THE FLOATING DUTCHMEN
Three Dimensional Driven-Arm Inverted Pendulum
Final Report for ECSE-4962 Control Systems Design

Team 2
Teresa Bernardi
Brian Lewis
Matthew Rosmarin

Monday, May 8, 2006

Rensselaer Polytechnic Institute
Abstract

The purpose of this paper is to give a synopsis of the senior design project for Control Systems Design. The goal of the senior design project is to explore control techniques and implement a controller to balance an inverted pendulum initially in two dimensions and finally an inverted pendulum in three dimensions. The challenge to successfully balance an unstable system is the motivation behind the project. This report contains the design strategy and the final results. The system was successfully balancing in two dimensions in all configurations, but balancing in three dimensions was unattainable due to the motor system design and excessive vibrations. With modifications to make the motor system design more stable, balancing in three dimensions should be possible.
# Table of Contents:

1. Introduction..................................................................................................................... 1
2. System Design .................................................................................................................. 2
  2.1 Design Specifications.................................................................................................. 2
  2.2 Modeling using Matlab SimMechanics ..................................................................... 2
    2.2.1 Two Dimensional Model ................................................................................... 2
    2.2.2 Three Dimensional Model ............................................................................... 3
  2.3 Equations of Motion .................................................................................................. 5
    2.3.1 Linearized Equations of Motion ........................................................................ 5
  2.4 Model Validation ....................................................................................................... 7
    2.4.1 Equations of Motion .......................................................................................... 7
    2.4.2 Natural Frequency of Oscillation ..................................................................... 8
    2.4.3 SolidWorks ...................................................................................................... 11
  2.5 Control Design .......................................................................................................... 14
    2.5.1 Matlab Feedback Control Loop ....................................................................... 14
    2.5.2 Batch Testing ................................................................................................... 15
    2.5.3 Labview Implementation ................................................................................. 18
  2.6 Controller Tuning .................................................................................................... 20
  2.7 Design Integration ................................................................................................... 22
    2.7.1 Mechanical Design of Three Dimensional System .......................................... 22
    2.7.2 Mechanical Design of Two Dimensional System ............................................ 23
    2.7.3 System Wiring ................................................................................................. 23
3.0 Results.......................................................................................................................... 24
  3.1 Two Dimensional Lower Configuration ................................................................... 24
  3.2 Two Dimensional Upper Configuration ..................................................................... 28
  3.3 Three Dimensional Upper Configuration ........................................................................ 30
  3.4 Two Dimensional Swing up .................................................................................... 32
4.0 Costs............................................................................................................................ 34
5.0 Conclusion .................................................................................................................. 35
6.0 Bibliography................................................................................................................. 36
Appendix A: Terms............................................................................................................ 37
Appendix B: Matlab for 2D Lower Configuration ............................................................. 38
Appendix C: Matlab for 2D Upper Configuration ................................................................. 41
Appendix D: Equations of Motion ...................................................................................... 44
Appendix E: Matlab Code for Equations of Motion............................................................ 54
Appendix F: Matlab Model and Code for Frequency Calculation of Actuating Arm ......... 56
Appendix G: Matlab Model and Code for Frequency Calculation of Balancing Arm ...... 58
Appendix H: Experimental Results for Parameter Identification Model Validation .......... 60
Appendix I: Inertia Calculations........................................................................................ 63
Appendix J: Matlab 2D Batch File Configuration ................................................................. 65
Appendix K: Matlab 3D Batch File Configuration ................................................................. 71
Appendix L: Batch File Results......................................................................................... 78
  2D Configuration .......................................................................................................... 78
  3D Configuration .......................................................................................................... 84
Appendix M: Labview VI Summary .................................................................................. 91
Appendix N: SolidWorks Drawings ................................................................................... 98
List of Figures

Figure 1: 2D Linearized Matlab SimMechanics Model .......................................................... 3
Figure 2: 3D Linearized Matlab SimMechanics Model .......................................................... 4
Figure 3: Period of Oscillating Actuating Arm ...................................................................... 9
Figure 4: Period of Oscillating Balancing Arm .................................................................. 10
Figure 5: Period of Oscillating .......................................................................................... 10
Figure 6: Mass Properties for Actuating Arm Assembly ..................................................... 12
Figure 7: Mass Properties for Balancing Arm Assembly ..................................................... 13
Figure 8: 2D Simulink Control Loop ................................................................................... 14
Figure 9: 2D Nonlinear Simulink Control Loop ................................................................. 15
Figure 10: 2D Nonlinear Simulink Control Loop for Simulation ........................................ 15
Figure 11: Batch Results for Actuating Arm Angle ........................................................... 17
Figure 12: Batch File Result for Balancing Arm Angle ..................................................... 17
Figure 13: State Space Control Gains ................................................................................ 20
Figure 14: System Gain ...................................................................................................... 21
Figure 15: Static Friction Compensation ............................................................................ 21
Figure 16: 2D System, Lower Configuration ....................................................................... 23
Figure 17: Balancing Result with no end Weight, Arm Angles .......................................... 24
Figure 18: Balancing result with no end Weight, Arm Velocities ....................................... 25
Figure 19: Balancing with .13kg end Mass, Arm Angles .................................................... 25
Figure 20: Balancing with .13 kg end Mass, Arm Velocities ............................................. 26
Figure 21: Balancing with .23kg end Mass, Arm Angles .................................................... 26
Figure 22: Lower Balancing with .23 kg end Mass, Arm Velocities .................................. 27
Figure 23: Lower Balancing with .68kg end Mass, Arm Velocities .................................... 27
Figure 24: Lower Balancing with .68 kg end Mass, Arm Velocities ................................... 28
Figure 25: 2D Upper Configuration .................................................................................. 28
Figure 26: Upper Balancing with .13 kg end Mass, Arm Angles ....................................... 29
Figure 27: Upper Balancing Configuration .13 end Mass, Arm Angles ............................... 29
Figure 28: SolidWorks: Three Dimension System ............................................................... 30
Figure 29: 3D Marginal Balancing with .13 kg end Mass, Y Angle ..................................... 31
Figure 30: 3D Marginal Balancing with .13 kg end Mass, X Angle ..................................... 31
Figure 31: 3D Marginal Balancing with .13 kg end Mass, Balancing Arm Angles .............. 32
Figure 32: Swing Up with no end Mass ............................................................................. 33
Figure 33: Swing up with .13kg end Mass .......................................................................... 33
Figure 34: Device Parts ..................................................................................................... 37
Figure 35: 2D Lower Configuration Matlab Plant .............................................................. 38
Figure 36: 2D Lower Configuration Matlab Simulink ......................................................... 38
Figure 37: Actuating Arm Position Information for Lower Configuration ......................... 39
Figure 38: Balancing Arm Position Information for Lower Configuration ......................... 39
Figure 39: 2D Upper Configuration Matlab Plant ............................................................... 41
Figure 40: 2D Upper Configuration Matlab Simulink .......................................................... 41
Figure 41: Actuating Arm Position Information for Upper Configuration .......................... 42
Figure 42: Balancing Arm Position Information for Upper Configuration ...................... 42
Figure 43: SimMechanics Model for Actuating Arm Frequency Calculation .................. 56
Figure 44: SimMechanics Model for Balancing Arm Frequency Calculation ................. 58
Figure 45: 2D Batch File Linearized Plant ................................................................. 65
Figure 46: 2D Batch File Simulink Diagram ............................................................... 65
Figure 47: 3D Batch File Plant ................................................................................. 71
Figure 48: 3D Batch File Diagram ........................................................................... 72
Figure 49: Batch File Results for Actuating Arm for End Weight of 0.0001 kg .............. 79
Figure 50: Batch File Results for Balancing Arm for End Weight of 0.0001 kg ............ 79
Figure 51: Batch File Results for Actuating Arm for End Weight of 0.1 kg............... 80
Figure 52: Batch File Results for Balancing Arm for End Weight of 0.1 kg ............... 80
Figure 53: Batch File Results for Actuating Arm for Balancing Arm of 0.2 m .......... 81
Figure 54: Batch File Results for Balancing Arm for Balancing Arm of 0.2 m .......... 81
Figure 55: Batch File Result for Actuating Arm for Balancing Arm of 2 m .............. 82
Figure 56: Batch File Result for Actuating Arm for Balancing Arm of 2 m .............. 82
Figure 57: Batch File Results for Actuating Arm for Actuating Arm of 0.22 m ....... 83
Figure 58: Batch File Results for Balancing Arm for Actuating Arm of 0.22 m ....... 83
Figure 59: Batch File Results for Actuating Arm in x-Axis for T = 10 N•m ............... 84
Figure 60: Batch File Results for Actuating Arm in y-Axis for T = 10 N•m ............... 85
Figure 61: Batch File Results for Balancing Arm in x-Axis for T = 10 N•m ............... 85
Figure 62: Batch File Results for Balancing Arm in y-Axis for T = 10 N•m ............... 86
Figure 63: Batch File Results for Actuating Arm in x-Axis for T = 0.083 N•m .......... 87
Figure 64: Batch File Results for Actuating Arm in y-Axis for T = 0.083 N•m .......... 87
Figure 65: Batch File Results for Balancing Arm in x-Axis for T = 0.083 N•m .......... 88
Figure 66: Batch File Results for Balancing Arm in y-Axis for T = 0.083 N•m .......... 88
Figure 67: Batch File Results for Actuating Arm in x-Axis for Constant Balancing Arm ................................................................. 89
Figure 68: Batch File Results for Actuating Arm in y-Axis for Constant Balancing Arm ................................................................. 89
Figure 69: Batch File Results for Balancing Arm in x-Axis for Constant Balancing Arm ................................................................. 90
Figure 70: Batch File Results for Balancing Arm in y-Axis for Constant Balancing Arm ................................................................. 90
Figure 71: LabView Variables .................................................................................. 91
Figure 72: Encoder Interface FPGA enhanced vel estimation.vi: ................................ 92
Figure 73: Current Amp Interface FPGA.vi .............................................................. 93
Figure 74: Joystick Interface FPGA.vi ..................................................................... 93
Figure 75: 2D Balancing VI .................................................................................... 95
Figure 76: Swing Up Vi .......................................................................................... 96
Figure 77: 3D VI ..................................................................................................... 97
List of Tables

Table 1: Natural Frequencies of System ................................................................. 11
Table 2: Experimental Frequency Data for Actuating Arm .............................. 60
Table 3: Experimental Frequency Data for Balancing Arm ............................ 60
Table 4: Summary of Arm Characteristics ............................................................ 61
Table 5: Summary of Experimental Results ......................................................... 62
1. Introduction

The inverted pendulum is a popular control application which is vastly used and studied throughout the scientific community. The motivation for this project is the significance behind the inverted pendulum. The significance is the system is inherently unstable. Balancing an unstable system plays an important role in understanding the dynamics of the human body. Many robotic applications such as the biped walking robot use the fundamentals of unstable systems such as inverted pendulums [1]. There are many different forms of inverted pendulums. One form is moving a cart back and forth to balance an inverted pendulum. Another example is a rotary inverted pendulum [2]. In a rotary configuration, the first arm which is driven by a motor rotates in a vertical plane balances the pendulum. A two dimensional arm-driven inverted pendulum is the most similar design to the proposed project.

The primary objective of the project is to design and balance a Three Dimensional Arm-Driven Inverted Pendulum. Since the size and dimensions of this system greatly influence how easy or difficult it will be to control, different mechanical configurations were created. These configurations consisted of different weights that affect the inertia and center of mass of the balancing arm. Part of the project is to identify what mechanical characteristics cause the system to be easy or difficult to control, using Matlab’s SimMechanics to simulate the system. See Appendix A for nomenclature.

To successfully complete the primary objective, the system is designed, built and modeled in two dimensions: in a lower configuration and an upper configuration. The purpose of this is to show the system can be balanced in two dimensions in both the pan and tilt axes separately. Other than the general specification of the system balancing, the inverted pendulum must also be able to compensate for disturbances in the form of a non-zero initial angle, perturbations to the system, and variation in end mass. The two areas of analyzing system performance will be first in simulation and then in final implementation. When the design approach is done in two dimensions, the final goal is to combine the results from the two systems to create a three dimensional inverted pendulum. Swing up was also a design implemented in the system. This was done successfully in the lower configuration.

The design approach has four main components: the design specifications, the model validations, control design and tuning, integration and implantation. Model validation consists of Matlab system modeling and verification by equations of motion, parameter identification using natural frequency and moment of inertia. The control design consists of linearization of the model, state space control design, and nonlinear simulation and evaluation. The mechanical fabrication of the system, the electrical aspects and the controller are completed in the integration and implementation stage. The results consist of the analyzed data from balancing the system in multiple configurations.
2. System Design

2.1 Design Specifications

The overall goal of the project is to balance a three dimensional inverted pendulum. In order to meet this goal, the system needs to be successfully balanced in two dimensions using both the tilt and pan axes separately. Once a controller is implemented for both axes in the upper configuration, the systems can be combined to balance the three dimensional system. Other than the general specification of the system balancing, the Inverted Pendulum must also be able to compensate for disturbances in the form of a non-zero initial angle, perturbations to the system, and variation in the arm inertia. The two areas of analyzing system performance will be first in simulation and then in final implementation.

2.2 Modeling using Matlab SimMechanics

In order to accurately model the different configurations of the inverted pendulum system, a Matlab Simulink add-on called SimMechanics is used. SimMechanics applies the Newtonian laws of physics to rigid body machines and their motion. This capability allows for easier design, simulation and virtual testing of mechanical systems, including nonlinear aspects of the system. SimMechanics also provides the capability for three dimensional visualization of a created model. Since this modeling method is purely of the mechanical system, creating a model of the three dimensional system will be no more difficult than for the two dimensional, which does not hold true for other modeling methods.

2.2.1 Two Dimensional Model

The two dimensional (2D) model represents the system when either motion is allowed in the xz-plane or in the yz-plane, but not for both simultaneously. There are two different configurations of the 2D model. The first is the lower configuration, where the center of gravity of the actuating arm is below the axis of rotation. The second is the upper configuration where center of gravity of both the actuating and balancing arms are above the axis of rotation. Considering the angles to be measured from the vertical, both configurations will have the same initial conditions for both the actuating and balancing arms - zero degrees. The major difference between the two configurations is the defining of the locations of the center of gravities and the direction that the arms point in (see Appendices B and C for 2D Upper and Lower Configuration Models). Since the only major difference is in how the arms are specified the physical design of the models are
identical, and only the properties of the system blocks need to be defined. The layout of the model for the upper, and thus the lower, configuration may be seen in Figure 1 below.

Figure 1: 2D Linearized Matlab SimMechanics Model

The model begins at the base of the figure, where the ground corresponds to the location where the motor is mounted to a structure in the physical system. The lower joint is the motor shaft which accepts initial conditions and torque commands. The position and velocity that the lower joint outputs are equivalent to the readings gleaned from the shaft encoder. The lower joint is firmly connected to the driven arm, which is analogous to the actuating arm. Then there is an upper joint that is equivalent to the universal joint in the physical system. There are inputs available to simulate a joint spring and damper, as well as take in any initial conditions for the joint. The position and velocity of the joint motion are read, much as with the encoders on the physical system. Above the upper joint is the pendulum, which corresponds to the balancing arm and end mass combination.

2.2.2 Three Dimensional Model

The three dimensional SimMechanics model simply requires the upper joint to be changed to a universal joint, and the addition of a second joint to the lower joint, as can be seen in Figure 2. There are also actuator inertias added to the 3D model, however these are assumed to be negligible and therefore the addition does not change the model.
significantly. The encoder equivalents must now collect information from the position and velocity in both the x- and y-axes. The weld between the actuator X inertia and the driven arm is there to ensure that the actuating arm is firmly affixed to the motor shaft.

Figure 2: 3D Linearized Matlab SimMechanics Model
2.3 Equations of Motion

The equations of motion for 2D balancing were previously derived in a thesis by Joshua Hurst (Hurst 2003) [3]. The final nonlinear equations of motion are

\[
\begin{align*}
(m_2 l_2 l_2 \cos \theta_2 + m_2 l_2^2 + l_2) \ddot{\theta}_2 + (m_2 l_1^2 + 2m_2 l_1 l_2 \cos \theta_2 + I_{z_1} + m_2 l_2^2 + I_{z_2} + m_1 l_1^2) \ddot{\theta}_1 - m_2 g L_2 \sin(\theta_1 + \theta_2) - m_2 l_1 l_2 \ddot{\theta}_1 - m_2 g l_1 \sin \theta_1 - 2m_2 l_1 l_2 \ddot{\theta}_1 \dot{\theta}_2 \sin \theta_2 = 0 \\
m_1 g L_{11} \sin \theta_1 = T - B_1 \dot{\theta}_1 - T_{f1} \text{sgn} \theta_1 \\
(2m_2 l_2 \dot{\theta}_2 + (m_2 l_1 l_2 \cos \theta_2 + m_2 l_2^2 + I_{z_2}) \ddot{\theta}_1 + m_2 l_1 l_2 \ddot{\theta}_1 \sin \theta_2 - m_2 g l_2 \sin(\theta_1 + \theta_2) = -B_2 \dot{\theta}_2 - T_{f2} \text{sgn} \theta_2
\end{align*}
\]

where \( m_1 \) is the mass of the actuating arm, \( m_2 \) is the mass of the balancing arm, \( L_1 \) is the length of the actuating arm, \( L_{11} \) is the distance from the motor axis to the center of gravity of the actuating arm, \( L_{21} \) is the distance from the axis of the universal joint to the center of gravity of the balancing arm, \( \theta_1 \) is the angle between the vertical and the actuating arm (positive in the counterclockwise direction), \( \theta_2 \) is the angle between the actuating arm and the balancing arm (positive in the counterclockwise direction), \( \dot{\theta}_1 \) is the angular velocity of the actuating arm, \( \dot{\theta}_2 \) is the angular velocity of the balancing arm, \( \ddot{\theta}_1 \) is the angular acceleration of the actuating arm, \( \ddot{\theta}_2 \) is the angular acceleration of the balancing arm, \( I_{1xx} \) is the inertia of the actuating arm about the x-axis with no coupling with any other axis, \( I_{2xx} \) is the inertia of balancing arm about the x-axis with no coupling with any other axis, \( g \) is the acceleration due to gravity, \( T \) is the torque of the motor applied to the system, \( B_1 \) is the viscous friction coefficient of the actuating arm, \( T_{f1} \) is the Coulomb friction associated with the actuating arm, \( B_2 \) is the viscous friction of the balancing arm, and \( T_{f2} \) is the Coulomb friction associated with the balancing arm. The complete derivation of these equations may be found in Appendix D.

2.3.1 Linearized Equations of Motion

Lower Balancing Configuration

When the system is linearized about the lower balancing configuration, \( \theta_1=\theta_2=\pi \), a state space controller may be found. According to Hurst, the state space matrixes become

\[
A = \frac{1}{K} \begin{bmatrix} 0 & K & 0 & 0 \\ (C_5 C_3 - C_4 C_1) & C_5 B_1 & (C_5 C_4 - C_4 C_1) - B_2 C_1 \\ 0 & 0 & 0 & K \\ (C_2 C_4 - C_3 C_1) - C_1 B_1 & (C_2 C_4 - C_4 C_1) & B_2 C_1 \end{bmatrix}, \quad B = \frac{1}{K} \begin{bmatrix} 0 \\ -C_5 \\ 0 \\ C_1 \end{bmatrix}
\]
\[
C = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\text{, and } D = \begin{bmatrix}
0 \\
0 \\
0 \\
0
\end{bmatrix}.
\]

The constants used in the equations above are defined as:

\[
C_1 = -m_2 L_1 L_{21} + m_2 L_{21}^2 + I_{2x} \\
C_2 = m_2 L_1^2 - 2m_2 L_1 L_{21} + I_{2x} + m_2 L_{21}^2 + I_{1x} + m_1 L_{11}^2 \\
C_3 = m_2 g L_1 - m_2 g L_{21} - m_1 g L_{11} \\
C_4 = -m_2 g L_{21} \\
C_5 = I_{2x} + m_2 L_{21}^2 \\
K = C_1^2 - C_2 C_5
\]

(The derivation of the linearized equations may be found in Appendix D.)

**Upper Balancing Configuration**

When the system is linearized about the upper balancing configuration, \( \theta_1=\theta_2=0 \), a state space controller may be found. According to Hurst, the state space matrixes become

\[
A = \frac{1}{K} \begin{bmatrix}
0 & K & 0 & 0 \\
(C_5 C_3 - C_4 C_1) & C_2 B_1 & (C_5 C_4 - C_4 C_1) & -B_2 C_1 \\
0 & 0 & 0 & K \\
(C_2 C_4 - C_3 C_1) & -C_1 B_1 & (C_2 C_4 - C_4 C_1) & B_2 C_2
\end{bmatrix}
\text{, and } B = \frac{1}{K} \begin{bmatrix}
0 \\
-C_5 \\
0 \\
C_1
\end{bmatrix},
\]

\[
C = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\text{, and } D = \begin{bmatrix}
0 \\
0 \\
0 \\
0
\end{bmatrix}.
\]

The constants used in the equations above are defined as:

\[
C_1 = m_2 L_1 L_{21} + m_2 L_{21}^2 + I_{2x} \\
C_2 = m_2 L_1^2 + 2m_2 L_1 L_{21} + I_{2x} + m_2 L_{21}^2 + I_{1x} + m_1 L_{11}^2 \\
C_3 = -m_2 g L_1 - m_2 g L_{21} - m_1 g L_{11} \\
C_4 = -m_2 g L_{21} \\
C_5 = I_{2x} + m_2 L_{21}^2 \\
K = C_1^2 - C_2 C_5
\]
2.4 Model Validation

The Matlab SimMechanics 2D model is validated in two manners. One of the model validation methods uses the equations of motion that are derived in the Hurst thesis [3]. The other method uses the parameter identification to ensure model validity.

2.4.1 Equations of Motion

By simultaneously plugging identical system values into the SimMechanics model and a Matlab script designed to generate the state space equations for a given set of input parameters, the model may be verified. The Matlab code used to generate the state space matrices from the equations of motion may be found in Appendix E. The format for the state space equations that is being considered here are

\[
\dot{X} = AX + BU \\
Y = CX + DU .
\]

For the lower configuration using the equations of motion, the state space matrices are

\[
A = \begin{bmatrix}
0 & 1 & 0 & 0 \\
-154.4025 & 0 & 223.0364 & 0 \\
0 & 0 & 0 & 1 \\
67.2594 & 0 & -36.2428 & 0 \\
\end{bmatrix}, \quad B = \begin{bmatrix}
0 \\
337.3337 \\
0 \\
-92.5014 \\
\end{bmatrix},
\]

\[
C = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}, \quad \text{and} \quad D = \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
\end{bmatrix}.
\]

The state space matrices for the upper configuration using the equations of motion are

\[
A = \begin{bmatrix}
0 & 1 & 0 & 0 \\
154.4147 & 0 & -223.0364 & 0 \\
0 & 0 & 0 & 1 \\
-241.5701 & 0 & 409.83 & 0 \\
\end{bmatrix}, \quad B = \begin{bmatrix}
0 \\
337.3337 \\
0 \\
-582.1659 \\
\end{bmatrix},
\]

\[
C = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}, \quad \text{and} \quad D = \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
\end{bmatrix}.
\]
For lower configuration using the Matlab model, the state space matrices are

\[
A = \begin{bmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-36.2428 & 67.2594 & 0 & 0 \\
223.0364 & -154.4147 & 0 & 0
\end{bmatrix}, \quad B = \begin{bmatrix}
0 \\
0 \\
-92.5014 \\
337.3337
\end{bmatrix}, \\
C = \begin{bmatrix}
0 & 57.2958 & 0 & 0 \\
0 & 0 & 0 & 57.2958 \\
57.2958 & 0 & 0 & 0 \\
0 & 0 & 57.2958 & 0
\end{bmatrix}, \quad \text{and} \quad D = \begin{bmatrix}
0 \\
0 \\
0 \\
0
\end{bmatrix}.
\]

The state space matrices for upper configuration using the Matlab model are

\[
A = \begin{bmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
409.83 & -241.5701 & 0 & 0 \\
-223.0364 & 154.4147 & 0 & 0
\end{bmatrix}, \quad B = \begin{bmatrix}
0 \\
0 \\
-582.1659 \\
337.3337
\end{bmatrix}, \\
C = \begin{bmatrix}
0 & 57.2958 & 0 & 0 \\
0 & 0 & 0 & 57.2958 \\
57.2958 & 0 & 0 & 0 \\
0 & 0 & 57.2958 & 0
\end{bmatrix}, \quad \text{and} \quad D = \begin{bmatrix}
0 \\
0 \\
0 \\
0
\end{bmatrix}.
\]

The differences in the C matrices are due to the fact that the equations of motion use radians for the angular information and the SimMechanics model uses degrees (the conversion factor for radians to degrees is 57.2958). The only difference between the manner in which the two state space models appear is the fact that the variables in the X matrices are in different orders. In the equations of motion, the order of the variables is actuating arm position, actuating arm velocity, balancing arm position, and balancing arm velocity. In the Matlab model, the order of the variables is balancing arm position, actuating arm position, balancing arm velocity, and actuating arm velocity. When the matrices produced by the Matlab simulation are reordered to have the same variable order as in the equations of motion, the state space matrices become identical to those produced by the equations of motion in both the upper and lower configurations. Therefore, according to the equations of motion, the Matlab SimMechanics 2D models for both the lower and upper configurations are valid.

### 2.4.2 Natural Frequency of Oscillation

The natural frequency of oscillation for the actual system’s balancing and actuating arms can be compared with the natural frequency of oscillation that is calculated using Matlab SimMechanics. The experimental procedure for determining the natural frequency is...
counting the amount of time required for the arm in question to traverse a specified number of oscillations. To determine the natural frequency by simulation, the system velocity response to an initial offset angle is recorded, and the frequency of the resulting sine wave is calculated. The frequency is calculated by marking the end of the first oscillation and noting the time at which the occurrence takes place. This time is the period of the arm oscillation. The frequency is equal to one divided by the period. There is a slight error in determining the period as the exact point where the sine wave crosses zero is not found, as can be seen in Figures 3 and 4 below.
The velocity information is used because the curve produced will have the same natural frequency as the position graph, but will provide a smoother curve. Figure 5 displays the graph of the position with respect to time of the balancing arm during oscillation. Note that the period of the position graph is identical to that of the velocity graph.
In order to determine the frequencies of the actuating and balancing arms independent of each other, the Matlab simulations are altered to allow the actuating arm to be held still during the test for the oscillation frequency of the balancing arm, and the balancing arm was removed during the test for the natural frequency of the actuating arm (see Appendices E and F for the SimMechanics models). The simulation results for rotation about the x- and y-axes are assumed to be the same, as the model assumes that the system is radially symmetric about the z-axis.

The results of the model validation by parameter identification are summarized in Table 1. The full derivation of the frequencies and resulting system inertias may be found in Appendix H.

<table>
<thead>
<tr>
<th>Rotation Axis</th>
<th>Frequency (Hz)</th>
<th>Actuating Arm</th>
<th>Balancing Arm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Experiment</td>
<td>Matlab</td>
<td>Experiment</td>
</tr>
<tr>
<td>X (Tilt)</td>
<td>0.9302</td>
<td>0.9276</td>
<td>0.676</td>
</tr>
<tr>
<td>Y (Pan)</td>
<td>0.9249</td>
<td></td>
<td>0.667</td>
</tr>
</tbody>
</table>

### 2.4.3 SolidWorks

The inertia of the system found through experimentation is also validated using the SolidWorks model of the system. Once the SolidWorks model of the system was built and the properties of each component were found and inserted into the model, the inertia of the system along with the overall mass and center of mass were determined by highlighting the desired parts in the model and looking up the mass properties. In order to compare the inertias of the SolidWorks model to those found for the actual system using the natural frequency, the axes of the SolidWorks model must be changed to those of the physical system. The x-axis is the same in both the SolidWorks model and the physical system; however the y- and z-axes are reversed in the SolidWorks model in comparison to the physical system. The mass properties table from SolidWorks for the actuating arm may be seen in Figure 6, where the inertia matrix in question is circled in red.
The inertias found through SolidWorks were 0.0039 and 0.0041 kg·m² for the moment of inertia about the x- and y-axes, respectively; whereas the inertias calculated experimentally are 0.004815 and 0.004881 kg·m² in the x- and y-axes, respectively. There is an error of 19% about the x-axis and 15% about the y-axis. Considering that the SolidWorks model does not include any screws, nuts, or other fasteners, the answer, while not extremely accurate, is within an acceptable tolerance.

The same process can be applied to the balancing arm, for which the mass properties can be seen in Figure 7. The SolidWorks moments of inertia about the x- and y-axes were both 0.0319 kg·m². The values for the moments of inertia about the x- and y-axes are 0.0287 and 0.0303 kg·m², respectively. The errors for the balancing arm inertias are 11% and 5% for the x- and y-axes, which is an acceptable amount of error. The inertias for the balancing and actuating arms are very similar to one another between what was found experimentally and what was determined using SolidWorks.
Figure 7: Mass Properties for Balancing Arm Assembly

The weights found using Solidworks are also the same or similar to what is determined experimentally. Using SolidWorks the weights are 0.90 and 0.84 lb for the actuating and balancing arm assemblies, respectively. The experimentally determined weights were found using a scale that had a tolerance of ±0.02 lb, and are found to be 0.94 lb for the actuating arm and 0.86 lb for the balancing arm. These values are almost within the tolerance of error for the scale that is used.

The centers of mass calculated for each of the arms both using SolidWorks and experimentally are also similar. Experimentally, the centers of mass values for the actuating and balancing arms were found to be 1.852 in and 11.5 in from the intersection of the x- and y-axes of rotation, respectively. The centers of mass determined by
SolidWorks are 1.53 in for the actuating arm and 12.27 in for the balancing arm about the intersection of the rotation axes.

Combining the similarities in the moments of inertia, weights, and centers of mass, the SolidWorks model can be assumed to be an accurate representation of the actual physical system.

2.5 Control Design

2.5.1 Matlab Feedback Control Loop

To create a state space controller, the linearized SimMechanics model is used. The linmod command in Matlab calls upon the model and from it creates the A, B, C and D state space matrices. Assuming that all states are weighted evenly, diagonal Q and R matrices are passed to the LQR command along with the state space matrices. The K matrix produced is used as the proportional gains that control the system.

In order to ensure that the controller is feasible, a closed-loop system is created in Simulink to simulate the inverted pendulum system. The actual system will use encoders to read the offset angles and angular velocity. These values will then be used to send a command to the motor in order to counteract any movement in the balancing arm. This process is exactly what the feedback loop in the Simulink model is all about: taking readings from the physical system and interpreting them into a motor reaction. The saturation torque of the motor is even included in the model (see Figure 8).

![Figure 8: 2D Simulink Control Loop](image)

Nonlinearities can also be introduced into the Simulink model. Figures 9 and 10 show examples of added nonlinearities, other than torque saturation. The zero order hold is representative of the sampling time of the control loop that will be implemented on the controller, in the case of this project the National Instrument cRIO. Quantization may also occur due to velocity estimation methods.
Simulink calls upon a linearized SimMechanics model as the plant of the system. Figure 10 shows that for certain conditions a simulation must stop. These cases are when the system is either uncontrollable or unstable. When stopped, the 3D visualization should cease and the program should stop running.

A Matlab script is created in order to initialize the variables of the system, find the necessary state space matrices, and produce the visualization. The script can also be programmed to run tests on the Simulink system. A plot of the angular position with respect to time is also made by the Matlab script. This chart allows the system response to the initial conditions prescribed to be monitored.

### 2.5.2 Batch Testing

To gain insight into controlling the system, first the system parameters are chosen. Batch files were run using Matlab Simulink and the SimMechanics model varying different parameters in order to understand when the system will be stable and what configurations would be more difficult to control. Initially, certain configurations were found to be uncontrollable. Upon further investigation, it was discovered that this was due to an error in the way the inertia of several system components was calculated. Since the inertia calculations have been fixed the uncontrollable system configurations have not been encountered.
The parameters that are varied in simulation are the mass of end weight, actuating arm length, and balancing arm length. The dimensions and weight of the universal joint and other structural components of the system are purposely kept at a minimum, so no batch file is created for these components. The inertia of the system is estimated using calculations based upon the length of the arms and the density of the materials they are made of (See Appendix I for the derivation of the inertia matrices).

The batch tests produce a plot of the angular position with respect to time for the various tests of different system parameters. Analyzing these charts, the system parameters for an optimized response may be obtained.

The results seen in Figures 11 and 12 are from varying both the balancing arm length (referred to as upper in the legend) from 0.2 m to 2 m and the mass of the end weight from 0 to 0.3 kg while keeping the length of the actuating arm (referred to as lower in the legend) constant at 0.11 m for the 2D configuration of the system. The initial conditions for the system are an offset angle of 1° for the balancing arm, no offset angle for the actuating arm, and there are no initial velocities in the system. There are a total of 52 successful runs, 81 unstable configurations, and no uncontrollable configurations.

The trend in the data shows that the system can compensate for larger angles when the length of the balancing arm is longer and the end weight has a larger mass. Due to torque saturation, however, there is a limit to the stability range for given parameters. The torque saturation of the motors being used will be the chief limiter of the stability and robustness of the overall system.
### Figure 11: Batch Results for Actuating Arm Angle

![Actuator Arm Angle (Deg) in Reference to Ground](image)

<table>
<thead>
<tr>
<th>Upper</th>
<th>Lower</th>
<th>End Mass</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>0.3</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>0.4</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>0.5</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>0.6</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>0.7</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>0.8</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>0.9</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.0</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.1</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.2</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.3</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.4</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.5</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.6</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.7</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.8</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.9</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>2.0</td>
<td>0.11</td>
<td>0</td>
</tr>
</tbody>
</table>

### Figure 12: Batch File Result for Balancing Arm Angle

![Balancing Arm Angle (Deg) in Reference to Actuating Arm](image)

<table>
<thead>
<tr>
<th>Upper</th>
<th>Lower</th>
<th>End Mass</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>0.3</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>0.4</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>0.5</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>0.6</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>0.7</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>0.8</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>0.9</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.0</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.1</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.2</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.3</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.4</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.5</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.6</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.7</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.8</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>1.9</td>
<td>0.11</td>
<td>0</td>
</tr>
<tr>
<td>2.0</td>
<td>0.11</td>
<td>0</td>
</tr>
</tbody>
</table>

The Matlab scripts for the 2D and 3D batch files may be found in Appendices J and K, respectively. A more complete analysis of the batch file results may be located in Appendix L.
### 2.5.3 Labview Implementation

The hierarchy for the Labview implementation can be separated into two parts: the FPGA code and the Real-Time Controller code.

The FPGA Code is executed on the FPGA. This code is designed to run as fast as possible and has a loop rate of several MHz.

- **Encoder Interface**
  - Position Calculation
  - Velocity Estimation
  - Velocity Averaging
- **Analog Input**
  - Reading Potentiometer Values
- **Analog Output**
  - Sending Command to Servo Amplifiers

To make this code reusable, it has been designed in a very hierarchal manner.

All Low Level functions have their own VIs and storage variables.

Pictures of the Main VIs can be found in Appendix M.
Real-Time Controller Code is executed on the Real-Time Controller. This code is designed to run at a fixed loop rate. This loop rate can be set in software and is on the order of 200Hz to 1 kHz. It is responsible for:

1. Scaling
   a. Converting to correct units
   b. Normalized Angle

2. Filtering
   a. Low-Pass filtering of signal inputs to cut out noise and vibrations. (Requires fast loop rates)

3. Balancing Control Loop
   a. Implements a State Space Controller

4. Swing-Up
   a. Implements a Swing up Controller

5. Selects between Controllers
   a. Switched between swing up and balancing controller based on linkage angle.

6. Recording Data
   a. Logs Data to internal storage unit on Real-Time Controller
There are many versions of the Real-Time VI because of the variety of system configurations. The different versions are organized into folders that are named based on the configuration. Pictures of one such configuration can be found in Appendix M

2.6 Controller Tuning

Once a controller is designed in the control design stage, it must be adapted to be implemented in the system. The following steps detail the procedure that is used to tune the balancing controller.

1. The four state space control gains that are generated in the control design are set in the Real-Time Control System VI.

2. The system is then placed by hand into the equilibrium position.
3. The motors are then enabled.

4. Then the system gain is raised until the system becomes stable without the need for human intervention. Note that pushing this gain too high will also cause instability.
5. At this point, the system is stable, but there are tuning steps that can be taken to improve stability.

   a. Simple Static Friction Compensation can be added. This will improve the controllability of the system, but may introduce vibrations if set too high.

   b. The velocity state space gains can be hand tuned to lessen vibrations. This is a matter varying both velocity states by 10% to 20% to try and achieve a better controller. Note that this can drastically change the performance of the system, but is found to be very useful in curbing.
Although is should have been possible to simply lock in a state space controller with a set system gain to balance the system, vibrations and mechanical problems such as backlash and compliance made fine tuning the system essential to balancing.

2.7 Design Integration

2.7.1 Mechanical Design of Three Dimensional System

The goal of designing the three dimensional physical system is to create a light weight universal joint. A universal joint allows the rod to bend at an angle in any direction relative to the other rod. For measuring the displacement angle of the balancing rod two encoders are used for the X and Y directions. The three dimensional system is also designed so that the actuating rod and the balancing arm could easily be interchanged with different lengths. For this, the system is designed so that rods could be twisted into position and tightened with a C-clamp to keep everything aligned properly. The weights are also threaded so that they can be easily interchangeable. See Appendix N for CAD drawings

A challenge when designing the system is to limit the amount of moment caused by the universal joint and the actuating arm. It is important to limit the moment because the higher the moment, the greater the torque requirements of the motor. The moment of the system is reduced by adding a counter weight to the opposite side of rotation. The material used for this system is brass. This is because it has a very high density when compared to steel and aluminum, and it is still easy to machine. Even though this
increased the inertia of the system, it greatly reduced the torque requirements of the motor. In addition another counter weight is used in the universal joint to balance out the weight of the encoder to make sure the system would be at true zero when hanging, and to make sure it would balance correctly when in the upper configuration.

2.7.2 Mechanical Design of Two Dimensional System

The two dimensional system is crucial to the development of the three dimensional system. Balancing the system in two dimensions using both the pan and tilt axes separately provide a basis to control the system in three dimensions. The three dimensional system was modified into a two dimensional system. See Appendix N for CAD drawings.

2.7.3 System Wiring

The wiring of the system is organized as specified in the wiring chart in Appendix O. The main connection system for the encoders uses CAT5e cabling. This is used to limit the amount of loose wiring for the system. Since there are four encoders in the system, each CAT5e cable holds the connections for two encoders. It is important not to send power to the motors using the CAT5e cables since there is not enough shielding between each of the wires to prevent the higher current or PWM interference from causing noise on the signals from the encoders.
3.0 Results

The following section presents the results from various balancing configurations. All configurations use an actuating arm of length 0.22 m and a balancing arm of length 0.6 m.

3.1 Two Dimensional Lower Configuration

The lower balancing configurations turn out to be the easiest to balance. Results are very encouraging and controllers need minimal hand tuning. The zone in which the pendulum can be balance is quite large (± 50 Degrees). Additionally, because of friction, the system enters into a very slow limit cycle.

![Figure 17: Balancing Result with no end Weight, Arm Angles](image)

Figure 17: Balancing Result with no end Weight, Arm Angles
The graphs below are for the two dimensional lower configuration. This is more stable than the system with no end mass.
Lower Balancing Configuration
with 0.13kg end mass
Arm Velocities

Figure 20: Balancing with .13 kg end Mass, Arm Velocities

Lower Balancing Configuration
with 0.23kg end mass
Arm Angles

Figure 21: Balancing with .23kg end Mass, Arm Angles
Figure 22: Lower Balancing with .23 kg end Mass, Arm Velocities

Figure 23: Lower Balancing with .68kg end Mass, Arm Velocities
3.2 Two Dimensional Upper Configuration

The upper configuration is found to have significant vibrations. It is also less robust in balancing. This behavior is attributed to the torque saturation of the motor and vibrations throughout the system.
Upper Balancing Configuration with 0.13kg end mass

Arm Angles

Figure 26: Upper Balancing with .13 kg end Mass, Arm Angles

Upper Balancing Configuration with 0.13kg end mass

Arm Velocities

Figure 27: Upper Balancing Configuration .13 end Mass, Arm Velocities
3.3 Three Dimensional Upper Configuration

Three dimensional balancing was achieved, but was marginal. Pendulum would be stable for less than a few seconds. Reasons for this are outlined in the conclusions section. Nevertheless, from these results, it can be seen that infinite three dimensional balancing is indeed possible with some modifications to the pan / tilt mechanism.

Figure 28: SolidWorks: Three Dimension System
3D Angle Y Axis (Unstable) with 0.13kg end mass

Figure 29: 3D Marginal Balancing with .13 kg end Mass, Y Angle

3D Balancing X (unstable) with 0.13kg end mass

Figure 30: 3D Marginal Balancing with .13 kg end Mass, X Angle
3.4 Two Dimensional Swing up

It was decided late in the project to add swing up to the list of objectives. This decision was made based on the premise that all work for swing up would be at the software level and would not require any hardware modifications. The swing up controller method is based on parts of Astrom and Furuta’s energy based control swing up [4]. The results are found to be very satisfactory, and swing up of the pendulum occurs within 3 to 5 swings.
Swing Up with no end mass

Figure 32: Swing Up with no end Mass

Swing Up with 0.13kg End mass

Figure 33: Swing up with .13kg end Mass
### 4.0 Costs

#### Parts:

<table>
<thead>
<tr>
<th>Part Number:</th>
<th>Description</th>
<th>Quantity</th>
<th>Price</th>
<th>Total Cost</th>
<th>Our Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>S2-2048-B</td>
<td>US Digital Encoder</td>
<td>4</td>
<td>$85.55</td>
<td>$342.20</td>
<td></td>
</tr>
<tr>
<td>GM8724S017</td>
<td>Pittman Motor</td>
<td>2</td>
<td>$210.00</td>
<td>$420.00</td>
<td></td>
</tr>
<tr>
<td>MSC: 74924887</td>
<td>Flange Bearings</td>
<td>4</td>
<td>$7.44</td>
<td>$29.76</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Pan gear A</td>
<td>1</td>
<td>$9.97</td>
<td>$9.97</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Pan gear B</td>
<td>1</td>
<td>$22.02</td>
<td>$22.02</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Tilt gear A</td>
<td>1</td>
<td>$7.95</td>
<td>$7.95</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Tilt gear B</td>
<td>1</td>
<td>$22.02</td>
<td>$22.02</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Pan belt</td>
<td>1</td>
<td>$3.92</td>
<td>$3.92</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Tilt belt</td>
<td>1</td>
<td>$4.00</td>
<td>$4.00</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yoke Motor Mount Plate</td>
<td>1</td>
<td>$1.42</td>
<td>$1.42</td>
<td>$1.42</td>
</tr>
</tbody>
</table>

**Parts Total**: $931.74  $1.42

#### Raw Materials:

<table>
<thead>
<tr>
<th>Part Number:</th>
<th>Description</th>
<th>Quantity</th>
<th>Price</th>
<th>Total Cost</th>
<th>Our Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSC: 32000820</td>
<td>Alloy 3003</td>
<td>2</td>
<td>$9.66</td>
<td>$19.32</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Aluminum Material:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Aluminum - Alloy 3003</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Outside Diameter: 1/2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Wall Thickness: 0.035 In.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSC: 02629335</td>
<td>Alloy 6061</td>
<td>1</td>
<td>$7.56</td>
<td>$7.56</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Aluminum Material:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Aluminum - Alloy 6061</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Diameter: 1 Length: 12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSC: 32012254</td>
<td>Alloy 6061</td>
<td>1</td>
<td>$12.38</td>
<td>$12.38</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Aluminum Material:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Aluminum - Alloy 6061</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Length: 72 Length: 6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Thickness: 1/4 Width: 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N/A</td>
<td>Brass</td>
<td>1.6 lb</td>
<td>$6 per lb</td>
<td>$10.00</td>
<td>$10.00</td>
</tr>
</tbody>
</table>

**Raw Materials Total**: $49.26  $10.00

#### Labor Costs

<table>
<thead>
<tr>
<th>Name:</th>
<th>Title</th>
<th>Hours</th>
<th>Cost</th>
<th>Total Cost</th>
<th>Our Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brian Lewis</td>
<td>Engineer</td>
<td>150.75</td>
<td>$40</td>
<td>$6,030</td>
<td>$ -</td>
</tr>
<tr>
<td>Teresa Bernardi</td>
<td>Engineer</td>
<td>170.75</td>
<td>$40</td>
<td>$6830</td>
<td>$ -</td>
</tr>
<tr>
<td>Matthew Rosmarin</td>
<td>Engineer</td>
<td>170.25</td>
<td>$40</td>
<td>$6810</td>
<td>$ -</td>
</tr>
<tr>
<td>Adjima Moreira</td>
<td>Engineer</td>
<td>20.5</td>
<td>$40</td>
<td>$820</td>
<td>$ -</td>
</tr>
</tbody>
</table>

**Labor Costs Total**: $20,490  $ -

**Final Cost**: $21,471  $11.42
5.0 Conclusion

Balancing is achievable in both the pan and tilt configurations in two dimensions in both the lower and upper configurations. Balancing is found to be more robust in the lower configuration since it requires less torque. Balancing in three dimensions was marginal. System would be balancing for a few seconds with considerable vibrations at critical frequencies. Vibrations in addition to backlash and compliance did not allow for infinite balancing. Balancing in three dimensions could be greatly improved by redesigning the actuation system to be more rigid, eliminate the reliance on couplings such as belts and gears, using a large motor on the pan axis.

These conclusions are based on the fact that the belt nonlinearities greatly affect the system. Swing up would work in one configuration, but if the system was started 360 degrees ahead, swing up would not work due to the nonlinear friction in the belt. One of the belts was completely stripped due to too much torque on the belt.

Swing up was also more successful with no end mass. This is partly due to the torque requirements. With the bigger end mass the belt would begin to strip.
6.0 Bibliography


Appendix A: Terms

Abbreviations:

- 2D = two dimensional
- 3D = three dimensional
- cRIO = Compact Remote Input Output or Compact RIO
- FPGA = Field Programmable Gate Array

Figure 34: Device Parts
Appendix B: Matlab for 2D Lower Configuration

Figure 35: 2D Lower Configuration Matlab Plant

Figure 36: 2D Lower Configuration Matlab Simulink
39

setup.m
% Sim Mechanics Simulation Batch Setup for 2D Balancing Pendulum
% in Lower Position
% April 18, 2006 M.R.

clc
clear

% Define desired parameters of actuated and balancing arms
Driven.Mass=0.4264;  % in kg
Driven.Length=0.2223; % in m
Driven.Cg=0.04704;   % in m
Balance.Mass=0.3901;  % in kg
Balance.Length=0.5921; % in m
Balance.Cg=0.2921; % in m

%Initial Conditions
BalanceArm.Angle=1; %deg
BalanceArm.Velocity=0; %deg/s

DrivenArm.Angle=00; %deg
DrivenArm.Velocity=0; %deg/s

SamplePeriod=0.01
Motor.TorqueSaturation=.5; %N-m
SystemGain=1;

%Control Design

[A B C D] = linmod('Plant') %Linearize System
SS = ss(A,B,C,D)
Q=[100 0 0 0
   0 100 0 0
   0 0 100 0
   0 0 0 100];
[K,S,e] = lqr(A,B,Q,1) %Create Controller

sim('HighLevel')

plot(tout,yout)
Appendix C: Matlab for 2D Upper Configuration

Figure 39: 2D Upper Configuration Matlab Plant

Figure 40: 2D Upper Configuration Matlab Simulink
setup.m
% Sim Mechanics Simulation Batch Setup for 2D Balancing Pendulum
% in Upper Position
% April 18, 2006 M.R.
clc
clear

% Define desired parameters of actuated and balancing arms
Driven.Mass=0.4264; % in kg
Driven.Length=0.2223; % in m
Driven.Cg=0.04704; % in m
Balance.Mass=0.3901; % in kg
Balance.Length=0.5921; % in m
Balance.Cg=0.2921; % in m

%Initial Conditions
BalanceArm.Angle=1; %deg
BalanceArm.Velocity=0; %deg/s

DrivenArm.Angle=00; %deg
DrivenArm.Velocity=0; %deg/s

SamplePeriod=0.01
Motor.TorqueSaturation=.5; %N-m
SystemGain=1;

%Control Design

[A B C D] = linmod('PlantUpper') %Linearize System
SS = ss(A,B,C,D)
Q=[100 0 0 0
   0 100 0 0
   0 0 100 0
   0 0 0 100];
[K,S,e] = lqr(A,B,Q,1) %Create Controller

sim('HighLevelUpper')
plot(tout,yout)
Appendix D: Equations of Motion

See next page
3. Math Model Development: Dynamics

In this section reference frames and position vectors are defined, then equations of motion are developed using the lagrange method.

3.1 Reference Frames and Definitions

Figure 3.1 shows the zero configuration for the system: the upright position, and Figure 3.2 shows an offset configuration.

Definitions:
0: origin for ground/base frame
1: origin for reference frame 1
2: origin for reference frame 2
a: location of center-of-gravity of link 1
b: location of center-of-gravity of link 2
L1: length of link 1
L2: length of link 2
L11: length from origin 1 to center-of-gravity of link 1
L12: length from center-of-gravity of link 1 to origin 2
L21: length from origin 2 to center-of gravity of link 2
\( \vec{h}_1 \): revolute axis for joint 1 (with base at origins 0 and 1)
\( \vec{h}_2 \): revolute axis for joint 2 (with base at origin 2)

Next define reference frame, revolute axes, transformation matrices and position vectors:

- **Reference Frame:** \( x = [1, 0, 0]^T, y = [0, 1, 0]^T, z = [0, 0, 1]^T \)

- **Revolute axes:** \( h_1 = h_2 = x = [1, 0, 0]^T \) (defined in respective base frames)

- **Transformation matrices:**

\[
R_{01} = e^{h_1 \theta_1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta_1 & -\sin \theta_1 \\ 0 & \sin \theta_1 & \cos \theta_1 \end{bmatrix}, \quad R_{12} = e^{h_2 \theta_2} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta_2 & -\sin \theta_2 \\ 0 & \sin \theta_2 & \cos \theta_2 \end{bmatrix}
\]

\[
R_{02} = R_{01}R_{12} = e^{h_1 \theta_1} e^{h_2 \theta_2} = e^x(\theta_1 + \theta_2) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_1 + \theta_2) & -\sin(\theta_1 + \theta_2) \\ 0 & \sin(\theta_1 + \theta_2) & \cos(\theta_1 + \theta_2) \end{bmatrix}
\]

- **Position vectors:** defined with respect to their base frames. To evaluate the vectors they must be transformed to the appropriate evaluation frame.

\[ p_{01} = 0 \]
Figure 3.3: Position vector definitions

\[
\begin{align*}
\vec{p}_{12} &= L_1 \hat{z} \text{ (in the 1 frame)} \\
\vec{p}_{1a} &= L_{11} \hat{z} \text{ (in the 1 frame)} \\
\vec{p}_{2b} &= L_{21} \hat{z} \text{ (in the 2 frame)} \\
\vec{p}_{1b} &= \vec{p}_{12} + \vec{p}_{2b}
\end{align*}
\]

3.2 Lagrange Method

The following is the common well known form of Lagrange’s equations:

\[
\frac{d}{dt} \left( \frac{\partial T}{\partial \dot{q}_i} \right) - \frac{\partial T}{\partial q_i} + \frac{\partial V}{\partial q_i} = Q_i
\]

\(T\) is the kinetic energy of the system, \(V\) is the potential energy of the system, and \(Q_i\) are the generalized forces for each generalized coordinate. To use the lagrange method, we must define these parameters, and perform the necessary operations.

Definition of new terms:

- \(\omega_a\): angular velocity vector of a (link1)
- \(\omega_b\): angular velocity vector of b (link2)
- \(\dot{\theta}_1\): angular velocity of a (link1)
- \(\dot{\theta}_2\): angular velocity of b (link2)
- \((\omega_a)_0\): angular velocity vector of a evaluated in ground frame
- \((\omega_b)_0\): angular velocity vector of b evaluated in ground frame
- \(V_a\): velocity vector of a
\( V_b \): velocity vector of b  
\( V_{b/2} \): velocity of b with respect to origin 2  
\( (V_a)_0 \): velocity vector of a evaluated in ground frame  
\( (V_b)_0 \): velocity vector of b evaluated in ground frame

### 3.2.1 Kinetic energy

- Angular Velocities:

\[
\begin{align*}
\omega_a &= \dot{\theta}_1 h_1 \\
\omega_b &= \omega_a + \dot{\theta}_2 h_2
\end{align*}
\]

Evaluated in newtonian frame \((0)\):

\[
\begin{align*}
(\omega_a)_0 &= \dot{\theta}_1 R_01 h_1 = \dot{\theta}_1 R_01 x \\
(\omega_b)_0 &= \dot{\theta}_1 R_01 h_1 + \dot{\theta}_2 R_02 h_2 = (\omega_b)_0 = \dot{\theta}_1 R_01 x + \dot{\theta}_2 R_02 x
\end{align*}
\]

- Velocities of center-of-gravity:

\[
\begin{align*}
V_a &= \omega_a \times p_1 a \\
V_b &= V_{b/2} + V_2 = \omega_b \times p_2 b + \omega_a \times p_{12}
\end{align*}
\]

Evaluated in newtonian frame \((0)\):

\[
\begin{align*}
(V_a)_0 &= (\omega_a)_0 \times R_01 p_1 a \\
(V_b)_0 &= (\omega_b)_0 \times R_02 p_2 b + (\omega_a)_0 \times R_01 p_{12}
\end{align*}
\]

- Inertias: (in general a \(3 \times 3\))

\[
I_1 = \begin{pmatrix}
I_{1xx} & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{pmatrix},
I_2 = \begin{pmatrix}
I_{2xx} & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{pmatrix}
\]

Evaluated in newtonian frame \((0)\):

\[
(I_1)_0 = R_01 I_1 R_01^T, (I_2)_0 = R_02 I_2 R_02^T
\]
With all quantities defined, kinetic energy can be stated:

\[
T = \frac{1}{2}m_1(V_a)_0^2 + \frac{1}{2}m_2(V_b)_0^2 + \frac{1}{2}(\omega_a)_0^2(I_1)_0 + \frac{1}{2}(\omega_b)_0^2(I_2)_0
\]

### 3.2.2 Potential energy

Reference is top of link 2 in zero configuration. Figure 3.4 shows the definitions for potential energy calculation.

![Figure 3.4: Potential energy definitions](image)

\[
H_1 = L_1 + L_2 - L_{11} \cos \theta_1
\]

\[
H_2 = L_1 + L_2 - L_{11} \cos \theta_1 - L_{21} \cos(\theta_2 + \theta_1)
\]

\[
V = -m_1 g H_1 - m_2 g H_2
\]

### 3.2.3 Generalized forces

\[
Q_1 = T - B_1 \dot{\theta}_1 - T f_1 sgn(\theta_1)
\]

\[
Q_2 = -B_2 \dot{\theta}_2 - T f_2 sgn(\theta_2)
\]

\[
T: \quad \text{input torque}
\]

\[
B_1, B_2: \quad \text{viscous friction coefficients}
\]

\[
T f_1, T f_2: \quad \text{coulomb friction coefficients}
\]

\[
g: \quad \text{gravity}
\]
3.2.4 Lagrange equations

After defining $T$, $V$, and $Q_i$ evaluate Lagrange equations:

$$\frac{d}{dt} \left( \frac{\partial T}{\partial \dot{\theta}_1} \right) - \frac{\partial T}{\partial \theta_1} + \frac{\partial V}{\partial \theta_1} = Q_1$$

$$\frac{d}{dt} \left( \frac{\partial T}{\partial \dot{\theta}_2} \right) - \frac{\partial T}{\partial \theta_2} + \frac{\partial V}{\partial \theta_2} = Q_2$$

Evaluation yields the following two coupled equations of motion:

$$(m_2 L_1 L_2 \cos \theta_2 + m_2 L_2 L_1^2 + I_{2xx}) \ddot{\theta}_2 + (m_2 L_1^2 + 2m_2 L_1 L_2 \cos \theta_2 + I_{2xx} + m_2 L_2 L_1^2 + I_{1xx} + m_1 L_1^2) \ddot{\theta}_1 - m_2 g L_21 \sin(\theta_1 + \theta_2) - m_2 L_1 L_2 \theta_2^2 \sin \theta_2 - m_2 g L_1 \sin \theta_1 - 2m_2 L_1 L_2 \dot{\theta}_1 \dot{\theta}_2 \sin \theta_2 - m_1 g L_11 \sin \dot{\theta}_1 = -T - B_1 \dot{\theta}_1 - T f_1 \text{sgn}(\theta_1)$$

$$(I_{2xx} + m_2 L_2 L_1^2) \ddot{\theta}_2 + (m_2 L_1 L_2 \cos \theta_2 + m_2 L_2 L_1^2 + I_{2xx}) \dot{\theta}_1 + m_2 L_1 L_2 \theta_1^2 \sin \theta_2 - m_2 g L_21 \sin(\theta_1 + \theta_2) = -B_2 \dot{\theta}_2 - T f_2 \text{sgn}(\theta_2)$$
5. Controller Design

After verifying the math model it can then be used as a basis for controller design.

5.1 Linearization

The pendubot contains 4 equilibrium points. The two of most interest are when link 2 is upright and in its unstable equilibrium and link 1 is both in its stable and unstable positions. These two situations are shown in Figures 5.1 and 5.2.

Figure 5.1: Unstable Equilibrium Position of Pendubot

Recall the equations of motion for this system:

\[(m_2L_1L_21 \cos \theta_2 + m_2L_21^2 + I_2_{xx})\ddot{\theta}_2 + (m_2L_1^2 + 2m_2L_1L_21 \cos \theta_2 + I_2_{xx} + m_2L_21^2 + I_1_{xx} + m_1L_11^2)\dot{\theta}_1 - m_2gL_21 \sin(\theta_1 + \theta_2) - m_2L_1L_21\dot{\theta}_2^2 \sin \theta_2 - m_2gL_1 \sin \theta_1 - 2m_2L_1L_21 \dot{\theta}_1 \dot{\theta}_2 \sin \theta_2 - m_1gL_11 \sin \theta_1 = T - B_1\dot{\theta}_1 - T_{f1}sgn(\theta_1)\]

\[(I_2_{xx} + m_2L_21^2)\ddot{\theta}_2 + (m_2L_1L_21 \cos \theta_2 + m_2L_21^2 + I_2_{xx})\ddot{\theta}_1 + m_2L_1L_21\dot{\theta}_1^2 \sin \theta_2 - m_2gL_21 \sin(\theta_1 + \theta_2) = -B_2\dot{\theta}_2 - T_{f2}sgn(\theta_2)\]
5.1.1 Linearization About $\theta_1 = 0$, $\theta_2 = 0$

Performing a Taylor series expansion about the operating point $\theta_1 = 0$, $\theta_2 = 0$ and keeping only the linear terms:

\[
\begin{align*}
&(m_2L1L21 + m_2L21^2 + I_{2xx})\ddot{\theta}_2 + (m_2L1^2 + 2m_2L1L21 + I_{2xx} + m_2L21^2 + I_{1xx} + m_1L11^2)\ddot{\theta}_1 + B_1\dot{\theta}_1 - T + (-m_2gL1 - m_2gL21 - m_1gL11)\theta_1 + (-m_2gL21)\theta_2 \\
&(I_{2xx} + m_2L21^2)\dddot{\theta}_2 + (m_2L1L21 + m_2L21^2 + I_{2xx})\dddot{\theta}_1 + B_2\dddot{\theta}_2 + (-m_2gL21)\dddot{\theta}_1 + \theta_1 + (-m_2gL21)\dddot{\theta}_2 \\
\end{align*}
\]

Define:

- $C_1 = m_2L1L21 + m_2L21^2 + I_{2xx}$
- $C_2 = m_2L1^2 + 2m_2L1L21 + I_{2xx} + m_2L21^2 + I_{1xx} + m_1L11^2$
- $C_3 = -m_2gL1 - m_2gL21 - m_1gL11$
• $C_4 = -m_2 g L_{21}$
• $C_5 = I_{2xx} + m_2 L_{21}^2$

The linearized equations become:

\[
C_1 \ddot{\theta}_2 + C_2 \ddot{\theta}_1 + B_1 \dot{\theta}_1 + C_3 \theta_1 + C_4 \theta_2 - T
\]

\[
C_5 \ddot{\theta}_2 + C_1 \ddot{\theta}_1 + B_2 \dot{\theta}_2 + C_4 \theta_1 + C_4 \theta_2
\]

Solving the two linear equations for $\ddot{\theta}_1$ and $\ddot{\theta}_2$ allows the equations to be put into state space form:

Define: $K = C_1^2 - C_2 C_5$

\[
A = \frac{1}{K} \begin{pmatrix}
0 & K & 0 & 0 \\
(C_5 C_3 - C_4 C_1) & C_5 B_1 & (C_5 C_4 - C_4 C_1) & -B_2 C_1 \\
0 & 0 & 0 & K \\
(C_2 C_4 - C_3 C_1) & -C_1 B_1 & (C_2 C_4 - C_4 C_1) & B_2 C_2
\end{pmatrix}
\]

\[
B = \frac{1}{K} \begin{pmatrix}
0 \\
-C_5 \\
0 \\
C_1
\end{pmatrix},
C = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix},
D = \begin{pmatrix}
0 \\
0 \\
0 \\
0
\end{pmatrix}
\]

5.1.2 Linearization About $\theta_1 = \pi, \theta_2 = \pi$

Following the same procedure as before define:

• $C_1 = -m_2 L_1 L_{21} + m_2 L_{21}^2 + I_{2xx}$
• $C_2 = m_2 L_1^2 - 2m_2 L_1 L_{21} + I_{2xx} + m_2 L_{21}^2 + I_1_{xx} + m_1 L_{11}^2$
• $C_3 = m_2 g L_1 - m_2 g L_{21} + m_1 g L_{11}$
• $C_4 = -m_2 g L_{21}$
• $C_5 = I_{2xx} + m_2 L_{21}^2$
• $K = C_1^2 - C_2 C_5$

With $A$, $B$, $C$ and $D$ defined the same as before.
Appendix E: Matlab Code for Equations of Motion

% Calculation of the State-Space Model for
% 2D Lower and Upper Configurations using
% the Model Provided by Joshua Hurst for a
% Given Set of Parameters
%
% T.B. April 30, 2006

m1=0.7348;           % in kg
m2=0.23587;           % in kg
L1=0.28575;          % in m
L11=0.0635;          % in m
L21=0.3937;           % in m
theta1=0;         % in rad
theta2=1;         % in rad
theta1_dot=0;    % in rad/sec
theta2_dot=0;    % in rad/sec
I1xx=0.000001;   % in kg*m^2
I2xx=0.000001;   % in kg*m^2
B1=0;
B2=0;
g=9.81;           % in m/sec^2

% Lower configuration state space
C1_l=-m2*L1*L21+m2*L21^2+I2xx;
C2_l=m2*L1^2-2*m2*L1*L21+I2xx+m2*L21^2+I1xx+m1*L11^2;
C3_l=m2*g*L1-m2*g*L21+m1*g*L11;
C4_l=-m2*g*L21;
C5_l=I2xx+m2*L21^2;
K_l=C1_l^2-C2_l*C5_l;
A_l=[0 K_l 0 0;(C5_l*C3_l-C4_l*C1_l) C5_l*B1 (C5_l*C4_l-C4_l*C1_l) -B2*C1_l;0 0
     0 K_l;0](C2_l*C4_l-C3_l*C1_l) -C1_l*B1 (C2_l*C4_l-C4_l*C1_l) B2*C2_l)/K_l;
B_l=[0;-C5_l;0;C1_l]/K_l;
C_l=eye(4);
D_l=[0;0;0;0]

% Upper configuration state space
C1_u=m2*L1*L21+m2*L21^2+I2xx;
C2_u=m2*L1^2+2*m2*L1*L21+I2xx+m2*L21^2+I1xx+m1*L11^2;
C3_u=-m2*g*L1+m2*g*L21+m1*g*L11;
C4_u=-m2*g*L21;
C5_u=I2xx+m2*L21^2;
K_u=C1_u^2-C2_u*C5_u;

54
\[
A_u = \begin{bmatrix} 0 & K_u & 0 & 0 \\ (C_5_u \cdot C_3_u \cdot C_4_u \cdot C_1_u) & C_5_u \cdot B_1 & (C_5_u \cdot C_4_u \cdot C_1_u) - B_2 \cdot C_1_u & 0 \\ 0 & 0 & K_u & (C_2_u \cdot C_4_u \cdot C_3_u \cdot C_1_u) - C_1_u \cdot B_1 \\ B_2 \cdot C_2_u & 0 & 0 & K_u \end{bmatrix} / K_u
\]

\[
B_u = \begin{bmatrix} 0 \\ -C_5_u \\ 0 \\ C_1_u \end{bmatrix} / K_u
\]

\[
C_u = \text{eye}(4)
\]

\[
D_u = [0; 0; 0; 0]
\]
Appendix F: Matlab Model and Code for Frequency Calculation of Actuating Arm

The code for calculating the natural frequency of the actuating arm of the balancing inverted pendulum system may be found below. The Matlab script defines the parameters of the physical system and produces the graph of the velocity of actuating arm as it oscillates.

`frequency.m`

```
% Sim Mechanics Simulation Batch Setup for 2D Balancing Pendulum
% for Calculating Frequency
% May 3, 2006 M.R. & T.B.

clc
clear

% Define desired parameters of actuated and balancing arms
Driven.Mass=0.4264; % in kg
Driven.Length=0.2223; % in m
Driven.Cg=0.04704; % in m
```
Balance.Mass=0.3901; % in kg
Balance.Length=0.5921; % in m
Balance.Cg=0.2921; % in m

% Initial Conditions
BalanceArm.Angle=1; % deg
BalanceArm.Velocity=0; % deg/s

DrivenArm.Angle=00; % deg
DrivenArm.Velocity=0; % deg/s

SamplePeriod=0.01
Motor.TorqueSaturation=.5; % N-m
SystemGain=1;

% Control Design

sim('Plant')

plot(tout,yout(:,2))
title('Oscillation of Actuating Arm')
xlabel('Time (sec)')
ylabel('Velocity (m/sec)')
The code for calculating the natural frequency of the balancing arm of the inverted pendulum system may be found below. The Matlab script defines the parameters of the physical system and produces the graph of the velocity of balancing arm as it oscillates.

`frequency.m`

% Sim Mechanics Simulation Batch Setup for 2D Balancing Pendulum
clc
clear

% Define desired parameters of actuated and balancing arms
Driven.Mass=0.4264;             % in kg
Driven.Length=0.2223;          % in m
Driven.Cg=0.04704;               % in m
Balance.Mass=0.3901;           % in kg
Balance.Length=0.5921;          % in m
Balance.Cg=0.2921;              % in m

% Initial Conditions
BalanceArm.Angle=1; % deg
BalanceArm.Velocity=0; % deg/s
DrivenArm.Angle=0; % deg
DrivenArm.Velocity=0; % deg/s
SamplePeriod=0.01
Motor.TorqueSaturation=.5; % N-m
SystemGain=1;

% Control Design

sim('Plant')

plot(tout,yout(:,4))
title('Oscillation of Balancing Arm')
xlabel('Time (sec)')
ylabel('Velocity (m/sec)')
### Appendix H: Experimental Results for Parameter Identification Model Validation

In order to determine the natural frequency for each the actuating and balancing arms, the respective arm is allowed to swing freely while the rest of the system is held rigid. The experiment is conducted at least twice in each the x- and y-axes for both arms.

<table>
<thead>
<tr>
<th>Rotation Axis</th>
<th>Number of Oscillations</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>X (Tilt)</td>
<td>5</td>
<td>5.422</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>5.344</td>
</tr>
<tr>
<td>Y (Pan)</td>
<td>5</td>
<td>5.531</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>5.297</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>5.406</td>
</tr>
</tbody>
</table>

**Table 2: Experimental Frequency Data for Actuating Arm**

<table>
<thead>
<tr>
<th>Rotation Axis</th>
<th>Number of Oscillations</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>X (Tilt)</td>
<td>50</td>
<td>74</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>37</td>
</tr>
<tr>
<td>Y (Pan)</td>
<td>50</td>
<td>75</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>37.5</td>
</tr>
</tbody>
</table>

**Table 3: Experimental Frequency Data for Balancing Arm**

The natural frequencies of the two arms are calculated about each axis by dividing the number of oscillations by the average amount of time required to complete the oscillations. The calculated natural frequencies for the actuating arm are

\[
\begin{align*}
    f_{nx} & = \frac{5}{5.375\text{sec}} = 0.9302\text{Hz} \quad \text{and} \\
    f_{ny} & = \frac{5}{5.406\text{sec}} = 0.9249\text{Hz},
\end{align*}
\]

and for the balancing arm the natural frequencies are

\[
\begin{align*}
    f_{nx} & = \frac{50}{74\text{sec}} = 0.676\text{Hz} \quad \text{and} \\
    f_{ny} & = \frac{50}{75\text{sec}} = 0.667\text{Hz}.
\end{align*}
\]

From the natural frequencies the inertias about the axes of rotation may be calculated using the formula

\[
\begin{align*}
    f_n & = \frac{1}{2\pi} \sqrt{\frac{mgr}{I_o}} \quad \text{or} \\
    I_o & = \frac{mgr}{(2\pi f_n)^2},
\end{align*}
\]
where $I_o$ is the inertia, $m$ is the mass of the arm, $g$ is the acceleration due to gravity, and $r$ is distance from the axis of rotation of the center of mass. A summary of the necessary characteristics of the actuating and balancing arms can be seen in Table 4.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Actuating Arm</th>
<th>Balancing Arm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight (lb)</td>
<td>0.94</td>
<td>0.86</td>
</tr>
<tr>
<td>$r_x$ (in)</td>
<td>1.852</td>
<td>11.5</td>
</tr>
<tr>
<td>$r_y$ (in)</td>
<td>1.852</td>
<td>11.5</td>
</tr>
</tbody>
</table>

Table 4: Summary of Arm Characteristics

The weight in the table above may be used to replace the product of the mass and acceleration due to gravity. The inertia equation now becomes

$$I_o = \frac{Wr}{(2\pi f)^2}.$$  

The calculations for the actuating arm inertias about the rotation axis become

$$I_{ox} = \frac{(0.94 lb)(1.852 \text{ ft})}{2\pi(0.9302Hz)^2} \left( \frac{1 \text{ slug} \cdot \text{ ft} / \text{ sec}^2}{1 \text{ lb}} \right) \left( \frac{14.59 \text{ kg}}{1 \text{ slug}} \right) \left( \frac{0.3048 m}{1 \text{ ft}} \right)^2 = 0.005757 \text{ kg} \cdot \text{ m}^2 \text{ and}$$  

$$I_{oy} = \frac{(0.94 lb)(1.852 \text{ ft})}{2\pi(0.9249Hz)^2} \left( \frac{1 \text{ slug} \cdot \text{ ft} / \text{ sec}^2}{1 \text{ lb}} \right) \left( \frac{14.59 \text{ kg}}{1 \text{ slug}} \right) \left( \frac{0.3048 m}{1 \text{ ft}} \right)^2 = 0.005823 \text{ kg} \cdot \text{ m}^2.$$  

Note that the units are converted from English units to metric units. The reason for this conversion is the fact that the Matlab SimMechanics model is set up to take in all variables in metric units. The model, however, is also designed for the inertia of the arms to be about their center of gravity. Therefore, the parallel axis theorem must be applied to find the inertias necessary for the simulation. The equation for the parallel axis theorem is

$$I_o = I_g + r^2 m,$$

where $I_o$ is the inertia about the axis of rotation, $I_g$ is the inertia about the center of gravity, and $r$ is the distance from the origin of the center of gravity to the origin of the axis of rotation. The equation to calculate the inertia about the center of gravity becomes

$$I_g = I_o - r^2 m = I_o - r^2 \frac{W}{g},$$

The calculations for the inertias about the center of gravity for the actuating arm are

$$I_{gx} = I_{ox} - r^2 \frac{W}{g}$$

$$I_{gx} = \left( 0.005757 \text{ kg} \cdot \text{ m}^2 \right) - \left( \frac{1.852 \text{ ft}}{12} \right)^2 \left( \frac{0.94 \text{ lb}}{32.2 \text{ ft} / \text{ sec}^2} \right) \left( \frac{14.59 \text{ kg}}{1 \text{ slug}} \right) \left( \frac{0.3048 m}{1 \text{ ft}} \right)^2$$

$$I_{gx} = \left( 0.005757 \text{ kg} \cdot \text{ m}^2 \right) - (0.000942 \text{ kg} \cdot \text{ m}^2) = 0.004815 \text{ kg} \cdot \text{ m}^2$$

and

$$I_{gy} = \left( 0.005823 \text{ kg} \cdot \text{ m}^2 \right) - (0.000942 \text{ kg} \cdot \text{ m}^2) = 0.004881 \text{ kg} \cdot \text{ m}^2.$$
There is no inertia about the z-axis at the center of gravity because the actuating arm is radially symmetric. Similarly the inertia calculations about the rotation axis for the balancing arm are

\[
I_{ox} = \frac{(0.86lb)\left(\frac{11.5}{12} \text{ ft}\right)}{2\pi(0.676Hz)^2} \left(\frac{1\text{ slug \cdot ft/sec}}{1\text{ lb}}\right) \left(\frac{14.59kg}{1\text{ slug}}\right) \left(\frac{0.3048m}{1\text{ ft}}\right)^2 = 0.06192kg \cdot m^2 \quad \text{and}
\]

\[
I_{oy} = \frac{(0.86lb)\left(\frac{1.852}{12} \text{ ft}\right)}{2\pi(0.667Hz)^2} \left(\frac{1\text{ slug \cdot ft/sec}}{1\text{ lb}}\right) \left(\frac{14.59kg}{1\text{ slug}}\right) \left(\frac{0.3048m}{1\text{ ft}}\right)^2 = 0.06360kg \cdot m^2,
\]

and the inertia calculations about the center of gravity are

\[
I_{gx} = \left(0.06192kg \cdot m^2\right) - \left(\frac{11.5}{12} \text{ ft}\right)^2 \left(\frac{0.94lb}{32.2 \text{ ft/}^2\text{sec}^2}\right) \left(\frac{1\text{ slug \cdot ft/}^2\text{sec}}{1\text{ lb}}\right) \left(\frac{14.59kg}{1\text{ slug}}\right) \left(\frac{0.3048m}{1\text{ ft}}\right)^2 = 0.02867kg \cdot m^2
\]

and

\[
I_{gy} = \left(0.06360kg \cdot m^2\right) - \left(0.03325kg \cdot m^2\right) = 0.03035kg \cdot m^2.
\]

A summary of the frequencies and inertias for both the actuating and balancing arms may be seen in Table 5 below.

<table>
<thead>
<tr>
<th></th>
<th>Actuating Arm</th>
<th>Balancing Arm</th>
</tr>
</thead>
<tbody>
<tr>
<td>(f_{ox} ) (Hz)</td>
<td>0.9302</td>
<td>0.676</td>
</tr>
<tr>
<td>(f_{oy} ) (Hz)</td>
<td>0.9249</td>
<td>0.667</td>
</tr>
<tr>
<td>(I_{ox} ) (kg\cdotm²)</td>
<td>0.005757</td>
<td>0.06192</td>
</tr>
<tr>
<td>(I_{oy} ) (kg\cdotm²)</td>
<td>0.005823</td>
<td>0.06360</td>
</tr>
<tr>
<td>(I_{gx} ) (kg\cdotm²)</td>
<td>0.004815</td>
<td>0.02867</td>
</tr>
<tr>
<td>(I_{gy} ) (kg\cdotm²)</td>
<td>0.004881</td>
<td>0.03035</td>
</tr>
</tbody>
</table>

Table 5: Summary of Experimental Results
Appendix I: Inertia Calculations

For end mass:

Considering the x-axis to be along the height of the end mass and the y- and z-axes to be in radial directions, the basic equations for the end mass are

\[
I_x = \frac{1}{2} mr^2 \quad \text{and} \quad I_y = I_z = \frac{1}{12} m(3r^2 + h^2),
\]

where r is the radius and h is the height. Just taking this into consideration, the inertia matrix is

\[
I = \begin{bmatrix}
\frac{1}{2} mr^2 & 0 & 0 \\
0 & \frac{1}{12} m(3r^2 + h^2) & 0 \\
0 & 0 & \frac{1}{12} m(3r^2 + h^2)
\end{bmatrix}.
\]

Since the inertia needs to be defined in terms of the density, the relation seen below must be employed –

\[
m = \rho h \pi r^2.
\]

This makes the inertia equations

\[
I_x = \frac{1}{2} \rho h \pi r^4 \quad \text{and} \quad I_y = I_z = \frac{1}{12} \rho h \pi (3r^4 + h^2 r^2).
\]

Since the end mass is symmetric about its center of gravity in the xy-, xz- and yz-planes, the product moments of inertia are

\[
I_{xy} = I_{xz} = I_{yz} = 0.
\]

This makes the inertia matrix

\[
I = \frac{1}{2} \rho h \pi \begin{bmatrix}
r^4 & 0 & 0 \\
0 & \frac{1}{6} (3r^4 + h^2 r^2) & 0 \\
0 & 0 & \frac{1}{6} (3r^4 + h^2 r^2)
\end{bmatrix}.
\]

For arm tubing:

Considering the x-axis to be along the height of the tubing and the y- and z-axes to be in radial directions, the equations for the moments of inertia will be derived from those for a circular cylinder,
where \( r \) is the radius and \( L \) is the length of the cylinder. The way that the tubing equations are derived is by considering the tubing to be two separate cylinders – the first using the outer diameter of the tubing and the second using the inner diameter of the tubing. The second cylinder will be considered to have negative inertia, since the material is absent from the actual tubing. The basic equations for the tubing inertia now become

\[
I_x = \frac{1}{2} m_{\text{outer}} r_{\text{outer}}^2 - \frac{1}{2} m_{\text{inner}} r_{\text{inner}}^2 \quad \text{and} \quad I_y = I_z = \frac{1}{12} m_{\text{outer}} \left( 3r_{\text{outer}}^2 + L^2 \right) - \frac{1}{12} m_{\text{inner}} \left( 3r_{\text{inner}}^2 + L^2 \right).
\]

Since the inertia will be a function of the length only, the mass with respect to the desired length of the tubing will be considered

\[
m_{\text{outer}} = \rho L \pi r_{\text{outer}}^2 \quad \text{and} \quad m_{\text{inner}} = \rho L \pi r_{\text{inner}}^2
\]

for the outer and inner cylinders, respectively.

The inertia equations are now

\[
I_x = \frac{1}{2} \rho \pi L r_{\text{outer}}^4 - \frac{1}{2} \rho \pi L r_{\text{inner}}^4 \quad \text{and} \quad I_y = I_z = \frac{1}{12} \rho \pi L \left( 3r_{\text{outer}}^4 + L^2 r_{\text{outer}}^2 \right) - \frac{1}{12} \rho \pi L \left( 3r_{\text{inner}}^4 + L^2 r_{\text{inner}}^2 \right).
\]

These equations can be simplified to

\[
I_x = \frac{1}{2} \rho \pi L \left( r_{\text{outer}}^4 - r_{\text{inner}}^4 \right) \quad \text{and} \quad I_y = I_z = \frac{1}{12} \rho \pi L \left[ 3 \left( r_{\text{outer}}^4 - r_{\text{inner}}^4 \right) + L^2 \left( r_{\text{outer}}^2 - r_{\text{inner}}^2 \right) \right].
\]

However, since the tubing is symmetric about all planes, the product moments of inertia should all be zero. This makes the inertia matrix for the tubing

\[
I = \frac{1}{2} \rho L \pi \begin{bmatrix}
  r_{\text{outer}}^4 - r_{\text{inner}}^4 & 0 & 0
  \\
  0 & \frac{1}{6} \left( 3 \left( r_{\text{outer}}^4 - r_{\text{inner}}^4 \right) + L^2 \left( r_{\text{outer}}^2 - r_{\text{inner}}^2 \right) \right) & 0
  \\
  0 & 0 & \frac{1}{6} \left( 3 \left( r_{\text{outer}}^4 - r_{\text{inner}}^4 \right) + L^2 \left( r_{\text{outer}}^2 - r_{\text{inner}}^2 \right) \right)
\end{bmatrix}
\]

64
Appendix J: Matlab 2D Batch File Configuration

Figure 45: 2D Batch File Linearized Plant

Figure 46: 2D Batch File Simulink Diagram
batch2D.m
% Sim Mechanics Simulation Batch Setup for 2D Balancing Pendulum
% Mar 21, 2006 M.R. & T.B.
clc
clear
close all
tic
% Varying parameters for the batch experimentation
% All other parameters should remain constant
JointMass=0.1542;
% Define the desired lengths of the actuated and balanced arms
minAA=110;    % Minimum length of actuated arm (in mm)
maxAA=110;    % Maximum length of actuated arm (in mm)
minBA=200;    % Minimum length of balanced arm (in mm)
maxBA=2000;   % Maximum length of balanced arm (in mm)
interval=100;  % Interval length in mm
% Specify density per unit length of arms
density_act=2740;   % Density for actuated arm
density_bal=2740;   % Density for balanced arm
% Specify radial dimensions of arms
ria=0.00455295;        % Inner diameter of actuated arm
roa=0.00635635;         % Outer diameter of actuated arm
rib=0.00455295;        % Inner diameter of balanced arm
rob=0.00635635;        % Outer diameter of balanced arm
% Specify End Mass Properties
minEM=0;     % minimum desired mass (g)
maxEM=300;    % maximum desired mass (g)
minterval=50;  % interval length in g
%Actuator
Actuator.Mass=.1;    %kg
Actuator.Inertia= eye(3)*0.002;  %kg*m^2
%Damping and Stiction
UpperJoint.Damping=.1;
LowerJoint.Damping=.1;
%Initial Conditions
BalanceArm.Angle=1; %deg
BalanceArm.Velocity=0; %deg/s
DrivenArm.Angle=0; %deg
DrivenArm.Velocity=0; %deg/s

Motor.TorqueSaturation=.083; %N-m

%begin batch code
run=0;

numerror.uncontrollable=0;
numerror.unstablenonlinearity=0;

%Batch for Endmass
for BE = minEM:mininterval:maxEM   %DrivenArm.Length=X:Y
    EndMass.Mass=BE/1000;
    % EndMass.Inertia=eye(3)/EndMass.Mass*.00001; %Define inertia tensor
    height_endmass=0.034;
    radius_endmass=0.0223;
    EndMass.Inertia=[0.5*EndMass.Mass*radius_endmass^2 0 0;0
                     EndMass.Mass*(3*radius_endmass^2+height_endmass^2)/12 0 0
                     0 0 0.0223*EndMass.Mass*(3*radius_endmass^2+height_endmass^2)/12]; %Define inertia tensor
end

%Batch for Actuated Arm
for BAA = minAA:interval:maxAA %DrivenArm.Length=X:Y
    DrivenArm.Length=BAA/1000;
    % DrivenArm.Inertia= [0.00001 0 0;0 0.00001 0; 0 0 0.00001]; %kg*m^2
    DrivenArm.Inertia= 0.5*density_act*pi*(roa^4-ria^4)*0.5+DrivenArm.Length^2*(roa^2-ria^2)/6; %kg*m^2
end

%Batch for Balanced Arm
for BBA=minBA:interval:maxBA
    BalanceArm.Length=BBA/1000;
    % BalanceArm.Inertia= [0.00001 0 0;0 0.00001 0; 0 0 0.00001]; %kg*m^2
BalanceArm.Inertia = density_bal*pi*0.5*[rob^4-rib^4 0 0 0.5*(rob^4-rib^4)+(BalanceArm.Length^2*(rob^2-rib^2))/6 0 0.5*(rob^4-rib^4)+BalanceArm.Length^2*(rob^2-rib^2)/6]; %kg*m^2

% Control Design
uncontrolable=0;
try
[A B C D] = linmod('Plant2DBatch'); % Linearize System
SS = ss(A,B,C,D);
Q=[100 0 0 0
  0 100 0 0
  0 0 100 0
  0 0 0 100];
[K,S,e] = lqr(A,B,Q,1); % Create Controller
Kc=1;
valid=0;
sim('HighLevel2DBatch');
catch
valid=0;
umerror.uncontrollable=numerror.uncontrollable+1;
uncontrolable =1;
end

if (valid)
  run=run+1;
data(run).DrivenArmLength=DrivenArm.Length;
data(run).BalanceArmLength=BalanceArm.Length;
data(run).Endmass=EndMass.Mass;
data(run).UpperJointPosition=yout(:,1)
data(run).LowerJointPosition=yout(:,2)
data(run).time=tout;
else
  if(uncontrolable==0)
    numerror.unstablennonlinearity=numerror.unstablennonlinearity+1;
  end
end

end
end
for x=1:run
    plot(data(x).time, data(x).UpperJointPosition)
    plottitle(x) = {strcat('Upper: ', num2str(data(x).BalanceArmLength), ' Lower: ', num2str(data(x).DrivenArmLength), 'EndMass:', num2str(data(x).Endmass))};
end
try
legend(plottitle);
end
figure(1)
hold all
figure(2)
hold all
for x=1:run
    plot(data(x).time, data(x).LowerJointPosition)
    plottitle(x) = {strcat('Upper: ', num2str(data(x).BalanceArmLength), ' Lower: ', num2str(data(x).DrivenArmLength), 'EndMass:', num2str(data(x).Endmass))};
end
try
legend(plottitle);
end
for x=1:run
    plot(data(x).time, data(x).UpperJointPosition)
    plottitle(x) = {strcat('Upper: ', num2str(data(x).BalanceArmLength), ' Lower: ', num2str(data(x).DrivenArmLength), 'EndMass:', num2str(data(x).Endmass))};
end
try
legend(plottitle);
end
clc
disp('Stats');
disp('Number of successful runs');
disp(run);
disp('Number of Uncontrollable configurations');
disp(numerror.uncontrollable);
disp('Number of Unstable configurations');
disp(numerror.unstablenonlinearity);
disp('Run Time (s)');
disp(toc);
Appendix K: Matlab 3D Batch File Configuration

Figure 47: 3D Batch File Plant
Batch3D_stops.m
% Sim Mechanics Simulation Batch Experimentation for 3D Balancing Pendulum
% Mar 5, 2006 M.R. & T.B.
clec
clear
tic

clc
clear
tic

% Define the desired lengths of the actuated and balanced arms
X_actuate=.01;       % Minimum length of actuated arm (in m)
Y_actuate=1.6;    % Maximum length of actuated arm (in m)
X_balance=0.2;       % Minimum length of balanced arm (in m)
Y_balance=0.2;    % Maximum length of balanced arm (in m)
interval=0.2;       % interval between tests (in m)

% Specify density per unit length of arms
density_act=2740;     % Density for actuated arm
density_bal=2740;     % Density for balanced arm

% Specify End Mass Properties
EndMass.Mass=0.1;      %Define desired mass
height_endmass=0.034;
radius_endmass=0.0223;
EndMass.Inertia=[0.5*EndMass.Mass*radius_endmass^2 0 0;0
EndMass.Mass*(3*radius_endmass^2+height_endmass^2)/12 0;0 0
EndMass.Mass*(3*radius_endmass^2+height_endmass^2)/12]; %Define inertia tensor

% Specify radial dimensions of arms
ria=0.00455295;       % Inner diameter of actuated arm
roa=0.00635635;     % Outer diameter of actuated arm
rib=0.00455295;       % Inner diameter of balanced arm
rob=0.00635635;     % Outer diameter of actuated arm
% System Parameters
Gravity = -9.81;  

% Actuator
Actuator.MassX = 0.01;  
Actuator.InertiaX = eye(3) * 0.00001;  
Actuator.MassY = 0.01;  
Actuator.InertiaY = eye(3) * 0.00001;  

% Damping and Stiction
UpperJoint.DampingX = 0;  
UpperJoint.DampingY = 0;  
Actuator.KineticFrictionX = 0;  
Actuator.StaticFrictionX = 0;  
Actuator.FowardStictionLimitX = 0;  
Actuator.ReverseStictionLimitX = 0;  
Actuator.KineticFrictionY = 0;  
Actuator.StaticFrictionY = 0;  
Actuator.FowardStictionLimitY = 0;  
Actuator.ReverseStictionLimitY = 0;  

% Initial Conditions
BalanceArm.AngleX = 1;  
BalanceArm.AngleY = 1;  
BalanceArm.VelocityX = 0;  
BalanceArm.VelocityY = 0;  
DrivenArm.AngleX = 0;  
DrivenArm.VelocityX = 0;  
DrivenArm.AngleY = 0;  
DrivenArm.VelocityY = 0;  
Motor.TorqueSaturationX = 0.083;  
Motor.TorqueSaturationY = 0.083;  

% Begin Batch Code
run = 0;
numerror.uncontrollable=0;
numerror.unstablenonlinearity=0;

% Batch for Actuated Arm
for DAL=X_actuate:interval:Y_actuate;
    DrivenArm.Length=DAL;
    DrivenArm.Mass=DrivenArm.Length*density_act*pi*(roa^2-ria^2)/4; %kg
    %DrivenArm.Inertia= [0.083 0 0;0 .083 0; 0 0 0]; % (kg*m^2)
    DrivenArm.Inertia= 0.5*density_act*pi*[roa^4-ria^4 0 0 0; 0 (roa^4-
        ria^4)*0.5+DrivenArm.Length^2*(roa^2-ria^2)/6 0 0 0.5*(roa^4-
        ria^4)+DrivenArm.Length^2*(roa^2-ria^2)/6]; %kg*m^2
%Batch for Balanced Arm
for BAL=X_balance:interval:Y_balance;
    BalanceArm.Length=BAL;
    BalanceArm.Mass=BalanceArm.Length*density_bal*pi*(rob^2-rib^2)/4; %kg
    %BalanceArm.Inertia= [0.083 0 0;0 .083 0; 0 0 0]; % (kg*m^2)
    BalanceArm.Inertia= density_bal*pi*0.5*[rob^4-rib^4 0 0 0; 0 0.5*(rob^4-
        rib^4)+(BalanceArm.Length^2*(rob^2-rib^2))/6 0 0 0.5*(rob^4-
        rib^4)+BalanceArm.Length^2*(rob^2-rib^2)/6]; %kg*m^2

%Control Design
uncontrolable=0;
try
    [A B C D] = linmod('Plant3D');   %Linearize System
    SS = ss(A,B,C,D);
    Q=[
        1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
        0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
        0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
        0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
        0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
        0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
        0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
        0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
    ];
    Q=Q*100;
    [K, S, e] = lqr(A,B,Q,[1 0;0 1]);   %Create Controller
    Kx=K(1,:);
    Ky=K(2,:);
    valid=0;
    sim('HighLevel3D_stops');
catch
    valid=0;
end
numerror.uncontrollable=numerror.uncontrollable+1;
uncontrolable =1;
end

if (valid)
    run=run+1;
data(run).DrivenArmLength=DrivenArm.Length;
data(run).BalanceArmLength=BalanceArm.Length;
data(run).Endmass=EndMass.Mass;
data(run).UpperJointPositionX=yout(:,1)
data(run).UpperJointPositionY=yout(:,2)
data(run).LowerJointPositionX=yout(:,3)
data(run).LowerJointPositionY=yout(:,4)

data(run).time=tout;
else
    if(uncontrolable==0)
        numerror.unstablenonlinearity=numerror.unstablenonlinearity+1;
    end
end

figure
plot(tout,yout)
end

figure(1)
hold all
for x=1:run
    plot(data(x).time,data(x).UpperJointPositionX)
    plottitle(x)=strcat('Upper: ',num2str(data(x).BalanceArmLength),',Lower: ',num2str(data(x).DrivenArmLength),',EndMass: ',num2str(data(x).Endmass));
end
try
    legend(plottitle);
end

figure(2)
hold all
for x=1:run
    plot(data(x).time,data(x).UpperJointPositionY)
    plottitle(x)={strcat('Upper: ',num2str(data(x).BalanceArmLength),', Lower: ',num2str(data(x).DrivenArmLength),', EndMass: ',num2str(data(x).Endmass))};
end
try
    legend(plottitle);
end
title('Balancing Arm Y Angle (Deg) in Reference to Actuating Arm');
xlabel('Time (sec)');
ylabel('Displacement (Deg)');
hold off

figure(3)
hold all
for x=1:run
    plot(data(x).time,data(x).LowerJointPositionX)
    plottitle(x)={strcat('Upper: ',num2str(data(x).BalanceArmLength),', Lower: ',num2str(data(x).DrivenArmLength),', EndMass: ',num2str(data(x).Endmass))};
end
try
    legend(plottitle);
end
title('Actuator Arm X Angle (Deg) in Reference to Ground');
xlabel('Time (sec)');
ylabel('Displacement (Deg)');
hold off

figure(4)
hold all
for x=1:run
    plot(data(x).time,data(x).LowerJointPositionY)
    plottitle(x)={strcat('Upper: ',num2str(data(x).BalanceArmLength),', Lower: ',num2str(data(x).DrivenArmLength),', EndMass: ',num2str(data(x).Endmass))};
end
try
    legend(plottitle);
end
title('Actuator Arm Y Angle (Deg) in Reference to Ground');
xlabel('Time (sec)');
ylabel('Displacement (Deg)');
hold off
clc
disp('Stats');
disp('Number of succesful runs');
disp(run);
disp('Number of Uncontrolable configurations');
disp(numerror.uncontrollable);
disp('Number of Unstable configurations');
disp(numerror.unstablennonlinearity);
disp('Run Time (s)');
disp(toc);
Appendix L: Batch File Results

2D Configuration

When varying the balancing arm from 0.2 m to 2 m while keeping the actuating arm at a constant 0.11 m and the end weight at a mass of 0.0001 kg, there are no unsuccessful runs. The printout for the batch statistics reads:

Stats
Number of successful runs
19

Number of Uncontrollable configurations
0

Number of Unstable configurations
0

Run Time (s)
64.3280

The plots of the actuating and balancing arm positions may be seen in Figures 49 and 50 below.
Figure 49: Batch File Results for Actuating Arm for End Weight of 0.0001 kg

Figure 50: Batch File Results for Balancing Arm for End Weight of 0.0001 kg
These results can be compared to when the end weight is increased to 0.1 kg. The number of successful runs is decreased to 8 and the number of unstable configurations is increased to 11. Figures 51 and 52 below show the batch file results for the successful runs. Insight is gained into the effect of the end weight on the system – the higher the end weight the shorter the balancing arm must be in order to maintain stability.
When setup is changed so that the length of the balancing arm is kept at 0.2 m of length and the end weight mass is varied from 0 to 0.3 kg, the figures change. Figures 53 and 54 show the results for the new batch file.

Figure 53: Batch File Results for Actuating Arm for Balancing Arm of 0.2 m

Figure 54: Batch File Results for Balancing Arm for Balancing Arm of 0.2 m
Whereas there were 6 successful runs when the balancing arm was 0.2 m long, there is only one when the arm is 2 m long. Again, this shows for a specific length of the balancing arm, there is a limit to the amount of weight that can be added to the end. The longer the balancing arm, the quicker the end weight comes to the saturation of the weight the system can handle. The batch result for the longer balancing arm may be seen in Figures 55 and 56 below.

![Actuator Arm Angle (Deg) in Reference to Ground](image1.png)

**Figure 55: Batch File Result for Actuating Arm for Balancing Arm of 2 m**

![Balancing Arm Angle (Deg) in Reference to Actuating Arm](image2.png)

**Figure 56: Batch File Result for Actuating Arm for Balancing Arm of 2 m**

When the actuating arm length is changed to 0.22 m and the balancing arm is changed back to 0.2 m for end weight mass varying from 0 to 0.3 kg, the number of successful runs becomes 5 out of 7. The stability and robustness of the system is decreased when the
actuating arm length is increased, as can be seen by comparing Figures 57 and 58 to Figures 53 and 54.

Figure 57: Batch File Results for Actuating Arm for Actuating Arm of 0.22 m

Figure 58: Batch File Results for Balancing Arm for Actuating Arm of 0.22 m
**3D Configuration**

In the three dimensional system, keeping the end weight constant at 0.1 kg while varying the balancing arm length from 0.2 m to 2 m and the actuating arm length from 0.1 m to 1 m for motor torque saturation set at 10 N·m, the statistics for the system become:

**Stats**

- **Number of successful runs**: 50
- **Number of Uncontrollable configurations**: 0
- **Number of Unstable configurations**: 0
- **Run Time (s)**: 43.5780

The results can be seen in Figures 59 through 60. The system is easier to control when the balancing arm is longer. This fact can be seen in the fact that the actuating and balancing arm angles have a lower maximum offset angle before settling back to an angle of zero when the balancing arm is longer.

![Figure 59: Batch File Results for Actuating Arm in x-Axis for T = 10 N·m](image-url)
### Figure 60: Batch File Results for Actuating Arm in y-Axis for $T = 10 \text{ N•m}$

<table>
<thead>
<tr>
<th>Actuator Arm Y Angle (Deg) in Reference to Ground</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upper:0.2  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:0.4  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:0.6  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:0.8  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:1    Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:1.2  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:1.4  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:1.6  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:1.8  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:2    Lower:0.1  EndMass:0.1</td>
</tr>
</tbody>
</table>

### Figure 61: Batch File Results for Balancing Arm in x-Axis for $T = 10 \text{ N•m}$

<table>
<thead>
<tr>
<th>Balancing Arm X Angle (Deg) in Reference to Actuating Arm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upper:0.2  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:0.4  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:0.6  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:0.8  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:1    Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:1.2  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:1.4  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:1.6  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:1.8  Lower:0.1  EndMass:0.1</td>
</tr>
<tr>
<td>Upper:2    Lower:0.1  EndMass:0.1</td>
</tr>
</tbody>
</table>

85
Figure 62: Batch File Results for Balancing Arm in y-Axis for T = 10 N·m

When the torque saturation for both axes are set to the actual value of 0.083 N·m while keeping all other values constant, fewer combinations are stable. Now there are 17 out a total of 50 configurations that are stable. No matter how small the end weight mass is, there are always 17 stable configurations out of a possible 50. This shows exactly what is being seen in experimentation. The motors are able to control the 3D system for a very small periods of time and then go unstable. In order to control even the smallest of the masses, motors with higher torque saturation are necessary.
Figure 63: Batch File Results for Actuating Arm in x-Axis for T = 0.083 N•m

Figure 64: Batch File Results for Actuating Arm in y-Axis for T = 0.083 N•m
To see how the actuating arm affects the system, the mass of the end weight is kept constant at 0.1 kg and the balancing arm length is maintained at 0.4 m for a torque saturation 0.083 N·m, while the actuating arm length is varied from 0.01 m to 4 m. There are a total of two successful runs out of a total twenty combinations. The trend shows that a longer actuating arm will be easier to control than a shorter length, as can be seen in Figures 67 through 70.
Figure 67: Batch File Results for Actuating Arm in x-Axis for Constant Balancing Arm

Figure 68: Batch File Results for Actuating Arm in y-Axis for Constant Balancing Arm
Figure 69: Batch File Results for Balancing Arm in x-Axis for Constant Balancing Arm

Figure 70: Batch File Results for Balancing Arm in y-Axis for Constant Balancing Arm
Appendix M: Labview VI Summary

FPGA: Main FPGA Enhanced Velocity Estimation.vi & Main FPGA Enhanced Velocity Estimation 3D.vi: This VI acts as a pass through from the Real-Time Controller to the FPGA.

Figure 71: LabView Variables
Encoder Interface FPGA enhanced vel estimation.vi: This VI takes care of the Quadrature Decoding of the encoder position, velocity estimation, and velocity averaging.

Figure 72: Encoder Interface FPGA enhanced vel estimation.vi:
Current Amp Interface FPGA.vi: This is a simple VI that updates the output of the Analog Output Terminals on the cRIO

Figure 73: Current Amp Interface FPGA.vi

Joystick Interface FPGA.vi: This is a simple VI that is used to read two potentiometers in order to manually control the actuator arm.

Figure 74: Joystick Interface FPGA.vi
Real-Time Control VIs:
There are three main templates for the Real-Time Control VI. All of the Real-Time Control configurations use some variant of the following VIs
Figure 75: 2D Balancing VI
Figure 76: Swing Up Vi
Figure 77: 3D VI
Appendix N: SolidWorks Drawings

See next page
UNLESS OTHERWISE SPECIFIED:

DIMENSIONS ARE IN INCHES
TOLERANCES:
FRACTIONAL ±
ANGULAR: MACH ±
BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±

INTERPRET GEOMETRIC TOLERANCING PER:

MATERIAL

FINISH

DO NOT SCALE DRAWING

SCALE: 1:1

WEIGHT:

SHEET 1 OF 7

TITLE: High Resolution Encoder

<table>
<thead>
<tr>
<th>NAME</th>
<th>DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BL</td>
<td>3/28/06</td>
</tr>
</tbody>
</table>

DRAWN
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:

SIZE DWG. NO. REV
A

SCALE: 1:1 WEIGHT:

SHEET 1 OF 7
UNLESS OTHERWISE SPECIFIED:

DIMENSIONS ARE IN INCHES

TOLERANCES:

FRACTIONAL ±
ANGULAR: MACH+ BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±

INTERPRET GEOMETRIC TOLERANCING PER:

MATERIAL: Aluminum
FINISH

TITLE: Actuated Arm Clamp

NAME DATE

DRAWN BL 3/28/06
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:

SCALE: 1:1
WEIGHT:
SHEET 2 OF 7
Balance Arm Holder

Dimensions are in inches.

Tolerances:
- Fractional: ±
- Angular: Machined Bend ±
- Two place decimal: ±
- Three place decimal: ±

Material: Aluminum

Finished

Scale: 1:1

Weight:

Sheet 3 of 7
Balance Arm Encoder Mount

DIMENSIONS ARE IN INCHES
TO LEANANCES:
FRACTIONAL:
ANGULAR: MACH BEND:
TWO PLACE DECIMAL:
TWO PLACE DECIMAL:
THREE PLACE DECIMAL:
INTERPRET GEOMETRIC
TOLERANCING PER:
MATERIAL: Aluminum
FINISH

UNLESS OTHERWISE SPECIFIED:

DRAWN
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:

NAME DATE
BL 3/28/06

TITLE:

SIZE DWG. NO. REV
A SHEET 4 OF 7

SCALE: 1:1 WEIGHT:
UNLESS OTHERWISE SPECIFIED:

DIMENSIONS ARE IN INCHES

TOLERANCES:

FRACTIONAL ±

ANGULAR: MACH, BEND ±

TWO PLACE DECIMAL ±

THREE PLACE DECIMAL ±

INTERPRET GEOMETRIC TOLERANCING PER:

MATERIAL: aluminum

FINISH

TITLE: UJOINT

DRAWN: BL 3/28/06

CHECKED

ENG APPR.

MFG APPR.

Q.A.

COMMENTS:

SHEET 5 OF 7
<table>
<thead>
<tr>
<th>ITEM NO.</th>
<th>PART NUMBER</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>UJOINT</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Balance Arm Encoder Mount</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>High Resolution Encoder</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>dowl</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>Balance Arm Holder</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>Balance Weight</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>Balance Arm</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>Actuated Arm</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>Actuated Arm Clamp</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>Weight</td>
<td>1</td>
</tr>
</tbody>
</table>

**3 D Configuration**

- **Name**: BL
- **Date**: 3/28/06

Dimensions are in inches, tolerances: fractional: ± angular; plus or minus; bend: ± two place decimal; ± three place decimal.

Interpret geometric tolerancing per: material: finish.
ITEM NO. | Description     | QTY.
----------|-----------------|------
1         | 3/8 dowel       | 1
2         | C-Clamp         | 2
3         | Locking Clamp   | 1
4         | Acuated Arm     | 1
5         | U-Joint         | 1
6         | High Resolution Encoder | 1
7         | longdowl        | 1
8         | C-Clamp         | 1
9         | Balancing Arm   | 1
10        | Weight          | 1
11        | Balance Weight  | 1
Appendix O: Wiring Chart

Definitions:

- **EncMX ChA**: Shaft Encoder X direction Channel A
- **EncMX ChB**: Shaft Encoder X direction Channel B
- **EncMY ChA**: Shaft Encoder Y direction Channel A
- **EncMY ChB**: Shaft Encoder Y direction Channel B
- **EncAX ChA**: Actuated Arm Encoder X direction Channel A
- **EncAX ChB**: Actuated Arm Encoder X direction Channel B
- **EncAY ChA**: Actuated Arm Encoder Y direction Channel A
- **EncAY ChB**: Actuated Arm Encoder Y direction Channel B
- **EncBX ChA**: Balanced Arm Encoder X direction Channel A
- **EncBX ChB**: Balanced Arm Encoder X direction Channel B
- **EncBY ChA**: Balanced Arm Encoder Y direction Channel A
- **EncBY ChB**: Balanced Arm Encoder Y direction Channel B

CAT5 Wire Assignments:

- **Orange**: NC
- **Orange/White**: NC
- **Blue**: Shaft Encoder Channel A
- **Blue/White**: Shaft Encoder Channel B
- **Green**: Actuated Arm Encoder direction Channel A
- **Green/White**: Actuated Arm Encoder direction Channel B
- **Brown**: Power
- **Brown/White**: GND

Pin Assignments:

cRIO 9401

<table>
<thead>
<tr>
<th>Pin</th>
<th>Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>DI00</td>
<td>14</td>
</tr>
<tr>
<td>DI01</td>
<td>16</td>
</tr>
<tr>
<td>DI02</td>
<td>17</td>
</tr>
<tr>
<td>DI03</td>
<td>19</td>
</tr>
<tr>
<td>DI04</td>
<td>20</td>
</tr>
<tr>
<td>DI05</td>
<td>22</td>
</tr>
<tr>
<td>DI06</td>
<td>23</td>
</tr>
<tr>
<td>DI07</td>
<td>25</td>
</tr>
<tr>
<td>DI0A</td>
<td>1</td>
</tr>
<tr>
<td>DI0B</td>
<td>9</td>
</tr>
<tr>
<td>DI1A</td>
<td>2</td>
</tr>
<tr>
<td>DI1B</td>
<td>10</td>
</tr>
<tr>
<td>DI2A</td>
<td>3</td>
</tr>
<tr>
<td>DI2B</td>
<td>11</td>
</tr>
<tr>
<td>DI2A</td>
<td>6</td>
</tr>
<tr>
<td>DI2B</td>
<td>13</td>
</tr>
</tbody>
</table>
## Appendix P: Physical System Parameters

<table>
<thead>
<tr>
<th>Component</th>
<th>Length (in)</th>
<th>Weight (lb)</th>
<th>Inner Diameter (in)</th>
<th>Outer Diameter (in)</th>
<th>Width (in)</th>
<th>Height (in)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actuating Arm</td>
<td>8.75</td>
<td>0.08</td>
<td>0.3585</td>
<td>0.5005</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Balancing Arm</td>
<td>23.3125</td>
<td>0.2</td>
<td>0.3585</td>
<td>0.5005</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>End Weight</td>
<td>N/A</td>
<td>0.3</td>
<td>N/A</td>
<td>1.7595</td>
<td>N/A</td>
<td>1.338</td>
</tr>
<tr>
<td>Counter Weight</td>
<td>2.494</td>
<td>0.6</td>
<td>N/A</td>
<td>N/A</td>
<td>1.009</td>
<td>0.802</td>
</tr>
<tr>
<td>Brass Plate</td>
<td>3.001</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>1.044</td>
<td>0.146</td>
</tr>
<tr>
<td>Actuating Arm</td>
<td>N/A</td>
<td>0.94</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Assembly</td>
<td>N/A</td>
<td>0.86</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Balancing Arm</td>
<td>N/A</td>
<td>0.86</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Assembly</td>
<td>N/A</td>
<td>0.86</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Component</th>
<th>Gear Ratio</th>
<th>Voltage Conversion (V/step)</th>
<th>Current Conversion (mA/V)</th>
<th>Torque Conversion (N·m/A)</th>
<th>Maximum Torque (N·m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pan Motor</td>
<td>4.36</td>
<td>3.23E-04</td>
<td>103.50</td>
<td>4.36E-02</td>
<td>0.064</td>
</tr>
<tr>
<td>Tilt Motor</td>
<td>3.00</td>
<td>3.23E-04</td>
<td>111.85</td>
<td>4.36E-02</td>
<td>0.047</td>
</tr>
</tbody>
</table>