

## ABSTRACT

**HUGHES, HUNTER D.** Optimal Control for Spacecraft Large Angle Maneuvers Using  $H_\infty$  Linear Varying Parameter Control Techniques. (Under the direction of Associate Professor Fen Wu).

This study investigates the possibility of designing and implementing a Linear Parameter Varying controller with  $H_\infty$  performance criteria integrated into the synthesis of the controller for a spacecraft undergoing a large angle maneuver about its principal axes. Towards this end, Cayley-Rodrigues parameters were used to model nonlinear spacecraft dynamics and to ensure up to  $180^\circ$  of rotation about the principal axis without singularities in the system. Several linear parameter varying (LPV) controllers with different parameter ranges were designed and the closed-loop performances were compared with respect to the  $H_\infty$  upper bound  $\gamma$ . The optimal  $\gamma$  value obtained is roughly .0026. Two resulting LPV controllers were then examined through a series of simulations in order to observe both power consumption and disturbance rejection capabilities for these controllers. The two controllers demonstrated fast response times and good disturbance rejection. It was also found that the controller with the smaller performance level  $\gamma$  did perform better. Both control systems seemed to show some positive signs of enhanced power consumption. There was chatter involved with certain aspects of the controller input profiles, but the simulations showed evidence that this was not caused by external disturbance, and can be eliminated by proper selection of weighting functions in the control design process.

**Optimal Control for Spacecraft Large Angle Maneuvers Using  $H_\infty$  Linear  
Varying Parameter Control Techniques**

by

**Hunter D. Hughes**

A thesis submitted to the Graduate Faculty of  
North Carolina State University  
in partial fulfillment of the  
requirements for the Degree of  
Master of Science

**Mechanical and Aerospace Engineering**

Raleigh

November 2006

**Approved By:**

---

Dr. Ramasubramanian

---

Dr. Fen Wu  
Chair of Advisory Committee

---

Dr. Larry Silverberg

Dedicated to my parents who have given me so much throughout the course of my education.

## Biography

Hunter Hughes was born in Chattanooga Tennessee to parents Doug and Lynne Hughes on December 26, 1981. He grew up in Chattanooga and attended school there. After high school, he moved to Cookeville Tennessee where he attended Tennessee Technological University. In December of 2004, he received a Bachelor of Science degree in Mechanical Engineering. After graduation, Hunter moved to Raleigh North Carolina to start his work on a Master's of Science degree in Mechanical Engineering.

## Acknowledgements

I would like to take the time here to thank all of the educators who have helped me complete this thesis. Especially Dr. Wu who has been with me every step of the way guiding me and encouraging me. Without his help, I would not have made it this far. I would also like to acknowledge of the people who have taught me over the years, and have inspired me to go into the field of mechanical engineering.

# Contents

<b>List of Figures</b>	<b>vii</b>
<b>List of Tables</b>	<b>ix</b>
<b>Nomenclature</b>	<b>x</b>
<b>1 Introduction</b>	<b>1</b>
1.1 The Importance of Studying Spacecraft Control Problems . . . . .	1
1.2 $H_\infty$ Control Technique . . . . .	2
1.3 LPV Control Techniques . . . . .	6
1.4 Thesis Outline . . . . .	9
<b>2 Spacecraft Modeling and Its LPV Control Designs</b>	<b>10</b>
2.1 Review of Different Control Design Techniques for Spacecraft . . . . .	10
2.2 Nonlinear Model of Spacecraft . . . . .	11
2.3 LPV Spacecraft Model . . . . .	13
2.4 LPV Control Designs for Spacecraft . . . . .	13
<b>3 Simulation Results</b>	<b>21</b>
3.1 Overview . . . . .	21
3.2 Simulations Without Disturbance In The Plant Dynamics . . . . .	22
3.3 Simulations With Disturbance In The Plant Dynamics . . . . .	27
<b>4 Conclusions</b>	<b>38</b>
4.1 Contributions of This Research Work . . . . .	38
4.2 Future Work . . . . .	38
<b>Bibliography</b>	<b>40</b>
<b>Appendices</b>	<b>42</b>
<b>A <math>H_\infty</math> LPV Controller Solution Code</b>	
mytry.m	<b>43</b>
<b>B mkbasis2.m</b>	<b>45</b>

<b>C</b>	<b>slpvsyn_sf.m</b>	<b>47</b>
<b>D</b>	<b>mkslpvk_sf.m</b>	<b>50</b>
<b>E</b>	<b>stability_test.m</b>	<b>51</b>
<b>F</b>	<b>plant_temp.m</b>	<b>52</b>
<b>G</b>	<b>control3_temp.m</b>	<b>55</b>

# List of Figures

1.1	Block Diagram of Uncontrolled System . . . . .	3
1.2	Block Diagram of Controlled System . . . . .	4
2.1	Simulink Block Diagram . . . . .	18
3.1	The Response of System One Due to an Initial Impulse Without Disturbance	22
3.2	The Control Input of System One Due to an Initial Impulse Without Disturbance . . . . .	23
3.3	The Angle of System One Due to an Initial Impulse Without Disturbance .	24
3.4	The Response of System Two Due to an Initial Impulse Without Disturbance	25
3.5	The Control Input of System Two Due to an Initial Impulse Without Disturbance . . . . .	26
3.6	The Angle of System Two Due to an Initial Impulse Without Disturbance .	27
3.7	The Response of System One Due to an Initial Displacement Without Disturbance . . . . .	28
3.8	The Control Input of System One Due to an Initial Displacement Without Disturbance . . . . .	29
3.9	The Angle of System One Due to an Initial Displacement Without Disturbance	30
3.10	The Response of System Two Due to an Initial Displacement Without Disturbance . . . . .	30
3.11	The Control Input of System Two Due to an Initial Displacement Without Disturbance . . . . .	31
3.12	The Angle of System Two Due to an Initial Displacement Without Disturbance	31
3.13	The Response of System One Due to an Initial Impulse With Disturbance .	32
3.14	The Control Input of System One Due to an Initial Impulse With Disturbance	32
3.15	The Angle of System One Due to an Initial Impulse With Disturbance . . .	33
3.16	The Response of System Two Due to an Initial Impulse With Disturbance .	33
3.17	The Control Input of System Two Due to an Initial Impulse With Disturbance	34
3.18	The Angle of System Two Due to an Initial Impulse With Disturbance . . .	34
3.19	The Response of System One Due to an Initial Displacement With Disturbance	35
3.20	The Control Input of System One Due to an Initial Displacement With Disturbance . . . . .	35
3.21	The Angle of System One Due to an Initial Displacement With Disturbance	36

3.22	The Response of System Two Due to an Initial Displacement With Disturbance	36
3.23	The Control Input of System Two Due to an Initial Displacement With Disturbance . . . . .	37
3.24	The Angle of System Two Due to an Initial Displacement With Disturbance	37

# List of Tables

2.1	Table of $\gamma$ values . . . . .	17
-----	------------------------------------	----

# Nomenclature

$\gamma$	$H_\infty$ performance index
$\hat{e}$	Unit vector in the direction of the principal axis of rotation
$\nu$	Parameter Variation Rate
$\omega$	Angular velocity
$\bar{\nu}_k$	Upper bound of parameter variation rate
$\Phi$	The principal angle of rotation
$\rho$	LPV Scheduling parameter vector
$\underline{\nu}_k$	Lower bound of parameter variation rate
$A$	System matrix of the plant
$A_{cl}$	System matrix for the closed loop system
$A_k$	System matrix for the controller
$B$	Input matrix of the plant
$B_{cl}$	Input matrix for the closed loop system
$B_k$	Input matrix for the controller
$C$	Output matrix of the plant
$C_{cl}$	Output matrix for the closed loop system
$C_k$	Output matrix for the controller

$D$	Feedthrough matrix of the plant
$d$	Disturbance vector
$D_{cl}$	Feedthrough matrix for the closed loop system
$D_k$	Feedthrough matrix for the controller
$e$	Output vector
$I$	Identity matrix
$J$	Polar moment of inertia matrix for the spacecraft
$k_1, k_2$	Gain scheduled state feedback gains
$N_R$	The bases of the null spaces of $\begin{bmatrix} B_2^T & D_{12}^T \end{bmatrix}$
$N_S$	The bases of the null spaces of $\begin{bmatrix} C_2 & D_{21} \end{bmatrix}$
$P$	Matrix associated with Lyapunov Function
$p$	Cayley-Rodrigues parameter vector
$q$	Vector of quaternion elements
$r$	Random vector representing noise and disturbance in the spacecraft model
$V$	Lyapunov Function
$x$	State variable vector of the plant
$x_{cl}$	State vector for the closed loop system
$x_k$	State variable matrix for the controller
$\mathcal{P}$	Parameter set
R	Optimization variables for $H_\infty$ problem
S	Optimization variables for $H_\infty$ problem
u	The control input vector
y	The measurement output vector

# Chapter 1

## Introduction

### 1.1 The Importance of Studying Spacecraft Control Problems

Spacecraft have found wide applications in human outer space exploration and other areas. Spacecraft are used on an almost daily basis in modern society today. From cell phones to global positioning systems to television, satellites and other spacecraft provide useful services to people all over the world. With the existing uses for spacecraft and the future of space exploration, it is important to find new and more efficient ways to control these nonlinear systems. Power is an expensive aspect of spacecraft operation. It would prove beneficial to try and create a controller that would improve upon the current power consumption of spacecraft. Disturbance can also be a problem in spacecraft flight. It is often important for spacecraft to undergo some sort of large angle rotation about its principal axis to accomplish a task. This can be seen when a satellite has to transmit something to earth, or when the space shuttle has to dock with a space station. In both of these applications, it is crucial to get a very precise alignment in the rotation because even a very small amount of error could have a very large impact to the functionality of the spacecraft. Since this is the case, it is important to design a controller that increases disturbance rejection capabilities. Spacecraft have inherent nonlinear characteristics that cannot be ignored. Its increasingly stringent operating requirements have continually demanded high performance flight control systems. For example, future commercial and military spacecraft are envisioned to have large angle maneuver capability and high pointing accuracy. In addition, limited power

capabilities necessitates optimal control strategy for long term spacecraft operations. These capabilities and operating requirements are imperative for the demanding tasks of future space missions, such as very large base interferometry, formation flying, and autonomous shuttle docking and servicing.

This study will look at control techniques for spacecraft undergoing rotation around all three principal axes during large angle maneuvers. Currently there are two basic approaches to controlling such a spacecraft. The first technique is a linear approach. This technique involves taking a linear approximation about an equilibrium point and then applying controller gains to the linearized region. The advantage of this technique is that the linearized region is easily optimized for disturbance rejection and is also stabilizable. The drawback of this technique is that the controller is only good for a very small range around the single equilibrium point. This means that large maneuvers cannot be handled by such a control technique. The second technique is a nonlinear approach to the controller design. This technique is in fact a solution throughout the entire range of motion, but it is difficult to optimize the controller for power consumption and disturbance rejection. In addition, it was very difficult to solve for stabilization with nonlinear techniques until recently. The goal of this study is to investigate the potential for combining the two aforementioned control techniques for the purpose of designing a controller that will be applicable to the entire range of motion as well as optimized for improved disturbance rejection and enhanced power consumption.

## 1.2 $H_\infty$ Control Technique

The background for building this type of controller is founded in the  $H_\infty$  and Linear Parameter Varying (LPV) control theory. Therefore, it will be important to understand the basic concepts of  $H_\infty$  control for the purposes of understanding this report. The  $H_\infty$  control problem can be solved using linear matrix inequality (LMI) techniques [5]. Using the system seen in figure 1.1, the Linear Time Invariant (LTI) system equations of  $G$  is

$$\dot{x}(t) = Ax(t) + Bd(t), \quad (1.1)$$

$$e(t) = Cx(t) + Dd(t), \quad (1.2)$$

where the state  $x \in \mathbf{R}^n$ , the disturbance  $d \in \mathbf{R}^{n_d}$  and controlled output  $e \in \mathbf{R}^{n_e}$ .

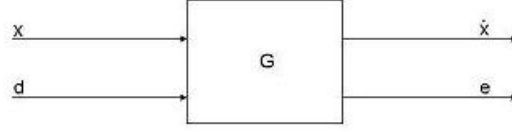


Figure 1.1: Block Diagram of Uncontrolled System

Assuming that the matrix  $A$  is asymptotically stable by letting the eigenvalue of  $A$  be less than zero, it is possible to determine the  $H_\infty$  performance from  $d$  to  $e$ . The  $H_\infty$  norm of the system  $G$  1.1-1.2 is defined as

$$\|G\|_\infty = \max_{d, \|d\|_2 \neq 0} \frac{\|e\|_2}{\|d\|_2}, \quad (1.3)$$

where the signal  $e$ 's 2-norm is  $\|e\|_2^2 = \int_0^\infty e^T(t)e(t)dt$  and similar for  $d$ .

Using the following Lyapunov function

$$V(x) = x^T P x, \quad P = P^T > 0, \quad (1.4)$$

to analyze the stability and system performance, the following equation will be yielded

$$\dot{V} + \frac{1}{\gamma} e^T e - \gamma d^T d < 0. \quad (1.5)$$

Since  $\frac{dV}{dt} = \frac{\partial V}{\partial x} \dot{x}$ , then the above equation becomes

$$\dot{x}^T P x + x^T P \dot{x} + \frac{1}{\gamma} e^T e - \gamma d^T d < 0,$$

which can be reduced to,

$$\|e\|_2 < \gamma \|d\|_2. \quad (1.6)$$

Therefore, the  $H_\infty$  norm of the system is bounded from above by  $\gamma$ .

From this equation, it can be seen that the smaller  $\gamma$  is, the greater the energy amplification of the system becomes. This will be important as one of the stated goals is to decrease power consumption. To minimize the  $\gamma$ , apply the following linear matrix inequality by finding a positive definite matrix  $P > 0$  such that

$$\begin{bmatrix} A^T P + P A & P B & C^T \\ B^T P & -\gamma I & D^T \\ C & D & -\gamma I \end{bmatrix} < 0. \quad (1.7)$$

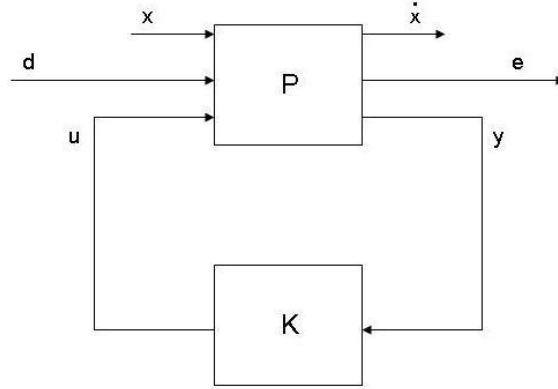


Figure 1.2: Block Diagram of Controlled System

This is known as the bounded real Lemma and is given by an LMI. LMI is a special type of convex problem which can be solved using efficient interior-point optimization techniques [2].

For the  $H_\infty$  control problem, assume that the open loop plant is in the following form

$$\dot{x}(t) = Ax(t) + B_1d(t) + B_2u(t), \quad (1.8)$$

$$e(t) = C_1x(t) + D_{11}d(t) + D_{12}u(t), \quad (1.9)$$

$$y(t) = C_2x(t) + D_{21}d(t) + D_{22}u(t), \quad (1.10)$$

where the control input  $u \in \mathbf{R}^{n_u}$  and the measurement output  $y \in \mathbf{R}^{n_y}$ . It is assumed that the triple  $(A, B_2, C_2)$  is stabilizable and detectable.

Now that the open loop plant has been defined, it is necessary to define the closed loop plant in order to solve the  $H_\infty$  optimization problem. This closed loop system can be seen in figure 1.2.

It is desirable to design an output feedback controller with the following set of equations.

$$\dot{x}_k(t) = A_kx_k(t) + B_ky(t), \quad (1.11)$$

$$u(t) = C_kx_k(t) + D_ky(t), \quad (1.12)$$

where  $x_k \in \mathbf{R}^{n_k}$  is the controller states. In order to find the optimized controller, it will be necessary to optimize the closed loop  $H_\infty$  performance using a dynamic controller. This optimized  $\gamma$  will be referred to as  $\gamma_{opt}$ . This problem will now need to be formulated as a Linear Matrix Inequality (LMI) optimization problem as well.

For a dynamic output feedback control in the form of equations 1.11-1.12, the closed loop system becomes,

$$\dot{x}_{cl}(t) = A_{cl}x_{cl}(t) + B_{cl}d(t), \quad (1.13)$$

$$e(t) = C_{cl}x_{cl}(t) + D_{cl}d(t), \quad (1.14)$$

where  $x_{cl} = \begin{bmatrix} x^T & x_k^T \end{bmatrix}^T$  and the state space matrices of the closed loop system are,

$$A_{cl} = \begin{bmatrix} A + B_2D_kC_2 & B_2C_k \\ B_kC_2 & A_k \end{bmatrix}, \quad (1.15)$$

$$B_{cl} = \begin{bmatrix} B_1 + B_2D_kD_{12} \\ B_kD_{12} \end{bmatrix}, \quad (1.16)$$

$$C_{cl} = \begin{bmatrix} C_1 + D_{12}D_kC_2 & D_{12}C_k \end{bmatrix}, \quad (1.17)$$

$$D_{cl} = D_{11} + D_{12}D_kD_{12}. \quad (1.18)$$

From this set of equations, an optimal controller can be found using  $H_\infty$  analysis condition 1.7. In order to solve the LMI optimization problem, the following three conditions must also be met to solve the control synthesis problem [5]: find positive definite matrices  $R, S > 0$  such that

$$\begin{bmatrix} N_R & 0 \\ 0 & I \end{bmatrix}^T \begin{bmatrix} AR + RA^T & RC_1^T & B_1 \\ C_1R & -\gamma I & D_{11} \\ B_1^T & D_{11}^T & -\gamma I \end{bmatrix} \begin{bmatrix} N_R & 0 \\ 0 & I \end{bmatrix} < 0, \quad (1.19)$$

$$\begin{bmatrix} N_S & 0 \\ 0 & I \end{bmatrix}^T \begin{bmatrix} A^T S + SA & SB_1 & C_1^T \\ B_1^T S & -\gamma I & D_{11}^T \\ C_1 & D_{11} & -\gamma I \end{bmatrix} \begin{bmatrix} N_S & 0 \\ 0 & I \end{bmatrix} < 0, \quad (1.20)$$

$$\begin{bmatrix} R & I \\ I & S \end{bmatrix} \geq 0, \quad (1.21)$$

where  $N_R$  and  $N_S$  are the bases of the null spaces of  $\begin{bmatrix} B_2^T & D_{12}^T \end{bmatrix}$  and  $\begin{bmatrix} C_2 & D_{21} \end{bmatrix}$  which are defined as

$$N_R = \ker \begin{bmatrix} B_2^T & D_{12}^T \end{bmatrix},$$

$$N_S = \ker \begin{bmatrix} C_2 & D_{21} \end{bmatrix}.$$

The control synthesis condition 1.19-1.21 is again in the form of LMIs and can be solved using a commercial software LMIlab [6]. This concludes the  $H_\infty$  synthesis of the LTI system. The next section will discuss how to take these results for LTI systems and extend them into LPV formulations.

### 1.3 LPV Control Techniques

Now that the  $H_\infty$  synthesis conditions have been defined, it is important to look at how this parameterization affects the control problem, and the synthesis conditions. The LPV system is a class of linear systems with its state space matrices depending on a time-varying vector  $\rho(t) \in \mathbf{R}^s$ ,

$$\dot{x}(t) = A(\rho(t))x(t) + B_1(\rho(t))d(t) + B_2(\rho(t))u(t), \quad (1.22)$$

$$e(t) = C_1(\rho(t))x(t) + D_{11}(\rho(t))d(t) + D_{12}(\rho(t))u(t), \quad (1.23)$$

$$y(t) = C_2(\rho(t))x(t) + D_{21}(\rho(t))d(t) + D_{22}(\rho(t))u(t), \quad (1.24)$$

It is assumed that the scheduling parameter  $\rho$  evolves continuously over time and its range is limited to a compact set  $\rho \in \mathcal{P}$ . In addition, its time derivative is often assumed to be bounded and satisfy  $\underline{\nu}_k < \dot{\rho}_k < \bar{\nu}_k$ ,  $k = 1, 2, \dots, s$ . Moreover, assume that

**(A1)** The matrix function triple  $(A(\rho), B_2(\rho), C_2(\rho))$  is parameter-dependent stabilizable and detectable,

**(A2)** The matrices  $\begin{bmatrix} C_2(\rho) & D_{21}(\rho) \end{bmatrix}$  and  $\begin{bmatrix} B_2^T(\rho) & C_{12}^T(\rho) \end{bmatrix}$  have full row rank for all  $\rho \in \mathcal{P}$ ,

**(A3)**  $D_{11}(\rho) = 0$  and  $D_{22}(\rho) = 0$ .

Similar to the open loop description, the LPV synthesis conditions also change with parameterization. For full state feedback,  $y = x$ , we consider the static state feedback control law in the form of

$$u = F(\rho)x \quad (1.25)$$

The synthesis condition of LPV state feedback control problem is given by

$$\begin{aligned} \begin{bmatrix} N_R(\rho) & 0 \\ 0 & I \end{bmatrix}^T & \begin{bmatrix} \left\{ \begin{array}{l} A(\rho)R(\rho) + R(\rho)A^T(\rho) \\ -\sum_{i=1}^s \{\mathcal{L}_i, \bar{\nu}_i\} \frac{\partial R}{\partial \rho_i} \end{array} \right\} & R(\rho)C_1^T(\rho) & B_1(\rho) \\ C_1(\rho)R(\rho) & -\gamma I & D_{11}(\rho) \\ B_1^T(\rho) & D_{11}^T(\rho) & -\gamma I \end{bmatrix} \\ & \times \begin{bmatrix} N_R(\rho) & 0 \\ 0 & I \end{bmatrix} < 0 \end{aligned} \quad (1.26)$$

for any  $\rho \in \mathcal{P}$ . Consequently, the resulting LPV state feedback control gain will be

$$F(\rho) = -(D_{12}^T(\rho)D_{12}^{-1}(\rho))^{-1} [\gamma B_2^T(\rho)R^{-1}(\rho) + D_{12}^T(\rho)C_1(\rho)]. \quad (1.27)$$

For the LPV output feedback control problem, the following controller formulation is needed.

$$\dot{x}_k(t) = A_k(\rho(t), \dot{\rho}(t))x_k(t) + B_k(\rho(t))y(t), \quad (1.28)$$

$$u(t) = C_k(\rho(t))x_k(t) + D_k(\rho(t))y(t). \quad (1.29)$$

Using a parameter-dependent quadratic Lyapunov function  $V(x) = x_{cl}^T P(\rho)x_{cl}$ , the solution of  $H_\infty$  synthesis problem of an LPV output feedback controller [1, 14] is to find a pair

of continuously differentiable matrix functions  $R(\rho)$ ,  $S(\rho) > 0$  which satisfy

$$\begin{bmatrix} N_R(\rho) & 0 \\ 0 & I \end{bmatrix}^T \begin{bmatrix} \left\{ \begin{array}{l} A(\rho)R(\rho) + R(\rho)A^T(\rho) \\ - \sum_{i=1}^s \{\underline{\nu}_i, \bar{\nu}_i\} \frac{\partial R}{\partial \rho_i} \end{array} \right\} & R(\rho)C_1^T(\rho) & B_1(\rho) \\ C_1(\rho)R(\rho) & -\gamma I & D_{11}(\rho) \\ B_1^T(\rho) & D_{11}^T(\rho) & -\gamma I \end{bmatrix} \\ \times \begin{bmatrix} N_R(\rho) & 0 \\ 0 & I \end{bmatrix} < 0, \quad (1.30)$$

$$\begin{bmatrix} N_S(\rho) & 0 \\ 0 & I \end{bmatrix}^T \begin{bmatrix} \left\{ \begin{array}{l} A^T(\rho)S(\rho) + S(\rho)A(\rho) \\ + \sum_{i=1}^s \{\underline{\nu}_i, \bar{\nu}_i\} \frac{\partial S}{\partial \rho_i} \end{array} \right\} & S(\rho)B_1(\rho) & C_1^T(\rho) \\ B_1^T(\rho)S(\rho) & -\gamma I & D_{11}^T(\rho) \\ C_1(\rho) & D_{11}(\rho) & -\gamma I \end{bmatrix} \\ \times \begin{bmatrix} N_S(\rho) & 0 \\ 0 & I \end{bmatrix} < 0, \quad (1.31)$$

$$\begin{bmatrix} R(\rho) & I \\ I & S(\rho) \end{bmatrix} \geq 0, \quad (1.32)$$

for all  $\rho \in \mathcal{P}$ . Similar to the LTI case,

$$\begin{aligned} N_R(\rho) &= \ker \begin{bmatrix} B_2^T(\rho) & D_{12}^T(\rho) \end{bmatrix}, \\ N_S(\rho) &= \ker \begin{bmatrix} C_2(\rho) & D_{21}(\rho) \end{bmatrix}. \end{aligned}$$

Now that the equations of motion, and the LPV solution techniques have been established, it is possible to find a solution to a given problem. It should be noted here that the synthesis conditions and closed loop dynamics of the plant are still LTI even though they have all been parameterized. This means that the LMI's and closed loop equations involve an infinite number constraints. To simplify this problem, the parameters will be discretized. This means that a finite number of points for each parameter in the system will be chosen to represent the system as a whole. It is important to realize that each of these discretized parameter points will represent a linear model of the system. These gridding points will be chosen in a way such that the discrete points are so close that the range for which each linearized region is valid overlaps with another linearized gridding point. In addition,  $R(\rho)$  and  $S(\rho)$  must also be parameterized using a finite number of basis functions [1]. This will

take the form of,

$$R(\rho) = \sum_{i=1}^{N_f} f_i(\rho)R_i, \quad (1.33)$$

$$S(\rho) = \sum_{j=1}^{N_g} g_j(\rho)S_j, \quad (1.34)$$

where  $f_i(\rho), i = 1, 2, \dots, N_f$  and  $g_j(\rho), j = 1, 2, \dots, N_g$  are user specified basis functions. Once this parameterization of the system has taken place, the synthesis problem can now be solved.

## 1.4 Thesis Outline

This thesis will explore the solution technique used to solve a nonlinear system using LPV and  $H_\infty$  control techniques. The next chapter in this report will discuss the control problem as well as the synthesis conditions for a spacecraft undergoing rotation about its principal axes. Chapter three will discuss the process for both computing the optimized controller as well as the computer simulation to test the controller. Chapter four will cover the conclusions from this study.

## Chapter 2

# Spacecraft Modeling and Its LPV Control Designs

### 2.1 Review of Different Control Design Techniques for Spacecraft

From existing literatures, there are several current techniques for solving the optimal nonlinear spacecraft control. Before discussing the solution approach used in this study, it will help the reader to look at past control techniques for the nonlinear spacecraft model. These control techniques are divided into two main strategies, nonlinear techniques and linear techniques.

First there are the nonlinear theories. These nonlinear theories are very difficult to work with because of the complexity involved with optimizing the nonlinear Lyapunov functions. In order to optimize a nonlinear system, one must apply the  $H_\infty$  analysis to the system. Doing this requires solving the Hamilton-Jacobi-Isaacs inequality [7]. This inequality is a partial differential inequality, and is difficult to solve for complex Lyapunov functions. In order to solve these inequalities, certain assumptions are made to simplify the Lyapunov functions [3]. These assumptions result in a suboptimal solution for the spacecraft. This method is cumbersome and numerically difficult to solve.

There is also a technique in which the inverse optimization is used [8]. This technique uses the modified Rodrigues parameters to define the spacecraft model. However, this technique

does not use the  $H_\infty$  optimization, but rather solves the nonlinear controller using stabilizing Lyapunov functions. Once the controller has been solved for, the inverse optimization technique finds the cost criteria for which the solved controller has been optimized. This is not a desirable technique because it is not possible to state what the desired optimized cost criteria is before the controller is solved.

In addition to nonlinear techniques, there are several types of linear techniques [13, 4]. The most basic linear technique involves solving the controller for the spacecraft near a certain equilibrium point by applying a Taylor series expansion to the system and neglecting all of the higher order terms. This solution is easy to work with, but it is only stable over a very small region around the point of linearization. To account for this small region of stability, the LPV approach has been used to parameterize a larger region of space in order to create a series of linear controllers to stabilize the entire region of motion. In the past, this approach has been done successfully using the quaternion as the parameterization variable [9]. This technique is suitable, but until recently, has not incorporated the  $H_\infty$  technique to optimize the controller performance. This study is an attempt to improve upon these previous design attempts.

## 2.2 Nonlinear Model of Spacecraft

The particular spacecraft and dynamics used for this project are set up so that

$$\omega = [\omega_1 \quad \omega_2 \quad \omega_3]^T \quad (2.1)$$

and

$$\tilde{\omega} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}, \quad (2.2)$$

where  $\omega$  is a vector of the rotational velocities about the principal axes [9]. Now let  $J$  be the matrix values of the polar moments of inertia and  $u$  be the control torques. This yields the following equation to govern the dynamics of the system.

$$J\dot{\omega} + \tilde{\omega}J\omega = u. \quad (2.3)$$

To ensure that no singularities are encountered during the rotation of this spacecraft, the quaternion (or Euler Parameters) are applied to the system. The end result of this yields

the following two equations

$$\dot{q} = \frac{1}{2}\tilde{q}\omega + \frac{1}{2}q_4\omega, \quad (2.4)$$

$$\dot{q}_4 = \frac{1}{2}q^T\omega, \quad (2.5)$$

where  $q = [q_1 \ q_2 \ q_3]^T$  and  $q_1^2 + q_2^2 + q_3^2 + q_4^2 = 1$ . Simplifying these equations, and applying them to the system dynamics equations yields the following equations [12].

$$\dot{\omega} = -J^{-1}\tilde{\omega}J\omega + J^{-1}u, \quad (2.6)$$

$$\dot{q} = \frac{1}{2}\tilde{q}\omega + \frac{1}{2}q_4\omega, \quad (2.7)$$

$$\dot{q}_4 = -\frac{1}{2}q^T\omega. \quad (2.8)$$

Now that the dynamics of the spacecraft are defined, it is necessary to model the non-linear spacecraft to a Linear Parameter Varying system. Before establishing this system by choosing the varying parameters of the system, the system should be simplified to reduce the intensity of the calculations. In order to simplify this system, the Cayley-Rodrigues parameters were chosen as in [11]. These parameters are defined below.

$$p_1 = \frac{q_1}{q_4}, \quad (2.9)$$

$$p_2 = \frac{q_2}{q_4}, \quad (2.10)$$

$$p_3 = \frac{q_3}{q_4}, \quad (2.11)$$

This gives the modified spacecraft dynamics as,

$$\dot{p} = H(p)\omega, \quad (2.12)$$

where

$$H(p) = \frac{1}{2}(I - \tilde{p} + pp^T), \quad (2.13)$$

$$\tilde{p} = \begin{bmatrix} 0 & -p_3 & p_2 \\ p_3 & 0 & -p_1 \\ -p_2 & p_1 & 0 \end{bmatrix}. \quad (2.14)$$

## 2.3 LPV Spacecraft Model

In order to obtain the LPV system model, treat  $p$  as a scheduling parameter such that  $\rho_1 = p_1$ ,  $\rho_2 = p_2$ ,  $\rho_3 = p_3$ , and  $\rho = \begin{bmatrix} \rho_1 & \rho_2 & \rho_3 \end{bmatrix}^T$ . If  $v = u - \tilde{\omega}J\omega$ , and this transformation is substituted into equation 2.3, then the dynamic equation becomes

$$\dot{\omega} = J^{-1}v, \quad (2.15)$$

or,

$$v = k_1(\rho)\omega + k_2(\rho)p. \quad (2.16)$$

Finally, the kinematic equations of spacecraft are given by

$$\dot{\omega} = J^{-1}v, \quad (2.17)$$

$$\dot{\rho} = H(\rho)\omega. \quad (2.18)$$

It is important to take into consideration at this point the fact that Cayley-Rodrigues parameters cannot account for the full eigen-axis rotation anymore. These parameters are only able to be used for rotations of  $180^\circ$  or less. In the interest of time, this solution technique has been deemed suitable.

This LPV modeling approach is better than previous models because it reduces the number of parameters in the system. In Lu, Wu, and Kim's paper, there are seven different parameters for the system [10]. This large number of parameters causes the computation to become very large and arduous. By reducing the number of parameters, the problem is simplified. This allows one to use a larger number of gridding points in the parameterization of the problem. This larger number of parameter points results in a more optimal solution to the problem for a given range of motion for the spacecraft.

## 2.4 LPV Control Designs for Spacecraft

This section will go through the LPV controller design, discuss a performance comparison between several different LPV controllers with different parameter ranges, and discuss the simulation results found using an LPV controller. Before going any further, it is worth mentioning to the reader that MATLAB 2006a with the Robust Control Toolbox 3.1.1

and Simulink were used extensively in solving the control problem and simulating its results. Also, this study used the High Performance Computing resource available through North Carolina State University. For more information on the hardware capabilities of this resource, please visit "<http://www.ncsu.edu/itd/hpc/Hardware/Hardware.php>", and read about the Henry2 cluster.

For this case study, the inertial matrix of the spacecraft will be defined below [9].

$$J = \begin{bmatrix} 10000 & 0 & 0 \\ 0 & 9000 & 0 \\ 0 & 0 & 12000 \end{bmatrix} kg - m^2. \quad (2.19)$$

Once the inertial matrix is established, it is important to determine what the different states of the controller are going to be. For this controller, full state feedback was chosen. (Note that for the full state feedback condition,  $S = 0$ .) The number of control states and the number of measured states were equal to three in this case. After this has been established, it is necessary to determine what the range of the parameters needs to be for the system. This poses a very delicate problem. If too many gridding points are chosen, then the calculations to solve the controller become very difficult for a computer to handle, and at the same time if too few points are chosen, then the controller will behave in a less efficient manner or become unstable. In addition to this, it is also important to remember that the Cayley-Rodrigues parameters only allow for a rotation less than  $180^\circ$  in either direction. As a starting point for this study  $\rho$  was chosen such that  $-\frac{1}{3} < \rho_1 < \frac{1}{3}$ ,  $-\frac{1}{3} < \rho_2 < \frac{1}{3}$ , and  $-\frac{1}{3} < \rho_3 < \frac{1}{3}$ . Since

$$\rho = \hat{e} \tan \frac{\Phi}{2}, \quad (2.20)$$

where  $\hat{e}$  is the unit vector in the direction of the principal axis of rotation, and  $\Phi$  is the principal angle of rotation. Since  $p_1$ ,  $p_2$ , and  $p_3$  correspond to  $\rho_1$ ,  $\rho_2$ , and  $\rho_3$  accordingly, and also

$$p_1^2 + p_2^2 + p_3^2 = \frac{1}{q_4^2} - 1 \quad (2.21)$$

with

$$q_4 = \cos \frac{\Phi}{2} \quad (2.22)$$

it can be seen that this choice of parameter values results in  $-\frac{\Pi}{3} \leq \Phi \leq \frac{\Pi}{3}$ . Through experimentation, it was discovered that the maximum amount of points that the parameter range could be subdivided into was seven. If more parameter points than this were chosen, the computer was unable to solve the system due to a lack of RAM (Random Access Memory). Choosing this number of gridding points generated  $7^3$  or 343 gridding points. Each of these gridding points represents a single point in space where the system has been linearized. This means that the system will operate efficiently as long as these linearized points are located close enough together. If the points are too far apart from each other, then the system could behave inefficiently as it goes from one point on the grid to the next, and in an extreme case could result in either an uncontrollable or unstabilizable system.

Now that the parameters for the system have been determined, it will be necessary to establish the state space equations for the open loop system. These equations are as follows,

$$A = \begin{bmatrix} 0 & 0 \\ \frac{(I-\rho+\rho\rho^T)}{2} & 0 \end{bmatrix}, \quad (2.23)$$

$$B = \begin{bmatrix} 0.01 \times J^{-1} & J^{-1} \\ 0 & 0 \end{bmatrix}, \quad (2.24)$$

$$C = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix}, \quad (2.25)$$

$$D = \begin{bmatrix} 0 & 0 \\ 0 & 0.1 \times I \end{bmatrix}, \quad (2.26)$$

where  $\rho$  will be a vector of all parameter points. The LPV and  $H_\infty$  Controller Solution MATLAB Code in Appendix A shows how this controller was solved for in detail. In the current design, a constant weighting function was used to penalize the controlled output and input force. A parametrically dependent weight function would be needed to get a more accurate result, but in the interest of time this function has been deemed suitable.

Having established the state space equations for the open loop system for every parameter point, it is now time to find  $R(\rho)$  and  $\gamma$ . Before this can be done however, the basis function for  $R(\rho)$  must first be found. If it is assumed that  $R(\rho)$  is parameterized in the form of

$$R(\rho) = R_0 + \rho_1 R_1 + \rho_2 R_2 + \rho_3 R_3, \quad (2.27)$$

then the corresponding basis function vector  $f(\rho)$  takes the form of

$$f(\rho) = \begin{bmatrix} 1 & \rho_1 & \rho_2 & \rho_3 \end{bmatrix} \quad (2.28)$$

for all parameter points. It will also be necessary to calculate the gradient of this vector,  $\frac{\partial f(\rho)}{\partial \rho}$ . Both of these calculations are done in the `mkbasis2` MATLAB code that can be found in Appendix B. Before  $R(\rho)$  and  $\gamma$  can be calculated, there is still one constant that must be set. This constant is  $\nu$ , which is the bound that is placed on  $\rho$  from equation 2.20.  $\nu$  can be defined as

$$|\dot{\rho}| \leq \nu. \quad (2.29)$$

For this case,  $\nu = [0.5 \ 0.5 \ 0.5]^T$ . This bound simply controls the rate at which  $\rho$  changes. Now that the basis function and constants have been set, all of the criteria is met to solve for  $R(\rho)$  and  $\gamma$ . This was done using the MATLAB code `slpvsyn_sf` in Appendix C.

Now that the  $R(\rho)$  and  $\gamma$  have been solved, the full state feedback LPV controller with  $H_\infty$  optimization can now be found. This calculation is done using the `mkslpvk_sf` MATLAB code found in Appendix D. This code will generate the controller for the system based off of  $R(\rho)$ ,  $\gamma$ , and  $\nu$ . This controller will be used in the simulation of this spacecraft as it undergoes principal axis rotation.

Before the simulation is discussed however, it is important to take a look at the trend of the  $\gamma$  for different parameter choices. Since the  $H_\infty$  optimization is based off of the reduction of  $\gamma$ , it is important to run this set of calculations for different parameter choices to see the different  $\gamma$  values. The data for these different parameter choices can be found in Table 2.1. In this table, six different parameter ranges were chosen. In each parameter range, seven equidistant points were taken. The values of  $\gamma$  suggest that as the range decreases from  $148^\circ$  to  $25^\circ$  the  $\gamma$  value also decreases. This decreased  $\gamma$  means an increase in the performance with respect to disturbance rejection and power consumption. This would seem to suggest that on this range of values at least, the spacecraft is in fact behaving in a desired fashion. The problem comes into play when one considers the fact that  $\gamma$  decreases again as the range of  $\Phi$  goes from  $148^\circ$  to  $167^\circ$ . The reason for this occurrence is that the gridding points for the range of  $\Phi = 167^\circ$  are too far apart. Since only seven points can be taken for each parameter resulting in a total of 343 gridding points, the points are spaced too

Table 2.1: Table of  $\gamma$  values

$\Phi$	$\rho$	$\gamma$
$-25^\circ$ to $25^\circ$	$-\frac{1}{8} \leq \rho \leq \frac{1}{8}$	0.0026
$-60^\circ$ to $60^\circ$	$-\frac{1}{3} \leq \rho \leq \frac{1}{3}$	0.0027
$-82^\circ$ to $82^\circ$	$-\frac{1}{2} \leq \rho \leq \frac{1}{2}$	0.004
$-120^\circ$ to $120^\circ$	$-1 \leq \rho \leq 1$	0.0131
$-148^\circ$ to $148^\circ$	$-2 \leq \rho \leq 2$	0.0308
$-167^\circ$ to $167^\circ$	$-5 \leq \rho \leq 5$	0.0069

far apart for an optimal solution. Though it is not possible to solve this system with more gridding points, it is possible to run a quick stability test for this case. This was done using the `stability_test` file found in Appendix E. This test shows that the system is unstable for only 343 parameter points on this range. This means that the points are too far apart to achieve a suitable solution.

Since the controller has been solved and its optimality has been established, it is important to look at a simulation of this controller. Before going into the details of this simulation, it is important for the reader to note that this simulation is purely theoretical since there were no available resources to test on an actual spacecraft. It will also be important to note that an assumption has been made that the propulsion systems of the spacecraft will be able to generate any desired  $\omega$  value. In reality this may not be the case, but this is simply a preliminary test to see if this solution technique is going to be feasible.

For the simulation of this controller, a mathematical model of both the spacecraft and the controller was built in Simulink. This block diagram took advantage of Simulink's S-function block (for more information about S-function blocks in general, please see the help file in Simulink). One S-function block was used to model the dynamics of the spacecraft, while another S-function block was used to model the controller. The block diagram for this system can be seen in figure 2.1. All of the output from the simulation was saved in data files as can be seen in the figure.



The block diagram deserves some further explanation. First consider the Plant block. This Simulink block models the dynamics of the spacecraft. This block is an S-function block, and so the code for this block is defined by the user. For this problem, the plant dynamics code `plant_temp` can be found in Appendix F. This block has a user input which is the initial conditions for the state vector. The code has three essential elements to it. These elements are the initialization of the program, the derivatives of the system, and the output of the system. The initialization section simply sets up the sizes of the variables in the program. It also reads in the initial conditions which must be defined by the user. The initialization section of the code is only run at the startup of the simulation, after that it is no longer necessary. The derivatives of the system are calculated in the next section of the program. These derivatives are defined as  $\begin{bmatrix} \dot{\omega} & \dot{p} \end{bmatrix}^T$ . From equations 2.17 and 2.18, it can be seen that this is the description of the system dynamics. These  $[\omega^T \ p^T]^T$  values are then sent to the output section of code. In the output section of the code, the derivatives are sent as the output from the plant block to the input of the control block. The control block code `control3_temp` can be seen in Appendix G. The control block has five user inputs, and of course the input from the plant block. The control code has two main sections, the initialization and the output. Since this code is designed for full state feedback, there is no need for a derivative section in this code. The initialization section for this code is very similar to that of the plant code. The initialization sets up the sizes of all the variables to be used in the code as well as reading in the user input to the system. The user inputs are the open loop state space matrices, the number of measured states, the number of controlled states,  $\gamma$ , and  $R(\rho)$ . The number of measured states is constant. For this study, there are six measured states because the controller has full state feedback. The number of controlled states is also constant. In this case, there are three controlled states because there are three thrusters to control the motion of the spacecraft. The other variables are calculated by running the LPV and  $H_\infty$  Controller Solution Code found in Appendix A. This Simulink block will be able to pull these variables from the workspace after having run the LPV and  $H_\infty$  Controller Solution Code found in Appendix A. Once these variables are read, and the initialization is complete, the program will go straight to the output portion of the code. In the output portion of the code, the A matrix of the open loop system will be defined as in equation 2.23, and  $R(\rho)$  is defined as it is in equation 2.27 for the appropriate value of  $\rho$ . These values along with number of control states and  $\gamma$  are input into the `mkslpvk_sf` code. This code then calculates the controlled system matrix,  $A_{cl}$ , at the given value for  $\rho$ . The

controller is then applied to the system to obtain the controller angular velocities using the following equation

$$u = \begin{bmatrix} k_1(p) & k_2(p) \end{bmatrix} \times x - \tilde{\omega}J\omega, \quad (2.30)$$

where  $x$  is the state vector at the given  $\rho$  value, and  $u$  is the control input vector of torques. This  $u$  vector now has the three control torques in it which are the output of the control code. This output is sent to the plant as input, and the plant applies the dynamics of the system with the new input, and the process starts all over again and runs until the designated ending time is reached. For this study, three minutes (180 seconds) of run time was allotted.

## Chapter 3

# Simulation Results

### 3.1 Overview

This chapter is dedicated to discussing the results of the study. First it will be important to see the different tests that have been performed. The LPV and  $H_\infty$  Controller Solution Code has been run with the parameter listed in table 2.1, but after running the simulation using the  $-\frac{1}{3} \leq \rho \leq \frac{1}{3}$  and  $-1 \leq \rho \leq 1$  conditions, it was decided that there was no need to run the simulation again for the other cases. Both the  $-\frac{1}{3} \leq \rho \leq \frac{1}{3}$  (from here on this system will be referred to as System One) and  $-1 \leq \rho \leq 1$  (from here on this system will be referred to as System Two) conditions will be examined for different simulation criteria. This criteria includes looking at each system both with and without disturbance introduced into the plant dynamics, looking at the system with an initial impulse, and looking at the system with an initial displacement. The disturbance in the system is defined as

$$d = .01 \times r, \quad (3.1)$$

and

$$J\dot{\omega} = v + d, \quad (3.2)$$

where  $r$  is a random matrix of size  $(3 \times 1)$ . The initial impulse is defined as  $\omega_i = [.5 \ .5 \ .5]^T$  where  $p_i = 0$ . The initial displacement case is defined as  $p_i = [-\frac{1}{3} \ -\frac{1}{3} \ -\frac{1}{3}]^T$  for System One and as  $p_i = [-1 \ -1 \ -1]^T$  for System Two where  $\omega_i = 0$  for both systems. A total of eight different simulations were run and will be examined in this chapter.

## 3.2 Simulations Without Disturbance In The Plant Dynamics

First consider the systems without any disturbance to the plant. The response of these systems will give insight to the nature of the controller in regards to both response time and its power consumption. For System One, consider the response of the system due to an initial impulse to the system. Figure 3.1 shows the response of this system.

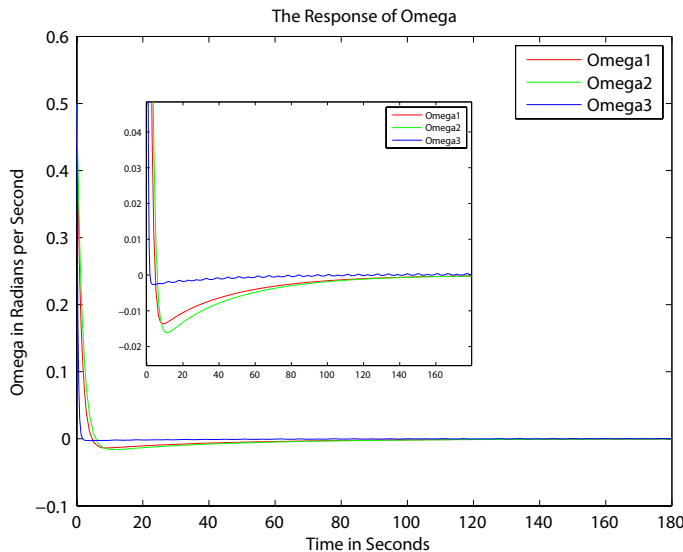


Figure 3.1: The Response of System One Due to an Initial Impulse Without Disturbance

Notice that the angular velocities of the spacecraft go to zero very quickly. This would suggest that the system has an acceptable response time. Now look at the controller forces for this simulation as can be seen in figure 3.2.

From this figure, one can see that there are a couple of concerns about the control forces. The first concern is that the initial torques that need to be generated by the actuators is large. It may not be likely that  $-18000$  Nm is achievable, but this is not the main concern of this paper as stated in earlier chapters. What is of greater concern is the fact that the controller output for the torque associated with the third axis does not seem to be asymptotically stable. There seems to be a considerable amount of chatter on this particular

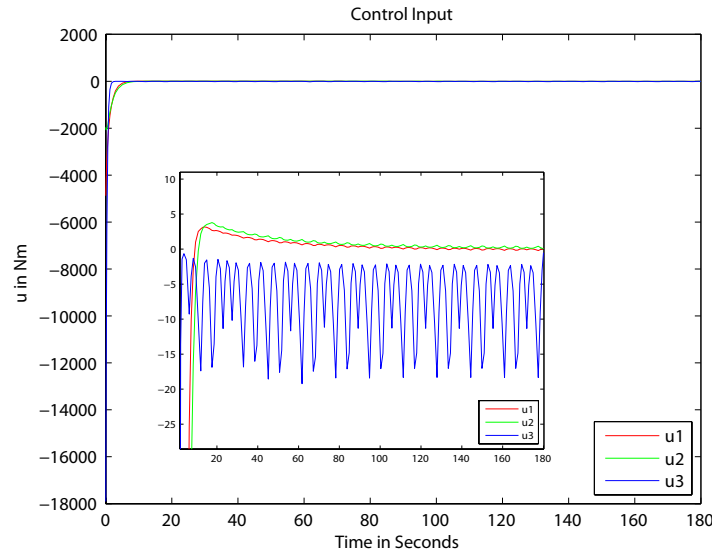


Figure 3.2: The Control Input of System One Due to an Initial Impulse Without Disturbance

control input torque, and a slight amount of chatter on the other control inputs. The reason for this is not certain, but as seen from figure 3.1, the response of the system still seems to be acceptable. The power consumption for this controller is good. The output of the controller seems to suggest that there will be a large controller force applied for a short period of time, and then a small controller force applied for a long period of time. This would suggest that the area under this curve is small, and thus the required control force is small. From figure 3.3, it can be seen that the angle of rotation does regulate about the principal axes.

For this case,  $\Phi = 0$  radians and then the spacecraft undergoes a rotation due to the impulse to the system. Once the controller takes effect, the system asymptotically approaches zero. From the three plots for this system, it can be seen that this system does in fact respond to the impulse in a suitable fashion.

Now look at the same set of conditions for System Two. Figure 3.4 shows the response of the angular velocities of the system. This system has a lower peak overshoot than what is seen in System One for the same set of conditions.

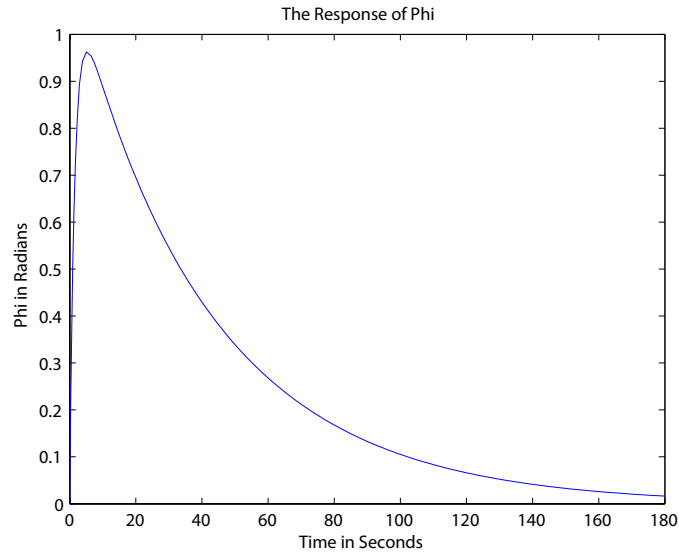


Figure 3.3: The Angle of System One Due to an Initial Impulse Without Disturbance

In figure 3.5, the control inputs for System Two can be seen for the initial impulse to the system. As with System One, there is some chatter in the control inputs. With this system however, the chatter is significant not only on  $u_3$ , but also on  $u_1$  and  $u_2$  as well. Also, it is important to see that the control forces could be higher than any achievable force. It is interesting to see here that the overshoot of the control inputs is less for System Two than it is for System One.

In figure 3.6,  $\Phi$  for System Two can be seen. Similar to System One, the angle for System Two is asymptotically approaching zero as well. The main difference to note is that System Two is not approaching zero as quickly as was seen in System One (figure 3.2). This is due mainly to the fact that System Two is less optimal than System One, thus resulting in a faster response time for System One.

Now that the impulse system response has been analyzed for both System One and System Two with no disturbance introduced into the plant dynamics, it is important to consider the system responses to the initial displacement with no disturbance introduced into the plant dynamics. First look at System One. In figure 3.7, the response of the angular velocities can be seen. Since this input to the system is a displacement, the initial velocity of the system

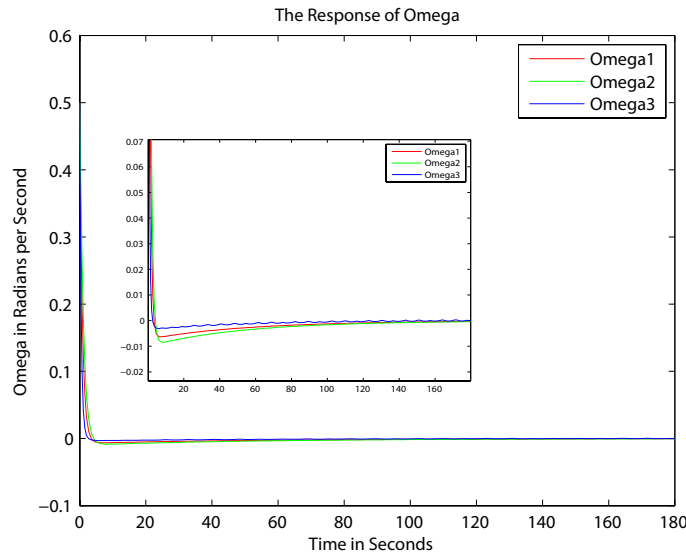


Figure 3.4: The Response of System Two Due to an Initial Impulse Without Disturbance

is zero. From this figure, one can see that the velocity of the system quickly increases until it reaches approximately .014 radians per second and then the velocity quickly approaches zero in an asymptotic fashion. Unlike the impulse response for this same system, the initial displacement response takes a much longer time to settle out. In addition to this, one should take notice that there is no chatter involved with the response of this system. Also, the angular velocities are an order of magnitude lower than with the initial impulse. This lower magnitude could be the reason for the lack of chatter in the response of this system. It is possible that when the velocity of the system is too great, as in the impulse simulation, the system does not have a fast enough response time to gain proper control of the spacecraft before it switches to the next gridding point. Choosing a different weight function could solve this chatter problem.

Figure 3.8 shows the control input to the system. This figure shows that again the controller forces may be high, but notice that they are much lower than the impulse response for this system. Also notice how the chatter is greatly reduced, and that it is only seen on  $u_3$ . This could mean that the chatter is somehow related to the inertial matrix for the system ( $u_3$  having the largest moment of inertia associated with it for this problem). Again, a different weight function could solve the chatter problem. Other than this phenomena,

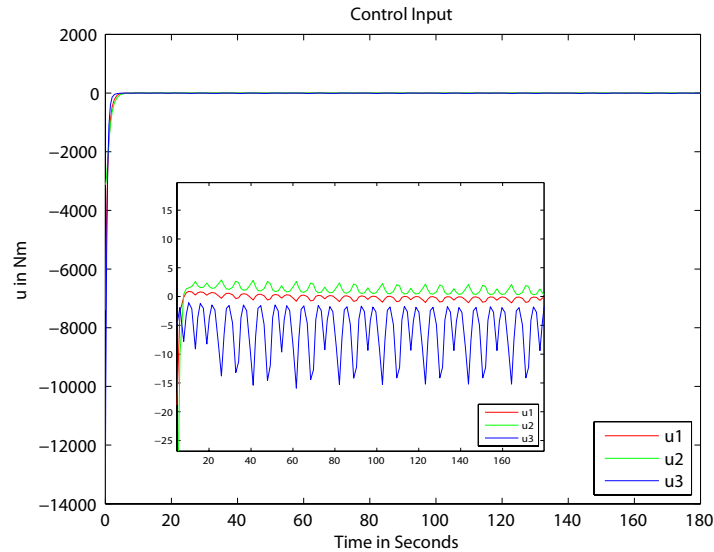


Figure 3.5: The Control Input of System Two Due to an Initial Impulse Without Disturbance

the control input to the system looks feasible. The control force would be high for a short period of time, and then would asymptotically approach zero. This supports the theory that this is an optimal controller.

Figure 3.9 shows angle  $\Phi$  of System One due to an initial displacement. The response of this angle asymptotically approaches zero. This is to be expected for the system.

Now compare the same set of data for the response of System Two to an initial displacement. Figure 3.10 shows the angular velocities of the system. The angular velocity of System Two is twice that of System One. Just as before, the system asymptotically goes to zero, but System Two takes slightly longer to reach zero. This is due to the fact that System Two has a larger displacement than System One.

Figure 3.11 shows the control input to System Two for the initial displacement case. From this figure, the chatter on the  $u_3$  control input can be seen. As with System One, the chatter on this control input appears to be small. Also notice that for System Two the control input values are about twice the size of that for System One.

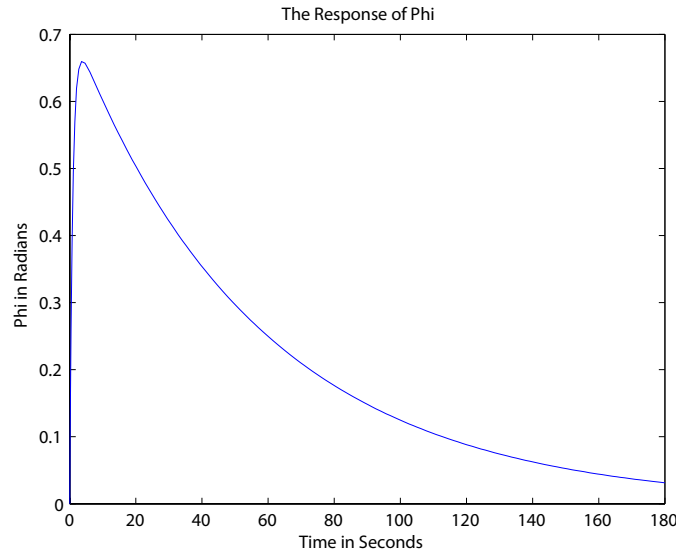


Figure 3.6: The Angle of System Two Due to an Initial Impulse Without Disturbance

Figure 3.12 shows the angle  $\Phi$  for System Two after an initial displacement. The displacement is greater for System Two than it is for System One, therefore the response time of System Two is slightly slower. The angle still approaches zero asymptotically as expected.

### 3.3 Simulations With Disturbance In The Plant Dynamics

Now that the two systems have been analyzed for the two different sets of initial conditions without disturbance to the system, it is time to look at the effect of adding disturbance to the system. Since one of the goals of this improved controller design is to increase disturbance rejection, one would expect to see little or no difference between the responses seen with disturbance and the responses seen without disturbance. The disturbance added to the system is defined in equation 3.1. The MATLAB has a built in random number generator function called `rand()`. This function was used to generate the matrix of random values  $r$  for the disturbance.

Figure 3.13 shows that the disturbance has little to no effect on the response of  $\omega$  for System One when compared to 3.1. When looking at the zoomed in portion on both of these plots, one can see that the graphs are almost identical. Comparing also for System One for

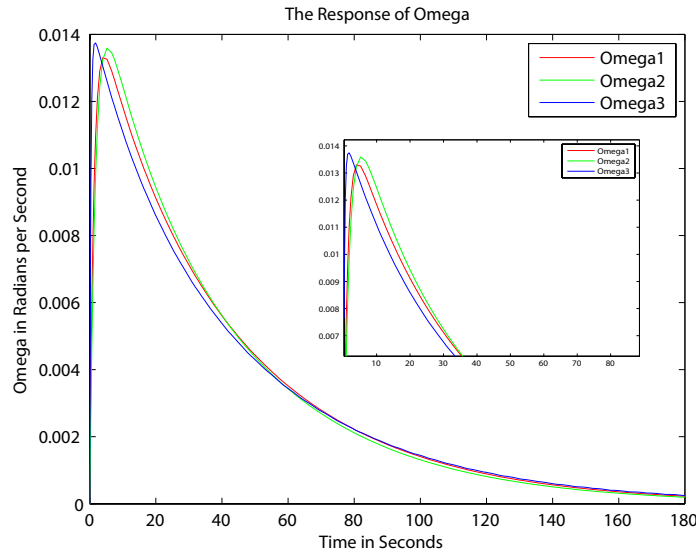


Figure 3.7: The Response of System One Due to an Initial Displacement Without Disturbance

the initial impulse figures 3.14 and 3.15 with figures 3.2 and 3.3 respectively, show that the system behaves in the same fashion both with and without disturbance in the system. The chatter in the system shows up in the same spot for both the disturbed and undisturbed system, so the chatter is mathematical as opposed to a disturbance force.

Now consider System Two with an initial impulse and disturbance introduced to the system. Figure 3.16 shows the response of the  $\omega$  values. When comparing this figure to figure 3.4, it can be seen again that the disturbance introduced to the system has no effect. The plots are identical. Comparing figures 3.17 and 3.18 with 3.5 and 3.6 support this idea. The control output is the same for the simulation run with disturbance as it is in the simulation run without disturbance. The same is true for the  $\Phi$  angle.

Now look at System One with an initial displacement with disturbance introduced to the system. Again as before, it can be seen that figures 3.19, 3.20, and 3.21 are almost identical to figures 3.7, 3.8, and 3.9. Again the data shows that the disturbance has little to no effect on the system.

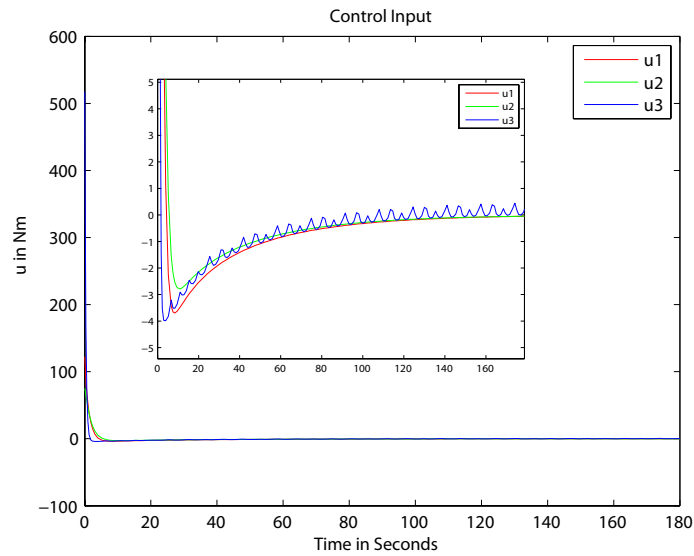


Figure 3.8: The Control Input of System One Due to an Initial Displacement Without Disturbance

Finally consider the last case where System Two has an initial displacement with disturbance introduced to the system. Again as before, it can be seen that figures 3.22, 3.23, and 3.24 are almost identical to figures 3.10, 3.11, and 3.12. Again the data shows that the disturbance has little to no effect on the system.

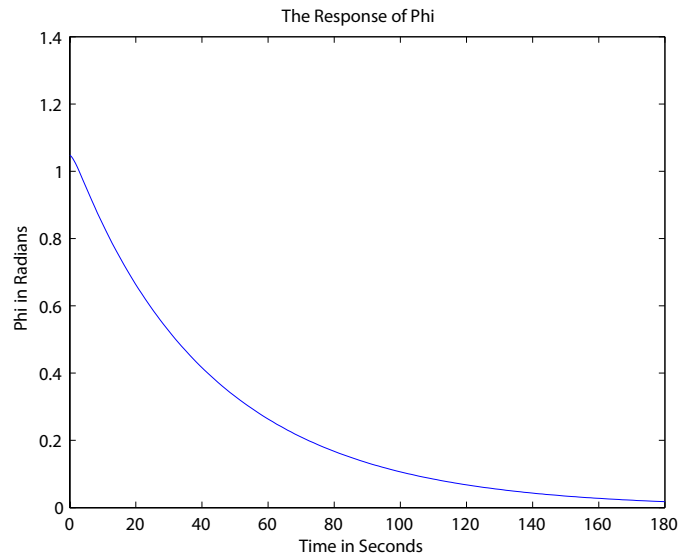


Figure 3.9: The Angle of System One Due to an Initial Displacement Without Disturbance

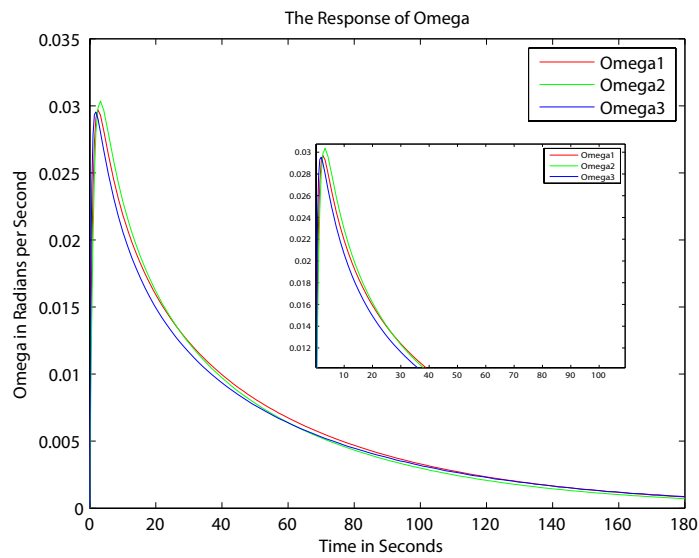


Figure 3.10: The Response of System Two Due to an Initial Displacement Without Disturbance

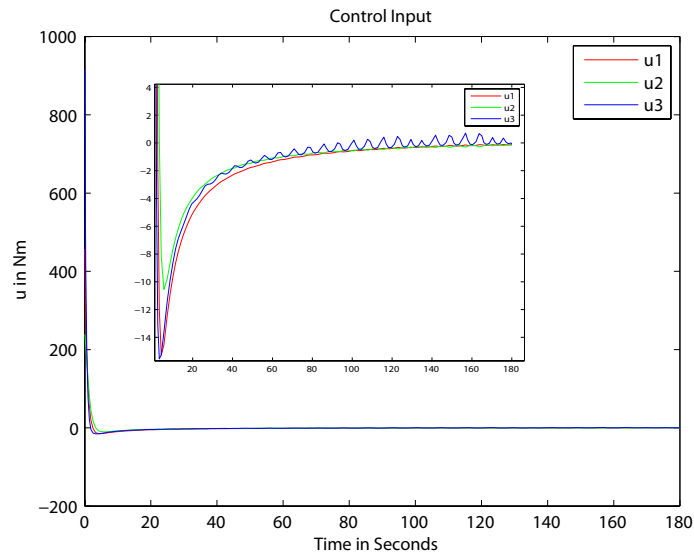


Figure 3.11: The Control Input of System Two Due to an Initial Displacement Without Disturbance

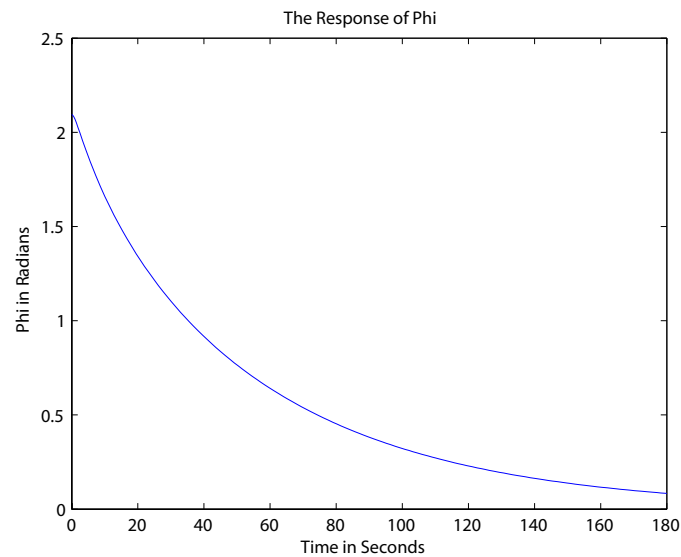


Figure 3.12: The Angle of System Two Due to an Initial Displacement Without Disturbance

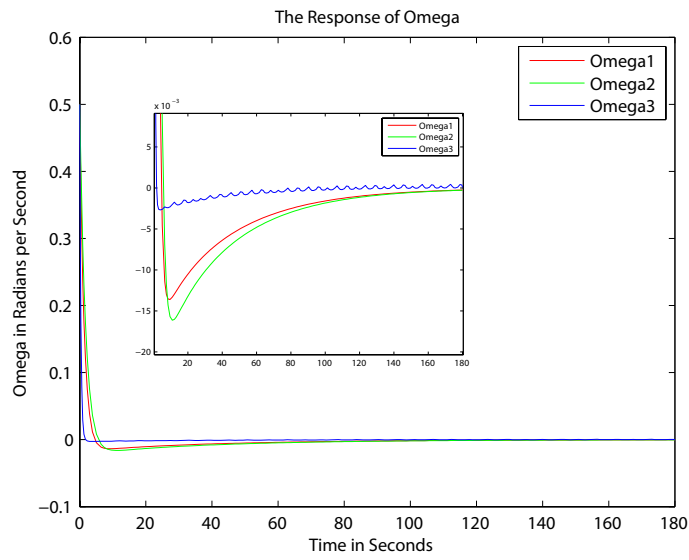


Figure 3.13: The Response of System One Due to an Initial Impulse With Disturbance

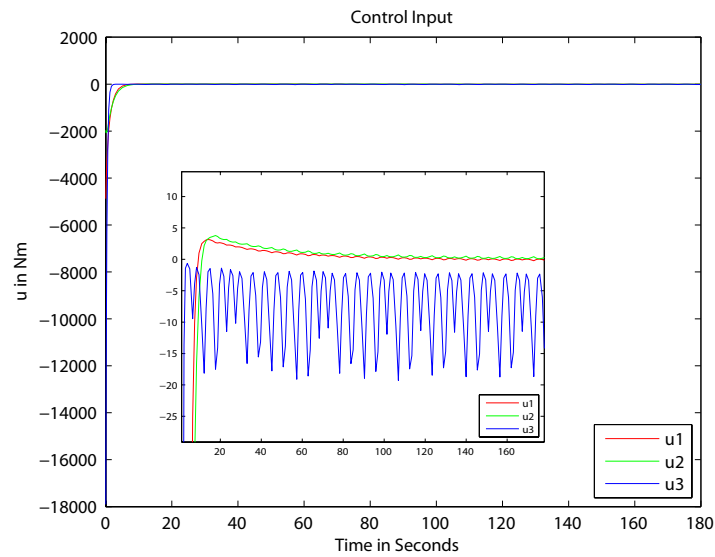


Figure 3.14: The Control Input of System One Due to an Initial Impulse With Disturbance

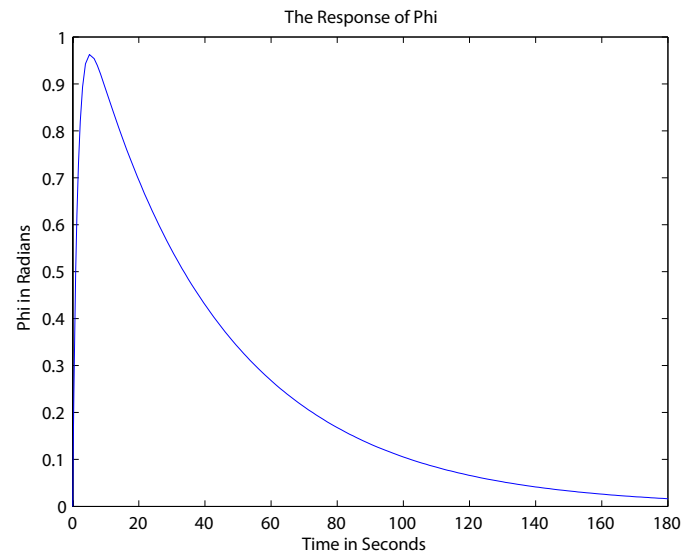


Figure 3.15: The Angle of System One Due to an Initial Impulse With Disturbance

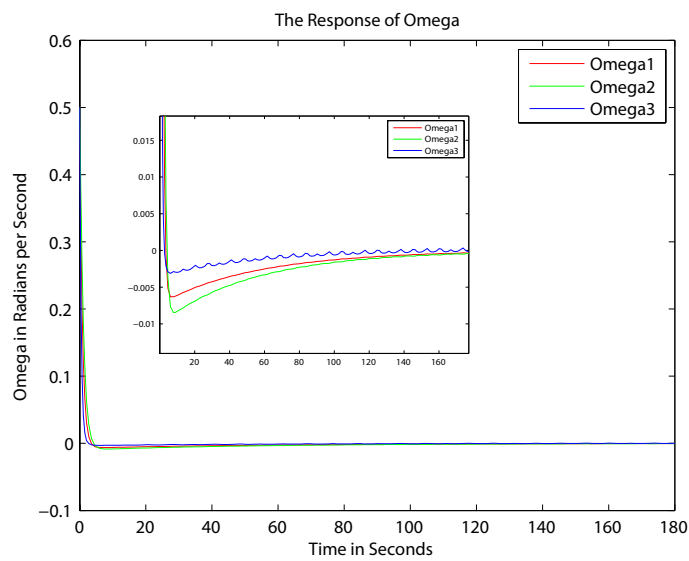


Figure 3.16: The Response of System Two Due to an Initial Impulse With Disturbance

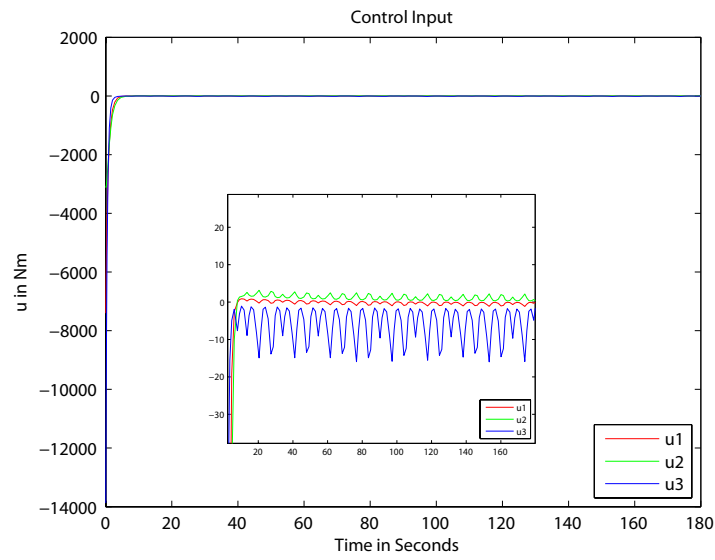


Figure 3.17: The Control Input of System Two Due to an Initial Impulse With Disturbance

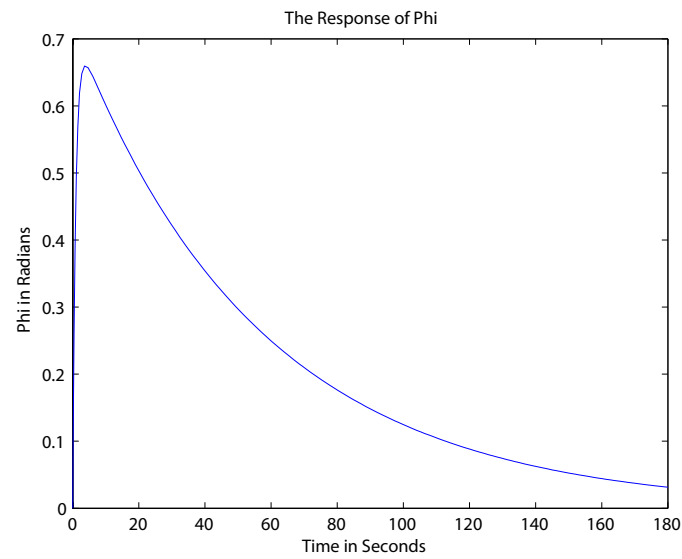


Figure 3.18: The Angle of System Two Due to an Initial Impulse With Disturbance

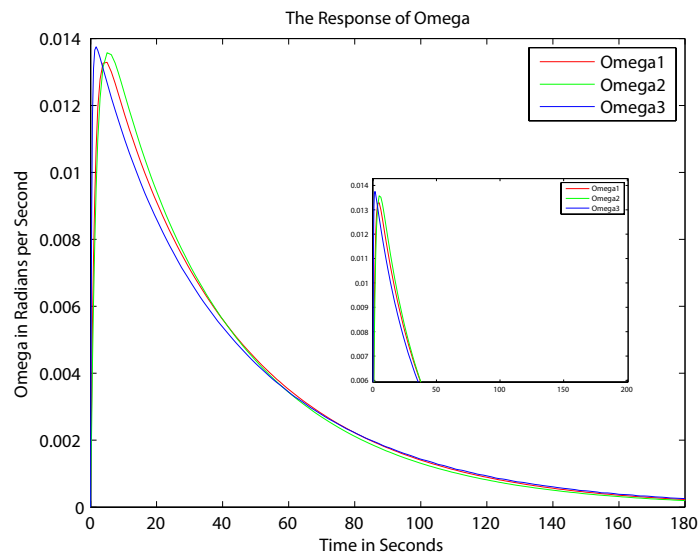


Figure 3.19: The Response of System One Due to an Initial Displacement With Disturbance

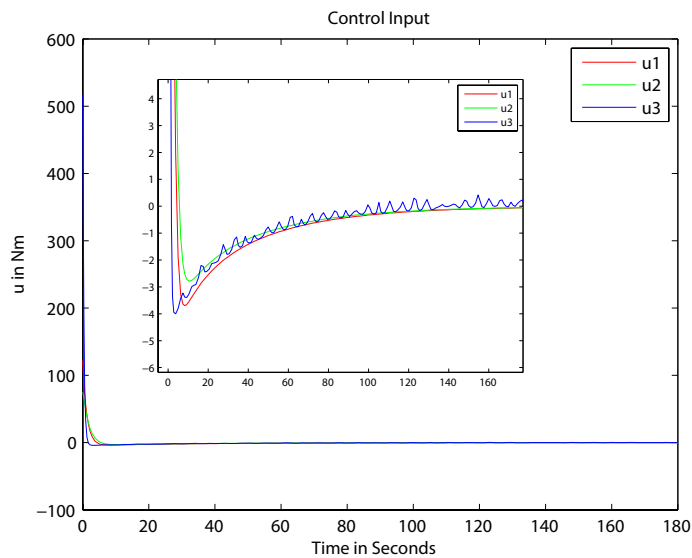


Figure 3.20: The Control Input of System One Due to an Initial Displacement With Disturbance

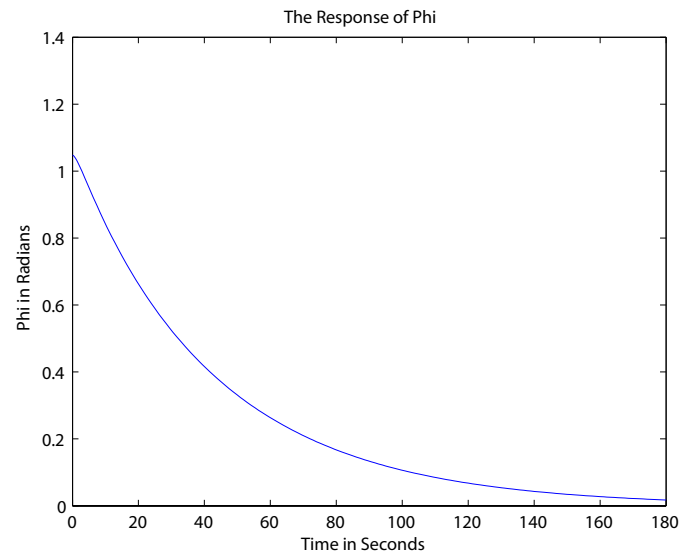


Figure 3.21: The Angle of System One Due to an Initial Displacement With Disturbance

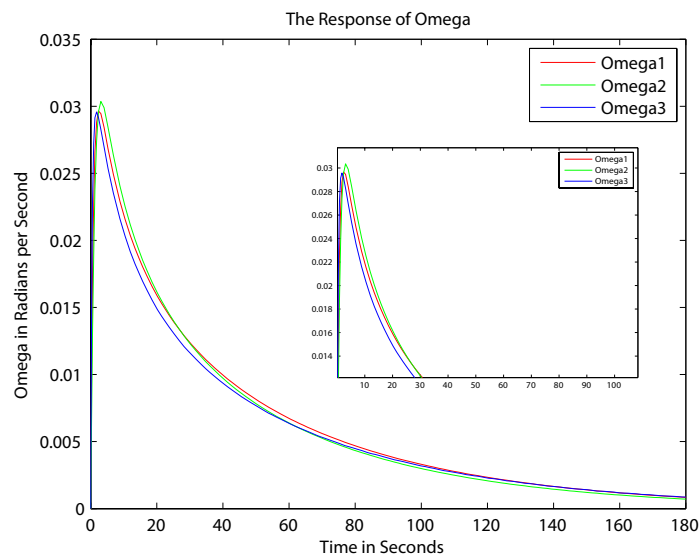


Figure 3.22: The Response of System Two Due to an Initial Displacement With Disturbance

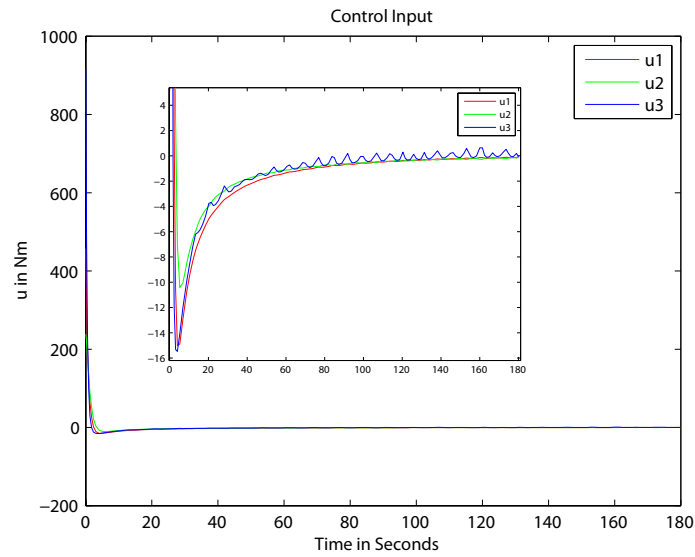


Figure 3.23: The Control Input of System Two Due to an Initial Displacement With Disturbance

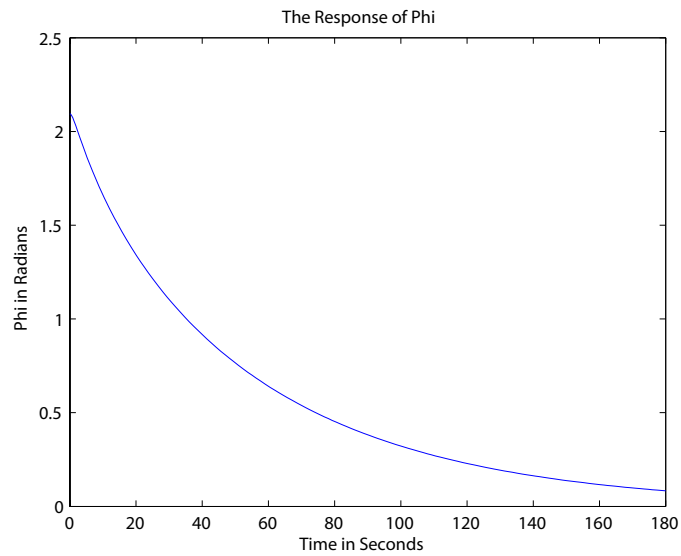


Figure 3.24: The Angle of System Two Due to an Initial Displacement With Disturbance

## Chapter 4

# Conclusions

### 4.1 Contributions of This Research Work

This study has been unique from previous work done relating to spacecraft control systems by establishing a unique way of applying the Cayley-Rodrigues parameters to reduce the number of LPV scheduling parameters in the system. This study has applied the  $H_\infty$  optimization to the LPV control technique. Using this solution approach for the controller design has been analyzed, simulated, and compared the results of different controllers. These simulations have shown that the controllers do in fact demonstrate a feasible solution with respect to enhanced disturbance rejection and power consumption. This gain scheduling technique has proven to be useful for solving nonlinear systems with optimality.

In addition to combining the  $H_\infty$  optimization analysis and LPV gain scheduling techniques, this study has been beneficial in laying out the ground work for future study in the area of optimized nonlinear controls. There are still some concerns that need to be addressed, but this study has shown that there is potential for using gain scheduling to achieve optimal control for a nonlinear system.

### 4.2 Future Work

Currently the Cayley-Rodrigues parameters are used here. These parameters are only good on a range that is less than  $180^\circ$ . If the modified Rodrigues parameters were applied to the system, the range could be increased to a full  $360^\circ$  of rotation without a singularity [11].

Since this study has shown that a parameter range can not be divided more than seven times with the current computer resources and the range of  $167^\circ$  is unstable, a solution to the problem requiring a larger angle of rotation is needed. A possible solution technique to this problem could be to use a switching algorithm [10]. This technique involves solving more than one controller over different set of parameter ranges, and then implementing a set of conditions that cause the controller to switch from one controller to another as the spacecraft passes through the parameter ranges of the different controllers. Using this technique, it would be possible to solve several different controllers for the entire range of motion up to  $360^\circ$  of rotation, and switch between these controllers accordingly.

Only a constant weighting function has been considered, and this weighting function can cause chatter on the control inputs. This chatter is undesirable because it results in additional control effort that causes the system to be less efficient with respect to power consumption. Future study to find a parametrically dependent weighting function could prove beneficial. With the proper weighting function, it may be possible to see a system with diminished power consumption.

It would also be beneficial to see a comparison between this solution technique and existing techniques to see the advantages of this solution. In the interest of time, only one solution technique has been covered. It would be beneficial to have a basis of comparison to see the improvements this technique has made on previous methods of nonlinear control. In addition to this comparison, it would also be interesting to see this technique applied to different systems. In addition to the regulation problem covered, it would be interesting to see this technique applied to the tracking problem. It may also be beneficial to see this technique simulated for more sets of initial conditions.

Insight into a new controller design technique has been offered, this technique affords the opportunity for optimal control. This technique still has room for improvement, but is still an improvement over previous nonlinear spacecraft control techniques.

# Bibliography

- [1] G. Becker and A. Packard. Robust performance of linear parametrically varying systems using parametrically-dependent linear feedback. *Systems and Control Letters*, 23:205–215, 1994.
- [2] S.P. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in Systems and Control Theory*. SIAM, Philadelphia, PA, 1994.
- [3] M. Dalsmo and O. Egeland. State feedback  $H_\infty$ -suboptimal control of a rigid spacecraft. *IEEE Transactions on Automatic Control*, 42(8):1186–1189, August 1997.
- [4] M. Elgersma, G. Stein, M.R. Jackson, and J. Yeichner. Robust controller for space station momentum management. *IEEE Control Systems Magazine*, 12(5):14–22, 1992.
- [5] P. Gahinet and P. Apkarian. A linear matrix inequality approach to  $H_\infty$  control. In *International Journal of Robust and Nonlinear Control*, volume 4, pages 421–448, 1994.
- [6] P. Gahinet, A. Nemirovski, A.J. Laub, and M. Chilali. *LMI Control Toolbox*. Mathworks, Natick, MA, May 1995.
- [7] W. Kang. Nonlinear  $H_\infty$  control and its application to rigid spacecraft. *IEEE Transactions on Automatic Control*, 40(7):1281–1288, July 1995.
- [8] M. Krstic and P. Tsiotras. Inverse optimal stabilization of a rigid spacecraft. *IEEE Transactions on Automatic Control*, 44(5):1042–1049, 1999.
- [9] S. Lim. New quaternion feedback control for efficient large angle maneuvers. *AIAA Guidance, Navigation and Control Conference*, August 2001.
- [10] B. Lu, F. Wu, and S. Kim. Switching lpv control of an f-16 aircraft via controller state reset. *IEEE Transactions on Control Systems Technology*, 14(2):1–11, March 2006.

- [11] P. Tsiotras. Stabilization and optimality results for the attitude control problem. *Journal of Guidance, Control, and Dynamics*, 19(4):1–9, July-August 1996.
- [12] B. Wie. *Space Vehicle Dynamics and Control*. AIAA, 1998.
- [13] B. Wie, K.W. Byun, and V.W. Warren. New approach to attitude/momentum control for the space station. *AIAA Journal of Guidance, Control and Dynamics*, 12:714–722, 1989.
- [14] F. Wu, X. Yang, A. Packard, and G. Becker. Induced  $l_2$ -norm control for lpv systems with bounded parameter variation rates. *International Journal of Robust and Nonlinear Control*, 6:983–998, 1996.

# Appendices

## Appendix A

# $H_\infty$ LPV Controller Solution Code mytry.m

```

% This Code Solves The H_infinity problem for an
% LPV and creates a controller based off of the
% chosen parameter ranges.
diary on
J11=10000;
J22=9000;
J33=12000;
JJ = daug(J11,J22,J33);
invJ = inv(JJ);
npts = 7;
nx = 6;
nctrl = 3;
nmeas = 3;

lpv = zeros(13*npts^3,13);
rho = zeros(npts,3);
index = 0;
for i = 1:npts
    for j = 1:npts
        for k = 1:npts
            index = index+1;
            %This is for phi from -120 to 120
            disp([i j k])
            p(1) = (i-1)*1.0/3-1.0;
            p(2) = (j-1)*1.0/3-1.0;
            p(3) = (k-1)*1.0/3-1.0;
            S = [0 p(3) -p(2);-p(3) 0 p(1);p(2) -p(1) 0];
            Psq = [p(1);p(2);p(3)]*[p(1) p(2) p(3)];
            A = [zeros(3,6);(eye(3)-S+Psq)/2 zeros(3,3)];
            B = [0.01*invJ invJ;zeros(3,6)];
            C = [zeros(3,3) eye(3);zeros(3,6)];
            D = [zeros(3,6);zeros(3,3) 0.1*eye(3)];
            sys = pck(A,B,C,D);
        end
    end
end

```

```

        lpv((index-1)*13+1:index*13,:) = sys;
        rho(index,:) = [i,j,k];
    end
end
end
invar = [1:1:npts^3]';
vlpv = vpck(lpv,invar);
vrho = vpck(rho,invar);
[vf,vg,vfgrad,vggrad] = mkbasis2(vrho);

% Solve LPV synthesis
nu = [0.5;0.5;0.5];
epsilon = 0.0;
[slpvgam,rmat] = slpvsyn_sf(vlpv,vf,vfgrad,nu,nctrl,epsilon);

% Construct controller gains
lpvk = zeros(3*npts^3,6);
for i = 1:npts^3
    lpv = xtracti(vlpv,i,1);
    rho = xtracti(vrho,i,1);
    r = rmat(:,1:6)+rho(1,1)*rmat(:,7:12)+rho(1,2)...
        *rmat(:,13:18)+rho(1,3)*rmat(:,19:24);
    lpvk((i-1)*3+1:i*3,:) = mkslpvk_sf(lpv,nctrl,slpvgam,r);
end
savefile = 'rmat.mat';
save(savefile, 'rmat');
savefile = 'lpvk.mat';
save(savefile, 'lpvk');
savefile = 'slpvgam.mat';
save(savefile, 'slpvgam');
savefile = 'vlpv.mat';
save(savefile, 'vlpv');
diary off

```

## Appendix B

### mkbasis2.m

```

% function [vf,vg,vfgrad,vggrad] = mkbasis(vrho)
%   Evaluates basis functions and gradients at desired
%   parameter points.  PARMVEC should be varying matrix
%   of parameter values.
function [vf,vg,vfgrad,vggrad] = mkbasis(vrho)
[systype,dum,dum,npts] = minfo(vrho);
if (systype == 'vary')
    tmp = xtracti(vrho,1,1);
    if (min(size(tmp)) ~= 1)
        error('Parameter must be a vector');
    end
elseif (systype == 'cons')
    if (min(size(vrho)) ~= 1)
        error('Parameter must be a vector');
    end
    vrho = vpck(vrho,1);
else
    error('Parameter data must constant or varying matrix');
end
vf = [];
vg = [];
vfgrad = [];
vggrad = [];
for i = 1:npts
    rho = xtracti(vrho,i,1);
%
% ENTER BASIS FUNCTION HERE
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% S fixed
fdef = [1;rho(1,1);rho(1,2);rho(1,3)];
gdef = [1];
fgraddef = [0 0 0;1 0 0;0 1 0;0 0 1];
ggraddef = [0 0 0];
% R fixed

```

```

% fdef = [1];
% gdef = [1;abs(rho(1,1));rho(1,2)];
% fgraddef = [0 0];
% ggraddef = [0 0;sign(rho(1,1)) 0;0 1];
% Both R, S parameter-dependent
% fdef = [1;abs(rho(1,1));rho(1,2)];
% gdef = [1;abs(rho(1,1));rho(1,2)];
% fgraddef = [0 0;sign(rho(1,1)) 0;0 1];
% ggraddef = [0 0;sign(rho(1,1)) 0;0 1];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
vf = [vf;fdef];
vg = [vg;gdef];
vfgrad = [vfgrad;fgraddef];
vggrad = [vggrad;ggraddef];
end
inv = [1:1:npts]';
vf = vpck(vf,inv);
vg = vpck(vg,inv);
vfgrad = vpck(vfgrad,inv);
vggrad = vpck(vggrad,inv);

```

## Appendix C

### slpvsyn\_sf.m

```

%function [slpvgam,rmat,smat] = slpvsyn_sf(vlpv,vf,vfgrad,nu,nctrl,epsilon)
% State-feedback code
% Synthesize optimal H-infinity gain SLPVGAM for LPV system with
% NU parameter variation and get feasible RMAT and SMAT matrices
% which are the basis of parameter-dependent Lyapunov function.
% It is assumed that for each fixed rho, D12(rho) and D21(rho) have full
% column and row ranks, respectively.
function [slpvgam,rmat,smat] = slpvsyn_sf(vlpv,vf,vfgrad,nu,nctrl,epsilon)

if (nargin == 0)
    disp('Usage: [slpvgam,rmat,smat] = slpvsyn(vlpv,vf,vfgrad,nu,nctrl,epsilon)')
    return;
end
if (nargin == 5)
    epsilon = 0;
end

[systype,dum,dum,npts] = minfo(vlpv);
sys = xtracti(vlpv,1,1);
[systype,no,ni,nx] = minfo(sys);
nd = ni-nctrl;
ne = no;

ndim = length(nu);
nvertix = 2^ndim;
fdat = xtracti(vf,1,1);
nfb = length(fdat);

combdat = corners(ndim);

setlmis([]);
% Set optimization variables
for i = 1:nfb
    lmivar(1,[nx 1]); % RMAT
end

```

```

idGAM = lmivar(1,[1 0]); % GAMMA

for i = 1:npts
    disp(i)

    fdat = xtracti(vf,i,1);
    fgraddat = xtracti(vfgrad,i,1);

    sys = xtracti(vlpv,i,1);
    [a,b,c,d] = unpck(sys);
    b1 = b(:,1:nd);
    b2 = b(:,nd+1:ni);
    c1 = c(1:ne,:);
    d11 = d(1:ne,1:nd);
    d12 = d(1:ne,nd+1:ni);

    Nr = null([b2' d12']);

    indx1 = (nvertix+1)*(i-1);
% LMI of RMat
    tempr = daug(Nr,eye(nd));

    for j = 1:nvertix
        lmiterm([indx1+j 0 0 0],tempr);
        for k = 1:nfb
            lmiterm([indx1+j 1 1 k],a,fdat(k),'s');
            lmiterm([indx1+j 2 1 k],c1,fdat(k));
            for l = 1:ndim
                lmiterm([indx1+j 1 1 k],[-combdat(j,l)*nu(l)*fgraddat(k,l),1);
            end
        end
        lmiterm([indx1+j 2 2 idGAM],-1,1);
        lmiterm([indx1+j 3 1 0],b1');
        lmiterm([indx1+j 3 2 0],d11');
        lmiterm([indx1+j 3 3 idGAM],-1,1);
    end
% Positive-definite condition
    for k = 1:nfb
        lmiterm([i*(nvertix+1) 1 1 k],[-fdat(k),1);
    end
end

lmis = getlmis;

nvar = decnbr(lmis);
disp([' Total variable numbers: ',num2str(nvar)])

nlmi = lminbr(lmis);
disp([' Total LMI numbers: ',num2str(nlmi)])

```

```

% Construct CVEC
nvar = degnbr(lmis);
cvec = zeros(nvar,1);
for i = 1:nvar
    [vGAMi] = defcx(lmis,i,idGAM);
    cvec(i,1) = vGAMi;
    for k = 1:nfb
        [vRi] = defcx(lmis,i,k);
        cvec(i,1) = cvec(i,1)+epsilon*trace(fdat(k)*vRi);
    end
end
% Call LMI optimization subroutine
[copt,xopt] = mincx(lmis,cvec,[1e-3 500 -1 0 0]);
% Convert to optimization variables to matrix form
rmat = [];
for i = 1:nfb
    r = dec2mat(lmis,xopt,i);
    rmat = [rmat r];
end
slpvgam = dec2mat(lmis,xopt,idGAM);

% Revised by Fen Wu on Oct. 9, 2006 based on the papers
% ‘‘Induced L2 norm control of LPV systems with bounded parameter
% variation rates’’
% by Fen Wu et al., IJRNC 1996 and
% ‘‘Additional results on parameter-dependent controllers for LPV
% systems’’
% by Greg Becker, IFAC Proc. 1996.

```

## Appendix D

### mkslpvk\_sf.m

```

% function [slpvk] = mkslpvk_sf(sys,nctrl,slpvgam,rmat)
% Build state-feedback LPV controller that solves
% H-infinity LPV system/nu-variation
% synthesis problem with general assumptions on
%state-space data.
function [slpvk] = mkslpvk_sf(sys,nctrl,slpvgam,rmat)
if (nargin == 0)
    disp(['usage: [slpvk] = mkslpvk_sf(sys,nctrl,slpvgam,rmat)']);
    return;
end
[systype,no,ni,nx] = minfo(sys);
nd = ni-nctrl;
ne = no;
gami2 = 1/slpvgam/slpvgam;
[a,b,c,d] = unpck(sys);
b1 = b(:,1:nd);
b2 = b(:,nd+1:ni);
c1 = c(1:ne,:);
d11 = d(1:ne,1:nd);
d12 = d(1:ne,nd+1:ni);
dh = eye(ne)/(eye(ne)-gami2*d11*d11');
c1dh = c1'*dh;
F = -(d12'*dh*d12)\(c1dh*d12+slpvgam*(rmat\b2))';
slpvk = F;

```

## Appendix E

### stability\_test.m

```
load results_167.mat
npts = 7;
[lpv,invar] = vumpck(vlpv);
index = 0;
for i = 1:npts
    for j = 1:npts
        for k = 1:npts
            index = index+1;
            [A,B,C,D]=umpck(lpv((index-1)*13+1:index*13,:));
            G=lpvk((index-1)*3+1:index*3,:);
            A1=(A-B(:,4:6)*G);
            [V,D1]=eig(A1);
            poles((index-1)*6+1:index*6,:)=D1;
        end
    end
end
savefile = 'poles.mat';
save(savefile, 'poles');
```

## Appendix F

### plant\_temp.m

```

function [sys,x0,str,ts] = plant_temp(t,x,u,flag,X0)
% nonlinear spacecraft dynamic model.
% Input: (1) u1 (2) u2 (3) u3
% Outputs: (1) omega1 (2) omega2 (3) omega3
%          (4) p1 (5) p2 (6) p3
% States: (1) omega1 (2) omega2 (3) omega3
%         (4) p1 (5) p2 (6) p3
nx = 6;
ni = 3;
no = 6;

switch flag,
    %%%%%%%%%%%
    % Initialization %
    %%%%%%%%%%%
    case 0,
        [sys,x0,str,ts]=mdlInitializeSizes(nx,ni,no,X0);
        %%%%%%%%%%%
        % Derivatives %
        %%%%%%%%%%%
    case 1,
        sys=mdlDerivatives(t,x,u);
        %%%%%%%%%%%
        % Outputs %
        %%%%%%%%%%%
    case 3,
        sys=mdlOutputs(t,x,u);

    case {2,4,9},
        sys = [];
        %%%%%%%%%%%
        % Unexpected flags %
        %%%%%%%%%%%
    otherwise
        error(['Unhandled flag = ',num2str(flag)]);

```

```

end

%=====
% mdlInitializeSizes
% Return the sizes, initial conditions, and sample times for the S-function.
%=====
function [sys,x0,str,ts]=mdlInitializeSizes(nx,ni,no,X0)
sizes = simsizes;
sizes.NumContStates = nx;
sizes.NumDiscStates = 0;
sizes.NumOutputs = no;
sizes.NumInputs = ni;
sizes.DirFeedthrough = 0;
sizes.NumSampleTimes = 1; % at least one sample time is needed
sys = simsizes(sizes);
% initialize the initial conditions
x0 = X0;
% str is always an empty matrix
str = [];
% initialize the array of sample times
ts = [0 0];
% end mdlInitializeSizes

%=====
% mdlDerivatives
% Return the derivatives for the continuous states.
%=====
function sys=mdlDerivatives(t,x,u)
% polar moments of inertia
J11=10000;
J22=9000;
J33=12000;
JJ = daug(J11,J22,J33);
invJ = inv(JJ);
% omega's and p's
omega = [x(1);x(2);x(3)];
p = [x(4);x(5);x(6)];
omega_scew = [0,-omega(3,1),omega(2,1);omega(3,1),0,...
    -omega(1,1);-omega(2,1),omega(1,1),0];
p_scew = [0,-p(3,1),p(2,1);p(3,1),0,-p(1,1);...
    -p(2,1),p(1,1),0];
H = (eye(3)-p_scew+p*p')/2;
% Nonlinear plant state derivatives
sys = [invJ*(omega_scew*JJ*omega+u+.01.*rand(3,1));
    H*omega];
% end mdlDerivatives

%=====
% mdlOutputs
% Return the block outputs.

```

```
%=====
function sys=mdlOutputs(t,x,u)
% Output equations
sys = [x];
% end mdlOutputs

%=====
% mdlTerminate
% Perform any end of simulation tasks.
%=====
function sys=mdlTerminate(t,x,u)
sys = [];
% end mdlTerminate
```

## Appendix G

### control3\_temp.m

```

function [sys,xk0,str,ts] = control3_temp(t,xk,uk,flag,OLIC,NMEAS,NCTRL,SLPVGAM,RMAT)
% full state feedback gain-scheduling controller.
% Inputs: (1) omega1 (2) omega2 (3) omega3
%          (4) p1 (5) p2 (6) p3
% Outputs: (1) u1 (2) u2 (3) u3
nxk = 0;
ni = 6;
no = 3;

switch flag,
    %%%%%%%%%%%
    % Initialization %
    %%%%%%%%%%%
    case 0,
        [sys,xk0,str,ts]=mdlInitializeSizes(nxk,ni,no);
    %%%%%%%%%%%
    % Derivatives %
    %%%%%%%%%%%
% case 1,
% %%%%%%%%%%%
% % Outputs %
% %%%%%%%%%%%
    case 3,
        sys=mdlOutputs(t,xk,uk,OLIC,NMEAS,NCTRL,SLPVGAM,RMAT);
    case {1,2,4,9},
        sys = [];
    %%%%%%%%%%%
    % Unexpected flags %
    %%%%%%%%%%%
    otherwise
        error(['Unhandled flag = ',num2str(flag)]);
end

%=====
% mdlInitializeSizes

```

```

% Return the sizes, initial conditions, and sample times for the S-function.
%=====
function [sys,xk0,str,ts]=mdlInitializeSizes(nxk,ni,no)
sizes = simsizes;
sizes.NumContStates = nxk;
sizes.NumDiscStates = 0;
sizes.NumOutputs = no;
sizes.NumInputs = ni;
sizes.DirFeedthrough = 1;
sizes.NumSampleTimes = 1; % at least one sample time is needed
sys = simsizes(sizes);
% initialize the initial conditions
xk0 = zeros(nxk,1);
% str is always an empty matrix
str = [];
% initialize the array of sample times
ts = [0 0];
% end mdlInitializeSizes

%=====
% mdlOutputs
% Return the block outputs.
%=====
function sys=mdlOutputs(t,xk,uk,OLIC,NMEAS,NCTRL,SLPVGAM,RMAT)
% Polar moment of inertia
J11 = 10000;
J22 = 9000;
J33 = 12000;
JJ = daug(J11,J22,J33);
invJ = inv(JJ);
% state variable
omega = [uk(1);uk(2);uk(3)];
p = [uk(4);uk(5);uk(6)];
omega_scew = [0,-omega(3,1),omega(2,1);omega(3,1),0,...
    -omega(1,1);-omega(2,1),omega(1,1),0];
p_scew = [0,-p(3,1),p(2,1);p(3,1),0,-p(1,1);...
    -p(2,1),p(1,1),0];
Hrho = (eye(3)-p_scew+p*p')/2;
OLIC(4:6,1:3) = Hrho;
rsym = RMAT(:,1:6)+p(1,1)*RMAT(:,7:12)+p(2,1)...
    *RMAT(:,13:18)+p(3,1)*RMAT(:,19:24);
lpvk = mkslpvk_sf(OLIC,NCTRL,SLPVGAM,rsym);
sys = [lpvk*uk - omega_scew*JJ*omega];
% end mdlOutputs

%=====
% mdlTerminate
% Perform any end of simulation tasks.
%=====
function sys=mdlTerminate(t,x,u)

```

```
sys = [];  
% end mdlTerminate
```