

ROBUST NONLINEAR ATTITUDE CONTROL WITH DISTURBANCE COMPENSATION

By

KEVIN J. WALCHKO

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2003

Copyright 2003

By

Kevin J Walchko

ACKNOWLEDGEMENTS

I would like to thank Dr. Paul Mason for his guidance of my education and research. He has always made every effort to aid me in my work and life. Paul has held many roles in my life such as mentor, academic advisor, friend, and best man at my wedding. He has become more than an advisor over the years, and will forever be part of my family. I will always be in his debt.

I would like to thank Dr. John Schueller who acted as my co-chair during this dissertation. He provided much advice and guidance in the bureaucracy of University of Florida. Schueller was always willing to help me with various paperwork and other such matters pertaining to fellowships, pay checks, and graduation. He is a good friend and I look forward to working with him in the future.

I would also like to thank my family who sacrificed much at times so this could be possible. My wife Nina took great care of me and our new son Michael. She made sure I had plenty of time to do my work and never complained about anything. She is the source from which I draw all of my strength, ambition, and dedication.

Finally, I would like to thank the horde, our pack of cats. They provided much amusement and stress relief through their wacky antics. I treasure each and every one of them, and will never forget those who have passed away.

TABLE OF CONTENTS

	<u>page</u>
ACKNOWLEDGEMENTS	iii
LIST OF FIGURES	viii
LIST OF TABLES	xi
LIST OF SYMBOLS	xiii
ABSTRACT	xv
1 INTRODUCTION	1
Formation Flying	1
General Problems Controlling Single Satellites	2
External Disturbances	2
Parametric Uncertainty	3
Literature Review	4
Fuel Slosh	4
Modelling fuel slosh dynamics	4
Controlling fuel slosh.....	6
Do we need to worry about fuel slosh?.....	9
Missions involving fuel slosh disturbances	12
Thermally Induced Vibrations: Solar Snap	12
Controlling thermal vibrations.....	13
Missions which involved solar snap	15
Parametric Uncertainty	16
Summary and Overview of the Thesis	17
2 CONTROLLING THE UNKNOWN: FUEL SLOSH	19
Introduction	19
Modelling Fuel Slosh Dynamics	20
Fuel Slosh Models	23
Simulation Model	23
Controller Model	24

3 DYNAMICS OF SOLAR ARRAYS	26
Introduction	26
Thermally Induced Vibrations	26
Heat Flux	27
Summing Forces	28
Nonhomogeneous Boundary Conditions	33
Separation of Variables	34
Solving the position function $F(x)$	35
Solving the time function $G(t)$	38
Summary of results	38
Thermally Induced Dynamics	39
Modal Equations	39
Disturbance Torques	40
Thickness Temperature Gradient	42
Conclusion	44
4 SATELLITE ATTITUDE DYNAMICS	46
Reference Frames	46
Earth Centered Inertial	46
Orbital Frame	47
Body Fixed Frame	47
Fun With Vectors and Rotating Coordinate Systems	47
Spacecraft Attitude	49
Quaternions	50
Rotations of Rigid Bodies in Space.	51
Spacecraft Equations of Motion	51
Attitude Kinematics	52
Spacecraft Dynamics	52
Simple Spacecraft	52
Angular Velocity	56
Internal Disturbances	56
Spacecraft with Reaction or Momentum Wheels	56
Fuel Slosh	60
Environmental Disturbances in Space	61
Gravity Gradient	62
Solar Snap	62
Conclusion	62
5 ATTITUDE CONTROL OF SPACECRAFT	64
Satellite Control Hardware	64
Control Computational Hardware	64
Control Actuators	65
Reaction and momentum wheels	65

Magnetic torque rods	66
Gas jets.....	67
Proportional Derivative Control	67
Sliding Mode Control Theory and Background	69
What is Sliding Mode?	69
Theory	69
Simple Example	72
Other Sliding Mode Designs for Spacecraft	73
Satellite Controller Design For This Work	75
Controller Derivation	75
Fuel Slosh Disturbance	77
Stability of Controller	78
Saturation	80
Comparing The Two Types of Controller	81
Conclusion	81
 6 SIMULATION AND RESULTS	 83
Simulation Dynamics and Parameters	83
Satellite	83
Satellite Dynamics	85
Fuel Tank	86
Reaction Wheels	86
Solar Panels	87
Control System	87
Finding an Optimal PD Gain	88
Fuel Slosh	93
IAE	94
ITAE	95
Settling Time	95
CM	96
Torque and Momentum Limits	96
Qualitative Analysis of Results	98
Solar Snap	98
Performance	98
Qualitative Analysis of Results	99
 7 CONCLUSION	 101
 A ATTITUDE REPRESENTATIONS AND ROTATION MATRICES	 104
Fixed Angle Rotations	104
Euler Angles	105
Quaternions	107
Quaternion Algebra	108
Rotations of Rigid Bodies in Space.	110

Attitude Errors in Quaternions	111
Summary of Quaternions	112
B MODAL EQUATIONS	113
Motion of Multi-Degree of Freedom System	113
Dampening	115
Simple Example	116
C SIMULATION CODE	117
Simulation Code	117
Matlab	117
C Programming Language	117
C++ Programming Language	118
cMathlib Header File	119
cRK4 Header File	126
REFERENCES	127
BIOGRAPHICAL SKETCH	134

LIST OF FIGURES

<u>Figure</u>	<u>page</u>
1-1 Finite element analysis of a fuel tank inside of a satellite.	6
1-2 PPT for EO-1.	10
1-3 Diagram of a PPT.	10
1-4 NEAR spacecraft orbiting the asteroid Eros.	11
1-5 Hubble Space Telescope diagram.	13
2-1 Traditional fuel slosh pendulum model where M is the total fuel mass center, m_p is the pendulum mass, m_s is the stationary mass fraction, L is the pendulum arm length, and g is the local acceleration due to some external force.	22
2-2 Propagation of fuel slosh in an elliptical tank over time. These images demonstrate the difficulty in modelling the fluid movement with simple representations.	22
2-3 Diagram of a single sloshing mode, with its mass attached to the fuel tank walls by springs and dampers aligned along the x , y , and z axes.	23
2-4 Plot of the effect of multiple modes in the velocity of a simply supported beam. This is representative of the fuel slosh disturbance in flight data.	24
3-1 The Soviet satellite Sputnik which orbited the Earth in October 1957.	27
3-2 Diagram of the penumbra and umbra regions which block a satellite from sunlight.	28
3-3 Another view showing the path of a satellite and how its orbit does effect the heat flux it is exposed to.	28
3-4 Cross section of solar panel with coordinate systems superimposed. Note that the S 's in the figures represent 's and M 's are moments.	30
3-5 Free body diagram of a section of the beam of size dx where $M(x,t)$ is a moment, $V(x,t)$ is a shear force, $f(x)$ is a force per unit length, and $u(x,t)$ is the height of the mass element at position x at time t	32

3-6	Plot of [3-49]. Intersections of the two lines are solutions of the characteristic equation.	36
3-7	A cross section of the beam in which a heat flux is applied to one side. Notice the beam starts off at a uniform temperature.	42
3-8	Plot of the temperature change of a beam. Notice that the temperature difference between the two sides remains constant eventually.	42
3-9	Hubble Space Telescope which shows the solar panels bent due to a constant temperature gradient.	43
3-10	Another picture of the Hubble Space Telescope showing deflection of the solar panels during maintenance by the space shuttle.	44
4-1	The three frames of reference commonly used in spacecraft dynamics.	47
4-2	A reference frame in motion relative to an inertially fixed frame of reference.	48
4-3	A rigid body satellite.	53
4-4	A rigid body spacecraft with a single reaction wheel.	57
4-5	A plot of common environmental disturbances a spacecraft is subjected too.	60
5-1	Two types of reaction wheels produced by Ithaco.	66
5-2	Torque rods produced by Ithaco.	67
5-3	Spacecraft engine produced by Boeing.	68
5-4	Two possible solutions for the double integrator, where (a) is and (b) is . These plots are typically referred to as phase portraits. They plot position and velocity in this example, but later (when we are doing controls with full state feed back) the portraits will represent error and error velocity.	70
5-5	Switching between the two controllers results in driving the system to zero in a stable manor.	71
5-6	Graphical representation of how effects the sliding surface's orientation in the phase plane. The dotted lines represent the sliding surface ($s = 0$).	72
6-1	EO-1 satellite with solar panel deployed.	83
6-2	Diagram of satellite's internal components.	84

6-3	Composition of a typical simple solar panel.	86
6-4	Cost function surface plot for the optimal gain of the PD controller.	89
6-5	ITAE results from the optimal PD search.	90
6-6	IAE results from the optimal PD search.	90
6-7	Settling time results from optimal PD search.	91
6-8	Control momentum or fuel used searching for the optimal PD controller.	92
6-9	A depiction of the step maneuver showing the orientation of the satellite at start and end. The angular distance and axis of rotation are also shown for better understanding of the terms.	94
6-10	Quaternion error and control effort of the PD controller rotating from -30 to 30.	96
6-11	Quaternion error and control of the SM controller rotating from -30 to 30.	97
6-12	Quaternion error and control of the SMCNF controller rotating from -30 to 30.	97
6-13	Arcsecond error of the PD controller during solar snap.	99
6-14	Arcsecond error of the SM controller during solar snap.	100
6-15	Arcsecond error of the SMCNF controller during solar snap.	100
A-1	Body reference frame attached to a rigid body. The x-axis points out the front of the vehicle.	105
B-1	Simple multi-degree of freedom system composed of mass and spring elements.	113
C-1	UML diagram of Mathlib class which contains C++ objects for vectors, matrices, and quaternion used in this work.	119

LIST OF TABLES

<u>Table</u>	<u>page</u>
1-1 Spacecraft with thermally induced disturbances.	15
3-1 Solar panel thermal material properties.	32
3-2 First four eigen modes of the beam	35
5-1 Comparison of common active attitude control hardware.	65
5-2 Summary of the switching law.	70
6-1 Technical Specifications for the EO-1 spacecraft	84
6-2 EO-1 fuel slosh parameters.	86
6-3 EO-1 wheel specifications.	86
6-4 EO-1 solar panel parameters.	87
6-5 Initial and final euler angles where roll, pitch, and yaw are equal.	93
6-6 IAE results with values having units of radians.	94
6-7 ITAE results with values having units of radian-seconds ²	95
6-8 Settling time results with values having units of seconds.	96
6-9 CM results with values having units of Nms.	96
6-10 Maximum wheel momentum achieved where units are in Nms.	98
6-11 Solar snap results of the three controllers.	99
A-1 Properties of a rotation matrix.	105
A-2 Comparison of the two major types of rotations, sequential and first and third axes rotation	106

A-3	Quaternion Algebra Summary	109
-----	----------------------------------	-----

LIST OF SYMBOLS

Thermally Induced Vibrations Symbols

α	Absorptivity
α_T	Coefficient of thermal expansion.
ω_n	Natural frequency of mode n .
ρ	Density of a material.
κ	Thermal diffusivity.
A	Cross sectional area of the solar panel.
c	Heat capacity.
E	Young's Modulus
$F(x)$	Position function from the separation of variables method.
F_n	Position function for mode n .
$G(t)$	Time function from the separation of variables method.
h	Thickness of the solar panel.
k	Thermal conductivity.
$M_T(t)$	Thermal moment.
q_o	Total heat flux on surface.
q_s	Heat flux from sun.
q_e	Earth emitted heat flux.
q_a	Earth reflected heat flux.
$u(x,t)$	Displacement of solar panel.
$V(x,t)$	Shear force.

Satellite Dynamics and Fuel Slosh Symbols

$\omega^{B/I}$	Angular velocity of satellite system relative to inertial frame.
$\omega^{B/O}$	Angular velocity of satellite system relative to orbital frame.
$\omega^{O/I}$	Angular velocity of orbital frame relative to inertial frame.

Satellite Dynamics and Fuel Slosh Symbols

μ	
C	Rotation matrix.
c	First moment of inertia.
c_{fn}	Damper coefficient for sloshing mode n .
F	External forces and torques.
h	Angular momentum.
J	Second moment of inertia.
k_{fn}	Spring coefficient for sloshing mode n .
L	Length of pendulum.
M	Mass matrix of satellite system.
m_p	Pendulum mass.
m_s	Stationary mass fraction.
p	Linear momentum.
q	Quaternion
\hat{q}	The imaginary part of the quaternion, 1-3
R_c	Distance of satellite from center of Earth.
T	Kinetic energy of satellite system.
V	Composite linear and angular velocity of satellite system.

Controls Symbols

λ	Weighting term on the sliding surface.
K_d	Derivative controller gain.
K_p	Proportional controller gain.
PD	Proportional Derivative controller
s	Sliding surface.
SM	Sliding Mode controller.
$SMCNF$	Sliding Mode controller with a colored noise filter
T_d	Gain ratio between the proportional and derivative gains.

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree Doctor of Philosophy

ROBUST NONLINEAR ATTITUDE CONTROL WITH DISTURBANCE
COMPENSATION

By

Kevin J Walchko

May 2003

Chair: Paul Mason

Cochair: John Schueller

Major Department: Mechanical and Aerospace Engineering

Attitude control of small spacecraft is a particularly important component for many missions in the space program: Hubble Space Telescope for observing the cosmos, GPS satellites for navigation, SeaWiFS for studying phytoplankton concentrations in the ocean, etc. Typically designers use proportional derivative control because it is simple to understand and implement. However this method lacks robustness in the presence of disturbances and uncertainties. Thus to improve the fidelity of this simulation, two disturbances were included, fuel slosh and solar snap.

Fuel slosh is the unwanted movement of fuel inside of a fuel tank. The fuel slosh model used for the satellite represents each sloshing mode as a mass-spring-damper. The mass represents the wave of fuel that propagates across the tank, the damper represents the baffling that hinders the movement, and the spring represents the force imparted to the spacecraft when the wave impacts the tank wall. This formulation makes the incorporation

of multiple modes of interest simple, which is an advance over the typical one sloshing mode, pendulum model.

Thermally induced vibrations, or solar snap, occur as a satellite transitions from the day-to-night or night-to-day side of a planet. During this transition, there is a sudden change in the amount of heat flux to the solar panels and vibrations occur. Few authors have looked at the effects of solar snap. The disturbance dynamics were based on the work by Earl Thorten. The simulated effects compared favorably with real flight data taken from satellites that have encountered solar snap.

A robust sliding mode controller was developed and compared to a more traditional proportional derivative controller. The controllers were evaluated in the presence of fuel slosh and solar snap. The optimized baseline proportional derivative controller used in this work showed little effort was needed to obtain better performance using sliding mode. In addition, a colored noise filter was developed to compensate for the fuel sloshing disturbance and incorporated into the sliding mode controller for greater performance increase at the expense of requiring a little more control effort.

CHAPTER 1 INTRODUCTION

This chapter provides the motivation and a literature review for this work. Additional literature will be presented in subsequent chapters as appropriate.

Formation Flying

As scientists aim for higher goals of discovery, satellites prove to be an increasingly important tool. Bigger, more powerful satellites can search the cosmos more easily for distant planets, map the Earth, or even try to discover the origins of the universe itself. However, when a satellite reaches a certain size it effectively can no longer be launched into space.

The only answer to bigger, more powerful satellites is either construction in space or formation flying. Construction in space is difficult and costly, but perhaps one day when the international space station (or its successor) is finished this will be a viable solution. Formation flying is the only near term solution. Thus, the idea of not one big satellite but rather many little satellites, each acting as an individual element of a much larger sensor array, is actively being pursued by National Aeronautical and Space Administration (NASA) and other agencies. These missions greatly improve the sensing capabilities available to NASA, while keeping the associated cost and risk down.

One of the main problems with formations and individual spacecraft in orbit is the amount of fuel consumed. Small differences in altitude between satellites will result in different velocities and orbits. This will require some satellites to expend more fuel in

order to continually maintain formation. In addition, if they are in different inclinations, they are in different orbital planes (planes that cut through the center of the Earth) and thus will naturally move farther apart or closer together depending on where they are in the orbit. Therefore fuel is required for formation maintenance, which inevitably leaves fuel tanks partially filled. Partially filled tanks can lead to fuel slosh which can degrade the pointing accuracy and therefore affect the science mission. Fuel slosh compensation is not an easy task and unfortunately is often neglected.

Satellites do not typically change their orbits unsupervised, but with formations there will have to be some level of autonomy where the satellites must make corrections to their orbits. Satellites will be required to change orbit to close gaps in formations or to avoid collisions from orbital debris. Gaps in the formation will occur as satellites run out of fuel or have failures. The control solution must be robust to account for uncertainty, produce stable motion, be easy to understand, and easy to implement. There has been much research on satellite formation control [1-8]; however these works assume satellites with robust individual control systems.

General Problems Controlling Single Satellites

To make a formation more stable, we must improve individual satellite control. This section will provide a brief overview of some of the problems with satellite control. Later greater detail will be presented on two specific problems: fuel slosh and solar snap.

External Disturbances

NASA engineers must contend with many types of disturbances when controlling a spacecraft. Low altitude satellites must fight gravitational and aerodynamic torque which can deteriorate their attitude accuracy and over time degrade their orbit. These

gravitational torques can be present on large satellites where gravity has more of an effect on one end of a satellite than the other.

In deep space or geosynchronous orbits, solar based disturbance becomes the dominate environmental disturbance since there is no friction or dampening. Also in these orbits, the gravity effects become negligible compared to effects produced by the sun. One of the major solar disturbances is solar snap which is thermally induced vibrations. Vibrations due to thermal changes have been known for a long time, but here on Earth there are few situations where these vibrations are of any interest.

Another disturbance due to the sun is called radiation torque. Photons emitted from the sun slam into the satellite (a force) which produces a torque about the center of mass of the craft. This radiation torque grows and wanes with solar flare activity, but none the less pushes and prods satellites until they slip behind the dark side of a planet or moon.

Parametric Uncertainty

Control systems are typically designed using mathematical models of the system to be controlled. Thus there is a direct correlation between the performance of a standard classical controller and the accuracy of the mathematical model of the plant. Unfortunately in satellite control, the inertial matrix is only known within 5% of its in orbit value. This model inaccuracy must be addressed in order to achieve the higher level of performance expected from the missions. Thus engineers are currently examining new control schemes which help to combat this problem and improve performance in the presence of modelling error.

Another problem that effects the inertia matrix is fuel slosh and fuel expenditure. Fuel can compose 40% of the total mass of a satellite prior to launch [9]. During orbit insertion,

some of this fuel will be burned to perform various Δv maneuvers to achieve a desired orbit. However, some of the fuel will be kept so that later orbital corrections can be made during the life span of the spacecraft. This remaining fuel poses a problem as it moves around inside of the satellite and thus changes the inertial matrix on orbit.

Literature Review

This section will provide a review of spacecraft that have had problems with fuel slosh and solar snap. The controllers developed to compensate for these disturbances will also be discussed. An analytical description of these disturbances will be provided in a later chapter.

Fuel Slosh

Modelling fuel slosh dynamics

Accurate analysis of the fuel requires the use of the Navier-Stokes equations, which are extremely computationally intensive. Since most satellites have a limited amount of processor power assigned to control, this is not practical for controller implementation. Therefore the most used models are the simplest ones such as the pendulum, mass-spring-dampener, and moving fuel tank walls [11]. For small motions, the objective is to reproduce the sloshing mode resonance frequencies. Typically a mass is attached to the wall of the fuel tank by a spring. The movement of this mass accounts for the free surface wave motion and the influence on the spacecraft's dynamics is represented by the force the spring exerts on the wall of the fuel tank. Dampeners are commonly introduced to represent the viscous dampening of the fuel. There is typically one set of mass-spring-dampeners for each sloshing mode that is being represented. Swirl effects of the fuel can also be included as reaction wheels with torsional springs attached. These models however

are only accurate in representing the sloshing modes when small linear or angular motions are performed. For accuracy during larger motions, better models must be utilized.

However these models are not ideal for control purposes. The typical pendulum model used for fuel slosh is derived from work done with rockets. Typically in this scenario, the rocket's main engine is firing and produces a local acceleration that pools the fuel at one end of the tank. The amount of fuel that remains pinned to this end of the tank is the stationary fuel mass fraction. The rest of the fuel is able to oscillate about the center line of the fuel tank in a pendulum-like motion. These oscillations are typically small since the fuel has only a small momentum compared to the forces produced by the local gravity.

Unfortunately this model is not as useful in the situation of satellite attitude control, since there is no external force produced by a thruster that is constantly firing throughout the life of the vehicle. Thus a model is needed in which the fuel has more freedom of movement, can exhibit a larger range of movement, and produce accurate results.

The work developed by Agrawal [10] for use on the INTELSAT IV, dual spin spacecraft, is a boundary layer model of the fuel for a spinning spacecraft. The author reached the conclusion that the analytical model gives the most accurate prediction of the fuel dynamics. The finite element model consisted of 81 nodes which represented a spherical fuel tank, shown in Figure 1-1. This fuel tank had a radius of 0.42 m, was 1.31 m from center of mass of the craft, and spun at 30 rpm. The solution for the fluid motion is obtained by solving three equations: an inviscous fluid problem, a boundary layer problem, and a viscous correction problem. The inviscid and viscous correction solutions are obtained by using finite element methods, while the boundary layer problem is solved analytically. The numerical results showed two slosh modes which could be accurately

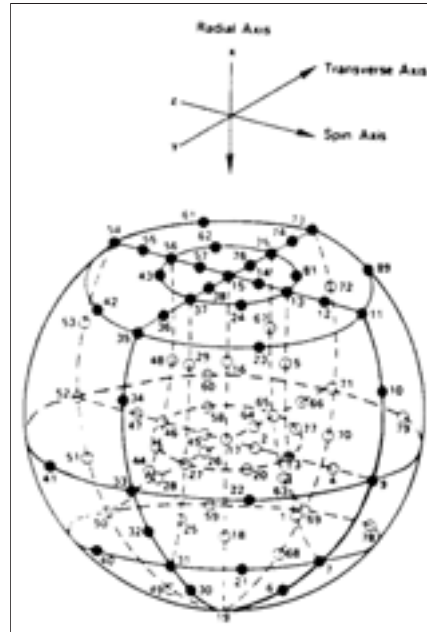


Figure 1-1. Finite element analysis of a fuel tank inside of a satellite.

modeled by the pendulum. However there were also first azimuth and elevation modes, lower modes to be inertial modes and higher modes to be a combination of inertial and slosh modes. These other modes would not be accurately modeled by a pendulum.

Controlling fuel slosh

Fuel slosh is an example of under-actuated control. The objective is to control both the rigid body dynamics of the spacecraft and the fluid dynamics of the fuel only having access to effectors that act directly on the spacecraft. The fuel's dynamics are controlled through the coupling of the two. One of the main problems with controlling this type of disturbance is that one can not measure the position, orientation, etc. of fuel in a satellite. One can only measure the effects the fuel slosh on the total system. The states and parameters of the fuel must be estimated, and a controller must try to account for the fuel's influence. Therefore it is difficult to control what can not be measured directly. Thus several types of passive methods are used to control slosh such as baffles [12], slosh

absorbers [13] to dissipate energy through sloshing, and large fuel tanks are broken up into smaller ones. However these steps add complexity and weight to a satellite's design. The added complexity increases the construction time, the cost, and the chances of mechanical failure of the satellite. The added weight also increases the cost of launching the satellite since it costs approximately \$10,000 per pound to put things into space.

Fuel slosh directly affects the performance of the attitude control system, in both pointing and tracking maneuvers. One of the simplest ways to handle fuel slosh is to identify the sloshing frequencies and eliminate the influence on the feedback with a notch filter. Unfortunately this denies the satellite the ability to operate in certain frequency ranges since the filter attenuates the slosh frequencies in the feedback. Hence, normal feedback without sloshing effects will also be attenuated. Also as fuel is expended and the sloshing frequency changes, the filter would need to be updated to notch out the new sloshing frequencies again.

Cho and McClaroach [14] developed a controller which utilized a common 2D pendulum model. Their equations only dealt with motion in a plane, but “a significant extension of what has been done previously, since it simultaneously controls both the rigid vehicle motion and the fuel slosh dynamics.” One of the major problems they set out to solve, which others were unsuccessful in solving, was the transverse motion of the craft. Their method utilized a Lyapunov function approach, and was able to control the system. They utilized additional control effectors to counter this movement (i.e., gas jet thrusters) while also controlling the pitch and suppressing the slosh dynamics.

Kuang and Leung [15] use a simple model of a liquid-filled spacecraft subject to external disturbances, and control it with a state feedback H_∞ controller. The spacecraft's

elliptical fuel tank is assumed to be completely filled with an ideal liquid in uniform vortex motion. Because of these assumptions, the slug of liquid can be represented by a finite number of variables. These variables can then be solved using Helmholtz equations. For the simulations, the liquid slug is located at the center of mass of the vehicle and surrounded by a viscous layer. An H_∞ attitude controller was then developed using the theory of Hardy-infinity optimal control theory. The controller was capable of attenuating the vortices in the fuel tank. The pointing accuracy remained constant regardless of small changes in the viscosity. The authors plan to next look at partially filled fuel tanks, which is a more difficult problem.

Thus far we have discussed more complex models to model liquids and said that in order to provide the level of control that a designer wants, one must use a complex model. An alternate example is Grundelius and Bernhardsson [16], who developed a minimum-time optimal controller for moving open rectangular fluid containers down an assembly line. Proper control of the containers is important. If the accelerations applied to the containers are too high, then the contents of the containers may spill out of the open top. This is obviously a loss to the company in material and results in a more expensive process. Also, if the top of the container is to be eventually sealed, the contents may spill on to an area where glue may be applied, thus making for a bad seal. This could result in customers disliking the product because of “cheap” design. When the speed of the system began to exceed two meters per second, fluid began to slosh out of the containers and the controller was not able to achieve the desired result. The authors incorporated a minimum-energy cost function into the design. Using only a simple linear model for the position of the surface of the liquid and the container, a controller that used a quadratic penalty on the

slosh and a quadratic penalty on the terminal state was able to produce the desired results without a complex hydro-dynamical model.

Do we need to worry about fuel slosh?

Fuel slosh is associated with liquid fuel which is used by thrusters. These types of thrusters are efficient and can provide a significant amount of control authority. Slosh forces can be eliminated via the use of alternate actuators. Let us take a look at some of the various types of control actuators available to satellites.

Wisniewski and Blanke [17] developed a three axis magnetorquer controller for satellites subject to a gravity gradient while in a polar orbit about the Earth. Magnetorquers are an attractive form of control for small inexpensive satellites since they use the Earth's magnetic field to control the orientation of the spacecraft. However magnetorquers produce very small torques, and their principle of operation is nonlinear and difficult to use. They are difficult due to the fact that control torques can only be generated perpendicular to the Earth's magnetic field. Also the size of the spacecraft might prohibit the use of magnetorquers since they produce low torque.

Chen et al. [18] developed an optimal controller which utilizes both thrusters and magnetorquers to unload extra momentum that has accumulated and saturated the reaction wheels of a spacecraft. Their method separates the required torques for the thrusters and magnetorquers using a simple optimal algorithm. Simulations conducted by Chen showed a 20% fuel savings using this method.

Liquid [19] apogee motors are common in geosynchronous spacecraft design. This results in liquid fuel constituting almost half of the vehicles' mass. In space, liquid fuel motion greatly influences the attitude stability and control. The level and nature of these

influences are dependent on the geometry and location of the fuel tanks, the ratio of spacecraft mass to fuel mass, the fill level of the fuel tanks, and the physical properties of the fuel itself. They also depend on the flexible structure of the spacecraft and the behavior of the estimation and control schemes that are present. Unfortunately, accurate prediction of the fuel dynamics is a difficult problem due to the complexity of the hydrodynamical equations of motion. LEASAT, which was launched in 1984, experienced attitude instability during the pre-apogee injection phase which immediately followed the activation of despin control. The instability was the result of the interaction between the lateral sloshing modes and the attitude control.

Pulsed plasma thrusters (PPT) are becoming a more popular control solution for small spacecraft [20]. This fact is due to the small electrical consumption and the use of a solid teflon fuel. The fuel is attractive because it is stable, compact, and does not provide complications due to sloshing liquids. EO-1 provided a successful demonstration of an attitude control system using these PPT's shown in Figure 1-2 and Figure 1-3. PPT's are capable of very high pulse width modulation frequencies (PPT's are only capable of on-off or bang-bang type of control). Although PPT's have some very favorable properties, they still have the liability of using a non-renewable fuel and are only capable of operating

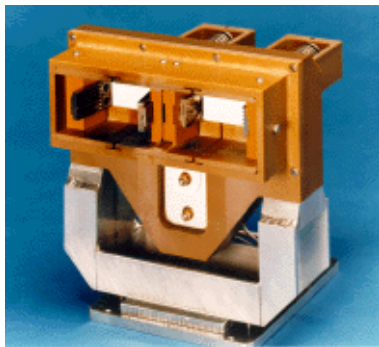


Figure 1-2. PPT for EO-1.

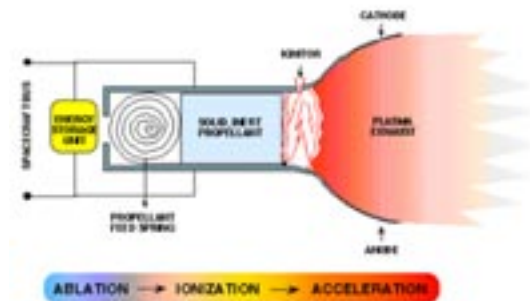


Figure 1-3. Diagram of a PPT.

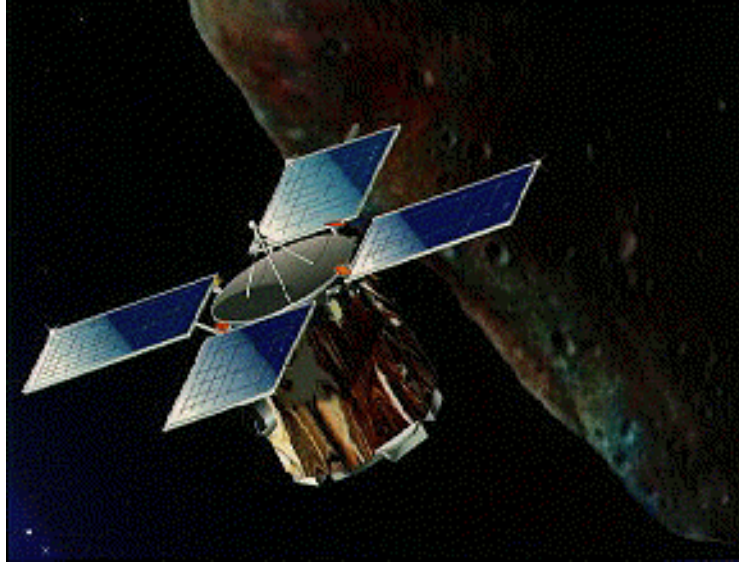


Figure 1-4. NEAR spacecraft orbiting the asteroid Eros.

on small spacecraft.

Another technology that shows promise for satellites ranging from 100 grams to 500 kg is MEMS engines [21]. MEMS (micro-electrical-mechanical systems) is an exciting new technology field which currently is beginning to make its way out of the R&D labs and into commercial products. MEMS are microscopic mechanical-electrical systems that are developed using a combination of micromachining and standard integrated circuit design. Some of the common MEMS components on the market are accelerometers, gyros, and GPS receivers. Here MEMS rocket engines are developed for space applications. The MEMS engines are the size of a human hair and have a thrust-to-weight ratio hundreds of times greater than the space shuttle.

Although there are many good alternatives to liquid thrusters for small spacecraft, liquid fueled thrusters still provide the best source of high thrust capability for both small

and large spacecraft. Therefore fuel slosh will always be present in these types of systems and thus affect high precession pointing and tracking accuracy.

Missions involving fuel slosh disturbances

We now understand why fuel slosh exists. Next let us look at missions where slosh has been observed. The Near Earth Asteroid Rendezvous (NEAR) spacecraft was sent to orbit and observe the Eros asteroid on December 23, 1998. The spacecraft, shown in Figure 1-4, had to fly by the asteroid following an unsuccessful firing of its main engine a few days earlier. A subsequent successful firing of the engine put NEAR on course to rendezvous with Eros to begin its planned year long orbital mission starting in mid-February 2000. The reason the main engine firing did not work is the on board control system terminated the burn when sensors registered higher than expected lateral accelerations. These accelerations were later determined to be caused by the spacecraft's fuel. The NEAR's mission had to be delayed for over a year because the control system was incapable of handling the coupled dynamics.

Situations like NEAR do not just happen around asteroids, but with the advent of the International Space Station similar problems could arise. As resupply ships travel to and from the station, similar problems could occur and a ship could crash into the station. This scenario is similar to the accident that occurred with MIR and one of its resupply vessels. Although that incident was not caused by fuel slosh, it still does represent a potential problem.

Thermally Induced Vibrations: Solar Snap

Thermally induced vibration or solar snap occurs when dynamics are excited by a rapid temperature change. This type of vibration has been known for some time but rarely

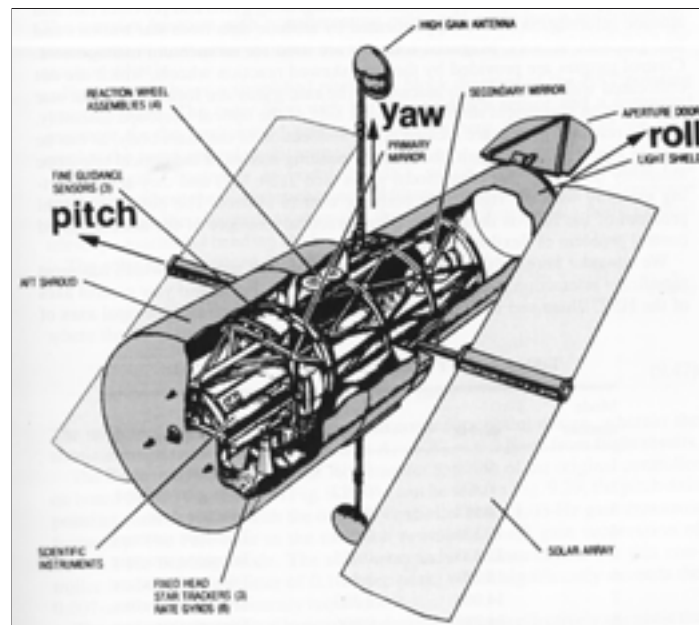


Figure 1-5. Hubble Space Telescope diagram.

appears in applications on Earth. However in the harsh, irradiated vacuum of space, without the presence of an atmosphere to dampen the vibrations, this disturbance can adversely affect the performance of high precision instruments and equipment. This section, like the proceeding one, will present missions that where adversely affected by solar snap and control systems that were designed to compensate for it.

Controlling thermal vibrations

The Hubble Space Telescope (HST), shown in Figure 1-5, was launched on 24 April 1990 with the hopes of making numerous new scientific discoveries. The HST was a 13 ton, free-flying spacecraft with a pointing precision requirement of 0.007 arc-sec over a 24 hour period (which is the most stringent requirement ever imposed). During its initial checkout period, a pointing “jitter” was discover. The jitter was produced by a thermally induced vibrations imitating from the solar panel array. Later in December 1993 the solar

arrays were replaced by the Space Shuttle *Endeavour* in an attempt to reduce the jitter. The solar panels exhibited an end-to-end bending oscillation when the spacecraft transitioned between sunlight and shadow. They also had a sideways oscillation on the day side of the Earth. At its worst, the 20 ft. solar arrays would deflect as much as 3 ft. This would have the effect of rendering any long time exposures of over 25 min. pointless. The original controller was a digital PID controller with an FIR filter in the rate path to attenuate high-frequency, main-body bending modes. This design was incapable of handling the induced disturbances from the solar arrays, which caused a peak pointing jitter of 0.1 arc-sec. ([22], p. 583)

Wie and Liu [25] redesigned the attitude control system using an H_∞ controller. This controller was designed to attenuate the two major modes from the solar panels which operated at 0.12 and 0.66 Hz. Through a trial-and-error process, they were able to tune the weights of the controller and properly achieve the mission requirements.

Singhose et al. [23], while working at M.I.T., designed a method to minimize vibrations in flexible spacecraft which utilized command profiles. The commands are designed by the following:

1. Formulate the equations of motion for a system subject to a sequence of impulses.
2. The equations are solved for the sequence of impulses that results in the smallest amount of residual vibrations in the system.
3. The sequence is convolved with the desired input to generate the command sequence which is issued to the real system.

The command sequence is a pulse width modulation (i.e., on-off sequence) of commands for the spacecraft's thrusters. Later at Georgia Tech., Ooten and Singhose [24] extended the work to sliding-mode control. The drawback for this work is, although flexible space-

craft are mentioned, all simulations involve only a mass-spring-damper system. Also, solving the equations of motion (step 2) for a nonlinear, flexible satellite would be very difficult.

Missions which involved solar snap

Originally spacecraft were very simple, but as technology progressed into the 1960s more appendages were being put on. These structures offered new challenges in controlling flexible structures and vibrations. Below is a table of spacecraft that had their performance compromised by thermal vibrations. These satellites do not include military and Soviet spacecraft.

Table 1-1: Spacecraft with thermally induced disturbances^a.

Spacecraft	Date	Description
Ulysses	1990-1991	Spin-stabilized solar probe experiences unexpected nutation due to thermally induced vibrations of beryllium-copper axial boom
Upper Atmosphere Research Satellite	1991	“Thermal snap” at orbital night/day transition attributed to rapid heating of large single solar array
Hubble Space Telescope	1990	Jitter phenomena attributed to thermally induced vibrations of FRUSA-type solar arrays
LANDSAT-4/5	1980s	Thermal elastic shock experienced during orbital night/day transition due to disturbance of large single solar array.
Communications Technology Satellite	1978	Three axis stabilized satellite experienced “thermal elastic shock” during orbital night/day transition
Voyager	1977	Low frequency oscillations of PRA booms during LEO operations attributed to “thermal flutter”
Apollo 15	1971	Thermally induced vibrations of twin Bi-STEM booms filmed during 64th lunar orbit by astronauts.
Explorer 45	1971	Unexpected nutation of spin stabilized satellite attributed to thermal bending of four radial booms.
NRL 161, 163, and 164	1969	Thermally induced vibrations of STEM booms used for gravity-gradient stabilization leads to large satellite motions.
OGO-IV	1967	NASA Orbiting Geophysical Observatory Satellite STEM booms experience strong thermally induced vibrations during orbital night/day transition.
OV-10	1966	US Air Force gravity-gradient satellite using STEM booms equipped with tip masses experiences thermal flutter and complete inversion in orientation.

Table 1-1: Continued

Spacecraft	Date	Description
Explorer XX	1964	NASA spin-stabilized satellite experiences spin decay due to interaction between STEM boom thermal bending and solar radiation pressure.
1964-83D and 1963-22A	1964/ 1963	Dynamic thermal bending of STEM booms on APL gravity-gradient satellites.
Aloutte 1	1962	Canadian spin-stabilized satellite experiences spin-rate decay due to thermal structural response of STEM booms similar to Explorer XX.

a. Information contained in this table comes from Thorten [26] on page 345.

“In spite of the practical importance of thermally induced vibrations to the successful operation of spacecraft, recurrent vibration problems from the 1960s to the present day suggest that the phenomena is either underemphasized or not well understood in spacecraft design” (Thorten [26], p. 344)

Parametric Uncertainty

A very simple way to model the effects of fuel and thermal vibrations is to represent them as external disturbances (i.e., deterministic noise). Depending on the mass fuel ratio, fuel slosh affects the mass distribution of the vehicle which affects the inertia matrix. Since control systems are model based, this will directly affect the level of control. If a control system could be designed that provided generic control, or model independent control, then the effects of these disturbances could be negated.

This idea of generic control was the subject of work done by Walchko and Mason [27] utilizing fuzzy logic. The idea was that fuzzy logic is rule based and not model based control. Therefore fuzzy logic would be capable of providing better control in the presence of uncertainty. Although several different configurations of fuzzy logic were developed (PD, PID, and sliding mode) none produced the desired results when there was a significant amount of uncertainty. In fact it was shown that the two best were sliding mode

and fuzzy sliding mode. However the non-fuzzy sliding mode had the added advantage that stability could be shown for the controller, whereas fuzzy logic lacks any general method to prove stability

Petroff et al. [28] attempted to address this problem of fuzzy logic stability. They looked at simple linear single input single output systems and were able to develop a stability analysis. However, this method did not naturally extend to multi-input multi-output systems and definitely not to nonlinear systems. Thus although fuzzy logic appeared to be attractive as a model independent control architecture, it was extremely weak with respect to stability analysis. This fatal flaw ultimately disqualified it for our applications.

Summary and Overview of the Thesis

What is needed are controllers that are easy to adapt from one satellite to another but still retain their robustness and efficiency. Problems due to simple control systems being implemented on satellites are numerous. This usually results in a redesign of the control system which is costly and a waste of time and resources. Now couple these problems with the difficulties of formation flying and things get very risky.

This work will primarily focus on fuel slosh and solar snap. Both of these phenomena are interesting in that they have rarely been observed. Rather, engineers have examined the data from satellites that have experienced unexplained problems and deduced these were the causes. Neither of these problems is generally looked at when a control system is developed, but rather they are dealt with only when they appear. This attitude proves to be a problem when applied to formation control. The possibility of one satellite going

unstable due to these phenomena then introduces the possibility of that satellite affecting the entire formation, thus endangering the mission.

Chapter 2 will provide a more in depth discussion of fuel slosh. Various models will be described that are used to model fuel slosh for simulation purposes and models that are used in controllers.

Chapter 3 will cover the interactions of solar panels with satellite dynamics. Thermally induced vibrations are described by the solution to their partial differential equations and modal equations.

Chapter 4 will cover satellite attitude dynamics and control. This chapter will introduce the reader to the fundamental dynamics of satellites.

Chapter 5 will introduce the basics of sliding mode control and other attitude control algorithms and control hardware. The specific controller for this work will also be derived and discussed.

Chapter 6 will provide an overview of the simulations. All parameters, equations, and assumptions will be presented here. Also all results will be discussed in this chapter.

Chapter 8 will provide conclusion to the work done.

CHAPTER 2

CONTROLLING THE UNKNOWN: FUEL SLOSH

The introduction covered much of the problems associated with fuel slosh and highlighted missions where this phenomena occurred. This chapter will expand on the introduction by focusing more on the fuel slosh disturbance. Two models were developed, one for use in the simulation to provide the disturbance torques on the satellite and another model to help the controller account for the fuel slosh disturbances.

Introduction

The sloshing behavior of liquids and its modeling has been studied in many different areas: fuel slosh in aerospace applications, movement of fluid filled containers in industrial/manufacturing applications, earth quakes, vehicle and ship dynamics. Unfortunately the choice of a model is not a simple one. Most analytic models that try to accurately represent the dynamics are three dimensional partial differential equations. These models are heavily dependant on boundary conditions and computationally expensive to solve. Thus they are typically not used in controller design. However, the nonlinear dynamics that appear from these complex models are important when designers wish to move large amounts of fluid rapidly.

The aerospace industry has looked at controlling the effects of fuel movement in aircraft fuel tanks for years. Most of the ideas (i.e., baffled fuel tanks, breaking one big tank up into many smaller ones, etc.) can be traced back to standard aircraft design.

The interaction of the fuel and the spacecraft is more complex in three-axis stabilized systems than spin-stabilized systems [29]. This is due to the fact that this system has constraints that reduce the number of degrees of freedom, simplify the model structure, and is numerically easier to compute. However, spacecraft will typically operate in several modes of operation over their life span and thus it is important to be able to control fuel slosh in any mode of operation.

Modelling Fuel Slosh Dynamics

Models for fuel slosh have not changed much since Abramson started in 1961 [9]. Models over the years have been: (single and multi) mass-spring-damper, pendulum, liquid slug, and CFD/FEA models. Typically, the controller uses a model to account for the fuel present inside of the spacecraft. Unfortunately, the models listed are too simple or based on a static representation of the fuel and do not attempt to modify the parameters of the model during run time.

The coordinate system for the fuel is typically the body coordinate system [29]. This has the result of making the spacecraft's motions appear as a body-force field in the Navier-Stokes equations. The boundary conditions at the wall also become stationary with only the free surface remaining transient while the viscous forces provide a dampening of the liquid. The fuel motion is influenced by four classes of forces: gravity, inertia, viscous, and capillary forces.

The early work with slosh dynamics and spacecraft centered around liquid filled rockets and their stability [29]. This topic is still of interest and the assumptions [9] for this problem typically are:

- The fuel has small displacements, velocities, and slopes of the liquid-free surface.

- The fuel tank is rigid.
- The fuel is assumed to be a nonviscous liquid.
- The fuel is also modelled as an incompressible, homogenous fluid.

Propellant slosh is the induced motion of liquid due to acceleration of the container. This motion along with its reactive forces can deteriorate the pointing performance. This is especially true for systems with large fuel/weight ratio. Due to the nonlinear dynamics of sloshing, it is not always possible to compensate for its affects using a standard attitude controller.

Typically a pendulum system, shown in Figure 2-1, is used to model a single sloshing mode. Since slosh dynamics typically display several modes, several pendulums should be included in the model of the system. Thus the more dominate modes are modelled, the more realistic the simulation.

A problem with this type of model is determining the values of the pendulum lengths, masses, springs, and dampeners. One way to do this is to look at actual flight data where you can determine the modal frequencies. Assuming the amount of mass in the slosh mode, the value of the spring constant can be solved. The damper value is picked so that the system's oscillations die out after an appropriate amount of time.

The main problem with liquid slosh dynamics is estimating the hydrodynamic pressure distribution, forces, and moments. One reason that this is so difficult is the dynamic boundary conditions at the free surface varies with time in a manor not known a priori, as depicted in Figure 2-2. Hydrodynamic pressure in many rigid tanks is composed of two parts. The first part is the fluid that moves with the tank in unison. The second component is the sloshing at the free surface. This component is typically modelled as a

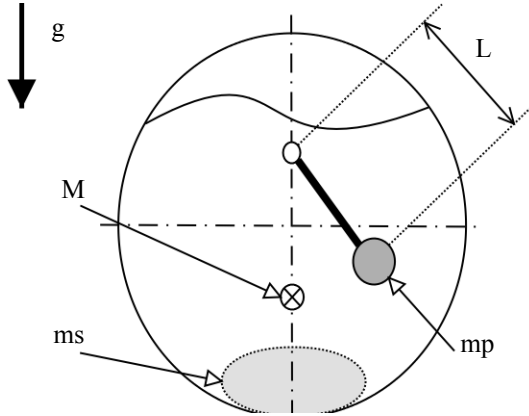


Figure 2-1. Traditional fuel slosh pendulum model where M is the total fuel mass center, mp is the pendulum mass, ms is the stationary mass fraction, L is the pendulum arm length, and g is the local acceleration due to some external force.

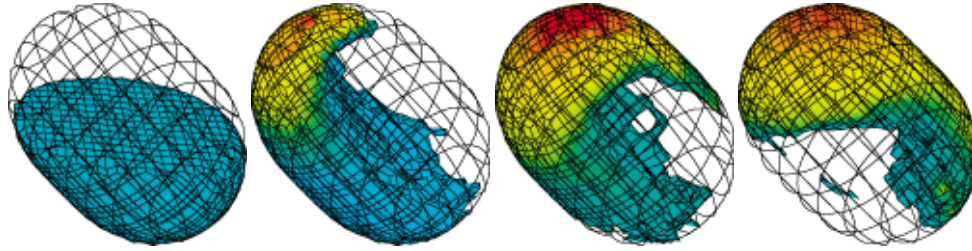


Figure 2-2. Propagation of fuel slosh in an elliptical tank over time. These images demonstrate the difficulty in modelling the fluid movement with simple representations.

mass-spring-damper or pendulum. Suggested values for the pendulum shown in Figure 2-1 are given by EL-Sayad et al. [30] as

$$L = \frac{R}{1.84} \coth\left(1.84 \frac{h}{R}\right) \quad mp = M_{total} \left(\frac{R}{2.2h}\right) \tanh\left(1.84 \frac{h}{R}\right) \quad [2-1]$$

where L is the length of the pendulum, R is the radius of the tank, and mp is the mass at the end of the pendulum.

Fuel Slosh Models

Two different models for the fuel will be utilized in this work. The first model will be included in the satellite dynamics and provide the proper disturbance torques on the system. The second model will be computationally simpler, but still contain the important aspects of the fuel's disturbance on the system.

Simulation Model

The model utilized is based on the mass-spring-damper models by Sidi [9] and Hughes [31] shown in Figure 2-3. Each sloshing mode will be composed of a mass, a spring, and a damper. The mass represents the mass of fuel in a specific mode. The spring represents the force exerted on the tank wall by a sloshing mode. The damper represents the baffling in the fuel tank that dissipates the sloshing movement.

The actual dynamics of this disturbance will be derived later when the dynamics of the satellite are covered. This is due to the fact that both the dynamics of the spacecraft and fuel are tightly coupled and it is easier to present them all at once.

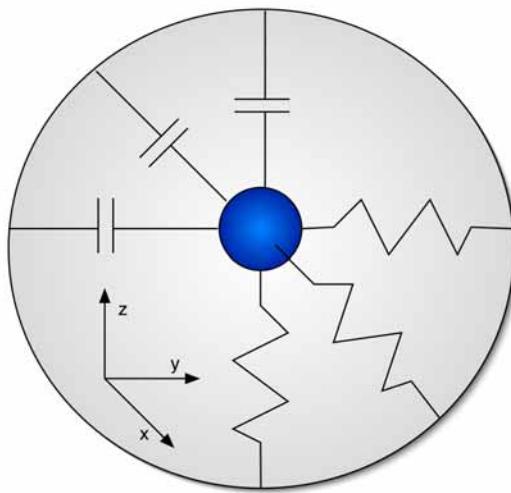


Figure 2-3. Diagram of a single sloshing mode, with its mass attached to the fuel tank walls by springs and dampers aligned along the x, y, and z axes.

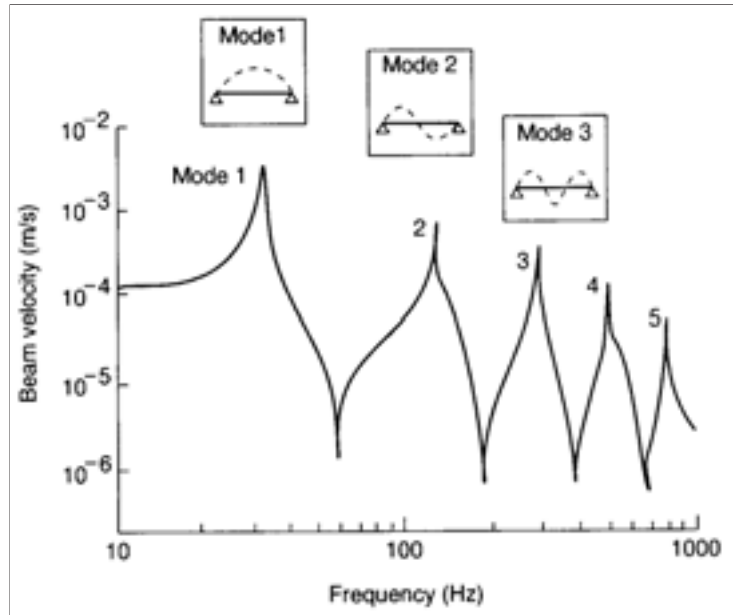


Figure 2-4. Plot of the effect of multiple modes in the velocity of a simply supported beam. This is representative of the fuel slosh disturbance in flight data.

Controller Model

The model used in the controller needs to be capable of representing several sloshing modes, but can not be computationally expensive. Figure 2-4 provides a frequency domain visualization of the proposed model. The model represents the sum of the slosh modes as a band pass filter. This will allow us to incorporate the effects of the multiple sloshing modes without having to specifically identify their individual characteristics.

A simple band pass filter is the combination of a low pass and high pass filter in series.

The standard transfer function for this filter is as follows:

$$\frac{as}{as + 1} \frac{1}{bs + 1} = \frac{as}{abs^2 + (a + b)s + 1} \quad [2-2]$$

$$a = \frac{1}{freq_{highpass}} \quad b = \frac{1}{freq_{lowpass}} \quad [2-3]$$

where the transfer function on the left is a high-pass filter and the one on the right is a low-pass filter. The variables a and b are the inverse of the high and low pass cutoff frequencies. Putting [2-2] into state space form yields

$$\begin{bmatrix} \dot{x} \\ x \\ x \end{bmatrix} = \begin{bmatrix} -\frac{(a+b)}{ab} & -\frac{1}{ab} \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ x \\ x \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u \quad [2-4]$$

$$y = \begin{bmatrix} \frac{1}{b} & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ x \\ x \end{bmatrix} \quad [2-5]$$

The interesting thing to note about 2-5 is the output given by the model is velocity and not position. This is due to the derivative term (i.e., s in the numerator) in 2-3. This is a reasonable characteristic, since the fuel slosh effects show up in the rate terms and not the position terms.

CHAPTER 3 DYNAMICS OF SOLAR ARRAYS

As discussed in the introduction, thermally induced vibrations are a source of problems for satellite attitude control. This chapter will provide the theoretical development of thermally induced vibrations in long, thin beams. These results will finally be used to generate the solar snap disturbance in the simulations.

Introduction

Research on thermally induced vibrations began back in the mid-1950's, before Sputnik orbited the Earth in October 1957. However it was not until the 1960's that applications for this new idea appeared. Initially spacecraft appendages (i.e., solar panels, communication arrays, booms, etc.) were thought to be simple systems like those used by Sputnik (Figure 3-1), but slowly it became apparent that they were much more complicated. Strange and unexplained spacecraft behavior was now being attributed to this new nonlinear behavior being observed.

Thermally Induced Vibrations

First, this section will derive and solve the partial differential equations associated with solar snap. Then differential equations will be obtained from the PDEs which are utilized in the simulations.

The derivation that follows will parallel work done by Thorten [26] and Rietz [67]. However, there was great difficulty in actually following their work. Fortunately, all key equations were independently derived. The results in this work were also similar to

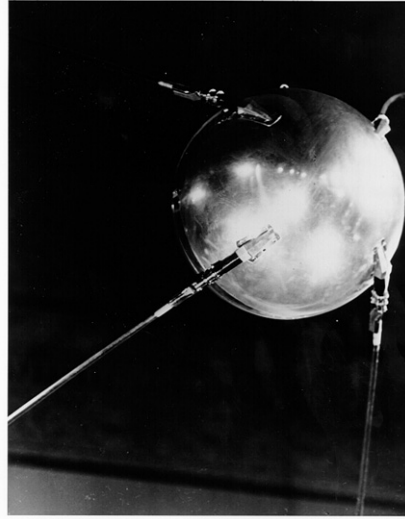


Figure 3-1. The Soviet satellite Sputnik which orbited the Earth in October 1957.

derivations performed on cantilever beams excited by sources other than thermal vibrations by Greenburg [32], Kreyszig [33], Meirovitch [34], and Zill and Cullen [35].

Heat Flux

Thermal vibrations are excited by a sudden change in heat flux to the object. In this work, heat flux changes greatly during the day-to-night and night-to-day transitions. Here the satellite passes into and out of the Earth's shadow.

The Earth's shadow is actually composed of two regions shown in Figure 3-2. The penumbra is a region of partial shadow and the umbra is a region of full shadow. When a satellite is in low Earth orbit (LEO), the penumbra is small and the satellite essentially transitions from the day side to the night side (i.e., umbra). This sudden change from day-to-night or night-to-day results in larger thermal vibrations. This is due to the sudden change in heat flux, much like turning on or off a light switch.

The amount of heat flux (q) seen by a satellite is composed of solar flux from the sun (q_s), Earth emitted radiation flux¹ (q_e), and Earth's reflected heat flux (q_a) from the sun.

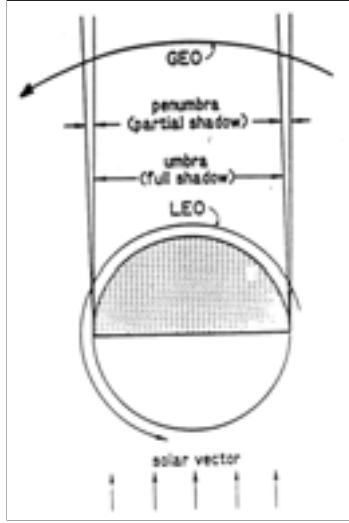


Figure 3-2. Diagram of the penumbra and umbra regions which block a satellite from sunlight.

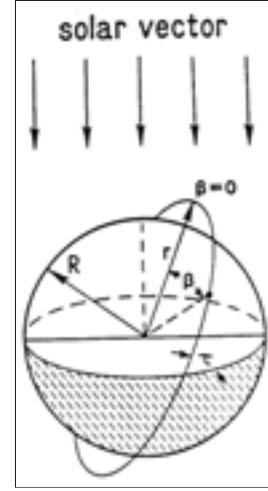


Figure 3-3. Another view showing the path of a satellite and how its orbit does effect the heat flux it is exposed to.

This amount changes with the satellite's position in orbit and its orientation relative to the solar flux vector (see Figure 3-3).

$$q = q_s + q_e + q_a \quad [3-1]$$

$$q_s = 1350 a_s \cos(\psi) \quad [3-2]$$

where a_s is the surface absorptivity for solar radiation and ψ is the angle between the surface normal and the solar flux vector.

Summing Forces

The solar panel is modelled as an Euler-Bernoulli cantilever beam subjected to a thermal moment. The beam, a cross section is shown in Figure 3-4, will have the following assumptions:

- Plane cross sections before bending will remain planes after bending.

1. The radiation referred to here is black body radiation.

- Lateral displacements in v and w are due to bending only. Deformations from shearing and changes in transverse beam dimensions are negligibly small.
- The beam is a Euler-Bernoulli cantilever beam.
- The system is uncoupled, meaning that changes in beam orientation do not effect the amount of heat flux entering the system.
- The temperature gradient is only one dimensional (i.e., through the thickness of the beam).

The moments on the system are defined as

$$M_z = \int_A \sigma_x y dA \quad [3-3]$$

$$M_T = \int_A E \alpha \Delta T y dA \quad [3-4]$$

Linear strain displacement is as follows:

$$\epsilon_x = \frac{\partial u}{\partial x} = \frac{du_0}{dx} - y \frac{\partial^2 v}{\partial x^2} - z \frac{\partial^2 w}{\partial x^2} \quad [3-5]$$

Hooke's uniaxial law is

$$\sigma_x = E \epsilon_x - E \alpha \Delta T \quad [3-6]$$

Then substituting [3-5] into [3-6] yields:

$$\sigma_x = E \left(\frac{du_0}{dx} - y \frac{\partial^2 v}{\partial x^2} - z \frac{\partial^2 w}{\partial x^2} \right) - E \alpha \Delta T \quad [3-7]$$

Now substituting [3-7] into [3-3] produces

$$M_z = \int_A \left(E \left(\frac{du_0}{dx} - y \frac{\partial^2 v}{\partial x^2} - z \frac{\partial^2 w}{\partial x^2} \right) - E \alpha \Delta T \right) y dA = -EI \frac{\partial^2 v}{\partial x^2} - EI_{yz} \frac{\partial^2 w}{\partial x^2} - M_T \quad [3-8]$$

where

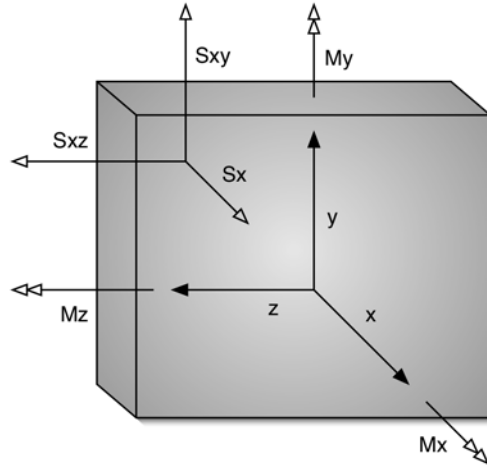


Figure 3-4. Cross section of solar panel with coordinate systems superimposed. Note that the S's in the figures represent σ 's and M's are moments.

$$I_z = \int_A y^2 dA \text{ and } I_{yz} = \int_A yz dA \quad [3-9]$$

Next, summing the forces in the y direction in Figure 3-5 results in

$$\sum F_y = -f(x) - V(x, t) + V(x, t) + \frac{\partial}{\partial x} V(x, t) dx + m \ddot{u}(x, t) = 0 \quad [3-10]$$

Note that from here on out, the terms in parenthesis will be dropped unless it is felt that they are necessary for clarity.

$$\rho A \frac{\partial^2 u}{\partial t^2} dx + \frac{\partial V}{\partial x} dx = f dx \quad [3-11]$$

$$\rho A \frac{\partial^2 u}{\partial t^2} + \frac{\partial V}{\partial x} = f \quad [3-12]$$

Summing the moments about the center of mass of the beam yields:

$$\sum M = M + \frac{\partial M}{\partial x} dx + \frac{1}{2} V dx + \frac{1}{2} \left(V + \frac{\partial V}{\partial x} dx \right) dx - M = 0 \quad [3-13]$$

$$\frac{\partial M}{\partial x} + V = 0 \quad [3-14]$$

$$-\frac{\partial M}{\partial x} = V \quad [3-15]$$

Substituting [3-8] into [3-5] and ignoring the middle term (since we are only looking at motion in the u direction and not the w direction).

$$\frac{\partial}{\partial x} \left(EI \frac{\partial^2 u}{\partial x^2} + M_T \right) = V \quad [3-16]$$

Now taking [3-12] and substituting [3-16] gives us:

$$\rho A \frac{d^2 u}{dt^2} + \frac{\partial}{\partial x} \left(\frac{\partial}{\partial x} \left(EI \frac{\partial^2 u}{\partial x^2} + M_T \right) \right) = f \quad [3-17]$$

$$\rho A \frac{d^2 u}{dt^2} + EI \frac{\partial^4 u}{\partial x^4} + \frac{\partial^2 M_T}{\partial x^2} = f \quad [3-18]$$

This equation can be further simplified by assuming that no uniform load ($f(x)$) on the solar panel exists and realizing the thermal moment is not a function of x , thus its second derivative is zero.

$$f = 0 \quad M_{Txx} = 0 \quad \beta^4 = \frac{EI}{\rho A} \quad [3-19]$$

Incorporating these, [3-18] becomes

$$u_{tt} + \beta^4 u_{xxxx} = 0 \quad [3-20]$$

where a subscript of x or t refers to differentiation with respect to that variable. The boundary conditions of the system are as follows:

$$u(x, t) = 0 \quad [3-21]$$

$$u_x(0, t) = 0 \quad [3-22]$$

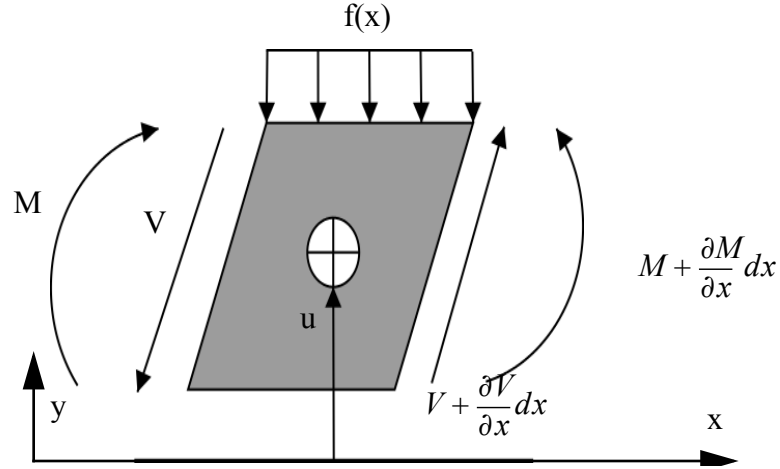


Figure 3-5. Free body diagram of a section of the beam of size dx where $M(x,t)$ is a moment, $V(x,t)$ is a shear force, $f(x)$ is a force per unit length, and $u(x,t)$ is the height of the mass element at position x at time t .

$$M_z(L) = -EIu_{xx} - M_T = 0 \quad [3-23]$$

$$V(L, t) = EIu_{xxx} + M_{Tx} = 0 \quad [3-24]$$

Table 3-1: Solar panel thermal material properties.

Variable	Description	Units
k	thermal conductivity	$\frac{W}{mK}$
q	heat flux	$\frac{W}{m^2K}$
ρ	density	$\frac{kg}{m^3}$
c	heat capacity	$\frac{J}{kgK}$
κ	thermal diffusivity	$\frac{m^2}{s}$
α	coefficient of thermal expansion	$\frac{1}{K}$

Nonhomogeneous Boundary Conditions

Unfortunately the boundary conditions for the system are nonhomogeneous and in order to solve the equations this must be corrected. Getting rid of the nonhomogeneous boundary conditions will be accomplished by changing the dependent variable.

$$u(x, t) = v(x, t) + \psi(x) \quad [3-25]$$

Substituting [3-25] into [3-23] yields:

$$u_{xx} = v_{xx} + \psi_{xx} = -\frac{M_T}{EI} \quad [3-26]$$

The second derivative of u must be equal to zero to make the system homogenous, thus ψ must negate the right hand side of the equation.

$$\psi_{xx} = -\frac{M_T}{EI} \quad [3-27]$$

Now we must solve for $\psi(x)$, thus integrating [3-27] twice results in an equation with two constants that can be solved by looking at the initial conditions.

$$\psi = -\frac{M_T}{2EI}x^2 + C_1x + C_2 \quad [3-28]$$

$$\psi(0) = 0 = C_2 \quad \psi_x(0) = 0 = C_1 \quad [3-29]$$

$$\psi(x) = -\frac{M_T}{2EI}x^2 \quad [3-30]$$

Thus [3-25] can be rewritten as

$$u(x, t) = v(x, t) - \frac{M_T x^2}{2EI} \quad [3-31]$$

Our new beam equation and boundary conditions using [3-31] are shown below. Notice how this is now a simpler homogeneous equation to solve.

$$v_{tt} + \beta^4 v_{xxxx} = 0 \quad [3-32]$$

$$v(0, t) = 0 \quad [3-33]$$

$$v_x(0, t) = 0 \quad [3-34]$$

$$v_{xx}(L, t) = 0 \quad [3-35]$$

$$v_{xxx}(L, t) = 0 \quad [3-36]$$

Separation of Variables

Now [3-32] with its homogeneous boundary conditions can be solved via separation of variables. We will assume that the function $v(x, t)$ can be broken up into two separate functions. Each of these two functions are dependent on only one variable, either position (x) or time (t).

$$v(x, t) = F(x)G(t) \quad [3-37]$$

Substituting [3-37] into [3-32] gives us an equation which can be separated.

$$F_{xxxx}G + \beta^4 FG_{tt} = 0 \quad [3-38]$$

$$-\frac{G_{tt}}{G} = \frac{F_{xxxx}}{\beta^4 F} = \omega^2 \quad [3-39]$$

$$\beta^4 = \frac{EI}{\rho A} \quad [3-40]$$

Since each side of the equation is dependent on a single variable, it must be equal to a constant. We will choose this constant to be ω^2 , which is squared to simplify the derivation. Each of the two sides can now be solved, which gives us one differential equation in x and one differential equation in t .

$$G_{tt} + \omega^2 G = 0 \quad [3-41]$$

$$F_{xxxx} - \omega^2 \beta^4 F = 0 \quad [3-42]$$

Solving the position function $F(x)$

We will begin by solving the $F(x)$ equation first. Assuming that the solution to this differential equation is an exponential, the following solution is found.

$$F(x) = B_1 \cosh(\beta_n x) + B_2 \sinh(\beta_n x) + B_3 \cos(\beta_n x) + B_4 \sin(\beta_n x) \quad [3-43]$$

$$F(0) = B_1 + B_3 = 0 \quad [3-44]$$

$$F_x(0) = B_2 + B_4 = 0 \quad [3-45]$$

$$F_{xx}(L) = B_1(\cosh(\beta_n L) + \cos(\beta_n L)) + B_2(\sinh(\beta_n L) + \sin(\beta_n L)) = 0 \quad [3-46]$$

$$F_{xxx}(L) = B_1(\sinh(\beta_n L) - \sin(\beta_n L)) + B_2(\cosh(\beta_n L) + \cos(\beta_n L)) = 0 \quad [3-47]$$

Combining the last two equations into a matrix give us:

$$\begin{bmatrix} \cosh(\beta_n L) + \cos(\beta_n L) & \sinh(\beta_n L) + \sin(\beta_n L) \\ \sinh(\beta_n L) - \sin(\beta_n L) & \cosh(\beta_n L) + \cos(\beta_n L) \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} = 0 \quad [3-48]$$

The determinate of this matrix gives us the characteristic equation from which we can solve for the values of $\beta_n L$ that result in zero. The characteristic equation is shown below and a plot (shown in Figure 3-6) which can be used to graphically solve the equation.

$$\cosh(\beta_n L) \cos(\beta_n L) + 1 = 0 \quad [3-49]$$

Table 3-2: First four eigen modes of the beam

n	1	2	3	4
$\beta_n L$	1.875104	4.694091	7.854757	10.995541

It is now possible to solve the equations in the matrix as a ratio B_2/B_1 . Thus it is possible to solve for 3 of the constants in terms of the fourth.

$$F_n(x) = \cosh(\beta_n x) - \cos(\beta_n x) + \alpha_n (\sinh(\beta_n x) - \sin(\beta_n x)) \quad [3-50]$$

$$\omega_n = \frac{(\beta_n L)^2}{L^2} \sqrt{\frac{EI}{\rho}} \quad \beta_n^4 = \frac{\rho A \omega_n^2}{EI} \quad \alpha_n = \frac{B_2}{B_1} = \frac{\sin(\beta_n L) - \sinh(\beta_n L)}{\cos(\beta_n L) + \cosh(\beta_n L)} \quad [3-51]$$

This represents the n^{th} spatial solution or mode shape. In order to calculate the complete response of the beam, a summation over all modes must be done.

$$F(x) = \sum_{n=1}^{\infty} a_n F_n(x) \quad [3-52]$$

The coefficient a_n is found by the orthogonality property of the modes. This coefficient is what defines the amplitude of the vibrations while $F_n(x)$ only defines the shape of the vibrating modes. Now because it is assumed that the eigenvectors form an orthonormal basis, the dot product of any one eigenvector with another is zero. For example, if we take the above equation and dot both sides with the first eigenvector only the first eigenvector

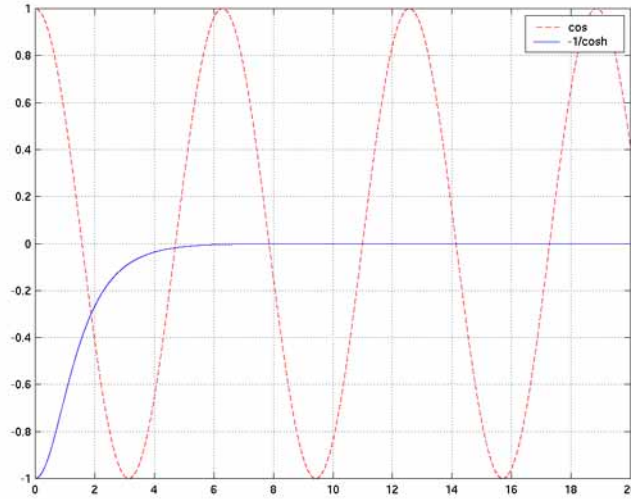


Figure 3-6. Plot of [3-49]. Intersections of the two lines are solutions of the characteristic equation.

and its coefficient will remain. We are now free to assume that the solution to the differential equation is a form of the first eigenvector and solve for the unknown coefficient.

$$a_n = \frac{\int_0^L \frac{M_t(0) \cdot x^2}{2EI} F_n(x) dx}{\int_0^L F_n^2(x) dx} \quad [3-53]$$

where the thermal moment ($M_t(t)$) comes from Johnson and Thorten which is similar to Reitz [66] shown below respectively.

$$M_T(t) = \frac{E\alpha wh^2}{2} \Delta T \quad [3-54]$$

$$M_T(t) = \frac{48EIq_o\alpha}{\pi^4\kappa} \left(\frac{\pi^4}{96} - \sum_{n=1}^{\infty} \frac{e^{-\frac{n^2\pi^2\kappa t}{h^2}}}{n^4} \right) \quad [3-55]$$

The ΔT in Thorten's equation is the temperature difference through the thickness of the solar panel. An equation for ΔT , assuming the solar panel can be modelled as a thin plate, will be derived later in the chapter. The temperature difference equation is dependent on the physical shape of the solar panel, while the thermal moment equation is more general.

A quick numerical example, to calculate the coefficients for the first mode, set $\beta_n L$ to 1.875104 and L to a position on the beam (w).

$$\begin{aligned} \omega_1 &= \frac{(1.875104)^2}{w^2} \sqrt{\frac{EI}{\rho}} & \beta_1 &= \frac{\beta_1 L}{L} = \frac{1.875104}{w} \\ \alpha_1 &= \frac{\sin(1.875104) - \sinh(1.875104)}{\cos(1.875104) + \cosh(1.875104)} \end{aligned} \quad [3-56]$$

Then solve for the coefficient a_n and finally for $F(x)$. Although the derivations show that the summation symbol goes from zero to infinity, in actuality you would only sum over a finite number of modes.

Solving the time function $G(t)$

The solution of the time part of the separation of variables method is a little complicated. The change of the dependent variable is what complicates finding the result. Our solution for the time variable has the form shown below.

$$G_{tt}(t) + \omega^2 G(t) = -M_{Ttt} \quad [3-57]$$

The solution of [3-57] will be in the form shown below, where the term G_p is the particular solution which can be determined if we have an equation for the thermal moment. The general form of the solution is shown below.

$$G(t) = A \sin(\omega_n t) + B \cos(\omega_n t) + G_p \quad [3-58]$$

Summary of results

We have taken a long, winding road in order to solve the equations of motion for the beam. The final solution of the beam, which describes the motion in one direction, will need to be numerically solved is shown below.

$$u(x, t) = v(x, t) + \psi(x) = F(x)G(t) - \frac{M_t(t)x^2}{2EI} \quad [3-59]$$

$$M_T(t) = \frac{E\alpha wh^2}{2} \Delta T \quad [3-60]$$

$$F(x) = \sum a_n [\cosh(\beta_n x) - \cos(\beta_n x) + \alpha_n (\sinh(\beta_n x) - \sin(\beta_n x))] \quad [3-61]$$

$$\omega_n = \frac{(\beta_n L)^2}{L^2} \sqrt{\frac{EI}{\rho}} \quad \beta_n = \frac{\beta_n L}{L} \quad \alpha_n = \frac{\sin(\beta_n L) - \sinh(\beta_n L)}{\cos(\beta_n L) + \cosh(\beta_n L)} \quad [3-62]$$

$$G(t) = A \sin(\omega_n t) + B \cos(\omega_n t) + G_p \quad [3-63]$$

$$a_n = \frac{\int_0^L \frac{M_t(0) \cdot x^2}{2EI} F_n(x) dx}{\int_0^L F_n^2(x) dx} \quad [3-64]$$

Again, summation occurs only over the number of modes of interest.

Thermally Induced Dynamics

Modal Equations

The preceding derivation results in the path of the beam at some point x over time. However for the simulation modal equations are developed which are ordinary differential equations (ODE). This section follows the ideas in McConnell [36] and Thompson [37] and will cover the formulation of the modal equations. Further explanation of the theory pertaining to modal equations can be found in Appendix B or Tongue [71].

Each mode of the system will be a second order equation as shown below.

$$\ddot{q}_n + C_n \dot{q}_n + \omega_n^2 q_n = Q_n \quad [3-65]$$

The subscript n denotes that the equation is for the n^{th} mode. The q 's are the generalized coordinates of the mode and Q is the generalized force applied to that mode. Note that in this formulation, the generalized force for the second mode will only effect the second mode and have no effect on the first mode or any other mode. This idea is due to orthogonality and as in the previous derivation, the modes are assumed to be orthogonal and do not interact with each other. The relationship between our physical parameter of distance x and the generalized coordinate q is as follows:

$$x = \sum_n F_n q_n \quad [3-66]$$

where F_n is our position function¹ for mode n from the previous derivation, note that again it's dependence on x is not shown for simplicity.

$$Q_p = \frac{1}{M_p} \int_0^L P(x) F_n dx \quad [3-67]$$

where $P(x)$ is some external distributed load per unit length. The mass, spring, and dampener matrices are shown below:

$$M_n = \int_0^L \rho A F_n^2 dx \quad [3-68]$$

$$K_n = \int_0^L EI \left[\frac{\partial^2 F_n}{\partial x^2} \right]^2 dx \quad [3-69]$$

$$C_n = \alpha + \beta \omega_n^2 \quad [3-70]$$

The dampening matrix is assumed to be the same magnitude as the mass and spring coefficient matrices. Thus the dampening matrix is just a scaled version of the two where α and β are scalar weighting coefficients that are tuned until the desired amount of dampening is present in the system. The natural frequencies (ω_n) are defined by [3-51] and are shown again below:

$$\omega_n = \frac{(\beta_n L)^2}{L^2} \sqrt{\frac{EI}{\rho}} \quad [3-71]$$

Disturbance Torques

In order to include the solar snap disturbance torques in a simulation, not all of the preceding steps need to be performed. This section will follow along the work of Johnson and Thorten [68]. Instead of solving [3-25], a differential equation is developed which

1. Thompson refers to F_n as a normal mode and uses the variable ϕ_n , see page 438 in [37]

draws on the material already presented. This solution is easier to understand and integrate into a computer simulation.

The equations Johnson and Thorten give for the coupled satellite solar panel system using a generalized form of Lagrangian equations is as follows:

$$I_{sat}\ddot{\theta} + \int_0^L \rho A (R_o + x) \ddot{u}(x, t) dx = 0 \quad [3-72]$$

An expression for the disturbance torque can be obtained by moving terms associated with movement of the solar panel to the right side of the equation. The disturbance torque (g_{ss}) is a composite of two torques. The first torque (g_{qs}) is generated from the quasi-static terms, or the nonhomogeneous boundary conditions. The second torque (g_v) is derived from the vibrational aspects of the beam.

$$g_{ss} = g_{qs} + g_v \quad [3-73]$$

This can be seen by substituting the second derivative of [3-25] into the disturbance torque equation obtained from [3-72].

$$g_{ss} = - \int_0^L \rho A (R_o + x) \ddot{u}(x, t) dx \quad [3-74]$$

$$\ddot{u}(x, t) = \frac{d^2}{dt^2} u(x, t) = \frac{d^2}{dt^2} (v(x, t) + \psi(x, t)) \quad [3-75]$$

$$g_{qs} = - \int_0^L \rho A (R_o + x) \frac{d^2}{dt^2} \psi(x, t) dx = - \frac{\rho A^2 h \alpha_T}{4I} \left(\frac{R_o L^3}{3} + \frac{L^4}{4} \right) \Delta \ddot{T} \quad [3-76]$$

$$g_v = - \int_0^L \rho A (R_o + x) \frac{d^2}{dt^2} v(x, t) dx = \rho A \sum_{n=1}^N \left[\int_0^L (R_o + x) \phi_n(x) dx \right] \ddot{q}_n \quad [3-77]$$

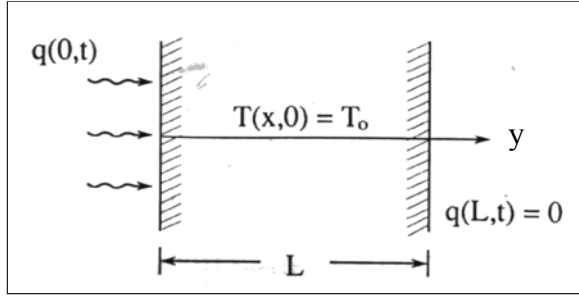


Figure 3-7. A cross section of the beam in which a heat flux is applied to one side. Notice the beam starts off at a uniform temperature.

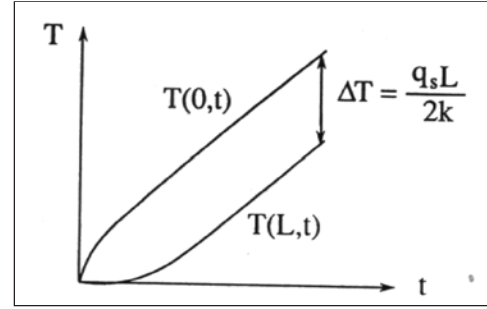


Figure 3-8. Plot of the temperature change of a beam. Notice that the temperature difference between the two sides remains constant eventually.

where ϕ_n is the shape function (which use to be F_n but is changed to match Johnson and Thorten's work) and q_n is the generalized coordinate for mode n , and ΔT is the temperature difference through the thickness (h) of the solar panel.

Thickness Temperature Gradient

The equation for the temperature in a thin plate of thickness h can be found in Thorten's book [26] on pg. 86 or Incropera and DeWitt [69]. A diagram of the solar panel cross section is shown in Figure 3-7, the equation for the temperature change through the thickness is given below.

$$T(y, t) = \frac{q_o h}{k} \left[\frac{\kappa t}{h^2} + \frac{1}{2} \left(\frac{y}{h} + \frac{1}{2} \right)^2 - \frac{1}{6} - \frac{2}{\pi^2} \sum_{n=1}^{\infty} \frac{(-1)^n}{n^2} \cos \left(n\pi \left(\frac{y}{h} + \frac{1}{2} \right) \right) e^{\frac{-n^2 \pi^2 \kappa t}{h^2}} \right] \quad [3-78]$$

The temperature difference through the thickness of the beam can be found by subtracting the temperature at the heated surface from the temperature at the insulated surface at time t . After subtracting the two equations and some algebra, the temperature gradient at time t , steady state at time ∞ , and its acceleration becomes:

$$\Delta T(t) = \frac{q_o h}{k} \left[\frac{1}{2} + \frac{2}{\pi^2} \sum_{n=1,3,5,\dots}^{\infty} \frac{-2}{n^2} e^{\frac{-n^2 \pi^2 \kappa t}{h^2}} \right] \quad [3-79]$$

$$\Delta T(\infty) = \frac{q_o h}{2k} \quad [3-80]$$

$$\Delta \ddot{T}(t) = \frac{d^2 \Delta T}{dt^2} \quad [3-81]$$

Notice that Figure 3-8 and [3-80] show the temperature gradient of the solar panel will eventually reach a steady state. The two sides of the beam never simultaneously achieve the same temperature while there is a heat flux constantly applied. This means there is always a moment on the solar panels, and thus always a deflection of the solar panels. Examples of this are shown in Figure 3-9 and Figure 3-10 where the solar panels of Hubble are deflected due to the constant temperature gradient. Hubble's solar panels are more complex than what is represented in this work. Hubble's solar panels incorporate bi-stems, solar blankets, and spreader bars which are very thin, light weigh structures that are highly susceptible to solar snap. Hubble's solar panels have been replaced several times



Figure 3-9. Hubble Space Telescope which shows the solar panels bent due to a constant temperature gradient.



Figure 3-10. Another picture of the Hubble Space Telescope showing deflection of the solar panels during maintenance by the space shuttle.

over the years, and the current ones are smaller and stiffer than the original design. This is an attempt to reduce the influence of solar snap on Hubble.

Even though there is a constant temperature gradient through the panels, this does not mean there is constant vibrations. The vibrations are driven by a change in the heat flux, and thus a change in the temperature gradient. These changes primarily occur during the day-to-night and night-to-day transitions of the satellite around the Earth.

Conclusion

Thermally induced vibrations or solar snap is a disturbance which occurs when a spacecraft transitions into and out of the Earth's shadow. A temperature gradient occurs in the long thin solar panel which in turn creates a thermal moment. This moment causes the panel to vibrate until the transients die out. During this time frame, the scientific mission has to be temporarily halted and the spacecraft put into a stable orientation.

The theory and math behind this phenomenon is not a mystery and the aim of this chapter was to provide the reader with a complete derivation of this problem for a simple solar panel. Here the assumption was made that the temperature gradient only occurs through the thickness of the beam which reduced the problem to one dimension.

CHAPTER 4

SATELLITE ATTITUDE DYNAMICS

Satellite attitude control can be difficult due to the various types of disturbances and limited control capabilities of satellites. In order to develop a controller, a good understanding of the dynamics of the system is needed for proper simulations to be conducted. This chapter will give an overview of the dynamics of rigid bodies in rotating reference frames.

Reference Frames

Several different coordinate systems will be used to develop the equations of motion for a satellite. This section will cover the reference frames used in this work and are shown in Figure 4-1.

Earth Centered Inertial

The first reference frame is the Earth centered inertial (ECI) frame. This frame is a non-rotating frame relative to the fixed stars¹. Another way to say this is, the ECI frame is an inertially fixed frame where Newton's laws are valid. This frame has its origin located at the center of the Earth with its z-axis along the Earth's mean axis of rotation. The y-axis and x-axis are pointing to some convenient set of stars. Note that the Earth is allowed to rotate in this fixed frame.

1. Actually the stars move, and thus our ECI frame moves with them. This movement is small enough to be ignored in most cases.

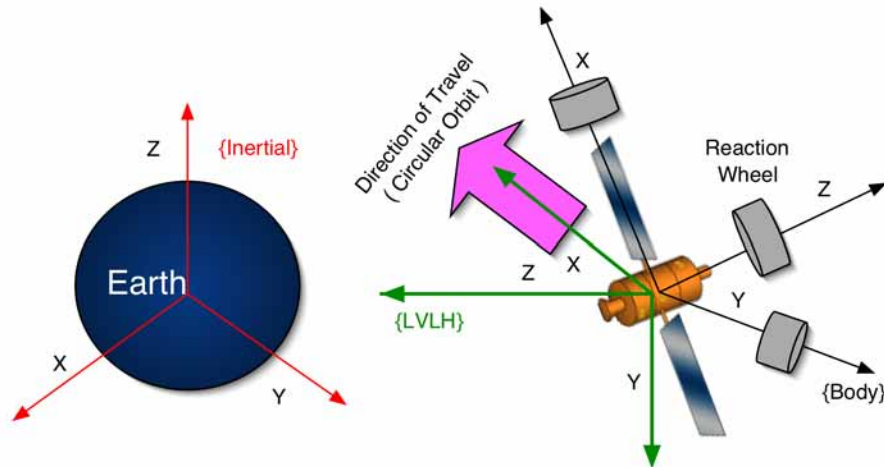


Figure 4-1. The three frames of reference commonly used in spacecraft dynamics.

Orbital Frame

The orbital reference frame or local vertical, local horizontal (LVLH) frame is the reference frame from which all of our spacecraft's angle will be defined. The orbital frame's origin is located at the center of mass of the spacecraft. The z-axis points towards the center of the Earth. The y-axis is perpendicular to the orbital plane. The x-axis points in the general direction of travel. When the orbit is circular, then the velocity vector and the x-axis are co-linear.

Body Fixed Frame

The body reference frame is located with its origin located at the center of mass of the spacecraft. The x, y, and z axis are located along the x, y, and z principle axis of the spacecraft.

Fun With Vectors and Rotating Coordinate Systems

This section will present some simple vector identities and properties which maybe useful to the reader. Although all of the important steps are shown in the following

derivations for spacecraft dynamics, at times some algebraic simplifications are not shown.

$\mathbf{a} \cdot (\mathbf{b} + \mathbf{c}) = \mathbf{a} \cdot \mathbf{b} + \mathbf{a} \cdot \mathbf{c}$	$\mathbf{a} \times \mathbf{b} = -\mathbf{b} \times \mathbf{a}$
$(\mathbf{a} \times \mathbf{b}) \cdot \mathbf{c} = \mathbf{a} \cdot (\mathbf{b} \times \mathbf{c})$	$\mathbf{a} \cdot \mathbf{b} = \mathbf{b} \cdot \mathbf{a}$
$\mathbf{a} \times (\mathbf{b} \times \mathbf{c}) = \mathbf{b}(\mathbf{a} \cdot \mathbf{c}) - \mathbf{c}(\mathbf{a} \cdot \mathbf{b})$	$\mathbf{a} \cdot \mathbf{b} = \mathbf{a}^T \mathbf{b}$

Conventions used:

- All vectors and matrices are bold face.
- \mathbf{I} and $\mathbf{0}$ represent the identity matrix and the zero matrix respectively.

The dynamics for a spacecraft are written with respect to several frames of reference. Two important reference frames will be the ECI inertially fixed frame and a body fixed frame which is allowed to rotate in space. Differentiating a vector relative to a rotating reference frame is given by:

$$\frac{\partial \mathbf{A}_I}{\partial t} = \frac{d\mathbf{A}_B}{dt} + \boldsymbol{\omega} \times \mathbf{A}_B \quad [4-1]$$

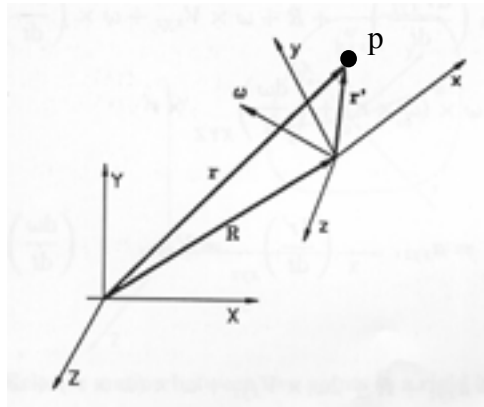


Figure 4-2. A reference frame in motion relative to an inertially fixed frame of reference.

where ω is the relative angular rotation rate between the inertially fixed frame I and the rotating body frame B .

Now for the point p located in the rotating system at position r , its velocity is:

$$\mathbf{v}_I = \frac{d\mathbf{r}_I}{dt} = \frac{d\mathbf{R}_I}{dt} + \frac{d\mathbf{r}'_I}{dt} = \dot{\mathbf{R}} + \frac{d\mathbf{r}'_B}{dt} + \boldsymbol{\omega} \times \mathbf{r}'_B = \dot{\mathbf{R}} + \mathbf{v}_B + \boldsymbol{\omega} \times \mathbf{r}'_B \quad [4-2]$$

$$\mathbf{r} = \mathbf{R} + \mathbf{r}' \quad [4-3]$$

where $\dot{\mathbf{R}}$ is the velocity of the origin of the rotating system relative to the fixed system and \mathbf{v}_B is the velocity of the point in the rotating system. A similar derivation can be shown for the acceleration of the point which results in:

$$\mathbf{a}_I = \mathbf{a}_B + \ddot{\mathbf{R}} + 2\boldsymbol{\omega} \times \mathbf{v}_B + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}') + \dot{\boldsymbol{\omega}} \times \mathbf{r}' \quad [4-4]$$

When dealing with rigid bodies, with the rotating frame representing a body fixed frame, the point p does not move within this frame. This is due to the fact that p is fixed in the body. Thus $\mathbf{v}_B = \mathbf{a}_B = 0$, and the proceeding equations reduce to:

$$\mathbf{v}_I = \dot{\mathbf{R}} + \boldsymbol{\omega} \times \mathbf{r}' \quad \mathbf{a}_I = \ddot{\mathbf{R}} + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}') + \dot{\boldsymbol{\omega}} \times \mathbf{r}' \quad [4-5]$$

where the velocity and acceleration are functions of the rotation rate of the rotating frame and its velocity and acceleration.

Spacecraft Attitude

Attitude refers to the orientation a spacecraft occupies in 3D space. Although there are several different ways to represent this (i.e., euler angles, Gibbs vector, fixed angles, Euler symmetric parameters, etc.), typically space applications utilize quaternions. This section does not attempt to provide the extensive understanding needed to employ quaternions but

rather a simple introduction. Further information can be found in Wertz [37] or Crane and Duffy [42], or Appendix A.

Quaternions

Quaternions were invented by William Rowan Hamilton in 1843. Prior to his discovery, it was believed impossible that any algebra could violate the laws of commutativity for multiplication. His work introduced the idea of hyper-complex numbers. Here real numbers can be thought of as hyper-complex numbers with a rank of 1, ordinary complex numbers with a rank of 2, and quaternions with a rank of 4. Hamilton's crucial rule that made this possible:

$$i^2 = j^2 = k^2 = ijk = -1 \quad [4-6]$$

Hamilton supposedly developed this rule while on his way to a party. When he realized what the solution was, he took out his pocket knife and carved the answer into a wooden bridge. This rule would forever change mathematics as was known at the time. Now mathematicians could look at algebra where commutativity did not work. This is where Gibbs and others developed algebra of vector spaces, and quickly eclipsed Hamilton's work until recently.

Quaternions, also known as Euler symmetric parameters, are more mathematically efficient ways to compute rotations of rigid and non-rigid body systems than traditional methods involving standard rotational matrices or Euler angles. Quaternions have the advantage of few trigonometric functions needed to compute attitude. Also, there exists a product rule for successive rotations that greatly simplifies the math, thus reducing processor computation time. Quaternions also hold the advantage of being able to interpolate between two quaternions (through a technique called spherical linear

interpolation or slerp) without the danger of singularities, maintaining a constant velocity, and minimum distance travelled between points.

The major disadvantage of quaternions is the lack of intuitive physical meaning. Most people would understand where a point was in cartesian space if they were given [1 2 3]. However, few would comprehend where a point was if given the quaternion [1 2 3 4].

Rotations of Rigid Bodies in Space.

Quaternions are able to represent a rotation of a rigid body in space. To perform a rotation (ϕ) of a rigid body about an arbitrary moving/fixed axis (\mathbf{e}) in space, the quaternion representation of this operation is

$$\mathbf{q} = \begin{bmatrix} \hat{\mathbf{q}} & q_4 \end{bmatrix}^T \quad \text{Norm}(\mathbf{q}) = 1 \quad [4-7]$$

$$\hat{\mathbf{q}} = \mathbf{e} \cdot \sin\left(\frac{\phi}{2}\right) \quad q_4 = \cos\left(\frac{\phi}{2}\right) \quad [4-8]$$

$$\mathbf{e} = \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix}^T \quad \text{Norm}(\mathbf{e}) = 1 \quad [4-9]$$

Notice that only one sine and one cosine function call is needed to calculate a quaternion, where an euler rotation matrix would require three sine and three cosine function calls, one each for roll, pitch, and yaw. Since trigonometric function calls are computationally expensive, this is a great savings.

Spacecraft Equations of Motion

In this section, the kinematic and dynamic equations of motion for a spacecraft are presented.

Attitude Kinematics

A spacecraft's orientation in space is represented by a quaternion. The quaternion kinematic equations of motion are given by Wertz [38] as,

$$\dot{\mathbf{q}} = \frac{1}{2}\mathbf{\Omega}\mathbf{q} = \frac{1}{2}\Xi(\mathbf{q})\boldsymbol{\omega} \quad [4-10]$$

where

$$\mathbf{\Omega} = \begin{bmatrix} -\boldsymbol{\omega} \times & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^T & 0 \end{bmatrix} \text{ and } \Xi(\mathbf{q}) = \begin{bmatrix} q_4 \mathbf{I}_{3 \times 3} + \hat{\mathbf{q}} \times \\ -\hat{\mathbf{q}}^T \end{bmatrix} \quad [4-11]$$

The terms with the \times by them signify a skew matrix.

Spacecraft Dynamics

First the equations of motion will be developed for a satellite and then expanded to include other dynamics. Thus a lengthy derivation is needed to derive something as simple as Euler's equation. This, however, is necessary so that when reaction wheels and fuel slosh are included, understanding how these dynamics are incorporated will be easier.

Simple Spacecraft

To model a satellite, we starting off with a couple of definitions for the total mass of the spacecraft, first moment of inertia, and the second moment of inertia respectively.

$$m = \sum_{n=1}^N m_n \quad \mathbf{c} = \sum_{n=1}^N m_n \mathbf{r}_n \quad \mathbf{J} = \sum_{n=1}^N m_n (\mathbf{r}_n^T \mathbf{r}_n \mathbf{I} - \mathbf{r}_n \mathbf{r}_n^T) \quad [4-12]$$

where m_n is a point mass in the spacecraft, N is the total number of points. These definitions are in reference to Figure 4-3 (Adapted from Hughes [31] p. 43). The second moment of inertia (\mathbf{J}) is typically just referred to as the moment of inertia. The moment of inertia has two important properties: symmetry and positive definite.

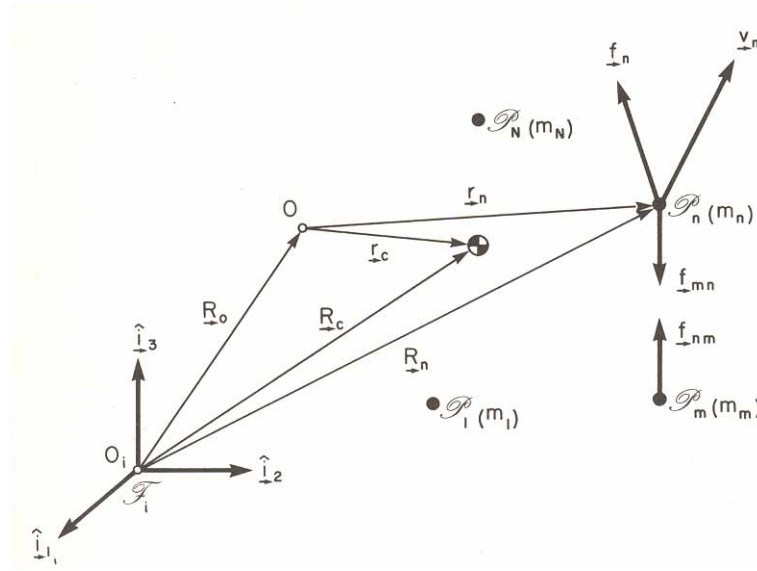


Figure 4-3. A rigid body satellite.

The momentum and velocity of each point in the spacecraft is

$$m_n \dot{\mathbf{v}}_n = \mathbf{f}_n + \sum_{m=1}^N \mathbf{f}_{mn} \quad \mathbf{v}_n = \mathbf{v}_o + \dot{\mathbf{r}}_n \quad [4-13]$$

where $\dot{\mathbf{R}} = \mathbf{v}_o$, \mathbf{f}_n is the external forces acting on point p , and \mathbf{f}_{mn} is the force exerted by point m on point n .

The linear momentum (\mathbf{p}) of the total spacecraft is

$$\mathbf{p} = \sum_{n=1}^N \mathbf{p}_n = \sum_{n=1}^N m_n \mathbf{v}_n = \sum_{n=1}^N m_n (\mathbf{v}_o + \dot{\mathbf{r}}_n) = m \mathbf{v}_o + \dot{\mathbf{c}} \quad [4-14]$$

This equation could be further simplified by realizing that a spacecraft¹ is a rigid body, and thus $\dot{\mathbf{c}} = \mathbf{0}$. Thus

$$\dot{\mathbf{p}} = \mathbf{f} = \sum_{n=1}^N \mathbf{f}_n \quad [4-15]$$

1. Actually since spacecraft are composed of light weight, flexible materials, some authors model them as flexible structures.

where

$$\sum_{n=1}^N \sum_{m=1}^N \mathbf{f}_{mn} = \mathbf{0} \quad [4-16]$$

which comes from Newton's third law, $\mathbf{f}_{mn} + \mathbf{f}_{nm} = \mathbf{0}$.

The angular momentum of the spacecraft is:

$$\mathbf{h}_o = \sum_{n=1}^N \mathbf{r}_n \times \mathbf{p}_n \quad [4-17]$$

Now differentiating this equation with respect to time, results in the following equation.

$$\dot{\mathbf{h}}_o = \sum_{n=1}^N (\dot{\mathbf{r}}_n \times \mathbf{p}_n + \mathbf{r}_n \times \dot{\mathbf{p}}_n) = \sum_{n=1}^N [(\mathbf{v}_n - \mathbf{v}_o) \times m_n \mathbf{v}_n + \mathbf{r}_n \times \mathbf{f}_n] \quad [4-18]$$

Realizing that $\mathbf{v}_n \times m_n \mathbf{v}_n = \mathbf{0}$ results in the following equation.

$$\dot{\mathbf{h}}_o = -\mathbf{v}_o \times \mathbf{p} + \mathbf{g}_o \quad [4-19]$$

where \mathbf{g}_o is the total external torque about the point O.

Now if the equations are defined about the center of mass of a rigid body spacecraft, then $\mathbf{c} = \mathbf{0}$, $\mathbf{v}_o = \mathbf{v}_n$, and the cross product with momentum in [4-19] disappears. The equations collapse to the more familiar ones for linear and angular momentum.

$$\mathbf{p} = m\mathbf{v}_o \quad \dot{\mathbf{h}}_o = \mathbf{g}_o \quad [4-20]$$

Now using the previous equations involving rotating reference frames, we can write the linear and angular momentum equations for a spacecraft in a rotating body fixed frame.

$$\mathbf{p} = m\mathbf{v}_o + \dot{\mathbf{c}} + \boldsymbol{\omega} \times \mathbf{c} \quad [4-21]$$

$$\dot{\mathbf{p}} = -\boldsymbol{\omega} \times \mathbf{p} + \mathbf{f} \quad [4-22]$$

$$\dot{\mathbf{h}}_o = -\boldsymbol{\omega} \times \mathbf{h}_o - \mathbf{v}_o \times \mathbf{p} + \mathbf{g}_o \quad [4-23]$$

As before, these equations can be simplified using certain assumptions. However, deriving the equations without the assumptions makes including internal dynamics (i.e., fuel slosh and reaction/momentum wheels) easier.

These equations put into a matrix form are as follows:

$$\boldsymbol{\rho} = \begin{bmatrix} \mathbf{p} & \mathbf{h}_o \end{bmatrix}^T \quad \mathbf{V} = \begin{bmatrix} \mathbf{v} & \boldsymbol{\omega} \end{bmatrix}^T \quad \mathbf{F} = \begin{bmatrix} \mathbf{f} & \mathbf{g}_o \end{bmatrix}^T \quad [4-24]$$

$$\boldsymbol{\rho} = \mathbf{M}\mathbf{V} \quad \mathbf{M} = \begin{bmatrix} m\mathbf{I} & -\mathbf{c} \times \\ \mathbf{c} \times & \mathbf{J} \end{bmatrix} \quad [4-25]$$

where \mathbf{M} is the mass matrix of the system. The kinetic energy T of the system is given by:

$$T = \frac{1}{2} \mathbf{V}^T \mathbf{M} \mathbf{V} \quad [4-26]$$

$$T = \frac{1}{2} m \mathbf{v}_o^T \mathbf{v}_o + \boldsymbol{\omega}^T \mathbf{c} \times \mathbf{v}_o + \frac{1}{2} \boldsymbol{\omega}^T \mathbf{J} \boldsymbol{\omega} \quad [4-27]$$

The dynamics can be obtained from the kinetic energy by using the following:

$$\frac{\partial T}{\partial \mathbf{v}} = \mathbf{p} \quad \frac{\partial T}{\partial \boldsymbol{\omega}} = \mathbf{h}_o \quad [4-28]$$

$$\frac{d}{dt} \left(\frac{\partial T}{\partial \mathbf{v}_o} \right) + \boldsymbol{\omega} \times \left(\frac{\partial T}{\partial \mathbf{v}_o} \right) = \mathbf{f} \quad \frac{d}{dt} \left(\frac{\partial T}{\partial \boldsymbol{\omega}} \right) + \boldsymbol{\omega} \times \left(\frac{\partial T}{\partial \boldsymbol{\omega}} \right) + \mathbf{v}_o \times \left(\frac{\partial T}{\partial \mathbf{v}_o} \right) = \mathbf{g}_o \quad [4-29]$$

These equations are called the quasi-Lagrangian equations and are shown in Hughes [31] and Meirovitch [39]. Meirovitch derives the equations using quasi-coordinates, which is somewhat difficult to understand and very long. However, this method has been used by some authors such as Miller et al. [40] to model various flexible appendages on a satellite and moving submasses inside of a satellite.

Angular Velocity

The spacecraft's orbit in this work is assumed to be circular. Since we are using an LVLH reference frame for our spacecraft, the angular velocity of the system must account for the orbital rate in addition to the maneuvers that the satellite is conducting.

$$\boldsymbol{\omega}^{B/I} = \boldsymbol{\omega}^{B/O} + \boldsymbol{\omega}^{O/I} \quad [4-30]$$

$$\boldsymbol{\omega}^{O/I} = -\mathbf{C}\mathbf{n} \quad \mathbf{n} = \begin{bmatrix} 0 & \sqrt{\frac{\mu}{R_c^3}} & 0 \end{bmatrix}^T \quad [4-31]$$

where B refers to the body frame, O refers to the orbital frame, I refers to the fixed inertial frame, $\boldsymbol{\omega}^{B/O}$ refers to angular velocity of B relative to O , C is a rotation matrix, μ is the gravitational parameter of the Earth, and R_c is the distance the spacecraft is from the center of the Earth.

Internal Disturbances

Various factors complicate the dynamics of a satellite's dynamics. Moving masses such as reaction wheels and sloshing fuel create additional linear and angular momentum that needs to be accounted for. This section will show how the above equations for a simple spacecraft need to be modified in order to account for these effects.

Spacecraft with Reaction or Momentum Wheels

Starting off with reaction wheels, we will introduce a method to account for any number of moving wheels in any configuration. However, since this work only deals with reaction wheels, we will not concern ourselves with momentum wheels which change their orientation and not their rate of spin.

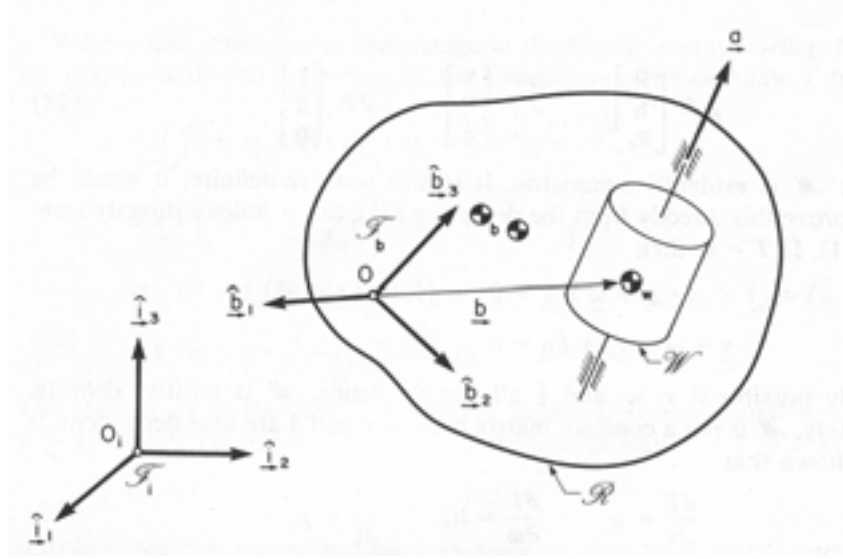


Figure 4-4. A rigid body spacecraft with a single reaction wheel.

Now we will add the contribution of the reaction or momentum wheels to the spacecraft dynamics. Looking at Figure 4-4 (adapted from Hughes [31] p. 66), we can define the following terms:

$$m = m_{sat} + m_w \quad \mathbf{c} = \mathbf{c}_{sat} + m_w \mathbf{b} \quad \mathbf{J} = \mathbf{J}_{sat} + I_w \mathbf{a} + m_w (\mathbf{b}^T \mathbf{b} \mathbf{I} - \mathbf{b} \mathbf{b}^T) \quad [4-32]$$

where terms with the subscript *sat* refer to the satellite and terms with the subscript *w* refer to the wheels. The linear momentum is given by the following equations:

$$\mathbf{p}_{sat} = m_{sat} \mathbf{v} + \boldsymbol{\omega} \times \mathbf{c}_{sat} \quad [4-33]$$

$$\mathbf{p}_w = m_w \mathbf{v} + m_w \boldsymbol{\omega} \times \mathbf{b} \quad [4-34]$$

$$\mathbf{p} = \mathbf{p}_{sat} + \mathbf{p}_w = m \mathbf{v} + \boldsymbol{\omega} \times \mathbf{c} \quad [4-35]$$

where \mathbf{b} is the vector from the point O to the wheel. The angular equations of motion are given by the following equations:

$$\mathbf{h}_{sat} = \mathbf{J}_{sat} \boldsymbol{\omega} + \mathbf{c}_{sat} \times \mathbf{v} \quad [4-36]$$

$$\mathbf{h}_w = I_s \boldsymbol{\omega}_w \quad [4-37]$$

$$\mathbf{h} = \mathbf{h}_{sat} + \mathbf{h}_a \quad [4-38]$$

$$\mathbf{h} = \mathbf{c} \times \mathbf{v} + \mathbf{J}\boldsymbol{\omega} + \mathbf{a}\mathbf{h}_w \quad [4-39]$$

where \mathbf{a} is a vector that describes the direction of the wheel axis when dealing with a single wheel. When there are multiple wheels, \mathbf{a} becomes a matrix where the columns are the individual direction vectors for each wheel. For example, a system with n wheels would look like:

$$\mathbf{a} = \left[\{\mathbf{a}_{w1}\} \ \{\mathbf{a}_{w2}\} \ \dots \ \{\mathbf{a}_{wn}\} \right] \quad [4-40]$$

where

$$\mathbf{a}_{wi} = \left[e_1 \ e_2 \ e_3 \right]^T \quad Norm(\mathbf{a}_{wi}) = 1 \quad [4-41]$$

The wheel speed $\boldsymbol{\omega}_w$ is a vector where each element is the speed of a specific wheel. Thus for a system with n wheel, the vector would be:

$$\boldsymbol{\omega}_w = \left[\omega_{w1} \ \omega_{w2} \ \dots \ \omega_{wn} \right]^T \quad [4-42]$$

The equations of motion can be derived from kinetic energy for the system by using the quasi-Lagrangian. The kinetic energy for the spacecraft with wheel dynamics is:

$$T = \frac{1}{2} m \mathbf{v}^T \mathbf{v} + \frac{1}{2} \boldsymbol{\omega}^T \mathbf{J} \boldsymbol{\omega} + \frac{1}{2} I_w \boldsymbol{\omega}_w^T \boldsymbol{\omega}_w - \mathbf{v}^T \mathbf{c} \times \boldsymbol{\omega} + I_w \boldsymbol{\omega}_w^T \mathbf{a}^T \boldsymbol{\omega} \quad [4-43]$$

Putting these equations into a matrix form results in the following equations.

$$\boldsymbol{\phi} = \left[\mathbf{p} \ \mathbf{h}_o \ \mathbf{h}_a \right]^T \quad \mathbf{V} = \left[\mathbf{v}_o \ \boldsymbol{\omega} \ \boldsymbol{\omega}_w \right]^T \quad \mathbf{F} = \left[\mathbf{f} \ \mathbf{g} \ \mathbf{g}_a \right]^T \quad [4-44]$$

$$\mathcal{P} = MV \quad M = \begin{bmatrix} mI & -c \times & \mathbf{0} \\ c \times & J & I_w \mathbf{a} \\ \mathbf{0}^T & I_w \mathbf{a}^T & I_w \mathbf{a}^T \end{bmatrix} \quad [4-45]$$

The equations of motion, like before, are as follows:

$$\dot{\mathbf{p}} = -\boldsymbol{\omega} \times \mathbf{p} + \mathbf{f} \quad [4-46]$$

$$\dot{\mathbf{h}}_o = -\boldsymbol{\omega} \times \mathbf{h}_o - \mathbf{v}_o \times \mathbf{p} + \mathbf{g}_o \quad [4-47]$$

$$\dot{\mathbf{h}}_a = \mathbf{g}_a \quad [4-48]$$

Although these equations look the same as the previous equations without wheel dynamics, remember the equations for linear and angular momentum are different. The linear and angular momentum of the spacecraft and wheels are now coupled in these equations.

Equations for spacecraft with wheel dynamics are often simpler than what are shown here. However, if the wheels are assumed to be located at the center of mass (which is physically impossible) then the equations are greatly simplified,

$$\dot{\mathbf{h}}_o = -\boldsymbol{\omega} \times \mathbf{h}_o + \mathbf{g}_o \quad [4-49]$$

since $\mathbf{p} = m\mathbf{v}_o$ and $\mathbf{v}_o \times \mathbf{p} = \mathbf{v}_o \times m\mathbf{v}_o = \mathbf{0}$. Further more, $\mathbf{h} = J\boldsymbol{\omega} + \mathbf{h}_w$ since $\mathbf{c} = \mathbf{0}$ and substituting this into [4-49] results in:

$$J\dot{\boldsymbol{\omega}} + \dot{\mathbf{h}}_w = -\boldsymbol{\omega} \times (J\boldsymbol{\omega} + \mathbf{h}_w) + \mathbf{g}_o \quad [4-50]$$

$$J\dot{\boldsymbol{\omega}} = -\boldsymbol{\omega} \times J\boldsymbol{\omega} + \boldsymbol{\omega} \times \mathbf{h}_w - \dot{\mathbf{h}}_w + \mathbf{g}_o \quad [4-51]$$

$$J\dot{\boldsymbol{\omega}} = -\boldsymbol{\omega} \times J\boldsymbol{\omega} + \boldsymbol{\omega} \times \mathbf{h}_w - \mathbf{g}_a + \mathbf{g}_o \quad [4-52]$$

This final equation is the common equation found in Wie [22] and Sidi [9].

Fuel Slosh

Fuel slosh, as stated in a previous chapter, is the unwanted movement of fuel inside a spacecraft. Each sloshing mode is modelled as a mass, with the forces exerted on the spacecraft body represented by a spring, and the effects of baffling in the fuel tank

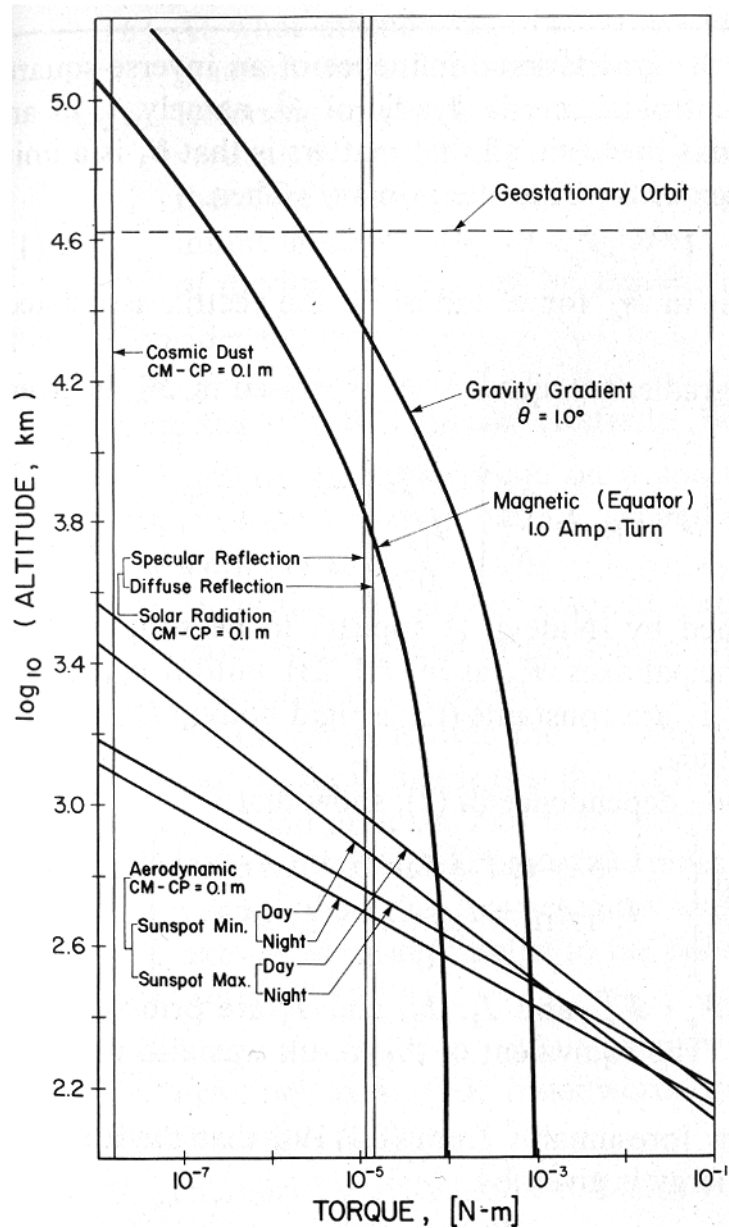


Figure 4-5. A plot of common environmental disturbances a spacecraft is subjected to.

represented by a dampener. The dynamics of this are incorporated into our simple satellite model by the following equations which will represent one sloshing mode.

$$\mathbf{p} = m\mathbf{v} - \mathbf{c} \times \boldsymbol{\omega} + m_f \dot{\mathbf{n}} \quad [4-53]$$

$$\mathbf{h} = \mathbf{c} \times \boldsymbol{\omega} + \mathbf{J}\boldsymbol{\omega} + m_f \mathbf{b} \times \dot{\mathbf{n}} \quad [4-54]$$

$$\mathbf{p}_f = m_f(\mathbf{v} - \mathbf{b} \times \boldsymbol{\omega} + \dot{\mathbf{n}}) \quad [4-55]$$

where \mathbf{n} is the position of the sloshing mode, m_f is the mass of the fuel, and \mathbf{b} is the vector from the center of mass of the spacecraft to the center of the fuel tank. These equations reflect how the linear, angular, and fuel slosh momentums change with the addition of fuel slosh's mass-spring-damper. The mass matrix now becomes:

$$\mathbf{M} = \begin{bmatrix} m\mathbf{I} & -\mathbf{c} \times & m_f \\ \mathbf{c} \times & \mathbf{J} & m_f \mathbf{b} \times \\ m_f & -m_f \mathbf{b} \times & m_f \end{bmatrix} \quad [4-56]$$

As with the wheel dynamics, the differential equations of motion for the linear and angular momentum remain the same. The new differential equation for the fuel slosh is:

$$\dot{\mathbf{p}}_f = -m_f \boldsymbol{\omega} \times (\mathbf{v} - \mathbf{r}_d \times \boldsymbol{\omega}) - c_f \dot{\mathbf{n}} - k_f \mathbf{n} \quad [4-57]$$

where the c_f and k_f are the dampening coefficient and spring constant for the sloshing mode.

Environmental Disturbances in Space

There are many environmental disturbances that plague a spacecraft in orbit. These disturbances constantly effect the performance of the spacecraft which necessitates the use of a control system that counter acts them. The disturbance torques produced by some of

these are shown in Figure 4-5 (adapted from Hughes [31] p. 271). A further explanation of the important disturbances follows.

Gravity Gradient

Euler and d'Alembert in 1749 first pointed out how the Earth's gravitational field was not uniform. Later in 1780, Lagrange used these ideas to explain why the moon always had the same side facing the Earth. Since the gravitational field over a rigid body is not uniform, the center of mass is not the center of gravity. Thus there are torques about the center of mass which depend on the orientation of the spacecraft.

The gravity gradient torque disturbance is found in [22], [31], and [38] as

$$\mathbf{g}_{gg} = 3\omega_e^2 \hat{\mathbf{o}}_3 \times \mathbf{I} \hat{\mathbf{o}}_3 \quad [4-58]$$

where ω_e is the orbital rate of the spacecraft, $\hat{\mathbf{o}}_3$ is the third column in the orbital frame to body frame rotation matrix. When a satellite is in low Earth orbit, gravity gradient torques become the predominate environmental disturbance.

Solar Snap

In a previous chapter, modelling solar snap was discussed. The disturbance torque due to solar snap is a combination of the quasi-static displacement of the solar panel and its vibration.

$$\mathbf{g}_{ss} = \mathbf{g}_{qs} + \mathbf{g}_v \quad [4-59]$$

Conclusion

This chapter showed the basic concepts of spacecraft and disturbances dynamics. These dynamics produce a system where the satellite body, wheels, and fuel are all coupled. The disturbances presented in this chapter were internal (fuel slosh) and external

(gravity gradient). A complete system can now be produced which has all of the dynamics of interest: satellite, wheel, fuel slosh, and thermally induced vibrations.

CHAPTER 5 ATTITUDE CONTROL OF SPACECRAFT

This chapter will introduce the reader to attitude control of a satellite and specifically to a variable structure method called sliding mode control. First however we will cover some of the typical hardware used in a satellite to control its attitude. Next the traditional proportional derivative controller and sliding mode controllers will be introduced. Later in this chapter we will compare and contrast the two controllers.

Satellite Control Hardware

This section will provide a brief overview of hardware used by a satellite to control attitude.

Control Computational Hardware

Satellites are typically equipped with only the bare minimum in computational ability in an effort to reduce the cost of the vehicle. Unfortunately the complexity and performance of the control system is constrained only to the simpler algorithms. In fact some spacecraft, such as the lunar explorer, had no on board control systems. Instead it was remotely controlled from Earth, which was a very brittle solution at best. This was hailed as a great step forward in reducing the cost of space flight, but what happens if you lose the connect between ground control and the satellite?

Any control algorithm used must be computationally inexpensive so it does not overburden the processor, and is typically designed to operate at 1 Hz. One of the simplest, easiest, and most used controller available is the venerable proportional

derivative (PD) controller. This is one reason that the PD has flourished in spacecraft control.

Control Actuators

In order to overcome environmental disturbances, satellites are typically equipped with various types of actuators that can provide an attitude control effort. Some of these are active control systems which utilized electrical power or consumable resources to effect the spacecraft's attitude. While others are passive, and rely on conservation of energy to effect the satellite's motion. Passive methods of control are generally relegated to dampening out disturbances during flight. These methods do not produce the same levels of torque as their active counterparts, but are typically employed along with active controls in an effort to reduce the power consumption of the spacecraft.

Some of the most common types of active control actuators will now be discussed. A quick comparison of their control torques is given below.

Table 5-1: Comparison of common active attitude control hardware.

Control Hardware	Control Force Range (Nm)
Magnetic Torque Rods	10E-6 (geostationary) 2.5E-3 (400 km)
Reaction Wheels	0.05-2
Thrusters	0.1-30

Reaction and momentum wheels

Reaction and momentum wheels are cylindrical masses that rotate about their center of mass driven by a motor. They are produced in a variety of types and sizes, some of which are shown in Figure 5-1. These devices are capable of changing or stabilizing the orientation of a spacecraft with respect to its axis of rotation through conservation of angular momentum. These are the best for delivering smooth continuous control efforts.

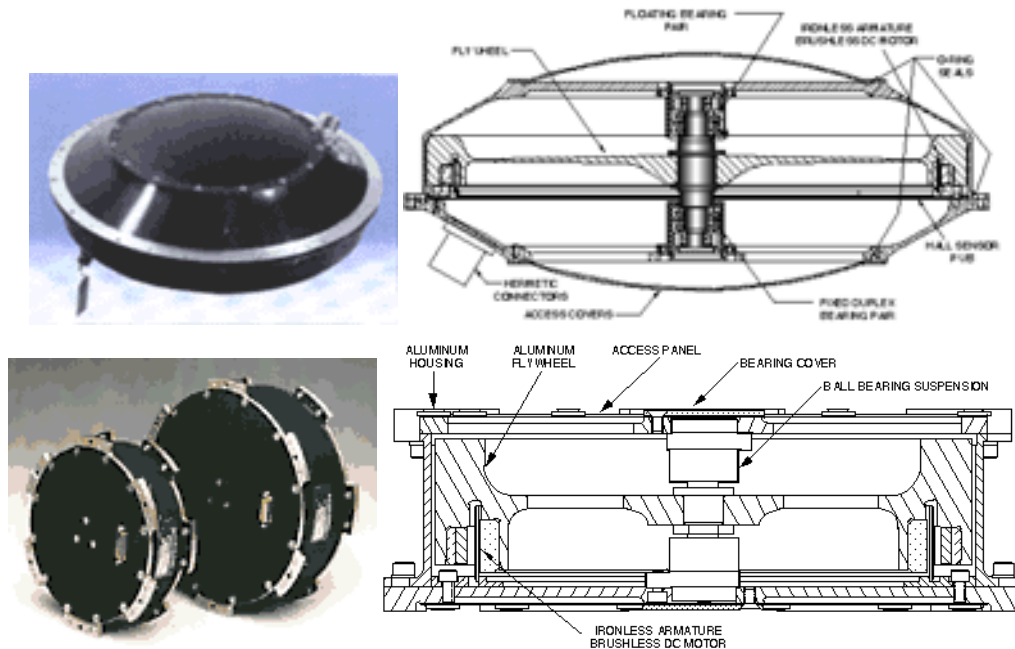


Figure 5-1. Two types of reaction wheels produced by Ithaco.

Wheels that have a nominal spin rate of zero¹ are called reaction wheels. Wheels that have a constant momentum are called momentum wheels.

Moment wheels are used to create a bias in the system which resists small parasitic environmental torques. Sometimes these wheels are also mounted in a gimbaled system that has one or two degrees of freedom. This allows the momentum wheels (with their constant momentum) to change the direction of their momentum vector.

Magnetic torque rods

Magnetic torque rods are another method of attitude control and are shown below in Figure 5-2. They are based on the idea of generating torque by magnetic fields. Here the rods produce a magnetic field which interacts with the magnetic field of the Earth. Obviously this method of attitude control produces small control forces compared to

1. This statement is not always true. Sometimes wheels are biased by a small amount to fight friction in the wheels and bearings. These zero-crossing problems, dead zone around zero rpm, can effect the accuracy of a satellite when it is gathering scientific data.

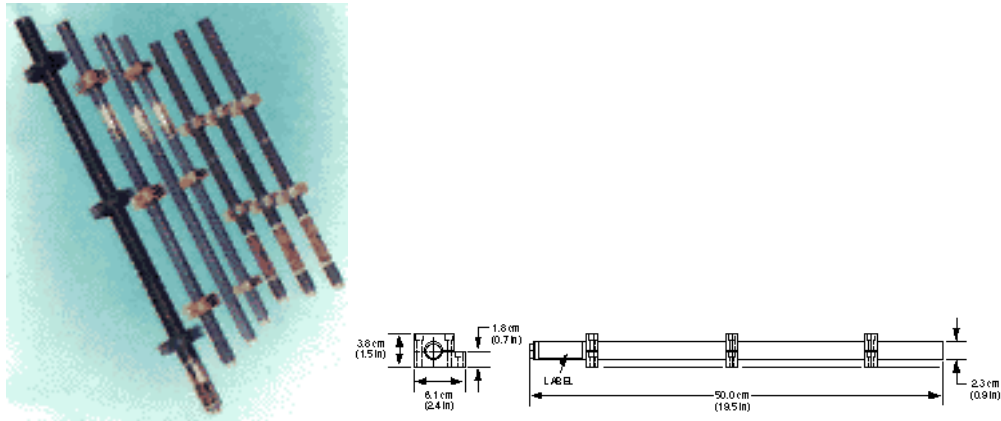


Figure 5-2. Torque rods produced by Ithaco.

reaction wheels. Also the maximum amount of control effort produced by this method is dependant on the altitude of the spacecraft since the Earth's magnetic field is stronger the closer the spacecraft is to the center of the Earth.

In addition to small torque capabilities, electromagnetic shielding to protect satellite components must be considered. This has the additional disadvantage of adding weight and increasing the complexity of the design process.

Gas jets

Gas jets or control jets (shown in Figure 5-3) are the most powerful hardware available for attitude control, but these are not the most desirable to use for two reasons. First they consume fuel which is a limited, non-renewable resource, and second they do not have variable control but rather "on-off". Thus bang-bang control schemes are created to provide pulse-width modulation control efforts to mimic a variable type of device.

Proportional Derivative Control

First let's start off with one of the simplest to design and computationally efficient to implement, the proportional derivative (PD) controller. Wie and Barba [41] developed several computationally efficient control schemes for large angle maneuvers. Many of



Figure 5-3. Spacecraft engine produced by Boeing.

these stabilizing schemes utilize quaternion and angular velocity feedback. The use of quaternion representation allows for more realistic, large angle maneuver control schemes. These schemes are formulated based on Lyapunov analysis, which produces a range of positive stable gains for that control law. Thus in order to meet desired performance, engineers must iterate through a significant number of gain combinations to obtain the desired response in a clean simulation. However, even when a satisfactory response is finally obtained, there are no guarantees how the satellite will behave in the presence of disturbances, noise, or uncertainties.

Bong Wie [22] defines three different PD controllers utilizing quaternions in his book where the origin of the quaternion system is $(0,0,0,1)$.

$$\mathbf{u} = -\mathbf{K}\hat{\mathbf{q}} - \mathbf{C}\boldsymbol{\omega} \quad [5-1]$$

$$\mathbf{K} = k\mathbf{I} \quad \mathbf{C} = \text{diag}(c_1, c_2, c_3) \quad [5-2]$$

$$\mathbf{K} = \frac{k}{q_4^3}\mathbf{I} \quad \mathbf{C} = \text{diag}(c_1, c_2, c_3) \quad [5-3]$$

$$\mathbf{K} = k\text{sgn}(q_4)\mathbf{I} \quad \mathbf{C} = \text{diag}(c_1, c_2, c_3) \quad [5-4]$$

where u is the control effort and all k 's and c 's are positive.

Sliding Mode Control Theory and Background

This section will present an overview of sliding mode to the reader. Further in depth explanation can be obtained from Slotine [42] or DeCarlo et al. [43].

What is Sliding Mode?

Nonlinear model based control systems offer a level of dynamic capabilities which linear techniques are incapable of providing when dealing with parameter uncertainties and unmodeled dynamics. Sliding mode, which has been studied in the Soviet Union for many years, is categorized as a variable structure control. This control structure has excellent stability, robustness, and disturbance rejection characteristics. Sliding mode is not a new control technique either, many researchers have utilized its robust properties to control a variety of systems such as missiles [44], mechanical systems [45], robotic manipulators [46][47][48], and submarines [49][50][51][52].

Theory

First let's look at a simple example of controlling a system. Given the system below,

$$\ddot{y} = u(t) \quad [5-5]$$

$$u(t) = -k \cdot y(t) \quad [5-6]$$

a solution to this differential equation can be found and is plotted in Figure 5-4. Notice that neither of the two solutions drive the system to zero at the origin, but rather remains oscillating with a stable behavior. The trick is to switch between the two solutions depending on which quadrant the system is in.

$$u(t) = \begin{cases} -k_1 \cdot y(t) & \text{if } \dot{y} < 0 \\ -k_2 \cdot y(t) & \text{otherwise} \end{cases} \quad [5-7]$$

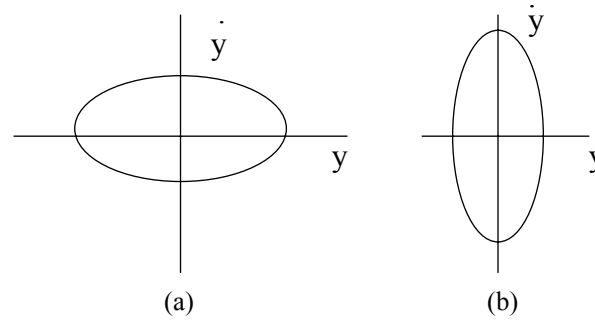


Figure 5-4. Two possible solutions for the double integrator, where (a) is $k = k_1$ and (b) is $k = k_2$. These plots are typically referred to as phase portraits. They plot position and velocity in this example, but later (when we are doing controls with full state feed back) the portraits will represent error and error velocity.

By switching, we can drive the system to the origin in a stable manor, as shown in Figure 5-5. Slotine took this idea and developed a systematic method for designing sliding mode controllers. Looking at Figure 5-5, we can develop the following table:

Table 5-2: Summary of the switching law.

Quadrant	\dot{y}	y	$u(t)$
1	+	+	$-k_2$
2	+	-	k_1
3	-	+	k_1
4	-	-	$-k_2$

This table can be summarized by the switching law.

$$\text{sgn}(s) = \begin{cases} -1 & s(y, \dot{y}) > 0 \\ 1 & s(y, \dot{y}) < 0 \end{cases} \quad [5-8]$$

where

$$s(y, \dot{y}) = \dot{y} + \lambda y \quad [5-9]$$

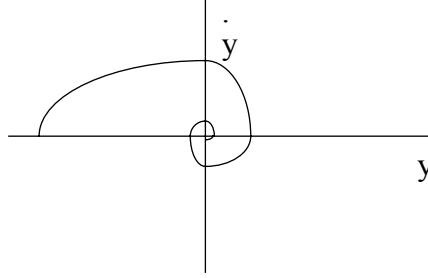


Figure 5-5. Switching between the two controllers results in driving the system to zero in a stable manor.

is called your sliding surface and the control effort is given by:

$$u(t) = -K \cdot \text{sgn}(s) \quad [5-10]$$

where K is typically a scalar or diagonal matrix of positive definite gains.

In order to design a controller, we will now replace our states (y, \dot{y}) with our errors $(\tilde{x}, \dot{\tilde{x}})$. This will now (as previously stated) drive our system errors to zero in a stable manor. Second, we will modify the control effort equation to include a feed forward aspect as follows.

$$u(t) = \hat{u} - K \cdot \text{sgn}(s) \quad [5-11]$$

The term \hat{u} is called the equivalent control effort. It is model based, thus if we have a perfect model of the system, we can calculate the required control effort to produce the desired results. However, that is never the case. There are always disturbances and unmodelled dynamics that influence our system. Thus the job of the second term in [5-11] is to reduce the error and error velocity produced by the equivalent control in a stable manor. Also note that the λ is a weighting factor between the error and error velocity. This term can be adjusted depending on which is more important, and its effects are depicted

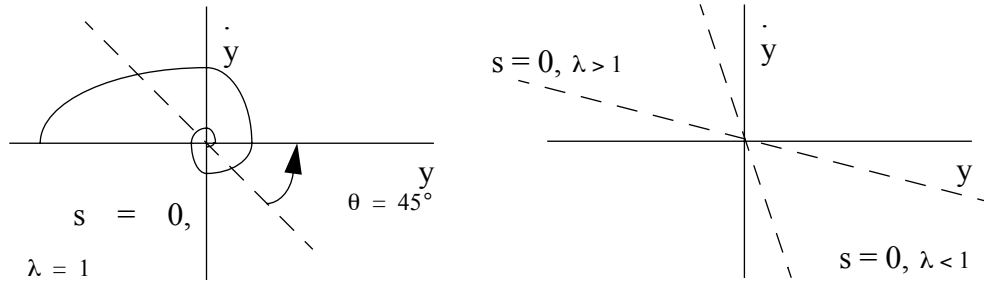


Figure 5-6. Graphical representation of how λ effects the sliding surface's orientation in the phase plane. The dotted lines represent the sliding surface ($s = 0$).

below in Figure 5-6. Typically this term is constant, but there is no reason it could not be dynamic.

Simple Example

For a simple example, we will develop a sliding mode controller for a mass-spring-dampener system. The dynamic equations for this system is

$$m\ddot{x} + c\dot{x} + kx = u. \quad [5-12]$$

Our sliding surface and equivalent control are

$$s = \dot{\tilde{x}} + \lambda\tilde{x}, \quad [5-13]$$

$$\dot{s} = \ddot{\tilde{x}} + \lambda\dot{\tilde{x}} = 0 \quad [5-14]$$

$$m^{-1}(\hat{u} - c\dot{\tilde{x}} - k\tilde{x}) - \ddot{\tilde{x}}_d + \lambda\dot{\tilde{x}} = 0 \quad [5-15]$$

$$\hat{u} = m(\ddot{\tilde{x}}_d - \lambda\dot{\tilde{x}}) + c\dot{\tilde{x}} + k\tilde{x} \quad [5-16]$$

Now looking at [5-16] we notice that if there is no error in the modelling of the system or disturbances, then this control effort is sufficient to control the system. Also if we had a model of any disturbances in the system, we could include them in our equivalent control

term. However in this example, there are no disturbances included in the system or we do not have a model for any disturbances seen by the system during run-time. Unfortunately there will always be error in modelling, noise, or disturbances. Thus we must include another term to account for these problems.

$$u = \hat{u} - K \cdot \text{sgn}(s) \quad [5-17]$$

Other Sliding Mode Designs for Spacecraft

Many authors have suggested many types of control architectures for satellite attitude control [53-59]. However, the one of the most interesting and simpler to implement types of control is sliding mode. The use of sliding mode control is not new to satellite attitude control. Lo and Chen [60] designed a sliding mode controller scheme which avoids the inverse of the inertia matrix, and smooths the control effort of the controller. Such a strategy provides for a more efficient use of fuel. Their simulations involved small uncertainties in the spacecraft inertia matrix and a sinusoidal disturbance. Simulations showed favorable results. The control effort was

$$u = \tilde{\omega} \times J\tilde{\omega} + J\dot{v}_d - \lambda J\tilde{q} - k \text{sgn}(s) \quad [5-18]$$

where the $\dot{v}_d \approx \dot{\omega}_d$ for all intensive purposes.

Boskovic et al. [61] and [62] developed a sliding mode controller, but specifically designed the controller such that it did not saturate the control effort. They were able to produce accurate results, and even tested the controller on a system with a larger inertia matrix than what the controller was designed with. Unfortunately they drew an incorrect conclusion that the controller is independent of the inertia matrix it is presented with. Sliding mode will be able to control a system with a larger inertia matrix, but may not control a system with a smaller inertia matrix. This is because the control effort is model

based, and thus it is possible to make a satellite go unstable by issuing too much control effort. The simulations utilized a square wave disturbance and small uncertainties in the inertia matrix. The controller developed was as follows:

$$u = \tilde{\omega} \times J\tilde{\omega} - \lambda J\dot{q} - \text{diag} \begin{bmatrix} \text{sgn}(s_1) & \text{sgn}(s_2) & \text{sgn}(s_3) \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \\ k_3 \end{bmatrix} \quad [5-19]$$

Crassidis et al. [63] developed an optimal variable-structure (sliding mode) controller for large-angle maneuvers on satellites. A cost function was developed, which when minimized, resulted in the optimal control effort. The the minimization lead to a two point boundary value problem. The cost function and the resulting optimal control effort were

$$\Pi(\omega) = \frac{1}{2} \int_{ts}^{\infty} (\rho \tilde{q}^T \tilde{q} + \tilde{\omega}^T \tilde{\omega}) dt \quad [5-20]$$

$$u = \omega \times J\omega + J \left(\frac{1}{2} \lambda \text{sgn}(\tilde{q}_4) [\Xi(q) \Xi(q_d) \omega_d - \Xi(q_d) \Xi(q) \omega] \right) + \dot{\omega}_d - k \text{sgn}(s) \quad [5-21]$$

Simulations of the controller on the MAP satellite showed favorable results.

Coleman and Godbole [64] conducted a performance trade study between fuzzy logic, PID, and sliding mode control. The controllers were tuned for and tested on three similar linear plants. In all three cases, the sliding mode controller out-performed the fuzzy logic controller. This is a typical result, where sliding mode tends to provide a superior model based performance compared to fuzzy logic, assuming the model is accurate enough.

However all of these authors utilized simple satellite dynamics with a simple external disturbance in their simulations. Fuel slosh, which is coupled with the dynamics of the satellite, is a more complex problem. It is difficult to accurately model the real disturbance and requires a control system that can account for this uncertainty.

Satellite Controller Design For This Work

This section will cover the sliding mode controller developed for this research.

Controller Derivation

This formulation, which is a full-state feedback technique, utilizes the existing dynamic model and compensates for uncertainty while formulating a control effort that tracks the desired trajectories. The equations of motion for a satellite's attitude (in quaternion space) are given by [22]

$$J\dot{\omega} = -\Omega J\omega + f + u \quad [5-22]$$

$$\dot{q} = \frac{1}{2}\Omega q + \frac{1}{2}q_4\omega \quad [5-23]$$

where

$$\Omega = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \quad q = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} \quad [5-24]$$

Also f in [5-22] represents the modelling error and potential disturbances in the system. This term will provide a feed forward aspect to the controller and should yield better results. The q is a vector composed of the three imaginary elements of a quaternion. However, in this formulation, we can take the state variable to be Δ or small euler angle orientation, and equate this to q .

$$\begin{bmatrix} \theta_x \\ \theta_y \\ \theta_z \end{bmatrix} = 2 \cdot \begin{bmatrix} q_x \\ q_y \\ q_z \end{bmatrix} \quad [5-25]$$

Thus the sliding surface (s) is given by:

$$s = \left(\frac{d}{dt} + \lambda \right)^{n-1} \tilde{x} = \tilde{\dot{x}} + \lambda \tilde{x} = \tilde{\dot{\omega}} + \lambda \tilde{q} \quad [5-26]$$

where the error terms are defined as:

$$\tilde{x} = x - x_{desired} \quad [5-27]$$

In order to calculate the equivalent control effort (\hat{u}), we need to take the derivative of the sliding surface and set it equal to zero.

$$\dot{s} = \tilde{\dot{\omega}} + \lambda \tilde{q} \quad [5-28]$$

Now multiplying both sides by the inertia matrix (J) will allow us to substitute in the equations of motion ([5-22] and [5-23]).

$$J\dot{s} = \hat{J}\tilde{\dot{\omega}} + \lambda\hat{J}\tilde{q} = \hat{J}\dot{\omega} + \lambda\hat{J}\tilde{q} - \hat{J}\dot{\omega}_{desired} \quad [5-29]$$

$$J\dot{s} = \Omega\hat{J}\omega + \hat{f} + \hat{u} + \lambda\hat{J}\tilde{q} - \hat{J}\dot{\omega}_{desired} \quad [5-30]$$

The hat over particular variables denotes that they are estimates, since accurate values for the inertia matrix and disturbances may not be known. Since the sliding surface does not move, its time derivative is zero, and thus the equation becomes

$$J\dot{s} = \Omega\hat{J}\omega + \hat{f} + \hat{u} + \lambda\hat{J}\tilde{q} - \hat{J}\dot{\omega}_{desired} = 0. \quad [5-31]$$

Now solving for the equivalent control effort (\hat{u}) from [5-31] yields:

$$\hat{u} = -\Omega\hat{J}\omega - \hat{f} - \lambda\hat{J}\tilde{q} + \hat{J}\dot{\omega}_{desired}. \quad [5-32]$$

The quaternion error rate in the above equation is defined as:

$$\tilde{q} = \frac{1}{2}\tilde{\Omega}q + \frac{1}{2}q^4\tilde{\omega} \quad [5-33]$$

$$\tilde{\boldsymbol{\Omega}} = \begin{bmatrix} 0 & -\tilde{\omega}_3 & \tilde{\omega}_2 \\ \tilde{\omega}_3 & 0 & -\tilde{\omega}_1 \\ -\tilde{\omega}_2 & \tilde{\omega}_1 & 0 \end{bmatrix} \quad [5-34]$$

The estimated disturbance in the above equation is equal to the estimated solar snap and fuel slosh disturbance.

$$\hat{\mathbf{f}} = \hat{\mathbf{f}}_{ss} + \hat{\mathbf{f}}_{fs} \quad [5-35]$$

where f_{ss} is the disturbance due to solar snap and f_{fs} is the disturbance due to fuel slosh.

Finally, the sliding mode control effort is given by:

$$\mathbf{u} = \hat{\mathbf{u}} - K \cdot \text{sgn}(\mathbf{s}) \quad [5-36]$$

The K term in [5-36] will be discussed when stability of the controller is covered. This switching term provides the appropriate control action to quickly drive the trajectories back onto the sliding surface. Since real actuators do not have an infinite bandwidth, the sign function tends to cause excessive chatter in the control effort. The sign function is typically replaced by a saturation function. The saturation function not only smooths the control response, it also reduces the amount of fuel used by the controller.

Fuel Slosh Disturbance

The effects of the fuel slosh are incorporated into the controller by the \hat{f}_{fs} term in the controller. A discrete state space system is used to track the effects of the fuel slosh:

$$\dot{\mathbf{x}}_{fs} = \mathbf{A}\mathbf{x}_{fs} + \mathbf{B}\mathbf{u} \quad [5-37]$$

$$\hat{\mathbf{f}}_{fs} = \mathbf{C}\mathbf{x}_{fs} \quad [5-38]$$

where the state vector is:

$$\mathbf{x}_{fs} = \begin{bmatrix} \dot{x} \\ x \\ x \end{bmatrix} \quad [5-39]$$

where x is the position of the fuel slosh and \dot{x} is its velocity. The input to the system u is the previous control effort used by the satellite. Also the state space matrices are discrete time representations to reduce the computational load of the controller.

Stability of Controller

The following Lyapunov candidate function is proposed. This function is clearly positive definite.

$$V = \frac{1}{2} \mathbf{s}^T \mathbf{J} \mathbf{s} \quad [5-40]$$

Taking its time derivative and substituting in previous equations of motion and control effort gives:

$$\dot{V} = \mathbf{s}^T \mathbf{J} \dot{\mathbf{s}} \quad [5-41]$$

where the true dynamics of the system are inserted into the equations are

$$\mathbf{J} \dot{\mathbf{s}} = -\tilde{\boldsymbol{\omega}} \times \mathbf{J} \tilde{\boldsymbol{\omega}} + \mathbf{u} + \mathbf{f} + \lambda \mathbf{J} \tilde{\mathbf{q}} \quad [5-42]$$

The modelled dynamics, which define the controller output are given by:

$$\mathbf{u} = \hat{\mathbf{u}} - K \text{sgn}(\mathbf{s}) \quad \text{and} \quad \hat{\mathbf{u}} = \tilde{\boldsymbol{\omega}} \times \hat{\mathbf{J}} \tilde{\boldsymbol{\omega}} - \lambda \hat{\mathbf{J}} \tilde{\mathbf{q}} \quad [5-43]$$

After some algebra, the dynamics cancel each other out, and the only terms left are the difference between the true disturbance forces and the estimated/modelled ones and the last terms on the control effort equation. Thus [5-41] becomes:

$$\dot{V} = \mathbf{s}^T (\mathbf{F} - K \text{sgn}(\mathbf{s})) \quad [5-44]$$

where

$$\mathbf{F} = |\mathbf{f} - \hat{\mathbf{f}}| \quad [5-45]$$

Note that any differences between the true inertia matrix and the real inertia matrix, plus any other modelling error or disturbance, is contained in \mathbf{f} . Where \mathbf{f} is the true disturbances, modelling errors, unmodelled higher order terms, etc.

For a stable system, the lyapunov derivative needs to be negative definite. All variables are known except for the scalar gain value K . Two of the three remaining terms are positive definite, K and F . Thus we need to do further simplifications and algebra to solve for the gain.

$$\mathbf{s}^T \mathbf{F} - \mathbf{s}^T K \text{sgn}(\mathbf{s}) < 0 \quad [5-46]$$

$$\mathbf{s}^T \mathbf{F} - K \langle \mathbf{s} \rangle < 0 \quad [5-47]$$

where

$$\langle \mathbf{s} \rangle = \mathbf{s}^T \text{sgn}(\mathbf{s}) \quad [5-48]$$

This operation is a cross between an absolute value operation (if it were a scalar) and a norm of a vector (if the sign function was not present). Thus this operation will be denoted by the angled brackets and not confused with either absolute value or norm of a vector. A quick example is shown below, and note that the result will always be scalar and positive.

$$\begin{bmatrix} 1 & -3 & 4 \end{bmatrix} \text{sgn} \left(\begin{bmatrix} 1 \\ -3 \\ 4 \end{bmatrix} \right) = \begin{bmatrix} 1 & -3 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} = 8 \quad [5-49]$$

Thus the scalar gain K is

$$K > \frac{|\mathbf{s}^T \mathbf{F}|}{\langle \mathbf{s} \rangle} \quad [5-50]$$

In order to guarantee the above inequality, we will add a little offset (η) that will always be positive.

$$K = \frac{|s^T|F}{\langle s \rangle} + \eta \quad [5-51]$$

Now substituting this gain back into [5-47] results in:

$$s^T F - \left(\frac{|s^T|F}{\langle s \rangle} + \eta \right) \langle s \rangle < 0 \quad [5-52]$$

This equation has two results depending on whether s is positive or negative. If s is negative then the two $s^T F$'s sum together and the entire equation is negative definite. If s is positive, then the two cancel each other out and the equation is still negative definite.

$$2s^T F - \eta \langle s \rangle < 0 \quad \text{or} \quad -\eta \langle s \rangle < 0 \quad [5-53]$$

Saturation

The sign function used in the sliding mode controller leads to an excessive amount of chattering in the control effort. A common modification to the sliding mode controller is to change the sign function to a saturation function with the following properties.

$$sat(a) = \begin{cases} -1 < a < 1 & a \\ else & \text{sgn}(a) \end{cases} \quad [5-54]$$

Thus our sliding mode control equation changes to

$$\mathbf{u} = \hat{\mathbf{u}} - K sat\left(\frac{s}{\phi}\right) \quad [5-55]$$

where the ϕ is a scalar value that can be used to adjust when the saturation function will saturate. All other sliding mode equations remain the same.

Comparing The Two Types of Controller

Slotine's sliding mode formulation and a PD controller have some very similar attributes when the sliding mode is in the sliding region. Simplifying the system some, assume that the equivalent control effort (\hat{u}) is zero, since this is really a feed forward term. Thus the control effort is now:

$$\mathbf{u} = -Ksat\left(\frac{\mathbf{s}}{\phi}\right) \quad [5-56]$$

The sliding surface s is very similar to a PD control effort.

$$\mathbf{s} = \left(\frac{d}{dt} + \lambda\right)^{n-1} \tilde{\mathbf{x}} = \tilde{\dot{\mathbf{x}}} + \lambda \tilde{\mathbf{x}} \quad [5-57]$$

where $\tilde{\dot{\mathbf{x}}}$ is a rate error and $\tilde{\mathbf{x}}$ is a position error. When the errors are on the sliding surface (meaning when the errors are small, they are in the saturation bounds), the saturation function returns a scaled version of the values passed to it (as opposed to a 1 or -1 when outside of the saturation bounds). Thus the control effort becomes scaled errors and scaled rate error multiplied by a gain K .

$$\mathbf{u} = -K\left(\frac{\mathbf{s}}{\phi}\right) \quad [5-58]$$

$$\mathbf{u} = -\frac{K}{\phi}\left(\tilde{\dot{\mathbf{x}}} + \lambda \tilde{\mathbf{x}}\right) \quad [5-59]$$

Then the PD parameters can be written as follows:

$$k_p = \frac{\lambda K}{\phi} \quad T_i = \frac{1}{\lambda} \quad [5-60]$$

Conclusion

This chapter introduced PD and sliding mode theory for satellite attitude control. An in depth discussion of sliding mode design, stability, and a comparison between sliding mode

and PD was provided. This laid the foundation for the similarities and differences between the two controller which will soon manifest themselves when the results are presented.

CHAPTER 6 SIMULATION AND RESULTS

This chapter will discuss the simulation, the gains chosen for the PD controller, and finally discuss the results. The first section will describe the specifics of the satellite and the simulation. All of the variable from proceeding chapters will be filled in and explained. Next the simulation, equations of motion, and the controllers that were developed are covered. Finally the results of the simulation are discussed.

Simulation Dynamics and Parameters

The simulation used to evaluate the performance of the controllers was written in C++. The first simulations were written in Matlab because of the ease of working with vectors and matrices. The problem with Matlab, however, was the amount of time each simulation took. Once the simulation were rewritten for C++, they were magnitudes faster. The reader is referred to Appendix C, where some of the mathematical code developed is presented.

Satellite

The satellite shown in Figure 6-1 and Figure 6-2 is Earth Observing 1 (EO-1), and will be the satellite used in all simulations. EO-1 was primarily constructed by Swales

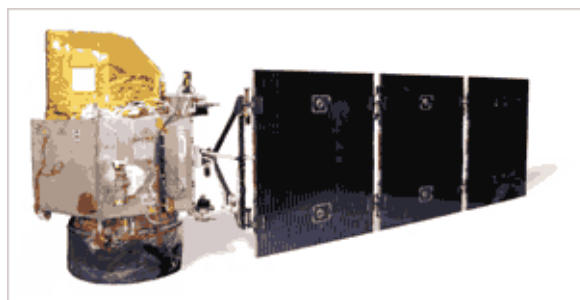


Figure 6-1. EO-1 satellite with solar panel deployed.

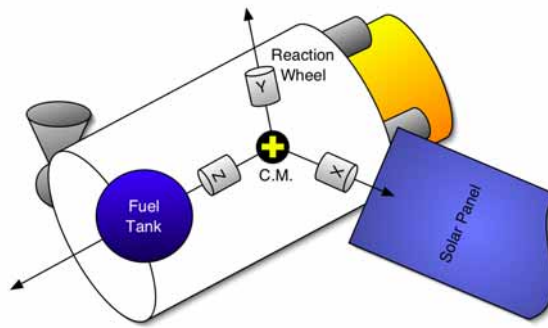


Figure 6-2. Diagram of satellite's internal components.

Aerospace and is part of the New Millennium Program which is dedicated to validating new technologies. The spacecraft was launched aboard a Boeing Delta II rocket from Vandenberg Air Force Base on November 21, 2000. Some of the new technology being tested is the Advanced Land Imager, hyperspectral imaging spectrometer called Hyperion, X band phased array, pulsed plasma thrusters, and formation flying with Landsat 7 spacecraft which is already in orbit.

Table 6-1: Technical Specifications for the EO-1 spacecraft

Dry Mass	568 kg
Volume of Bus	1.5m x 1.5m x 2m
Inertia Tensor	[443 179 429] kg m ²
ACS	Zero Momentum, 3 axis stabilized
Command and Data Handling Bus Architecture	Mongoose V, Rad Hard at 12 Mhz, RISC Architecture
Solar Arrays	3 panel / Si w/GaAs / Articulating
Bus Structure	Hexagonal with aluminum honeycomb
Propulsion	1 fuel tank with 4 thrusters
Propellant Capacity	23 kg
Mission Design Life	1.5 years

Satellite Dynamics

The true equations of motion for the satellite's linear and angular momentum are the now familiar ones:

$$\dot{\mathbf{p}} = -\boldsymbol{\omega} \times \mathbf{p} \quad [6-1]$$

$$\dot{\mathbf{h}} = -\boldsymbol{\omega} \times \mathbf{h} - \mathbf{v} \times \mathbf{p} + \mathbf{g} \quad [6-2]$$

The differential equation for the linear momentum of sloshing mode i is:

$$\dot{\mathbf{p}}_i = m_{fi} \boldsymbol{\omega} \times (\mathbf{v} - \mathbf{r}_{fi} \times \boldsymbol{\omega}) - c_f \dot{\mathbf{n}}_i - k_{fi} \mathbf{n}_i \quad [6-3]$$

where for this work there are two sloshing modes. The wheel dynamics are:

$$\dot{\mathbf{h}}_a = \mathbf{g}_a \quad [6-4]$$

The mass matrix of the spacecraft, which contains the linear and angular momentums for the various components, is:

$$\boldsymbol{\rho} = \mathbf{M}\mathbf{V} \quad [6-5]$$

$$\boldsymbol{\rho} = \begin{bmatrix} \mathbf{p} & \mathbf{h}_o & \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{h}_a \end{bmatrix}^T \quad \mathbf{V} = \begin{bmatrix} \mathbf{v} & \boldsymbol{\omega} & \dot{\mathbf{n}}_1 & \dot{\mathbf{n}}_2 & \boldsymbol{\omega}_s \end{bmatrix}^T \quad [6-6]$$

$$\mathbf{M} = \begin{bmatrix} m\mathbf{I} & -\mathbf{c} \times & \mathbf{I}m_{f1} & \mathbf{I}m_{f2} & 0 \\ \mathbf{c} \times & \mathbf{J} & m_{f1}\mathbf{b} \times & m_{f2}\mathbf{b} \times & \mathbf{I}_s\mathbf{a} \\ \mathbf{I}m_{f1} & -\mathbf{I}m_{f1}\mathbf{b} \times & -\mathbf{I}m_{f1} & 0 & 0 \\ \mathbf{I}m_{f2} & -\mathbf{I}m_{f2}\mathbf{b} \times & 0 & -\mathbf{I}m_{f2} & 0 \\ 0 & \mathbf{I}_s\mathbf{a} & 0 & 0 & \mathbf{I}_s\mathbf{I} \end{bmatrix} \quad [6-7]$$

Fuel Tank

The location of the fuel tank in the body frame is $\mathbf{b} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$ and the remaining parameters are shown below.

Table 6-2: EO-1 fuel slosh parameters.

	Mode 1	Mode 2
Fuel Mass (m_f)	15 kg	8 kg
Natural Freq. (ω)	0.1 Hz	0.3 Hz
Dampening Ratio (ζ)	0.5	0.5
Neutral Location from C.M. (b)	1 m	1 m

Reaction Wheels

The reaction wheels used on EO-1 are customized versions Ithaco TW-4A12. Their specifications used in this work are shown below.

Table 6-3: EO-1 wheel specifications.

Mass	2.55 kg	
Speed	\pm 5100 rpm	
Dimensions	Dia. 20.5 cm	Height 6.4 cm
Limits	Torque 0.025 Nm	Momentum 4 Nms
Location from C.M.	.5 m	

Each of the three reaction wheels is located on one of the three prime axes, half a meter from the center of mass.

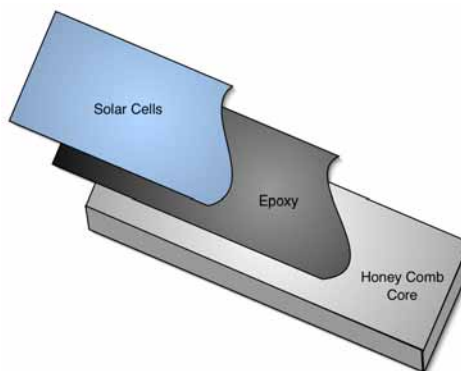


Figure 6-3. Composition of a typical simple solar panel.

Solar Panels

The parameters used for the solar panel in this work are shown below in Table 6-4. Since a solar panel is actually a composite of various materials and components as shown in Figure 6-3. Thus the parameters are primarily based on the largest component, the aluminum honey comb core. Johnson and Thorten [68] provides a formula for calculating

Table 6-4: EO-1 solar panel parameters.

Material	Aluminum 6061 honey comb
Dimensions (w x L x h)	1.428 m x 3.8 m x 0.03 m
Density (ρ)	300 kg/m ³
Natural Freq. (ω)	0.48, 3.0, 8.4 Hz
Structural Dampening Ratio (ζ)	0.1
Structural Stiffness (EI)	2E4 N m ²
Specific Heat (c)	920 J/kg K
Thermal Conductivity (k)	1.2 W/m K
Coefficient of Thermal Expansion (α_T)	22.7E-6 1/K
Location from C.M. (R_o)	1 m

the material constants taking into account that the solar panel is a composite structure. However the formula appeared in a journal article with no explanation where it came from, nor of the constants used to weight the different parameters.

These parameters have been determined based on what little information that is publicly available from the manufacturer of EO-1's solar panel. Note that the dampening ratio used was a very small number since structural dampening is assumed.

Control System

The control system issues control corrections to the spacecraft at 1 Hz with the colored noise filter running at a faster 4 Hz. The controllers operate with a 5% error in the inertia matrix.

$$\text{PD: } K_p = 1.5 \quad T_d = 0.1$$

$$\text{SM: } \lambda = 0.1 \quad \eta = 0.3 \quad \phi = 0.1$$

Finding an Optimal PD Gain

In order to compare the PD controller to the sliding mode controller, an attempt to find the best gains was conducted. A simulation was run which looped through various values of the PD controller.

$$\mathbf{u} = k_p \mathbf{e} + k_d \dot{\mathbf{e}} = k_p (\mathbf{e} + T_d \dot{\mathbf{e}}) \quad [6-8]$$

where u is the control effort, k_p is the proportional gain, k_d is the derivative gain, e is the attitude error in the system, \dot{e} is the rotational rate error, and $T_d = \frac{k_d}{k_p}$. The dynamics used for the satellite were basically Euler's equation, which are commonly used to represent a simple spacecraft.

$$\dot{\mathbf{h}}_{sat} = -\boldsymbol{\omega} \times \mathbf{h}_{sat} - \boldsymbol{\omega} \times \mathbf{h}_w - \mathbf{N}_w + \mathbf{N}_{ext} \quad [6-9]$$

where h_{sat} is the angular momentum of the satellite, h_w is the angular momentum of the wheels, and N_{ext} are the external disturbance torques. These dynamics were used because they are simpler and allowed the search to proceed at a greater rate. Also no mathematical model ever exactly matches the real system. Thus it was seen as undesirable to train the satellite on the same model that the experimenting would be conducted on. The reasoning, again, is that in real life whatever model the controller was trained on before the spacecraft was put into space would have some error, big or small.

The satellite was commanded to perform a step maneuver which rotated the satellite through a rotation from euler angles [-30 -30 -30] to [30 30 30]. Each time the values for K_p and T_d were changed and various performance criteria were calculated. The

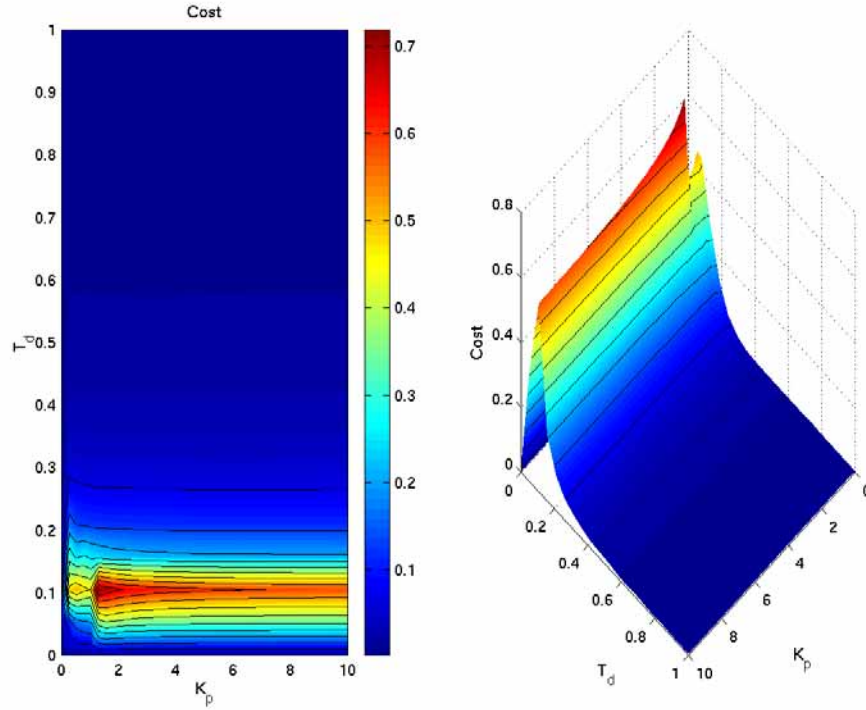


Figure 6-4. Cost function surface plot for the optimal gain of the PD controller.

performance measures used were the Integration of Absolute Error (IAE), Integration of Time times Absolute Error (ITAE), and settling time (t_s) of the system.

$$IAE = \int_0^t |q_{error}| dq_{error} \quad [6-10]$$

$$ITAE = \int_0^t |q_{error}| t dt \quad [6-11]$$

The error used in the calculations came from the quaternion error (q_{error}) and the absolute value bars indicate the euclidian norm of the quaternion error vector.

Then a cost function was calculated using these individual measures to determine the optimal controller gains.

$$Cost = Cost_{itae} \cdot Cost_{iae} \cdot Cost_{t_s} \quad [6-12]$$

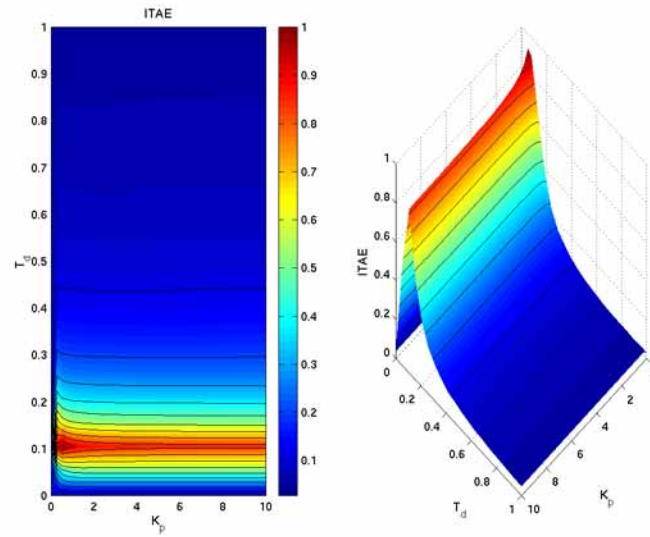


Figure 6-5. ITAE results from the optimal PD search.

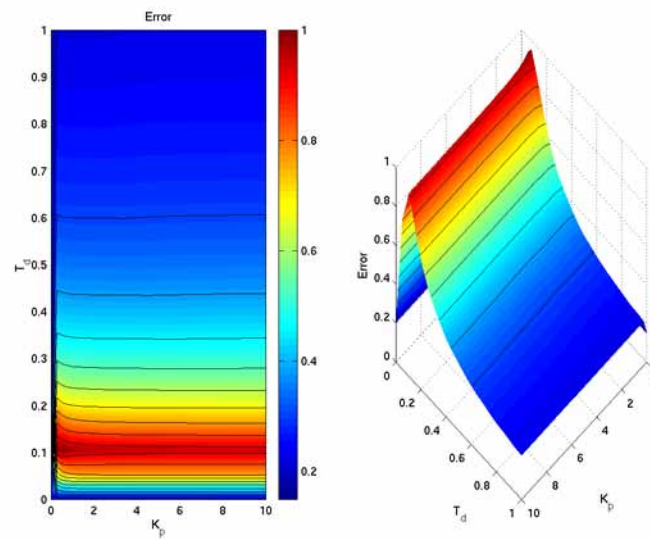


Figure 6-6. IAE results from the optimal PD search.

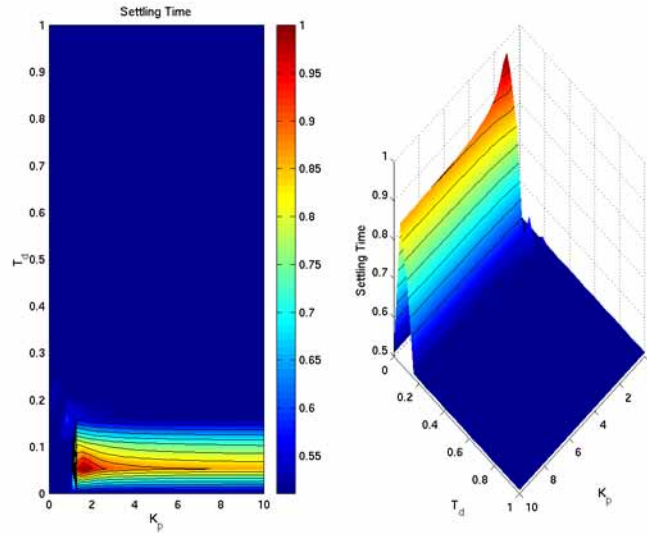


Figure 6-7. Settling time results from optimal PD search.

This cost function was used because these were some of the performance measurement used to evaluate the different controllers. Thus optimizing the PD controller's gains for the performance measurements would help it perform better.

Each of the individual measures were also normalized by dividing them by the maximum value found after the search was completed. For example, the ITAE would have been calculated as follows:

$$itae = \frac{1}{ITAE} \quad [6-13]$$

$$Cost_{itae} = \frac{itae}{\max(itae)} \quad [6-14]$$

Thus the values for the cost functions that interest us most are the highest values (i.e., 1.0). The smaller error in the system and the shortest settling time will all have a value of one. Also the cost function is now unit less since there is a division by the maximum value (i.e., the units cancel).

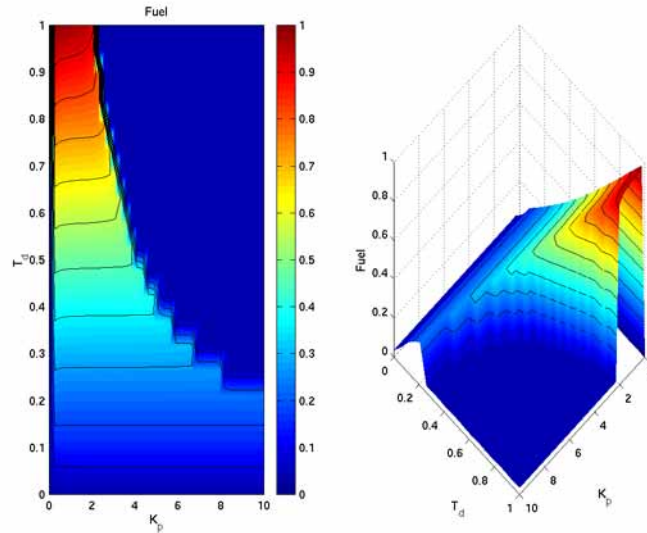


Figure 6-8. Control momentum or fuel used searching for the optimal PD controller.

The resulting surface plot of the cost function is shown in Figure 6-4. Here the optimal gains were: $K_p = 1.5$ and $T_d = 0.1$.

Looking at the individual components of the cost function, the IAE and ITAE in Figure 6-6 and Figure 6-5 produce similar surfaces. Although the ITAE produced a narrower ridge of gains, both criteria prefer lower gain values that run along a ridge approximately located at $T_d = 0.1$.

Next looking at the settling time results in Figure 6-7, a similar ridge to the error surfaces was formed. The ridge again showed a preference for lower gains like the error surfaces. Here, however, the ridge resides slightly lower than the error gains; running approximately along $T_d = 0.06$.

The cost function for finding an optimal gain was composed of the error terms and the settling time. Originally, however, a term for the amount of power / fuel / control momentum was intended on being included. The results of this performance criteria is

shown in Figure 6-8. This term was not used since it favors higher T_d values, but when looking at pervious surfaces, it becomes apparent that performance would suffer greatly.

Fuel Slosh

This section and the one following it will talk about the performance of three different controllers. The controller used in this work were the proportional derivative (PD), sliding mode (SM), and sliding mode with a colored noise filter to compensate for fuel slosh (SMCNF).

A step simulation was used to demonstrate the controller's ability to react to the two main disturbances: fuel slosh and gravity gradient torque. The simulation had all euler angle set equal to each other, but the initial and final conditions where differentiated by sign. Table 6-5 shows the three sets of initial and final angles used along with the angular distance between the two orientations.

Table 6-5: Initial and final euler angles where roll, pitch, and yaw are equal.

Start (deg)	Stop (deg)	Angular Distance (deg)
-10	10	34.46
-30	30	98.99
-60	60	165.64

The results of the simulations were compared by looking at four performance criteria. They were ITAE, IAE, T_s (settling time), and CM (Control Momentum in Nms). The CM was determined by integrating the euclidian norm of the control effort ($u(t)$) used by the controller from time 0 to the end of the simulation (t_{final}).

$$CM = \int_0^{t_{final}} |u(t)| dt \quad [6-15]$$

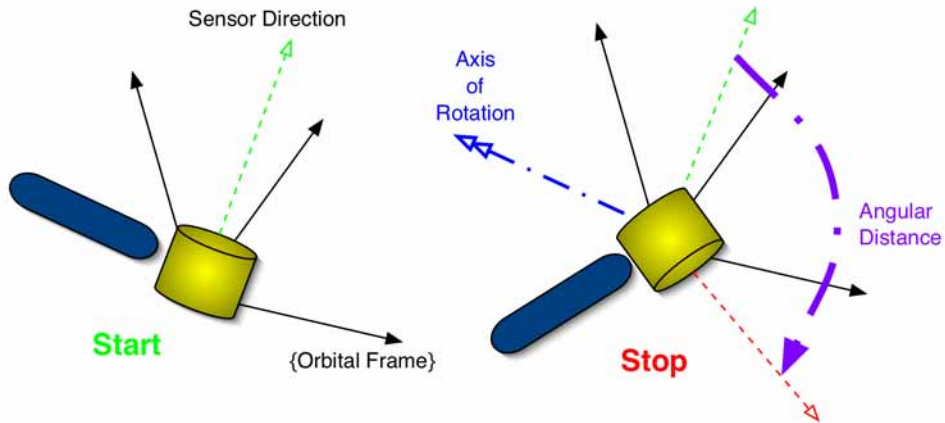


Figure 6-9. A depiction of the step maneuver showing the orientation of the satellite at start and end. The angular distance and axis of rotation are also shown for better understanding of the terms.

IAE

The IAE is the sum of error in the system. The amount of error in a system is proportional to the amount of control effort used to get reduce the error. In a dynamic system, such as the satellite, there are other factors that contribute to the amount of error and control effort used, such as momentum and external disturbances. The results for the simulations were:

Table 6-6: IAE results with values having units of radians.

Sim	PD	SM	SMCNF
10	230	198	197
30	887	861	858
60	2292	2169	2142

The sim column in Table 6-6 refers to the euler angles. Thus the 10 row means the satellite rotated from euler angles $[-10 \ -10 \ -10]$ to $[10 \ 10 \ 10]$. Notice that the numbers decrease from left to right. This means that the SM controller produced less error than the

PD controller, and the SMCNF controller produced less error than the SM and PD controllers. The only exception to this was the results for the PD controller's IAE at 60.

ITAE

The ITAE is a measurement similar to the IAE, but ITAE keeps track of when an error occurs. Errors that occur early in a simulation are weighted very small while errors that occur later in the simulation are weighted larger. This performance criteria has the effect of favoring systems with short settling times, but does not penalize controller that have large overshoots very early in the simulation.

Table 6-7: ITAE results with values having units of radian-seconds².

Sim	PD	SM	SMCNF
10	4249	2756	2745
30	18036	16476	16377
60	84504	71924	68917

The numbers in Table 6-7 again decrease from left to right, meaning that the SMCNF controller performed better than the previous two.

Settling Time

The settling time is the amount of time the system takes to reach a steady state value. The spacecraft reached steady state when its quaternion error was below 100 arcsec¹.

Table 6-8: Settling time results with values having units of seconds.

Sim	PD	SM	SMCNF
10	509	278	278
30	576	370	362
60	918	730	721

1. An arc second (arcsec) is 1/3600 of a degree or one degree is equal to 3600 arcsec. This is a very fine unit of measurement used for high precision orientation.

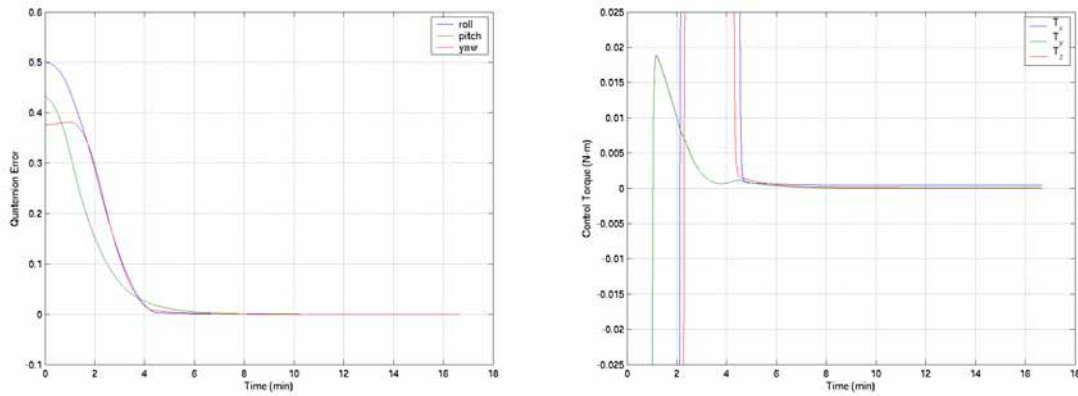


Figure 6-10. Quaternion error and control effort of the PD controller rotating from -30 to 30.

The results shown in Table 6-8 show that the SM produces faster setting times than the PD, but the SMCNF does not always produce faster setting times than the SM.

CM

The control momentum (CM) is a measurement of the amount of energy used by each controller. The results for the three simulations are shown in Table 6-9.

Table 6-9: CM results with values having units of Nms.

Sim	PD	SM	SMCNF
10	4.83	5.32	5.35
30	10.62	10.64	10.86
60	19.37	18.01	17.55

The SM and SMCNF controllers utilized more fuel than the PD controller, while the addition of the CNF to the SM produced a very small decrease in the amount of fuel consumed.

Torque and Momentum Limits

The final point of interest for this section was the maximum control efforts. The momentum wheels are limited to 0.025 Nm of torque by the simulation. The simulation

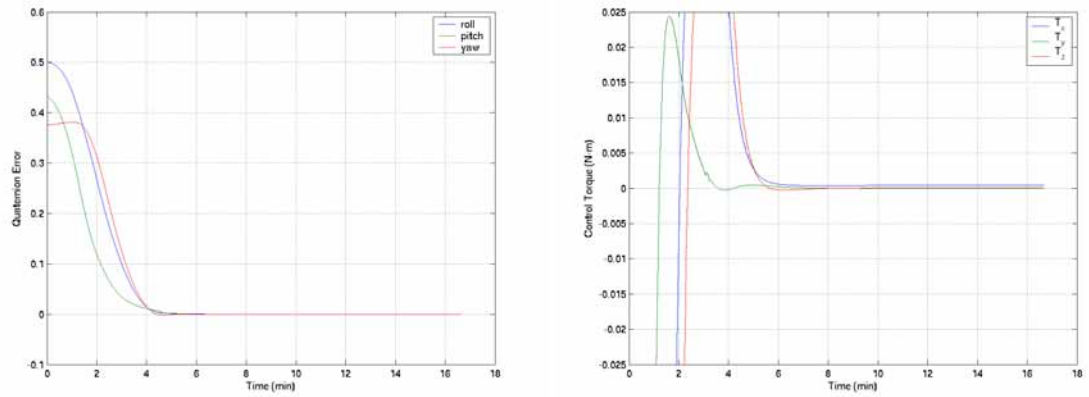


Figure 6-11. Quaternion error and control of the SM controller rotating from -30 to 30.

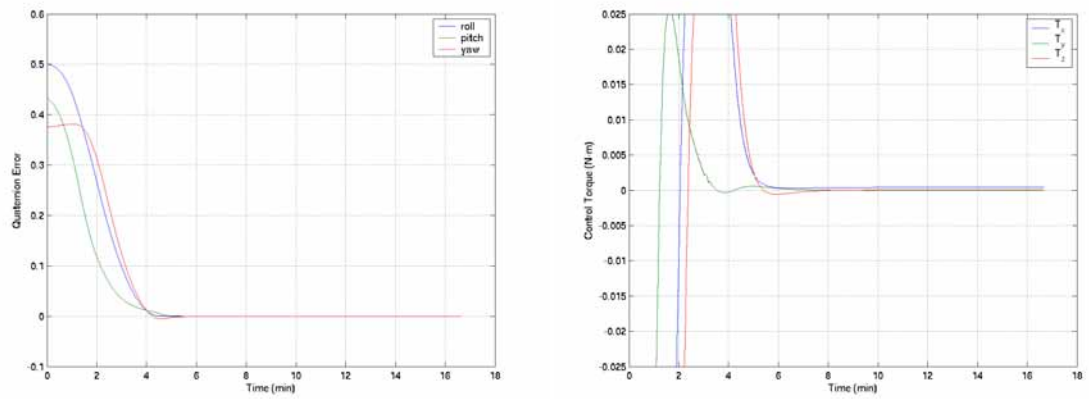


Figure 6-12. Quaternion error and control of the SMCNF controller rotating from -30 to 30.

does nothing to limit the momentum that the wheels can produce, but the maximum wheel momentum produced by the controllers was recorded.

Table 6-10: Maximum wheel momentum achieved where units are in Nms.

Sim	PD	SM	SMCNF
10	1.59	1.66	1.67
30	3.44	3.43	3.45
60	5.15	4.72	4.70

The wheels on EO-1 have a momentum limit of 4 Nms. Thus during the final maneuver of 60, the wheels would have saturated and the control system. The thruster are used to unload momentum and the wheels are slowed down to some nominal value. This maneuver would require a much more complex control system involving a thruster with bang-bang control which is outside of the scope of this work.

Qualitative Analysis of Results

Plotting the quaternion error for the 30 degree maneuver showed a smooth decrease of the error for each controller, as shown in Figure 6-10, Figure 6-11, and Figure 6-12. Thus there was little overshoot and waste of energy in the system. Although there was a definite difference between the PD and sliding mode controllers, the two sliding mode controllers showed little difference in their error plots. This was expected since the addition of the colored noise filter only slightly increases the performance of the sliding mode controller.

Solar Snap

The final simulation done involved looking at the effects of solar snap on the controllers.

Performance

Here the satellite tried to maintain a nadir¹ orientation as it orbits the Earth. The solar panel was then subjected to a heat flux which induced vibrations.

Table 6-11: Solar snap results of the three controllers.

	PD	SM	SMCNF
IAE (rad)	3.7	1.0	0.9
ITAE (rad-sec ²)	14.9	2.3	2.1
CM (Nms)	0.52	0.57	0.50
t_s (sec)	139	46	45

1. A nadir orientation means the satellite is pointing towards the center of the Earth.

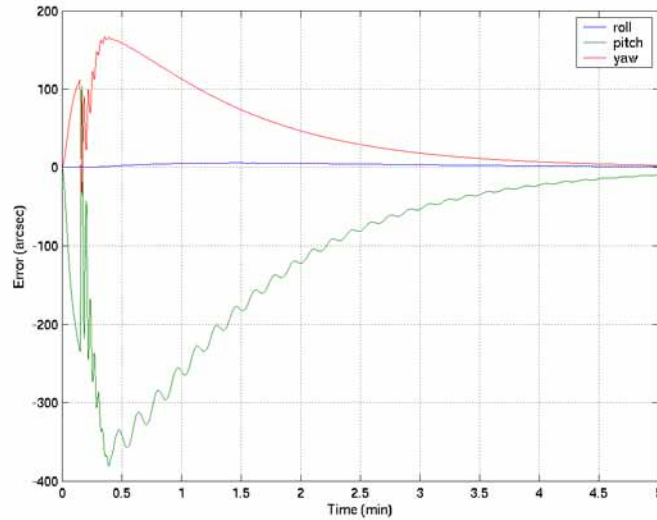


Figure 6-13. Arcsecond error of the PD controller during solar snap.

The results in Table 6-11 show that the SMCNF produced less error and a quicker settling time than the other two controllers, but at the expense of using more control effort.

Qualitative Analysis of Results

Again looking at the error produced by the three different controllers reveals the sliding mode controllers have better performance than the PD controller, as shown in Figure 6-13, Figure 6-14, and Figure 6-15. The error curves for the two sliding mode controllers look identical.

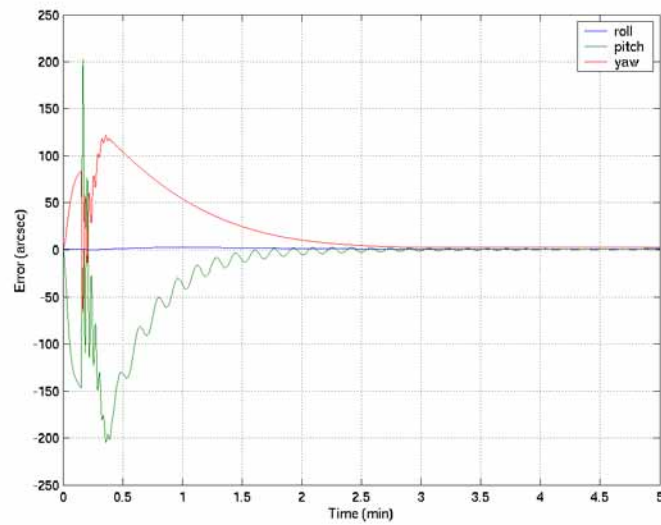


Figure 6-14. Arcsecond error of the SM controller during solar snap.

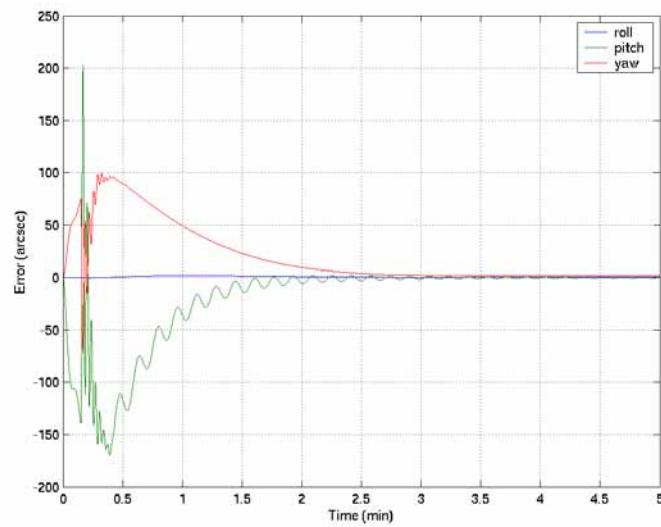


Figure 6-15. Arcsecond error of the SMCNF controller during solar snap.

CHAPTER 7

CONCLUSION

This work revolved around comparing two different types of attitude controllers: proportional derivative (PD) and sliding mode (SM). In order to make the comparison as good as possible, the PD controller was optimized to produce the best effort possible in this simulation. The SM was tuned by trial and error to achieve an acceptable results. The tuning took only a short amount of time, and easily produced better results than the PD controller. Thus the SM parameters used in this work may not be globally optimal.

The simulation written to test these controllers was written in C++. The benefits of programming in a language such as C++, as opposed to using prototyping software such as Matlab, was speed. The solar snap disturbances required very small time steps to avoid numerical instability, and under Matlab this translated into very long simulation times. Thus code written in C++ allowed equations to be written in clear mathematical expressions, but produced very fast execution times. A simulation in Matlab would take approximately 10 minutes, while the same simulation written in C++ would take 20 seconds. Since simulation times were now short, much experimentation could easily be conducted in a short amount of time.

Fuel slosh disturbances are a difficult problem to handle. They are impossible to measure or estimate without making assumptions about the form of the sloshing modes (e.g. pendulum, mass-spring-damper, liquid slug, etc.). These assumptions greatly simplify the problem to something that can be realized in an embedded system. Otherwise

very complex models which require massive amounts of computing power would need to be employed which is impossible in an embedded system like a satellite.

The SM controller was shown to provide better control than the venerable PD controller. The SM controller was designed to handle inaccuracies in the system model. Here the inaccuracies are movement of the fuel which change the inertia matrix during maneuvers. In order to improve the performance of the SM controller, a colored noise filter (CNF) was added to account for the disturbances. The results shown in the previous chapter illustrated how the CNF improved the SM controller's performance over the PD.

The SM and SMCNF controllers produced good results when encountering solar snap activity. Typically satellites stop their scientific mission during times of possible solar snap disturbance because the errors generally exceed the allowable range. Here the SM and SMCNF produce much quicker settling times and less than half the error of the PD controller. Thus the science mission could be run with less down time due to solar snap and could possibly even be run during solar snap disturbances if the SM or SMCNF errors are within an acceptable range.

This work clearly showed the superiority of SM or SMCNF over a PD controller. The question now becomes, which of the two types of sliding mode controllers was better. That really depends on the mission for which they would be applied. The SM is less computationally expensive than the SMCNF, and for satellites where fuel sloshing disturbances are small, computational power is very limited, and high precision is unnecessary, the SM controller would most likely be the better choice. This would be especially true where engineers are unfamiliar with sliding mode, and this is their first opportunity to work with it. However, in the situation where the opposite is true, then the

benefits of the CNF at the expense of increased computational cost and increased development time become attractive. Thus, in this situation the SMCNF would be a better controller to implement.

APPENDIX A

ATTITUDE REPRESENTATIONS AND ROTATION MATRICES

When dealing with a subject such as inertial navigation, controlling spacecraft, or computer vision, it is common to have to deal with multiple reference frames and have to relate the position of one point in one frame to its position in another reference frame. Thus it is important to know how to rotate points from one frame to another efficiently and understand what limitations may be present in various methods. This Appendix will attempt to familiarize the reader with quaternions, rotations, attitude in space, and euler angles. Comparisons will be drawn between euler angles and quaternions to show their strengths and weaknesses. However it is assumed that the reader is already somewhat familiar with euler angles.

Fixed Angle Rotations

Typically when trying to relate a common point between different reference frames, a rotation matrix will be constructed. This matrix will allow easy transformation between frames.

$$P^b = R_a^b P^a \quad [A-1]$$

Here a point (P) is rotated from where it is originally defined, in reference frame a, to another reference frame b. Note that the rotation matrix R does not include any translation, but only rotation. Note also that R has the following properties:

Table A-1: Properties of a rotation matrix.

Rotation Matrix R
$R^{-1} = R^T$
$R_a^b = \begin{bmatrix} x_a & y_a & z_a \end{bmatrix} = \begin{bmatrix} x_b \\ y_b \\ z_b \end{bmatrix}$
$\ R\ = 1$
$\ column_i(R)\ = \ row_i(R)\ = 1$
$R_a^c = R_b^c R_a^b$

When a rigid body is rotated about a fixed inertial reference frame, it is referred to as a fixed angle rotation. These types of rotations are common in robotics where one is trying to determine the joint angles of a serial robot relative to some point in space. What is important to understand, is rotations of this type are made about the inertial axis which is inconvenient. A better method would be rotations about the body axes. These type of rotations are known as Euler Angles.

Euler Angles

Euler angles are typically thought of in terms of roll, pitch, and yaw angle about body axes. These terms are shown graphically below for a rigid body in space.

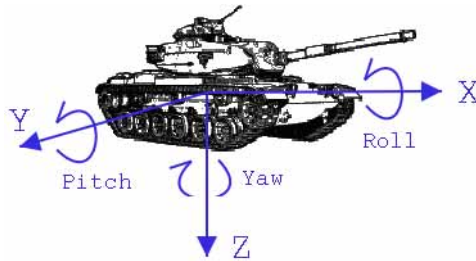


Figure A-1. Body reference frame attached to a rigid body. The x-axis points out the front of the vehicle.

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} roll \\ pitch \\ yaw \end{bmatrix} \quad [A-2]$$

There are two different groups of euler rotations out of the possible twelve. The two groups are distinguished by their singularity locations. Typically aerospace and navigational applications use the 1-2-3 rotation where this singularity only effects fighter aircraft who dive or climb at such steep angles. Some space applications, such as orbits around the Earth, use the 3-1-3 rotation sequence.

Table A-2: Comparison of the two major types of rotations, sequential and first and third axes rotation

	Singularities in Pitch	Possible Rotation Sequence
Type I	+/-90	1-2-3, 1-3-2, 2-1-3, 2-3-1, 3-1-2, 3-2-1
Type II	0/180	1-2-1, 1-3-1, 2-1-2, 2-3-2, 3-1-3, 3-2-3

In order to perform these operations on the rigid body, a rotation matrix (R) is used to find the new orientation of the spacecraft given the old orientation. Given below are the attitude matrices for rotations about the x, y, and z-axes.

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{bmatrix} \quad [A-3]$$

$$R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \quad [A-4]$$

$$R_z(\psi) = \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad [A-5]$$

Now, subsequent rotations about these primary axes can be accomplished by multiplying the matrices together. Thus, successive rotations about the z-axis, x-axis, and y-axis are given by:

$$R_{zyx}(\psi, \theta, \phi) = R_z(\psi) \cdot R_y(\theta) \cdot R_x(\phi) \quad [\text{A-6}]$$

Quaternions

Quaternions were invented by William Rowan Hamilton in 1843. Prior to his discovery, it was believed impossible that any algebra could violate the laws of commutativity for multiplication. His work introduced the idea of hyper-complex numbers. Here real numbers can be thought of as hyper-complex numbers with a rank of 1, ordinary complex numbers with a rank of 2, and quaternions with a rank of 4. Hamilton's crucial rule that made this possible:

$$i^2 = j^2 = k^2 = ijk = -1 \quad [\text{A-7}]$$

Hamilton supposedly developed this rule while on his way to a party. When he realized what the solution was, he took out his pocket knife and carved the answer into a wooden bridge. This rule would forever change mathematics as was known at the time. Now mathematicians could look at algebra where commutativity did not work. This is where Gibbs and others developed algebra of vector spaces, and quickly eclipsed Hamilton's work until recently.

Quaternions, also known as Euler symmetric parameters, are more mathematically efficient ways to compute rotations of rigid and non-rigid body systems than traditional methods involving standard rotational matrices or Euler angles. Quaternions have the advantage of few trigonometric functions needed to compute attitude. Also, there exists a product rule for successive rotations that greatly simplifies the math, thus reducing

processor computation time. Quaternions also hold the advantage of being able to interpolate between two quaternions (through a technique called spherical linear interpolation or slerp) without the danger of singularities, maintaining a constant velocity, and minimum distance travelled between points.

The major disadvantage of quaternions is the lack of intuitive physical meaning. Most people would understand where a point was if they were given [1 2 3]. However, few would comprehend where a point was if given the quaternion [1 2 3 4].

This section does not attempt to provide the extensive understanding needed to employ quaternions but rather a simple introduction. Further information can be found in Wertz [37] or Crane and Duffy [42].

Quaternion Algebra

The quaternion is composed of a scalar and a vector part. The scalar is a redundant element that prevents singularities from occurring since the four elements are all dependent upon each other.

$$q = \begin{bmatrix} q_x & q_y & q_z & q_r \end{bmatrix}^T = \begin{bmatrix} q_1 & q_2 & q_3 & q_4 \end{bmatrix}^T = \begin{bmatrix} \mathbf{v}, r \end{bmatrix} = ix + jy + kz + r^1 \quad [\text{A-8}]$$

where the \mathbf{v} stands for vector and the r is a scalar real. Given below is a table that will summarize the important mathematical operations of quaternions.

-
1. The order of the quaternion elements is not standardized. I have chosen the element order that NASA uses which is the imaginary part first then the real. Some authors put the real first then the imaginary which is similar to complex numbers. Thus before you use any equations that utilize quaternions, make sure to understand how the author is arranging the elements.

Table A-3: Quaternion Algebra Summary

Operation	Formula
Add / Subtract	$q \pm q' = [q_x \pm q'_x \ q_y \pm q'_y \ q_z \pm q'_z \ q_r \pm q'_r]^T$
Scalar Multiplication	$\alpha q = [\alpha q_x \ \alpha q_y \ \alpha q_z \ \alpha q_r]^T$
Norm	$\ q\ = \sqrt{q_x^2 + q_y^2 + q_z^2 + q_r^2}$
Quaternion Multiplication	$q' \cdot q = \begin{bmatrix} q_4 & q_3 & -q_2 & q_1 \\ -q_3 & q_4 & q_1 & q_2 \\ q_2 & -q_1 & q_4 & q_3 \\ -q_1 & -q_2 & -q_3 & q_4 \end{bmatrix} \cdot q'$ ^a
Conjugate	$q^* = [-q_x \ -q_y \ -q_z \ q_r]^T$
Inverse	$q^{-1} = \frac{q^*}{\ q\ }$

a. Notice that the order of the quaternions in the matrix multiplication have been reversed.

Multiplication is an important operation for quaternions, thus we will elaborate on it a little.

$$pq = r = r_4 + ir_1 + jr_2 + kr_3 \quad [\text{A-9}]$$

where

$$r_4 = p_4q_4 - p_1q_1 - p_2q_2 - p_3q_3 \quad [\text{A-10}]$$

$$r_1 = p_4q_1 + p_1q_4 + p_2q_3 - p_3q_2 \quad [\text{A-11}]$$

$$r_2 = p_4q_2 - p_1q_3 + p_2q_4 + p_3q_1 \quad [\text{A-12}]$$

$$r_3 = p_4q_3 + p_1q_2 - p_2q_1 + p_3q_4 \quad [\text{A-13}]$$

or rewritten in matrix form.

$$pq = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \end{bmatrix} = \begin{bmatrix} p_4 & -p_3 & p_2 & p_1 \\ p_3 & p_4 & -p_1 & p_2 \\ -p_2 & p_1 & p_4 & p_3 \\ -p_1 & -p_2 & -p_3 & p_4 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} q_4 & q_3 & -q_2 & q_1 \\ -q_3 & q_4 & q_1 & q_2 \\ q_2 & -q_1 & q_4 & q_3 \\ -q_1 & -q_2 & -q_3 & q_4 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix} \quad [\text{A-14}]$$

The last part is the same representation present in the table of quaternion operations above. Notice how only the signs change when we reverse the order of the multiplication in the matrix-vector form of the quaternion multiplication. This is because of the cross product relationship with the vector part: $q \times p = -p \times q$.

Rotations of Rigid Bodies in Space.

Quaternions are able to represent a unique rotation in space. To perform a rotation (ϕ) of a rigid body about an arbitrary moving/fixed axis (e) in space, the quaternion representation of this operation is:

$$q_{1-3} = e \cdot \sin\left(\frac{\phi}{2}\right) \quad q_4 = \cos\left(\frac{\phi}{2}\right) \quad [\text{A-15}]$$

$$e = \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix}^T \quad [\text{A-16}]$$

Notice that only one sine and one cosine function call is needed to calculate a quaternion, where euler would require three sine and three cosine function calls, one each for roll, pitch, and yaw. Since trigonometric function calls are computationally expensive, this is a great savings.

Rotation of a point in space with a quaternion is done as follows:

$$P^b = qP^a q^* \quad [\text{A-17}]$$

$$P^a = \begin{bmatrix} x & y & z & 0 \end{bmatrix}^T \quad [\text{A-18}]$$

Since the quaternion has four elements, the point P must have a zero appended on to it in place of the real part of the quaternion. Thus the point P values constitute the imaginary (vector) part of the quaternion. This rotation can also be done similar to euler and fixed angle rotations by creating a rotation matrix from the quaternions. The attitude matrix (A) or rotation matrix (R) for this is

$$A(q) = R = \begin{bmatrix} q_1^2 - q_2^2 - q_3^2 + q_4^2 & 2(q_1q_2 + q_3q_4) & 2(q_1q_3 - q_2q_4) \\ 2(q_1q_2 - q_3q_4) & -q_1^2 + q_2^2 - q_3^2 + q_4^2 & 2(q_2q_3 + q_1q_4) \\ 2(q_1q_3 + q_2q_4) & 2(q_2q_3 - q_1q_4) & -q_1^2 - q_2^2 + q_3^2 + q_4^2 \end{bmatrix} \quad [A-19]$$

Successive rotations can be accomplished by multiplying the attitude matrices together.

Hopefully it can be seen that quaternions are better than other Euler operations to determine attitude. Quaternions lack singularities due to one redundant element. They also lack the computationally intensive trigonometric functions, and contain a simplified way to determine successive rotations about an arbitrary axis.

Attitude Errors in Quaternions

Since quaternions can represent attitudes and not just rotations, it is important to be able to calculate the difference between where you want to point and where you are pointing. For example, say you had to control where a satellite was to point and you need an error term to feed into your control system.

$$q_E = q_S^{-1} q_T = \begin{bmatrix} q_{T4} & q_{T3} & -q_{T2} & q_{T1} \\ -q_{T3} & q_{T4} & q_{T1} & q_{T2} \\ q_{T2} & -q_{T1} & q_{T4} & q_{T3} \\ -q_{T1} & -q_{T2} & -q_{T3} & q_{T4} \end{bmatrix} \begin{bmatrix} -q_{S1} \\ -q_{S2} \\ -q_{S3} \\ p_{S4} \end{bmatrix} \quad [A-20]$$

where q_E is the quaternion error, q_S is the attitude of the satellite, and q_T is the target attitude. Note that the quaternions are in reversed order in the matrix and the spacecraft quaternion has the imaginary part negated due to the inversion.

Summary of Quaternions

1. Quaternions have singularities at angles of rotation of 180 degrees, but this is generally not a problem. Typically if an application encounters a rotation greater than 180 degrees, it is shorter to go in the other direction.
2. The quaternion is a compact method of representing a unique rotation with only four elements without the redundant information present in Euler rotation's nine elements.
3. Quaternions are difficult to visualize without extracting out the axis of rotation and rotation angle. Although the quaternion can be thought of as a sphere in 4D space, this does the majority of people little good, since they can not perceive a 4D surface.
4. The use of quaternions allows you to compute the shortest distance between two different attitudes. If lerp is used, the resulting velocity between the two attitudes will not be constant but more bell shaped. If slerp is used, the resulting velocity will be constant. This process is difficult using Euler angles, which contain many singularities that must be avoided and the solution is not guaranteed to be the shortest distance. Also the solutions that result from Euler angles are not uniformly distributed along the surface of a sphere. Thus it again is not always possible to transition from one orientation to another using the shortest distance, because these empty areas must be skirted around to arrive at the desired location.

APPENDIX B MODAL EQUATIONS

Motion of Multi-Degree of Freedom System

Looking at the simple multi-degree of freedom (MDOF) system in Figure B-1, we can write the equation of motion as

$$M\ddot{x} + Kx = f \quad [B-1]$$

Assuming $f = 0$, [B-1] will have the simple harmonic solution

$$x_n(t) = A_n e^{j\omega t} \quad [B-2]$$

where x_n is the displacement and A_n is the amplitude of the n th mass. Thus substituting [B-2] into [B-1] gives

$$-\lambda Ma + Ka = 0 \quad a = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_N \end{bmatrix} \quad \lambda = \omega^2 \quad [B-3]$$

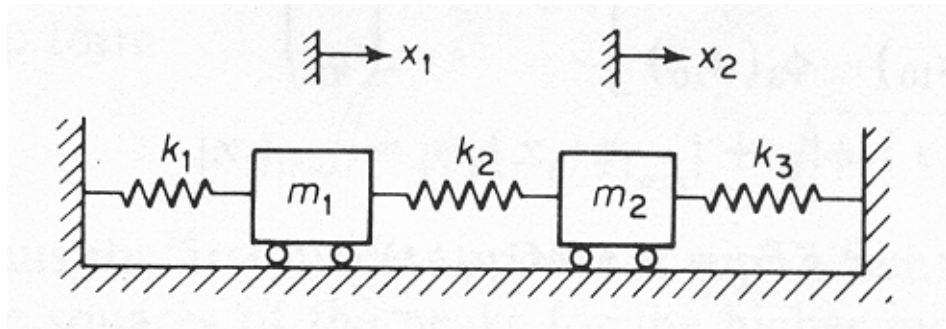


Figure B-1. Simple multi-degree of freedom system composed of mass and spring elements.

Now dividing through by the mass and simplifying the equation a little we have the characteristic equation.

$$\left| \lambda I - \frac{K}{M} \right| = 0 \quad [\text{B-4}]$$

This is assuming that $a \neq 0$. The roots of this equation are eigenvalues which provide the resonance frequency of the system. The roots must be real and positive if M and K are positive definite. Since the eigenvalues provide the resonance frequencies, the eigenvectors (ψ) provide the mode shapes of the system. Thus the nth mode shape of the system is

$$\psi_n = \begin{bmatrix} A_1^n \\ A_2^n \\ \dots \\ A_N^n \end{bmatrix} \quad [\text{B-5}]$$

The mode shapes are arbitrary to a constant value, thus the modal vector is typically normalized. Two common methods to scale the mode shape are by using the mass matrix or by setting the maximum value of the modal vector to one.

$$\psi^T M \psi = 1 \quad \max |\psi| = 1 \quad [\text{B-6}]$$

The mode shapes also follow the following property of orthogonality.

$$\psi^T M \psi = \delta_{nm} M = M_n \quad \psi^T K \psi = \delta_{nm} K = K_n \quad [\text{B-7}]$$

where δ is the Kronecker delta function which is defined as

$$\delta_{nm} = \begin{cases} 0, & n \neq m \\ 1, & n = m \end{cases} \quad [\text{B-8}]$$

The mode shapes form a basis which can be used to solve for x . To accomplish this, we will now define a modal matrix (Ψ) whose columns are made up of modal vectors. The solution to the system is now assumed to take the form

$$x(t) = \Psi q(t) \quad [\text{B-9}]$$

where $q(t)$ is a generalized coordinate or the modal displacement. Substituting [B-9] into [B-1] and pre-multiplying by the transpose of the of the modal matrix resulting in the equations of motion.

$$\Psi^T M \Psi \ddot{q} + \Psi^T K \Psi q = \Psi f \quad [\text{B-10}]$$

This can be simplified to:

$$M_n \ddot{q} + K_n q = \Psi f \quad [\text{B-11}]$$

This can further be manipulated by dividing the equation through by the mass matrix. This results in an equation with a matrix of natural frequencies and the modal force is now defined and inserted in the equation.

$$\ddot{q} + \omega_n^2 q = F_n \quad [\text{B-12}]$$

$$F_n = M_n^{-1} \Psi f \quad [\text{B-13}]$$

The initial position and velocity of the system are defined by:

$$q_n(0) = \Psi_n^T M x(0) \quad \dot{q}_n(0) = \Psi_n^T M \dot{x}(0) \quad [\text{B-14}]$$

Dampening

Throughout these derivations, we have assumed that there was no dampening in the system. The dampening coefficient matrix is assumed to be a linear combination of the mass and spring constant coefficient matrix.

$$M_n \ddot{q} + C_n \dot{q} + K_n q = \Psi f \quad [\text{B-15}]$$

$$C_n = \alpha M_n + \beta K_n \quad [\text{B-16}]$$

or represented another way:

$$\ddot{q} + C_n \dot{q} + \omega_n^2 q = F_n \quad [\text{B-17}]$$

$$C_n = \alpha + \beta \omega_n^2 \quad [\text{B-18}]$$

Simple Example

Thus for a system with two modes, the state space representation would be:

$$\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} C_1 & 0 \\ 0 & C_2 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} \omega_1^2 & 0 \\ 0 & \omega_2^2 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \end{bmatrix} \quad [\text{B-19}]$$

Again notice that the coefficient matrices are diagonal due to orthogonality of the modes in the system. Also note that the input forces for the nth mode only effects the nth mode. In order to derive this system, only the diagonal mass matrix, natural frequencies, and the normal modes need to be calculated.

APPENDIX C SIMULATION CODE

This Appendix will describe the satellite simulation, which was eventually written in the C++ programming language.

Simulation Code

Matlab

The simulation originally was created in Matlab to utilize the quick development environment. The main problem with Matlab, however, was the slow speed at which the simulations ran. Thus a decision was made to port the code over to C to gain the advantage of speed.

C Programming Language

All of the mathematical benefits of Matlab had to be created for use in C. The book *Numerical Recipes in C* [73] was a great aid in developing some of the vector and matrix operations. The functions written to perform the various operations were designed to be as efficient and fast as possible. Thus most mathematical function required two inputs and an output, so that processor time was not wasted making expensive calls to malloc to create a return value.

The C programming language produced fast simulations, however the dynamics equation code was difficult to read and follow due to the functional architecture of C. For example of how C's functional aspects obscure the mathematics, see the example below.

```
1  #include "mathlib.h"  
2
```

```

3  int main(void){
4      vector_t h_dot, w, h, tmp, u; // create vectors
5
6      initVector(h_dot,3); // initialize the vectors to size 3
7      ... // initialize remaining vectors
8
9      // Euler's equation:  $h\_dot = w \times h + u$ 
10     cross(w,h,tmp); // calc cross product and store in tmp vector
11     add(tmp,u,h_dot); // add the two vectors together and store in h_dot
12
13 }

```

Although Euler's equation is calculated properly using the two functions `cross()` and `add()`, where the first two arguments are the inputs and the third is the output, this is not as clear as Matlab code and therefor can lead to errors. Thus the code was moved from C to C++ which is an object oriented language where concepts such as vectors and matrices can exist.

C++ Programming Language

C++ provided cleaner mathematical code without sacrificing performance. This was due to C++'s object oriented nature and the ability to overload operators such as: `+`, `-`, `*`, and `/`. Basically the C code previously written was modified for C++ by following some of the examples in [72]. An example of the previous C code rewritten in C++ is shown below.

```

1  #include "mathlib.h"
2
3  int main(void){
4      cVector h_dot(3), w(3), h(3), u(3); // create vectors of size 3
5
6      // Euler's equation:  $h\_dot = w \times h + u$ 
7      h_dot = cross(w,h) + u;
8
9  }
10

```

Notice that the temporary vector `tmp` is no longer required. This is due to the fact that the math library developed here automatically creates and reuses temporary vectors and

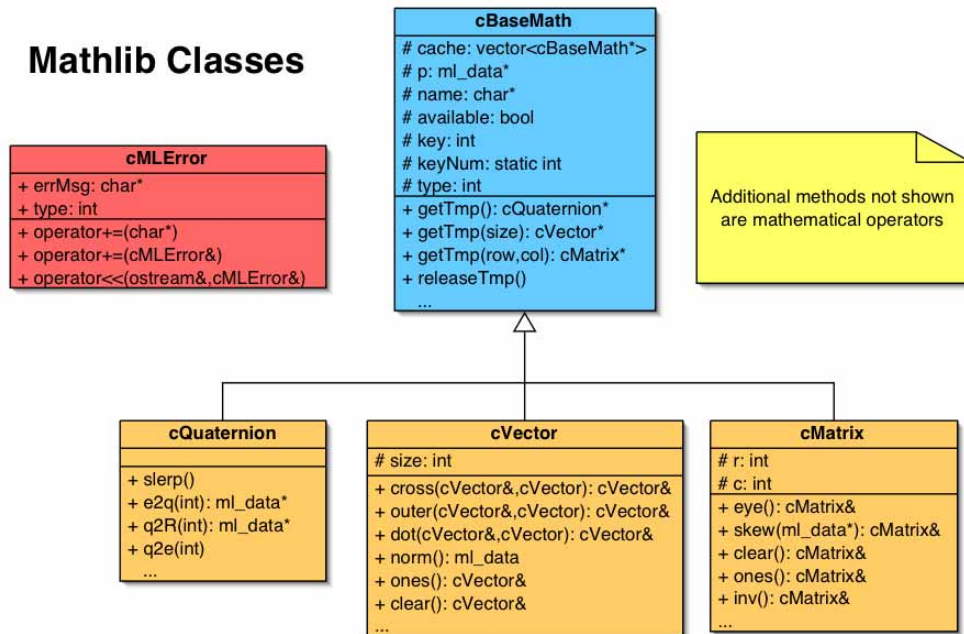


Figure C-1. UML diagram of Mathlib class which contains C++ objects for vectors, matrices, and quaternion used in this work.

matrices for mathematical operations. This means that the new operator¹ does not need to be called to return temporary values. This is accomplished by creating a cache of vectors, matrices, and quaternions that are need using an STL (standard template library) data structure called a vector². Further explanation of STL can be found in [75], which covers programing with the wide variety of data structures (i.e., linked lists, hash tables, trees, etc.) available in STL.

cMathlib Header File

This header file provides the foundation of the programming work. Here there are definitions for vectors (**cVector**), matrices (**cMatrix**), quaternions (**cQuaternion**), and error handling (**cMLError**).

-
1. The new operator is equivalent to malloc in C. For example, `double *a = new double` in C++ is basically the same as `double *a = (double*)malloc(sizeof(double))` in C.
 2. Note the STL vector is not a mathematical vector, but rather a dynamic array data structure.

The `cVector`, `cMatrix`, and `cQuaternion` are mathematical objects and are easy to comprehend. The `cMLError` or class Math Library Error is an object that is created when an error occurs. The C++ language has a method of error handling called catch and throw. Basically when an error occurs in a function, an error (in this case `cMLError`) is thrown by the offending piece of code. The code that called the function then catches the error and handles the problem appropriately. Further information on catch and throw can be found in Deitel and Deitel [74].

```

1  #ifndef CMATHLIB_H
2  #define CMATHLIB_H
3
4  //--- C++ ---
5  #include <iostream>
6  #include <vector>
7  #include <algorithm>
8
9  //--- C ---
10 #include <cmath>
11 #include <cstring>
12 #include <cstdlib>
13
14 //--- defines ---
15 #define ML_CHECK
16 #define ERROR_STRING_SIZE 255
17 #define ml_data double
18
19 #define R2D (180.0/M_PI)
20 #define D2R (M_PI/180.0)
21
22 class cBaseMath;
23 class cMatrix;
24 class cQuaternion;
25 class cVector;
26 template < class T > class cArray;
27
28 class cMLError {
29 public:
30     cMLError(int,char*);
31     ~cMLError(void);
32     friend std::ostream &operator<<(std::ostream&,cMLError&);
33     void operator+=(cMLError&);
34     void operator+=(char*);
35
36     enum {NONFATAL,FATAL};
37     char errMsg[255];

```



```

38  int type;
39  };
40
41
42  /*!
43   \warning NEVER pass a temp matrix, vector, or
44   quaternion to or from a function. A
45   temp should ALWAYS be passed to a user
46   created matrix, vector, or quaternion.
47   */
48  class cBaseMath {
49  public:
50      cBaseMath(void);
51      ~cBaseMath(void);
52      static cQuaternion* getTmp(void);
53      static cVector* getTmp(int);
54      template<class T> static cArray<T>* getTmpArray(int);
55      static cMatrix* getTmp(int,int);
56      void releaseTmp(void) const;
57      inline void setKey(int a){key = a;}
58      inline int getKey(){return key;}
59      void setName(char*);
60      char* getName(void){return name;}
61      inline bool valid(void){return p == NULL ? false : true;}
62      inline int getType(void){return type;}
63      ml_data get(const int i) const; // {return p[i];}
64      ml_data& set(const int i); // {return p[i];}
65
66      //protected:
67      enum {ML_VECTOR,ML_MATRIX,ML_QUATERNION,ML_ARRAY};
68      static std::vector<cBaseMath*> cache;
69      int key;
70      int type;
71      bool available;
72      ml_data *p;      // pointer to data
73      char *name;
74      static int keyNum;
75  };
76
77
78  template < class T >
79  cArray<T>* cBaseMath::getTmpArray(int size){
80      cBaseMath *b = NULL;
81      cArray<T> *a = NULL;
82      unsigned int index = 0;
83
84      for(index = 0; index<cache.size();index++){
85          b = (cBaseMath*) cache[index];
86          if ( b->type == cBaseMath::ML_ARRAY && b->available ){
87              a = (cArray<T>*) b;
88              if (a->getSize() >= size){
89                  a->available = false;
90                  //printf(" using array[%d](%d)\n",a->getKey(),size);
91                  break; // get out of for loop

```

```

92     }
93     else{
94         a = NULL;
95     }
96 }
97 }
98
99 if( a == NULL ){
100     a = new cArray<T>(size);
101     a->setKey(keyNum++);
102     a->available = false;
103     //printf(" + array[%d](%d)\n",a->getKey(),size);
104     cache.push_back(a);
105 }
106
107 memset(a->array,0,sizeof(T)*size);
108
109 return a;
110 }
111
112 //////////////////////////////////////
113
114 template < class T >
115 class cArray : public cBaseMath {
116 public:
117     cArray(int);
118     ~cArray(void);
119     inline int& operator[](int a){return array[a];}
120     inline int getSize(void){return size;}
121     //protected:
122     T *array;
123     int size;
124 };
125
126 template < class T >
127 cArray<T>::cArray(int s){
128     type = ML_ARRAY;
129     key = 0;
130     size = s;
131
132     array = NULL;
133     array = new T[size];
134
135     if(array == NULL){
136         throw cMLError(cMLError::FATAL,"could not create array");
137     }
138 }
139
140 template < class T >
141 cArray<T>::~~cArray(void){
142     delete array;
143     size = 0;
144 }
145

```

```

146
147
148 //////////////////////////////////////
149
150 class cVector : public cBaseMath {
151     friend cVector& operator+(const cVector&, const cVector&);
152     friend cVector& operator-(cVector&, cVector&);
153     friend cVector& operator-(cVector&);
154     friend cVector& operator*(cVector&, ml_data);
155     friend cVector& operator*(ml_data, cVector&);
156     friend cVector& operator/(cVector&, ml_data);
157     friend cVector& operator/=(cVector&,ml_data);
158     friend cVector& operator*=(cVector&,ml_data);
159     friend cVector& operator*=(ml_data,cVector&);
160     friend cVector& operator+=(cVector&, cVector&);
161     friend cVector& operator-=(cVector&, cVector&);
162     friend std::ostream& operator<<(std::ostream&,cVector&);
163     friend ml_data dot(cVector&,cVector&);
164     friend cVector& cross(cVector&,cVector&);
165     friend cMatrix& outer(cVector&,cVector&);
166
167 public:
168     cVector(int,char* =NULL,ml_data* =NULL);
169     cVector(void){size=0;name=NULL;p=NULL;};
170     cVector(cVector&);
171     ~cVector();
172     cVector& operator=(cVector&);
173     cVector& operator=(cVector*);
174     cVector& operator=(ml_data*);
175     inline bool operator==(cVector &a){return (size==a.size ? true : false);}
176     inline bool operator!=(cVector &a){return (size==a.size ? false : true);}
177
178     ml_data& operator()(int)const;
179     ml_data norm(void);
180     cVector& ones(void);
181
182     inline void set(ml_data *d){memcpy(p,d,sizeof(ml_data)*size);}
183     inline void set(ml_data a, ml_data b, ml_data c){p[0]=a;p[1]=b;p[2]=c;}
184     inline void clear(void){memset(p,0,sizeof(ml_data)*size);}
185     inline int getSize(void){return size;}
186     void resize(int);
187
188 //private:
189 int size;
190 };
191
192 class cMatrix : public cBaseMath {
193     friend cMatrix& operator+(cMatrix&,cMatrix&); //matrix addition
194     friend cMatrix& operator-(cMatrix&,cMatrix&); //matrix subtraction
195     friend cMatrix& operator-(cMatrix&); //negate a matrix
196     friend cMatrix& operator*(cMatrix&,cMatrix&); //matrix multiplication
197     friend cMatrix& operator*(ml_data,cMatrix&); // scalar multiplication
198     friend cMatrix& operator*(cMatrix&,ml_data); // scalar multiplication
199     friend cVector& operator*(cMatrix&,cVector&); //matrix addition

```

```

200 friend cMatrix& operator/(ml_data,cMatrix&); // scalar division
201 friend cMatrix& operator/=(cMatrix&,ml_data); // scalar division
202 friend cMatrix& operator+=(cMatrix&,cMatrix&); //matrix addition
203 friend cMatrix& operator-=(cMatrix&,cMatrix&); //matrix subtraction
204 friend cMatrix& operator*=(cMatrix&,cMatrix&); //matrix multiplication
205 friend cMatrix& skew(ml_data*,int=3);
206 friend cMatrix& eye(int);
207 friend cMatrix& zeros(int,int);
208 friend cMatrix& ones(int,int);
209
210 friend std::ostream &operator<<(std::ostream &s, cMatrix &mat); // print matrix
211
212 public:
213 cMatrix(int,int,char* =NULL,ml_data* =NULL);// constructor
214 cMatrix(void){r=c=0,name=NULL;p=NULL;};
215 ~cMatrix(); // destructor
216 cMatrix(const cMatrix &m);// copy constructor
217
218 cMatrix& eye(void);
219 cMatrix& ones(void);
220 cMatrix& diag(ml_data*);
221 cMatrix& inv(void);
222 cMatrix& trans(void);
223 cMatrix& skew(ml_data*);
224 ml_data trace(void);
225 void resize(int,int);
226 inline void reset(int a, int b, char *n=NULL){resize(a,b);setName(n);}
227
228 ml_data& operator()(int i, int j)const;// index
229 cMatrix& operator=(cMatrix&);
230 cMatrix& operator=(cMatrix*);
231 cMatrix& operator=(ml_data*);
232 cMatrix& fill(int,int,int,int,ml_data*);
233 inline bool operator==(cMatrix &a){return (r==a.r && c==a.c ? true : false);}
234 inline bool operator!=(cMatrix &a){return (r==a.r && c==a.c ? false : true);}
235
236 inline int getRow(void){return r;}
237 inline int getCol(void){return c;}
238 void setR(ml_data*,int);
239 void setC(ml_data*,int);
240 inline void set(cMatrix &m,int a, int b){set(m.p,a,b,m.r,m.c);}
241 inline void set(ml_data *d){memcpy(p,d,sizeof(ml_data)*r*c);}
242 inline void set(ml_data *d,int rr,int cc){set(d,rr,cc,r,c);}
243 void set(ml_data*,int,int,int,int);
244 inline void clear(void){memset(p,0,sizeof(ml_data)*r*c);}
245
246 //private:
247 int r,c; // index ranges
248 };
249
250
251 class cQuaternion : public cBaseMath {
252 friend cQuaternion& operator*(cQuaternion&,cQuaternion&);
253 friend inline cQuaternion& inv(cQuaternion &q){return q.inv();}

```

```

254 friend cQuaternion& slerp(cQuaternion&,cQuaternion&,int);
255 friend std::ostream &operator<<(std::ostream&, cQuaternion&); // print quaternion
256 //friend void operator=(ml_data*,cQuaternion&);
257
258 public:
259 cQuaternion(char*);
260 cQuaternion(const cQuaternion&); // copy constructor
261
262 // set data
263 inline void set(ml_data a, ml_data b, ml_data c, ml_data d){ p[0]=a; p[1]=b; p[2]=c; p[3]=d;
264 }
265 inline void set(ml_data *a){set(a[0],a[1],a[2],a[3]);}
266 inline ml_data& operator[](int i){return p[i];} // 0 based
267 inline ml_data& operator()(int i){return p[i-1];} // 1 based
268 cQuaternion operator*=(cQuaternion&);
269 void operator=(cQuaternion&);
270 void operator=(cQuaternion*);
271 void operator=(ml_data*);
272
273 // get data
274 inline ml_data& x(void){return p[0];}
275 inline ml_data& y(void){return p[1];}
276 inline ml_data& z(void){return p[2];}
277 inline ml_data& r(void){return p[3];}
278 inline ml_data* getImg(void){return p;}
279 inline ml_data getReal(void){return p[3];}
280 ml_data* getAxis(void);
281 ml_data getAngle(void);
282 ml_data norm(void);
283 void copy(ml_data*);
284
285 // misc
286 cQuaternion& inv(void);
287 cQuaternion& conj(void);
288 void normalize(void);
289 void clear(void);
290 void error(cQuaternion&,cQuaternion&);
291
292 // conversions
293 void e2q(ml_data,ml_data,ml_data,int);
294 inline void e2q(ml_data *e,int t){e2q(e[0],e[1],e[2],t);}
295 ml_data* q2e(int=7);
296 ml_data* q2R(int);
297
298 private:
299 };
300
301
302 #endif
303
304
305

```

cRK4 Header File

This code was responsible for integrating the equations of motion using a Rung-Kutta integration method. Notice that it utilizes the cVectors from cMathlib shown above.

```

1  #ifndef KEVINS_RUNGE_KUTTA
2  #define KEVINS_RUNGE_KUTTA
3
4  #include <cmath>
5  #include "cMathlib.h"
6
7
8  class cRK4 {
9  public:
10         cRK4(cVector& (*model)(cVector&,cVector&,cVector&),
11             ml_data dt,
12             int x_size,
13             int u_size
14             );
15
16         ~cRK4(void);
17         cVector& integrate(void);
18
19         //protected:
20         ml_data time,dt;
21         cVector k1x,k2x,k3x,k4x,x,xx,y,u,dist;
22         cVector& (*model)(cVector&,cVector&,cVector&);
23     };
24
25
26 #endif
27
28

```

REFERENCES

- [1] Kang, W., Sparks, A., and Banda, S., "Coordinated Control of Multisatellite Systems," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 3, March-April 2001, pp. 360-368.
- [2] Mesbahi, M. and Hadaegh, F., "Formation Flying Control of Multiple Spacecraft via Graphs, Matrix Inequalities, and Switching," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 3, March-April 2001, pp. 369-377.
- [3] Chichka, D., "Satellite Clusters with Constant Apparent Distribution," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 1, January-February 2001, pp. 117-122.
- [4] Kapila, V., Sparks, A. G., Buffington, J., and Yan, Q., "Spacecraft Formation Flying Dynamics and Control," *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 3, May-June 2000, pp. 561-564.
- [5] Queiroz, M. S., Kapila, V., and Yan, Q., "Adaptive Nonlinear Control of Multiple Spacecraft Formation Flying," *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 3, May-June 2000, pp. 385-390.
- [6] Ocampo, C., and Rosbrough, G., "Multiple-Spacecraft Orbit Transfer Problem: Weak-Booster and Strong-Booster Cases," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 6, November-December 1999, pp. 897-904.
- [7] Ocamp, C., and Rosborough, W., "Multiple-Spacecraft Orbit-Transfer Problem: The No-Booster Case," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 5, September-October 1999, pp. 650-657.
- [8] Wang, P., Hadaegh, F., and Lau, K., "Synchronized Formation Rotation and Attitude Control of Multiple Free-Flying Spacecraft," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 1, January-February 1999, pp. 28-35.
- [9] Sidi, M., *Spacecraft Dynamics and Control*, Cambridge University Press, New York, 1997.
- [10] Agrawal, B., "Dynamic Characteristics of Liquid Motion in Partially Filled Tanks of a Spinning Spacecraft," *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 4, July-August 1993, pp. 636-640.

- [11] Genevaux, J. and Lu, D., "On Sloshing in a Container with Moving Wall," *Journal of Fluids and Structures*, Vol. 14, 2000, pp. 275-278.
- [12] Anderson, J., Turan, O., and Semercigil, S., "Experiments to Control Sloshing in Cylindrical Containers," *Journal of Sound and Vibration*, Vol. 240, No. 2, 2001, pp. 398-404.
- [13] Anderson, J., Turan, O., and Semercigil, S., "A Standing-wave-type Sloshing Absorber to Control Transient Oscillations," *Journal of Sound Vibrations*, Vol. 232, No. 5, 2000, pp. 839-856.
- [14] Cho, S. and McClaroach, N., "Feedback Control of a Space Vehicle with Unactuated Fuel Slosh Dynamics," *AIAA Guidance, Navigation, and Control Conference*, Denver, CO, August 14-17, 2000.
- [15] Kuang, J. and Leung, A. Y., " H_∞ Feedback for Attitude Control of Liquid-Filled Spacecraft," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 1, January-February 2001, pp. 46-55.
- [16] Grundelius, M. and Bernhardsson, B., "Control of Liquid Slosh in a Industrial Packing Machine," *IEEE Conference on Control Applications -- Proceedings*, Vol. 2, Kohala Coast, HI, 1999, pp. 1654-1659.
- [17] Wisniewski, R. and Blanke, M., "Fully Magnetic Attitude Control for Spacecraft Subject to Gravity Gradient," *Automatica*, Vol. 35, 1999, pp. 1201-1214.
- [18] Chen, X., Steyn, W. H., Hodgart, S., and Hashida, Y., "Optimal Combined Reaction-Wheel Momentum Management for Earth-Pointing Satellites," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 4, July-August 1999, pp. 543-550.
- [19] Agrawal, B. N., "Dynamic Characteristics of Liquid Motion in Partially Filled Tanks of a Spinning Spacecraft," *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 4, July-August 1993, pp. 636-640.
- [20] Hoskins, W., Cassady, R., Campbell, M., and Rayburn, C., "A Micro Pulsed Plasma Thruster (PPT) for the "Dawgstar" Spacecraft," *IEEE Aerospace Conference Proceedings*, Vol. 4, Big Sky, MT, 2000, pp. 7-13.
- [21] Wilson, J., "Major New Thrust for MEMS Engines," *Aerospace America*, February, 2003, pp.34-38.
- [22] Wie, B., *Space Vehicle Dynamics and Control*, AIAA Education Series, AIAA Inc., Reston, 1998.

- [23] W. Singhose, K. Bohlke, and W. Seering, "Fuel-Efficient Shaped Command Profiles for Flexible Spacecraft," AIAA Guidance, Navigation, and Control Conference. Baltimore, MD, 1995.
- [24] Ooten, E. and Singhose, W., "Command Generation for Flexible Systems using Numerator Dynamics and Sliding Mode Control", Master's Thesis, Georgia Institute of Technology, 2000.
- [25] Wie, B. and Liu, Q., "Classical and Robust H_∞ Controller Redesign for the Hubble Space Telescope," Journal of Guidance, Control, and Dynamics, Vol. 16, No. 6, November-December 1993, pp. 1069-1077.
- [26] Thorten, E., *Thermal Structures for Aerospace Applications*, AIAA Education Series, AIAA Inc., Reston, 1996.
- [27] Walchko, K. and Mason, P. A. C., "Development of a Generic Hybrid Fuzzy Controller," Artificial Neural Networks in Engineering, St. Louis, Ms, 1-4 Nov.1998.
- [28] Petroff, N., Mason, P., Walchko, K., "Numerical Stability Analysis of a Fuzzy Controller," Artificial Neural Networks in Engineering, St. Louis, Ms, 1-4 Nov.1998.
- [29] Vreeberg, J., "Acceleration Measurements on Slosat FLEVO for Liquid Force Location Determination," 4th ESA International Conference on Spaceflight Guidance, Navigation, and Control Systems, Noordwijk, Netherlands, 18-21 October 1999.
- [30] El-Sayad, M., Hanna, S., and Ibrahim, R., "Parametric Excitation of Nonlinear Elastic Systems Involving Hydrodynamic Sloshing Impact," Nonlinear Dynamics, Vol. 18, 1999, pp. 25-50.
- [31] Hughes, P., *Spacecraft Attitude Dynamics*, John Wiley & Sons, New York, 1986.
- [32] Greenburg, M., *Advanced Engineering Mathematics*, Prentice Hall, New York, 1998, pp. 1033-1034.
- [33] Kreyszig, E., *Advanced Engineering Mathematics*, John Wiley & Sons, New York, 1999, pp. 598-599.
- [34] Meirovitch, L., *Analytical Methods in Vibrations*, Collier-MacMillan Limited, New York, 1967.
- [35] Zill, D. and Cullen, M., *Advanced Engineering Mathematics*, Jones and Bartlett Publishers International, New York, 2000, pp. 703-704.
- [36] McConnell, K., *Vibration Testing, Theory and Practice*, John Wiley and Sons, New York, 1995.

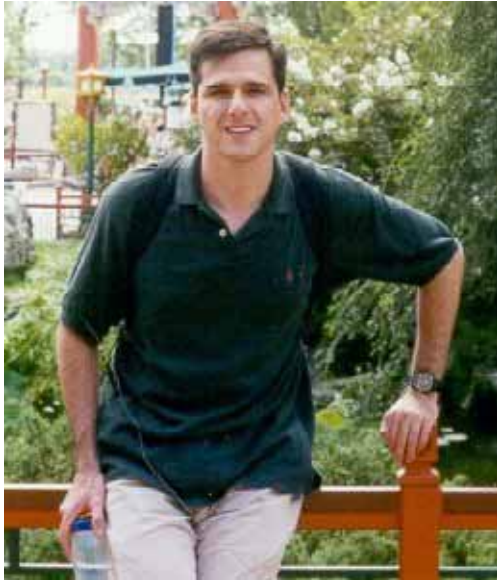
- [37] Thompson, W., *Theory of Vibrations with Applications*, Prentice Hall, New York, 1998.
- [38] Wertz, J., *Spacecraft Attitude Determination and Control*, D. Reidel Publishing Co., Boston, 1978.
- [39] Meirovitch, L., *Methods of Analytical Dynamics*, McGraw-Hill Book Co., New York, 1970.
- [40] Miller, A., Gray, G., and Massoleni, A., "Nonlinear Spacecraft Dynamics with a Flexible Appendage, Dampening, and Moving Internal Submasses," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 3, May-June 2001, pp. 605-615.
- [41] Wie, B., and Barba, P., "Quaternion Feedback for Spacecraft Large Angle Maneuvers," *Journal of Guidance, Control and Dynamics*, Vol. 8, No. 3, May-June 1985, pp. 360-365.
- [42] Slotine, J. and Li, W., *Applied Nonlinear Control*, Pearson Education, New York, 1990.
- [43] DeCarlo, R., Zak, S., and Mathews, G., "Variable Structure Control of Nonlinear Multivariable Systems: A Tutorial," *Proceeding of the IEEE*, Vol. 76, No. 3, March 1988, pp. 212-232.
- [44] Zhou, D., Mu, C., and Xu, W., "Adaptive Sliding-Mode Guidance of a Homing Missile," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 4, July-August 1999, pp. 589-594.
- [45] Lewis, A. S. and Sinha, A., "Sliding Mode Control of Mechanical Systems with Bounded Disturbances via Output Feedback," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 2, March-April 1999, pp. 235-240.
- [46] Erbatur, K., Kaynak, O., Sabanovic, A., and Rudas, I., "Fuzzy Parameter Adaptation for a Sliding Mode Controller as Applied to the Control of an Articulated Arm," *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, Albuquerque, NM, April 1997, pp. 817-822.
- [47] Hsu, F. and Fu, L., "Nonlinear Control of Robot Manipulators Using Adaptive Fuzzy Sliding Mode Control," *International Conference on Intelligent Robots and Systems*, Vol. 1, Pittsburgh, PA, August 5 - 9, 1995, pp. 156 - 161.
- [48] Bekit, B., Whidborne, J., and Seneviratne, L., "Fuzzy Sliding Mode Control for a Robot Manipulator," *The 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, Monterey, CA, July 10-11, 1997, pp. 320-325.

- [49] Kiriazov, P., Kreuzer, E., and Pinto, F. C., "Robust Feedback Stabilization of Underwater Robotic Vehicles," *Robotics and Autonomous Systems*, Vol. 21, 1997, pp. 415-423.
- [50] Song, G., Longman, R., and Mukherjee, R., "Integrated Sliding-Mode Adaptive-Robust Control," *IEE Proceedings of Control Theory Applications*, Vol. 146, No. 4, July 1999, pp.341-347.
- [51] Conte, G. and Serrani, A., "Robust Control of a Remotely Operated Underwater Vehicle," *Automatica*, Vol. 34, No. 2, 1998, pp. 193-198.
- [52] Alessandri, A., Hawkinson, T., Healey, A., and Veruggio, G., "Robust Model-Based Fault Diagnosis for Unmanned Underwater Vehicles Using Sliding Mode-Observers," 11th International Symposium on Unmanned Untethered Submersible Technology, Autonomous Undersea Systems Institute, Durham, NH, Aug. 22-25, 1999, pp. 1-8.
- [53] Livneh, R. and Wie, B., "Asymmetric Body Spinning Motion with Energy Dissipation and Constant Body-Fixed Torques," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 2, March-April 1999, pp. 322-328.
- [54] Tsiotras, P., Haijun, S., and Hall, C., "Satellite Attitude Control and Power Tracking with Energy/Momentum Wheels," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 1, January-February 2001, pp. 23-35.
- [55] Collins, E. G. and Song, T., "Robust H_∞ Estimation and Fault Detection of Uncertain Dynamic Systems," *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 5, September-October 2000, pp. 857-864.
- [56] Fowell, R. A., Milford, R. I., and Yocum, J. F., "Spacecraft Spin Stabilization Using a Transverse Wheel for Any Inertia Ratio," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 6, November-December 1999, pp. 768-775.
- [57] Kumar, K. K., and Nishita, K., "Robustness Analysis of Neural Networks with Application to System Identification," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 5, September-October 1999, pp. 695-701.
- [58] Shen, H. and Tsiotras, P., "Time-Optimal Control of Axisymmetric Rigid Spacecraft Using Two Controls," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 5, September-October 1999, pp. 682-694.
- [59] Crassidis, J. L., "Robust Control of Nonlinear Systems Using Model-Error Control Synthesis," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 4, July-August 1999, pp. 595-601.

- [60] Lo, S. and Chen, Y., "Smooth Sliding-Mode Control for Spacecraft Attitude Tracking Maneuvers," *Journal of Guidance, Control, and Dynamics*, Vol. 18, No. 6, November-December 1995, pp. 1345-1348.
- [61] Boskovic, J. D., Li, S., and Mehra, R. K., "Robust Adaptive Variable Structure Control of Spacecraft Under Control Input Saturation," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 1, January-February 2001, pp. 14-22.
- [62] Boskovic, J. D., Li, S., and Mehra, R. K., "Robust Adaptive Variable Structure Control of Spacecraft Under Control Input Saturation," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 1, January-February 2001, pp. 14-22.
- [63] Crassidis, J. L., Vadali, S. R., and Markley, F. L., "Optimal Variable-Structure Control Tracking of Spacecraft Maneuvers," *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 3, May-June 2000, pp. 564-566.
- [64] Coleman, C. P. and Godbole, D., "A Comparison of Robustness: Fuzzy Logic, PID, & Sliding Mode Control," *Proceedings of the 3rd IEEE International Conference on Fuzzy Systems*, pp. 1654-1560.
- [65] Yousefi-Koma, A. and Vukovick, G., "Vibration Suppression of Flexible Beams with Bonded Piezotransducers Using Wave-Absorbing Controllers," *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 2, March-April 2000, pp. 347-354.
- [66] Reitz, R., *Nonlinear Feedback Control of a Slewing Flexible Structure*, Master's Thesis, State University of New York at Buffalo, 1992.
- [67] Reitz, R., *Dynamics and Control of Thermally Induced Vibrations*, Ph.D. Dissertation, State University of New York at Buffalo, 1996.
- [68] Johnston, J. and Thorten E., "Thermally Induced Dynamics of Satellite Solar Panels," *Journal of Spacecraft and Rockets*, Vol. 37, No. 5, September-October, 2000, pp. 604-613.
- [69] Incropera, F. and DeWitt, D., *Fundamentals of Heat and Mass Transfer, 4th Ed.*, John Wiley & Sons, New York, 1996.
- [70] Brown, R. and Hwang, P., *Introduction to Random Signals and Applied Kalman Filtering*, John Wiley & Sons, New York, 1997.
- [71] Tonque, B., *Principles of Vibrations*, Oxford University Press, New York, 1996.
- [72] Capper, D., *Introducing C++ for Scientists, Engineers, and Mathematicians*, Springer-Verlag, New York, 1994.

- [73] Press, H., Teukolsky, S., Vetterling, W., and Flannery, B., *Numerical Recipes in C, 2nd Ed.*, Cambridge University Press, New York, 1992.
- [74] Deitel, H. and Deitel, P., *C++ How to Program*, 2nd Ed., Prentice Hall, Upper Saddle River, New Jersey, 1998.
- [75] Josuttis, N., *The C++ Standard Template Library, A Tutorial and Reference*, Addison-Wesley, New York, 1999.

BIOGRAPHICAL SKETCH



Kevin J. Walchko was born on 1 November 1972 in Sarasota, FL, and attended the University of Florida starting in the fall of 1991. After several years, a B.S. degree in mechanical engineering was earned in 1997. His college was paid for through the Montgomery G.I. Bill during his 7 years of service in the Florida Army National Guard as a Light Cavalry Scout. During his military career, Kevin participated in humanitarian relief for hurricane Andrew and Opal.

Kevin next received a master's degree in spring of 1999 for his work with NASA and satellite attitudes control. Most of the research Kevin conducted during this time was in controls and dynamics. Specifically, the research involved intelligent controls such as fuzzy logic and neural networks. Continuing his work with NASA, Kevin stayed at the University of Florida for a Ph.D. in mechanical engineering awarded in the spring of 2003.

Kevin has also conducted extensive research in robotics and autonomous vehicles. He has been a main member of the autonomous submarine team, Subjugator, for several years. Working closely with several members of the Machine Intelligence Laboratory in

the Electrical and Computer Engineering Department at the University of Florida, he concurrently completed a master's degree in electrical engineering during the summer of 2002 which involved low cost inertial navigation for autonomous vehicles.

During his time in graduate school, Kevin also fell in love and finally married Nina C. Massie on 20 October 2001. Their first son, Michael Jacob Walchko, was born on 20 January 2003.

Kevin's future plans are to return to the military by joining the U.S. Air Force as a developmental engineer. Here he will conduct research into new technologies and supervise research projects for the Air Force.