Spring 4-2020

Dynamic and Control of Air-Bearing Spacecraft Simulator

Jacob Joseph Korczyk
Embry-Riddle Aeronautical University

DYNAMICS AND CONTROL OF AIR-BEARING SPACECRAFT SIMULATOR

By

Jacob Joseph Korczyk

A Thesis Submitted to the Faculty of Embry-Riddle Aeronautical University

In Partial Fulfillment of the Requirements for the Degree of

Master of Science in Aerospace Engineering

April 2020

Embry-Riddle Aeronautical University

Daytona Beach, Florida

DYNAMICS AND CONTROL OF AIR-BEARING SPACECRAFT SIMULATOR

By

Jacob Joseph Korczyk

This Thesis was prepared under the direction of the candidate's Thesis Committee Chair, Dr. Troy Henderson, Department of Aerospace Engineering, and has been approved by the members of the Thesis Committee. It was submitted to the Office of the Senior Vice President for Academic Affairs and Provost, and was accepted in the partial fulfillment of the requirements for the Degree of Master of Science in Aerospace Engineering.

THESIS COMMITTEE

Troy A. Henderson
Digitally signed by Troy A. Henderson
Date: 2020.04.05 13:45:20 -04'00'

Chairman, Dr. Troy Henderson

Richard J. Prazenica
Digitally signed by Richard J. Prazenica
Date: 2020.04.05 13:53:59 -04'00'

Member, Dr. Richard Prazenica

Morad Nazari
Digitally signed by Morad Nazari
Date: 2020.04.05 14:06:40 -04'00'

Member, Dr. Morad Nazari

Dr. Magdy S. Attia
Digitally signed by Dr. Magdy S. Attia
Date: 2020.04.05 15:03:58 -04'00'

Graduate Program Coordinator, Dr. Magdy Attia

Date

Maj Dean Mirmirani
Digitally signed by Maj Dean Mirmirani
DN: cn=Maj Dean Mirmirani, o=Embry-Riddle Aeronautical University, ou, email=mirmiram@erau.edu, c=US
Date: 2020.04.06 08:51:22 -04'00'

Dean of the College of Engineering, Dr. Maj Mirmirani

Date

Christopher Grant
Digitally signed by Christopher Grant
Date: 2020.04.09 07:21:21 -04'00'

Associate Provost of Academic Support, Dr. Christopher Grant

Date

ACKNOWLEDGEMENTS

I would like to thank Dr. Bogdan Udrea for providing me the opportunity to work on a project that, at the time, seemed intimidating, but has opened my eyes to a passion I never knew I had. Dr. Troy Henderson, Dr. Richard Prazenica, and Dr. Morad Nazari, thank you for all of the time you have invested in my development as a young engineer learning my way around spacecraft dynamics and control, if I could repay you for all of time I have taken from you I truly would. Thank you, Dr. Francisco Franquiz and Daniel Posada for your unconditional support, without you I would have been lost from day one. Dr. Aaron Altman, thank you for pushing me to pursue my passion; There is not a doubt in my mind that without your support and guidance I would have been a lost soul form the beginning, and would have never discovered a passion for dynamics and control. To my grandparents, Babcia, Dziadek and Papa, dziękuję za pokazanie mi wartości edukacji, i zawsze wymaganie najlepszych od mną, zawsze będziesz ze mną. Dalton, I can never thank you enough for being the best brother and friend a little boy could ever wish for, I know Papa and Unlce Tony are proud. I don't say it enough but I love you. Mom and Dad, I could never put into words my love and gratitude. The example you have set for Dalton and I, along with the lessons we have learned from you have been invaluable. The support and sacrifice you have made for Dalton and I to follow our dreams has been nothing short of divine.

ABSTRACT

An air bearing is being designed as a spacecraft rotational motion simulator, featuring the Sawyer Robot and its control box. The objective is to maneuver the robot as desired, performing operations specific to on-orbit servicing operations while maintaining stability of the system. Before the control can be designed, the dynamics of the platform and the robot must be modeled. The dynamics of the robot can be derived utilizing a Newton-Euler recursive approach. By beginning with a simple pendulum, then adding links (degrees of freedom) to more closely resemble the Sawyer arm, the equations of motion for the robot can be developed. After the equations of motion for the robot are derived, the next step is to model the dynamics of the entire platform, which adds three more degrees of freedom to the system. The Newton-Euler recursive approach is not compatible with the system with the addition of the spherical joint; therefore a new approach is adopted to model the attitude dynamics in terms of Euler angles. Once the dynamics are modeled, control design can take place, where an incremental non-linear dynamic inversion controller is designed to reject the disturbances of the robot performing its maneuver, while also actuating the platform to a desired attitude.

TABLE OF CONTENTS

LIST OF FIGURES

Figure Page

LIST OF TABLES

ABBREVIATIONS

ISS	International Space Station
MSS	Mobile Servicing System
JAXA	Japan Aerospace Exploration Agency
JEMRMS	Japanese Experiment Module Robotic Manipulator System
ESA	European Space Agency
ERA	European Robotic Arm
EVA	Extravehicular Activity
SCARA	Selective Compliance Assembly Robot Arm
NASA	National Aeronautics and Space Administration
D-H	Denavait-Hartemburg
NLDI	Non-Linear Dynamic Inversion
INLDI	Incramental Non-Linear Dynamic Inversion
PI	Proportional Integral
CMG	Control Moment Gyro

NOMENCLATURE

α_{DH}	DH parameter for rotation about x-axis
β	Standard 3-2-1 rotation matrix
$\mathbf{I}$	Identity Matrix
Φ	Gravity matrix used for Newton-Euler Recursive Method
ϕ	Roll
ψ	Yaw
θ	Pitch
θ_{DH}	DH parameter for rotation about z-axis
$\vec{\alpha}$	Angular acceleration vector
$\vec{\omega}$	Angular velocity vector
$\vec{\tau}$	Torque vector
$\vec{\Theta}$	Vector or Euler angles
$\vec{F}$	Force vector
$\vec{H}$	Angular momentum vector
$\vec{M}$	Moment vector
$\vec{r}$	Position vector
$\vec{v}$	Linear velocity vector
C	Centrifugal matrix used for Newton-Euler Recursive Method
d_{DH}	DH parameter for distance z-axis

g	Earth gravitational constant
I	Inertia Matrix
M	Mass matrix used for Newton-Euler Recursive Method
m	Mass
q	Rotation about the z-axis
r_{DH}	DH parameter for distance along common normal
R_x^y	Rotation matrix from arbitrary frame x to y

1. Introduction

As humans endeavor to conduct space missions of increasing complexity, both in Earth orbit and beyond, operations such as asteroid mining, space debris removal, and on-orbit servicing will have to be performed routinely. To model the complexities of the contact dynamics involved in these procedures, a robotics test-bed is designed consisting of two interacting robotic arms. A robotic arm is mounted on an air-bearing platform, acting as the servicing satellite, while a second arm is stationary, acting as the client spacecraft, as shown in Figure 1.1.

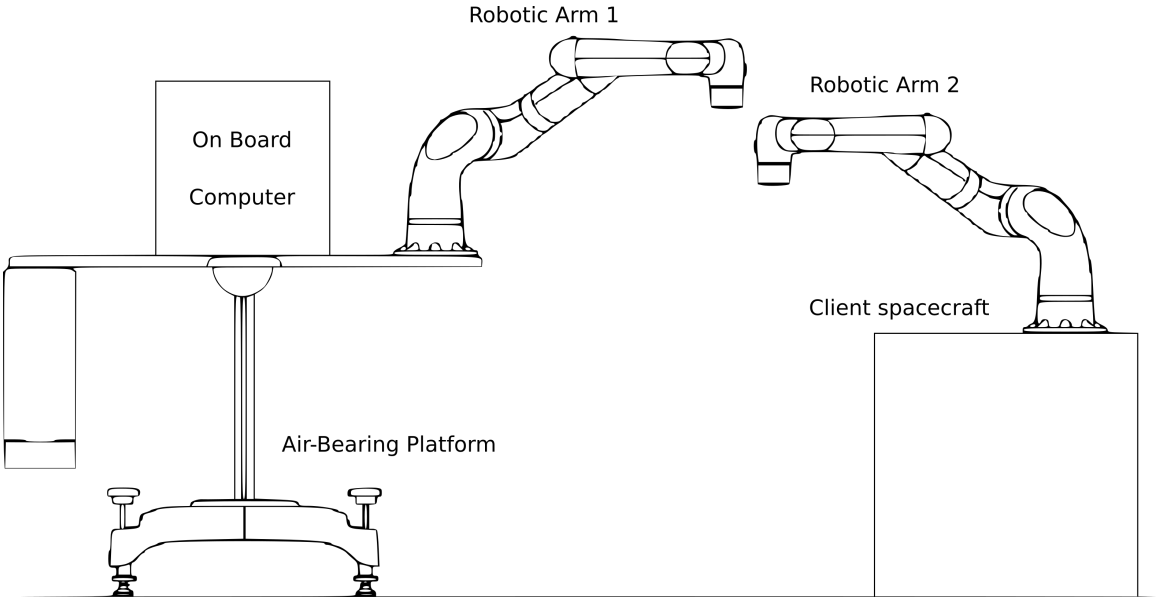


Figure 1.1 Space Robot System

Different examples of robotic systems in space can be found in the literature (Currie & Peacock, 2002) Currie & Peacock (2002). Three of them reside on the International Space Station (ISS): The Canadian Mobile Servicing System (MSS), which is used to perform maintenance to the station and provide aid for approaching maneuvers. The Japanese (JAXA) Experiment Module Remote Manipulator System (JEMRMS), which is used for experimentation and payload maneuvers. And finally, the European Space Agency (ESA) European Robotic Arm (ERA), used for maintenance and EVA support.

These examples are all implemented on systems with considerably more mass and

Figure 1.2 Sawyer Robot. From "Rethink Robotics Leads in Research and Education with Open Source SDK," by Rethink Robotics, 2016 (http://www.prnewswire.com/news-releases/rethink-robotics-leads-in-research-and-education-with-open-source-sdk-300339747.html)

volume than the manipulator. However, with the current trend in miniaturization, smaller, more agile spacecraft will need to carry out similar missions, comparable to what has been done by Antonello in 2018 Antonello et al. (2018) using thrusters, and Schwartz in 2004 Schwartz (2004) for decentralized control. Because of the small size of this satellite, when the robot arm is manipulated in orbit, as there are no external influences, the attitude of the satellite will change as a consequence of this motion. This effect is undesirable, as it can cause unwanted maneuvers that could have drastic effects on communication, power generation, as well as compromise the mission. Using the air-bearing test-bed, these dynamics can be studied, and their effects better understood. Using the dynamics, a controller can be developed to control the system or reject disturbances, utilizing different methods.

The dynamics are simulated using the Simscape^{TM} Multibody environment The MathWorks, Inc. (n.d.) and validated by comparing the predicted torque of the recursive method to the torque output from the Simscape model. Both models are

compared to the output from the real Sawyer robot, Figure 1.2, and are deemed accurate, without regard for frictional and damping effects that exist in the physical robot. After validation of the recursive method, the model is expanded to include the air bearing table. Due to incompatibility of the spherical joint with the recursive method, the dynamics of the system are re-developed using a Newtonian approach, starting with one link and then building up to all seven links, with the additional three degrees of freedom of the spherical joint. This model determines the amount of torque on the spherical air-bearing due to the robot as well as its control box. Again the method is compared to a simulation in Simscape and validated, leading to the design of a control system for the pseudo spacecraft.

The objective of this research is to develop the rotational dynamics of the spacecraft in order to design a controller to actuate the system to a desired state of Euler angles as well as reject any disturbances. Many controllers are able to achieve this result. For a time-invariant system, meaning that the physical system does not change with time a simple state feedback controller may be able successfully complete this task. While a more complex controller will be necessary for the time varying systems such as systems where fuel is exiting the system, or a large mass moving within the system. However, this servicing satellite features a robotic arm that causes a change in mass distribution (inertia) as time progresses, causing the system to vary with time as well as providing highly non-linear characteristics.

More robust controllers are required for such applications. The type of control that is used is incremental, non-linear dynamic inversion (INLDI). This type of controller utilizes two loops, a proportional-integral controller for the inner loop and a proportional controller for the outer loop. This dynamic inversion controller allows for the disturbance and fully determined dynamics to be fed forward to the controller and negated, all the while actuating the satellite to a specific attitude.

The controller is tuned to respond to several maneuvers each producing a favorable response. A critically damped response that uses a realistic amount of

torque, with a settling time of over a minute in most cases, is generally used. Next, noise introduced to the system is seen to cause large issues with controller, limiting the ability to converge to the targeted values, instead entering a limit cycle, oscillating constantly around the targeted attitude. This sets up perfectly for non-linear estimation, not covered in this work, which succeeds in developing the attitude dynamics as well as a controller to command the system.

2. Literature Review

Space systems feature many diverse and complex subsystems, such as power, propulsion, and control. For the system as a whole to be successful, the subsystems must first be tested to ensure safety and reliability. Most, if not all, are tested on the ground before flight. The control system in particular involves testing the system in its entirety on Earth, as if it were in space. To test sensors, controllers, actuators or, even control laws the friction-less, torque-free environment of space must be replicated on Earth.

2.1. Existing Air-Bearing Systems

One way to simulate the environment of space is using an air-bearing simulator. While air-bearing simulators do not emulate the micro-gravity of space, they are able to provide a friction-less environment to allow motion in several degrees of freedom. There are many types of simulators for their different applications, although they are typically separated by the type of motion they facilitate: translational or rotational.

Translational simulators are often planar, similar to that of an air-hockey table, where procedures such as docking or robotic proximity operations can be tested and simulated. An air-hockey table supplies air through the table to "float" the desired object, the puck. Instead, a planar air-bearing system supplies air through the body being supported; in the air-hockey example it would be as if the puck was supplying its own ability to glide effortlessly along the table. The air is supplied through a permeable surface creating a film of air between the supported body and planar surface Rybus & Seweryn (2016). As long as air is constantly supplied to the system, the film supports the body rendering the system "friction-less" because the body never comes in contact with the surfaces. This friction-less environment allows for planar motion and one rotational degree of freedom, about the vertical axis, resulting in three degrees of freedom.

Many professional, governmental and educational institutions use planar air-bearing simulators, of course ranging in degree of complexity depending on

application and funding. In 1967, the North American Rockwell Group built a system capable of supporting a 200-lb test vehicle with the option to feature a human pilot (Fornoff, 1967)Fornoff (1967). An air-bearing test facility at Stanford's Aerospace Robotics Laboratory features multiple air-bearings used to explore the use of robots for on-orbit operations such as construction and to what degree humans need to be involved (Schwartz et al., 2003) Schwartz et al. (2003). A single robotic arm at the University of Victoria is featured on a planar air-bearing, examining the optimal joint trajectory to reduce strain during maneuvers (Pond & Sharf, 1999) Pond & Sharf (1999). Even international organizations such as Japanese corporations are using a planar air-bearing to develop controllers for robots to replace EVA's (Toda et al., 1992) Yoshitugu Toda (1992) as shown in, Figure 2.1. In this system, two 3 deree of freedom SCARA robots are featured on the system that is supported by N_2 gas being expelled out of eight thrusters. This system has a mass of about 150 kg and without the manipulators is 700 mm^3.

Rotational simulators, similar to planar systems, use air to create a film that "floats" the structure, allowing the system to rotate freely in three axes. The difference between planar systems and rotational systems is that rotational air-bearing simulators use a hemisphere and a cup to support the rotating system.

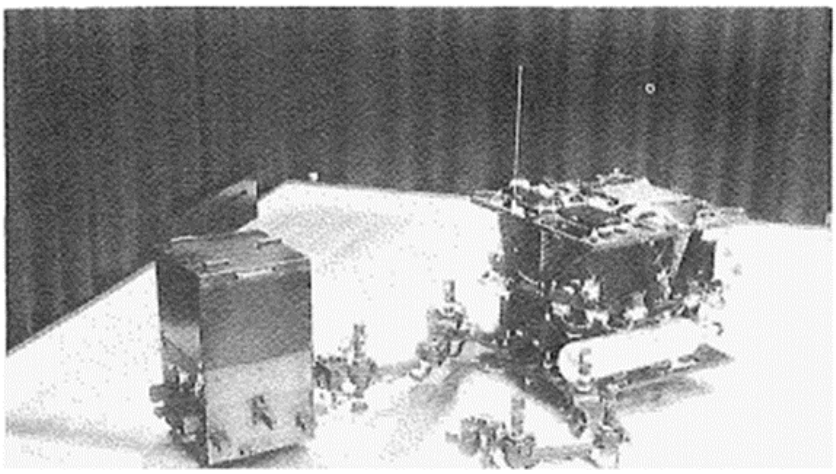


Figure 2.1 Japanese Planar Air-Bearing Simulator. (Toda et al., 1992, p. 33)

Using pressurized air, there is a film created between the cup and the semi-sphere, allowing the semi-sphere to "float", leading to relatively unfettered rotational ability based upon the components of the system above it. While ideally the use of a sphere would allow the air-bearing unrestricted movement, the rotation about the z-axis is often unlimited, however the pitch and roll axes are not. Rotation about the x and y-axes is restricted so that the structure on top of the sphere will not come in contact with the stand holding the air-bearing, causing possible damage to either the stand, the air-bearing, or the structure. Adding complexity, rotational systems must rotate about a point of rotation; however if this does not coincide with the center of gravity, the imbalance will cause a change in angular rate as well as attitude, and the platform remains horizontal when at rest (Guo et al., 2017).

Many institutions interested in space applications use spherical air-bearing simulators for the purposes of attitude control because of the ability to simulate practically unfettered rotational dynamics. Typically, these systems use an umbrella configuration; this arrangement sets the system above the bearing, allowing for these systems to rotate more freely. In 1975, an air-bearing was used to determine the center of mass of a system at Stanford University. A U.S. Army Ballistic Missile Agency that later merged with NASA in 1960 created the spherical air-bearing, shown in Figure 2.2, that was used to investigate bearing imperfections and aided the research into hydrodynamic air-bearings. This configuration enabled the system to achieve a range of $\pm 120^{\circ}$ about the pitch and roll axes (Haeussermann & Kennel, 1960) Haeussermann & Kennel (1960).

Air-bearing systems, both planar and rotational, aid in testing of components, equipment development, as well as control law design. Institutions around the world have different methods of designing these simulators, each for a different purpose. Some use air, others use multiple gasses. Some support massive loads, while others shoulder only a couple kilograms. Some are used for educational applications while others are used for classified missions. The wide range of air-bearings and their

applications provides an immense amount of background and guidance to develop a spherical air-bearing spacecraft simulator featuring a robotic arm to simulate on-orbit operations.

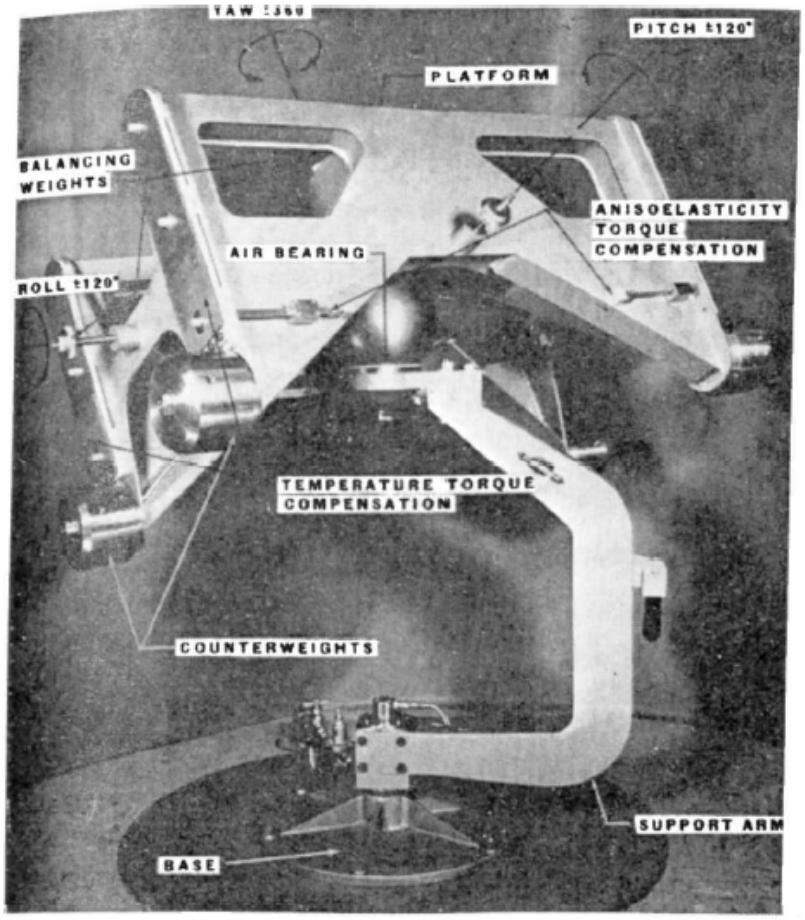


Figure 2.2 U.S. Army Ballistic Missile Agency, Spherical Air-Bearing. From "A Satellite Motion Simulator," by W. Haeussermann and H. Kennel, 1960, *Astronautics, Vol. 5* p. 22, 23, 90, 91. Coppyright 1960 by the American Rocket Society

2.2. D-H Parameters

To develop an air-bearing spacecraft simulator, dynamics need to be modeled to ensure that the design is feasible and realistic, and the motion is understood. The first step in developing these dynamics is defining reference frames. Reference frames are essential to the description of the kinematics: determining where each body is located in space and how these bodies move. For robotic systems, reference frames are necessary to describe the location of each joint and link segment leading to the

dynamical description. In robotics, there are several methods to determine these different reference frames. For this application, D-H parameters were utilized. This particular method uses four characteristics to locate reference frames in succession from the first, in this case, the inertial frame, as shown in Figure 2.3.

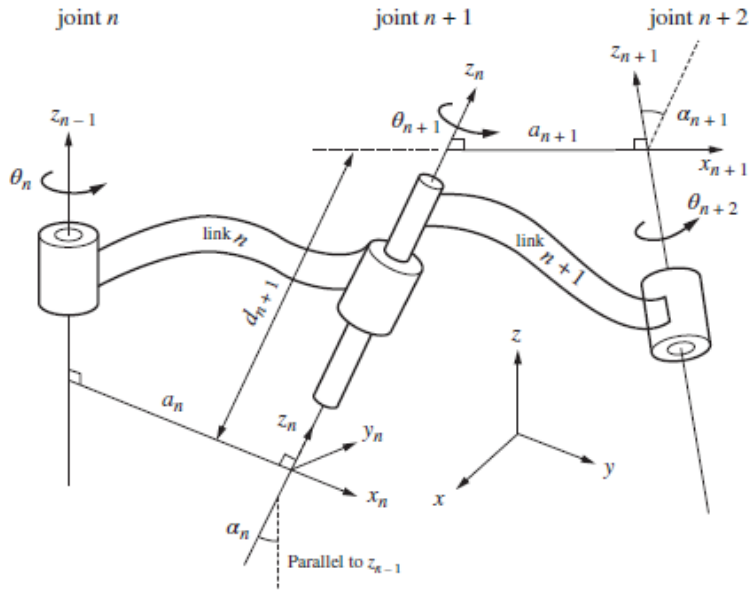


Figure 2.3 D-H Parameters Graphical Representation. Adapted From *Dynamics and Control of Robotic Manipulators with Contact and Friction* (p. 16) by S. Liu and G. S. Chen, 2019, Hoboken, NJ: John Wiley & Sons, Inc. Copyright 2019 by John Wiley & Sons, Inc.

The first attribute is the distance, d_{DH}, along the z-axis of the $i-1$ frame to the common normal. The common normal is the vector perpendicular to the z-axis of the $i-1$ frame and the i^{th} frame. The second attribute is θ_{DH}, the rotation about the z-axis from the $i-1$ frame to the i^{th} frame. For the purposes of this development, this quantity is the generalized coordinate denoted as q. The third attribute is the distance along the common normal, r_{DH}, from the $i-1$ frame to the i^{th} frame. The final attribute is α_{DH}, which is the rotation about the new x-axis to align the new z-axis with the axis of rotation. *Table* 2.1 provides the description of the Denavait-Hartenburg, D-H, parameters used to describe the reference frames of the Sawyer robot.

While the D-H parameters describe the reference frames in the correct manner,

the 2^{nd} frame has an extra 90 - degree rotation about the y - axis that is not captured through the D-H parameter description. This extra rotation leads to an extra rotation of the link from the expected orientation based upon the D-H parameters; however it is necessary to emulate the Sawyer robot.

Table 2.1

D-H Parameter for Sawyer Robot

Link	$\boldsymbol{d_{DH}}$ **[meters]**	$\boldsymbol{\alpha_{DH}}$ **[radians]**	$\boldsymbol{\theta_{DH}}$	$\boldsymbol{r_{DH}}$ **[meters]**
Joint 1	0.2370	$-\frac{\pi}{2}$	q_1	-0.081
Joint 2	0.1925	$\frac{\pi}{2}$	q_2	0
Joint 3	0.4000	$-\frac{\pi}{2}$	q_3	0
Joint 4	-0.1685	$\frac{\pi}{2}$	q_4	0
Joint 5	0.4000	$-\frac{\pi}{2}$	q_5	0
Joint 6	0.1363	$\frac{\pi}{2}$	q_6	0
Joint 7	0.1100	0	q_7	$8.08e^{-7}$

2.3. Introduction to Lie Algebra

Equations of motion describe the motion of individual bodies in a dynamical system. For rigid body motion, there is typically one vector equation per body. Therefor six separate scalar equations can be formed for each body, describing the motion of a particular body both translationally and rotationally. The Sawyer robot has seven links, meaning there would normally be seven complete vector equations of motion, each dependent on the motion of the previous links. This dynamic coupling leads to rather long equations that are difficult to reproduce and accurately test. A more compact form of these equations, developed by Park, is known as the Newton-Euler Recursive algorithm. This method relies on a form of mathematics called Lie Algebra (Murray, 1994; Park, 1991) Murray (1994) Park & Bobrow (1994) to structure the dynamics in a way that becomes compact and easy to work with.

Many of the kinematic quantities are known in their Lie groups, also known as differentiable manifolds, which are smooth and continuous. However, to synthesize the numerical equations of motion, these quantities must be mapped to their more useful

Lie algebras through the use of the adjoint representation. The linear mapping creates 6×6 matrices that are used in the recursive algorithm to kinematically describe the motion, leading to the dynamics of the rigid links. A mapping is composed of the configuration, $X = \left(\Theta, \vec{b}\right) \in SE(3)$, of each body for every instance of time:

$$\begin{bmatrix} \Theta & \vec{b} \\ 0 & 1 \end{bmatrix}$$

Where Θ is the orientation, in the form of a rotation matrix, and $\vec{b}$ is the position vector relating separate frames. To determine the transpose of Ad_X, the dual operator Equation (2.1), Ad_X^* Equation (2.2), is used:

$$Ad_X(x) = \begin{bmatrix} \Theta & 0 \\ \vec{b} \times \Theta & \Theta \end{bmatrix} \tag{2.1}$$

$$Ad_X^*(x) = \begin{bmatrix} \Theta^T & \Theta^T \left[\vec{b}^{\times}\right]^T \\ 0 & \Theta^T \end{bmatrix} \tag{2.2}$$

Using the adjoint and dual adjoint mappings, generalized forces and velocities of bodies in their respective body frames can be transformed, and therefore mapped into frames that could be more useful to arithmetic development. To emulate a mathematical cross product, the Lie bracket, $ad_X(y) = [x, y]$, is used, which is developed using the adjoint representations:

$$ad_x = \begin{bmatrix} \vec{\omega}^{\times} & 0 \\ \vec{v}^{\times} & \vec{\omega}^{\times} \end{bmatrix} \tag{2.3}$$

$$ad_x^* = \begin{bmatrix} -\vec{\omega}^{\times} & -\vec{v}^{\times} \\ 0 & -\vec{\omega}^{\times} \end{bmatrix} \tag{2.4}$$

A more detailed description of the adjoint representations and their development

can be found in Ploen's dissertation, *Geometric Algorithms for the Dynamics and Control of Multibody Systems* (Ploen, 1998), as well as in, *A Lie group Formulation of Robot Dynamics* (Park et al., 1995), and in, *A Mathematical Introduction to Robotic Manipulation* (Murray, 1994).

Before dynamic equations can be developed, kinematic relationships must be accounted for and described to accurately model any system. To demonstrate the kinematic motion between frames, the product of exponential formula can be used. Frame i can be described relative to the $i-1$ frame by using the matrix exponential, Equation (2.5):

$$M_i e^{P_i q_i} \tag{2.5}$$

Where M_i is the configuration matrix of frame i, q_i is the rotation q for link i, and P_i is the vector S_i described later to be a 6×1 vector composed of the unit vector along the axis of rotation and the zero vector. To determine the n^{th} frame related to the base frame, the product of exponentials can be used:

$$f(q_1, \ldots, q_n) = M_1 e^{P_1 q_1} \ldots M_n e^{P_n q_n} \tag{2.6}$$

Here the matrix exponentials can be calculated analytically:

$$exp\left(\begin{bmatrix} \vec{\omega}^{\times} & \vec{v} \\ 0 & 0 \end{bmatrix} \theta\right) = \begin{bmatrix} exp\left(\vec{\omega}^{\times}\theta\right) & \vec{b} \\ 0 & 1 \end{bmatrix} \tag{2.7}$$

where:

$$\vec{b} = \left(\theta I + (1 - \cos\theta)\,\vec{\omega}^{\times} + (\theta - \sin\theta)\left[\vec{\omega}^{\times}\right]^2\right)\vec{v} \tag{2.8}$$

$$exp\left(\vec{\omega}^{\times}\theta\right) = I + \vec{\omega}^{\times}\sin\theta + (1 - \cos\theta)\,\vec{\omega}^{\times 2} \tag{2.9}$$

Notice, the angular velocity unit vector $\vec{\omega}$ is turned into the skew symmetric matrix, also known as the cross product matrix, denoted by the $\times$ superscript. in this development I denotes the identity matrix while θ, in this description developed by both Park and Murray, is the generalized coordinate in the recursive algorithm known as q. While developed analytically, the matrix exponential for the purposes of this numerical validation is generated numerically, for calculation speed.

2.4. Newton-Euler Recursive Method

Newton, Euler, and Lagrange each have different ways of developing and representing the equations of motion. The Lagrangian approach is based upon the total energy of the system:

$$L = T(q, \dot{q}) - V(q) \tag{2.10}$$

Where T is the kinetic energy and V is the potential energy. Using the Lagrange equation:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_k} - \frac{\partial L}{\partial q_k} = Q_k \tag{2.11}$$

Where Q_k is the torque of body k, leads to a compact form:

$$\tau = M(q)\ddot{q} + C(q, \dot{q})\dot{q} + \phi(q) \tag{2.12}$$

Where M is the mass matrix, C is the centrifugal matrix, ϕ represents the gravitational terms, and τ is the applied torque. Similarly, the equations of motion can be developed using a combination of Newton and Euler's equation. Taking the sum of the total forces and the total moments acting on the system:

$$\sum \vec{F} = m\vec{a} \tag{2.13}$$

$$\sum \vec{M} = I\dot{\vec{\omega}} + \vec{\omega} \times I\vec{\omega} + \vec{r} \times m\vec{a} \tag{2.14}$$

Both approaches lead to the same Equation (2.12), but the formulation is much different. The Lagrangian approach lends itself to an analytical formulation while a numerical demonstration is better suited for the Newton-Euler recursive method. Because the Sawyer robot can generate useful data that can be recorded and utilized, a numerical approach is more beneficial. The recursive method easily allows for the comprehensive comparison of torques to those generated by the robot.

To begin with the recursive method, the mass, centrifugal, and gravity matrices must first be broken up into the quantities that create them, based upon physical and kinematic characteristics of the Sawyer robot:

$$M(q) = S^T G^T J G S \tag{2.15}$$

$$C(q, \dot{q}) = S^T G^T (J G ad_{S_{\dot{q}}} \Gamma + ad_V^* J) G S \tag{2.16}$$

$$\Phi(q) = S^T G^T J G P_0 \dot{V}_0 \tag{2.17}$$

A required matrix in these calculations is S which is 42×7. This matrix is composed of smaller vectors, s_i, that are composed of the zero vector and the unit vector along the axis of rotation. This group exists in $SE(3)$, and because $SE(3)$ is isomorphic to $\mathbb{R}^6$ (Nazari et al., 2018) Nazari et al. (2018) it can be represented as a 6×1 vector; thus it sets up nicely for the recursive algorithm that is developed from

primarily 6×6 matrices.

$$s_i = \begin{bmatrix} \vec{\omega}_i \\ 0 \end{bmatrix} \rightarrow s_i = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.18}$$

$$S = \begin{bmatrix} s_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & s_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & s_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & s_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & s_5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & s_6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & s_7 \end{bmatrix} \tag{2.19}$$

The J matrix is the inertia matrix, composed of the masses as well as the distance, $\vec{r}_i$, each mass is centered away from the body frame of joint i. Notice, the distance $\vec{r}_i$ is a vector but is turned into a skew symmetric matrix as denoted by the $\times$ superscript. The I_i is the inertia matrix of link i about the center of mass while the $\mathbf{I}$ matrix is the 3×3 identity matrix.

$$J = \begin{bmatrix} I_i - m_i \left[\vec{r}_i^{\times}\right] \left[\vec{r}_i^{\times}\right]^T & m_i \left[\vec{r}_i^{\times}\right] \\ -m_i \left[\vec{r}_i^{\times}\right] & m_i \mathbf{I} \end{bmatrix} \tag{2.20}$$

$$J = \begin{bmatrix} j_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & j_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & j_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & j_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & j_5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & j_6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & j_7 \end{bmatrix} \tag{2.21}$$

The adjoint representation is utilized to develop the remaining matrices. The $ad_{S\dot{q}}$ and ad_v^* matrices represent the linear and angular velocities. $ad_{S\dot{q}}$ is the adjoint representation of the magnitude of the angular velocity multiplied by the unit vector in the direction of rotation, while ad_v^* is the dual adjoint of the velocity vector.

$$ad_{S\dot{q}} \begin{bmatrix} -\left[(S\dot{q}_i)^{\times}\right] & 0 \\ 0 & -\left[(S\dot{q}_i)^{\times}\right] \end{bmatrix} \tag{2.22}$$

$$ad_v^* = \begin{bmatrix} -\left[\vec{\omega}_i^{\times}\right] & -\left[\vec{v}_i^{\times}\right] \\ 0 & -\left[\vec{\omega}_i^{\times}\right] \end{bmatrix} \tag{2.23}$$

$$ad_{S\dot{q}} = \begin{bmatrix} ad_{S\dot{q}_1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & ad_{S\dot{q}_2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & ad_{S\dot{q}_3} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & ad_{S\dot{q}_4} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & ad_{S\dot{q}_5} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & ad_{S\dot{q}_6} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & ad_{S\dot{q}_7} \end{bmatrix} \tag{2.24}$$

$$ad_v^* = \begin{bmatrix} ad_{v_1}^* & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & ad_{v_2}^* & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & ad_{v_3}^* & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & ad_{v_4}^* & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & ad_{v_5}^* & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & ad_{v_6}^* & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & ad_{v_7}^* \end{bmatrix} \tag{2.25}$$

The interesting connection between frames is all managed by the G matrix. The G matrix is based upon the adjoined representation of the kinematic dependencies between frames. $f_{i-1,i}$ is the configuration of link i with relation to frame $i-1$, determined by the matrix exponential that can be determined both numerically and analytically. The analytical expression is shown by Equation (2.26):

$$f_{i-1,i} = M_i e^{[\omega_i^\times]q_i} \rightarrow f_{i-1,i} = \begin{bmatrix} \Theta_i & \vec{b}_i \\ 0 & 1 \end{bmatrix} \tag{2.26}$$

Where M_i, not to be confused with the mass matrix M, is the configuration matrix of joint i:

$$M_i = \begin{bmatrix} R_i^{i-1} & \vec{r}_i \\ 0 & 1 \end{bmatrix} \tag{2.27}$$

Here $\vec{r}_i$ is the distance and R_i^{i-1} is the rotation matrix both with the relationship from reference frame $i-1$ to i, yielding a 4×4 matrix. Once the configuration matrix is generated for each link up to the final link, $f_{n-1,n}$ the Γ and G matrices can be generated. Γ is an off-diagonal matrix composed of the adjoined representations of each of the $f_{i-1,i}^{-1}$ matrices, $Ad_{f_{i-1,i}^{-1}}$.

$$f_{i-1,i}^{-1} = \begin{bmatrix} \Theta_i^T & \Theta_i^T \vec{b}_i \\ 0 & 1 \end{bmatrix} \tag{2.28}$$

$$Ad_{f_{i-1,i}^{-1}} = \begin{bmatrix} \Theta_i^T & 0 \\ \vec{b}_i \Theta_i^T & \Theta_i^T \end{bmatrix} \tag{2.29}$$

$$\Gamma = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Ad_{f_{1,2}^{-1}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & Ad_{f_{2,3}^{-1}} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & Ad_{f_{3,4}^{-1}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & Ad_{f_{4,5}^{-1}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & Ad_{f_{5,6}^{-1}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & Ad_{f_{6,7}^{-1}} & 0 \end{bmatrix} \tag{2.30}$$

The G matrix can then be generated from the relation:

$$G = (I - \Gamma)^{-1} \tag{2.31}$$

The final two matrices $\dot{V}_0$ and P_0 are only used in ϕ, which accounts for the gravity terms. $\dot{V}_0$ is the gravity vector and P_0 accounts for the kinematic relationship between the first and inertial frames.

$$\dot{V}_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ g \end{pmatrix} \tag{2.32}$$

$$P_0 = \begin{bmatrix} Ad_{f_{0,1}^{-1}} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.33}$$

These separate matrices are primarily block diagonal matrices made up of smaller matrices dependent on physical and kinematic relationships. The dimension of these matrices depends on the number of links in the robot. The robot has seven links; therefore, the size of many of these larger matrices is 42×42, although many of the elements are zero as a result of the block diagonal nature of these large matrices. The smaller matrices are each 6×6 due to the manner in which adjoined representations are assembled. However, the matrix S is 42×7 due to the nature of the smaller vectors 6×1 making up the large S matrix. Similarly, $\dot{V}_0$ is also a vector, being the 6×1 gravity vector of the system. P_0 is a 42×6 matrix relating the first and inertial frames using the same adjoint representation present in the Γ matrix.

Lie Algebra provided a window into robot dynamics that was effective in illustrating how to model the complex system. Using this basis for development, platform dynamics can be developed to further explore dynamics of the system to include the robot, platform and control box. After the development of the platform dynamics, a controller can be developed to stabilize and control the system.

3. Dynamic Buildup of Air-Bearing Platform

Test-bed environments similar to the one proposed in Figure 1.1 are essential to continue the innovation process and improve space exploration, as the interest in servicing missions has increased. By testing here on earth, the dangers of on-orbit servicing can be evaluated and assessed. Different configurations can be designed to test different conditions and simulate a multitude of missions. Although, before this test-bed can aid in the testing and design of different missions the pseudo spacecraft needs to be dynamically modeled as well as physically constructed.

3.1. Introduction to the System

The system is composed of three major systems: the air-bearing, the spacecraft and the robotic system. The air-bearing is of spherical design, being composed of a semi-sphere that rests atop a cup in which pressurized air can be ran through to create a thin film to float the system. While a functional architecture has been generated, shown in Figure 3.1, the spacecraft only features the platform, that is home to a single Sawyer robot and its control box. The robotic system is composed of two Sawyer robots, one mounted on the platform and the other fixed in space, acting as a client spacecraft that will be serviced. This research focuses on the dynamics and control of the spacecraft system that features three main components: the robotic arm, its control box and the platform itself.

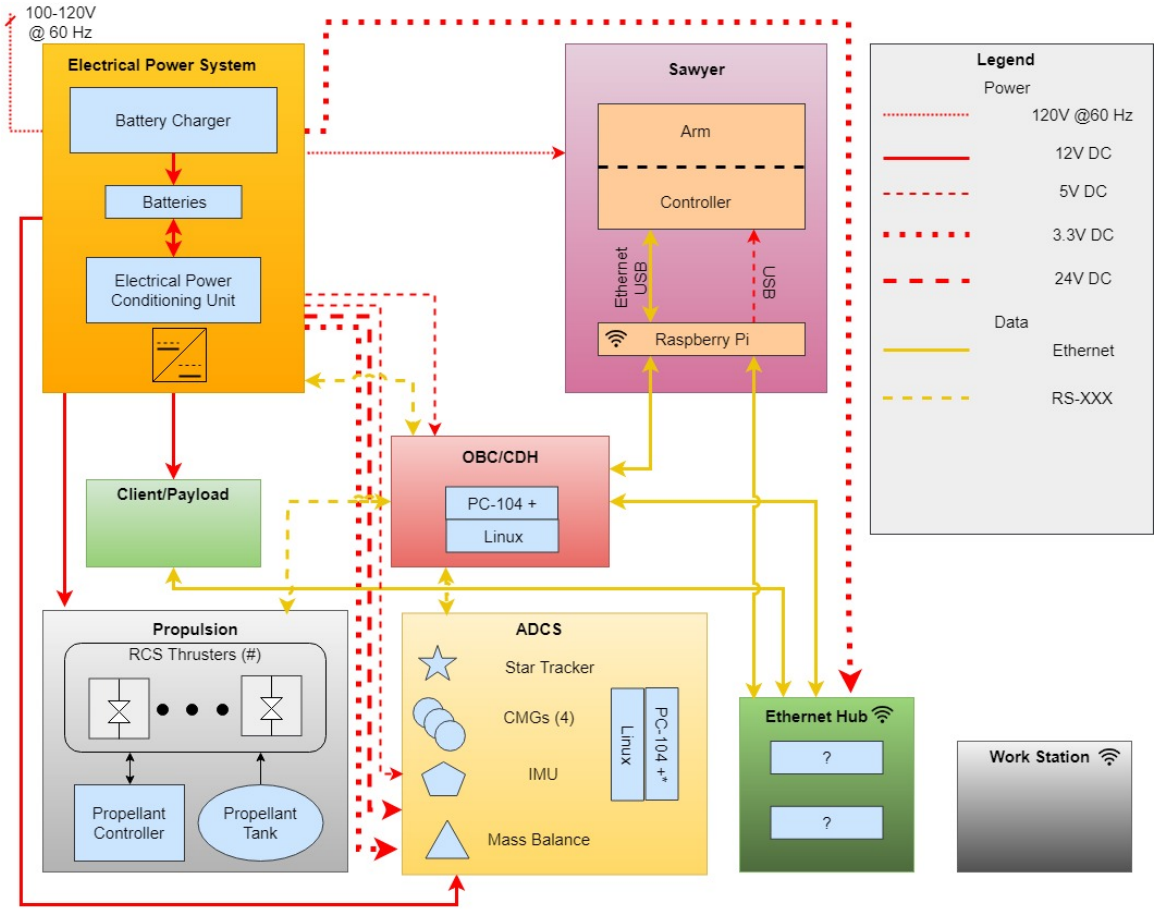
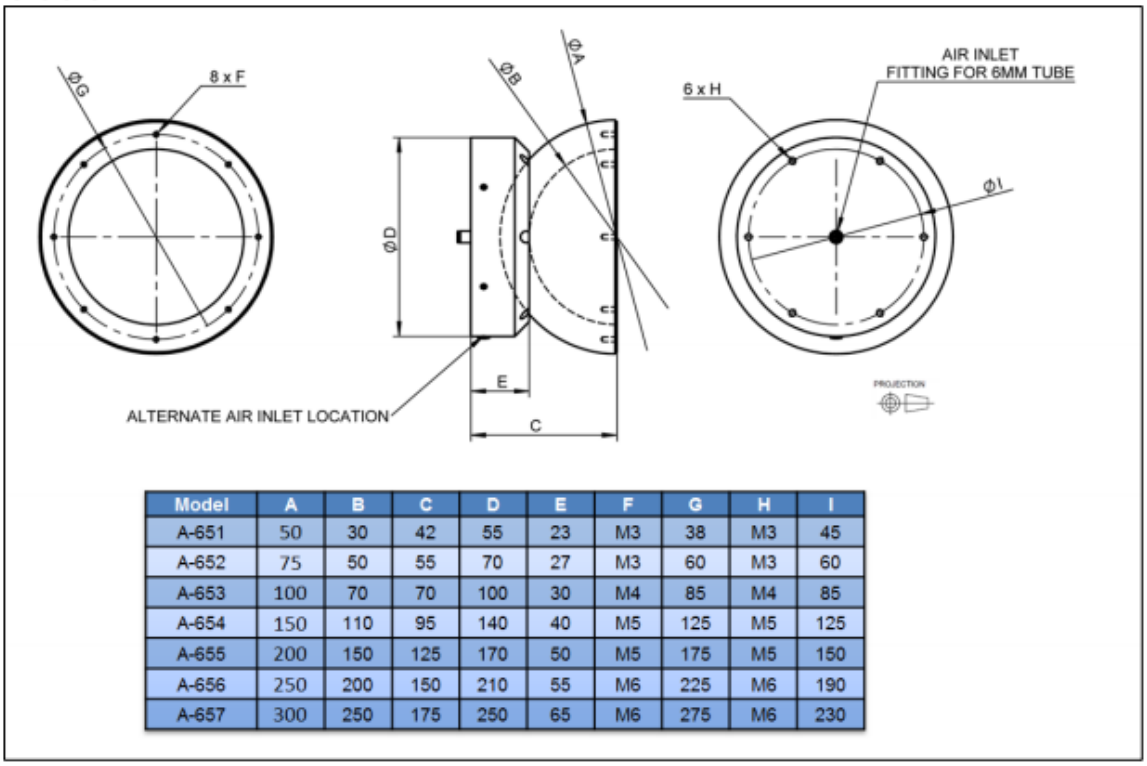


Figure 3.1 Functional Architecture of Air-Bearing Spacecraft Simulator

3.1.1. Air-Bearing System

To simulate on-orbit operations for space conditions frictionless conditions must be mimicked. With the use of an air-bearing system similar to that described in the previous chapter, a spacecraft can be built above it to study the dynamics of these close proximity maneuvers. Using the PIglide HB, model A-655, by Physik Instrumente, allows for a platform to be set-up on top of a spherical air-bearing Figure 3.2.

There are two parts to the air-bearing, the semi-sphere and the cup. There are minute openings in the cup that air is pumped through at 80 psi, providing a small film of air between itself and the semi-sphere. The air-bearing has a load capacity of 265 kg (PIglide HB Hemispherical Air Bearings User Manual, 2016) meaning the



Model	A	B	C	D	E	F	G	H	I
A-651	50	30	42	55	23	M3	38	M3	45
A-652	75	50	55	70	27	M3	60	M3	60
A-653	100	70	70	100	30	M4	85	M4	85
A-654	150	110	95	140	40	M5	125	M5	125
A-655	200	150	125	170	50	M5	175	M5	150
A-656	250	200	150	210	55	M6	225	M6	190
A-657	300	250	175	250	65	M6	275	M6	230

Figure 3.2 Air-Bearing Model A-655 (*Piglide HB Hemispherical Air Bearings User Manual*)

"satellite" built on top of the air bearing will need to weigh less than this capacity for the air-bearing to create a frictionless environment created by the film of air. This frictionless situation between the semi-sphere and the bowl allows for the semi-sphere and the structure on top to rotate freely along the z-axis and rotate up to 45 degrees along the x and y-axes. However, the structure of the system above the air-bearing is limited to less than 40 degrees of rotation about the y-axis due to the configuration, to ensure that the control box of the Sawyer does not come in contact with the stand and damage either component.

3.1.2. Platform

While the air bearing is a vital component of the test-bed, the platform is the structure capable of supporting the "satellite" on top of it. A stable, well designed structure is necessary to ensure the components attached to it remain in their secured positions while in motion. Made out of 80/20® Aluminum, the structure features a mass of 22.5 kg. 80/20 Aluminum was chosen because of the cross sections; they

provide rigidity while also maintaining a lightweight system. The goal when designing the platform was to ensure that the weight of both the robot and the control box would not deflect the platform more than two tenths of a millimeter in the z-direction. Several cross sections were tested using Femap, a finite element software by SIEMENS. These cross sections were 30×60 mm, lightweight but allowed too much deflection, 0.534 mm; the 45×90 mm, rigid although heavy with a mass of 39.12 kg; and finally 40×80 mm lite, rigid enough to make it worthwhile to build a lighter system 23.37 kg, all the while deflecting only 0.253 mm. It deflected more that the desired 0.2 mm, however, considered acceptable as 0.2 mm was an approximate estimation. The shape of the platform added more rigidity to the system. Initially the design was three long beams similar to Figure 3.3, but instead of slanted beams providing more rigidity with the triangle shape, horizontal beams were in place. The design change to a triangular support structure provided more rigidity to the system while still facilitating a relatively lightweight structure.

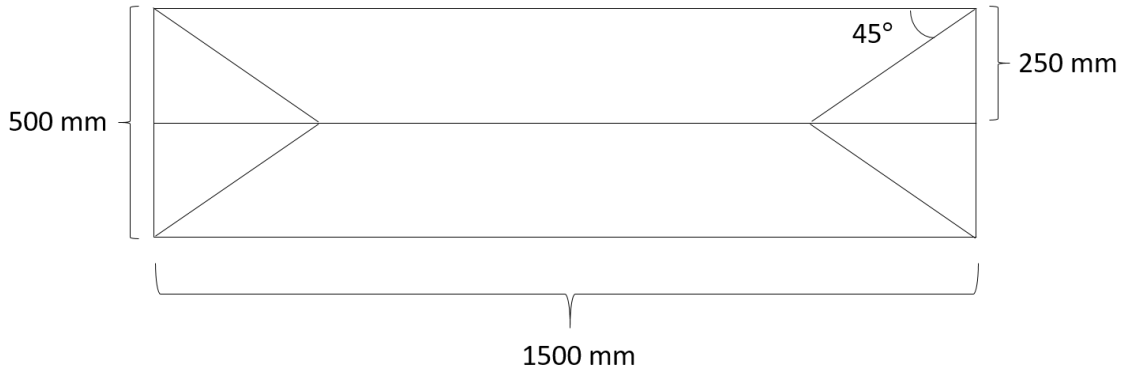


Figure 3.3 Platform Design

3.1.3. Robotic System

The payload of this "satellite" is the Sawyer and its brain, the control box. Like all satellites, the center of mass as well as the magnitude of the weight are of paramount importance. To ensure that the system has a degree of balance, the two components of this robotic system reside on opposite ends of the platform. The seven

revolute joint robot on one end, with a mass of 19 kg, and the control box, exhibiting a mass of 20.4 kg on the other side.

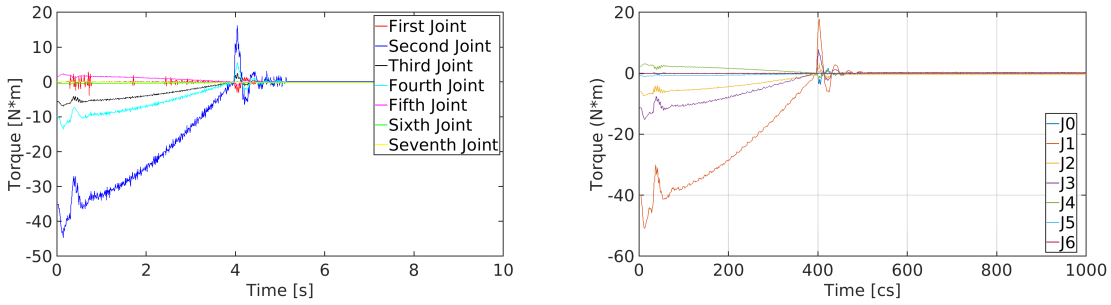
The Sawyer robot belongs to a class of robots known as cobots, more commonly known as collaborative robots. Cobots are often seen working with their human counterparts in industrial settings and as a result have a variety of safety features (Sawyer Safety Overview, 2017). Some of these safety features, such as contact detection and backdrivable joints, lend themselves to not only industrial settings but also educational and research situations. The backdrivable joints allow for the robot to be repositioned manually regardless of its powered condition, allowing the robot to be easily overpowered by a researcher if necessary. The robot is equipped with "Series Elastic Actuators" (Sawyer Safety Overview, 2017) that measure torque at each joint and can sense contact and halt motion in the event of hazardous collision. In addition to safety features, the robot lends itself to academic research because of its ability to run from the proprietary software known as Intera®, or utilized with its python-based functions in a software of the user's choice. For its purpose in the following spacecraft simulator, the robot has been conditioned to respond to inputs from MATLAB®, allowing for control and data collection from the actuators and sensors.

3.1.4. Goal of the System

In order to successfully simulate on-orbit maneuvers, the test bed needs to accurately mimic a frictionless environment of outer-space. With the use of a spherical air-bearing, the frictionless environment can be replicated and a precise model of the attitude dynamics of the spacecraft can be created. After dynamic modeling, a controller can be designed that has the ability to orient the spacecraft at will. For purposes of this research, the controller was designed to reject the disturbances created by the Sawyer robot while maintaining the ability to orient the spacecraft to a desired attitude.

3.2. System Dynamics

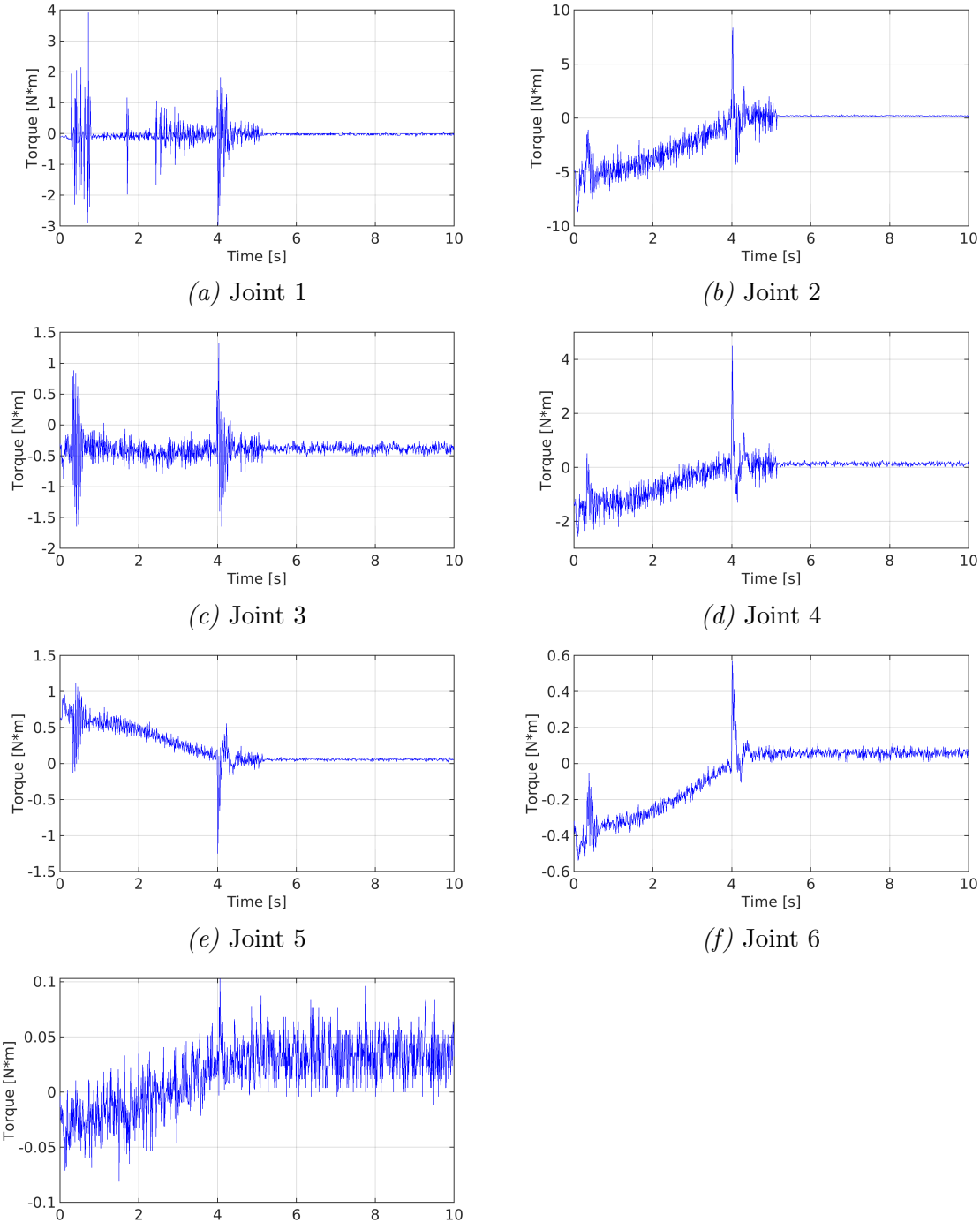
The objective of stabilizing the spacecraft during operation entails dynamic modeling as a precursor to control design. In the previous chapter a Newton-Euler recursive method was used to model the dynamics of the robot. The recursive method yields torques at each joint paving the way for a program to be written in MATLAB that allowed for the prediction of torque at each joint based on a prescribed set of joint motion characteristics such as position, velocity and acceleration. Having the physical robot and a program generating torque profiles at each joint allowed for the comparison between the two torque generation methods, as shown in Figure 3.4b and Figure 3.4a.



(a) Simulated Torque Profile *(b)* Sawyer Torque Profile

Figure 3.4 Torque Profile Sawyer Simple Movement

Provided the same kinematic conditions, moving the robot from the rest position, fully extended upward in the z-direction, similar to the a human arm extend fully upwards in the air, to its zero position, fully extended outward in the x-direction, similar to a human arm extended fully outwards in front of the individual; both methods generate torque profiles. The relative error between the models can be generated by subtracting the Sawyer data from the recursive model output to quickly validate the recursive model, Figure 3.5a to Figure 3.5g. There is still a degree of error, one reason is due to the omission of frictional and damping effects that exist within the real robot, however are not modeled with the recursive method.



(a) Joint 1

(b) Joint 2

(c) Joint 3

(d) Joint 4

(e) Joint 5

(f) Joint 6

(g) Joint 7

Figure 3.5 Simulation Relative Error Sawyer Simple Movement

The largest error is seen in Figure 3.5b. Being the first pitching joint, joint two must resist and overpower the gravitational effects of each subsequent link. Moving such a large amount of mass means that it will undoubtedly result in the highest

torque, predicted by Figure 3.4a and shown to be true by Sawyer Figure 3.4b. While the profiles are similar, a possibility for the difference in magnitude could be how the gravity is modeled using the lie group algebra of each joint.

With the robot dynamically modeled the platform would simply add three degrees of freedom with the addition of the spherical joint. The spherical joint can be represented as three coincident revolute joints that each act about a different respective axis. However, applying this notion to the recursive method presented by Ploen causes values to be lost that need to be dynamically represented to accurately describe torque about the three axes. While the recursive method is convenient for single degree of freedom joints, it can become problematic for joints with additional degrees of freedom. Instead of using the recursive method, using Newtonian and Eulerian mechanics makes way for a new dynamic model dependent on the joint characteristics capable of accurately modeling the torque about the spherical joint.

3.2.1. Analytical Approach

The spacecraft simulator is composed of nine different rigid bodies, the control box, platform and each link of the Sawyer robot. In space, the dynamic motion of these rigid bodies would be the most important source of torque generation. However, because the system is designed on Earth the gravitational force will also generate a torque on the system pulling each body toward the center of the Earth. To dynamically model the system, the spherical air bearing, point O of Figure 3.6, will be locked in place as the controller will seek to keep the spacecraft at a constant position. It should also be noted that for simplicity the center of mass of the spacecraft is assumed to be about the center of rotation, point O.

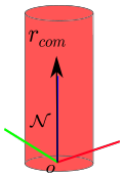


Figure 3.6 Single Link on Top of Air-Bearing

3.2.2. Simplest Case

Moving forward, instead of beginning with the fully realized dynamic problem which could lead to countless problems and errors, the simplest form of the problem can be first considered. This situation is configured as one link with the center of mass along the center of rotation as well as this link being located on top of the center of rotation of the platform. By making these simplifications a basic form of the torque equation can be easily derived and verified. However even before dynamics are considered, the static position of the control box will cause a torque about the spherical joint that can be easily determined through a quick sum of moments about point O.

$$\sum \vec{M}_O = \vec{r} \times \vec{F} \tag{3.1}$$

Using the cross product, the torque in the inertial frame can be generated from the weight of the control box about the spherical joint. This equation will be used to also generate the torque produced from the weight of each link of the robot at every point in time. Now that the static portion of the model has been generated, the one link situation can be studied with the gravity turned off, meaning the weight of the robot will provide no source of torque. Consider one link mounted on top of the spherical air bearing, with the center of mass along the axis of rotation, as shown in Figure 3.6. The angular momentum $\vec{H}$ is known to be the matrix product of inertia multiplied by the angular velocity vector.

$$^B\vec{H}_O = {}^BI_O\vec{\omega} \tag{3.2}$$

The inertia matrix here is about the point O. Initially the inertia of the link is about the center of mass, which is along the axis of rotation, displaced in only the z-direction from the point of rotation. Because the inertia of the link is known about an arbitrary point, this inertia will need to be determined about the point of rotation,

point O. To determine the inertia about the point of rotation the parallel axis theorem Equation (3.3) must be employed.

$$ {}^{B}I_{O} = {}^{B}I_{CoM} - m\left[{}^{B}\vec{r}^{\times}\right]\left[{}^{B}\vec{r}^{\times}\right]^{T} \tag{3.3} $$

where the vector $\vec{r}$ is the position vector from the current point to the point of interest. In this case it is the vector from the link center of mass to point O. After expanding the inertia term, the angular momentum equation now appears as:

$$ {}^{B}\vec{H}_{O} = {}^{B}I_{CoM}\vec{\omega} - m\left[{}^{B}\vec{r}^{\times}\right]\left[{}^{B}\vec{r}^{\times}\right]^{T}\vec{\omega} \tag{3.4} $$

Additionally, notice the angular momentum is expressed in the body frame of the spherical air-bearing. To produce the torque, the derivative of the angular momentum vector must be taken with respect to the inertial frame. To get the torque of the system into the inertial frame, the transport theorem must be applied to the system Equation (3.5).

$$ {}^{N}\dot{\vec{H}}_{O} = {}^{B}\dot{\vec{H}}_{O} + \vec{\omega} \times {}^{B}\vec{H}_{O} \tag{3.5} $$

By expanding terms, it can be seen that for this system the inertia does not change with respect to the body frame and because the center of mass is along the axis of rotation, the velocity vector goes to zero. As a result, the body frame derivative of angular momentum is the well-known equation of torque, Equation (3.6).

$$ {}^{B}\dot{\vec{H}}_{O} = {}^{B}I_{O}\dot{\vec{\omega}} \longrightarrow {}^{B}\dot{\vec{H}}_{O} = I_{O}\vec{\alpha} \tag{3.6} $$

Although, because this system is taken from the body frame to the inertial frame the cross product of the angular velocity vector with the angular momentum must be

added onto the system yielding Equation (3.7).

$$ {}^{N}\dot{\vec{H}}_O = {}^{B}I_O\dot{\vec{\omega}} + \vec{\omega} \times {}^{B}I_O\vec{\omega} \tag{3.7} $$

This is the torque about the spherical air-bearing, point O, due to one link, directly on top of the joint.

3.2.3. Increasing Model Fidelity

Next, moving the center of mass of the link to its actual location within the body and adding the displacement from the spherical joint to the mounted location on the platform Equation (3.8), Figure 3.7 creates interesting dynamics to explore. Notice that the position vector is still described in the body-frame, so the fixed inertial length between the spherical joint and the fixed position of the robot, ${}^{N}\vec{r}_l$, is rotated into the body frame using a rotation matrix R_N^B which rotates from the inertial frame N to the body-frame B.

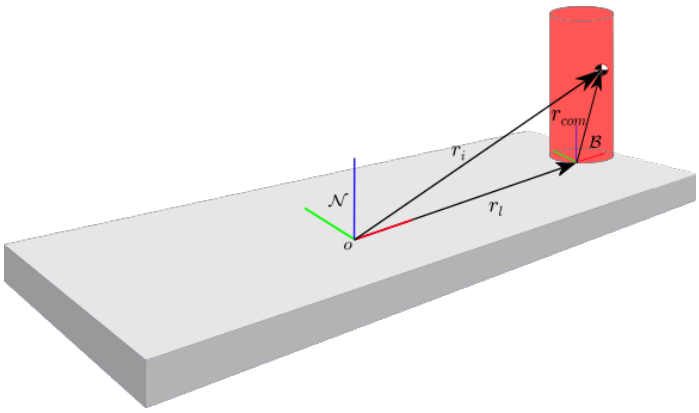


Figure 3.7 Platform Model Simplification

$$ {}^{B}\vec{r} = {}^{B}\vec{r}_{CoM} + \left[R_N^B\right]{}^{N}\vec{r}_l \tag{3.8} $$

Again, starting from angular momentum Equation (3.4), and taking the derivative to get to the inertial torque Equation (3.5), it can be seen that there is a body derivative of angular momentum as well as the cross product with the angular velocity

vector. To develop the body derivative of the angular momentum, two quantities can be further derived, the inertia about the center of mass as well as the displacement portion of the parallel axis theorem. Here the body derivative of the inertia with relation to the body frame is not changing, making its time derivative zero. However, because of the fact that the center of mass is no longer along the axis of rotation, the time derivative of the position of the center of mass in the inertial frame is no longer zero, creating a pseudo derivative of inertia, $\dot{I}$, given in Equation (3.10).

$$
\begin{aligned}
{}^{B}\dot{\vec{H}}_O &= \frac{d}{dt}\left({}^{B}I_{com} + m\left[{}^{B}\vec{r}^{\times}\right]\left[{}^{B}\vec{r}^{\times}\right]^{T}\right)\vec{\omega} \qquad (3.9)\\
&= \left[\frac{d}{dt}\left({}^{B}I_{CoM}\right) + m\,\frac{d}{dt}\left(\left[{}^{B}\vec{r}^{\times}\right]\left[{}^{B}\vec{r}^{\times}\right]^{T}\right)\right]\vec{\omega}
\end{aligned}
$$

$$
{}^{B}\dot{\vec{H}}_O = \underbrace{m\left(\left[{}^{N}\dot{\vec{r}}^{\times}\right]\left[{}^{B}\vec{r}^{\times}\right]^{T} + \left[{}^{B}\vec{r}^{\times}\right]\left[{}^{N}\dot{\vec{r}}^{\times}\right]^{T}\right)}_{\dot{I}_O}\vec{\omega} \qquad (3.10)
$$

Notice the position of the center of mass is still in the body frame while the velocity is about the inertial frame, requiring another transport theorem Equation (3.11). The angular velocities in these transport theorems represent the relationship between the frames; therefore the angular velocity used is the angular velocity of the body frame with respect to the inertial frame.

$$
{}^{N}\dot{\vec{r}} = {}^{B}\dot{\vec{r}} + \vec{\omega} \times {}^{B}\vec{r} \qquad (3.11)
$$

Combining the derivative of the body-frame angular momentum and the cross product of the angular velocity with the angular momentum gives way to the torque Equation (3.13) , the torque of one link about the spherical joint, O.

$$
{}^{N}\dot{\vec{H}}_O = {}^{N}\dot{I}_O\vec{\omega} + {}^{B}I_O\dot{\vec{\omega}} + \vec{\omega} \times {}^{B}I_O\vec{\omega} \qquad (3.12)
$$

$$
\begin{aligned}
{}^{N}\dot{\vec{H}}_O = m\left(\left[{}^{N}\dot{\vec{r}}^{\times}\right]\left[{}^{B}\vec{r}^{\times}\right]^{T} + \left[{}^{B}\vec{r}^{\times}\right]\left[{}^{N}\dot{\vec{r}}^{\times}\right]^{T}\right)\vec{\omega} + \dots \\
+\left({}^{B}I_{CoM} - m\left[{}^{B}\vec{r}^{\times}\right]\left[{}^{B}\vec{r}^{\times}\right]^{T}\right)\dot{\vec{\omega}} + \vec{\omega}\times\left({}^{B}I_{CoM} - m\left[{}^{B}\vec{r}^{\times}\right]\left[{}^{B}\vec{r}^{\times}\right]^{T}\right)\vec{\omega}
\end{aligned}
\tag{3.13}
$$

Equation (3.13) can be simplified to the form:

$$
\begin{aligned}
{}^{N}\dot{\vec{H}}_O = \dot{I}_O{}^{B}\vec{\omega} \; + \; \left({}^{B}I_{CoM} + m\left[{}^{B}\vec{r}^{\times}\right]\left[{}^{B}\vec{r}^{\times}\right]^{T}\right){}^{B}\dot{\vec{\omega}} \; + \dots \\
+ {}^{N}\vec{\omega}\times\left({}^{B}I_{CoM} + m\left[{}^{B}\vec{r}^{\times}\right]\left[{}^{B}\vec{r}^{\times}\right]^{T}\right){}^{B}\vec{\omega}
\end{aligned}
\tag{3.14}
$$

To gain a better picture of the system, adding links is the next obvious step. By adding a second link, the system is now exhibits five degrees of freedom, although for the sake of this problem, because the spherical joint will remain fixed to determine the torque acting upon it, there is effectively only two degrees of freedom. Similar to the single link case, the second link is derived using the same process, beginning with the angular momentum about the spherical joint then moving to the transport theorem to determine the torque about the joint. The key difference in this link is that the position of its center of mass is now dependent on the movement of the first link as well as its own movement.

$$
\vec{r}_2 = \left[R_N^{B_2}\right]\vec{r}_l + \left[R_{B_1}^{B_2}\right]\vec{l}_1 + \vec{r}_{CoM_2} \tag{3.15}
$$

Notice the position is still in the body frame of this link, link 2. Requiring the transport theorem Equation (3.12), the torque calculation utilizes the position of the center of mass in the link body frame while the velocity is again in the inertial frame. This development results in an identical equation. Naturally, different values are being used such as the position, and as a result the velocity, of the link in relation to the spherical air-bearing. The angular velocity, relates the body-frame of link two to the inertial frame instead of the first to the inertial frame. Finally the inertia tensor and mass are also different, representing the physical properties of the second link.

Taking the sum of these two equations, yields the total torque about the spherical joint, Equation (3.16).

$$^{N}\dot{\vec{\tau}}_{O} = {}^{N}\dot{\vec{H}}_{1_O} + {}^{N}\dot{\vec{H}}_{2_O} \tag{3.16}$$

3.2.4. Generalization

Now after generating the torque for two links, a trend can be seen, as the general torque equation for each link is the same. The difference is in the kinematic and physical quantities that are being used to determine the torque. To determine a torque of each link about the spherical joint, a general formula can be derived, beginning with the angular momentum of the i^{th} link, where the inertia is determined from the parallel axis theorem, Equation (3.17).

$$^{B}I_{CoM_i} + m_i \left[{}^{B}\vec{r}_i^{\times}\right] \left[{}^{B}\vec{r}_i^{\times}\right]^{T} \tag{3.17}$$

where the general position vector is:

$$\vec{r}_i = \left[R_N^{B_i}\right] \vec{r}_l + \left[R_N^{B_i}\right] \vec{l}_{i-1} + \vec{r}_{CoM_i} \tag{3.18}$$

and $\vec{l}_{i-1}$ represents previous link lengths to the current link Equation (3.20). For example, if position of the center of mass of link three was of interest, the lengths of link 1 and link 2 would need to be rotated from their respective body-frames to the link three body-frame and added together to get the distance from the fixed position on the platform to the location of the third revolute joint in the third body-frame reference frame.

From here the transport theorem can be used in conjunction with the angluar momentum to generate the torque of the i^{th} link about the inertial point O, Equation (3.19).

$$
\begin{aligned}
{}^{N}\dot{\vec{H}}_{i_O} &= m_i \left(\left[{}^{N}\dot{\vec{r}}_i^{\times} \right] \left[{}^{B}\vec{r}_i^{\times} \right]^T + \left[{}^{B}\vec{r}_i^{\times} \right] \left[{}^{N}\dot{\vec{r}}_i^{\times} \right]^T \right) \vec{\omega}_i + \dots \\
&+ \left({}^{B}I_{i_{CoM}} - m_i \left[{}^{B}\vec{r}_i^{\times} \right] \left[{}^{B}\vec{r}_i^{\times} \right]^T \right) \dot{\vec{\omega}}_i + \vec{\omega} \times \left({}^{B}I_{CoM} - m \left[{}^{B}\vec{r}^{\times} \right] \left[{}^{B}\vec{r}^{\times} \right]^T \right) \vec{\omega}
\end{aligned}
\tag{3.19}
$$

where the inertial velocity is determined by applying the transport theorem to i^{th} center of mass position. These positions each depend on the position of the joints before them, leading to a recursive-type formula for the body position of the i^{th} center of mass Equation (3.20).

$$
\begin{aligned}
{}^{B}\vec{r}_1 &= \left[R_N^{B_1} \right] \vec{r}_l + \vec{r}_{CoM_1} \\
{}^{B}\vec{r}_2 &= \left[R_N^{B_2} \right] {}^{N}\vec{r}_2 + {}^{B}\vec{r}_{CoM_2} \\
{}^{B}\vec{r}_3 &= \left[R_N^{B_3} \right] {}^{N}\vec{r}_3 + {}^{B}\vec{r}_{CoM_3} \\
&\vdots \\
{}^{B}\vec{r}_i &= \left[R_N^{B_i} \right] {}^{N}\vec{r}_{i-1} + {}^{B}\vec{r}_{CoM_i}
\end{aligned}
\tag{3.20}
$$

Equation (3.19) describes the dynamic torque of each body on the platform; gravity is not considered. However, as mentioned at the beginning of this section, static torques due to the weight of each of the bodies, the control box, and each link, exist. Taking the sum of the static torques and the dynamic torques at each time step for the i^{th} link leads to a general form of the torque of the i^{th} link about the inertial point O, the spherical air-bearing. To get the total torque about point O, the summation of the torque of each link about point O can be taken, Equation (3.21).

$$
{}^{N}\vec{\tau}_O = \sum_{i=1}^{8} {}^{N}\dot{\vec{H}}_{i_O} + {}^{N}\vec{r}_i \times {}^{N}\vec{F} \tag{3.21}
$$

Notice the summation goes to eight while there are only seven links. The control box can be considered to be another link, although the control box does not move,

meaning it will not generate any dynamic torque, although it will generate a static torque that needs to be accounted for.

3.3. Simulation

Dynamic modeling is the backbone of the design, however, there is no way to tell if the model is accurate; that is, without simulation. When modeling only the robotic arm using the recursive Newton-Euler method, testing was done to ensure that the simulation results matched the real robot; however for the platform that has yet to be fully assembled, comparison to a real, physical system is not yet possible. Although simulation was done in an environment within Simulink. The environment in Simulink that was used is a physics engine known as Simscape. Using Simscape, physical systems can be modeled using physical components that actually exist in these systems. Joints can be accurately modeled with sensor data taken directly from the simulated joints as well as fixed connections that separate different bodies of a system.

The Sawyer Robot was accurately modeled for comparison with the recursive method, Figure 3.8, to provide clarity about the accuracy of the recursive algorithm. Before testing with the real robot, it needed to be determined that the recursive method simulated with MATLAB was representative of a physical system, using Simscape. Notice connections seem to lead out of the figure. This is because the Simscape representation is much larger than the two links show; however the other five links of the Sawyer robot are of the exact same structure as the ones shown, so a zoomed in portion of the model is presented here to better visualize the Sawyer Simscape architecture.

The world frame is to the left, being an inertial fixed frame from which all subsequent simulation can be attached and referenced. The blocks that have figures resembling joints are in-fact revolute joints, where the B input connects the joint to the previous frame while the F denotes the follower frame. The q input is where motion is fed to the joint. For this simulation, a trajectory, *Table* 3.1, has been specified, meaning the angular position, velocity and acceleration of each joint has

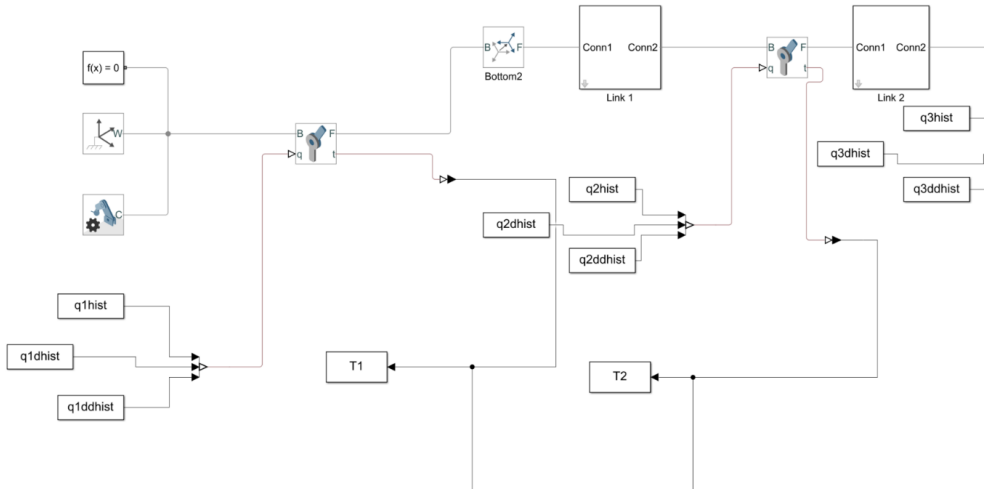


Figure 3.8 Sawyer Simscape Model

Table 3.1

Sinusoidal Trajectory for Sawyer Robot

Joint	$\boldsymbol{q}$	$\dot{\boldsymbol{q}}$	$\ddot{\boldsymbol{q}}$
1	$\sin t$	$\cos t$	$-\sin t$
2	$\cos t$	$-\sin t$	$-\cos t$
3	$\sin t$	$\cos t$	$-\sin t$
4	$\cos t$	$-\sin t$	$-\cos t$
5	$\sin t$	$\cos t$	$-\sin t$
6	$\cos t$	$-\sin t$	$-\cos t$
7	$\sin t$	$\cos t$	$-\sin t$

been prescribed for every moment in time. The output t represents the torque on the joint based upon prescribed motion as well as the physical system attached to it. The links are represented as white boxes with a label denoting which link the box represents. Inside these boxes, Figure 3.9 center of mass positions are specified, as well as the reference frames needed to precisely position the link to mimic reality. In the representation of link 2 below, the first reference frame attaches the body to the revolute joint while the second reference frame reflects the D-H parameter transformation, described in section 2.2., to the next revolute joint that will begin the next link.

Simulation of the recursive method and the Simscape model in the plots below, shown in Figure 3.10a and Figure 3.10b. It can be seen that the results look

practically identical but perhaps not exactly the same. By subtracting one from the other, it can be seen that they are actually the same results within a degree of MATLAB precision error, as shown in Figure 3.11a to Figure 3.11g.

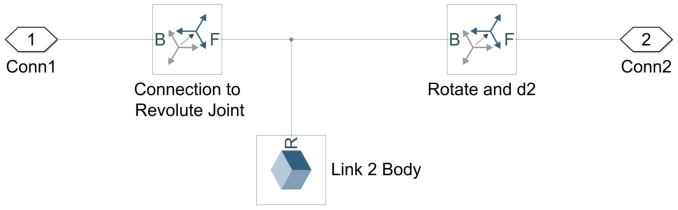
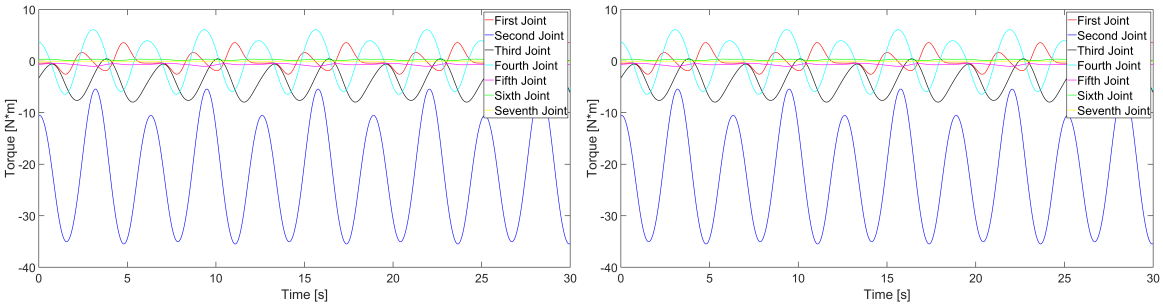


Figure 3.9 Sawyer Simscape Link Model

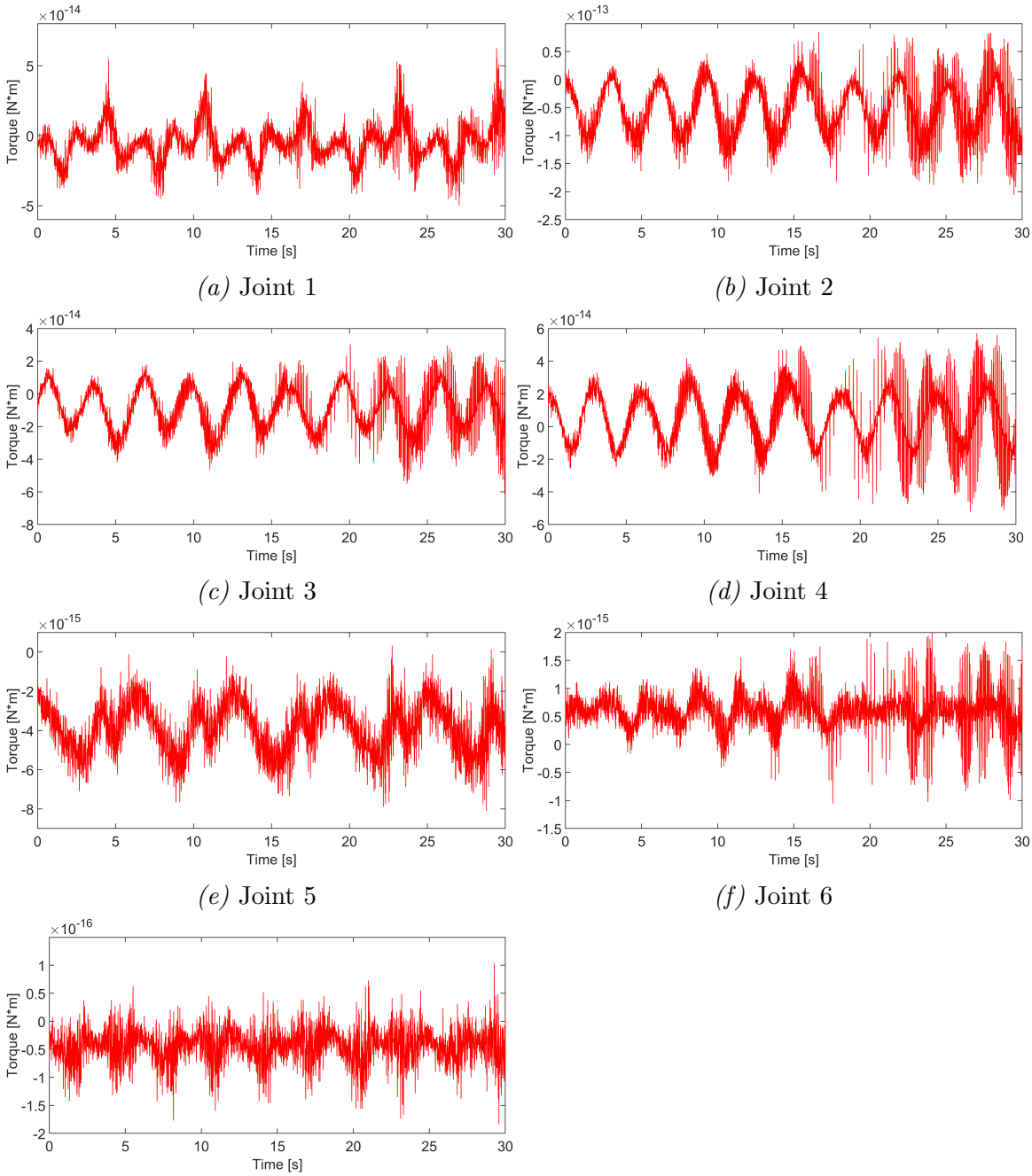


(a) Generated by the Recursive Method *(b)* Generated by Simscape

Figure 3.10 Torque Profile Based on *Table* 3.1 Trajectory

After simulating the Sawyer robot, naturally the next step is to model and simulate the spacecraft air-bearing platform as an entire system; Figure 3.12 depicts the robot arm attached to the air-bearing Figure 3.13. Using the same robot model from the recursive simulation additional bodies can be added to mimic the spacecraft simulator, as shown in Figure 3.14. The model now looks like several large boxes housing different models. The robot block houses the same model as seen before, but these new blocks represent the control box as well as the platform itself.

The difference between the robot and these separate bodies is these bodies do not move. They are dependent on the motion of the spherical air-bearing, as seen in Figure 3.14. Being dependent on the movement of only the spherical air-bearing these blocks have no other joints, meaning they will only move if the air-bearing moves. The spherical air-bearing is effectively locked in place for the simulation because the



(a) Joint 1 *(b)* Joint 2

(c) Joint 3 *(d)* Joint 4

(e) Joint 5 *(f)* Joint 6

(g) Joint 7

Figure 3.11 Relative Error between Recursive Method and Simscape^{TM}

interest of this development is to determine the torque about the spherical air-bearing while it maintains an attitude of 0 degrees of rotation in all axes: roll ϕ, pitch θ and yaw ψ. Therefore, the only dynamic torque is due to the motion of the robotic arm; the rest are static torques as a result of the weight of bodies within the system. Using

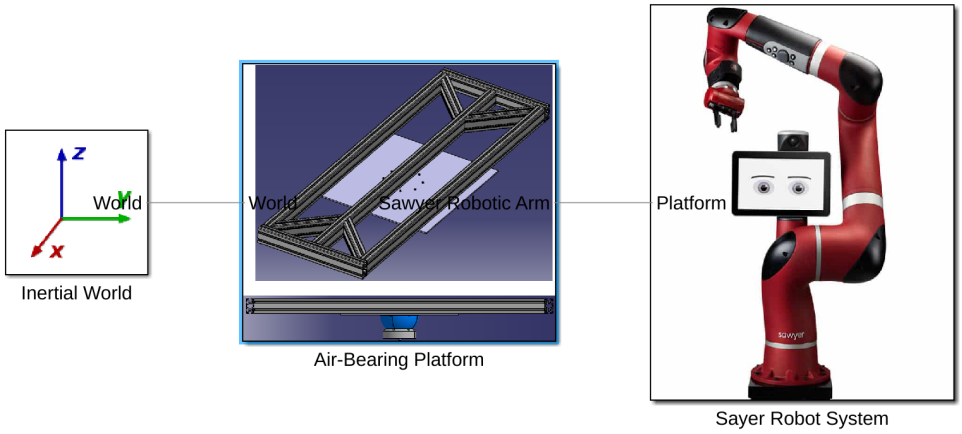


Figure 3.12 Complete Simulink Model Built in Simscape

Figure 3.13 Air-bearing Platform Modeled as a Gimbal

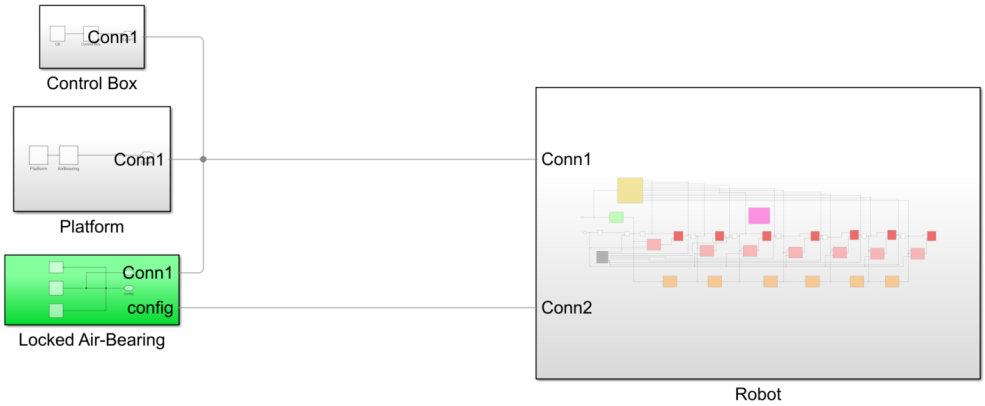
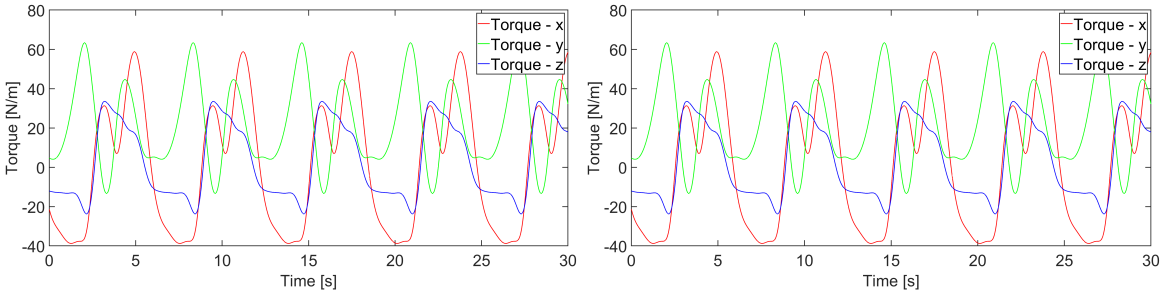


Figure 3.14 Air-Bearing Spacecraft Simulator Simscape Model

this knowledge, the Simscape model of this system may be quickly verified by switching gravity to zero and ensuring the only torque created is due to motion of the robot. The opposite may also be verified by ensuring the motion is zero and ensuring

the only torque present in the model is due to gravitational effects.

Comparing the MATLAB model, that is representative of the dynamics outlined in section 3.2.4., with the Simscape model, it can be seen in Figure 3.15a and Figure 3.15b that these two models again look similar but have minor differences. After again subtracting one result from the other, it can be seen in Figure 3.16 that they are infact the same again within a level of precision error. Notice the error is on an order of magnitude of 1×10^{-11}; this is due to the variation in differentiation methods used to determine the velocity. A forward differentiation was used in both the Simscape and MATLAB model but there is still precision error in Figure 3.16 that is able to propagate throughout the system causing an elevated error magnitude. Figure 3.17a and Figure 3.17b show and example corresponding to the differentiation of a sine function.



(a) Generated by Recursive Method *(b)* Generated by Simscape

Figure 3.15 Torque Profile of the Air-Bearing System

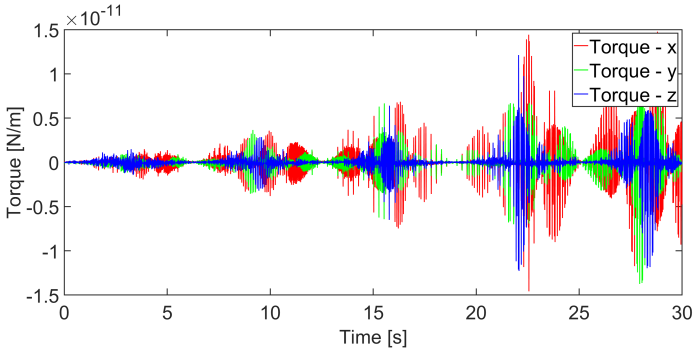
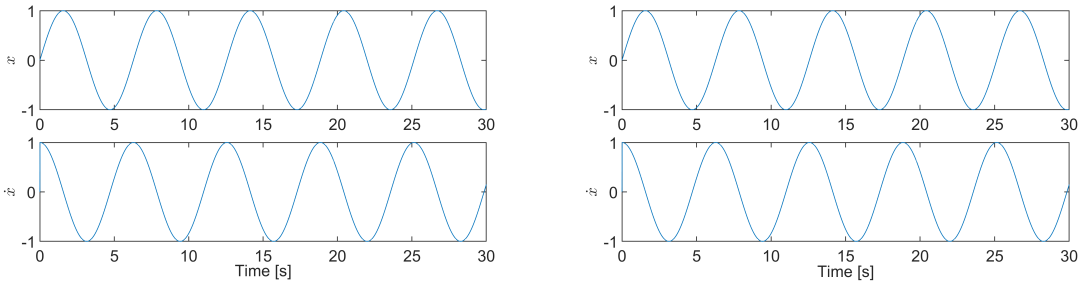


Figure 3.16 Torque Calculation Relative Error



(a) Computed with Matlab *(b)* Computed with Simscape

Figure 3.17 Derivative of $\sin x$

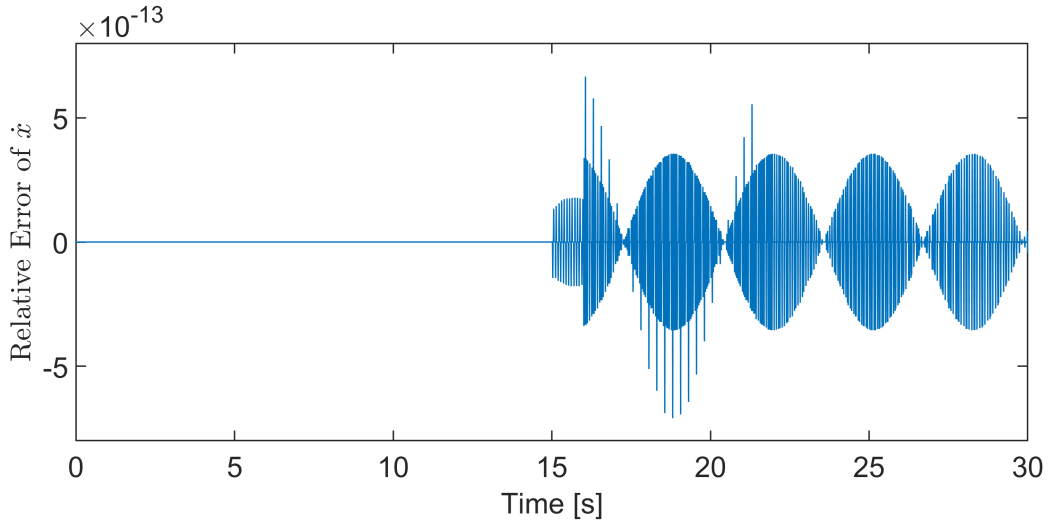


Figure 3.18 Derivative Calculation Relative Error Error

3.4. Periodicity

The pattern of a system is known as its periodicity. Many systems are periodic offering insights into their stability and boundedness. The periodicity of the system is typically seen through a phase plot; these depictions are often the derivative of a state plotted against the original state, in most cases x vs $\dot{x}$. For the case of the torque about the spherical air-bearing, torque is the state in question. The phase plot is the time derivative of torque, about a particular axis, plotted vs the torque about that same axis. Figure 3.19 to Figure 3.21 depict the phase plots for the x, y, and z axes.

In this simulation the torque in all three axes, meaning the entire torque vector, is periodic. The time of this simulation was ten minutes, sufficient time to see if the system converges to specific states or diverges showing instability. The initial torque

values are seen as the green circles while the torque at the final time, 10 minutes, is the red circle. The final time does not indicate the point at which the state will end; it simply corresponds to the point at which the simulation had been cut off. Furthermore, the system will continue tracing this path for all time due to the periodic forcing input. Instead of the system diverging or converging to a specific point, the system will continue on this trajectory for all time. However, this is not to say that the torque in general for this system is determined to be a limit cycle. It is purely dependent on the motion of the robot.

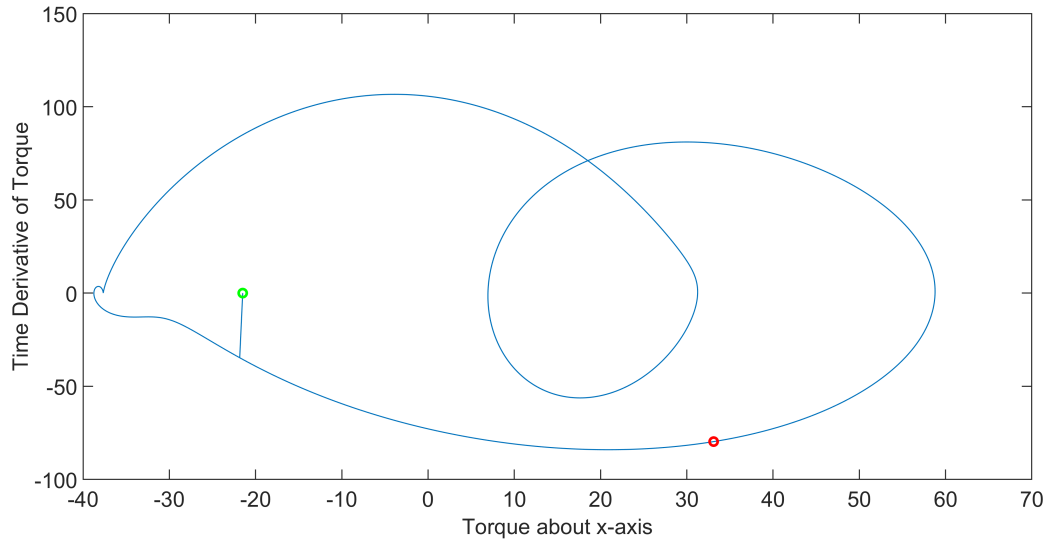


Figure 3.19 Phase Portrait about the x-axis

The previous cycle appears to be a limit cycle because the robot, in all joints, is exhibiting periodic behavior, following sinusoidal, bounded trajectories for all time *Table* 3.1. If the motion of the robot was different, perhaps a maneuver more realistic for a servicing satellite, such as a trajectory reaching to grab a tool as in Figure 3.22a, then returning to it previous position as in Figure 3.22b, described by the joint positions *Table* 3.2, the torque profile would not appear periodic. This motion would stop, if only for a short time before the robot began its servicing operation, meaning the torque would reach an equilibrium; the torque would remain at a value for all time, if not acted on by an external force.

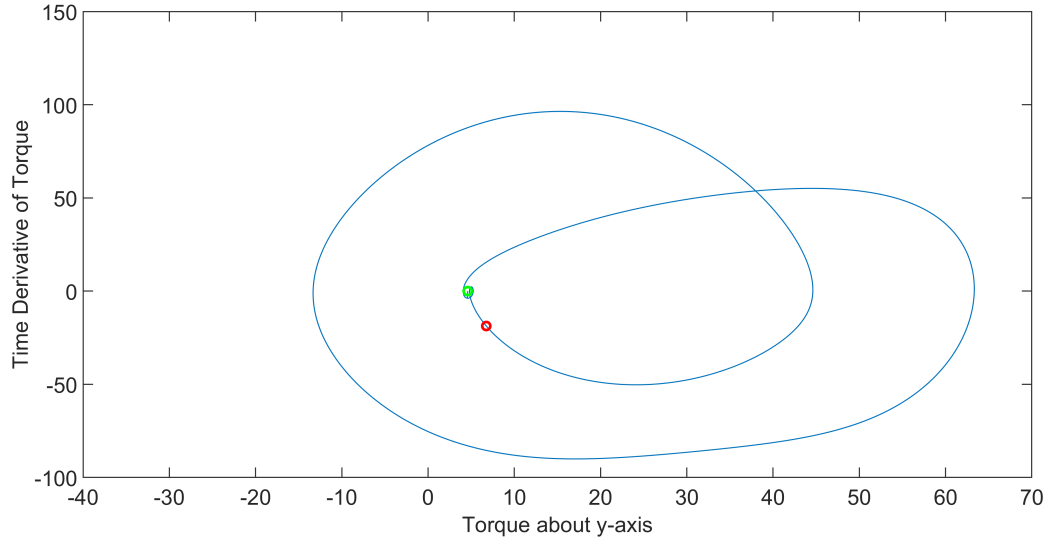


Figure 3.20 Phase Portrait about the y-axis

Table 3.2

Robot Joint Angular Positions [rad] for Servicing Maneuver

Joint	**Position 1**	**Position 2**	**Position 3**	**Position 4**	**Position 5**
1	0	π	π	π	0
2	0	0	0	0	0
3	$-\frac{\pi}{2}$	0	0	0	$-\frac{\pi}{2}$
4	0	$-\frac{\pi}{2}$	$-\frac{2\pi}{6}$	$-\frac{\pi}{2}$	0
5	π	0	0	0	π
6	0	$\frac{\pi}{2}$	$\frac{2\pi}{6}$	$\frac{\pi}{2}$	0
7	0	0	0	0	0

Shown in Figure 3.23 through Figure 3.25, the torque about the spherical joint does reach a final, equilibrium state because the motion of the robot has stopped. Now the torques on the spherical air-bearing are the torques created by gravity of the control box as well as the robot links, nonetheless the system is in equilibrium. The system is no longer periodic, it is now chaotic. However the system is bounded and reaches an equilibrium point meaning the system is stable.

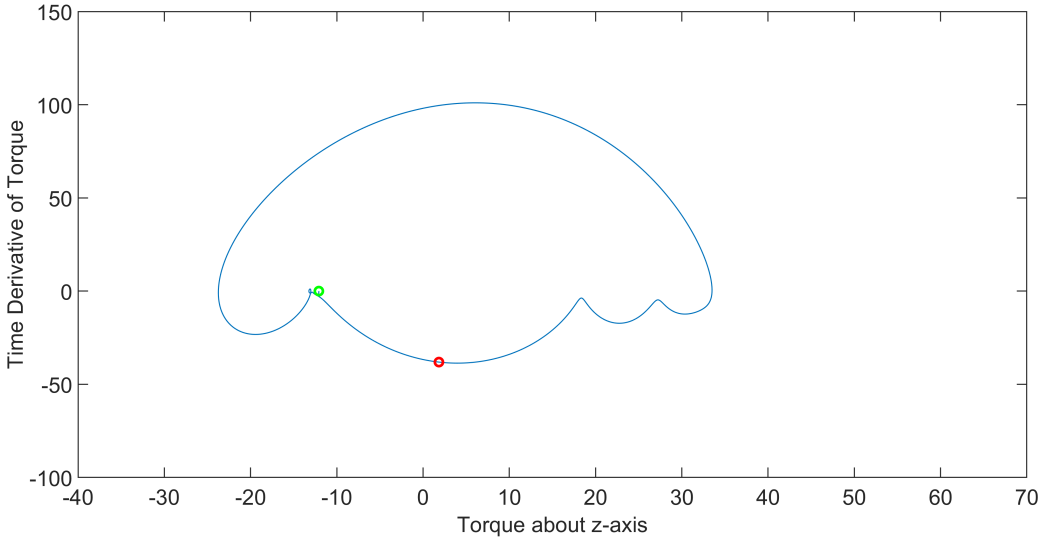
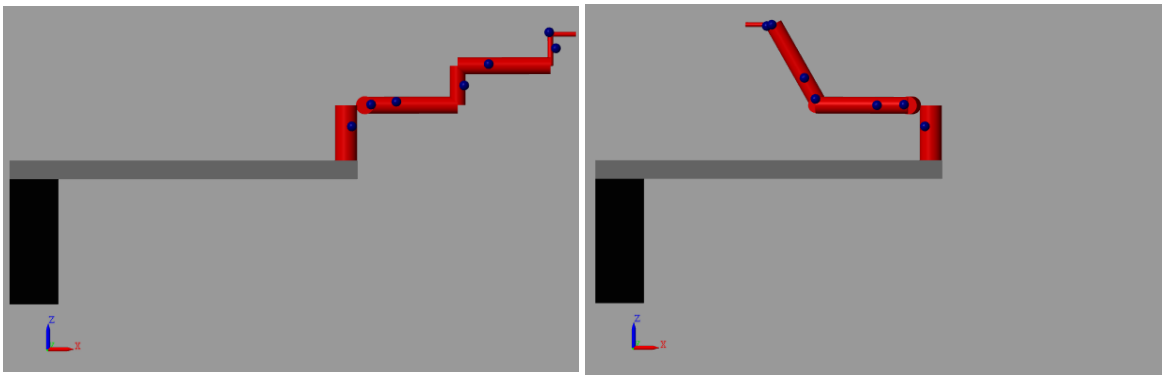


Figure 3.21 Phase Portrait about the z-axis



(a) Servicing Position *(b)* Tool Position

Figure 3.22 Tool Grab Maneuver

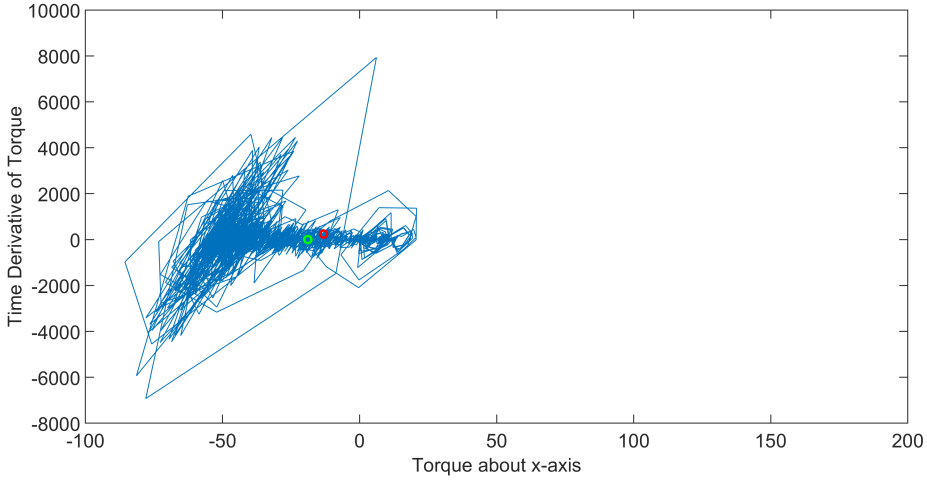


Figure 3.23 Phase Portrait about the x-axis Servicing Maneuver

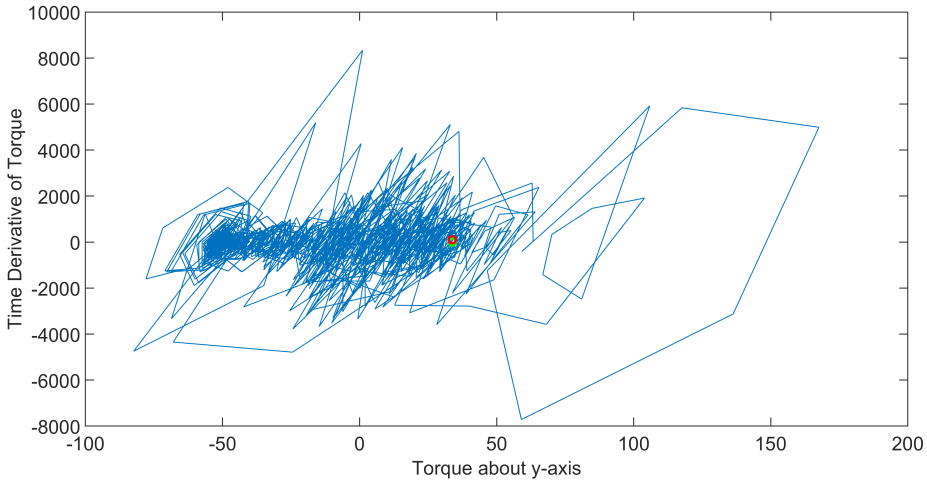


Figure 3.24 Phase Portrait about the y-axis Servicing Maneuver

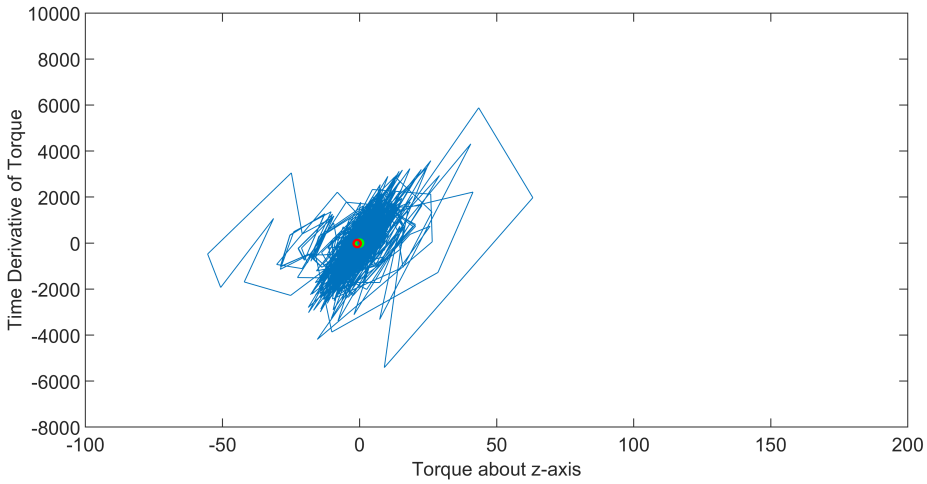


Figure 3.25 Phase Portrait about the z-axis Servicing Maneuver

4. Control Design

With the dynamic model of the system created, the next logical step in the course of creating a spacecraft attitude simulator is to design a controller to control the system's dynamics. The goal of this controller is to ensure the system can reject the torque generated by the robot's motion, all the while keeping the platform level at a fixed position. Different types of controllers can be designed to minimize error between a desired state and its actual value.

State-feedback controllers are a popular set of controllers, most often used in linear systems, that feed the current state to the controller, which will apply effort to minimize the error between the behavior of the desired system and the current system. Disturbance accommodating controllers are a popular set of controllers that are used to act as their namesake suggests, responding to the system's incoming disturbance all the while maintaining authority over the system to actuate or maintain a desired state. Non-linear controllers often need aspects of different controllers to aid in the command over the systems they were designed for allowing them to act non-linear in nature with the robust ability to handle a plethora of different conditions. This class of controller can often handle the described non-linearities of specific systems, track a desired state, and even eliminate disturbances. All of these qualities will be needed to design a controller for the spacecraft simulator.

Once designed, the controller can be tuned to respond in a desired fashion. Perhaps the goal is to actuate and control the system quickly; or perhaps the opposite is true, it might be necessary to minimize control effort. A desired frequency and damping ratio might be paramount. However, for space systems the goal is typically a critically damped system with no overshoot. The systems respond in a reasonable time, typically less than a minute while trying to limit control effort.

4.1. Possible Controllers

There are several options, as previously stated, to use as blueprints to design a controller for the spacecraft simulator. To select which class to work with, the states

need to first be identified. The states of interest for this system are the Euler angles as well as the angular rates, leading to the state space representation in Equation (4.2):

$$\vec{x} = \begin{Bmatrix} \vec{\Theta} \\ \vec{\omega} \end{Bmatrix} \tag{4.1}$$

$$\dot{\vec{x}} = \begin{Bmatrix} \dot{\vec{\Theta}} \\ \dot{\vec{\omega}} \end{Bmatrix} = \begin{Bmatrix} \beta\left(\Theta\right)\vec{\omega} \\ {}^{B}I^{-1}\left(\vec{M}_{ext} - {}^{B}\dot{I}\vec{\omega} - \vec{\omega} \times {}^{B}I\vec{\omega}\right) \end{Bmatrix} \tag{4.2}$$

4.2. State-Feedback Control

One option is a state-feedback controller, often associated with pole or eigenvalue placement. State-feedback offers the ability to add a control input to the system to exhibit the desired effects. This can be achieved through eigenvalue placement. Consider the linear time invariant system:

$$\dot{\vec{x}} = A\vec{x} + B\vec{u} \tag{4.3}$$

$$\vec{u} = -K\vec{x} \tag{4.4}$$

Here A represents the linear dynamics of the open-loop, uncontrolled, system. The matrix B represents how the control input is transmitted to the system. To develop the control law Equation (4.4), an eigenvalue placement approach can be taken where the coefficients of K can be solved for by matching the characteristic polynomial, Equation (4.5), generated from the determinant, to the characteristic equation generated by choosing desired eigenvalues for the system, as defined in Equation (4.6).

$$f(\lambda) = det\left|\lambda I - (A - BK)\right| \tag{4.5}$$

$$f(\lambda) = (\lambda - \lambda_1)(\lambda - \lambda_2)\ldots(\lambda - \lambda_i) \tag{4.6}$$

Using this matching technique, a controller can be quickly developed to emulate the desired dynamics set forth by desired eigenvalues. The spacecraft system can be simply linearized to develop a feedback controller. While this would not be the best controller for the system, it would provide a useful starting point to develop a non-linear controller. However, two separate issues prove problematic when trying to design a state-feedback controller. First, the disturbance created by the robot is not handled with the use of this state-feedback controller. These dynamics could possibly be lumped in with the dynamics of the platform through linearization, however this could prove to be a gross simplification of the system. In addition, the temporal character of a system utilizing constant-gain state-feedback must be time invariant, meaning the system does not explicitly change with time. The open loop dynamics of the spacecraft simulator will however change directly with time. The inertia of the system is dependent on the Sawyer robot, meaning while the system can be linearized about an equilibrium point, the A matrix will still remain time-varying. The time-varying aspect of this system limits the ability to use constant-gain state-feedback control laws, meaning a controller that can efficiently deal with time varying systems is necessary.

4.3. Time Varying Optimal Controller

A way to mitigate disturbances to a system is to have a portion of the controller aptly designed to deal with such dynamics. Disturbance accommodating controllers as well as robust controllers can respond to such changes. A large problem of this system is how to design a controller to handle the time-varying aspect of the system in conjunction with disturbances as these types of controllers are often designed to handle the tasks separately. However, a control law put forth by Gongyou, Yandong and Baolin, examines this exact problem. In this controller, the system being modeled is a time varying system with persistent disturbances.

$$\dot{\vec{x}}(t) = A(t)\,\vec{x}(t) + B(t)\,\vec{u}(t) + D(t)\,\vec{v}(t) \tag{4.7}$$

Notice the dynamic, controller, and disturbance matrices are all time-varying. A system such as this could resemble the spacecraft simulator where the disturbance is the torque generated by the robotic arm. The controller here will be subject to the optimal control cost function:

$$J = \int_{t_0}^{\infty} \left[\vec{x}^T(t)\, Q(t)\, \vec{x}(t) + \vec{u}^T(t)\, R(t)\, \vec{u}(t) \right] dt \tag{4.8}$$

where Q is positive semi-definite and R is positive definite. Using this cost function an optimal control law can be developed that rejects the disturbance all the while controlling the system in a feedback manner.

$$\vec{u}^{\,*}(t) = -R^{-1}(t)\, B^T(t) \left[P(t)\, \vec{x}(t) + \overline{P}(t)\, \vec{W}(t) \right] \tag{4.9}$$

In Equation (4.9) P is the position semi-definite solution to the Matrix Riccati differential equation:

$$\begin{aligned} -\dot{P}(t) = P(t)\, A(t) + A^T(t)\, P(t) - P(t)\, B(t)\, R^{-1}(t)\, B^T(t)\, P(t) + \ldots \\ +Q(t)\,, P_{\infty} = 0 \end{aligned} \tag{4.10}$$

while $\overline{P}$ is the solution to the matrix differential equation below that serves as the feedforward portion of the controller.

$$\begin{aligned} -\dot{\overline{P}}(t) = \left[A(t) - B(t)\, R^{-1}(t)\, B^T(t)\, P(t) \right]^T \overline{P}(t) + \overline{P}(t)\, G + \ldots \\ +P(t)\, D(t)\, HP(t)\,, \overline{P}_{\infty} = 0 \end{aligned} \tag{4.11}$$

The proof to the validity of this control law can be found in *Feedforward and Feedback Optimal Control for Linear Time-Varying Systems with Persistent Disturbances* (Gongyou et al., 2006). The crux of this control law hinges on the ability to generalize

the disturbance, $\vec{v}$, to a model:

$$\vec{v}(t) = H\vec{w}(t) \tag{4.12}$$

$$\dot{\vec{w}}(t) = G\vec{w}(t) \tag{4.13}$$

If the dynamics of the system can not be represented by a separate state space system as in Equation (4.12) and Equation (4.13), then the creation of the necessary G and H matrices is impossible.

For this linear time varying optimal controller to be successful there must be solutions for both the matrix ricatti equation, and the differential equation in terms of $\overline{P}$, as both P and $\overline{P}$ are necessary for the feedforward optimal controller. The matrix ricatti equation can be simply solved with the selection of satisfying Q and R matrices; although the second differential equation is different. The equation for $\overline{P}$ requires the use of D, G, and H, the matrices responsible for describing the generalized disturbance as described above. Because the torque values for this robot are completey deterministic, the development of generalized matrices is difficult and inaccurate. Without the generalized disturbance these matrices cannot be generated, meaning the optimal controller will not feature a feedforward disturbance accommodation behavior. Devoid of the ability to feedforward disturbance information to the controller, the system will not accurately represent the spacecraft with the Sawyer robot in motion.

The disturbance acting on the spacecraft satellite simulator, for the purpose of this research, is only the torque generated from the robotic arm. Perhaps disturbances for future iterations, such as solar pressure or gravitational gradient torques on the system, might be better suited for this control application although is not within the scope of this investigation. While this feedforward optimal controller is time varying and can reject disturbances, without the ability to feed through the

dynamics of the perturbation by generating the G and H matrices, this controller cannot be applied effectivley to the spacecraft air-bearing simulator.

4.4. Non-Linear Dynamic Inversion

Time-varying non-linear systems require more robust controllers, one such being a non-linear dynamic inverter, NLDI. This type of controller is a complex system usually consisting of multiple steps to achieve a response that is satisfactory. Initially, NLDI controllers have the ability to invert specific dynamics, effectively eliminating them from the system, thus resulting in a more manageable dynamical system. In some cases, NLDI's contain multiple loops for incremental non-linear dynamic inversion INLDI.

These separate loops, an inner and outer loop as shown in Figure 4.3, are used to generate a desired response. The inner loop is typically the "fast loop" meaning these are the dynamics that progress faster in time, in the sense they respond quicker than their outer loop counterparts. The "slow loop" is the outer loop that is responsible for a slower developing dynamical state. The inner-loops feed into the outer loop with the idea that stabilizing the "fast mode" dynamics will provide the ability to control the outer loop. Developed by Acquatella et al., in 2012 and demonstrated by Perez Rocha in 2016, the attitude of an aerospace system may be controlled in this very manner. The attitude, Euler angles, of the system serves as the outer loop while the inner loop consists of the angular rates of the system.

Recall the system described at the beginning of this chapter to be:

$$\dot{\vec{x}} = \begin{Bmatrix} \dot{\vec{\Theta}} \\ \dot{\vec{\omega}} \end{Bmatrix} = \begin{Bmatrix} \beta\left(\Theta\right)\vec{\omega} \\ {}^{B}I^{-1}\left(\vec{M}_{ext} - {}^{B}\dot{I}\vec{\omega} - \vec{\omega} \times {}^{B}I\vec{\omega}\right) \end{Bmatrix}$$

Notice that the Euler angular rates, are directly related to the angular rates of the system through the 3-2-1 rotation $\beta\left(\Theta\right)$ about the roll ϕ, pitch θ, and yaw ψ axes.

$$\beta(\Theta) = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \tag{4.14}$$

$$\dot{\vec{\Theta}} = \beta(\Theta)\,\vec{\omega} \tag{4.15}$$

While the dynamics of the system are expressed as:

$$\dot{\vec{\omega}} = {}^{B}I^{-1}\left(\vec{M}_{ext} - {}^{B}\dot{I}\vec{\omega} - \vec{\omega} \times {}^{B}I\vec{\omega}\right) \tag{4.16}$$

With the representation of the external torques of the system M_{ext}, the control input as well as the disturbance torque generated from chapter 3. are introduced to the system as in Equation (4.17).

$$\dot{\vec{\omega}} = {}^{B}I^{-1}\left({}^{B}\vec{\tau}_{D} + \vec{u}_{DI} - {}^{B}\dot{I}\vec{\omega} - \vec{\omega} \times {}^{B}I\vec{\omega}\right) \tag{4.17}$$

Here the first NLDI loop, Figure 4.1, is implemented to eliminate the robot generated disturbance as well as control the angular rate of the system to be the reference angular rate prescribed by the outer loop of the system in Figure 4.2. This inner loop utilizes a proportional-integral, PI, controller to facilitate the dynamic inversion, as shown in Equation (4.18) and Equation (4.19).

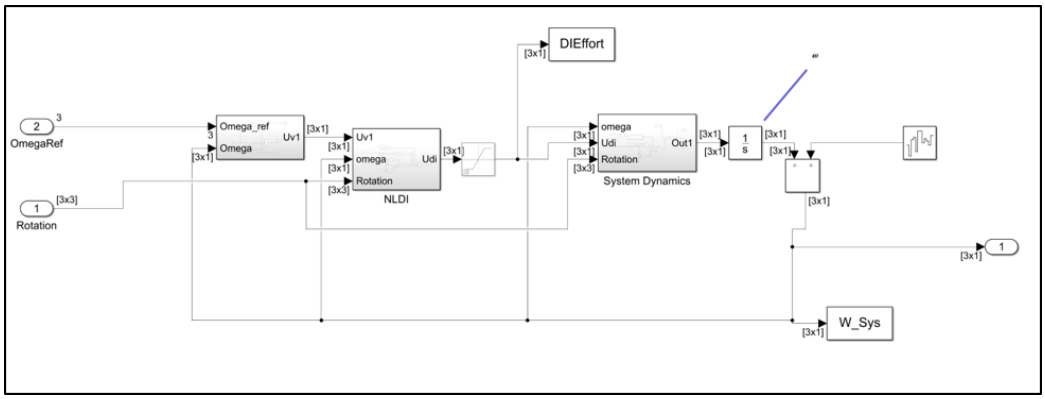


Figure 4.1 Inner Loop Fast Dynamics

$$\vec{u}_{DI} = {}^{B}\dot{I}\vec{\omega} + \vec{\omega} \times {}^{B}I\vec{\omega} - {}^{B}\vec{\tau}_{D} + {}^{B}I\vec{u}_{PI} \tag{4.18}$$

$$\vec{u}_{PI_1} = Kp_1\left(\vec{\omega}_{ref} - \vec{\omega}\right) + K_{i_1}\int\left(\vec{\omega}_{ref} - \vec{\omega}\right)dt + \dot{\vec{\omega}}_{ref}$$

$$\vec{e} = \vec{\omega}_{ref} - \vec{\omega}$$

$$\vec{u}_{PI} = Kp_1\vec{e} + Ki_1\int \vec{e}dt + \dot{\vec{\omega}}_{ref} \tag{4.19}$$

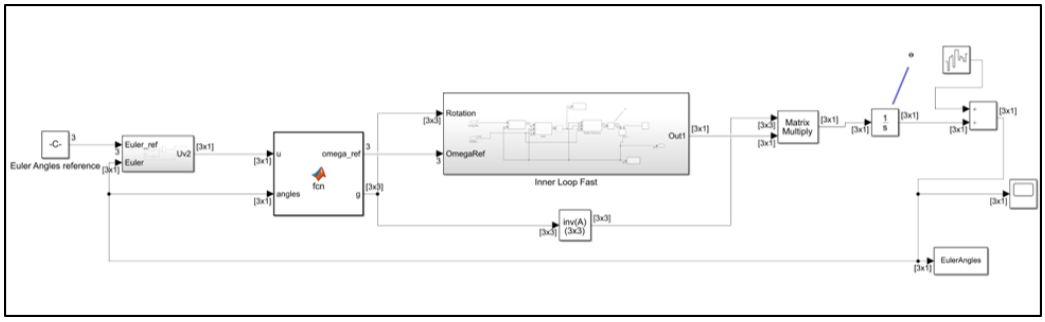


Figure 4.2 Outer Loop Slow Dynamics

The Euler angle dynamics are slower than those of inner loop, the angular rates. As a result, it is safe to assume that $\omega \approx \omega_{ref}$ (Perez Rocha, 2016). Furthermore, the desired angular rates, for the purposes of dynamic inversion will be of the form:

$$\vec{\omega}_{ref} = \beta^{-1}\left(\Theta\right)\vec{u}_{P} \tag{4.20}$$

The control input $\vec{u}$ can be selected to stabilize the slow dynamics of the system to track a desired angular position. The control input for this outer loop is a proportional controller; further explanation will take place in the tuning section. This leads to the simple inversion when fed back through to get the Euler angles

$$\dot{\vec{\Theta}} = \vec{u}_{P} \tag{4.21}$$

Together the system architecture is as follows:

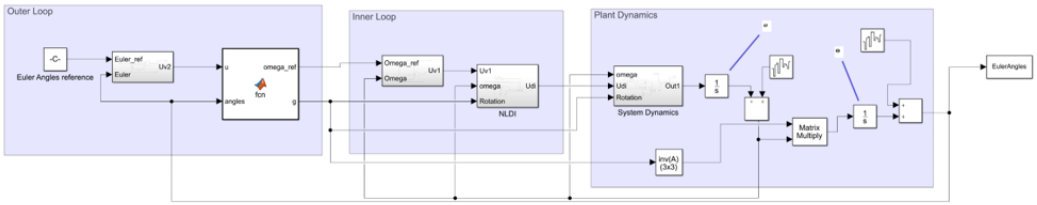


Figure 4.3 Closed Loop System Architecture

Notice the disturbance is also transformed; this is due to the nature of how the disturbance torque was developed, from the inertial frame, where it is required to be in the body frame for the development of the INLDI.

By simulating the system, it can be seen that the system can actuate to a specified vector of Euler angles of $\{0, 0, 90\}^T$ degrees, as well as reject the disturbance due to the robotic arm, based upon the trajectory from *Table* 3.1.

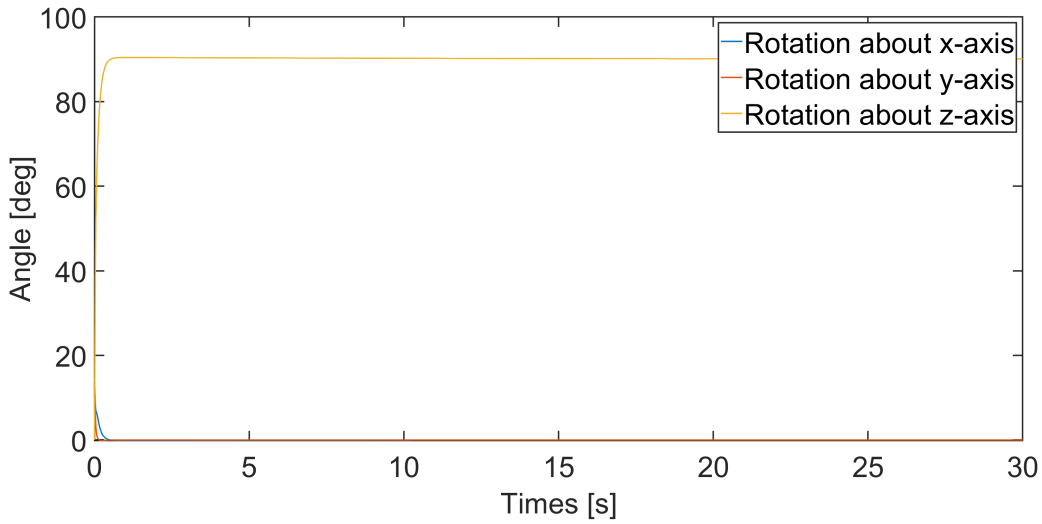


Figure 4.4 Tracking of specified Euler Angles $\begin{Bmatrix} 0° \\ 0° \\ 90° \end{Bmatrix}$

Here the closed-loop system is shown to be a critically damped system that responds in what appears to be less than one second, as shown in Figure 4.5, which is very fast. However, the effort required to actuate the system to these conditions so quickly is far too high to be realistic. To try and yield a quickly controlled system all

the while utilizing realistic effort is necessary and can be achieved by tuning this controller.

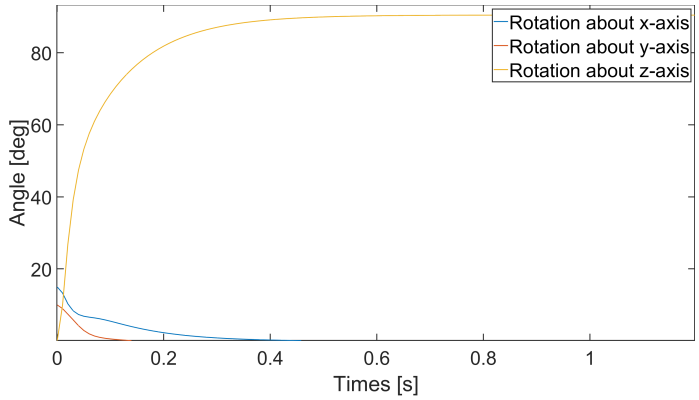


Figure 4.5 Tracking of specified Euler Angles First Second

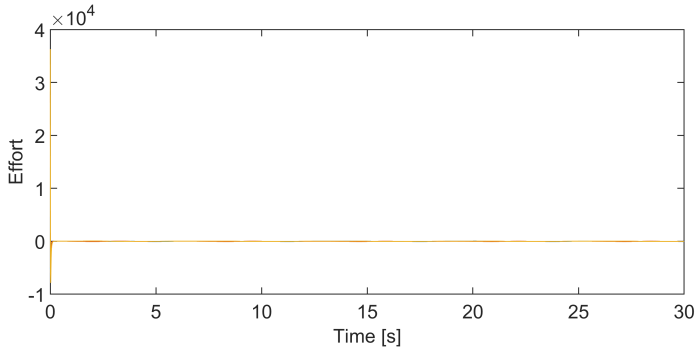


Figure 4.6 Control Effort to Control System in under 1 second

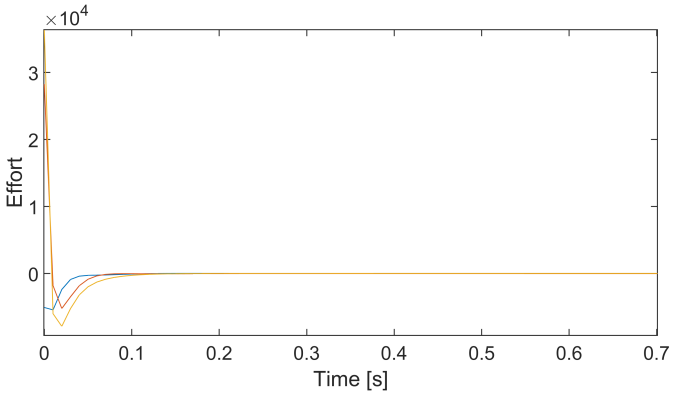


Figure 4.7 Control Effort under 1 second

4.5. Tuning

To ensure that the desired response is generated using a controller tuning is often necessary. Controllers have specific coefficients that are intended to scale dynamic

values to achieve a specific result. Linear quadratic regulators use variable Q and R matrices to achieve desired results. Using feedback controllers, the K can be manipulated to yield the best outcome. For this INLDI, there are two sets of gains that can be tuned to achieve a desired result. These correspond to a PI controller that affects the angular rate as well as the proportional controller that scales the error value of the Euler angles.

The desired response is to reject the disturbance created by the robot while actuating to a specific set of Euler angles. Initially the goal is to capture a rotation 90 degrees about the yaw or z-axis while aligning with 0 degrees in both the pitch and roll axes. The controller can more than handle this maneuver under perfect conditions, meaning limitless control effort, all the while quickly responding in a critically damped fashion. However, in reality this amount of control effort would require several large control moment gyro's, CMG's. To increase the amount of time the system takes to fully capture the desired angles and reduce the required control torques, the proportional gains of the controllers must be decreased. When lowered to 0.2 and 0.05 for the K_p of the inner loop and outer loop respectively, the control effort is reduced and the settling time, the time until steady state is achieved, is increased. Instead of a critically damped response, the system now exhibits large oscillations. To reduce these oscillations, the integral action of both controllers is reduced to 0.1 and 0 for the inner and outer loops respectively. By reducing the outer loop integral action to zero, the controller is reduced to a simple proportional controller. This tuned controller is able to produce a critically damped system, as shown in Figure 4.8, with a much longer settling time, over 100 seconds, while drastically reducing the control effort, Figure 4.9.

This control effort is still rather high at the beginning of the maneuver however this is actuating the system from the following initial conditions:

$$\Theta_0 = \begin{Bmatrix} 15° \\ 10° \\ 90° \end{Bmatrix}, \quad \omega_0 = \begin{Bmatrix} 0 \\ 0 \\ 2 \end{Bmatrix} \frac{\text{deg}}{\text{s}}$$

while rotating the system 90 degrees about the z-axis. This control effort can be drastically reduced by limiting the rotation about the z-axis to 45 degrees, as shown inFigure 4.10a and Figure 4.10b, or even 0 degrees, as depicted in Figure 4.11a and Figure 4.11b.

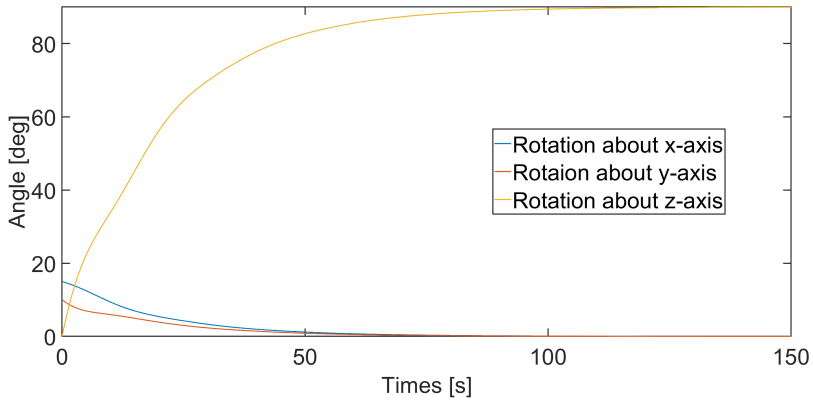


Figure 4.8 Tracking Euler Angles $\begin{Bmatrix} 0° \\ 0° \\ 90° \end{Bmatrix}$, Longer Settling Time

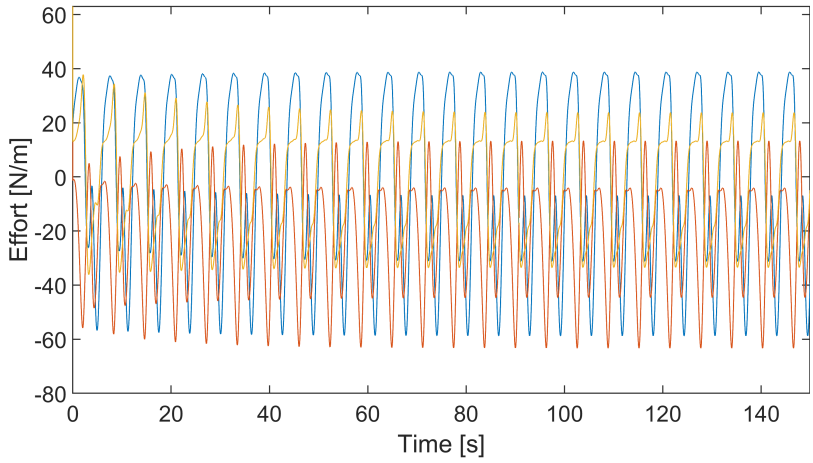
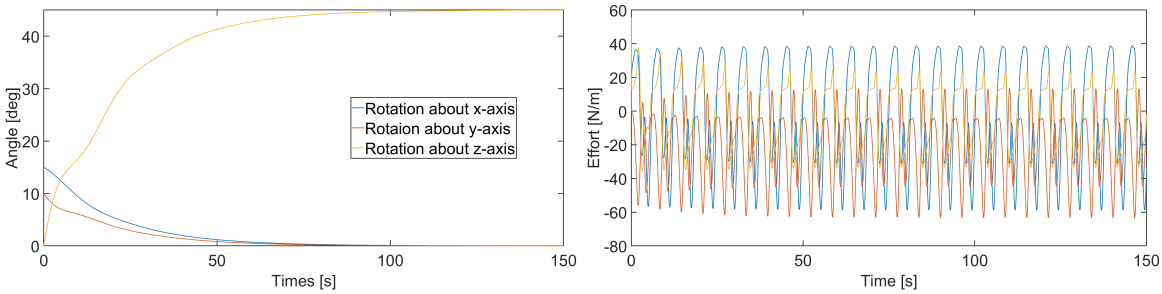
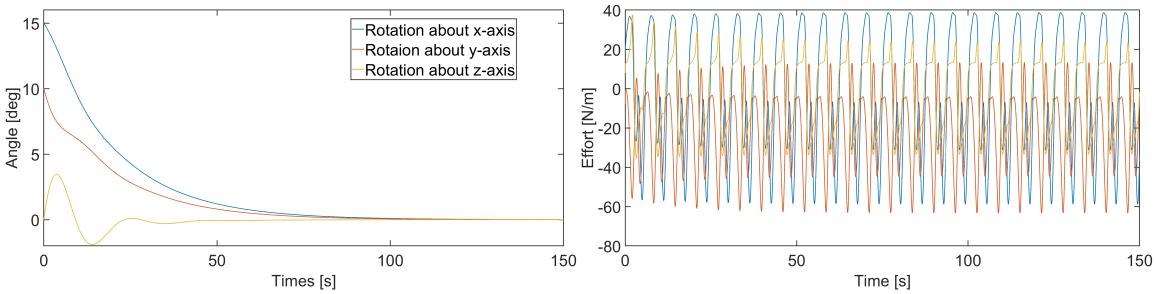


Figure 4.9 Control Effort for Longer Settling Time



(a) Dynamic Response *(b)* Control Effort

Figure 4.10 Euler Angles $\begin{Bmatrix} 0° \\ 0° \\ 40° \end{Bmatrix}$



(a) Dynamic Response *(b)* Control Effort

Figure 4.11 Euler Angles $\begin{Bmatrix} 0° \\ 0° \\ 0° \end{Bmatrix}$

Notice the controller is producing negative torque as well as positive torque. This is generated by the direction of rotation of the reaction wheels; however this profile is in response to the sinusoidal trajectory of each link of the robot as described by *Table* 3.1.

A more useful trajectory for the robot to follow is described by *Table* 3.2, the tool maneuver. Using this set of position, velocity and acceleration data that come along with it from the the robot, the torque profile can be generated, using the dynamics from chapter 3., shown in Figure 4.12. Using this data, the controller can be tuned to reject this disturbance while actuating the system to:

$$\Theta = \begin{Bmatrix} 0° \\ 0° \\ 0° \end{Bmatrix}$$

from the initial conditions:

$$\Theta_0 = \begin{Bmatrix} 15° \\ 30° \\ 0° \end{Bmatrix}, \quad \omega_0 = \begin{Bmatrix} 0 \\ 0 \\ 0 \end{Bmatrix} \frac{\text{deg}}{\text{s}}$$

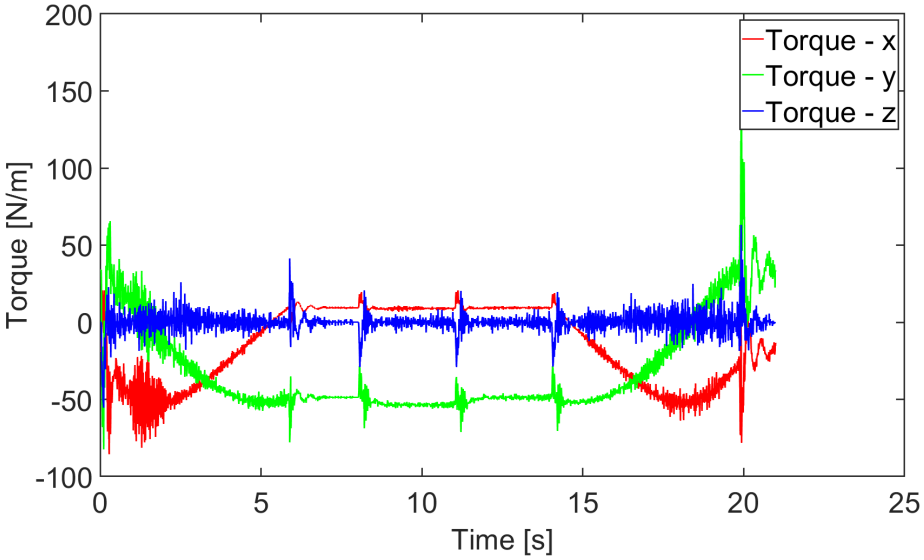


Figure 4.12 Torque Generated from Tool Maneuver

Here the K_p of the outer loop was increased to 0.2 to ensure the system stabilized in 20 seconds, the length of the maneuver. If reduced settling time of the system was not an objective, the gain could be lowered by an order of magnitude, increasing the time of stabilization and lowering the amount of torque necessary to actuate the system.

Noise is another characteristic to consider when designing a controller. The controller will in most cases not have access to the exact angular rate of the system, rather it receives its input based on a sensor reading provided by a rate gyro or

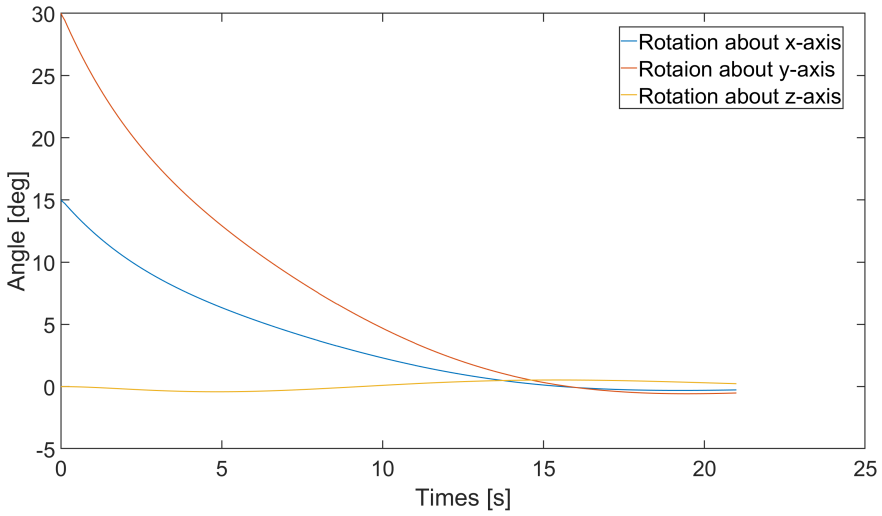


Figure 4.13 Tracking of Specified Euler Angles for Tool Maneuver

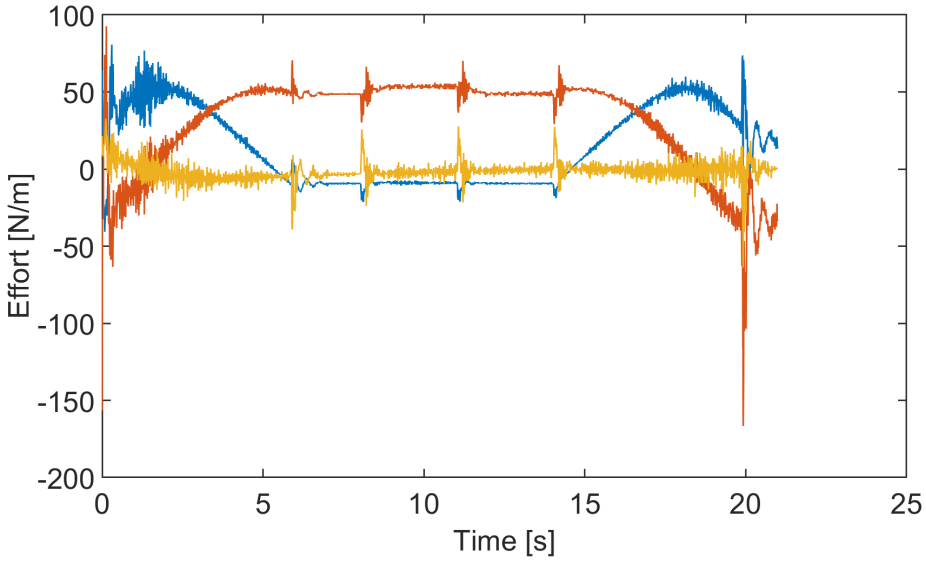


Figure 4.14 Control Effort for Tool Maneuver

accelerometer. These measurements acquire noise, which is generated from many different sources, such as vibrations, radiation from the earth or other celestial bodies. These sources of noise can be categorized as Gaussian, for their normal distribution within the time domain and can be modeled as such. By adding noise into the system where there would be sensor readings into the system, namely, the input of Euler angles as well as angular rates back into the system, noise can be effectively modeled.

It must be noted that noise was simply added into the simulated measurements

that were input back into the controller. The controller istelf was not designed with the intent to filter or effectivley manage noise. Measurement noise was simply added to the simulation for the purpose of observing its effect on the controller. The specifications of the VectorNav200 are used to model the noise, where the angular accuracy has an error of less than 0.05 degrees, and the resolution of the angular velocity is less than 0.02 degrees per second.

As can be seen in Figure 4.15, the system makes an attempt to settle around the desired Euler angles of $\{0°, 0°, 0°\}$ but the controller cannot handle the noise. Therefore the system is rocking constantly. The best way to avoid this situation is to design a Kalman filter to estimate the current states instead of using direct feeback of sensor measurements. However, this requires non-linear estimation which is not in the scope of this research; however this could be addressed in future work.

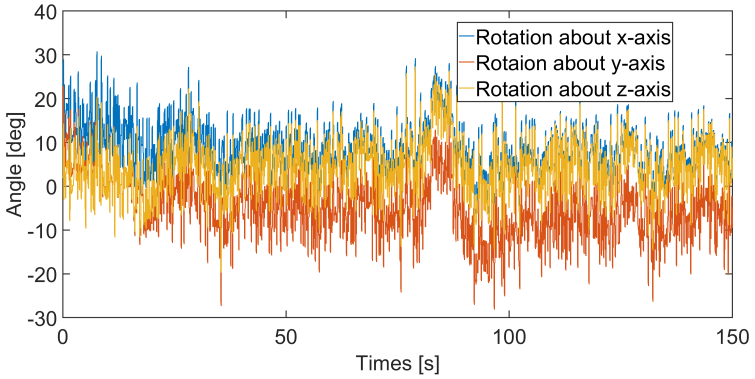


Figure 4.15 Tracking of Specified Euler Angles with Noise

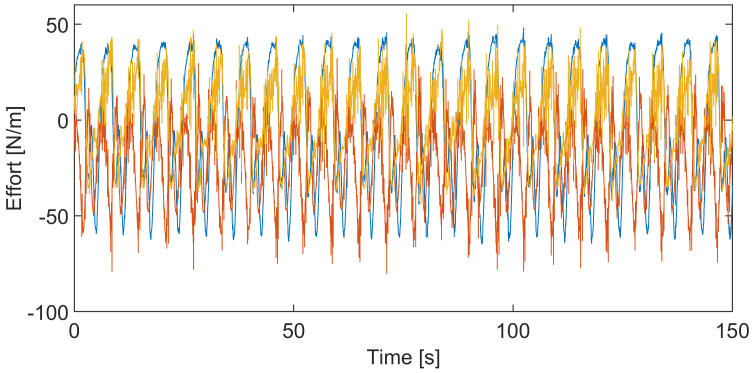


Figure 4.16 Control Effort with Noise

5. Conclusion

The air-bearing spacecraft simulator is a dynamical system that features a spacecraft with the goal of servicing a client satellite. Two robotic arms are utilized in the system; however the dynamics and control of the servicing satellite provide the interesting physics discussed in this work. The servicing satellite consists of a Sawyer robot, its control box, and a structure to house these objects, the platform. Designing the platform with the dynamical system in mind it will not deflect more than three tenths of a millimeter making the simulation results representative of reality with little need to consider flexible beam bending. The goal of this investigation was to develop the dynamics of the system, leading to the design of a controller capable of not only responding to the disturbance of the robot but also to actuate the spacecraft to a desired attitude.

With the goal of a controller in mind, the first step is to dynamically model the system. The Sawyer robot was initially modeled with a recursive method to determine the torques about each joint, without regard for the platform. Simulating this Newton-Euler approach with a physics simulation software inside Simulink as well as the physical robot, three methods of determining torque were compared. These comparisons were all similar meaning the Newton-Euler approach is relatively indicative of reality, while ignoring joint damping and frictional effects.

The next step was to model the system including the platform, adding three more degrees of freedom. The recursive method was not developed for joints with multiple degrees of freedom; trying to include a spherical joint in the development resulted in singularities that were not representative of the actual system; thus a new analytical development took place to develop the equations of motion of the spacecraft attitude. The system was again simulated using Simscape, as the physical system has yet to be constructed. The system initially was seen to be periodic based upon the input kinematics, position, velocity and acceleration. However, it was shown that with different inputs, those more representative of a satellite servicing maneuver, the

system was in fact chaotic, all the while stable.

After the dynamics had been developed, the control design took place. The goal of the controller was to reject the disturbance created by the robot while maintaining stability of the air-bearing spacecraft. Three approaches were hypothesized; a state-feedback controller, feedback-feedforward optimal control, and non-linear dynamic inversion. The system could be linearized about an equilibrium point, however that does not eliminate the time dependence of the system, meaning a state-feedback controller would be unsuccessful. The disturbance was not easily generalized into a state space system meaning that the feedback-feedforward optimal controller could not be designed, leaving the final option of non-linear dynamic inversion, NLDI. Using the NLDI, the dynamics of the system could be separated into two loops, an inner and outer loop. The inner loop pertains to the faster dynamics of the system, the angular velocity, while the outer loop regulates the Euler angles. After a bit of tuning, the NLDI was effective at rejecting the disturbance of the robot while actuating the system to a desired attitude. The initial control effort spike is large; however, this is under the assumption that the actuators will be starting from rest which is most likely incorrect.

5.1. Future Work

Noise is an artifact of the sensor measurements used. This simulation feeds the actual state through to the control when in reality, the utilized states will come from sensor measurements, adding in noise. Most systems do not directly feed sensor data into their controllers. In most cases there is an estimator that will estimate the necessary state to ensure that control system dynamics that utilize states have access to a noise free estimation of the state. With the addition of noise into the system, as if the states were being received from sensors, the controller has difficulty converging causing the system to result in a periodic error rocking back and forth in a harmonic fashion in an attempt to converge to the desired Euler angles. Non-linear state estimation would smooth out and reduce the noise entering the system, allowing for

the controller to converge to a state the is provided not by sensors or the physical system but an estimation of the real dynamics.

Another way to increase the fidelity of this simulation with reality is to model the actuator dynamics, including saturation limits. The actuators, either control moment gyros, CMGs or a set of reaction wheels, will not simply respond as step functions and instantly provide the necessary amount of torque. Instead they will most likely be spinning at a low rate and quickly spool up when necessary, although they also have saturation limits. The actuators might not be able to provide the maximum amount of control effort, in this case torque, necessary to control the system, resulting in a saturated controller, most often resulting in an unstable system. By modeling the actuator dynamics, the simulation will be better representative of the actuator in reality, providing a better estimate of the stability and even how quickly maneuvers can occur for the system to remain stable, or if they are even possible or will they saturate the controller.

To possibly reduce the control effort necessitated by the controller, perhaps a robust sub-optimal controller could be used. This controller would be designed to the upper bound of the disturbance, meaning the greatest disturbance that could be possibly created by the robot. By designing to this upper bound the controller would be able to manage the system provided any possible disturbances. The current PI controller uses constant gains whereas the sub-optimal controller would feature time dependent gains for the purposes of optimization, and as a result should be able to lower the necessary control effort.

Finally, perhaps the most important future installment is to select a mission for which to cater the design. Be it on-orbit repairs, refueling or asteroid mining, the specific mission will provide a better idea of what kind of maneuvers the robot will be making in order to more specifically size the actuators necessary to control the system on Earth. The specific mission will also aid in determining optimization parameters, be it for time, torque or mission dependent. Repairs will be many small precise

maneuvers of small variations, generating a steady amount of low torque. On the contrary, refueling my be short bursts of large disturbances from the robot needed to operate about a large path but in sparse time segments, generating a high burst of torque to reject disturbances and provide minor station-keeping.

5.2. Concluding Remarks

Regardless ofthe mission or necessary actuators, a test bed has been designed modeling the dynamics of not only the robot but also the attitude of the entire spacecraft. Using these dynamics, a non-linear dynamic inversion controller was designed that rejected the robot disturbance torque as well as actuating the system to the desired attitude. The next step is to design a non-linear estimator that will feed this controller to reduce the effect of noise from sensor measurements.

REFERENCES

Acquatella, P., Falkena, W., van Kampen, E.-J., & Chu, Q. P. (2012). Robust nonlinear spacecraft attitude control using incremental nonlinear dynamic inversion. In *Aiaa guidance, navigation, and control conference* (p. 4623).

Antonello, A., Valverde, A., & Tsiotras, P. (2018). Dynamics and control of spacecraft manipulators with thrusters and momentum exchange devices. *Journal of Guidance, Control, and Dynamics*, *42*(1), 15–29.

Brockett, R. W. (1984). Robotic manipulators and the product of exponentials formula. In *Mathematical theory of networks and systems* (pp. 120–129).

Currie, N. J., & Peacock, B. (2002). International space station robotic systems operations a human factors perspective. In *Proceedings of the human factors and ergonomics society annual meeting* (Vol. 46, pp. 26–30).

Fornoff, H. (1967). Final report for air bearing platform t50-2. *NASA Contractor Report NASACR-97588*.

Guo, Y., Kang, G., Pan, J., Jin, C., & Qiao, S. (2017). Research on three axis air bearing platform simulation system for small satellite attitude control. In *2017 36th chinese control conference (ccc)* (pp. 10322–10327).

Haeussermann, W., & Kennel, H. (1960). A satellite motion simulator. *Astronautics*, *5*(12).

Heney, P. (2017, Aug). *Rethink robotics expands global reach with new distribution partners.* Retrieved from https://www.designworldonline.com/rethink-robotics-expands-global-reach-with-new-distribution-partners/

Junkins, J. L., & Schaub, H. (2009). *Analytical mechanics of space systems*. American Institute of Aeronautics and Astronautics.

Liu, S., & Chen, G. S. (2019). *Dynamics and control of robotic manipulators with contact and friction*. John Wiley & Sons.

Murray, R. M. (1994). *A mathematical introduction to robotic manipulation.* CRC Press.

Nazari, M., Maadani, M., Butcher, E. A., & Yucelen, T. (2018). Morse-lyapunov-based control of rigid body motion on tse (3) via backstepping. In *2018 aiaa guidance, navigation, and control conference* (p. 0602).

Park, F. C. (1991). The optimal kinematic design of mechanisms.

Park, F. C., & Bobrow, J. E. (1994). A recursive algorithm for robot dynamics using lie groups. In *Proceedings of the 1994 ieee international conference on robotics and automation* (pp. 1535–1540).

Park, F. C., Bobrow, J. E., & Ploen, S. R. (1995). A lie group formulation of robot dynamics. *The International journal of robotics research*, *14*(6), 609–618.

Piglide hb hemispherical air bearings user manual [Computer software manual]. (n.d.).

Ploen, S. R. (1998). Geometric algorithms for the dynamics and control of multibody systems.

Pond, B., & Sharf, I. (1999). Experimental demonstration of flexible manipulator trajectory optimization. In *Guidance, navigation, and control conference and exhibit* (p. 4302).

Prado, J., Bisiacchi, G., Reyes, L., Vicente, E., Contreras, F., Mesinas, M., & Juárez, A. (2005). Three-axis air-bearing based platform for small satellite attitude determination and control simulation. *Journal of Applied Research and Technology*, *3*(3), 222–237.

Rocha, A. E. P. (2016). Development of fault tolerant adaptive control laws for aerospace systems.

Rybus, T., & Seweryn, K. (2016). Planar air-bearing microgravity simulators: Review of applications, existing solutions and design parameters. *Acta Astronautica*, *120*, 239–259.

Sawyer safety overview [Computer software manual]. (n.d.).

Schwartz, J. L. (2004). *The distributed spacecraft attitude control system simulator: From design concept to decentralized control* (Unpublished doctoral dissertation). Virginia Polytechnic Institute and State University.

Schwartz, J. L., Peck, M. A., & Hall, C. D. (2003). Historical review of air-bearing spacecraft simulators. *Journal of Guidance, Control, and Dynamics*, *26*(4), 513–522.

Selig, J. M. (2013). *Geometrical methods in robotics*. Springer Science & Business Media.

Spong, M. W., & Vidyasagar, M. (2008). *Robot dynamics and control*. John Wiley & Sons.

Tang, G., Zhao, Y., & Zhang, B. (2006). Feedforward and feedback optimal control for linear time-varying systems with persistent disturbances. *Journal of Systems Engineering and Electronics*, *17*(2), 350–354.

Tao, L. (2014). Research on abroad video satellite development. *Space International*, *9*(429), 50–56.

The MathWorks, Inc. (n.d.). Simscape user's guide r2019b [Computer software manual].

Yoshitsugu, T., Toshiaki, I., Kazuo, M., Akiko, O., Hidetoshi, T., & Yasuo, S. (1991). Development of flying telerobot model for ground experiments. *The Journal of Space Technology and Science*, *7*(2), 2_15–2_22.

Yoshitugu Toda, Y. F., Akiko Otuka. (1992). Development of free-flying space telerobot, ground experiments on 2-dimensional flat test bed. In *Proceedings guidance, navigation and control conference, 1992* (pp. 33–39).

Zuhui, S. (2010). *Mechanics of materials (i)[m].* Beijing: Higher Education Press.