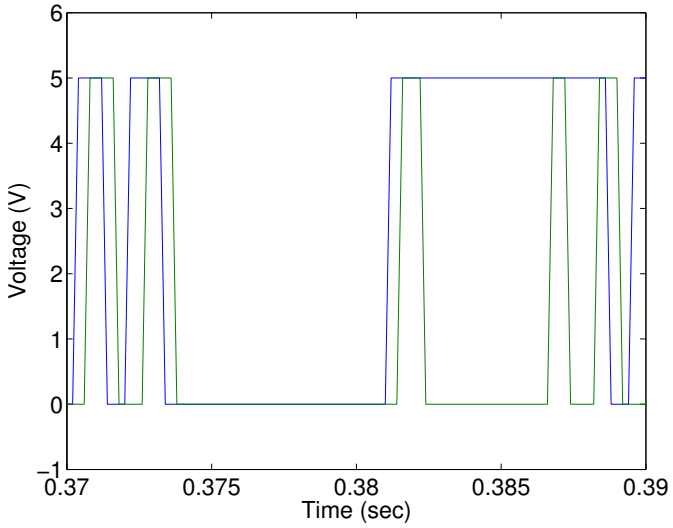

Figure 2.12: Encoder signals after software processing for noise. Test #2. (Zooming in on Figure 2.11.)

2.2.3 Test 4

Figure 2.13 shows the angle vs. time response for the fourth test preformed on 01/28/02. Just as in tests 1 and 2, there is a flat spot in this line around 35°. Figure 2.14 zooms in on the portion of Figure 2.13 with the flat spot.

Figure 2.15 shows the digital encoder signals for the same time as in Figure 2.14. Figure 2.16 zooms in on the portion of Figure 2.15 where there is a flat spot it Figure 2.14.

Figure 2.16 seems to indicate that there is a sensor problem slightly more definitively than Figure 2.12 did. In Figure 2.16, channel B appears to continue reading as if the arm is moving with constant velocity while channel A appears to stop. This reinforces the constant velocity motion that would be expected and seems verified by the naked eye observation of the motion. It appears that something was wrong with channel A during this portion of this test. Perhaps the encoder lines for channel A are damaged or dirty in the portion of the encoder wheel around 35° for joint 2.

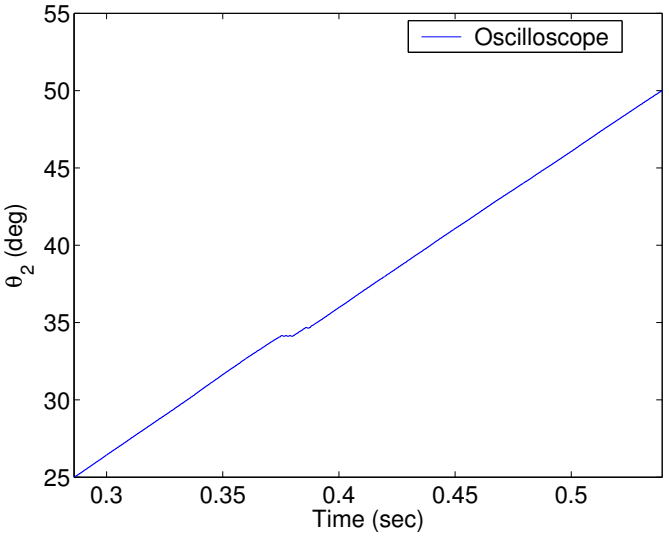

Figure 2.14: Encoder readings with trouble-shooting wire connected directly to the encoder. Test #4 . (Zooming in on Figure 2.13.)

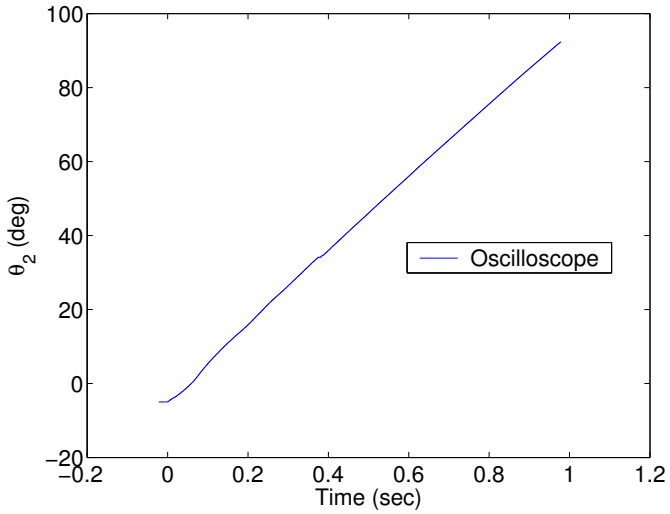

Figure 2.13: Encoder readings with trouble-shooting wire connected directly to the encoder. Test #4. Sampling frequency of 10kHz. The +5V is externally supplied and the encoder is not connected to Quanser at all.

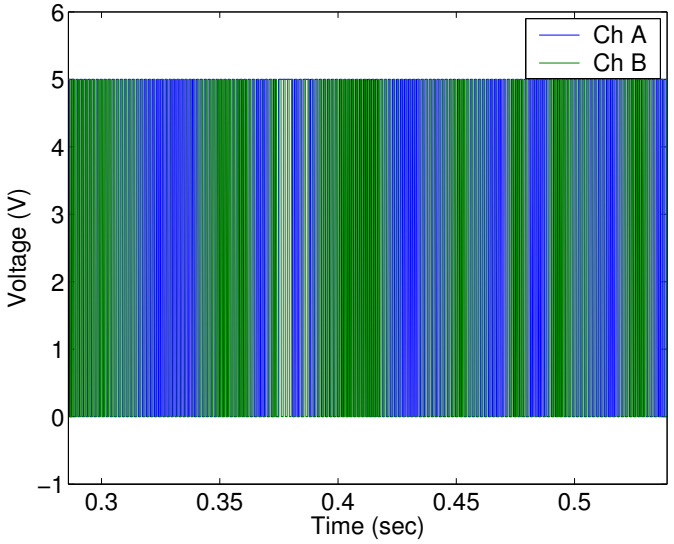

Figure 2.15: Encoder signals after software processing for noise. Test #4. Sampling frequency of 10kHz.

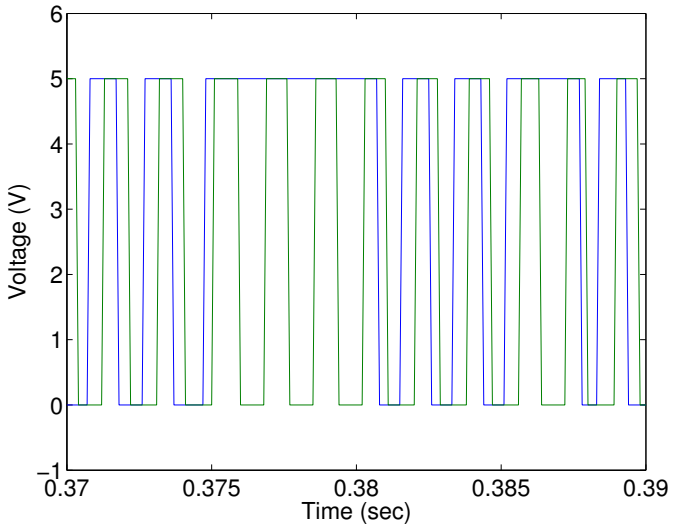

Figure 2.16: Encoder signals after software processing for noise. Test #4. (Zooming in on Figure 2.15.)

2.3 Conclusion and Encoder Cleaning

Based on the results of these tests, it was concluded that there is something wrong with the encoder (this is not a software nor does it appear to be a wiring problem - a loose wire would not be this consistent).

As a first attempt to solving this problem, the encoder was taken apart and the wheel was cleaned. The problem has not reoccurred since the encoder wheel has been cleaned. Unless the problem begins to reoccur, I will assume that the problem has been solved.

A.1 Matlab Files Used

A.1.1 Main Matlab File

The main Matlab file is `process_encoder_data_01_28_04.m` This file processes the encoder data from testing done on 01/28/02. The encoder signals are first cleaned up. Anything below 1 volt is set to zero and anything above 3 volts is set to 5. The two channels of the encoder are then processed to determine the angle vs. time signal. The angle versus time as well as encoder signals versus time are plotted. Based on a vector of x-axis limits determined manually, plots are generated that zoom in on the relevant portions of the curves. Encoder data was not recorded with the Quanser software for these tests becuase previous testing had already showed good agreement between Quanser and the oscilloscope mesaurements.

A.1.2 Additional Matlab Files

`process_encoder_data_01_23_04.m`

This file processes the encoder data from testing done on 01/24/02. The encoder signals are first cleaned up. Anything below 1 volt is set to zero and anything above 3 volts is set to 5. The two channels of the encoder are then processed to determine the angle vs. time signal. The angle versus time is plotted. The results of this processing algorithm are overlayed with the angle recorded using the Quanser software. The agreement is quite good, showing that this algorithm can be trusted and that the oscilloscope is sampling fast engouh that counts are not being missed.

A.1.3 Verbatim Matlab Files

`encoder_troubleshoot_fullstory.pdf`

Chapter 3

Transfer Matrix Analysis

3.1 Transfer Matrix Derivation for a Beam Element

The shear force can be written as

$$V = A_1 \cosh\left(\frac{\beta\, x}{l}\right) + A_2 \sinh\left(\frac{\beta\, x}{l}\right) + A_3 \cos\left(\frac{\beta\, x}{l}\right) + A_4 \sin\left(\frac{\beta\, x}{l}\right) \tag{3.1}$$

The shear force is related to the displacement according to

$$\frac{dV}{dx} = -\mu\omega^2 w \tag{3.2}$$

This can be rearranged to give

$$w = \frac{-1}{\mu\omega^2}\frac{dV}{dx} \tag{3.3}$$

β is defined to be

$$\beta^4 = \frac{\omega^2 l^4 \mu}{EI} \tag{3.4}$$

Solving equation 3.4 for $\mu\omega^2$ and substituting the result in equation 3.3 gives the following equation:

$$w = \frac{-l^4}{\beta^4 EI}\frac{dV}{dx} \tag{3.5}$$

Differentiating equation 3.1 and substituting the result into equation 3.5 gives

$$w = -l^3\left(A_1 \sinh\left(\frac{\beta\, x}{l}\right) + A_2 \cosh\left(\frac{\beta\, x}{l}\right) - A_3 \sin\left(\frac{\beta\, x}{l}\right) + A_4 \cos\left(\frac{\beta\, x}{l}\right)\right)\beta^{-3} EI^{-1} \tag{3.6}$$

By definition,

$$\psi = -\frac{dw}{dx} \tag{3.7}$$

Differentiating equation 3.6 gives

$$\psi = l^2 \left(A_1 \cosh\left(\frac{\beta x}{l}\right) + A_2 \sinh\left(\frac{\beta x}{l}\right) - A_3 \cos\left(\frac{\beta x}{l}\right) - A_4 \sin\left(\frac{\beta x}{l}\right)\right) \beta^{-2} EI^{-1} \tag{3.8}$$

By definition,

$$M = EI\frac{d\psi}{dx} \tag{3.9}$$

Differentiating equation 3.8 gives

$$M = l \left(A_1 \sinh\left(\frac{\beta x}{l}\right) + A_2 \cosh\left(\frac{\beta x}{l}\right) + A_3 \sin\left(\frac{\beta x}{l}\right) - A_4 \cos\left(\frac{\beta x}{l}\right)\right) \beta^{-1} \tag{3.10}$$

Equations 3.1, 3.6, 3.8, and 3.10 can be rewritten in matrix form as

$$\mathbf{z}(x) = \mathbf{B}(x)\mathbf{a} \tag{3.11}$$

where

$$\mathbf{z} = \begin{bmatrix} -w \\ \psi \\ M \\ V \end{bmatrix} \tag{3.12}$$

$$\mathbf{a} = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{bmatrix} \tag{3.13}$$

$$\mathbf{B} = \begin{bmatrix} al\sinh\left(\frac{\beta x}{l}\right)\beta^{-3} & al\cosh\left(\frac{\beta x}{l}\right)\beta^{-3} & -al\sin\left(\frac{\beta x}{l}\right)\beta^{-3} & al\cos\left(\frac{\beta x}{l}\right)\beta^{-3} \\ a\cosh\left(\frac{\beta x}{l}\right)\beta^{-2} & a\sinh\left(\frac{\beta x}{l}\right)\beta^{-2} & -a\cos\left(\frac{\beta x}{l}\right)\beta^{-2} & -a\sin\left(\frac{\beta x}{l}\right)\beta^{-2} \\ l\sinh\left(\frac{\beta x}{l}\right)\beta^{-1} & l\cosh\left(\frac{\beta x}{l}\right)\beta^{-1} & l\sin\left(\frac{\beta x}{l}\right)\beta^{-1} & -l\cos\left(\frac{\beta x}{l}\right)\beta^{-1} \\ \cosh\left(\frac{\beta x}{l}\right) & \sinh\left(\frac{\beta x}{l}\right) & \cos\left(\frac{\beta x}{l}\right) & \sin\left(\frac{\beta x}{l}\right) \end{bmatrix} \tag{3.14}$$

and

$$a = \frac{l^2}{EI} \tag{3.15}$$

Evaluating equation 3.11 at each end of the beam gives

$$\mathbf{z}(0) = \mathbf{B}(0)\mathbf{a} \tag{3.16}$$

and

$$\mathbf{z}(L) = \mathbf{B}(L)\mathbf{a} \tag{3.17}$$

Solving equation 3.16 for $\mathbf{a}$ gives

$$\mathbf{a} = [\mathbf{B}(0)]^{-1}\,\mathbf{z}(0) \tag{3.18}$$

Substituting equation 3.18 into equation 3.17 gives

$$\mathbf{z}(L) = \mathbf{B}(L)\left[\mathbf{B}(0)\right]^{-1}\mathbf{z}(0) \tag{3.19}$$

The transfer matrix between $\mathbf{z}(0)$ and $\mathbf{z}(L)$ can then be written as

$$\mathbf{U}_{L0} = \mathbf{B}(L)\left[\mathbf{B}(0)\right]^{-1} \tag{3.20}$$

$\mathbf{B}(0)$ is given by

$$\mathbf{B}(0) = \begin{bmatrix} 0 & \frac{al}{\beta^3} & 0 & \frac{al}{\beta^3} \\ \frac{a}{\beta^2} & 0 & -\frac{a}{\beta^2} & 0 \\ 0 & \frac{l}{\beta} & 0 & -\frac{l}{\beta} \\ 1 & 0 & 1 & 0 \end{bmatrix} \tag{3.21}$$

and $\mathbf{B}(L)$ is given by

$$\mathbf{B}(L) = \begin{bmatrix} \frac{al\sinh(\beta)}{\beta^3} & \frac{al\cosh(\beta)}{\beta^3} & -\frac{al\sin(\beta)}{\beta^3} & \frac{al\cos(\beta)}{\beta^3} \\ \frac{a\cosh(\beta)}{\beta^2} & \frac{a\sinh(\beta)}{\beta^2} & -\frac{a\cos(\beta)}{\beta^2} & -\frac{a\sin(\beta)}{\beta^2} \\ \frac{l\sinh(\beta)}{\beta} & \frac{l\cosh(\beta)}{\beta} & \frac{l\sin(\beta)}{\beta} & -\frac{l\cos(\beta)}{\beta} \\ \cosh(\beta) & \sinh(\beta) & \cos(\beta) & \sin(\beta) \end{bmatrix} \tag{3.22}$$

Substituing equations 3.21 and 3.22 into equation 3.20 gives

$$\mathbf{U}_{L0} = \begin{bmatrix} 1/2\,\cosh(\beta) + 1/2\,\cos(\beta) & 1/2\,\frac{l(\sinh(\beta)+\sin(\beta))}{\beta} \\ 1/2\,\frac{\beta(\sinh(\beta)-\sin(\beta))}{l} & 1/2\,\cosh(\beta) + 1/2\,\cos(\beta) \\ 1/2\,\frac{\beta^2(\cosh(\beta)-\cos(\beta))}{a} & 1/2\,\frac{\beta l(\sinh(\beta)-\sin(\beta))}{a} \\ 1/2\,\frac{\beta^3(\sinh(\beta)+\sin(\beta))}{al} & 1/2\,\frac{\beta^2(\cosh(\beta)-\cos(\beta))}{a} \end{bmatrix}$$

$$\begin{bmatrix} 1/2\,\frac{a(\cosh(\beta)-\cos(\beta))}{\beta^2} & 1/2\,\frac{al(\sinh(\beta)-\sin(\beta))}{\beta^3} \\ 1/2\,\frac{a(\sinh(\beta)+\sin(\beta))}{\beta l} & 1/2\,\frac{a(\cosh(\beta)-\cos(\beta))}{\beta^2} \\ 1/2\,\cosh(\beta) + 1/2\,\cos(\beta) & 1/2\,\frac{l(\sinh(\beta)+\sin(\beta))}{\beta} \\ 1/2\,\frac{\beta(\sinh(\beta)-\sin(\beta))}{l} & 1/2\,\cosh(\beta) + 1/2\,\cos(\beta) \end{bmatrix} \tag{3.23}$$

Rewriting the transfer matrix in terms of subexperssions gives

$$\mathbf{U}_{L0} = \begin{bmatrix} c(1) & 1/2\,\frac{lc(4)}{\beta} & 1/2\,\frac{ac(3)}{\beta^2} & 1/2\,\frac{alc(2)}{\beta^3} \\ 1/2\,\frac{\beta c(2)}{l} & c(1) & 1/2\,\frac{ac(4)}{\beta l} & 1/2\,\frac{ac(3)}{\beta^2} \\ 1/2\,\frac{\beta^2 c(3)}{a} & 1/2\,\frac{\beta lc(2)}{a} & c(1) & 1/2\,\frac{lc(4)}{\beta} \\ 1/2\,\frac{\beta^3 c(4)}{al} & 1/2\,\frac{\beta^2 c(3)}{a} & 1/2\,\frac{\beta c(2)}{l} & c(1) \end{bmatrix} \tag{3.24}$$

where

$$c = \begin{bmatrix} 1/2\,\cosh(\beta) + 1/2\,\cos(\beta) \\ \sinh(\beta) - \sin(\beta) \\ \cosh(\beta) - \cos(\beta) \\ \sinh(\beta) + \sin(\beta) \end{bmatrix} \tag{3.25}$$

Alternatively, the transfer matrix between $\mathbf{z}(L)$ and $\mathbf{z}(0)$ can then be written as (reversing the order of multilplication)

$$\mathbf{U}_{0L} = \mathbf{B}(0)\left[\mathbf{B}(L)\right]^{-1} \tag{3.26}$$

which can be written as

$$\mathbf{U}_{0L} = \begin{bmatrix} 1/2\,\cosh(\beta) + 1/2\,\cos(\beta) & -1/2\,\frac{l(\sinh(\beta)+\sin(\beta))}{\beta} \\ 1/2\,\frac{\beta\,(-\sinh(\beta)+\sin(\beta))}{l} & 1/2\,\cosh(\beta) + 1/2\,\cos(\beta) \\ 1/2\,\frac{\beta^2(\cosh(\beta)-\cos(\beta))}{a} & 1/2\,\frac{\beta\,l(-\sinh(\beta)+\sin(\beta))}{a} \\ -1/2\,\frac{\beta^3(\sinh(\beta)+\sin(\beta))}{al} & 1/2\,\frac{\beta^2(\cosh(\beta)-\cos(\beta))}{a} \end{bmatrix}$$

$$\begin{bmatrix} 1/2\,\frac{a(\cosh(\beta)-\cos(\beta))}{\beta^2} & 1/2\,\frac{al(-\sinh(\beta)+\sin(\beta))}{\beta^3} \\ -1/2\,\frac{a(\sinh(\beta)+\sin(\beta))}{\beta\,l} & 1/2\,\frac{a(\cosh(\beta)-\cos(\beta))}{\beta^2} \\ 1/2\,\cosh(\beta) + 1/2\,\cos(\beta) & -1/2\,\frac{l(\sinh(\beta)+\sin(\beta))}{\beta} \\ 1/2\,\frac{\beta\,(-\sinh(\beta)+\sin(\beta))}{l} & 1/2\,\cosh(\beta) + 1/2\,\cos(\beta) \end{bmatrix} \tag{3.27}$$

which can be rewritten in terms of subexperssions to give

$$\mathbf{U}_{0L} = \begin{bmatrix} c0L(1) & -1/2\,\frac{lc0L(4)}{\beta} & 1/2\,\frac{ac0L(3)}{\beta^2} & 1/2\,\frac{alc0L(2)}{\beta^3} \\ 1/2\,\frac{\beta\,c0L(2)}{l} & c0L(1) & -1/2\,\frac{ac0L(4)}{\beta\,l} & 1/2\,\frac{ac0L(3)}{\beta^2} \\ 1/2\,\frac{\beta^2 c0L(3)}{a} & 1/2\,\frac{\beta\,lc0L(2)}{a} & c0L(1) & -1/2\,\frac{lc0L(4)}{\beta} \\ -1/2\,\frac{\beta^3 c0L(4)}{al} & 1/2\,\frac{\beta^2 c0L(3)}{a} & 1/2\,\frac{\beta\,c0L(2)}{l} & c0L(1) \end{bmatrix} \tag{3.28}$$

where

$$c0L = \begin{bmatrix} 1/2\,\cosh(\beta) + 1/2\,\cos(\beta) \\ -\sinh(\beta) + \sin(\beta) \\ \cosh(\beta) - \cos(\beta) \\ \sinh(\beta) + \sin(\beta) \end{bmatrix} \tag{3.29}$$

Using $\mathbf{B}(0)$ and $\mathbf{z}(0)$ to determine the coefficents $\mathbf{a}$ gives

$$\mathbf{a}_0 = \begin{bmatrix} 1/2\ V0 \\ 1/2\ \frac{\beta\ M0}{l} \\ 1/2\ V0 \\ -1/2\ \frac{\beta\ M0}{l} \end{bmatrix} \tag{3.30}$$

3.1.1 Cantilevered Beam

To illustrate the difference between $\mathbf{U}_{0L}$ and $\mathbf{U}_{L0}$, consider the response of a cantilevered beam. The boundary conditions for a cantilevered beam are given by

$$z_0 = \begin{bmatrix} 0 \\ 0 \\ M0 \\ V0 \end{bmatrix} \tag{3.31}$$

and

$$z_L = \begin{bmatrix} -w_L \\ \psi_L \\ 0 \\ 0 \end{bmatrix} \tag{3.32}$$

In order for the equation $\mathbf{z}(0) = \mathbf{U}\mathbf{z}(L)$ to hold for all values of $w(L)$ and $\psi(L)$ the submatrix

$$\mathbf{U}_{(1:2,1:2)} = \begin{bmatrix} 1/2\ \cosh(\beta) + 1/2\ \cos(\beta) & -1/2\ \frac{l(\sinh(\beta)+\sin(\beta))}{\beta} \\ 1/2\ \frac{\beta\,(-\sinh(\beta)+\sin(\beta))}{l} & 1/2\ \cosh(\beta) + 1/2\ \cos(\beta) \end{bmatrix} \tag{3.33}$$

must have zero determinant. This determinant can be written as

$$|\mathbf{U}_{(1:2,1:2)}| = 1/2\ \cosh(\beta)\cos(\beta) + 1/2 \tag{3.34}$$

Setting this determinant equal to zero gives a transcendental equation that can be solved for β. For the first mode of a cantilevered beam,

$$\beta = 1.8751 \tag{3.35}$$

Substituing this value for β into equation 3.25, gives

$$c = \begin{bmatrix} 1.5189 \\ 2.23 \\ -3.637 \\ 4.1381 \end{bmatrix} \tag{3.36}$$

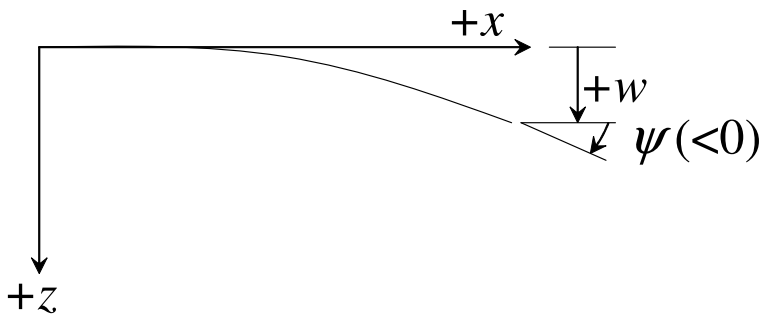

Figure 3.1: The first mode of a cantilevered beam with the sign conventions used for this derivation.

Substituing this value for β into the $\mathbf{U}_{0L}$ and $\mathbf{U}_{L0}$ matrices gives

$$\mathbf{U}_{0L} = \begin{bmatrix} 1.5189 & -1.1034\,l & 0.51721\,a & -0.16912\,al \\ -2.0907\,l^{-1} & 1.5189 & -1.1034\,\frac{a}{l} & 0.51721\,a \\ 6.3939\,a^{-1} & -2.0907\,\frac{l}{a} & 1.5189 & -1.1034\,l \\ -13.641\,\frac{1}{al} & 6.3939\,a^{-1} & -2.0907\,l^{-1} & 1.5189 \end{bmatrix} \tag{3.37}$$

and

$$\mathbf{U}_{L0} = \begin{bmatrix} 1.5189 & 1.1034\,l & 0.51721\,a & 0.16912\,al \\ 2.0907\,l^{-1} & 1.5189 & 1.1034\,\frac{a}{l} & 0.51721\,a \\ 6.3939\,a^{-1} & 2.0907\,\frac{l}{a} & 1.5189 & 1.1034\,l \\ 13.641\,\frac{1}{al} & 6.3939\,a^{-1} & 2.0907\,l^{-1} & 1.5189 \end{bmatrix} \tag{3.38}$$

The first row of the matrix equation $\mathbf{z}(0) = \mathbf{U}_{0L}\mathbf{z}(L)$ is

$$-1.5189\,w_L - 1.1034\,l\psi_L = 0 \tag{3.39}$$

While the first row of the matrix equation $\mathbf{z}(0) = \mathbf{U}_{L0}\mathbf{z}(L)$ (note that this is using the incorrect $\mathbf{U}$) is

$$-1.5189\,w_L + 1.1034\,l\psi_L = 0 \tag{3.40}$$

Figure 3.1 shows the sign convention for this derivation and how it applies to the first mode of a cantilevered beam. ψ is measured from horizontal with counter-clockwise being positive. w is positive in the downward direction. Positive w will produce negative ψ. This is what is found in equation 3.39, but the opposite of what is predicted with equation 3.40.

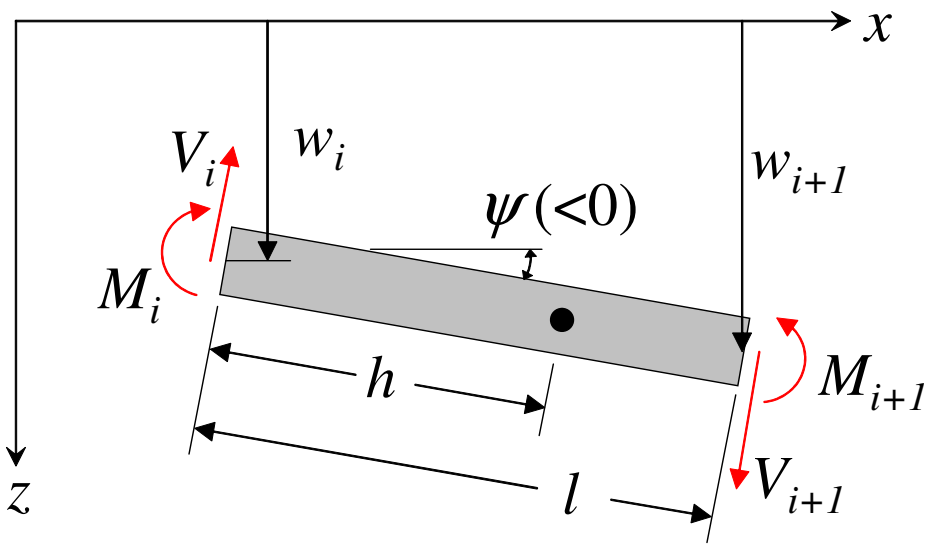

Figure 3.2: Sketch of the rigid body being analyzed in this section.

B.0.2 Matlab File Used

Matlab File Summary

This file uses the Matlab file `Bmatrix_deriv.m` This file derives the transfer matrix for a beam element based on the solution of the differential equation as in Chapter 5 of Pestel.

The B transfer matrix is derived for transfering the state vector from left to right or from right to left: i.e. $z_R = U_{RL} z_L$ or $z_L = U_{LR} z_R$.

Verbatim Matlab Text

`Bmatrix_deriv_editted_verb_apndx.pdf`

3.1 Rigid Mass Transfer Matrix Derivation

3.1.1 $\mathbf{R}_{i,i+1}$

If w is positive downward and ψ is positive in the counter-clockwise direction, then from Figure 3.2 it can be shown that

$$w_{i+1} = w_i - \psi\, l \tag{3.41}$$

To derive the matrix $\mathbf{R}_{i,i+1}$ that transfers from station $i+1$ to station i (i.e. $\mathbf{z}_0 = \mathbf{R}_{01}\mathbf{z}_1$), equation 3.41 must be solved for w_i:

$$w_i = w_{i+1} + \psi\, l \tag{3.42}$$

This equation gives the first row of the $\mathbf{R}_{i,i+1}$ matrix

$$\mathbf{R}_{i,i+1}(1,:) = \begin{bmatrix} 1 & -l & 0 & 0 \end{bmatrix} \tag{3.43}$$

Note that $-w$ is used in the state vector so that the first row of the $\mathbf{R}_{i,i+1}$ matrix represents the equation

$$-w_i = \mathbf{R}_{i,i+1}(1,:) \begin{bmatrix} -w_{i+1} \\ \psi_{i+1} \\ M_{i+1} \\ V_{i+1} \end{bmatrix} \tag{3.44}$$

Because the link is rigid, the value of ψ does not change from one end to the other, so that the second row of the matrix is

$$\mathbf{R}_{i,i+1}(2,:) = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix} \tag{3.45}$$

The sum of the moments about any point on a rigid body is given by

$$\sum \mathbf{M}_A = m\mathbf{r}_{g/A} \times \mathbf{a}_A + \dot{\mathbf{H}}_A \tag{3.46}$$

To derive the matrix $\mathbf{R}_{i,i+1}$ that transfers from station $i+1$ to station i, the moments must be summed about point i (to avoid introducing V_i into the equation). For this case, point A in equation 3.46 is replaced by point i, the left end of the rectangle in Figure 3.2. This choice of point A gives $\mathbf{r}_{g/A} = -h\mathbf{i}$ and $\mathbf{a}_A = \ddot{w}_i\mathbf{k}$ so that equation 3.46 can be written as

$$M_{i+1} - M_i - V_{i+1}\, l = mh\ddot{w}_i + \dot{\mathbf{H}}_A \tag{3.47}$$

For the planar motion case being considered,

$$\dot{\mathbf{H}}_A = I_i\, \alpha \tag{3.48}$$

where

$$I_i = I_{cg} + mh^2 \tag{3.49}$$

and

$$\alpha = -\omega^2\psi \tag{3.50}$$

Substituing these results into equation 3.47 gives

$$M_{i+1} - M_i - V_{i+1}\, l = mh\ddot{w}_i - \left(I_{cg} + mh^2\right)\omega^2\psi \tag{3.51}$$

In order to solve for M_i in terms of only quantities at station $i+1$, $\ddot{w}_i$ must be rewritten as

$$\ddot{w}_i = -\omega^2\left(w_{i+1} + \psi\, l\right) \tag{3.52}$$

Substituting this experssion into equation 3.51 and solving for M_i gives

$$M_i = \left(mlh + I_{cg} + mh^2\right)\omega^2\psi + mhw_{i+1}\,\omega^2 - V_{i+1}\, l + M_{i+1} \tag{3.53}$$

This equation gives the third row of the $\mathbf{R}_{i,i+1}$ matrix

$$\mathbf{R}_{i,i+1}(3,:) = \begin{bmatrix} -m\omega^2 h & \left(mlh + I_{cg} + mh^2\right)\omega^2 & 1 & -l \end{bmatrix} \tag{3.54}$$

Summing forces for the link gives

$$V_{i+1} - V_i = ma_{cg} \tag{3.55}$$

In order to solve for V_i in terms of only quantities at station $i+1$, a_{cg} must be rewritten as

$$a_{cg} = \ddot{w}_{i+1} + (l-h)\,\ddot{\psi} \tag{3.56}$$

or

$$a_{cg} = -\omega^2 w_{i+1} - (l-h)\,\omega^2\psi \tag{3.57}$$

Substituting equation 3.57 into equation 3.55 and solving for V_i gives

$$V_i = m\omega^2 w_{i+1} + m\,(l-h)\,\omega^2\psi + V_{i+1} \tag{3.58}$$

This equation gives the fourth row of the $\mathbf{R}_{i,i+1}$ matrix

$$\mathbf{R}_{i,i+1}(4,:) = \begin{bmatrix} -m\omega^2 & m\,(l-h)\,\omega^2 & 0 & 1 \end{bmatrix} \tag{3.59}$$

The entire $\mathbf{R}_{i,i+1}$ matrix can now be written as

$$\mathbf{R}_{i,i+1} = \begin{bmatrix} 1 & -l & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -mh\omega^2 & (mhl + I_{cg} + mh^2)\,\omega^2 & 1 & -l \\ -m\omega^2 & m\,(l-h)\,\omega^2 & 0 & 1 \end{bmatrix} \tag{3.60}$$

3.1.2 $\mathbf{R}_{i+1,i}$

To derive the matrix $\mathbf{R}_{i+1,i}$ that transfers from station i to station $i+1$ (i.e. $\mathbf{z}_1 = \mathbf{R}_{10}\mathbf{z}_0$), equation 3.41 can be used directly to get the first row of the matrix

$$\mathbf{R}_{i,i+1}(1,:) = \begin{bmatrix} 1 & l & 0 & 0 \end{bmatrix} \tag{3.61}$$

Because the link is rigid, the value of ψ does not change from one end to the other, so that the second row of the matrix is

$$\mathbf{R}_{i+1,1}(2,:) = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix} \tag{3.62}$$

If the matrix $\mathbf{R}_{i+1,i}$ (that transfers from station i to station $i+1$) is sought, the moments must be summed about point $i+1$ (to avoid introducing V_{i+1} into the equation). For this case, point A in equation 3.46 is replaced by point $i+1$, the right end of the rectangle in Figure 3.2. This choice of point A gives $\mathbf{r}_{g/A} = (l-h)\mathbf{i}$ and $\mathbf{a}_A = \ddot{w}_{i+1}\mathbf{k}$ so that equation 3.46 can be written as

$$M_{i+1} - M_i - V_i\,l = -m\,(l-h)\,\ddot{w}_{i+1} + \dot{\mathbf{H}}_A \tag{3.63}$$

For this case,

$$\dot{\mathbf{H}}_A = I_{i+1}\,\alpha \tag{3.64}$$

Iip1sub

$$I_{i+1} = I_{cg} + m\,(l-h)^2 \tag{3.65}$$

and (as in the previous case)

$$\alpha = -\omega^2\psi \tag{3.66}$$

Substituing these results into equation 3.63 gives

$$M_{i+1} - M_i - V_i\,l = -m\,(l-h)\,\ddot{w}_{i+1} - \left(I_{cg} + m\,(l-h)^2\right)\omega^2\psi \tag{3.67}$$

In order to solve for M_{i+1} in terms of only quantities at station i, $\ddot{w}_{i+1}$ must be rewritten as

$$\ddot{w}_{i+1} = -\omega^2\,(w_i - \psi\,l) \tag{3.68}$$

Substituting this experssion into equation 3.67 and solving for M_{i+1} gives

$$M_{i+1} = \left(-m\,(l-h)\,l - I_{cg} - m\,(l-h)^2\right)\omega^2\psi + m\,(l-h)\,w_i\,\omega^2 + V_i\,l + M_i \tag{3.69}$$

This equation gives the third row of the $\mathbf{R}_{i+1,i}$ matrix

$$\mathbf{R}_{i+1,i}(3,:) = \left[\begin{array}{cccc} -m\,(l-h)\,\omega^2 & \left(-m\,(l-h)\,l - I_{cg} - m\,(l-h)^2\right)\omega^2 & 1 & l \end{array}\right] \tag{3.70}$$

In order to solve for V_{i+1} in terms of only quantities at station i, a_{cg} must be rewritten as

$$a_{cg} = \ddot{w}_i - h\ddot{\psi} \tag{3.71}$$

or

$$a_{cg} = -\omega^2 w_i + \omega^2\psi\,h \tag{3.72}$$

Substituting equation 3.72 into equation 3.55 and solving for V_{i+1} gives

$$V_{i+1} = m\left(-\omega^2 w_i + \omega^2\psi\,h\right) + V_i \tag{3.73}$$

This equation gives the fourth row of the $\mathbf{R}_{i+1,i}$ matrix

$$\mathbf{R}_{i+1,i}(4,:) = \left[\begin{array}{cccc} m\omega^2 & mh\omega^2 & 0 & 1 \end{array}\right] \tag{3.74}$$

The entire $\mathbf{R}_{i+1,i}$ matrix can now be written as

$$\mathbf{R}_{i+1,i} = \left[\begin{array}{cccc} 1 & l & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -m\,(l-h)\,\omega^2 & \left(-m\,(l-h)\,l - I_{cg} - m\,(l-h)^2\right)\omega^2 & 1 & l \\ m\omega^2 & mh\omega^2 & 0 & 1 \end{array}\right] \tag{3.75}$$

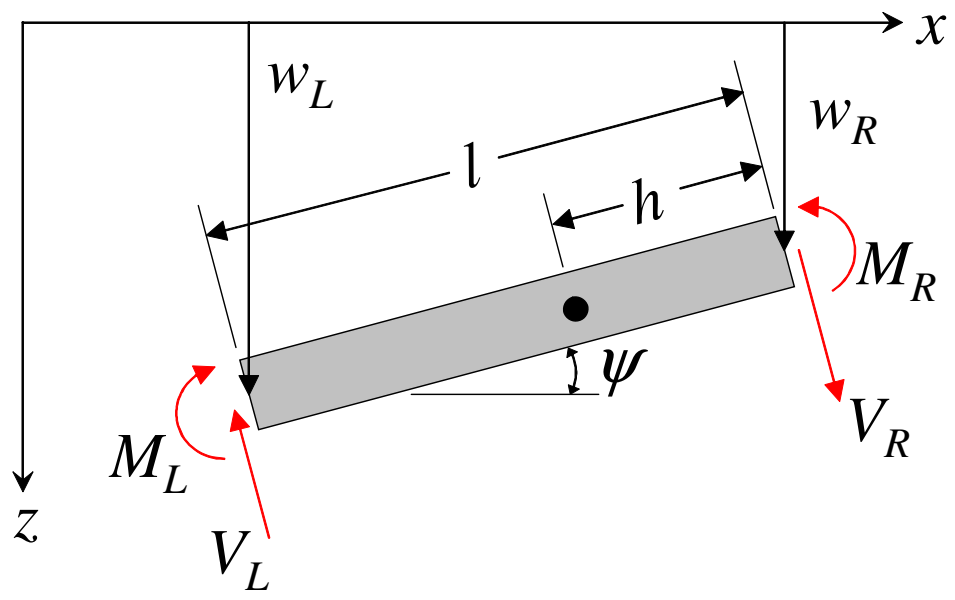

Figure 3.3: Sketch of the rigid body being analyzed in this section.

C.0.3 Matlab File Used

Matlab File Summary

This file uses the Matlab file `Rmatrix_deriv.m` This file contains my first derivation of the transfer matrix for a rigid mass. I had some questions about the rigid mass transfer matrix as it is given in Dr. Book's thesis.

I later re-did this derivation with a different convention for h (the distance to the center of mass). This later derivation was done to compare results with Dr. Book's lab notebook. The later derivation is done in `Rmatrix_deriv_DrBook_notes.m` which generates the file `Rmatrix_deriv_wjb_notes.tex`. It is this later derivation that is used in the remainder of my analysis and in the utility function `rigidmasstm.m`.

Verbatim Matlab Text

`Rmatrix_deriv_verb_apndx.pdf`

3.1 Rigid Mass Transfer Matrix (WJB notes)

3.1.1 $\mathbf{R}_{RL}$

This is an attempt to recreate the $\mathbf{R}_{RL}$ matrix (i.e. a transfer matrix for a rigid body that transfers from a state vector on the left-hand side to a state vector on the right-hand side: $\mathbf{z}_R = \mathbf{R}_{RL}\mathbf{z}_L$) that Dr. Book derived in his notebook (not exactly what is in his thesis).

If w is positive downward and ψ is positive in the counter-clockwise direction, then from Figure 3.3 it can be shown that

$$w_R = w_L - \psi\, l \tag{3.76}$$

The state vector is given by

$$\mathbf{z} = \begin{bmatrix} -w \\ \psi \\ M \\ V \end{bmatrix} \tag{3.77}$$

Because the state vector uses $-w$, equation 3.76 should be multiplied by -1:

$$-w_R = -w_L + \psi\, l \tag{3.78}$$

This equation gives the first row of the $\mathbf{R}_{RL}$ matrix

$$\mathbf{R}_{RL}(1,:) = \begin{bmatrix} 1 & l & 0 & 0 \end{bmatrix} \tag{3.79}$$

Because the link is rigid, the value of ψ does not change from one end to the other, so that the second row of the matrix is

$$\mathbf{R}_{RL}(2,:) = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix} \tag{3.80}$$

Summing forces gives

$$m a_{cg} = V_R - V_L \tag{3.81}$$

where

$$a_{cg} = \ddot{w}_L - \ddot{\psi}\ (l - h) \tag{3.82}$$

or

$$a_{cg} = -\omega^2 w_L + \omega^2 \psi\ (l - h) \tag{3.83}$$

Substituing equations 3.83 into equation 3.81 and solving for V_R gives

$$V_R = m\left(-\omega^2 w_L + \omega^2 \psi\ (l - h)\right) + V_L \tag{3.84}$$

This equation gives the fourth row of the $\mathbf{R}_{RL}$ matrix

$$\mathbf{R}_{RL}(4,:) = \begin{bmatrix} m\omega^2 & m\omega^2\,(l-h) & 0 & 1 \end{bmatrix} \tag{3.85}$$

Summing moments about the center of gravity gives

$$I\ddot{\psi} = M_R - M_L - V_L\ (l - h) - V_R\, h \tag{3.86}$$

Substituing for V_R from equation 3.84 and replacing $\ddot{\psi}$ with $-\omega^2\psi$ gives

$$-I\omega^2\psi = M_R - M_L - V_L\ (l - h) - \left(m\left(-\omega^2 w_L + \omega^2 \psi\ (l - h)\right) + V_L\right) h \tag{3.87}$$

Solving equation 3.87 for M_R gives

$$M_R = \left(\left(hl - h^2\right)\omega^2 m - I\omega^2\right)\psi - hm\omega^2 w_L + M_L + V_L\, l \tag{3.88}$$

This equation gives the third row of the $\mathbf{R}_{RL}$ matrix

$$\mathbf{R}_{RL}(3,:) = \begin{bmatrix} hm\omega^2 & (hl-h^2)\,\omega^2 m - I\omega^2 & 1 & l \end{bmatrix} \tag{3.89}$$

The entire $\mathbf{R}_{RL}$ matrix can now be written as

$$\mathbf{R}_{RL} = \begin{bmatrix} 1 & l & 0 & 0 \\ 0 & 1 & 0 & 0 \\ hm\omega^2 & (hl-h^2)\,\omega^2 m - I\omega^2 & 1 & l \\ m\omega^2 & m\omega^2\,(l-h) & 0 & 1 \end{bmatrix} \tag{3.90}$$

The $\mathbf{R}_{RL}$ matrix is used to calculate $\mathbf{z}_R = \mathbf{R}_{RL}\mathbf{z}_L$ or

$$\begin{bmatrix} -w \\ \psi \\ M \\ V \end{bmatrix}_R = \begin{bmatrix} 1 & l & 0 & 0 \\ 0 & 1 & 0 & 0 \\ hm\omega^2 & (hl-h^2)\,\omega^2 m - I\omega^2 & 1 & l \\ m\omega^2 & m\omega^2\,(l-h) & 0 & 1 \end{bmatrix} \begin{bmatrix} -w \\ \psi \\ M \\ V \end{bmatrix}_L \tag{3.91}$$

Alternatively, the $\mathbf{R}_{LR}$ matrix is given by $[\mathbf{R}_{RL}]^{-1}$

$$\mathbf{R}_{LR} = \begin{bmatrix} 1 & -l & 0 & 0 \\ 0 & 1 & 0 & 0 \\ m\omega^2\,(l-h) & (-hl+h^2)\,\omega^2 m + I\omega^2 & 1 & -l \\ -m\omega^2 & hm\omega^2 & 0 & 1 \end{bmatrix} \tag{3.92}$$

and the $\mathbf{R}_{LR}$ matrix can be used to calculate $\mathbf{z}_L = \mathbf{R}_{LR}\mathbf{z}_R$ or

$$\begin{bmatrix} -w \\ \psi \\ M \\ V \end{bmatrix}_L = \begin{bmatrix} 1 & -l & 0 & 0 \\ 0 & 1 & 0 & 0 \\ m\omega^2\,(l-h) & (-hl+h^2)\,\omega^2 m + I\omega^2 & 1 & -l \\ -m\omega^2 & hm\omega^2 & 0 & 1 \end{bmatrix} \begin{bmatrix} -w \\ \psi \\ M \\ V \end{bmatrix}_R \tag{3.93}$$

Testing the output of my `rigidmasstm.m` function,

$$\mathbf{R}_{RL} = \begin{bmatrix} 1 & l & 0 & 0 \\ 0 & 1 & 0 & 0 \\ hm\omega^2 & (hl-h^2)\,\omega^2 m - I\omega^2 & 1 & l \\ m\omega^2 & m\omega^2\,(l-h) & 0 & 1 \end{bmatrix} \tag{3.94}$$

Testing the augmented output of my `rigidmasstm.m` function (the augmented form is used to find the forced system response),

$$\mathbf{R}_{RLA} = \begin{bmatrix} 1 & l & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ hm\omega^2 & (hl - h^2)\,\omega^2 m - I\omega^2 & 1 & l & 0 \\ m\omega^2 & m\omega^2\,(l - h) & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.95}$$

D.0.2 Matlab File Used

Matlab File Summary

This file uses the Matlab file `Rmatrix_deriv_DrBook_notes.m` This function documents the derivation of the rigid mass transfer matrix that I use in my analysis. I had some questions concerning the rigid mass transfer matrix that Dr. Book gives in his thesis and in a chapter of a book he is writing on flexible robotics. I believe that the derivation included in this file is correct and it is the one I am moving forward with. It agrees fairly closely, but not completely, with the derivation in Dr. Book's lab manual. There are some significant differences between this derivation and the results in Dr. Book's thesis.

This derivation is the basis of the transfer matrix utility function `rigidmasstm.m` which is in the `rwk_matlab_functions` directory.

Verbatim Matlab Text

`Rmatrix_deriv_wjb_notes_verb_apndx.pdf`

3.1 Free Cantilever Analysis

3.1.1 Natural Frequency Comparison

Results from the tranfer matrix method:

$$f_1 = 6.302 \text{ Hz} \tag{3.96}$$

$$f_2 = 39.494 \text{ Hz} \tag{3.97}$$

Comparison with exact solution: The exact solution comes from solving

$$\cos\beta \cosh\beta = -1 \tag{3.98}$$

Using the natural frequencies from the transfer matrix to evaluate this expression gave

$$\cos\beta \cosh\beta = -1.0000000000 \tag{3.99}$$

$$\cos\beta \cosh\beta = -1.0000000000 \tag{3.100}$$

3.1.2 Mode Shape Comparison

Figure 3.4 overlays the first mode shape from the transfer matrix with the exact solution. Figure 3.5 overlays the second mode shape from the transfer matrix with the exact solution. The agreement for both mode shapes is good. This gives a high degree of confidence that the transfer matrix method for a cantilever beam has been correctly implemented in Matlab.

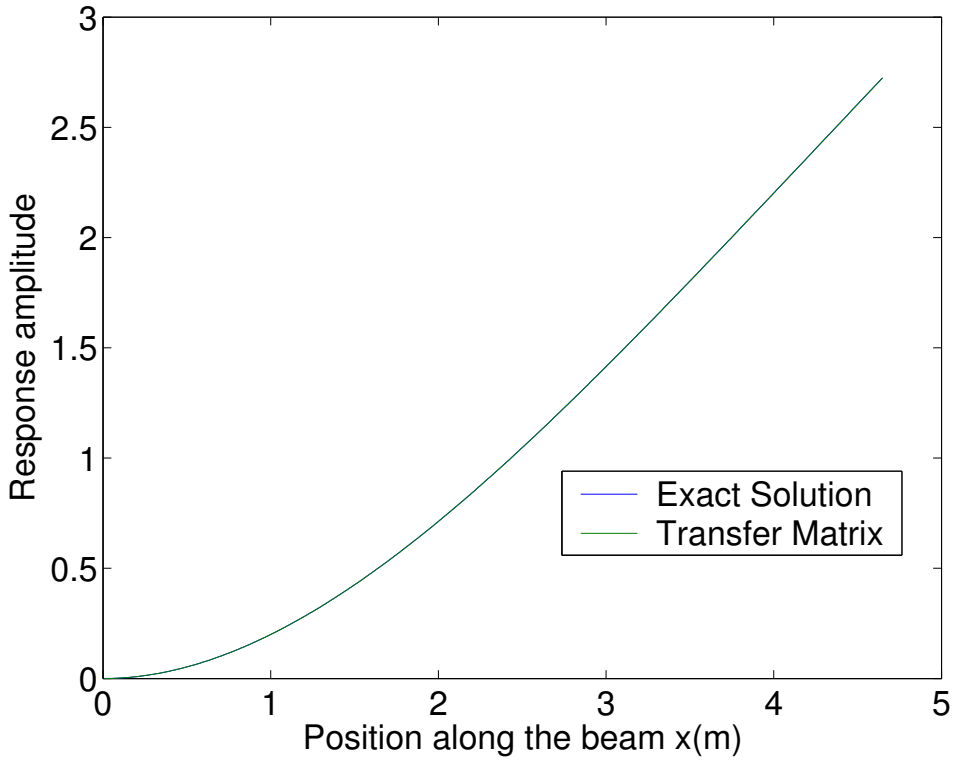

Figure 3.4: Comparison of the first mode shape from the transfer matrix method and the exact solution for a free cantilever.

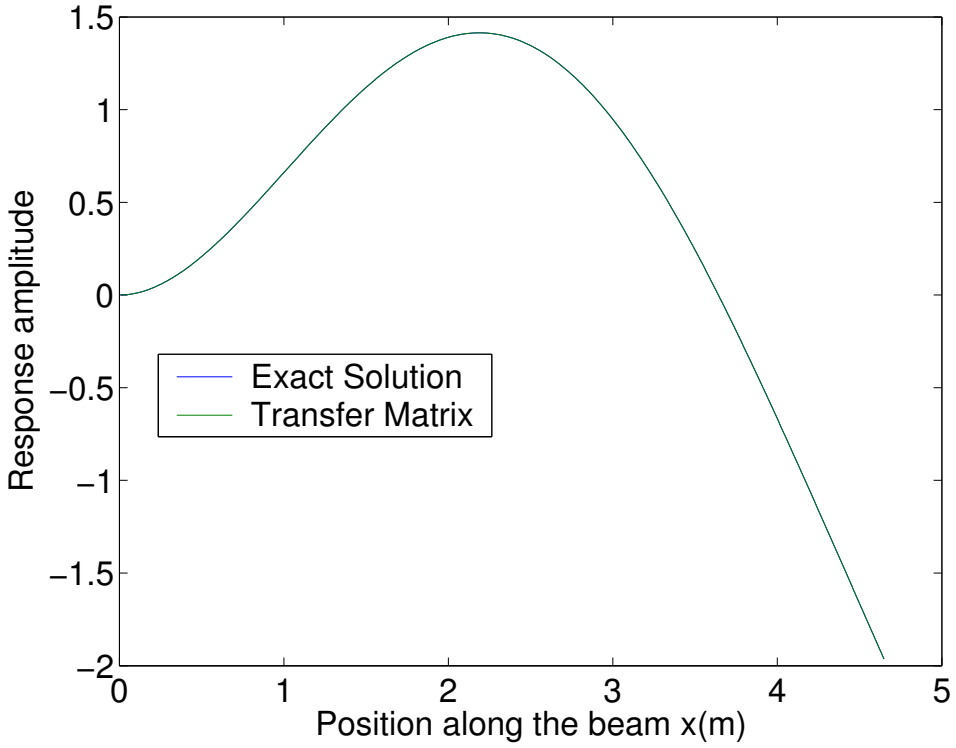

Figure 3.5: Comparison of the second mode shape from the transfer matrix method and the exact solution for a free cantilever.

E.0.3 Matlab File Used

Matlab File Summary

This file uses the Matlab file `freecantilever_wnums.m` To verify that my transfer matrix code is working correctly, I compared results between the transfer matrix method and other methods as I went along. The first step was to compare the results for a single cantilever beam element with the exact solution (from Meirovitch). That is what this file does. Natural frequencies and mode shapes are compared between the transfer matrix method and the exact solution (since the transfer matrix is and exact solution, this is just a verifcation of the code and that I am correctly implementing the method).

Verbatim Matlab Text

`freecantilever_wnums_verb_apndx.pdf`

3.1 Cantilever with Point End Mass Analysis

3.1.1 Natural Frequency Comparison

Results from the tranfer matrix method:

$$f_1 = 2.5754 \text{ Hz} \tag{3.101}$$

$$f_2 = 28.888 \text{ Hz} \tag{3.102}$$

Results from the assumed modes method:

$$f_1 = 2.5754 \text{ Hz} \tag{3.103}$$

$$f_2 = 28.888 \text{ Hz} \tag{3.104}$$

3.1.2 Mode Shape Comparison

Figure 3.6 overlays the first mode shape from the transfer matrix method and the assumed modes method. Figure 3.7 overlays the second mode shape from the transfer matrix method and the assumed modes method. The agreement for both mode shapes is good. This gives a high degree of confidence that the transfer matrix method for a cantilever beam has been correctly implemented in Matlab.

3.2 Cantilever with Real End Mass Analysis

3.2.1 Natural Frequency Comparison

Results from the tranfer matrix method:

$$f_1 = 2.2386 \text{ Hz} \tag{3.105}$$

$$f_2 = 19.346 \text{ Hz} \tag{3.106}$$

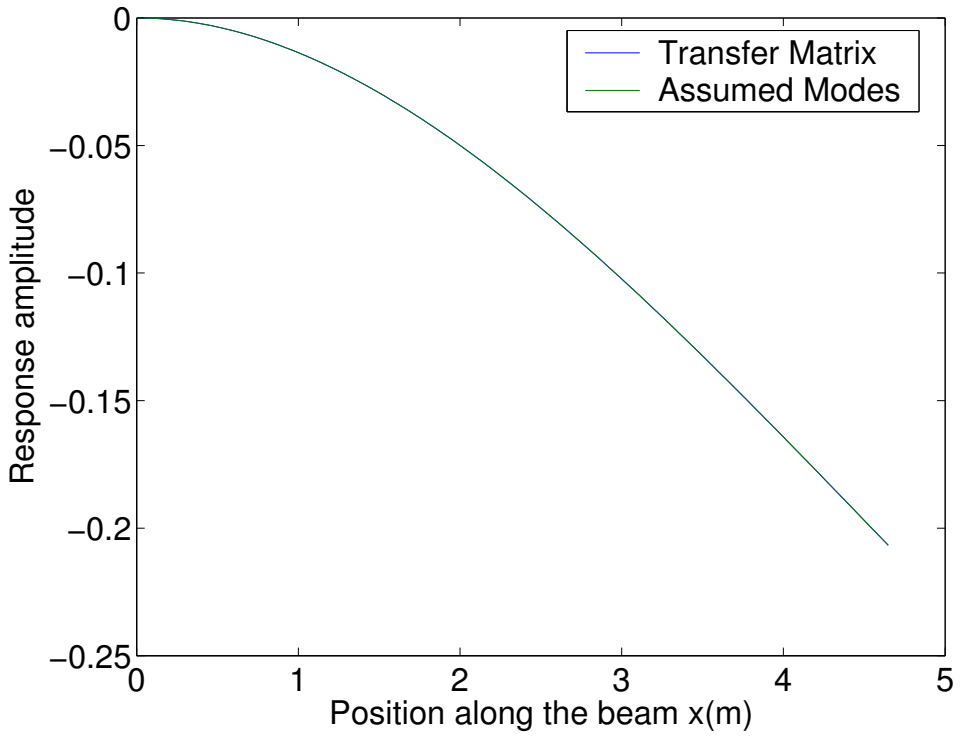

Figure 3.6: Comparison of the first mode shape from the transfer matrix method and the assumed modes method for a cantilever with a point mass on the end.

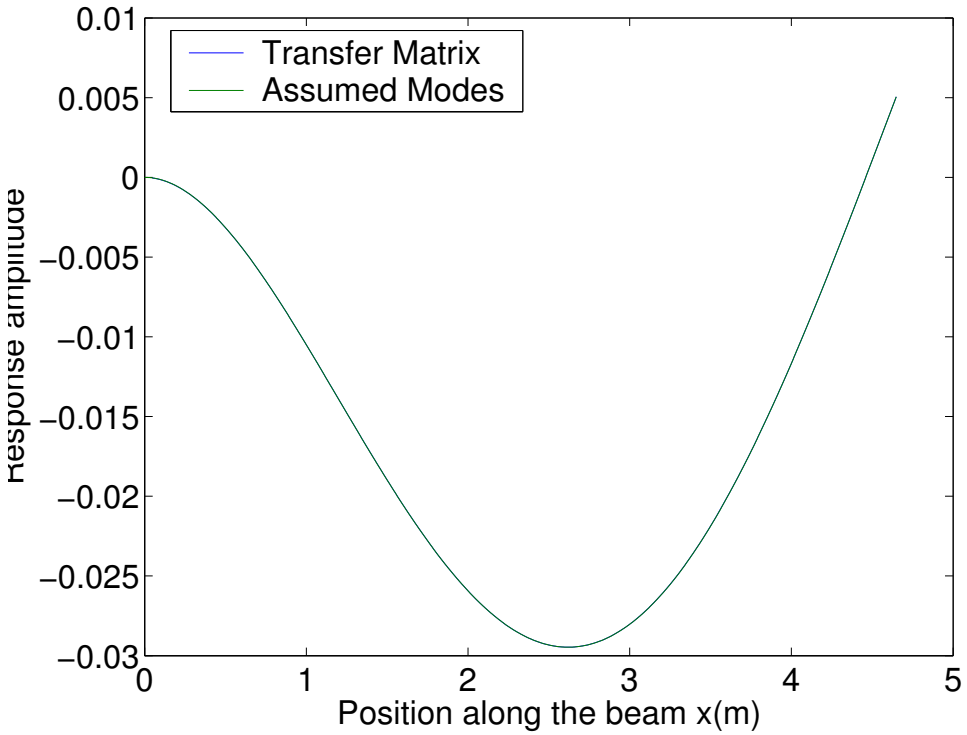

Figure 3.7: Comparison of the second mode shape from the transfer matrix method and the assumed modes method for a cantilever with a point mass on the end.

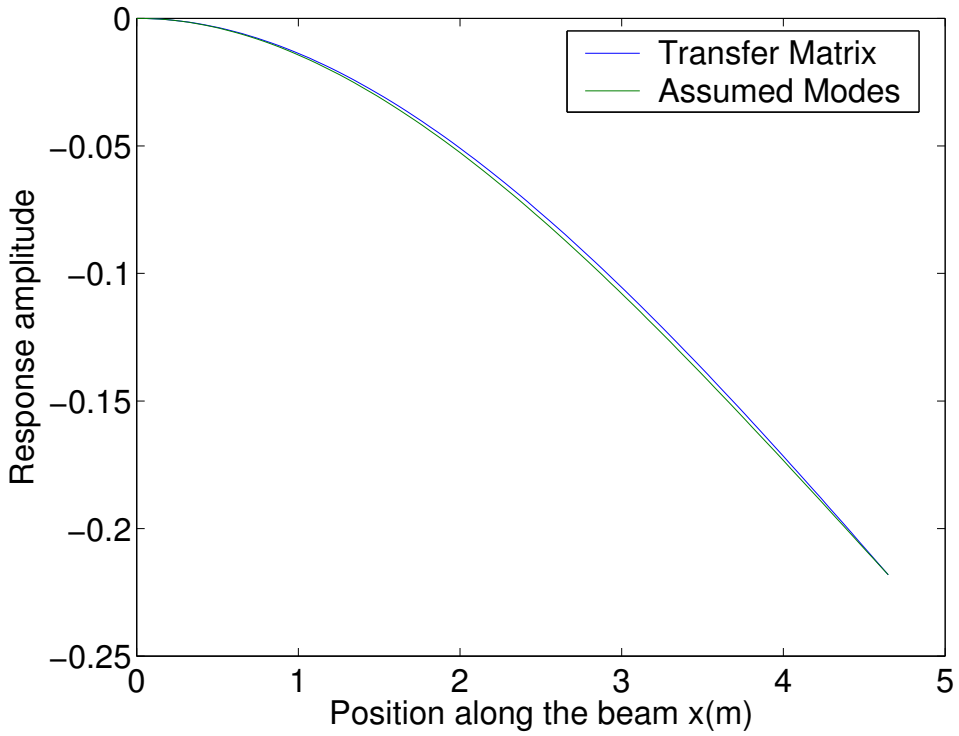

Figure 3.8: Comparison of the first mode shape from the transfer matrix method and the assumed modes method for a cantilever with a real mass on the end.

Results from the assumed modes method:

$$f_1 = 2.4485 \text{ Hz} \tag{3.107}$$

$$f_2 = 28.918 \text{ Hz} \tag{3.108}$$

3.2.2 Mode Shape Comparison

Figure 3.8 overlays the first mode shape from the transfer matrix method and the assumed modes method. Figure 3.9 overlays the second mode shape from the transfer matrix method and the assumed modes method. Note that the rotational kinetic energ of the end mass was not included in the assumed modes method analysis.

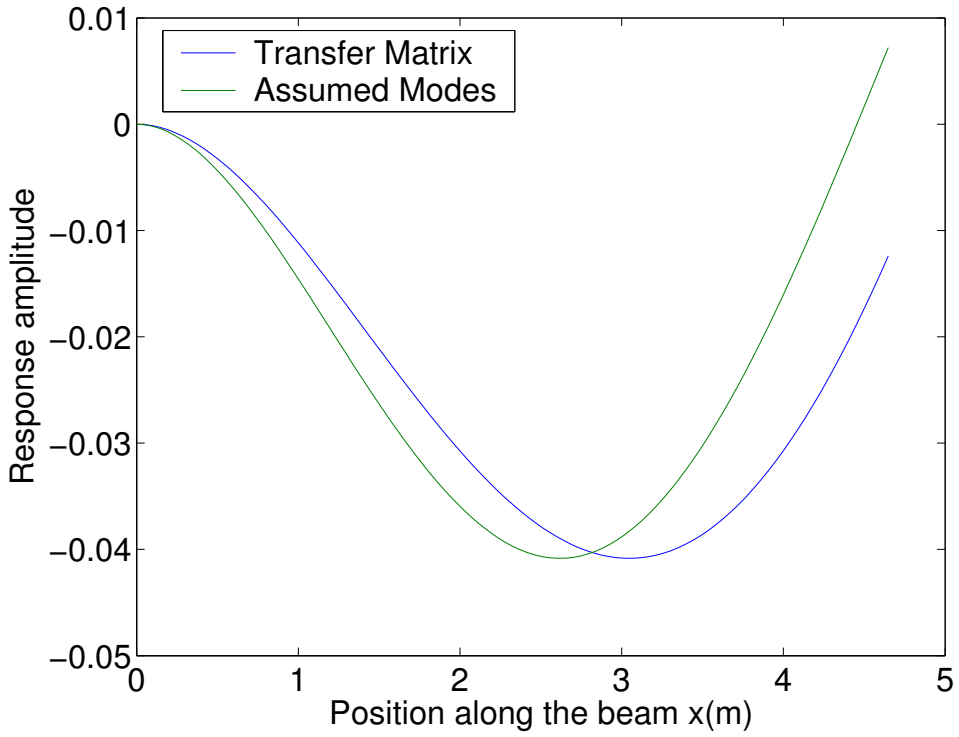

Figure 3.9: Comparison of the second mode shape from the transfer matrix method and the assumed modes method for a cantilever with a point mass on the end.

F.0.3 Matlab File Used

Matlab File Summary

This file uses the Matlab file `cantileverwrealmass_wnums.m` As a next step in complexity, this file uses the transfer matrix method to find the natural frequencies and mode shapes of a cantilever beam with a real mass on the end (i.e. one with non-zero length and second moment of ineteria). The results from the transfer matrix method are compare to results from the assumed modes method.

Verbatim Matlab Text

`cantirealmass_wnums_verb_apndx.pdf`

Chapter 4

Transfer Matrix Utilities

A fair ammount of time this semester was spent developing Matlab code for straightforward implemenation of the transfer matrix method for arbitrary systems. I hope these functions will be able to be used for some time into the future and built upon by others. I tried to make them able to handle as general a tranfer matrix problem as possible.

G.1 Matlab Files Used

G.1.1 Main Matlab File

The main Matlab file is `tmsystem.m` This function attempts to find the transfer matrix for an arbitrary system. This function is intended to be called by fminsearch to find the natural frequencies. To accomplish this, the first input is omega and the additional inputs are for discribing the system. The second input is a cell array of strings containing the transfer matrices in the order they are to be multiplied together (i.e. {'B','R','B','B','R'}). The third input is a cell array of the parameters corresponding to each transfer matrix.

G.1.2 Additional Matlab Files

`beammodeshape.m`

This function determines the mode shape of a beam element by calling the `beamtm.m` function for varying lengths from x=xstep to L. Apparently, I forgot that I wrote this and wrote the new funciton `tmbeammodeshape.m` to do basically the same thing. `tmbeammodeshape.m` is very similar but is newer and should be used in stead of this one.

`beamtm.m`

This function returns the transfer matrix for a beam element. The inputs are the frequency, a cell array of system parameters={L, E, I, mu}, and a cell array containing optoinal arguments. Right now {'aug=1'} is the only supported optional argument, which when evaluted would specify that an augmented matrix should be returned for use in determining the force response of dynamic systems.

`forcetm.m`

This function returns an augmented transfer matrix used as input point for external forcing on a dynamic system. The output is an identity matrix augmented with a column of the forced values and a fifth row that is $\begin{bmatrix} 0 & 0 & 0 & 0 & 1 \end{bmatrix}$.

`rigidmasstm.m`

This function returns the transfer matrix for a beam element. The inputs are the frequency, a cell array of system parameters={Lr, mr, Ir, h} (where h is the distance from the RIGHT or far side of mass to the center of gravity), and a cell array containing optoinal arguments. Right now {'aug=1'} is the only supported optional argument, which when evaluted would specify that an augmented matrix should be returned for use in determining the force response of dynamic systems.

For a sketch of the rigid mass elements with the various parameters on it or to see a derivaiton of this matrix, please refer to `Rmatrix_deriv_DrBook_notes.m` and `Rmatrix_deriv_wjb_notes.tex`

This matrix transfers from the left to the righthand side of a rigid mass, z_L to z_R according to $z_R = \mathbf{R} z_L$ as in the derivation in `Rmatrix_deriv_DrBook_notes.m` which outputs the file `Rmatrix_deriv_wjb_notes.tex`

`tmbeammodeshape.m`

This function calls the beamtm function with length varying from L/numx to the total length of the beam element at numx increments. Based on zL, the boundary conditions at the left end of the beam, and the beam transfer matrix calculated at the current length increment, the response at numx points along the length of the beam is found.

beamparams needs to be in the same order as the sysparams argument of the beamtm.m function (L=sysparams{1}; E=sysparams{2}; I=sysparams{3}; mu=sysparams{4};).

G.1.3 Verbatim Matlab Files

`tmutilies_verb_apndx.pdf`