2007-12

Distributed autonomous control of multiple spacecraft during close proximity operations

McCamish, Shawn B.

Monterey, California. Naval Postgraduate School, 2007.

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

DISSERTATION

DISTRIBUTED AUTONOMOUS CONTROL OF MULTIPLE SPACECRAFT DURING CLOSE PROXIMITY OPERATIONS

by

Shawn B. McCamish

December 2007

Dissertation Supervisors: Xiaoping Yun
Marcello Romano

Approved for public release; distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** December 2007	**3. REPORT TYPE AND DATES COVERED** Dissertation	
4. TITLE AND SUBTITLE Distributed Autonomous Control of Multiple Spacecraft During Close Proximity Operations			**5. FUNDING NUMBERS**
6. AUTHOR(S) Shawn B. McCamish			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200 words)

This dissertation reports the development of an autonomous distributed control algorithm for multiple spacecraft during close proximity operations, including rendezvous and docking. The proposed control algorithm combines the efficiency of Linear Quadratic Regulator (LQR) and robust collision avoidance capability of Artificial Potential Function method (APF). The LQR control effort serves as attractive force toward goal positions, while the APF-based repulsive functions provide collision avoidance for obstacles. The combination of the LQR and APF multiple spacecraft close proximity control algorithms yielded promising results as demonstrated by the simulations reported in this dissertation.

The multiple spacecraft close proximity control algorithm was developed, refined, and thoroughly simulated using high fidelity six DOF spacecraft models. A versatile multiple spacecraft model validation and simulation visualization technique using a MATLAB-STK (Satellite Tool Kit) interface was created to propagate spacecraft models, compare against STK generated ephemeris, and animate for analysis. The MATLAB-STK interface efficacy was demonstrated during the evaluation and analysis of the innovative control algorithm. Additionally, the LQR/APF multiple spacecraft control algorithm was thoroughly analyzed via Monte-Carlo simulations to validate its stability and robustness. Finally, the LQR/APF multiple spacecraft control algorithm was evaluated by virtual hardware-in-the-loop implementation at the NPS Spacecraft Robotics Laboratory.

14. SUBJECT TERMS APF, LQR, control, multiple spacecraft, rendezvous, docking, assembly, maneuver, close proximity operations, simulation, STK, virtual hardware-in-the-loop.			**15. NUMBER OF PAGES** 281
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited.

DISTRIBUTED AUTONOMOUS CONTROL OF MULTIPLE SPACECRAFT DURING CLOSE PROXIMITY OPERATIONS

Shawn B. McCamish
Major, United States Air Force
B.S., Electrical Engineering, Iowa State University, 1994
M.S., Electrical Engineering, Air Force Institute of Technology, 1995

Submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 2007

Author: _____
Shawn B. McCamish

Approved by:

_____	_____
Xiaoping Yun	Marcello Romano
Professor of ECE	Assistant Professor of MAE
Dissertation Committee Chair	Dissertation Co-Advisor
_____	_____
Roberto Cristi	Tri Ha
Professor of ECE	Professor of ECE
_____	
Ravi Vaidyanathan	
Assistant Professor of SE	

Approved by: _____
Jeffrey Knorr, Chair, Department of Electrical and Computer Engineering

Approved by: _____
Julie Filizetti, Associate Provost for Academic Affairs

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This research contributes to multiple spacecraft control by developing an autonomous distributed control algorithm for close proximity operations of multiple spacecraft systems, including rendezvous and docking scenarios. The proposed control algorithm combines the efficiency of the Linear Quadratic Regulator (LQR) and the robust collision avoidance capability of the Artificial Potential Function (APF) method. The LQR control effort serves as the attractive force toward goal positions, while the APF-based repulsive functions provide collision avoidance for both fixed and moving obstacles. The combination of the LQR and APF control logics, referred to as the LQR/APF control algorithm, yielded promising results as demonstrated by the numerous multiple spacecraft maneuver simulations reported in this dissertation.

In order to validate the proposed control approach, a multiple spacecraft model validation and visualization technique was developed using a versatile MATLAB-Satellite Toll Kit (STK) interface to propagate the spacecraft models, compare against STK generated ephemeris, and animate for analysis. The MATLAB-STK interface efficacy was demonstrated during the evaluation and analysis of the innovative LQR/APF multiple spacecraft control algorithm.

The LQR/APF multiple spacecraft close proximity control algorithm was developed, refined, and thoroughly simulated using high fidelity six Degree of Freedom (DOF) spacecraft models. In order to evaluate the stability and robustness of the control approach a Monte-Carlo simulations set was run. The LQR/APF control algorithm was further evaluated by virtual hardware-in-the-loop implementation at the NPS Spacecraft Robotics Laboratory. The laboratory hosts the Autonomous Docking and Spacecraft Servicing testbed which allows for on-the-ground testing of close proximity multiple spacecraft control concepts.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACRONYMS

2D	Two-Dimensional
3D	Three-Dimensional
ACAT	Advanced Close Approach Tool
ADCS	Attitude Determination Control Subsystem
AFRL	Air Force Research Laboratory
AGI	Analytical Graphics Incorporated
AMPHIS	Autonomous Multi-agent Physically Interacting Spacecraft
AOCS	Attitude and Orbital Control Subsystem
APF	Artificial Potential Field
API	Application Program Interface
ARTIMIS	Autonomous Rendezvous and rapid Turnout Experiment Maneuverable Inspection Satellite
ATV	Automated Transfer Vehicle
AUDUSS	Autonomous Docking and Servicing Spacecraft
BIBO	Bounded-Input Bounded-Output
BSC	Base Sensor Control
COESA	Committee on Extension to the Standard Atmosphere
COM	Center of Mass
DARPA	Defense Advanced Research Projects Agency
DART	Demonstration for Autonomous Rendezvous Technology
DCM	Direction Cosine Matrix
DoD	Department of Defense
DOF	Degree of Freedom
ECI	Earth Centered Inertial Coordinate System
EGM	Earth Gravity Model
ESA	European Space Agency
FOV	Field-of-View
FREND	Front-end Robotics Enabling Near-term Demonstration

GEO	Geosynchronous Orbits
GG	Gravitational Gradient
GPS	Global Positioning System
GSFT	Ground-Based Satellite frame Testing
GUI	Graphical User Interface
HPOP	High Precision Orbital Propagator
ISS	International Space Station
LEO	Low Earth Orbits
LQG	Linear Quadratic Gaussian
LQR	Linear Quadratic Regulator
MED	Momentum Exchange Devices
MEO	Middle Earth Orbits
MIT	Massachusetts Institute of Technology
NASA	National Aeronautics and Space Administration
NGC	Navigation, Guidance, and Control
NPS	Naval Postgraduate School
NRL	Naval Research Laboratory
RSSI	Received Signal Strength Indicator
RW	Reaction Wheel
SPHERES	Synchronized Position Hold Engage Reorient Experimental Satellites
SRL	Spacecraft Robotics Laboratory
STK	Satellite Tool Kit
STS	Space Transportation System
SUMO	Spacecraft for the Universal Modification of Orbits
TCP/IP	Transmission Control Protocol and Internet Protocol
TICS	Tiny, Independent, Coordinating Spacecraft
TORU	Telerobotically Operated Rendezvous Unit
UDP	Universal or User Datagram Protocol
UT1	Universal Time

VHIL	Virtual Hardware-In-the-Loop
VO	STK Visual Option VO Commands
WGS	World Geodetic System
XSS	Experimental Satellite Systems

THIS PAGE INTENTIONALLY LEFT BLANK

Nomenclature

A, B, C, D	State Space Matrix
A_s	Spacecraft's Cross-Sectional Area Normal to its Velocity Vector
a	Orbital Semi-Major Axis
a	Acceleration
$\vec{a}_{3body}$	Acceleration due to Third Body Gravitational Forces
$\vec{a}_{co}$	Acceleration of the Chase Spacecraft toward Obstacle
$\vec{a}_{drag}$	Acceleration due to Atmospheric Drag
$\vec{a}_g$	Desire Acceleration of Chase Spacecraft toward Goal
$\vec{a}_{LQR}$	LQR Based Acceleration of Chase Spacecraft toward Goal
$a_{\max}$	Maximum Acceleration Allowed by Actuation
$\vec{a}_{thrust}$	Acceleration due to Thrust
$\vec{a}_{total}$	Desired Acceleration due Goal and Obstacle APF
B_B	Magnetic Field with respect to Spacecraft Body
b_g	Goal Velocity Decay Shaping function
C_{DCM}	Coordinate Transformation Matrix or Direction Cosine Matrix
C_m	LQR Velocity Additive Gain Magnitude Shaping Function
C_v	LQR Velocity Additive Gain for Collision Avoidance
c_1	Direction Cosine of Radius Vector in Spacecraft's Body Frame
c_D	Drag Coefficient
D_{stop}	Minimum Stopping Distance
D_o	Obstacle Region of Influence
d_a	Acceleration Decay Constant
d_e	LQR Additive Velocity Gain Constant
d_g	Velocity Decay Constant
d_o	Obstacle Region of Influence Constant
d_R	Velocity Ramping Constant

e	Eccentricity
$\vec{e}_{axis}$	Euler axis
eig	Eigenvalue
F	Force Acting Upon an Object
F_g	Force of Earth's Gravitation Field acting upon Spacecraft
F_{thrust}	Force of Spacecraft's Motor Thrust
G	Universal Gravitational Constant
g	Earth's Gravitational Acceleration
g_c	Acceleration Force on Chase Spacecraft
$\vec{g}_k$	Gradient Vector
g_t	Acceleration Force on Target Spacecraft
H	Angular Momentum
h	Angular Momentum of the Momentum Exchange Device
h_{mw}	Angular Momentum of Momentum Wheel
h_{RW}	Angular Momentum of Reaction Wheel
I_{sp}	Specific Impulse
i	Inclination
J	Inertia
$J(\vec{x})$	Jacobian
J_{LQR}	LQR Cost Function
J2	Second-Order Earth Potential Zonal Harmonic
J3	Third-Order Earth Potential Zonal Harmonic
J4	Fourth-Order Earth Potential Zonal Harmonic
K_{LQR}	LQR State Feedback Gain
k_a	Acceleration Shaping Function
k_d	Quaternion Feedback Derivative Gain
k_g	Velocity Shaping Function Based on Goal Potential
k_{mt}	Reaction Wheel Momentum Gain
k_o	Velocity Shaping Function Based on Obstacle Potential

k_p	Quaternion Feedback Proportional Gain
k_Q	LQR Velocity State Gain
k_R	Relative Velocity Ramping Function
k_s	Safety Shaping Function
k_v	Velocity Shaping Function
L_o	Exterior Surface of Obstacle
m	Mass
m_e	Mass of Earth
m_p	Mass of Propellant
m_s	Mass of Spacecraft
M	Magnetic Dipole Moment
N	LQR Coupling Gain Matrix
P, Q, W	Perifocal Coordinates
P_T	Period of Spacecraft Orbit
$\vec{p}_k$	Search Direction
q	Quaternion
$\vec{q}_E$	Quaternion Attitude Error
Q	LQR State Gain Matrix
Q_c	State Space Controllability Matrix
R, S, W	Relative Spacecraft Coordinates
R	LQR Control Effort Gain Matrix
R_e	Radius of the Earth
R_o	State Space Observability Matrix
r	Euclidean (2-norm) Distance
$\vec{r}$	Relative Position Vector
$\vec{r}_{em}$	Relative Distance from Earth to Moon
$\vec{r}_{eS}$	Relative Distance from Earth to Sun
$\vec{r}_{sm}$	Relative Distance from Spacecraft to Moon

$\vec{r}_{sS}$	Relative Distance from Spacecraft to Sun
r_a	Orbital Radius of Apogee
r_p	Orbital Radius of Perigee
r_t	Orbital Radius of Target Spacecraft
$\vec{r}_c$	Position of Chase Spacecraft
r_{cg}	Distance of Chase Spacecraft from Goal
$\vec{r}_{cg}$	Relative Position of Chase Spacecraft with respect to Goal
r_{co}	Distance of Chase Spacecraft from Obstacle
$\vec{r}_{co}$	Relative Position of Chase Spacecraft with respect to Obstacle
$\vec{r}_0$	Initial Relative Position Vector
$\vec{r}_f$	Final Relative Position Vector
$\vec{r}_g$	Relative Position of Goa with respect to Chase Spacecraft
r_{init}	Initial Distance of Chase Spacecraft from Goal
r_{max}	Maximum Allowable Distance of Chase Spacecraft from Goal
$\vec{r}_o$	Relative Position of Obstacle with respect to Goal
r_{sg}	Distance of Proximity Spacecraft from its Goal
$\vec{r}_t$	Position of Target Spacecraft (Inertial Frame)
S_p	Solar Radiation Pressure
T	Torque
T_{CMD}	Torque Commanded by Attitude Control Law
T_{DAMP}	Torque Damping due to Magnetotorquers
T_{GG}	Torque due to Gravitational Gradient
T_{MED}	Torque due to Momentum Exchange Device
T_{mw}	Torque due to Momentum Wheel
T_{REQ}	Torque Required by Magnetotorquers Control Law
T_{RW}	Torque due to Reaction Wheels
t	Time

t_d	Time Duration of Maneuver
u^*	LQR Optimal Control feedback
$u_{\max}$	Maximum Control Effort along Cartesian Axis
v	Velocity
$\vec{v}_{atm}$	Spacecraft Velocity Vector with respect to Earth's Atmosphere
v_0	Initial Velocity
$\vec{v}_{co}$	Relative Velocity of Chase Spacecraft with respect to Obstacle
$\vec{v}_g$	Desired Relative Velocity of Chase Spacecraft with respect to Goal
$\vec{v}_o$	Desired Relative Velocity of Chase Spacecraft with respect to Obstacle
$\vec{v}_{ECI}$	Velocity Vector with respect to ECI Coordinate System
v_f	Final Velocity
$v_{\max}$	Maximum Desired Relative Velocity
$\vec{v}_{offset}$	Velocity due to Stationary Offset Position with respect to RSW
$\vec{v}_{atm}$	Spacecraft Velocity Vector with respect to Earth's Atmosphere
$V(x)$	Lyapunov Potential Function
V_g	Attractive Potential of Goal
V_o	Repulsive Potential of Obstacle
w	Orbital Argument of Perigee
x, y, z	System States
$x*$	Nominal Point
$\vec{X}(t)$	System State Vector
X_B, Y_B, Z_B	Spacecraft Body Frame Coordinates
Z_{RW}	Reaction Wheel Alignment Matrix with respect to Spin Axis
A	Performance Surface Contour Shaping Matrix
α	Euler Angle
$\vec{\alpha}$	Angular Acceleration Vector
α_Q	LQR State Gain Matrix Numerator Constants
β_{R_1}	LQR Control Effort Gain Matrix Numerator Constants
Δv	Change in Velocity Magnitude

Δt	Time Increment
ε	Innovation, Residual, or Error
φ	Roll Angle of Rotation about X_B-axis
λ_k	Step Size or Magnitude Shaping Function
λ_g	Goal Potential Shaping Function
λ_o	Obstacle Potential Shaping Function
μ	Gravitational Parameter of Earth
μ_m	Gravitational Parameter of Moon
μ_S	Gravitational Parameter of Sun
μl	Momentum Learning Coefficient
μk	Levenberg-Marquardt Coefficient
Φ	State Transition Matrix
θ	Pitch Angle of Rotation about Y_B-axis
ρ	Atmospheric Density
σ	Standard Deviation for Obstacle Region of Influence
υ	Orbital True Anomaly
Ω	Orbital Right Ascension of Ascending Node
ω	Orbital Angular Velocity
$\vec{\omega}_{BN}$	Angular Velocity of Spacecraft Body Frame with respect Inertial Frame
$\vec{\omega}_{BO}$	Angular Velocity of Spacecraft Body Frame with respect to Orbital Frame
ω_e	Angular Velocity of the Earth
ω_{mw}	Spin Rate of Momentum Wheel
$\vec{\omega}_{ON}$	Angular Velocity of an Orbital Frame with respect to an Inertial Frame
ψ	Yaw Angle of Rotation about Z_B-axis

EXECUTIVE SUMMARY

As spacecraft technology has advanced, simultaneous control of multiple cooperative spacecraft has become a desired mission capability. The first generation of spacecraft were individually stabilized in orbit by manual control from ground stations. The second generation of spacecraft were placed in stable constellations by ground stations, with some basic controls automated on-board the spacecrafts. The next generation will be required to reliably perform autonomous multiple spacecraft close proximity operations while avoiding collisions. Therefore, a need exists for robust and efficient automated and distributed control of multiple spacecraft for emerging servicing missions, involving simultaneous rendezvous and docking scenarios.

In this research a novel multiple spacecraft close proximity control algorithm was developed, refined, and thoroughly simulated using high fidelity six degree of freedom spacecraft models. The developed control algorithm combines the dynamic optimization of a Linear Quadratic Regulator (LQR) and collision avoidance capability of Artificial Potential Field (APF) approaches. Development and evaluation of the LQR/APF control algorithm was supported by the realization of a MATLAB and Satellite Tool Kit (STK) simulation interface technique. This versatile MATLAB-STK simulation interface allowed for both multiple spacecraft model validation and simulation visualization. The LQR/APF multiple spacecraft control algorithm was further evaluated by virtual hardware-in-the-loop (VHIL) configuration at the NPS Spacecraft Robotics Laboratory (SRL). The VHIL structure utilizes independent processors which simulate spacecraft and interact as multiple spacecraft.

Analysis of numerous close proximity maneuvers proved the LQR/APF to be both effective and efficient in conducting simultaneous spacecraft missions. The LQR/APF avoids actuator saturation while avoiding both stationary and moving obstacles. Monte Carlo simulations showed the multiple spacecraft control to be both stable and robust. It also established a parametric baseline for future multiple spacecraft close proximity control algorithms, by evaluating collision avoidance requirement, fuel efficiency, and maneuver duration objectives. The LQR/APF control algorithm's desirable performance

gives spacecraft designers and mission planners a useful means of developing and forecasting maneuvers. Finally, successful implementation of the multiple spacecraft control algorithm in a VHIL configuration paves the way for future terrestrial and orbital hardware testing.

ACKNOWLEDGMENTS

I would like to thank everyone who has contributed to my academic progress. There were countless teachers and classmates who motivated, challenged, and inspired me over the years. I was fortunate to have the support of a dissertation committee consisting of expert educators and researchers. Professor Xiaoping Yun has been a valuable mentor and patient teacher throughout the entire process. Assistant Professor Marcello Romano continually offered guidance and course corrections, which were vital to the completion of my research. Assistant Professor Ravi Vaidyanathan was a constant source of inspiration and motivation, with fresh ideas and enthusiasm. Professor Roberto Cristi was a selfless educator who helped me keep my sense of humor. Professor Tri Ha offered practical advice and perspective. With the knowledge and experiences that they have given me, I hope to improve the field of knowledge for those who follow.

I have found that it is easy to suggest a quick solution when you do not completely understand the problem. As you endeavor to further understand the problem, continue to attempt numerous solutions. This allows you to evaluate options along the way as you expand your understanding of the problem and the relevance of possible solutions. This is a researcher path to success.

Finally, I would especially like to thank my wife for her support, understanding, and patience through the late nights of studying. You mean more to me than I can adequately express with words…but I am working on an equation.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. MOTIVATION AND BACKGROUND

As spacecraft technology has advanced, simultaneous control of multiple cooperative spacecraft has become a desired mission capability. Therefore, there is a need to develop an autonomous distributed control algorithm for multiple spacecraft during close proximity operations. The first generation of spacecraft were individually stabilized in orbit by manual control from ground stations. The second generation of spacecraft were placed in stable constellations, and formations by ground stations, with some basic controls automated on-board the spacecrafts. The next generation of spacecraft will be required to reliably perform autonomous close proximity operations, including multiple spacecraft dispersion, rendezvous, and docking maneuvers.

There are numerous mission concepts that involve the convergence of multiple spacecraft in close proximity. At present, these missions are parameterized through pre-determined (*a priori*) orientations and trajectories, and executed with centralized manual control. This approach is extremely cumbersome with respect to time and computational expense, relies on high communication capacity between spacecraft, and allows for no dynamic reconfiguration in spacecraft control. While, large spacecraft formation tracking and keeping has received a great deal of study, research in the area of practical multiple spacecraft close proximity operations is limited [1]. Current spacecraft rendezvous and docking require that the spacecraft cooperate and are deliberately designed to work together. Approaching spacecraft usually use tracking based algorithms, with advanced sensors and processors, to approach the desired position. Additionally, these close proximity spacecraft maneuvers are typically manually operated. The need for advanced sensors, processors, and actuators limits the application of previous spacecraft control algorithms to a wide variety of spacecraft platforms and mission.

These issues could be addressed with a computationally efficient, robust, distributed control algorithm allowing for multiple spacecraft close proximity operations, with the capacity for dynamic reconfiguration for collision avoidance. Research and experience with terrestrial based robots have matured the application of iterative artificial

potential field (APF) based control algorithms [2][3][4][5]. These APF algorithms are geometrically based with converging minimization searches along the direct Steepest Descent path. The APF have been refined to include obstacle repulsion potentials which allow for robots to avoid collisions in moderately dynamic environments [6][7]. However, these algorithms have not been widely applied to the unique challenges presented in spacecraft control. The simplicity of the potential field based control algorithms are a good match for spacecraft application with limited proximity sensors and processing capability. Previously proposed spacecraft potential field based controllers have been task and platform specific and not robust in the full range of possible close proximity operations. Also, fuel efficiency and optimization has been limited to maintaining spacecraft formations. The consideration of efficiency while maintaining collision avoidance in close proximity operations has been limited, and typically requires dramatic increases in computation or centralization.

Efficiency concerns have driven spacecraft control system designers toward optimal control algorithms [8]. However, the restricted computation capability onboard the current generation of spacecraft forces trade-offs between such algorithms and simple feedback architectures which can be hosted locally. An original solution to the efficiency demands and the collision avoidance capability required for the emerging autonomous close proximity operations is the combination of optimal control and geometric collision avoidance. An iterative Linear Quadratic Regulator (LQR) control algorithm can be used as driving force toward the desired goal, replacing the more conventional force due to the attractive potential. The LQR has the advantage of incorporating the spacecraft linearized relative dynamics by utilizing variable state and control effort gain matrices in order to solve for an optimal cost solution at each iteration. These allow variation for state weighting during convergence toward the goal.

The fusion of a LQR, including relative dynamics, and an APF based collision avoidance capability yields a new and promising multiple spacecraft control algorithm. The LQR control effort serves as the attractive force toward goal positions, while the APF-type repulsive functions provide collision avoidance for both fixed and moving obstacles. This LQR/APF multiple spacecraft proximity control algorithm offers desirable performance in a robust close-proximity, and establishes a baseline for fuel

efficiency while maintaining collision free operations. Critical evaluation of the multiple spacecraft control algorithm utilized high fidelity six DOF spacecraft models, allows assessment with realistic spacecraft dynamics and constraints. The multiple spacecraft close proximity control algorithm was evaluated for several multiple spacecraft emerging maneuvers, which may require gathering, rendezvous, and docking operations.

This dissertation includes the requirement, development, simulation, refinement, evaluation, and analysis of an autonomous distributed control algorithm for multiple spacecraft during close proximity operations. Literature reference and review material is presented in each applicable dissertation chapter. This ensures topical material is presented as research concepts are presented and developed. Chapter I closes with a summary of the research and its contributions. The mission profile and requirements are presented in Chapter II. The space environment and a high fidelity six DOF spacecraft model are developed in Chapter III. The equations of relative motion between multiple spacecraft in close proximity and the typical rendezvous control are introduced in Chapter IV. The close proximity spacecraft control algorithm is developed in Chapters V–VII. A detailed evaluation of the LQR/APF performances during convergence, rally, rendezvous, and docking maneuvers is presented in Chapter VIII, followed by a Monte-Carlo method analysis of the heuristic control algorithm in Chapter IX. Finally in Chapter X, a virtual hardware-in-the-loop (VHIL) implementation of the multiple spacecraft control algorithm is discussed.

B. RESEARCH GOALS

This research is intended to advance the field of multiple spacecraft control by developing an autonomous distributed control algorithm for multiple spacecraft in close proximity operations, including simultaneous rendezvous and docking. The architecture of the controller is designed to convolve the collision avoidance capacity of APF approaches with the LQR capacity to address dynamic platform constraints. The developed approach, referred to as the LQR/APF control algorithm, is an iterative feedback based algorithm which allows for efficient and timely proximity spacecraft maneuvers. The LQR/APF control algorithm combines efficient LQR performance,

utilizing linearized relative dynamics, and APF with geometric collision avoidance. This convergence of efficiency and collision avoidance into a control algorithm is an enabler for future spacecraft missions.

The developed control algorithm was thoroughly simulated with a high fidelity six degree-of-freedom (DOF) spacecraft model operating in 3D space. Next, the simulation model was fully developed with consideration of both perturbation forces and torques. The control algorithm takes into account considerations on realistic actuators and sensors performances. Finally, the control algorithm was tested and evaluate by VHIL implementation in the dedicated Spacecraft Servicing and Robotics Laboratory located at the Naval Postgraduate School (NPS). This allows for validation of the control algorithm with independent processors simulating spacecraft performance based on limited and incremental state information. The laboratory also hosts the Autonomous Docking and Spacecraft Servicing test-bed, whose task is to simulate on-the-ground the navigation and control of the docking between multiple free-flying small spacecraft.

C. MAIN CONTRIBUTIONS

Specific research contributions include:

- Development of a MATLAB-Simulink interface with STK which allows for multiple spacecraft model validation and dynamic environment visualization. This developer friendly tool proved critical for engineering analysis of control algorithm performance. The use of STK, which is a key spacecraft industry standard, allows for clear presentation of developed control algorithm performance to the spacecraft field.
- Development of a multi-purpose APF based control algorithm which performs close proximity operations in the spacecraft environment with realistic spacecraft constraints. This algorithm was refined with the addition of collision avoidance capability to be effective in a broad range of spacecraft maneuvers involving multiple spacecraft and obstacles.
- Development of an iterative LQR control algorithm, with variable gain matrices, which performs close proximity operations in the spacecraft environment with realistic spacecraft constraints.
- Combination of the LQR and APF control concepts. The iterative LQR generated control effort serves as the attractive force toward goal positions, while the APF repulsive functions provide collision avoidance for both fixed and moving obstacles. This algorithm was further refined for a full range of maneuvers involving stationary and moving obstacles.

- Thorough comparison of maneuver performance, efficiency, and duration. This research established a parametric baseline for future multiple spacecraft close proximity control algorithms, by evaluating collision avoidance requirement, fuel efficiency, and maneuver duration objectives.
- Implementation and evaluation of the control algorithm in a VHIL configuration. Utilizes independent processors which simulate spacecraft and interact as multiple spacecraft.

THIS PAGE INTENTIONALLY LEFT BLANK

II. OVERVIEW OF MULTIPLE SPACECRAFT MISSIONS

A. MULTIPLE SPACECRAFT CONTROL

The controlled spatial interaction of systems involving multiple vehicles, robots, aircraft, or spacecraft in close proximity is a complex task. How multiple vehicles are controlled depends on a wide range of operating environments, missions, objectives, and organizational combinations. Developing systems consisting of multiple autonomous vehicles that cooperatively perform a task or behavior is of paramount importance to the engineering community [9]. The distinctive space environment makes for particularly interesting and challenging control algorithm requirements based on orbital dynamics, communication limitations, and tight spacecraft sensitive to disturbances and constraints. The dynamics of the orbital spacecraft is affected by disturbances and perturbations. Traditionally, individual spacecraft maneuvers have been based on fuel efficiency requirements, since fuel is a critical resource in the life of a spacecraft. During large early staging maneuvers, the conservation of fuel is generally more critical then the timeliness of any particular task. Once in the desired mission orbit, the only regular translational maneuvers which most individual spacecraft perform are for general station keeping. Also, due to the relatively large distances between typical orbital objects collision avoidance maneuvers are rarely necessary and seldom executed. However, due to spacecraft technology improvements, there is now a greater desire to control multiple spacecraft in close proximity operations. These multiple spacecraft, close proximity operations require collision avoidance while each spacecraft executes the desired close proximity task.

Advances in technology continue to decrease the size, weight of components while increasing the capability of payloads, sensors and processors. This trend has enabled the space industry to decrease the size of spacecraft. Small satellites with mass less than a few hundred kg are becoming more common. Launch opportunities are more readily available for smaller spacecraft since more launch vehicle can support them. Many of these relatively small satellites can be deployed into orbit from the same launch vehicle. Once in orbit the spacecraft will need to disperse or converge, relative to each other depending on their mission. As spacecraft get closer to each other the execution of

collision avoidance, rendezvous, and docking maneuvers over a short timeline may take priority over optimizing individual spacecraft's fuel consumption. During these close proximity maneuvers, the desires are to maintain efficient fuel management while accomplishing the maneuver objective quickly. The longer each spacecraft stays in transition during close proximity the more precise station keeping is required; therefore more propellant may be used.

B. MULTIPLE SPACECRAFT MISSIONS

Control algorithms for multiple spacecraft need to take the similarity and differences of each spacecraft into consideration, which is discussed in detail in Chapter II.B.1. In this research cooperative homogeneous spacecraft with similar sensors and control algorithms are considered. Additionally, it is useful to define the phase of the spacecraft mission in relationship to the distances from other spacecraft or the goal position. A discussion of relative orbital mission phases is presented in Chapter II.B.2. Also, the organizational grouping of spacecraft depends on the manner in which the spacecraft maintain position with respect to each other. The various organizational groupings of spacecraft are discussed in Chapter II.B.3. Precision control may be desired in some applications, such as pointing spacecraft with high resolution imagery payloads, and loose control may be desired in other applications, such as station keeping of communication satellites. Mission motivations for autonomously controlled and synchronized multiple spacecraft in precise spatial configurations are numerous, and include interferometry, communications, and power generation. These cooperative spacecraft may need to be able to converge or diverge in a safe and efficient fashion. For servicing missions, it may be necessary for two or more spacecraft to rendezvous, or even dock.

First, multiple spacecraft interferometry is based on the idea that the precise control of the relative position and orientation of multiple spacecraft payloads could result in performance equivalent to a much larger spacecraft payload. Multiple spacecraft are positioned in order to form a distributed array. Distributing spacecraft payloads (e.g., sensors) into precise spatially configurations, or assembling multiple spacecraft on orbit, require higher level control algorithms. Required control algorithm capabilities include

the establishment and maintenance of a given formation and any possible reconfiguration of the formation. Formation maintenance control has been given its due of attention [10] [11].

Second, reconfigurable cooperative formation mission concepts are based on collectives of small spacecraft working in unison to perform a greater function. For the purpose of spacecraft formation control it is generally assumed that each spacecraft is relatively homogeneous and executes centralized control strategies in order to minimize fuel consumption during formation flight. Each small spacecraft coordinates with others in the group and shares processing, communication, and payload or mission functions. Defense Advanced Research Projects Agency (DARPA) is currently studying the idea of fractionated spacecraft architectures which decompose the overall spacecraft function into a free-flying network of component modules. The traditional monolithic spacecraft might be replaced with a group of smaller spacecraft interacting wirelessly [12]. Individual spacecraft may be required to perform close proximity maneuvers autonomously while the mission or configuration changes. These fractionated, modular spacecraft have advantages over traditional monolithic spacecraft, as analyzed and assessed in [12][13]. Upgrading the group can be done iteratively in order to increase overall performance and mission duration. This can substantially enhances the flexibility, responsiveness, robustness, and lifecycle of the overall spacecraft function.

Third, rendezvous mission concepts are usually based on a cooperative spacecraft approaching another spacecraft within a common spatial region. The goal position is usually occupied by a cooperative spacecraft in the same orbital plane as the maneuvering spacecraft. The Space Transportation System (STS), often referred to as the Space Shuttle, and the ISS are often shown docking in orbit. The Space Shuttle is astronaut controlled as it docks to the ISS. This process is not autonomous, but there is autonomous docking of Russian Soyuz and Progress spacecraft with the ISS. The process of automated spacecraft docking was pioneered by the Soviet Union; however the automated system occasionally fails to complete the task. According to NASA, current state of the art Russian automated rendezvous and docking systems have a current failure rate of approximately 10-15 % [14]. As a result, ISS's Zvezda module is equipped with the Russian built Telerobotically Operated Rendezvous Unit (TORU) manual docking

system which can be operated by cosmonauts [15]. However, manual rendezvous control can not be relied upon since human space flight is very dangerous, impractical, and unnecessary for most on orbit missions. The European Space Agency (ESA) is currently developing the Automated Transfer Vehicle (ATV) to automatically dock with the ISS. However, close proximity operations of spacecraft are often complicated by momentary communication and autopilot navigation failures.

The motivation to rendezvous is not limited to manned-spaceflight. Rendezvous technology has also evolved with small spacecraft development, such as the NASA's Demonstration for Autonomous Rendezvous Technology (DART) [16], Air Force Research Laboratory's (AFRL) Experimental Satellite Systems (XSS)-10 and XSS-11 [17][18], Naval Research Laboratory's (NRL) Spacecraft for the Universal Modification of Orbits (SUMO) [19], and DARPA's Orbital Express [20]. The SUMO program has evolved into the Front-end Robotics Enabling Near-term Demonstration (FREND) program, which maintains the objective of autonomous rendezvous and grapple operations. For a full discussion on the SUMO/FREND in context of the current status of autonomous rendezvous and capture missions, refer to Creamer [21]. Although, the most well known of these programs may be the XSS-11. The AFRL Space Vehicle Directorate at Kirtland Air Force Base in New Mexico developed the XSS-11 in order to exhibit the ability for a small satellite to autonomously plan and rendezvous with passive and cooperative objects in LEO [22]. The use of micro-satellites to monitor, inspect, service, repair, and re-fuel larger spacecraft is a long term goal. The closest the XSS-11 approached and maneuvered around another object in space was approximately 500 meters. The XSS-11 used on-board laser range finders to measure the distance to target objects. Most of the XSS-11 flight is being conducted manually with autonomous planners running in the background. In addition, DARPA's Orbital Express Advanced Technology Demonstration Program is intended to validate the technology and techniques for on-orbit refueling and reconfiguration of two satellites. The mission, which launched in late 2006, is intended to perform seven autonomous rendezvous and capture scenarios [23]. These will include component exchange and propellant transfer events. There is also research to apply rendezvous technology to smaller spacecraft, such as DARPA's Tiny, Independent, Coordinating Spacecraft (TICS) program [24]. These

small spacecraft programs are leading the way for advanced autonomous close proximity operations by supporting the development of enabling technologies.

There are several research groups contributing to the development of autonomous formation flight and docking control algorithms. Examples of recent progress in the field include such projects as Massachusetts Institute of Technology's (MIT) Space Systems Laboratory's Synchronized Position Hold Engage Reorient Experimental Satellites (SPHERES) [25] [26] and AFRL TechSat 21 projects. The TechSat 21 program, which was cancelled in 2003, was intended to develop the concept of using clusters or formations of collaborative small satellites to fulfill complex missions, such as distributed aperture remote sensing, geo-location, of moving ground targets. As with most fractionated spacecraft architectures, the key motivating design concepts were that distributed mission architecture is more tolerant to damage and can also reduce the overall system cost. The SPHERES are intended to be used as a test bed for formation flight and reconfiguration, as well as autonomous rendezvous and docking technologies [26]. The first SPHERES satellite reached the ISS in May 2006 and has begun testing scenarios. These major research activities emphasize the challenge and ongoing requirement of developing a multiple spacecraft control algorithm.

Maintaining tight formations for interferometry and other cooperative missions for long durations, is extremely taxing on spacecraft fuel. Also, thruster firings result in out gassing of propellant in close proximity may cause problems for spacecraft sensors and payloads. So, one way to ensure proper positioning and attitude relative to multiple spacecraft is to actually connect them in some manner. The connection of spacecraft on orbit is referred to as docking. Efficient and safe docking is the ultimate close proximity operational goal. The ability to dock spacecraft, in a safe, robust, and autonomous manner, is the cornerstone in being able to re-supply, service, upgrade, and reconfigure spacecraft while in orbit.

1. Homogenous and Heterogeneous Spacecraft

Any given group of spacecraft may have characteristics that make them similar or different. The likeness among vehicles is of significant consideration in the coordinated

control of a group of spacecraft. Spacecraft are termed homogenous, if the spacecraft are identical, or heterogeneous, if the spacecraft are different. The likeness evaluation of spacecrafts may be based on a large variety of parameters, varying from physical characteristics (e.g., mass, volume, and structure/configuration) to the spacecrafts capabilities (e.g., communication, data handling, power, payloads, and control systems). Limited types of launch vehicle capabilities and spacecraft bus designs give an upper bound to the mass and volume of orbital spacecraft. For instance, the static launch envelope inside the launch vehicle fairing limits the volume of a spacecraft [27]. Increased modularity of systems allows for a larger variation in payloads, sensors, and configurations. This research will be based on homogenous spacecraft, including similar control system sensors and actuators. However, dissimilar size and shape spacecraft may be considered as long as the dynamics of each spacecraft is properly modeled. Exploring the varying perturbations dynamics of multiple heterogeneous spacecraft is beyond the scope of this work. In this research, all commanded spacecraft are assumed to use the same basic control scheme and be equipped with sensors and actuators which offer the same level of precision.

2. Orbiting Spacecraft Mission Phases

The multiple spacecraft control algorithm used depends on the phase of spacecraft operations. In order to distinguish between phases of spacecraft operations based on physical proximity, it is helpful to adapt some terminology from missile engineering. In missile interception, four fundamental stages of flight have been defined [28]. These stages are commonly referred to as launch, midcourse, terminal, and endgame stages of flight. They can be extended in spacecraft phase proximity operations, referred to as launch, midcourse, rendezvous, and docking. Here are the four spacecraft proximity phases:

1. Launch phase ends after the satellite separates from the launch vehicle upper stage (booster) and it is in operational orbit. All large orbital maneuvers are performed during the launch phase.

2. Midcourse phase begins when the spacecraft has been stabilized into its operational orbit and it can perform station keeping [29]. Small orbital corrections may be performed during this midcourse phase.

3. Rendezvous, or terminal, phase is the when the spacecraft converges to a common point in its operational orbit. This rendezvous phase begins when the spacecraft receives its rendezvous command to translate toward a point in space. This space may be occupied by another spacecraft or be an empty space which the spacecraft will move into and occupy. If a group of spacecraft are commanded to rendezvous to empty spaces relative to each other they will form a spacecraft formation. On the other hand, if a group of spacecraft are converging on the same location in space they are converging for docking.

4. Docking, or endgame, phase begins when spacecraft on-board proximity sensor acquire the target location/vehicle. Spacecraft docking is a very precise maneuver, since uncertainties in guidance and attitude need to be corrected quickly and effectively during this engagement scenario. For this research, short duration, close proximity operations will be limited to 1.0 km of separation between spacecraft in nearly circular orbits. These assumptions are consistent with the staging required to get cooperative spacecraft into the same spatial region. Long term fuel optimal formation keeping and larger orbital eccentricity variations are considered separate issues which are not addressed in this research.

3. Multiple Spacecraft Groups

Before addressing the control of multiple spacecraft, it is useful to define the differences between constellations, formations, and swarms/clusters. Constellations are groups of spacecrafts in relative motion, or orbit(s), but their positions and attitudes are not dynamically coupled in any way [30]. This means that a change in position or velocity of one spacecraft does not impact the others. For instance, the GPS constellation is made up of at least 24 satellites with four satellites in six different GEO planes. Each

satellite is maintained in its orbit independently via ground commanding. Maintaining the orbital position of the spacecraft, referred to as station keeping, is required due to several perturbation influences (e.g., Sun and Moon gravitational fields, variations in Earth's gravitational field, aerodynamic drag). Constellations are usually maintained via centralized ground control.

Formations are groups of spacecrafts which are dynamically coupled through a control law [30]. The motion of one spacecraft can give relative state information about at least one other spacecraft. Formation flight control of multiple small spacecraft is the task of maintaining the spatial formation, via control of the motion of the individual spacecraft in order to maintain the overall formation shape. The control law typically uses the position and velocity of one spacecraft to command another spacecraft [30]. The lead spacecraft moves along a commanded path and the following spacecraft(s) maintain a relative position, attitude, and velocity with respect to the leader. This is a common concept in leader/follower tracking schemes [31], where the relative motion is like a flock of geese in flight. Collisions are generally avoided through strict control of the following spacecraft's motion. For spacecraft formation control a virtual structure control approach is commonly used [11]. In spacecraft orbital terms, the lead spacecraft is referred to as the “Target” and the tracking spacecraft is called the “Chaser.” The target motion may be represented by an imaginary spacecraft. The idea of an imaginary target leads better understanding of a swarm/cluster.

A swarm/cluster is a group of spacecrafts that move in concert with one another, but without strict control of relative positions, attitudes, or velocities. In the idea of a bee swarm, an outside observer can see the relatively smooth motion of an entire swarm but can not determine the specific relationship between any two spacecrafts. Think of the center of the swarm as an imaginary target and all of the spacecrafts as Chasers which only stay within a specific range. A tight swarm would be represented by a small acceptable distance between the Target and Chasers. However, the specific position, attitude, and velocity of each Chaser spacecraft are not centrally controlled. Each spacecraft autonomously manages its own motion and is only influence by its range from the target and the need not to collide with other spacecrafts.

The desire is to offer a control algorithm which bridges the gap between single cost optimal fuel trajectory tracking of rigid formation and the emergent behavior of swarms. Using APF based control, allows for incorporation of collision avoidance directly into the close proximity control algorithm. The goal position of each spacecraft is explicit and the general spacecraft path will be in the predictable direction of the goal. The obstacle potential functions can be defined and used to determine navigation paths which are robust enough to allow for both convergence and collision avoidance. For a detailed development of APF control refer to Chapter V.

THIS PAGE INTENTIONALLY LEFT BLANK

III. SINGLE SPACECRAFT KINEMATICS AND DYNAMICS MODELING

A. INTRODUCTION

The first computational step of developing a control algorithm is to establish the system model. For this research, the fundamental system is a high fidelity six DOF spacecraft orbiting the Earth. This chapter introduces the dynamics and kinematics model used through the rest of the dissertation. The orbital perturbations included are fourth order harmonics in the Earth gravitational potential field, atmospheric drag on the spacecraft, third-body (Sun and Moon) forces, solar-radiation pressure, and mass variation due to thruster firings. Refer to Chapter X. for detailed description of the spacecraft model validation technique.

A spacecraft orbit is usually expressed relative to a right-hand inertial (X, Y, Z) coordinate system with its origin at the center of the Earth and the center of the Earth in the plane of the spacecraft's orbit. This reference frame is called Earth Centered Inertial (ECI), shown in Figure 3.1. The X-axis points toward the vernal equinox, the Y-axis is $90°$ counterclockwise from the X-axis in the equatorial plane, and the Z-axis extends through the North Pole [32]. For scaling reference, it is worth noting that the radius of the Earth spheroid is approximately, $R_e = 6{,}378.1$ km.

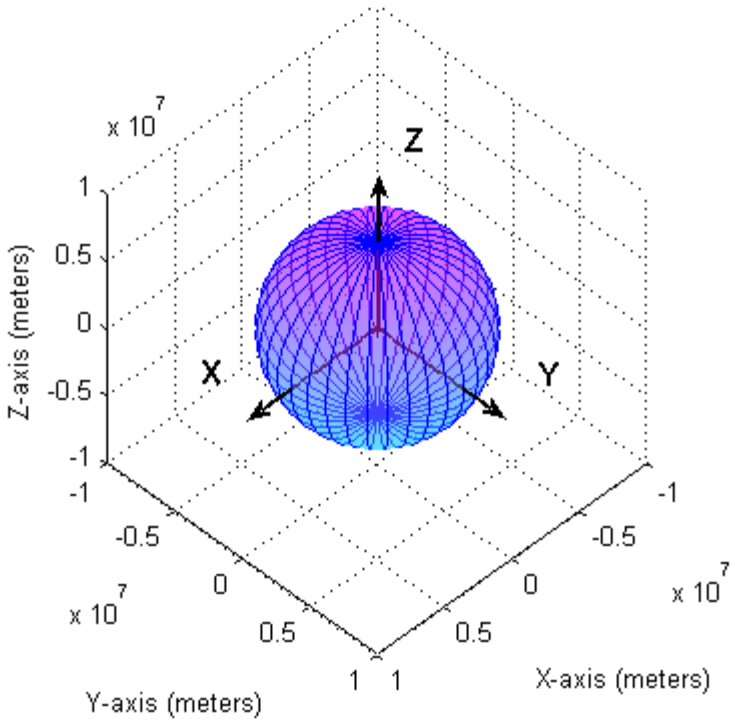

Figure 3.1 Earth with ECI Coordinate System.

A spacecraft orbit is determined by the gravitational forces acting upon it. The spacecraft orbital dynamics, which is described in Chapter III.B, can be simplified as the solution to a two body problem; refer to Chapter III.B.1. The description of a spacecraft's orbit can be characterized by a finite number of parameters, which are discussed in Chapter III.B.2. However, the relative orbital position and velocity of a spacecraft depends on the reference frame of interest. For instance, in order to describe the spacecraft position with respect to the Earth, an inertial frame like that described above would be sufficient. However, there is also a need to describe one spacecraft position and velocity relative to another. Therefore, additional reference frames need to be defined; refer to Chapter III.C. With objects in multi-dimensional space, there is not only a need to define a position and velocity in space, but also a need to define the objects orientation. The attitude dynamic of the spacecraft is discussed in Chapter III.D. There are other forces and perturbations which influence the spacecraft orbit; these are discussed in Chapter III.E. Finally, the overall characteristics of the six DOF spacecraft model used in this research are outlined in Chapter III.F.

B. KEPLERIAN ORBITAL DYNAMICS

1. Two Body Problem

The fundamental description of a spacecraft orbit is determined by the gravitational force of the central body. In this document the Earth is assumed to be the primary body of interest and any spacecraft are in orbit around it. Simple two body orbital motion is described by Newton's Law of Universal Gravitation [33].

$$F_g = -\frac{G\, m_e\, m_s}{r^2}\left(\frac{\vec{r}}{r}\right) \tag{3.1}$$

where F_g is the force of Earth's gravity acting upon the spacecraft, G is the Universal Gravitational Constant $(6.673\times10^{-11} m^3\, s^{-2}\, kg^{-1})$, m_e is the mass of the Earth $(5.9733328\times10^{24}\, kg)$, m_s is the mass of the spacecraft, $\vec{r}$ is the relative distance vector from the center of mass (COM) of the Earth to spacecraft, and r is the Euclidean (2-norm) distance of the spacecraft from the COM of the Earth. A 3D position vector in Earth inertial reference frame, can be represented as

$$\vec{r} = r_1\hat{X} + r_2\hat{Y} + r_3\hat{Z} \tag{3.2}$$

The Euclidean (2-norm) of a 3D vector is

$$r = \sqrt{(r_1)^2 + (r_2)^2 + (r_3)^2} = \left|\vec{r}\right| \tag{3.3}$$

The assumptions with this simple two body problem are that the Earth and spacecraft can be modeled as point masses, the gravitational field is symmetric, and no other external forces are acting on the Earth or spacecraft. Additional orbital forces and perturbations will be considered as the dynamic model is further developed in Chapter III.E.

The left side of equation (3.1) can be expanded with Newton's Second Law [33],

$$F = m\,a \tag{3.4}$$

where F is the force resulting from the mass (m) and acceleration (a) of an object. The relative acceleration of the spacecraft can be determined by subtracting the acceleration

of the Earth from the acceleration of the spacecraft, and substituting equation (3.1) and equation (3.4) results in

$$\vec{a} = -\frac{G(m_e + m_s)}{r^2}\left(\frac{\vec{r}}{r}\right) \tag{3.5}$$

The relative acceleration is the second derivative of the relative position vector $\left(\ddot{\vec{r}}\right)$ and the term on the far right is the normalized position vector. Next, make use of the gravitational parameter, defined as

$$\mu = G\,m_e = 398600.4418 \times 10^9\, m^3 s^{-2} \tag{3.6}$$

The value of μ is based on modeling of the Earth, such as the World Geodetic System (WGS-84) [32]. For most applications, the mass of the spacecraft is negligible compared to the mass of the earth, so the two body motion equation simplifies as follows

$$\ddot{\vec{r}} + \frac{\mu}{r^2}\left(\frac{\vec{r}}{r}\right) = 0 \tag{3.7}$$

This is the core equation for determining spacecraft position. Given the initial values of the relative position and velocity of the spacecraft, the orbit can be propagated using numerical analysis. The integration of Equation (3.7) leads to the classical Keplerian orbits, whose shape can vary according to the spacecraft initial position and velocity. In particular closed (circle, ellipse) and open (parabola, hyperbola) orbits are the possible trajectories for the spacecraft.

2. Orbital Elements

Any spacecraft specific orbit position can be also represented by six classical, or Keplerian, orbital elements, which can substitute the Cartesian coordinates. The six classical orbital elements are the following:

1. Semi-major axis (a) determines the size of the orbital ellipse as

$$\mathrm{a} = \frac{r_a + r_p}{2} \tag{3.8}$$

Orbits are typically referred to in categories that relate to general radial size of their orbit. Low Earth orbits (LEO) are typically around an altitude of 1,000 km. Middle Earth orbits (MEO) are typically around an altitude of 10,000 km. Geosynchronous orbits (GEO) occupy an altitude of 36,000 km, and remain in a relatively stationary position relative to the Earth, depending on inclination. A special case of this type of orbit is the geostationary orbit, which remains in a stationary position over the Earth's equator, due to an inclination of zero degrees. Using the semi-major axis, the period (P_T), or duration, of the spacecraft's orbit can be determined as

$$P_T = 2\pi\sqrt{\frac{a^3}{\mu}} \tag{3.9}$$

2. Eccentricity (e) determines the shape of the orbit, such that when $e = 0$ the orbit is circular. As the eccentricity increases from 0 up to 1, the shape becomes less circular and more ellipse-like. For e =1 the orbit is a parabola and for e>1 the orbit is a hyperbola.

3. Inclination (i) is the tilt angle relative to the Earth's equator. If $i = 0$ then the orbit is in plane with the Earth's equator.

4. Longitude of ascending node (Ω) (also referred to as the right ascension of the ascending node) is the relative swivel angle of the orbital ellipse. This angle is measured from the inertial X-axis to the line defining the point where the orbit crosses the equator moving from the south to the north.

5. Argument of perigee (w) is the angle between the ascending node and the closest radial distance (lowest altitude) from the orbit to the Earth (perigee).

6. True anomaly (υ) is the angle from the perigee to the actual spacecraft location, thus it varies throughout the orbit. This can be thought of as the phasing of the spacecraft in the orbital ellipse, or circle.

These six orbital parameters are useful for visualizing the spacecraft's orbit, as shown in 0.

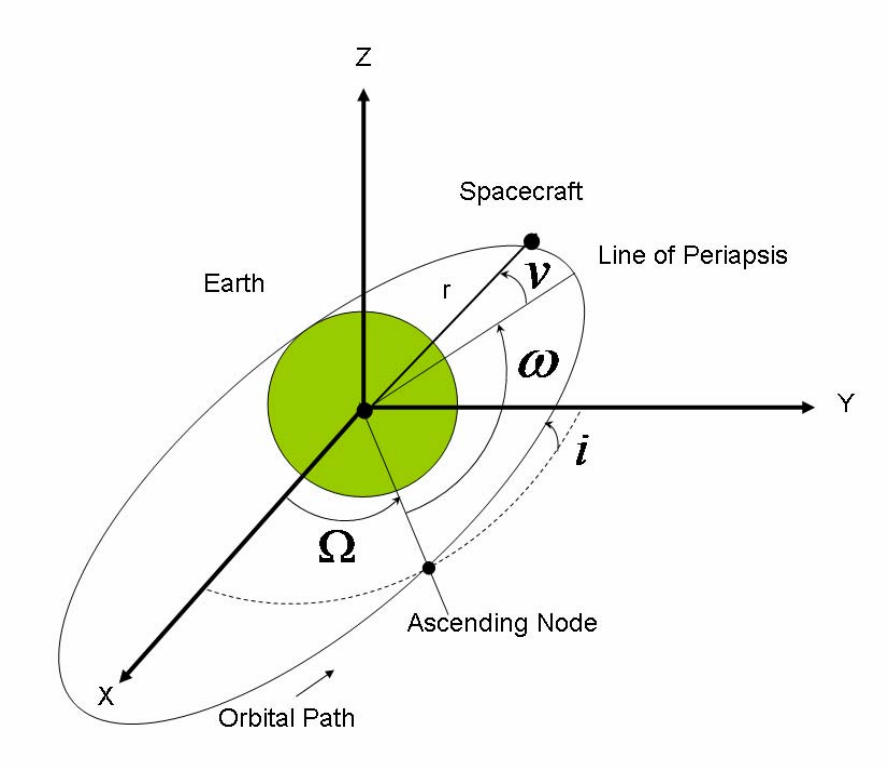

Figure 3.2 Classical Orbital Parameters.

C. REFERENCE FRAMES AND TRANSFORMATIONS

When discussing positions, velocities, and attitudes of orbiting spacecraft, one must have a suitable reference system defined. Although useful, the classical orbital elements are not the only way to compute orbital propagation. In some instances, the spacecraft position ($\vec{r}$) and velocity ($\vec{v}$) in the ECI frame may be used for translational computations. The use of the right-handed ECI frame is intuitively satisfying since it is centered upon the primary object of interest, the Earth. However, the ECI coordinate is independent of the spacecraft and its orbit and does not meet all of our needs. For instance, on-board spacecraft sensors are not centered in the ECI frame.

1. Earth and Body Centered Reference Frames

There are several coordinate systems which are based on the plane of the spacecraft orbit. These typically use the spacecraft orbit as the fundamental plane, and are denoted by the first two axes in a right-handed coordinate system. As an example, one such system with uses characteristics of both the ECI and classical orbital elements is the perifocal coordinate system (P,Q,W), as shown in Figure 3.3. In the perifocal coordinate system, the Earth is the origin, the P-axis points toward the orbital perigee, the Q-axis is 90° ahead in the orbital direction, and the W-axis is normal to the plane of orbit. The P-axis and Q-axis are in the orbital plane and orientated based on the argument of perigee. If the eccentricity changes and the perigee rotates, then the PQW coordinate system will also change. The inclination tilts the PQW coordinate system by tilting the orbital plane. Although, useful in some situations, the PQW coordinate system is not always well defined. As for all systems based on classical orbital parameters, circular orbits in the equatorial plane require special rules. This is due to argument of perigee and semi-major axis not being defined for a circular orbit, and the longitude of ascending node not being defined for an equatorial orbit. Since the ECI and PQW coordinate systems centered at the same origin, the transformation from one system to the other is purely a matter of rotation.

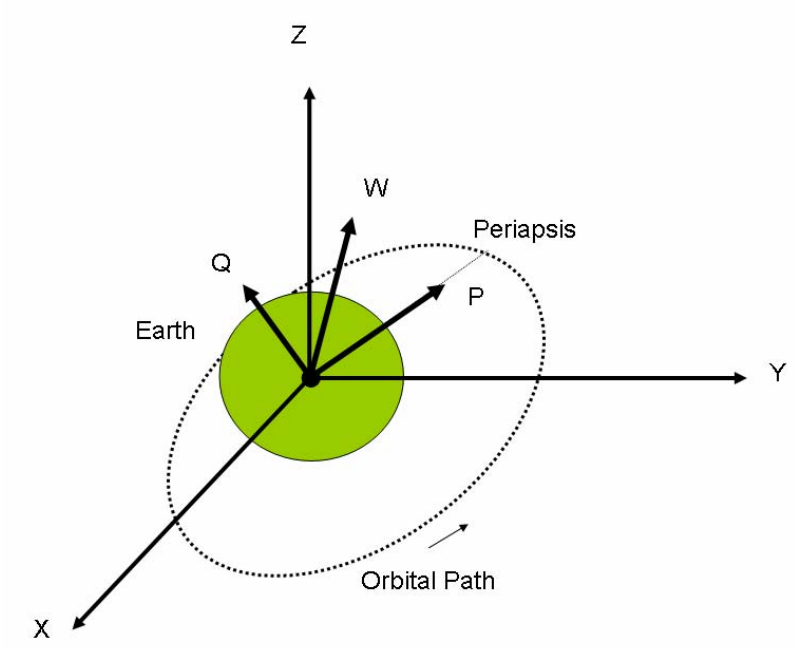

Figure 3.3 Perifocal and ECI Coordinate Systems.

There are numerous situations when a reference frame centered on the spacecraft or its relative motion is needed. Each spacecraft, assumed to be a rigid-body, has a body fixed-fixed reference frame centered at the spacecraft's center of mass. This spacecraft body frame (X_B, Y_B, Z_B) is typically aligned with the principle axis of inertia. The spacecraft body frame is not inertial and is free to rotate as the spacecraft rotates along any of its three axes. The rotation around the body axes are represented by three angles: roll (φ), pitch (θ), and yaw (ψ) as shown in Figure 3.4. This frame is useful when determining the spacecraft attitude and rotation rates, similar to the reference frame used for aircraft.

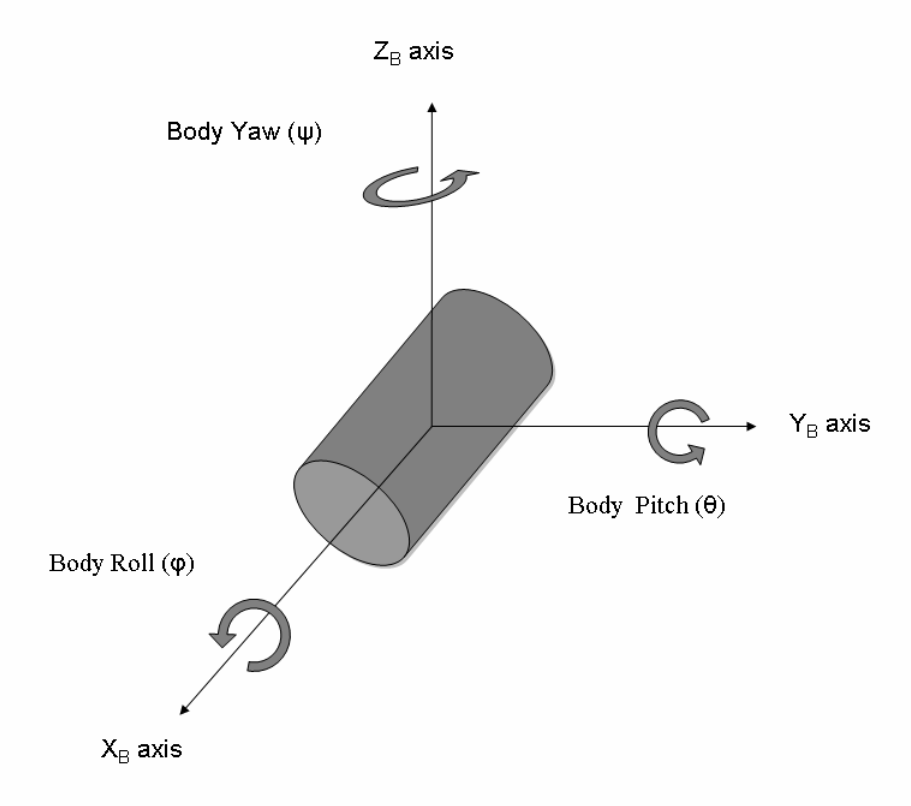

Figure 3.4 Body-fixed Coordinate System.

Of particular interest is the spacecraft coordinate system (R,S,W) which is used to determine relative motion between objects in orbit. The spacecraft coordinate system is aligned with the R-axis along the radial direction from the Earth to the spacecraft, the S-axis is along the direction of the spacecraft translational track, and the W-axis is cross-

track (normal) to the orbital plane. One possible orientation of the spacecraft, , as shown in Figure 3.5, is with the R-axis aligned opposite of body yaw, S-axis aligned with body roll, and W-axis aligned opposite the body pitch axis. In Figure 3.5, the spacecraft coordinate system W-axis points directly up from the page and the body-fixed Y_B axis points directly down into the page. As with the ECI and PQW coordinate systems, the body-fixed and RSW coordinate systems are centered at the same origin. Therefore, transformation from body-fixed to RSW is purely a matter of rotation. It is worth noting that the velocity vector is only aligned with the S-axis when the orbit is perfectly circular, and at the apogee and perigee of elliptical orbits. The angular difference from the local horizontal (line perpendicular to the radial vector) and the velocity vector and is usually referred to as the flight path angle.

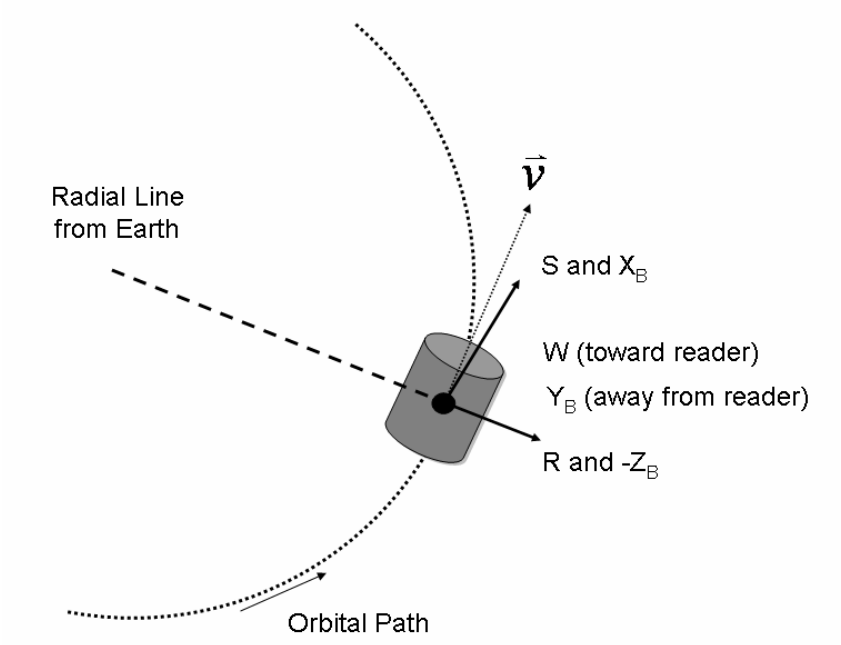

Figure 3.5 Body-fixed and RSW Coordinate Systems.

The RSW and body-fixed reference frames are not inertial. The spacecraft coordinate system orbits along the orbital path in the ECI reference frame, as shown in Figure 3.6.

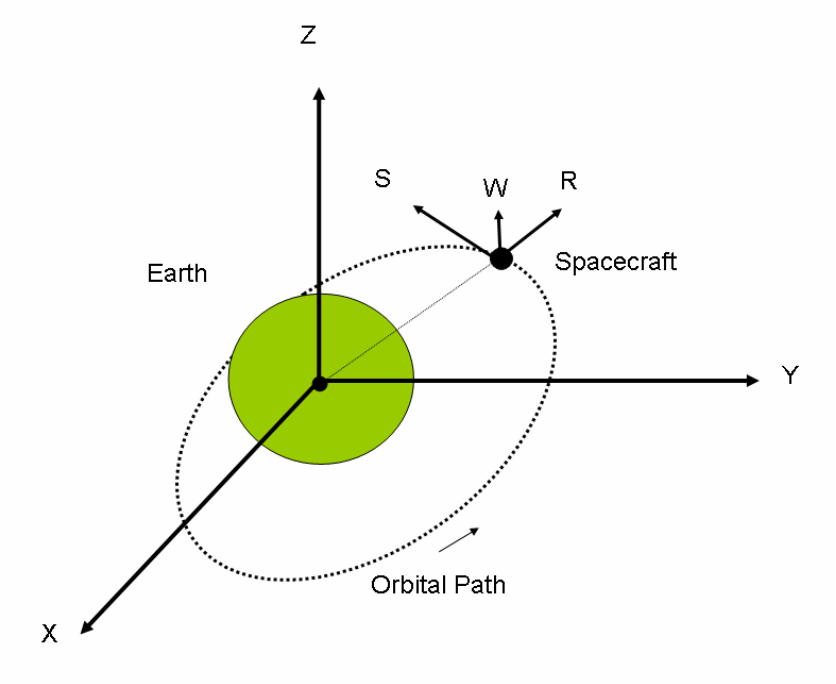

Figure 3.6 ECI and Spacecraft Body Coordinates.

For spatial ranging and positioning information, each spacecraft will have on-board sensors and receivers. These sensors are likely to include Global Positioning System (GPS) receivers and directional range finders (e.g., sonar, laser, or infra-red) to determine distances away from the desired location and from other objects. Most payload sensors, with the exception of GPS receiver data, give information relative to their position on-board the spacecraft. Figure 3.7 shows an elliptical, equatorial orbit with the spacecraft coordinate W-axis and ECI Z-axis both pointing directly out of the page. In a 2D or 3D space, r is the actual geometric distance between objects in the space, as computed in Equation (3.3). In order to transform data from the RSW to the ECI coordinate system both a rotation and translation is required.

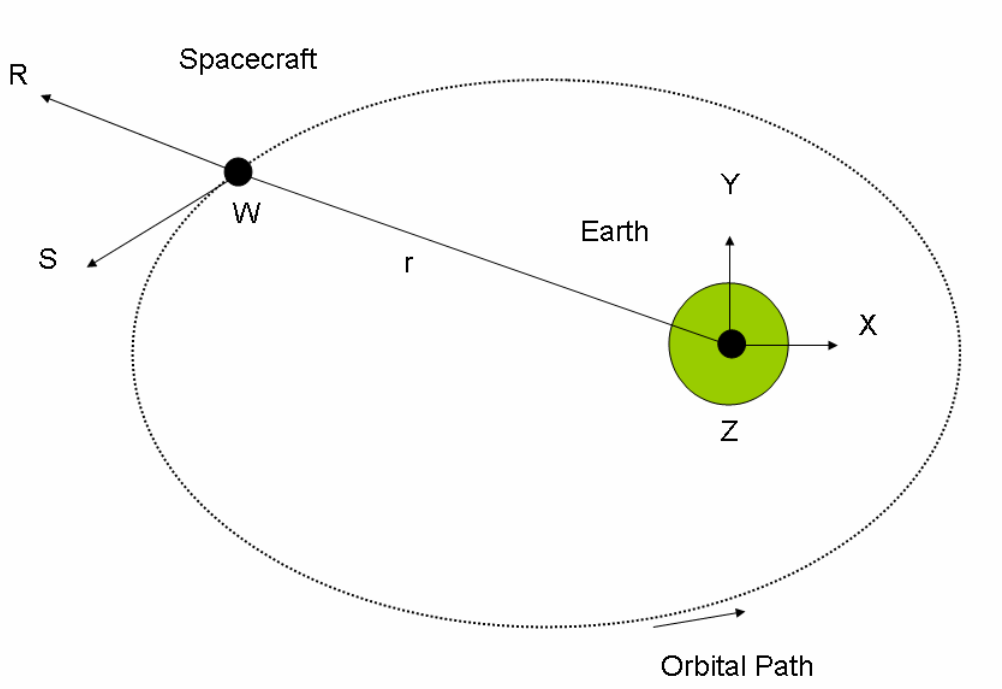

Figure 3.7 Euclidean Distance.

2. Transformations

Transforming between two different reference frames with the same origin requires a rotation. For reference frames that do not have the same origin, the axis can be aligned by rotation and then translated with vector addition. For Cartesian 3D systems there are three degrees of freedom for angular displacement and one degree of freedom along each axis, resulting in nine element matrix with six relationships between them. The coordinate transformation matrix (C_{DCM}), also referred to as a Direction Cosine Matrix (DCM) [34], is represented by

$$C_{DCM} = \begin{bmatrix} c(\theta)c(\psi) & c(\theta)s(\psi) & -s(\theta) \\ s(\varphi)s(\theta)c(\psi) - c(\varphi)s(\psi) & s(\varphi)s(\theta)s(\psi) + c(\varphi)c(\psi) & s(\varphi)c(\theta) \\ c(\varphi)s(\theta)c(\psi) + s(\varphi)s(\psi) & c(\varphi)s(\theta)s(\psi) - s(\varphi)c(\psi) & c(\varphi)c(\theta) \end{bmatrix} \quad (3.10)$$

where the angles of yaw (ψ), pitch (θ), and roll (φ) correspond to the rotation described in the body-fixed reference frame, c(--) is the cosine function, and s(--) is the sine function. The order of angular rotation is important, since different order of rotation will result in a different transformation matrix. The C_{DCM}, shown in equation (3.10), is for a

yaw, pitch, roll sequence. There are 11 other possible sequences and resulting transformation matrixes. The C_{DCM} matrix is useful for physical insight of the transformation, but it is not efficient for computation due to the trigonometric functions.

The most used parameterization for spacecraft attitude transformations are quaternions. The definition of quaternion is based on Euler's theorem which states that the general displacement of a rigid body with one fixed point is a rotation about a fixed axis. This axis is called the eigenaxis or Euler axis ($\vec{e}_{axis}$). The angle of rotation is called the Euler angle or the principal Euler angle (α). The Euler axis is the eigenvector of the rotation matrix associated with the eigenvalue 1. Thus, every rotation matrix has an eigenvalue that is equal to positive one [35]. This fact justifies the term eigenaxis for the Euler axis vector, denoted by

$$\vec{e}_{axis} = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} \tag{3.11}$$

It is useful to compute Euler angles from a given rotation matrix, since there is a need to be able to compute $\vec{e}_{axis}$ and α. The Euler parameter set, also known as a quaternion, is a four-parameter set with some computational efficiency and singularity advantages over the Euler angles set. Although useful, it may be difficult to visualize the physical meaning of the quaternion for most applications. The quaternion vector ($\vec{q}$) is a 3 x 1 [34], represented by

$$\vec{q} = \begin{bmatrix} e_1 \sin(\alpha/2) \\ e_2 \sin(\alpha/2) \\ e_3 \sin(\alpha/2) \end{bmatrix} \tag{3.12}$$

The quaternion (q) is a 4 x 1 algebraic vector [34], with a scalar component as the fourth component

$$q = \begin{bmatrix} q_1 \sin(\alpha/2) \\ q_2 \sin(\alpha/2) \\ q_3 \sin(\alpha/2) \\ \cos(\alpha/2) \end{bmatrix} \tag{3.13}$$

The spacecraft coordinate frame is not inertial, therefore it is moving with respect to the ECI frame. The kinematics of a non-inertial frame can be represented by the following equation for the velocity of the spacecraft with respect to the ECI coordinate system

$$\vec{v}_{ECI} = \dot{\vec{r}} + \vec{\omega}_{BN} \times \vec{r} \tag{3.14}$$

The acceleration of the spacecraft with respect to the ECI coordinate system, can be determined by differentiating equation (3.14),

$$\vec{a}_{ECI} = \ddot{\vec{r}} + 2(\omega_{BN} \times \dot{\vec{r}}) + (\dot{\omega}_{BN} \times \vec{r}) + \omega_{BN} \times (\omega_{BN} \times \vec{r}) \tag{3.15}$$

These results can be generalized with respect to any two reference frames, and will be used to derive the equations for the relative motion between two spacecraft, refer to Chapter IV.

D. ATTITUDE DYNAMICS

1. Spacecraft Rotational Dynamics

Applying Newton's Second Law to rotational dynamics, the torque on a system is defined by

$$\vec{T} = \dot{\vec{H}} = J\,\vec{\alpha} \tag{3.16}$$

The angular momentum is described by

$$\vec{H} = J\,\vec{\omega} \tag{3.17}$$

The Inertia matrix of a 3D system is a three by three matrix,

$$J = \begin{bmatrix} J_{xx} & J_{xy} & J_{xz} \\ J_{yx} & J_{yy} & J_{yz} \\ J_{zx} & J_{zy} & J_{zz} \end{bmatrix} \tag{3.18}$$

where the diagonal terms are Moments of Inertia and the off-diagonal terms are Products of Inertia. The inertia matrix is positive definite and symmetrical, with the Moments of

Inertia always positive or zero. The primary axes of the body can be defined along the principal axis, which are defined by the minimum and maximum moments of inertia.

The Euler Equation for rotation follows the form of equation (3.14), and describes the torque in each axis [29] as

$$\dot{\vec{H}} + \vec{\omega} \times \vec{H} = \vec{T} \tag{3.19}$$

For the system with the origin of the body frame at the center of mass, the Euler Equations [29] can be described by

$$\begin{aligned} J_x \dot{\omega}_x + (J_z - J_y)\,\omega_z\,\omega_y &= T_x \qquad (a) \\ J_y \dot{\omega}_y + (J_x - J_z)\,\omega_x\,\omega_z &= T_y \qquad (b) \\ J_z \dot{\omega}_z + (J_y - J_x)\,\omega_y\,\omega_x &= T_z \qquad (c) \end{aligned} \tag{3.20}$$

These three first-order differential equations are in components along the body axes of the angular velocity of the body frame with respect to the inertia frame. They are non-linear and coupled, so they are usually solved by numerical integration. The angular rate of the spacecraft body with respect to the initial frame is given by

$$\vec{\omega}_{BN} = \vec{\omega}_{BO} + \vec{\omega}_{ON} \tag{3.21}$$

where $\vec{\omega}_{BO}$ is the angular velocity of the spacecraft body frame with respect to the orbital reference frame and $\vec{\omega}_{ON}$ is the angular velocity of the orbital reference frame with respect to the inertial reference frame. For a circular orbit with small angles and angular rates, the angular velocity of the spacecraft with respect to inertial is

$$\begin{aligned} \omega_x &= \dot{\alpha}_x - \omega_{ON}\,\alpha_z \qquad (a) \\ \omega_y &= \dot{\alpha}_y - \omega_{ON} \qquad (b) \\ \omega_x &= \dot{\alpha}_z + \omega_{ON}\,\alpha_x \qquad (c) \end{aligned} \tag{3.22}$$

2. Gravity Gradient

Gravitational forces acting upon an orbiting spacecraft depend on the mass distribution of the body. Since most spacecraft are not perfectly symmetrical, the gravitational forces on a spacecraft are not uniform and result in a disturbance torque around the spacecraft's center of mass. The gravity gradient tends to align the

spacecraft's minimum axis of inertia with the radial vector from the Earth. For example, a pencil orbiting in space will tend to align with the point toward the Earth. The Gravitational Gradient (GG) torque can be expressed along the principle axes of the body frame as

$$T_{GGx} = \left(\frac{3\mu}{r^3}\right)(J_z - J_y)\, c_2\, c_3 \qquad (a)$$

$$T_{GGy} = \left(\frac{3\mu}{r^3}\right)(J_x - J_z)\, c_1\, c_3 \qquad (b) \qquad (3.23)$$

$$T_{GGz} = \left(\frac{3\mu}{r^3}\right)(J_y - J_x)\, c_1\, c_2 \qquad (c)$$

where c_1, c_2, and c_3 are the direction cosines of the radius vector with respect to the spacecraft's body frame. The GG torque is perpendicular to the local vertical. Therefore, the GG torque only has components in the roll and pitch axis. In fact, if one of the principle axes is aligned with the local vertical, then the GG torques is null. The GG torque can be used as a simple control method in order to stabilize a spacecraft attitude.

E. ORBITAL PERTURBATIONS

Non uniform Earth mass distribution and non spherical Earth shape, atmospheric drag on the spacecraft, third-body (Sun and Moon) forces, and solar-radiation pressure act as disturbances on a spacecraft Keplerian orbit [32]. The significance of these forces often depends on spacecraft size, position and altitude. In this research, interest is in relatively small spacecraft (less than 400 kg) at low orbital altitudes (less than 3,000 km). These conditions determine which perturbation forces have the largest influence on the spacecraft orbit. For relative low latitudes, the Earth's oblateness and atmospheric drag are more dominant. Also of importance is the time duration of interest. Some of these force are of great significance over the course of a satellites lifetime, but negligible for the duration of a single orbit. Simulations will include perturbations when practical based on the time duration and spacecraft characteristics and orbit.

1. Non-Symmetrical Earth

The Earth is not symmetrical in its overall shape and mass distribution. This results in the gravitational field also not being symmetric. The Earth's oblateness, or equatorial bulge, can be described by the zonal harmonics coefficients in conventional Legendre polynomials, refer to [32] for detailed discussion. The first three zonal coefficient terms in the Legendre polynomial are J2, J3, and J4, where $J2 = 1.08262668355 \times 10^{-3}$ is the second-order zonal harmonic. The third-order zonal harmonic is $J3 = 2.53265648533 \times 10^{-6}$ and the fourth-order zonal harmonic is $J4 = 1.08262668355 \times 10^{-3}$. J2 is the equatorial bulge term which has the most significant effect on spacecraft orbits. J2 effects are often classified as short period oscillations, while J4 results in long period variations.

In this work, the EGM-96 coefficients and WGS-84 reference shape are used for calculations, refer to [32] for further details. Various, detailed Earth models are available through applications, such as STK [36]. Higher order polynomials will not be discussed in this research, although may be included in imported dynamic models.

2. Atmospheric Drag

The density of particles in the Earth's atmosphere is variable and changes due to solar interaction and magnetic field influences. Particles in the altitude of the spacecraft's orbit act upon the body of the spacecraft and slow it down. Atmospheric drag can impart both a translational disturbance force and rotational (attitude) disturbance torque. The basic equation of aerodynamic drag [32] is

$$\vec{a}_{drag} = -\frac{1}{2}\left(\frac{c_D A_s}{m_s}\right)\rho \vec{v}_{rel}^{\,2}\left(\frac{\vec{v}_{rel}}{|\vec{v}_{rel}|}\right) \tag{3.24}$$

where c_D is the spacecraft drag coefficient, $\vec{v}_{rel}$ is the spacecraft velocity vector with respect to the atmosphere, A_s is the spacecraft's cross-sectional area normal to its velocity vector, m_s is the spacecraft's mass, and ρ is the atmospheric density. The velocity vector relative to the Earth's rotating atmosphere is

$$\vec{v}_{rel} = \dot{\vec{r}} - \left(\vec{\omega}_e \times \vec{r}\right) \tag{3.25}$$

where $\vec{\omega}_e$ is the angular rotation vector of the Earth.

Comparable to the gravity models, numerous atmospheric models are available for reference. In this work, an exponential atmospheric density model based on the U. S. Standard Atmosphere (1976) was used [32]. This model, adopted by the U. S. Committee on Extension to the Standard Atmosphere (COESA), should be sufficient for LEO orbits with orbital altitudes between 100 km and 1,000 km [32].

3. Third Body Effects (Sun and Moon)

The gravitation attraction from other bodies, such as the Sun and Moon, also affects spacecraft orbital Keplerian motion. The three body equation of motion for the relative acceleration of a spacecraft in the Earth's inertial frame [32] is

$$\vec{a}_{3body} = -\frac{\mu}{r^2}\left(\frac{\vec{r}}{r}\right) + \mu_m\left(\frac{\vec{r}_{sm}}{r_{sm}^{\ 3}} - \frac{\vec{r}_{em}}{r_{em}^{\ 3}}\right) + \mu_S\left(\frac{\vec{r}_{sS}}{r_{sS}^{\ 3}} - \frac{\vec{r}_{eS}}{r_{eS}^{\ 3}}\right) \tag{3.26}$$

where $\vec{r}_{em}$ is the relative distance from Earth to Moon, $\vec{r}_{eS}$ is the relative distance from Earth to Sun, $\vec{r}_{sm}$ is the relative distance from spacecraft to Moon, $\vec{r}_{sS}$ is the relative distance from spacecraft to Sun, $\mu_m = 4902.798882 \times 10^6 m^3 s^{-2}$ is the Gravitational Parameter of Moon, and $\mu_S = 13271.2428 \times 10^{13} m^3 s^{-2}$ is the Gravitational Parameter of Sun. If the terms in equation (3.26), especially with respect to the Sun, are too small for numerical precision then the acceleration can be approximated using Taylor series expansion [32].

4. Solar-Radiation Pressure

The radiation being emitted by the Sun exerts a force on the spacecraft, somewhat like the wind on a sail. The magnitude and direction of the solar-radiation force is dependent on several factors, such as the position of the Sun relative to the spacecraft, the attitude and shape of the spacecraft, the intensity of the solar-radiation and the reflectivity

of the spacecraft [32]. Solar pressure can impart both a translational disturbance force and rotational (attitude) disturbance torque. For this research, the spacecraft is treated as a "black body," which absorbs all radiation.

In general, the force of solar pressure per unit area on a spacecraft orbiting Earth is $S_p = 4.51 \times 10^{-6} N/m^2$. A small spherical spacecraft at lower altitudes will be influence less by this solar-radiation force. There may even be durations of the orbit where the spacecraft may be in eclipse and does not experience this effect. Although not expanded here, more precise equations for the force can be developed using the reflectivity of the spacecraft and the exposed area of the spacecraft.

5. Thrust

Thrusting, and any other control actuator output, from the spacecraft can be considered as a perturbation to the orbit. The thrust is the result of a controller's command to the propulsion system, in order to attain a desired orbit or attitude. The force of the thrust is

$$F_{thrust} = I_{sp}\left(\frac{dm}{dt}\right) \tag{3.27}$$

where F_{thrust} is the force of spacecraft motor, I_{sp} is the specific impulse of the engine, and $\frac{dm}{dt}$ is the spacecraft motor mass flow rate. The change in velocity, Δv, of the spacecraft due to the thruster firing is

$$\Delta v = g\, I_{sp} \ln\left(\frac{m_s}{m_s - m_p}\right) \tag{3.28}$$

where g = 9.80665 m/s² is the gravitational acceleration at the Earth's surface, and m_p is the mass of the burnt propellant. The minimum and maximum Δv are determined based on the duration of the thruster firing. For a given thruster force, the Δv can be approximated as

$$\Delta v = \frac{F_{thrust}\,\Delta t}{m_s} \tag{3.29}$$

Thruster sizing for a spacecraft depends on mission and orbit. Station keeping, drag compensation and three axis stabilization requirements all contribute to the total Δv. In this work, each thruster is assumed to be hydrazine fueled with the compiled parameters listed in Table 3.1 [37]. The thruster(s) are assumed to be directional and mapped to deliver the commanded Δv in the body frame of the spacecraft.

Thruster Parameter	Thruster Range	Simulation Baseline
Propellant Type	Hydrazine	Monopropellant Hydrazine
Force	0.5 - 2 N	1 N
Specific Impulse (I_{sp})	50 - 200 sec	200 sec
Steady State Impulse	50 - 200 sec	200 sec
Min pulse duration	0.01 - 0.5 sec	0.05 sec
Min Δv	0.01 - 5 mm/s	0.0005 m/s
Max total Δv	10 - 800 m/s	60 m/s
Mass	3-7% of Spacecraft Mass	3% of Spacecraft Mass

Table 3.1 General Thruster Parameters.

An on-off thrust profile is never ideal, due to variations in ignition, tail-off, mass flow rate, and specific impulse [32]. These variations can be managed by applying thrust modulation methods. The pulse width (duration of fire) and pulse frequency (rate of fire) can be used to limit the hysteresis, or dead zone lag. Using pulse width pulse frequency (PW-PF) modulation an approximately linear response can be achieved from on-off thruster.

F. SPACECRAFT CONTROL SYSTEM

The spacecraft control system, referred to as the Attitude and Orbital Control Subsystem (AOCS) [27], is based on the spacecraft's kinematics and dynamics. Determining the actual position and velocity of the spacecraft in an orbit is the function of the Navigation, Guidance, and Control (NGC) subsystem [27]; whereas, determining

attitude is the function of the Attitude Determination Control Subsystem (ADCS) [27]. A typical control system block diagram is shown in Figure 3.8. If the attitude of the spacecraft is not important then the spacecraft can be treated as a point, or sphere, mass. However, most payloads require a specific pointing direction for their sensors, so the attitude of the spacecraft must be maintained. As the spacecraft location converges, the attitude of the spacecraft becomes more significant. For instance, if two spacecraft are to dock then their docking mechanisms must be aligned. The station keeping and attitude control (pointing) accuracy of spacecraft often vary significantly and are dependent on their sensors and actuators. Spacecraft attitude sensors include, earth, Sun, and star sensors, gyroscopes, magnetometers, and differential GPS receivers. Typical attitude actuators are divided into two categories: passive and active. Passive actuators include gravity gradient stabilization, spin stabilization, and dampers. Active actuators consist of thrusters, magnetic torquers, and momentum control devices (e.g., momentum wheels, reaction wheels, and control moment gyroscopes).

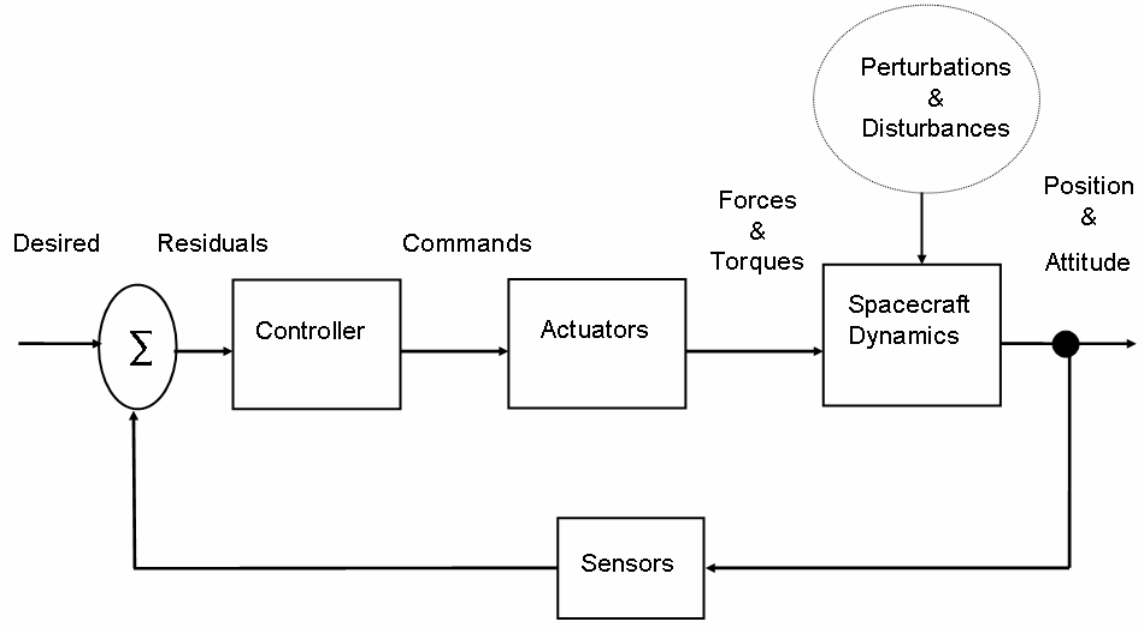

Figure 3.8 Control System (adapted from [27]).

1. Translational Control

Translational control is the major theme of this dissertation, with translational control achieved by position and velocity state feedback. The translational control is

discussed in detail in Chapters V, VI, and VII. Chapter V is an overview of spacecraft translational control concepts. Chapter VI introduces the specific APF control algorithm and Chapter VII discusses the LQR/APF control algorithm.

Thruster firings, or any mass expelling system, can be used to propel a spacecraft and guide its center of mass position and velocity in space. The primary kinds of thrusters used on spacecraft are cold gas, hot gas and electric. They each have torque and specific impulse limitations, based on equation (3.28), as

$$I_{sp} = \frac{T}{\dot{m}\, g} \tag{3.30}$$

The propulsion is primarily dependent on the mass flow rate. The advantage of using thrusters is that they can be used to torque about any axis in any orbit. However, thrusters generally have on-off limitations which may result in sensitivity to chattering, dead zones, and hysteresis issues. These issues are of particular concern in the final stages of docking, due to the need for precise control. These challenges can be overcome by modulating the thruster firing duration and frequency. But these modulation schemes add to the overall complexity of the control law being used.

Also, various sizes and numbers of thrusters can be positioned at desired locations on the spacecraft. Smaller thrusters are used for fine adjustments and larger thrusters are used for larger maneuvers. In spacecraft motion thrusters are typically used to impart a Δv in the desired direction and then another Δv in the opposite direction once the spacecraft is close to the desired spatial region. This second thruster firing serves as a braking maneuver, such as a car approaching a stop sign. For short duration firings, thruster firings can be treated as generating instantaneous velocity variation of the spacecraft, not changing its position.

2. Attitude Control

Attitude control was achieved via quaternion feedback commands to three axis aligned reaction wheels with magnetotorqers for momentum damping. The non-linear quaternion feedback control law utilized is based upon on attitude error described by the

quaternion and the angular rate (body with respect to inertial). The commanded torque due the quaternion error and angular rate is

$$T_{CMD} = -k_p \, \vec{q}_E - k_d \, \vec{\omega}_{BN} \tag{3.31}$$

where k_p and k_d are the proportional and derivative feedback control gains, respectively. These gains were tuned in order to get a damped attitude response with an actuation limited maximum torque. Of course, the maximum torque performance is dependent on the attitude control actuator. Additionally, quaternion rotation logic is necessary to ensure minimum rotations are commanded. Since the quaternion represents a vector in a 4-D unit sphere with an axis and an angle, it is possible for two different quaternions to represent the same orientation. In particular if the opposite eigenaxis is considered the angle rotation is 360 degrees less, represented as

$$q = \begin{bmatrix} \vec{q} \\ \cos(\alpha/2) \end{bmatrix} \; or \; q = \begin{bmatrix} -\vec{q} \\ \cos(\alpha/2) - 2\pi \end{bmatrix} \tag{3.32}$$

The quaternion with the positive fourth element represents the shortest rotation, and its complement is the longest. Selection of quaternion with minimum angular rotation is generally desired for efficient attitude control.

Momentum Exchange Devices (MED), such as, momentum wheels, reaction wheels, and control moment gyros, are commonly used for providing control torque. The general attitude dynamics for a spacecraft with a MED is

$$J \, \dot{\vec{\omega}}_{BN} + \vec{\omega}_{BN} \times J \, \vec{\omega}_{BN} = \vec{T} + \vec{T}_{MED} \tag{3.33}$$

which follows from Equation (3.17) and Equation (3.19). The spacecraft torque due to the MED ($\vec{T}_{MED}$) is described by

$$\vec{T}_{MED} = -\left(\left(\vec{\omega}_{BN} \times Z \, \vec{h} \right) + \left(Z \, \dot{\vec{h}} \right) + \left(\dot{Z} \, \vec{h} \right) + \left(\dot{J} \, \vec{\omega}_{BN} \right) \right) \tag{3.34}$$

where $\vec{h}$ is the angular momentum of the MED and Z represents the axis about which the MED spins. The mass of the spinning wheel acts as inertia to the spacecraft system and provides disturbance rejection in the perpendicular axis.

Each of the three primary types of MED work in a slightly different manner. The simple momentum-wheel is a fixed-axis wheel spinning at a large angular rate. This is the most simple of the MED, due to the constant mass, fixed axis, and reasonably constant spin rate. For instance, a torque due to momentum wheel rotating about the spacecraft's pitch axis is

$$\vec{T}_{mw} = -\left(\vec{\omega}_{BN} \times \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} h_{mw} \right) - \left(\begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} \dot{h}_{mw} \right) \tag{3.35}$$

where the last two terms on the right side of Equation (3.34) are now zero. This can be simplified for a single momentum wheel, by substituting equation (3.22) into equation (3.35), as

$$\vec{T}_{mw} = \begin{bmatrix} -\left(h_{mw}\,\dot{\alpha}_z\right) - \left(h_{mw}\,\omega_{mw}\,\alpha_x\right) \\ \dot{h}_{mw} \\ \left(h_{mw}\,\dot{\alpha}_x\right) - \left(h_{mw}\,\omega_{mw}\,\alpha_z\right) \end{bmatrix} \tag{3.36}$$

The reaction wheel is more commonly used. The reaction wheel is a fixed wheel that varies its spin to deliver desired torque. Three reaction wheels are typically aligned along the principle axes along with three magnetotorquers. The orientation of the reaction wheel, Z, is a constant, so $\dot{Z} = 0$ and the $\dot{Z} * h$ term in (3.34) can be dropped. For a typical spacecraft configuration, with three reaction wheel along each axis and rotating in the same direction:

$$Z_{RW} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.37}$$

with the spin direction defined to be with respect to each axis. Mass of the spacecraft is still considered relatively constant, so $\dot{J} = 0$ and the $\dot{J} * \vec{\omega}_{BN}$ term can be dropped. The angular rate of the reaction wheel is variable and therefore $\dot{h} \neq 0$ (different than momentum wheel). Therefore, the general control law for the reaction wheel can be reduced to

$$T_{RW} = -(\omega_{BN} \times Z_{RW} h_{RW} + Z_{RW} \dot{h}_{RW}) \tag{3.38}$$

which can be slightly expanded as follows

$$T_{RW} = -\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} h_{RW} - \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \dot{h}_{RW} \tag{3.39}$$

It is useful to note that that the rate-of-change of the wheel momentum is as follows

$$\dot{h}_{RW} = Z_{RW}^{-1}(-T_{CMD} - \omega_{BN} \times Z_{RW} * h_{RW}) \tag{3.40}$$

Equation (3.40) is used to describe the dynamics of the reaction wheel. The commanded torque is designed to saturate at the maximum torque available to the reaction wheel along each axis.

The magnetotorquers may be used to de-saturate the reaction wheels, as necessary. The reaction wheel momentum is multiplied by a gain, k_{mt}, in order to determine the torque required, T_{REQ}, by the magnetotorquers control law,

$$T_{REQ} = -k_{mt} h_w \tag{3.41}$$

This T_{REQ} is used to determine the magnetic dipole momentum, M, of each magnetotorquer, as

$$M = \frac{B_B \times T_{REQ}}{(B_B \bullet B_B)} \tag{3.42}$$

where B_B is the magnetic field in the body frame. The maximum dipole moment of each torquerod, M, is limited. Also, the pitch axis is not torquable at 11.7 degree orbital inclination since the spacecraft is at the magnetic equator. At the magnetic equator the pitch axis is aligned with the magnetic field and can not create torque along that axis. With the limited dipole moment and the magnetic filed with respect to the body frame, the de-saturation torque generated by the magnetotorquers is given as

$$T_{DAMP} = M \times B_B \tag{3.43}$$

This torque tends to align the dipole with the magnetic field.

Each spacecraft model will be assumed to be equipped with an attitude control system. The spacecraft attitude control actuator and damper reactions will be based on reasonable attitude control systems for small spacecraft. Standard spacecraft attitude and rate sensors will also be assumed, with typical statistical sensor noises. Development of specific sensors and actuators is beyond the scope of this research.

It is worth noting that although thrusters are the most common actuators to control spacecraft translational motion they can also be used for external spacecraft torque. The thrusters are usually positioned at equal and opposite radial distances from the center of mass in order to create torque. Various sizes and number of thrusters can be used, but they usually range from four to twelve per spacecraft [29]. Thrusters can also be used to supplement momentum exchange devices, by desaturating spin rates and changing momentum by providing external spacecraft torque.

G. HIGH FIDELITY SIX DOF SPACECRAFT MODEL

The characteristics of each of the multiple spacecraft in the group are assumed to be the same unless explicitly stated in the simulation results. The spacecraft altitudes are limited to LEO orbits in the range of 300-2,000 km. The distance between the Target and Chaser spacecraft initial positions is limited to less than 1.0 km in RSW coordinates. The basic characteristics of the spacecraft considered in this research are listed in Table 3.2. These characteristics were selected to represent realistic and current operational capabilities of a small spacecraft. Rules for spacecraft and subsystem sizing were determined based on design rules of thumb [37]. For instance, the spacecraft volume was selected to be approximately $1/100$ of the spacecraft mass [37]. The center of mass of the spacecraft is assumed to be located at the geometric center. Position, attitude, and ranging sensors are assumed to provide near ideal information. Translational motion is conducted via six cold gas pulsed thrusters with a maximum thrust of $1.0\,N$. The attitude actuators are three reaction wheels along each spacecraft axis, along with magnetotorquers to dump the reaction wheel momentum. Disturbance perturbations are adjusted as necessary for model and simulations development. The attitude actuator

specifications are loosely based on the Honeywell Model HR 0610 Reaction Wheel [38]. The translations thrust actuators were generalized from cold gas thruster performance and operating characteristics; refer to [37].

Spacecraft Physical Characteristics	Length and Width	1.0 m
	Height	1.0 m
	Mass	100 kg
	Moment of Inertia about X	16.67 - 50 kg m^2
	Moment of Inertia about Y	16.67 - 50 kg m^2
	Moment of Inertia about Z	16.67 - 50 kg m^2
Spacecraft Actuators	Number of Thrusters	1-6
	Mass of Propellant (% Mass)	3-6%
	Max Thrust (each thrusters)	1.0 N
	Number of Reaction Wheels (RW)	3
	RW Max Torque (each axis)	0.055 N m
	RW Max Angular Momentum	4-12 N m s
	Initial Angular Rate of RW	0 RPM
	Inertia about RW spinning axis	0.14258 Kg m^2
	Number of Magnetotorquers	3
	Max dipole moment (each torquerod)	100 A m^2

Table 3.2 Main Characteristics of Chaser Spacecraft Simulators.

Initially these spacecraft characteristics were used to model spacecraft using *The MathWorks Incorporated* products MATLAB and Simulink [39]. The spacecraft characteristics may be further expanded and refined as hardware-in-the loop structures are developed at the NPS Spacecraft Servicing and Robotics Laboratory. Mass and dimensions greatly affect both the perturbations on the spacecraft and the commanded actuation by the control algorithm. Variations in the moment of inertia for three basic spacecraft shapes, spherical, cylindrical, and cubic, are considered. Mass changes due to thruster firings are modeled with appropriate moment of inertia changes. In the absence of actual sensor readings and actuators responses, equivalently accurate input and outputs are provided into the simulations. For instance, attitude determination sensor reading, such as star trackers, can be simulated based on orbital propagation calculations. On the other hand, actual GPS reading in the laboratory may be used to determine spacecraft positioning on the Autonomous Docking and Spacecraft Servicing testbed.

IV. SPACECRAFT ORBITAL RELATIVE MOTION

A. EQUATIONS OF RELATIVE MOTION

As two spacecraft obit around the Earth their relative positions change based on the characteristics of their orbits. Two different orbits are shown in Figure 4.1. This view shows a circular and an elliptical equatorial orbit from the perspective of high above the North Pole. The difference in inertial position and velocity between spacecraft orbits, appears as dynamic relative motion between orbital spacecraft.

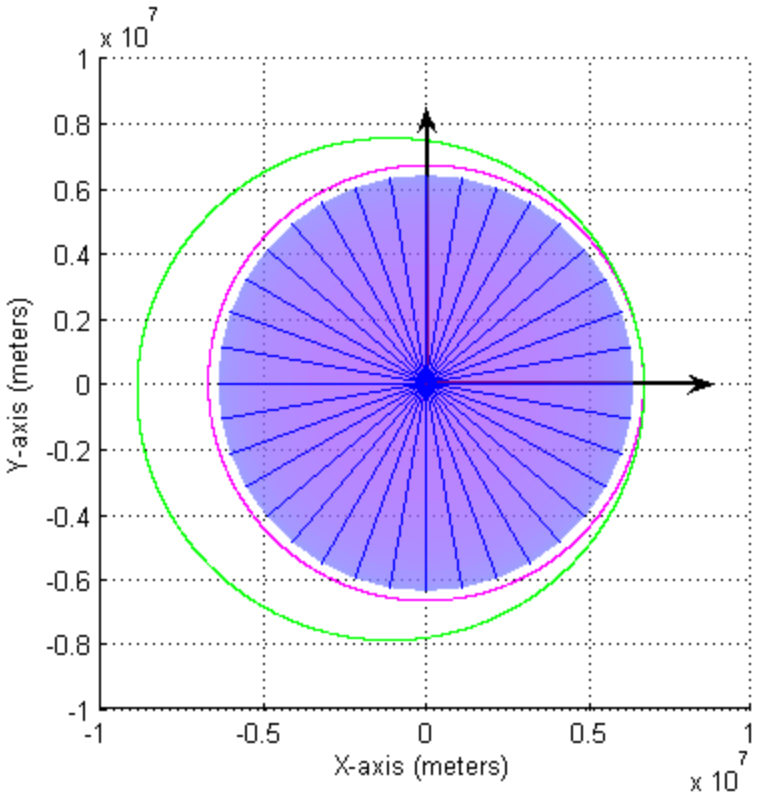

Figure 4.1 Two Spacecraft Orbits in 2D.

In order to establish the equations of motion between spacecraft consider one of the spacecraft as the primary and all of others as secondary. The primary spacecraft will be denoted as the Target (t). The secondary spacecraft will each be denoted as Chasers (c) with numerical designations to distinguish them apart. The Target and Chase

spacecraft reference is used during the rendezvous maneuver, when the Target spacecraft maintains its orbit and the Chase spacecraft maneuvers to the Target's position.

The relative motion can be developed from the position vectors of the two spacecraft. From Figure 4.2, it is apparent that vector addition gives

$$\vec{r}_c = \vec{r}_t + \vec{r} \tag{4.1}$$

where $\vec{r}_t$ and $\vec{r}_c$ are the position vectors of the Target and Chase spacecraft in the ECI coordinates and $\vec{r}$ is the relative position of the Chase with respect to the Target.

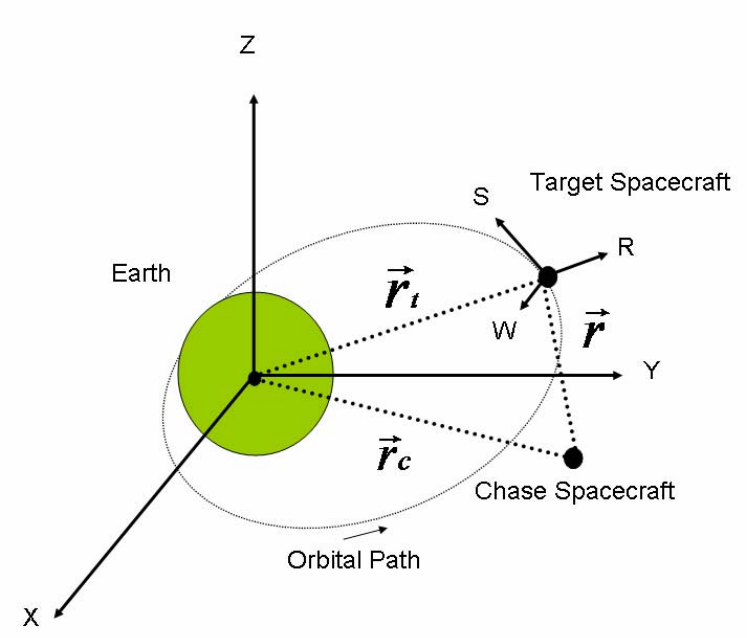

Figure 4.2 Relative Distance.

Using equation (3.14), with the Target spacecraft as the origin of the rotating reference frame, the relative translational acceleration of the Chase spacecraft is given by

$$\ddot{\vec{r}}_c = \ddot{\vec{r}}_t + 2(\vec{\omega}_{BN} \times \dot{\vec{r}}) + (\dot{\vec{\omega}}_{BN} \times \vec{r}) + \vec{\omega}_{BN} \times (\vec{\omega}_{BN} \times \vec{r}) \tag{4.2}$$

In order to solve for the relative motion, the position vectors can be expressed in the Target spacecraft coordinates. The relative position vector is

$$\vec{r} = x\hat{R} + y\hat{S} + z\hat{W} \tag{4.3}$$

with the (x, y, z) adopted as standard Cartesian coordinate notation. The position of the Target spacecraft is displaced from the ECI coordinate along its radial axis

$$\vec{r}_t = -r_t \hat{R} \tag{4.4}$$

Substituting equations (4.3) and (4.4) into equation (4.1), the position vector of the Chase spacecraft becomes

$$\vec{r}_c = (x - r_t)\hat{R} + y\hat{S} + z\hat{W} \tag{4.5}$$

Considering the force of gravity acting on the spacecraft from the two-body problem from equation (3.7), the acceleration of the Target spacecraft is denoted as

$$\ddot{\vec{r}}_t = -\frac{\mu}{r_t^2}\left(\frac{\vec{r}_t}{r_t}\right) = -g_t \hat{R} \tag{4.6}$$

The acceleration of the Chase spacecraft is denoted as

$$\ddot{\vec{r}}_c = -\vec{g}_c + \vec{a}_{thrust} \tag{4.7}$$

where $\vec{a}_{thrust}$ is the acceleration due to control actions and $\vec{g}_c$ is the acceleration of the Chase spacecraft due to the force of gravity. Acceleration on the Chase spacecraft due to gravity can be resolved into the Target spacecraft coordinates [40]**,**

$$\vec{g}_c = -\left(g_c \frac{(x - r_t)}{r_c}\right)\hat{R} - \left(g_c \frac{y}{r_c}\right)\hat{S} - \left(g_c \frac{z}{r_c}\right)\hat{W} + \vec{a}_{thrust} \tag{4.8}$$

where g_c is the vector norm, as denoted in equation (3.3). Next, these resolved components can be substituted into equation (4.2) with the assumption that the angular velocity of the of the Target spacecraft coordinate system is

$$\vec{\omega}_{BN} = \omega \hat{W} \tag{4.9}$$

This angular velocity represents that the orbital rotation only occurs around the spacecraft's coordinate axis perpendicular to the fundamental orbital plane. By performing the cross products and simplifying, the acceleration equations of the Chase spacecraft along each of the Target spacecraft's axis components are

$$\ddot{x} = \left\{ -\left(g_c \frac{(x - r_t)}{r_c} \right) + A_x + g_t - 2(\omega \dot{y}) - (\dot{\omega} y) - (\omega^2 x) \right\} \hat{R} \qquad (a)$$

$$\ddot{y} = \left\{ -\left(g_c \frac{y}{r_c} \right) + A_y + 2(\omega \dot{x}) + (\dot{\omega} x) - (\omega^2 y) \right\} \hat{S} \qquad (b) \qquad (4.10)$$

$$\ddot{z} = \left\{ -\left(g_c \frac{z}{r_c} \right) + A_z \right\} \hat{W} \qquad (c)$$

This set of equations represents the nonlinear equations of relative motion [40]. Using the same orbits as shown in Figure 4.1, the relative motion of the Chase spacecraft elliptical orbit with respect to the Target spacecraft circular orbit is shown in Figure 4.3. The y-axis is the radial axis from the Earth and the x-axis is transverse position of the Chase spacecraft. The plot is orientated for a counterclockwise orbital rotation with the velocity vector pointing to the left. Notice that the motion of the Chase spacecraft drifts away from the initial (0, 0) position. Additional translational acceleration forces can be incorporated into the equations in the same manner as the gravitational and control forces; refer to Chapter III.E.

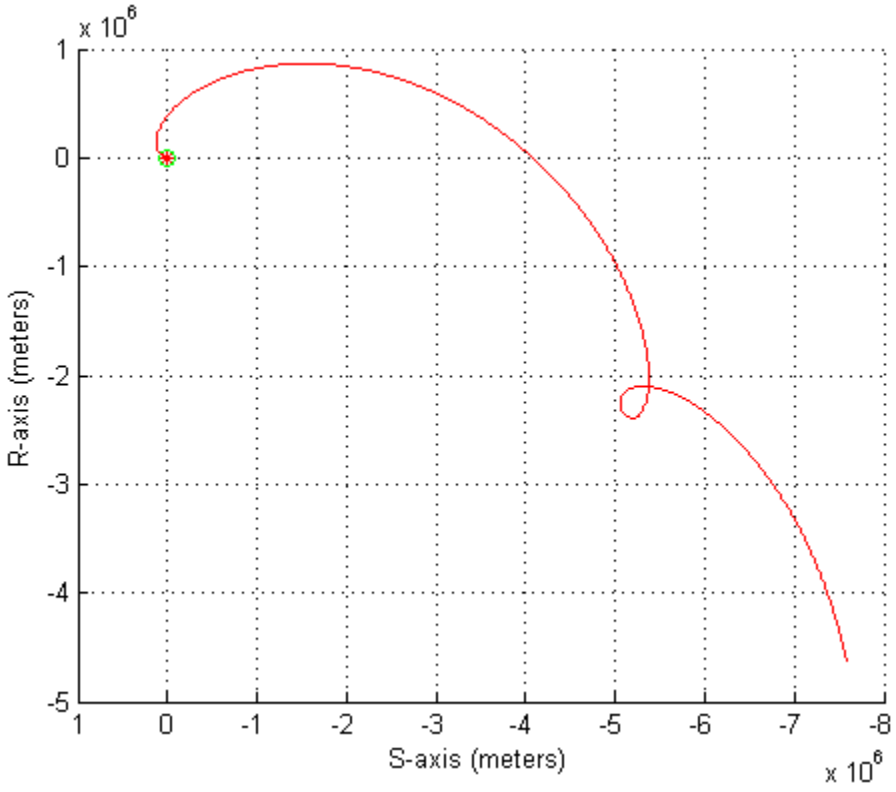

Figure 4.3 Nonlinear Relative Motion.

Several algorithms have been developed to implement these equations. They vary in accuracy based on the gravitational and reference orbit assumptions. The most common is based on a spherical Earth and a circular reference orbit [41]**.** Although for long term formation flight, geometric methods which consider reference orbit eccentricity and differential gravitational perturbations may be more accurate [42]. However, as the relative motion methods increases in accuracy they also increase in complexity. Detailed analysis of the orbital eccentricity and perturbation effects can be evaluated in order to determine which relative motion theory should be applied to a relative motion of two spacecraft [43]. Although for this research pertaining to short duration proximity operations the Clohessy-Wiltshire equations are sufficient.

B. THE CLOHESSY-WILTSHIRE EQUATIONS OF MOTION

The nonlinear equations for relative motion, shown in Equation (4.10), may be simplified and linearized in order to provide closed form analytical solutions. The Clohessy-Wiltshire equations are the well known linearized relative motion equations for near circular orbits. First, assume that the relative distance between the two spacecraft (r) is much smaller than the orbital radius of the Target spacecraft (r_t)

$$r_t >> r \tag{4.11}$$

This research assumes that the relative distance is within a few kilometers and that the orbital altitude is at least a few hundred kilometers. This leads to the following, linearized estimations [32][40]:

$$r = \sqrt{(x - r_t)^2 + (y)^2 + (z)^2} \approx r_t \left(\frac{x}{r_t} - 1 \right) \qquad (a)$$

$$g = \left(\frac{g_t r_t^2}{r^2} \right) \approx g_t \left(1 - \frac{2x}{r_t} \right) \qquad (b)$$

$$g\left(\frac{y}{r} \right) \approx g_t \left(\frac{y}{r_t} \right) \qquad (c) \qquad (4.12)$$

$$g\left(\frac{z}{r} \right) \approx g_t \left(\frac{z}{r_t} \right) \qquad (d)$$

$$g\left(\frac{x - r_t}{r} \right) \approx g_t \left(1 - \frac{2x}{r_t} \right) \qquad (e)$$

Substituting the equation (4.12) estimates into equation (4.10) yields the following

$$\ddot{x} = \left\{ -g\left(1 - \frac{2x}{r} \right) + a_x + g + 2(\omega \dot{y}) + \dot{\omega} y) + (\omega^2 x) \right\} \hat{R} \qquad (a)$$

$$\ddot{y} = \left\{ -\left(g \frac{y}{r} \right) + a_y - 2(\omega \dot{x}) - (\dot{\omega} x) + (\omega^2 y) \right\} \hat{S} \qquad (b) \qquad (4.13)$$

$$\ddot{z} = \left\{ -\left(g \frac{z}{r} \right) + a_z \right\} \hat{W} \qquad (c)$$

Next, assume that the Target spacecraft is in a near circular orbit. This assumption yields a simple estimate of the orbital angular velocity, in radians per second, as

$$\omega = \sqrt{\frac{\mu}{r^3}} \qquad (4.14)$$

Based on this estimate the angular velocity is constant, therefore the angular acceleration is zero ($\dot{\omega} = 0$). These assumptions yield the simple linearized equations known as the on Clohessy-Wiltshire equations

$$\ddot{x} - 2(\omega \dot{y}) - 3(\omega^2 x) = a_x \qquad (a)$$

$$\ddot{y} + 2(\omega \dot{x}) = a_y \qquad (b) \qquad (4.15)$$

$$\ddot{z} + (\omega^2 z) = a_z \qquad (c)$$

where the acceleration due to control actions is zero for freely orbiting spacecraft.

These equations can be written in general state space form, $\dot{x} = Ax + Bu$ and $y = Cx$, with the following six states

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 3\omega^2 & 0 & 0 & 0 & 2\omega & 0 \\ 0 & 0 & 0 & -2\omega & 0 & 0 \\ 0 & 0 & -\omega^2 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \tag{4.16}$$

$$\begin{bmatrix} r_x \\ r_y \\ r_z \\ v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \tag{4.17}$$

The nonlinear coupling weakens as the range between spacecraft decreases and remains much smaller than the Target's orbital radius [41]. These state space equations can be evaluated for controllability, observability, and stability. The linear system is completely controllable, since the controllability matrix, Q_c, is full rank.

$$Q_c = \begin{bmatrix} B & AB & \cdots & AB^{n-1} \end{bmatrix} \tag{4.18}$$

and completely observable, since the observability matrix, R_o, is also full rank.

$$R_o = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix} \tag{4.19}$$

The unforced relative dynamic system is *un*stable, since the six eigenvalues of the A matrix consists of two repeated roots at zero and two repeated imaginary complex pairs. For the zero relative velocity initial condition, the stability of the system can be conceptually thought of as a two axis of stability along the eigenvectors of the zeros and planes described by the complex eigenvalues related vectors. Relatively close initial

Chase spacecraft positions along the S-axis, will stay within a bound and may be considered equilibrium points. Also, there is an axis slightly off of the S-axis with velocity changes along the same axis which may be considered an axis of equilibrium. However, the Chase spacecraft must meet the condition of being relatively close, within a few kilometers, and near the same circular orbit as the target spacecraft. The other possible equilibrium locations may be considered dynamically unstable due to their strict dependence on both position and velocity relationships. These purely imaginary eigenvalues explain the oscillatory behavior of spacecraft relative dynamics. The system stability is further addressed in Chapter VII.D.

Using the same orbits as shown above, the linear relative motion of the Chase spacecraft's elliptical orbit with respect to the Target spacecraft's circular orbit is shown in Figure 4.4. Once again, the y-axis is the radial axis from the Earth and the x-axis is the transverse position of the Chase spacecraft, with the velocity vector to the left. Notice that the motion of the Chase spacecraft is now periodic and does not drift due to eccentricity of the Chase orbit.

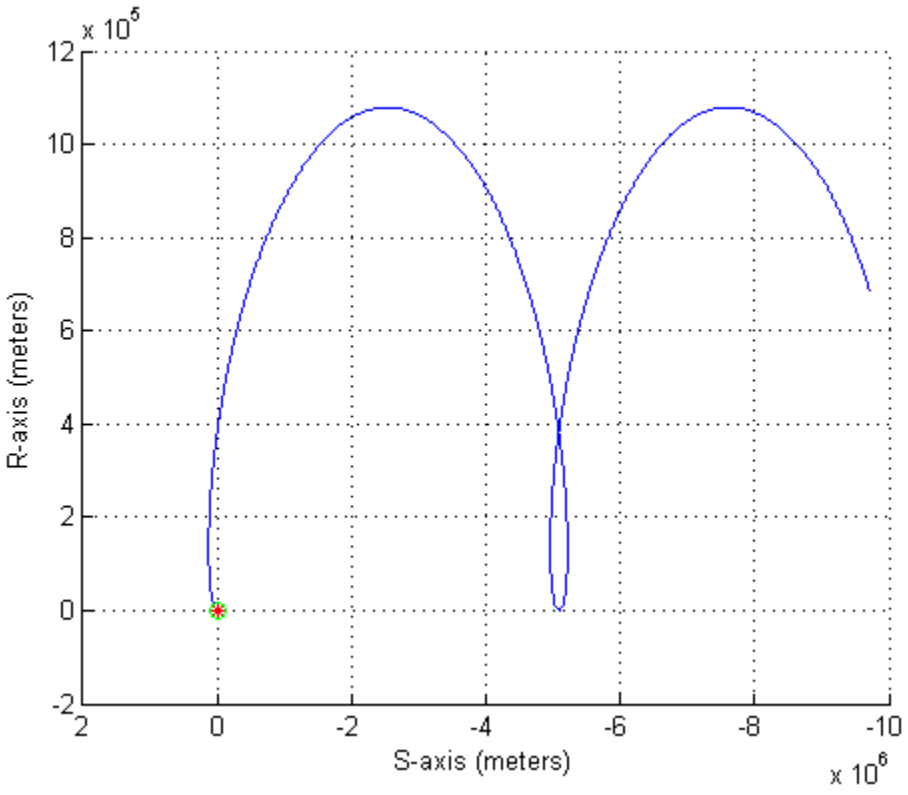

Figure 4.4 Linear Relative Motion.

Next, Figure 4.5 shows both the linear and nonlinear motions on the same plot. The Clohessy-Wiltshire equations track the true relative motion for short time periods. The absolute relative differences between the nonlinear and linear equations for the orbits are shown in Figure 4.6. In this instance, the Clohessy-Wiltshire equations vary less than 70 meters from true for the first 30 minutes of the orbital propagation. As a practical rule of thumb, any step duration longer than a quarter of an orbital period should be avoided due to the 90° rotation of the RSW coordinate system. The set of ordinary differential equations, in equation (4.15), have been used extensively in rendezvous and docking algorithms. They have been useful in analyzing trends for initial positions and velocities [32]. The relative motion equations of the spacecraft can be further simplified into a state transition matrix form; refer to Chapter IV.C.

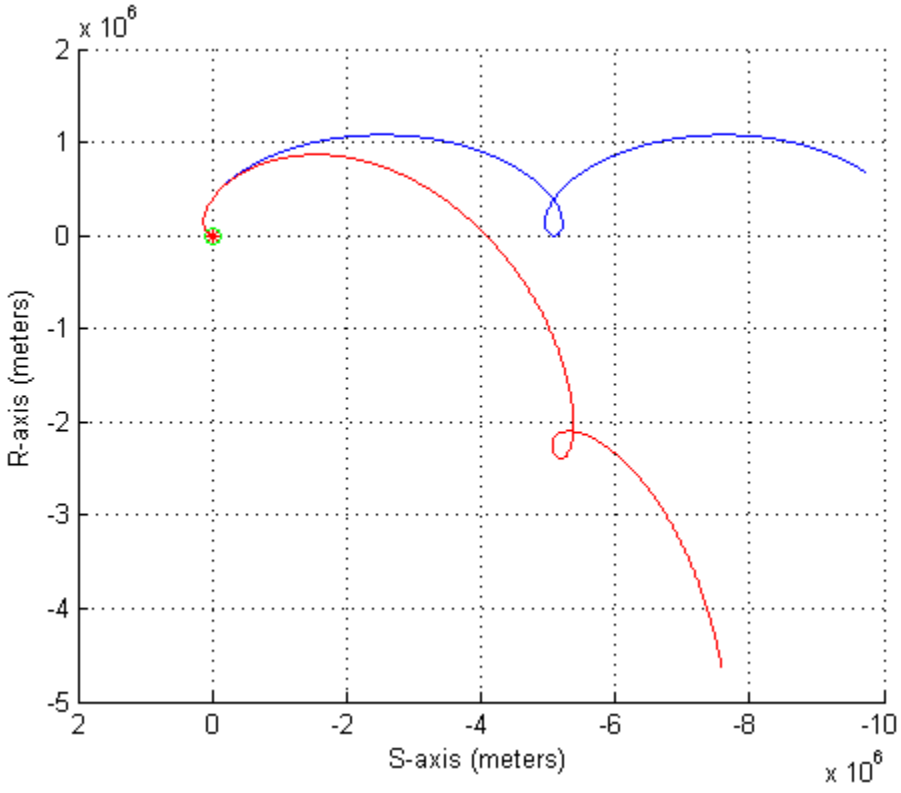

Figure 4.5 Comparison of Relative Motion.

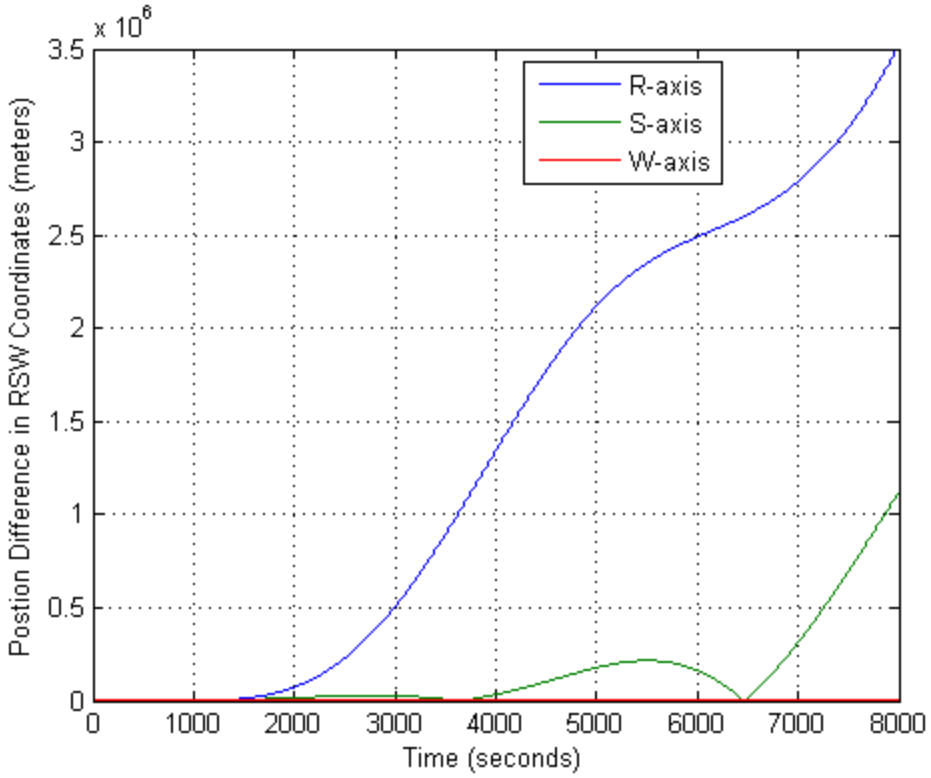

Figure 4.6 Absolute Difference in Relative Position.

C. STATE TRANSITION MATRIX FOR RELATIVE MOTION

The unforced ordinary differential equations, in equation (4.15), can be solved using Laplace operators. The solutions of these Clohessy-Wiltshire equations are known as Hill's equations. The position solutions, based on constant initial values, [32] are

$$x(t) = \left(\frac{\dot{x}_0}{\omega}\right)\sin(\omega t) - \left(3x_0 + \frac{\dot{y}_0}{\omega}\right)\cos(\omega t) + \left(4x_0 + \frac{2\dot{y}_0}{\omega}\right) \qquad (a)$$

$$y(t) = \left(6x_0 + \frac{4\dot{y}_0}{\omega}\right)\sin(\omega t) + \left(\frac{2\dot{x}_0}{\omega}\right)\cos(\omega t) - \left(6\omega x_0 + 3\dot{y}_0\right)t + \left(y_0 + \frac{2\dot{x}_0}{\omega}\right) \qquad (b) \qquad (4.20)$$

$$z(t) = -z_0\,\omega\sin(\omega t) + \frac{\dot{z}_0}{\omega}\cos(\omega t) \qquad (c)$$

where the x_0, y_0, and z_0 are the initial position values and $\dot{x}_0$, $\dot{y}_0$, and $\dot{z}_0$ are the initial velocity values for the Chase spacecraft with respect to the Target spacecraft's RSW coordinate system. The velocity solutions [32] are

$$\dot{x}(t) = \dot{x}_0\cos(\omega t) + \left(3\omega x_0 + 2\dot{y}_0\right)\sin(\omega t) \qquad (a)$$

$$\dot{y}(t) = \left(6\omega x_0 + 4\dot{y}_0\right)\cos(\omega t) - 2\dot{x}_0\sin(\omega t) - \left(6\omega x_0 + 3\dot{y}_0\right) \qquad (b) \qquad (4.21)$$

$$\dot{z}(t) = -z_0\,\omega\sin(\omega t) + \frac{\dot{z}_0}{\omega}\cos(\omega t) \qquad (c)$$

The R and S coordinates are still closely coupled, and the W term is a simple harmonic oscillator.

The position and velocity equations can be placed into general state-space form

$$\bar{X}(t) = \Phi\,\bar{X}_0 \qquad (4.22)$$

where $X(t)$ is the current state vector, Φ is the state transition matrix, and X_0 is the initial valued state vector. The state vector is composed of the position and velocity of the Chase spacecraft, defined as

$$\bar{X} = [x, y, z, \dot{x}, \dot{y}, \dot{z}]^T = [\vec{r}, \dot{\vec{r}}]^T \qquad (4.23)$$

The state transition matrix relating to this state vector is

$$\Phi = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \qquad (4.24)$$

where A, B, C, and D [40] are given as

$$A = \begin{bmatrix} 4-3\cos(\omega t) & 0 & 0 \\ 6(\sin(\omega t)-\omega t) & 1 & 0 \\ 0 & 0 & \cos(\omega t) \end{bmatrix} \qquad (a)$$

$$B = \begin{bmatrix} \frac{1}{\omega}\sin(\omega t) & \frac{2}{\omega}(1-\cos(\omega t)) & 0 \\ \frac{2}{\omega}(\cos(\omega t)-1) & \frac{4}{\omega}\sin(\omega t)-3t & 0 \\ 0 & 0 & \frac{1}{\omega}\sin(\omega t) \end{bmatrix} \qquad (b)$$

$$C = \begin{bmatrix} 3\omega\sin(\omega t) & 0 & 0 \\ 6\omega(\cos(\omega t)-1) & 0 & 0 \\ 0 & 0 & -\omega(\sin(\omega t) \end{bmatrix} \qquad (c)$$

$$D = \begin{bmatrix} \cos(\omega t) & 2\sin(\omega t) & 0 \\ -2\sin(\omega t) & -3+4\cos(\omega t) & 0 \\ 0 & 0 & \cos(\omega t) \end{bmatrix} \qquad (d) \qquad (4.25)$$

This state transition matrix can be used to efficiently calculate the initial and final velocity of the Chase spacecraft required to achieve a desired position. Any required magnitude change in the Chase spacecraft's velocity corresponds to thruster firings, denoted as Δv. All necessary Δv's can be summed and used as a measure of the cost of the spacecraft's maneuver. The minimum total Δv maneuver can be determined as a baseline for evaluating controller command performance. This will be discussed in more detail in Chapter IV.D.

D. OPTIMAL TWO IMPULSE RENDEZVOUS

The co-planar two thrust maneuver, based on the Hohmann transfer concept, is the classic way to plan the rendezvous of two spacecraft. The Hohmann maneuver is mathematically accepted as the most fuel efficient transfer between two circular, and elliptical, planar orbits [44]. The maneuver, first suggested by Walter Hohmann in 1925,

involves two tangential thruster firings for start and stop of the transfer [45]. The two thruster firings are each approximated as impulse changes in velocity. This maneuver minimizes the Δv required, but takes a longer time to complete the maneuver. The maneuver time is one half of the transfer orbit's period. There are cases where Hohmann transfers can be used in series, called bi-elliptic transfers, in order to achieve less Δv [32]. The two impulse rendezvous maneuver can be used for fuel efficiency comparison, and the bi-elliptical maneuver will not be explored in this research. This orbital transfer maneuver is a classic example of a known fuel efficient result, which can be used to validate the high-fidelity six DOF spacecraft model.

The two impulse maneuver is not physically possible or practical for spacecraft close proximity operation. The impulse velocity change is physically impossible for a spacecraft with non zero mass and momentum. The force required to gain the desired impulsive is not usually possible with the thrusters onboard spacecraft. Additionally, the second impulse occurs at the goal position and does not leave much safety margin. Position uncertainties and rapid convergence at the rendezvous of two spacecraft make the second impulse undesirable. Despite these limitations, the two impulse maneuver serves as a baseline for minimum fuel efficiency and maximum duration of a spacecraft maneuver between two points. Therefore, the two-impulse maneuver is useful, as the baseline, for comparing the performance of spacecraft maneuver control algorithms.

The Hohmann maneuver thruster firings are predetermined to take place at a time corresponding to apogee of the initial and transfer orbit. Small change in the time of the thruster firing can result in drastic variation in the Δv for a general point-to-point transfer. For evaluating the general point-to-point orbital transfer, rewrite the state-transition matrix for a final position ($\vec{r}_f$) and velocity ($\dot{\vec{r}}_f$) at some final time (t_f)

$$\begin{aligned} (\vec{r}_f)^T &= A(\vec{r}_0)^T + B(\dot{\vec{r}}_0)^T \qquad (a) \\ (\dot{\vec{r}}_f)^T &= C(\vec{r}_0)^T + D(\dot{\vec{r}}_0)^T \qquad (b) \end{aligned} \tag{4.26}$$

where $\vec{r}_0 = [x_0, y_0, z_0]$ and $\dot{\vec{r}}_0 = [\dot{x}_0, \dot{y}_0, \dot{z}_0]$ are the initial relative position and velocity. The starting relative velocity of the Target and Chase spacecraft is selected to be zero,

such that the Chase and Target velocities are equal. To calculate the initial velocity to intercept the Target position, equation (4.26) can be rearranged

$$(\dot{\vec{r}}_0)^T = B^{-1}((\vec{r}_f)^T - A(\vec{r}_0)^T) \tag{4.27}$$

where the inverse of B is

$$B^{-1} = \begin{bmatrix} \dfrac{-4\omega\sin(\omega t)+3\omega^2 t}{(-8+3\omega t\sin(w t)+8\cos(w t))} & \dfrac{-2\omega(\cos(\omega t)-1)}{(-8+3\omega t\sin(w t)+8\cos(w t))} & 0 \\ \dfrac{2\omega(\cos(\omega t)-1)}{(-8+3\omega t\sin(w t)+8\cos(w t))} & \dfrac{-\omega\sin(\omega t)}{(-8+3\omega t\sin(w t)+8\cos(w t))} & 0 \\ 0 & 0 & \dfrac{\omega}{\sin(\omega t)} \end{bmatrix} \tag{4.28}$$

For a given initial and desired final position, the final velocity can be calculated by substituting equation (4.27) into equation (4.26) to yield

$$(\dot{\vec{r}}_f)^T = C(\vec{r}_0)^T + D[B^{-1}((\vec{r}_f)^T - A(\vec{r}_0)^T)] \tag{4.29}$$

For the situation where the Chase spacecraft is starting and finishing at rest relative to the Target spacecraft, the total Δv is determined by the initial and final velocities. The initial change in velocity is

$$\Delta v_0 = \left|\dot{\vec{r}}_0\right| \tag{4.30}$$

and the final change in velocity is

$$\Delta v_f = \left|-\dot{\vec{r}}_f\right| \tag{4.31}$$

The final velocity, from equation (4.29), is the actual velocity of the spacecraft at the final position. The final change in velocity needs to be counter act this velocity in order to stop the relative motion at the goal. The total change in velocity of the maneuver is simply the sum of all of the two impulse changes in velocity

$$\Delta v = \Delta v_0 + \Delta v_f \tag{4.32}$$

However, the value of Δv depends on the time selected to perform the impulse thruster firings. Therefore care must be taken in establishing the optimized Δv, based on

a functional minimization algorithm which contains time dependent minima. This is better illustrated in the following two examples, which can serve as a basis for comparison.

In the first case, the Chase spacecraft is directly ahead of the goal position. The Chase spacecraft is initially at [0, 70, 0] meters in the Target's RSW coordinate system and maneuvers to the Target's position of [0, 0, 0] meters. For a data set sampled every second for approximately 20,000 seconds, using the state transition equations, determined in Chapter IV.D., yields maneuver duration of

$$t_d = 16,278\,s \tag{4.33}$$

and a total optimized Δv of

$$\Delta v = 2.8636 \times 10^{-3}\, m/s \tag{4.34}$$

These results are comparable to those in [40]. The orbital maneuver could have been done more efficiently over a longer duration.

In the second case, the Chase spacecraft is offset from a goal position. The Chase spacecraft is initially at [50, -100, -50], maneuvers to the Target's position of [0, 0, 0]. For a data set sampled every second for approximately 20,000 seconds, the optimal time of transfer was determined to be

$$t_d = 3,972\,\mathrm{s} \tag{4.35}$$

which yields a total optimized Δv of

$$\Delta v = 0.19675\ m/s \tag{4.36}$$

These results are on the same order of magnitude as those in obtained by [40]. The optimal trajectory path will vary if the final time is allowed to vary. Therefore, the optimal trajectory described here is a function of both the time and Δv of the maneuver. Using different time constraints will result in different paths to the final position. This idea can be applied to the situation where an obstacle is in the optimal path and a secondary path needs to be selected in order to avoid collision. This will result in an efficient, but not Δv optimal, trajectory.

Minimizing the Δv based on the Clohessy-Wiltshire equations, will result in optimal trajectory for point-to-point transfer maneuver. However, this maneuver is not flexible and is compromised if any course correction is necessary [40]. Small changes in desired waypoints or timing of the impulse thrust can result in large trajectory variations and offsets from the desired position. However, these optimal solutions for minimum Δv maneuvers can serve as useful baselines against which an artificial potential field based control algorithm can be evaluated.

V. CLOSE PROXIMITY SPACECRAFT CONTROL

A. CONTROL ALGORITHM DISCUSSION

There are several different control strategies and schemes that may yield satisfactory results for a given scenario. They all have advantages and disadvantages. These strategies range from strict single constraint optimal control to flexible adaptive control. The optimal fuel controllers are useful for large spacecraft maneuvers which require minimum use of fuel. While, the flexible adaptive controllers are applicable for spacecraft in deep space that are likely to experience large communications lags, sensor failures, and unknown situations. By evaluating the disadvantages of each, it can be determined that neither would be ideal for close proximity spacecraft operations. Optimal path planners have the disadvantage of being too slow and computational intensive for real-time application in the presence of obstacles [19]. While, adaptive schemes tend to exhibit more arbitrary motion and require hierarchical structures to maintain fuel efficiency. Although, previous consideration of artificial potential fields for on-orbit assembly [40], motivates the development and application of a potential function based controller for multiple spacecraft in close proximity operations.

Optimal control is typically based on minimizing a single cost function, which may consist of many diverse parameters. Various parameters, including distance, time, energy and path/sensor mapping can be optimized for path planning. However, only one of these parameters can truly be optimal for any particular cost function. All others parameters are treated as constraints. As multiple parameters are included in a cost function the range of possible solutions exhibit the same weighting trade-offs as experienced in classical control with multiple gain tuning. In order to perform path planning, knowledge of the spacecrafts' working environment is required so that coordinate interaction can be determined. With precise knowledge of the working environment and each spacecraft's location, one may be able to specifically command and control all spacecrafts' interaction optimally with a centralized control scheme. However, one must allow for the exhaustive constraint evaluation, algorithm computation time, and universal spacecraft communication requirements which globally optimal algorithms require. As the DOF of the spacecrafts' spatial working environment and the

number of spacecrafts increase, the computation time increases dramatically. For multiple spacecraft organization, in either 2D or 3D space, there are an infinite number of theoretical initial conditions and desired final configurations. This requires calculation of the best course of action for each and every spacecraft at a specific point in time and then instantaneously commanded them into a precise course of action. In addition, the simultaneous communication requirement between the centralized controller and each spacecraft is intensive.

Due to the variable positions and attitudes that a group of satellites may be arranged in, a more flexible adaptive control algorithm may be desired. Decentralized, adaptive and evolving control algorithms allow for variations in the environment and spacecrafts, but are seldom efficient. As the environment changes, the robust controller can be designed to function even though the exact scenario was not anticipated. Some of these adaptive controllers focus use genetic algorithms to allow each spacecraft to function with limited to no global knowledge or communication. These schemes are harder to predict and may not even arrive at a solution to a reasonable spatial command. As the control system shifts in pursuit of better fitness in multiple spacecraft response, there is a delay in commanding. The spacecrafts' movements often appear clumsy and slow. Although with proper development, autonomous and distributed functioning can replace strict control and centralization [46]. Simple spacecraft (insect like) self organizing theories may be interesting in order to show complex collective behavior. However, allowing for emergent behavior of spacecraft is not a risk that most space agencies, or companies, are willing to take with their multi-million dollar spacecraft. Therefore, it is desirable to have each spacecraft efficiently control their own action within the execution of an overarching general command.

There have been a numerous spacecraft formation control schemes which are based on dual level algorithms. The first level determines the overall position and orientation of the formation and the second level commands the individual spacecraft to achieve and maintain their desired position. The selection of the position and orientation of the formation may be determined based on the mission or on the initial position of the spacecraft. Both leader-follower and virtual structure formation approaches have advantages for rigid formation control [10]. However, neither is effective for multiple

spacecraft close proximity operations. Use of a potential function algorithm can incorporate the favorable insensitivity to leader perturbation and limited communication requirement of leader-follower algorithms with the favorable coordinated behavior of virtual structures. The primary attraction point of the potential field acts both as a leader and the geometric center of the virtual structure. The attraction point does not have the single point of failure issues of a true leader and does not rely on the rigid relative positioning of a virtual structure.

Previous spacecraft work usually limited collision avoidance to predictive states which must be include into maneuver planning. These are based on numerically complex and computationally intensive uncertainty ellipsoids. The results were collision proximity and collision probability indicators that needed to be looped back into mission planning decision schemes [47]. This processing may be acceptable for ground stations with access to space situational awareness data, but not typically practical for on-board control algorithms. The direct inclusion of uncertainty information, such as disturbance or sensor covariance, into obstacle's region of influence eliminates the need for uncertainty ellipsoid generation and propagation. Even if these ellipsoids can be generated, they are highly dependent on sensor and communication reliability.

The goal of this research is to develop an efficient control algorithm for use during proximity operations for distributed autonomous multiple spacecrafts. The distributed nature of the spacecrafts will require that global knowledge is not available to each spacecraft [11]. A centralized controller is assumed not to exist, such that each spacecraft must perform their portion of the operation with local information and limited communications. Each spacecraft must also perform efficient fuel consumption maneuvers, while avoiding collisions, and satisfying their commanded goal position and orientation. The primary control algorithm is based on APF development from sensory inputs. An overview of APF control and related topics follows.

B. ARTIFICIAL POTENTIAL FIELD (APF) OVERVIEW

Artificial potential field functions have been used extensively in robot navigation and control [2][3][4][5][6][7][9][48]. The idea of the use of potential functions for robot tasking was pioneered by Khatib [2]. Khatib applied the problem to robot navigation for obstacle avoidance with perfect information. The topological challenge of the use of the potential functions in feedback control laws in order to drive robots was explored by Koditschek [3]. The idea that a conservative system will settle toward the local minima of their potential energy, is extended to establish an artificial potential minima corresponding to an spacecraft's goal position. The goal position acts as an attraction point for the spacecraft. Additionally, forbidden positions, such as obstacles, can be assigned sufficiently large potentials values such that the spacecraft is repelled from these regions. The spacecraft moves along the negative gradient of the topographic potential field to approach the minimum artificial potential value in its environment. Obstacle avoidance with perfect *a priori* information was further refined by Rimon and Koditschek [7]. Newman and Hogan [6] extended the application of potential functions for time varying goal and obstacles. They showed that APF are effective in simple obstacle environments and safer than most path planning algorithms in highly dynamic environments.

For static environments the artificial potential field and global minimums can be determined off-line. However when the environment is unknown and the presence of obstacles are determined by on-board sensors, the local potential field is only known at that instant. The real-time motion planning of the spacecrafts can lead to local minimums. Overcoming the presence of undesired local minimums can be difficult due to the complex shape of potential obstacles. Although, there are various ways to overcome the local minimum problem, including defining artificial potential fields that only have a minimum at the goal position and developing methods for escaping from local minimum [4]. Due to the limited number of obstacles in the spacecraft environment and the dynamics of motion, local minimums do not tend to occur. However, as multiple spacecraft converge to a goal position they pack into a spherical goal region which may cause each spacecraft to chatter about a local minimum. Also chattering can occur during passage between two obstacles. The repulsion field of the obstacles may influence the

APF based controller to bounce back and forth between the obstacles as it passes between them. Precise and smooth motion around obstacles may require an spacecraft to look-up pre-computed information about obstacles in a respective region and adjust the potential field accordingly [48]. Adjustment of repulsive fields around obstacles and time dependent smoothing of potential field was used, along with sliding mode control, by Guldner and Utkin [48] in order to navigate robots about fixed obstacles. Refinement of the obstacle repulsive field may be incorporated into the control algorithm by using minimum communication to periodically update the position of cooperative spacecraft.

As the spacecraft rendezvous and approach docking, multi-spacecraft control may necessitate task assignment among the spacecrafts. Order of docking and servicing may matter either due to heterogeneous functions of each spacecraft or efficiency of relative motion. The distribution of tasks and computation may require the development of a communication protocol. Sliding mode swarm control, with artificial potentials was further developed by Gazi [5][9]. Yang *et al.* [1] proposed a fuel optimal spacecraft formation reconfiguration using multi-spacecraft task assignment. They established a communication protocol which allowed for optimal distributed control algorithm to select the most fuel efficient task for each spacecraft based on the cost to the entire group. A similar communication algorithm seems promising for terminal convergence of spacecraft using a potential field based control algorithm. This docking stage with multiple spacecraft can also be considered the terminal phase of on-orbit assembly [49]. This final stage may require that the local attractive potentials be re-assigned or proportionally weighted for each spacecraft. Since assembly sequences can and usually are predetermined before launch, it may be possible to design the docking behavior into the control algorithm. In addition, predictive and smoothing filters may be applied to the potential field as finite control resources allow.

Artificial potential field guidance was considered for orbital vehicles by C. R. McInnes in 1993 [50]. It has been expanded to consider distributed control [51], autonomous rendezvous with fixed obstacle avoidance [52], autonomous control of on-orbit assembly [40], and fuel efficiency constraints for cluster formation [53]. Recent application of potential functions in controlling swarms of micro-utility spacecraft also shows promise [54][55]. This research explores the use potential function based on

velocity error, as opposed to position errors, for controlling a small spacecraft. In general, the artificial potential shows promise in stable rendezvous, collision avoidance, while maintaining reasonable fuel efficiency. Autonomous and distributed development of local potential field algorithm depends on local sensor information. With very little communication requirements, cooperative spacecraft obstacles and unknown obstacles can be delineated.

Knowledge about cooperative spacecraft can be used to refine obstacle avoidance potential fields. This may be especially useful for terminal stages of the docking. In addition, the space environments itself is relatively obstacle free and most potential obstacles can be characterized by size. The knowledge that unknown obstacles larger than a few meters in diameter are highly unlikely, is useful when estimating an obstacles region with only surface detection sensors. In addition, obstacles crossing the orbital path will usually be at high enough velocity that collision avoidance maneuvers are not necessary or possible for a small satellite with local sensor information. These sorts of cosmic collision avoidance require guidance control from higher level system with access to additional information.

The determination of the local potential filed is dependent on the local sensor information and limited communication of cooperative spacecraft current states. However, the APF control algorithm can be refined to be made more efficient. The initial definition of the APF control algorithm is usually forced to choose between absolute collision avoidance, stable convergence, or optimal fuel efficiency. Exploration and refinement of a time varying control algorithm, based on analytic potential functions, should be able to ensure collision-free, Lyapunov stable, and near-optimal fuel efficiency. True fuel optimization would require global knowledge, intensive computations, and centralized coordination between all group spacecraft. Local knowledge and limited communication/coordination with function constraints will not give the fuel minimized solution. However, simple iterative optimal schemes may be computationally practical while improving efficiency.

The control of the spacecraft is based on recursively stepping through an APF. The potential field is time varying as the sensor readings are updating. This potential

field is evaluated by the control algorithm in order to determine an instantaneous step size and direction for the control actuators. The step size is translated into thruster firing duration. The orientation of the thrusters must allow for motion along the opposite of the potential field's gradient. Steepest Descent algorithms restrict the search direction orthogonal to the contour lines. Refer to Appendix A for a full discussion of optimal search algorithms. In some instances convergence performance may be improved if the search direction is adjusted to be conjugate. This usually means more steps are required for convergence. In fact, convergence is not guaranteed. However, having a known location of the goal position will allow us to force convergence by using this position information to limit the influence of any possible local minimums. A fuel efficient time-varying potential field control algorithm can be developed for autonomous, distributed spacecraft close-proximity operations, to include rendezvous and docking maneuvers.

C. PATH PLANNING CONTROL SCHEME

Typical path planning algorithms presume a priori knowledge of the spacecraft's geometry and environment. The algorithm establishes an optimal trajectory based on some cost function. Then the controller tracks the spacecraft along the trajectory al closely as possible. An example of a current satellite trajectory optimization model is presented in [45]. These types of optimal controllers are designed to minimize fuel for during large spacecraft maneuver or long term formation maintenance. Designing these fuel efficient maneuvers for a single vehicle is time consuming, involving iterative check to ensure minimization of fuel consumption. This translates into heavy processing requirements that seem to be beyond the current on-orbit capability. These optimized controllers seldom account for multiple spacecraft and the spatial interactions between spacecraft. They typically are only concerned with moderately static formation maintenance, such as in [56]. If they do address collision avoidance, by defining a relatively larger safe distance constraint in their performance evaluation, the results are usually over constrained and one can question weather these methods are the most effective in time or energy. For an example of a trajectory planning algorithm related to collision avoidance during docking; refer to Breger and How's research [57]. The existence of numerous constraints often results in trajectory planning that is not obvious

and computationally intensive. Additionally, in a dynamic environment with maneuvering obstacles or limited environmental knowledge the optimal control must be iteratively computed and updated. For autonomous distributed control there is a much simpler solution.

Instead of focusing on the computational intensive optimal solution for every possible multiple spacecraft scenario, a feasible, fuel efficient, and safe solution for robust, autonomous, and distributed close proximity operations is desired. In order to satisfy these requirements, a suitable potential function based closed loop feedback control approach was researched. This control strategy allows for complex multiple spacecraft behaviors to arise from simple single vehicle action. It is generally true that as spacecrafts move from their initial position into the new commanded position, or configuration, they must avoid collision with each other and potential obstacles. This is especially true in orbital operations, where obstacles may include portion of the launch vehicle and unresponsive spacecraft. This collision avoidance in path planning has usually been neglected for spacecraft, due to the large working space and relative distance in which spacecraft usually operate. However, for proximity operations of spacecraft constellations, formations, and swarms/clusters the control of the spacecraft must also consider the relative motion of each spacecraft. As selected spacecrafts converge or disperse in a controlled manner, each must maintain a collision free path. If the relative motions are too fast then control forces and torques may reach saturation and risk overshoot and collision [49]. If the relative motions are too slow the orbital perturbations in close proximity will cause excessive control to maintain safe positioning. The selection of velocity dampening terms in the potential field is dependent on the spacecraft's dynamic environment the actuators responses [3]. The total effect of the spacecraft motion can be spring-like as the spacecraft is pulled toward goals and pushed away from obstacles.

Proper selection of attraction and repulsion potential values for the multiple spacecraft environment offers promising navigation results. The selection of attraction potentials approaches the fuel and time optimal solutions. The resulting maneuvers tend to differ from a true minimum fuel maneuver moderately due to the short time constraints and consideration of multiple parameters. The selection of repulsion potentials around

obstacles and group spacecraft serves as collision avoidance. Sequential calculations of the repulsion potential for converging spacecraft can be considered a near-miss penalty function [53]. The superposition of the goal and obstacle potentials function results in a local artificial potential field for the spacecraft. By descending the lowest path toward the minimum values, the spacecraft approaches the goal similar to a stream flowing down a hillside.

The close proximity operation may be considered complete once the total velocity of the spacecrafts falls below a certain threshold within a desired spatial region. The potential function is not static, due to its dependence on the relative position of spacecrafts and obstacles. Also, sensor uncertainties, dynamic perturbations, and non-ideal actuators affect the rate of change of the potential function. Therefore, as the spacecrafts converge on the final close-proximity goal position they may be subject to chattering phenomena [5]. Resolving chattering usually requires rigorous analysis of the control commands and model uncertainties.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. APF CONTROL ALGORITHM APPROACH

The control of multiple spacecraft during close proximity operations by implementing a recursive APF based algorithm was explored. A robust and unique APF with collision avoidance control algorithm was developed for close proximity spacecraft missions. The APF relates position based potentials to modify the Chase spacecraft desired velocity. The collision avoidance used the obstacle positions to dampen the Chase spacecraft velocity components in the direction of obstacles. The use of Chase spacecraft positioning and velocity relationships in the APF gains allows for straightforward adjustment for spacecraft constraints. Additionally, relative positioning logic limits the effect of obstacles for precision missions, such as docking. Successful simulations for multiple spacecraft during simultaneous maneuvering for convergence, rally, rendezvous, docking operations were conducted. Refer to Chapter VIII for simulation analysis details.

The artificial potential field of each spacecraft is determined by the arithmetic superposition of the goal and all obstacle potential functions in its working area [6]. The APF is an intuitive geometric method of determining the behavior of groups of robots [58]. The overall potential field will serve as the performance surface for the control algorithm, of the form

$$V = V_g + V_o \tag{6.1}$$

where V_g is the attractive potential of the goal point and V_o is the repulsive potential of obstacles [59]. Selection of the potential functions is critical in ensuring smooth potential fields that are stable and provide the desired performance. It may be useful to think of APF control algorithms as being geometrically based, with ranges to the goal and from obstacles being the primary influences on performance. One strategy to selecting potential functions is to base them on the desirable characteristics of Lyapunov functions; refer to Chapter VI.A. Selection of a potential function with the minimum at the goal position can be used to attract spacecraft; refer to Chapter VI.B. In addition, potential functions with relatively large maximums can be used to push away spacecraft; refer to Chapter VI.C. Generally, the most elementary functions that can characterize an obstacle

will yield the best results. In 3D space, spherical or ellipsoid shape obstacle fields are effective in minimizing local minima. However, the obstacle must have a repulsive field in order to ensure collision avoidance. The challenge is to define a region of influence around obstacles. If the obstacle region of influence is too large, it will severely limit motion and causes excessive avoidance. On the other hand, if the obstacle region of influence is too small then collision with the boundary is possible.

A. LYAPUNOV STABILITY FOR POTENTIAL FUNCTIONS

The stability of a dynamic system is of interest when engineering a control algorithm. The description of potential function stability with respect to a dynamic system is generally heuristic. One conceptual method of generating a stable control algorithm is to apply the concept of Lyapunov stability. The definitive papers of Kalman and Bertram [60][61] applied the Lyapunov method to a wide variety of control problems. The basic concept is that for any isolated system, if the rate of change of the energy is negative except for a single minimum equilibrium point, then the energy of the system will continue to decrease until the system arrives at the equilibrium point. This idea can be expanded into a mathematical definition.

A dynamical system, with states (x) is asymptotically stable about some equilibrium point (x_e) if there exists a Lyapunov function ,$V(x)$, with the following properties:

1) Lyapunov function is positive definite, except at equilibrium point.

$$V(x) > 0 \ \ except\ at\ \ x \neq x_e \tag{6.2}$$

2) Rate of change of Lyapunov function is negative definite, except at equilibrium point:

$$\dot{V}(x) < 0 \ \ except\ \ at\ \ \ x \neq x_e \tag{6.3}$$

3) Lyapunov function is zero at equilibrium point:

$$V(x_e) = 0 \tag{6.4}$$

4) Rate of change of Lyapunov function is zero at equilibrium point:

$$\dot{V}(x_e) = 0 \tag{6.5}$$

The asymptotic stable equilibrium point can be considered global asymptotically stable if it is the only equilibrium point in the domain and the Lyapunov function is radially unbounded. A more formal, rigorous, and complete mathematical discussion of stability can be found in [62].

The Lyapunov function can be extended into a term referred to as a potential function. The potential function can be thought of as a topographical representation of the state vector of a dynamical system with respect to a desired goal. The system can be driven toward equilibrium based on the rate of change of the potential function. A sample potential function, shown in Figure 6.1, converges in the direction of the negative rate of change. Figure 6.1 shows a 3D plot of quadratic function with the equilibrium located at the origin of the R-S plane. This quadratic function meets all of the above properties of a Lyapunov function. This type of equilibrium point is often referred to as a stable node, or minimum.

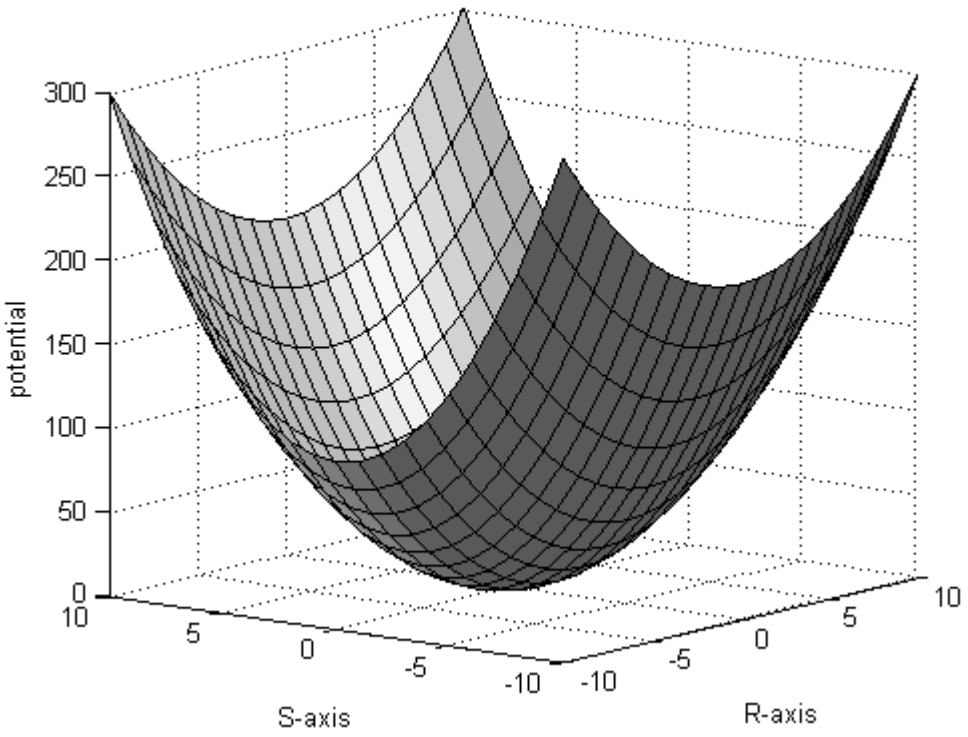

Figure 6.1 Simple Quadratic Function.

The same quadratic function is shown on a contour plot, Figure 6.2, with lines of equal potential appearing as concentric ellipses. Vectors of positive slope point toward the outside of the contour ellipses and away from the minimum at the origin, while the vectors of negative slope point toward the inside of the contour ellipses and in the general direction of the minimum point. The steepest slope is characterized by the gradient vector, which is perpendicular to each contour ellipse. The negative gradient is the path of the Steepest Descent toward the minimum. Refer to Appendix A for discussion of Steepest Descent search method.

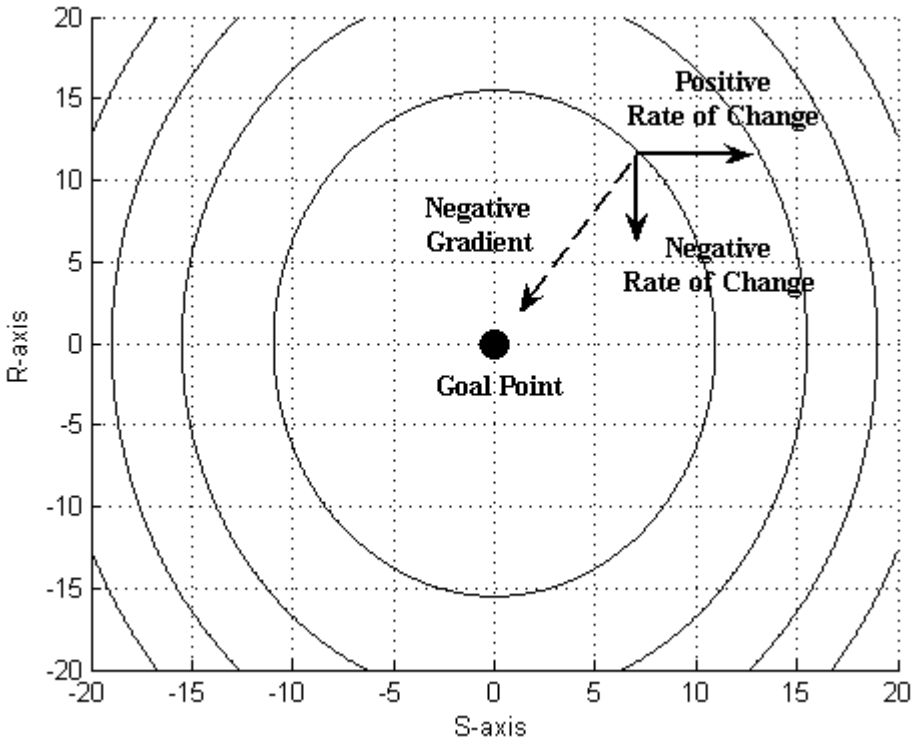

Figure 6.2 Simple Contour Plot.

Since the potential function based on the state vector is a Lyapunov function, it maintains the same characteristics listed in equations (6.2) - (6.5). For instance, any path following the negative rate of change will arrive at the equilibrium point.

B. GOAL POTENTIAL FUNCTION

The goal potential is the minimum potential in the spacecraft working area. This equilibrium point serves as the global attraction point for the Chase spacecraft. A simple quadratic goal potential function based on the relative position of the Chase spacecraft to the goal position is

$$V_g = \frac{\lambda_g}{2}(\vec{r}_c - \vec{r}_g)^T (\vec{r}_c - \vec{r}_g) \tag{6.6}$$

where $\vec{r}_c$ is the relative position vector of the Chase spacecraft, $\vec{r}_g$ is the relative position vector of the goal, and λ_g is a non-negative scaling parameter. The quadratic nature of this potential function V ensures that it is positive semi-definite, such that it is positive for all values of $\vec{r}_c \neq \vec{r}_g$ and zero only at $\vec{r}_c = \vec{r}_g$. This potential function takes advantage of the useful characteristics of both performance functions and Lyapunov functions.

Next, ensure that this goal potential function rate of change is negative semi-definite. Differentiating the goal potential function, equation (6.6)**,** yields the following

$$\dot{V}_g = \lambda_g (\vec{r}_c - \vec{r}_g)^T \vec{v} \tag{6.7}$$

where $\vec{v}$ is the velocity vector of the Chase spacecraft. The control algorithm ensures that the rate of change of the potential function is negative for any $\vec{r}_c \neq \vec{r}_g$ and zero when the Chase spacecraft arrives at the goal position, $\vec{r}_c = \vec{r}_g$. This requires that Chase spacecraft velocity is maintained along vectors toward the negative slope of the potential function. The desired velocity of the Chase spacecraft may be determined as

$$\vec{v}_g = -k_g \frac{\nabla V}{|\nabla V|} \tag{6.8}$$

where k_g is a positive parameter or a magnitude shaping function [40] and ∇V is the gradient of the potential function. The normalized gradient function points the Chase spacecraft in the Steepest Descent direction, so that $\dot{V}$ is always negative semi-definite. The selection of k_g determines the convergence of the control algorithm by relating the

potential function to a convergence velocity. Large values of k_g cause the algorithm to converge quickly toward the area of the goal position, but oscillate around the actual goal position. Small values of k_g ensure slow steady convergence toward the goal position in a damped manner. This is the more desirable of the possible behaviors for spacecraft rendezvous; however convergence to the goal may take more iterations and longer amounts of time. The maneuver is controlled along the Chase spacecraft's relative velocity vector coinciding with the Steepest Descent approach.

The shaping function, k_g, can be directly related to the actuator, such as on-off jet thruster performance. The shaping function can be tuned for different spacecraft by taking into consideration the minimum thrust pulse duration. The thruster mapping algorithm for transforming the control algorithm commands into the thrust components is not considered as part of the APF development. This mapping is dependent on the thruster configuration and performance on the specific spacecraft.

C. OBSTACLE POTENTIAL FUNCTION

An obstacle potential aids the spacecraft to avoid collisions. The obstacle potential is a useful tool for avoiding special regions. For instance, free orbiting objects and regularly maneuvering spacecraft may be avoided. However, collision avoidance is not intended for active evasion of high-speed celestial objects. The location of an obstacle may be from a priori information, such as known space debris in a particular orbit. Otherwise, obstacle location information may come from onboard proximity sensors. If the obstacle is known to be another spacecraft then its size and center of mass can be estimated in order to refine the obstacles potential function. The center of mass and geometric center of an obstacle can usually be assumed to be the same. Otherwise, a Chase spacecraft's local range sensor may be the only information on which to base the obstacle potential function. In this case, the obstacle potential function region of influence must be larger or equal to the actual physical region occupied by the obstacle. It is assumed that the slight loss of fuel efficiency due to over estimating the size of an obstacle is worth the effective execution of collision free navigation.

The goal potential and attractive potential can be formulated separately, but must have properly scaled parameters in order to allow for the desired system motion [58]. An obstruction potential function, based on [52] and [40], is

$$V_o = \lambda_o e^{-\frac{(\vec{r}_C - \vec{r}_O)^T (\vec{r}_C - \vec{r}_O)}{\sigma}} \tag{6.9}$$

where $\vec{r}_o$ is position vector of the obstacle, λ_o is a scaling parameter for the size of the potential function, and σ is the standard deviation of the region of influence. This potential function is a Gaussian with λ_o and σ selected to ensure that the obstacles region of influence (D_o) is equal to or larger then the actual dimensions of the object to be avoided. Numerous candidates could be selected for the obstacle avoidance function, such as spherical power-law and super quadratic functions. However, these functions are more complex to define and generate then the Gaussian function and do not guarantee better performance [40]. This complexity would require more a priori of the obstacles, which is not assumed in this research.

Previous obstruction potential functions have been scaled in order to be equal or greater than the goal potential at the initial position [52]. This selection is mathematically useful so that the obstacle potential is large enough to ensure that the Chase vehicle avoids the obstacle region. However, a potential function based only on relative positions may result in circular local minimums around obstacles. Also in the presence of multiple obstacles, the global minimum may be shifted due to the superposition of obstacle potentials [52]. If the APF function is purely position based then velocity variations may cause unacceptable overshooting position oscillations in the region of obstacles and the goal location. Relating the potential function with Chase vehicle's desired velocity may eliminate such performance limitations. The proper consideration of velocity in the application of APF methods is essential for application on practical systems with limited actuation. The velocity relationship can be included of as a velocity error potential function, as in [54] and [55], or as coupled relative state APF, as in [63][64]. The coupled relative state APF is developed in this research as a position based APF which drives the desired velocity. The magnitude and slopes of the potentials

are driven by the relative position between the Chase spacecraft and the goal and obstacles. A novel selection of the APF relationships and parameters for the multiple spacecraft was determined.

D. APF FOR MULTIPLE SPACECRAFT PROXIMITY OPERATIONS

The APF can be refined for application to the multiple spacecraft close proximity operation. The goal and obstacle APF functions for relative spacecraft dynamics was selected for precision of relative position, steady convergence, and control efficiency. The simplified APF control block diagram is shown in Figure 6.3.

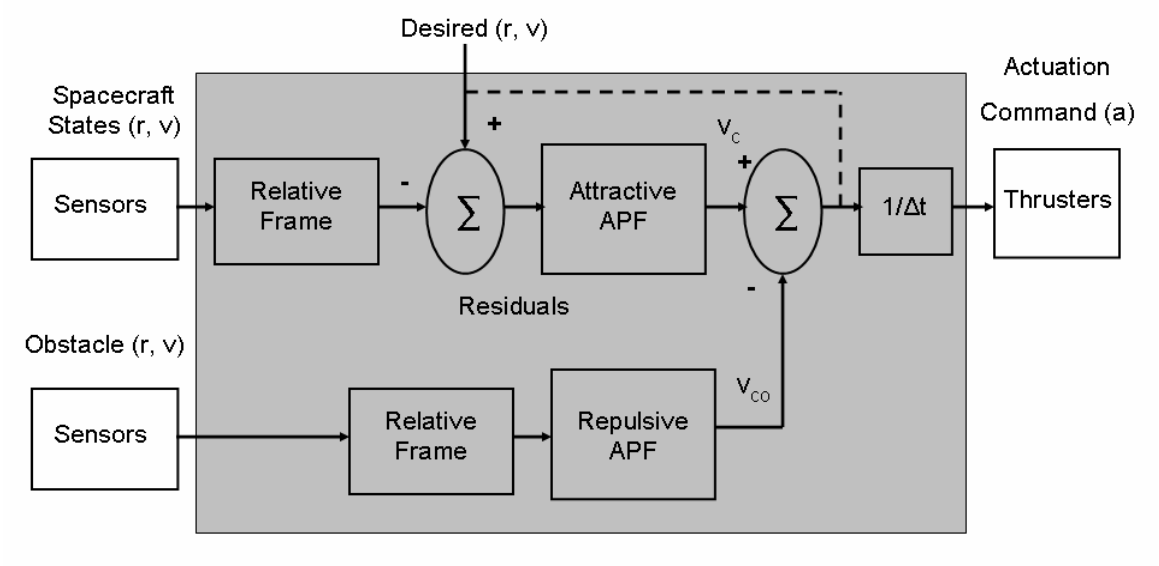

Figure 6.3 APF Control Block Diagram.

The goal, or attractive, potential remains the same as quadratic equation (6.6), with the positive scaling parameter $\lambda_g = 1/r_{cg} = 1/\left|(\vec{r}_c - \vec{r}_g)\right|$, such that

$$V_g = \frac{1}{2} r_{cg} \tag{6.10}$$

This goal potential function results in a linear scaling of the range. Range maintains a positive semi-definite relationship based on the vector normalization.

Application of APF to the spacecraft environment requires very careful considerations of acceptable relative velocity. Stable relative orbital dynamics and limited spacecraft translational actuation seriously limits acceptable velocity magnitudes and variations. Due to these spacecraft relative dynamics and control relationship, the goal APF is used to determine the spacecraft desired velocity in an exponential manner. This acts as a damping of the rendezvous dynamics necessary for position precision in the presence of actuator limitations. This goal potential is used to scale the desired relative velocity of the Chase spacecraft, as in equation (6.8). The desired velocity magnitude shaping function k_g is a positive semi-definite shaping function. The shaping function, k_g, was determined to be

$$k_g = \left(\frac{r_{init}}{r_{\max}} \right) v_{\max} \left(1 - e^{-\left(b_g V_g \right)} \right) \tag{6.11}$$

where the ratio of the initial range, r_{init}, and maximum range, $r_{\max}$, linearly scales the maximum relative velocity, $v_{\max}$. The variable, b_g, is used to shape the exponential decay. The value allows for slower decay, and can be scalable to the initial conditions of the Chase spacecraft, such as

$$b_g = \left(\frac{1}{d_g} \right) \left(\frac{r_{\max}}{r_{init}} \right) \tag{6.12}$$

where d_g is a positive scaling constant. In this research $d_g = 17$ was selected for favorable velocity decay characteristics. The initial position determines the initial desire velocity magnitude which is exponential shaped as the Chase spacecraft approaches the goal position. The resulting Chase spacecraft's desired velocity, based on the attraction toward the goal position, is

$$\vec{v}_g = -k_g \frac{\nabla V}{\left| \nabla V \right|} = -\left(\frac{r_{init}}{r_{\max}} \right) v_{\max} \left(1 - e^{-\left(b_g V_g \right)} \right) \frac{\vec{r}_{cg}}{r_{cg}} \tag{6.13}$$

This desired velocity shaping function decreases the Chase spacecraft desired velocity as it approaches the goal. Based on an assumption of zero starting relative velocity, the initial velocity transient is often large and causes the control actuator to saturate. In order to avoid this saturation an exponential ramping function, k_R, can be added to Equation (6.13), such as

$$\vec{v}_g = -k_R \left(\frac{r_{init}}{r_{\max}} \right) v_{\max} \left(1 - e^{-\left(b_g V_g \right)} \right) \frac{\vec{r}_{cg}}{r_{cg}} \tag{6.14}$$

where this velocity ramping term can be represented as

$$k_R = \frac{d_R}{r_{init}} \left(1 - e^{-t} \right) \tag{6.15}$$

For this research the velocity ramping constant, d_R, was selected to be $d_R = 10$. This allowed for more comparable performance with the control algorithm developed in Chapter VII. This ramping term only influences the initial velocity transient by allowing a more gradual increase related to the start-up of the control algorithm and the initial range from the goal. For instance, maneuver of approximately 100 meters result in a ramp up to the maximum velocity is about 60 seconds.

The actual relative velocity is subtracted from the desire velocity to determine the Δv required by the control effort. This desired change in velocity is used to determine the Chase spacecraft's desired control actuation, in terms of acceleration

$$\vec{a}_g = \frac{\left(\vec{v}_g - \vec{v} \right)}{\Delta t} \tag{6.16}$$

The goal potential allows for convergence to the goal position, however an obstacle potential is required to avoid collision with other spacecraft and sensed objects. The obstacle avoidance function is best thought of as a damping of the geometric forces toward an obstacle, such as relative velocity and acceleration. Due to the manifold of relative dynamics, actual pushing away from an object can lead to instability. The obstacle potential selected to be a Gaussian function of the form

$$V_o = \lambda_o \left(e^{-\left(\frac{r_{co}^2}{2\sigma^2}\right)} - e^{-\left(\frac{D_o^2}{2\sigma^2}\right)} \right) \tag{6.17}$$

where , r_{co} is the distance from the Chase spacecraft to the center of the obstacle, D_o is the region of influence from the center of the obstacle, σ is the standard deviation of the bell-shaped curve, and λ_o is a positive function which serves as a repulsive potential magnitude shaping function. The repulsion potential curve is a smooth function that increases from the boundary of the region of influence to the surface of the obstacle. A smooth transition is ensured by subtracting the value of the Gaussian function at the edge of the region of influence. The employment of a Gaussian function allows for direct implementation of uncertainty in an obstacle state. The shaping parameter λ_o is used to ensure that the value of the repulsive potential at the obstacle surface is equal to the initial attraction potential. The obstacle shaping parameter used in this research is

$$\lambda_o = \left(\frac{r_{init}}{2} \right) \left(e^{-\left(\frac{L_o^2}{2\sigma^2}\right)} - e^{-\left(\frac{D_o^2}{2\sigma^2}\right)} \right)^{-1} \tag{6.18}$$

where L_o is the exterior surface radius of the obstacle, such as another spacecraft. The region of influence of the obstacle, D_o, and the standard deviation, σ, are functions of the size of the obstacle, L_o. This selection of λ_o ensures that the value of V_o equals the initial value of V_g at the surface of the obstacle. The region of influence, D_o, is determined from the size of the obstacle, the velocity of the Chase spacecraft, $\vec{v}$, and the maximum acceleration, $a_{\max}$, allowed by the control actuation. The minimum stopping distance, D_{stop} required by a spacecraft is determined as

$$D_{stop} = \frac{\vec{v}^2}{4 a_{\max}} \tag{6.19}$$

This functional link allows the speed and responsiveness of the Chase spacecraft in the rendezvous region to determine the buffer distance which is required for obstacles.

$$D_o = d_o \left(L_o + D_{stop} \right) \tag{6.20}$$

where the constant multiple factor, d_o allows for a smooth breaking region for avoiding impact with obstacles. For this research, $d_o = 3$ was selected for favorable avoidance characteristics. The standard deviation σ is selected so that the obstacle surface is within one standard deviation as the spacecraft velocity approaches zero, such that

$$\sigma = \frac{D_o}{3} \tag{6.21}$$

This relationship allows a reasonable safety region around obstacles and a smooth Gaussian repulsive potential function. The velocity change due to applying maximum thruster acceleration is shown in Figure 6.4. From an initial velocity and a maximum acceleration the minimum stopping distance is determined by equation (6.19) and the resulting velocity during the braking is shown as a solid line. The actual braking can not be steeper than this maximum condition. Using the obstacle shaping parameter, equation (6.18), and the standard deviation, equation (6.21), a smooth Gaussian velocity change can be commanded, as shown as a dashed line. Notice that both functions are equal to zero at the obstacle's outer surface. For practical application the Gaussian must not be steeper than the maximum braking condition at any point. Also, the area of the maximum stopping function must be less to the area under the Gaussian curve. This is achieved in the algorithm by using the three standard deviations with an additive term.

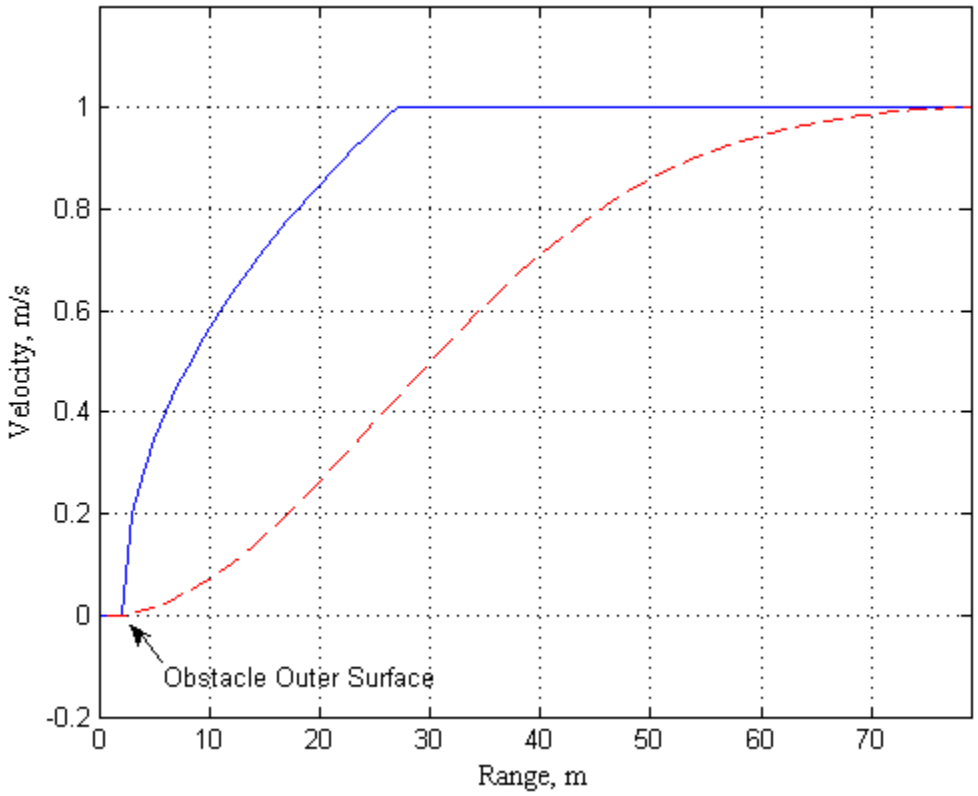

Figure 6.4 Spacecraft Velocity in Obstacle Region.

However, application of this position based obstacle repulsion potential requires that it is related to the attractive potential. This obstacle potential is used to modify the desired relative velocity of the Chase spacecraft, as in equation (6.8). The desired velocity shaping function k_o is a positive semi-definite shaping function used push the Chase spacecraft away from the obstacle. The shaping function, k_o, was determined to be

$$k_o = \frac{k_g V_o}{\left(\frac{r_{init}}{2}\right)} \tag{6.22}$$

This unique shaping function allows for k_o being equal to k_g at the surface of the obstacle. The resulting Chase spacecraft's desired velocity due to the obstacle, based on the repulsion away from the obstacles position, is

$$\vec{v}_o = k_o \frac{\vec{r}_{co}}{r_{co}} = \frac{k_g V_o}{\left(\frac{r_{init}}{2}\right)} \frac{\vec{r}_{co}}{r_{co}} \tag{6.23}$$

The obstacle shaping function dampens the Chase spacecraft approach in the direction of the obstacle as spacecraft advances toward the goal position. Proper selection of damping based on components of the relative velocity vector results in a smooth flow around spherical obstacles. Additionally, this insight and use of damping shaping functions should eliminate of any possible narrow passageway oscillations, such as discussed by Masoud [65]. The attraction velocity vector of the goal is toward the goal position and the repulsive velocity vector of obstacles away from each obstacle. The effect of each obstacle within the spacecraft's spatial region is summed in order to achieve the total repulsive force on the Chase spacecraft. The total control force is determined by vector addition of the potential derived velocity minus the current actual velocity vector of the Chase spacecraft, as

$$\vec{a}_{total} = \frac{\left(\vec{v}_g + \sum_{obs=0}^{n} \vec{v}_o - \vec{v}\right)}{\Delta t} \tag{6.24}$$

where the number of obstacles is not limited, but can result in Chase spacecraft not being able to arrive at the goal location. Obstacles may be other spacecraft or stationary exclusion zones. The other spacecraft are generally moving relative to the Target and Chase spacecraft. In this research, other spacecraft are typically additional Chase spacecraft converging toward a goal within the same region. Stationary obstacles are in a fixed position relative to the goal location. These may be extensions of Target spacecraft or exclusion zones due to physical objects, such as solar panels, or nonphysical items, such as thrust plume or radiation beam regions [52].

Selection of the magnitude of the repulsion shaping function must be related to the attraction shaping function in order to achieve desired critically damped performance. Proper selection of a repulsion shaping function based on the attraction shaping function allows for safety in selecting goal positions and efficiency when avoiding obstacles. For instance if the region of influence of the obstacle is too small and the slope of the

repulsive potential shaping function is too steep then a thrust limited actuator may not be able to avoid collision with the obstacle. On the other hand, if the obstacle region is too large then the Chase spacecraft may be less efficient in both control effort and maneuver duration as it avoids obstacles.

An obstacle's repulsive region of influence causes a potential minimum or saddle point to occur in the area between the obstacle outer region of influence and the surface of the obstacle. The location of this local minimum depends on the obstacles location with respect to the goal position. This local minimum can cause difficulty if the overall potential function is the only driving function for determining control effort. However, the attractive and repulsive desired velocity shaping functions, k_g and k_o respectively, allow for velocity damping around regions of concern. This ensured that the Chase spacecraft slows as it approaches the goal position and avoids obstacles. These shaping functions are also selected so that the desired velocity determined from the repulsive potential balances with the attractive potential at the surface of the obstacle. This allows for the goal position to be placed in the center of a spacecraft and the control algorithm to converge to the surface of the Target spacecraft. This is a vital capability for docking algorithms. The control algorithm is precise within a millimeter of the goal position or the Targets spacecraft's outer surface, assuming ideal sensor measurements.

As multiple spacecraft and obstacles occupy the Chase spacecraft's region, some simple control logic must be applied. First, Chase spacecraft are only influence by obstacles when they are within the obstacles region of influence, D_o. Second, only obstacles which are equal distance or closer to the goal position than the Chase spacecraft are allowed to influence the Chase spacecraft. For instance, the spacecraft is looking toward the goal like an automobile on the road. The next spacecraft converging into the goal region will then avoid contact with Chase spacecraft. In most cases, other spacecraft are simply treated the same as obstacles. However, additional logic is needed if multiple spacecraft are converging within the same goal position. The third control logic condition uses a safety function, k_s, to modify the desired repulsive velocity between maneuvering spacecraft as they approach the goal. This safety influence between multiple converging spacecraft ensures collision avoidance while achieving the closest

possible convergence to the goal. This safety function is a trade-off between collision avoidance and docking. The safety function, k_s, between converging spacecraft multiplies the current $\vec{v}_o$ and results in the following modification to (6.23) as

$$\vec{v}_o = k_s \, k_o \, \frac{\vec{r}_{co}}{r_{co}} \tag{6.25}$$

where k_s is usually greater than or equal to one. If $k_s = 1$, there is an ideal balance between $\vec{v}_o$ and $\vec{v}_g$ at the surface of the goal location boundary. This may allow contact between Chase spacecraft converging at the goal location, which obviously can be undesirable. A value of $k_s = 1.01$ is large enough to ensure that multiple spacecraft converging upon the exact same goal position do not collide. However, the multiple spacecraft rendezvous to the exact same goal position results in a staggered convergence. The first Chase spacecraft to arrive converges to the goal position. The next Chase spacecraft converges to within millimeters of the first Chase spacecraft. The third Chase spacecraft has the additive repulsion of the first two spacecraft and converges to a radial position further away. Any additional spacecraft will converge to a safe point slightly further away. This staggered cluster may be a desirable result for spacecraft rendezvousing to an unknown formation, where additional command maneuvering may need to occur.

For multiple spacecraft docking maneuvers, the staggered cluster effect of the additive repulsion may not be desired. In this case, the goal location is an actual Target spacecraft. To allow the later arriving spacecraft to converge toward docking, while avoiding collision, the safety function, k_s, is adapted to be a decaying exponential of the attractive potential based on the goal position, such as

$$k_s = 1 - e^{-\left(V_g - \frac{L_o}{2}\right)} \tag{6.26}$$

This results in the repulsion due to other spacecraft decaying toward zero as the Chase spacecraft reaches the outer bound of the Target spacecraft. In this case, the multiple spacecraft converge relatively tightly around the Target spacecraft. Limitations in the

Target spacecraft's outer boundary surface area and local minimums due to saddle points may cause some delays for spacecraft which arrive late. This is usually only an issue for a second wave of spacecraft which arrive as the first few converging spacecraft are settling into position. It is envisioned that each spacecraft is commanded to a specific docking port, therefore this clustered convergence is not an operational issue. Also, more spacecraft will not typically be commanded to converge toward docking on a spacecraft which can not support additional spacecraft.

Some simple logic is used to ensure that obstacle influence and goal convergence is reasonable. For instance, any obstacle outside the relative range of the goal position is not allowed to influence the Chase spacecraft. This ensures that late arriving spacecraft do not cause docking spacecraft to bump into the Target spacecraft. The Chase spacecraft's collision avoidance motion is dependent on three primary logical conditions, as follows:

1. Chase spacecraft must be within region of influence of an obstacle, such that $r_{co} \le D_o$.
2. Chase spacecraft range to its goal must be greater than the obstacle's range to its goal, such that $r_{cg} \ge \left(r_{sg} - L_o\right)$. This allows the Chase spacecraft to be influenced only by obstacles closer to their goal, so that a closer Chase spacecraft is not disrupted from its goal due to a farther converging spacecraft. The inclusion of the L_o term acts as a safety margin ensuring that the Chase spacecraft does not clip the side of obstacles as it resumes it free-space motion.
3. Chase spacecraft range to its goal must be greater than the distance to the obstacle, such that $r_{cg} \ge \left(r_{co} - L_o / 2\right)$. Obstacles on the far side of a goal location are not allowed to influence the convergence toward the goal. The inclusion of the L_o term acts as a safety margin ensuring that other docked spacecraft are avoided while having a limited influence.

These logical conditions limit the collision avoidance in obstacle dense environments. The most sensitive of these environments occurs when multiple spacecraft are simultaneously converging toward a common Target spacecraft. If the desire is for multiple spacecraft to simultaneously dock, then the obstacle influence of other Chase spacecraft is decayed near the goal location, as shown in Equation (6.26). The third

logical condition allows for the Target spacecraft repulsive influence to decay along the docking port's axis. This decay results in cardioids shapes similar to that discussed by Lopez and McInnes [52], without the need for additional transformations. The Target spacecraft's region of influence is due to the spherical, Gaussian based, region of influence with a roughly conic shaped wedge cut from around the docking port. Although, a full 3D representation is not necessary since an obstacle's region of influence is primarily range dependent.

This multiple spacecraft APF control algorithm appears to be directly applicable to any spacecraft in a full range of close proximity maneuvers. The parameters for the potentials and shaping functions are not system specific, so intensive tuning is not required. The performance appears to be robust in the full range of close proximity operations. Maneuver ranges of over one kilometer, docking precision within millimeters, and relatively high allowable speeds establish improvements in timeliness and robustness. Direct comparisons with previous APF control algorithms for spacecraft is limited, since few which have been fully developed and simulated. For initial consideration refer to previous research conducted by McQuade [40] or the system specific algorithm recently developed by Neubauer [54][55].

VII. LQR/APF CONTROL ALGORITHM APPROACH

The LQR approach can be recursively applied to the multiple spacecraft close proximity control. The inclusion of dynamics in the LQR computations allows for optimality considerations. Also, there is a desire to combine the APF repulsive collision avoidance capability with the refined LQR for spacecraft rendezvous. The developed LQR with APF-based collision avoidance allows for efficiency based on the environmental dynamics combined with gain weighting and logic to allow for primary mission achievement. The simplified LQR/APF control block diagram is shown in Figure 7.1.

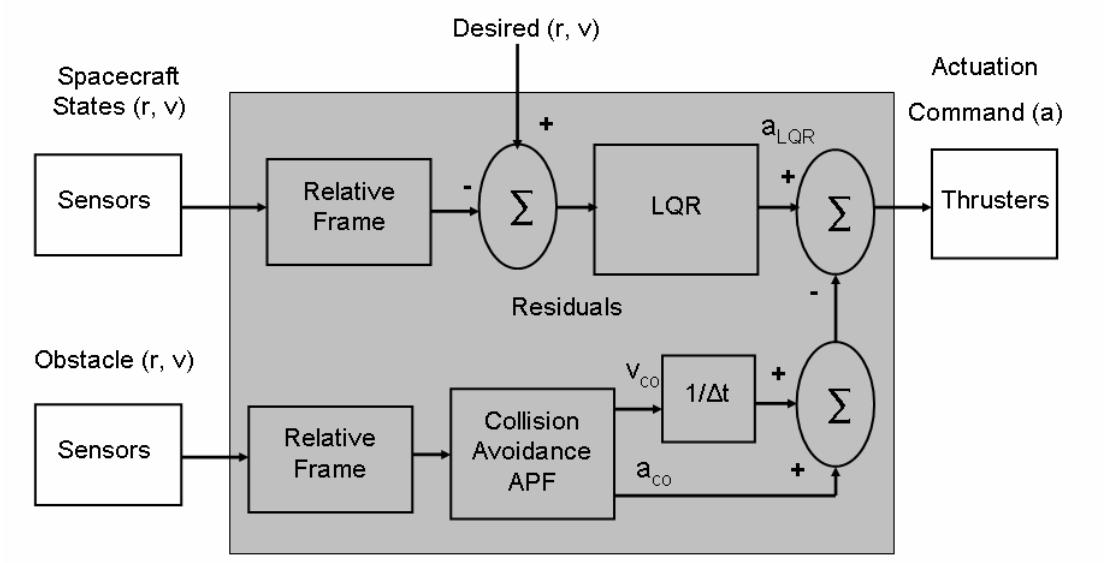

Figure 7.1 LQR/APF Control Block Diagram.

A. GENERAL LQR

The LQR uses the state space dynamics of the system to determine the optimal control effort based on solving Riccati equation for a selected cost function with full state feedback. A quadratic cost function can take into account control effort for a linear system is referred to a LQR. The quadratic cost function is of the general form

$$J_{LQR} = \frac{1}{2}\int_0^T \left(\vec{x}^T\, Q\, \vec{x} + \vec{u}^T\, R\, \vec{u} + \vec{u}^T\, N\, \vec{x} \right) dt \tag{7.1}$$

The gain matrixes for the states, Q, and control effort, R, are of primary interest. The coupling gain matrix, N, is typically zero. These gains are used to solve the Riccati equation.

$$S\,A + A^T\,S + Q - \left(S\,B + N\right) R^{-1} \left(B^T\,S + N^T\right) = 0 \tag{7.2}$$

where S is the solution of the algebraic Riccati equation. The state feedback to minimize the quadratic cost function can be determined from this solution. The optimal state feedback gain, K_{LQR}, for a particular cost function is determined by

$$K_{LQR} = R^{-1} \left(B^T\,S + N^T\right) \tag{7.3}$$

This optimal feedback gain is then used to determine the optimal control effort, such as

$$\vec{u}^* = -R^{-1}\,(B^T\,S + N^T)\,\vec{x} = -\,K_{LQR}\,\vec{x} \tag{7.4}$$

The computation for a limited state LQR can be relatively efficient, as compared to more complicated optimal routines. Increasing the magnitude of the state weighting matrix, Q, will result in a faster convergence to goal. Increasing the magnitude of the control effort weighting matrix, R, will result in more efficiency. The trade-off is between the time of convergence and the control efficiency. The K_{LQR} gain is mostly a constant gain, which decreases as the states converge to the final value.

The determination of the gain matrices is usually based on the normalizing the quadratics along the diagonal. First attempts typically use the maximum allowable values of the states and control effort as the initial values, such as

$$Q = \begin{bmatrix} \dfrac{\alpha_{Q_1}}{\left(x_{1\max}\right)^2} & & 0 \\ & \ddots & \\ 0 & & \dfrac{\alpha_{Q_n}}{\left(x_{n\max}\right)^2} \end{bmatrix} \tag{7.5}$$

with α_Q adjusted based on the state performance. Similarly, the R weighting matrix is selected based on the maximum allowable control effort, such as

$$R = \begin{bmatrix} \dfrac{\beta_{R_1}}{\left(u_{1\max}\right)^2} & & 0 \\ & \ddots & \\ 0 & & \dfrac{\beta_{R_n}}{\left(u_{n\max}\right)^2} \end{bmatrix} \tag{7.6}$$

with β_R adjusted based on the control effort demanded. The selection of these gain variables require insight into the desired response of the system with modifications based on simulation results.

B. LQR/APF FOR MULTIPLE SPACECRAFT PROXIMITY OPERATIONS

The LQR control response for the multiple spacecraft maneuver requires consideration of system response and control efficiency. System response includes the concepts of maneuver duration and precision upon reaching the goal. For the multiple spacecraft rendezvous problem, a critically damped relative position response with limited control effort is desired. As with all spacecraft maneuvers, control efficiency during multiple spacecraft close proximity operations must be considered. However, the convergence maneuver is assumed to be operationally significant and must be performed in finite time duration. For this research, close proximity maneuver durations of one quarter of an orbital period, approximately 30 minutes, are desired. This duration is based on the spacecraft starting from an initial relative position of within one kilometer. The close proximity maneuver is considered successful once the spacecraft converges within a spherical region from its desired goal position. This precision may be much greater than the typical one meter used to evaluate most rendezvous maneuvers. The intent is to be able to use the developed control algorithm for docking maneuvers.

The balancing factor between spacecraft relative position and control effort efficiency is the relative convergence rate. However, the relative spacecraft dynamics causes rendezvous challenges if the relative convergence rate is too slow or rapid. If the

rate of convergence is slow the goal position is spirally orbited as the minimal control actuation is used. The slow converges can dramatically increase the maneuver duration as the spacecraft approaches close to the goal position. The long duration within close proximity of multiple spacecraft can unnecessarily increase the danger of collision due to perturbations. On the other hand, if the rate of convergence is too rapid the limited actuation will result in a collision danger due to relative position overshoot. Even if collision is avoided initially, the spacecraft continues to orbit through the goal position due to overshooting oscillation effects, often referred to as a *rubberband* effect. This limits both the close proximity maneuver precision and efficiency, with the resulting motion of the Chase spacecraft oscillating about the goal position.

Taking the above issues into consideration, the attractive potential of the APF algorithm is replaced with an iterative LQR for multiple spacecraft in proximity operations. Collision avoidance was incorporated based on the APF-type shaping function. The LQR becomes the driving control toward the goal and the APF based collision avoidance is successfully applied. The resulting LQR with APF-based collision avoidance incorporates linearized dynamics for free-space optimal convergence and utilizes avoidance based on obstacle geometric relationships. The developed multiple spacecraft LQR/APF control algorithm is control effort efficient and effectively avoids collisions while successfully conducting a wide range of close proximity operations.

1. LQR Attractive Component

The iterative LQR gain matrices for multiple spacecraft were selected after evaluating the APF control responses. The gains were determined for six states, position and velocity along each axis, along with control effort along each axis. The resulting gains matrixes, based on equations (7.5) and (7.6), are as follows,

$$Q = \begin{bmatrix} \frac{\alpha_{Q_1}}{\left(x_{\max}\right)^2} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{\alpha_{Q_2}}{\left(y_{\max}\right)^2} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{\alpha_{Q_3}}{\left(z_{\max}\right)^2} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\alpha_{Q_4}}{\left(\dot{x}_{\max}\right)^2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\alpha_{Q_5}}{\left(\dot{y}_{\max}\right)^2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{\alpha_{Q_6}}{\left(\dot{z}_{\max}\right)^2} \end{bmatrix} \tag{7.7}$$

$$R = \begin{bmatrix} \frac{\beta_{R_1}}{\left(u_{x\max}\right)^2} & 0 & 0 \\ 0 & \frac{\beta_{R_2}}{\left(u_{y\max}\right)^2} & 0 \\ 0 & 0 & \frac{\beta_{R_3}}{\left(u_{z\max}\right)^2} \end{bmatrix} \tag{7.8}$$

The relative position error along each axis is equally weighted, so that

$$x_{\max} = y_{\max} = z_{\max} = r_{cg} \tag{7.9}$$

The selection of maximum distance as the current distance to the goal, r, allows position to become more important as the spacecraft approaches the goal. The relative velocity error along each axis is also equally weighted, such that

$$\dot{x}_{\max} = \dot{y}_{\max} = \dot{z}_{\max} = \left(\frac{r_{init}}{r_{\max}}\right) v_{\max} \tag{7.10}$$

This desired velocity is determined by scaling the maximum relative velocity, $v_{\max}$ by the ratio of the initial range, r_{init}, and maximum range, $r_{\max}$. For this research, the maximum relative velocity was conservatively selected as

$$v_{\max} = 1.0 \ m/s \tag{7.11}$$

This selection of velocity limits the transients due to the initially neutral relative velocity and also limits the convergence rate for safe operations. However, this $v_{\max}$ is approximately ten times greater than the relative velocities allowed by previous APF proximity control [40][54].

With the denominators of the diagonal gains selected for desired maximum position and velocity. The numerator position error and velocity errors along each axis are equally weighted. Therefore, the α_Q gains are all equal and scaled as the Chase's distance to the goal position converges

$$\alpha_{Q_1} = \alpha_{Q_2} = \alpha_{Q_3} = \alpha_{Q_4} = \alpha_{Q_5} = \alpha_{Q_6} = r_{cg} \tag{7.12}$$

The actuator control effort is the acceleration imparted due to the translational thrusters. For this research, the thrust along each axis is limited to a maximum acceleration of

$$a_{\max} = \frac{F_{thrust}}{m_s} = 0.01 \ m/s^2 \tag{7.13}$$

based on a thrust force of 1.0 Newton and a spacecraft mass of 100 kg. This maximum acceleration is the maximum control effort along each axis.

$$u_{x\max} = u_{y\max} = u_{z\max} = a_{\max} \tag{7.14}$$

The scaling of the control effort is also scaled as the spacecraft relative position changes. Therefore, the numerators of the control effort matrix diagonal gains are

$$\beta_{R_1} = \beta_{R_2} = \beta_{R_3} = r_{cg} \tag{7.15}$$

A minimum scaling factor for the numerator can be selected such, as the range to goal approaches zero, that numerical problems and chattering is avoided. For instance as the r_{cg} approaches zero the value of r_{cg} may be limited to some nominal minimum value, such as $r_{cg} \geq 0.05\ m$.

The control effort resulting from the LQR algorithm, refer to Equation (7.4), can be thought of in terms of a desired acceleration, such as

$$\vec{a}_{LQR} = -K_{LQR}\,\vec{x} \tag{7.16}$$

where $\vec{x}$ is a generalized representation of the Chase spacecraft's position and velocity states.

2. APF-Based Collision Avoidance Component

The LQR drives the control effort based on the relative spacecraft systems linearized state dynamics. The LQR algorithm control effort varies the Chase spacecraft's position and velocity states in a more complicated manner then the previous geometric based APF. This more complicated relationship requires a modification to both velocity and acceleration in the region of influence of obstacles. The component of the Chase's velocity in the direction of the obstacle is determined as

$$\vec{v}_{co} = \frac{\vec{r}_{co} \bullet \left(\vec{v} + \vec{v}_{offset}\right)}{r_{co}} \left(\frac{\vec{r}_{co}}{r_{co}}\right) \tag{7.17}$$

where v_{offset} is the velocity offset correction due to the fact that relative stationary obstacles actually maintain a velocity in the orbital plane. This correction is critical for potential sub centimeter precision maneuver, such as docking. Similarly, the component of the acceleration in the direction of the obstacle is determined as

$$\vec{a}_{co} = \frac{\vec{r}_{co} \bullet \vec{a}_{LQR}}{r_{co}} \left(\frac{\vec{r}_{co}}{r_{co}}\right) \tag{7.18}$$

Using the vector components of the velocity and acceleration in the direction of obstacles helps limit superposition issues. The resulting iterative spacecraft control algorithm is driven by optimal LQR cost convergence, with associated dynamics, and smooth collision avoidance responses.

The APF obstacle potential parameters, represented in (6.22), can be combined to generate a Gaussian function which is unity at the obstacle boundary. This function becomes the LQR velocity shaping function due to obstacle position.

$$k_v = \left(e^{-\left(r_{co}{}^2/\left(2\sigma^2\right)\right)} - e^{-\left(D_o{}^2/\left(2\sigma^2\right)\right)} \right) \left(e^{-\left(L_o{}^2/\left(2\sigma^2\right)\right)} - e^{-\left(D_o{}^2/\left(2\sigma^2\right)\right)} \right)^{-1} \tag{7.19}$$

This gain, k_v, is multiplied by the negative relative velocity to ensure the Chase spacecraft slows to zero at the boundary of the obstacle.

Next, the attractive acceleration due to the LQR recursive function is shaped. There is no change to it when the Chase spacecraft is outside obstacle regions of influence. However, if the Chase is within the region of influence then acceleration toward the obstacle must be decreased. The LQR acceleration shaping parameter is selected as

$$k_a = e^{-d_a\left(r_{co} - L_o\right)} \tag{7.20}$$

where the positive constant, d_a, is used to establish the parameters rate of decay. In this research $d_a = 1$ was selected in order for the decay directly related to relative position. The k_a parameter is multiplied by the negative component of LQR acceleration to ensure that the LQR derived control effort does not drive into an obstacle. Finally, the safety shaping parameter, from Equation (6.26), is modified to replace the potential function with the Chase spacecraft's range from the goal.

$$k_s = 1 - e^{-\left(d_a r_{cg}\right)} \tag{7.21}$$

The safety function allows the obstacle repulsion to decay faster as the Chase spacecraft approaches the goal position. This function enables precision maneuvers, such as docking, in regions where the relative ranges between spacecraft are small. If the obstacle is the Target spacecraft then the safety function ensures that the Chase only approaches in the vicinity of the docking port.

The overall control effort for the multiple spacecraft LQR with collision avoidance is

$$\vec{a} = \vec{a}_{LQR} - \sum_{obs=0}^{n} \left(\left(k_v \vec{v}_{co} / \Delta t \right) + k_s k_a \vec{a}_{co} \right) \tag{7.22}$$

The control algorithm only decreases velocity and acceleration toward obstacles. It does not actually push away from obstacles. This is useful in maintaining the relative stability in a bounded system. Fortunately the relative dynamics result in forces which help the control algorithm escape local minima in densely packed obstacle regions. The consequence is similar to that achieved by APF wall-following methods [4]. The efficiency gained by the LQR derived control effort is more significant when implemented in a limited number of obstacles environment.

C. SENSOR NOISE AND MODEL UNCERTAINTY

In actual applications, the potential field and related gradient are estimates dependent on noisy sensor measurements and spacecraft modeling uncertainties. A realistic system model typically has imperfect initial estimates and dynamic disturbances. Measurements of the system states are limited and imperfect. These issues can be considered in the simulation by applying variance as a measure of the uncertainty. The current state of the spacecraft and the latest sensor measurements can be filtered in order to determine the best estimate of the spacecraft. For instance, the Kalman filter determines this estimate in a minimum mean square error manner [66]. The Kalman filter can be implemented as predictor-corrector algorithm. The filter predicts the next position of the spacecraft and the value of the next sensor measurement. Next, the next sensor measurement is used to determine the residual error between the predicted and

actual position. This residual is multiplied by a weighting gain and used to correct the predicted state to determine the current estimate.

The number, type, and location of sensors are dependent on the degree of maneuver precision required and knowledge of actuation responses. For simple docking mechanisms such as that used on Orbital Express operations architecture, the connection of spacecraft to a hardened docking attachment ring does not require complex manipulator coordination. The entire spacecraft is positioned and oriented for docking. The need for additional sensors is limited to the end-point of the simple docking mechanism. The actuator control is usually limited to translational thruster and attitude reaction wheels (or possibly jets). Thruster actuation variations, such as those mentioned in Chapter III.E.5., can be due to thermal changes and fuel supply levels.

The use of manipulator arms may be necessary for some docking and assembly scenarios. This is especially important for complex on-orbit assembly of structures, such as solar arrays and power stations. As the assembled structure increases in size the flexibility, thermal, and other disturbances acting on the body assembly requires more detailed analysis. If the docking and assembly is conducted with free flying manipulator space robots, then the modal structural characteristics of the structures must be accounted for with high precision and minimum sensors. Research at MIT [67] shows promising progress for extension of the Base Sensor Control (BSC) method for estimation of actuation forces and torques on space robot manipulators with limited sensing. Both the force and torque on a seven DOF bi-arm manipulator can be effectively measured with only one six-axis force/torque sensor [67]. The savings in sensor hardware complexity is highly desirable.

Even docking/assembly missions with manipulator arms must get within range for manipulator arms to operate. These close proximity operations must be conducted within the safe operating ranges established for small spacecraft. The sensors must be precise enough to measure state characteristics at least to the level of precision required for the spacecraft operation. In addition, the actuators must also be able to operate in a manner which allows for the desired operational precision. This may also restrict which actuators can be used at certain distances and orientations. For instance, in close proximity to

another spacecraft thruster firings should be restricted to avoid plum impingement on the sensors and solar panels [68]. These restrictions are dynamic, since they are dependent on the relative position and orientation of the spacecraft. As with any parametric used as a constraint or optimization cost function, trade-offs in performance must be properly weighted. For instance, selection of different collision avoidance, fuel efficiency, plume impingement and time duration considerations will vary the trajectory followed by the spacecraft. Each of these metrics may have different levels of sensitivity to model uncertainties and sensor noise. These considerations in trajectory and path planning raise a large level of concern with the any multiple parameter cost function's claims on optimality. To avoid this level of ambiguity, the close proximity LQR/APF control algorithm uses only relative spatial considerations to determine the direction and rates of spacecraft motion.

Estimations of motion between any object, target or obstacles can be improved by associating sensor measurements with the actual object position. Adopting the general convention that the target states information is assumed to be known, these relative motions can be determined by simple vector arithmetic. The difference between this expected and measured sensor data is considered the innovation [69]. The covariance of the innovation can be directly utilized in determining the collision avoidance thresholds for obstacles. These uncertainties are immediately applicable to the Gaussian based collision avoidance functions and the LQG algorithm which can incorporate process and measurement noise as Gaussian white noises with covariance. The use of Gaussian based collision avoidance lends itself to direct inclusion of any measurement noise in to the obstacle relative range and velocity. Therefore, the overall control algorithm structure allows for convenient inclusion of known, or estimated, sensor uncertainties.

D. STABILITY AND ROBUSTNESS

Stability and robustness of collision avoidance control algorithms are not guaranteed. The stability is highly dependent on the reference frame and specific conditions applied. Generally speaking, once in orbit all passive objects are stable within the gravitational region of the primary body. Only by adding energy can the satellite escape the primary body. However, once the discussion is limited to the relative range

between two orbiting bodies the issue of stability becomes more complicated. The boundary region can typically be drawn such that Chase spacecraft stays within it. However, the open loop relative dynamics of spacecraft are inherently unstable, with non-zero initial conditions and external forces causing spacecraft to drift over time [70]. The A matrix of the linearized dynamics, shown in equation (4.16), yields six eigenvalues consisting of two repeated roots at zero and two repeated imaginary complex pairs. Therefore, the system is bounded-input, bounded-output (BIBO) unstable in the orbital plane and BIBO stable along the out-of-plane axis [71]. It is well known that close loop control can be used to achieve stable and convergent spacecraft relative motion. The stability of distributed spacecraft with individual feedback in a common reference field was rigorously addressed by Kang, Sparks, and Banda [72]. Although, their discussion was primarily based on formation maintenance it directly applies to convergence to any desired relative orbital position. The multiple spacecraft control algorithm balances the convergence to a desired goal position with the safety of collision avoidance. The convergence of the algorithm in a free environment can be explained by using common heuristic techniques and shown via simulation. However, in the presence of obstacles there may not be a unique solution. If the working space is not over constrained, multiple solutions will exist for the spacecraft path in the presence of multiple obstacles. There is also freedom in the closeness of the solution, based on acceptable goal achievement tolerance. Any attempt at a rigid stability proof may need to show convergence within a desired velocity and position ball, such as suggested by Neubauer and Swartwout's research [55]. It is generally possible to select a closed map over which the algorithm is stable and robust. For these closure operations, preliminary initial conditions and simulation parameters must be carefully selected.

Even if a global minimum exists, the spatial dynamics must allow freedom of motion for obstacle avoidance. Obstacles may be arranged, or maneuvered, to block the path of the spacecraft, beyond the capability of the control logic or the available control effort. It the absences of obstacles the multiple spacecraft control algorithm converges to the optimal trajectory. This convergence within a bounded area around an equilibrium state can be shown for each individual spacecraft, but can not be directly applied to the case of multiple spacecraft with uncertainty in relative states. Despite this lack of a

rigorous stability proof, the control algorithm can be shown to be rather robust in the presence of perturbations. The general stability of spacecraft formations in the presence of disturbances was shown by Acikmese *et al* [73]. The stability definition is motivated by BIBO definitions and the propagation of disturbances through the interconnected system. Sensor and actuator uncertainties can always be increased to a level that handicaps the control algorithm. Therefore, in the presence of high levels of noise and uncertainty, additional control logic needs to be implemented to ensure safety. The stability of control in the presence of multiple dynamic obstacles has yet to be proved. In this research global convergence was achieved by representing obstacles with spherical regions of influence. This minimizes the local minimization and chattering problems, since nonlinear dynamics act as persistent perturbations which will require control effort. Any local minimums encountered tend to be momentary and unstable. Also, only allowing obstacle influences to damp relative motion in the direction of the obstacle helps ensure stability. Motion along the path of the obstacle is restricted only to avoid collision, but movement along the convergent direction is unimpeded. This damping allows convergence as long as there is a free path to maneuver around an obstacle toward the goal. Any obstacle avoidance which allows divergent motion within the relative space must take into consideration the compounding effects of orbital rotations and escape velocities.

Spacecraft translation actuation is often discretely implemented in the continuous dynamics of relative motion. It is worth mentioning that there is promising work in the area of hybrid control that lays the framework for evaluation of such systems [70][74]. The hybrid control is well suited to the spacecraft translational problem with flow set (flow map) relating to free-floating continuous time and jump sets (jump maps) relating to control actuation at discrete time. The discussion involves weakly invariant sets and proper selection of Lyapunov functions.

Both the stability and robustness of the close proximity control algorithm is demonstrated implicitly by Monte Carlo simulation results; refer to Chapter IX. There are three conditions which are considered control algorithm failures. These failures are based on the collision avoidance requirement, desire for short duration close proximity maneuvers, and the limited spacecraft propellant. First, any spacecraft collisions are

considered failures. Second, any maneuver durations lasting longer than 90 minutes is considered a failure. Finally, any spacecraft which uses all of its propellant is considered a failure. None of these failures were detected during the Monte Carlo simulations. The close proximity control algorithm is considered stable due to the unfailing convergence to accomplish the desired maneuver. Neither local minimums nor environmental disturbances appeared to prevent Chase spacecraft from achieving desired maneuver positions. Also, the close proximity control algorithm appears to be robust to a wide envelope of initial conditions and configurations.

VIII. CLOSE PROXIMITY OPERATIONS EVALUATION

Control algorithm performance evaluation of a wide range of close proximity operations was conducted. For each maneuver, the multiple spacecraft are initially assumed to be within a spatial sphere with a one kilometer radius and negligible relative velocity to a Target spacecraft in a thrust free circular trajectory. For most close proximity maneuvers results cubic shaped spacecraft were simulated. The cubic shape proved to be the most challenging of the three basic shapes, spherical, cylindrical, and cubic spacecraft shapes for collision avoidance. More complicated shapes may require more conservative safety margins for the simultaneous maneuvering of multiple spacecraft. The communication of the goal, or target, position is the minimum information required. Desired mission ranges for inspection, regions of control for rendezvous, and spatial tolerances for docking may be dependent on both sensor and actuator performance. In particular collision avoidance is a balance between situational awareness, which is dependent upon sensor accuracy, and spacecraft's reaction, which is based upon actuator capabilities.

The primary parameters used to evaluate control algorithm performance are based on the maneuver duration time and required control effort. The duration of time required to successfully accomplish the desired close proximity maneuver is t_d, in seconds. The general desire is that close proximity maneuvers can be conducted in approximately 30 minutes. The control effort is related to the velocity change, Δv in meters per second, required to complete the maneuver. Control effort should be efficient while maintaining desired performance. These two basic figures of merit, t_d and Δv, must be accompanied by engineering analysis, since they are typically inversely related to each other. This is a conceptual simplification since these metrics are a result of the minimization of a cost or potential function with numerous constraints. Evaluation must take into account that any control effort which is severely, or continually, saturated may be a safety hazard in the vicinity of multiple spacecraft. Additional control effort may not be available to perform collision avoidance. Any control effort response that saturates the available control effort is denoted with an asterisk. Also, the overall convergence toward a goal should be

sufficiently damped in order to avoid oscillation in the goal region. Any oscillation in close vicinity to other spacecraft will result in pulsing responses of all other local spacecraft.

Comprehensive performance evaluation of the LQR/APF and APF control algorithms was conducted for a wide range of close proximity operations, including convergence, rally, rendezvous, and docking maneuvers. All simulated maneuvers included orbital perturbations due to variations in the Earth's shape and mass, atmospheric drag on the spacecraft, third-body (Sun and Moon) forces, and solar-radiation pressure. Similarly, the gravity gradient, atmospheric drag, and solar pressure torques were included as perturbations in the attitude control loop. The close proximity operations evaluated all began with Chase spacecraft within 1.0 km of the goal position or Target spacecraft. The convergence maneuver is simply moving to a goal position in free-space. The rally maneuver is gathering of multiple spacecraft to a common goal region in free space. Rendezvous maneuvers require the convergence of multiple spacecraft to a Target spacecraft. Docking maneuvers require precise convergence to the outer boundary of a Target spacecraft while avoiding collision. The parameter results listed for each maneuvers are representative of the general control algorithm performance. It should be understood that improvement to a particular parameter for a selected maneuver may be possible. The following results were generated using the same basic logic and gains for all maneuvers.

A. CONVERGENCE MANEUVERS

Close proximity operations begins when spacecraft independent course corrections and phasing has placed the spacecraft within a kilometer of each other. The convergence maneuver is intended to be a baseline maneuver, without collision avoidance, used for determining control algorithm performance. In this maneuver, the Chase spacecraft maneuvers from its initial location to within 1.0 - 2.0 mm of goal position. Both the recursive LQR/APF and APF control algorithms were successful in converging to within 1.0 mm of a goal position. The convergence maneuver was used to tune algorithms gains for similar performance based on maneuver duration and control

effort efficiency. This terminal range is much less than typically required for general close proximity control and serves to establish legitimacy for application of the control algorithm.

The control algorithm performance results for six independent near and six independent far initial spacecraft positions are listed in Table 8.1 and Table 8.2, respectively. Each one of the near maneuvers starts approximately 100 meters way from a goal; whereas the far maneuvers start approximately 1,000 meters from a goal. Comparison with the two impulse maneuvers previously discussed, yields further insight into control algorithm performance. The first Chase spacecraft, listed in Table 8.1, arrives approximately 15 times faster with over 60 times more control effort than the two-impulse maneuver, which is discussed in Chapter IV.D. This is an extreme case of time and control effort trade-offs due to spacecraft phase dynamics. The second spacecraft, listed in Table 8.1, arrives approximately 3.7 times faster with only 1.5 times more control effort then the two-impulse maneuver, which is discussed in Chapter IV.D. The two-impulse maneuver is used for illustration purposes only, since execution of the two-impulse maneuver with precision necessary for close proximity operations is unfeasible. Realistic control response for multiple spacecraft close proximity maneuvers do not allow for implementation of two-impulse maneuvers. Comparison illustrates parameter trade-offs which must consider while developing multiple spacecraft control algorithms.

Near Convergence Maneuver	**LQR/APF**	**APF**
Near Convergence Initial RSW [0, 70, 0] m	$\Delta v = 0.1877$ m/s	$\Delta v = 0.1905$ m/s
	$t_d = 1041$ s	$t_d = 1264$ s
Near Convergence Initial RSW [50, -100, -50] m	$\Delta v = 0.3105$ m/s	$\Delta v = 0.3123$ m/s
	$t_d = 1068$ s	$t_d = 1298$ s
Near Convergence Initial RSW [100, 100, 100] m	$\Delta v = 0.4900$ m/s	$\Delta v = 0.5121$ m/s
	$t_d = 1068$ s	$t_d = 1317$ s
Near Convergence Initial RSW [100, 0, 0] m	$\Delta v = 0.3077$ m/s	$\Delta v = 0.3215$ m/s
	$t_d = 1056$ s	$t_d = 1284$ s
Near Convergence Initial RSW [-50, 100, -100] m	$\Delta v = 0.3889$ m/s	$\Delta v = 0.3912$ m/s
	$t_d = 1082$ s	$t_d = 1295$ s
Near Convergence Initial RSW [0, 0, 100] m	$\Delta v = 0.2486$ m/s	$\Delta v = 0.2548$ m/s
	$t_d = 1062$ s	$t_d = 1279$ s

Table 8.1 Six Spacecraft Near Convergence Maneuver.

For near convergence maneuvers the LQR/APF control algorithm is more control efficient and reaches the goal location faster then the APF control algorithm. This is as expected since the LQR/APF is not being affected by collision avoidance and is well within the linear performance region of the Clohessy-Wiltshire equations, refer to Equation (4.15). The APF has included a velocity ramping function to limit control saturation due to initial transients, refer to Equation (6.15). This allows for better comparison of performance in relative position, relative velocity, and control effort. The relative position, velocity and control effort performance of the APF control algorithm for the second Chase spacecraft, listed in Table 8.1, are shown in Figure 8.1 through Figure 8.3, respectively. The relative position, velocity and control effort performance of the LQR/APF control algorithm for the same Chase spacecraft are shown in Figure 8.4 through Figure 8.6, respectively.

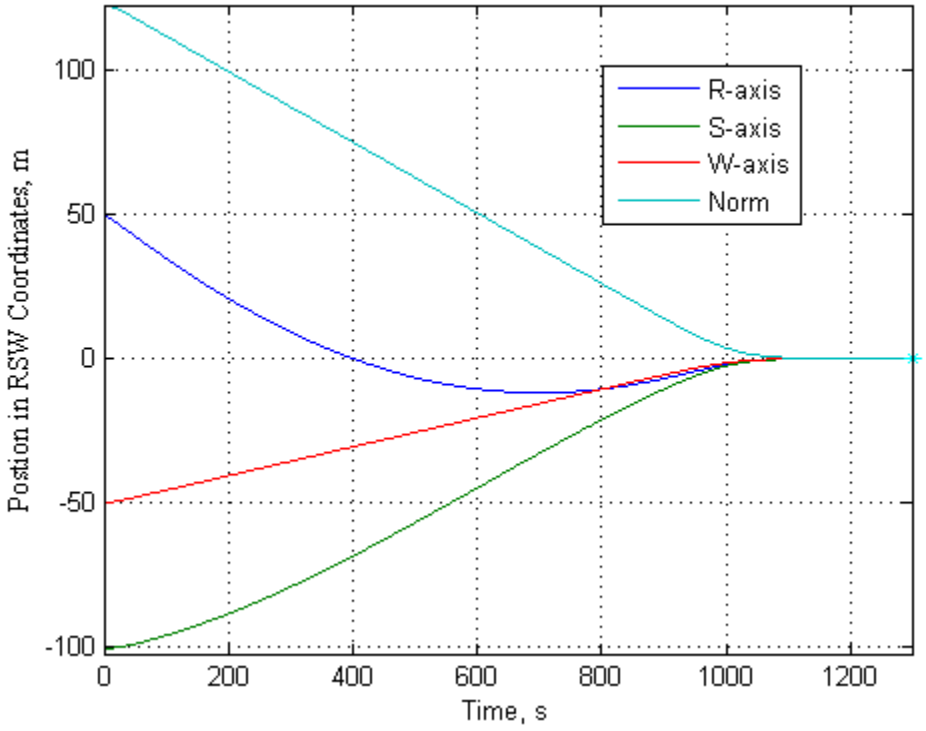

Figure 8.1 Chase Spacecraft Relative Position Using APF for Near Convergence.

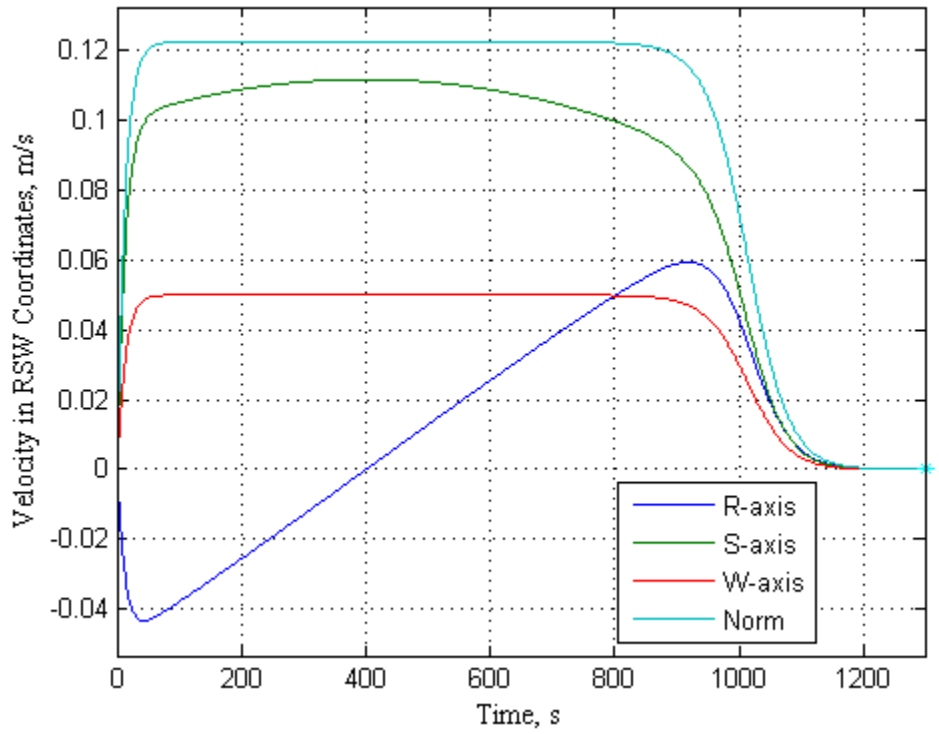

Figure 8.2 Chase Spacecraft Relative Velocity Using APF for Near Convergence.

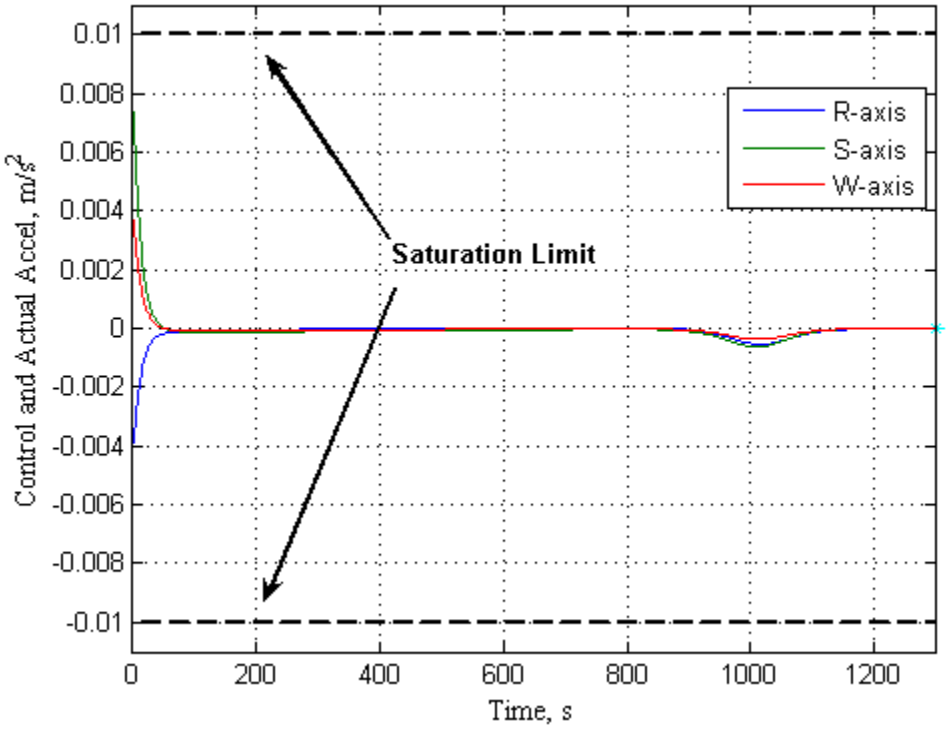

Figure 8.3 Chase Spacecraft Control Effort Using APF for Near Convergence.

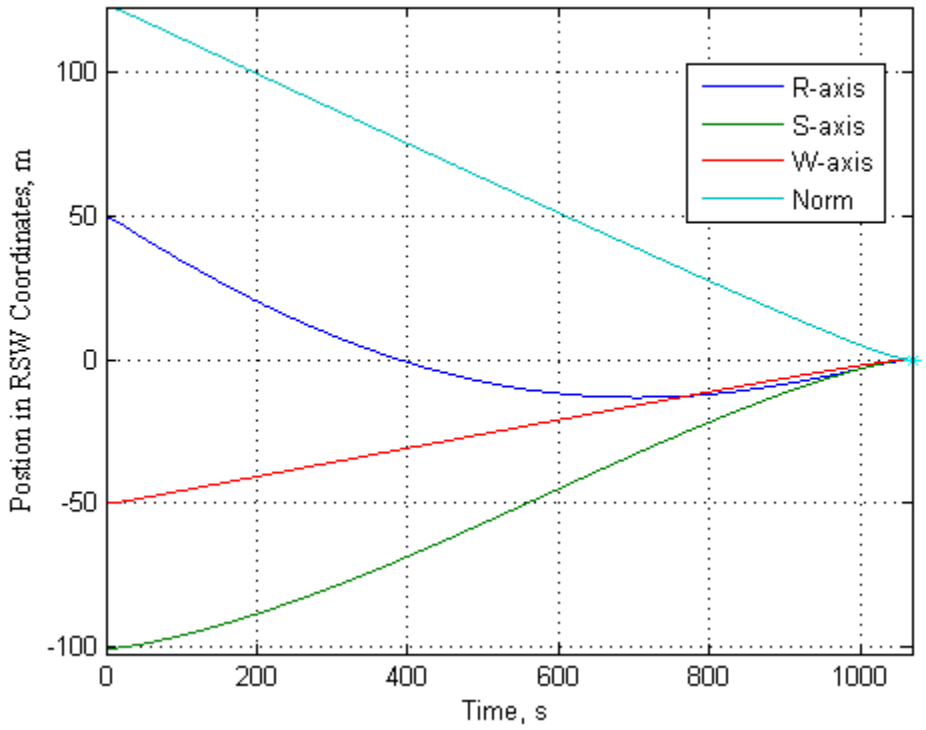

Figure 8.4 Chase Spacecraft Relative Position Using LQR/APF for Near Convergence.

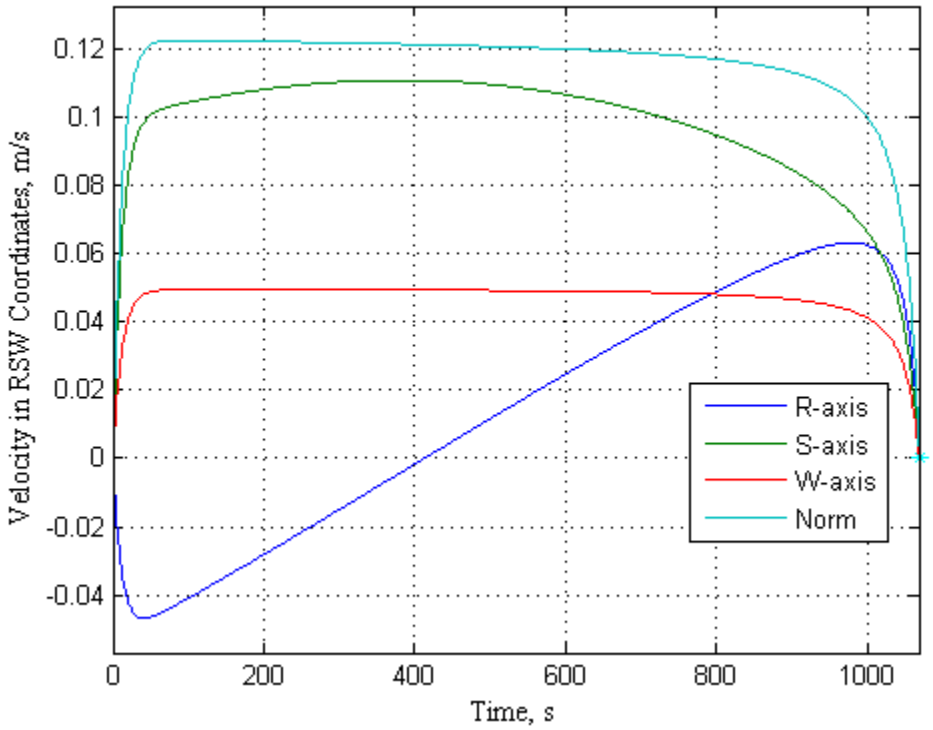

Figure 8.5 Chase Spacecraft Relative Velocity Using LQR/APF for Near Convergence.

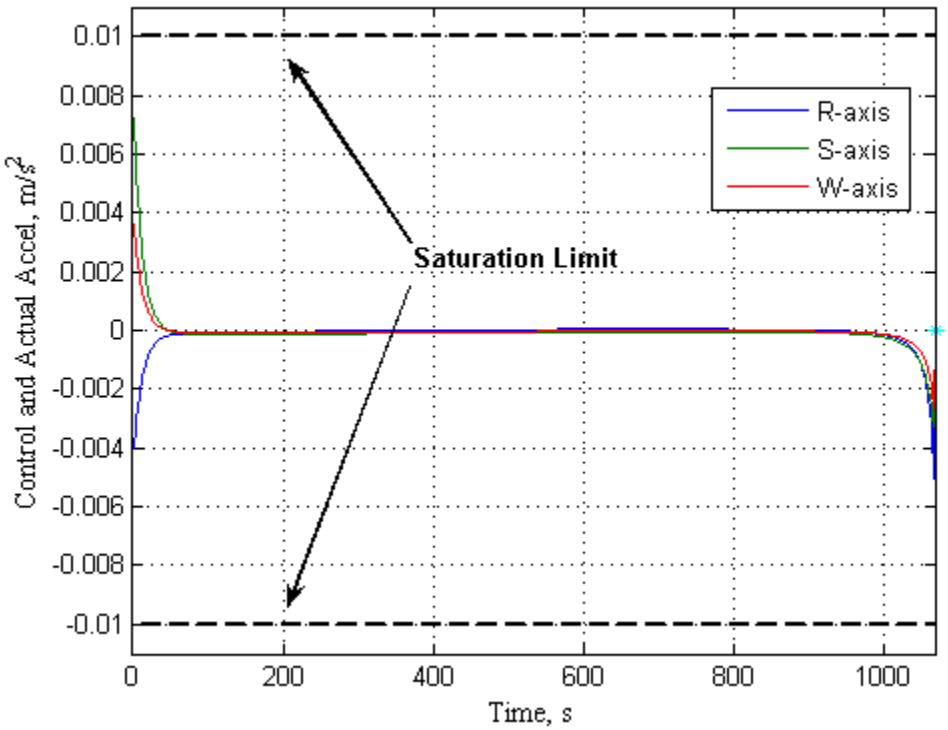

Figure 8.6 Chase Spacecraft Control Effort Using LQR/APF for Near Convergence.

As shown, the LQR/APF and APF can both exhibit the desired close proximity performances. The control effort of the APF algorithm is primarily driven by the desired velocity. This is particularly apparent in the control effort response corresponding to velocity shaping function influence, as shown in Figure 8.3. By comparison, the LQR/APF control effort is balanced by the velocity and position states, as shown in Figure 8.6. The LQR/APF control algorithm generally results in an initial control action to begin convergence and a terminal control effort to stop at the desired goal.

Additionally, far convergence maneuver results are listed in Table 8.2. The relative position, velocity and control effort performance of the APF control algorithm for the second Chase spacecraft, listed in Table 8.2, are shown in Figure 8.7 through Figure 8.9, respectively. The relative position, velocity and control effort performance of the LQR/APF control algorithm for the same Chase spacecraft are shown in Figure 8.10 through Figure 8.12, respectively. The velocity shaping and control effort responses are more pronounced for these far maneuvers figures. Notice that higher velocity is achieved

during the execution of the far convergence maneuver. The APF control response reaches actuator saturation for the fourth Chase spacecraft, as denoted by the asterisk. This is generally an undesirable condition, but may not be completely avoidable in a dynamic environment without underutilizing the actuators.

Far Convergence Maneuver	LQR/APF	APF
Far Convergence Initial RSW [0, 1000, 0] m	$\Delta v = 2.5514$ m/s	$\Delta v = 2.2800$ m/s
	$t_d = 1368$ s	$t_d = 1446$ s
Far Convergence Initial RSW [412,-812,-412] m	$\Delta v = 2.5454$ m/s	$\Delta v = 2.1884$ m/s
	$t_d = 1371$ s	$t_d = 1453$ s
Far Convergence Initial RSW [575,575,575] m	$\Delta v = 2.7375$ m/s	$\Delta v = 2.9295$ m/s
	$t_d = 1369$ s	$t_d = 1459$ s
Far Convergence Initial RSW [1000,0,0] m	$\Delta v = 3.2204$ m/s	$\Delta v = 3.4209$ m/s *
	$t_d = 1367$ s	$t_d = 1449$ s
Far Convergence Initial RSW [0,0,1000] m	$\Delta v = 2.0063$ m/s	$\Delta v = 2.0201$ m/s
	$t_d = 1389$ s	$t_d = 1445$ s
Far Convergence Initial RSW [707,707,0] m	$\Delta v = 2.9865$ m/s	$\Delta v = 3.231$ m/s
	$t_d = 1361$ s	$t_d = 1454$ s

Table 8.2 Six Spacecraft Far Convergence Maneuver.

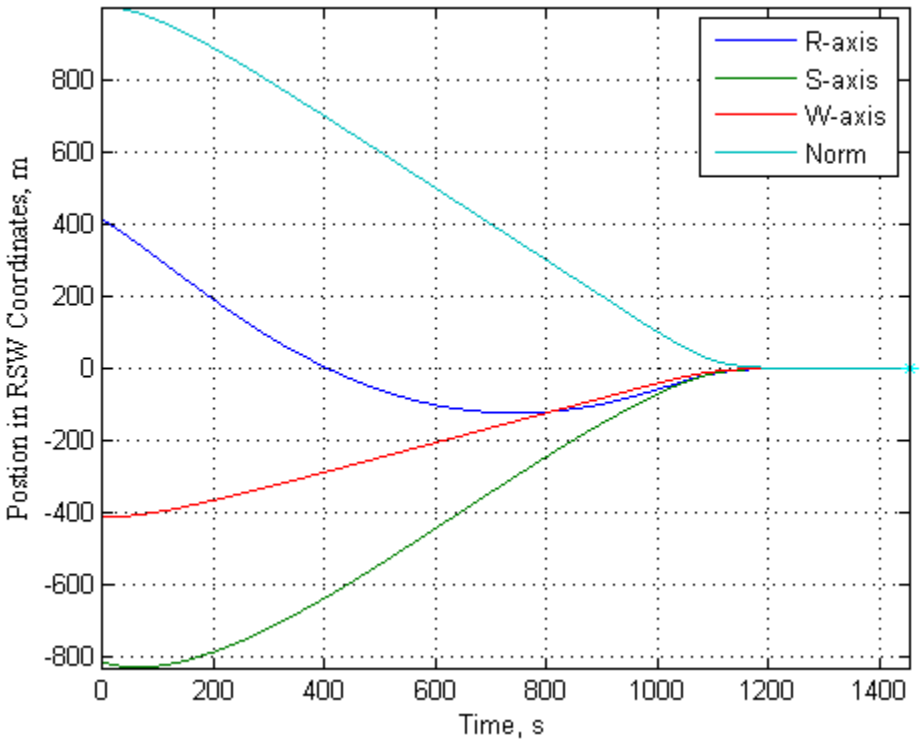

Figure 8.7 Chase Spacecraft Relative Position Using APF for Far Convergence.

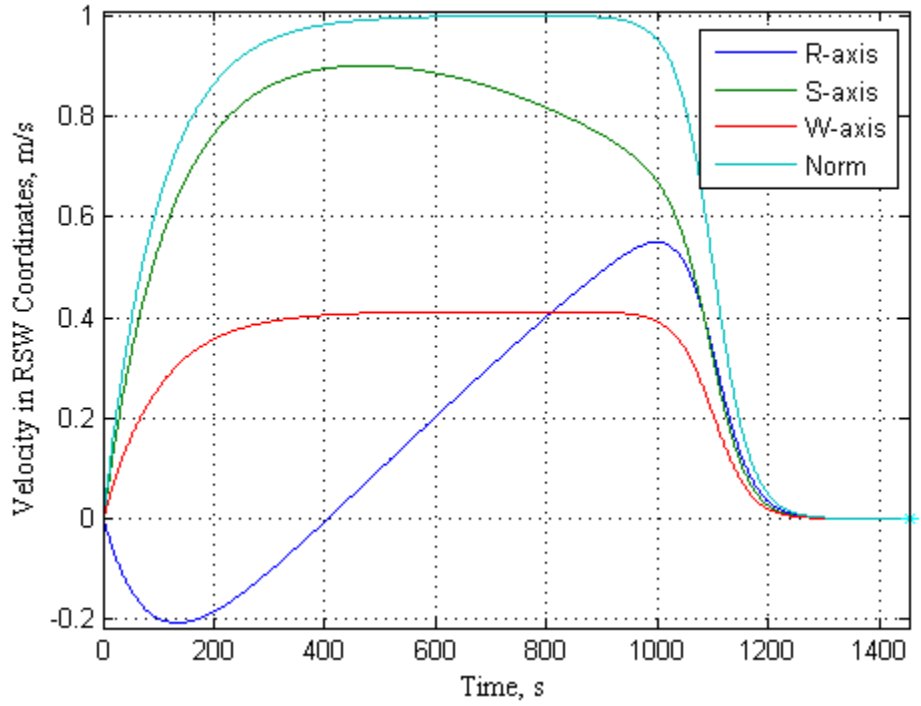

Figure 8.8 Chase Spacecraft Relative Velocity Using APF for Far Convergence.

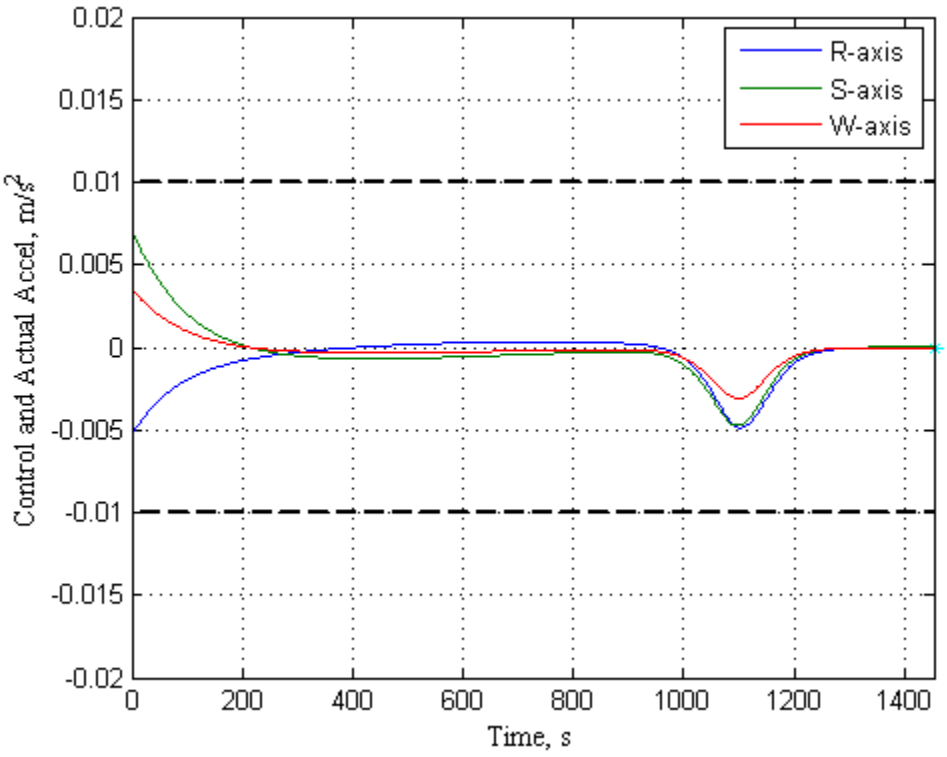

Figure 8.9 Chase Spacecraft Control Effort Using APF for Far Convergence.

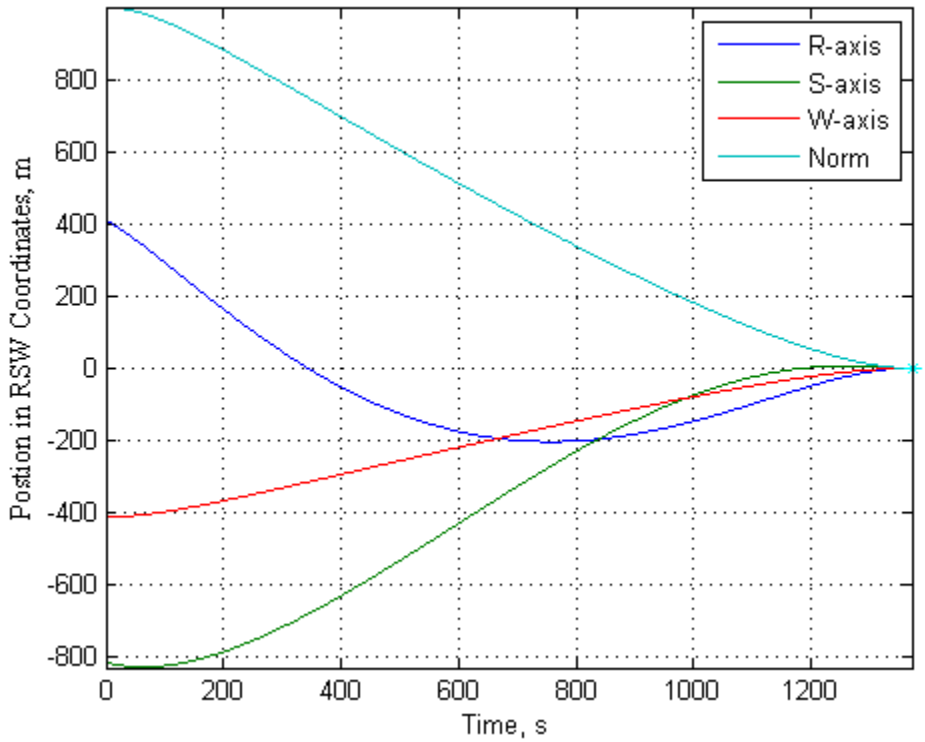

Figure 8.10 Chase Spacecraft Relative Position Using LQR/APF for Far Convergence.

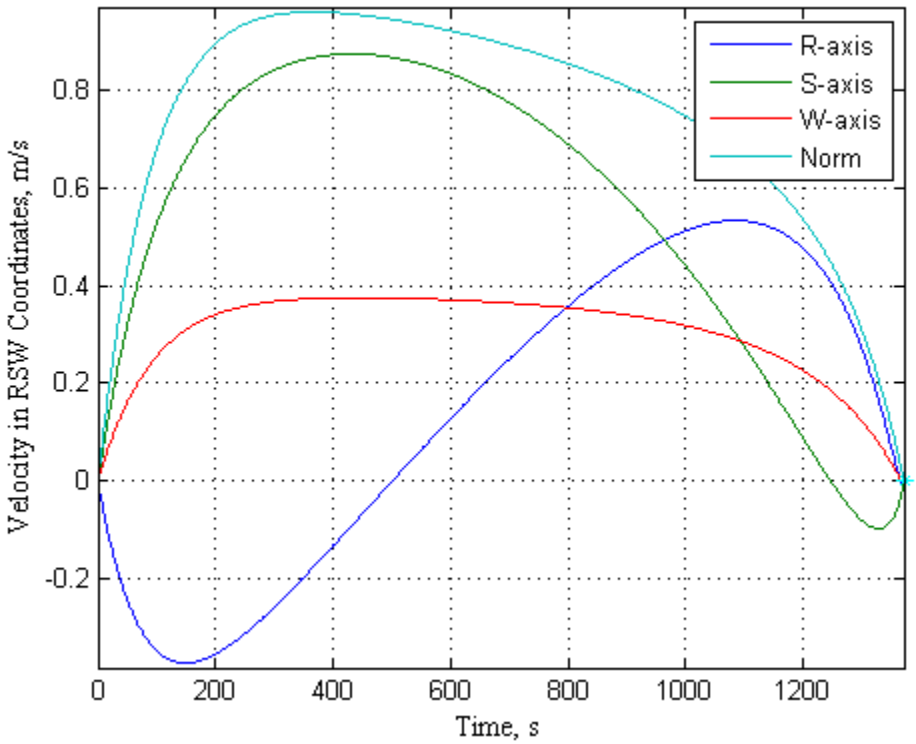

Figure 8.11 Chase Spacecraft Relative Velocity Using LQR/APF for Far Convergence.

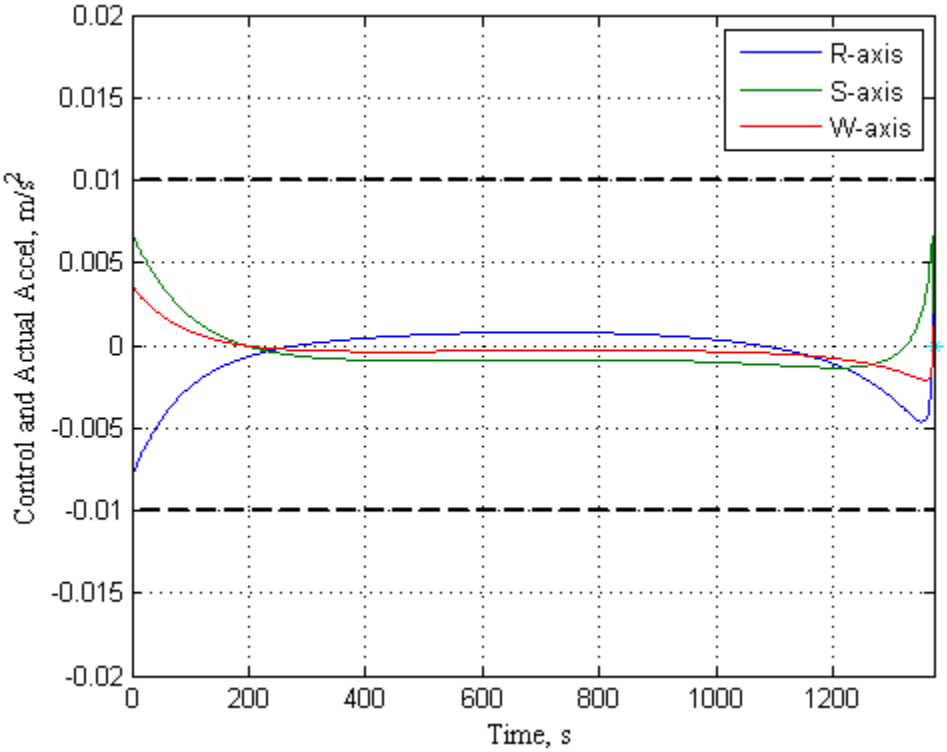

Figure 8.12 Chase Spacecraft Control Effort Using LQR/APF for Far Convergence.

As expected, for both controllers the closer maneuvers took less time to complete. The LQR/APF results are generally more control efficient at similar, or faster, maneuver durations. For the closer rendezvous the multiple spacecraft LQR/APF was better in all metrics. This is primarily due to the LQR consideration of dynamics for the convergence control. In particular, the LQR algorithm results in a smooth initial increase in velocity while the APF requests a step increase in velocity. It was observed that the duration of some more distant maneuvers may be a few seconds shorter using the APF, if the APF initial velocity errors are not ramped. However, this is not considered significant, due to the average 30 minute maneuver duration. The 1.0 millimeter accuracy is used to show the capabilities of the algorithms, although sensor accuracy is not expected to be able to support this level of precision. Preliminary comparison to other APF applications, such as that presented in [52], [40] and [55], indicate that the algorithms accomplish higher precision within much shorter maneuver duration. Improved efficiency is generally accomplished, with direct comparisons strongly dependent on the density of the obstacle environment and initial conditions. Most previous algorithms delineate each phase of sequential maneuvering spacecraft with independent logic determining varying results, as in [54]. This makes the comparison of sequentially and simultaneous multiple spacecraft maneuvers in a dynamic environment difficult and unwarranted. However, this simultaneous and effective maneuvering of several spacecraft may be essential for a number of spacecraft missions.

B. RALLY MANEUVERS

The LQR/APF and APF control algorithms can be used to rally, or cluster, a group of spacecraft to a common point. Collision avoidance is used to avoid obstacles and other spacecraft while converging to within a desired range of a rally point. The results for three cubic Chase spacecraft simultaneously rallying proximity maneuver with collision avoidance are listed in Table 8.3. For these collision avoidance maneuvers, the goal position is a near rally point in free space. This requires the Chase spacecraft to converge to within 0.5 meters of the goal point while avoiding impact with other converging spacecraft. In addition, stationary obstacles are placed at positions along the path of each of the Chase spacecraft, for the sake of testing the algorithms. These

obstacles have an actual diameter of 2.0 meters and are positioned so that the Chase spacecraft encounter them at approximately 500-600 seconds into the maneuver. This point was selected to ensure that the obstacle was encounter while the Chase spacecraft was at relatively high velocity. The three Chase spacecraft motion during the rally maneuver is illustrated in Figure 8.13. Notice that the maneuver duration may lengthen for the later congregating spacecraft, such as the case for the third Chase spacecraft listed in Table 8.3. This is due to other Chase spacecraft arriving and loitering in the rally region. The spacecraft which arrive first may limit the approach of the following spacecraft. This spatial constraint must be considered before determining the acceptable rally region in which the Chase spacecraft are being commanded.

Near Rally Maneuver	**LQR/APF**	**APF**
Near Rally with Obstacle Initial RSW [0, 70, 0] m	$\Delta v = 0.3078$ m/s	$\Delta v = 0.2901$ m/s
	$t_d = 1102$ s	$t_d = 1052$ s
Near Rally with Obstacle Initial RSW [50, -100, -50] m	$\Delta v = 0.5029$ m/s	$\Delta v = 0.4683$ m/s
	$t_d = 1094$ s	$t_d = 1049$ s
Near Rally with Obstacle Initial RSW [100, 100, 100] m	$\Delta v = 0.8306$ m/s	$\Delta v = 0.8188$ m/s *
	$t_d = 1492$ s	$t_d = 1204$ s

Table 8.3 Three Spacecraft Near Rally Maneuver with Collision Avoidance.

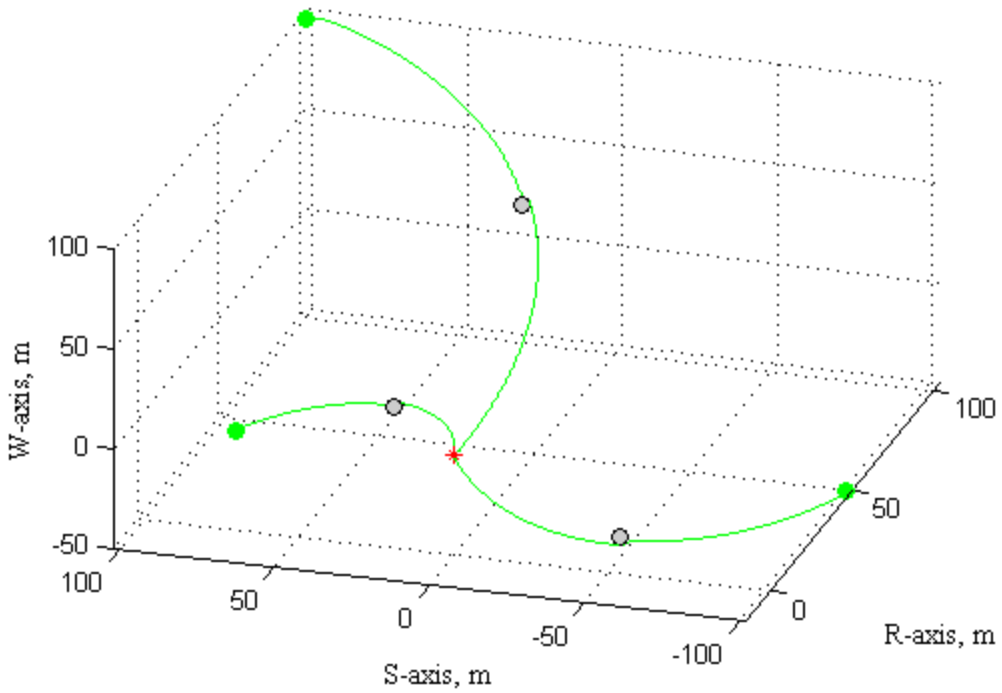

Figure 8.13 Three Spacecraft Relative Motion During Rally Maneuver.

The performance evaluation is once again extended to relatively far initial positions. The results for three cubic Chase spacecraft simultaneously rallying far maneuver with collision avoidance are listed in Table 8.4.

Far Rally Maneuver	LQR/APF	APF
Far Rally with Obstacle Initial RSW [0, 1000, 0] m	$\Delta v = 4.1887$ m/s	$\Delta v = 3.7436$ m/s *
	$t_d = 1615$ s	$t_d = 1243$ s
Far Rally with Obstacle Initial RSW [412, -812, -412] m	$\Delta v = 4.0973$ m/s *	$\Delta v = 3.9324$ m/s *
	$t_d = 1503$ s	$t_d = 1242$ s
Far Rally with Obstacle Initial RSW [575, 575, 575] m	$\Delta v = 4.2541$ m/s	$\Delta v = 4.8076$ m/s *
	$t_d = 1487$ s	$t_d = 1258$ s

Table 8.4 Three Spacecraft Far Rally Maneuver with Collision Avoidance.

In order to further challenge the control algorithms the simultaneous rally maneuvers were conducted with six Chase spacecraft. The near rally maneuver with collision avoidance results are listed in Table 8.5 and the far rally maneuver with collision avoidance results are listed in Table 8.6. In these six spacecraft maneuvers the rally

sphere is extended to 2.0 meters. Such each spacecraft must simultaneously converge to within 2.0 meters of the goal position while avoiding each other. The convergence sphere increases as the number of spacecraft increases, due to practical packing considerations. In realistic operational scenarios, safely margins would be maintained between spacecraft to allow for autonomous activity.

Near Rally Maneuver	LQR/APF	APF
Near with Obstacle Initial RSW [0, 70, 0] m	$\Delta v = 0.3034$ m/s $t_d = 1068$ s	$\Delta v = 0.2745$ m/s $t_d = 1012$ s
Near with Obstacle Initial RSW [50, -100, -50] m	$\Delta v = 0.5363$ m/s $t_d = 1075$ s	$\Delta v = 0.4400$ m/s $t_d = 1017$ s
Near with Obstacle Initial RSW [100, 100, 100] m	$\Delta v = 0.7945$ m/s $t_d = 1108$ s	$\Delta v = 0.7426$ m/s $t_d = 1042$ s
Near with Obstacle Initial RSW [100, 0, 0] m	$\Delta v = 0.4487$ m/s $t_d = 1050$ s	$\Delta v = 0.5370$ m/s $t_d = 1073$ s
Near with Obstacle Initial RSW [-50, 100, -100] m	$\Delta v = 0.7016$ m/s $t_d = 1159$ s	$\Delta v = 0.5875$ m/s $t_d = 1027$ s
Near with Obstacle Initial RSW [0, 0, 100] m	$\Delta v = 0.4646$ m/s $t_d = 1252$ s	$\Delta v = 0.4634$ m/s $t_d = 1162$ s

Table 8.5 Six Spacecraft Near Rally Maneuver with Collision Avoidance.

Far Rally Maneuver	LQR/APF	APF
Far with Obstacle Initial RSW [0, 1000, 0] m	$\Delta v = 4.1598$ m/s $t_d = 1506$ s	$\Delta v = 3.7031$ m/s * $t_d = 1226$ s
Far with Obstacle Initial RSW [412, -812, -412] m	$\Delta v = 4.0729$ m/s $t_d = 1507$ s	$\Delta v = 3.8766$ m/s * $t_d = 1219$ s
Far with Obstacle Initial RSW [575, 575, 575] m	$\Delta v = 4.3607$ m/s $t_d = 1496$ s	$\Delta v = 4.7925$ m/s * $t_d = 1237$ s
Far with Obstacle Initial RSW [1000, 0, 0] m	$\Delta v = 4.8436$ m/s $t_d = 1489$ s	$\Delta v = 4.8994$ m/s * $t_d = 1239$ s
Far with Obstacle Initial RSW [0, 0, 1000] m	$\Delta v = 3.346$ m/s $t_d = 1552$ s	$\Delta v = 3.6122$ m/s * $t_d = 1227$ s
Far with Obstacle Initial RSW [707, 707, 0] m	$\Delta v = 2.7601$ m/s $t_d = 1319$ s	$\Delta v = 3.111$ m/s $t_d = 1168$ s

Table 8.6 Six Spacecraft Far Rally Maneuver with Collision Avoidance.

The first Chase spacecraft to arrive in the goal region use less control effort. They do not need to avoid collisions with other spacecraft, whereas latter arriving spacecraft

may need to maneuver between to complete their rally maneuver. As the number of spacecraft increase direct comparison between spacecraft maneuvers become challenging. Each spacecraft maneuver is dependent upon every other spacecraft and obstacle within region of influence. Generally, the algorithms are comparable with maneuver durations being accomplished within a couple minutes of each other. Efficiency of the algorithms is dependent on the number and location of obstacles encountered and the desired relative velocities. For instance, the APF/LQR algorithm appears to be generally more efficient in a low density obstacle close proximity environment. However, the geometric based velocity management of the APF may converge faster in high density obstacle environments. This faster convergence comes at the risk of saturating the actuators.

The LQR/APF and APF both continue to perform satisfactory in high density obstacle environments. However, the APF continues to saturate control effort. In some instances the saturation actually helps the APF control algorithms control efficiency, such as the first and second Chase spacecraft in Table 8.6. The efficiency appears to be better then the LQR/APF control algorithm, but this is only due to heavy actuator saturation which is generally undesired. The relative position, velocity and control effort performance of the APF control algorithm for the second Chase spacecraft, listed in Table 8.6, are shown in Figure 8.14 through Figure 8.16, respectively. The relative position, velocity and control effort performance of the LQR/APF control algorithm for the same Chase spacecraft are shown in Figure 8.17 through Figure 8.19, respectively.

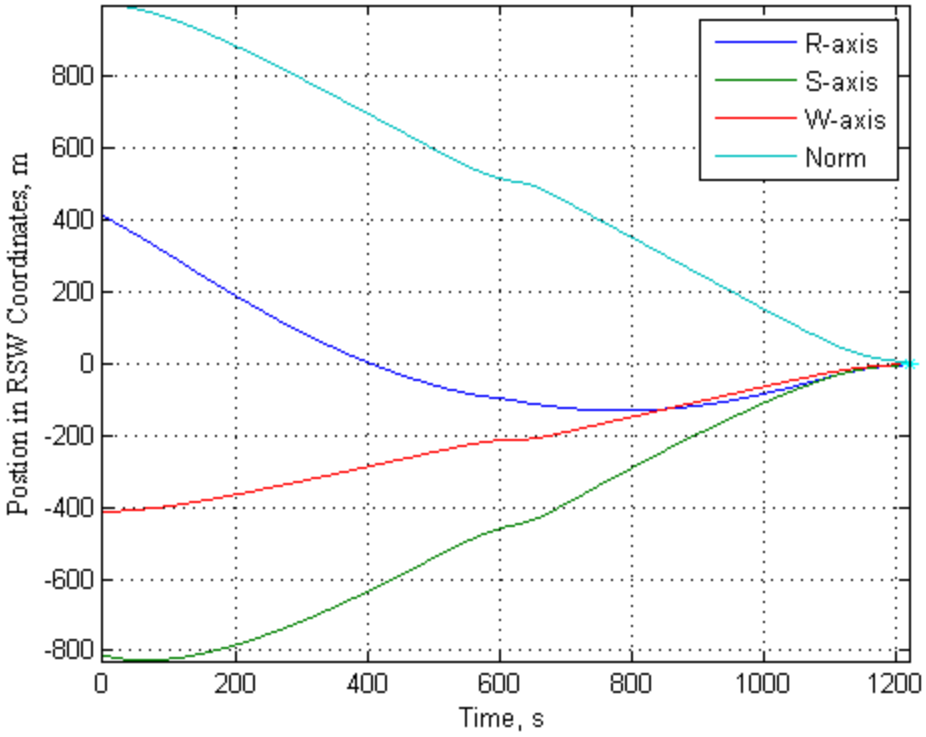

Figure 8.14 Chase Spacecraft Relative Position Using APF for Far Rally.

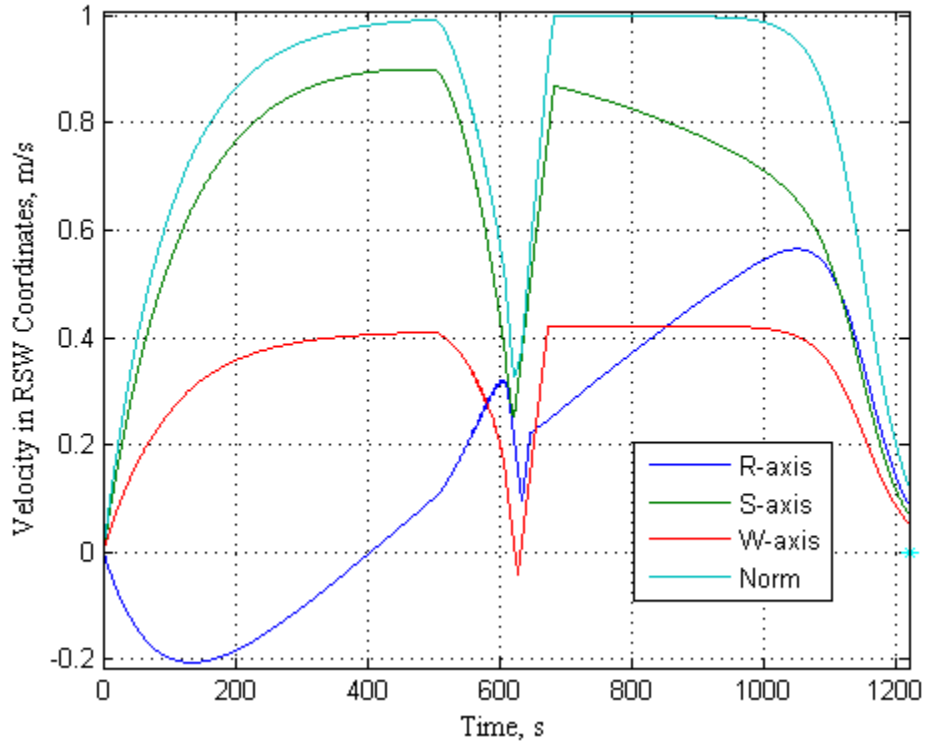

Figure 8.15 Chase Spacecraft Relative Velocity Using APF for Far Rally.

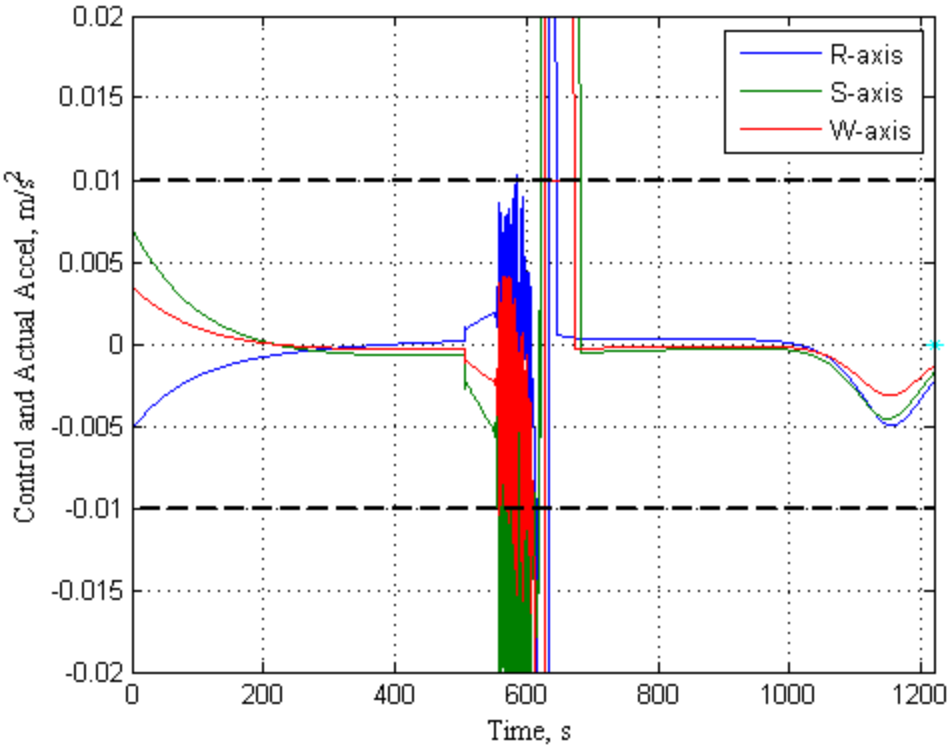

Figure 8.16 Chase Spacecraft Control Effort Using APF for Far Rally.

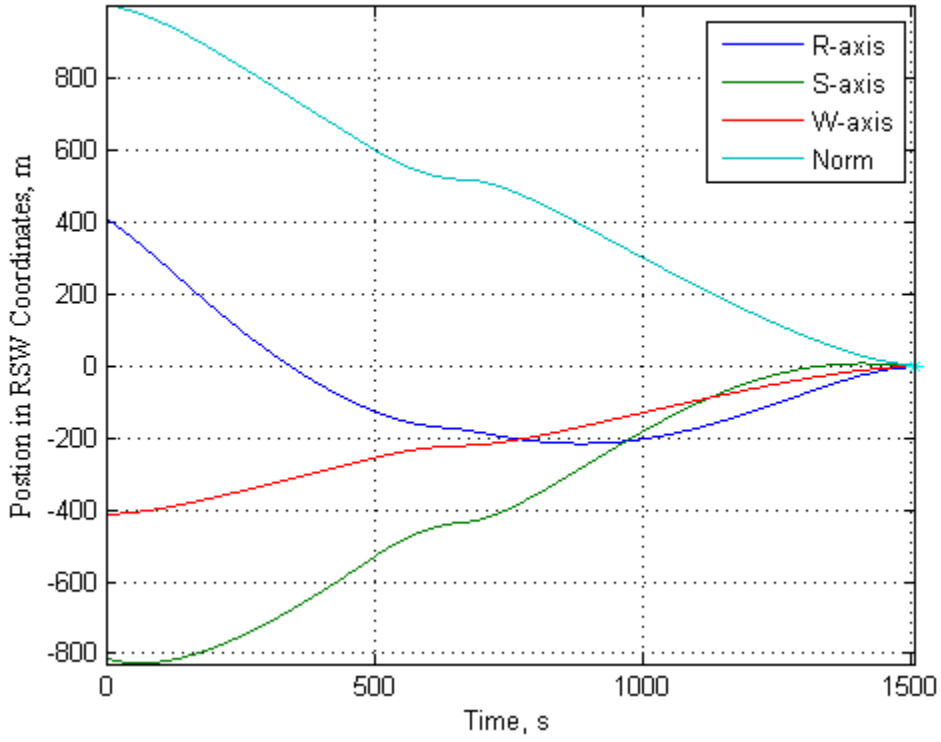

Figure 8.17 Chase Spacecraft Relative Position Using LQR/APF for Far Rally.

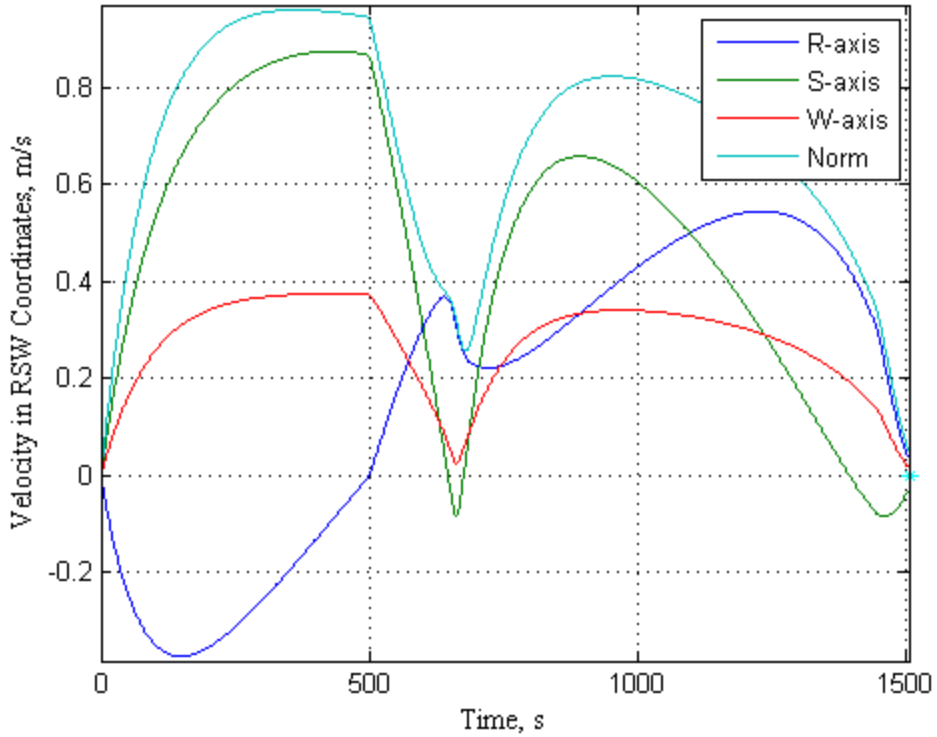

Figure 8.18 Chase Spacecraft Relative Velocity Using LQR/APF for Far Rally.

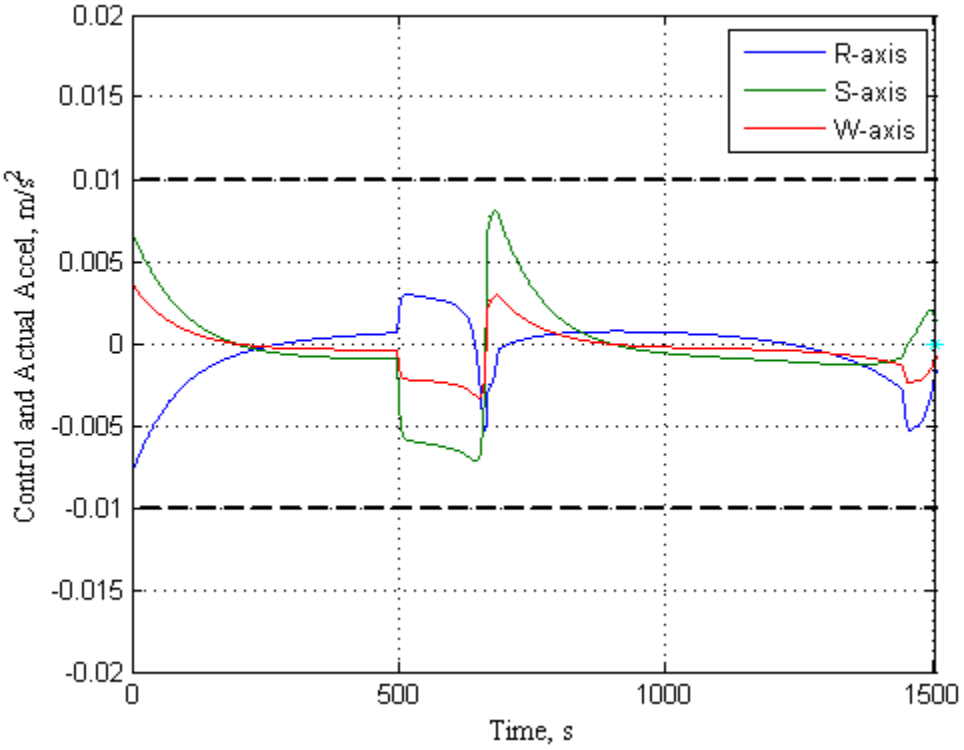

Figure 8.19 Chase Spacecraft Control Effort Using LQR/APF for Far Rally.

These results validate the LQR/APF control algorithm for the multiple spacecraft close proximity maneuvers. The saturation seen in the Chase spacecraft's control effort due to the APF, refer to Figure 8.16, is not exhibited by the LQR/APF, refer to Figure 8.19 The LQR/APF smoothly handles collision avoidance while successfully rallying multiple spacecraft to a desired location.

C. RENDEZVOUS MANEUVERS

Additionally, the LQR/APF and APF control algorithms with collision avoidance are used to avoid obstacles and other spacecraft while converging to within a specified range of a Target spacecraft's outer boundary. For the three Chase spacecraft simultaneous rendezvous maneuvers the Chase spacecraft approach to within 5.0 centimeters of the Target spacecrafts outer surface. The results for the three spacecraft simultaneously conducting a near rendezvous maneuver with collision avoidance are listed in Table 8.7. Similarly, the results for three spacecraft simultaneously conducting a far rendezvous maneuver with collision avoidance are listed in Table 8.8. For these collision avoidance maneuvers, the goal position is the center of the Target spacecraft. This requires that the Target spacecraft's repulsion to allow the Chase spacecraft to converge while avoiding impact. Stationary obstacles are placed at positions along the unobstructed path of the Chase spacecraft. These obstacles are encountered at approximately 500-600 seconds into the maneuver, in order to ensure that the Chase spacecraft was at relatively high velocity. These obstacles have an actual diameter of 2.0 meters. Generally, avoiding larger obstacles requires more control effort and time. The performance accuracy shows that convergence and rendezvous can be achieved while ensuring collision avoidance, as illustrated in Figure 8.20. Although, densely packed obstacle regions tend to result in a superposition of repulsion forces which keep later converging spacecraft at further distances away from the mutual goal location.

Near Rendezvous Maneuver	LQR/APF	APF
Near with Obstacle Initial RSW [0, 70, 0] m	$\Delta v = 0.3886$ m/s	$\Delta v = 0.3339$ m/s
	$t_d = 1342$ s	$t_d = 1103$ s
Near with Obstacle Initial RSW [50, -100, -50] m	$\Delta v = 0.6149$ m/s	$\Delta v = 0.5198$ m/s
	$t_d = 1314$ s	$t_d = 1092$ s
Near with Obstacle Initial RSW [100, 100, 100] m	$\Delta v = 0.8301$ m/s	$\Delta v = 0.8245$ m/s
	$t_d = 1540$ s	$t_d = 1117$ s

Table 8.7 Three Spacecraft Near Rendezvous Maneuver with Collision Avoidance.

Far Rendezvous Maneuver	LQR/APF	APF
Far with Obstacle Initial RSW [0, 1000, 0] m	$\Delta v = 4.1934$ m/s	$\Delta v = 3.7902$ m/s *
	$t_d = 1982$ s	$t_d = 1276$ s
Far with Obstacle Initial RSW [412, -812, -412] m	$\Delta v = 4.1455$ m/s	$\Delta v = 3.9995$ m/s *
	$t_d = 1720$ s	$t_d = 1276$ s
Far with Obstacle Initial RSW [575, 575, 575] m	$\Delta v = 4.4081$ m/s	$\Delta v = 4.8637$ m/s *
	$t_d = 1729$ s	$t_d = 1287$ s

Table 8.8 Three Spacecraft Far Rendezvous Maneuver with Collision Avoidance.

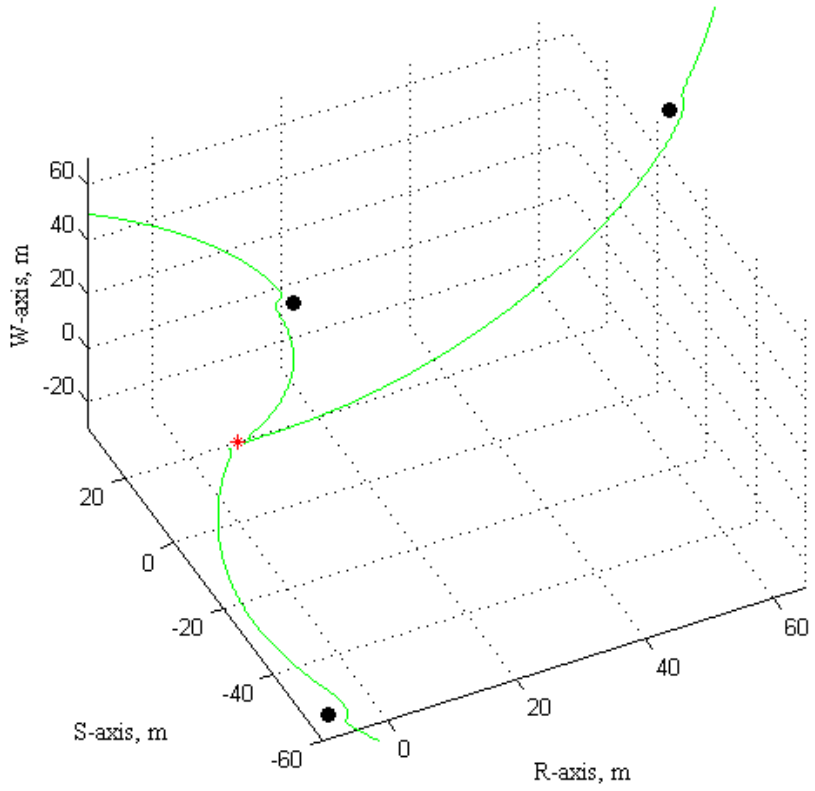

Figure 8.20 Three Spacecraft Collision Avoidance During Rendezvous Maneuver.

In order to further challenge the control algorithms the simultaneous rendezvous maneuvers were conducted with six Chase spacecraft. For the six Chase spacecraft simultaneous rendezvous maneuvers the Chase spacecraft approach to within 1.0 meter of the Target spacecrafts outer surface. The near rendezvous maneuver with collision avoidance results are listed in Table 8.9 and the far rendezvous maneuver with collision avoidance results are listed in Table 8.10.

Near Rendezvous Maneuver	LQR/APF	APF
Near with Obstacle Initial RSW [0, 70, 0] m	$\Delta v = 0.3138$ m/s	$\Delta v = 0.2763$ m/s
	$t_d = 1066$ s	$t_d = 1011$ s
Near with Obstacle Initial RSW [50, -100, -50] m	$\Delta v = 0.5294$ m/s	$\Delta v = 0.4424$ m/s
	$t_d = 1073$ s	$t_d = 1017$ s
Near with Obstacle Initial RSW [100, 100, 100] m	$\Delta v = 0.8097$ m/s	$\Delta v = 0.7362$ m/s
	$t_d = 1116$ s	$t_d = 1041$ s
Near with Obstacle Initial RSW [100, 0, 0] m	$\Delta v = 0.4611$ m/s	$\Delta v = 0.5287$ m/s
	$t_d = 1049$ s	$t_d = 1069$ s
Near with Obstacle Initial RSW [-50, 100, -100] m	$\Delta v = 0.6939$ m/s	$\Delta v = 0.5854$ m/s
	$t_d = 1148$ s	$t_d = 1026$ s
Near with Obstacle Initial RSW [0, 0, 100] m	$\Delta v = 0.4605$ m/s	$\Delta v = 0.4551$ m/s
	$t_d = 1249$ s	$t_d = 1159$ s

Table 8.9 Six Spacecraft Near Rendezvous Maneuver with Collision Avoidance.

Far Rendezvous Maneuver	LQR/APF	APF
Far with Obstacle Initial RSW [0, 1000, 0] m	$\Delta v = 4.1568$ m/s	$\Delta v = 3.7170$ m/s *
	$t_d = 1504$ s	$t_d = 1235$ s
Far with Obstacle Initial RSW [412, -812, -412] m	$\Delta v = 4.0249$ m/s	$\Delta v = 3.9096$ m/s *
	$t_d = 1499$ s	$t_d = 1225$ s
Far with Obstacle Initial RSW [575, 575, 575] m	$\Delta v = 4.3766$ m/s	$\Delta v = 4.8191$ m/s *
	$t_d = 1523$ s	$t_d = 1255$ s
Far with Obstacle Initial RSW [1000, 0, 0] m	$\Delta v = 4.8820$ m/s *	$\Delta v = 4.9273$ m/s *
	$t_d = 1531$ s	$t_d = 1251$ s
Far with Obstacle Initial RSW [0, 0, 1000] m	$\Delta v = 3.3261$ m/s	$\Delta v = 3.6335$ m/s *
	$t_d = 1547$ s	$t_d = 1232$ s
Far with Obstacle Initial RSW [707 ,707, 0] m	$\Delta v = 2.8049$ m/s	$\Delta v = 3.1474$ m/s
	$t_d = 1317$ s	$t_d = 1174$ s

Table 8.10 Six Spacecraft Far Rendezvous Maneuver with Collision Avoidance.

The efficiency of the maneuvers is not as clear in this maneuver due to non-optimal collision avoidance affecting both algorithms. The collision avoidance cause a decrease in acceleration and velocity as the Chase spacecraft approaches an obstacle. The LQR/APF algorithm tends to respond in a smooth manner while avoiding collisions. The APF algorithm tends to saturate the thrusters in the vicinity of obstacles. The rigid desired velocity following characteristic the APF algorithm may allow quick and efficient convergence during some maneuvers. However, there is a greater danger of collision as thrusters saturate and may not be able to respond to additional maneuver demands. The collision avoidance required at the Target spacecraft further decreases the efficiency of the LQR/APF control algorithm by making it stop short of its optimized goal.

As expected, it generally takes longer for spacecraft to cover longer distance. Collision avoidance may further delay rendezvous due to the need to maneuver around obstacles. Notice that the duration of the simultaneous maneuvers increases for the latter arriving spacecraft due to repulsion of the other rendezvous spacecraft. The collision avoidance algorithm logic ensures that converging spacecraft are not perturbed by latter converging spacecraft. This ensures safety in the convergence, but may cause the rendezvous of later spacecraft to be delayed due to the congestion at the shared goal position. For instance, the Chase spacecraft avoiding obstacles can be delayed due to the obstacle avoidance maneuver in free space and experience additional convergence delay due to latter arrival at the rendezvous. This last delay usually affects the third, or more, spacecraft approaching the rendezvous point. This delay can be resolved by dedicating different rendezvous points for each spacecraft. This concept is explicitly be addressed by assigning different goal locations for each spacecraft, such as in docking maneuvers.

D. DOCKING MANEUVERS

The final stage and ultimate goal of rendezvous may be the docking of multiple spacecraft. The two spacecraft docking maneuver is the basis for on orbit servicing and assembly. As multiple spacecraft are required to perform docking maneuvers, several potential complications arise. The docking mechanisms and the docking order need to be addressed. The forces and torque tolerance of the docking mechanism and the overall spacecraft need to be considered. Also, the docking mechanisms must be arranged on

each spacecraft to allow for clear fields of view for sensors and clear approach zones. For spacecraft the docking order may be predetermined. This is typically the case for heterogeneous spacecraft that must be assembled in a specific order or manner. For homogenous spacecraft, the order of docking may not be as important, but there may be limitations due to docking mechanism number and position. Each docking mechanism may be function as a female, male, or both. Also, the spacecraft structure may also limit the docking mechanism placement on the spacecraft. For instance, a possible cubic spacecraft may dock on any of its six sides; refer to Figure 8.21. The dotted lines represent male to female docking orientations. A spacecraft with only one male and female connection is very limited in versatility of assembly scenarios.

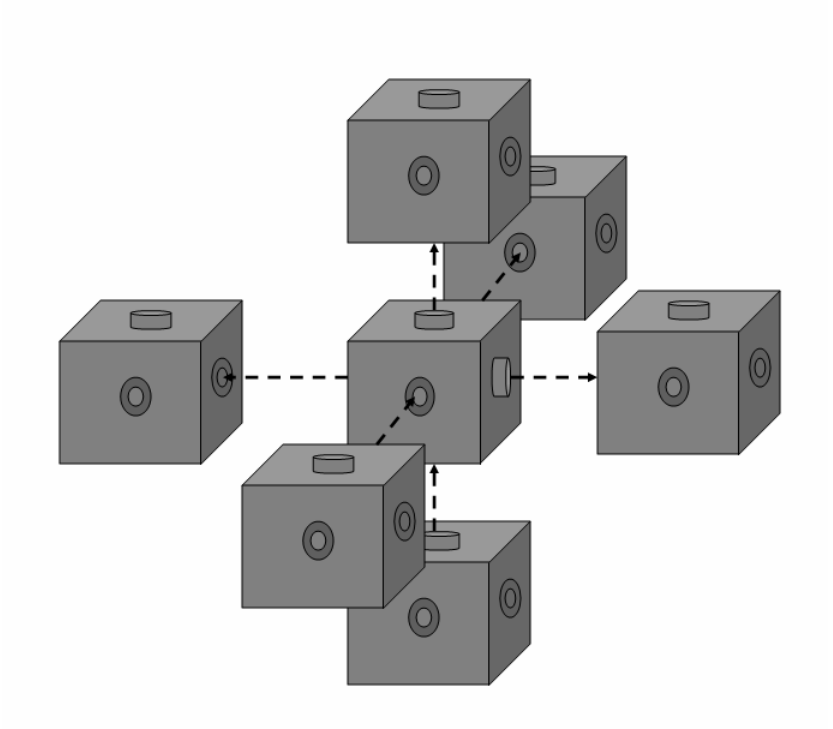

Figure 8.21 Cubic Spacecraft Docking Positions.

The developed LQR/APF and APF control algorithms were evaluated for the docking maneuver. As with the other maneuvers, the docking maneuver time and fuel efficiency are the primary metrics used for evaluating the performance of a control algorithm. Each of Chase spacecraft was assigned a desired goal location on the outer surface of the Target spacecraft. Both LQR/APF and APF control algorithms with collision avoidance are used to avoid obstacles and other spacecraft while converging to

within 2.0 millimeter of the docking position on the Target spacecraft's outer boundary. The results of the three spacecraft near docking proximity maneuver with collision avoidance are listed in Table 8.11. The same initial positions and stationary obstacles were selected for direct comparison. The results of the three spacecraft far docking proximity maneuver with collision avoidance are listed in Table 8.12.

Near Docking Maneuver	LQR/APF	APF
Near with Obstacle Initial RSW [0, 70, 0] m	$\Delta v = 0.3355$ m/s	$\Delta v = 0.2959$ m/s
	$t_d = 1135$ s	$t_d = 1282$ s
Near with Obstacle Initial RSW [50, -100, -50] m	$\Delta v = 0.5694$ m/s	$\Delta v = 0.5397$ m/s
	$t_d = 1118$ s	$t_d = 1296$ s
Near with Obstacle Initial RSW [100, 100, 100] m	$\Delta v = 0.9915$ m/s	$\Delta v = 0.9823$ m/s *
	$t_d = 1355$ s	$t_d = 1386$ s

Table 8.11 Three Spacecraft Near Docking Maneuver with Collision Avoidance.

Far Docking Maneuver	LQR/APF	APF
Far with Obstacle Initial RSW [0, 1000, 0] m	$\Delta v = 4.1575$ m/s	$\Delta v = 3.7561$ m/s *
	$t_d = 1522$ s	$t_d = 1477$ s
Far with Obstacle Initial RSW [412, -812, -412] m	$\Delta v = 4.4056$ m/s	$\Delta v = 4.0500$ m/s *
	$t_d = 2119$ s	$t_d = 1479$ s
Far with Obstacle Initial RSW [575, 575, 575] m	$\Delta v = 4.6149$ m/s	$\Delta v = 5.003$ m/s *
	$t_d = 1878$ s	$t_d = 1531$ s

Table 8.12 Three Spacecraft Far Docking Maneuver with Collision Avoidance.

For sub-centimeter docking precision, the exact relative velocity of the docking port must be taken into consideration. Objects appearing to be stationary in relative position actually have a velocity which directly relates to the orbital rotation of the RSW frame. This velocity offset can limit precession if not determined for all objects appearing relatively stationary in the RSW frame. This velocity offset, v_{offset}, is determined by simply multiplying the docking port's offset position from the Target center by the orbital rotation rate.

$$v_{offset} = \begin{bmatrix} 0 & \omega & 0 \\ -\omega & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{8.1}$$

This resulting velocity is that which is required for an object to appear stationary with respect to the center of the RSW frame. Without this correction for LEO orbits, the resulting convergence error is approximately 3.0-4.0 cm for every meter of offset. This offset velocity correction is only necessary for relatively fixed objects and does not impact freely orbiting objects.

Ultimately the docking mechanism must be sized for the spacecraft. Consideration must be made for forces and torques due to manner of docking, such as grappling, magnetic, or male/female. The final position and velocity error control algorithm must be within the tolerance of the docking mechanism. For instance, if the final position is expected to be within 2 cm then the docking mechanism may need to have an adaptor with a radius of 2 cm plus the size of the docking lever; refer to Figure 8.22. The velocity error of the final position will result in forces and torques imparted between spacecraft as they make contact. This docking impact may result in transient translational and rotations which need to be controlled. For instance, even if a mother spacecraft can support the docking of multiple spacecraft, the docking order may need to be staggered in order to allow the assembled space structure to stabilize after each docking. This is less significant for small spacecraft docking to relatively more massive spacecraft.

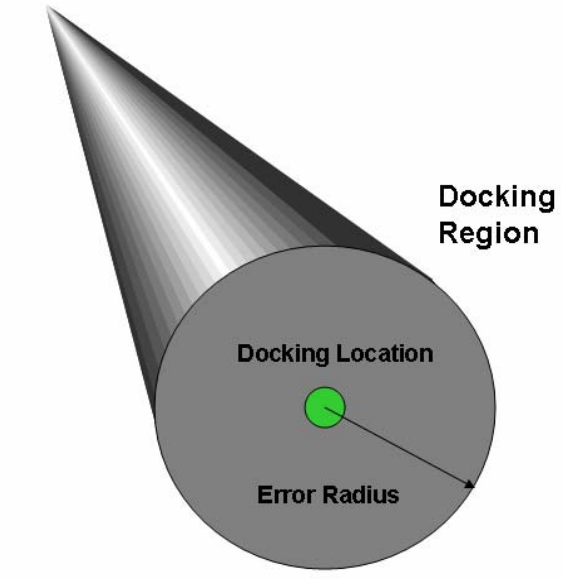

Figure 8.22 Spacecraft Docking Region.

The control algorithms are applied to the more complicated maneuver involving six Chase spacecraft simultaneously docking to a single Target spacecraft. For the six Chase spacecraft docking maneuvers, the ports are centered on each side of a cubic Target spacecraft at RSW locations of [1, 0, 0], [0, -1, 0], [0, 1, 0], [-1, 0, 0], [0, 1, 0], and [0, -1, 0], respectively. The docking locations were randomly assigned and not related to the initial positions of the Chase spacecraft. This allows for robustness evaluation of the control algorithm, since it requires that each Chase spacecraft maneuver in close proximity. Both LQR/APF and APF control algorithms with collision avoidance are used to avoid obstacles and other spacecraft while converging to within 2.0 millimeter of their assigned docking position on the Target spacecraft's outer boundary. The results for six Chase spacecraft conducting simultaneously near docking maneuvers, with collision avoidance, are listed in Table 8.13. The results for six Chase spacecraft simultaneously conducting far docking maneuvers, with collision avoidance, are listed in Table 8.14.

Near Docking Maneuver	**LQR/APF**	**APF**
Near with Obstacle Initial RSW [0, 70, 0] m	$\Delta v = 0.3355$ m/s	$\Delta v = 0.2959$ m/s
	$t_d = 1135$ s	$t_d = 1282$ s
Near with Obstacle Initial RSW [50, -100, -50] m	$\Delta v = 0.5694$ m/s	$\Delta v = 0.5648$ m/s *
	$t_d = 1118$ s	$t_d = 1304$ s
Near with Obstacle Initial RSW [100, 100, 100] m	$\Delta v = 0.9915$ m/s	$\Delta v = 1.1742$ m/s *
	$t_d = 1355$ s	$t_d = 1516$ s
Near with Obstacle Initial RSW [100, 0, 0] m	$\Delta v = 0.6902$ m/s	$\Delta v = 0.7264$ m/s *
	$t_d = 1307$ s	$t_d = 1444$ s
Near with Obstacle Initial RSW [-50, 100, -100] m	$\Delta v = 0.9969$ m/s	$\Delta v = 0.7238$ m/s *
	$t_d = 1521$ s	$t_d = 1397$ s
Near with Obstacle Initial RSW [0, 0, 100] m	$\Delta v = 0.7399$ m/s	$\Delta v = 0.6944$ m/s *
	$t_d = 1794$ s	$t_d = 1923$ s

Table 8.13 Six Spacecraft Near Docking Maneuver with Collision Avoidance.

Far Docking Maneuver	LQR/APF	APF
Far with Obstacle Initial RSW [0, 1000, 0] m	$\Delta v = 4.1792$ m/s	$\Delta v = 3.7513$ m/s *
	$t_d = 1530$ s	$t_d = 1478$ s
Far with Obstacle Initial RSW [412, -812, -412] m	$\Delta v = 4.5243$ m/s	$\Delta v = 4.1084$ m/s *
	$t_d = 1870$ s	$t_d = 1488$ s
Far with Obstacle Initial RSW [575,575, 575] m	$\Delta v = 4.7083$ m/s	$\Delta v = 5.2832$ m/s *
	$t_d = 1719$ s	$t_d = 1549$ s
Far with Obstacle Initial RSW [1000, 0, 0] m	$\Delta v = 4.9268$ m/s *	$\Delta v = 5.2024$ m/s *
	$t_d = 1520$ s	$t_d = 1602$ s
Far with Obstacle Initial RSW [0, 0, 1000] m	$\Delta v = 3.6151$ m/s	$\Delta v = 3.8136$ m/s *
	$t_d = 1678$ s	$t_d = 1496$ s
Far with Obstacle Initial RSW [707, 707, 0] m	$\Delta v = 3.0789$ m/s	$\Delta v = 5.1804$ m/s *
	$t_d = 1463$ s	$t_d = 1509$ s

Table 8.14 Six Spacecraft Far Docking Maneuver with Collision Avoidance.

The relative position, velocity and control effort performance of the APF control algorithm for the second Chase spacecraft, listed in Table 8.14, are shown in Figure 8.23 through Figure 8.25, respectively. The relative position, velocity and control effort performance of the LQR/APF control algorithm for the same Chase spacecraft are shown in Figure 8.26 through Figure 8.28, respectively. These responses are similar to those seen for previous maneuvers, which illustrate the general similarity in multiple spacecraft close proximity maneuvers. Although, additional control response is often needed in the later stages of precision docking maneuvers, refer to the control effort response in Figure 8.28.

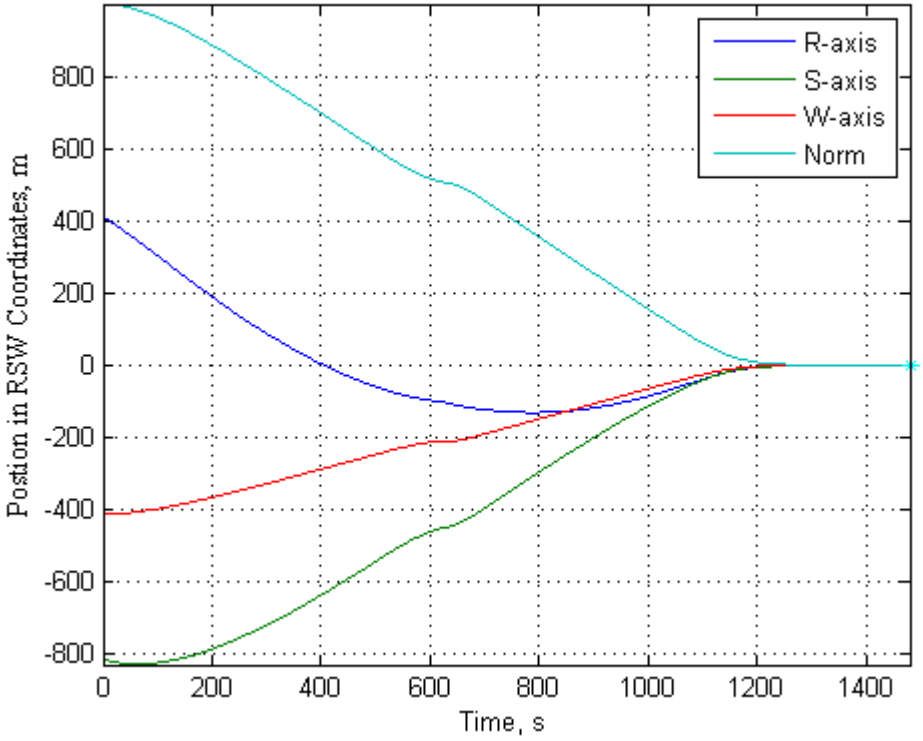

Figure 8.23 Chase Spacecraft Relative Position Using APF for Far Rally.

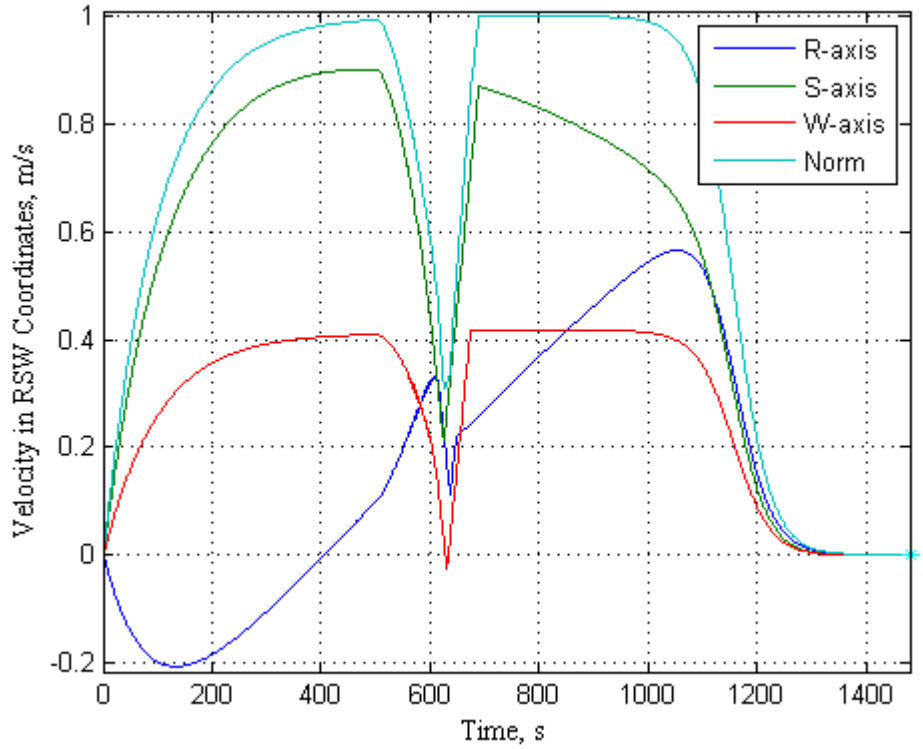

Figure 8.24 Chase Spacecraft Relative Velocity Using APF for Docking.

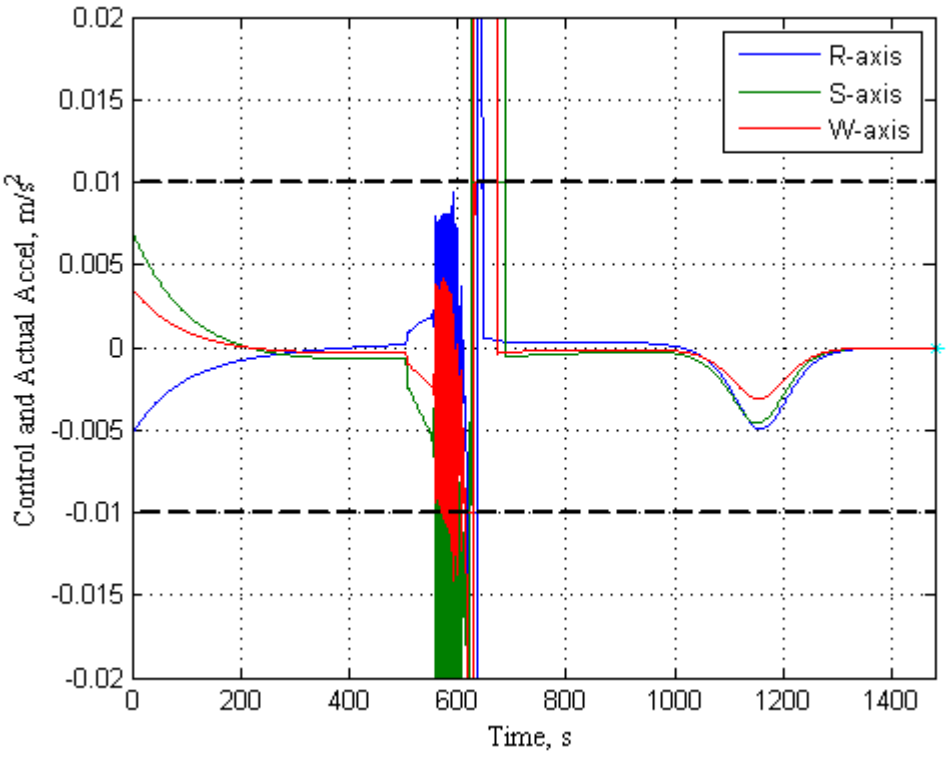

Figure 8.25 Chase Spacecraft Control Effort Using APF for Far Docking.

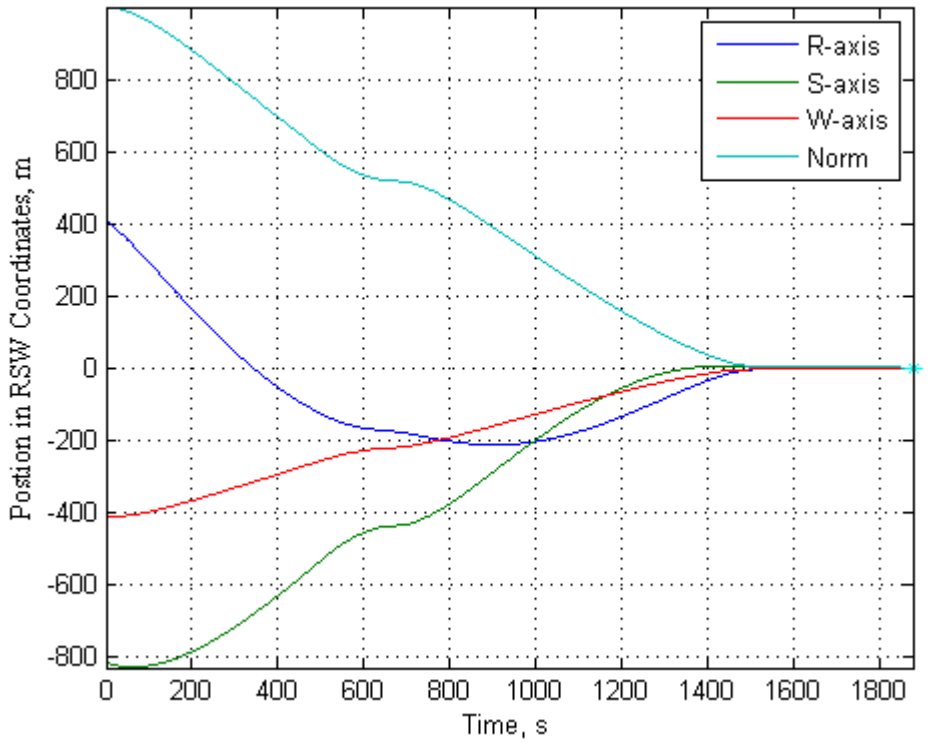

Figure 8.26 Chase Spacecraft Relative Position Using LQR/APF for Far Docking.

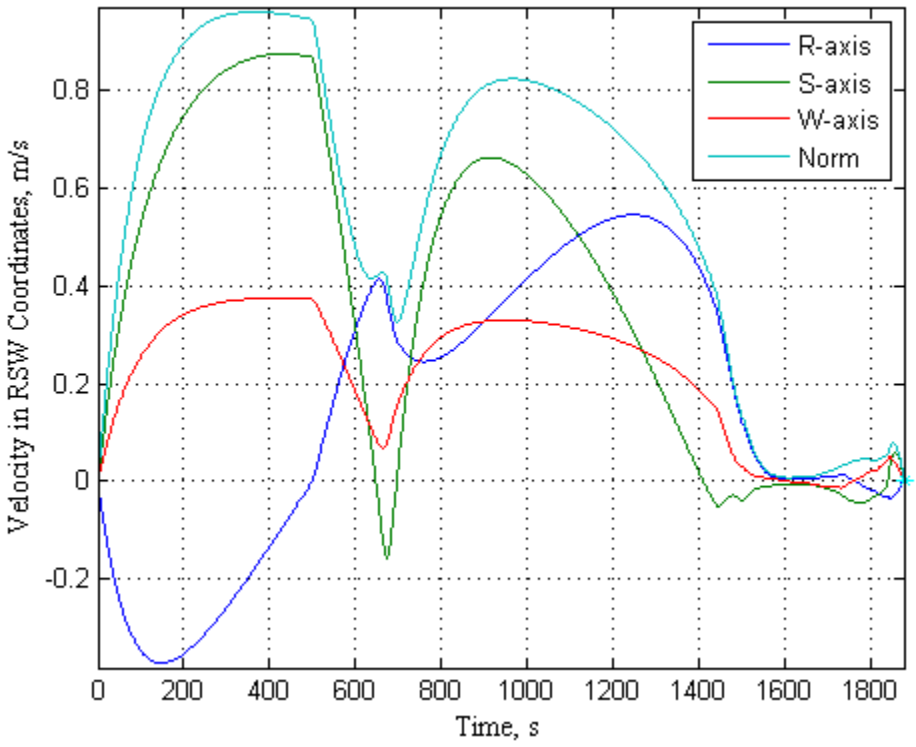

Figure 8.27 Chase Spacecraft Relative Velocity Using LQR/APF for Far Docking.

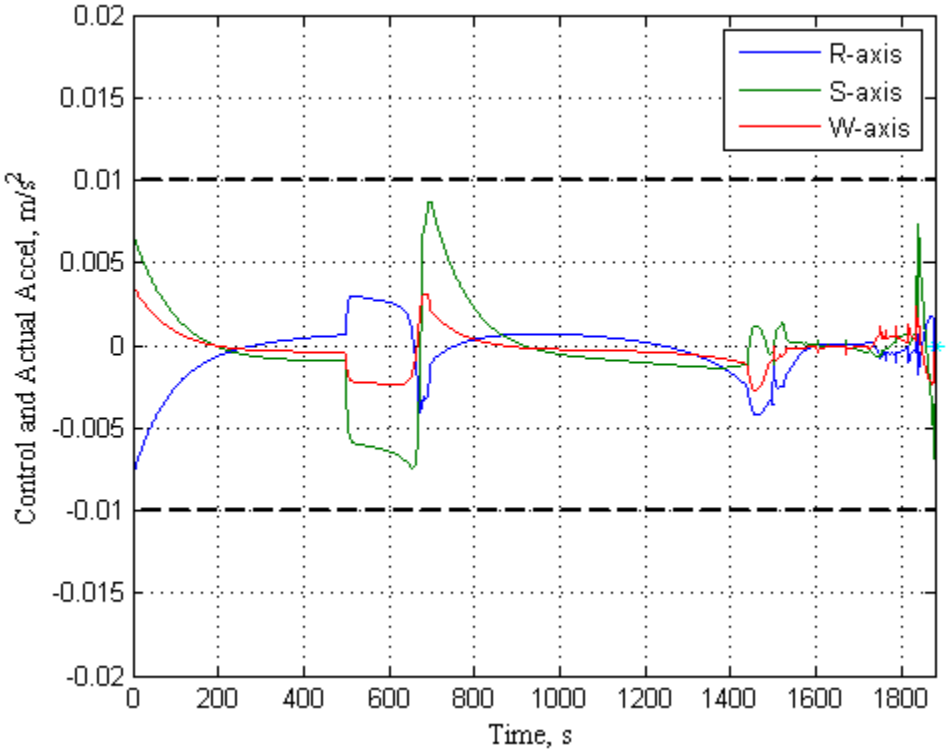

Figure 8.28 Chase Spacecraft Control Effort Using LQR/APF for Far Docking.

The most interesting part of multiple spacecraft docking is best shown with animated graphic. This allows evaluation of control logic and convergence performance with respect to how each spacecraft maneuvers with relationship to both time and space. Without animation, a basic assessment of the LQR/APF and APF controller's relative position performance can be illustrated with a path comparison. The relative path of the second Chase spacecraft, from Table 8.13 and Table 8.14, is shown in Figure 8.29. The commanded path for each of the control algorithms is obviously different, with variations having cascading effects for collision avoidance with other spacecraft. The stationary obstacle collision avoidance and docking regions are highlighted in Figure 8.30 and Figure 8.31, respectively. The stationary obstacle avoidance response is straight forward; however the docking regions path requires further explanation. The Chase spacecraft arrives at the Target spacecraft in a different order based on LQR/APF or APF control. Using the APF, this Chase spacecraft arrives as the second spacecraft in the docking region, so modest collision avoidance in the docking region is required. For the LQR/APF this particular Chase spacecraft arrives last and must fully employ collision avoidance. Variations in commanded path, arriving sequence, and relative position of the other Chase spacecraft make one to one performance comparison challenging. The control behavior of each Chase spacecraft may appear to be intelligent behavior, but is actually the result of the control algorithm's basic computation and logic. These control algorithms can be expanded to handle additional logical situations of particular maneuvers.

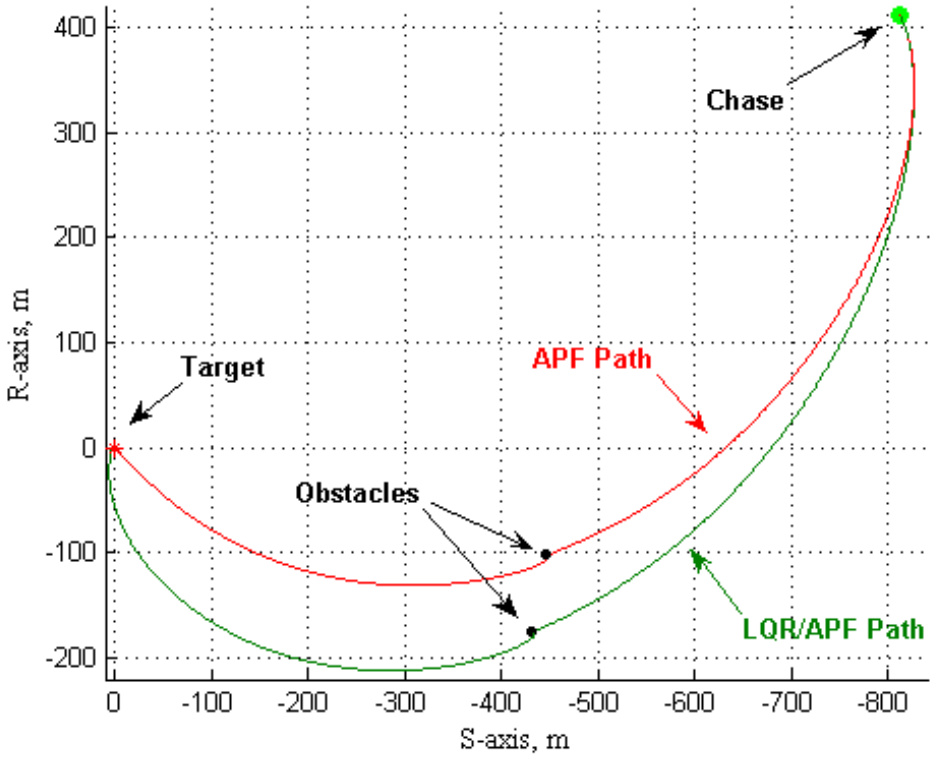

Figure 8.29 LQR/APF and APF Path Comparison.

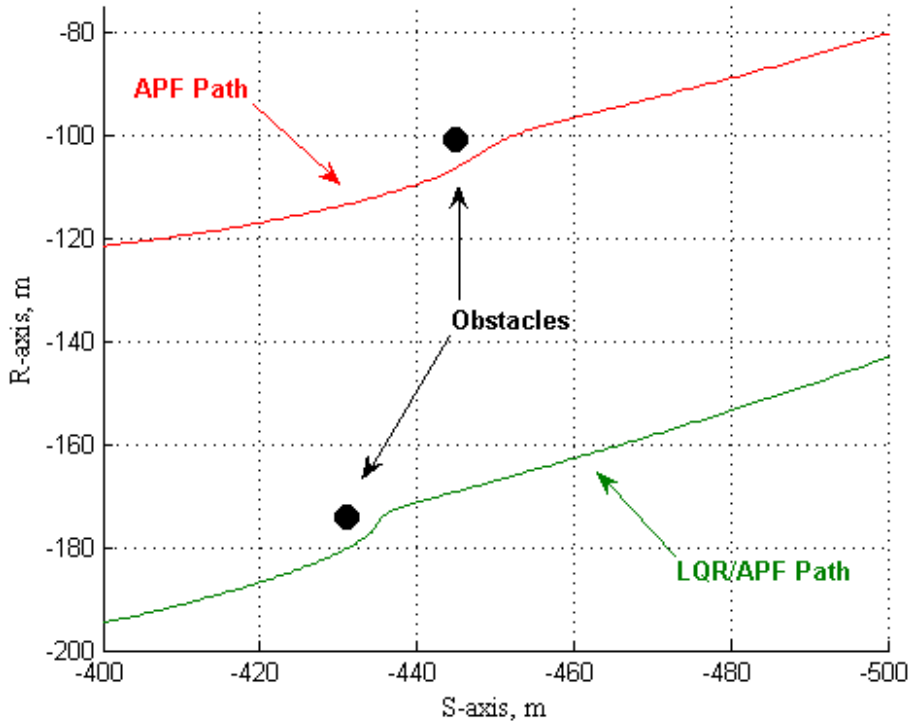

Figure 8.30 LQR/APF and APF Collision Avoidance Path Comparison.

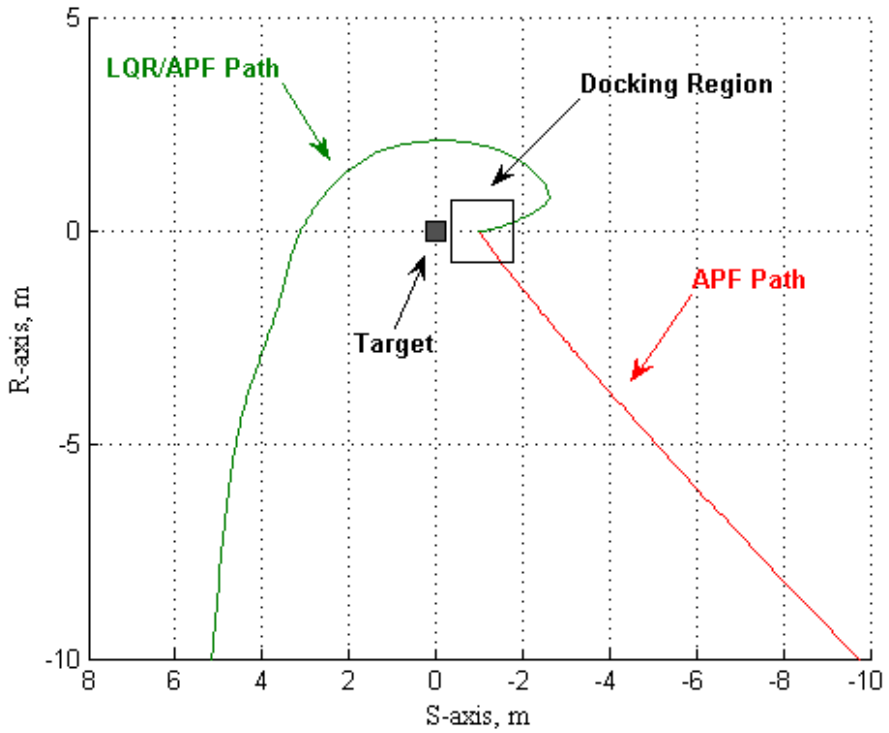

Figure 8.31 LQR/APF and APF Docking Region Path Comparison.

Selecting which spacecraft should dock at which docking mechanism may be a function of a higher order control algorithm. For instance a docking assignment algorithm may select the closest relative spacecraft to each docking mechanism that is available. Multiple spacecraft docking assignment depends on the mission and characteristic of each spacecraft. One example of a candidate task assignment algorithm was developed in [1] and could be adapted for the docking task. This algorithm is based on dynamics programming and allows for the spacecraft order to be based on fuel efficiency of the maneuver. This algorithm can be readily adapted to potential function relative positioning without a centralized computation and a simple communication protocol. However, once a hierarchical tasking algorithm is applied additional communication and computation may be required.

E. MANEUVER EVALUATION CONCLUSIONS

Based on the wide range of close proximity maneuvers evaluated, the LQR/APF control algorithm is an excellent candidate for multiple spacecraft close proximity operations. The convergence rate and control effort efficiency is comparable to that of a highly tuned APF control algorithms. The incorporation of the linearized dynamics in the LQR/APF control algorithm allows it to overcome limitations of APF control algorithms for actual implementation. The iterative calculations allow for practical computation for real time applications. The LQR/APF simulations demonstrate successful completion of multiple spacecraft conducting simultaneous, rally, rendezvous, and docking maneuvers. Inclusion of high fidelity spacecraft perturbation forces during the simulations proves that the LQR/APF algorithm is also effective in disturbance rejection.

The control effort expended during the close proximity maneuver can be related to the quantity of propellant used. Spacecraft have an estimated margin of 3-6% for propellant mass, refer to Table 3.2. Based on 100 kg spacecraft, the initial propellant mass would be at least 3.0 kg. The far simultaneous docking maneuvers are potential the most control effort intensive maneuvers. The control effort for the six Chase spacecraft docking maneuvers, refer to Table 8.14, are represented in propellant mass in Figure 8.32. From this perspective, a Chase spacecraft could perform about ten far docking maneuvers. This is an extremely conservative estimate based on far initial positions, short maneuver duration, modest volume of propellant, and a dense obstacle environment. The average Chase should be able to perform several close proximity maneuvers. Therefore, the LQR/APF algorithm appears to be a promising new development for the field of multiple spacecraft close proximity maneuver control. Monte-Carlo method analysis allows for reasonable statistical estimates of the mean and standard deviation of maneuver t_d and Δv. Refer to Chapter IX for multiple spacecraft control algorithm Monte Carlos simulation results.

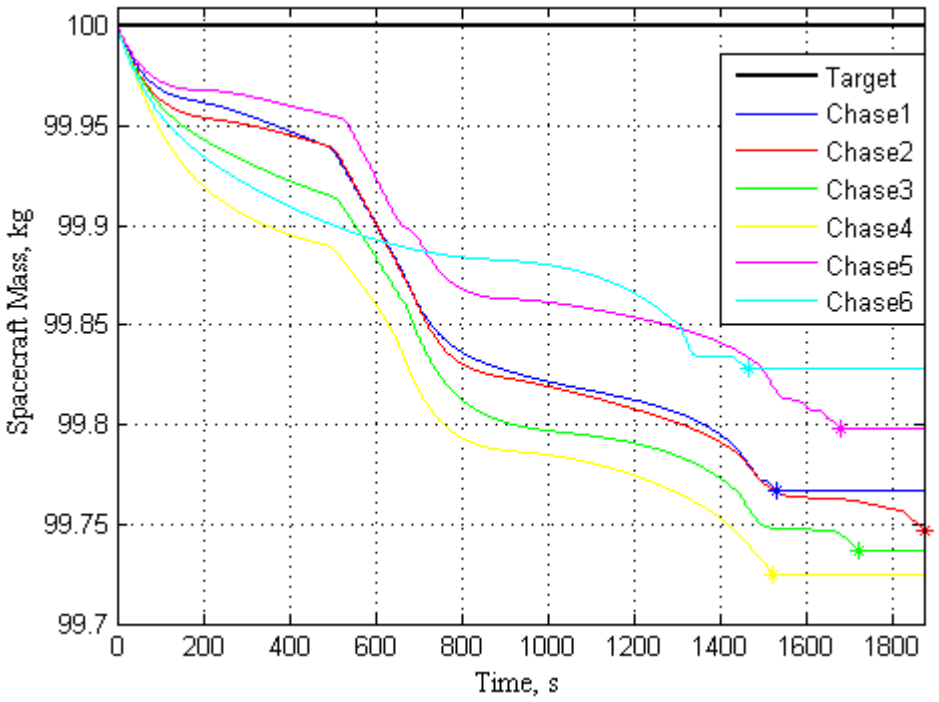

Figure 8.32 Propellant Usage in Relation to Mass.

IX. MONTE-CARLO ANALYSIS OF CLOSE PROXIMITY MULTIPLE SPACECRAFT MANEUVERS

Numerous simulations of multiple spacecraft close proximity maneuvers were conducted in order to generate a sample distribution of maneuver parameters. Using *Monte-Carlo methods*, estimates of the mean and standard deviation of maneuver duration, t_d, and delta-v, Δv, were determined. Two hundred convergence, rally, rendezvous, and docking maneuver simulations were conducted for both the APF and LQR/APF control algorithms. Each simulation involved six Chase spacecraft performing simultaneous maneuvers. The Law of large numbers permits the approximation of sample statistics via Monte-Carlo methods. As expected, the large sample size generally approaches a Gaussian distribution. The normalized data distribution allows for estimates of the maneuver parameter means and standard deviations. The statistical data is presented in both *per* spacecraft and *per* maneuver format. The *per* spacecraft statistics use each Chase spacecraft of each maneuver for a total sample size of 1,200 spacecraft. The *per* maneuver statistics use the maximum parameters of each maneuver for a total sample size of 200. The average close proximity maneuver duration is valuable to spacecraft operators. Similarly, the average Δv is valuable to both spacecraft designers and mission planners.

None of the three control algorithm failures conditions, as discussed in Chapter VII.D, were experienced by either the refined APF or the developed LQR/APF control algorithm. First, no spacecraft collisions were detected. Second, all spacecraft maneuvers were successfully performed within 90 minutes. Finally, no spacecraft was required to use all of its propellant during maneuvering. Therefore, both control algorithms proved to be effective in performing close proximity operations. A statistical analysis of both the APF and LQR/APF during the convergence and docking maneuvers follows.

A. INITIAL CONDITIONS AND SPACECRAFT PARAMETERS

For each simulation, the Chase spacecraft initial positions were uniformly randomly distributed while all other simulation parameters were maintained. Each of the six Chase spacecraft was initially positioned within a 1.0 km sphere with respect to a Target spacecraft position. However, the Chase spacecraft initial position was assumed to be at least 10 m from the Target. This stand off range, representing the center-to-center distance between the Target and Chase spacecraft, was used so that spacecraft size variation of a few meters could be readily simulated. The Target spacecraft was assumed to be in a circular LEO of 500 km altitude. The initial relative velocity was assumed to be negligible. This neutral velocity state in the relative frame, suggests an elliptical orbital phase for the Chase spacecraft. For the sake of control evaluation, this neutral situation is reasonable and serves to avoid bias due to favorable velocity conditions. The relatively high velocity management of both the APF and LQR/APF allows for some initial velocity. The initial velocity was used as an experimental control variable, while the initial position is treated as a uniformly randomly distributed independent variable.

The initial range of each Chase spacecraft is within a 10 - 1,000 m sphere of the Target Spacecraft. The initial positions of all Chase spacecraft, with respect to the Target spacecraft, are shown in Figure 9.1. This includes all 1,200 Chase spacecraft of the 200 simulations. The initial position of the first Chase spacecraft of each simulation is shown in Figure 9.2. This subset of 200 spacecraft initial positions shows that the Chase spacecraft are randomly distributed for each simulation. The Chase spacecraft Monte-Carlo simulation initial range statistics are listed in Table 9.1. First, the initial range mean and standard deviation is listed for all 1,200 Chase spacecraft in the 200 simulations. The Chase spacecraft are uniformly randomly distributed, as shown in Figure 9.3. Next, the mean and standard deviation is listed for the maximum initial Chase spacecraft range of each of the 200 simulations. The maximum Chase spacecraft's initial range distribution for each maneuver, as shown in Figure 9.4. This maximum initial range metric drives the overall t_d and Δv for each multiple spacecraft maneuver. Both the LQR/APF and APF control algorithm are analyzed over the same random range distribution. This allows for direct comparison of performance for each maneuver.

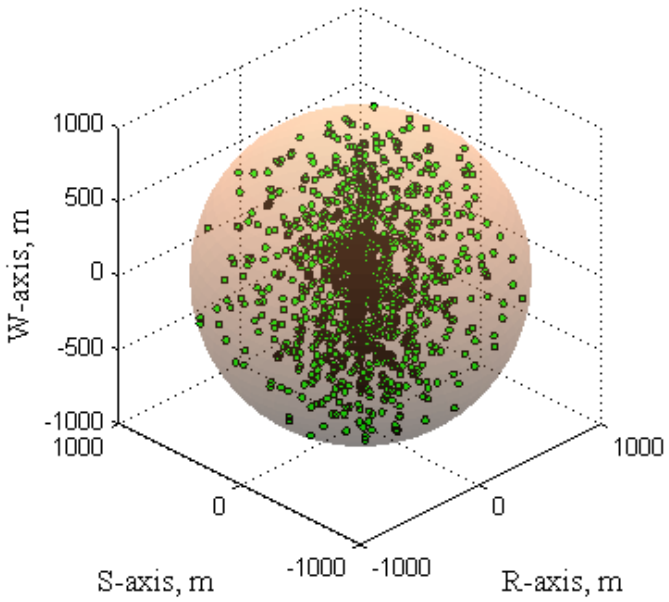

Figure 9.1 Initial Position of All Chase Spacecraft.

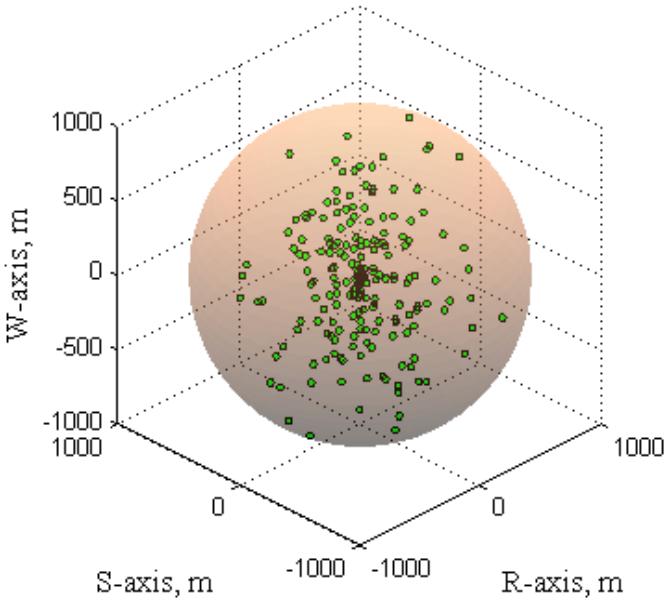

Figure 9.2 Initial Position of First Chase Spacecraft.

Initial Range Statistics	Mean	Standard Deviation
Initial Range (1200 samples)	501.8 m	282.1 m
Max Initial Range (200 samples)	840.0 m	122.0 m

Table 9.1 Chase Spacecraft Range Distribution Statistics.

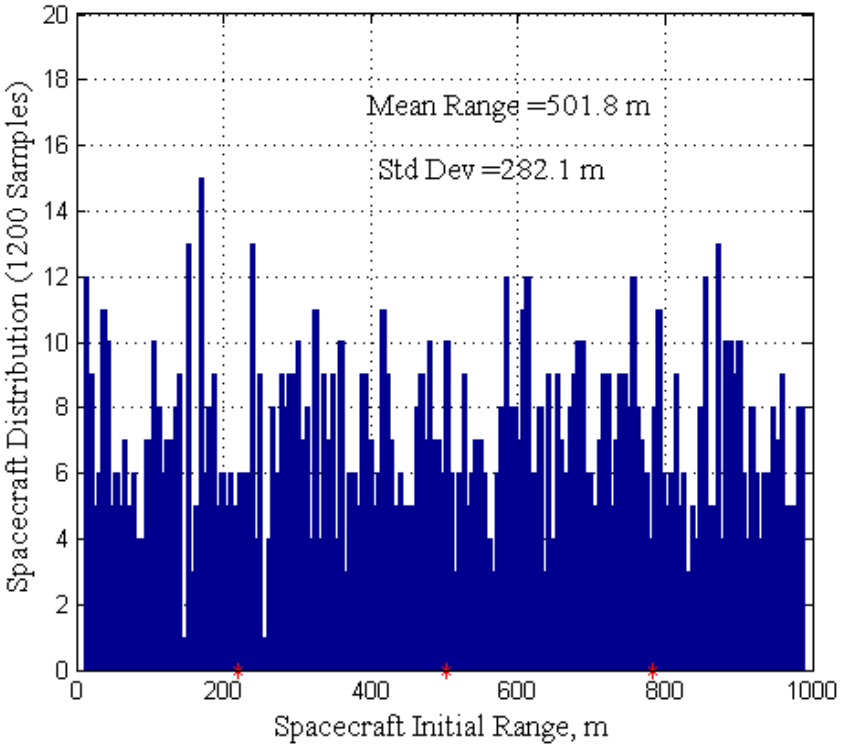

Figure 9.3 Initial Chase Spacecraft Range Distribution.

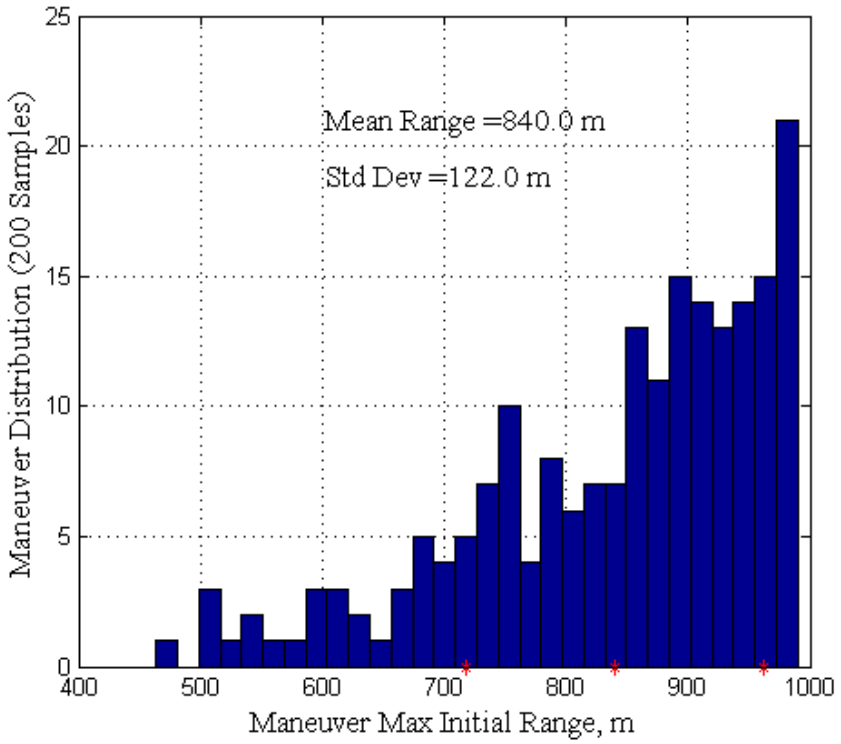

Figure 9.4 Maximum Maneuver Initial Range Distribution.

The Chase spacecraft parameters were held constant for all simulations. All spacecraft are homogeneous cubes of 1.0 m. Each spacecraft has an initial mass of 100 kg; however the mass of each varies as it uses propellant. The general Monte-Carlo simulation method parameters are listed in Table 9.2. Each high fidelity spacecraft model is subject to all of the perturbing forces and torques discussed in Chapter III.

Monte Carlo Analysis Simulation Parameters	
Number of Simulations	200
Spacecraft Shape	Cubic
Spacecraft Size	1.0 m
Initial Spacecraft Mass	100 kg
Initial Propellant Mass	3 kg
Target Spacecraft Orbit	Circular orbit at 500 km altitude
Number of Chase Spacecraft per Maneuver	6
Chase Spacecraft Initial Position	10 – 1,000 m from Target
Chase Spacecraft Initial Velocity	Negligible relative velocity
Spacecraft Perturbation Forces	High Fidelity Model

Table 9.2 Monte Carlo Simulation Parameters.

B. MONTE-CARLO ANALYSIS OF CONVERGENCE MANEUVERS

The simple convergence maneuver, once again, serves as the baseline for evaluating control algorithm performance. The individual spacecraft distribution statistics for t_d and Δv are listed in Table 9.3. Overall, the LQR/APF performs simple convergence maneuvers better, on a *per* spacecraft basis. The maneuvers are accomplished in less time using less control effort. This is as expected due to the iterative optimization of the LQR algorithm. The LQR/APF's and APF's convergence t_d spacecraft distributions are shown in Figure 9.5 and Figure 9.6. The LQR/APF's and APF's convergence Δv spacecraft distributions are shown in Figure 9.7 and Figure 9.8.

Convergence Spacecraft Statistics (1,200 Samples)		LQR/APF	APF
t_d	Mean	1212.9 s	1364.0 s
	Standard Deviation	102.6 s	56.5 s
Δv	Mean	1.252 m/s	1.276 m/s *
	Standard Deviation	0.708 m/s	0.732 m/s

Table 9.3 Convergence Spacecraft Statistics.

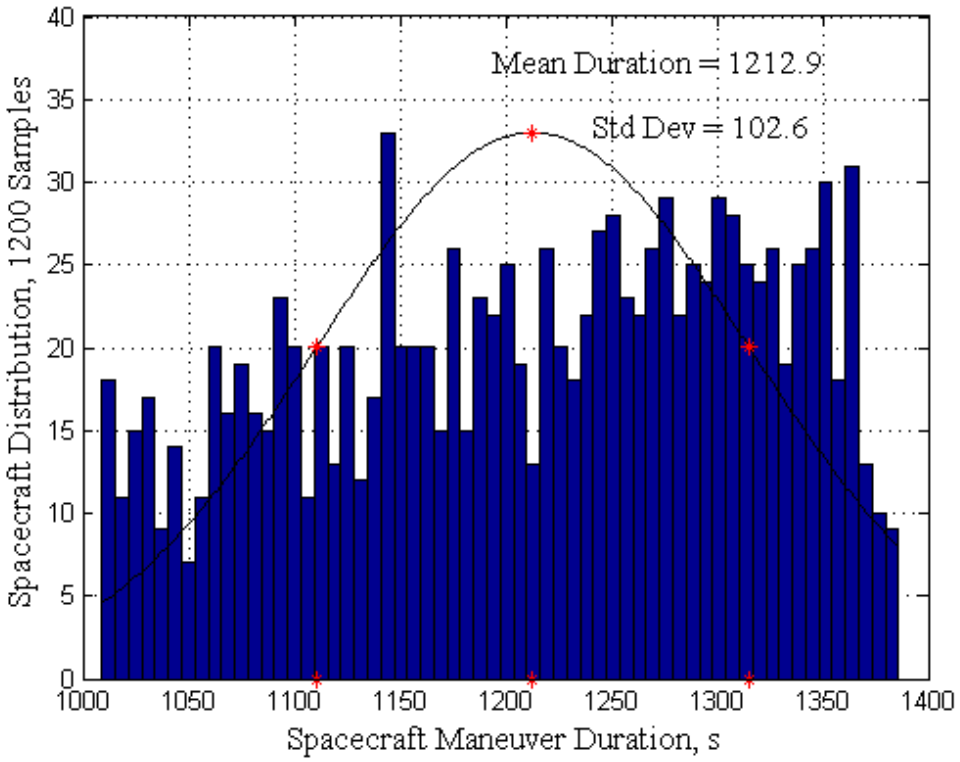

Figure 9.5 LQR/APF Spacecraft Convergence Maneuver Duration Distribution.

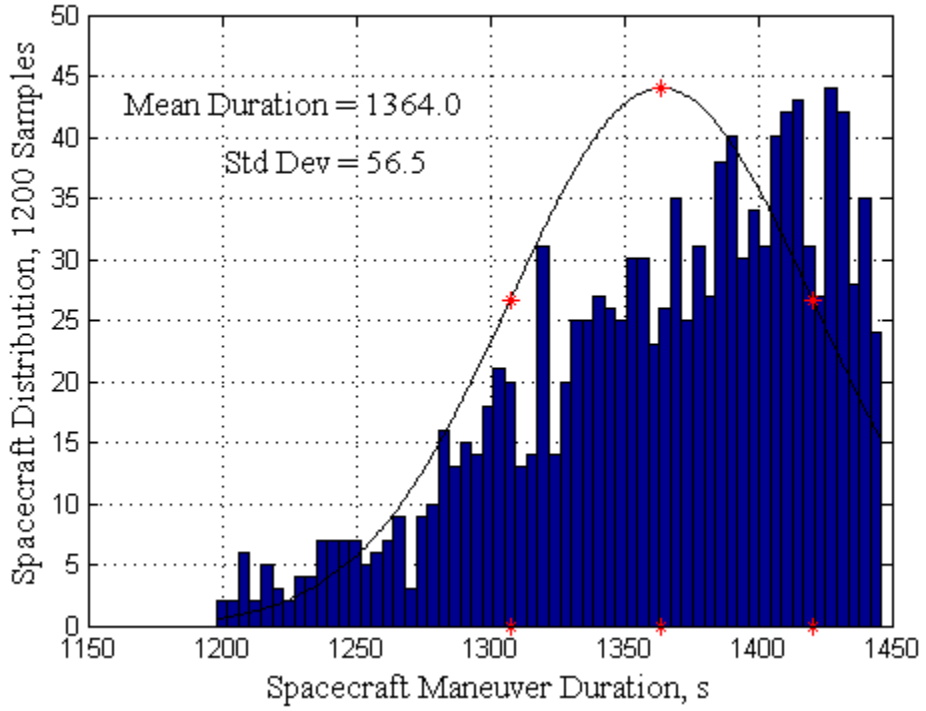

Figure 9.6 APF Spacecraft Convergence Maneuver Duration Distribution.

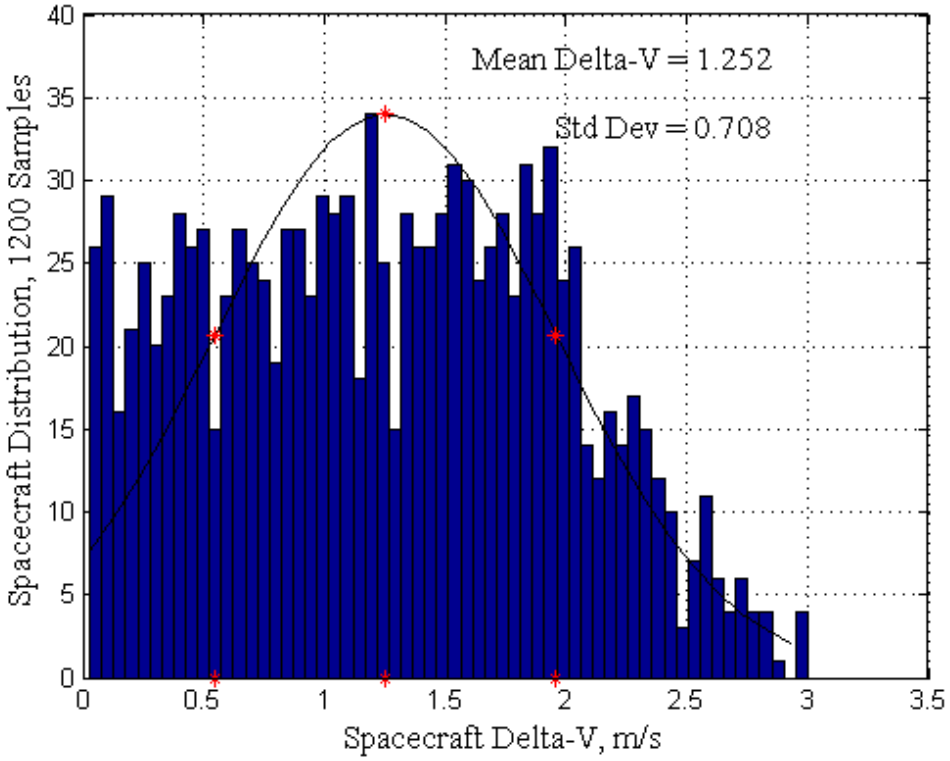

Figure 9.7 LQR/APF Spacecraft Convergence Delta-V Distribution.

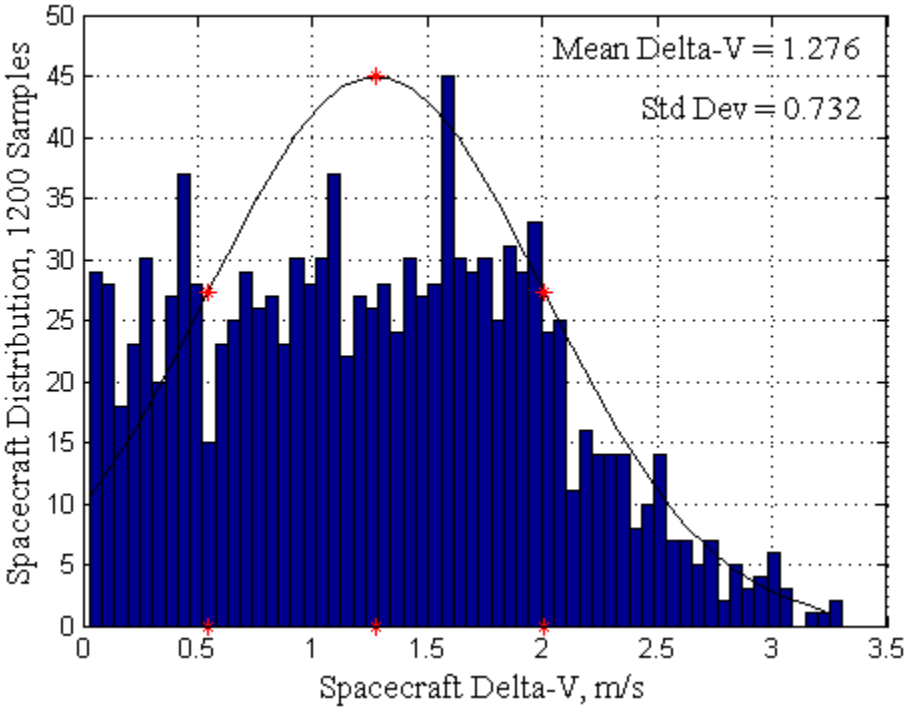

Figure 9.8 APF Spacecraft Convergence Delta-V Distribution.

The statistical analysis is extended to a *per* maneuver basis. The maximum t_d and Δv are selected for each six Chase spacecraft of a convergence maneuver. Additionally, the Δv of all six maneuvering Chase spacecraft are summed to obtain the total Δv for each maneuver. The mean and standard deviation of these metrics are listed in Table 9.4. The LQR/APF control algorithm performs simple convergence maneuvers well. The maneuver durations are shorter on average. Both the maximum and total maneuver Δv have a smaller average and tighter standard deviation. The tighter standard deviation is helpful for spacecraft designers and mission planners. The LQR/APF's and APF's convergence maneuver maximum t_d distributions are shown in Figure 9.9 and Figure 9.10. The maximum duration for each convergence maneuver is similar to the maximum initial range distribution. The LQR/APF's and APF's convergence maneuver maximum Δv distributions are shown in Figure 9.11 and Figure 9.12. The LQR/APF's and APF's total convergence maneuver Δv distributions are shown in Figure 9.13 and Figure 9.14.

Convergence Maneuver Statistics (200 Samples)		LQR/APF	APF
Max t_d	Mean	1330.7 s	1423.4 s
	Standard Deviation	38.7 s	17.5 s
Max Δv	Mean	2.147 m/s	2.219 m/s
	Standard Deviation	0.404 m/s	0.459 m/s
Total Δv	Mean	7.515 m/s	7.656 m/s
	Standard Deviation	1.899 m/s	1.951 m/s

Table 9.4 Convergence Maneuver Statistics.

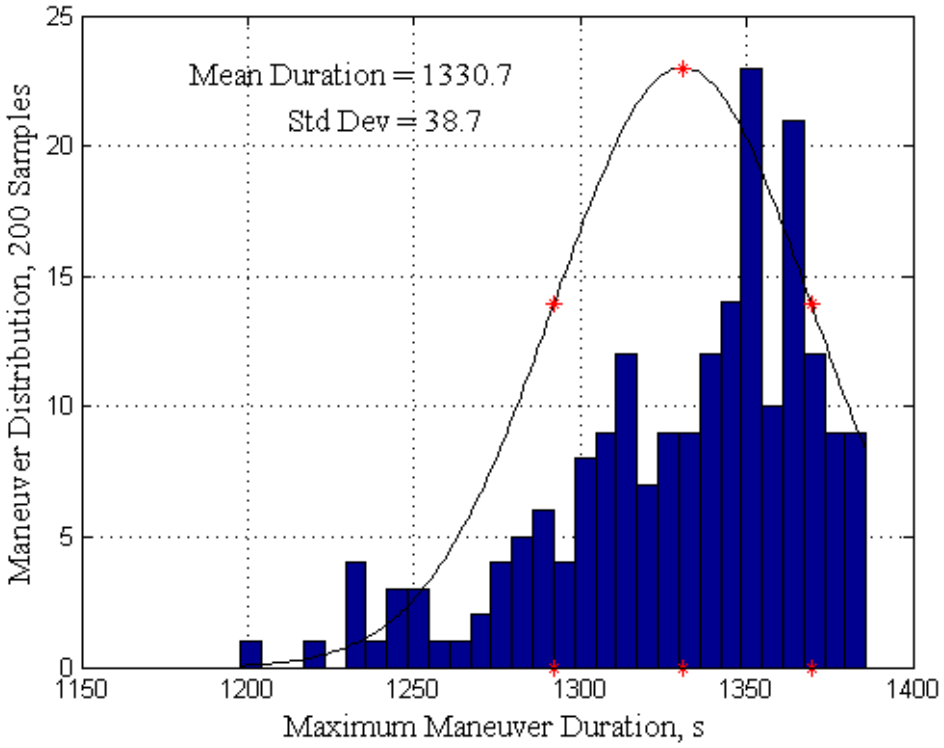

Figure 9.9 LQR/APF Convergence Maneuver Maximum Duration Distribution.

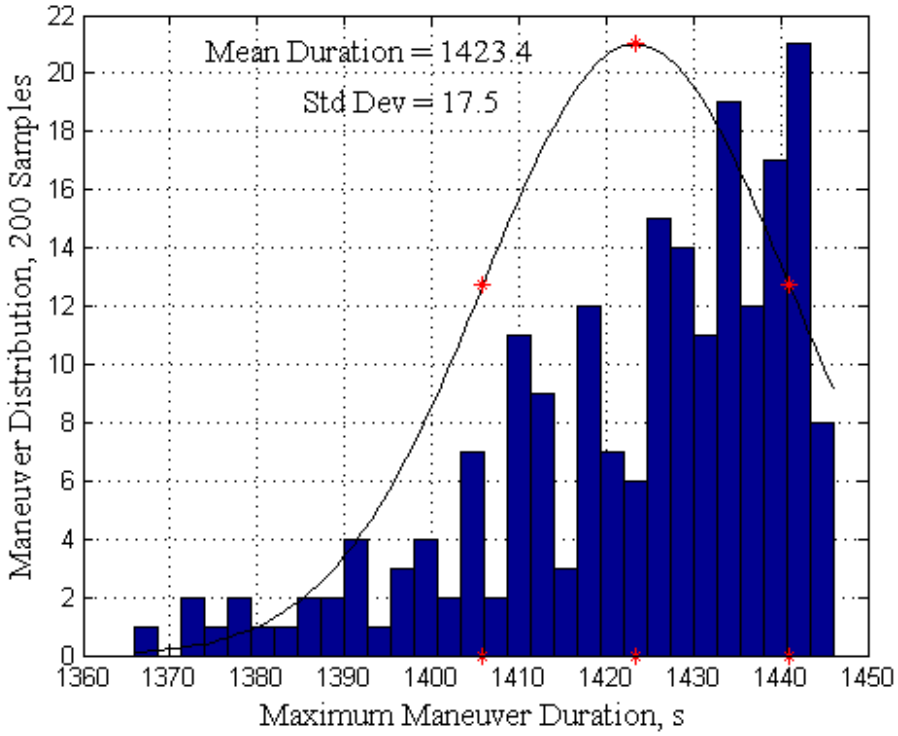

Figure 9.10 APF Convergence Maneuver Maximum Duration Distribution.

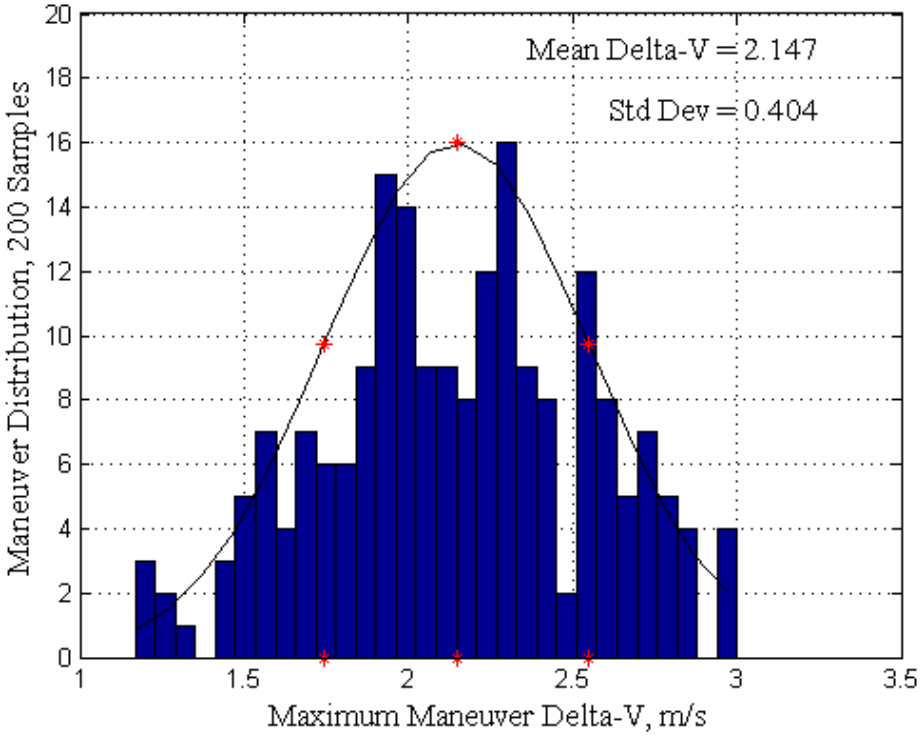

Figure 9.11 LQR/APF Convergence Maneuver Maximum Delta-V Distribution.

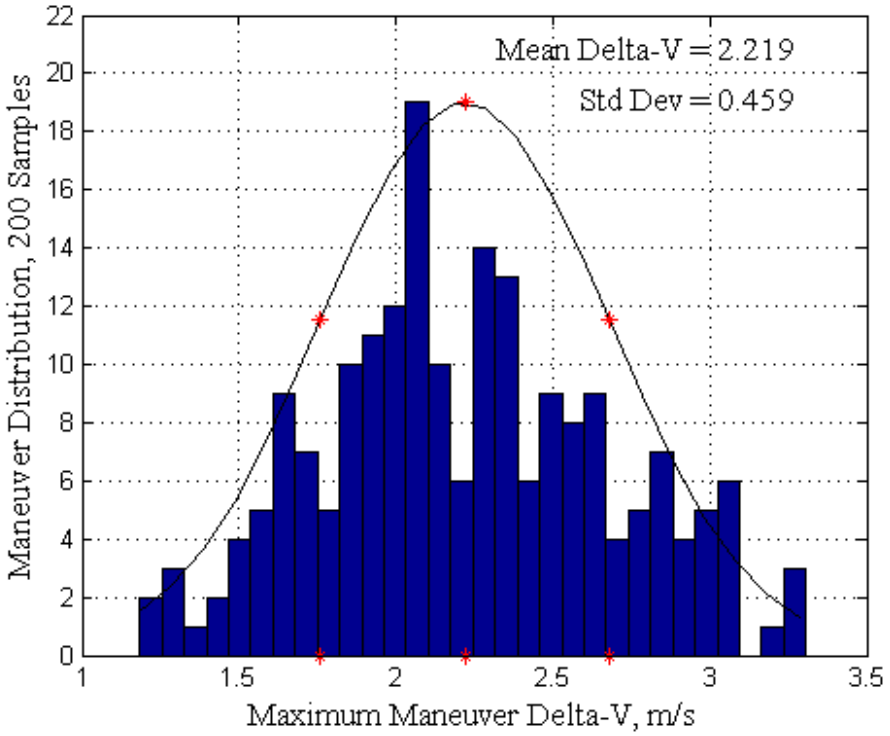

Figure 9.12 APF Convergence Maneuver Maximum Delta-V Distribution.

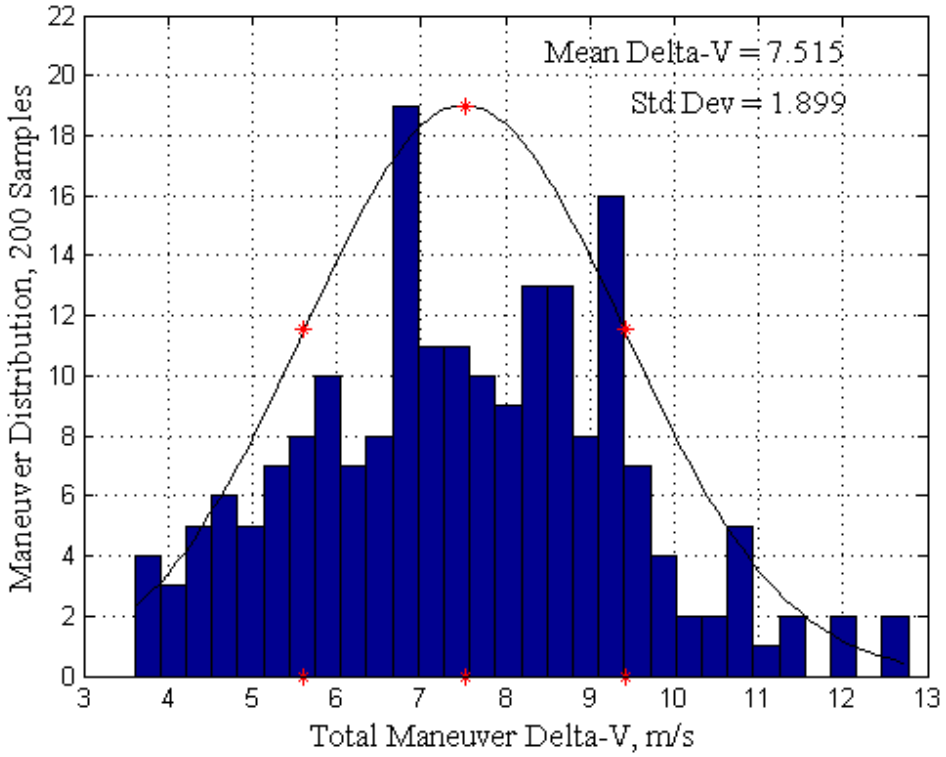

Figure 9.13 LQR/APF Convergence Maneuver Total Delta-V Distribution.

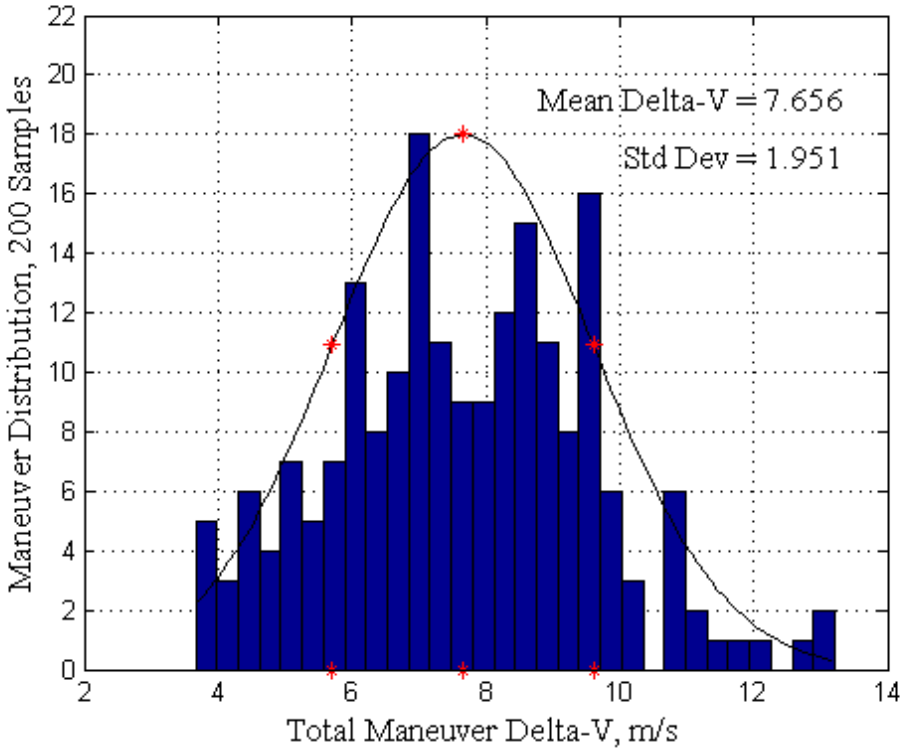

Figure 9.14 APF Convergence Maneuver Total Delta-V Distribution.

During convergence maneuvers, the APF control algorithm significantly saturated the control actuation in approximately 3-6% of all convergence maneuvers. These would represent situations where no further control effort is available for collision avoidance. This saturation allows the APF control effort to appear comparatively efficient, but is not a desired state for high density obstacle environments.

C. MONTE-CARLO ANALYSIS OF RALLY MANEUVERS

For the Monte-Carlo six spacecraft rally maneuvers the rally sphere is 3.5 meters. Each cubic spacecraft must simultaneously converge to within 3.5 meters of the goal position while avoiding each other. This spherical range represents the maximum cross-section of two cubic spacecraft. The individual spacecraft distribution statistics for t_d and Δv are listed in Table 9.5. Overall, the LQR/APF performs six spacecraft rally maneuvers better, on a per spacecraft basis. The maneuvers are accomplished in slightly more time using less control effort. This is as expected due to the strict velocity control

of the APF control algorithm. The LQR/APF's and APF's rally t_d spacecraft distributions are shown in Figure 9.15 and Figure 9.16. The LQR/APF has a couple duration outliers due to Chase spacecraft loitering in front of later arriving Chase spacecraft. The LQR/APF's and APF's rally Δv spacecraft distributions are shown in Figure 9.17 and Figure 9.18.

Rally Spacecraft Statistics (1,200 Samples)		**LQR/APF**	**APF**
t_d	Mean	1170.5 s	1065.2 s
	Standard Deviation	179.1 s	90.2 s
Δv	Mean	1.151 m/s	1.167 m/s
	Standard Deviation	0.706 m/s	0.720 m/s

Table 9.5 Rally Spacecraft Statistics.

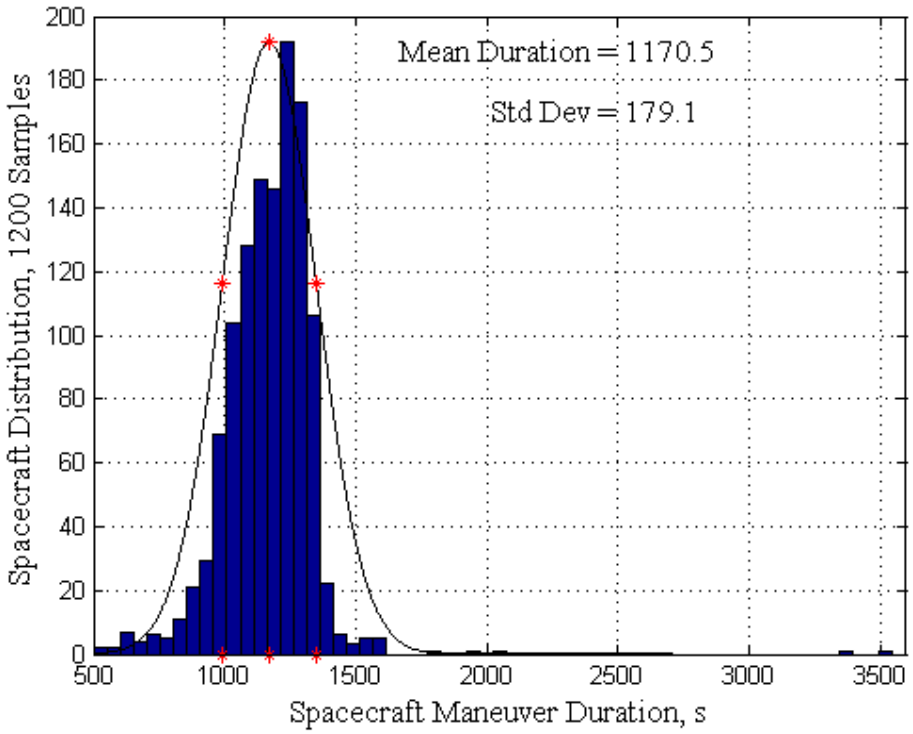

Figure 9.15 LQR/APF Spacecraft Rally Maneuver Duration Distribution.

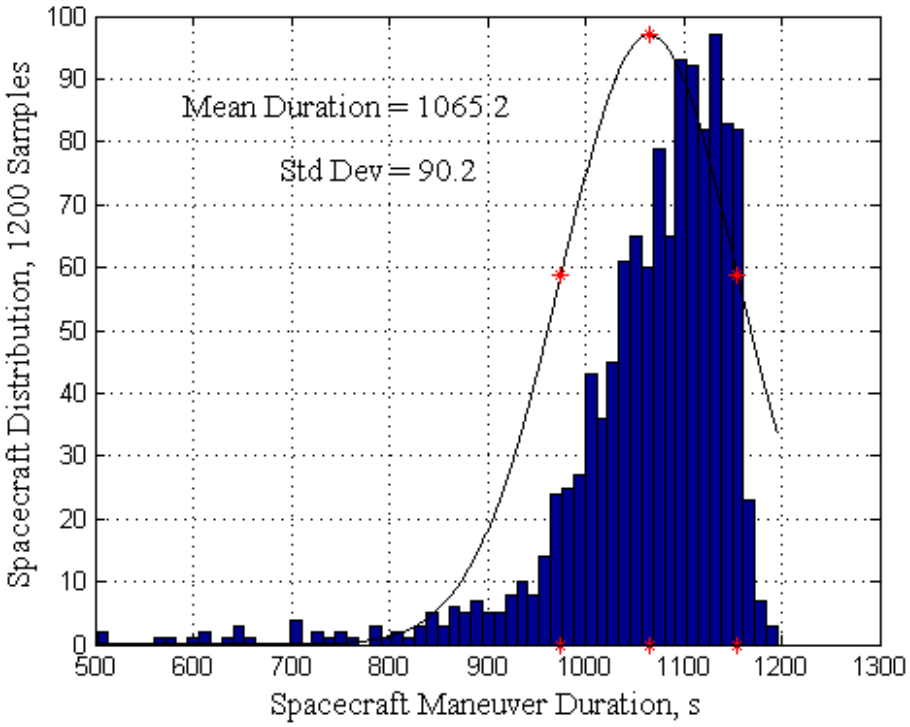

Figure 9.16 APF Spacecraft Rally Maneuver Duration Distribution.

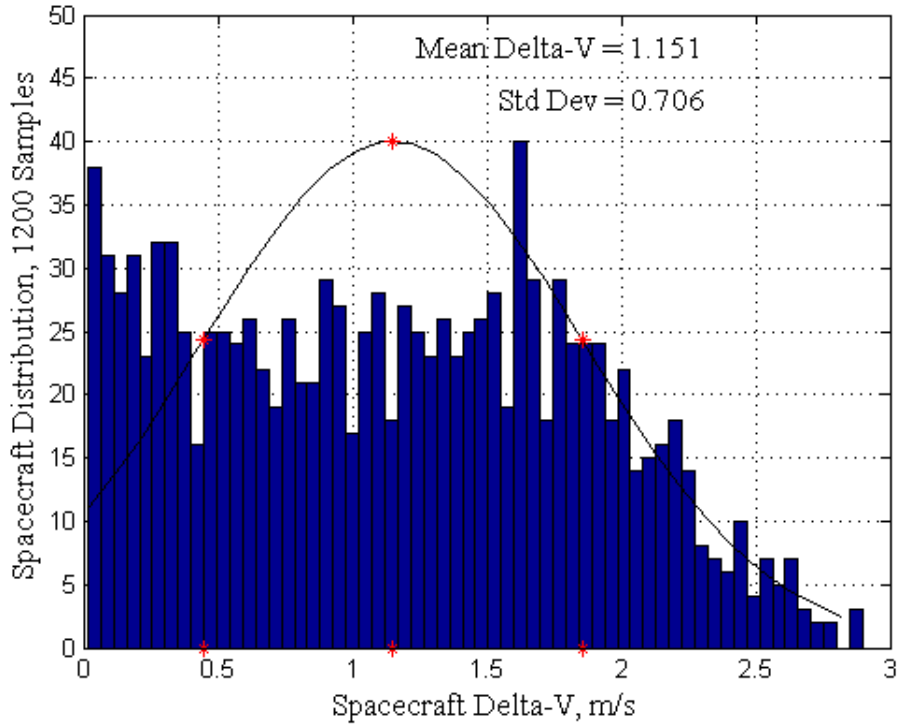

Figure 9.17 LQR/APF Spacecraft Rally Delta-V Distribution.

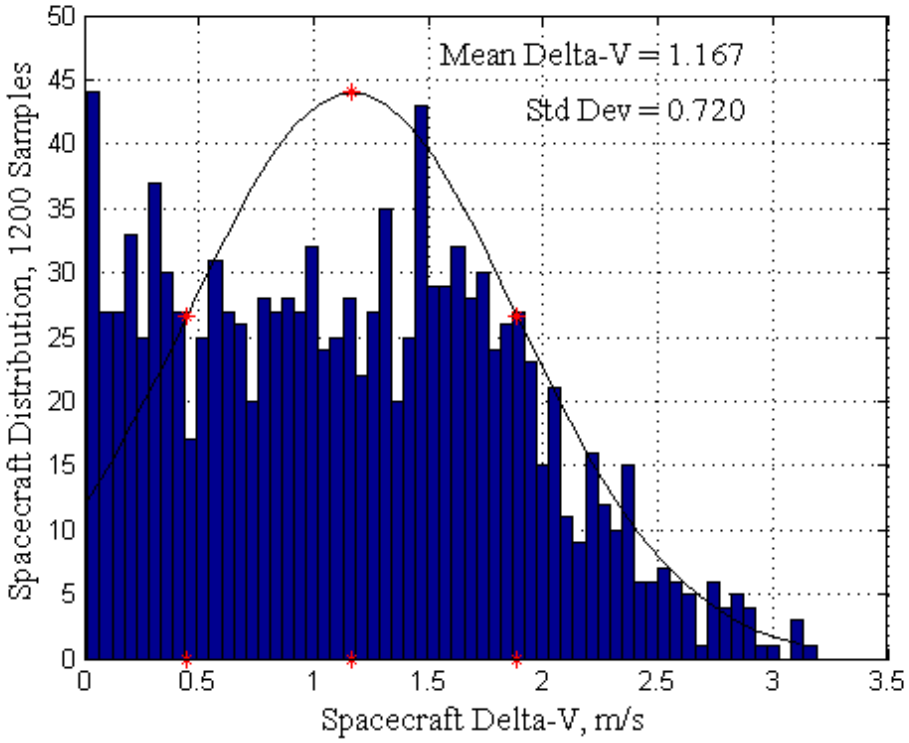

Figure 9.18 APF Spacecraft Rally Delta-V Distribution.

The statistical analysis is extended to a *per* rally maneuver basis, with the mean and standard deviation of t_d and Δv metrics listed in Table 9.6. The LQR/APF control algorithm performs rally maneuvers well. The average maneuver duration is slightly longer, but both the maximum and total maneuver Δv has smaller average and tighter standard deviation. The APF control algorithm significantly saturated the control actuation in approximately 5-10% of all rally maneuvers. The LQR/APF's and APF's rally maneuver maximum t_d distributions are shown in Figure 9.19 and Figure 9.20. The maximum duration for each rally maneuver is similar to the maximum initial range distribution. The LQR/APF's and APF's rally maneuver maximum Δv distributions are shown in Figure 9.21 and Figure 9.22. The LQR/APF's and APF's total rally maneuver Δv distributions are shown in Figure 9.23 and Figure 9.24.

Rally Maneuver Statistics (200 Samples)		LQR/APF	APF
Max t_d	Mean	1341.9 s	1143.4 s
	Standard Deviation	184.1 s	21.2 s
Max Δv	Mean	2.055 m/s	2.109 m/s
	Standard Deviation	0.389 m/s	0.454 m/s
Total Δv	Mean	6.907 m/s	7.003 m/s
	Standard Deviation	1.808 m/s	1.881 m/s

Table 9.6 Rally Maneuver Statistics.

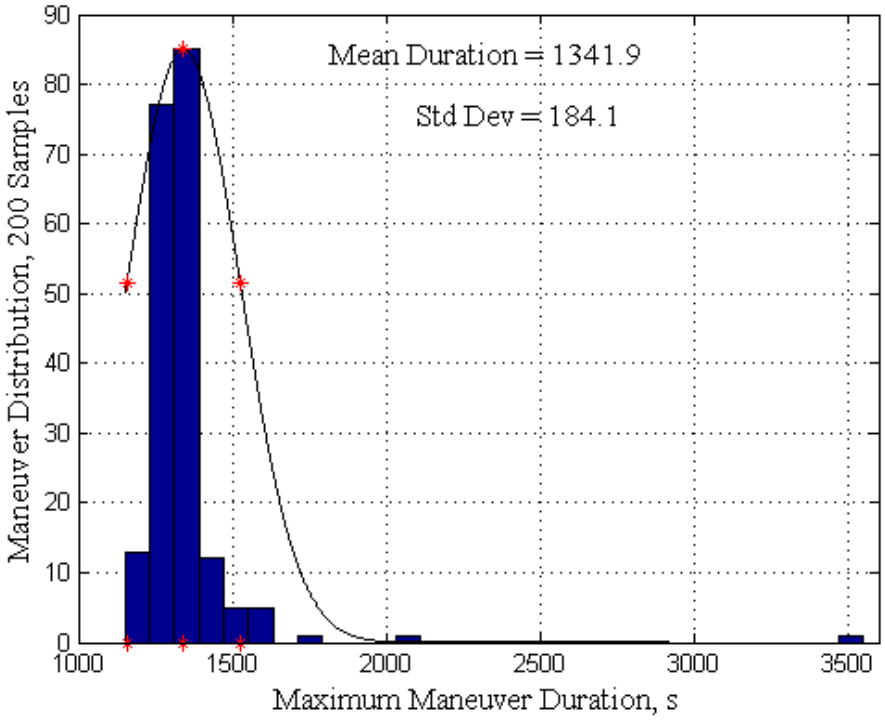

Figure 9.19 LQR/APF Rally Maneuver Maximum Duration Distribution.

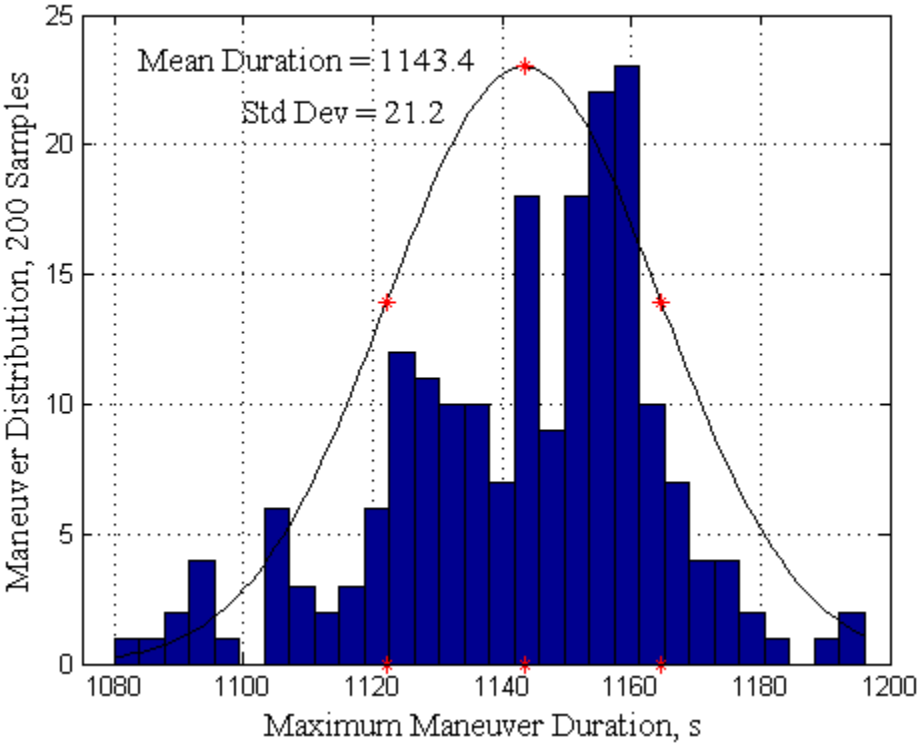

Figure 9.20 APF Rally Maneuver Maximum Duration Distribution.

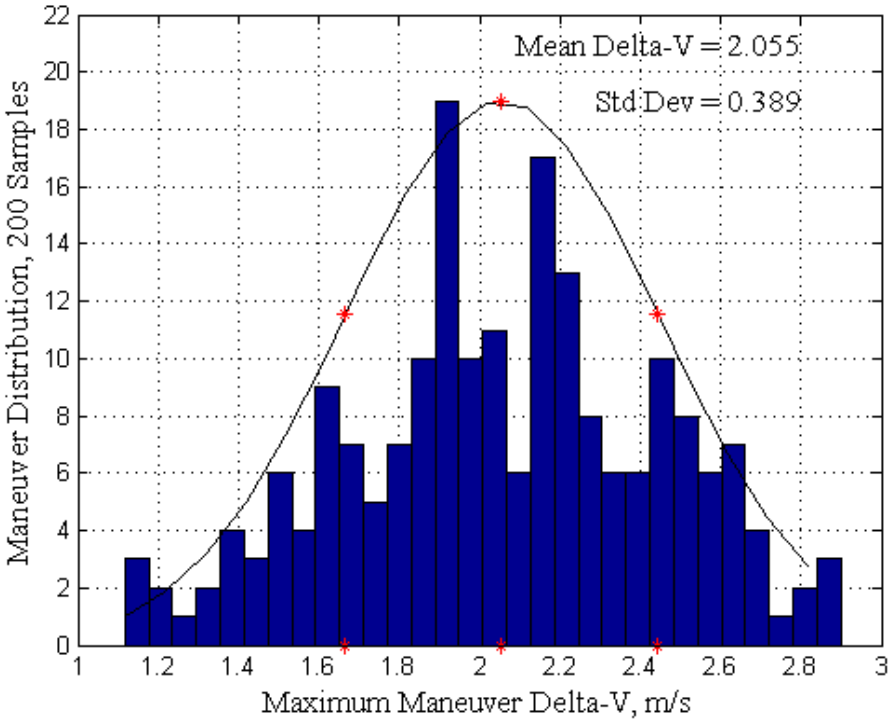

Figure 9.21 LQR/APF Rally Maneuver Maximum Delta-V Distribution.

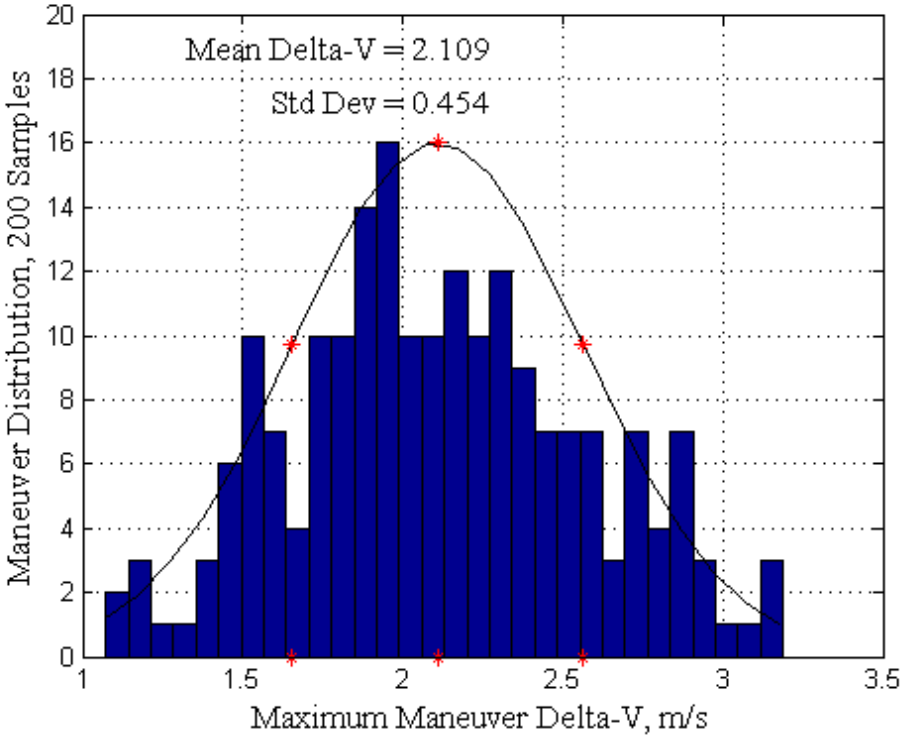

Figure 9.22 APF Rally Maneuver Maximum Delta-V Distribution.

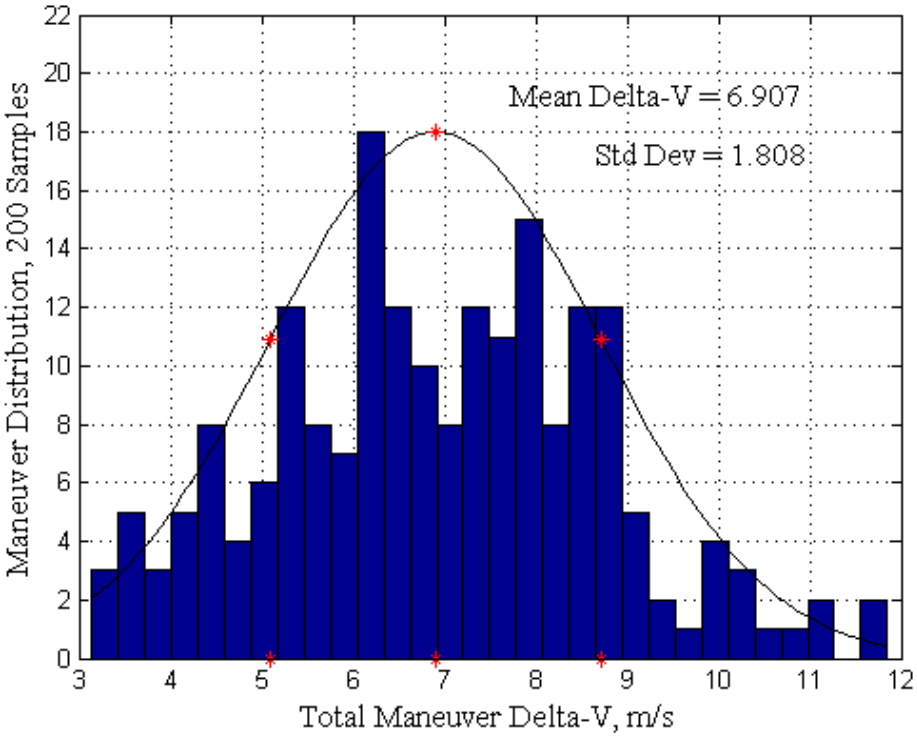

Figure 9.23 LQR/APF Rally Maneuver Total Delta-V Distribution.

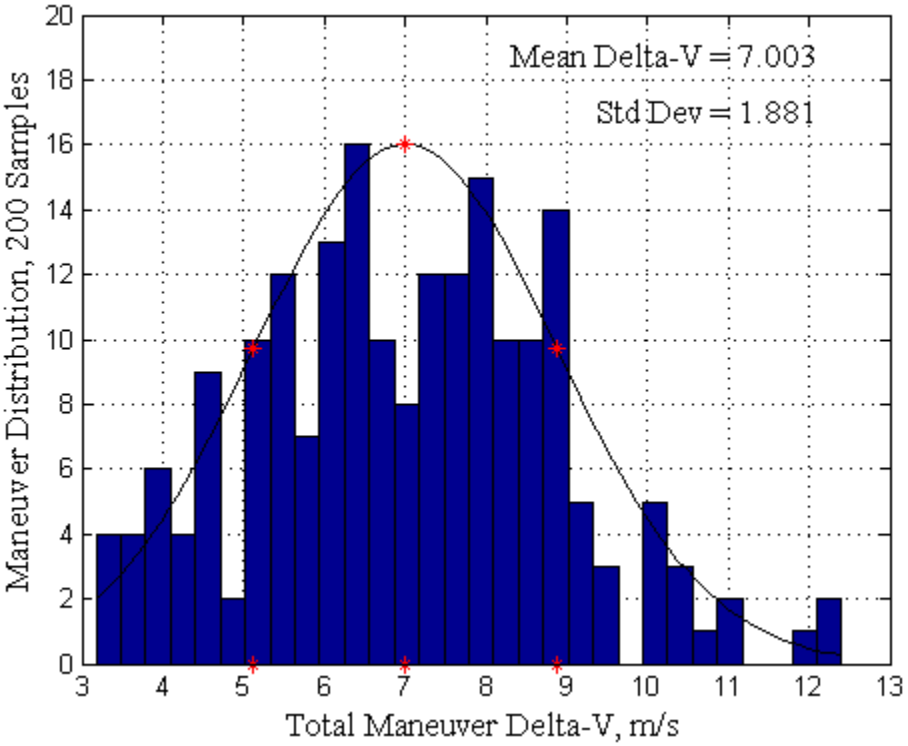

Figure 9.24 APF Rally Maneuver Total Delta-V Distribution.

D. MONTE-CARLO ANALYSIS OF RENDEZVOUS MANEUVERS

The six spacecraft rendezvous maneuver sphere is similar to that of the rally maneuver. Each spacecraft must simultaneously converge to within 3 meters of the surface of the Target spacecraft while avoiding each other. This allows rendezvous within twice the maximum cross-section of the cubic spacecraft. Any closer rendezvous requires thorough operational consideration of practical maneuver space and end goal points. The individual spacecraft distribution statistics for t_d and Δv are listed in Table 9.7. Similar to the rally maneuver, the LQR/APF performs six spacecraft rendezvous maneuvers in slightly more time using less control effort. The LQR/APF's and APF's rendezvous t_d spacecraft distributions are shown in Figure 9.25 and Figure 9.26. As seen in the rally maneuver, the LQR/APF has a couple duration outliers. The LQR/APF's and APF's rendezvous Δv spacecraft distributions are shown in Figure 9.27 and Figure 9.28.

Rendezvous Spacecraft Statistics (1,200 Samples)		LQR/APF	APF
t_d	Mean	1171.9 s	1065.0 s
	Standard Deviation	181.3 s	90.1 s
Δv	Mean	1.150 m/s	1.166 m/s
	Standard Deviation	0.704 m/s	0.719 m/s

Table 9.7 Rendezvous Spacecraft Statistics.

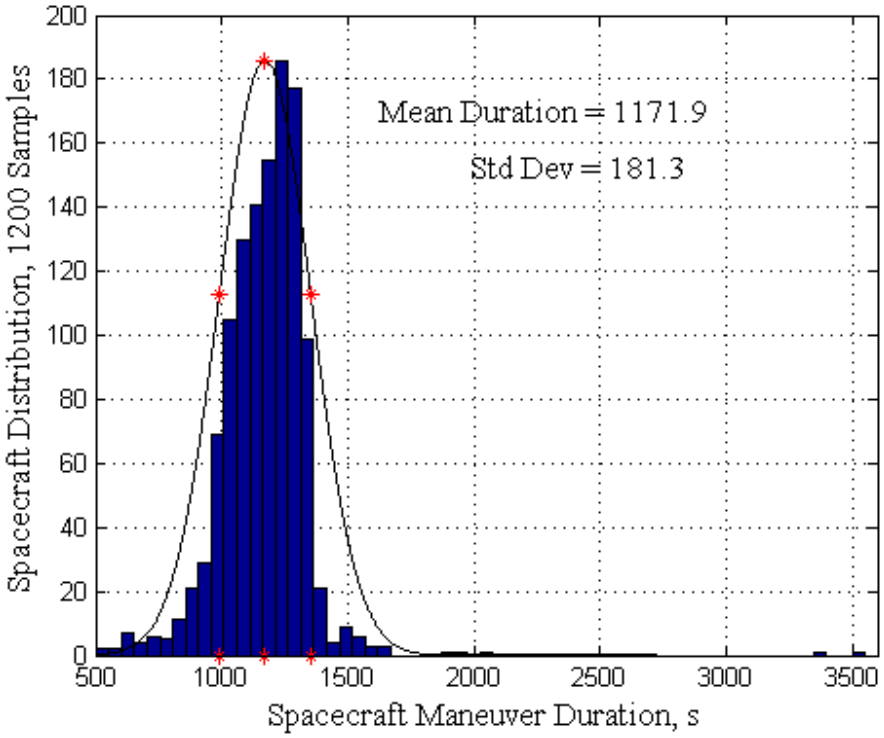

Figure 9.25 LQR/APF Spacecraft Rendezvous Maneuver Duration Distribution.

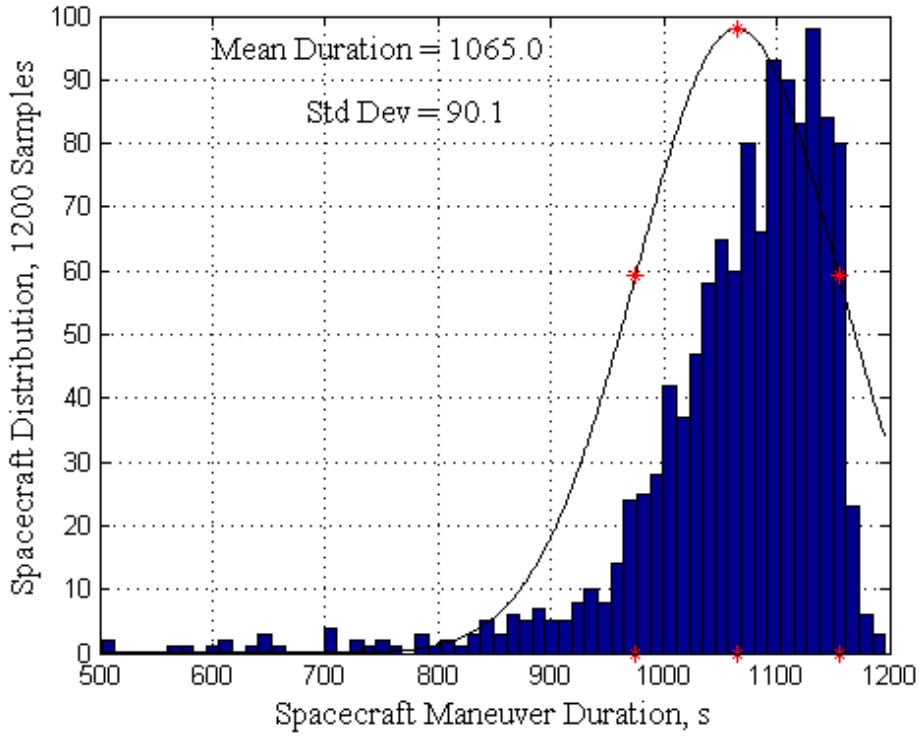

Figure 9.26 APF Spacecraft Rendezvous Maneuver Duration Distribution.

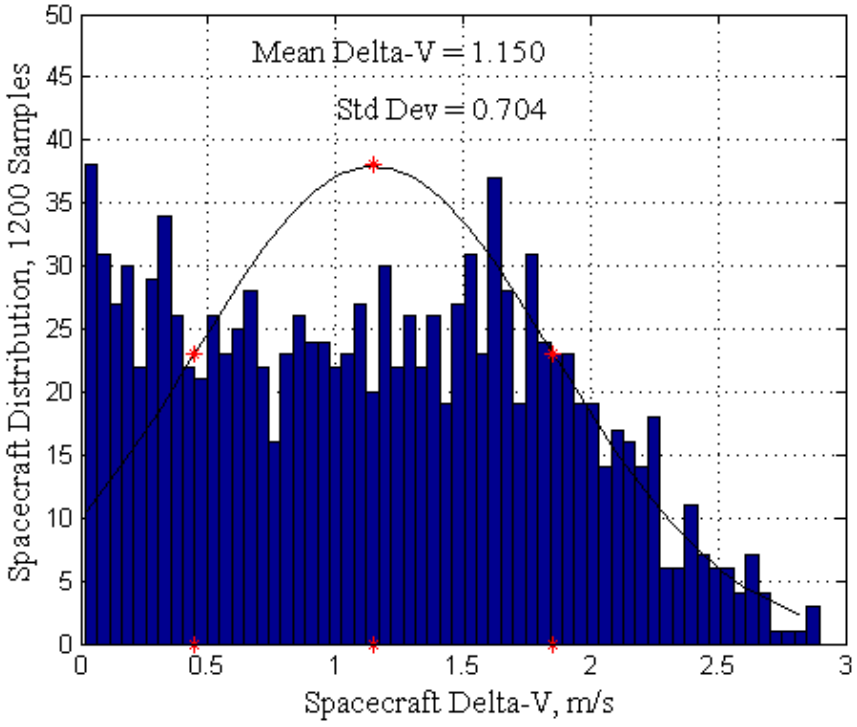

Figure 9.27 LQR/APF Spacecraft Rendezvous Delta-V Distribution.

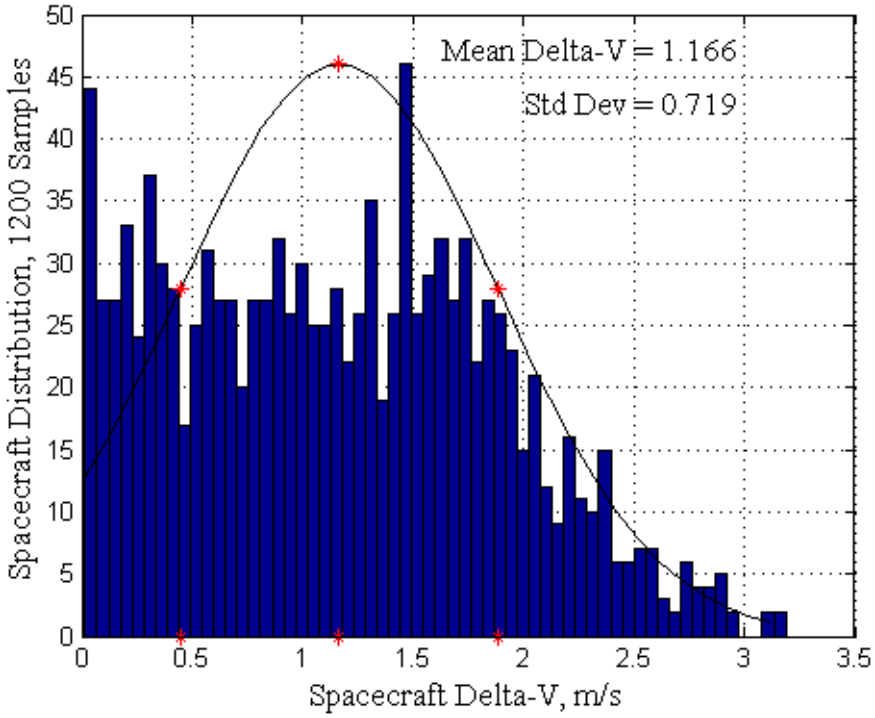

Figure 9.28 APF Spacecraft Rendezvous Delta-V Distribution.

The statistical analysis is extended to a *per* rendezvous maneuver basis, with the means and standard deviations of t_d and Δv listed in Table 9.8. The LQR/APF control algorithm performance is similar to that of the rally maneuver, with slightly longer duration and smaller average Δv and tighter Δv standard deviation. Again, the APF control algorithm significantly saturated the control actuation in approximately 5-10% of all rendezvous maneuvers. The LQR/APF's and APF's rendezvous maneuver maximum t_d distributions are shown in Figure 9.29 and Figure 9.30. The maximum duration for each rendezvous maneuver is similar to the maximum initial range distribution. The LQR/APF's and APF's rendezvous maneuver maximum Δv distributions are shown in Figure 9.31 and Figure 9.32. The LQR/APF's and APF's total rendezvous maneuver Δv distributions are shown in Figure 9.33 and Figure 9.34.

Rendezvous Maneuver Statistics (200 Samples)		LQR/APF	APF
Max t_d	Mean	1347.8 s	1143.2 s
	Standard Deviation	188.0 s	21.3 s
Max Δv	Mean	2.054 m/s	2.107 m/s
	Standard Deviation	0.386 m/s	0.454 m/s
Total Δv	Mean	6.902 m/s	6.994 m/s
	Standard Deviation	1.793 m/s	1.878 m/s

Table 9.8 Rendezvous Maneuver Statistics.

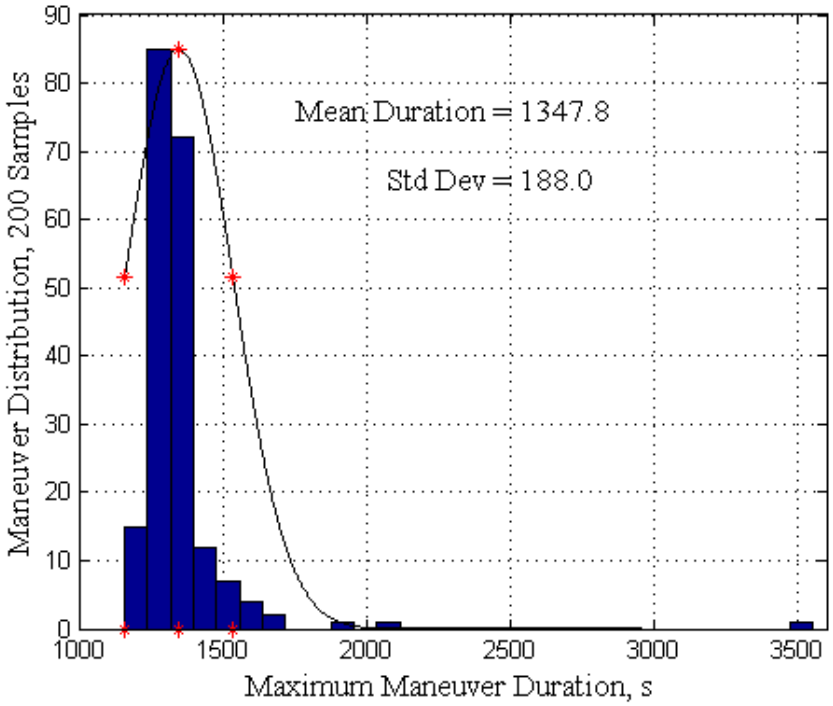

Figure 9.29 LQR/APF Rendezvous Maneuver Maximum Duration Distribution.

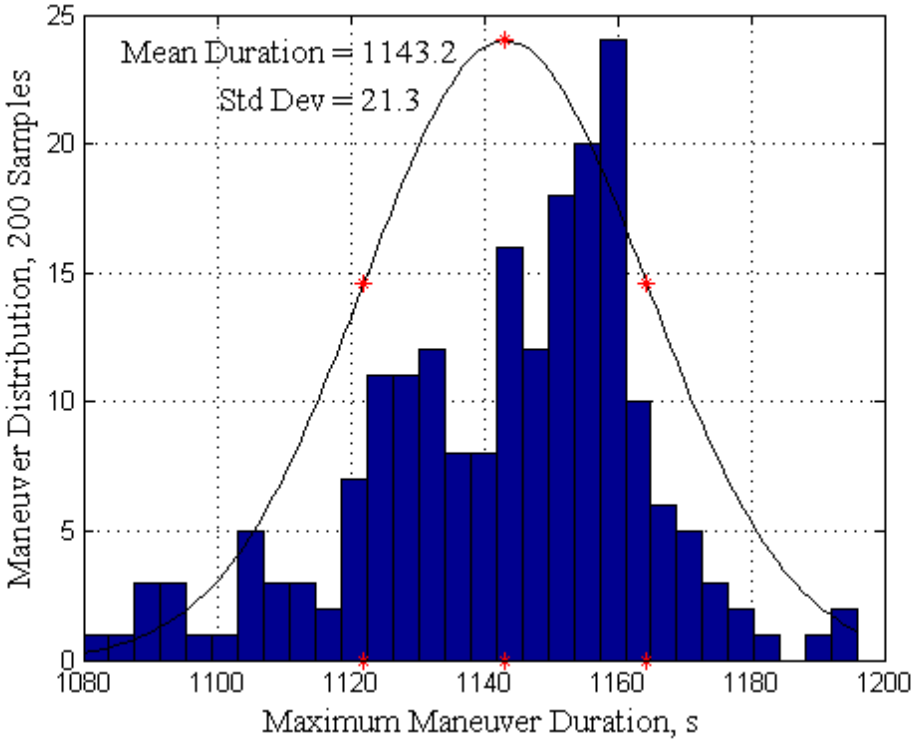

Figure 9.30 APF Rendezvous Maneuver Maximum Duration Distribution.

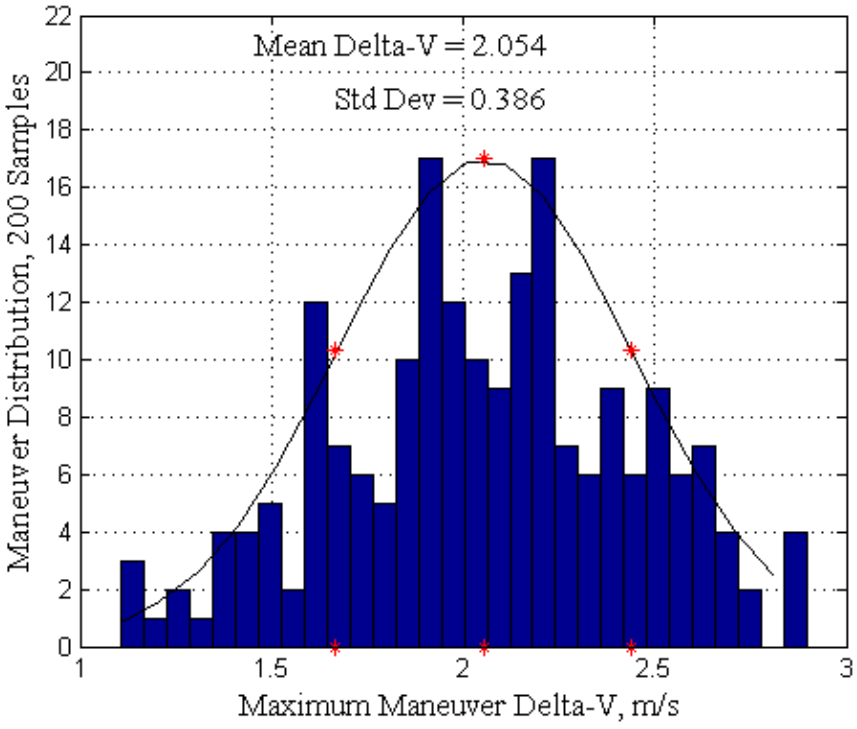

Figure 9.31 LQR/APF Rendezvous Maneuver Maximum Delta-V Distribution.

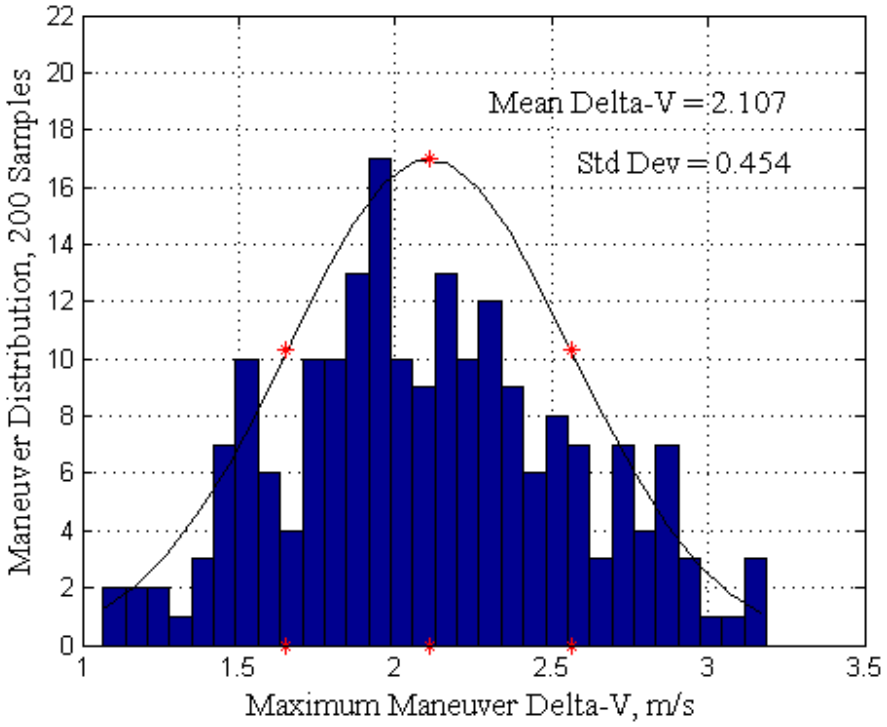

Figure 9.32 APF Rendezvous Maneuver Maximum Delta-V Distribution.

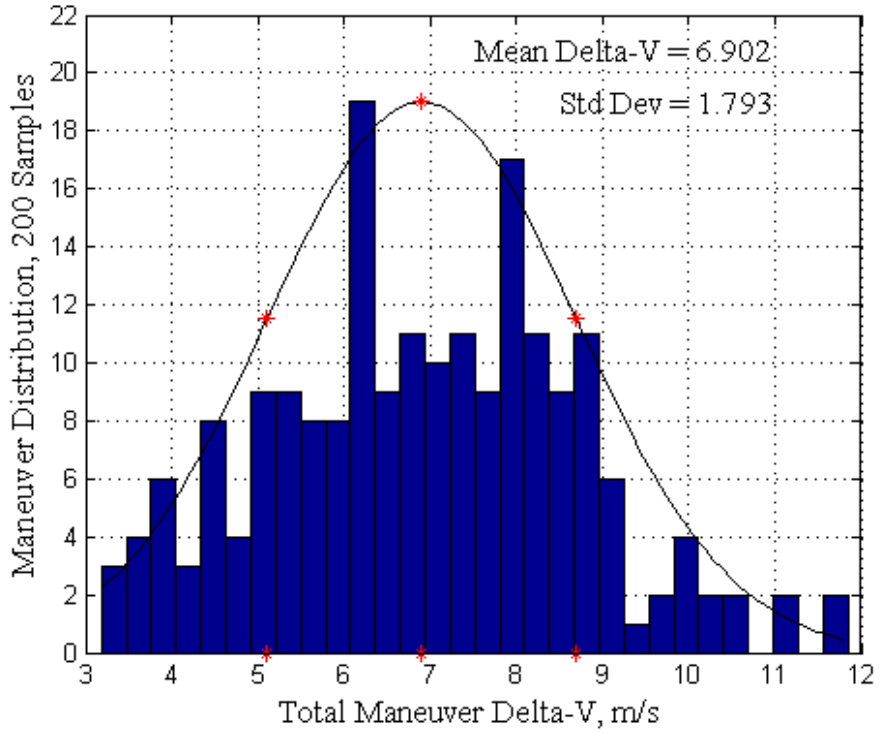

Figure 9.33 LQR/APF Rendezvous Maneuver Total Delta-V Distribution.

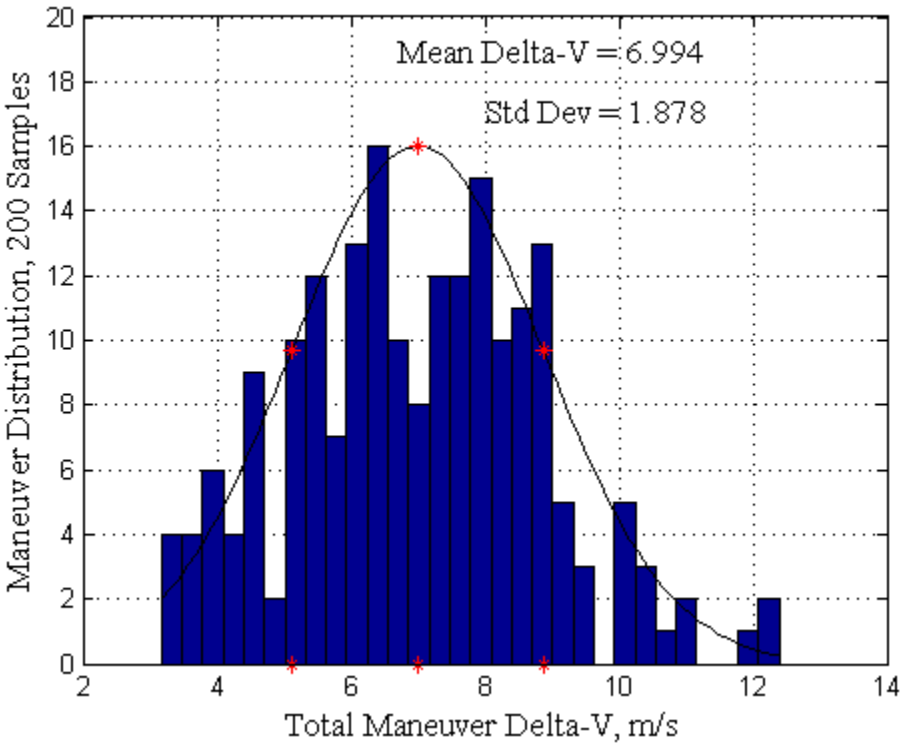

Figure 9.34 APF Rendezvous Maneuver Total Delta-V Distribution.

E. MONTE-CARLO ANALYSIS OF DOCKING MANEUVERS

The six spacecraft simultaneous docking maneuver requires that all six spacecraft avoid each other while converging to within 2.0 millimeter of their assigned docking position on the Target spacecraft's outer boundary. The docking port positions, on the center of each cubic face of the Target spacecraft, are randomly assigned. The individual spacecraft distribution statistics for t_d and Δv are listed in Table 9.9. The LQR/APF still performs well, however the collision avoidance maneuvering close to the Target favors the APF controller. The maneuvers are accomplished in slightly more time and more control effort. This is as expected due to the intrinsic collision avoidance capability of the APF control algorithm. Although, the APF control efficiency is only achieved by saturating the available control actuation. The LQR/APF's and APF's docking t_d spacecraft distributions are shown in Figure 9.35 and Figure 9.36. The LQR/APF's and APF's docking Δv spacecraft distributions are shown in Figure 9.37 and Figure 9.38.

Docking Spacecraft Statistics (1,200 Samples)		LQR/APF	APF
t_d	Mean	1460.9 s	1438.0 s
	Standard Deviation	202.6 s	122.0 s
Δv	Mean	1.423 m/s	1.382 m/s
	Standard Deviation	0.735 m/s	0.764 m/s

Table 9.9 Docking Spacecraft Statistics.

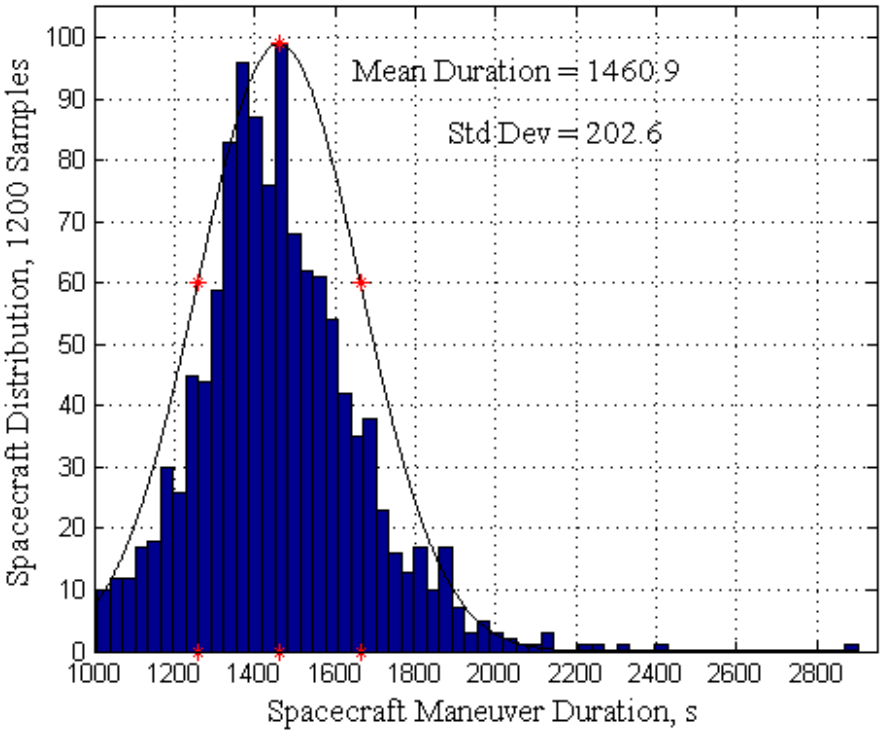

Figure 9.35 LQR/APF Spacecraft Docking Maneuver Duration Distribution.

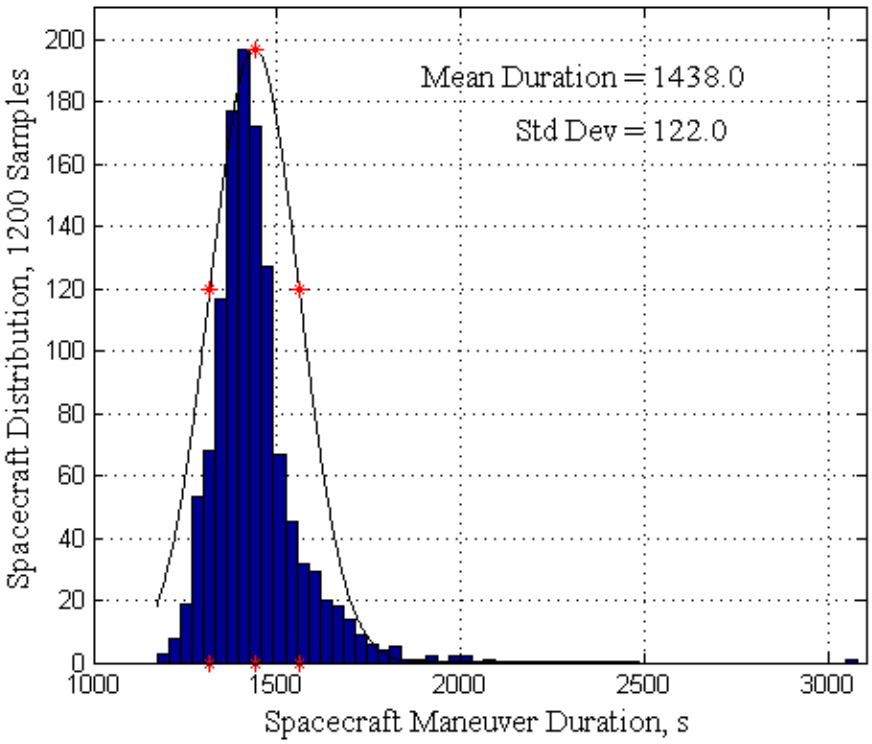

Figure 9.36 APF Spacecraft Docking Maneuver Duration Distribution.

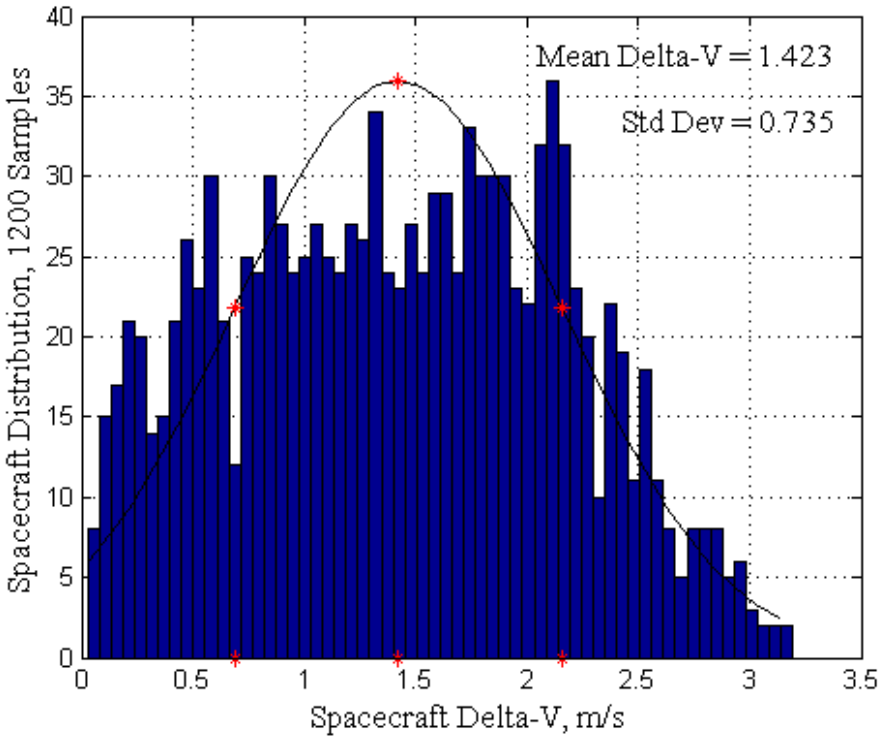

Figure 9.37 LQR/APF Spacecraft Docking Delta-V Distribution.

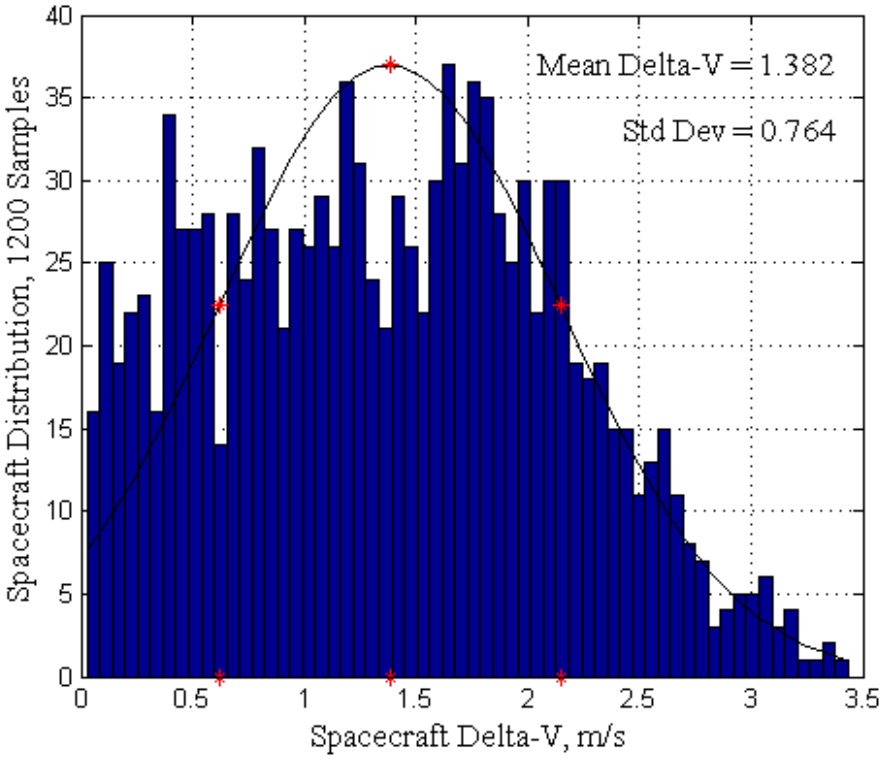

Figure 9.38 APF Spacecraft Docking Delta-V Distribution.

The statistical analysis is extended to a *per* docking maneuver basis, with the means and standard deviations of t_d and Δv listed in Table 9.10 The LQR/APF control algorithm performance continues to maintain a tighter Δv standard deviation. This confirms the LQR/APF control algorithm's more predictable performance. While on the other hand, the APF control algorithm significantly saturated the control actuation in approximately 10-20% of all docking maneuvers. This increase in actuator saturation is undesirable and increases the risk of collision in high density obstacle regions. The LQR/APF's and APF's docking maneuver maximum t_d distributions are shown in Figure 9.39 and Figure 9.40. The LQR/APF's and APF's docking maneuver maximum Δv distributions are shown in Figure 9.41 and Figure 9.42. The LQR/APF's maximum Δv is lower on average with a tighter distribution. The LQR/APF's and APF's total docking maneuver Δv distributions are shown in Figure 9.43 and Figure 9.44.

Docking Maneuver Statistics (200 Samples)		LQR/APF	APF
Max t_d	Mean	1672.0 s	1602.2 s
	Standard Deviation	194.5 s	162.9 s
Max Δv	Mean	2.348 m/s	2.362 m/s
	Standard Deviation	0.394 m/s	0.456 m/s
Total Δv	Mean	8.537 m/s	8.291 m/s
	Standard Deviation	1.918 m/s	1.989 m/s

Table 9.10 Docking Maneuver Statistics.

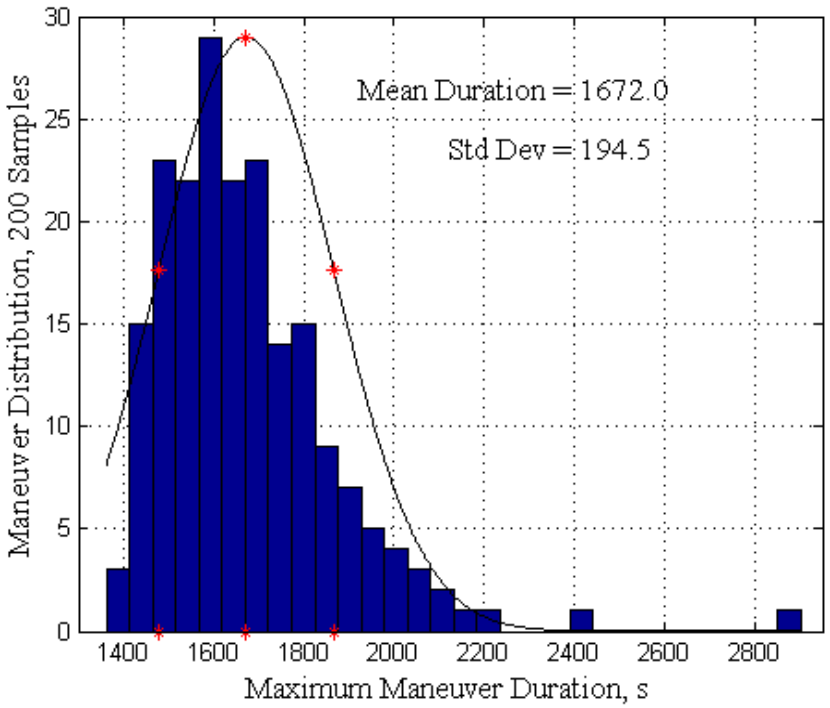

Figure 9.39 LQR/APF Docking Maneuver Maximum Duration Distribution.

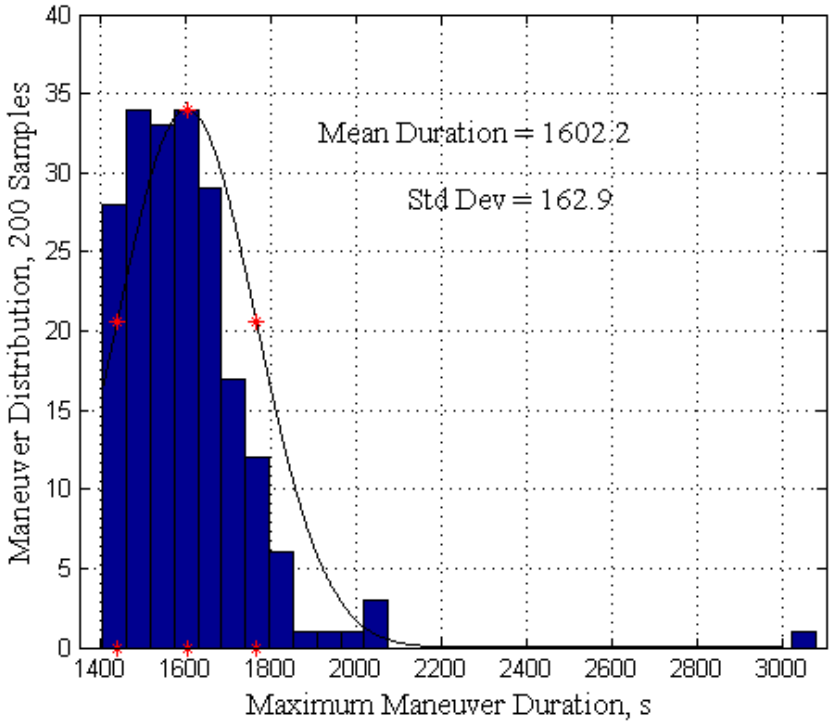

Figure 9.40 APF Docking Maneuver Maximum Duration Distribution.

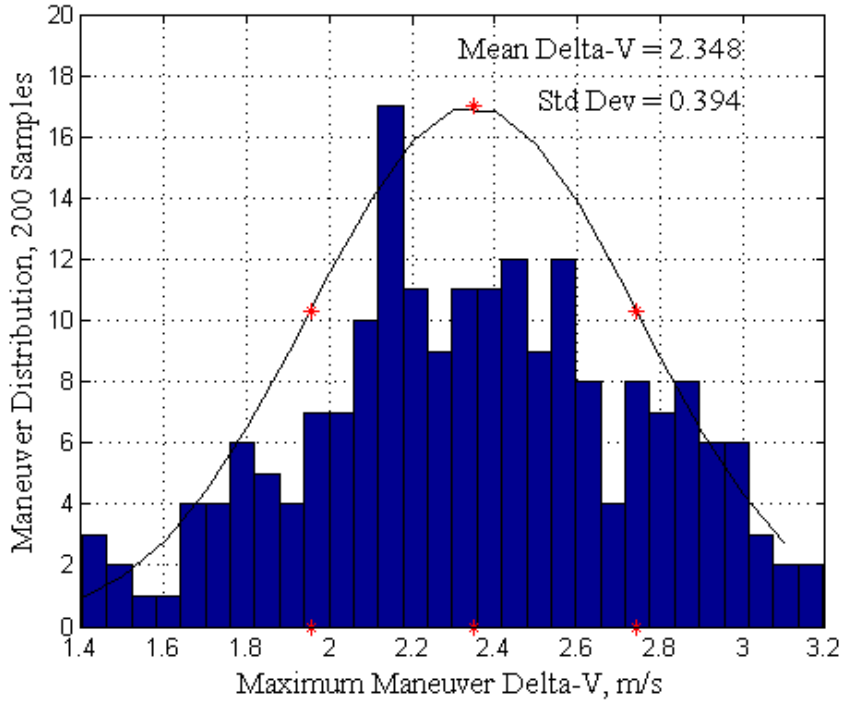

Figure 9.41 LQR/APF Docking Maneuver Maximum Delta-V Distribution.

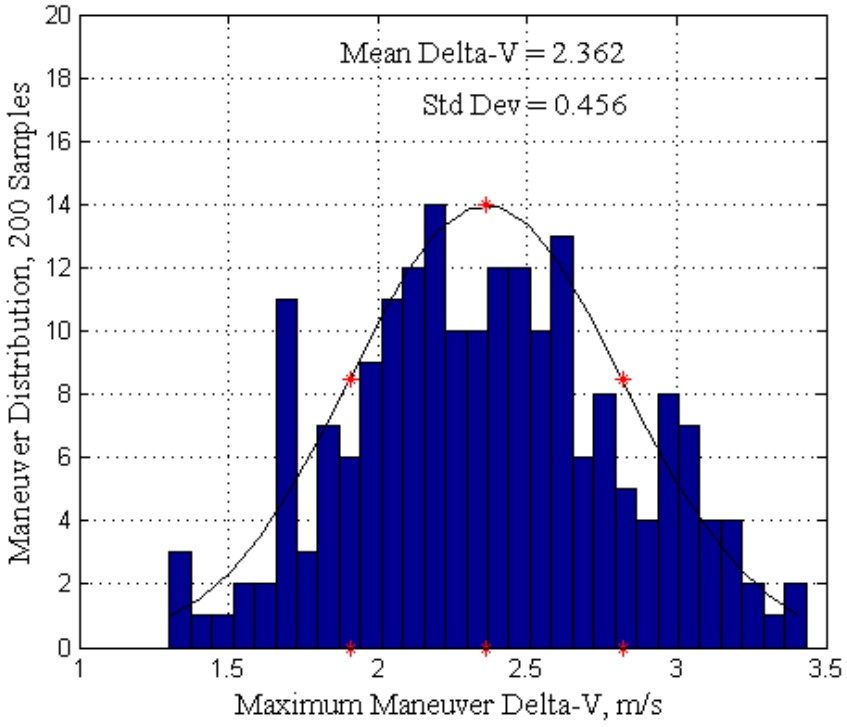

Figure 9.42 APF Docking Maneuver Maximum Delta-V Distribution.

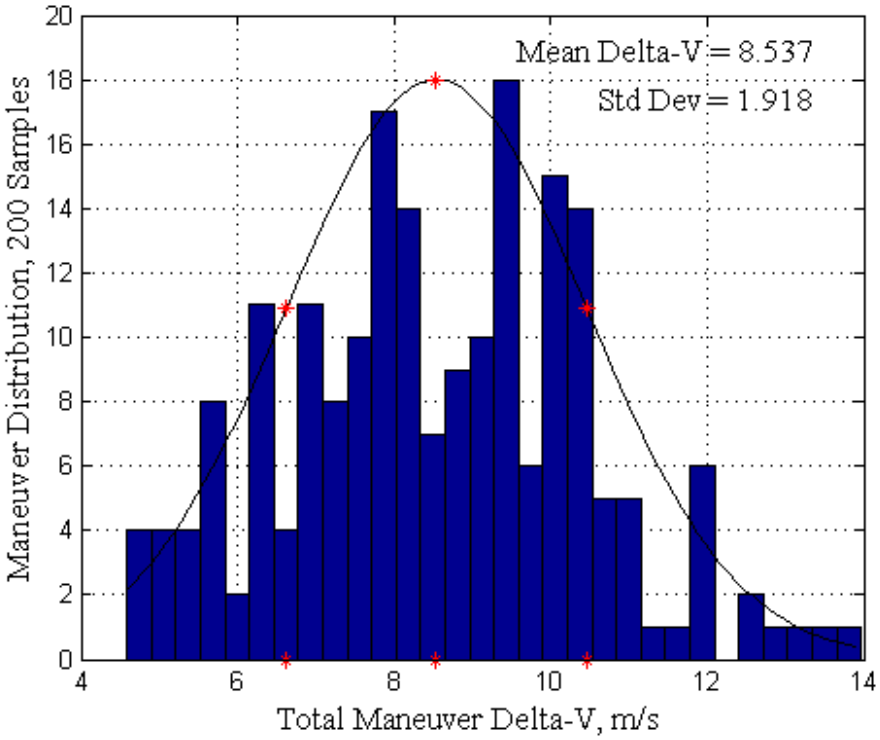

Figure 9.43 LQR/APF Docking Maneuver Total Delta-V Distribution.

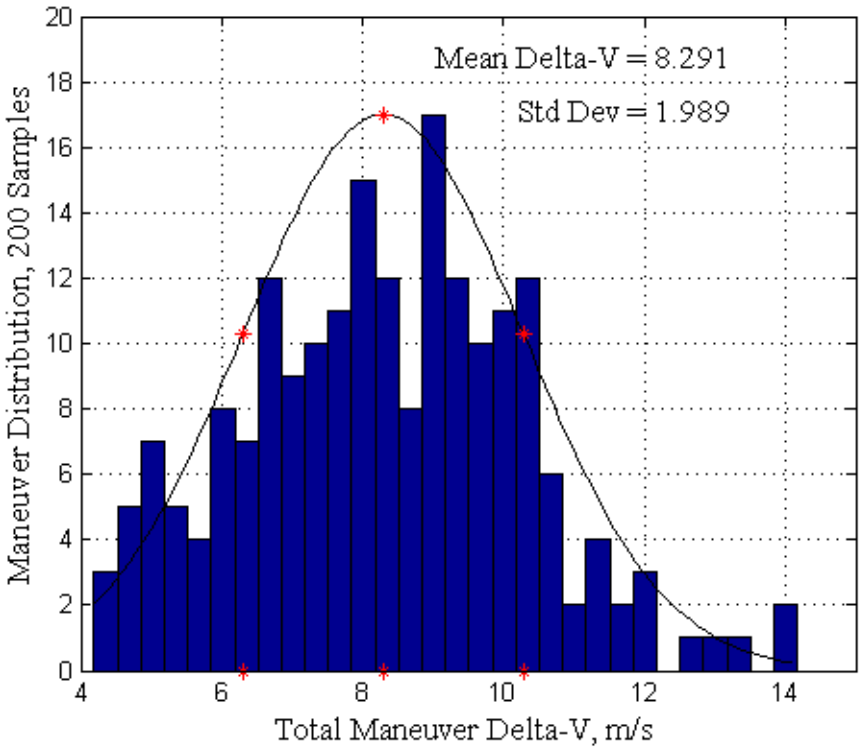

Figure 9.44 APF Docking Maneuver Total Delta-V Distribution.

F. MONTE-CARLO ANALYSIS CONCLUSIONS

The Monte-Carlo method analysis of close proximity maneuvers confirms that the LQR/APF control algorithm is a practical candidate for multiple spacecraft close proximity operations. Estimates of the mean and standard deviation of maneuver t_d and Δv show that average proximity maneuver control effort, Δv, efficiency is generally better than that of a highly tuned APF control algorithm. The LQR/APF showed a 0.5-1.0% efficiency improvement on a *per* spacecraft maneuver basis. The standard deviation of the LQR/APF control effort is consistently narrower then that of the APF. The LQR/APF showed a 10-20% narrower standard deviation. This narrow Δv standard deviation is valuable to both spacecraft designers and mission planners. It allows effective propellant sizing for close proximity operations. It also gives operational planners a useful tool for developing and forecasting maneuvers. The average mission duration, t_d, of all close proximity operations were maintained below the desired 30 minutes. The wider t_d standard deviation of the LQR/APF control algorithm is due to

velocity variations allowed by the algorithm. This standard deviation of approximately three minutes is acceptable for spacecraft operations which have traditionally been measured is terms of hours or days. Both the LQR/APF's and APF's Δv efficiency improves if the convergence rate of the Chase spacecraft is slowed. Therefore, the relative velocity can be used as a design trade-off between maneuver duration and efficiency.

Based on this Monte-Carlo analysis, the LQR/APF control algorithm appears suitable for application to emerging multiple spacecraft operations. Both the control efficiency and maneuver duration are reasonable for current spacecraft designs. Based on this analysis, the average Chase spacecraft could perform 20 - 40 close proximity maneuvers. This is more maneuvering than typically discussed in even the most aggressive spacecraft servicing operations. Therefore, the LQR/APF algorithm appears to be practical for multiple spacecraft close proximity maneuver control. Variation in spacecraft physical characteristics and orbital assumptions may cause some fluctuations in the total number of close proximity maneuvers which can be performed. However, it the LQR/APF control algorithm performs reliably for a wide range of maneuvers.

X. PERFORMANCE VALIDATION AND VISUALIZATION

A refined and validated multiple spacecraft six DOF simulation was configured for the purpose of control algorithm development during close proximity operations. The simulator incorporates a six DOF MATLAB and Simulink numerical model with 3D STK visualization. Both the model validation and 3D visualization are crucial to successful engineering evaluation during control algorithm development. A MATLAB-STK simulation interface was developed in order to validate the spacecraft models and provide detailed animation of simulations [75].

An effective test scenario is one which dependably simulates the environment in which the control algorithm is expected to operate. The application of the control algorithm for use on multiple spacecraft in close proximity operations drives the requirements that it be tested with computer-generated orbital dynamics and kinematics. Spacecraft model validation gives confidence that results are consistent and reliable. This research produced a method of spacecraft model validation by comparison with STK spacecraft analysis software developed by AGI [36]. STK is used as an orbital propagator for both simulation and emulation of the desired spacecraft models.

Accurate rendering of 3D spacecraft during compound close proximity maneuvers permit additional confidence in derived results. In addition to spacecraft model validation, STK can be used for detailed animation of spacecraft simulations. 2D animations based on complex numerical translational and attitude data often fail to convey true physical relationships. Even with multiple animated views these 2D representations may lead to conceptual misunderstandings. By utilizing 3D simulations more accurate engineering analysis can be conducted, allowing for undesirable performance to be discovered and corrected. This visualization is especially important for missions which require simultaneous control of multiple spacecraft maneuvering in close proximity. STK visualization can enhance spacecraft engineering analysis of relative translational motion and attitude while allowing for variations in spacecraft parameters and constraints. An overview of the MATLAB-STK simulation interface used for model validation and visualization is presented.

A. MATLAB-STK SIMULATION INTERFACE

Due to its ubiquitousness in engineering applications, a MATLAB based interface was desired. The common understanding of MATLAB code among both electrical and mechanical engineering researchers allows for a common starting point for multiple spacecraft simulation. In order to develop high fidelity spacecraft models, MATLAB and Simulink code is extremely useful and relevant. The high level language allows for ease in defining multiple parameters and numerical computation. The STK core modules allow for analytical and numerical orbital propagation of multiple spacecraft. Additional educational modules allow STK to be used in conjunction with MATLAB [39]. For this development MATLAB version 7.3.0267 (R2006b) and STK version 7.0.1 were utilized [39] [36]. Instructions for the installation and configuration of STK and MATLAB can be found at the AGI website [36]. By transferring information between MATLAB and STK, MATLAB constructed simulation dynamics and kinematics can be verified and visualized.

The MATLAB-STK interface takes advantage of STK's standard connection protocol for initializing STK from MATLAB. STK developers have enabled their code to be executed by MATLAB via an application program interface (API) called *AgConnec*t [36]. The connection allows for properly formatted commands to be sent via a Transmission Control Protocol and Internet Protocol (TCP/IP) port. By exploiting this feature, formatted data, such as ephemeris and attitude date can be passed between applications. For instance, the interface allows all spacecraft physical characteristics and simulation parameters to be defined in MATLAB and related to STK. Also, STK data can be passed to MATLAB for analysis The MATLAB-STK simulation interface exploits the strength of each software application. The general functions and information flow between MATLAB and STK are shown in Fig. 1. This is a broad overview of the primary functions and how both applications are employed.

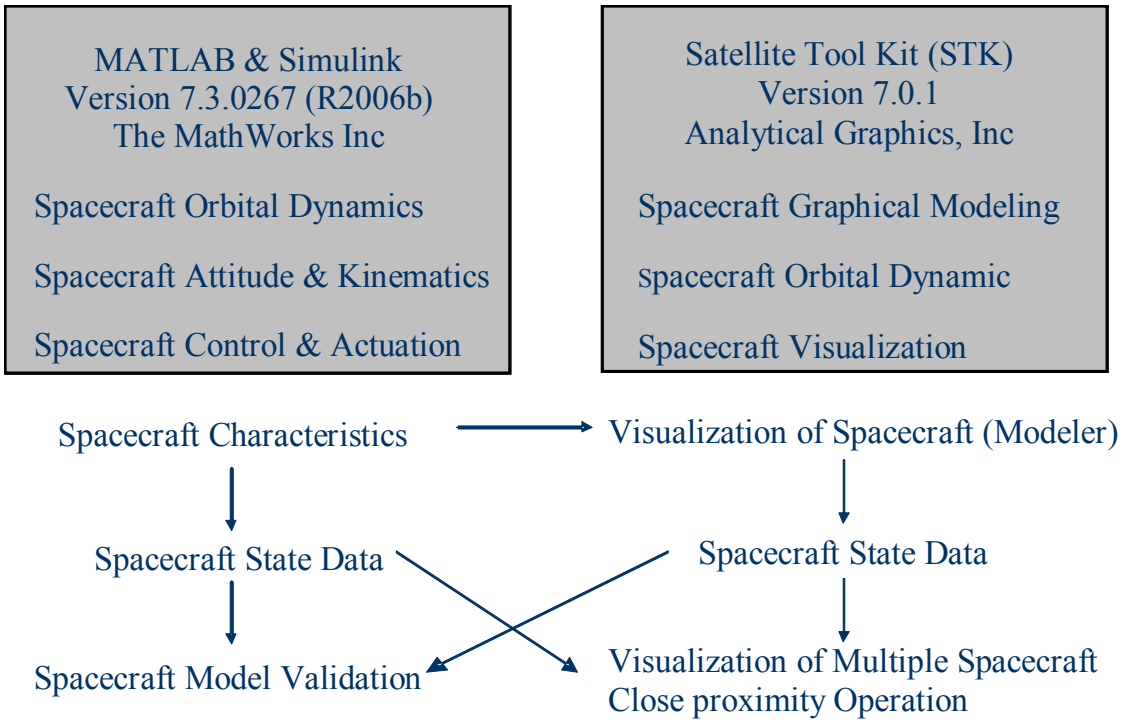

Figure 10.1 MATLAB-STK Simulation Interface Overview.

1. Overview of MATLAB/Simulink Spacecraft Modeling

Generalized spacecraft characteristics are modeled using MATLAB and Simulink [39]. The MATLAB and Simulink software components were modular structured allowing for variation in the number of spacecraft, orbital perturbations, spacecraft and obstacles parameters, and desired maneuvers. A high level Simulink spacecraft model is shown in Figure 10.2. The primary blocks are labeled with their functional calculations. The labeled blocks may have several sub layers with corresponding MATLAB code. The wiring has been simplified for visual flow, and some variables are shared between the kinematics and dynamics blocks. All multiple spacecraft simulation parameters are assigned, or defined, in MATLAB and implemented by a single Simulink model. Spacecraft states are concatenated into vectors and passed through the Simulink model. This vector of multiple spacecraft states passing through a single Simulink model is more computationally efficient and flexible then having multiple Simulink models to run simultaneously. The modeled space environment perturbations included variations in the

Earth's shape and mass, atmospheric drag on the spacecraft, third-body (Sun and Moon) forces, and solar-radiation pressure, and mass variation due to thruster firings; refer to Chapter III.E.

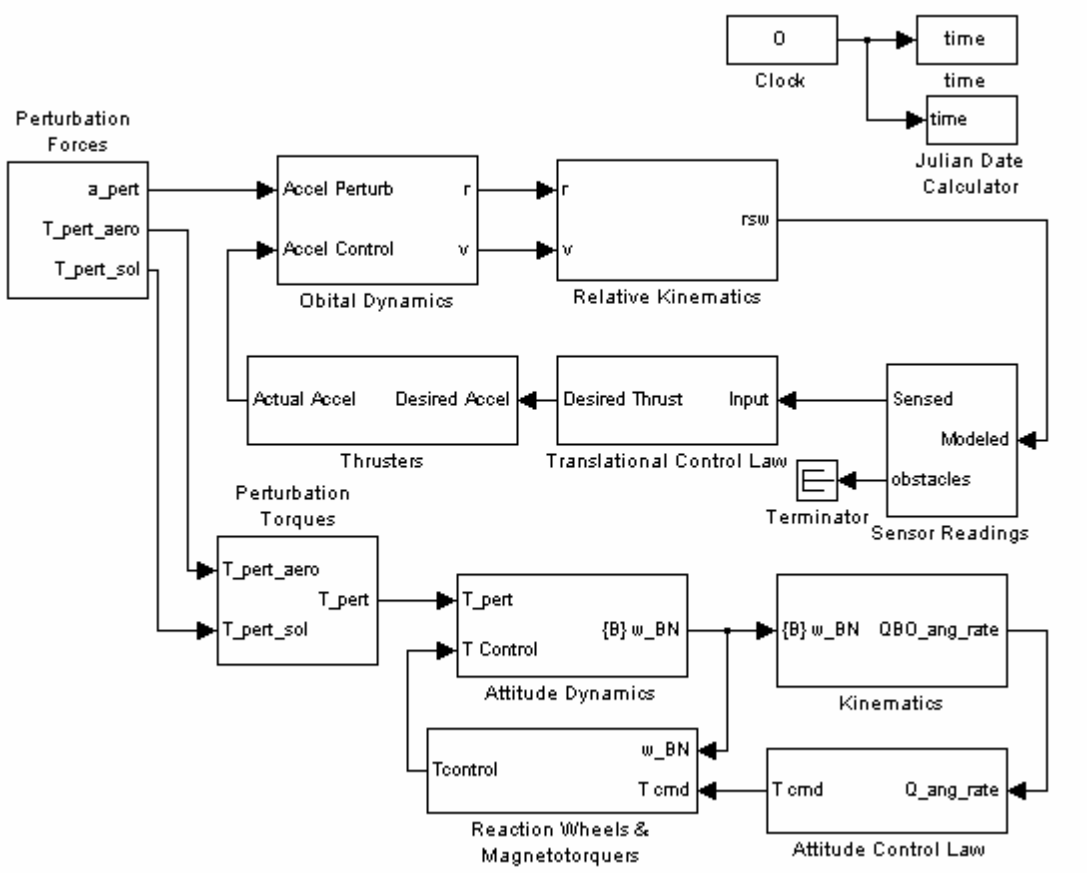

Figure 10.2 High Level Simulink Model for Multiple Spacecraft Simulation.

Several parameters must be defined, or assigned, before multiple spacecraft simulations can be executed. These parameters can include the number of simulations, initial time, selection of perturbations, number of spacecraft and obstacles, selection of control algorithms, and duration of simulation. For each simulation, a reference start time must be selected. This time can either be the current clock time, or a default reference time, which will be used for perturbation calculations, such as solar drag and third body effects. In this research, the Universal Time (UT1) in the format of [year, month, day, hour, minute, second] was used. With respect to this time, it is typical to select a duration

and time step for the simulation. A subset of simulation conditions are summarized in Table 10.1. The defining of spacecraft proximity operation to be within one kilometer is consistent with other researchers [76]. Although, this research considers rendezvous a subset of the close proximity maneuvers within the one kilometer range. It is worth mentioning that this meaning may vary from the phased rendezvous and proximity operations mission sequences outlined in the STK/Astrogator module [36].

Target Spacecraft	Target Minimum Altitude	300 km
	Target Maximum Altitude	2,000 km
Chase Spacecraft	Number of Chase Spacecraft	1 - 14
Chase Spacecraft Initial Position	R-axis	10 – 1,000 m
	S-axis	10 - 1,000 m
	W-axis	10 - 1000 m
Chase Spacecraft Initial Velocity	R-axis	0 m/s
	S-axis	0 m/s
	W-axis	0 m/s

Table 10.1 Close Proximity Spacecraft Maneuver Simulation Parameters.

In addition to simulation parameters, there are adjustable parameters for each spacecraft, such as initial position, attitude, size, shape, mass, and control actuators. Refer to Chapter III.G. for discussion of these parameters. The initial position of spacecraft could be specifically assigned, as in Chapter VIII, or randomly determined, as in Chapter IX. Selection of the desired maneuver prompts the fine-tuning of the control algorithm logic, such as decreasing the region of influence due to other Chase spacecraft during the terminal stage of docking. Visualization of the system responses required 3D evaluations of multiple spacecraft operating in close proximity. Although, 3D visualization is possible using MATLAB it is limited and leads to some programming challenges. For instance, changing views during animations may be beyond the ability of most users. For a majority of applications simple 3D point mass dynamic representation with 2D plots of associated metrics might be acceptable, such as used in [58]. However, close proximity maneuvers of high fidelity 3D spacecraft require more realistic

simulation for detailed performance evaluation of collision avoidance and docking. The visualization of spacecraft orbits and maneuvers is a specific field which is greatly simplified by the use of STK.

2. Overview of Satellite Tool Kit (STK)

The STK is a useful spacecraft environment simulator and propagator with a sophisticated graphical user interfaces (GUI) for spacecraft particular applications. However, since STK is not as commonly used in the engineering fields as MATLAB, development of a simulation interface between the two proved be useful. In order to visualize and evaluate the controller's performance, the MATLAB generated data was passed to STK. The spacecraft modeling and visualization capabilities of STK and its related modules and plug-ins are extensive. The visual analysis of the multiple spacecraft is invaluable during the evaluation of complex maneuvers. Both attitude and translational dynamics can be viewed from a wide variety of coordinate systems. Sensor field-of-view can be added to spacecraft models to provide an additional level of fidelity to the simulations. The STK/Connect module provides the means for STK to communicate with applications, such as MATLAB, through the use of a TCP/IP socket. This allows data to be transferred between MATLAB and STK. The sending and receiving of information to and from STK is intended to accurately model sensor performance between multiple spacecraft. Additional development may use the STK/Communications module to emulate communication propagation delay, frequency and bandwidth limitation, and bit error rates between all spacecraft [77]. This communication evaluation could be used to verify that higher order controller commands are properly implemented.

Typical STK users select scenario parameters via the GUI interface. This allows primary objects, such as spacecraft, to be loaded with desired constraints. Additionally sensors can be defined and attached to the satellites. The GUI has numerous drop-down menus with many layers for defining basic characteristic, 2D graphic animation, 3D graphic animation, and related constraints. A sample STK scenario GUI interface is shown in Figure 10.3. The numerous button toolbars are listed along the top of the GUI. The primary objects and attachments are listed along the left column. The 3D graphics

animation is shown in the scenario screen, with the general STK user page and 2D graphic animation tabbed along the bottom of the screen.

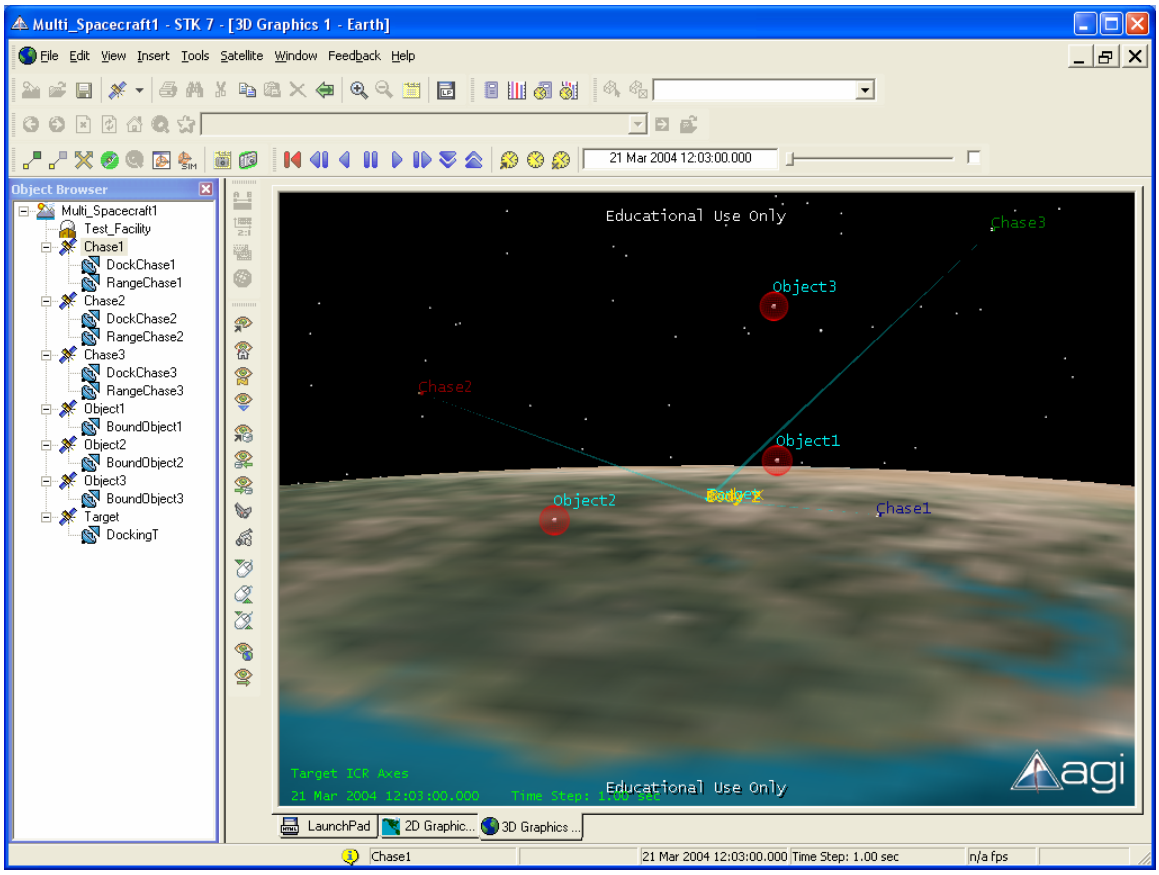

Figure 10.3 Sample STK Scenario GUI.

For those interested, several additional STK supported modules may be useful for spacecraft proximity operations research. These modules supplement the core STK visualization environment and assist in the evaluation of mission sequences. In particular, the Astrogator module can be used to support detailed maneuver analysis and operations [76]. The standard API links directly to the STK simulation environment. The Astrogator module is particularly useful for detailed mission analysis bring more fidelity to the entire mission control phasing and sequences. Further integration and discussion concerning the Astrogator module is beyond the scope of this research, refer to [76] and [36].

Similarly, the Advanced Close Approach Tool (ACAT) may be useful in collision avoidance maneuver planning. This STK module allows for additional situational awareness of any spacecraft or object in a referenced database. The ACAT module enables proximity indications and visual cues for variable close approach calculations. The use of uncertainty ellipsoids is more of a collision prediction tool for spacecraft path planning; refer to [47]. The analysis may be useful when evaluating potential sensor failure and communication link limitations. These modules may be more useful as space mission analysis and situational awareness applications.

3. MATLAB-STK Interface

The MATLAB-STK interface allows overall simulation parameters and spacecraft physical characteristics to be defined in MATLAB and related to STK via *mexConnect* commands, or formatted native STK commands. This interface technique takes advantage of STK's standard connection protocol for initializing STK from MATLAB. Once initialized, native STK commands can be called from MATLAB. The specific requirements of each particular STK command will determine which path and parameters are necessary. In addition to commands, formatted ephemeris and attitude files can be passed from MATLAB to STK. Likewise, data can be pulled from STK by using the `stkReport` command. Pulling ephemeris from STK and passing it to the MATLAB engine allows for comparison of independently generated STK and MATLAB spacecraft propagation. Evaluating the results of this comparison allows for validation of a developed MATLAB model based on the STK's High Precision Orbital Propagator (HPOP). The satellite of interest must be initialized and assigned with HPOP, before the propagation parameters and perturbations can be tailored via HPOP commands. By systematically enabling and comparing perturbation effects, each component of the spacecraft model can be validated. Once the numerical model has been validated, complex multiple spacecraft maneuvering can be animated for further assessment. However, precision close proximity maneuvers require refinement of the typical STK animation and views. For instance, the spacecraft's physical characteristics, such as size, shape and mass, may need to be modified. These parameters can be assigned as MATLAB variables and executed in STK by placing them as parameters in STK native

commands. The STK graphical model is primarily modified by using an assortment of STK Visual Option (VO) commands. The MATLAB-STK simulation interface code is written in MATLAB (*.m file* format) with some MATLAB/Simulink files nested within it. These sub-files serve as modular components of the MATLAB-STK interface, which allow for easy simulation variation and modification between users.

a. Instructions to Establish MATLAB-STK Interface Configuration

For the standard MATLAB-STK connection, both MATLAB and STK applications should be loaded and launched. Once launched, STK can be initialized from MATLAB by using the command `stkInit` or `agiInit`. Information on the path setting established can be found by using `agiGetConfig`. The next step in the connection processes is to open a socket, by using `stkOpen`. This configures the path and assigns a socket variable and a MATLAB variable, called `STKError`, for reference. Via the established MATLAB/STK connection, STK can be commanded by using either a select number of *mexConnect* commands or native STK commands. The command paths for these two methods are slightly different. The *mexConnect* commands are a limited subset of core MATLAB/STK interface commands which can be found by exploring MATLAB Help. These *mexConnect* commands are all prefixed with `stk`, such as the `stkInit` commands mentioned above. Since the *mexConnect* commands are limited, there is a general command which allows native STK commands to be executed from MATLAB. The general execution of STK commands via MATLAB can be conducted by using `stkExec(ConID, 'Command Path Parameter')`. A full listing of native STK commands which can be implemented in this fashion are listed in the *STK Help* menu under the path *<Automate/Extend/Integrate>, <Command Listings>, <Alphabetical Listing>*. The first step is usually to open a new STK scenario and ensure that any previously scenarios are closed. This ensures that a clean STK workspace is being established. The initial MATLAB-STK interface code is as follows:

```
stkInit
%initializes the STK/MATLAB Interface
remMachine = stkDefaultHost;
delete(get(0,'children'));        %clear open MATLAB charts
conID=stkOpen(remMachine);        %Open the Connect to STK
```

```
%%Check to see if a scenario is open
scen_open=stkValidScen;
if scen_open == 1
   rtn = questdlg('Close the current STK Scenario?');
   if ~strcmp(rtn,'Yes')
      stkClose(conID)
      return
   else
      stkUnloadv('/*')
   end
end
scen_nam=['Multi_Spacecraft',num2str(CONST.simnum)];
stkNewObj('/','Scenario',scen_nam);
```

The `conID` variable serves as a numeric representation of the established connection. This `conID` variable will be used often in `stkExec` commands. The STK scenario name was determined by a MATLAB variable, such as `CONST.simnum`, which can selected by the user to represent the number of simulations or scenarios desired. The sceneries name was then used to establish the new STK scenario. The `stkNewObj` command path is establishing the new scenario at the highest level path level. All subsequent STK objects, such as spacecraft, will be assigned under this current scenario.

b. STK Scenario Time Synchronization

The selected simulation initial date and time must be properly formatted and passed to STK. The date and time can be represented in several formats. One such format is [year, month, day, hour, minute, second], with a specific example to [2007, 07, 21, 12, 30, 0]. However, for STK applications the date and time must be rearranged and passed in the format of [21 Jul 2007 12:30:00.0]. If the MATLAB `datestr` command is used to determine the date and time, then a method of re-formatting and passing the date and time to STK is as follows:

```
para_now=datestr(now);
para_now(3)=' ';
para_now(7)=' ';
PARA.TIME=clock;     %sets formatted time for MATLAB
stkSetTimePeriod(para_now,para_now,'GREGUTC');
stkSetEpoch(para_now,'GREGUTC');
stkSyncEpoch;
newpara=strcat('SetValues_','"',para_now, '"',' 0.2 0.1');
newpara(10)=' ';
rtn=stkConnect(conID,'Animate',['Scenario/',scen_nam], newpara);
rtn=stkConnect(conID,'Animate',['Scenario/',scen_nam],'Reset');
```

In the `newpara` variable, the second to last number is animation time step, in seconds, and the last number is the highest speed or refresh rate, in seconds. The scale of these rates will influence the simulation by changing the animation of the scenario. These values can also be assigned as variables. When variables are passed into the body of an STK formatted command, brackets must be employed. This is apparent in the `stkConnect` command, where the variable `scen_nam` is used. Code listed in this research is only intended to be representative of the capability of the MATLAB-STK simulation interface, since there may be numerous methods of producing similar results.

c. Simple Satellite Object

Multiple spacecraft simulations require several spacecraft to be created. These spacecraft will need to be assigned physical and mass characteristics, attitude, and initial position and velocity. Along with the time determined above, these will serve as the basic parameters for initializing orbital propagation. The following sample code shows a spacecraft, called Target, being created:

```
stkNewObj('*/','Satellite','Target');
stkExec(conID,'VO */Satellite/Target Model Filename"s/c model"');
stkExec(conID, ['SetMass */Satellite/Target Value',…
num2str(Mass)])
stkExec(conID, ['SetMass */Satellite/Target Matrix',…
num2str(Inertia)]);
stkExec(conID,  ['SetAttitude  */Satellite/Target  Profile  InertFix
Quat', num2str(Quaternion0),' "CentralBody/Earth J2000"']);
stkSetPropCart('*/Satellite/Target','HPOP','J2000',tstart,…
tstopdummy,stepsize,orbitepoch,STATE.ri(1:3)',STATE.vi(1:3)');
```

The VO command is used to call a general spacecraft graphical model which is available to STK. The spacecraft graphical model can be selected from a default list, usually located at `C:\Program Files\AGI\STK 7\STKData\VO\Models\Space`, or a custom user generated spacecraft graphical model. The development of simple spacecraft graphical models will be discussed in more details in Chapter X.C. In this sample code several variables are passed from MATLAB to STK. First, the mass characteristics are described by the scalar `Mass` and the vector `Inertia`. `Inertia` is the inertia matrix of the spacecraft in vector form. Next, the initial quaternion attitude of the Target spacecraft, called `Quaternion0`, is assigned. For this research, an inertial fixed attitude about the Earth is assigned using quaternion in the inertial fixed coordinate

system (J2000). The propagation requires an initial start and stop time, called `tstart` and `tstopdummy` respectively. The stop time is temporarily assigned at this point to allow for assignment of the HPOP integrator. Next, the `orbitepoch` variable is assigned. This variable is usually equal to zero. Finally, the initial position vector, called `STATE.ri`, and velocity vector, called `STATE.vi`, are passed into STK. These parameters define the initial Target spacecraft states, but additional HPOP settings will allow for selection of integration and perturbations features. These will enable for our systematic spacecraft numerical model validation.

B. SPACECRAFT MODEL VERIFICATION

The high fidelity six DOF spacecraft model and dynamics was verified by comparison of developed MATLAB and Simulink orbital modeling with custom STK propagators. The STK High Precision Orbital Propagator (HPOP) was used to ensure spacecraft propagation conditions matched all model variations. Due to its commonality in both Simulink and STK, fourth order Runga-Kutta numerical integration method was used. Details on numerical integration applied to orbital propagation are provided in [32]. The position and velocity states of non-maneuvering spacecraft were compared for different propagation variations. The modeled space environment perturbations, including variations in the Earth's shape and mass (J2-J4 coefficients), atmospheric drag on the spacecraft, third-body (Sun and Moon) forces, and solar-radiation pressure, were sequentially compared. In addition, the simple two-body dynamics serves as a baseline case and all of these perturbations were included into a total perturbation case. The comparisons of the MATLAB/Simulink and STK propagations show that spacecraft model validation via a MATLAB-STK simulation interface is achievable.

1. MATLAB-STK Model Validation Interface

The multiple spacecraft model was modularly configured to allow each of the perturbation models to be sequentially evaluated and validated. The modular perturbations are triggered by an enabling constant. For each module a variable, called `CONST.PERT.choice`, enabled the proper Simulink blocks and assigned the

corresponding HPOP settings for STK. The MATLAB and Simulink numerical modeling of the perturbations was conducted as discussed in Chapter III and [32]. The desired precision of the numerical representations can vary as preferred by the user.

a. HPOP Integrator Setting

The HPOP command allows for assignment of both the numerical integration method, such as fourth order Runge-Kutta, and the Earth Gravitational model, such as EGM96 [36][32]. The HPOP command is of the general form, `stkExec(ConID, 'HPOP Path Parameter')`. Once the HPOP is assign to a spacecraft the perturbation forces can be further defined. The desired perturbations can be related to the enabling condition on the Simulink model, so that the various perturbations can be independently evaluated. The HPOP settings can be initialized for each spacecraft with the following code:

```
stkExec(conID,'HPOP */Satellite/Target Integrator IntegMethod RK4
ReportOnFixedStep On');
if CONST.PERT.choice==1;          %Earth Oblateness
   stkExec(conID,'HPOP */Satellite/Target Force Gravity "C:/Program
   Files/AGI/STK 7/STKData/CentralBodies/Earth/EGM96.grv" 4 0');
   stkExec(conID,'HPOP */Satellite/Target Drag Off');
   stkExec(conID,'HPOP */Satellite/Target Force SolarRad Off');
   stkExec(conID,'HPOP */Satellite/Target Force ThirdBodyGravity Moon
   Off');
   stkExec(conID,'HPOP */Satellite/Target Force ThirdBodyGravity Sun
   Off');
elseif CONST.PERT.choice==2;           %Aero Drag
   stkExec(conID,'HPOP */Satellite/Target Force Gravity "C:/Program
   Files/AGI/STK 7/STKData/CentralBodies/Earth/EGM96.grv" 0 0');
   stkExec(conID,['HPOP */Satellite/Target Drag On ',Cd,' 0.01 "1976
   Standard"']);
   stkExec(conID,'HPOP */Satellite/Target Force SolarRad Off');
   stkExec(conID,'HPOP */Satellite/Target Force ThirdBodyGravity Moon
   Off');
   stkExec(conID,'HPOP */Satellite/Target Force ThirdBodyGravity Sun
   Off');
elseif CONST.PERT.choice==3;           %Solar Drag
   stkExec(conID,'HPOP */Satellite/Target Integrator IntegMethod RK4
   ReportOnFixedStep On');
   stkExec(conID,'HPOP */Satellite/Target Force Gravity "C:/Program
   Files/AGI/STK 7/STKData/CentralBodies/Earth/EGM96.grv" 0 0');
   stkExec(conID,'HPOP */Satellite/Target Drag Off');
   stkExec(conID,['HPOP */Satellite/Target Force SolarRad On ',Cd,'
0.01']);
   stkExec(conID,'HPOP */Satellite/Target Force ThirdBodyGravity Moon
Off');
```

```
   stkExec(conID,'HPOP */Satellite/Target Force ThirdBodyGravity Sun
Off');
elseif CONST.PERT.choice==4;            %3-Body
   stkExec(conID,'HPOP */Satellite/Target Integrator IntegMethod RK4
   ReportOnFixedStep On');
   stkExec(conID,'HPOP */Satellite/Target Force Gravity "C:/Program
   Files/AGI/STK 7/STKData/CentralBodies/Earth/EGM96.grv" 0 0');
   stkExec(conID,'HPOP */Satellite/Target Drag Off');
   stkExec(conID,'HPOP */Satellite/Target Force SolarRad Off');
   stkExec(conID,'HPOP */Satellite/Target Force ThirdBodyGravity Moon
   On FromCB');
   stkExec(conID,'HPOP */Satellite/Target Force ThirdBodyGravity Sun
   On FromCB');
elseif CONST.PERT.choice==5;            %Total
   stkExec(conID,'HPOP */Satellite/Target Integrator IntegMethod RK4
   ReportOnFixedStep On');
   stkExec(conID,'HPOP */Satellite/Target Force Gravity "C:/Program
   Files/AGI/STK 7/STKData/CentralBodies/Earth/EGM96.grv" 4 0');
   stkExec(conID,['HPOP */Satellite/Target Drag On ',Cd,' 0.01 "1976
   Standard"']);
   stkExec(conID,['HPOP */Satellite/Target Force SolarRad On ',Cd,'
   0.01']);
   stkExec(conID,'HPOP */Satellite/Target Force ThirdBodyGravity Moon
   On FromCB');
   stkExec(conID,'HPOP */Satellite/Target Force ThirdBodyGravity Sun
   On FromCB');
else                                %Two Body with no perturbations
   stkExec(conID,'HPOP */Satellite/Target Integrator IntegMethod RK4
   ReportOnFixedStep On');
   stkExec(conID,'HPOP */Satellite/Target Force Gravity "C:/Program
   Files/AGI/STK 7/STKData/CentralBodies/Earth/EGM96.grv" 0 0');
   stkExec(conID,'HPOP */Satellite/Target Drag Off');
   stkExec(conID,'HPOP */Satellite/Target Force SolarRad Off');
   stkExec(conID,'HPOP */Satellite/Target Force ThirdBodyGravity Moon
Off');
   stkExec(conID,'HPOP */Satellite/Target Force ThirdBodyGravity Sun
Off');
end
```

Selection of the perturbation parameter, `CONST.PERT.choice`, activates the desired perturbation code. The selection may include all or none of the considered perturbations. For instance, the selection of `CONST.PERT.choice=6` results in the simple two-body propagation of the spacecraft orbit. Additionally, the two numbers in the `Gravity` settings command, at the beginning of each conditional setting, define the degree and order of the zonal coefficient terms, respectively [36]. The coefficients can be synchronized with any particular user defined model by modifying the *.grv file* selected.

As previously discussed, variable can be passed into the STK command. For instance, the coefficient of drag, represented by `Cd`, was incorporated into the `Drag` and `SolarRad` command settings.

Once the HPOP settings are completed the actual propagation can be computed. The initial stop time, `tstopdummy`, is replaced with the final desired propagation time, `tstop`. The spacecraft propagation command is as follows:

```
stkSetPropCart('*/Satellite/Target','HPOP','J2000',tstart,tstop,…
stepsize,orbitepoch,STATE.ri(1:3)',STATE.vi(1:3)');
```

where the propagation parameters, `stepsize`, `orbitepoch`, `STATE.ri`, and `STATE.vi` are the same as in the initial propagation command; refer to Chapter X.A.3.c.

b. Output STK Data to MATLAB Workspace

The data from STK propagation can be passed to the MATLAB Workspace. This will allow any desired post processing and evaluation of the data. Proper variable assignment of the data allows for state variable comparison. STK spacecraft state data can be passed in several formats. For this research the ECI position and velocity were desired. Sample code for passing spacecraft ephemeris is as follows:

```
[stkData,stkName]=stkReport('*/Satellite/Target','J2000 ECI
Position Velocity');
STATE.STK.time=stkFindData(stkData{1},'Time');
STATE.STK.r(:,1)=stkFindData(stkData{1},'x');
STATE.STK.r(:,2)=stkFindData(stkData{1},'y');
STATE.STK.r(:,3)=stkFindData(stkData{1},'z');
STATE.STK.v(:,1)=stkFindData(stkData{1},'vx');
STATE.STK.v(:,2)=stkFindData(stkData{1},'vy');
STATE.STK.v(:,3)=stkFindData(stkData{1},'vz');
STATE.STK.ECI=[STATE.STK.r, STATE.STK.v];
```

with the variables, prefixed with STATE.STK, arbitrary assigned. The STK ephemeris data is in column format with each row representing a numerical integration step. Although not the focus of this research, the user can also pass spacecraft attitude data back to the MATLAB Workspace. For completeness, format of the attitude code is as follows:

```
[stkData,stkNames]=stkReport('*/Satellite/Target','Attitude
Quaternions');
STATE.STK.Qbn(:,iqi+1)=stkFindData(stkData{1},'q1');
STATE.STK.Qbn(:,iqi+2)=stkFindData(stkData{1},'q2');
STATE.STK.Qbn(:,iqi+3)=stkFindData(stkData{1},'q3');
STATE.STK.Qbn(:,iqi+4)=stkFindData(stkData{1},'q4');
```

In this STK quaternions format, the first three elements are the vector components and the fourth element of the quaternion is the scalar (rotational) term.

2. Spacecraft Model Validation Results

Once the independently generated STK spacecraft data is passed into MATLAB, the propagated data can be compared against that of the MATLAB/Simulink model. Sample average differences between equivalent MATLAB and STK propagation methods are reported in Table 10.2. The results of this particular comparison appear to validate the MATLAB/Simulink numerical model. These averages were estimated over several LEO orbital propagations of 10 spacecraft at various initial conditions. Each orbit was propagated for 30 minutes and one minute durations with fixed one second step size. The step size introduces floating point errors in the calculation, such that relatively large integration step sizes will result in less precision. Also, as the propagation time duration increases the error between the propagations slowly and steadily increases. For instance, if the iteration step size is shortened to 0.5 seconds, the total perturbation position and velocity error for a 30 minute duration improves to 1.251 m and 1.895×10^{-3} m/s, respectively. The 30 minute time period was chosen in our simulations as the sample duration for a close proximity maneuver. The one minute duration represents an estimated time for slow communication or sensor update rates. Errors are a result of slight differences in the numerical integration techniques used for orbit propagations and numeric precision of constants/coefficients used during the calculation of perturbation forces. Overall, the relatively small average error in position and velocity validates the MATLAB model. Either the STK HPOP settings or the MATLAB/Simulink model can be refined and adjusted in order to achieve equivalent results. Our goal was not to achieve identical results, but to validate our spacecraft model by showing similar performance. The performance comparison thresholds may vary depending on the user's specific application. Also, some of the computational discrepancies between generated

data have periodic characteristics which result in variations of average errors due to the time period, or duration, selected. For instance, the average differences between two sets of ephemeris over one half of an orbital period may be less then those generated over one quarter of an orbital period.

Perturbation Model	Average Difference	30 minutes	1 minute
Simple Two Body	Position	8.237×10^{-10} m	3.391×10^{-10} m
	Velocity	6.890×10^{-13} m/s	6.082×10^{-13} m/s
J2 Perturbation	Position	4.650×10^{-1} m	1.102×10^{-3} m
	Velocity	6.908×10^{-4} m/s	5.346×10^{-5} m/s
J4 Perturbation	Position	8.286×10^{-1} m	1.452×10^{-3} m
	Velocity	1.256×10^{-3} m/s	7.183×10^{-5} m/s
Aero Drag	Position	1.725×10^{-6} m	1.301×10^{-8} m
	Velocity	2.904×10^{-9} m/s	6.291×10^{-10} m/s
Solar Drag	Position	2.075×10^{-3} m	6.656×10^{-8} m
	Velocity	4.267×10^{-6} m/s	3.277×10^{-9} m/s
Third Body Effects	Position	5.721×10^{-2} m	6.330×10^{-5} m
	Velocity	1.065×10^{-4} m/s	3.777×10^{-6} m/s
Total Perturbation	Position	1.360 m	3.739×10^{-3} m
	Velocity	2.047×10^{-3} m/s	3.357×10^{-4} m/s

Table 10.2 Average Difference of MATLAB and STK Propagation.

Due to the numerical precision and numerous environmental model subtle differences, comparison between different propagations techniques can be challenging. The numerical integration and step size precision, mentioned above, is a common source of discrepancy. However, orbital propagation algorithms can also vary greatly due to small precision and significant digit variations on constants used in near space orbital environment. For instance, the most common value for the Gravitational Parameter of Earth is $\mu = 398600.4418\times10^{9} m^{3} s^{-2}$ [32], however some common applications are based on models which use a default value of $\mu = 398600.4415\times10^{9} m^{3} s^{-2}$ [36]. This appears at first glance to be a small change in the last digit, however due to the magnitude of the value it can cause large variations in the gravitational effects on the spacecraft model.

The orbital discrepancy can be as large as several kilometers, over an orbital duration. These challenges are only compounded as the numerous perturbation effects are included into the spacecraft dynamics model. Large relative scale differences in computational constants can result in loss of desired numerical precision. The spacecraft models used in this research is scaled for precision on the order of meters.

Nevertheless, the relatively short duration for the considered maneuver limits the divergence due to perturbations. Minor propagation differences can cause difficulty for optimization based algorithms that depend on long duration path propagation and tracking. However, close proximity feedback based spacecraft control does not usually see these large perturbation changes. The difference between propagators for the duration of a sensor measurement cycle, typically less then one minute, are much smaller then those above. The short duration of the close proximity maneuvers and the disturbance rejection of feedback control make these systems rather tolerant of orbital perturbations [63][64]. Onboard sensors are capable of providing relative position and velocity data at rates much less then one minute. Therefore, the values in right column of Table 10.2 are well within an acceptable range for the close proximity maneuvers. Position error for short durations is approximately two millimeters. This propagation position error may be a good metric for determining the size requirements of on-orbit docking mechanisms. However, if onboard sensors and communication fail the propagation model used by each spacecraft should be standardized so that position estimation can be maintained during such blackout periods

In the future, the multiple spacecraft model can be compared against other high fidelity six DOF spacecraft models. For nano-satellite comparison, the Autonomous Rendezvous and Rapid Turnout Experiment Maneuverable Inspection Satellite (ARTIMIS) testbed at the University of Texas at Austin may be a likely candidate [78]. However, determining the cause of variations in model performance is very labor intensive. Simulation comparison is especially difficult if different numerical integration methods, synchronization, and software or coding are implemented. Ultimately, some variation in simulation performance is expected as perturbations are modeled based on empirical estimates of data. Precise model refinement would require feedback of

operational or test data for actual spacecraft, or components, in the same size and mission range. This consideration is not necessary or practical at the algorithm developmental level.

3. Spacecraft Model Propagation Challenges

Due to the numerical precision and numerous environmental model subtle differences, comparison between different propagations techniques can be challenging. The numerical integration and step size precision, mentioned above, is a common source of discrepancy. However, orbital propagation algorithms can also vary greatly due to small precision and significant digit variations on constants used in near space orbital environment. For instance, the most common value for the Gravitational Parameter of Earth is $\mu = 398600.4418 \times 10^9 m^3 s^{-2}$ [32], however some common applications are based on models which use a default value of $\mu = 398600.4415 \times 10^9 m^3 s^{-2}$ [36]. This appears at first glance to be a small change in the last digit, however due to the magnitude of the value it can cause large variations in the gravitational effects on the spacecraft model. The orbital discrepancy can be as large as several kilometers, over an orbital duration. These challenges are only compounded as the numerous perturbation effects are included into the spacecraft dynamics model. Large relative scale differences in computational constants can result in loss of desired numerical precision. The spacecraft models used in this research is scaled for precision on the order of meters, not the standard orbital scale of kilometers.

The advantage of close proximity research is in the relatively short duration of the spacecraft maneuvers. This relatively short duration, usually on the order of a quarter of an orbit, limits the divergence due to perturbations. Minor propagation differences can cause difficulty for optimization based algorithms that depend on long duration path propagation and tracking. However, feedback based control does not see these large perturbation changes. The difference between propagators for the duration of a sensor measurement cycle, typically less than one minute, are much smaller then those above. The short duration of the close proximity maneuvers and the feedback of relative position information make the APF controller rather tolerant to perturbation forces. Onboard

sensors are expected to provide relative position and velocity data at rates much less than one minute. Therefore, the values in right column of Table 10.2 are well within an acceptable range for the close proximity maneuvers. Position error for short durations is approximately two millimeters. This propagation position error may be a good metric for determining the size requirements of on-orbit docking mechanisms. However, if onboard sensors and communication fail the propagation model used by each spacecraft should be standardized so that position estimation can be maintained during such blackout periods.

C. SPACECRAFT MODEL ANIMATION

Accurate 3D simulation and visualization of multiple spacecraft close proximity maneuvers support engineering analysis; communicating the complexity and risk in these operations [76]. Multiple spacecraft maneuvers were initiated via MATLAB and animated via STK. The MATLAB initiates an STK TCP/IP connection to send commands over a specified port. The STK scenario is developed by executing STK commands. Spacecraft dimensional models for STK visualization must be pre-written in STK Modeler. Once the Model is loaded and the scenario is established data can be passed between STK and MATLAB. The coding for versatile spacecraft model selection and visualization can be extensive and complex. For this research visualization was limited to basic cubic, spherical, and cylindrical spacecraft models. More detailed models are available for use and reference in the STK model library.

Typically all control algorithm development is performed in MATLAB with STK used for performance visualize and verification. Comparison of control algorithm metrics is can be numerically evaluated in MATLAB. However, even with extensive metrics and spacecraft state information plots, standardized visualization of the 3D spacecraft environment is desired. In order to animate the maneuver, the MATLAB generated spacecraft ephemeris can be formatted and passed to STK to be viewed in 2D or 3D. The final STK scenario is very useful for visualizing the multiple spacecraft maneuver and assist in understanding and troubleshooting any issues that may arise. A snapshot of a sample 3D animation of multiple spacecraft is shown in Figure 10.4. Three cubic spacecraft are converging toward a common Target spacecraft, with a spherical obstacle in the background on the right. Both the view point and proximity are easily

adjusted with a standard computer mouse click and drag interface. The animation time steps can also be easily adjusted via toolbar buttons. These features allow the user to continually evaluate multiple spacecraft simultaneously performing proximity maneuvers. For instance as spacecraft are maneuvering through free-space the time step can be kept relatively high, but can be slowed down as spacecraft converge. Also, the viewpoint may be changed as one spacecraft passes close to an obstacle or eclipses another spacecraft. This is extremely useful for evaluating simultaneous docking maneuvers of multiple spacecraft. Finally, the user can zoom in on the any desired spacecraft to visually check that maneuver constraints, such as spatial safety margins, are properly maintained.

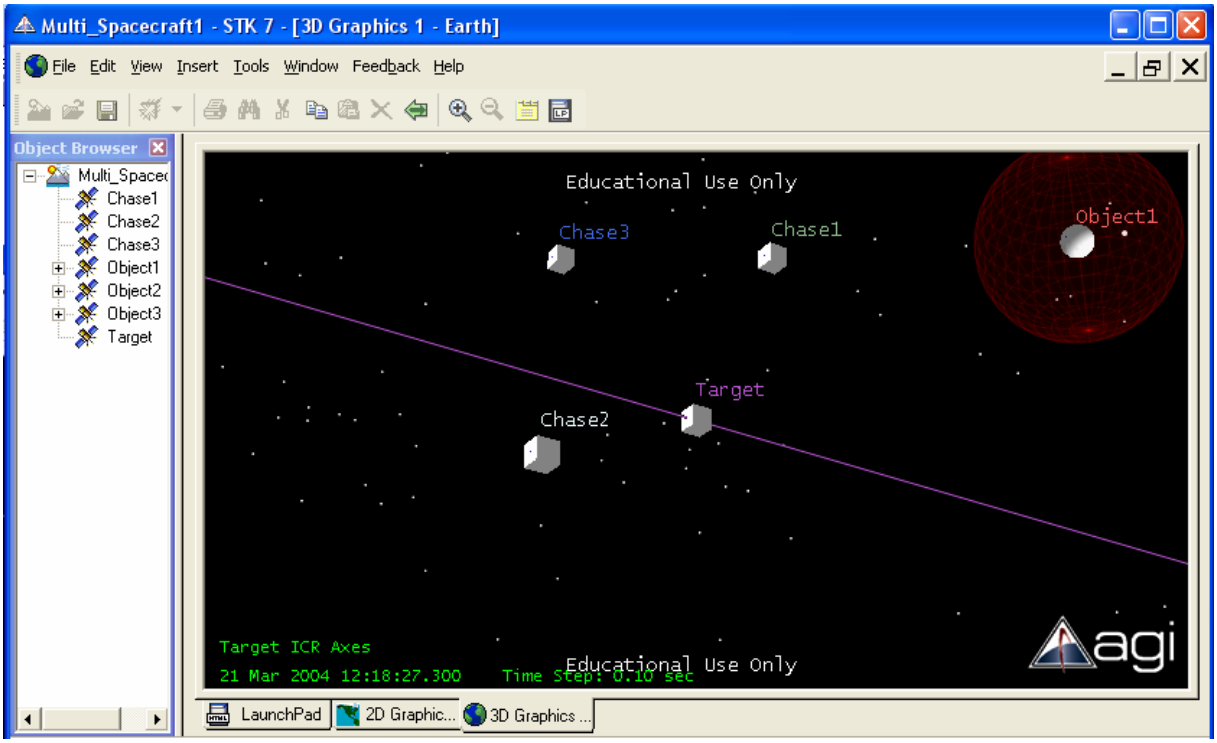

Figure 10.4 Sample 3D STK View of Animation Frame.

1. STK Spacecraft Model

For standard STK model assignment, a STK *.mdl file* must be available. This file is assigned to the desired STK satellite object using the following command:

```
stkExec(conID,['VO */Satellite/Target Model Filename
"C:\Userfiles\STK\Scenario\mdl_cube',num2str(isi),'.mdl"']);
```

where `isi` is a numerical variable which was used to distinguish multiple uses of the same basic model file. This is useful for the modeling of multiple spacecraft with the same basic model properties. A sample spacecraft model filename may be `mdl_cube1.mdl`. This would assign the STK satellite object named, Target, with the model spacecraft model described by `mdl_cube1.mdl`. The VO command can be used to call a general spacecraft model, which is available in STK. The spacecraft model can be selected from a default list, located at `C:\Program Files\AGI\STK 7\STKData\VO\Models\Space`, or a custom user generated spacecraft graphical model. These models can be manipulated an explored by opening the STK Modeler application and loading the desired spacecraft model. Some spacecraft advanced visualization setting can be modified from the basic model, by executing selected VO commands. These modifications may include scaling, labeling, and sensor descriptions.

a. STK Graphical Model Development

Three simple and distinctive spacecraft models were developed in this research. They are based on a simple spherical, cylindrical, or cubic spacecraft designs. Once again the `fprintf` command serves as our primary formatting tool. In this example the cubic shape serves as the primary body component with supplemental docking ports and thrusters components. The sample cubic spacecraft model code is presented for discussion and reference as follows:

```
filename=['C:\Userfiles\STK\Scenario\mdl_cube',num2str(isi),'.mdl'];
mdl_cube=fopen(filename,'wt');
fprintf(mdl_cube,'Component face\n');
fprintf(mdl_cube,'\t Polygon\n');
fprintf(mdl_cube,'\t\t Specularity\t 0.2\n');
fprintf(mdl_cube,'\t\t Shininess\t 25.6\n');
fprintf(mdl_cube,'\t\t Translucency\t 0.025\n');
fprintf(mdl_cube,'\t\t FaceColor burlywood\n');
fprintf(mdl_cube,'\t\t BackfaceCullable\t No\n');
fprintf(mdl_cube,'\t\t Translucency 0.025\n');
fprintf(mdl_cube,'\t\t NumVerts\t 4\n');
fprintf(mdl_cube,'\t\t Data\n');
fprintf(mdl_cube,['\t\t -',num2str(L/2),'\t ',num2str(L/2),'\t -
',num2str(L/2),'\n']);
```

```
fprintf(mdl_cube,['\t\t ',num2str(L/2),'\t ',num2str(L/2),'\t -
',num2str(L/2),'\n']);
fprintf(mdl_cube,['\t\t ',num2str(L/2),'\t ',num2str(L/2),'\t
',num2str(L/2),'\n']);
fprintf(mdl_cube,['\t\t -',num2str(L/2),'\t ',num2str(L/2),'\t
',num2str(L/2),'\n']);
fprintf(mdl_cube,'\t EndPolygon\n');
fprintf(mdl_cube,'EndComponent\n');

fprintf(mdl_cube,'Component left\n');
fprintf(mdl_cube,'\t Rotate 0 0 90\n');
fprintf(mdl_cube,'\t Refer\n');
fprintf(mdl_cube,'\t Component face\n');
fprintf(mdl_cube,'\t EndRefer\n');
fprintf(mdl_cube,'EndComponent\n');

fprintf(mdl_cube,'Component top\n');
fprintf(mdl_cube,'\t Rotate -90 0 0\n');
fprintf(mdl_cube,['\t Translate 0 0 ',num2str(L),'\n']);
fprintf(mdl_cube,'\t Refer\n');
fprintf(mdl_cube,'\t Component face\n');
fprintf(mdl_cube,'\t EndRefer\n');
fprintf(mdl_cube,'EndComponent\n');

fprintf(mdl_cube,'Component back\n');
fprintf(mdl_cube,['\t Translate 0 -',num2str(L),' 0\n']);
fprintf(mdl_cube,'\t Refer\n');
fprintf(mdl_cube,'\t Component face\n');
fprintf(mdl_cube,'\t EndRefer\n');
fprintf(mdl_cube,'EndComponent\n');

fprintf(mdl_cube,'Component right\n');
fprintf(mdl_cube,'\t Rotate 0 0 90\n');
fprintf(mdl_cube,['\t Translate ',num2str(L),' 0 0\n']);
fprintf(mdl_cube,'\t Refer\n');
fprintf(mdl_cube,'\t Component face\n');
fprintf(mdl_cube,'\t EndRefer\n');
fprintf(mdl_cube,'EndComponent\n');

fprintf(mdl_cube,'Component bottom\n');
fprintf(mdl_cube,'\t Rotate -90 0 0\n');
fprintf(mdl_cube,'\t Refer\n');
fprintf(mdl_cube,'\t Component face\n');
fprintf(mdl_cube,'\t EndRefer\n');
fprintf(mdl_cube,'EndComponent\n');
```

This section of code generates a primary box shape by defining and rotating a simple face. The variable, L, represents user defined dimensional parameters which can be implemented in the model. In this code the face component was defined first. Next, this component was translated and rotated as necessary to determine all size sides of the cubic.

Additional components can be defined, such as a simple cone. The code for a conic docking component is as follows:

```
fprintf(mdl_cube,'Component dockingpoint\n');
fprintf(mdl_cube,'\t Cylinder\n');
fprintf(mdl_cube,'\t\t Translucency\t 0.25\n');
fprintf(mdl_cube,'\t\t BackfaceCullable\t Yes\n');
fprintf(mdl_cube,'\t\t NumSides\t 32\n');
fprintf(mdl_cube,['\t\t Face1Position\t ',num2str(L/10),' 0 0\n']);
fprintf(mdl_cube,['\t\t Face1Radius\t ',num2str(L/20),'\n']);
fprintf(mdl_cube,'\t\t Face1Normal\t 1 0 0\n');
fprintf(mdl_cube,['\t\t Face2Position\t -',num2str(L/10),' 0 0\n']);
fprintf(mdl_cube,['\t\t Face2Radius\t ',num2str(L/100),'\n']);
fprintf(mdl_cube,'\t\t Face2Normal\t -1 0 0\n');
fprintf(mdl_cube,'\t EndCylinder\n');
fprintf(mdl_cube,'EndComponent\n\n');
```

More detailed components can be generated by combining components. For instance, a conic thruster can be combined with a flame, as follows:

```
fprintf(mdl_cube,'Component Thrust_Cone\n');
fprintf(mdl_cube,'\t Cylinder\n');
fprintf(mdl_cube,'\t\t Translucency 0.05\n');
fprintf(mdl_cube,'\t\t BackfaceCullable\t No\n');
fprintf(mdl_cube,'\t\tFaceColor\t gray16\n');
fprintf(mdl_cube,'\t\t NumSides\t 32\n');
fprintf(mdl_cube,['\t\t Face1Position\t -',num2str(L/10),' 0 0\n']);
fprintf(mdl_cube,['\t\t Face1Radius\t ',num2str(L/40),'\n']);
fprintf(mdl_cube,'\t\t Face1Normal\t -1 0 0\n');
fprintf(mdl_cube,['\t\t Face2Position\t ',num2str(L/10),' 0 0\n']);
fprintf(mdl_cube,['\t\t Face2Radius\t ',num2str(L/200),'\n']);
fprintf(mdl_cube,'\t\t Face2Normal\t 1 0 0\n');
fprintf(mdl_cube,'\t EndCylinder\n');
fprintf(mdl_cube,'EndComponent\n');

fprintf(mdl_cube,'Component Thrust_Flame\n');
fprintf(mdl_cube,'PolygonMesh\n');
fprintf(mdl_cube,'\t FaceColor\t black\n');
fprintf(mdl_cube,'\t FaceEmissionColor\t gray100\n');
fprintf(mdl_cube,'\t NoDiffuseLighting\n');
fprintf(mdl_cube,'\t SmoothShading\t No\n');
fprintf(mdl_cube,'\t Translucency\t 0\n');
fprintf(mdl_cube,'\t Specularity\t 0\n');
fprintf(mdl_cube,'\t Shininess 51\n');
fprintf(mdl_cube,'\t Texture\n');
fprintf(mdl_cube,'\t\t RGB flametex-white\n');
fprintf(mdl_cube,'\t\t Alpha flamealpha3\n');
fprintf(mdl_cube,'\t\t Parm AA\n');
fprintf(mdl_cube,'\t EndTexture\n');
fprintf(mdl_cube,'\t NumVerts 12\n');
fprintf(mdl_cube,'\t DataTx\n');
fprintf(mdl_cube,'\t\t 0\t 0.001\t -1.518\t 0\t 0\n');
fprintf(mdl_cube,'\t\t 0\t 0.001\t 1.287\t 0\t 1\n');
```

```
fprintf(mdl_cube,'\t\t -10.55\t -0.001\t 1.287\t 1\t 1\n');
fprintf(mdl_cube,'\t\t -10.55\t -0.001\t -1.518\t 1\t 0\n');
fprintf(mdl_cube,'\t\t 0\t 1.21\t 0.586\t 0\t 0\n');
fprintf(mdl_cube,'\t\t 0\t -1.21\t -0.817\t 0\t 1\n');
fprintf(mdl_cube,'\t\t -10.55\t -1.21\t -0.817\t 1\t 1\n');
fprintf(mdl_cube,'\t\t -10.55\t 1.21\t 0.586\t 1\t 0\n');
fprintf(mdl_cube,'\t\t 0\t -1.21\t 0.586\t 0\t 0\n');
fprintf(mdl_cube,'\t\t 0\t 1.21\t -0.817\t 0\t 1\n');
fprintf(mdl_cube,'\t\t -10.55\t 1.21\t -0.817\t 1\t 1\n');
fprintf(mdl_cube,'\t\t -10.55\t -1.21\t 0.586\t 1\t 0\n');
fprintf(mdl_cube,'\t NumPolys 3\n');
fprintf(mdl_cube,'\t Polys\n');
fprintf(mdl_cube,'4 3 2 1 0\n');
fprintf(mdl_cube,'4 7 6 5 4\n');
fprintf(mdl_cube,'4 11 10 9 8\n');
fprintf(mdl_cube,'\t EndPolygonMesh\n');
fprintf(mdl_cube,'EndComponent\n');
```

The `Thrust_Flame` component uses textures, such as `flametex-white` and `flamealpha3,` to render a more complex image. For detailed discussion of textures refer to AGI [36].

Once the basic components are defined, they can be reproduced within a loop. In this example six thruster cones, based on `Thrust_Cone`, will be placed on the center of each face of the cubic spacecraft. Their respective rotations and positions can be determined from a user defined matrix, such as `thrustr(thrusti,:)` and `thrustl(thrusti,:)`.

```
for thrusti=1:6       %thruster body positions (-z,-x,x,z,-y,y)
   fprintf(mdl_cube,['\t Component
Thrust_Cone',num2str(thrusti),'\n']);
   fprintf(mdl_cube,['\t\t Rotate\t ',thrustr(thrusti,:),'\n']);
   fprintf(mdl_cube,['\t\t Translate\t ',thrustl(thrusti,:),'\n']);
   fprintf(mdl_cube,'\t\t Refer\n');
   fprintf(mdl_cube,'\t\t Component Thrust_Cone\n');
   fprintf(mdl_cube,'\t\t EndRefer\n');
   fprintf(mdl_cube,'\t EndComponent\n');

   fprintf(mdl_cube,['\t Component Thrust_Flame',
   num2str(thrusti),'\n']);
   fprintf(mdl_cube,['\t\t Rotate\t ',thrustr(thrusti,:),'\n']);
   fprintf(mdl_cube,['\t\t Translate\t ',thrustl(thrusti,:),'\n']);
   fprintf(mdl_cube,'\t\t Refer\n');
   fprintf(mdl_cube,'\t\t Component Thrust_Flame\n');
   fprintf(mdl_cube,'\t\t EndRefer\n');
   fprintf(mdl_cube,['\t\t Articulation\t Thrust_Flame',
   num2str(thrusti),'\n']);
   fprintf(mdl_cube,'\t\t uniformScale\t Size\t 0\t 0\t 1\n');
   fprintf(mdl_cube,'\t\t EndArticulation\n');
   fprintf(mdl_cube,'\t EndComponent\n');
end
```

The STK `Articulation` command is further discussed in Chapter X.C.2.b.

Now that the primary spacecraft components, including the cubic faces, docking ports, and thrusters, are defined, they can be assembled. Each of the six cubic components will make the basic spacecraft shape. Next, a thruster with flames articulation will be centered on each face of the cubic spacecraft. Finally, the desired number of docking ports will be added based on the total number of spacecraft. For instance, the Target spacecraft, numerically represented by `isi=1` will have six docking ports. While, all other Chase spacecraft will have single color coordinated docking port assigned from user defined matrices, such as `dockc(isi-1,:)`, `dockr(dockpi,:)`, and `dockl(dockpi,:)`, for color, rotation, and position, respectively. Each component is referred onto the graphical spacecraft model as follows:

```
fprintf(mdl_cube,'Component spacecraft\n');
fprintf(mdl_cube,'Root\n');
fprintf(mdl_cube,'\t Refer\n');
fprintf(mdl_cube,'\t\t Component face\n');
fprintf(mdl_cube,'\t EndRefer\n');
fprintf(mdl_cube,'\t Refer\n');
fprintf(mdl_cube,'\t\t Component left\n');
fprintf(mdl_cube,'\t EndRefer\n');
fprintf(mdl_cube,'\t Refer\n');
fprintf(mdl_cube,'\t\t Component top\n');
fprintf(mdl_cube,'\t EndRefer\n');
fprintf(mdl_cube,'\t Refer\n');
fprintf(mdl_cube,'\t\t Component back\n');
fprintf(mdl_cube,'\t EndRefer\n');
fprintf(mdl_cube,'\t Refer\n');
fprintf(mdl_cube,'\t\t Component right\n');
fprintf(mdl_cube,'\t EndRefer\n');
fprintf(mdl_cube,'\t Refer\n');
fprintf(mdl_cube,'\t\t Component bottom\n');
fprintf(mdl_cube,'\t EndRefer\n');

for thrusti=1:6 %% thruster positions (-z,-x,x,z,-y,y)
   fprintf(mdl_cube,'\t Refer\n');
   fprintf(mdl_cube,['\t\t Component Thrust_Cone',num2str(thrusti),
   '\n']);
   fprintf(mdl_cube,'\t EndRefer\n');
   fprintf(mdl_cube,'\t Refer\n');
   fprintf(mdl_cube,['\t\t Component Thrust_Flame',num2str(thrusti),
   '\n']);
   fprintf(mdl_cube,'\t EndRefer\n');
end

if isi==1
   for dockpi=1:6
```

```
    fprintf(mdl_cube,'\t Refer\n');
    fprintf(mdl_cube,'\t\t Component dockingpoint\n');
    fprintf(mdl_cube,['\t\t FaceColor\t ',dockc(dockpi,:),'\n']);
    fprintf(mdl_cube,['\t\t Rotate\t ',dockr(dockpi,:),'\n']);
    fprintf(mdl_cube,['\t\t Translate\t ',dockl(dockpi,:),'\n']);
    fprintf(mdl_cube,'\t EndRefer\n');
    end
  elseif isi<=7
    fprintf(mdl_cube,'\t Refer\n');
    fprintf(mdl_cube,'\t\t Component dockingpoint\n');
    fprintf(mdl_cube,['\t\t FaceColor\t ',dockc(isi-1,:),'\n']);
    fprintf(mdl_cube,['\t\t Rotate\t ',dockr(3,:),'\n']);
    fprintf(mdl_cube,['\t\t Translate\t ',dockl(3,:),'\n']);
    fprintf(mdl_cube,'\t EndRefer\n');
  else
    fprintf(mdl_cube,'\t Refer\n');
    fprintf(mdl_cube,'\t\t Component dockingpoint\n');
    fprintf(mdl_cube,['\t\t FaceColor\t ',dockc(1,:),'\n']);
    fprintf(mdl_cube,['\t\t Rotate\t ',dockr(3,:),'\n']);
    fprintf(mdl_cube,['\t\t Translate\t ',dockl(3,:),'\n']);
    fprintf(mdl_cube,'\t EndRefer\n');
  end
  fprintf(mdl_cube,'EndComponent\n\n');
  fclose(mdl_cube);
```

The entire graphical modeling section of code can be called as one MATLAB file, such as `model_cubic_dock_thrust.m`. This allows this graphical modeling to be isolated from the other MATLAB-STK simulation interface code.

b. STK 3D Visualization Options

Several graphics and VO parameters can be tailored for clear visualization of multiple spacecraft maneuvers. The primary scaling of the model view is accomplished by selecting the following VO command:

```
stkExec(conID, 'VO */Satellite/Target ScaleModel Ratio');
```

where `Ratio` is the number of times larger that the spacecraft model appears during animation. For instance, spacecraft being observed from a lunar perspective may nee to appear hundreds of times larger in order to appear form this perspective. In close proximity operations, `Ratio=1` was selected in order that spacecraft appear strictly as they are modeled. The label and graphic features of the spacecraft can be modified as follows:

```
stkExec(conID,'VO */Satellite/Target LabelOffsetInPixels Off');
stkExec(conID,['VO */Satellite/Target LabelXYZ 0.0 ',num2str(L),'
0.0']);
stkExec(conID,'Graphics */Satellite/Target Basic Color cyan LineWidth
4.2');
```

The orbital path is typically shown for spacecraft propagation. However, for multiple spacecraft these projections can result in visual confusion. These orbital pass projections can be modified as follows:

```
stkExec(conID,'VO */Satellite/Target Pass OrbitLead None');
stkExec(conID,'VO */Satellite/Target Pass OrbitTrail None');
```

In addition to labeling the spacecraft, inclusion of a body axis may be useful for attitude reference. A 3D spacecraft body reference frame can be added to a model as follows:

```
stkExec(conID, 'VO */Satellite/Target SetVectorGeometry Data
FixVectorAxesScaleToModel On Scale 0.01');
stkExec(conID,'VO */Satellite/Target SetVectorGeometry Modify
"Satellite/Target Body Axes" Show On ShowLabel On Color Gold
ArrowType 3D');
```

Besides straightforward labeling and graphics modifications, additional sensors may be desired. These sensors may represent actual ranging and communication devices, or useful visual projections for engineering analysis. Each sensor must be uniquely named, so that STK dos not confuse identically named sensors in the same scenario. In this research a conical ranging and docking sensor was added to each Chase spacecraft. The ranging sensor represents the local sensor view from the Chase spacecraft to the Target spacecraft, or position. The simple conic range sensor is assigned as follows:

```
stkExec(conID,['Location */Satellite/',Chase,'/Sensor/Range',Chase,'
Fixed Cartesian ',num2str(L/2+0.001),' 0 0']);
stkExec(conID,['Define */Satellite/',Chase,'/Sensor/Range',Chase,'
SimpleCone 0.25']);
stkExec(conID,['Point */Satellite/',Chase,'/Sensor/Range',Chase,'
Targeted Tracking Satellite/Target Hold']);
stkExec(conID,['VO */Satellite/',Chase,'/Sensor/Range',Chase,'
Projection SpaceProjection ',num2str(rsw_norm(1,isi+1)-L)]);
stkExec(conID,['Graphics */Satellite/',Chase,'/Sensor/Range',Chase,'
SetColor cyan']);
```

First, the sensor is positioned and defined on a spacecraft with an exclusive name. In this example, `Chase` is an alpha-numeric label unique to each Chase spacecraft. Next, the pointing direction of the sensor is determined. The range sensor continually points toward the Target spacecraft and adjusts its projection based on the relative distance from the Target, represented by `rsw_norm`. This modification of the projection may not be necessary for most sensor applications. Finally, the color of the sensor projection can be modified as desired. Similarly, a docking sensor may be added to the spacecraft graphical model as follows:

```
stkExec(conID,['Location */Satellite/',Chase,'/Sensor/Dock',Chase,'
Fixed Cartesian ',num2str(L/2+0.001),' 0 0']);
stkExec(conID,['Define */Satellite/',Chase,'/Sensor/Dock',Chase,'
SimpleCone 45.0']);
stkExec(conID,['Point */Satellite/',Chase,'/Sensor/Dock',Chase,'
Fixed YPR 321 0 90 0']);
stkExec(conID,['VO */Satellite/',Chase,'/Sensor/Dock',Chase,'
Projection SpaceProjection ',num2str(L/2)]);
stkExec(conID,['Graphics */Satellite/',Chase,'/Sensor/Dock',Chase,'
SetColor ',dockc(isi-1,:)]);
```

The docking sensor is located at the same position, but has a much wider cone of 45 degrees. The docking sensor points in a fixed direction from the spacecraft body axis, with a limited projection of half of the spacecraft's length. Next, the sensor projection color is selected from a matrix, called `dockc`. These simple sensors are useful in visually representing ranges and FOV as spacecraft are animated through their dynamics and kinematics. A sample cubic spacecraft with labels and sensor projections is shown in Figure 10.5. The three body axis are shown in yellow with labels. The green projection on the right side of the spacecraft is the docking sensor projection and the small cyan line extending to the bottom left is the range sensor. The white protrusions on ht top and right are thruster firings.

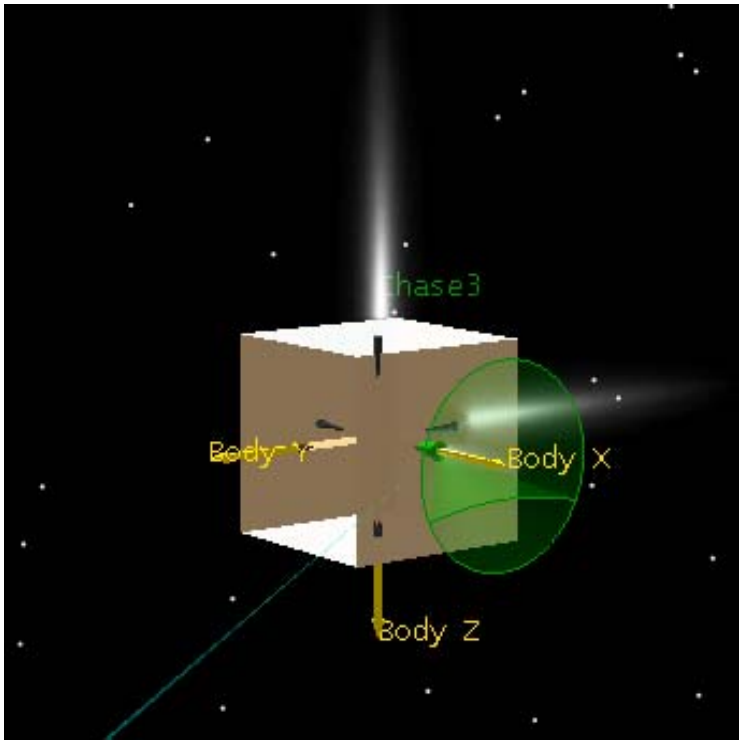

Figure 10.5 Sample Cubic Spacecraft Graphical Model.

One additional sensor projection was useful in visualizing spacecraft exclusion regions during close proximity operations. A boundary sensor projection was placed around regions that spacecraft were intended to avoid. These areas were referred to as obstacles and labeled with `Obstcl`. Spherical projections encompassing these regions can be generated using conical sensor defined with 360 degrees FOV, such as follows:

```
stkExec(conID,['Define */Satellite/',Obstcl,'/Sensor/Bound',Obstcl,'
Conical 0 180 0 360']);
stkExec(conID,['SetConstraint
*/Satellite/',Obstcl,'/Sensor/Bound',Obstcl,' Range Max
',num2str(OBS.DoL)]);
stkExec(conID,['Location
*/Satellite/',Obstcl,'/Sensor/Bound',Obstcl,' Fixed Cartesian 0 0
0']);
stkExec(conID,['VO */Satellite/',Obstcl,'/Sensor/Bound',Obstcl,'
Projection SpaceProjection ',num2str(OBS.DoL)]);
stkExec(conID,['Graphics
*/Satellite/',Obstcl,'/Sensor/Bound',Obstcl,' SetColor red']);
stkExec(conID,['VO */Satellite/',Obstcl,'/Sensor/Bound',Obstcl,'
Translucency 95']);
stkExec(conID,['VO */Satellite/',Obstcl,'/Sensor/Bound',Obstcl,'
TranslucentLines On']);
```

The spherical sensor projections were made translucent in order not to obscure the view of maneuvering spacecraft. A sample spherical boundary sensor projection about a spherical object is shown in Figure 10.6. The sensor projection is in translucent red with the Chase spacecraft in the upper right. These sensors allowed for robust engineering evaluation of collision avoidance function of spacecraft control algorithms. From these basic examples, the user can develop a vast array of representative sensors. Sensor projections ranges and colors can be varied based on logical condition. For instance, ranging sensors may be programmed to modify their color as the distance to a target varies. This may allow for quick visual analysis of multiple spacecraft maneuver performance. However, users may also want to terminate some projections during presentations, since multiple overlapping sensors may be visually confusing to some audiences. Ultimately, the animation graphic and level of detail can be tailored from the core MATLAB-STK simulation interface code described in this research.

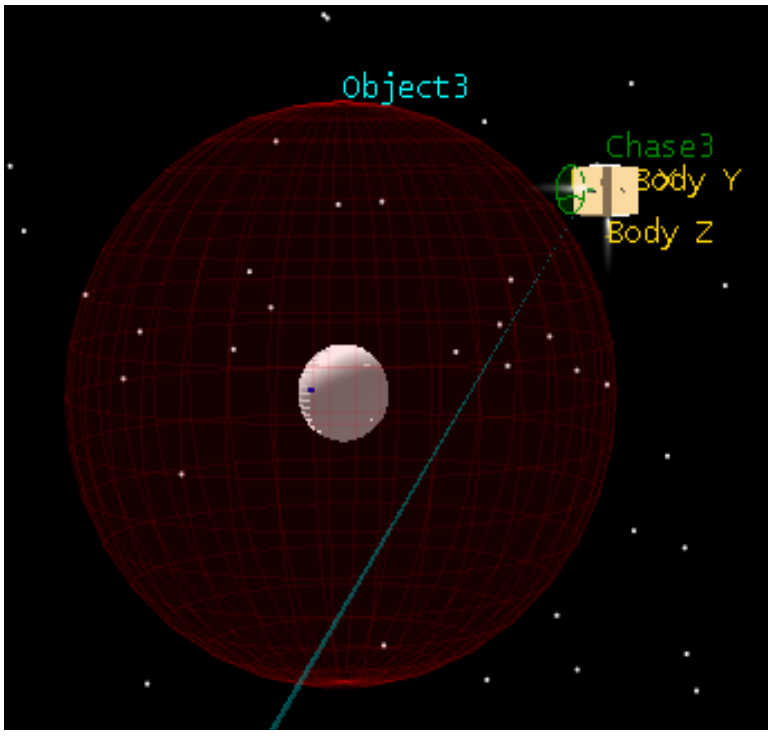

Figure 10.6 Sample Spherical Boundary Sensor.

2. Formatting MATLAB Data for STK Files

The visualization and animation of the MATLAB/Simulink data with the desired spacecraft model is a useful tool for engineering analysis and evaluation. In particular, it was vital in the development of a multiple spacecraft control algorithm during simultaneous close proximity operations [63][64]. The 3D representation for spacecraft during collision avoidance maneuvers enabled effective troubleshooting. For instance, the undesired clipping of the corners of Chase spacecraft while converging to the Target spacecraft during docking maneuvers was easily shown in STK animation. Due to this evaluation, the control algorithm collision avoidance logic was modified and its performance robustness was improved. Additionally, visualization of the simultaneous multiple spacecraft maneuvers in STK allows for logical point of view changes and animation rate changes. These capabilities allow for clear presentation of spacecraft dynamics and control results to those interested multiple spacecraft operations. In fact, STK scenarios can be easily edited and transferred into video format for presentations. Modifications of camera key frames (camera position and angle) and animation rates allow versatile video composition. The video file's format, encoding, bit rate, and size can all be selected in STK 7.1, or later versions [36]. The basic STK scenario can be modified and animated to meet the individual user's requirements. Ultimately, the same STK scenario used for engineering evaluation can serve as a demonstration of engineering capabilities to a wider audience.

a. Ephemeris and Attitude Data

The animation of the spacecraft can be based on the data generated by STK or the MATLAB/Simulink model. STK generated data is already in the proper format. However, MATLAB/Simulink data must be proper formatted in order to be used by STK. STK allows for several specific data file formats, such as STK ephemeris and attitude files. These files can be properly written by executing short MATLAB scripts utilizing `fprintf` commands. A sample STK ephemeris file can be created in MATLAB as follows:

```
filename='C:\Userfiles\STK\Scenario\ephemeris_t.e';
ephemeris=fopen(filename,'w');
fprintf(ephemeris,'stk.v.4.3\n\n');
fprintf(ephemeris,'BEGIN Ephemeris\n\n');
fprintf(ephemeris,'NumberOfEphemerisPoints\t
%6.0f\n',length(STATE.time));
fprintf(ephemeris,'ScenarioEpoch\t %s\n',num2str(para_now));
fprintf(ephemeris,'InterpolationMethod\t Lagrange\n');
fprintf(ephemeris,'InterpolationOrder\t 5\n');
fprintf(ephemeris,'CentralBody\t Earth\n');
fprintf(ephemeris,'CoordinateSystem\t J2000\n');
fprintf(ephemeris,'EphemerisTimePosVel\n\n');
data=[STATE.time(:,1),r(:,ici+1:ici+3),v(:,ici+1:ici+3)]';
fprintf(ephemeris,'%16.14e\t %16.14e\t %16.14e\t %16.14e\t %16.14e\t
%16.14e\t %16.14e\n',data);
fprintf(ephemeris,'\n END Ephemeris\n');
fclose(ephemeris);
```

The STK ephemeris file name and path, in the first line, are arbitrary, as long as it has the proper `.e` suffix. Although, it is useful to save all related STK scenario files in the same folder. The first few fprintf lines are used to establish the desired reference frame and interpolation for the data. The variables `STATE.time` and `para_now` are used to synchronize the data points with the time constraints, as discussed in Chapter I. C. 2. The data is formatted into seven columns, represented by time, position vector, and velocity vector. Each row represents an iteration, or stepsize, of the data. Similarly, a STK quaternion attitude file can be created as follows:

```
filename2='C:\Userfiles\STK\Scenario\attitude_t.a';
attitude=fopen(filename2,'w');
fprintf(attitude,'stk.v.5.0\n\n');
fprintf(attitude,'BEGIN Attitude\n\n');
fprintf(attitude,'NumberOfAttitudePoints\t
%6.0f\n',length(STATE.time));
fprintf(attitude,'ScenarioEpoch\t %s\n',num2str(para_now));
fprintf(attitude,'BlockingFactor\t 20\n');
fprintf(attitude,'InterpolationOrder\t 5\n');
fprintf(attitude,'CentralBody\t Earth\n');
fprintf(attitude,'CoordinateAxes\t J2000\n');
fprintf(attitude,'AttitudeTimeQuaternions\n\n');
data=[STATE.time(:,1),Qbn(:,iqi+1:iqi+4)]';
fprintf(attitude,'%16.14e\t %16.14e\t %16.14e\t %16.14e\t
%16.14e\n',data);
fprintf(attitude,'\n END Attitude\n');
fclose(attitude);
```

The STK attitude file name and path are arbitrary, as long as it has the proper *.a* suffix. This sample STK attitude file is formatted into five columns with time, the three element quaternion vector, and the quaternion scalar (rotational) term.

b. STK Model Articulation Files

The articulation of a STK spacecraft graphic model component is executed by padding formatted data to the STK scenario. The articulation file must have the *.sama* suffix, be named the same as the spacecraft object, such as `Target`, and saved in the same folder as the STK graphical model, such as the `mdl_cube1.mdl`. A sample articulation file for each Thrust_Flame component is as follows:

```
filename3=['C: \Userfiles\STK\Scenario\Target.sama'];
articulate=fopen(filename3,'wt');
fprintf(articulate,'STARTTIME\t 0\n\n');
fprintf(articulate,'DURATION\t %6.0f\n',stoptime);
fprintf(articulate,'DEADBANDDURATION\t 0.0\n');
fprintf(articulate,'ACCELDURATION\t 0.0\n');
fprintf(articulate,'DECELDURATION\t 0.0\n');
fprintf(articulate,'DUTYCYCLEDELTA\t 0.0\n');
fprintf(articulate,'PERIOD\t 0.0\n');
fprintf(articulate,'ARTICULATION\t Thrust_Flame#\n');
fprintf(articulate,'TRANSFORMATION\t Size\n');
fprintf(articulate,'STARTVALUE\t 0\n');
fprintf(articulate,'ENDVALUE\t 1\n');
fclose(articulate);
```

where `Thrust_Flame#` is the name of the component being articulated and `Size` is the magnitude of the articulation desired. This strict formatting is limiting due to the start and end values of each articulation iteration. A more useful presentation, defines the articulation in a spreadsheet which loops through the time increments, `ti`, and assigns the desired articulation magnitude for each thruster, `tii`, on the spacecraft. Sample articulation code is as follows:

```
for ti=1:length(time)
   for tii=1:6
   filename3=['C: \Userfiles\STK\Scenario\Target.sama'];
   articulate=fopen(filename3,'wt');
   fprintf(articulate,'SPREADSHEET\n\n');
   fprintf(articulate,['ARTICULATION,',num2str((ti-1)*stepsize),',',
   num2str(stepsize),',0,0,0,0,0,Thrust_Flame',num2str(tii),',Size,',
   num2str(tmag(ti,tii)),',',num2str(tmag(ti+1,tii)),'\n']);
   end
end
fclose(articulate);
```

The spreadsheet data may be based on desired thruster firing magnitude at each time step of the simulation. This data may be generated from the MATLAB/Simulink spacecraft model's control algorithm response. The desired control response can be mapped to the appropriate thruster. This mapped response can be assigned into the articulation file to animate each spacecraft's thrusters firing during the multiple spacecraft maneuver.

It may be useful to include a reload command at the end of the articulation file assignment to ensure that all components have the current data. This reload command is of the form:

```
stkExec(conID,'VO */Satellite/Target ReloadArticFile');
```

The articulation of multiple components can be useful in generating complex spacecraft models. Articulating components may include such components as thrusters, solar panels, and momentum exchange devices. Although visually stimulating, it may be more practical to only articulate the necessary components for each engineering application. The articulation sections may be commented out when not necessary. Once performance is complete, the articulation may be reinstated for presentations.

D. MATLAB-STK CONCLUSION

A MATLAB-STK simulation interface was developed for spacecraft model validation and visualization. Both MATLAB/Simulink and STK have the capability to perform high precision orbital propagation. Utilizing this capability, the spacecraft model validation is conducted by comparing various spacecraft state date, such as position and velocity. Each modular perturbation component can be independently enabled for evaluation. The desired level of model similarity and accuracy can be achieved by matching all parameters that are being considered by the designer. Once the model is validated, the spacecraft can be animated by passing spacecraft state data to STK. The spacecraft characteristics and parameters, defined in MATLAB/Simulink, can be formatted and passed into STK. Utilizing STK via MATLAB is a versatile and effective method of animating six DOF spacecraft models. The resulting STK animation of spacecraft propagations is useful for engineering evaluation and result presentations. This interface was utilized during the development of a multiple spacecraft control

algorithm for close proximity operations. Model validation gave confidence in the performance results and visualization allowed for straightforward evaluation. The animation allowed for immediate identification of undesirable performance such that modifications and improvements could be made to the control algorithm. This MATLAB-STK simulation interface proved to be a useful tool for the development and evaluation of spacecraft models and control systems.

XI. VHIL EVALUATION

The challenging characteristics of microgravity orbital space make it difficult to simulate in the laboratory environment. So, a series of test methods exist for incremental evaluation of the control algorithm. First, the control algorithm processing can be implemented on distributed independent hardware. This VHIL testbed represents a set of simulated spacecraft which conduct the control algorithm based on their local state information. Next, a terrestrial based spacecraft robot testbed can be utilized. These 3D spacecraft robots can actual physically actuate and perform limited two-axis translational and one axis rotational maneuvers. Finally, the control algorithm can be loaded onto actual free flying spacecraft in the orbital environment.

A. VIRTUAL HARDWARE-IN-THE LOOP (VHIL) TESTBED

Hardware testing for spacecraft related components and systems are restricted due to high costs and environmental limitations. In order to perform incremental testing of the developed control algorithm, a VHIL testbed was developed at NPS. This test bed consists of a distributed set of independent processors linked in a virtual environment. The block diagram for the VHIL testbed is shown in Figure 11.1.

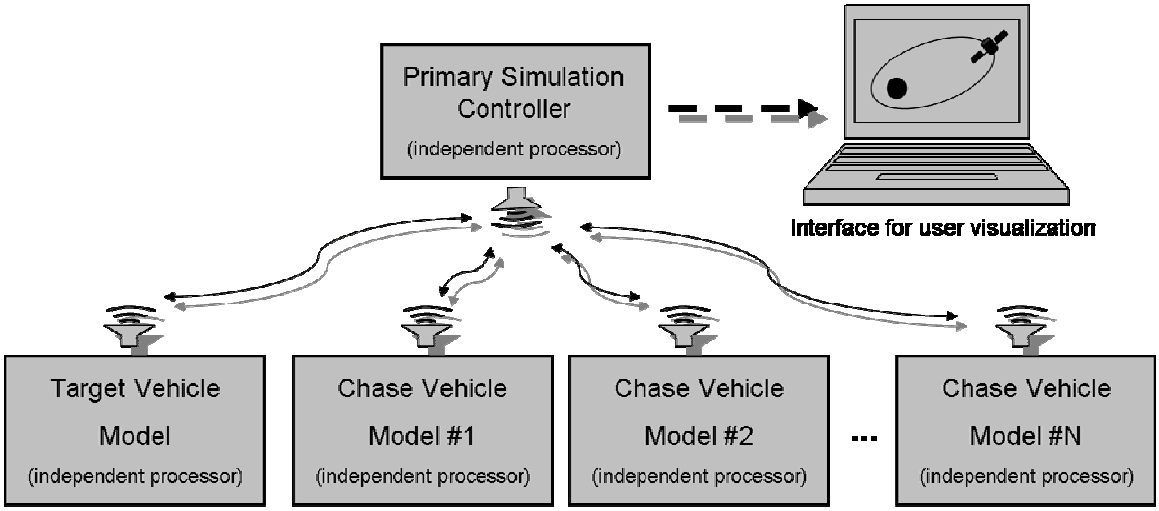

Figure 11.1 VHIL Block Diagram.

Each processor represents a stand alone Chase spacecraft which independently propagates its maneuver ephemeris based on a validated spacecraft model. Any external data concerning the position and velocity of other spacecraft is passed via UDP. This data simulates possible sensor derived information. A desktop computer may serve as the Target spacecraft. This desktop may also provide processing of the STK visualization for the multiple spacecraft maneuvers. The same software used in the original development was used in the distributed control algorithm, primarily MATLAB version 7.3.0267 (R2006b) and STK version 7.0.1 [36][39]. All MATLAB spacecraft dynamics and control functions were converted into embedded MATLAB functions for efficient C code programming language. The virtual spacecraft were represented by small mobile computers: refer to Table 11.1 for details. The multiple spacecraft VHIL testbed is shown in Figure 11.2. This image shows four distributed processors wirelessly connected, with a shared monitor, keyboard and mouse.

Characteristics	Name	LG-P625F
	Processor	Intel Pentium M 1.7 GHz
	Hard drive	100 GB 7200 RPM
	Memory	1 GB DDR400
	DC Power	12 V
Size	Width	0.155 m
	Depth	0.255 m
	Height	0.055 m
	Mass	< 3.0 kg
Features	Thermal Control Technology	Fan less
	Built-in Wireless	802.11B/G
	Slot Loading Disc Drive	CDRW with DVD
	Interfaces	USB 2.0, Firewire
Operating Systems	Microsoft Windows	XP (80 GB)
	Linux	Gentoo (20 GB)

Table 11.1 Virtual Spacecraft Computers.

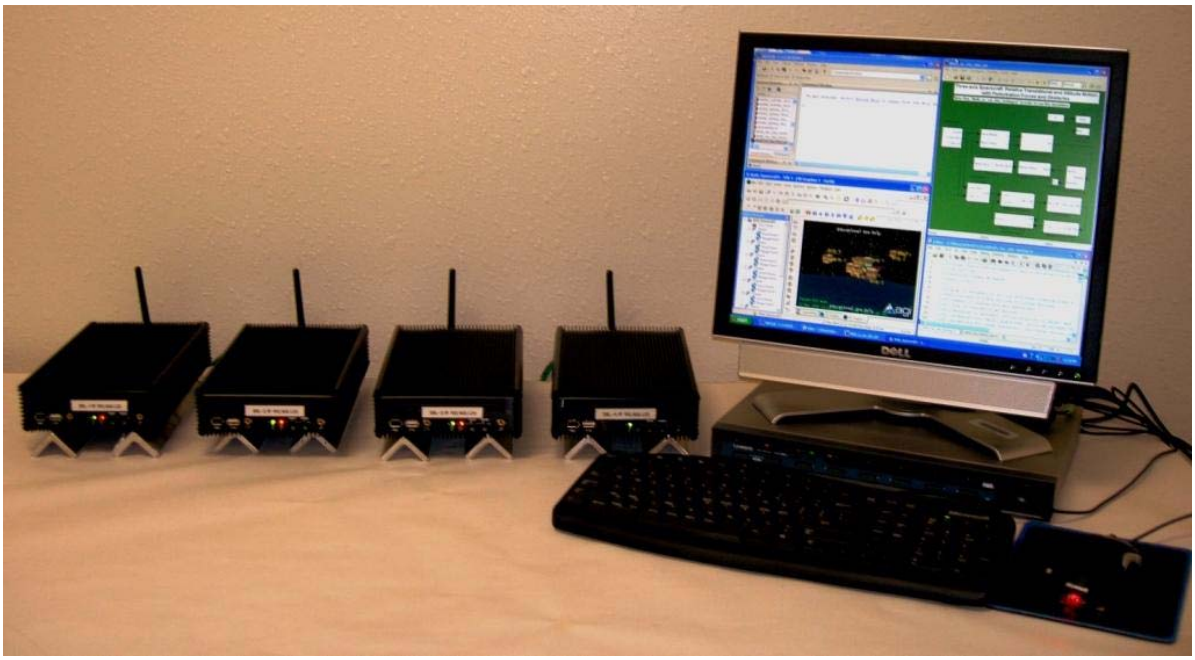

Figure 11.2 Multiple Spacecraft Testbed.

The VHIL distribution of the control algorithm ensures that limited state information and reasonable data rates are being utilized. Initial assessments of processor, sensor, and timing limitations can be conducted. Each processor independently simulates the spacecraft environment. Each virtual spacecraft processing rate can be roughly synchronized by using Simulink Simulation Pace blocks to standardize the ratio of

simulation seconds to internal clock seconds. This serves as a versatile soft-real time technique for implementing the VHIL via Simulink [36].

The control algorithm executes based on the autonomous spacecraft's simulated sensor information. This data can be passed directly between Chase processors or routed via the Target processor. Drop out of one or more Chase spacecraft processors do not negatively affect the VHIL simulation of other spacecraft. UDP communication default simply holds the last value successfully received [36]. Initial VHIL testing focused on control algorithm development and not sensor or communication link limitations. Obviously, obstacle and other Chase spacecraft states must be know at some level in order permit collision avoidance. The distributed state data for each Chase spacecraft is only centralized at the conclusion of simulations for the purpose of evaluation and visualization. The VHIL testing paves the way for additional testing on terrestrial and orbiting testbeds.

B. AMPHIS TESTBED

In order to fully develop the concepts for close proximity operations of multiple spacecraft through hardware-in-the-loop testing, it is necessary to have sufficient space for maneuvering of the testbed vehicles. A Proximity Operations Simulator Facility was been previously developed [79][80] and will be used for any new systems. This facility consists of a 4.9 m by 4.3 m wide Epoxy Floor Surface, approximately 21 square meters, used as base for the floatation of spacecraft simulators. The use of air pads on the simulators reduces the friction to negligible level. Due to an average residual slope angle of $\sim 2.6 \cdot 10^{-3}$ deg for the floating surface, the average value of the residual gravity acceleration affecting the dynamics of floating vehicles is $\sim 1.4 \cdot 10^{-3}$ $m * s^{-2}$. This value of acceleration, measured by analyzing the free motion of the Chaser spacecraft simulation, is two orders of magnitude lower than the nominal amplitude of the acceleration fluctuation obtained during the reduced gravity phases of parabolic flights [81].

Given the dimensional constraints of the flat epoxy floor, in order to simultaneously operate at least three robotic spacecraft a significantly lighter and smaller vehicle is required, as compared to the previous AUDUSS spacecraft robot simulator [79]

[80]. Additionally, the NPS SRL research and testing of multiple sensor packages requires multiple spacecraft robots. The new generation of robotic spacecraft at the SRL is referred to as AMPHIS, as shown in Figure 11.3.

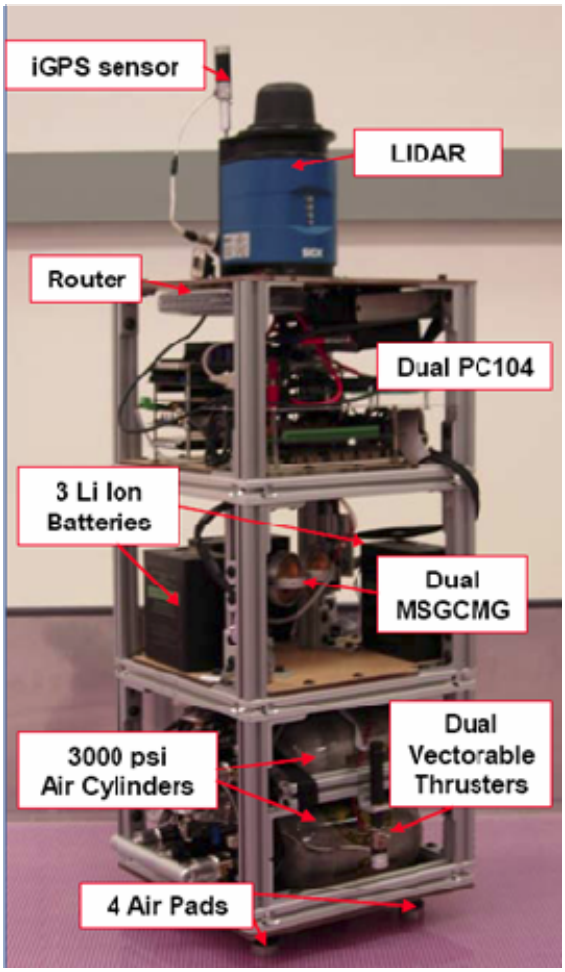

Figure 11.3 AMPHIS Robotic Spacecraft Simulator at NPS [82].

Recent enhancements in the AMPHIS robotic spacecraft simulator enabled a significant reduction in size and mass while providing for easy reconfiguration [82][83]; refer to Table 11.2. The details on the AMPHIS robot are presented completeness. The AMPHIS is continually evolving as research is conducted at NPS.

Size	Length and Width	0.30 m
	Height	0.69 m
	Mass	37 kg
	Moment of Inertia z_{veh}	0.75 kg m^2
Propulsion	Propellant	Air
	Equiv Storage Cap @ 21 MPa (3000 PSI)	0.002 m^3
	Operating Pressure	0.827 MPa
	Thrust per Thruster	0.28 N
Attitude Control	MSGCMG Max Torque	0.668 Nm
	MSGCMG Max Angular Momentum	0.098 Nms
Electrical & Electronic	Battery Type	Lithium-Ion
	Storage Cap @ 28 V	12 Ah
	Computers	2 PC-104, Pentium III
Sensors	Fiber Optic Gyro Bias	±20º/hr
	LIDAR	SICK 360 º
	iGPS Sensor Accuracy	<.050 mm
	Accelerometers Bias	$\pm 8.5 \times 10^{-3}$ g
Floatation	Propellant	Air
	Equiv Storage Cap @ 21 MPA (3000 PSI)	.002 m^3
	Operating Pressure	0.35 MPa
	Linear Air Bearings Diameter	32 mm

Table 11.2 Key Parameters on the AMPHIS Testbed [82][83].

It is worth mentioning that other Hardware-in-the-Loop testbeds are being developed world wide. All terrestrial spacecraft testbed has limitations due to the need to imitate the orbital environment. The concurrent development of terrestrial testbeds may allow for comparison of algorithm performance results. Besides the NPS SRL, another promising satellite formation flying testbed is located at the University of Southampton, in the United Kingdom [84]. This facility enables control algorithms to be tested on 5 DOF mechanical mock-up satellite frames, referred to as ground-based satellite frame testing (GSFT) [84]. The control algorithm testing relies on simulated sensor, actuator, and physical dynamics. The combination of these virtual components and the GSFT overlaps for systematic spacecraft environmental simulations. The accuracy of the testbed currently aims at centimeter accuracy, but may be improved for higher precision [84].

C. SPHERES TESTBED

On-orbit testing is a tremendous opportunity for development a multiple spacecraft control algorithm. On board the ISS, testing of the multiple spacecraft close proximity control algorithm is now possible with the MIT Space Systems Laboratory's SPHERES [25] [26]. The SPHERES are intended to be used as a test bed for formation flight and reconfiguration, as well as autonomous rendezvous and docking technologies [26]. The first SPHERES satellite reached the ISS in May 2006 and has begun testing scenarios. There are currently three SPHERES available for ISS investigation sessions.

Before on-orbit testing can be conducted, software modification and synchronization would have to be conducted. This would include ground testing with MIT Space Systems Laboratory's team. Funding and co-operative support for this work has been discussed. DARPA, NASA, MIT, NPS and DoD participants have all shown interest. Scheduling and timeline concerns have limited the inclusion of this testing in this current body of research.

THIS PAGE INTENTIONALLY LEFT BLANK

XII. CONCLUSIONS AND FUTURE WORK

A. CONCLUSIONS

The developed LQR/APF multiple spacecraft proximity control algorithm offers desirable performance and establishes a baseline for fuel efficiency while maintaining collision free operations. Evaluation of the LQR/APF control algorithm proved it to be effective in a broad range of multiple spacecraft close proximity maneuvers, including rendezvous and docking maneuvers. In the presence of obstacles and other maneuvering spacecraft the LQR/APF exhibited a smooth and effective control response which avoided actuator saturation. Therefore, the LQR/APF control algorithm shows efficient control response with reliable collision avoidance

Furthermore, Monte-Carlo method analysis of close proximity maneuvers confirms that the LQR/APF control algorithm is a practical candidate for multiple spacecraft close proximity operations. Estimates of the mean and standard deviation of maneuver t_d and Δv show that average proximity maneuver LQR/APF control Δv efficiency is better than that of a highly tuned APF control algorithm. The LQR/APF showed a distinct control efficiency improvement on a *per* spacecraft maneuver basis. The standard deviation of the LQR/APF control effort is consistently narrower then that of the APF controller. This narrow Δv standard deviation is valuable to both spacecraft designers and mission planners. It allows effective propellant sizing for close proximity operations. It also gives operational planners a useful tool for developing and forecasting maneuvers.

In conjunction with the controller research, a MATLAB-STK simulation interface was developed for multiple spacecraft model validation and dynamic environment visualization. This MATLAB-STK simulation interface was extensively utilized during the LQR/APF control algorithm refinement development. Model validation gave confidence in performance results and visualization allowed for straightforward evaluation. The STK animation allowed for immediate identification of undesirable performance such that modifications and improvements could be made to the control algorithm. This MATLAB-STK simulation interface is a useful tool for the development

and evaluation of spacecraft models and control systems. The STK animation allows for clear presentation of developed control algorithm performance to the spacecraft field.

Finally, the LQR/APF control algorithm was implemented and evaluated in a VHIL configuration. The VHIL utilizes independent processors which simulate spacecraft and interact as multiple spacecraft. VHIL control algorithm test results were consistent with the LQR/APF performance during the evaluation and analysis presented in this research. Successful VHIL experimentation paves the way for future hardware implementation on both terrestrial based and orbital spacecraft robot testbeds.

B. FUTURE WORK

Further investigation could establish the control algorithm robustness by conducting additional Monte-Carlo simulations including a higher number of spacecraft, heterogeneous spacecraft, noisy sensor, and more restrictive actuator limitations and failures. These studies could address the potential scope of application for this control algorithm. Elliptical orbital considerations could be included into the dynamics [43] [42] [44]. Specific on-orbit applications need to take into account the actual spacecraft sensor and actuators in order to determine if the control algorithm's performance meets mission requirements.

Refinement of the APF control algorithm could be carried out based on a more sophisticated Newton method or conjugate gradient search algorithms. The Steepest Descent algorithm initially adopted is based on the system measurement and control rates being nearly the same. However, the control effort may need to be two or three times faster then the measurement cycle. In this case, the quadratic characteristics of the potential functions should allow the faster control rate to achieve more efficient iterative minimization. In particular the conjugate gradient Fletcher Reeves iteration scheme efficiency is due to varying the step magnitude and direction variation while convergence is achieved in two iterations. Refer to Appendix A for alternate search algorithm discussion.

Refinement of the LQR control algorithm to include direct incorporation of collision avoidance terms into the state and control effort gain matrices could be

investigated. If the cost function can be minimized based on collision avoidance, then it should be possible to include additional parameters into the LQR gains which are related to collision avoidance. The desire would be to add collision avoidance gains without adding additional states. The selection of these gains needs to maintain a geometric relationship to collision avoidance, while preserving the desired system response. Preliminary success was achieved by incorporating obstacle relative velocity components into the LQR state gain matrix. In general, collision avoidance was achieved; however the tuning of these parameters proved to be very sensitive as the states approached the goal. Therefore, docking precision seen by the hybrid algorithm could not be achieved while maintaining collision avoidance. More rigorous study of the gain parameters need to be conducted before confidence can be given to this incorporated LQR collision avoidance method. Refer to Appendix B for preliminary LQR with collision avoidance approach and results.

Specific actuator limitations in both the translational and attitude control could be further investigated. Evaluating controller algorithm performance with limited numbers or capabilities in actuation may offer interesting practical maneuver limitations. For instance two maneuverable thrusters may be able to perform all desired maneuvers. However, specific types of thrusters may be constrained due to pluming regions around sensitive payloads. Additionally, the coupling of translational and attitude control via thruster actuation is an interesting development.

In this research, initial goal locations and docking ports were commanded, however future work may allows for an outer control loop to determine when and where each spacecraft is tasked. Higher level control can be implemented based on neural networking or dynamic programming multi-vehicle task assignment algorithms [85][86] [1]. However, these algorithms may require additional communication between spacecraft for efficient tasking decisions, or voting, to be accomplished. This communication requirement may lead to more research into wireless sensor networking of multiple spacecraft. In such a networking, micro chips with control with modem capabilities, such as Maxstream's small Xbee chips [87], can be used for communication and can also support ranging data. The wireless signal has signal strength data, referred to as Received Signal Strength Indicator (RSSI) that can be combined with ranging

sensors, such as GPS. The signal strength is determined from the decibel magnitude of the last packet. With directional antenna, the decibel reading can be directly transferred into ranging data. This additional information can improve range precision and supplement when other sensor data. An example of wireless mesh networking on autonomous vehicles using the Xbee chip is discussed in detail by Bledsoe in [88]. This application can be related to the communication of multiples spacecraft robots on a terrestrial testbed during proximity operations.

The multiple spacecraft control algorithm developed in this research showed robust disturbance rejection based on deterministic measurements. However, comprehensive performance evaluation including process and sensor noise characteristics for specific spacecraft configurations may be desired. Any known sensor statistics can be used to re-characterize the stochastic process. This uncertainty can be included into the control algorithm by modifying the LQR as a LQG regulator, which can incorporate process and measurement noise as Gaussian white noises with covariance. Also the use of Gaussian based collision avoidance lends itself to direct inclusion of any measurement noise in to the obstacle relative range and velocity. The algorithm structure allows for convenient inclusion of known or estimated sensor uncertainties. These uncertainties may be particularly suitable to the estimation with Kalman Filtering techniques suggested by Cristi *et al* [89]. Large control iteration time steps, of one second, were maintained to allow for fusing or filtering of sensor measurements. Even with noisy state measurements the probable shortening of control step iterations will allow for acceptable algorithm performance. Although, detailed control algorithm performance evaluation based on varying levels of filtered, or individual, sensor noise would offer confidence in practical and valid safety margins.

Full evaluation of docking mechanisms and their related performance requirements could be undertaken. Various potential docking mechanisms could be incorporated in evaluation of the control algorithm [80][81]. For instance, spatial and attitude tolerances may allow for less rigorous terminal control. Also passive docking mechanism may actually require Chase spacecraft to engage the Target spacecraft with some terminal force. These latch mechanism are envisioned for on-orbit assembly, and not for autonomous spacecraft with sensitive payloads [81]. However, even simple

adhesive docking ports would require some terminal contact force. The trade-off between a firm dock and the jarring of the spacecraft could be researched.

Most importantly, the close proximity control algorithm can be integrated onto spacecraft robots with different vehicle and sensor properties. Ground testing could include integration onto the NPS AMPHIS in the Spacecraft Servicing and Robotics Laboratory. The laboratory hosts the Autonomous Docking and Spacecraft Servicing testbed, which may allow for future testing of on-the-ground dynamics with close proximity operations. The AMPHIS is built upon previous research on the original AUDUSS and the AUDUSS II [79] [80]. Additionally, ground testing may also be conducted at MIT's Space System Laboratory as preparation for on-orbit flight testing of the control algorithm on the SPHERES onboard the ISS [25] [26]**.** The SPHERES testbed enables maturation and validation of the control algorithm in the micro gravity environment of the ISS as a risk reduction before integration on future spacecraft. This will require incorporation of the SPHERES system details into the control algorithm with baseline simulations and ground testing.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A: OPTIMAL SEARCH ALGORITHMS

As discussed during the APF development, previous APF control algorithms have used the immediate Steepest Descent approach to navigation. This is a reasonable assumption as to the best course for the Chase spacecraft to proceed. However, the implementation of iterative searches in spacecraft dynamics yields only a momentary current steep gradient. This direction of approach may not be the most efficient path. There are alternate ways for selecting both the search direction and length of the step size. In particular, Newton's method and Conjugate Gradient optimal search algorithms appear promising for selection of the search direction and step size varies for Spacecraft APF based control algorithms However, in order to utilize these search methods the spacecraft's sensor update rates may need to be considered in the controls systems iterative convergence algorithm. While the Steepest Descent method has proved extremely effective in this research, it may not be the best for all multiple vehicle control algorithms. Consideration and comparison of other search techniques, such as Newton's method and Conjugate Gradient, should be made before a final determination can be made.

A. STEEPEST DESCENT

Whether a potential or cost function is used as the guiding parameter for the control algorithm, some discussion of optimal search is warranted. Broadly speaking, search procedures for optimization problems may be divided into three main categories: calculus-based, enumerative, and heuristic. Calculus-based procedures use either analytical or numeric models of the solution space as the basis for the search. Enumerative or "brute force" procedures search systematically and do not incorporate any sophisticated methods to reduce the problem space or refine proposed solutions. Heuristic procedures attempt to improve on the search efficiency of enumerative methods without direct incorporation of models of the solution space, which are often unavailable.

Performance surfaces are functions which are designed such that the quantitative value is smallest when the desired performance is obtained. The global minimum point is

considered the equilibrium or optimal point [85]. The path which the system searches for and approaches this minimum point is dependent on the optimization algorithm used. The iterative selection of the search direction and step size varies for each type of optimization algorithms. While a wide breadth of search algorithms exists, those most applicable to this work are the Steepest Descent, Newton's method, and conjugate gradient. Given its efficiency in implementation, the Steepest Descent method was employed in this research.

For a performance function, $V(x)$, the general iterative search algorithm is of the form

$$V(\vec{x}_{k+1}) = V(\vec{x}_k + \Delta\vec{x}_k) \tag{A.1}$$

where x_k is the current state, x_{k+1} is the next state, and Δx_k is the change in states from one iteration to the next. The change in states is described as

$$\Delta\vec{x}_k = \vec{x}_{k+1} - \vec{x}_k = \lambda_k \, \vec{p}_k \tag{A.2}$$

where the vector $\vec{p}_k$ is the search direction and the scalar λ_k is the step size. Equation (A.2) can be rearranged to for iteration to give the next state as

$$\vec{x}_{k+1} = \vec{x}_k + \lambda_k \, \vec{p}_k \tag{A.3}$$

In order for search algorithm to converge, the next value of the performance function must be less than the current for each iteration, such as

$$V(\vec{x}_{k+1}) < V(\vec{x}_k) \quad for \quad \vec{x}_{k+1} \neq \vec{x}_k \tag{A.4}$$

with the search direction vector, $\vec{p}_k$, being less than zero and the step size being small, but greater than zero [85]**.**

There are various ways for selecting both the search direction and the step size. The Steepest Descent method searches in the direction of the largest negative rate of change for the performance function. The step size for each iteration can be either fixed or variable. The performance function is assumed to be an analytic function, such that its derivatives exist. This is logical since our performance functions are based on position

states, so the first derivative is velocity, and the second derivative is acceleration. Any performance function could be represented by a Taylor Series expansion, such as

$$V(x) = V(x^*) + \frac{d}{dx}V(x)\bigg|_{x=x^*}(x - x^*) + \left(\frac{1}{2}\right)\frac{d^2}{dx^2}V(x)\bigg|_{x=x^*}(x - x^*)^2 + \ldots \tag{A.5}$$

where the state x is a scalar and the function is expanded about some nominal point, x^*. For the neighborhood of the nominal point, the higher order terms can be dropped for a quadratic estimate of the function about that point. Since the states are most likely a vector, such as a position vector, the Taylor series can be expanded into a vector notation with $V(\vec{x}) = V(x_1, x_2, \ldots x_n)$, as

$$\begin{aligned} V(\vec{x}) = V(\vec{x}^*) &+ \frac{\partial}{\partial x_1}V(\vec{x})\bigg|_{x=x^*}(x_1 - x_1^*) + \frac{\partial}{\partial x_2}V(\vec{x})\bigg|_{x=x^*}(x_2 - x_2^*) + \ldots \\ &+ \frac{\partial}{\partial x_n}V(\vec{x})\bigg|_{x=x^*}(x_n - x_n^*) + \left(\frac{1}{2}\right)\frac{\partial^2}{\partial x_1^2}V(\vec{x})\bigg|_{x=x^*}(x_1 - x_1^*)^2 + \\ &\left(\frac{1}{2}\right)*\frac{\partial^2}{\partial x_2^2}V(\vec{x})\bigg|_{x=x^*}(x_2 - x_2^*)^2 + \ldots + \left(\frac{1}{2}\right)\frac{\partial^2}{\partial x_n^2}V(\vec{x})\bigg|_{x=x^*}(x_n - x_n^*)^2 + \ldots \end{aligned} \tag{A.6}$$

Since this format is cumbersome, it is more convenient to write in matrix form

$$V(\vec{x}) = V(\vec{x}^*) + \nabla V(\vec{x})^T\big|_{x=x^*}(\vec{x} - \vec{x}^*) + \left(\frac{1}{2}\right)(\vec{x} - \vec{x}^*)^T \nabla V^2(\vec{x})\big|_{\vec{x}=\vec{x}^*}(\vec{x} - \vec{x}^*) + \ldots \tag{A.7}$$

where ∇V is the gradient of the potential function, and is defined as

$$\nabla V(\vec{x}) = \left[\frac{\partial}{\partial x_1}V(\vec{x}), \frac{\partial}{\partial x_2}V(\vec{x}), \ldots, \frac{\partial}{\partial x_n}V(\vec{x})\right]^T \tag{A.8}$$

and ∇V^2 is the Hessian, and is defined as

$$\nabla V^2(\vec{x}) = \begin{bmatrix} \frac{\partial^2}{\partial x_1^{\,2}} V(\vec{x}) & \frac{\partial^2}{\partial x_1 \partial x_2} V(\vec{x}) \ldots & \frac{\partial^2}{\partial x_1 \partial x_n} V(\vec{x}) \\ \frac{\partial^2}{\partial x_2 \partial x_1} V(\vec{x}) & \frac{\partial^2}{\partial x_2^{\,2}} V(\vec{x}) \ldots & \frac{\partial^2}{\partial x_2 \partial x_n} V(\vec{x}) \\ \vdots & \vdots & \vdots \\ \frac{\partial^2}{\partial x_n \partial x_1} V(\vec{x}) & \frac{\partial^2}{\partial x_n \partial x_2} V(\vec{x}) \ldots & \frac{\partial^2}{\partial x_n^{\,2}} V(\vec{x}) \end{bmatrix} \tag{A.9}$$

The Hessian is the Jacobian of the gradient, where the Jacobian of a function, $f(\vec{x})$, is defined as a matrix of its first partial derivatives

$$J(\vec{x}) = \begin{bmatrix} \frac{\partial}{\partial x_1} f_1(\vec{x}) & \frac{\partial}{\partial x_2} f_1(\vec{x}) \ldots & \frac{\partial}{\partial x_n} f_1(\vec{x}) \\ \frac{\partial}{\partial x_1} f_2(\vec{x}) & \frac{\partial}{\partial x_2} f_2(\vec{x}) \ldots & \frac{\partial}{\partial x_n} f_2(\vec{x}) \\ \vdots & \vdots & \vdots \\ \frac{\partial}{\partial x_1} f_m(\vec{x}) & \frac{\partial}{\partial x_2} f_m(\vec{x}) \ldots & \frac{\partial}{\partial x_n} f_m(\vec{x}) \end{bmatrix} \tag{A.10}$$

The gradient and the Hessian can give valuable information about the performance surface, which may also be referred to as the potential field. The gradient is the direction of maximum slope from the current position. On the performance surface, any direction which is orthogonal to the gradient will have zero slope [85].

The baseline quadratic performance function is of the form

$$V(\vec{x}) = \frac{1}{2}(\vec{x})^T \mathrm{A}\,\vec{x} + d^T \vec{x} + c \tag{A.11}$$

where $\vec{x}$ are the states of the function, A is a scaling matrix which relates to the shape of the contour ellipses, d is a scaling factor which determines where equilibrium point is located, and c is a scaling constant which changes the value of the performance function at the equilibrium point. The gradient and Hessian of a generic quadratic function are

$$\nabla V(\vec{x}) = \mathrm{A}\,\vec{x} + d \tag{A.12}$$

$$\nabla V^2(\vec{x}) = \mathrm{A} \tag{A.13}$$

The equilibrium point is generally selected to be at the origin ($d = 0$) with a minimum value equal to zero ($c = 0$), so the quadratic function simplifies as

$$V(\vec{x}) = \frac{1}{2}(\vec{x})^T \, \mathrm{A} \, \vec{x} \tag{A.14}$$

In this case, the gradient and Hessian of the simple quadratic become

$$\nabla V(\vec{x}) = \mathrm{A} \, \vec{x} \tag{A.15}$$

$$\nabla V^2(\vec{x}) = \mathrm{A} \tag{A.16}$$

Notice that both the gradient and the Hessian depend on the performance surface contour shaping matrix, A.

$$\vec{x}_{k+1} = \vec{x}_k - \lambda_k \, \mathrm{A} \, \vec{x} \tag{A.17}$$

Using this information about the gradient and substituting into the general performance search algorithm in Equation (A.3), yields the Steepest Descent algorithm.

$$\vec{x}_{k+1} = \vec{x}_k - \lambda_k \, \vec{g}_k \tag{A.18}$$

where the search vector, $\vec{p}_k$, is in the direction of the negative gradient, $-\vec{g}_k$, which relates as

$$\vec{p}_k = -\nabla V(\vec{x})\big|_{\vec{x}=\vec{x}_k} = -\vec{g}_k \tag{A.19}$$

The Steepest Descent algorithm for the simple quadratic becomes

$$\vec{x}_{k+1} = \vec{x}_k - \lambda_k \, \mathrm{A} \, \vec{x} \tag{A.20}$$

The Steepest Descent algorithm applied to a simple quadratic gives insight into the optimization schemes characteristics. For a simple quadratic performance surface, with A being an identity matrix and the state values being the relative distance from the equilibrium point, the performance surface is shown in Figure A.1. In this simple case the minimum is located at the origin and the contour lines of equal potential are perfect circles. An iterative search along the steepest gradient can be shown to converge, but the convergence path will vary dependent on the step size parameter.

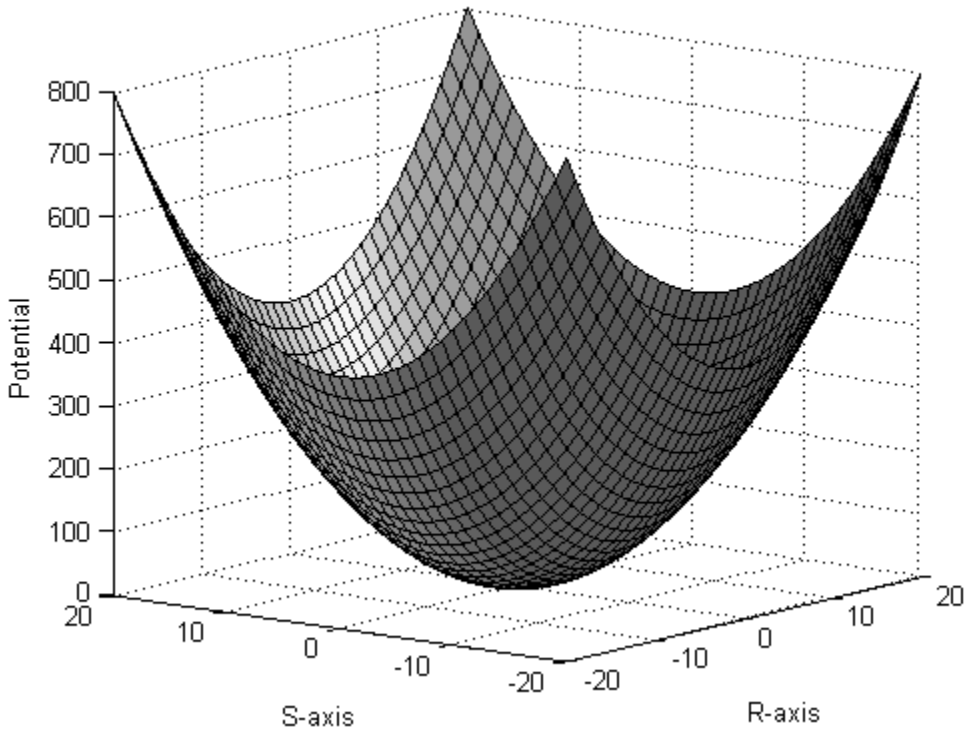

Figure A.1 Sample Performance Surface.

Determination of the step size parameter, λ_k ,can be constant or variable. Constant step size parameters are general based on a desired convergence response. Evaluating the largest eigenvalues of the Hessian can ensure convergence and even provide an underdamped response. For example, consider a system described by the quadratic function with

$$\mathrm{A} = \begin{bmatrix} 2 & 0 \\ 0 & 50 \end{bmatrix} \tag{A.21}$$

and its minimum at the origin. This function is steeper along the y-axis then the x-axis and has eigenvalues of (2, 50). The maximum eigenvalue, $eig_{\max}$, of the quadratic function can be used to determine the bounds on the step size parameter. For convergence, the step size must meet the following condition,

$$0 < \lambda < \left(\frac{2}{eig_{\max}} \right) \tag{A.22}$$

For an overdamped response, the step size must meet the following condition,

$$0 < \lambda < \left(\frac{1}{eig_{\max}} \right) \tag{A.23}$$

The convergence characteristics of four different step sizes, using the function described in Equation (A.21), are shown with the 3D performance surface in Figure A.2 and as contours in Figure A.3. For the step size meeting the conditions of Equation (A.23), $\lambda = 0.01$, the Steepest Descent method converges with an overdamped behavior. For the step size meeting the condition of Equation (A.23), $\lambda = 0.03$, the Steepest Descent method converges with some overshoot behavior. For the case where the step size is equal to the upper range condition of Equation (A.23), $\lambda = 0.04$, the method does not converge and continues to oscillate. If the step size is increase beyond this condition, $\lambda = 0.041$, the method diverges. Notice that the search directions are always opposite the gradient and thus perpendicular to the contour lines.

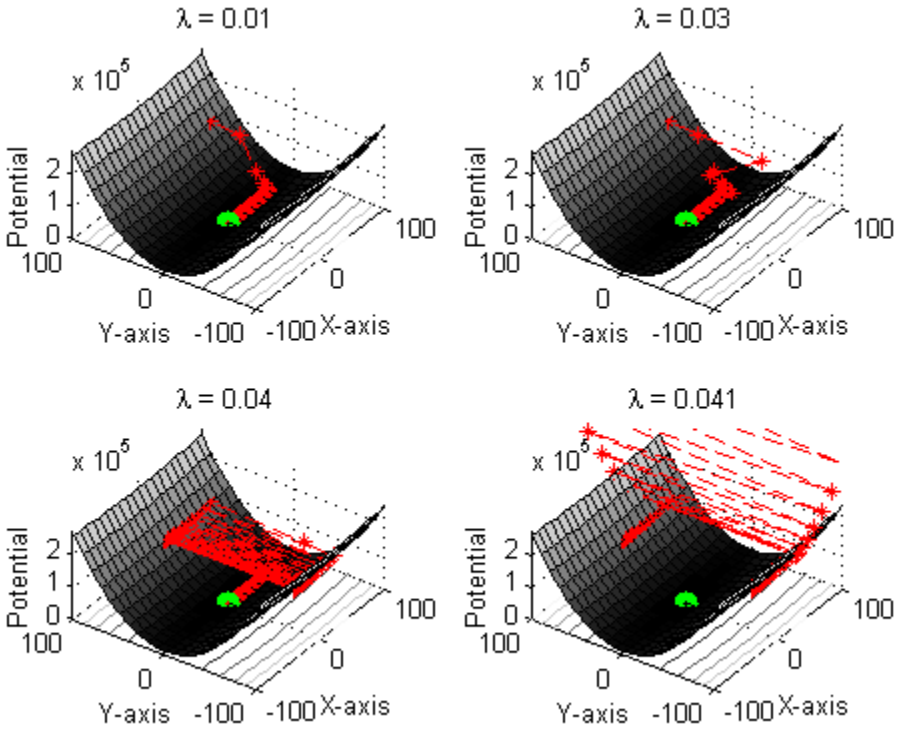

Figure A.2 Steepest Descent Performance for Fixed Step Sizes in 3D.

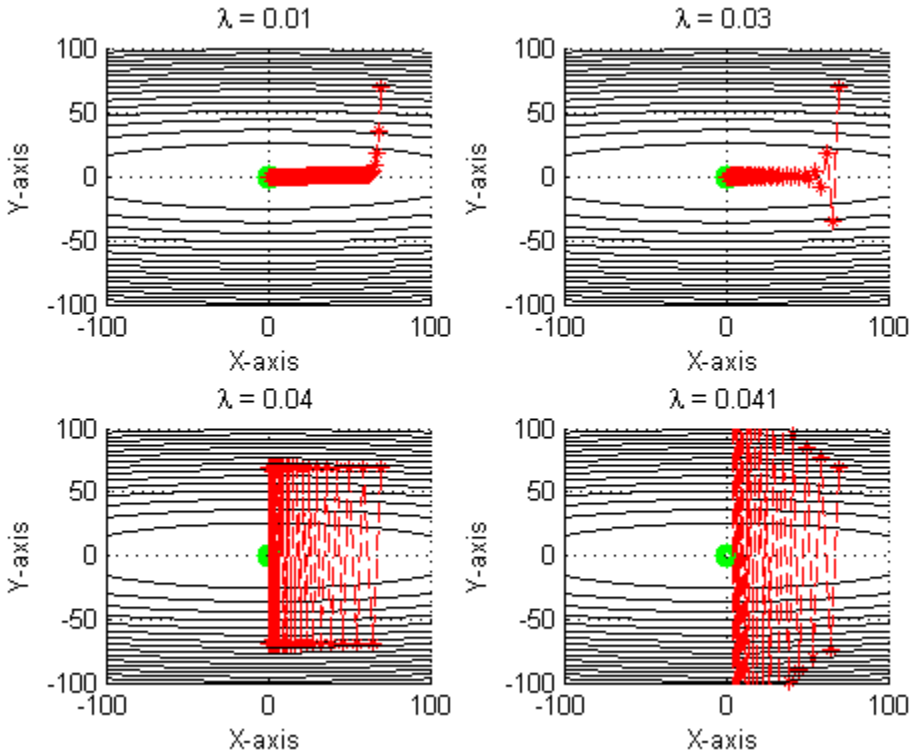

Figure A.3 Steepest Descent Contour for Fixed Step Sizes.

Variable step sizes may be based on minimization-on-a-line or momentum learning methods [85]. Minimization-on-a-line allows variation of the step size at each iteration, k. The analytical minimization for a quadratic function is satisfied by setting its derivative equal to zero and solving for the step size [85].

$$\lambda_k = -\left(\frac{\vec{g}_k{}^T \vec{p}_k}{\vec{p}_k{}^T \mathrm{A}^T \vec{p}_k} \right) \tag{A.24}$$

The Hessian for simple quadratic functions is symmetric, so the transpose may be dropped in these cases. The convergence characteristics of the variable step size, using the function described in Equation (A.24), are shown with the 3D performance surface in Figure A.4 and as contours in Figure A.5. The convergence generally takes less iterations, but dramatically oscillates due to the changing of the search direction at each iteration. Each step is orthogonal to the previous step and parallel to every other step. This pattern is due to the primary axis of the contour shape and the tangential intercept of

the contour lines. This direction change may not be reasonable with systems which have large momentum and can not change direction rapidly.

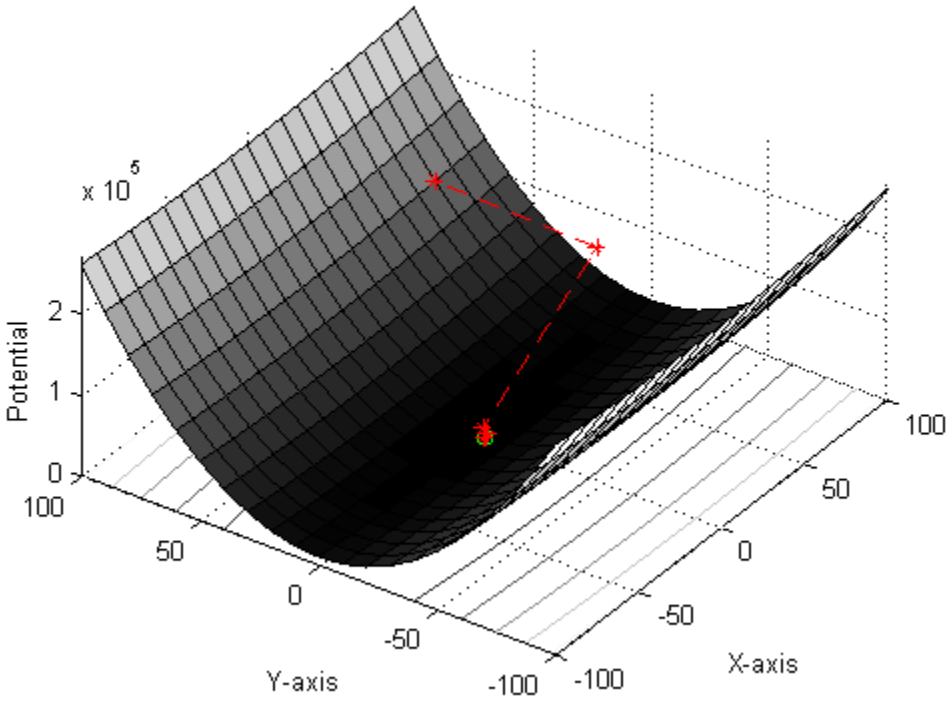

Figure A.4 Steepest Descent Performance for Variable Step Sizes in 3D.

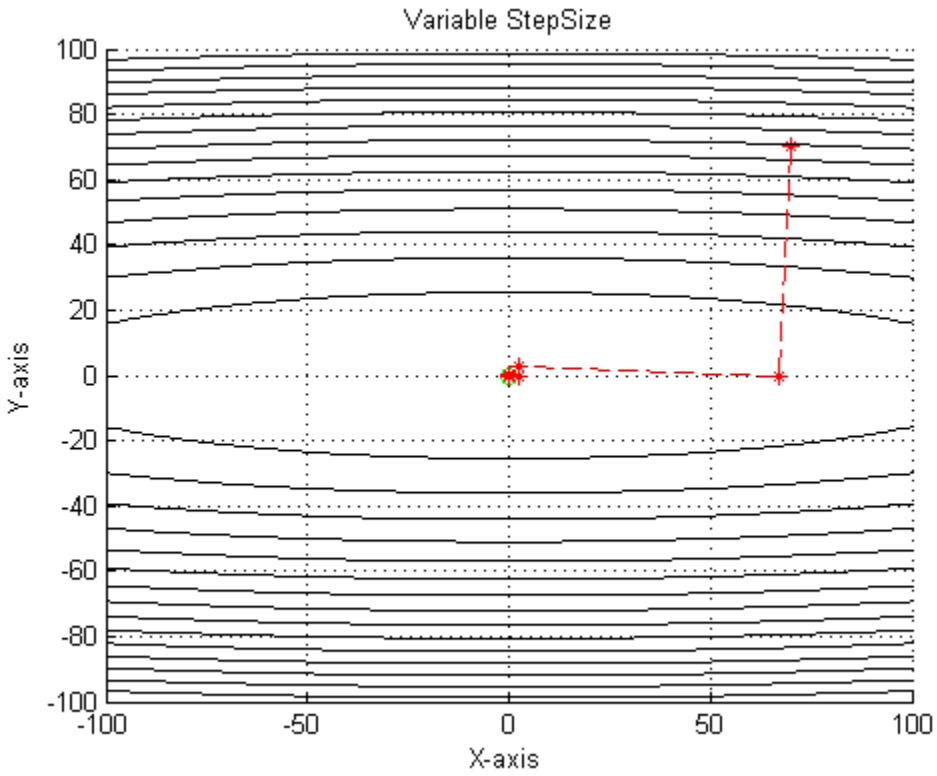

Figure A.5 Steepest Descent Contour for Variable Step Sizes.

Momentum learning smoothes out the iterations and may speed convergence for consistent trajectories [85]. This is done by considering both the current and previous iteration information and weighting them to reduce oscillations. The momentum learning coefficient, μl, acting as a low-pass filter on the search direction oscillation, weighs the new search direction with respect to the previous search.. The Steepest Descent with fixed step size and momentum learning yields the following algorithm

$$\vec{x}_{k+1} = \vec{x}_k - \lambda_k \left[(1-\mu l)\, \vec{g}_k - (\mu l)\, \vec{g}_{k-1} \right] \tag{A.25}$$

For $\mu l = 0$, the function has no memory of previous searches and is the original Steepest Descent algorithm. For $\mu l = 1$, the search direction is not updated from iteration to iteration. Momentum learning is not generally useful for variable step size implementation. The step size is determined from search directions which are damped by the momentum and as a result overshoot usually occurs. This may increase the number of iterations necessary for convergence. For those interested, a discussion of both Newton's Method and Conjugate Gradient search algorithms follows.

B. NEWTON'S METHOD

The Newton's Method is based on local second order approximation, as an extension of the Steepest Descent first order approximation. The Newton's Method algorithm is

$$\vec{x}_{k+1} = \vec{x}_k - \mathrm{A}_k^{-1} \cdot \vec{g}_k \tag{A.26}$$

For simple quadratic functions, Newton's method converges in one step. This is dependent on the Hessian being positive definite. The Newton's method usually converges faster than the Steepest Descent method. However, the second order approximation is computationally expensive, due to the need to calculate a Hessian and its inverse.

A first order variation, referred to as the Gauss-Newton Method, only requires the calculation of the Jacobian matrix, J , and its inverse. The Gauss-Newton algorithm is

$$\vec{x}_{k+1} = \vec{x}_k - J(\vec{x}_k)^{-1} \cdot v(\vec{x}_k) \tag{A.27}$$

where $v(\vec{x}_k)$ comes from rewriting Equation (A.11) as $V(\vec{x}) = v(\vec{x})^T * v(\vec{x})$. This method neglects second order terms and can result in the Hessian not being invertible.

Another variation, called the Levenberg-Marquardt Scheme, converges faster at the cost of additional memory. This is allows for a single parameter adjustment, μk , to vary the search performance from a Gauss-Newton scheme to the Steepest Descent with a small step size. The Levenberg-Marquardt algorithm is

$$\vec{x}_{k+1} = \vec{x}_k - \left[J^T(\vec{x}_k) \cdot J(\vec{x}_k) + \mu k \cdot I \right]^{-1} * J^T(\vec{x}_k) \cdot v(\vec{x}_k) \tag{A.28}$$

where μk is determined so that the inverse matrix exists. As $\mu k \to 0$ the algorithm approaches the Gauss-Newton algorithm and as $\mu k \to \infty$ the algorithm approaches the Steepest Descent with a small learning rate. This method converges fast, but has a drawback of requiring more memory.

C. CONJUGATE GRADIENT

Since using a second order approximation may be too expensive, a first order approach called conjugate gradient may be effective. This approach steps along the

conjugate directions of the function, with the initial step being in the direction of negative gradient. The step size can be adjusted to minimize the function along the selected search direction. For larger problems the conjugate gradient method is more efficient, since it does not require computation of the eigenvalues of the Hessian or its inverse. It usually converges in a number of interactions which is equal to the number of states. For instance a simple three state, [x, y, z], quadratic function would converge in three steps. However, as the function varies from quadratic then the iteration scheme will require more steps to converge. A primary conjugate gradient algorithm of interest is called the Fletcher Reeves iteration scheme. This iteration scheme iterates as follows

$$\vec{x}_{k+1} = \vec{x}_k + \lambda_k \cdot \vec{p}_k \tag{A.29}$$

$$\vec{g}_{k+1} = \mathrm{A} \cdot x_{k+1} + d \tag{A.30}$$

$$\vec{p}_{k+1} = -\vec{g}_{k+1} + \beta_{k+1} \cdot \vec{p}_k \tag{A.31}$$

$$B_{k+1} = \left(\frac{\vec{g}_{k+1}{}^T \cdot \vec{g}_{k+1}}{\vec{g}_k{}^T \cdot \vec{g}_k} \right) \tag{A.32}$$

The step size, λ_k, varies as minimization along a line, refer to Equation (A.24). The gradient, $\vec{g}_k$, is calculated as previously discussed. However, now the search direction, $\vec{p}_k$, varies from the Steepest Descent after the first iteration. The scalar variation of search direction, B_k, can take may forms, but all are based on Gram-Schmidt orthogonalization concepts [85]. Notice that the initial search direction is along the Steepest Descent, $\vec{p}_0 = -\vec{g}_0$, such that $B_0 = 0$.

The convergence characteristics of the conjugate gradient Fletcher Reeves iteration scheme, using the function described in Equations (A.29) - (A.32), are shown with the 3D performance surface in Figure A.6 and as contours in Figure A.7. The search direction of the two iteration steps is slightly greater than 90 degrees. This illustrates the variation of the search direction which allows the conjugate gradient approach to converge quicker then the Steepest Descent method, even though they are using the same variable step size.

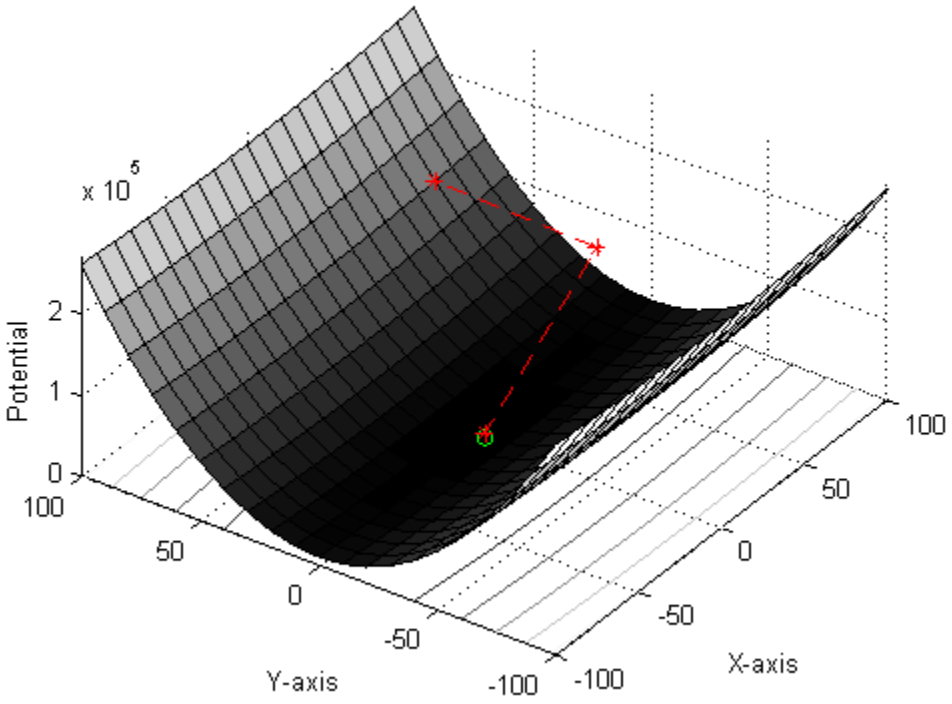

Figure A.6 Conjugate Gradient with Fletcher Reeves Scheme.

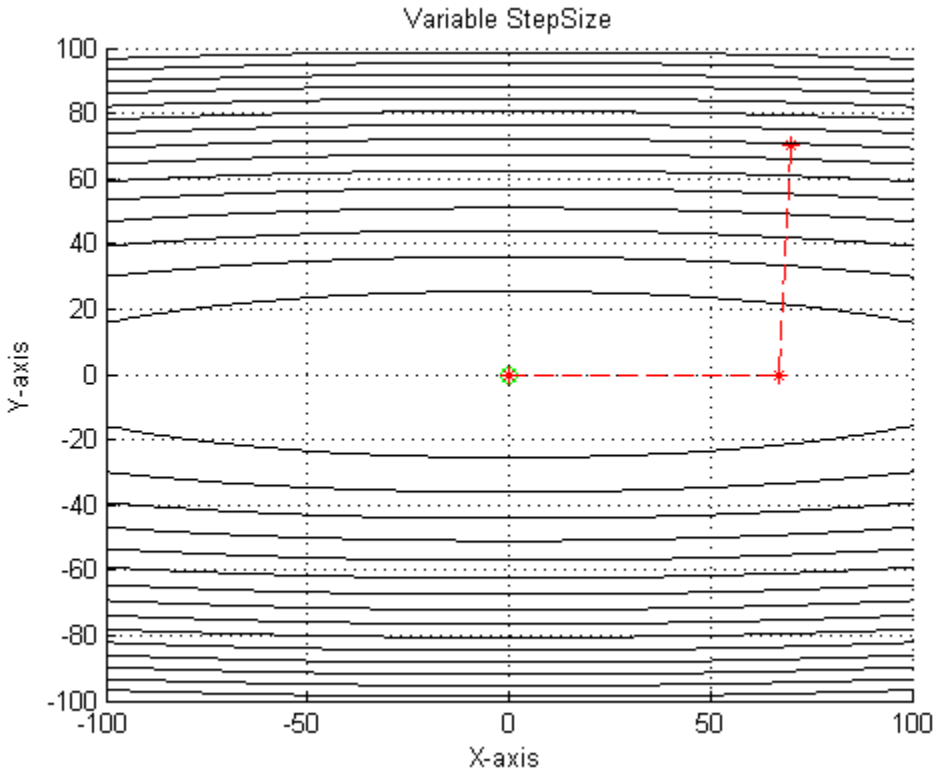

Figure A.7 Conjugate Gradient Contour with Fletcher Reeves Scheme.

D. DISCUSSION

The Steepest Descent search method is the simplest and most straight forward of the discussed methods. The Newton's Method and Conjugate Gradient approaches require additional computation. In the dynamics environment the additional computation may not yield any improvement in the control algorithm performance. However, if the control algorithm cycle is at least twice as fast as the measurement cycle then implementation of the Conjugate Gradient search algorithm should be considered. Practical limitations in actuator performance may show that the desired changes in search directions are impractical. For instance, drastic directional changes are impossible for physical system with momentum and limited actuation, such as spacecraft. Although, the same might be observed in a Steepest Descent algorithm which does not decrease in step size as it approaches equilibrium point.

Modification may also be made in the state weighting to influence the initial and final search directions. This could be useful for constrained docking approaches where convergence along a relative axis is necessary. The weighting of the states should have direct correlation with the cost and performance functions discussed previously. By modifying the contour shape of the quadratic surface the search algorithm can be prejudiced to approach from the desired direction. These types of modification usually negatively influence efficiency due to the forcing of required, but artificial, maneuver. This has a direct comparison with the collision avoidance capability.

It is not envisioned that modifications to the search algorithm will dramatically improve the APF convergence performance. Although, some improvement may be possible in control efficiency or maneuver duration. It was initially thought that replacing the state scaling function, in Equation (A.30), with the Clohessy-Wiltshire's A matrix from Equation (4.16) may allow for the linearized spacecraft dynamics to be incorporated into the APF. This relationship appears to have merit when used for the spatial states, but requires further analysis. In the APF development velocity considerations were the primary driving states for both successful convergence and collision avoidance. Determination of desirable velocity parameters may become much less intuitive for various close proximity maneuvers. So, simple reliance on the geometrically based APF may be limited regardless of the search method applied.

APPENDIX B: LQR WITH COLLISION AVOIDANCE GAINS

During the course of developing the LQR/APF with collision avoidance control algorithm, consideration was given to the direct incorporation of collision avoidance parameters into the LQR gain matrixes. It was believed to be a logic step to consider the inclusion of collision avoidance weighting into the gain matrixes of the LQR's cost function; refer to Equation (7.1). Modification to the gain matrixes for the states, Q,and control effort, R, continued to be of primary interest. There was some initial success in selecting additive functions that resulted in successful collisions avoidance, however rigorous evaluation is still necessary. For instance, it is unclear what effect additive gain terms modification may have on stability, convergence, and efficiency.

The approach was to add positive semi-definite parameters into the state gain matrix without increase the number of states. Although adding relative position and velocity states for each obstacle could conceptually work, including these full or partial states for each potential obstacle causes the computational requirements to increase dramatically. To avoid this consequence, the inclusion of obstacle information might be accomplished by increasing the cost of core spacecraft's states in the established relative structure. This should influence the control effort to avoid obstacle regions.

Numerous attempts to include additive terms in the gain matrixes were made. Additive positive, semi-definite terms were only considered, since the gains matrixes must maintain the proper form. It was determined that relative position information could not be incorporated with only additive terms, due to the complications of weighting position terms of obstacles in the presence of the goal. However, the inclusion of additive velocity terms, C_v , showed potential. The additive velocity term is based on the Gaussian shaping function, k_v from Equation (7.19), and the magnitude of the Chase spacecraft's velocity component in the direction of the obstacle, $\left|\bar{v}_{co}\right|$ based on Equation (7.17). If the Chase spacecraft velocity is positive in the relative direction of the obstacle then the additive velocity gain terms was determined to be

$$C_v = \frac{C_m k_v}{\left(v^2 + d_e\right)} \left| \frac{\vec{r}_{co}}{r_{co}} \right| \tag{B.1}$$

where the small positive constant, d_e, is included to prevent numerical difficulty due to division by zero. The additive velocity gain magnitude shaping function, C_m was selected to increase influence in response to velocity toward obstacles while decreasing in influence in the vicinity of the goal. This decrease in influence is a scaled version of the safety function, k_s, determined in Equation (7.21). The resulting magnitude shaping function is

$$C_m = \left(\frac{r_{cg}}{r_{init}} k_s + 2 \left| \vec{v}_{co} \right|^2 \right) \left| \frac{\vec{r}_{co}}{r_{co}} \right| \tag{B.2}$$

The additive terms for the velocity gain components are included along the diagonal of the LQR state weighting matrix and modify Equation (7.7) as follows

$$Q = \begin{bmatrix} \frac{1}{r_{cg}} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{r_{cg}} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{r_{cg}} & 0 & 0 & 0 \\ 0 & 0 & 0 & (1 + C_{v1}) k_Q & 0 & 0 \\ 0 & 0 & 0 & 0 & (1 + C_{v2}) k_Q & 0 \\ 0 & 0 & 0 & 0 & 0 & (1 + C_{v3}) k_Q \end{bmatrix} \tag{B.3}$$

where k_Q was determined from Equations (7.10) and (7.12) to be

$$k_Q = \frac{r_{cg}}{\left(v_{\max} \frac{r_{init}}{r_{\max}} \right)^2} \tag{B.4}$$

No modification was made to the control effort weighting matrix, R, since control effort should not be limited in the presence of obstacles. The incorporation of terms that would decrease weighting was not considered due to the challenges of maintaining the proper cost relationship between various states.

The LQR additive collision avoidance gains were generally successful in avoiding obstacles and converging toward the goal position. Preliminary results show that the LQR with collision avoidance gains control algorithm could avoid obstacles and other spacecraft while converging to within 4.0 meters of the Target spacecraft's outer boundary. The results for the six close proximity maneuvers with collision avoidance are listed in Table B.1. For these collision avoidance maneuvers, the goal position is the center of the Target spacecraft. This requires that the Target spacecraft's repulsion to allow the Chase spacecraft to converge while avoiding impact. Stationary obstacles are placed at positions along the unobstructed path of the Chase spacecraft. These obstacles have an actual diameter of 2.0 meters. Generally, avoiding larger obstacles requires more control effort and time. The performance accuracy shows that general convergence and rendezvous can be achieved while enabling collision avoidance. The relative position, velocity and control effort plots of the first Chase spacecraft are shown in Figure B.1 through Figure B.3. Although densely packed obstacle regions tend to have an additive repulsion force, which keep later converging spacecraft at further distances away from the mutual goal location. For instance, the three Chase spacecraft in the near rendezvous maneuvered to within 1.0, 2.0, and 4.0 meters of the Target spacecraft, based on their order of arrival.

Rendezvous Maneuver	**LQR with CA Gains**
Near with Obstacle Initial RSW [0, 70, 0] m	$\Delta v = 0.6760$ m/s
	$t_d = 1210$ s
Near with Obstacle Initial RSW [50, -100, -50] m	$\Delta v = 0.5196$ m/s
	$t_d = 1073$ s
Near with Obstacle Initial RSW [100, 100, 100] m	$\Delta v = 0.6845$ m/s
	$t_d = 1069$ s

Table B.1 Rendezvous Maneuver with Collision Avoidance Results.

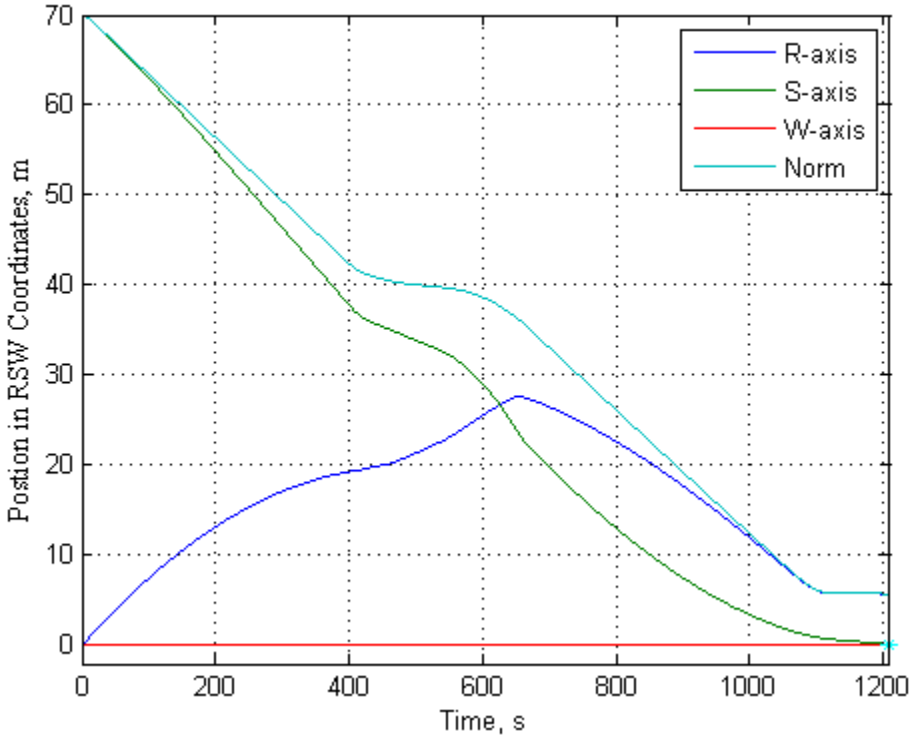

Figure B.1 Chase Spacecraft Relative Position.

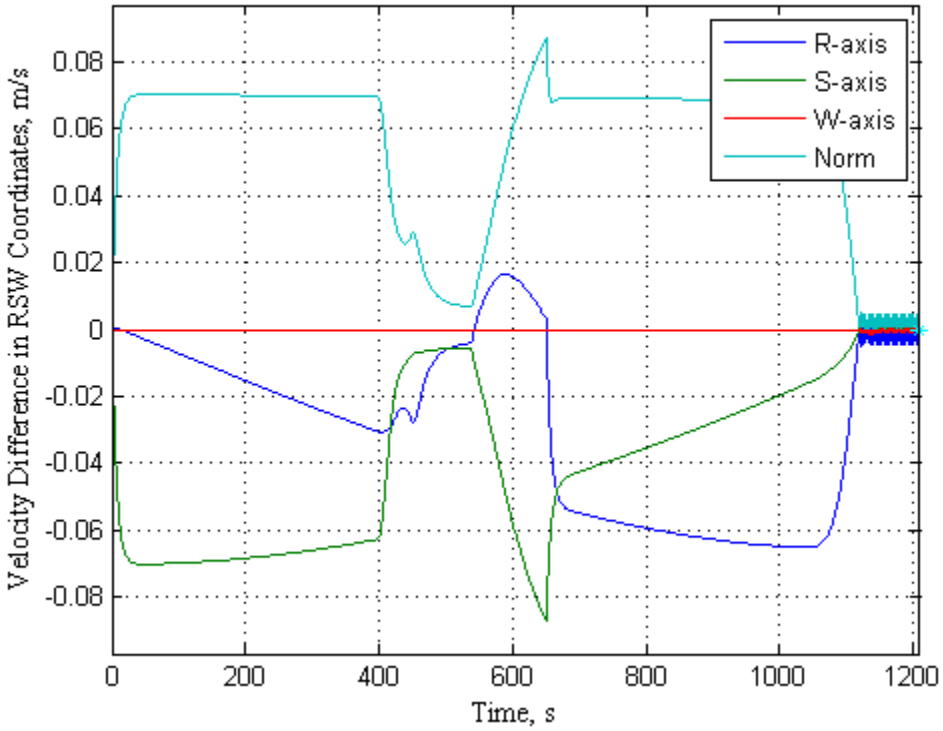

Figure B.2 Chase Spacecraft Relative Velocity.

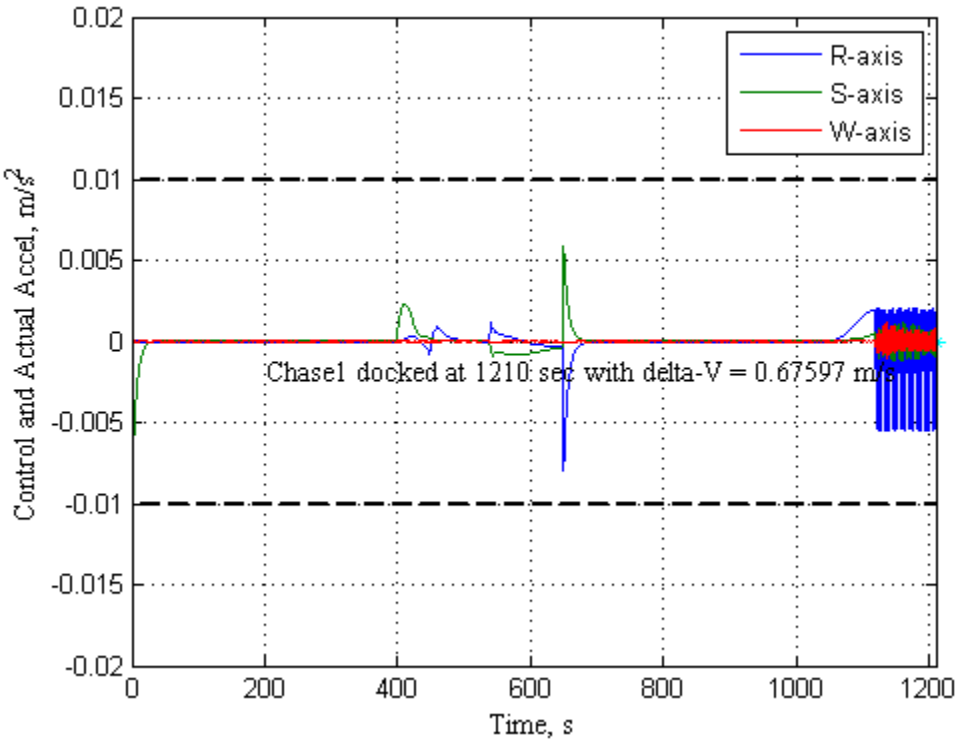

Figure B.3 Chase Spacecraft Control Effort.

There were several difficulties and deficiencies this initial approach. First, Chase spacecraft approaching along the relative coordinate frame's W-axis did not properly avoid obstacles directly along their path. In these cases the Chase spacecraft slowed in the region of the obstacle, but continued to push toward converges. If the obstacle gain was increased, Chase would avoid collision by stopping and no longer converging. This problem may preventable by perturbing the Chase off of this perfectly balanced approach. This could be done by adding a shifting factor if an obstacle is detected directly along this W-axis path. Also, adjusting the weighting the W-axis motion could influence the Chase spacecraft to move along the orbital plane.

Next, the focus on velocity terms may lead to some chattering when a Chase spacecraft skirts along an obstacle's region of influence. In this situation the relative velocity toward the obstacle fluctuates on the exterior of the avoidance arch. These fluctuations in close proximity would result in large cost variation. This causes a chattering phenomenon, as shown in Figure B.3. This issue may require additional gain

terms or control logic to be included. The velocity gains as initially set to increase near obstacles, but may need more complex functional relationships as they evade in the region of influence.

Finally, docking in the presence of obstacle was not successful. Problems developed when state cost decrease during goal convergence could not be properly tuned with collision avoidance in this terminal region. If too large, the collision avoidance terms prevented convergence; if too small, the obstacles were not properly avoided. The superposition of numerous obstacles in a region also, resulted in challenges for this conceptual approach. These problems were previously overcome by applying decaying obstacle influence along the geometric region of approach. The application the APF based techniques did not work for this approach. The cost variations would change the feedback response in a fashion that would restrict the usefulness of the logic. For instance, if a the obstacle's gain parameters increase the cost function to much, such that the Chase spacecraft is not longer approaching, then damping of that function only slows divergence. This leads to local minimums around some obstacles. The obstacle avoidance parameters may be shaped to avoid these cost circumstances away from the goal position, but there is less possible cost variation in the region of the Target spacecraft.

The high degree accuracy needed for more precision rendezvous and docking maneuvers was not achieved, so direct comparison with Chapter VI results could not be made. Attempts to balance the gains of position and velocity in the vicinity of the Target spacecraft tended to result in chattering. Additional research may still prove successful in the incorporation of additive collision avoidance terms in the LQR's state gain matrix. Any success will need to take complex weighing and state relationships into consideration. These relationships may prove more complicated then practically realizable. The benefits of such an algorithm are not yet apparent.

LIST OF REFERENCES

[1] G. Yang, Q. Yang, V. Kapila, D. Palmer, and R. Vaidyanathan, "Fuel Optimal Manoeuvres for Multiple Spacecraft Formation Reconfiguration Using Multi-Agent Optimization," *International Journal of Robust and Nonlinear Control*, Volume 12, February-March 2002, pp. 243-283.

[2] O. Khatib, "Real-Time Obstacle Avoidance For Manipulators and Mobile Robots," *Proceedings of the International Conference on Robotics and Automation*, Volume 2, March 1985, pp. 500-505.

[3] D. E. Koditschek, "Exact Robot Navigation by Means of Potential Functions: Some Toplological Considerations," *Proceedings of the International Conference on Advanced Robotics (ICAR),* Volume 4, March 1987, pp. 1-6.

[4] X. Yun and K. Tan, "A Wall-Fallowing Method for Escaping Local Minima in Potential Field Based Motion Planning," *Proceedings of the International Conference on Advanced Robotics (ICAR),* July 1997, pp. 421-426.

[5] V. Gazi, "Swarm Aggregation Using Artificial Potentials and Sliding Mode Control," *Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, Hawaii, Volume 2, December 2003, pp. 2041-2046.

[6] W. S. Newman and N. Hogan, "High Speed Robot Control and Obstacle Avoidance using Dynamic Potential Functions," *Proceedings of the IEEE International Conference on Robotics and Automation*, Volume 4, March 1987, pp. 14-24.

[7] E. Rimon and D. E. Koditschek, "Exact Robot Navigation Using Artificial Potential Functions," *IEEE Transactions on Robotics and Automation*, Volume 8, Issue 5, October 1992, pp. 501-518.

[8] P. Palmer, "Optimal Relocation of Satellites Flying in Near-Circular-Orbit Formations," *Journal of Guidance Control and Dynamics*, Volume 29, Number 3, May-June 2006, pp. 519-526.

[9] V. Gazi, "Swarm Aggregation Using Artificial Potentials and Sliding Mode Control," *IEEE Transactions on Robotics and Automation*, Volume 21, Issue 6, December 2005, pp. 1208-1214.

[10] W. Ren and R. W. Beard, "Formation Feedback Control for Multiple Spacecraft via Virtual Structure," *IEEE Proceedings of Control Theory and Applications*, Volume 151, Issue 3, 23 May 2004, pp. 357-368.

[11] Y. Q. Chen and Z. Wang, "Formation Control: a Review and a New Consideration,," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Edmonton Alberta, Canada, August 2005, pp. 3181-3186.

[12] O. Brown, P. Ermenko, and C, Roberts, "Cost Benefit Analysis of a Notational Fractionated SATCOM Architecture," AIAA 2006-5328, *24th AIAA International Communications Satellite Systems Conference (ICSSC)* , San Diego, CA, June 2006.

[13] C. Mathieu and A. L. Weigel, "Assessing the Flexibility provided by Fractionated Spacecraft," AIAA 2005-6700, *Proceedings of Space 2005 Conference*, Long Beach, CA, August 2005.

[14] R. T. Gavin, T. M. Erkenswick, and A. M. Foti, "Russian Automated Rendezvous & Docking (AR&D) Statistics", *NASA AR&D Team Memo*, Johnson Space Center (JSC), Houston, TX , 2 August 2006.

[15] T. Malik, "Russian Cargo Ship Begins Trip to Space Station," http://www.space.com/missionlaunches/060424_exp13_prog21_launch.html, SpaceFlight, 24 April 2006,[cited 12 June 2006].

[16] NASA, Dart Mission, http://www.nasa.gov/missions/science/dart_into_space.html [cited 22 August 2006].

[17] T. M. Davis et. al., "XSS-10 Micro-Satellite Flight Demonstration Program," *Proceedings of the 17th Annual AIAA/USU Conference on Small Satellites*, Logan, Utah, August 2003.

[18] AFRL, XSS-11, http://www.vs.afrl.af.mil/FactSheets/XSS11-MicroSatellite.pdf [cited 22 August 2006].

[19] A. B. Bosse et. al, "SUMO: Spacecraft for the Universal Modification of Orbits," *Defense and Security Symposium Proceedings*, Orlando, Florida, April 2004, pp 36-46.

[20] *DARPA, Orbital Express,* http://www.darpa.mil/tto/programs/oe.htm [cited 22 August 2006].

[21] G. Creamer, "The SUMO/FREND Project: Technology Development for Autonomous Grapple of Geosynchronous Satellites," 30th Annual American Astronautical Society Guidance and Control Conference, Breckenridge, CO, February 2007.

[22] L. David, "Military Micro-Sat Explores Space Inspection, Servicing Technologies" http://www.space.com/businesstechnology/050722_XSS-11_test.html, Technology, 22 July 2005 [cited 22 August 2006].

[23] M. A. Dornheim, "Orbital Express to Test Full Autonomy for On-Orbit Service," *Aviation Week & Space Technology (AW&ST)*, http://www.aviationnow.com/avnow/news/channel_awst_story.jsp?id=news/aw060506p1.xml, 4 June 2006 [cited 22 August 2006].

[24] F. Kennedy, A. Epstein, S. Dubowsky, and S. P. Worden, "Tiny, Independent, Coordinating Spacecraft (TICS): Leapfrogging the Microsatellite Revolution." *Proceedings of the 20th Annual AIAA/USU Conference on Small Satellites*, Logan, Utah, August 2006.

[25] MIT, Space System Laboratory, SPHERES, http://ssl.mit.edu/spheres/ [cited 22 August 2006].

[26] A. S. Otero, A. Chen, D. W. Miller, and M. Hilstad, "SPHERES: Development of an ISS Laboratory for Formation Slight and Docking Research," *2002 IEEE Aerospace Conference Proceedings*, Big Sky, MT, 9-16 March 2002, pp. 59-73.

[27] J. J. Sellers, *Understanding Space: An Introduction to Astronautics*, McGraw Hill, Boston, Massachusetts, revised 2nd edition, 2004.

[28] G. M. Anderson, "Comparison of Optimal Control and Differential Game Intercept Missile Guidance Laws," *Journal of Guidance, Control and Dynamics*, Volume 4, pp. 109-115, 1981.

[29] M. J. Sidi, *Spacecraft Dynamics and Control*, Cambridge University Press, New York, NY, 1997.

[30] D. P Scharf, F. Y. Hadaegh, and S. R. Ploen, "A Survey of Spacecraft Formation Flying Guidance and Control (Part II): Control," *2004 American Control Conference Proceedings*, Volume 4, 30 June-2 July 2004, pp. 2976-2985.

[31] S. McCamish, M. Pachter and J. J. D'Azzo, "Optimal Formation Flight Control," AIAA 1996-3868, Proceedings of the AIAA Guidance, Navigation and Control Conference, San Diego, CA, 29-31 July 1996.

[32] D. A. Vallado, *Fundamentals of Astrodynamics and Applications*, Microcosm Press, El Segundo, California, 2nd Edition, 2001.

[33] E. W. Gettys, F. J. Keller, M. J. Skove, *Classical and Modern Physics*, McGraw-Hill Book Company, New York, NY, 1989.

[34] B. Wie, *Space Vehicle Dynamics and Control*, American Institute of Aeronautics and Astronautics (AIAA), Reston, Virginia, 1998.

[35] V. A. Chobotov, *Spacecraft Attitude Dynamics and Control*, Krieger Publishing Company, Malabar, Florida, 1991.

[36] Satellite Tool Kit (STK), Analytical Graphics Incorporated (AGI), Exton, PA, https://www.agi.com/products/desktopApp/stkFamily/modules/core/stk/ [cited 10 September 2006].

[37] W. J. Larson and J. R. Wertz, *Space Mission Analysis and Design*, Microcosm Press, El Segundo, California, 3rd Edition, 2004.

[38] Honeywell International Inc., Products and Services, HR 0610 Reaction Wheel, http://www.honeywell.com/sites/aero/Pointing-Momentum-Control.htm [Cited 13 March 2007].

[39] MATLAB and Simulink, The MathWorks Incorporated, http://www.mathworks.com/ [cited 18 September 2006].

[40] F. McQuade, "*Autonomous Control for On-Orbit Assembly Using Artificial Potential Functions"*, PhD Thesis, Department of Aerospace Engineering, University of Glasgow, Scotland, United Kingdom, 1997.

[41] W. H. Clohessy and R. S. Wiltshire, "Terminal Guidance System for Satellite Rendezvous," *Journal of the Aerospace Sciences*, Volume 27, Number 9, pp. 653-658, 1960.

[42] D. W. Gim and K. T. Alfriend, "State Transition Matrix of Relative Motion for the Perturbed Noncircular Reference Orbit," *Journal of Guidance, Control and Dynamics*, Volume 26, pp. 956-971, 2003.

[43] K. T. Alfriend and H. Yan, "Evaluation and Comparison of Relative Motion Theories," *Journal of Guidance, Control and Dynamics*, Volume 28, Issue 2, pp. 254-261, 2005.

[44] D. F. Lawden, "Optimal Transfer Via Tangential Ellipses," *British Interplanetary Society Journal*, Volume 11, Issue 6, 1952.

[45] P. Mendy, "*Multiple Satellite Trajectory Optimization*," Masters Thesis, Department of Mechanical and Astronautical Engineering, Naval Postgraduate School, Monterey, California, 2004.

[46] E. Bonabeau, M. Dorigo, G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press, New York, NY, 1999.

[47] M. E. Campbell and B. Udrea, "Collision Avoidance in Satellite Clusters," Proceedings of the American Control Conference, Volume 2, Anchorage, AK, May 2002, pp. 1686-1692.

[48] J. Guldner and V. I. Utkin, "Sliding Mode Control for Gradient Tracking and Robot Navigation Using Artificial Potential Fields," *IEEE Transactions on Robotics and Automation*, Volume 11, Issue 2, April 1995, pp. 247-254.

[49] M. Ayre, D. Izzo, L. Pettazzi, "Self Assembly in Space Using Behaviour Based Intelligent Components," *Proceedings of Towards Autonomous Robotic Systems (TAROS)*, London, UK, 12-14 September 2005, pp. 1-8.

[50] C. R. McInnes, "Autonomous Proximity Maneuvering Using Artificial Potential Functions," *European Space Agency Journal*, Volume 17, Number 2, 1993, pp. 159-169.

[51] C. R. McInnes, "Distributed Control of Manouvring Vehicles for On-Orbit Assembly," *Journal of Guidance Control and Dynamics*, Volume 18, Number 5, September-October 1995, pp. 1204-1206.

[52] I. Lopez and C. R. McInnes, "Autonomous Rendezvous Using Artificial Potential Function Guidance," *Journal of Guidance Control and Dynamics*, Volume 18, Number 2, March-April 1995, pp. 237-241.

[53] H. Umehara and C. R. McInnes, "Penalty-Function Guidance for Multiple-Satellite Cluster Formation," *Journal of Guidance Control and Dynamics*, Volume 28, Number 1, January 2005, pp. 182-185.

[54] J. S. Neubauer, "*Controlling Swarms of Micro-Utility Spacecraft*," DSc Thesis, Department of Mechanical and Aerospace Engineering, Washington University, Saint Louis, Missouri, 2006.

[55] J. S. Neubauer and M. A. Swartwout, "Controlling Swarms of Bandit Inspector Spacecraft," *Proceedings of the 20th Annual AIAA/USU Conference on Small Satellites*, Logan, Utah, August 2006.

[56] A. Sparks, "Satellite Formationkeeping Control in the Presence of Gravity Perturbations," *Proceedings of the American Control Conference,* Chicago, IL, June 2000, pp. 844-848.

[57] L. Breger and J. P. How, "Safe Trajectories for Autonomous Rendezvous of Spacecraft," AIAA Guidance, Navigation and Control Conference, Keystone, CO, August 2006.

[58] L. F. Lee, R Bhatt and V. Krovi, "Comparison of Alternate Methods for Distributed Motion Planning of Robot Collectives within a Potential Field Framework," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Barcelona, Spain, April 2005, pp. 99-104.

[59] J. C. Latombe, *Robot Motion Planning*, Kluwer Academic Publishers, Boston, Massachusetts, 1991.

[60] R. E. Kalman and J. E. Bertram, "Control System Analysis and Design Via the Second Method of Lyapunov, Part I: Continuous Systems," *Journal of Basic Engineering*, Volume 82, June 1960, pp. 371-393.

[61] R. E. Kalman and J. E. Bertram, "Control System Analysis and Design Via the Second Method of Lyapunov, Part II: Discrete Systems," *Journal of Basic Engineering*, Volume 82, June 1960, pp. 394-400.

[62] H. K. Khalil, *Nonliner Systems*, Third Edition Prentice Hall, Upper Saddle River, New Jersey, 2002.

[63] S. McCamish, M. Romano, and X. Yun, "Autonomous Distributed LQR/APF Control Algorithm for Multiple Small Spacecraft during Simultaneous Close Proximity Operations," Proceedings of the 21st Annual AIAA/USU Conference on Small Satellites, Logan, Utah, August 2007 (submitted for publication).

[64] S. McCamish, M. Romano, and X. Yun, "Autonomous Distributed Control Algorithm for Multiple Small Spacecraft during Close Proximity Operations," Proceedings of the AIAA Guidance, Navigation and Control Conference, Hilton Head, SC, August 2007 (submitted for publication).

[65] A. Masoud, "Solving the Narrow Corridor Problem in Potential Field-Guided Autonomous Robots," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Barcelona, Spain, April 2005, pp. 2909-2914.

[66] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, "*Estimation with applications to Tracking and Navigation, Theory, Algorithms and Software*," John Wiley & Sons, New York, NY, 2001.

[67] P. Boning and S Dubowsky, "Identification of Actuation Efforts using limited Sensory Information for Space Robots," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Orlando, Florida, May 2006, pp. 3873-3878.

[68] S. Matsumoto, S. Jacobsen, S. Dubowsky, and Y. Ohkami, "Approach Planning and Guidance for Uncontrolled Rotating Satellite Capture Considering Collision Avoidance," *Proceedings of the 7th International Symposium on Artificial Intelligence and Robotics & Automation in Space (i-SAIRAS)*, Nara, Japan, 2003.

[69] M. Caccia et al, "Acoustic Motion Estimation and Control for an Unmanned Underwater Vehicle in a Structured Environment," *International Federation of Automatic Control (IFAC) Control Engineering Practice*, Volume 6, Number 5, May 1998, pp. 661-670.

[70] R. K. Yedavalli and A. G. Sparks, " Satellite Formation Flying Control Design Based on Hybrid Control System Stability Analysis," *Proceedings of the American Control Conference,* June 2000, pp. 2210-2214.

[71] S. Starin, R. K. Yedavalli, A. G. Sparks, "Design of a LQR controller of reduced inputs for multiple spacecraft formation flying," *Proceedings of the American Control Conference,* Arlington, VA, June 2001, pp. 1327-1332.

[72] W. Kang, A. Sparks, and S. Banda, "Multi-Satellite Formation and Reconfiguration," *Proceedings of the American Control Conference,* Chicago, IL, June 2000, pp. 379-383.

[73] B. Acikmese, F. Y. Hadaegh, D. P. Schard, and S. R. Ploen, "A Formulation of Stability for Spacecraft Formations," Proceedings of the American Control Conference, Minneapolis, MN, June 2006, pp. 3581-3587.

[74] R. G. Sanfelice, M. J. Messina, S. E. Tuna, and A. R. Teel, "Robust Hybrid Controllers for Continuous-time Systems with Applications to Obstacle Avoidance and Regulation to Disconnected Set of Points," *Proceedings of the American Control Conferences*, June 2006, pp. 3352-3357.

[75] S. McCamish and M. Romano, "Simulations of Relative Multiple-Spacecraft Dynamics and Control by MATLAB-Simulink and Satellite Tool Kit," Proceedings of the AIAA Modeling and Simulation Technologies, Conference, Hilton Head, SC, August 2007 (submitted for publication).

[76] T. Carrico et al, "Proximity Operations for Space Situational Awareness," Advanced Maui Optical and Space Surveillance Technologies Conference, Wailea Maui, Hawaii, September 2006.

[77] S. M. Endres, "*Simulation and Emulation of the Space Networking Environment*," MS Thesis, Department of Electrical Engineering and Computer Science, Case Western Reserve University, January 2005.

[78] E. D Silvia, A. L. Kelly, and R. H. Bishop, "Development of Guidance and Control Algorithms for Nanosatellite Rendezvous Applications," 30th Annual American Astronautical Society Guidance and Control Conference, Breckenridge, CO, February 2007.

[79] D. A. Friedman, "*Laboratory Experimentation of Autonomous Spacecraft Docking Using Cooperative Vision Navigation*," MS Thesis, Department of Mechanical and Astronautical Engineering, Naval Postgraduate School, December 2005.

[80] T. J. Shay, "Design and Fabrication of Planar Autonomous Spacecraft Simulator with Docking and Fluid Transfer Capability," MS Thesis, Department of Mechanical and Astronautical Engineering, Naval Postgraduate School, December 2005.

[81] M. Romano, D. A. Friedman, T. J. Shay, "Laboratory Experimentation of Autonomous Spacecraft Approach and Docking to a Collaborative Target," AIAA Journal of Spacecraft and Rockets, Vol. 43, No. 1, January-February. 2007.

[82] J. S. Hall and M. Romano, "A Novel Robotic Spacecraft Simulator with Mini-Control Moment Gyroscopes and Rotating Thrusters," *Proceedings of the 2007 IEEE/ASME International Conference on Advanced Intellegent Machatronics (AIM)*, Zurich, Switzerland, September 2007.

[83] J. S. Hall, "Design and Integration of a Three Degrees-of-Freedom Robotic Vehicle with Control Moment Gyro for the Autonomous Multi-Agent Physically Interacting Spacecraft (AMPHIS) Testbed," MS Thesis, Department of Mechanical and Astronautical Engineering, Naval Postgraduate School, 2006.

[84] S. Veres, N. Lincoln, and S. Gabriel, "Testbed for Satellite Formation Flying Under Ground Conditions," Proceedings of the European Control Conference, Kos, Greece, July 2007.

[85] M. T. Hagan, H. B. Demuth, and M. H. Beale, *Neural Network Design*, Boulder, Colorado, University of Colorado Bookstore, 1996.

[86] G. Yang, V. Kapila, and H. Wong, "Fuel Optimal Initialization of a Spacecraft Formation," *Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, Hawaii, December 2003, pp. 3591-3596.

[87] Maxstream, Xbee Chip, http://www.maxsteam.com/ [cited 20 April 2007].

[88] R. H. Bledsoe, "Knowledgeable Network Addressable Terminals Applied to the Mirco Morphing Air Land Vehicle," MS Thesis, Department of Electrical and Computer Engineering, Naval Postgraduate School, June 2007.

[89] R. Cristi et al, "Motion Estimation and Modeling of the Environment for Underwater Vehicles," *International Journal of Systems Science*, Volume 29, Number 10, May 1998, pp. 1135-1143.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Air Force Insitute of Technology (AFIT)
   Wright Patterson AFB, Dayton, Ohio

4. Marcello Romano
   Naval Postgraduate School
   Monterey, California

5. Xiaoping Yun
   Naval Postgraduate School
   Monterey, California

6. Roberto Cristi
   Naval Postgraduate School
   Monterey, California

7. Tri Ha
   Naval Postgraduate School
   Monterey, California

8. Ravi Vaidyanathan
   University of Southampton
   Southampton, United Kingdom