

**Design Methodologies for Simple Adaptive Controllers with
Applications to Spacecraft Proximity Operations**

by
Andriy Predmyrskyy

A thesis submitted to the Faculty of Graduate and Postdoctoral Affairs
in partial fulfillment of the requirements for the degree of

Master of Applied Science
in
Aerospace Engineering

Carleton University
Ottawa, Ontario

© 2021
Andriy Predmyrskyy

This work is dedicated to my father.
To my family and friends who support me endlessly.
To everyone who finds this work helpful.

Per aspera ad astra

Through hardships, to the stars

Abstract

Space debris in low-Earth Orbit is increasing year-on-year, with inaction threatening *Kessler Syndrome*, a point where debris collisions are self-sustaining and space launches are impossible. Conservative estimates suggest removing at least 10 pieces of large debris per year [1], with each piece of debris having different unknown mass properties. Conventional control techniques make it infeasible to manage the rendezvous, docking, and deorbit multiple pieces of debris every year. Advanced controllers, such as adaptive controllers which can sense and correct for deviations in systems with unknown or time-varying characteristics, are able to manage debris uncertainty without requiring costly or time-intensive design reformulations upon contact with each target. Simple adaptive control offers the ability to manage unknown or time-varying systems with guaranteed performance, and without intervention.

Simple adaptive control varies linear control gains until adequate system performance is achieved. The current work improves implementation of simple adaptive control through novel design heuristics, application of optimization, and disturbance accommodation. Techniques are experimentally verified, and tested on a multiple-input-multiple-output simulation of a spacecraft's attitude and position during spacecraft rendezvous, docking, and post-docking control.

Experimental results show that optimization is able to decrease the convergence time of a simple adaptive controller, and that disturbance compensation increases the model tracking of a simple adaptive controller. Design heuristics are able to provide a tangible method for determining simple adaptive control parameters. Furthermore, simulations show that simple adaptive control can uncouple unknown system dynamics, while improving the response. The provided work provides several methods and techniques to help designers implement simple adaptive control in physical systems, and improving those designs once they are implemented. Finally, several avenues for further research are proposed.

Acknowledgements

So many people and opportunities had to coincide to allow this work to take place, I cannot possibly thank them all. I want to thank you the reader, regardless of your background, for giving me the time to share my thoughts with you.

I cannot thank Dr. Steve Ulrich enough. Thank you for your understanding, your help, and your patience. You gave me the room I needed to study my passion, and more importantly, to make mistakes. You were always there with a suggestion for a paper, and I cannot count how many times our meetings made me go from feeling lost and unsure of myself to confident and ready to tackle challenges head on. I will never forget the way you said one of my results was “Powerful”.

No less important to my success is the support of Dr. Jeremy Dillon. As a friend and a mentor, your passion for learning opened me up to a world of possibilities. You made me understand that it does not take much to push the boundary of scientific understanding. I would never have convinced myself to take on the world of control systems theory if you had not shown me how simple these concepts could be, one blackboard at a time. I hope I can someday help someone the way you helped me.

To my friends and colleagues; each of you are infinitely appreciated. No small thanks goes to Kirk Hovell and Alex Crain, who have played a part in almost all SRCL work since they arrived. You each brought hours of troubleshooting down to minutes, and your openness to help and solve problems saved me immeasurable grief. The lab will suffer when you are gone. To my friends who were always open to airing my worries, being there when I needed a hand, and helping me to answer all my ridiculous questions. Adam Bisson, Justin Bourassa, I cannot wait to trade more ideas with you, and show you what I have done.

A big thanks goes out to Jeff Hough, whose been my research buddy through all of this. Getting to know you better over the course of the year, working through problems in and outside the classroom, and bouncing ideas back and forth has been a lifesaver. I hope to to see you some more in higher places!

To my family I give my unending admiration. I am certain it got tiring, listening

to me talk about things that could not possibly make sense. I am sure it was not easy, helping me write for months. For supporting me in doing what I love, thank you.

This research was financially supported in part by the Natural Sciences and Engineering Research Council of Canada, through an Alexander Graham Bell Canada Graduate Scholarship. Further financial support was provided by graduate student entrance scholarships and teaching assistant positions provided by the Mechanical and Aerospace Engineering Department.

Contents

Abstract	iii
Acknowledgements	iv
List of Tables	xi
List of Figures	xii
List of Symbols	xvii
Chapter 1 Introduction	1
1.1 Motivation	2
1.2 Problem Statement	8
1.3 Previous Work	10
1.3.1 Early Control Systems Research and Linear Analysis	11
1.3.2 Nonlinear Control Theory	12
1.3.3 Adaptive Control Theory	14
1.3.4 Optimal Control and Optimization Techniques	15
1.4 Thesis Objectives	18
1.5 Contributions	19
1.6 Organization	20
Chapter 2 Problem Statement and Underlying Theory	22
2.1 Introduction	22
2.2 Uncontrolled Target Acquisition and Control	24

2.2.1	Frames of Reference	24
2.2.2	Orbital Dynamics	26
2.2.3	Formation Flying Dynamics	30
2.2.4	Orientation and Position Dynamics for Controlled and Combined Spacecraft	34
2.2.5	Combined Target-Chaser System	35
2.3	Linear Dynamics and Controller Design	40
2.3.1	Continuous Linear Dynamics	41
2.3.2	Discrete Linear System Analysis	44
2.3.3	Example of Linear Analysis of Spacecraft Systems	46
2.3.4	Linear Controller Design and Implementation	47
2.4	Nonlinear System Analysis	53
2.5	Adaptive Control Design	60
2.5.1	Simple Adaptive Control Formulation	62
2.5.2	Negative Adaptation of Simple Adaptive Controllers	68
2.5.3	Model Matching Conditions	70
2.5.4	Disturbance Accommodating SAC	73
2.5.5	System Augmentation: SAC for Non-ASPR Systems	76
2.5.6	SAC Drawbacks	79
2.6	Optimization and Metaheuristics	80
2.6.1	Convex Optimization	82
2.6.2	Example Convex Optimization through Successive Approximation	91
2.6.3	Nonlinear Optimization	94
2.6.4	Metaheuristics	99
2.7	Chapter Summary	106
Chapter 3 Simple Adaptive Controller Implementations		109
3.1	Introduction	109
3.2	Nonlinear Optimization and Metaheuristics in SAC Design	109

3.2.1	Preliminary Metaheuristic Optimization of a SAC	110
3.3	Simple Adaptive Control Design and Heuristics	116
3.3.1	Ideal Model as Guidance	117
3.3.2	Ideal Models for Nonlinear Systems; Feasible Models	119
3.3.3	Adaptation Parameter Selection and Design Heuristics	123
3.3.4	Steady State Gains Under Noise and Signal Covariance	137
3.3.5	Proportional Adaptation as Quadratic or Boundary Control	141
3.4	SAC Disturbance Compensation in Feedforward Parallelized Plants	145
3.5	SAC Design Summary	149
3.6	Chapter Summary	153
 Chapter 4 Simple Adaptive Controller Design Experimental Veri-		
fication		155
4.1	Introduction	155
4.2	System Formulation	155
4.2.1	Experimental Apparatus	156
4.2.2	Software Implementation and Simulation	158
4.2.3	Nonlinear SAC Parameter Search	160
4.2.4	SAC Positional Tracking Experimental Setup and Controller Development	165
4.2.5	SAC Disturbance Compensation Experimental Setup	170
4.3	SAC Position Tracking Experimental Results	174
4.4	SAC Disturbance Compensation Experimental Results	185
4.5	Discussion of Optimized SAC Results	189
4.6	Discussion of Disturbance Compensation Results	192
4.7	Summary of Experimental Results	194
 Chapter 5 Implementation of SAC for Uncontrolled Spacecraft Cap-		
ture and Control		195
5.1	Introduction	195
5.2	Problem Formulation	196

5.2.1	Physical Parameters	196
5.2.2	Combined Target-Chaser System Parameters	199
5.3	Measurement and Command Signals	200
5.4	Path Planning	202
5.4.1	Rendezvous Path-Planning	202
5.4.2	Docking Phase	203
5.4.3	Post-Docked Behaviour	205
5.5	Linear Controller Design	205
5.6	SAC Design	207
5.6.1	Feedforward Parallelization	207
5.6.2	Feasible Model Development	208
5.6.3	MIMO SAC Design	211
5.7	Simulation and Results	220
5.8	Discussion of Simulation	220
Chapter 6	Conclusions	222
6.1	Thesis Summary	222
6.2	Significance of Work	225
6.2.1	Conference Proceedings	225
6.2.2	Journal Articles	226
6.3	Recommendations for Future Work	226
6.3.1	Spacecraft Implementation	226
6.3.2	Bursting Phenomena	226
6.3.3	Comparison or Equivalence of Augmentation Techniques	227
6.3.4	Guidance for Linear and Nonlinear Ideal Models	228
6.3.5	Extended Disturbance Accommodation	228
6.3.6	Kalman Filters: Incorporating Navigation Techniques	228
6.3.7	Real-Time Correlation Engines	228
Bibliography		230

Appendix A Essential Operations and Notation	236
A.1 Matrix Operations	236
A.2 Reference Frame Construction and Conversion	238
A.3 Classical Orbital Elements	240
A.4 Orbital Perturbations	243
Appendix B Linear Systems Theory and Results	245
B.1 Introduction	245
B.1.1 Time-Varying Signal	245
B.1.2 Linear Differential Equations and State-Space Representations	247
B.1.3 Solutions to Linear State-Space Systems	248
B.1.4 Laplace Transform and Linear System Analysis Tools	249
B.1.5 Block-Scheme Diagram Analysis	252
B.2 Discrete Systems	253
B.3 Additional Properties of Linear Systems	254
B.3.1 Minimal Realization	255
Appendix C Additional Linear System Analysis and Controller Design	257
C.1 Continuous Time SAC Stability Proof	257
C.2 Discrete Time SAC Stability Proof	261
C.3 Parallel Feedforward Compensation of non-ASPR systems	265

List of Tables

3.1	Metaheuristic Optimized SAC Parameters	113
4.1	Nonlinearly Optimized SAC Parameters	165
4.2	Nonlinear Search and Metaheuristic Optimized SAC Parameters	170
4.3	Disturbance Compensation Testing Gains	172
4.4	Linear-Quadratic Cost of SAC Designs	175

List of Figures

1.1	Infographic on LEO Deorbit Time	4
1.2	Hole on Canadarm Due to Debris.	5
1.3	LEO Spacecraft Launches Over Time.	6
1.4	Operations Required to Land the Curiosity Rover On Mars.	7
1.5	Spacecraft Robotics and Control Laboratory	19
2.1	Earth Centered Inertial Reference Frame	25
2.2	Local-Vertical-Local-Horizontal Reference Frame	27
2.3	Kepler’s Laws	27
2.4	Orbital Elements $\{a, e, i, \Omega, \omega, \theta\}$ Describe an Individual Orbit	29
2.5	Control Systems Before and After Docking	35
2.6	Docking Point Vectors For Target and Chaser Spacecraft	36
2.7	The First Photo From Himawari 9	41
2.8	Example of Spacecraft Guidance, Navigation, and Control Processing	46
2.9	Simplified Spacecraft Processing	47
2.10	Simplified Spacecraft Processing	47
2.11	Examples of Nonlinear Systems	54
2.12	Nonlinear Spring Example, with $x_0 = [10, 0]^T$	59
2.13	Block Scheme Diagram for a SAC	65
2.14	Correlated Signals Phase Shifted by ϕ . The Outer Product Integral Over a Wavelength is Small or Negative if Correlated Signals are Suf- ficiently Phase Shifted.	69
2.15	SAC with Feedforward Parallelization	78
2.16	Exaggerated example of Bursting Phenomena	80

2.17	Example Convex Function, With Convexity Constraint Highlighted	82
3.1	LQR and Manually Tuned SAC Performance	114
3.2	DE and SaDE Optimized SAC Performance	114
3.3	PSO and SPSO Optimized SAC Performance	115
3.4	Ideal Model Requires More Thrust Than System Can Deliver	120
3.5	Origin of the Feasible Model Architecture. A Satisfying Controller Ensures the Feasible Response Approaches the Ideal Response.	122
3.6	Feasible Model and Satisfying Controller for Example Thruster Satu- ration	123
3.7	Default SAC Simulation	126
3.8	Uninformed Estimate Results	132
3.9	Maximum Value Estimate Results	134
3.10	Maximum Value Estimate Iteration Results	134
3.11	Simulated Forcing Gain Estimate Results	136
3.12	Simulated Forcing Gain Estimate Iteration Results	137
3.13	Gains Under Noise and Appropriate σ Selection	140
3.14	Poorly Tuned System With and Without Proportional Adaptation	143
3.15	Default System Under Only Proportional Adaptation	145
3.16	Disturbance Acting on ASPR Plant	147
3.17	Disturbance Acting on Augmented ASPR Plant	147
3.18	Disturbance Acting on non-ASPR Plant	148
3.19	Disturbance Managed by SAC with Feedforward Parallelization	148
3.20	Equivalent Disturbance Compensation SAC Block Diagrams.	149
3.21	SAC Design with Feedforward Parallelization Using Stabilizing Con- troller $H(s)$	151
4.1	Three-Degree-Of-Freedom Frictionless Testbed, the SPOT.	157
4.2	The “Red” Test Platform, Beside “Black” Test Platform	158
4.3	Top Level of the SPOT Software	159

4.4	Feasible Model and Satisfying Controller for Example Thruster Saturation	168
4.5	Satisfying Controller in SPOT Simulation and Experimental Verification	169
4.6	Virtual Disturbances Act Similarly to Real Disturbances.	171
4.7	LQR Controller Position Graph and 3σ Standard Deviation	175
4.8	LQR Controller Force Output, with 3σ Standard Deviation	176
4.9	LQR Controller Position Error to the Command, with 3σ Standard Deviation	176
4.10	Manually Designed Controller Position Graph and 3σ Standard Deviation	177
4.11	Manually Designed Controller Force Output, with 3σ Standard Deviation	177
4.12	Manually Designed Controller Tracking Error, with 3σ Standard Deviation	178
4.13	Manually Designed Controller Position Error to the Command, with 3σ Standard Deviation	178
4.14	Nonlinearly Optimized Controller Position Graph and 3σ Standard Deviation	179
4.15	Nonlinearly Optimized Controller Force Output, with 3σ Standard Deviation	179
4.16	Nonlinearly Optimized Controller Tracking Error, with 3σ Standard Deviation	180
4.17	Nonlinearly Optimized Controller Position Error to the Command, with 3σ Standard Deviation	180
4.18	SaDE Optimized Controller Position Graph and 3σ Standard Deviation	181
4.19	SaDE Optimized Controller Force Output, with 3σ Standard Deviation	181
4.20	SaDE Optimized Controller Tracking Error, with 3σ Standard Deviation	182
4.21	SaDE Optimized Controller Position Error to the Command, with 3σ Standard Deviation	182
4.22	SPSO Optimized Controller Position Graph and 3σ Standard Deviation	183
4.23	SPSO Optimized Controller Force Output, with 3σ Standard Deviation	183
4.24	SPSO Optimized Controller Tracking Error, with 3σ Standard Deviation	184

4.25	SPSO Optimized Controller Position Error to the Command, with 3σ Standard Deviation	184
4.26	SAC Position Response Graph and 3σ Standard Deviation Under Linear Disturbance	185
4.27	SAC Force Output, with 3σ Standard Deviation Under Linear Disturbance	186
4.28	SAC Tracking Error, with 3σ Standard Deviation Under Linear Disturbance	186
4.29	SAC Position Error to the Command, with 3σ Standard Deviation Under Linear Disturbance	187
4.30	SAC Position Response Graph and 3σ Standard Deviation Under Linear Disturbance with Compensation	187
4.31	SAC Force Output, with 3σ Standard Deviation Under Linear Disturbance With Compensation	188
4.32	SAC Tracking Error to the Command, with 3σ Standard Deviation Under Linear Disturbance with Compensation	188
4.33	SAC Position Error, with 3σ Standard Deviation Under Linear Disturbance with Compensation	189
5.1	SPHERES Platforms on the ISS	197
5.2	LVLH Optimized Rendezvous Trajectory	203
5.3	Docking Command Mixing	203
5.4	Position and Orientation Commands After Docking	205
5.5	Position Satisfying Controller	209
5.6	Orientation Satisfying Controller	211
5.7	PID Position Response During Rendezvous and Docking	215
5.8	SAC Position Response During Rendezvous and Docking	215
5.9	PID Position Responses Post-Dock	216
5.10	PID Orientation Responses Post-Dock	217
5.11	SAC Position Responses Post-Dock	218

5.12 SAC Orientation Responses Post-Dock	219
A.1 Orbital Elements $\{a, e, i, \Omega, \omega, \theta\}$ Describe an Individual Orbit	240
A.2 Orbital Perturbation as compiled by Montenbruck and Gill in <i>Satellite Orbits: Models, Methods, and Applications</i> [2]	244
B.1 Examples of Linear System Analysis Tools For $H(s) = \frac{2s^2+5s+1}{s^2+2s+3}$	252
B.2 A Single Transfer Function $H(s)$ with Inputs $U(s)$ and Outputs $Y(s)$	252
B.3 The Block Diagram Operations of Multiplication, Addition, Feedback, and Shifting	253

Nomenclature

Roman Symbols

a	Semi-major axis of an ellipse [m] Placeholder variable name, Duffing equation coefficient Acceleration in Newtonian double-integrator examples
\mathbf{A}	State matrix, for LTI state-space systems
a_1 through a_5	Random distinct integers for SaDE search
\mathbf{A}_c	ASPR system state matrix after feedback
$\mathbf{A}_{c,d}$	Discrete ASPR system matrix after feedback
\mathbf{A}_d	State matrix, for discrete state-space system
$\mathbf{A}_{ff}(\mathbf{X}_{tot}(k))$	State-space approximation of formation flying equations given states \mathbf{X}_{tot} at iteration k
$A_{k,set}$	Set of active constraints at the k^{th} iteration
\mathbf{A}_{LQR}	State matrix, for LQR design
\mathbf{A}_m	State matrix, for SAC ideal model
$\mathbf{A}_{m,p}$	State matrix, for SAC position ideal model in uncontrolled spacecraft simulation
$\mathbf{A}_{m,\theta}$	State matrix, for SAC orientation ideal model in uncontrolled spacecraft simulation
\mathbf{A}_p	State matrix, for plant
$\mathbf{A}_{sc}(i)$	The i^{th} second order cone affine transformation matrix
\mathbf{A}_{SPOT}	State matrix approximation of SPOT hardware
\mathbf{A}_{SPOTd}	State matrix discrete approximation of SPOT hardware with feedforward parallelization
b	Placeholder variable name, Duffing equation coefficient
\mathbf{B}	Input matrix, for LTI state-space systems
\mathbf{b}_c	Equality constraint matrix of a convex optimization problem
b_d	Forced mutation dimension in DE and SaDE search

\mathbf{B}_d	Input matrix, for discrete state-space system
\mathbf{B}_{ff}	Input matrix for state-space approximation of formation flying equations
\mathbf{B}_{LQR}	Input matrix, for LQR design
\mathbf{B}_m	Input matrix, for SAC ideal model
$\mathbf{B}_{m,p}$	Input matrix, for SAC position ideal model in uncontrolled spacecraft simulation
$\mathbf{B}_{m,\theta}$	Input matrix, for SAC orientation ideal model in uncontrolled spacecraft simulation
\mathbf{B}_p	Input matrix, for plant
$\mathbf{b}_{sc}(i)$	The i^{th} second order cone translation matrix
\mathbf{B}_{SPOT}	Input matrix, approximation of SPOT hardware
\mathbf{B}_{SPOTd}	Input matrix, discrete approximation of SPOT hardware with feedforward parallelization
c	Placeholder variable name, Duffing equation coefficient
\mathbf{C}	Output matrix, for LTI state-space systems
\mathcal{C}	Set of second-order cones of a convex optimization problem
\mathcal{C}^*	Set of second-order cones of a convex optimization problem's Dual
$C(s)$	SISO Controller transfer function
\mathbf{C}_d	Output matrix, for discrete state-space system
\mathbf{C}_{ff}	State constraint for trajectory optimization problem
$\mathbf{C}_{\mathcal{LI}}$	Rotation from ECI to LVLH
\mathbf{C}_{LQR}	Output matrix, for LQR design
\mathbf{C}_m	Output matrix, for SAC ideal model
$\mathbf{C}_{m,p}$	Output matrix, for SAC position ideal model in uncontrolled spacecraft simulation
$\mathbf{C}_{m,\theta}$	Output matrix, for SAC orientation ideal model in uncontrolled spacecraft simulation
\mathbf{C}_p	Output matrix, for plant

\mathbf{C}_{SPOT}	Output matrix, approximation of SPOT hardware
\mathbf{C}_{SPOTd}	Output matrix, discrete approximation of SPOT hardware with feedforward parallelization
\mathbf{C}_Δ	Full sensor blending augmentation matrix
$\Delta\mathbf{C}$	Additional sensor blending matrix
d	Placeholder variable name, Duffing equation coefficient
\mathbf{D}	Feedthrough matrix, for LTI state-space systems
\mathcal{D}	Lyapunov function input space
$\mathbf{d}_c(i)$	The i^{th} second order cone inequality affine translation matrix
d_d	Dimension index in DE, PSO, SaDE, or SPSO search
\mathbf{D}_d	Feedthrough matrix, for discrete state-space system
\mathbf{d}_{ff}	State constraint for trajectory optimization problem
\hat{d}_k	Step direction after solving QP subproblem in SQP
\mathbf{D}_m	Feedthrough matrix, for SAC ideal model
$\mathbf{D}_{m,p}$	Feedthrough matrix, SAC position ideal model in uncontrolled spacecraft simulation
$\mathbf{D}_{m,\theta}$	Feedthrough matrix, SAC orientation ideal model in uncontrolled spacecraft simulation
d_n	Variable of optimization for QP subproblem in SQP
\mathbf{D}_p	Feedthrough matrix, for plant
e	The Euler-Mascheroni constant
	Eccentricity of an ellipse [unitless]
\mathbf{e}	MIMO error of PID controller
\mathcal{E}	The invariant set of the Lyapunov function
$\mathbf{E}(s)$	Laplace domain error signal
\mathbf{e}_c	Unit vector of second order cone directions
$\mathbf{e}_{max,y}$	Maximum expected tracking error \mathbf{e}_y
\mathbf{e}_n	Tracking error with noise
\mathbf{e}_p	Column matrix of position errors in uncontrolled spacecraft simulation, PID controller

\mathbf{e}_x	Ideal state error
\mathbf{e}_y	Error from plant response to model response, called the tracking error
\mathbf{e}_{ya}	Augmented tracking error
$\mathbf{e}_{y,b}$	Boundary tracking error for proportional adaptation estimate
\mathbf{e}_θ	Column matrix of orientation errors in uncontrolled spacecraft simulation, PID controller
F	Force in Newtonian double-integrator examples [N] Scaling factor in a DE and SaDE search
$f(\bullet)$	Example function, objective function
\mathbf{f}_c	Affine objective of a convex optimization problem
\mathbf{F}_c	Chaser spacecraft force actuation column matrix [N] Affine constraint matrix of a convex optimization problem
\mathcal{F}_c	Chaser spacecraft body-fixed reference frame vectrix
\mathbf{F}_d	Linear disturbance model state matrix
$\hat{\mathbf{F}}_d$	Estimated linear disturbance model state matrix
F_i	Scaling factor for the i^{th} agent in a SaDE search
$\vec{\mathcal{F}}_I$	ECI reference frame vectrix
$f_{k,m}$	the number of failures for the m^{th} trial vector generation function at the k^{th} iteration in a SaDE search
$\vec{\mathcal{F}}_L$	LVLH reference frame vectrix
\vec{F}_t	Gravitational force vector on target spacecraft [N]
$\vec{\mathcal{F}}_t$	Target spacecraft body-fixed reference frame vectrix
F_x	Chaser spacecraft LVLH force x component [N]
F_y	Chaser spacecraft LVLH force y component [N]
F_z	Chaser spacecraft LVLH force z component [N]
\mathbb{G}	Universal gravitational constant [$\text{m}^3\text{kg}^{-1}\text{s}^{-2}$]
\mathcal{G}	The largest invariant set of \mathcal{E}
$g(i)$	The i^{th} second order cone magnitude
$\mathbf{G}(s)$	Matrix of transfer functions of the plant

$\mathbf{G}_a(s)$	Augmented ASPR transfer function of the plant
$\mathbf{G}_c(s)$	Closed loop stable plant with feedback control of $\mathbf{H}(s)$
g_i	i^{th} inequality constraints for optimization problem
G_i	Secondary scaling factor for the i^{th} agent in a SaDE search
h	Timestep interval [sec]
$\mathbf{H}(s)$	Matrix of transfer functions of the stabilizing controller
$H^{-1}(s)$	Feedforward parallelization, SISO transfer function
$\mathbf{H}^{-1}(s)$	Feedforward parallelization, matrix of transfer functions
h_i	i^{th} equality constraint for optimization problem
H_k	The Hessian of a nonlinear programming problem at the k^{th} iteration
H_{tot}	Example control system SISO transfer function
i	Inclination of an orbit [rad]
	Counting integer
\mathbf{I}_K	Gain impulse
$\hat{\mathbf{I}}_K$	Estimate of gain impulse
i_q	The first basic quaternion
$\vec{\mathcal{I}}_x$	ECI reference frame x-direction unit vector
$\vec{\mathcal{I}}_y$	ECI reference frame y-direction unit vector
$\vec{\mathcal{I}}_z$	ECI reference frame z-direction unit vector
j	Initializing parameter for convex optimization problem
\mathbf{J}	Moment of inertia [kg m ²]
\mathbf{J}_c	Chaser spacecraft Moment of Inertia, [kg m ²]
$\mathbf{J}_{cmb,c}$	Moment of inertia contribution to combined system due to chaser moment of inertia [kg m ²]
$\mathbf{J}_{cmb,t}$	Moment of inertia contribution to combined system due to target moment of inertia [kg m ²]
j_i	Cost associated to \mathbf{p}_i
j_q	The second basic quaternion
\mathbf{J}_t	Target spacecraft moment of inertia, [kg m ²]

$\mathbf{J}_{worstcase}$	Worst case moment of inertia [kg m ²]
k	Iteration number
\mathbf{K}	Compacted SAC gain
\mathbb{K}	Constant, mean of expectation of X with Y in covariance example
\mathbf{K}^*	Ideal gain matrix for SAC
\mathbf{K}_0	Initial gain matrix for SAC
\mathbf{K}_b	Proportional adaptation parameter heuristic, boundary gain
K_d	Derivative gain scalar
\mathbf{K}_d	Proportional gain matrix for PID controller
$\mathbf{K}_{d,p}$	Matrix of derivative position gains in uncontrolled spacecraft simulation, PID controller
$\mathbf{K}_{d,sat}$	Feasible model satisfying controller derivative gain
$\mathbf{K}_{d,s,p}$	Derivative gain for satisfying controller for position feasible model in uncontrolled spacecraft simulation
$\mathbf{K}_{d,s,\theta}$	derivative gain matrix for satisfying controller for orientation feasible model in uncontrolled spacecraft simulation
$\mathbf{K}_{d,\theta}$	Matrix of derivative orientation gains in uncontrolled spacecraft simulation, PID controller
\mathbf{K}_e	SAC feedback gain matrix
\mathbf{K}_{eI}	SAC integral gain due to feedback error
\mathbf{K}_{eP}	SAC proportional gain due to feedback error
$\tilde{\mathbf{K}}_e$	Feedback gain for ASPR system
\mathbf{K}_F	Forcing gain
$\hat{\mathbf{K}}_F$	Forcing gain estimate
\mathbf{K}_i	Integral gain matrix for PID controller
\mathbf{K}_I	Integral SAC gains in nonlinear optimization
\mathbf{K}_{LQR}	Gain matrix for LQR designs
\mathbf{K}_m	Maximum acceptable linear SAC gains during implementation
\mathbf{K}_{max}	Maximum feedback gain matrix that stabilizes a plant

$\mathbf{K}_{max,e}$	Maximum acceptable feedback gain matrix for adaptation parameter heuristic
$\mathbf{K}_{max,u}$	Maximum acceptable command gain matrix for adaptation parameter heuristic
$\mathbf{K}_{max,x}$	Maximum acceptable model gain matrix for adaptation parameter heuristic
\mathbf{K}_{min}	Minimum feedback gain matrix that stabilizes a plant
K_p	Proportional gain scalar
\mathbf{K}_p	Proportional gain matrix for PID controller
$\mathbf{K}_{p,p}$	Matrix of proportional position gains in uncontrolled spacecraft simulation, PID controller
$\mathbf{K}_{p,sat}$	Feasible model satisfying controller proportional gain
$\mathbf{K}_{p,s,p}$	Proportional gain for satisfying controller for position feasible model in uncontrolled spacecraft simulation
$\mathbf{K}_{p,s,\theta}$	Proportional gain matrix for satisfying controller for orientation feasible model in uncontrolled spacecraft simulation
$\mathbf{K}_{p,\theta}$	Matrix of proportional orientation gains in uncontrolled spacecraft simulation, PID controller
k_q	The third basic quaternion
\mathbf{K}_{ss}	Steady state feedback error under noise (without noise amplification)
\mathbf{K}_u	SAC input command gain matrix
\mathbf{K}_u^*	SAC matching condition ideal input gain
\mathbf{K}_{uI}	SAC integral gain due to input command
\mathbf{K}_{uP}	SAC proportional gain due to input command
\mathbf{K}_x	SAC ideal model gain matrix
\mathbf{K}_x^*	SAC matching condition ideal state gain
\mathbf{K}_{xI}	SAC integral gain due to ideal model states
\mathbf{K}_{xP}	SAC proportional gain due to ideal model states
\mathbf{K}_{XY}	Example covariance matrix of X with Y

\mathbf{K}_z	SAC disturbance accommodation gain
\mathbf{K}_{zI}	SAC disturbance accommodation Integral gain
\mathbf{K}_{zP}	SAC disturbance accommodation Proportional gain
\mathbf{L}	Discrete ASPR stability condition matrix
L_P	Learning period in SaDE search
$\vec{\mathcal{L}}_x$	LVLH reference frame x-direction unit vector
$\vec{\mathcal{L}}_y$	LVLH reference frame y-direction unit vector
$\vec{\mathcal{L}}_z$	LVLH reference frame z-direction unit vector
m	Number of outputs to a state-space system
	Number of inputs and outputs to a SAC
	Mass [kg]
	The index of a trial vector generation function in a SaDE search
M	Number of SaDE trial vector generation function strategies
\mathbf{M}	Composite matrix of plant state-space matrices
	Mutated vector in a DE and SaDE search
$M(s)$	SISO model transfer function
m_b	Mass of parent body [kg]
m_c	The number of inequality constraints of a convex optimization problem
	Mass of chaser spacecraft [kg]
m_{cmb}	Combined system mass[kg]
M_{d_d}	Entry of the d_d^{th} dimension of the mutation vector \mathbf{M}
m_e	Number of constraints in the active set
m_n	Number of inequality constraints for nonlinear optimization problem
m_t	Mass of target spacecraft [kg]
n	System order for state-space system, ideal model system order
\mathbf{N}	Inverse of composite matrix of plant state-space matrices
\mathcal{N}	Optional control-state weight matrix for LQR design
\mathbf{N}_{11} through \mathbf{N}_{22}	Block diagonal decomposition of \mathbf{N}

n_c	Number of optimization parameters
O	Objective value
p	Number of inputs to a state-space system
p	Number of poles of a transfer function
\mathbf{P}	Linear symmetric Lyapunov stability matrix
p_c	The number of affine constraints of a convex optimization problem
\mathbf{P}_d	Discrete ASPR symmetric Lyapunov stability matrix
p_{d_d}	Entry of the d_d^{th} dimension of the position \mathbf{p}_i being considered
\mathbf{p}_g	Swarm's best known position in PSO and SPSO searches
\mathbf{p}_i	Best known parameters for the i^{th} agent in a swarm for a metaheuristic search
\mathbf{P}_{LQR}	Symmetric Lyapunov stability matrix for LQR design
p_n	Variable of optimization for QP subproblem using active set methods in SQP
p_n	Number of equality constraints for nonlinear optimization problem
\mathbf{p}_p	Agent's best known position in PSO and SPSO searches
q	Order of ideal model
\mathbf{q}	Best known parameters for a swarm in a metaheuristic search
\mathbf{Q}	Linear positive definite Lyapunov stability matrix
\mathcal{Q}	State/error weight matrix for LQR design
$q(i)$	Dimension of the i^{th} second order cone constraint
\mathbf{q}_e	Error quaternion
\mathbf{q}_{cmd}	Orientation quaternion command
$\mathbf{q}_{cmd,d}$	Orientation commands while docking
$\mathbf{q}_{cmd,r}$	Orientation commands during rendezvous
\mathbf{q}_m	Measured quaternion
\mathbf{q}_t	Target orientation quaternion
\mathbf{r}	Reference signal
\mathbf{R}	Rotation matrix [unitless]

\mathcal{R}	Control activation weight matrix for LQR design
\mathbb{R}	Euclidean real-number space
\vec{r}	Radius (position) [m]
\mathbf{r}_a	Docking axis for trajectory optimization problem[m]
$\mathbf{r}_{a,c}$	Chaser spacecraft docking point, distance from center of mass [m]
$\mathbf{r}_{a,cmb}$	Distance between individual target and chaser centers of mass, while docked [m]
$\mathbf{r}_{a,t}$	Chaser spacecraft docking point, distance from center of mass [m]
\mathbf{r}_b	Boundary reference signal for proportional adaptation estimate
r_c	LVLH composite radius, variable simplification [m]
\mathbf{r}_c	Chaser position column matrix [m]
\vec{r}_c	Radius vector of the chaser spacecraft [m]
\mathbf{r}_{c0}	Initial position of the chaser spacecraft[m]
\mathbf{r}_{cmb}	Combined system position column matrix [m]
\mathbf{r}_{cmd}	Position command[m]
$\mathbf{r}_{cmd,r}$	Optimized rendezvous position command in LVLH[m]
r_g	Random value for PSO and SPSO search
\mathbf{r}_{max}	Column matrix of maximum reference signal values
r_p	Random value for PSO and SPSO search
\mathbf{r}_t	Target position column matrix [m]
\vec{r}_t	Radius vector of the target spacecraft [m]
\mathbf{r}_{t0}	Initial position of the target spacecraft
s	The Laplace domain complex variable
S	Success fraction of each trial vector generation function in a SaDE search
\mathbf{s}	Dual problem cone variable for a convex optimization problem
\mathbb{S}	Circular set
\mathbf{S}	Convex optimization dual parameter arrowhead matrix
\mathbf{S}_{11}	SAC matching condition first matrix

\mathbf{S}_{12}	SAC matching condition second matrix
$s_{k,m}$	The number of successes for the m^{th} trial vector generation function at the k^{th} iteration in a SaDE search
t	Time [sec], parameter input
\mathbf{T}	Torque [Nm]
\mathbf{T}_c	Chaser spacecraft torque actuation column matrix [Nm]
t_d	Docking instant, time since mission start[sec]
\mathbf{T}_d	Disturbance torque due to combined system offset thrusters [Nm]
t_d^+	Instant after docking, time since mission start[sec]
t_d^-	Instant before docking, time since mission start[sec]
t_f	Final time, for formation flying optimization, simulations [sec], Final iteration for nonlinear optimization
T_{max}	Max thrust in formation flying trajectory optimization
\mathbf{u}	Input command, for LTI state-space systems
\mathbf{u}	Input commands to formation flying trajectory optimization state-space approximation
\mathbf{U}	Schur decomposition matrix of eigenvectors
\mathcal{U}	The set of states with Lyapunov function derivative equal to 0
$\hat{\mathbf{u}}_d$	Estimated disturbance output
$\mathbf{U}_{1,1}$ through $\mathbf{U}_{2,2}$	Schur decomposition eigenvector block diagonal decomposition element
u_c	Input command to a controller
\mathbf{u}_c	Input command, for SAC ideal model and update equations
$\mathbf{U}_c(s)$	Laplace domain command output
\mathbf{u}_d	Input for discrete state-space system Disturbance output
$\hat{\mathbf{u}}_d$	SAC control output for disturbance accommodation
u_{d_d}	Dimension d_d of trial vector \mathbf{u}_i
\mathbf{u}_i	Trial vector for i^{th} agent of a DE or SaDE search
\mathbf{u}_p	Input control power to plant from controller output

u_p	Scalar input to plant
\mathbf{u}_p	Inputs to the plant
$\mathbf{u}_{p,b}$	Output command desired at boundary for proportional adaptation estimate
$\mathbf{u}_{p,p}$	Thrust command to the plant for position control in uncontrolled spacecraft simulation
$\mathbf{u}_{p,\theta}$	Torque command to the plant for orientation control in uncontrolled spacecraft simulation
v	Order of linear disturbance model
V	Lyapunov function
\mathcal{V}	Objective function unrestricted input space
\vec{v}	Velocity [m/s]
\mathcal{V}^*	The convex set of the dual of an optimization problem
\mathbf{v}_c	Velocity column matrix of the chaser spacecraft [m/s]
\mathbf{V}_c	Column matrix of chaser spacecraft velocities[m/s]
\vec{v}_c	Velocity vector of the chaser spacecraft [m/s]
\mathbf{v}_{c0}	Initial velocity of the chaser spacecraft[m/s]
\mathbf{v}_{cmb}	Combined system velocity column matrix [m/s]
v_{ex}	Exhaust velocity of spacecraft thrusters[m/s]
\mathbf{v}_i	Velocity of the i^{th} agent in a metaheuristic search
\mathbf{v}_t	Velocity column matrix of the target spacecraft [m/s]
\vec{v}_t	Velocity vector of the target spacecraft [m/s]
\mathbf{v}_{t0}	Initial velocity of the target spacecraft[m/s]
\mathbf{w}	Random noise variable
\mathbf{W}	Discrete ASPR condition matrix
x	State in nonlinear system dynamics examples
X	Example signal correlated with Y
\mathbf{X}	Convex Optimization primal parameter arrowhead matrix
x^*	Parameters that are an optima of a nonlinear optimization problem

$\bar{\mathbf{x}}$	Displacement of combined system center of mass from chaser center of mass [m]
\mathbf{x}_0	Initial parameters for convex optimization problem
x_c	Convex input parameter
\mathbf{x}_c	Parameters of an optimization problem
\mathbf{X}_c	Formation flying trajectory optimization chaser spacecraft states
x_{cmd}	x position command
\mathbf{x}_d	System states for discrete state-space system
\mathbf{x}_e	Equilibrium point states
\bar{x}_{frac}	Distance of combined center of mass along $\mathbf{r}_{a,cmb}$, as fraction of one [unitless]
x_i	Parameters for the i^{th} agent in a swarm for a metaheuristic search
x_k	Nonlinear optimization problem parameters at the k^{th} iteration
x_{k+1}	Parameter update after solving QP subproblem in SQP
x_L	LVLH x-direction [m]
\mathbf{x}_m^*	Ideal plant states
x_p	Position, Newtonian double-integrator examples, LQR design [m]
\mathbf{x}_p	States for the plant
\mathbf{X}_t	Formation flying trajectory optimization target spacecraft states
\mathbf{X}_{tot}	Formation flying trajectory optimization problem states
y	System output Output in Newtonian double-integrator state-space system example
Y	Example signal correlated with X
\mathbf{y}	Output, for LTI state-space systems Measured MIMO system output

\mathbf{y}_0	Initial parameters for dual of convex optimization problem
\mathbf{y}_a	Augmented output of a system, now ASPR
y_c	Second Convex Input parameter
\mathbf{y}_c	The set of parameters for the dual of a convex optimization problem
y_{cmd}	y position command
y_L	LVLH y-direction [m]
y_m	Output of ideal model, output of feasible model if present
\mathbf{y}_m	Output, for SAC ideal model
y_m^*	Output of ideal model sent to satisfying controller
\mathbf{y}_p	Output of the plant
\mathbf{y}_p^*	Ideal plant output
\mathbf{y}_s	Slack variable of a convex optimization problem
z	Number of zeros of a transfer function
\mathbf{Z}	LQR design Associated Hermitian Matrix
\mathbb{Z}	Domain of integer numbers
z_0	Taylor expansion approximation coefficient of $e^{z_c(t)}$
z_c	Transformed chaser spacecraft mass
\mathbf{z}_d	Linear disturbance model states
$\hat{\mathbf{z}}_d$	Estimate of disturbance generator states
$\hat{\mathbf{z}}_d$	Estimated Linear disturbance model states
Z_k	Nullspace of the active set at iteration k
z_L	LVLH z-direction [m]
z_m	Mixing variable during docking

Greek Symbols

α	Cone half-angle for trajectory optimization docking cone constraint
α_n	Step size in SQP, in addition to \hat{d}_k
β	Cone half-angle for trajectory optimization thrust decontamination constraint

γ	Scaling parameter for central path solution of convex optimization problem
Γ_{eI}	SAC integral adaptation rate due to feedback error
Γ_{eP}	SAC proportional adaptation rate due to feedback error
Γ_I	Integral adaptation parameter
$\hat{\Gamma}_I$	Integral adaptation parameter estimate through heuristic
Γ_P	Proportional adaptation parameter
Γ_{uI}	SAC integral adaptation Rate due to input command
Γ_{uP}	SAC proportional adaptation Rate due to input command
Γ_{xI}	SAC integral adaptation Rate due to ideal model states
Γ_{xI1}	Integral adaptation parameter for the first model state
Γ_{xI2}	Integral adaptation parameter for the second model state
Γ_{xP}	SAC proportional adaptation rate due to ideal model states
Γ_{xP1}	Proportional adaptation parameter for the first model state
Γ_{xP2}	Proportional adaptation parameter for the second model state
Γ_{zI}	SAC disturbance accomodation integral adaptation
Γ_{zP}	SAC disturbance accomodation proportional adaptation
δ	Scaling parameter
δ	Variance of random variable \mathbf{w}
δ_d	Crossover rate in a DE and SaDE search
δ_i	Random variable for crossover ratio for the i^{th} agent in a SaDE search
δ_m	Crossover ratio median in a SaDE search
ϵ	A small number
ζ	Damping coefficient of a second order system
θ	True anomaly [rad]
θ	Pitch Euler angle [rad]
Θ_d	Linear disturbance output matrix
$\hat{\Theta}_d$	Estimated Linear disturbance output matrix
Θ_c	Ideal model orientation angle input command
θ_e	Pitch Euler angle error[rad]
$\Theta_{e,m}$	Ideal model orientation angle error

Θ_m	Ideal model orientation angle response
κ	Feasibility variable for homogenous and self-dual realization of convex optimization problem
κ_0	Initial infeasibility parameter of convex optimization problem
λ	Lagrangian multiplier
$\boldsymbol{\lambda}$	Schur decomposition upper triangular matrix containing eigenvalues
λ_c	Convex mixing coefficient
λ_i	i^{th} Lagrangian multiplier for a nonlinear programming problem
μ_{\oplus}	Standard gravitational parameter of the Earth [km^3/s^2]
$\boldsymbol{\mu}_0$	Variable of initialization for convex optimization problem
π	Archimedes' Constant
$\boldsymbol{\rho}$	Relative displacement column matrix [m]
$\vec{\rho}$	Relative displacement vector [m]
$\dot{\boldsymbol{\rho}}$	Relative velocity column matrix [m/s]
$\dot{\vec{\rho}}$	Relative velocity vector [m/s]
σ	Standard deviation
$\boldsymbol{\sigma}$	Sigma modification matrix for SAC
σ_c	nondimensionalized chaser thrust magnitude
$\boldsymbol{\sigma}_z$	SAC disturbance accommodation degradation parameter
ς	Safety factor
τ	Variable of integration
$\boldsymbol{\tau}$	Unboundedness variable for homogenous and self-dual realization of convex optimization problem
$\boldsymbol{\tau}_0$	Initial unboundedness parameter of convex optimization problem
$\boldsymbol{\tau}_c$	Normalized chaser thrust
ϕ	Phase shift of a signal[rad] Roll Euler angle [rad]
ϕ_e	Roll Euler angle error[rad]
ϕ_g	Attraction to swarm's best known position in PSO and SPSO searches
ϕ_p	Attraction to agent's best known position in PSO and SPSO searches

ψ	Yaw Euler angle [rad]
ψ_e	Yaw Euler angle error[rad]
ω	Argument of perigee[rad], Duffing equation angular velocity [rad/s]
$\boldsymbol{\omega}$	Angular velocity column matrix[rad/s]
Ω	Right-ascension of the ascending node (RAAN)[rad]
$\boldsymbol{\omega}_c$	Chaser spacecraft angular velocity column matrix [rad/s]
Ω_c	Ideal model angular velocity input command
$\boldsymbol{\omega}_{c0}$	Initial angular velocity of the chaser spacecraft
$\boldsymbol{\omega}_{cmb}$	Combined system angular velocity [rad/s]
ω_g	Damping factor in PSO and SPSO searches
$\boldsymbol{\omega}_{\mathcal{LI}}$	Angular velocity of LVLH frame with respect to ECI frame [rad/s]
ω_n	Natural frequency of a second order system[rad/s]
$\boldsymbol{\omega}_t$	Target spacecraft angular velocity column matrix [rad/s]
$\boldsymbol{\omega}_{t0}$	initial angular velocity of the target spacecraft[rad/s]
ω_x	x component of angular velocity [rad/s]
ω_y	y component of angular velocity [rad/s]
ω_z	z component of angular velocity [rad/s]

Superscripts

*	Ideal value of a signal
+	Value immediately after nominal value
	Positive set only
–	Value immediately before nominal value

Subscripts

c	As of, or pertaining to the chaser spacecraft
d_d	The d^{th} dimension of a variable
i	The i^{th} element of a list

I	As of, or pertaining to the integral SAC adaptation
k	The k^{th} iteration of a process
m	As of, or pertaining to the reference model
p	As of, or pertaining to the plant
P	As of, or pertaining to the proportional SAC adaptation
t	As of, or pertaining to the target spacecraft
\oplus	As of, or pertaining to the Earth
0	An initial value of a variable, such as initial conditions

Other Symbols and Operators

A	Bolded variable, Matrix form of a variable
A	Unbolded variable, Scalar form of a variable
$ \bullet $	Absolute value operation
$\ \bullet\ _2$	Euclidean 2-norm
$\dot{\bullet}$	First time derivative, $\frac{d}{dt}\bullet$
$\ddot{\bullet}$	Second Time derivative
\bullet^\times	Skew Symmetric operator of a column matrix
\mathbf{I}_n	Square identity matrix of size $\mathbb{R}^{n \times n}$
$\mathbf{0}_n$	Square matrix of zero elements of size $\mathbb{R}^{n \times n}$
$\mathbf{0}$	Square matrix of zero elements of appropriate size to adjacent block-diagonal matrix elements
\bullet^T	Matrix transpose
$\{\bullet, \bullet, \bullet, \bullet\}$	System composed of matrices contained in curly brackets
$\bullet(s)$	Denoting the value of a variable in the Laplace domain
$\bullet(t)$	Denoting the value of a variable in the time domain
$\int_a^b dc$	Integral from a to b under the variable of integration c
\bullet^{-1}	Inverse of a matrix, inverse of a quaternion
\bullet^*	Conjugate of a quaternion, see related superscript
$\frac{d}{dt}\bullet$	Time derivative

$\nabla \bullet$	The gradient operator. Subscripts denote the variables over which the gradient is being taken
$\text{diag}(\bullet, \bullet)$	Block diagonal matrix form of the input matrices
$\max(\bullet)$	The maximum value of a signal
$\sin(\bullet)$	Sinus function
$\cos(\bullet)$	Cosinus function
$\ln(\bullet)$	Natural logarithm function
$\Sigma_{a=b}^c \bullet$	Sum from $a = b$ to c of the expression

Acronyms, Abbreviations, and Initialisms

3D	Three-Dimensional
3DOF	Three-Degrees-of-Freedom
6DOF	Six-Degrees-of-Freedom
ADCS	Attitude Determination and Control Subsystem
ASPR	Almost Strictly Positive Real
COM	Center of Mass
DE	Differential Evolution
ECI	Earth-Centered Inertial
ESA	European Space Agency
GNC	Guidance, Navigation, and Control
ISS	International Space Station
JAXA	Japanese Aerospace eXploration Agency
JPL	Jet Propulsion Laboratory
LED	Light Emitting Diode
LEO	Low-Earth Orbit
LMI	Linear Matrix Inequality
LQR	Linear Quadratic Regulator
LTI	Linear Time-Invariant
LVLH	Local-Vertical-Local-Horizontal

MGTF	Microgravity Test Facility
MIMO	Multiple-Input-Multiple-Output
MIT	Michigan Institute of Technology
MRAC	Model Reference Adaptive Control
NASA	National Aeronautics and Space Administration
PID	Proportional-Integral-Derivative
PSO	Particle Swarm Optimization
RAAN	Right Ascension of the Ascending Node
SAC	Simple Adaptive Control(ler)
SaDE	Strategy-adaptive Differential Evolution
SISO	Single-Input-Single-Output
SPOT	Spacecraft Proximity Operations Testbed
SPR	Strictly Positive Real
SPSO	Selection Particle Swarm Optimization
SQP	Sequential Quadratic Programming
SRCL	Spacecraft Robotics and Control Laboratory
YALMIP	Yet Another LMI Parser

Chapter 1

Introduction

The exploitation of space is a stunning human accomplishment in its own right; not only is our planet uniquely suited to life, it just so happens that the chemical and electrical potential on it are sufficient to exit the atmosphere and accelerate objects into orbit. Orbit itself can seem magical, objects speed over the surface of the planet fast enough for the horizon to fall away; spacecraft manage to miss the ground.

Behind the weird quirks of physics that enable spaceflight lies hidden potential. Scientific observation, communication, and exploration are all possible by placing just the right hardware in just the right spot: rovers have allowed humanity to glimpse the surfaces of other planets; radar has penetrated the thick veil of Venus; and commercial spacecraft have brought accessible internet to every reach of the Earth. Even more is soon to come: the James Webb space telescope will see beyond space dust and deep into the universe's past; sample return missions from Mars will uncover the secrets of Earth's barren cousin; and the development of space infrastructure will expand in-orbit operations and further lower launch costs, opening space to a wider audience.

Behind all the hopes of advancements is one reality: humans cannot control each of these space systems individually. It is expensive and dangerous to send humans on space missions. Future advances demand a significant presence in space. Megaconstellations, such as Starlink, require hundreds or thousands of spacecraft to be useful¹. In-orbit assembly or servicing increases the number of spacecraft to be managed with the number of spacecraft being built or serviced. Any form of sample-return mission,

¹<https://spacenews.com/air-force-laying-groundwork-for-future-military-use-of-commercial-megaconstellations/>

such as the European Space Agency’s (ESA) upcoming mission², requires more autonomous spacecraft as the amount of material being returned increases. Several key technologies rely on the ability to increase autonomy as operations expand.

This work is to tackle the issue of spacecraft autonomy, through design analysis and validation of a specific adaptive control technique: simple adaptive control. Adaptive control is a key candidate for improving spacecraft autonomy, since large distances and slow communication make it difficult to service or modify space systems after they have been deployed. It is important that adaptive control is well understood to allow for spacecraft autonomy to be increased in lock step with the capabilities that we require from our space systems.

1.1 Motivation

The future of spacecraft operations lies in the ability to remotely and autonomously adapt control systems to ensure they achieve the required performance despite varying operating conditions. Spacecraft operations include activities of high scientific and economic value that also require a high degree of autonomy or self-reliance. Currently, performance of these systems is maximized through human supervision, slow operation, or extensive *a priori* knowledge of the individual operation.

Remote vehicle operations on other bodies such as the curiosity rover [3], spacecraft repair missions such as the Hubble telescope servicing mission [4] and active debris de-orbiting missions such as ESA’s proposed ClearSpace-1 [5] involve human presence or significant active oversight from ground crews. Each of these tasks requires additional effort to manage a simple underlying problem: remote systems are not sufficiently autonomous for them to be able to manage problems without intervention.

The curiosity rover must perform operations slowly on the surface of Mars, partly due to communication lag, and partly because it cannot be trusted to complete activities safely without observation. For ESA’s ClearSpace-1 system, variability in the debris being deorbited will require ground crews to determine how to manage that

²https://www.esa.int/Science_Exploration/Human_and_Robotic_Exploration/Exploration/Mars_sample_return

debris, slowing down operations. To service the Hubble telescope astronauts had to be put at risk because robotic systems could not complete the repairs themselves. In each of these operations efficiency, success, and sometimes human safety is affected by the need for consistent human interaction.

The need for autonomous systems is only growing in the space industry. Debris deorbiting, sample-return missions, and spacecraft repair are just three examples of tasks with inherent variability that must be done repetitively to be of value. These repetitive and variable tasks must furthermore be completed in the high-risk environment of spaceflight, where a single error can and has resulted in millions of dollars of losses³. Even worse, though, it is becoming clear that improper debris management in Low-Earth Orbit (LEO) is threatening catastrophe, and must be immediately managed.

Spacecraft can be present in LEO for decades after they leave service [6]. Figure 1.1 demonstrates how long a single spacecraft might take to deorbit from a given altitude above the Earth. The time to deorbit increases drastically with small increases in altitude, and it is not unusual for a spacecraft to take decades to deorbit.

Many pieces of debris are present in LEO, each going at several kilometers per second relative to one another. Collisions between uncontrolled objects are high-energy affairs producing a shower of additional particles all travelling at similar speeds and inundating the LEO environment with debris like the break shot in a game of pool. Collisions produces debris that are ready to create more collisions, making the process more like runaway nuclear fission than snooker. With enough debris in orbit a runaway reaction occurs, covering the LEO environment with tiny pieces of high-speed junk that leaves space inaccessible to ground launches, which has been dubbed *Kessler syndrome* [7]. What lies on the other side of Kessler syndrome is the inability to access space for possibly decades due to an impenetrable cloud of high-velocity debris, making a successful space ascent comparable to dodging a hailstorm made of bullets. Kessler syndrome is by no means an impossibility.

An example of how quickly debris can be made was unceremoniously created by the Chinese government, which intentionally destroyed the decommissioned Fengyun-1C

³<https://priceconomics.com/the-typo-that-destroyed-a-space-shuttle/>

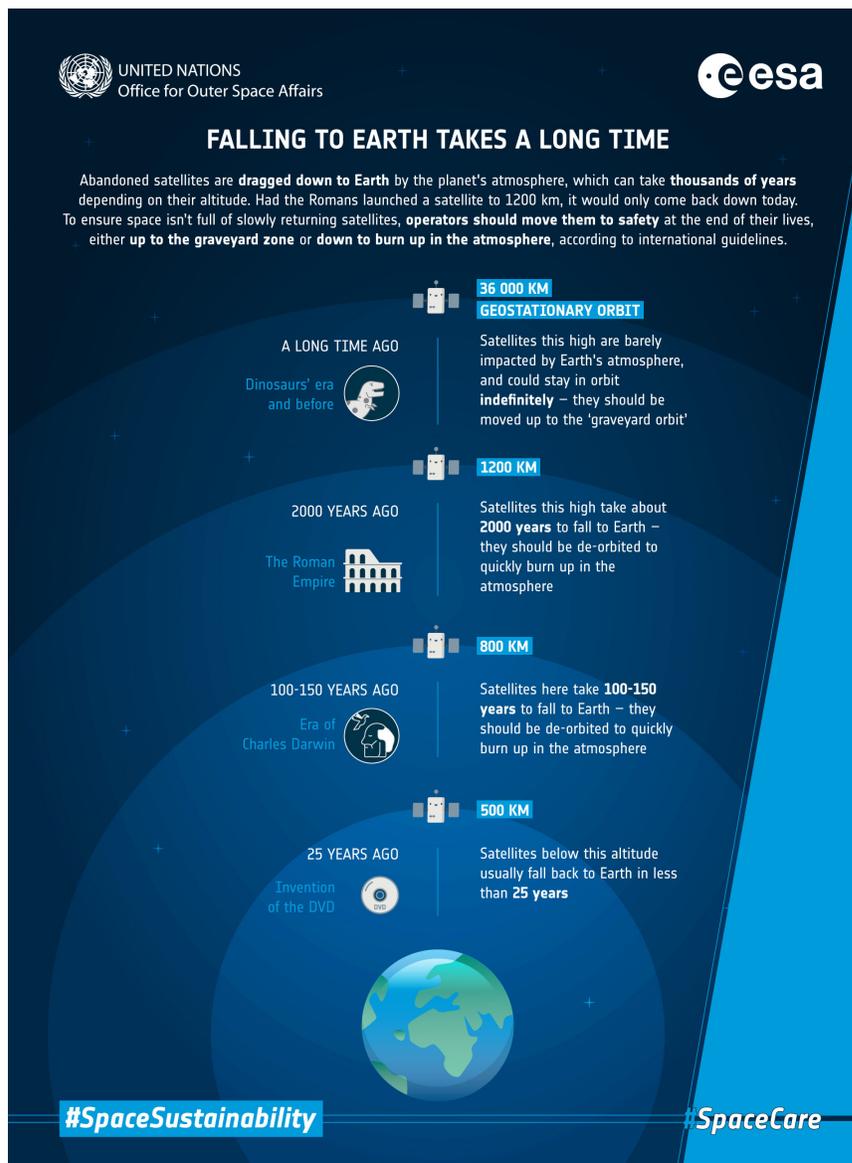


Figure 1.1: Infographic on LEO Deorbit Time. Courtesy ESA.⁴

spacecraft in 2007 to produce over 2000 pieces of new debris over 10 cm in diameter. The incident accounted for an immediate increase of 30% in the number of previously catalogued LEO debris; Fengyun-1C did in one instant 30% of what took 50 years previously to produce. Almost all of the debris created by the Fengyun-1C event is still in orbit to this day. Projections suggest that it will take between 15 and 40 years for half of the Fengyun-1C debris to deorbit, and likely over a century for less than

10% of the original debris to remain [8]. During the entirety of each piece of debris' lifetime there is opportunity to collide with other pieces of debris, and create more pollution in the LEO environment.

But debris events are not always as abrupt or notable as the Fengyun-1C event, and the rate of collisions is already increasing. In early June 2021, the Canadarm-2 aboard the International Space Station (ISS) was hit by a piece of debris leaving a 5 mm hole shown in Fig. 1.2. The debris that hit Canadarm-2 may have been as small as a piece of dust, with the large relative velocities between the ISS enough to create the hole.



Figure 1.2: Hole on Canadarm Due to Debris. Image courtesy Canadian Space Agency.⁵

The amount of spacecraft in orbit is only increasing. Figure 1.3 shows the amount of spacecraft in LEO, with the number of commercial spacecraft dramatically increasing in 2020 and 2021. Commercial launches have not been without incident, either. In April 2021 a OneWeb and Space-X spacecraft in LEO had a “conjunction event”, in which both spacecraft were at close separation to one another⁶. What counts as a “conjunction event”, the separation of the spacecraft, and the measures taken by each company are still in discussion, however it is clear that such an event would not have

⁴https://www.esa.int/var/esa/storage/images/esa_multimedia/images/2021/02/falling_to_earth_takes_a_long_time/23161085-4-eng-GB/Falling_to_Earth_takes_a_long_time_pillars.jpg

⁵<https://www.asc-csa.gc.ca/eng/iss/news.asp>

⁶<https://docs.fcc.gov/public/attachments/FCC-21-48A1.pdf>

occurred without the increase in commercial spacecraft in LEO. Furthermore, such “conjunction events” will only become more common as greater volumes of traffic are present in the LEO environment.

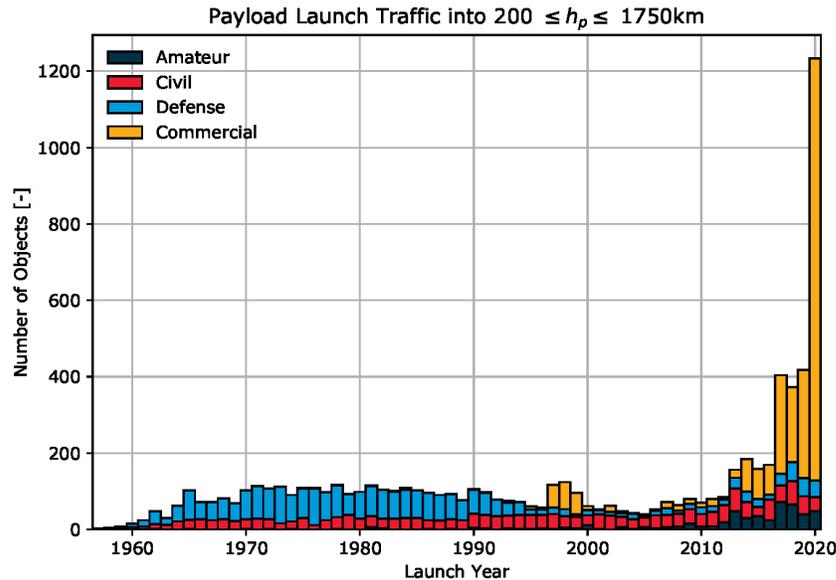


Figure 1.3: LEO Spacecraft Launches Over Time. Image courtesy ESA.⁷

Several space agencies such as the ESA and The National Aeronautics and Space Association (NASA), have already realized the severity of this problem and have made calls for technology to de-orbiting problematic debris. The most promising of these technologies are active techniques [9], which require a chaser spacecraft that identifies, rendezvous, and then manipulates a target in order to remove it from orbit. Analyses of orbital debris cleanup scenarios also favour scenarios where active deorbiting is done on few large objects as opposed to many small objects [1], with cleanup of as few as five to twenty pieces of large debris every year resulting in severely improved outcomes for LEO pollution. To make space safe, it will be necessary to deorbit multiple pieces of debris each year. Each piece of debris will be of uncertain size, mass, orientation, and angular velocity, all of which will need to be managed repeatedly and successfully. The increase in autonomy necessary to manage these systems with unknown or poorly defined properties necessitates an increase in spacecraft autonomy.

⁷http://www.esa.int/Safety_Security/Space_Debris/ESA_s_Space_Environment_Report_2021

on the Curiosity rover mission. To survive Mars entry, dubbed “the seven minutes of terror” and demonstrated in Fig. 1.4, every detail of the entry, descent, and release had to be meticulously calculated and planned; if any element of the descent was misunderstood, the whole mission would be lost. And yet, despite the available resources, an unexpected event on the surface of Mars led to the Curiosity rover becoming trapped in small sand dunes that rendered its wheels ineffective. The only solution available was to implement adaptive wheel speed control to improve Curiosity’s autonomy, helping it traverse the unpredictable dunes of the red planet [10]. The long delay present in Mars-Earth communications means that there is no method to ensure Curiosity does not become stuck during terrain traversal other than Curiosity itself ensuring it cannot become stuck.

As it becomes mandatory for either a large number of space missions to occur concurrently, for spacecraft to make decisions before they can be relayed to a ground-station, or to maximize performance of uncertain systems, adaptive control strategies will be crucial in ensuring missions can continue to occur despite minimal interaction.

The application of adaptability to spacecraft operations is more than a theoretical desire, rather, it is a justifiable necessity.

1.2 Problem Statement

The development of the space sector is strongly contingent on the achievable guidance, navigation, and control objectives. As control improves, the cost of individual missions decreases, the amount of missions that can be safely done increases, and the scope of missions that can be performed increases. On the contrary, the ramifications for not improving efficiency and accuracy rapidly enough may lead to irreparable consequences, ranging from the loss of any single vehicle, to the loss of accessibility to space travel altogether. In order to meet and exceed the required pace of progress it will be necessary to develop and mature novel methods of autonomous spacecraft operation.

To complete a mission autonomously a spacecraft must be able to transition between an initial state, to the final completed state. What that transition looks like varies from system to system, and it is the controller that synthesizes the actions

necessary to reach the final objective. Control systems are therefore directly responsible for a spacecraft's ability to meet the accuracy and efficiency requirements of the mission. The control system design onboard a spacecraft is the direct interface between the mission objectives, and the actions that meet those objectives.

Control architectures must operate under disturbances, measurement noise, and the system response. An example of a system response can be felt when the accelerator is pushed in a car; the placement of the accelerator affects the fuel flow to the engine, which accelerates the vehicle. The time it takes for changes in the accelerator placement to manifest as acceleration of the vehicle is the system response of the acceleration to the accelerator input. A controller manages the output of that system. You, as a driver, make use of the accelerator to control the vehicle acceleration, and thereby control the position of the car. Control of the car position can then be used to quickly and effectively change the position of the passenger, which is done whenever the driver decides to transition their position from one location to another. The human driver is naturally autonomous, managing to control the car accurately despite changes in its mass, the friction of the tires, the condition of the road, and any other unavoidable variability.

The foundation of automatic control systems are linear feedback controllers, which are used in a wide variety of systems [11]. Linear feedback controllers take a measurement of the system state at every available instant then use a combination of linear operations to produce a final control to be used by the system to achieve the desired response. Returning to the analogy of a car, a driver might monitor their velocity, depressing the vehicle accelerator if they are going too slow, and letting go of the accelerator if they are going too fast.

Since the controller is responsible for the ability of a spacecraft to achieve a command, improvements to controller formulations are the best method to improve the autonomy of spacecraft more generally. One prominent candidate to improve the autonomy and performance of controllers in the face of uncertainty are the class of control architectures known as *adaptive controllers*. Among these, one of the architectures with the most promise for minimizing the amount of onboard computation while achieving a high-level of performance and generality, is the Simple Adaptive

Controller, or SAC.

The SAC is a direct adaptive control architecture that can be easily implemented in a wide variety of systems. The controller is made up of an ideal model, control gains, and adaptation mechanism that all work together to ensure that the required system performance can be achieved and maintained despite significant uncertainty in the system. The ideal model dictates to the controller what performance is desired, and the adaptation mechanism changes the control gains to make the actual system response approach the ideal response. The major benefit of SAC is the simplicity of the adaptation mechanism, along with the large variety of systems for which it is applicable. Many questions still remain on how to best design such controllers, such as what parameters are necessary to ensure optimal performance in a given environment and to mitigate adverse effects.

This thesis aims to improve existing knowledge on the design, implementation, and performance of the SAC architecture. In particular, the applicability of adaptive control in improving autonomy of spacecraft proximity operations is explored. This text will also act as a collection and summary of existing knowledge on the design of SACs for any system, while improving knowledge on design and implementation wherever possible. In particular, heuristics and design tools are presented to allow the designer to aim for reliable operation, optimal performance, and/or compensate for disturbances. Where possible, performance of the techniques are compared and contrasted with linear design techniques for reference.

1.3 Previous Work

The control of systems has been an important topic for any designer who wishes to ensure a particular system acts in the way the designer intends it to. In the right light, everything from shoes to submarines can be considered a control system, mostly due to the abstract applicability of system analysis. Simple systems allow for simple designs to achieve simple goals. Many complex systems also require control, however, and control theory has expanded to consider systems of arbitrary size and complexity.

Linear control theory, optimal control theory, robust control theory, nonlinear control theory, and many more branches and subbranches of control theory exist to

tackle any and all aspects of the control problem. How do you design a system that can go from A to B, and how can it be done well, reliably, or repeatedly?

The current research on adaptive control techniques relies on the accomplishments of linear control analysis, nonlinear control analysis, and optimal control all in turn. In order to improve adaptive control formulations for spacecraft control it is therefore necessary to thoroughly understand linear and nonlinear control theory, as well as optimization techniques.

1.3.1 Early Control Systems Research and Linear Analysis

One of the first examples of a control system was the use of governors for the regulation of engine speed in 1868 by James Clark Maxwell [12]. Further research led to the development of the Routh-Hurwitz stability criterion [13], which when paired with the Laplace transform provided a simple way to show if a given linear system was stable. Control theory development stagnated until it was required once more to improve the reliability of flight controls in aircraft during World War II [14], where inherent stability of aircraft began to be traded for increased manoeuvrability in combat.

In the late 1950s and early 1960s research into control theory began to explode; countless small advancements, such as the development of discrete control theory for implementation in computers [15] and descriptions of the dynamics of servomechanical systems [14] had handily made it into textbooks on the subject of control [16] just in time for use in the space-race. Since then, control theory has entered every aspect of modern life: even simple mechanical thermostats employ a hysteresis controller in the form of a circuit that is closed when it is too cold and open when it is too hot [17]. Discrete control is present in modern automobiles not only to control the fuel flow to the engine [18] but as disturbance rejection when climbing hills, speed control when activating cruise, and in upcoming attempts for automated driving [19]. Most pressing is the requirement for control systems in space hardware, where human oversight is difficult to impossible, resource efficiency is paramount, system dynamics are unfamiliar or unstable, and failure can mean millions of dollars in damages to hardware or irreparable loss of human life.

The theory of linear systems has matured significantly and produced many insights

into the solutions and trajectories of certain classes of control system, particularly linear time-invariant (LTI) systems. The work of early pioneers is now collected in textbooks on the subject of linear analysis [20], from which we are able to quantify almost every behaviour of a linear system.

The wealth of information on the behaviour of linear systems allows for strong understanding of controller designs that keep the system linear. One common example of a linear controller is the Proportional-Integral-Derivative (PID) controller. The controller error, its derivative, and its integral are multiplied by values chosen by the designer and supplemented by ample design tools to achieve desirable system response characteristics. A plurality of control systems are managed by PID controllers or some small modification thereupon. So ubiquitous is the PID controller that Dr. Tariq Samad, who participated in a survey of the use of control techniques in industry, when asked said “98% of loops are controlled by PID controllers” and found that 100% of surveyed control engineers agree that PID controllers have had a high-impact on control theory [11]. Due to their ubiquity and ease of use, PID controllers have been analysed and built upon for decades.

The maturation of various optimization theories alongside linear system analysis lead to the development of the first optimal linear controller, namely, the Linear Quadratic Regulator (LQR) controller. By weighing the importance of error minimization and control activation, controllers can be produced that strike a balance between both.

1.3.2 Nonlinear Control Theory

Nonlinear control is a broad term used to characterize any system that does not meet the linear conditions. The term “nonlinear” itself is unnecessarily obtuse; Stanislaw Ulam, a nonlinear scientist once said⁹:

Using a term like nonlinear science is like referring to the bulk of zoology as the study of non-elephant animals

Almost nothing is linear. In fact, all physical systems will eventually encounter nonlinearities, either in the form of saturation values, slew rates, delays, or other

⁹<https://physics.sciences.ncsu.edu/research/nonlinear-dynamics/>

physical constraints. Even rigid body kinematics become nonlinear when approaching the speed of light.

Linear analysis tools allowed designers to create controllers with good performance for some nonlinear systems. Linear approximations necessitated that controllers were only applied to situations where those approximations were valid. In some cases, the applicable linear regions were very small. For example, aircraft control depends nonlinearly on the velocity of the aircraft, the angle of attack, the thrust of the engines, and the angle of sideslip, all of which vary the aerodynamic properties of the aircraft [21]. Linear controller design could create controllers that functioned well for small variations of these design variables, such as in trim and level flight, while performing poorly or unstably for other variations. Early attempts to increase the control envelope involved creating multiple linear controllers that would change their gains depending on the current state of the aircraft, called *gain-scheduled control* [22]. Gain-scheduled control requires significant knowledge of the system dynamics in order to identify different linear regions, design controllers for those regions, and change controllers during operation.

Other classes of nonlinear control problem involve measurable or well-defined nonlinearities that can be quantified. Attitude dynamics in rotating systems are nonlinear, however those dynamics are well understood through theoretical analysis and depend on measurable properties such as moment of inertia and angular velocity. Additionally, the dynamics created by these nonlinearities act as their own separate terms in the equations of motion, and as such are not coupled with other system dynamics. Well understood, uncoupled, and observable nonlinear dynamics such as these can be actively compensated through a technique known as *feedback linearization* which treats them as known disturbances. If the disturbance can be counteracted then the control system reverts to a linear control problem. Feedback linearization is commonly added to aircraft control formulations to compensate for nonlinear rotation dynamics and the nonlinear aerodynamic coefficients where possible [23].

The seminal work on nonlinear control theory would come in 1890s Soviet Russia, undertaken by A. M. Lyapunov, although his work would not be properly understood

by Western academia until the 1960s. Lyapunov gave valuable insights into the stability of nonlinear systems, an incredible accomplishment. Lyapunov's use of precise notions of stability and trajectories of "energies" in nonlinear system gave rise to Lyapunov stability theorem [24]. The introduction of Lyapunov's stability theorem was revolutionary; it was now possible to prove that some classes of nonlinear controller are stable.

The usefulness of Lyapunov's stability theorem to prove the stability of nonlinear systems cannot be understated. Use of the theorem allowed for the creation of innumerable nonlinear control laws for a wide variety of systems. *Backstepping* and *sliding mode* controllers are two popular examples [25], but the diversity and variety of controllers made using Lyapunov's stability theorem are legion, each with their own small variations for the control problem at hand. The vast amount of nonlinear controller designs present are therefore too numerous to cover here.

1.3.3 Adaptive Control Theory

Adaptive control theory sprang from the need to manage systems with time-varying or unknown processes, or where *a priori* information on the system evolution or initial state is scarce. Gain-scheduled controllers can be considered an early example of an adaptive controllers, but are characterized by their reliance on *a priori* information. Other styles of adaptive controller emerged to combat scenarios where operating conditions could not be explicitly predicted or where variability between controlled systems within the design space was unpredictable.

Adaptive control theory is broadly split into two classes. The first class consists of changes applied to the controller in response to measurable variations in the plant. When certain parameters, such as the mass of the control system, can be determined through measurement or sensing techniques the controller architecture can be indirectly changed through those measurements to produce a new controller. Controllers that adapted their architecture based on these indirect measurement of plant characteristics are classified as *indirect adaptive controllers*. Indirect adaptive control can be simple for some systems and complex for others. For example, aircraft control power depends on the airspeed over the wings. The ability for an aircraft to measure

the airspeed is an example of a simple indirect adaptation mechanism. Since the airspeed is known, the control power of each surface can be deduced, and the deflection commanded by the controller can be changed with measurements of the airspeed.

For many systems, however, measurements of the system parameters are not available. In order to control linear systems with unknown or time-varying characteristics an adaptive control system must be able to change itself directly from how the system responds, without any parameter measurements or estimates. Such systems are called *direct adaptive controllers* since their architecture is adapted directly from measurements of the system outputs, and not indirectly through observations of the system parameters.

One of the most prominent direct adaptive techniques is the Model-Reference Adaptive Controller (MRAC) which allows for designers to ensure that the unknown plant approaches a known reference response given that the plant and model are of the same order [26]. Further development of adaptive control techniques allowed for the creation of the SAC architecture that not only ensured stability for all cases in which MRAC was applicable, but also extended to a wider variety of systems. Significant work by Barkana and other throughout the 1980s to the present day has expanded the number and type of system that can be controlled by SAC [27].

Since then, SAC has been used to improve or match the performance of previous control methodologies while also allowing for the inclusion of adaptation to systems that previously lacked it. Ulrich and deLafontaine [28] showed that the technique was able to outperform previous control models for an uncertain descent through the atmosphere of Mars, even under the presence of large variations in the re-entry conditions through Monte Carlo simulation. Prabhakar et al. [29] applies SAC to a quadcopter, and shows that the SAC formulation can also be used to manage linear disturbances. Rusnak et al. [30] was able to show that SAC could be applied to a missile autopilot system to improve performance.

1.3.4 Optimal Control and Optimization Techniques

The cost of space travel places a large emphasis on minimizing resource usage since every gram of fuel necessary to complete a mission can contribute significantly to

launch costs. Since minimizing resource usage is important for lowering mission costs, and minimizing error is one of the explicit goals of control, engineers have turned to optimization techniques to get the best of both worlds. Control engineers have incorporated traditional optimization analysis and techniques, along with more approximate methods to optimize their controller formulations.

The study of optimality comprises its own mathematical field, for which various techniques, theories, and definitions have been used to determine optimal controllers for specific classes of problems. The major advantage of the class of *convex optimization* problems is the presence of a single global minimum, that is to say there is only one optimal solution across the entire parameter space which can be easily found. Solutions to convex problems are guaranteed to be the ideal solution for the system [31].

Non-convex problems can be much harder to manage. In non-convex problems it may be possible to show that any algorithmically determined solutions are better than the surrounding solutions, however it can be substantially more difficult to determine if the combination of parameters found are the global minimum or if it is simply the smallest value found so far, called a local minimum, without exploring the entirety of the search space.

It remains difficult to optimize control for nonlinear plants or for nonlinear controllers [32]. Nonlinear optimization problems, much like nonlinear control systems, are a large and unwavering body of problems that cannot be easily solved using standard methods. Nonlinear optimization techniques attempt to tackle these problems. The *Sequential Quadratic Programming* (SQP) technique [31], can be leveraged in nonlinear optimization problems to identify local minima.

Problems for which the shape of the cost function throughout the search space is not well defined, or where significant nonlinearities are present cannot be easily solved by standard optimization techniques. Problems for which normal optimization techniques perform poorly are known as hard optimization problems. Techniques known as *metaheuristics* are used to solve hard optimization problems which provide valuable estimates of optimal solutions while minimizing computational requirements and broadly searching the design space [33].

When very little information is available, or a wide search space must be parsed, population-based metaheuristics are able to determine solutions using numerous “agents” that test and wander the search space. Each agent is composed of a vector of the design parameters to be varied. The target design vectors are then tested using a cost function to determine the cost of that set of parameters. Information is exchanged between the agents to determine the next locations to test. Metaheuristic searches counter the complexity of the problem through stochastic sampling. By tasking multiple agents with finding strong solutions instead of trying to solve for a single optimal solution, the complexity of the problem is sidestepped at the cost of uncertainty of the solution optimality.

A large variety of population based metaheuristics have been developed, with many of them doing so in analogy to natural systems. It can be difficult to quantify the efficacy of a given metaheuristic, since each technique uses a different method to strike a balance between exploration and exploitation of the problem. Standard test batteries exist to determine how a new metaheuristic can be compared to previously established metaheuristics [34]. Of the common metaheuristics, particle swarm optimization (PSO) developed by Kennedy and Eberhart in 1995 [35], and differential evolution (DE) developed by Storn and Price in 1997 [36] are among the most popular. Further modifications of these techniques also exist, such as Strategy-adaptive Differential Evolution (SaDE) [37] and Selection Particle Swarm Optimization (SPSO) [34].

Metaheuristics have seen widespread adoption in fields where hard optimization problems exist. Particle swarm optimization has been used for controller tuning, vehicle routing, and mechanism design [38] and many more applications. Similarly, DE has seen wide adoption in the field of electrical engineering [39], and has been used for model identification [40].

In addition to the development of the linearly optimal LQR controller, a great deal of work has been placed into leveraging mathematical optimization for controllers [41]. Convex optimization, nonlinear optimization, and metaheuristic optimization have all been leveraged in the past to create some notion of optimal controller. The most well-developed of these are the various classes of *finite-horizon optimal control* techniques,

in which a model of the system dynamics is used in tandem with optimization methods to produce a control power time-history that minimizes the cost over a set time period, the finite-horizon. These techniques tend to require computational power and substantial *a priori* knowledge on the system dynamics which reduce their applicability to space systems with limited on-board computation that contend with unknown dynamic variability. Furthermore, very little has been done to optimize adaptive controllers. Takagi et al. [36] applied metaheuristic optimization to SAC, with the optimized SAC improving on the performance of manually tuned efforts.

1.4 Thesis Objectives

This thesis aims to test and refine SAC design techniques more generally, which are then applied specifically to proximity spacecraft operations. Design techniques are validated in simulation, as well as through experiments in Carleton’s spacecraft robotics and control laboratory (SRCL).

Theoretical results are collected and expanded to determine useful SAC design heuristics. By collecting, understanding, and developing the various existing design elements for SAC it will be possible for designers to quickly and easily create SAC formulations that result in improved system performance and safety.

Design optimization of SAC is considered for the spacecraft trajectory tracking problem, and tested using the spacecraft robotics and control laboratory’s three-degree-of-freedom (3DOF) laboratory, called the Spacecraft Proximity Operations Testbed (SPOT). Figure 1.5 shows the SRCL SPOT in operation. Nonlinear optimization tools are applied alongside a linear system approximation to develop optimal control parameters for SAC, which are then compared and contrasted with parameters developed by metaheuristic optimization techniques with the SRCL’s nonlinear simulation. The performance of both nonlinear and metaheuristic optimization techniques are compared and contrasted. Control of a spacecraft in proximity operations is verified through experiments for each of the controller designs. The performance of SAC under disturbance is also compared alongside disturbance compensation.

Finally, the performance of a SAC is explored for the rendezvous, docking, and post-dock transport of an uncontrolled target in simulation. The control problem

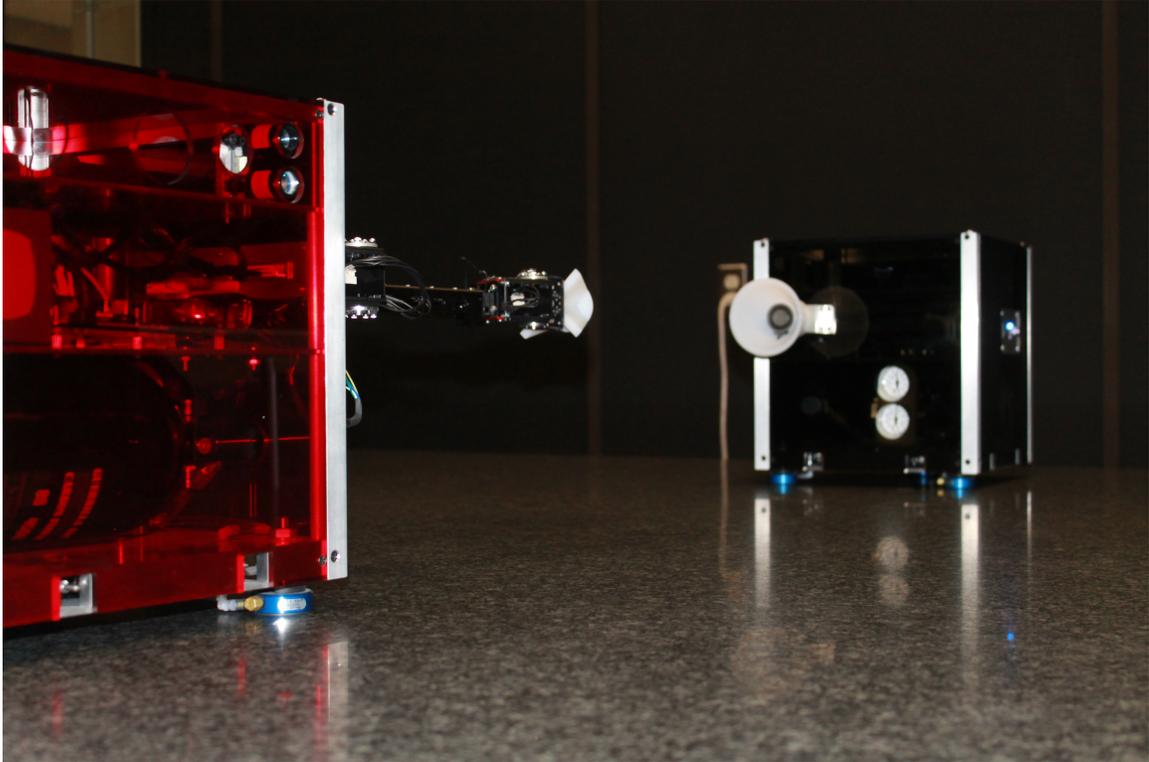


Figure 1.5: Spacecraft Robotics and Control Laboratory

includes six degrees of freedom (6DOF), namely control of the 3D position and orientation of the spacecraft throughout the three phases of flight. Adaptive control is able to improve on linear controller performance while compensating for unknown system cross coupling.

1.5 Contributions

Several contributions to the design workflow, implementation, and optimality of simple adaptive controllers are presented.

- Previously discovered elements of simple adaptive controller design have been collected for easy reference.
- Novel SAC design heuristics are presented to simplify selection of design parameters and bound output gains.

- Novel SAC design heuristics are used to develop a SAC design for the SPOT, which is then experimentally verified.
- The development of optimization scripts for SAC design, namely sequential quadratic programming, particle swarm optimization, selection particle swarm optimization, differential evolution, and strategy-adaptive differential evolution optimization scripts. The performance of each of these optimization techniques for determining optimal SAC designs is experimentally verified and compared on a 3DOF spacecraft testbed.
- An adaptive disturbance compensation component utilizing the SAC adaptation mechanism is developed for the spacecraft trajectory problem and experimentally verified.
- The development of a simulation environment that models LEO spacecraft motion, as well as rendezvous, docking, and post-dock control of an uncontrolled target. The simulation includes unavoidable cross-coupling between the orientation and position response.
- The design of a SAC architecture for rendezvous, docking, and transportation of an uncontrolled target spacecraft tumbling about a single axis.

1.6 Organization

This work is divided into multiple chapters for ease of reading. The concepts required to understand the later chapters of the text are presented early and developed until the final chapter, which presents a fully designed and implemented SAC for a spacecraft entrusted with autonomous completion of the rendezvous, docking, and post-dock transport of an uncontrolled spacecraft.

Chapter 2 presents background theory for all of the SAC design topics covered in the text. First, the equations of relative and inertial motion of spacecraft in orbit are presented. Linear control theory and nonlinear control theory are developed before describing the update equations for a SAC design. Previously understood design elements of SAC are collected, including stability proofs, augmentation of non-ASPR

systems, and ideal matching conditions. The equations for linear and optimal controllers are presented to be compared and contrasted with the adaptive control efforts. The equations for linear disturbance compensation in SAC are presented. Finally, the elements of optimization theory used for trajectory generation and optimal controller design are outlined.

Chapter 3 relates the various theoretical developments in SAC design. Optimal parameter selections for a SAC are found through several search techniques. Novel design heuristics are developed and demonstrated for SAC, alongside considerations of performance of SAC under linear disturbances. Finally, design best-practices for the implementation of a SAC in simulation and real systems are summarized for future design efforts.

Chapter 4 experimentally verifies SAC optimal design techniques and disturbance compensation using the SRCL's SPOT. The performance of three optimization techniques and their determined controllers are compared and contrasted. The performance of the SAC disturbance compensation component under disturbances of known frequency is compared to the performance without compensation. The results of optimization and disturbance compensation are discussed.

Chapter 5 presents a possible future implementation of SAC through the example of a 6DOF simulation of the rendezvous, docking, and post-dock transport of an uncontrolled target spacecraft of unknown mass and moment of inertia. A sufficient linear controller is created to control the system. Design principles outlined in Chapter 3 are used to develop a SAC that is able to manage strong uncertainty in the plant. Performance of the linear and adaptive control efforts are compared.

Chapter 6 presents a summary of the work done, and a discussion of the final results. Areas of future research are suggested to not only improve the current design principles of SAC, but to generally improve the performance of adaptive controllers and minimize negative aspects of their implementation.

Chapter 2

Problem Statement and Underlying Theory

The research presented here depends on linear, nonlinear, and adaptive control theories alongside optimization techniques. This chapter presents the relevant background theory necessary to improve SAC formulations in physical implementations, thereby increasing spacecraft autonomy. Due to the wide range of theoretical topics covered, readers are invited to return to each section when deeper knowledge of a topic is required. Each section is a thorough overview of the background material, and as such may be overwhelming for readers that are not interested in the entirety of the design methodologies presented throughout the text.

2.1 Introduction

Spacecraft assemblies include multiple subsystems that must communicate and operate in unison to produce a desired outcome. The subsystem that drastically improves spacecraft autonomy is the attitude determination and control subsystem (ADCS), which uses guidance, navigation, and control (GNC) techniques to manage the position and orientation of the spacecraft. Any additional systems that include motion will also require the guidance and control provided by ADCS. Improving spacecraft autonomy requires expanding the capabilities of ADCS, whose performance is heavily impacted by the presence and efficacy of GNC processing. Proper implementation of GNC techniques allow for effective measurements of current system states using onboard sensors and the generation of actuator commands that manage those states. Guidance and navigation, although vital to proper operation, is of little use without the controller that implements the actions necessary to ensure mission objectives are met.

However, each gramme of mass in a spacecraft contributes exponentially towards

increasing launch costs. As such the computational, sensing, and actuation capabilities that GNC relies upon are deliberately kept as low as feasible in order to minimize cost. The only way to verify if capabilities are sufficient is to test the performance of GNC.

Due to the prohibitive cost of spacecraft flight hardware and the inaccessibility of true 6DOF microgravity test environments, simulations of hardware performance are often used for evaluation. Implementations can be explored and tested in simulation before being applied to one of the limited 3DOF testbeds such as Carleton's SPOT, or true 6DOF microgravity test environments, such as the ISS or NASA's Microgravity Test Facility (MGTF)¹. In this text simulations are used to develop and explore implementations of SAC that are then tested in Carleton's SPOT to determine if simulation is able to successfully consider and compensate for the response of the real system. Since it is possible for elements to be missed in simulation that are present in hardware, it is critical to have a strong understanding of the equations of motion of the uncontrolled spacecraft problem. Without a strong understanding of the background elements of spacecraft system dynamics, linear and nonlinear control theory, or optimization techniques, hardware implementations may succumb to fatal flaws that are not apparent in simulation.

The following sections will present the background necessary to understand the uncontrolled spacecraft problem, as well as the theory underlying adaptive control, and optimization techniques that are leveraged to improve existing controller formulations.

First, the uncontrolled spacecraft rendezvous, docking, and control problem will be introduced and the relevant dynamics considered; second, the theory behind linear dynamic systems, linear controllers, nonlinear dynamics, and adaptive controllers will be introduced, alongside stability proofs and performance considerations where possible; finally, optimization methods and techniques are presented and explained to determine how adaptive controller implementations can maximize their strengths while mitigating their weaknesses.

¹https://www.nasa.gov/centers/ames/research/technology-onepaggers/microgravity_test_facility.html

2.2 Uncontrolled Target Acquisition and Control

Whether it be spacecraft repair, sample return missions, or debris deorbiting, the problem of docking with and then controlling a target of unknown mass properties must be mastered if spacecraft autonomy is to be increased. The general uncontrolled spacecraft problem will be developed and understood before tackling the control theory that allows it to be solved by adaptive control systems. Active debris deorbiting will be taken as the example scenario for this problem.

To deorbit debris, an uncontrolled object must be placed on a trajectory that ensures it deorbits and cannot harm currently active spacecraft. Such an operation would require an actively controlled spacecraft, hereafter called the *chaser*, to dock with and then control the previously uncontrolled object, called the *target*. Knowledge of the target's position can be leveraged to approach the target, rigidly connect to it, and then control both the target and chaser together to deorbit the target. The problem of ascertaining the position of an uncontrolled target, followed by rendezvous, docking, and then changing the path of the target is called *the uncontrolled spacecraft problem*.

The above example can be quickly generalized: it is desirable to determine how to make a chaser spacecraft rendezvous, dock, and control a target spacecraft that is not controlled. The dynamics of the uncontrolled target, as well as the dynamics of spacecraft trajectories have been well studied [42]. The reference frames of this problem, and the equations of motion before and after docking are covered in the following subsections. The dynamics presented in this section are leveraged in future sections to determine control formulations and to optimize trajectories.

2.2.1 Frames of Reference

The lack of static features in orbit necessitates precise and accurate definitions of frames of reference; a spacecraft orbiting the Earth, which is orbiting the Sun, which is speeding through the Milky Way galaxy, must use precise notions of position and orientation to ensure that it is in the right place at the right time. Reference frames ensure positions and orientations of a spacecraft in orbit are well-defined. The Earth-Centered Inertial (ECI) frame, as well as the Local-Vertical-Local-Horizontal (LVLH)

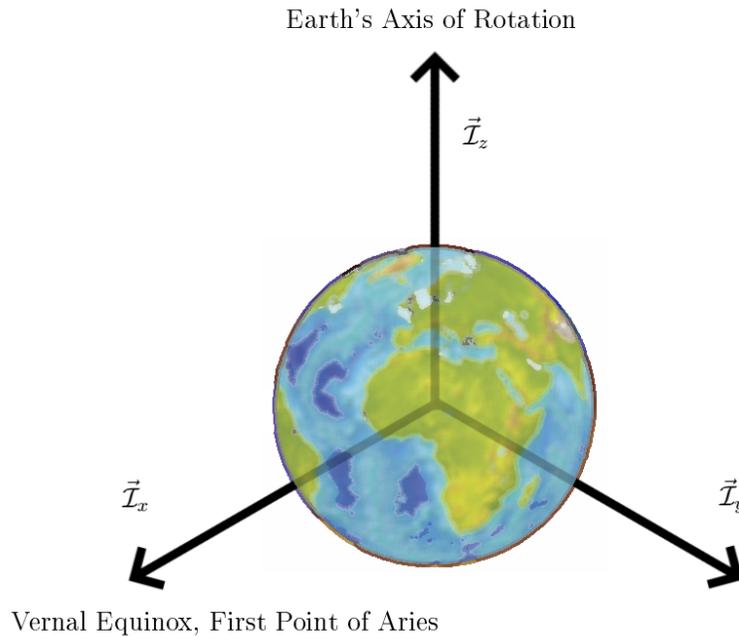


Figure 2.1: Earth Centered Inertial Reference Frame

frame are used in order to quickly and accurately describe the motion of a body around the Earth (ECI) or around a spacecraft of interest (LVLH). Reference frames are denoted using vectrix notation, that is to say vectors in ECI are denoted by premultiplication of the vectrix $\vec{\mathcal{F}}_I^T$, while vectors in LVLH are premultiplied by $\vec{\mathcal{F}}_L^T$. For convenience, a brief overview of standard vectrix notation is available in Appendix A, alongside other fundamentals of linear algebra and typical notation for the field.

The ECI reference frame is an inertial frame centred on the Earth that simplifies calculation of the accelerations on near-Earth spacecraft. The direction of the unit z-vector $\vec{\mathcal{I}}_z$ is aligned with the Earth's rotational axis, while the unit x-vector $\vec{\mathcal{I}}_x$ is aligned with the vernal equinox, also called the first point of Aries. Finally, the unit y-vector $\vec{\mathcal{I}}_y$ completes the triad of orthonormal basis vectors. An example of the ECI reference frame unit vector is shown in Fig. 2.1. Note that the ECI frame does not rotate with the Earth while the origin is fixed to the center of the planet.

A second non-inertial reference frame is used to simplify relative motion dynamics, called the Local-Vertical-Local-Horizontal reference frame of the target spacecraft. The LVLH reference frame has its origin at the center of mass of the target spacecraft. The unit x-vector in LVLH $\vec{\mathcal{L}}_x$, called the *radial direction*, is aligned with the center of the Earth and is found using the target position vector \vec{r}_t in any reference frame. The unit z-vector $\vec{\mathcal{L}}_z$ is in the direction of the orbital angular momentum vector, and is called the *cross-track direction*. The unit y-vector $\vec{\mathcal{L}}_y$ completes the orthonormal triad, and represents some notion of a *local horizontal* vector. The equations for determining the LVLH unitary orthonormal bases from the target inertial position \vec{r}_t and velocity \vec{v}_t are given by

$$\vec{\mathcal{L}}_x = \frac{\vec{r}_t}{r_t} \quad \vec{\mathcal{L}}_y = \vec{\mathcal{L}}_z \times \vec{\mathcal{L}}_x \quad \vec{\mathcal{L}}_z = \frac{\vec{r}_t \times \vec{v}_t}{|\vec{r}_t \times \vec{v}_t|} \quad (2.1)$$

Note that normally $\vec{\mathcal{L}}_z$ does not vary, while $\vec{\mathcal{L}}_x$ and $\vec{\mathcal{L}}_y$ rotate throughout the orbit, with the origin of the LVLH reference frame as seen from other reference frames varying in time following the orbit of the target.

An example of the LVLH basis vectors are shown in Fig. 2.2, where $\vec{\rho}$ is the relative position vector in the LVLH frame, developed later.

2.2.2 Orbital Dynamics

The developed reference frames can be used to describe the motion of spacecraft around a parent body, called *the two-body problem*. The two body problem has been studied for centuries, with the first model of orbital motion being developed in the 1600s following dedicated research efforts by Tycho Brahe. The first detailed descriptions of orbital motion were constructed by Johannes Kepler, followed by the time-varying analytical equations constructed by Sir Isaac Newton and the description of large-scale behaviour by Albert Einstein as part of his theory of general relativity. In all of the dynamic scenarios considered in this text, the chaser spacecraft will be in a near-Earth orbit, with a large part of the spacecraft motion unable to be directly controlled due to the large inertial velocities and accelerations present during orbital

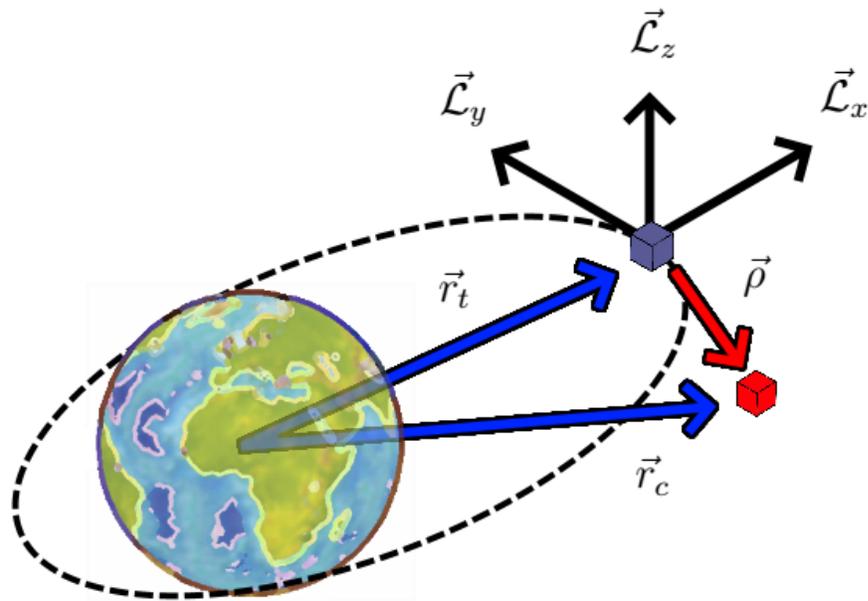


Figure 2.2: Local-Vertical-Local-Horizontal Reference Frame

motion. All relevant gravitational effects are captured through the use of Newtonian dynamics, the relevant results of which are developed and discussed here.

Classical Orbital Dynamics

In 1621 Kepler described three laws of planetary motion for each planet around the sun, and by extension the motion of any body around the Earth. Kepler's laws of elliptical orbits, law of equal area, and law of periods are represented in Fig. 2.3.

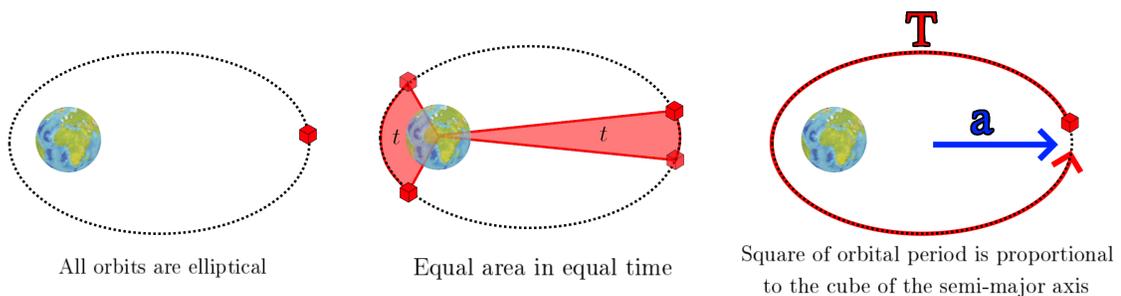


Figure 2.3: Kepler's Laws

Stuck home during the bubonic plague outbreak of 1665, Sir Isaac Newton developed the first notions of the law of universal gravitation.² Once mature, Newton's law of gravitation would describe gravity as acting proportionally to the mass of both objects, and inversely to the square of the distance between both objects. Newton's law of gravitation for a target spacecraft outside a massive body is formulated following

$$\vec{F}_t = -\frac{\mathbb{G}m_b m_t}{|\vec{r}_t|^3} \vec{r}_t \quad (2.2)$$

where \vec{F}_t is the force acting on the target, in N, m_b is the mass of the primary body in kg, m_t is the mass of the target in kg, and \vec{r}_t is the distance vector from the massive body to the target. The experimentally determined universal gravitational constant is denoted by the letter \mathbb{G} and identifies the fundamental strength of the gravitational field permeating space, which has been determined to hold a value of $\mathbb{G} = 6.67408 \times 10^{-11} \text{ Nm}^2/\text{kg}^2$ [43]. Equation 2.2 is often simplified to include the mass of both objects and the universal gravitational constant as a single term in the numerator. Analysis has yielded analytical time-varying solutions to the two body problem when both objects are under no other forces.

Newton's law of universal gravitation provides a simple formula to determine the propagation of accelerations throughout a gravitational system. If it can be assumed that the parent body is much larger than the spacecraft, that is to say $m_b \gg m_t$, then the target's mass has a negligible effect on the acceleration of the parent body. If only the acceleration of the target is desired, the orbital motion equation can be simplified to

$$\ddot{\vec{r}}_t = -\mu_{\oplus} \frac{\vec{r}_t}{|\vec{r}_t|^3} \quad (2.3)$$

where now \vec{r}_t is the position of the target in an inertial reference frame and $\mu_{\oplus} = \mathbb{G}m_b$ denotes a constant for the Earth-spacecraft two body system. NASA's DE440 document publishes a list of these parameters, where μ_{\oplus} is called the standard gravitational constant for Earth, and has a value of $398\,600.435436 \text{ km}^3/\text{s}^2$ [43]. Equation 2.3 is a second order differential equation for the acceleration of the spacecraft in the inertial

²<https://www.nationaltrust.org.uk/woolsthorpe-manor/features/year-of-wonders>

reference frame which can be propagated through time to produce the position, velocity, and acceleration of the spacecraft at any point in time for a given set of initial conditions.

A large range of intuitive and unintuitive dynamics arise from the application of the orbital equation of motion to spacecraft. When a spacecraft is going a sufficiently large speed at a sufficiently large distance, the spacecraft can orbit the parent body. Any set of initial positions and velocities corresponds to an orbit, which can be defined using a set of values called the *classical orbital elements*. The set of classical orbital elements $\{a, e, i, \Omega, \omega, \theta\}$ are described in more detail in Appendix A.

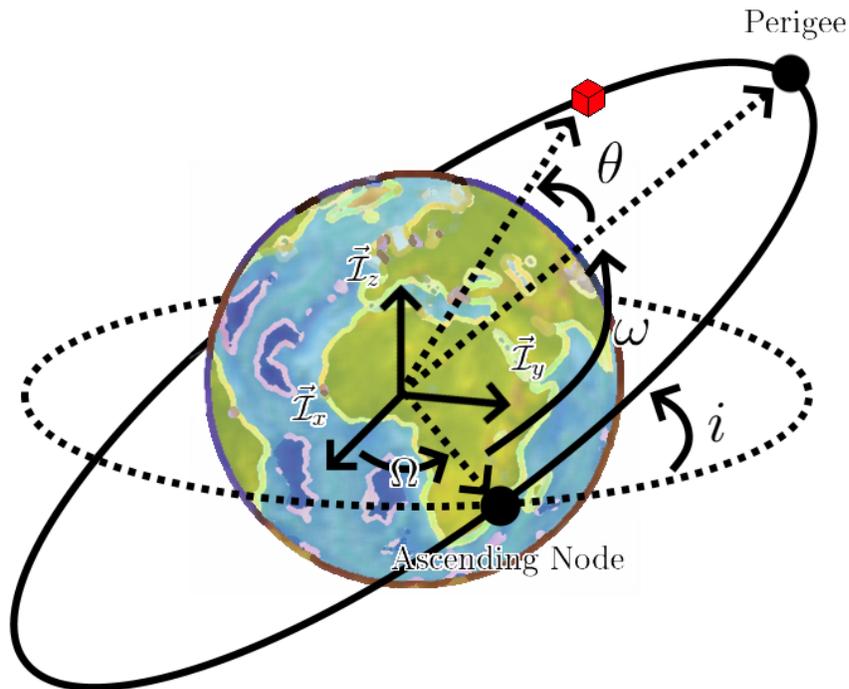


Figure 2.4: Orbital Elements $\{a, e, i, \Omega, \omega, \theta\}$ Describe an Individual Orbit

The result is that all orbits are made up of an ellipse fixed to a single plane which crosses the equatorial plane at some longitude and at some angle. The orbit must have a point of closest approach to the Earth that occurs at some angle along the orbital plane. Furthermore, the orbital plane is required to pass through the center of mass of the planet, and any attempts to change the orbital path of the spacecraft

will eventually, once acceleration is stopped, revert to an orbital plane that passes through the center of mass of the planet.

2.2.3 Formation Flying Dynamics

In order to rendezvous with a target, it is necessary that the target's orbital path overlaps with the chaser's, and that both spacecraft are at the overlapping point at similar times and with similar velocities. If these requirements are not met, the two spacecraft will not rendezvous, either arriving at the same point at different times, or speeding past one another.

The orbital trajectories of a spacecraft can be determined by propagating Eq. (2.3) through time to determine the positions and velocities of the spacecraft. For a given spacecraft with initial position \vec{r}_0 , initial velocity \vec{v}_0 , then its time-varying position $\vec{r}(t)$ and velocity $\vec{v}(t)$ for the gravitation acceleration $\ddot{\vec{r}}(t)$ given by Eq. (2.3) is governed by

$$\vec{v}(t) = \vec{v}_0 + \int_0^t \ddot{\vec{r}}(\tau) d\tau \quad (2.4)$$

$$\vec{r}(t) = \vec{r}_0 + \int_0^t \vec{v}(\tau) d\tau \quad (2.5)$$

Propagation is repeated for both the target and chaser spacecraft to determine both the target's time varying position $\vec{r}_t(t)$ and velocity $\vec{v}_t(t)$ as well as the chaser's time varying position $\vec{r}_c(t)$ and velocity $\vec{v}_c(t)$ in the ECI reference frame. The relative position of the spacecraft can then be found in the ECI reference frame through vector difference to yield the relative position vector $\vec{\rho}$ in ECI

$$\vec{\rho} = \vec{r}_c - \vec{r}_t = \vec{\mathcal{F}}_I^T(\mathbf{r}_c - \mathbf{r}_t) = \vec{\mathcal{F}}_I^T \boldsymbol{\rho} \quad (2.6)$$

$$\dot{\vec{\rho}} = \vec{v}_c - \vec{v}_t = \vec{\mathcal{F}}_I^T(\mathbf{v}_c - \mathbf{v}_t) = \vec{\mathcal{F}}_I^T \dot{\boldsymbol{\rho}}_I \quad (2.7)$$

where $\boldsymbol{\rho}_I \in \mathbb{R}^3$ are the elements of the relative position vector in ECI. Throughout this text bolded variables will be used to denote matrix elements, such as the components

of a vector.

Formation flying dynamics can be more easily understood in the LVLH reference frame, which is specifically constructed to have the target and chaser overlap at $\boldsymbol{\rho}_I = \mathbf{0}$. The distance vector $\vec{\rho}$ can be transformed from ECI to LVLH using standard operations.

In order to determine the distance vector in the LVLH reference frame the difference must be rotated. The rotation sequence from ECI to LVLH can be found using the reference frame vector presented previously, namely that the rotation matrix $\mathbf{C}_{\mathcal{LI}} \in \mathbb{R}^{3 \times 3}$ which rotates from the ECI to LVLH reference frames is given by

$$\mathbf{C}_{\mathcal{LI}} = \vec{\mathcal{F}}_{\mathcal{L}} \cdot \vec{\mathcal{F}}_{\mathcal{I}}^T \quad (2.8)$$

which, when applied to the vector difference yields the difference vector in the LVLH reference frame

$$\vec{\mathcal{F}}_{\mathcal{L}}^T \boldsymbol{\rho} = \mathbf{C}_{\mathcal{LI}} \vec{\mathcal{F}}_{\mathcal{I}}^T \boldsymbol{\rho}_I \quad (2.9)$$

with $\boldsymbol{\rho} \in \mathbb{R}^3$ denoting the components of the relative position vector in LVLH.

Due to their ubiquity in aerospace applications, many alternative methods exist to determine the rotation vectors between the ECI and LVLH reference frames.

Due to the angular velocity of the LVLH frame relative to the ECI frame, the relative velocity in the LVLH reference frame $\vec{\mathcal{F}}_{\mathcal{L}}^T \dot{\boldsymbol{\rho}}$ is subject to an additional velocity component. Namely, for the relative angular velocity vector $\boldsymbol{\omega}_{\mathcal{LI}} \in \mathbb{R}^3$ between the ECI and LVLH reference frame, the velocity of the chaser spacecraft in the LVLH reference frame is given by

$$\vec{\mathcal{F}}_{\mathcal{L}}^T \dot{\boldsymbol{\rho}} = \mathbf{C}_{\mathcal{LI}} \vec{\mathcal{F}}_{\mathcal{I}}^T (\dot{\boldsymbol{\rho}} - \boldsymbol{\omega}_{\mathcal{LI}}^\times \boldsymbol{\rho}) \quad (2.10)$$

where the superscript \times on the relative angular velocity $\boldsymbol{\omega}_{\mathcal{LI}}$ denotes the skew symmetric matrix corresponding to the relative angular velocity column matrix, and corresponds to the matrix equivalent of the cross product. The relative angular velocity itself can be determined through a simple cross product

$$\boldsymbol{\omega}_{\mathcal{LI}} = \frac{\mathbf{r}_t^\times \mathbf{v}_t}{\|\mathbf{r}_t\|^2} \quad (2.11)$$

Given the above transformations from the ECI to the LVLH reference frame, Alfriend and Terry [44] provide a full derivation of the equations of motion for each of the three directions in the LVLH reference frame. Only the results are reproduced here.

First, the LVLH distance vector is deconstructed into its components, namely that

$$\boldsymbol{\rho} = \begin{bmatrix} x_L \\ y_L \\ z_L \end{bmatrix} \quad (2.12)$$

for the LVLH displacements $x_L, y_L, z_L \in \mathbb{R}$ in the LVLH x, y, and z axes, respectively. The relative dynamics are significantly affected by the target spacecraft's orbit, and as such the dynamics require calculation of the difference equations for the target spacecraft's true anomaly $\theta \in \mathbb{R}$ and radius value $r_t \in \mathbb{R}$ which are determined through the two differential equations

$$\ddot{r}_t = \dot{\theta}^2 r_t - \frac{\mu_\oplus}{r_t^2} \quad (2.13)$$

$$\ddot{\theta} = -2 \frac{\dot{r}_t \dot{\theta}}{r_t} \quad (2.14)$$

and integrated through time to determine their evolution. If the forces acting on the chaser are defined as

$$\mathbf{F}_c = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} \quad (2.15)$$

and the additional variable simplification

$$r_c = \sqrt{(r_t + x_L)^2 + y_L^2 + z_L^2} \quad (2.16)$$

is made, then the equations of motion in the LVLH x, y, and z directions are

$$\ddot{x}_L - \dot{\theta}^2 x_L - 2\dot{\theta} \left(\dot{y}_L - y_L \frac{\dot{r}_t}{r_t} \right) - \frac{\mu_{\oplus}}{r_t^2} = -\frac{\mu_{\oplus}}{r_c^3} (r_t + x_L) + \frac{F_x}{m_c} \quad (2.17)$$

$$\ddot{y}_L - \dot{\theta}^2 y_L + 2\dot{\theta} \left(\dot{x}_L - x_L \frac{\dot{r}_t}{r_t} \right) = -\frac{\mu_{\oplus}}{r_c^3} y_L + \frac{F_y}{m_c} \quad (2.18)$$

$$\ddot{z}_L = -\frac{\mu}{r_c^3} z_L + \frac{F_z}{m_c} \quad (2.19)$$

The formation flying equations include coupling between the x and y axes. Similarly, the rate of change of the true anomaly has an effect on the dynamics. All nonlinear effects, however, diminish as the distance between spacecraft decrease, and the relative velocities decrease. That is to say the nonlinear effects of formation flying dynamics are lessened if both target and chaser share similar orbits. The nonlinearities in the equation diminish even further if a circular orbit is considered, in which case $\ddot{\theta}$ goes to zero, and $\dot{\theta}$ stays constant. Finally, at sufficiently small distances and relative velocities, dynamics return to the familiar Newtonian double-integrator dynamics, $F = ma$.

Control of spacecraft relative motion will be performed in future chapters. The greatest challenges inherent in controlling relative spacecraft motion occurs when attempting to manage rendezvous opportunities, fuel consumption, and determining trajectories. Many papers have been dedicated to these topics, which fall broadly under the Guidance, Navigation, and Control (GNC) umbrella. For the current exercise, optimization of spacecraft trajectory commands will be performed to determine rendezvous opportunities. By managing major nonlinearities through trajectory optimization the adaptive controller implementation no longer needs to consider and manage the nonlinearities in Eq. (2.17) through (2.19), but instead only needs to ensure that it is able to achieve the trajectory commands demanded of it. If trajectory commands are near the spacecraft position, then only the double-integrator dynamics in Eqs. (2.17) through (2.19) need to be managed by the control law, while the nonlinearities are managed by guidance methods through determining a rendezvous opportunity. The control task, therefore, is simplified dramatically by introducing

appropriate guidance.

2.2.4 Orientation and Position Dynamics for Controlled and Combined Spacecraft

The uncontrolled spacecraft problem can be broken into two distinct phases: first, the target must be ascertained, approached, and then docked with; subsequently, the chaser must control a new system composed of itself and the uncontrolled target. The dynamics of both systems are similar, however a large change in parameters occurs at the moment of rigid attachment. The dynamics of both systems, as well as the change in properties at the moment of docking are developed here. The disjoint dynamics presented will be tested in future chapters by a linear and adaptive controller in simulation to determine performance of both techniques when presented with a target of unknown mass and moment of inertia.

Chaser Only System

Control of the chaser spacecraft before docking with the target follows familiar dynamic formulations. The chaser will have a mass denoted by $m_c \in \mathbb{R}$, and moment of inertia $\mathbf{J}_c \in \mathbb{R}^{3 \times 3}$. The position dynamics of the chaser follow the relative motion dynamics presented in Eq. (2.17), with the chaser accelerations in the LVLH reference frame \ddot{x}_L , \ddot{y}_L , and \ddot{z}_L well approximated for small displacements by Newton's second law

$$\ddot{\boldsymbol{\rho}} = \mathbf{F}_c/m_c \tag{2.20}$$

for the vector of forces acting on the chaser $\mathbf{F}_c \in \mathbb{R}^3$ in N, and LVLH position vector $\boldsymbol{\rho} \in \mathbb{R}^3$ in m. The position dynamics in ECI can also be described the classical gravitation dynamics in Eq. (2.3) for position $\mathbf{r}_c \in \mathbb{R}^3$. The only control forces considered here are the control thrusts provided by the chaser. The attitude dynamics of the chaser in the body fixed reference frame follows the standard attitude dynamic equation

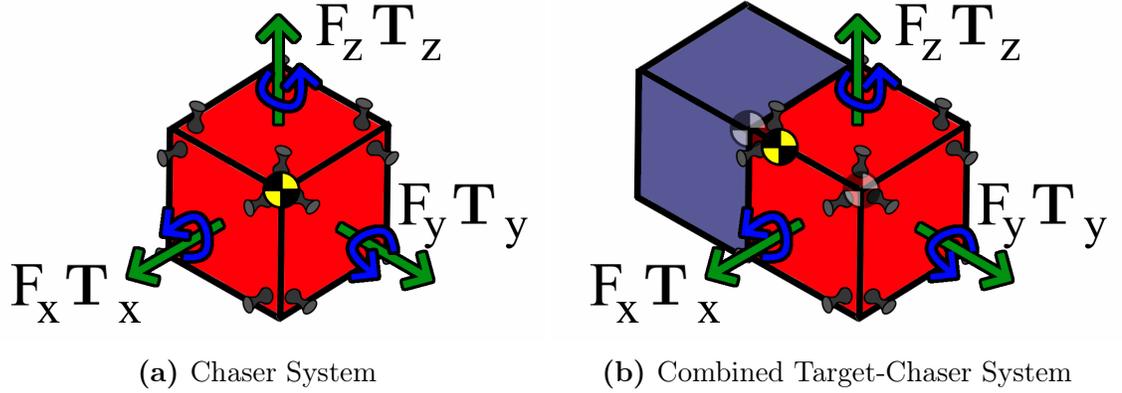


Figure 2.5: Control Systems Before and After Docking

$$\mathbf{J}_c \dot{\boldsymbol{\omega}}_c = -\boldsymbol{\omega}_c^\times \mathbf{J}_c \boldsymbol{\omega}_c + \mathbf{T}_c \quad (2.21)$$

with the angular velocity of the chaser in rad/s being denoted by $\boldsymbol{\omega}_c \in \mathbb{R}^3$, and the control torques acting on the chaser in Nm being given by $\mathbf{T}_c \in \mathbb{R}^3$.

These dynamics are identical for the target system before docking, with mass $m_t \in \mathbb{R}$, and inertia $\mathbf{J}_t \in \mathbb{R}^{3 \times 3}$, target position $\mathbf{r}_t \in \mathbb{R}^3$, and angular velocity $\boldsymbol{\omega}_t \in \mathbb{R}^3$. Notably, due to energy dissipation during elastic deformation, if the target spacecraft has been left for sufficiently long without control its rotation will likely be about its major inertial axis [45].

2.2.5 Combined Target-Chaser System

The combined target-chaser system, called the *combined system* for brevity, is the system to be controlled once the target and chaser are docked. First, the transition from chaser-only to the combined system will be briefly considered before the dynamics of the combined system are derived.

After docking, the spacecraft transitions to a combined system that includes both the chaser and target that can only be controlled by the actuators of the chaser. The combined system is modeled as a single rigid body with new moment of inertia, mass, center of mass location, and offset thrusters. The dynamics of the new rigid body system can be determined from previous parameters. An example schematic of the chaser-only and combined system is demonstrated in Fig. 2.5

Effects that occur during the moment that rigid docking occurs, such as conservation of linear and angular momentum, as well as collision dynamics, are not considered at present. Effects that occur at the moment of docking do not help to clarify the benefits and drawbacks of adaptive control in spacecraft operations. Instead, the role of adaptive control post-dock is emphasized to determine the suitability of adaptive control when the target parameters are unknown. Docking will be considered to occur instantaneously once the relative position and velocities of the target and chaser docking ports are sufficiently small. The boundary conditions of the chaser-only and combined system will need to be determined, as well as the combined system dynamics.

The chaser body-fixed reference frame is introduced, and denoted as $\vec{\mathcal{F}}_c$, with its origin at the Center of Mass (COM) of the chaser, and orthonormal axes dependant on spacecraft construction. The measured chaser position is unchanged before and after docking occurs. Where t_d is the instant of docking, t_d^+ is the instant after docking, and t_d^- is the instant before docking, then

$$\mathbf{r}_c(t_d^-) = \mathbf{r}_c(t_d^+) \quad (2.22)$$

The distance between the COM of the chaser and the chaser docking point is denoted by $\mathbf{r}_{a,c} \in \mathbb{R}^3$. The corresponding distance between the COM of the target and its docking point is given by $\mathbf{r}_{a,t} \in \mathbb{R}^3$. A demonstration of the docking point vectors is shown in Fig. 2.6.

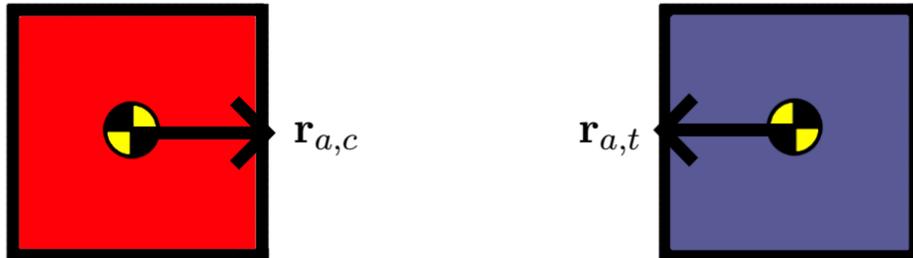


Figure 2.6: Docking Point Vectors For Target and Chaser Spacecraft

The docking points, as defined through $\mathbf{r}_{a,t}$ and $\mathbf{r}_{a,c}$, are coincident at the moment

of docking. For all examples in this text, the docking point displacements vectors will also be taken to align with the docking axis, and as such $\mathbf{r}_{a,c}$ and $\mathbf{r}_{a,t}$ will also be collinear at the time of docking. Following the requirement for coincidence and collinearity, the distance between the target and chaser COMs in the combined system is

$$\vec{\mathcal{F}}_c^T \mathbf{r}_{a,cb} = \vec{\mathcal{F}}_c^T (\mathbf{r}_{a,c} - \mathbf{r}_{a,t}) \quad (2.23)$$

Due to the docking constraints, the position of the combined system COM location must occur some distance along $\mathbf{r}_{a,cb}$, and corresponds to the weighted sum of the masses of the target and chaser spacecraft. The dimensionless distance of the COM along the combined docking axis vector, beginning from the initial chaser position, is thus

$$\bar{x}_{frac} = \frac{m_c}{m_c + m_t} \quad (2.24)$$

The distance in meters from the chaser COM to the new COM location in the chaser body fixed frame, for use in dynamics calculations, is thus

$$\bar{\mathbf{x}} = \bar{x}_{frac} \mathbf{r}_{a,cb} \quad (2.25)$$

The combined system therefore has an initial COM location of

$$\mathbf{r}_{cmb}(t_d^+) = \mathbf{r}_c(t_d^+) + \bar{\mathbf{x}} = \mathbf{r}_c(t_d^-) + \bar{\mathbf{x}} \quad (2.26)$$

which provides the initial position condition of the combined system. At the moment of docking, the target and chaser will be forced to share similar velocities, and as such the initial condition for the combined system velocity $\mathbf{v}_{cmb}(t_d^+)$ is simply taken as

$$\mathbf{v}_{cmb}(t_d^+) = \mathbf{v}_c(t_d^-) \quad (2.27)$$

The combined system will have a different moment of inertia from the chaser-only system, due to the sudden addition of the target's mass to the chaser. The moment of inertia in the combined system body-fixed reference frame $\vec{\mathcal{F}}_{cmb}$, which is aligned with

the chaser body fixed reference frame but is centered at the combined system center of mass, is calculated through parallel axis theorem about the new COM location. The moment of inertia contribution to the combined system due to the chaser inertia $\mathbf{J}_{cmb,c}$ is [46]

$$\mathbf{J}_{cmb,c} = \mathbf{J}_c + m_c[\bar{\mathbf{x}}^T \bar{\mathbf{x}} \mathbf{I}_3 - \bar{\mathbf{x}} \bar{\mathbf{x}}^T] \quad (2.28)$$

where \mathbf{I}_3 is simply the identity matrix in $\mathbb{R}^{3 \times 3}$. The target's moment of inertia contribution to the combined system must be rotated 180° about the body z-axis to align its contributions to the combined system body-fixed reference frame. If the reference frame $\vec{\mathcal{F}}_t$ is related to a second reference frame $\vec{\mathcal{F}}_{cmb}$ by a rotation matrix $\mathbf{R} \in \mathbb{R}^{3 \times 3}$ by

$$\vec{\mathcal{F}}_{cmb} = \mathbf{R} \vec{\mathcal{F}}_t \quad (2.29)$$

the transformation of a moment of inertia vector \mathbf{J} in reference frame $\vec{\mathcal{F}}_t$, to the same moment of inertia in the second reference frame $\vec{\mathcal{F}}_{cmb}$ is given by

$$\vec{\mathcal{F}}_{cmb} \mathbf{J} = \mathbf{R} \vec{\mathcal{F}}_t \mathbf{J} \mathbf{R}^T \quad (2.30)$$

Equation (2.30) is used to transform the target inertia \mathbf{J}_t into the combined body reference frame by rotation \mathbf{R} . The contribution to the combined system moment of inertia from the target is given by

$$\mathbf{J}_{cmb,t} = \mathbf{R} \mathbf{J}_t \mathbf{R}^T + m_t [(\bar{\mathbf{x}} - \mathbf{r}_{a,cmb})^T (\bar{\mathbf{x}} - \mathbf{r}_{a,cmb}) \mathbf{I}_3 - (\bar{\mathbf{x}} - \mathbf{r}_{a,cmb})(\bar{\mathbf{x}} - \mathbf{r}_{a,cmb})^T] \quad (2.31)$$

where the parallel axis distance is now $(\bar{\mathbf{x}} - \mathbf{r}_{a,cmb})$. Finally the combined system moment of inertia is the sum of the two contributions

$$\mathbf{J}_{cmb} = \mathbf{J}_{cmb,c} + \mathbf{J}_{cmb,t} \quad (2.32)$$

Using the mass of the combined system $m_{cmb} = m_c + m_t$ and moment of inertia \mathbf{J}_{cmb} the dynamics of the combined system can now be found using the previous dynamics system in Eqs. (2.13) through (2.19) for the position and Eq. (2.21) for the rotation.

In the combined system no forces or torques can be provided by the target spacecraft, and position measurements occur at the COM of the chaser. Cross-coupling occurs between the position and orientation responses due to offset thrusts and offset position measurements.

The moments due to thrusts \mathbf{F}_c are dictated by the offset between the original chaser COM and the new combined system COM. The additional torques created by the new COM location is given by

$$\mathbf{T}_d = \bar{\mathbf{x}}^\times \mathbf{F}_c \quad (2.33)$$

where \mathbf{F}_c is the output thrust of the control system and actuation on the combined system.

The position measurement of the control system still occurs at

$$\mathbf{r}_c = \mathbf{r}_{cmb} - \bar{\mathbf{x}} \quad (2.34)$$

Since \mathbf{r}_c is no longer the COM, rotations of the combined system will result in velocities of the point \mathbf{r}_c . From rigid body kinematics the derivative of \mathbf{r}_c given the combined system velocity $\mathbf{v}_{cmb} \in \mathbb{R}^3$ and angular velocity $\boldsymbol{\omega}_{cmb} \in \mathbb{R}^3$ will be

$$\mathbf{v}_c = \mathbf{v}_{cmb} - \boldsymbol{\omega}_{cmb}^\times \bar{\mathbf{x}} \quad (2.35)$$

The first term of which is simply the velocity found through the equations of motion. The second term is the motion of the chaser COM due to rotation. The linear approximation for velocity change due to rotation can be used for linear system analysis, and is given through linear approximation of Eq. (2.21) as

$$\frac{d(\boldsymbol{\omega}_{cmb}^\times \bar{\mathbf{x}})}{dt} = \dot{\boldsymbol{\omega}}_{cmb}^\times \bar{\mathbf{x}} \approx (\mathbf{J}_{cmb}^{-1} \mathbf{T}_c)^\times \bar{\mathbf{x}} \quad (2.36)$$

Leveraging the properties of skew symmetric matrices allows for the form

$$\frac{d(\boldsymbol{\omega}_{cmb}^\times \bar{\mathbf{x}})}{dt} \approx (\mathbf{J}_{cmb}^{-1} \mathbf{T}_c)^\times \bar{\mathbf{x}} = (\bar{\mathbf{x}}^\times)^T \mathbf{J}_{cmb}^{-1} \mathbf{T}_c \quad (2.37)$$

Now the linear approximation for the combined system response can be established

using Eqs. (2.20) and (2.21), alongside Eqs. (2.33) through (2.36). The linear approximation of the effects of control thrusts and torques on position and angular velocity is found to be

$$\begin{bmatrix} \ddot{\mathbf{r}}_{cmb} \\ \dot{\boldsymbol{\omega}}_{cmb} \end{bmatrix} = \begin{bmatrix} 1/(m_{cmb})\mathbf{I}_3 & -\bar{\mathbf{x}}^\times \mathbf{J}_{cmb}^{-1} \\ \mathbf{J}_{cmb}^{-1} \bar{\mathbf{x}}^\times & \mathbf{J}_{cmb}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{F}_c \\ \mathbf{T}_c \end{bmatrix} \quad (2.38)$$

As a final consideration, in a real system it is likely that any docked configuration would have limitations on which thrusters could fire from the chaser without interfering with the target, unless the design was made specifically to consider this eventuality. For the current mission this would restrict linear accelerations in the body negative x direction, as well as reduce the maximum available torque. The problem of designing a control system that can eliminate negative x direction thrusts is not considered here, and so the requirement for thruster output to not intersect with the target body is dropped.

The target, chaser, and combined system dynamic equations developed here will be used in Chapter 5 to develop a simulation of spacecraft rendezvous, docking, and post-docking control under linear and adaptive control.

2.3 Linear Dynamics and Controller Design

The purpose of placing spacecraft in orbit is to enact some meaningful work, observations, or change that can only be achieved from space. It can be difficult, for example, to monitor worldwide weather patterns from the ground without implementing numerous ground stations all across the planet and in the water. However, the same mission can be completed almost trivially with a spacecraft. The Japanese Himawari-9 spacecraft is currently in a geosynchronous orbit, with its orbital trajectory matching the rotation of the Earth perfectly. Himawari-9 is able to continuously monitor the weather of almost an entire hemisphere simultaneously. Furthermore, by making use of multiple cameras, lenses, and other sensors it is able to treat each pixel of each of its sensors as though it were a single ground station on the Earth's surface. The first photo taken from Himawari-9, showcasing its observation potential, is shown in Fig. 2.7.



Figure 2.7: The First Photo From Himawari 9, Courtesy JAXA³.

However, in order to maintain its geosynchronous orbit, the spacecraft must be able to stay in one place and reject any orbital perturbations that might cause it to drift. In many situations, such as the one above, a knowledge of linear dynamics and linear control theory allow designers to create simple controllers with guaranteed performance that ensure manoeuvres can be performed and disturbances are rejected. Without linear control systems, the Himawari-9 spacecraft would have to be continuously monitored by ground crews, its position and orientation considered, and corrected when disturbances grew too large. By introducing linear control law, repeated and simple control actions like stationkeeping can be offloaded to an automatic system, allowing ground crews to consider higher level and more impactful problems surrounding operations. Linear control law that can handle these types of control problems are described and developed here.

2.3.1 Continuous Linear Dynamics

A great deal of linear control theory is covered in textbooks on the subject that are covered in undergraduate and post-graduate courses. In particular, Antsaklis and

³http://www.jma.go.jp/jma/jma-eng/satellite/news/himawari89/20170124_himawari9_first_images.html

Michel's *Linear Systems Primer* [20] covers a great deal of linear system theory in great depth. What is covered here is a review of the fundamental results that will be useful for improving adaptive control implementations covered in future chapters. The use of Laplace space representation of linear differential systems, the associated state-space realizations of those systems, and the discrete update equations will be used interchangeably, with the understanding that under the right conditions these descriptions are identical. Similarly, a knowledge of block scheme diagrams of systems will be leveraged in Chapter 3. A review of the pertinent details of linear systems are reviewed in Appendix B, and for those inclined, in reference [20].

State-space systems will be used extensively in future sections, so it is useful to define them briefly here.

A wide variety of systems can be represented as, or are well approximated by, linear difference equations, such as Newtonian acceleration, harmonic oscillators, or spring damper systems. Linear difference equations can also be placed into linear state-space form.

DEFINITION 1 (LINEAR TIME-INVARIANT STATE-SPACE REPRESENTATION) A linear time-invariant (LTI) state-space system is the collection of matrices of order $n \in \mathbb{Z}^+$, inputs $p \in \mathbb{Z}^+$, outputs $m \in \mathbb{Z}^+$, associated matrices $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\mathbf{B} \in \mathbb{R}^{n \times p}$, $\mathbf{C} \in \mathbb{R}^{m \times n}$, and $\mathbf{D} \in \mathbb{R}^{p \times m}$, such that

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \quad (2.39)$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \quad (2.40)$$

for inputs column matrix $\mathbf{u}(t) \in \mathbb{R}^{p \times 1}$, state column matrix $\mathbf{x}(t) \in \mathbb{R}^{n \times 1}$, and system output column matrix $\mathbf{y}(t) \in \mathbb{R}^{m \times 1}$, all time-varying. The collection of matrices that represent a linear state-space system are denoted as $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$

REMARK The LTI state-space representation is so called because its transition matrices $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$ do not vary in time, and are linear with respect to the inputs, states, and output. Properties of this system are discussed in Appendix B.

Two classes of linear system, the Strictly Positive Real (SPR) and Almost Strictly Positive Real (ASPR) system, have properties that are useful for adaptive controllers.

DEFINITION 2 (ALMOST STRICTLY POSITIVE REAL SYSTEM [47]) A system $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$ is almost strictly positive real if by some constant feedback error gain the system becomes strictly positive real. That is to say that for the constant feedback gain $\tilde{\mathbf{K}}_e \in \mathbb{R}^{m \times p}$, and transition matrix

$$\mathbf{A}_c = \mathbf{A} - \mathbf{B}\tilde{\mathbf{K}}_e\mathbf{C} \quad (2.41)$$

a second system $\{\mathbf{A}_c, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$ is strictly positive real, satisfying the Kalman-Yakubovich-Popov conditions

$$\mathbf{P}\mathbf{A}_c + \mathbf{A}_c^T\mathbf{P} = -\mathbf{Q} \quad (2.42)$$

$$\mathbf{P}\mathbf{B} = \mathbf{C}^T \quad (2.43)$$

for some positive definite matrices $\mathbf{P} \in \mathbb{R}^{n \times n}$ and $\mathbf{Q} \in \mathbb{R}^{n \times n}$.

Corollary 2.3.0.1 ([48]) *The transfer function of any system meeting the SPR condition must have a relative order of -1, 0, or 1. That is to say that the difference between the number of poles and zeros in the transfer function must be -1, 0, or 1.*

Likewise, the difference in phase between any input frequency and output frequency must be within $\pm 90^\circ$ of the frequency input.

All SPR systems are Hurwitz stable. Finally, all ASPR system remains stable as the feedback gain approaches infinity.

The tools and definitions available in linear system analysis allow for a wide variety of system to be accurately controlled, and their performance measured. An overview of the techniques available in linear system analysis is briefly touched on in Appendix B, and will be useful when designing SAC with the recommendations made in Chapter 3.

2.3.2 Discrete Linear System Analysis

The rise and ubiquity of digital computation devices means that the vast majority of modern control systems are discrete. Measurement devices on control systems can only provide data at discrete intervals and computations necessarily takes some time to complete. The devices that turn controller outputs into control actions can only accept input at fixed intervals, and so, systems that update in discrete steps must be considered. Discrete system analysis provides the tools to ensure that even systems that update at discrete intervals can be accurately described.

Similar to continuous linear system theory, it is assumed that the reader has a grasp of discrete linear system theory. Of particular note is the existence of discrete counterparts to the ASPR condition mentioned in Sec. 2.3.1, and described in Appendix B. The ASPR condition is briefly developed insofar as it applies to direct adaptive control.

The discrete version of a linear state-space system shares many similarities with its continuous counterpart.

DEFINITION 3 (DISCRETE LINEAR STATE-SPACE SYSTEM [20]) The discrete linear state-space realization of a system at any timestep $k \in \mathbb{Z}$ is described by the system matrices $\{\mathbf{A}_d, \mathbf{B}_d, \mathbf{C}_d, \mathbf{D}_d\}$ following the update equations

$$\mathbf{x}_d(k+1) = \mathbf{A}_d \mathbf{x}_d(k) + \mathbf{B}_d \mathbf{u}_d(k) \quad (2.44)$$

$$\mathbf{y}_d(k) = \mathbf{C}_d \mathbf{x}_d(k) + \mathbf{D}_d \mathbf{u}_d(k) \quad (2.45)$$

for order $n \in \mathbb{Z}$, input size $p \in \mathbb{Z}$, output size $m \in \mathbb{Z}$, and the associated matrices $\mathbf{A}_d \in \mathbb{R}^{n \times n}$, $\mathbf{B}_d \in \mathbb{R}^{n \times p}$, $\mathbf{C}_d \in \mathbb{R}^{m \times n}$, and $\mathbf{D}_d \in \mathbb{R}^{p \times m}$, that translate the time-varying inputs column matrix $\mathbf{u}_d(k) \in \mathbb{R}^{p \times 1}$, into time-varying state column matrix $\mathbf{x}_d(k) \in \mathbb{R}^{n \times 1}$, and system output column matrix $\mathbf{y}_d(k) \in \mathbb{R}^{m \times 1}$.

REMARK Although the discrete state-space descriptor system is similar to its continuous counterpart in form, it should be noted that the state update Eq. (2.44) outputs the state at the next iteration, instead of its derivative to be integrated. As such, for

small timesteps, \mathbf{A}_d approaches the identity matrix, and \mathbf{B}_d becomes a matrix of zero elements.

The method for converting a continuous linear state-space system into its discrete counterpart is discussed in Appendix B.

The discrete analogue of the SPR and ASPR conditions is also useful when considering adaptive control in discrete systems. A summary by Hoagg et al. [49] provides the equivalent conditions for a discrete SPR system, and by extension the conditions for a discrete ASPR system.

DEFINITION 4 (DISCRETE ALMOST STRICTLY POSITIVE REAL SYSTEM [49]) A discrete state-space system $\{\mathbf{A}_d, \mathbf{B}_d, \mathbf{C}_d, \mathbf{D}_d\}$ is discrete almost strictly positive real if by some constant feedback error gain the system becomes strictly positive real. That is to say that there exists a constant feedback gain $\tilde{\mathbf{K}}_e \in \mathbb{R}^{m \times p}$, and transition matrix

$$\mathbf{A}_{c,d} = \mathbf{A}_d - \mathbf{B}_d \tilde{\mathbf{K}}_e \mathbf{C}_d \quad (2.46)$$

such that a second system $\{\mathbf{A}_{c,d}, \mathbf{B}_d, \mathbf{C}_d, \mathbf{D}_d\}$ is discrete strictly positive real. A discrete state-space system is strictly positive real if there exists a positive definite matrix $\mathbf{P}_d \in \mathbb{R}^{n \times n}$, matrices $\mathbf{L} \in \mathbb{R}^{m \times n}$, $\mathbf{W} \in \mathbb{R}^{m \times m}$, and scalar $\delta > 0 \in \mathbb{R}^+$ such that

$$\mathbf{P}_d - \delta \mathbf{P}_d - \mathbf{A}_{c,d}^T \mathbf{P}_d \mathbf{A}_{c,d} = \mathbf{L}^T \mathbf{L} \quad (2.47)$$

$$\mathbf{C}_d - \mathbf{A}_{c,d}^T \mathbf{P}_d \mathbf{B}_d = \mathbf{L}^T \mathbf{W} \quad (2.48)$$

$$\mathbf{D}_d^T + \mathbf{D}_d - \mathbf{B}_d^T \mathbf{P}_d \mathbf{B}_d = \mathbf{W}^T \mathbf{W} \quad (2.49)$$

Similarly, the system is positive real in the case of $\delta = 0$, in which case

$$\mathbf{P}_d - \mathbf{A}_{c,d}^T \mathbf{P}_d \mathbf{A}_{c,d} = \mathbf{L}^T \mathbf{L} \quad (2.50)$$

$$\mathbf{C}_d - \mathbf{A}_{c,d}^T \mathbf{P}_d \mathbf{B}_d = \mathbf{L}^T \mathbf{W} \quad (2.51)$$

$$\mathbf{D}_d^T + \mathbf{D}_d - \mathbf{B}_d^T \mathbf{P}_d \mathbf{B}_d = \mathbf{W}^T \mathbf{W} \quad (2.52)$$

Corollary 2.3.0.2 *In general it is not true that a continuous time system with the SPR property has a discrete counterpart that is SPR.*

An additional consequence of the definition is that a discrete system cannot be ASPR if it has direct passthrough gain $\mathbf{D}_d = 0$ [49].

The discrete analogues of continuous time systems allow for many of the same techniques used for analysis of continuous systems to apply to discrete systems. However, care must always be taken when applying continuous time principles to discrete systems. Although discrete systems approach the continuous systems for sufficiently small timesteps, there are a variety of unexpected behaviours that can occur when the timestep is insufficiently small.

2.3.3 Example of Linear Analysis of Spacecraft Systems

Linear system analysis makes it simple to create complex and accurate descriptions of system behaviour from simple linear approximations of the systems involved. For example, a flowchart for the signal processing of a spacecraft might be similar to the one represented in Fig. 2.8.

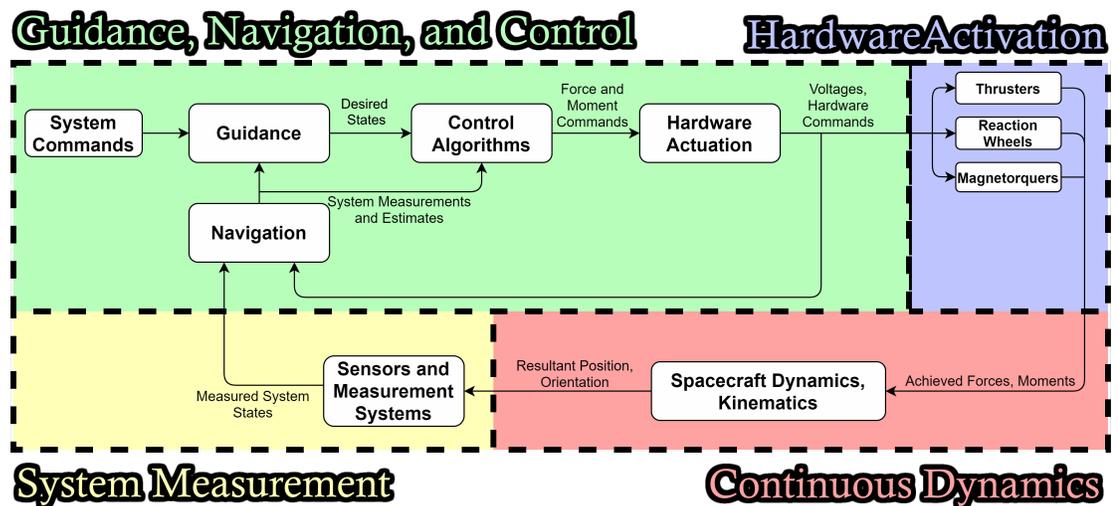


Figure 2.8: Example of Spacecraft Guidance, Navigation, and Control Processing

If, for control purposes, the measured signal is similar to the actual state of a system, and actuator dynamics are adequately described by a linear model, then the

entire system might be simplified to the block diagram seen in Fig. 2.9.

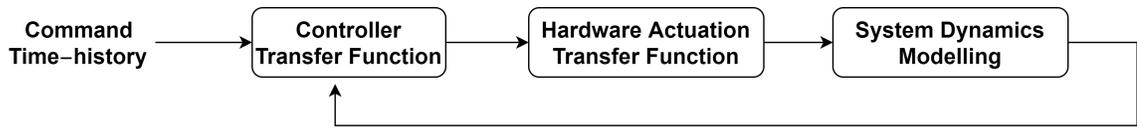


Figure 2.9: Simplified Spacecraft Processing

If all the blocks in the diagram are filled in with reasonably linear approximations of the real systems, the response of the system can be interpreted using linear design tools. A single-input-single-output (SISO) control loop for the position response of a spacecraft might resemble the diagram in Fig. 2.10. The position command $u_c(s)$ becomes the output $y(s)$ after it passes through the controller, actuators, and system dynamics.

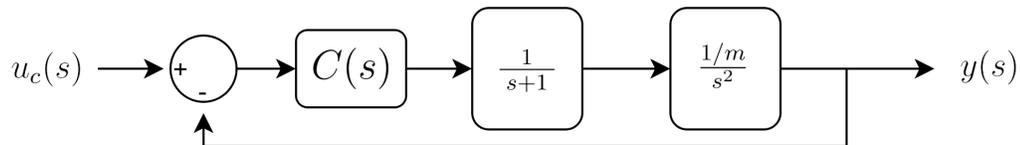


Figure 2.10: Simplified Spacecraft Processing

The frequency response of the system, its stability, and its response to a step command can all be probed to determine if the system reaches requirements. Notice, however, that the controller $C(s)$ has yet to be decided for the system. The system in Fig. 2.10 can be simplified to the transfer function

$$H_{tot}(s) = \frac{C(s)1/m}{s^3 + s^2 + C(s)1/m} \quad (2.53)$$

where the controller $C(s)$ for the system can still be chosen to achieve the desired performance. The selection of the controller $C(s)$ is the subject of linear controller design.

2.3.4 Linear Controller Design and Implementation

Once a system has been modelled to create a transfer function, state-space representation, or any other representation, it is up to the designer to determine what

controller designs are able to manage the relevant control signals to reach the desired performance. For linear systems it will be desirable to keep the final system linear in order to use the same linear design tools that are able to determine the performance of the uncontrolled system. Figure 2.10 and Eq. (2.53) show that the feedback controller $C(s)$ can be chosen to reach the desired performance of the final system, and could be composed of any number of transfer functions.

Practical limitations often reduce which controllers can be feasibly implemented. The current text aims to highlight the problems of implementation as they relate to adaptive controllers, so it will be helpful to touch on the implementation issues of linear controllers here to more easily highlight their solutions for adaptive systems later.

One method, called *full-state feedback control*, suggests it can achieve almost any performance desired by the designer for any linear system [20]. Each of the controllable poles and zeros of a full-state feedback controlled system can be moved to any location along the complex plane in Laplace space, allowing for any number of responses to be achieved. In practice, this is not the case. Linear system representations do not include common nonlinearities, such as random noise, signal saturation, derivative noise, or signal delay, which all hamper performance of full-state feedback controllers drastically.

The design of any linear controller $C(s)$ falls prey to similar issues as full-state feedback; any number of rational polynomials could be chosen as the controller transfer function to stabilize the plant, however the complex variable “ s ” represents a derivative of the input or output. In effect, the addition of each pole or zero in the controller to change the response of the original plant necessitates the introduction of a derivative or integral when calculating the desired control power. Since system measurements necessarily introduce noise into the controller, often only one or two derivatives of a signal can be taken before the noise on a signal’s derivative overpowers the information of that derivative. Similarly, without proper consideration, it is also possible for the introduction of integrators into calculations to continue integrating when control cannot be achieved, called *integrator windup*, which may degrade performance or cause instability. Due to these drawbacks, linear control designers have

settled on a simple and versatile controller design that uses minimal signal manipulation to allow for the production of a variety of controller designs and system responses for various applications.

The fundamental linear controller is known as the Proportional-Integral-Derivative (PID) controller, so called due to its control equations, which are given by

$$\mathbf{e}(t) = \mathbf{u}_c(t) - \mathbf{y}_p(t) \quad (2.54)$$

$$\mathbf{u}_p(t) = \mathbf{K}_p \mathbf{e}(t) + \mathbf{K}_I \int_0^t \mathbf{e}(t) dt + \mathbf{K}_d \dot{\mathbf{e}}(t) \quad (2.55)$$

for the output signal $\mathbf{y}_p \in \mathbb{R}^p$ to be controlled, the input command $\mathbf{u}_c \in \mathbb{R}^p$ to be tracked, system control power $\mathbf{u}_p \in \mathbb{R}^m$, and the constant proportional, integral, and derivative gains $\mathbf{K}_p, \mathbf{K}_i, \mathbf{K}_d \in \mathbb{R}^{m \times p}$, respectively. The Laplace transform of these equations lead to the transfer function, before feedback and creation of the signal $\mathbf{e}(t)$ of

$$\mathbf{C}(s) = \frac{\mathbf{U}_c(s)}{\mathbf{E}(s)} = \frac{\mathbf{K}_d s^2 + \mathbf{K}_p s + \mathbf{K}_i}{s} \quad (2.56)$$

Which has the ability to create two zeros to directly affect the plant response, and a gain by which to manage the placement of the poles and zeros through feedback. This controller is useful in physical system implementations, since it only requires a single derivative and integral, mitigating the effects of integral windup and derivative noise. The designer can choose the values of the three gain terms \mathbf{K}_p , \mathbf{K}_i , and \mathbf{K}_d which act as simple conversions from the error values to control activations. Although trying to quantify the meaning of the gains, such as translating from position offsets to engine voltages, may seem nonsensical, the trajectories followed by PID controllers manage to outperform humans in some applications [21].

The PID controller effectively manages a large variety of physical systems; both Newtonian double integrator dynamics and rotational dynamics can be managed by PID control. Most systems can be adequately controlled by a well-tuned PID [11].

While designing a PID controller for some system, its natural to ask “is there a *best* controller for this system?” Linear Quadratic Regulators, or LQRs, are a class

of linear controller that are provably optimal for linear systems and the cost function

$$O = \int_0^t \mathbf{x}_p^T(t) \mathcal{Q} \mathbf{x}_p(t) + \mathbf{u}_p^T(t) \mathcal{R} \mathbf{u}_p(t) dt \quad (2.57)$$

where $O \in \mathbb{R}$ denotes the final cost of the system, $\mathcal{Q} \in \mathbb{R}^{n \times n}$ is the error weight matrix, and $\mathcal{R} \in \mathbb{R}^{p \times p}$ is the control activation weight matrix. It should be noted that sometimes a cross-coupling term \mathcal{N} is included in the cost function, however this is usually omitted. Equation (2.57) is known as the linear quadratic cost function, and the controller that optimizes its value attempts to lower the magnitude of the system states as fast as possible while also minimizing the control activations required to reach those states. It should be mentioned that LQR formulations originally required that the controller drive the system to a zero-state $\mathbf{x}_p = \mathbf{0}$, hence the inclusion of the term “regulator” and not “controller”, however the controller formulation equations are equally applicable to a system where the error is driven to zero instead of the state and varies with the input control command.

Theorem 2.3.1 (Linear Quadratic Regulator [50]) *For a system $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$ and the associated cost function*

$$O = \int_0^t \mathbf{x}_p^T(t) \mathcal{Q} \mathbf{x}_p(t) + \mathbf{u}_p^T(t) \mathcal{R} \mathbf{u}_p(t) dt \quad (2.58)$$

the matrix of constant feedback gains $\mathbf{K}_{LQR} \in \mathbb{R}^{p \times n}$ in the control action

$$\mathbf{u}_p = -\mathbf{K}_{LQR} \mathbf{x}_p \quad (2.59)$$

that minimizes the objective function is found through the solution to the equations

$$\mathbf{K}_{LQR} = \mathcal{R}^{-1} \mathbf{B}^T \mathbf{P}_{LQR} \quad (2.60)$$

$$\mathbf{A}^T \mathbf{P}_{LQR} + \mathbf{P}_{LQR} \mathbf{A} + \mathcal{Q} = \mathbf{P}_{LQR} \mathbf{B} \mathcal{R}^{-1} \mathbf{B}^T \mathbf{P}_{LQR} \quad (2.61)$$

The solution for \mathbf{K}_{LQR} depends on \mathbf{P}_{LQR} , which is found through the solution of the algebraic Riccati equation in Eq. (2.61) [51].

REMARK The fact that Eq. (2.60) and (2.61) correspond to the solution to the LQR system is a fundamental result of control system theory that took many years to achieve. The proofs underlying this result are long, however the most accessible paper on proving Theorem 2.3.1 is given in J. Willem's *Least Squares Stationary Optimal Control and the Algebraic Riccati Equation* [52]. Other useful texts for deeper dives into this subject are V. Mehrmann's *The Autonomous Linear Quadratic Control Problem: Theory and Numerical Solution* [50], and any linear analysis textbook, such as *A Linear System's Primer* [20].

The solutions to the algebraic Riccati equation, Eq. (2.61), still need to be determined in order for an LQR to be found.

Theorem 2.3.2 (Schur Method for Solving Algebraic Riccati Equations [53])

The solution to the algebraic Riccati equation

$$\mathbf{A}^T \mathbf{P}_{LQR} + \mathbf{P}_{LQR} \mathbf{A} + \mathbf{Q} = \mathbf{P}_{LQR} \mathbf{B} \mathcal{R}^{-1} \mathbf{B}^T \mathbf{P}_{LQR} \quad (2.62)$$

for matrix \mathbf{P}_{LQR} can be found using the eigenvalue decomposition of a related Hermitian matrix

$$\mathbf{Z} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \mathcal{R}^{-1} \mathbf{B}^T \\ -\mathbf{Q} & -\mathbf{A}^T \end{bmatrix} \quad (2.63)$$

The Schur decomposition of \mathbf{Z} is given by

$$\mathbf{Z} = \mathbf{U} \boldsymbol{\lambda} \mathbf{U}^T \quad (2.64)$$

for the upper-triangular matrix $\boldsymbol{\lambda} \in \mathbb{R}^{2n \times 2n}$ and matrix of basis vectors $\mathbf{U} \in \mathbb{R}^{2n \times 2n}$. The matrix \mathbf{U} corresponds to the basis vectors of \mathbf{Z} , and can be broken down into the blocks

$$\mathbf{U} = \begin{bmatrix} \mathbf{U}_{1,1} & \mathbf{U}_{1,2} \\ \mathbf{U}_{2,1} & \mathbf{U}_{2,2} \end{bmatrix} \quad (2.65)$$

Each of dimension $n \times n$. The algebraic Riccati equation corresponding to Eq. (2.62) and therefore \mathbf{Z} is solved by the composition of basis vectors

$$\mathbf{P}_{LQR} = \mathbf{U}_{2,1} \mathbf{U}_{1,1}^{-1} \quad (2.66)$$

REMARK Other methods of solving the algebraic Riccati equation exist, such as finite time optimization methods and iterative solution techniques. These methods are not of particular note to typical control systems, but have been leveraged in infinite dimension control problems [54] and other complex control systems [55].

Numerical methods to solve the algebraic Riccati equation have been implemented into popular mathematical software, and solutions to this equation can be easily accessed. Once the \mathbf{P}_{LQR} matrix is found through the Riccati equation solution, the state-feedback matrix \mathbf{K}_{LQR} can be solved through Eq. (2.60) to determine the optimal gains for the system.

The solution to the LQR problem may require feedback of states, however in many cases these states are accessible directly from measurements. For example, the use of the cost weighing matrices

$$\mathcal{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \mathcal{R} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (2.67)$$

to determine an LQR for the double integrator state-space system modelled by the second-order state space model

$$\begin{bmatrix} \dot{x}_p \\ \ddot{x}_p \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_p \\ \dot{x}_p \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} F \quad (2.68)$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_p \\ \dot{x}_p \end{bmatrix} + [0]F \quad (2.69)$$

of input force F , mass m , states x_p and \dot{x}_p with output position y only requires knowledge of the position and velocity, which are both available from position measurements and their derivatives. Furthermore, only the ratio of the weighing matrices \mathcal{Q} and \mathcal{R} affect the output gains of the LQR design equations.

Implementations of LQR have some drawback; improper selection of the weighing matrices may suggest gains that are drastically too large for the final system in the presence of noise and nonlinearities, or gains with an unacceptably slow response. Although the optimality of the controller is guaranteed for linear control formulations, nothing is guaranteed in terms of the system robustness. Indeed, by attempting to maximize regulation of the control signal and minimize perturbations due to disturbances, the gains recommended by LQR formulations tend to be larger than designers would normally recommend for a system. Additionally, the robustness of LQR designs to plant uncertainty is usually worse than other techniques are able to achieve, again, due to the somewhat larger gains.

Nevertheless, LQR formulations allow for a quick way to produce optimal controllers that can be changed if necessary to meet other design requirements.

2.4 Nonlinear System Analysis

Although many systems are well approximated by linear state-space formulations, almost no systems is fully linear. It might be possible to say a car is well approximated by a state-space system, but attempting to accelerate it from 0 to 60 km/h in one second will demand an incredible acceleration that cannot possibly be achieved by the physical system. In practice, all physical systems have this limitation.

The name of the field itself is unhelpfully broad and obtuse. Systems with predictable and regular dynamics, like

$$\dot{x} = -x \cdot |x| \tag{2.70}$$

are placed alongside much more complex systems such as the Duffing equation

$$\ddot{x} + d\dot{x} + ax + bx^3 = c \cos(\omega t) \tag{2.71}$$

with radically different and chaotic behaviour, both of which are demonstrated in Fig. 2.11.

Without a specific dynamic system in mind and tools that are able to characterize the response of that nonlinear system, very little can be said about nonlinear systems

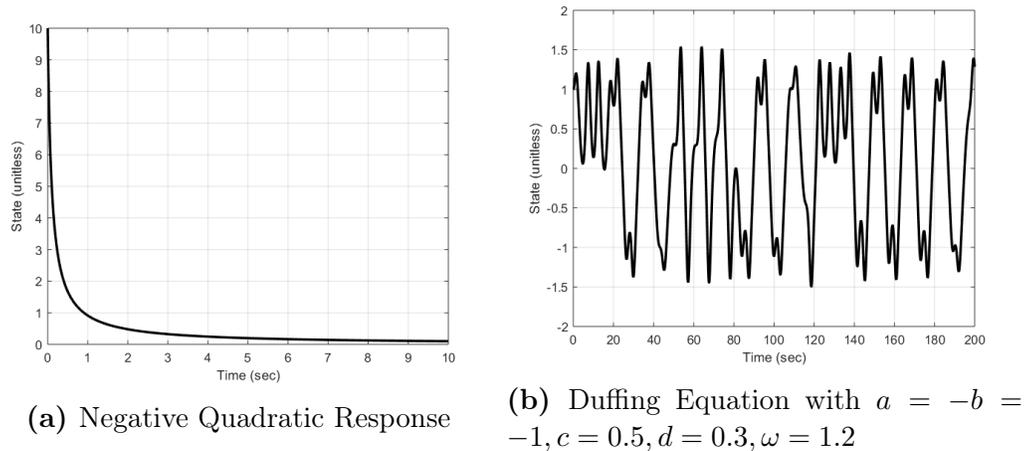


Figure 2.11: Examples of Nonlinear Systems

in general.

Nevertheless, dynamicists and control engineers have spent a great deal of time characterizing and analyzing nonlinearities in order to understand their impact on dynamic systems and their responses.

There is one tool that is widely applicable to nonlinear systems and allows for stability analyses to be performed on even the most obtuse dynamic system.

Lyapunov stability theorem was pioneered by Aleksandr Mikhailovich Lyapunov in 1892, however it did not see active usage until the cold war, when it was used for stability analysis of guidance systems in the Soviet Union. Lyapunov’s work made its way to North American academic circles in the late 1900s, where it was embraced for its amazing ability to describe the behaviour of a wide array of nonlinear systems. It is Lyapunov stability theory that will prove that adaptive systems are stable.

Fundamentally, Lyapunov stability theorem asks questions about the trajectories of some notion of “energy” in a system, called the Lyapunov function.

Theorem 2.4.1 (Lyapunov Stability Theorem [24]) *A function $V : \mathcal{D} \rightarrow \mathbb{R}$ is a positive definite function in \mathcal{D} when [24]:*

- $V(0) = 0, 0 \in \mathcal{D}$
- $V(x) > 0, \forall x \in \mathcal{D} - \{0\}$
- $|x| \rightarrow \infty \implies V(x) \rightarrow \infty$

That is to say that the function is greater than 0 for all points that are not the origin in \mathcal{D} , and 0 at the origin. If more points than just the origin in \mathcal{D} are mapped to a value of 0 then the function is said to be positive semi-definite.

For the system $\dot{x} = f(x)$, $\mathcal{D} \subset \mathbb{R}^n$, $f : \mathcal{D} \rightarrow \mathbb{R}^n$, and equilibrium point $\mathbf{x}_e = \mathbf{0}$, $\dot{\mathbf{x}} = f(\mathbf{x}_e) = \mathbf{0}$, if there exists a continuous Lyapunov function $V : \mathcal{D} \rightarrow \mathbb{R}$ such that

- $V(0) = 0$, $0 \in \mathcal{D}$
- $V(x) > 0$, $\forall x \in \mathcal{D} - \{0\}$
- $|x| \rightarrow \infty \implies V(x) \rightarrow \infty$
- $\frac{d}{dt}V(x) < 0$, $\forall x \in \mathcal{D} - \{0\}$

then the system must be asymptotically stable and the system must approach the equilibrium \mathbf{x}_e .

REMARK Lyapunov described his theorem in terms of system “energies”, which provides an intuitive method of understanding the theorem. The Lyapunov function takes as inputs all of the system states, and is greater than zero at all points excepting the origin. Similar to energy in classical dynamic systems, the Lyapunov function maps any one set of states to a single scalar value. It must also be true that larger system states create larger Lyapunov function values, approaching infinity as the states approach infinity. Since the Lyapunov function is at its minimum at the origin, if it can be shown that the value of the Lyapunov function through time is always decreasing, then it follows that the system states must reach the lowest point on the Lyapunov function, which is uniquely the origin.

Similar to proving that the energy of a classic system is always decreasing implies that the system will reach its lowest energy state, Lyapunov’s notion of stability says that declaring a system energy, then proving that energy is always decreasing with time implies the system must reach its lowest energy state.

Corollary 2.4.1.1 *In order for a system to be stable in the sense of Lyapunov, it is required that the time derivative of the Lyapunov function be dependant on all system states. If a state is not represented in the Lyapunov function or its derivative, the Lyapunov stability theorem cannot show that state must reach equilibrium.*

Amazingly, the Lyapunov function $V(x)$ of the system only needs to be dependent on the system states, and does not need to be any representations of what those states are, how they interact, or how they propagate. The information of how the dynamics of the system change is contained in the value $\frac{d}{dt}V(x)$. Stability of the equilibrium point at the origin is sufficiently proven by monitoring the output of the positive definite Lyapunov function, and showing that its output must always be decreasing in time.

An example application of Lyapunov's stability theorem is completed here. The example system $\dot{\mathbf{x}} = f(\mathbf{x})$ has update equations for states x_1 and x_2 of

$$\dot{x}_1 = x_2 \tag{2.72}$$

$$\dot{x}_2 = -x_1 - 0.1 \cdot x_2^3 \tag{2.73}$$

The Lyapunov function for this system might be chosen as

$$V(x) = x_1^2 + x_2^2 \tag{2.74}$$

then the derivative of the Lyapunov function would be

$$\dot{V}(x) = 2x_1\dot{x}_1 + 2x_2\dot{x}_2 \tag{2.75}$$

$$= 2x_1x_2 - 2x_2x_1 - 2 \cdot 0.1 \cdot x_2^4 \tag{2.76}$$

$$= -2 \cdot 0.1 \cdot x_2^4 \tag{2.77}$$

which is negative for all values of x_2 . The system response to the initial condition of $[x_1, x_2]^T = [10, 0]^T$ is shown in Fig. 2.12a.

Notably, in this case all that has been proven is that the Lyapunov function decreases for all values of x_2 , following corollary 2.4.1.1, nothing can be said about x_1 . For the situations where Lyapunov is not sufficient, Lasalle's invariance theorem can also be used.

Theorem 2.4.2 (Lasalle's Invariance Principle [56]) For $V : \mathbb{R}^n \rightarrow \mathbb{R}$, and $f :$

$\mathcal{D} \rightarrow \mathbb{R}^n$ both C^1 continuous functions, suppose that $\dot{V}(x) = \langle \nabla V(x), f(x) \rangle \leq 0$ for all $x \in \mathcal{U} \subset \mathbb{R}^n$. We define $\mathcal{E} = \{x \in \mathcal{U} : \dot{V} = 0\}$ to be the invariant set of V . If the largest invariant set in \mathcal{E} is called \mathcal{G} , then every solution of $\dot{x} = f(x)$ that is bounded and in \mathcal{U} must converge to \mathcal{G} for $t \geq 0$ and as $t \rightarrow +\infty$.

REMARK The set \mathcal{E} in Lasalle's invariance principle is the set of points where the value of \dot{V} is zero. The set \mathcal{E} , then, is all the points along the Lyapunov function where Lyapunov's stability theorem cannot tell us how the system propagates, since the Lyapunov function derivative does not change its value.

What Lasalle proposes is to look at the set \mathcal{E} , where nothing can be ascertained from the Lyapunov function, and consider how states that begin in \mathcal{E} must evolve. Considering all the states in \mathcal{E} , there must be some set that, once entered, states no longer exit. The largest such set, where states do not exit once they entered, is called \mathcal{G} . It does not matter what the states do once they are in \mathcal{G} , whether they approach a limit cycle, stay static, or chaotically vary, so long as they do not exit \mathcal{G} once it has been entered. The set \mathcal{G} , furthermore, must be a subset of \mathcal{E} , since if the set \mathcal{G} had elements with nonzero Lyapunov function derivatives, then they would no longer be invariant or a member of \mathcal{G} . What Lasalle was able to show is that if the set \mathcal{G} can be determined, then all members of the set \mathcal{E} must eventually enter and stay in \mathcal{G} .

By determining the size of the set \mathcal{G} the set of final system values can also be determined. The terminology can be confusing, so application of LaSalle's invariance principle will be clarified through continuing the nonlinear system example.

Theorem 2.4.3 *The example system in Eq. (2.72) and (2.73) is stable to the state $x_1 = x_2 = 0$.*

PROOF The invariant set of the Lyapunov function is the set of states such that

$$\dot{V}(x) = 0 \tag{2.78}$$

which implies by Eq. (2.77) that

$$x_2 = 0 \tag{2.79}$$

There are two cases then

Case 1 $x_1 \neq 0$ implies that $\dot{x}_2 \neq 0$ in Eq. (2.73), therefore the system leaves the invariant set.

Case 2 $x_1 = 0$, then the system is stable to the point $x_1 = x_2 = 0$.

Since the system either leaves the set of $\dot{V} = 0$ when $x_1 \neq 0$, or stays at the point $x_1 = x_2 = 0$, the largest stable invariant set is the origin of phase space, therefore the system must be stable to the point $x_1 = x_2 = 0$ following Lasalle's invariance principle. ■

The results of the above proof can also be seen in Fig. 2.12, where the system trajectory oscillates, while approaching zero position and velocity. Although the system approaches zero position and velocity more and more slowly with each second of simulation, Lyapunov stability theory and Lasalle's invariance principle ensures us that the system must reach equilibrium as time approaches infinity.

The final tool in the nonlinear analysis arsenal is Barbalat's lemma, which states that

Theorem 2.4.4 (Barbalat's Lemma [57]) *For the function $f : \mathbb{R} \rightarrow \mathbb{R}^n$, and where $\lim_{t \rightarrow \infty} f(t)$ is finite for $\dot{f}(t)$ uniformly continuous (or, equivalently, $\ddot{f}(t)$ bounded), then $\dot{f}(t) \rightarrow 0$ as $t \rightarrow \infty$.*

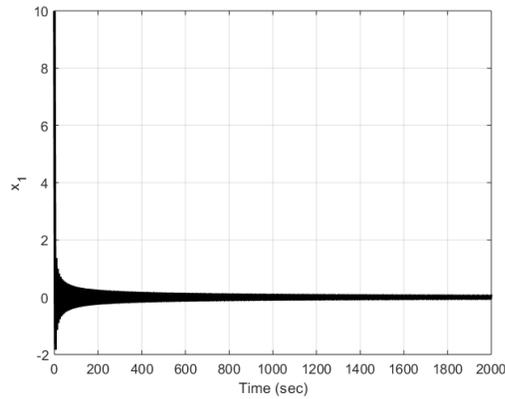
Barbalat's lemma can be useful when determining if a system is stable when other assumptions about the form of the dynamic equations cannot be made.

Lyapunov's stability theorem can also be applied to discrete systems, with a similar formulation and justification.

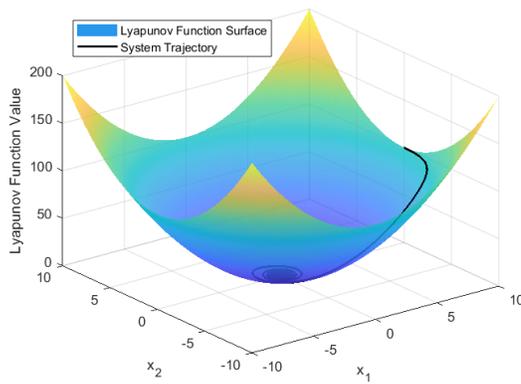
Theorem 2.4.5 (Discrete Lyapunov Stability Theorem [56]) *For the discrete autonomous system of iteration $k \in \mathbb{Z}$ and update equation*

$$\mathbf{x}(k+1) = f(\mathbf{x}(k)) \tag{2.80}$$

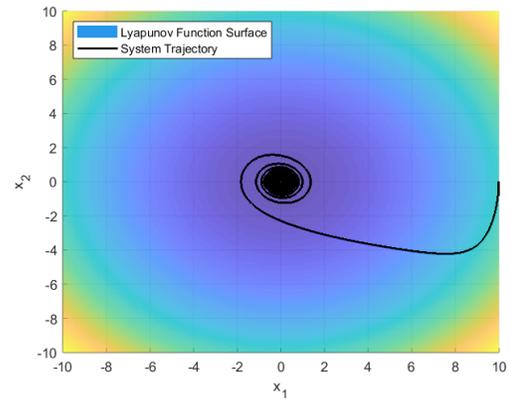
for $f : \mathcal{D} \rightarrow \mathbb{R}^n$, then for a positive definite function $V : \mathcal{D} \rightarrow \mathbb{R}$ such that



(a) Timeseries



(b) Lyapunov Function



(c) Overhead View

Figure 2.12: Nonlinear Spring Example, with $x_0 = [10, 0]^T$

$$V(f(\mathbf{x}(k))) - V(\mathbf{x}(k)) = \Delta V \leq 0, \quad \forall \mathbf{x} \in D \quad (2.81)$$

then $\mathbf{x} = 0$ is stable. And if the further condition

$$V(f(\mathbf{x}(k))) - V(\mathbf{x}(k)) = \Delta V < 0, \quad \forall \mathbf{x} \in D = \{\mathbf{0}\} \quad (2.82)$$

is met, then the system is asymptotically stable.

REMARK Similar to the continuous Lyapunov stability theorem, proving that the

discrete energy of the system is always decreasing for a positive definite Lyapunov function implies the system reaches the origin.

Lyapunov's stability theorem, Lasalle's invariance principle, and Barballat's lemma together are able to provide powerful insights into the behaviour of systems with nonlinear dynamics.

2.5 Adaptive Control Design

The advances to nonlinear system analysis made in the wake of Lyapunov's stability theorem enabled control systems engineers to develop controller architectures that departed from the standard linear control design. Now that the stability of a system could be guaranteed even when nonlinear controller formulations were present, a large number of controllers could be dreamt up, their stability proved through Lyapunov's method, and their performance probed.

Some popular examples of these nonlinear controllers include the sliding mode controller, backstepping controllers, and most importantly for our purposes, adaptive controllers.

Adaptive controllers are a class of controller that change in time to achieve performance objectives despite variable or uncertain system dynamics. The knowledge a designer has about the system he is controlling is known as *a priori* knowledge, and in adaptive controllers the primary objective is to develop a controller architecture or design that is able to reach strong performance while minimizing *a priori* knowledge. Adaptive control objectives differ somewhat subtly from robust control objectives, in that both deal with uncertainty in the plant response: robust control aims to manage uncertainty in a predefined range through a single versatile time-invariant controller, whereas adaptive control aims to reach the desired performance by varying the controller parameters in time. One possibility of adaptive control, for example, is to reach the designed performance exactly for all configurations of the plant, whereas robust control must accept some degradation whenever the plant is not nominal.

Adaptive controllers come in two major varieties, indirect and direct adaptive controllers. Indirect adaptive control updates from measurements of system parameters

that affect the control response, effectively adapting the controller through inference or measurement of the system states. Indirect adaptive control evolves naturally from situations in which changes in plant dynamics can be directly or indirectly measured.

In aircraft, for example, the control surfaces can produce more force or torque at higher airspeeds; as the aircraft reaches larger velocities, the achievable aerodynamic forces of each control surface is increased proportionally to the square of the velocity. Since the velocity of the aircraft can, and usually is, measured during flight, it is simple to include velocity in the aircraft control formulation to ensure that the commanded control deflection corresponds to the desired torques or forces on the aircraft. The control law adapts its commanded output to compensate for variability in the response of the plant to measurable variations in airspeed. More complicated arrangements of measurements and prediction can be used to determine other major or minor effects on the control response of the plant to varying parameters, however all of these are rooted in a very simple algorithm: using *a priori* knowledge of the plant dynamics, measure, predict, or synthesize knowledge of those properties that affect the plant dynamics, then compensate for those effects.

Indirect adaptive control includes industry standard techniques such as feedback linearization and gain scheduling. Indirect adaptive control efforts may also rely on more complex techniques, such as Kalman filtering, signal fusion, or nonlinear dynamics observers, to measure or predict values of the relevant parameters when they cannot be directly measured.

Indirect adaptive control relies fundamentally on knowledge of the structure of the plant. Although control engineers typically deal with systems whose dynamics have been rigorously analyzed, measured, or tested, there are yet other situations where the properties of a system cannot be directly measured and must instead be managed.

Direct adaptive control methodologies seek to minimize *a priori* knowledge of a system while still maintaining adequate control performance. Control is achieved by knowledge of the form of the system dynamics only, and proving that the dynamics of the controller are stable in transitioning from the initial controller to a final system which matches a desired response.

2.5.1 Simple Adaptive Control Formulation

Direct adaptive control methodologies have one simple goal: given a linear state-space system and some adaptive controller, make the response of the linear plant approach an ideal response. The simple adaptive control (SAC) architecture will be developed here, and it will be shown how a linear plant response can be made to approach an ideal response.

The state-space model $\{\mathbf{A}_p, \mathbf{B}_p, \mathbf{C}_p, \mathbf{0}\}$ of the plant is represented by

$$\dot{\mathbf{x}}_p(t) = \mathbf{A}_p \mathbf{x}_p(t) + \mathbf{B}_p \mathbf{u}_p(t) \quad (2.83)$$

$$\mathbf{y}_p(t) = \mathbf{C}_p \mathbf{x}_p(t) \quad (2.84)$$

For plant order $n \in \mathbb{Z}$, inputs $m \in \mathbb{Z}$, and identical size of outputs m , to yield system matrices of sizes $\mathbf{A}_p \in \mathbb{R}^{n \times n}$, $\mathbf{B}_p \in \mathbb{R}^{n \times m}$, and $\mathbf{C}_p \in \mathbb{R}^{m \times n}$. The desired system that the adaptive controller will attempt to match is called the *ideal model* $\{\mathbf{A}_m, \mathbf{B}_m, \mathbf{C}_m, \mathbf{0}\}$, with update equation

$$\dot{\mathbf{x}}_m(t) = \mathbf{A}_m \mathbf{x}_m(t) + \mathbf{B}_m \mathbf{u}_c(t) \quad (2.85)$$

$$\mathbf{y}_m(t) = \mathbf{C}_m \mathbf{x}_m(t) \quad (2.86)$$

model order $q \in \mathbb{Z}$ alongside input and output of sizes of m . The signal dimensions for the reference model are thus $\mathbf{x}_m \in \mathbb{R}^q$ states, $\mathbf{u}_m \in \mathbb{R}^m$ inputs, and $\mathbf{y}_m \in \mathbb{R}^m$ outputs, as well as $\mathbf{A}_m \in \mathbb{R}^{q \times q}$, $\mathbf{B}_m \in \mathbb{R}^{q \times m}$, and $\mathbf{C}_m \in \mathbb{R}^{m \times q}$.

The error between the real-system response and the ideal response is given by

$$\mathbf{e}_y(t) = \mathbf{y}_m(t) - \mathbf{y}_p(t) \quad (2.87)$$

called the tracking error. The tracking error is the primary signal that will help determine how the adaptive control gains should be modified to approach the ideal model.

Since the ideal model is known either from precomputation or during onboard

computation, the reference model states \mathbf{x}_m and the reference model output \mathbf{y}_m are available to be used by the controller. The available signals are thus the tracking error \mathbf{e}_y , model states \mathbf{x}_m , and model command input \mathbf{u}_c . These three signals can be used alongside adaptive gains to produce a time-varying command to be sent to the plant. Such an example control signal would be constructed following

$$\mathbf{u}_p(t) = \mathbf{K}_e(t)\mathbf{e}_y(t) + \mathbf{K}_x(t)\mathbf{x}_m(t) + \mathbf{K}_u(t)\mathbf{u}_c(t) \quad (2.88)$$

with time-varying gains $\mathbf{K}_e \in \mathbb{R}^{p \times m}$, $\mathbf{K}_x \in \mathbb{R}^{p \times q}$, and $\mathbf{K}_u \in \mathbb{R}^{p \times m}$. Now that each of the relevant control signals is being multiplied by an appropriately sized gain to produce a control signal, all that remains is to leverage knowledge of the system response to create a stable command.

First, the gain is split two portions, one of which will perform adaptation to adapt the gains to match the ideal model, while the other will be used to improve convergence only. By analogy, these two portions of the adaptive gain are called the *proportional adaptive* part, given by the subscript “*P*”, and the *integral adaptive* part given by the subscript “*I*”. The proportional and integral adaptive gains are summed to produce the total gain for either the tracking error, model states, or input command signals, following

$$\mathbf{K}_e(t) = \mathbf{K}_{eP}(t) + \mathbf{K}_{eI}(t) \quad (2.89)$$

$$\mathbf{K}_x(t) = \mathbf{K}_{xP}(t) + \mathbf{K}_{xI}(t) \quad (2.90)$$

$$\mathbf{K}_u(t) = \mathbf{K}_{uP}(t) + \mathbf{K}_{uI}(t) \quad (2.91)$$

where the proportional gains follow the adaptation rule of

$$\mathbf{K}_{eP}(t) = \mathbf{e}_y(t)\mathbf{e}_y^T(t)\mathbf{\Gamma}_{eP} \quad (2.92)$$

$$\mathbf{K}_{xP}(t) = \mathbf{e}_y(t)\mathbf{x}_m^T(t)\mathbf{\Gamma}_{xP} \quad (2.93)$$

$$\mathbf{K}_{uP}(t) = \mathbf{e}_y(t)\mathbf{u}_c^T(t)\mathbf{\Gamma}_{uP} \quad (2.94)$$

for some proportional adaptation parameters to be chosen by the designer $\mathbf{\Gamma}_{eP} \in \mathbb{R}^{m \times m}$, $\mathbf{\Gamma}_{xP} \in \mathbb{R}^{q \times q}$, and $\mathbf{\Gamma}_{uP} \in \mathbb{R}^{m \times m}$.

Notice that the inclusion of $\mathbf{e}_y(t)$ and $\mathbf{u}_c(t)$ in the adaptation for \mathbf{K}_{eP} , \mathbf{K}_{xP} , and \mathbf{K}_{uP} requires that the number of SAC outputs \mathbf{u}_p match the number of inputs \mathbf{y}_p . To clarify why the plant must be square, consider the case where only proportional adaptation is used, the equation must expand to become

$$\mathbf{u}_p(t) = \mathbf{e}_y(t)\mathbf{e}_y^T(t)\mathbf{\Gamma}_{eP}\mathbf{e}_y(t) + \mathbf{e}_y(t)\mathbf{x}_c^T(t)\mathbf{\Gamma}_{xP}\mathbf{x}_m(t) + \mathbf{e}_y(t)\mathbf{u}_c^T(t)\mathbf{\Gamma}_{uP}\mathbf{u}_c(t) \quad (2.95)$$

where the dimensions of \mathbf{u}_p must now match the dimensions of each of the terms on the right, which can only be true if in all cases the tracking error \mathbf{e}_y has the same size as the output command \mathbf{u}_p . Similar reasoning also holds for the integral adaptations that follow.

The integral portion follows similar adaptation rules

$$\dot{\mathbf{K}}_{eI}(t) = \mathbf{e}_y(t)\mathbf{e}_y^T(t)\mathbf{\Gamma}_{eI} \quad (2.96)$$

$$\dot{\mathbf{K}}_{xI}(t) = \mathbf{e}_y(t)\mathbf{x}_c^T(t)\mathbf{\Gamma}_{xI} \quad (2.97)$$

$$\dot{\mathbf{K}}_{uI}(t) = \mathbf{e}_y(t)\mathbf{u}_c^T(t)\mathbf{\Gamma}_{uI} \quad (2.98)$$

for some integral adaptation parameters to be chosen by the designer $\mathbf{\Gamma}_{eI} \in \mathbb{R}^{m \times m}$, $\mathbf{\Gamma}_{xI} \in \mathbb{R}^{q \times q}$, and $\mathbf{\Gamma}_{uI} \in \mathbb{R}^{m \times m}$. Both the adaptive portion of the gain and the integral portion of the gain have the same sizes.

Some simplifications can be made to the system to clarify its operation. The tracking error, ideal model states, and ideal model input can be compacted together to create a single signal called the *reference signal* $\mathbf{r}(t)$ following

$$\mathbf{r}(t) = \begin{bmatrix} \mathbf{e}_y(t) \\ \mathbf{x}_m(t) \\ \mathbf{u}_c(t) \end{bmatrix} \quad (2.99)$$

which simplifies the SAC design equations to

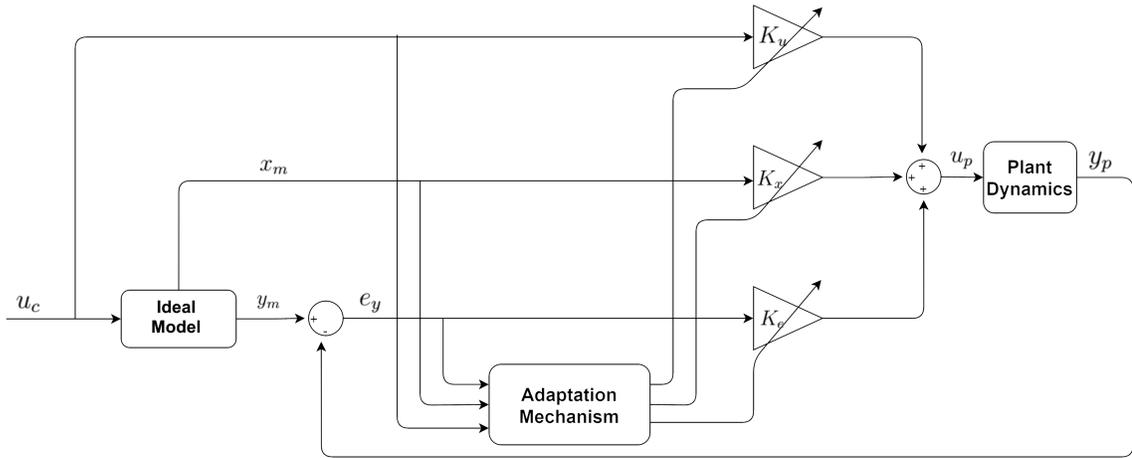


Figure 2.13: Block Scheme Diagram for a SAC

$$\mathbf{u}_p(t) = \mathbf{K}(t)\mathbf{r}(t) \quad (2.100)$$

$$\mathbf{K}(t) = \mathbf{K}_P(t) + \mathbf{K}_I(t) \quad (2.101)$$

$$\mathbf{\Gamma}_P = \begin{bmatrix} \mathbf{\Gamma}_{eP} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Gamma}_{xP} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{\Gamma}_{uP} \end{bmatrix} \quad (2.102)$$

$$\mathbf{\Gamma}_I = \begin{bmatrix} \mathbf{\Gamma}_{eI} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Gamma}_{xI} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{\Gamma}_{uI} \end{bmatrix} \quad (2.103)$$

$$\mathbf{K}_P(t) = \mathbf{e}_y(t)\mathbf{r}^T(t)\mathbf{\Gamma}_P \quad (2.104)$$

$$\dot{\mathbf{K}}_I(t) = \mathbf{e}_y(t)\mathbf{r}^T(t)\mathbf{\Gamma}_I \quad (2.105)$$

All that remains is for the designer to choose values of the individual proportional and integral adaptation parameters $\mathbf{\Gamma}_P$ and $\mathbf{\Gamma}_I$, and initial gain values.

The SAC architecture can be difficult to parse in equations alone. An example of the block scheme diagram for a SAC is presented in Fig. 2.13. The arrows overlapping the gains \mathbf{K}_e , \mathbf{K}_x , and \mathbf{K}_u refer to the adaptation of each gain in time using the adaptation mechanism, Eqs. (2.100) through (2.105).

On the surface, the SAC scheme appears to be a sensible way to determine the

relationship between the tracking error and a reference signal $\mathbf{r}(t)$. However, it is not clear by inspection if the SAC equations are stable. Significant work by Barkana [58] [59] [27] [47] [26] has been undertaken to show that this is indeed the case.

Theorem 2.5.1 (Continuous SAC Stability [47]) *For a linear time-invariant plant $\{\mathbf{A}_p, \mathbf{B}_p, \mathbf{C}_p, \mathbf{0}\}$ that is ASPR, the SAC control Eqs. (2.100) through (2.105) stabilize the system and result in perfect tracking*

PROOF The proof for stability in SAC is reproduced in full from [47] in Appendix C. After defining the ideal system state error

$$\mathbf{e}_x = \mathbf{x}_m - \mathbf{x}_p \quad (2.106)$$

and given ideal system gains \mathbf{K}^* it is shown that the Lyapunov function

$$V = \mathbf{e}_x^T(t) \mathbf{P} \mathbf{e}_x(t) + \text{Trace}[(\mathbf{K}(t) - \mathbf{K}^*)^T \mathbf{\Gamma}_I (\mathbf{K}(t) - \mathbf{K}^*)] \quad (2.107)$$

results in the Lyapunov function derivative

$$\dot{V} = -\mathbf{e}_x^T(t) \mathbf{Q} \mathbf{e}_x(t) \quad (2.108)$$

when using the ASPR system conditions, resulting in the positive-definite matrix \mathbf{Q} , which also implies \dot{V} is negative for all $|\mathbf{e}_x(t)| > 0$. ■

REMARK Following the previous discussion on Lyapunov stability proofs, the continuous time SAC stability proof is invariant under SAC gain $\mathbf{K}(t)$, ideal gain \mathbf{K}^* , and adaptation parameter $\mathbf{\Gamma}_I$. The stability result implies that any selection of $\mathbf{\Gamma}_I$ that is positive definite will result in a stable system, and that the system will be stable regardless of the gains $\mathbf{K}(t)$. Similarly, work by Barkana [47] has shown that ideal model following can occur even when the SAC gains do not reach the ideal gains.

The stability proof has also been done in previous works for the proportional adaptation, which does not affect the stability result, and does not clarify the stability analysis. The textbook *Direct Adaptive Control Algorithms: Theory and Applications*

[60] covers the development of fundamental proofs and theorems of SAC in great detail.

A discrete system proof of SAC stability is also available and reproduced in Appendix C. The proof shows that the system is boundedly stable in the discrete case. The conditions for discrete SAC stability are equivalent to their continuous counterpart. In order to be stable, a discrete SAC implementation must be controlling a discrete ASPR system.

The use of ASPR system properties in the stability proof means that it is necessary for the plant to be ASPR in order for the stability conditions to be met. The need for the plant to be ASPR can also be understood through how Eq. (2.105) makes adaptation possible at all.

Each of the terms for the gain adaptation in Eq. (2.105) make use of the outer product between the tracking error \mathbf{e}_y and an associated signal. Consider the form of a cross covariance matrix for two sets of data X and Y . The cross-covariance $\mathbf{K}_{X,Y}$ is given by

$$\mathbf{K}_{X,Y} = \text{cov}(X, Y) = \mathbf{E}[XY^T] - \mathbb{K} \quad (2.109)$$

where \mathbb{K} is a constant due to the mean of the cross-covariance. Notice that the calculation of the cross-covariance is dependant on the expected value of the outer product $\mathbf{E}[XY^T]$. It is computationally expensive to calculate the cross-covariance for two signals in a system at every timestep, however. The outer product used in the SAC gain adaptations behaves similarly to a rough approximation of the cross-covariance; if two signals are positively correlated the associated entry in the outer product will be positive, if two entries are negatively correlated their associated entry in the outer product will likewise be negative. Signals that are uncorrelated may have momentary spikes in the outer product but the integral of the outer product should nonetheless go to zero over time, since they are uncorrelated. By increasing the gain for each signal that has a correlation with the tracking error, the tracking error will decrease and the set of control gains that allow for zero tracking reference error will be approached.

However, it is not always true that the outer product of two correlated signals will

be in the same direction as the correlation. When two correlated signals are delayed from one another, it is possible for the outer product to return the wrong direction for the correlation. Figure 2.14 considers a system supplied with an input sinusoid that has a 1:1 correlated output sinusoid phase shifted by some amount ϕ . When the input and output sinusoid are only somewhat phase shifted, as in Fig. 2.14a, the overlap between both ensures that the integral of the outer product in Eq. (2.105) is in the same direction as the correlation; positive correlation leads to positive adaptation. In the case of the SAC, the correlation between these two signals will be used to modify the gain and decrease the tracking error. However, if the output sinusoid is sufficiently phase shifted, the value of the outer product throughout the period of one sinusoid decreases until at $\pm 90^\circ$ the integral of the outer product becomes zero. Figure 2.14b shows how the integral of the outer product goes to 0 when phase shifted by $\pi/2$ rad, and that further phase shifting leads to negative outer product integrals in Fig. 2.14c despite both signals being positively correlated.

There is one type of system ensures that there is overlap between an input signal and its resulting output. The frequency condition of ASPR systems from Corollary 2.3.0.1 ensures that no input frequency can be shifted in phase in the output by more or less than 90° , ensuring the SAC adaptation in Eq. (2.105) remains stable.

2.5.2 Negative Adaptation of Simple Adaptive Controllers

While SAC relies on correlating signals to the system response error, there remains a quirk to be addressed for imperfect systems. The tracking error gain adaptation following

$$\dot{\mathbf{K}}_{eI}(t) = \mathbf{e}_y(t)\mathbf{e}_y^T(t)\mathbf{\Gamma}_{eI} \quad (2.110)$$

incorporates the outer product of a signal with itself. Notably, the outer-product $\mathbf{e}_y(t)\mathbf{e}_y^T(t)$, much like the error covariance that it represents, must always be positive. Any input values for \mathbf{e}_y are effectively squared and must therefore always be positive. An unfortunate side-effect of implementation in real systems is the presence of noise, disturbances, mechanical errors, and any number of other factors that will mean that true reference model following such that $\mathbf{e}_y = \mathbf{0}$ cannot be achieved.

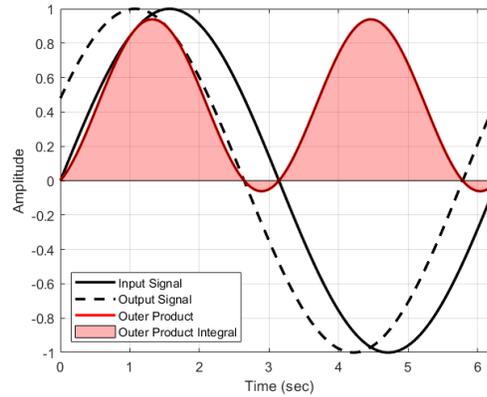
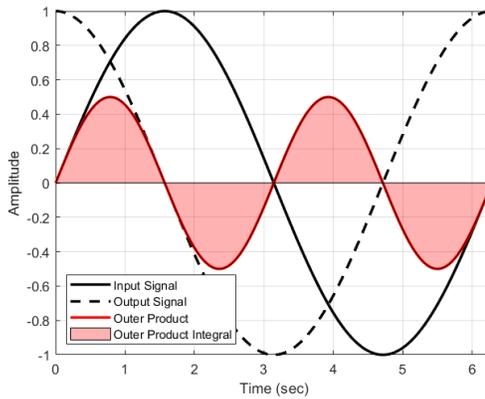
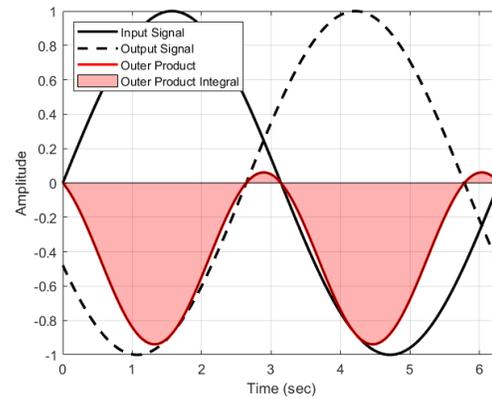
(a) $\phi = 0.5$ rad(b) $\phi = \pi/2$ rad(c) $\phi = \pi + 0.5$ rad

Figure 2.14: Correlated Signals Phase Shifted by ϕ . The Outer Product Integral Over a Wavelength is Small or Negative if Correlated Signals are Sufficiently Phase Shifted.

Unbounded adaptation of the tracking error gain \mathbf{K}_e is countered by a σ -modification. Fradkov [61] provided one of the first descriptions of this modification for SAC systems.

To combat the progressive accumulation of errors to this gain a regression term is added that causes a decrease in the gain value over time. The adaptation for each gain is then updated to include a regression term which produces the new gain adaptation equation

$$\dot{\mathbf{K}}_I(t) = \mathbf{e}_y(t)\mathbf{r}^T(t)\mathbf{\Gamma}_I - \mathbf{K}_I(t)\boldsymbol{\sigma} \quad (2.111)$$

for some regression term $\sigma \in \mathbb{R}^{m \times m}$. The inclusion of the regression term ensures that dynamics can maintain stability in the presence of noise, at the cost of causing the error tracking to be boundedly stable.

2.5.3 Model Matching Conditions

In order for SAC to achieve ideal model tracking, there must exist gains that allow for ideal model following to occur. Barkana [47] has derived matching conditions that must be satisfied in order for a known plant to be perfectly tracked by a known ideal model.

Theorem 2.5.2 (Simple Adaptive Control Matching Conditions [47]) *For the plant $\{\mathbf{A}_p, \mathbf{B}_p, \mathbf{C}_p, \mathbf{0}\}$ and model reference system $\{\mathbf{A}_m, \mathbf{B}_m, \mathbf{C}_m, \mathbf{0}\}$, composite matrices \mathbf{N} and \mathbf{M} are constructed following*

$$\mathbf{M} = \begin{bmatrix} \mathbf{A}_p & \mathbf{B}_p \\ \mathbf{C}_p & \mathbf{0} \end{bmatrix} \quad (2.112)$$

$$\mathbf{N} = \mathbf{M}^{-1} = \begin{bmatrix} \mathbf{N}_{11} & \mathbf{N}_{12} \\ \mathbf{N}_{21} & \mathbf{N}_{22} \end{bmatrix} \quad (2.113)$$

with \mathbf{M} provably nonsingular for an ASPR system [47]. Ideal model following occurs when the ideal model gains \mathbf{K}_x^* and \mathbf{K}_u^* satisfy the matching condition

$$\begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} \\ \mathbf{K}_x^* & \mathbf{K}_u^* \end{bmatrix} = \begin{bmatrix} \mathbf{N}_{11} & \mathbf{N}_{12} \\ \mathbf{N}_{21} & \mathbf{N}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{S}_{11}\mathbf{A}_m & \mathbf{S}_{11}\mathbf{B}_m \\ \mathbf{C}_m & \mathbf{0}_n \end{bmatrix} \quad (2.114)$$

equivalently given by the set of equations

$$\mathbf{N}_{11}\mathbf{S}_{11} - \mathbf{S}_{11}\mathbf{A}_m^{-1} = -\mathbf{N}_{12}\mathbf{C}_m\mathbf{A}_m^{-1} \quad (2.115)$$

$$\mathbf{S}_{12} = \mathbf{N}_{11}\mathbf{S}_{11}\mathbf{B}_m \quad (2.116)$$

$$\mathbf{K}_x^* = \mathbf{N}_{21}\mathbf{S}_{11}\mathbf{A}_m + \mathbf{N}_{22}\mathbf{C}_m \quad (2.117)$$

$$\mathbf{K}_u^* = \mathbf{N}_{21}\mathbf{S}_{11}\mathbf{B}_m \quad (2.118)$$

PROOF From [47]. For ideal output following to occur the plant output \mathbf{y}_p^* must match the model output \mathbf{y}_m . From the SAC architecture Eqs. (2.83) through (2.105),

$$\mathbf{y}_p^*(t) = \mathbf{C}_p \mathbf{x}_p^*(t) = \mathbf{C}_m \mathbf{x}_m(t) = \mathbf{y}_m(t) \quad (2.119)$$

for the ideal plant state trajectories \mathbf{x}_p^* . The time solution to continuous LTI system states is linear to inputs, which implies that the ideal model states are linear to the inputs. The ideal model gains, similarly, are constant, and take both the model states and model inputs to produce the input to the plant. It follows then that the ideal model states must be linear with respect to the model input and model states, which for linear matrices $\mathbf{S}_{11} \in \mathbb{R}^{n \times q}$ and $\mathbf{S}_{12} \in \mathbb{R}^{n \times m}$ are described as

$$\mathbf{x}_p^*(t) = \mathbf{S}_{11} \mathbf{x}_m(t) + \mathbf{S}_{12} \mathbf{u}_c(t) \quad (2.120)$$

Taking the derivative of Eq. (2.119), yields

$$\dot{\mathbf{y}}_p^*(t) = \mathbf{C}_p \dot{\mathbf{x}}_p^*(t) = \mathbf{C}_m \dot{\mathbf{x}}_m(t) = \dot{\mathbf{y}}_m(t) \quad (2.121)$$

which implies that $\dot{\mathbf{x}}_p^*$ must be linear with respect to $\dot{\mathbf{x}}_m$. The matrix of proportionality constants between $\dot{\mathbf{x}}_p^*$ and $\dot{\mathbf{x}}_m$ has already been defined as \mathbf{S}_{11} , and it follows that

$$\dot{\mathbf{x}}_p^* = \mathbf{S}_{11} \dot{\mathbf{x}}_m \quad (2.122)$$

$$\dot{\mathbf{x}}_p^* = \mathbf{S}_{11} \mathbf{A}_m \mathbf{x}_m + \mathbf{S}_{11} \mathbf{B}_m \mathbf{u}_c \quad (2.123)$$

$$\dot{\mathbf{x}}_p^* = \begin{bmatrix} \mathbf{S}_{11} \mathbf{A}_m & \mathbf{S}_{11} \mathbf{B}_m \end{bmatrix} \begin{bmatrix} \mathbf{x}_m \\ \mathbf{u}_c \end{bmatrix} \quad (2.124)$$

This provides one equation for the state derivative from the perspective of its relationship to the model input and states during perfect following. The same plant state update can now be considered from the commands of the SAC.

Since ideal model following only occurs when the tracking error is zero, we have that the ideal command output of the controller $\mathbf{u}_p^* \in \mathbb{R}^m$ is

$$\mathbf{u}_p^*(t) = \mathbf{K}_x^* \mathbf{x}_m(t) + \mathbf{K}_u^* \mathbf{u}_c(t) \quad (2.125)$$

which is applied to the plant update equation. Equation (2.120) is leveraged to produce the new state update equation

$$\dot{\mathbf{x}}_p^*(t) = \mathbf{A}_p \mathbf{S}_{11} \mathbf{x}_m(t) + \mathbf{A}_p \mathbf{S}_{12} \mathbf{u}_c(t) + \mathbf{B}_p \mathbf{K}_x^* \mathbf{x}_m(t) + \mathbf{B}_p \mathbf{K}_u^* \mathbf{u}_c(t) \quad (2.126)$$

$$\dot{\mathbf{x}}_p^*(t) = \begin{bmatrix} \mathbf{A}_p & \mathbf{B}_p \end{bmatrix} \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} \\ \mathbf{K}_x^* & \mathbf{K}_u^* \end{bmatrix} \begin{bmatrix} \mathbf{x}_m(t) \\ \mathbf{u}_c(t) \end{bmatrix} = \begin{bmatrix} \mathbf{S}_{11} \mathbf{A}_m & \mathbf{S}_{11} \mathbf{B}_m \end{bmatrix} \begin{bmatrix} \mathbf{x}_m(t) \\ \mathbf{u}_c(t) \end{bmatrix} \quad (2.127)$$

The requirement to match output values is included in the matrix equality in Eq. (2.127), before the identical vectors on both sides of the equation are dropped to yield the matching conditions

$$\begin{bmatrix} \mathbf{A}_p & \mathbf{B}_p \\ \mathbf{C}_p & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} \\ \mathbf{K}_x^* & \mathbf{K}_u^* \end{bmatrix} = \begin{bmatrix} \mathbf{S}_{11} \mathbf{A}_m & \mathbf{S}_{11} \mathbf{B}_m \\ \mathbf{C}_m & \mathbf{0} \end{bmatrix} \quad (2.128)$$

which must be met for model tracking to occur. The composite matrix \mathbf{M} is created for the system plant

$$\mathbf{M} = \begin{bmatrix} \mathbf{A}_p & \mathbf{B}_p \\ \mathbf{C}_p & \mathbf{0} \end{bmatrix} \quad (2.129)$$

which is provably nonsingular for an ASPR system [47]. The inverse of the composite matrix is now defined as

$$\mathbf{N} = \mathbf{M}^{-1} = \begin{bmatrix} \mathbf{N}_{11} & \mathbf{N}_{12} \\ \mathbf{N}_{21} & \mathbf{N}_{22} \end{bmatrix} \quad (2.130)$$

and updates the matching conditions to become

$$\begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} \\ \mathbf{K}_x^* & \mathbf{K}_u^* \end{bmatrix} = \begin{bmatrix} \mathbf{N}_{11} & \mathbf{N}_{12} \\ \mathbf{N}_{21} & \mathbf{N}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{S}_{11} \mathbf{A}_m & \mathbf{S}_{11} \mathbf{B}_m \\ \mathbf{C}_m & \mathbf{0} \end{bmatrix} \quad (2.131)$$

which provides a set of solvable equations for the ideal gains.

$$\mathbf{N}_{11}\mathbf{S}_{11} - \mathbf{S}_{11}\mathbf{A}_m^{-1} = -\mathbf{N}_{12}\mathbf{C}_m\mathbf{A}_m^{-1} \quad (2.132)$$

$$\mathbf{S}_{12} = \mathbf{N}_{11}\mathbf{S}_{11}\mathbf{B}_m \quad (2.133)$$

$$\mathbf{K}_x^* = \mathbf{N}_{21}\mathbf{S}_{11}\mathbf{A}_m + \mathbf{N}_{22}\mathbf{C}_m \quad (2.134)$$

$$\mathbf{K}_u^* = \mathbf{N}_{21}\mathbf{S}_{11}\mathbf{B}_m \quad (2.135)$$

■

REMARK Equation (2.115) is a Sylvester equation, while the others are linear combinations of the known conditions or the previous solutions.

Sylvester equations are iteratively solved using the Bartels-Stewart algorithm [62], which is available in most mathematical computation software. The remaining equations are solved analytically using the value for \mathbf{S}_{11} . The system of equations will not have a solution when either the plant is not ASPR, in which case the inversion of the matching matrix \mathbf{M} may fail, or if the eigenvalue of \mathbf{N}_{11} are not distinct from the eigenvalues of \mathbf{A}_m^{-1} , in which case no solution exists for the Sylvester equation.

Knowing not only that a set of ideal gains exists for any given plant, but also what those values are is an incredibly useful tool in SAC design.

2.5.4 Disturbance Accommodating SAC

Like any controller, SAC is able to compensate for disturbances present in systems using feedback control. The SAC adaptation equations are useful for determining the magnitude of gains for minimizing the tracking error, so it is natural to ask if SAC adaptation might be useful for determining the gains for other system components as well. Following this same train of thought, Prabhakar et al. [29] leverages SAC to create a disturbance accommodating controller for the management of linear disturbances applied to a SAC.

Although any standard disturbance estimation or compensation scheme, such as nonlinear dynamic inversion, can be used in parallel with SAC to combat disturbances, none makes use of the adaptation mechanisms present in SAC directly. The hope is

to use the SAC adaptation mechanism to identify the magnitude of a disturbance using a linear model.

Assume that the disturbance can be modelled using a linear disturbance generator of the form

$$\dot{\mathbf{z}}_d(t) = \mathbf{F}_d \mathbf{z}_d(t) \quad (2.136)$$

$$\mathbf{u}_d(t) = \mathbf{\Theta}_d \mathbf{z}_d(t) \quad (2.137)$$

for disturbance model order $v \in \mathbb{Z}^+$, disturbance model states $\mathbf{z}_d \in \mathbb{R}^v$, state update matrix $\mathbf{F}_d \in \mathbb{R}^{v \times v}$, output transition $\mathbf{\Theta}_d \in \mathbb{R}^{m \times v}$, and output disturbance $\mathbf{u}_d \in \mathbb{R}^m$. The similar estimated linear disturbance generator is given by

$$\dot{\hat{\mathbf{z}}}_d(t) = \hat{\mathbf{F}}_d \hat{\mathbf{z}}_d(t) \quad (2.138)$$

$$\hat{\mathbf{u}}_d(t) = \hat{\mathbf{\Theta}}_d \hat{\mathbf{z}}_d(t) \quad (2.139)$$

with the “ $\hat{\cdot}$ ” symbol denoting the estimate of each of the values of \mathbf{F}_d , \mathbf{z}_d , \mathbf{u}_d , and $\mathbf{\Theta}_d$ with identical sizes. The adaptation for the disturbance accommodating gain follows from the previous adaptive gain formulations as

$$\mathbf{u}_z(t) = \mathbf{K}_z(t) \hat{\mathbf{z}}_d(t) = \mathbf{K}_{zP}(t) + \mathbf{K}_{zI}(t) \quad (2.140)$$

$$\mathbf{K}_{zP}(t) = \mathbf{e}_y(t) \hat{\mathbf{z}}_d^T(t) \mathbf{\Gamma}_{zP} \quad (2.141)$$

$$\dot{\mathbf{K}}_{zI}(t) = \mathbf{e}_y(t) \hat{\mathbf{z}}_d^T(t) \mathbf{\Gamma}_{zI} - \mathbf{K}_{zI}(t) \boldsymbol{\sigma}_z \quad (2.142)$$

with $\mathbf{u}_z \in \mathbb{R}^m$ being the output SAC control due to disturbance compensation, and $\mathbf{K}_z \in \mathbb{R}^{m \times v}$ the matrix of disturbance state gains. The addition of disturbance accommodation increases the size of the reference signal, making the new full SAC equations

$$\mathbf{r}(t) = \begin{bmatrix} \mathbf{e}_y(t) \\ \mathbf{x}_m(t) \\ \mathbf{u}_m(t) \\ \hat{\mathbf{z}}_d(t) \end{bmatrix} \quad (2.143)$$

$$\mathbf{u}_p(t) = \mathbf{K}(t)\mathbf{r}(t) \quad (2.144)$$

$$\mathbf{K}(t) = \mathbf{K}_P(t) + \mathbf{K}_I(t) \quad (2.145)$$

$$\mathbf{\Gamma}_P = \begin{bmatrix} \mathbf{\Gamma}_{eP} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Gamma}_{xP} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{\Gamma}_{uP} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Gamma}_{zP} \end{bmatrix} \quad (2.146)$$

$$\mathbf{\Gamma}_I = \begin{bmatrix} \mathbf{\Gamma}_{eI} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Gamma}_{xI} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{\Gamma}_{uI} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Gamma}_{zI} \end{bmatrix} \quad (2.147)$$

$$\boldsymbol{\sigma} = \begin{bmatrix} \boldsymbol{\sigma}_e & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\sigma}_x & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\sigma}_u & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\sigma}_z \end{bmatrix} \quad (2.148)$$

$$\mathbf{K}_P(t) = \mathbf{e}_y(t)\mathbf{r}^T(t)\mathbf{\Gamma}_P \quad (2.149)$$

$$\dot{\mathbf{K}}_I(t) = \mathbf{e}_y(t)\mathbf{r}^T(t)\mathbf{\Gamma}_I - \mathbf{K}_I\boldsymbol{\sigma} \quad (2.150)$$

The UAV simulation results presented in Prabhakar [29] are promising for managing disturbances of known frequency and unknown amplitude.

2.5.5 System Augmentation: SAC for Non-ASPR Systems

Very few systems are ASPR, and previous sections have directly discussed methods of SAC parameter selection that maintain the ASPR assumptions. Necessary for the application of these techniques, however, is the control of an ASPR or *almost* ASPR system. Most systems are not ASPR, and as such SAC cannot be directly applied to these systems. For example, the simple double integrator system

$$F = ma \tag{2.151}$$

fails at the go to meet the ASPR requirements, since it has a double pole at the origin. Nonetheless, controlling the position of a system is a useful control task that would benefit from the application of adaptation. What has been done to apply direct adaptive control to non-ASPR systems is called *augmentation*, where an augmented plant that is ASPR and similar to the real plant is controlled by the adaptive system.

Due to the nature of the control systems field, techniques to improve and modify controllers are constantly being added, changed, or rejected from widespread appeal. Despite having determined a method to create a discrete ASPR plant in 1986 [59], Barkana is still developing methods to apply SAC to non-ASPR systems while preserving performance, having released a paper on discrete-time SAC implementation in non-ASPR systems as recently as 2014 [63]. Due to the rapid and constant change in the field while chasing the “best” controller formulations, two methods of augmenting a plant have been developed, with no clear distinction between them as to which is the superior technique. The methods of augmenting a non-ASPR plant to comply with SAC requirements are discussed and illustrated briefly here, however no method can be broadly classified as the “correct” method of augmenting a plant.

When designers are creating a SAC implementation, they should choose one of the below methods to augment their system. Without augmentation, application of SAC to control a non-ASPR system will result in odd choices of adaptation parameters being stable, necessarily large proportional adaptations to maintain stability, and overall poor performance.

Feedforward Parallelization

Barkana [27] presented the first method for allowing a non-ASPR plant to be augmented to resemble a similar ASPR plant, called *feedforward parallelization*.

The general case is that some non-ASPR system $\mathbf{G}(s)$ is desired to be controlled by a SAC. In order to control the system, it must be converted into an “augmented plant” $\mathbf{G}_a(s)$ which is ASPR.

Theorem 2.5.3 [58]

Consider the system $\mathbf{G}(s)$ not necessarily stable or minimum phase. Let $\mathbf{H}(s)$ be a proper output feedback transfer system with number of zeros $z \in \mathbb{Z}^+$ and number of poles $p \in \mathbb{Z}^+$ such that $p \leq n$. If the closed loop transfer function

$$\mathbf{G}_c(s) = [\mathbf{I} + \mathbf{G}(s)\mathbf{H}(s)]^{-1}\mathbf{G}(s) \quad (2.152)$$

is asymptotically stable, then the augmented open loop transfer function

$$\mathbf{G}_a(s) = \mathbf{G}(s) + \mathbf{H}^{-1}(s) \quad (2.153)$$

has $z + p$ poles and $z + p$ zeros and is therefore ASPR.

REMARK Barkana also proves this is the case for discrete systems in [59]. Intuition behind why any stabilizing controller $\mathbf{H}(s)$ must also create an ASPR augmented system when placed as a parallel feedforward component of the plant is developed in Appendix C.

The use of a stabilizing controller to render any plant ASPR proves a powerful tool for ensuring that SAC can be applied to any system while maintaining stability of that system. All that a designer need do to create an ASPR system is to find a stabilizing controller of that system. Furthermore, due to the inverse of the stabilizing controller being taken, any stable high-gain controller for a system will suffice to render the system ASPR.

An example of a system with feedforward parallelization is shown in Fig. 2.15. The stabilizing controller $\mathbf{H}(s)$ is placed in parallel with the plant to create the augmented system output \mathbf{y}_a

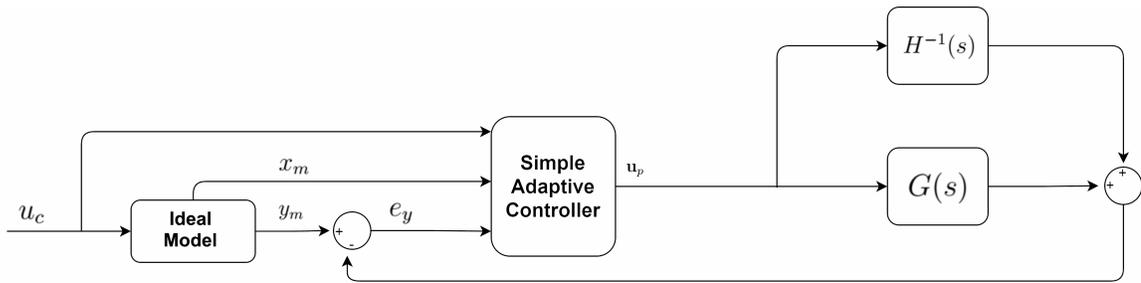


Figure 2.15: SAC with Feedforward Parallelization

$$\mathbf{y}_a(s) = (\mathbf{G}(s) + \mathbf{H}^{-1}(s))\mathbf{u}_p(s) \quad (2.154)$$

Signal Fusion

Another method of rendering a plant ASPR is to take some combination of the measured signals to produce an augmented plant which is ASPR. Sensor blending for the purposes of augmenting a non-ASPR plant was explored in the context of aircraft by Balas and Frost [64].

By taking the original signal

$$\mathbf{y}_p(t) = \mathbf{C}_p \mathbf{x}_p(t) \quad (2.155)$$

and taking the new output

$$\mathbf{y}_a(t) = \mathbf{C}_\Delta \mathbf{x}_p(t) = (\mathbf{C} + \Delta \mathbf{C}) \mathbf{x}_p(t) \quad (2.156)$$

for an appropriate selection of $\Delta \mathbf{C}$, Balas and Frost were able to show that the augmented plant could be used with direct adaptive control by proving that finite unstable transmission zeros could be converted to stable transmission zeros. Furthermore they provide a systematic method to do so.

Augmentation of Feedback

A rarely-seen modification to SAC can be made that mitigates the negative aspects of feedforward parallelization. Instead of taking the full SAC output \mathbf{u}_p as the input to

the stabilizing controller in feedforward parallelization $\mathbf{H}^{-1}(s)$, if the ASPR property need only be preserved for feedback control of the system, then perhaps only the feedback control $\mathbf{K}_e(t)\mathbf{e}_y(t)$ of the system needs to be passed to the augmentations. The augmented output is now

$$\mathbf{y}_a(s) = \mathbf{G}(s)\mathbf{u}_p(s) + \mathbf{H}^{-1}(s)(\mathbf{K}_e(s)\mathbf{e}_y(s)) \quad (2.157)$$

The first written record of this method occurs in Shibata et al. [65], where a stability proof of the method is included. The stability proof of the method is compared to the stability proof for other discrete ASPR SAC implementations in Appendix C.

It is unclear how prevalent this method has become, since the original paper has not been cited in other works, however the technique resurfaces in Takagi et al. [66] without accreditation. Since the modification does not have a name, it is difficult to track.

2.5.6 SAC Drawbacks

Simple adaptive control in particular suffers from some drawbacks that are not present for linear controllers.

The biggest benefit of using direct adaptive control in any application is the minimal amount of *a priori* knowledge required to design a useful controller. If the plant is ASPR, then just the baseline SAC formulation is sufficient to ensure stability. The lack of adaptation rates $\mathbf{\Gamma}_I$ in the continuous time SAC stability proof suggests that the choice of adaptation rates does not affect the stability of the final controller, so long as the matrix of adaptation rates is positive definite. Nevertheless, it is trivial to show that a system can be rendered unstable by sufficiently large choices of adaptation rates.

There are additional control effects that are present when SAC is being employed. The most prominent of the observed phenomena is an effect known as *bursting*. Bursting is a phenomena where the command forces, tracking error, or adaptive gains suddenly increase while oscillating, before reaching a limit and returning to the original behaviour. Bursting may occur in small or large scales, reach a bounded limit, or go unstable. Bursting also only occurs occasionally for some SAC designs, and may not

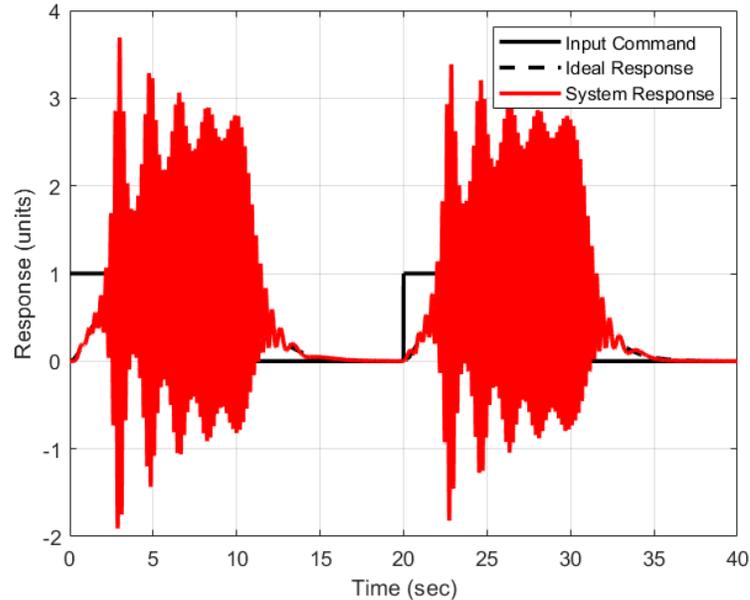


Figure 2.16: Exaggerated example of Bursting Phenomena

include any significant change in the command, state, or error signal that would be an obvious cause of the phenomena. Bursting occurs more frequently for higher values of the adaptation parameters. Until the bursting phenomena can be adequately explained many designers will feel uncomfortable with adaptive control, as they will be unsure if bursting might occur in their system at an inopportune time and cause damage. An exaggerated example of what significant SAC bursting looks like in a simulation is shown in Fig. 2.16.

The purpose of this thesis is to mitigate the drawbacks and increase the benefits of SAC in physical implementations of adaptive controllers. The problem of choosing stable adaptation parameters, mitigating bursting, and improving the response of adaptive controllers will be tackled in Chapter 3. For now, the most apparent way to improve SAC performance is to determine how optimization can be leveraged.

2.6 Optimization and Metaheuristics

Simple adaptive control offers the possibility of creating a control system that is not only robust to disturbances and uncertainties, but also guarantees performance

and stability. In practice, a surprisingly large amount of the SAC design relies on the ability of the designer to choose parameters that provide “good” control of the system. The obvious next step, then, would be to determine if the task of adaptive controller design can be simplified through the use of optimality search techniques.

The field of mathematical optimization is composed of many problems that can all be linked to one fundamental question: given an objective function $f : \mathcal{V} \rightarrow \mathbb{R}$ that is dependant on values $x \in \mathcal{V}$ following

$$O = f(x) \tag{2.158}$$

what is the minimum value of O and the parameters of x that achieve that minimum? Conversely, what parameters x achieve a maximum value of O ?

If certain forms of the objective function and its constraints are present then different techniques can be used to solve the problem. The various branches of optimization theory pertain to the forms, and therefore the assumptions that can be made about the objective O , and the constraints placed on the parameter space.

Convex optimization, nonlinear optimization, and metaheuristic optimization are three sets of techniques that have been used in the presented SAC research to simplify the task of designing controllers. The field of mathematical optimization is broad and complex, however. The best that can be accomplished here is a reflection of the main results of mathematical optimization theory as they apply to adaptive controller design. One of the great features of optimization techniques is that the algorithms and methods used to solve optimization problems can be used without intimate knowledge of their operations. Most useful optimization techniques now come bundled in mathematical software, such as MATLAB[®], to allow for solutions to any problem that can be stated as an optimization problem. It is not necessary to understand the fundamentals of optimization theory to query `fmincon(·)`, however a review of the fundamental concepts will be useful in understanding the results and limitations of each technique.

Resources for further reading are provided for both convex and nonlinear optimization, as the bulk of mathematical optimization theory is beyond the scope of this work.

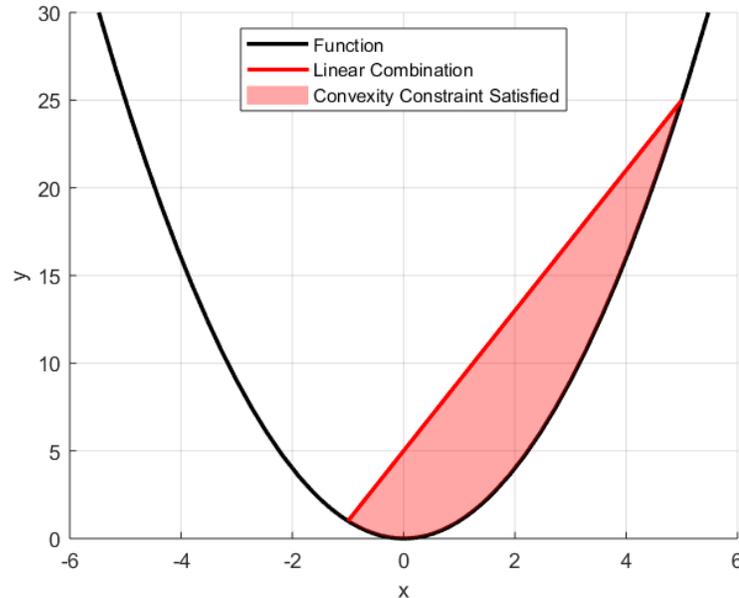


Figure 2.17: Example Convex Function, With Convexity Constraint Highlighted

2.6.1 Convex Optimization

The most well developed of the optimization fields is the branch of convex optimization. Convexity is a property of a function that is highly beneficial to the solution of optimization problems.

DEFINITION 5 (CONVEX FUNCTION [31]) A function is convex if it is true that for any parameters $x_c, y_c \in \mathcal{V}$ and mixing parameter $0 \leq \lambda_c \leq 1$ that the function $f : \mathcal{V} \rightarrow \mathbb{R}$ satisfies

$$f(x_c \lambda_c + y_c(1 - \lambda_c)) \leq \lambda_c f(x_c) + (1 - \lambda_c) f(y_c) \quad (2.159)$$

REMARK In effect, given two test points any intermediary test point is guaranteed to be equal to, or of lesser value than, the linear combination of the test points. An example of the convexity condition can be seen in Fig. 2.17.

A set is said to be convex if it follows a similar property, namely that the linear combination of any two elements of the set are always a member of the set.

The convexity condition has been leveraged to quickly and accurately determine the optimum value of a function. Moreover, if an optimization problem is provably

convex, it can be guaranteed that only a single optimum point exists and the input that corresponds to that optimum can be found using computationally efficient methods [31]. Constraints, such as maximum and minimum parameter values, equality conditions, and any other design factors in place while solving an optimization problem are included in optimality searches and limit the parameter space. For convex optimization problems the constraints placed on such problems must also be convex.

DEFINITION 6 (CONVEX OPTIMIZATION PROBLEM [31]) A convex optimization problem is any problem that can be defined as

$$\underset{x \in \mathcal{V}}{\text{minimize}} \quad O = f(x) \quad (2.160)$$

$$\text{subject to} \quad g_i(x) \leq 0, \quad i = 1, \dots, m_c \quad (2.161)$$

$$h_i(x) = 0, \quad i = 1, \dots, p_c \quad (2.162)$$

for the objective $O \in \mathbb{R}$, objective function $f : \mathcal{V} \rightarrow \mathbb{R}$ which is convex in \mathcal{V} , and where each of $g_i : \mathcal{V} \rightarrow \mathbb{R}$ are convex, and each of $h_i : \mathcal{V} \rightarrow \mathbb{R}$ are affine for $m_c \in \mathbb{Z}$ and $p_c \in \mathbb{Z}$ maximum constraints.

The goal of convex optimization as described in this paper is to create a single homogenous function that can be stepped through using Newton's method to determine the solution. Along the way, it will be shown that inequality constraints are equivalent to equality constraints with additional *slack variables* that can be varied, that a second-order cone problem can be restated as a dual problem, and that the cone constraints can be restated as a homogenous equality constraint between the primal and dual problem. The original second order cone problem is then restated as an equivalent problem that is self-dual and homogeneous, and whose central path can be determined and followed using a Newton's method search to find the solution that minimizes the objective function while meeting the constraints. The final equation, Eq. (2.188), embodies all of these properties, where the convex problem has been restated as a homogenous problem, and a central path can be followed to end up at the problem's solution.

A significant amount of the theory and theorems behind these results must necessarily be overlooked in constructing a rudimentary overview of convex optimization theory. The standout textbook in the field is Boyd's *Convex Optimization* [31] which covers in detail and clarity the method to solve convex second-order cone problems that is only summarized in this section. The results of convex optimization theory are accessible in all pieces of major mathematical software, and will be leveraged in future chapters to simplify solutions to the uncontrolled spacecraft problem.

Inequality Constraints

An inequality constraint can be formulated as an equality constraint with an additional parameter that does not affect the objective, called a *slack variable*. For the inequality constraint

$$\mathbf{F}_c \mathbf{x}_c \leq \mathbf{b}_c \quad (2.163)$$

the introduction of the slack variable \mathbf{y}_s allows for the inequality constraint to be converted to the equality constraint

$$\mathbf{F}_c \mathbf{x}_c + \mathbf{y}_s = \mathbf{b}_c \quad (2.164)$$

where the value of the slack variable \mathbf{y}_s given the parameter \mathbf{x}_c indicates that a candidate solution is infeasible if negative, and feasible if positive. The problem definition given in Eq. (2.160) now allows any inequality constraints to be converted to equality constraints with an additional parameter whose value tells us information about the state of the solution.

Additionally, if the slack variables are included as part of the parameter column matrix \mathbf{x}_c , then the equality constraint simplifies to

$$\mathbf{F}_c \mathbf{x}_c = \mathbf{b}_c \quad (2.165)$$

A convex optimization problem can now be placed into the form

$$\begin{array}{ll} \text{minimize} & \mathbf{f}_c^T \mathbf{x} \\ \mathbf{x} \in \mathcal{V} & \end{array} \quad (2.166)$$

$$\text{subject to} \quad \mathbf{F}_c \mathbf{x}_c = \mathbf{b}_c, \quad (2.167)$$

$$\mathbf{x} \in \mathcal{C} \quad (2.168)$$

where n_c is the order of the optimization space including slack variables, $m_c + p_c$ is the order of the constraints, $\mathbf{x}_c \in \mathbb{R}^{n_c}$ is the parameter vector including slack variables, $\mathbf{f}_c \in \mathbb{R}^{n_c}$ is a linear objective, $\mathbf{F}_c \in \mathbb{R}^{(m_c+p_c) \times n_c}$ is the linear constraint vector, $\mathbf{b}_c \in \mathbb{R}^{(m_c+p_c)}$ is the column matrix of equality constraints, and \mathcal{C} is the set of convex cone constraints for the system. Like the inequality constraints, the second-order cone constraints can also be converted to a set of linear constraints, but must be done by making use of the dual problem.

Self-Dual and Homogeneous Problem

Convex optimization techniques also make use of a clever problem redefinition to determine the solution to a problem faster and more accurately. The objective and constraint of a problem represent two distinct parts of an optimization problem: the objective is the value to minimize, while the constraints are the relationships to maintain. It just so happens that convex problems also have what is called a *dual problem*. The dual of a problem has an objective function that is similar to the constraints of the *primal* problem, and constraints that are similar to the objective of the primal problem.

Theorem 2.6.1 [67] *The dual of a second-order cone problem is given by*

$$\begin{array}{ll} \text{maximize} & \mathbf{b}_c^T \mathbf{y}_c \\ \mathbf{y}_c \in \mathcal{V}^* & \end{array} \quad (2.169)$$

$$\text{subject to} \quad \mathbf{F}_c^T \mathbf{y}_c + \mathbf{s} = \mathbf{f}_c, \quad (2.170)$$

$$\mathbf{s} \in \mathcal{C}^* \quad (2.171)$$

where \mathcal{V}^* is the equivalent convex space of the dual parameter, and \mathcal{C}^* is the set of cones for the dual problem such that

$$\mathcal{C}^* = \{\mathbf{s} : \mathbf{s}^T \mathbf{x} \geq 0 \ \forall \mathbf{x} \in \mathcal{C}\} \quad (2.172)$$

REMARK The dual of a problem includes the constraints of the primal problem as its objective, and the objective of the primal problem as its constraints. By reusing computations of the objective and constraints between the primal and dual problems, computation is drastically reduced. Furthermore, if a dual does not exist for the problem, then it is infeasible and cannot be solved.

What is incredible about second-order cone problems, is that they can be formulated as *self-dual problems*, meaning that for every second-order conic optimization problem, there also exists a related problem that is its own dual. Calculating the constraints of the problem acts as a computation of the system objective, and vice-versa. Computation is drastically reduced for a self-dual problem. Anderson [67] provides the method to convert a second-order conic optimization problem to a homogeneous and self-dual problem through the addition of two variables $\boldsymbol{\tau} \in \mathbb{R}^+$ and $\boldsymbol{\kappa} \in \mathbb{R}^+$ that now also determine if a problem is unbounded or infeasible.

Theorem 2.6.2 [67]

A primal second-order cone problem and its dual can be formulated as a Goldman-Tucker homogeneous and self-dual system

$$\mathbf{F}_c \mathbf{x}_c - \mathbf{b}_c \boldsymbol{\tau} = 0 \quad (2.173)$$

$$\mathbf{F}_c^T \mathbf{y}_c + \mathbf{s} - \mathbf{f}_c \boldsymbol{\tau} = 0 \quad (2.174)$$

$$-\mathbf{f}_c^T \mathbf{x}_c + \mathbf{b}_c^T \mathbf{y}_c - \boldsymbol{\kappa} = 0 \quad (2.175)$$

with additional parameters $\boldsymbol{\tau} \in \mathbb{R}^+$ and $\boldsymbol{\kappa} \in \mathbb{R}^+$. Solutions to the system must satisfy

$$\boldsymbol{\tau} + \boldsymbol{\kappa} > 0 \quad (2.176)$$

REMARK The homogeneous and self-dual system have properties that are conducive to iterative searches of the design space. However, the second-order cone constraints are not yet included in the system.

Second-Order Cone Constraints

A second-order cone is a cone that begins at some point and extends out in a direction, increasing in radius linearly with distance. The second-order cone constraint is also sometimes referred to as the ‘‘Lorentz cone’’ or the ‘‘Ice-cream cone’’ [67] due to its shape. An optimization problem can include second-order cone constraints, that is to say some number of parameters must fall inside of a designated cone, without losing convexity.

Theorem 2.6.3 [67] *A second order cone constraint can always be defined through*

$$\|\mathbf{A}_{sc}(i)\mathbf{x}_c - \mathbf{b}_{sc}(i)\| \leq \mathbf{d}_c^T(i)\mathbf{x} - g(i) \quad (2.177)$$

for $i \in \mathbb{Z}^+$ second order cone constraints, $q(i) \in \mathbb{Z} \forall i$ dimension of each cone-constraint, constraint matrices $\mathbf{A}_{sc}(i) \in \mathbb{R}^{q(i) \times n}$, $\mathbf{b}_{sc}(i) \in \mathbb{R}^{q(i)}$, and $\mathbf{d}^T(i) \in \mathbb{R}^{1 \times n}$ alongside constant $g(i) \in \mathbb{R}$.

Then, for the arrowhead matrices of \mathbf{X} in x_1 and \mathbf{S} in s_1 , defined for the $\mathbf{x} = [x_1, x_2, \dots, x_n]^T$ and $\mathbf{s} = [s_1, s_2, \dots, s_n]^T$ as

$$\mathbf{X} = \begin{bmatrix} x_1 & x_{2:n}^T \\ x_{2:n} & x_1 \mathbf{I}_{(n-1)} \end{bmatrix} \quad (2.178)$$

$$\mathbf{S} = \begin{bmatrix} s_1 & s_{2:n}^T \\ s_{2:n} & s_1 \mathbf{I}_{(n-1)} \end{bmatrix} \quad (2.179)$$

$$(2.180)$$

any second order cone constraint can be converted to an equivalent constraint

$$\mathbf{X}\mathbf{S}\mathbf{e}_c = 0 \quad (2.181)$$

where $\mathbf{e}_c \in \mathbb{R}^n$ is the unit vector corresponding to the cone dimension of the center of the cone constraint. If multiple cone constraints are present, blocks of arrowhead matrices \mathbf{X} and \mathbf{S} and cone unit vectors \mathbf{e}_c can be included as block diagonals to the constraint in Eq. (2.181), and additional unit vectors vertically concatenated for \mathbf{e}_c .

REMARK The equality constraint created in Eq. (2.181) is only met if the cone constraints are met. The equality constraint can also be repeated for as many cones as are present in the system, and can be used alongside the previous system to search for the solution.

Any second order cone constraints can now be formulated as an equality constraint for a homogeneous and self-dual problem. With that, any second-order cone problem can be redefined as a homogeneous and self-dual problem.

The full set of equations for the problem is now

$$\mathbf{F}_c \mathbf{x}_c - \mathbf{b}_c \boldsymbol{\tau} = \mathbf{0} \quad (2.182)$$

$$\mathbf{F}_c^T \mathbf{y}_c + \mathbf{s} - \mathbf{f}_c \boldsymbol{\tau} = \mathbf{0} \quad (2.183)$$

$$-\mathbf{f}_c^T \mathbf{x}_c + \mathbf{b}_c^T \mathbf{y}_c - \boldsymbol{\kappa} = \mathbf{0} \quad (2.184)$$

$$\mathbf{X} \mathbf{S} \mathbf{e}_c = \mathbf{0} \quad (2.185)$$

$$\boldsymbol{\tau} \boldsymbol{\kappa} = \mathbf{0} \quad (2.186)$$

Central Path Method: The Solution to Homogeneous and Self-Dual Problems

What is interesting about homogeneous and self-dual systems is that their solution can be synthesized from the system constraints. There is a method to create a *central path* for the problem, and all that is needed to find the system solution is to follow the central path to the optimal solution.

The initial variables, denoted by the subscript “0”, are constructed using the structure of the problem. The parameters \mathbf{x}_0 are set to 1 for each nonnegative variable, 1 for the first variable in each cone, and 0 otherwise. Each value of \mathbf{y}_0 is set to 0, while each value of $\boldsymbol{\tau}_0$ and $\boldsymbol{\kappa}_0$ is set to 1. Each value of \mathbf{s}_0 is set to 1 if it is a nonnegative

variable, otherwise the first variable of each cone is set to 1 while the rest are set to 0.

Then, where $j \in \mathbb{Z}$ is the number of nonzero elements in \mathbf{x}_0 , $\boldsymbol{\mu}_0$ is created following

$$\boldsymbol{\mu}_0 = \frac{\mathbf{x}_0^T \mathbf{s}_0 + \boldsymbol{\tau}_0 \boldsymbol{\kappa}_0}{j + 1} \quad (2.187)$$

The central path for the solution can be constructed from these initial conditions. The central path is the set of conditions for $0 \leq \gamma \leq 1$ that leads to a feasible and optimal solution at $\gamma = 0$. The central path of the homogeneous and self-dual problem is found to be [67]

$$\mathbf{F}_c \mathbf{x}_c - \mathbf{b}_c \boldsymbol{\tau} = \gamma(\mathbf{F}_c \mathbf{x}_0 - \mathbf{b}_c \boldsymbol{\tau}_0) \quad (2.188)$$

$$\mathbf{F}_c^T \mathbf{y}_c + \mathbf{s} - \mathbf{f}_c \boldsymbol{\tau} = \gamma(\mathbf{F}_c^T \mathbf{y}_0 + \mathbf{s}_0 - \mathbf{f}_c \boldsymbol{\tau}_0) \quad (2.189)$$

$$-\mathbf{f}_c^T \mathbf{x}_c + \mathbf{b}_c^T \mathbf{y}_c - \boldsymbol{\kappa} = \gamma(-\mathbf{f}_c^T \mathbf{x}_0 + \mathbf{b}_c^T \mathbf{y}_0 - \boldsymbol{\kappa}_0) \quad (2.190)$$

$$\mathbf{X} \mathbf{S} \mathbf{e}_c = \gamma \boldsymbol{\mu}_0 \mathbf{e}_c \quad (2.191)$$

$$\boldsymbol{\tau} \boldsymbol{\kappa} = \gamma \boldsymbol{\mu}_0 \quad (2.192)$$

Theorem 2.6.4 [67] *Given the central path for the homogeneous and self-dual system in Eqs. (2.188) through (2.192), the Monteiro-Zhang search direction for the system is*

$$\mathbf{F}_c \mathbf{d}_x - \mathbf{b}_c \mathbf{d}_\tau = (\gamma - 1)(\mathbf{F}_c \mathbf{x}_0 - \mathbf{b}_c \boldsymbol{\tau}_0) \quad (2.193)$$

$$\mathbf{F}_c^T \mathbf{d}_y + \mathbf{d}_s - \mathbf{f}_c \mathbf{d}_\tau = (\gamma - 1)(\mathbf{F}_c^T \mathbf{y}_0 + \mathbf{s}_0 - \mathbf{f}_c \boldsymbol{\tau}_0) \quad (2.194)$$

$$-\mathbf{f}_c^T \mathbf{d}_x + \mathbf{b}_c^T \mathbf{d}_y - \mathbf{d}_\kappa = (\gamma - 1)(-\mathbf{f}_c^T \mathbf{x}_0 + \mathbf{b}_c^T \mathbf{y}_0 - \boldsymbol{\kappa}_0) \quad (2.195)$$

$$\mathbf{X}_0 \mathbf{d}_s + \mathbf{S}_0 \mathbf{d}_x = -\mathbf{X}_0 \mathbf{S}_0 \mathbf{e}_c + \gamma \boldsymbol{\mu}_0 \mathbf{e}_c \quad (2.196)$$

$$\boldsymbol{\tau}_0 \mathbf{d}_\kappa = \boldsymbol{\tau}_0 \boldsymbol{\kappa}_0 + \gamma \boldsymbol{\mu}_0 \quad (2.197)$$

which is a linear set of equations that can be solved for the step sizes with respect to the input variables $(\mathbf{d}_x, \mathbf{d}_\tau, \mathbf{d}_y, \mathbf{d}_s, \mathbf{d}_\kappa)$. For step sizes of $\alpha \in (0, 1]$ the update equation

for solutions of the system along the central path are found using

$$\begin{bmatrix} \mathbf{x}(k+1) \\ \boldsymbol{\tau}(k+1) \\ \mathbf{y}(k+1) \\ \mathbf{s}(k+1) \\ \boldsymbol{\kappa}(k+1) \end{bmatrix} = \begin{bmatrix} \mathbf{x}(k) \\ \boldsymbol{\tau}(k) \\ \mathbf{y}(k) \\ \mathbf{s}(k) \\ \boldsymbol{\kappa}(k) \end{bmatrix} + \alpha \begin{bmatrix} \mathbf{d}_x(k) \\ \mathbf{d}_\tau(k) \\ \mathbf{d}_y(k) \\ \mathbf{d}_s(k) \\ \mathbf{d}_\kappa(k) \end{bmatrix} \quad (2.198)$$

and the initial conditions

$$\begin{bmatrix} \mathbf{x}(1) \\ \boldsymbol{\tau}(1) \\ \mathbf{y}(1) \\ \mathbf{s}(1) \\ \boldsymbol{\kappa}(1) \end{bmatrix} = \begin{bmatrix} \mathbf{x}_0 \\ \boldsymbol{\tau}_0 \\ \mathbf{y}_0 \\ \mathbf{s}_0 \\ \boldsymbol{\kappa}_0 \end{bmatrix} \quad (2.199)$$

REMARK By employing Newton's method to the homogeneous and self-dual system, a set of linear equations can be solved to find the step size of each variable. The system is provably convergent on the solution, which must satisfy the equality constraints of the self-dual system, the second-order cone constraints of the primal system, the feasibility and boundedness constraints, while also being the optimal solution.

What the theorem above provides, which is truly remarkable, is a repeatable set of steps that take an optimization problem and determine the solution to that problem. While there are a lot of individual steps involved, the ability to find the solution to any set of inequality constraints or second-order cone constraints cannot be understated. The process may seem long and convoluted, but the individual steps are repeatable and powerful.

The formulation of a general and computationally efficient method of solving second-order cone problems has allowed a wide variety of solutions to be found for linear cost functions. The above method for solving convex optimization problems, provided by Andersen and Roos [67], is also the method employed by MATLAB®'s

internal convex solver⁴. Convex optimization techniques are used in this text to determine optimal trajectories of the chaser spacecraft that result in rendezvous with the target spacecraft.

Finding the optimal trajectory for a spacecraft through convex optimization is done in the following section to serve as both an example and clarification of how convex optimization is implemented.

2.6.2 Example Convex Optimization through Successive Approximation

Although few systems can be formulated to be convex, there are several motivating examples that provide ample incentive to employ convex optimization where possible. Convex optimization can be used, for example, to minimize fuel cost during trajectory planning of spacecraft rendezvous and docking manoeuvres.

The nonlinear formation flying Eqs.(2.13) through (2.17) make it difficult to determine a trajectory between two relative points in orbit while also minimizing fuel consumption. Optimization techniques can be used to drastically reduce the difficulty of finding such trajectories.

The formation flying equations do not initially satisfy the convexity condition, however Lu and Liu [68] use a method of successive approximation of the nonlinear terms to quickly and accurately determine an optimal docking trajectory. The authors' technique is used to construct the trajectory command used for rendezvous and docking, and is briefly summarized here.

The system dynamics for the chaser spacecraft are formulated in the optimization problem as the dimensionless equations

$$\dot{\mathbf{r}}_{c,n} = \mathbf{V}_{c,n} \quad (2.200)$$

$$\dot{\mathbf{V}}_{c,n} = -\frac{1}{\|\mathbf{r}_c\|^3} \mathbf{r}_c + \boldsymbol{\tau}_c \quad (2.201)$$

$$\dot{z}_c = \frac{-1}{v_{ex}} \sigma_c \quad (2.202)$$

where $\boldsymbol{\tau}_c = \mathbf{T}_c/m_c \in \mathbb{R}^3$ is the nondimensionalized thrust of the chaser, $\mathbf{r}_{c,n} =$

⁴<https://www.mathworks.com/help/optim/ug/cone-programming-algorithm.html>

$\mathbf{r}_c/R_\oplus \in \mathbb{R}^3$ is the position of the chaser spacecraft around the parent body, nondimensionalized by the parent body radius $R_\oplus \in \mathbb{R}$, in an inertial reference frame, while $z_c = \ln(m_c) \in \mathbb{R}$ is a change of variables of the spacecraft mass, and $\sigma_c = \|\mathbf{T}\|/m \in \mathbb{R}$ is a nondimensionalized thrust magnitude. The state matrix of the target is also constructed, however the target is uncontrolled, and as such does not have a control matrix. The chaser state vector \mathbf{y}_c , chaser control vector \mathbf{u}_c , and target state vector \mathbf{y}_t of the system are used to create the state and control matrices

$$\mathbf{X}_c = \begin{bmatrix} \mathbf{r}_{c,n} \\ \mathbf{V}_{c,n} \\ z_c \end{bmatrix}, \quad \mathbf{u}_c = \begin{bmatrix} \tau_c \\ \sigma_c \end{bmatrix} \quad (2.203)$$

$$\mathbf{X}_t = \begin{bmatrix} \mathbf{r}_{t,n} \\ \mathbf{V}_{t,n} \\ z_t \end{bmatrix} \quad (2.204)$$

Equations (2.200) through (2.202) are used with

$$\mathbf{X}_{tot} = \begin{bmatrix} \mathbf{X}_c \\ \mathbf{X}_t \end{bmatrix} \quad (2.205)$$

and

$$\mathbf{u} = \mathbf{u}_c \quad (2.206)$$

to construct a state-space system.

In order to preserve convexity and allow for use of standard optimization techniques, a linear approximation of the state-space system must be constructed. Call $\mathbf{A}_{ff}(\mathbf{X}_{tot}(k))$ the linear approximation of Eqs. (2.200) through (2.202) given \mathbf{X}_{tot} at iteration $k \in \mathbb{Z}$. The dynamic equations of motion can be converted to the form

$$\dot{\mathbf{X}}_{tot}(k) = \mathbf{A}_{ff}(\mathbf{X}_{tot}(k))\mathbf{X}_{tot}(k) + \mathbf{B}_{ff}\mathbf{u}(k) \quad (2.207)$$

which is linear with respect to \mathbf{X}_{tot} and \mathbf{u} . The thrust constraints to ensure the chaser does not thrust above the value T_{max} , are determined to be

$$0 \leq \sigma_c \leq T_{max} e^{-z_c(t)} \approx T_{max} e^{-z_0(t)} [1 - (z_c(t) - z_0(t))] \quad (2.208)$$

$$\|\boldsymbol{\tau}\| \leq \sigma_c \quad (2.209)$$

with Eq. (2.208) making use of the Taylor expansion of e^x to maintain linearity. The uncontrolled target thrust is constrained to zero. The objective function of the problem, dependant only on the total system thrust, is defined as

$$O = \int_0^{t_f} \sigma_c dt \quad (2.210)$$

where t_f is the final time, and Lu [68] shows that σ_c does indeed converge to $\|\boldsymbol{\tau}\|$ in the solution.

The system dynamics are encoded in the optimization problem through equality constraints of the state space update equations. For iteration k and timestep interval h [68]

$$\begin{aligned} \left(\mathbf{I}_{14} - \frac{h}{2} \mathbf{A}_{ff}(\mathbf{X}_{tot}(k)) \right) \mathbf{X}_{tot}(k) &= \left(\mathbf{I}_{14} + \frac{h}{2} \mathbf{A}_{tot}(\mathbf{X}_{tot}(k-1)) \right) \mathbf{X}_{tot}(k-1) \\ &+ \frac{h}{2} \mathbf{B}_{ff} \mathbf{u}(k) + \frac{h}{2} \mathbf{B}_{ff} \mathbf{u}(k-1) \end{aligned} \quad (2.211)$$

is an equality constraint for the state space dynamics given the state transition matrix at timestep k of $A_{ff}(\mathbf{X}_{tot}(k))$, time varying states $\mathbf{X}_{tot}(k)$ and $\mathbf{X}_{tot}(k-1)$, time varying commands $\mathbf{u}(k)$ and $\mathbf{u}(k-1)$, and identity matrix $\mathbf{I}_{14} \in \mathbb{R}^{14 \times 14}$ under Euler integration. The dynamics equality constraint is repeated for each timestep, made to match the initial conditions, as well as the final system conditions. If a certain system state is desired, such as zero velocity and zero displacement from the target at the final time, it is met using the constraint

$$\mathbf{C}_{ff}(k) \mathbf{X}_{tot}(k) = \mathbf{d}_{ff}(k) \quad (2.212)$$

where $\mathbf{C}_{ff}(k)$ is a matrix that selects elements of the states $\mathbf{X}_{tot}(k)$ which must be equal to the desired states $\mathbf{d}_{ff}(k)$. If the docking axis is given by the unit vector \mathbf{r}_a , then an approach cone constraint is added using half angle α following

$$\|\mathbf{r}_c(k) - \mathbf{r}_t(k)\| \cos(\alpha) \leq \mathbf{r}_a^T (\mathbf{r}_c(k) - \mathbf{r}_t(k)) \quad (2.213)$$

Similarly, if a thrust decontamination half-angle β must be maintained, the constraint

$$\mathbf{r}_a^T \boldsymbol{\tau}(k) \leq \sigma_c(k) \cos(\beta) \quad (2.214)$$

is applied. Together the system constraints in Eqs. (2.208) through (2.214) are consistent with a second-order conic optimization problem. As was seen in Sec. 2.6.1, the second-order conic optimization problem here can be converted to a self-dual and homogeneous system that can be quickly, efficiently, and accurately solved. This second-order cone problem for path planning can be coded into mathematical optimization software, solved, and the solution used to determine a new time-history of the radius vector \mathbf{r}_c , with successive solutions approaching the true solution.

The authors comment on the nature of successive iteration for solving such problems [68], noting that the viability of successive iterations of convex approximations can be obtuse; determining if successive iteration will converge to the true value of a given problem is akin to solving the problem using successive iteration itself. As such, the applicability of successive iteration of convex optimization to non-convex problems may be limited.

2.6.3 Nonlinear Optimization

Although some problems can be converted to similar convex problems, such as in Lu and Liu [68], other problems resist convexification. Moreover, there is currently no way to guarantee that any given non-convex problem can or cannot be converted into a similar convex one. Problems that cannot be solved using traditional convex methods are known as nonlinear optimization problems.

Similar to convex problems, nonlinear optimization problems are defined as

$$\underset{x \in \mathcal{V}}{\text{minimize}} \quad O = f(x) \quad (2.215)$$

$$\text{subject to} \quad g_i(x) \leq 0, \quad i = 1, \dots, m_n \quad (2.216)$$

$$h_i(x) = 0, \quad i = 1, \dots, p_n \quad (2.217)$$

with no limitations on the properties of $f(x)$, or the constraints g_i and h_i . Due to their generality, few methods of direct analysis exist for determining solutions to nonlinear problems. Furthermore, since nothing can be said about the structure of the solution, it is possible for nonlinear optimization problems to contain multiple optimum solutions. At each optima the constraints of the problem may be met in some region around the solution but not outside that region. The optimal point for some neighbourhood is called the local optimum for that neighbourhood. It is currently very difficult to determine if an optimum is local, or for what values of the parameter space that value is optimal. There are furthermore very few methods to determine the solution to a nonlinear optimization problem, and methods that are typically useful for other classes of optimization problem become unreliable or unstable when tasked with a nonlinear problems. Several classes of nonlinear solver exist, with each method attempting to determine solutions to similar optimization problems with good behaviour, then determining if the nonlinear optimization problem has similar behaviour.

The nonlinear programming method used in MATLAB[®] is the Sequential Quadratic Programming (SQP) method, which has quick convergence behaviour on a wide class of problems⁵, and is inspired by the method implemented by Schittkowski [69]. A wonderful overview of SQP problems, their solutions, and the many considerations that need to be made when creating a solver for SQP problems is available in the 17th chapter of Nocedal and Wright's *Numerical Optimization* [70]. Nocedal and Wright tackle all of the numerical considerations, edge cases, and best-practices of the field of nonlinear programming within their text.

⁵<https://www.mathworks.com/help/optim/ug/constrained-nonlinear-optimization-algorithms.html>

In order to solve nonlinear optimization problems, the original nonlinear problem will be approximated as a subproblem whose solution can be quickly determined. The solution to the subproblem can be used to update the search, determine a new subproblem and solve that subproblem. The process repeats until the determined solution is found to be an optimal point.

To determine if an optimal point has been found, the properties of optimal points need to be understood. The Karush-Kuhn-Tucker equations describe the form that optimal solutions to constrained nonlinear optimization problems must take.

Theorem 2.6.5 (Karush-Kuhn-Tucker Equations [31]) *For objective function $O = f(x)$ and $m_c \in \mathbb{Z}^+$ constraints*

$$g_i(x) = 0 \quad \forall i = 1, \dots, m_c \quad (2.218)$$

then for m_e constraints that do not affect the cost, and remaining constraints affecting the cost, it must be true at any optimal point x^ that*

$$\nabla f(x^*) + \sum_{i=1}^m \lambda_i \nabla g_i(x^*) = 0 \quad (2.219)$$

$$\lambda_i g_i(x^*) = 0, \quad i = 1, \dots, m_e \quad (2.220)$$

$$\lambda_i \geq 0, \quad i = m_e + 1, \dots, m_c \quad (2.221)$$

$$(2.222)$$

for the lagrangian multipliers $\lambda_i \in \mathbb{R}, \forall i = 1, \dots, m_c$ that satisfy the equation

$$\nabla_x f(x) = \nabla_x \sum_{i=1}^m \lambda_i g_i(x) \quad (2.223)$$

Furthermore, the optimal point to the constrained optimization problem occurs when the Lagrangian function

$$\mathcal{L}(x, \lambda) = f(x) - \sum_{i=1}^m \lambda_i g_i(x) \quad (2.224)$$

is at a stationary value. That is to say it must be true that

$$\nabla_{x,\lambda}\mathcal{L}(x^*, \lambda) = 0 \quad (2.225)$$

REMARK The Lagrangian multipliers represent a conversion from a constraint to the effects of the constraint on the objective function. It can be shown that unique Lagrangian multipliers exist at an optimum. Locations for which the gradient of the Lagrangian are zero represent points where the gradient of the objective is balanced by the effects of the constraints. If the objective has no gradient at the solution, then it is a local optimum, but if the objective has a gradient at the optimal point, then that gradient must be caused by the constraints of the system. The solution closest to the local optimum and meeting the constraints conditions will have a Lagrangian with zero gradient, and Lagrangian multipliers that balance the gradient of the objective.

What SQP does is attempt to meet the Karush-Kuhn-Tucker (KKT) conditions by solving Quadratic Programming (QP) subproblems. First, the nonlinear equations are evaluated at the initial point to produce the initial cost. The gradient of the objective at the initial point is numerically calculated, alongside numerical estimates of the Lagrangian, the Hessian of the Lagrangian, and various merit functions used to approximate the Karush-Kuhn-Tucker equations and update search parameters. The detailed inner workings of this technique are outside the scope of this paper, and represent only one of the myriad ways that nonlinear programming problems are tackled within the optimization community.

The Hessian is the second derivative of the Lagrangian function, and its value at iteration k is called H_k , found through numerical estimation. The Hessian and the gradient of the objective at point x_k are used together to create a quadratic programming subproblem of the form

$$\underset{d_n \in \mathbb{R}^n}{\text{minimize}} \quad \frac{1}{2}d_n^T H_k d_n + \nabla f^T(x_k)d_n \quad (2.226)$$

$$\nabla g_i^T(x_k)d_n + g_i(x_k) = 0, \quad i = 1, \dots, m_e \quad (2.227)$$

$$\nabla g_i^T(x_k)d_n + g_i(x_k) \leq 0, \quad i = (m_e + 1), \dots, m \quad (2.228)$$

using the gradients at x_k of $\nabla g_i^T(x_k)$ and $\nabla f^T(x_k)$ over minimization parameter d_n . The QP problem can then be solved using normal techniques, such as those outlined in Schittkowski [69]. The equality constraints in Eq. (2.227) can be used to determine optimal solutions using the family of optimization techniques known as *active set methods*.

DEFINITION 7 (ACTIVE SET [31]) A constraint

$$g_i(x_k) \geq 0 \tag{2.229}$$

is called *active* if

$$g_i(x_k) = 0 \tag{2.230}$$

and *inactive* if

$$g_i(x_k) > 0 \tag{2.231}$$

REMARK If a constraint is part of the active set it is likely the solution stays within the active set. Inactive constraints likely affect the objective value. Estimates of the active set reduce the complexity of searches in QP problems.

Schittkowski's method involves determining directions orthogonal to the active set to increase convergence. The set of active constraints $A_{k,set}$ found from $\nabla g_i^T(x_k)$ is horizontally concatenated for each equality constraint and each sufficiently small inequality constraint. The active set is then put through QR decomposition to determine the nullspace of $A_{k,set}$. By evaluating parameters in the direction of the nullspace of $A_{k,set}$, updated parameter estimates remains in the active set. The subspace that is within the nullspace of the active set is notated Z_k . The solution to the QP problem within the subspace Z_k is found through determining the optimum point for the quadratic problem within the null-space of the active set, which is found to be

$$Z_k^T H_k Z_k p_n = -Z_k^T \nabla f^T(x_k) \tag{2.232}$$

optimizing over the variable p_n . Now the direction of the step \hat{d}_k that stays within the nullspace of the active set is

$$\hat{d}_k = Z_k p \quad (2.233)$$

The step to be taken that minimizes the QP problem, and should minimize the nonlinear objective function, is taken to be

$$x_{k+1} = x_k + \alpha_n \hat{d}_k \quad (2.234)$$

where the selection of step size α_n can further be refined based on its improvement to a merit function of the QP problem. The new step location is then taken to be the center of a new QP problem and the process is repeated.

Several modifications and numerical stability considerations are taken when solving a SQP problem. The varied and numerous minutiae of nonlinear programming design cannot be covered here, and do not help to clarify the role of optimization for adaptive controller design. Nonetheless, nonlinear optimization techniques offer the possibility to determine powerful solutions to problems that cannot be solved using convex methods.

Nonlinear solvers have been implemented in many modern pieces of software. Of particular note is MATLAB[®]'s SQP solver, which is used further in this text to determine solutions to the nonlinear SAC design equations. The ability to determine optimal solutions to even nonlinear optimization problems can be invaluable to problems in which linear or convex functions cannot approximate the fundamental behaviour that must be optimized.

2.6.4 Metaheuristics

Beyond convex and nonlinear optimization problems, there exists a further class of optimization problem, called *hard optimization problems*. Hard optimization problems resist traditional methods of solution through either their strong nonlinearity, presence of many local optima, large complexity, or highly limited ability to make function evaluations.

A large variety of metaheuristics exist for optimization, and it is not immediately apparent which of these is most applicable for any given problem. Wahab [34] performed a thorough review of metaheuristic techniques on several standardized datasets of hard optimization problems. Of the metaheuristics sampled, several techniques performed consistently well across datasets.

Differential Evolution (DE), makes use of genetic mutation to create candidate solutions, while a similar variant, Strategy-adaptive Differential Evolution (SaDE), chooses between various mutation strategies to improve convergence. Particle Swarm Optimization (PSO) gives each agent a velocity through the parameter space, and attraction to other members of the population to ensure convergence. The further addition of selection pressure to PSO creates Selection Particle Swarm Optimization (SPSO) to further increase convergence. Each of these metaheuristics will be outlined, and used in future chapters to improve SAC parameter selection. Of these techniques, Takagi [36] has previously used DE to determine SAC parameters that improve SAC formulations, where it was found that the DE search technique was able to improve convergence of the designed SAC's response.

Each of the four metaheuristic techniques mentioned here will be used in future efforts to determine their usefulness for improving SAC designs, and are described individually in the following sections. The term *position* is used to refer to a single combination of parameters within the search space as an analogy. The subscript i is used to refer to a parameter specific to an individual agent within the swarm, while the subscript d_d is used to refer to a given dimension within one of the agent's values.

Particle Swarm Optimization

The PSO metaheuristic uses the analogy of position and velocity in order to determine agent updates. The update to each agent's position, described in Eq. (2.235), is dependant on that agent's previous velocity, the position of the agent's best known position \mathbf{p}_p , and the swarm's best known position \mathbf{p}_g . The designer chooses the parameters ω_p , ϕ_p , and ϕ_g . The parameter ω indicates how quickly an agent loses its previous velocity, while ϕ_p affects each agent's attraction to the agent's best known position, and ϕ_g affects the agent's attraction to the swarm's best known position.

To begin a PSO search [35] with designed velocity dampening factor ω_p , agent optimum attraction ϕ_p , and global optimum attraction ϕ_g :

1. Randomly assign an initial position \mathbf{x}_i for each agent in the search space.
2. Set the best position \mathbf{p}_i for each agent to \mathbf{x}_i .
3. Determine the cost for each agent's position, called O , and set the swarm's minimum cost g to the lowest cost ($g = \min(O(\mathbf{p}_i)) \forall i$), with an associated best position \mathbf{q} . Call the current best known minimum value for each agent j_i .
4. Determine the initial velocity \mathbf{v}_i of each agent as a random vector.
5. Repeat iteratively until the completion criteria are met, for each agent i :
 - (a) Pick random values r_p and r_g between 0 and 1.
 - (b) Update the agent's velocity as:

$$\mathbf{v}'_i = \omega_p \mathbf{v}_i + \phi_p r_p (\mathbf{p}_i - \mathbf{x}_i) + \phi_g r_g (\mathbf{q} - \mathbf{x}_i) \quad (2.235)$$

- (c) Update the agent's position following:

$$\mathbf{x}'_i = \mathbf{x}_i + \mathbf{v}'_i \quad (2.236)$$

- (d) Check the cost of the new position $O(\mathbf{x}'_i)$.
- (e) If the new cost is lower than the agent's best cost j_i , make the agent's new best position \mathbf{p}_i equal to the current position and update the agent's best cost j_i .
- (f) If the agent's new cost is lower than the swarm's best cost g , make the swarm's best position \mathbf{q} the current agent's position and make the new best cost g equal to the current agent's cost j_i .

Differential Evolution:

The DE search makes use of mutation to produce randomized test points to decrease the cost. Test positions are created through combination of a mutation vector with the agent's best known position. The mutation vector itself is a random combination of the best known positions of two other members of the swarm. If an agent's test position is better than that agent's current position, then the test position becomes the agent's new position. Unlike PSO, which is constantly changing the test position of each agent, DE agents are always located at their best known location and several test points can be tested before the agent moves to a lower cost position. By producing test points using combinations of the swarm's best known positions the swarm tends to approach the best performing members of the swarm. Two variables are available to the designer. The crossover rate δ_d affects the chance that elements of the mutation vector will be used in the next test vector, while the scaling factor F affects the spread of mutation vector positions.

To perform a DE search [36] with designed crossover rate δ_d and scaling factor F :

1. Initialize each agent with a random position \mathbf{p}_i within the search space.
2. Determine the cost of each agent j_i at position \mathbf{p}_i .
3. Determine which agent's position has the lowest cost, marking it as \mathbf{q} , and saving the lowest cost as g .
4. Repeat iteratively until the completion criteria are met, for each agent i :
 - (a) Pick two random integers from 1 to the size of the swarm population that are not identical and not i , calling them a_1 and a_2 .
 - (b) Produce a mutated vector \mathbf{M} from the a_1^{th} and a_2^{th} agent following:

$$\mathbf{M} = \mathbf{q} + F(\mathbf{p}_{a_1} - \mathbf{p}_{a_2}) \quad (2.237)$$

- (c) Choose an integer b_d at random from 1 to the dimensionality of the problem.

- (d) Create a trial vector by checking the following for each dimension d_d using the design parameter δ_d :
- i. Pick a random value ρ_d from 0 to 1.
 - ii. The trial vector \mathbf{u}_i is defined for each dimension d_d as:

$$u_{d_d} = \begin{cases} M_{d_d}, & d_d = b_d \text{ or } \rho_d < \delta_d \\ p_{d_d}, & \text{otherwise} \end{cases} \quad (2.238)$$

Each dimension u_{d_d} is collected to form the trial vector \mathbf{u}_i .

- (e) Determine the current cost of the trial vector \mathbf{u}_i .
- (f) If the cost of \mathbf{u}_i is lower than the current position cost j_i , the agent's best known position \mathbf{p}_i becomes equal to the trial vector \mathbf{u}_i .
- (g) If the cost of \mathbf{u}_i is lower than the best global cost g , the global best known position \mathbf{q} becomes the trial vector \mathbf{u}_i .

Selection Particle Swarm Optimization:

The SPSO search increases the convergence of the PSO algorithm by adding selection pressure to each of the agents. Directly before the update step, the half of the swarm with the highest cost is randomly given a position from the half of the swarm with the lowest cost. The PSO velocity update and position update steps are then performed normally.

To begin an SPSO search [71]:

1. Initialize similarly to PSO steps (1) through (4)
2. Iteratively until the completion criteria are met:
 - (a) Sort the population by cost and mark the half of the agents with the highest cost for selection. Replace the current positions \mathbf{x}_i of each agent marked for selection randomly with one of the agents not marked for selection. Replace the current velocity \mathbf{v}_i of each agent marked for selection randomly with one of the agents not marked for selection. Keep the best known position \mathbf{p}_i of each agent unchanged.

- (b) Proceed with the optimization identically to the iterative steps in the PSO search from step (5a) through (5f).

Strategy-Adaptive Differential Evolution

The SaDE strategy leverages the many varied mutation strategies that have been developed for DE searches into one cohesive metaheuristic. A pool of mutation strategies is kept by SaDE and used to improve the search. Strategies that are more likely to result in success are more likely to be chosen in future. A success is recorded as any time a strategy produces a trial vector that decreases the best cost of an agent, with the total success of the m^{th} strategy during the current iteration k being denoted by $s_{k,m}$. When the cost does not decrease it is recorded as a failure, denoted by $f_{k,m}$. Pooling mutation strategies and recording which are most effective for the current optimization problem increases the exploitation of discovered minima, while decreasing the exploration of the SaDE technique when compared with DE.

To begin a SaDE search with M trial vector generation function strategies, and learning period L_P [37]:

Initialize similarly to DE steps (1) through (3), then repeat iteratively, calling the iteration number k , until the completion criteria are met:

1. If the current iteration number k is greater than the designed learning period L_P then:
 - (a) Calculate the success fraction S of each strategy throughout the learning period following:

$$S_{k,m} = \frac{\sum_{t=k-L_P}^k s_{k,t}}{\sum_{t=k-L_P}^k s_{k,t} + \sum_{t=k-L_P}^k f_{k,t}} + \epsilon \quad (2.239)$$

Values for $s_{k,t}$ and $f_{k,t}$ are recorded later. A small number ϵ is added for numerical stability.

- (b) Determine the probability of choosing the m^{th} strategy $P_{k,m}$ following

$$P_{k,m} = \frac{S_{k,m}}{\sum_{t=1}^M S_{k,t}} \quad (2.240)$$

- (c) Determine the new crossover ratio median δ_m as the average of crossover ratios δ_i that resulted in successful trial vectors over the last L_P iterations. The record of successful crossover ratios is made in a later step.
2. If $k < L_P$, set the probability for each trial vector generation function $P_{k,m}$ to be equal, such that each function has an equal likelihood of being chosen.
3. For each agent i :
- (a) Use probabilities $P_{k,m}$ that sum to 1, and a random variable from 0 to 1 to choose a strategy index m .
 - (b) Choose F_i and G_i as normal random values with a standard deviation of 0.3 and median of 0.5.
 - (c) Choose δ_i as a normally distributed random value with median of δ_m and standard deviation of 0.1, ensuring $0 < \delta_i < 1$.
 - (d) Use the m^{th} strategy to create the trial vector \mathbf{u}_i . Example trial vector generation strategies are compiled later.
 - (e) Determine the current cost of the trial vector \mathbf{u}_i .
 - (f) If the cost of \mathbf{u}_i is lower than j_i , set j_i to the current cost, and set $\mathbf{p}_i = \mathbf{u}_i$. Increase the number of successes $s_{k,m}$ for the chosen strategy m by one and add the crossover ratio δ_i used to the list of successful crossover ratios. If the cost of \mathbf{u}_i was not lower than the current cost j_i , increase the number of failures $f_{k,m}$ for the strategy m by one.
 - (g) If the cost of the trial vector \mathbf{u}_i is lower than the best global cost g , the global best known position \mathbf{q} becomes the trial vector \mathbf{u}_i .

The four strategies used are outlined in Eqs. (2.241) through (2.244) and are taken from Qin and Huang [37]. When used in parallel the methods require five random distinct integers a_1, a_2, a_3, a_4, a_5 with values from 1 to the swarm population that correspond to indices of members of the population that are not the agent being considered. These strategies also require an integer b_d between 1 and the dimensionality of the problem, a random value ρ_d between 0 and 1, the previously determined F_i ,

G_i , and δ_i values, the value of any i^{th} agent's position in the d_d^{th} dimension \mathbf{p}_{i,d_d} and the best known position in that dimension \mathbf{q}_{d_d} . The strategies and their names are listed below.

1. DE/rand/1/bin: For each dimension d_d :

$$\mathbf{u}_{i,d_d} = \begin{cases} \mathbf{p}_{a_1,d_d} + F_i(\mathbf{p}_{a_2,d_d} - \mathbf{p}_{a_3,d_d}), \rho_d < \delta_i \text{ or } d_d = b_d \\ \mathbf{p}_{i,d_d}, \text{ otherwise} \end{cases} \quad (2.241)$$

2. DE/rand-to-best/2/bin: For each dimension d_d :

$$\mathbf{u}_{i,d_d} = \begin{cases} \mathbf{p}_{i,d_d} + F_i(\mathbf{q}_{d_d} - \mathbf{p}_{i,d_d}) + F_i(\mathbf{p}_{a_1,d_d} - \mathbf{p}_{a_2,d_d}) \\ \quad + F_i(\mathbf{p}_{a_3,d_d} - \mathbf{p}_{a_4,d_d}), \rho_d < \delta_i \text{ or } d_d = b_d \\ \mathbf{p}_{i,d_d}, \text{ otherwise} \end{cases} \quad (2.242)$$

3. DE/rand/2/bin: For each dimension d_d :

$$\mathbf{u}_{i,d_d} = \begin{cases} \mathbf{p}_{a_1,d_d} + F_i(\mathbf{p}_{a_2,d_d} - \mathbf{p}_{a_3,d_d}) \\ \quad + F_i(\mathbf{p}_{a_4,d_d} - \mathbf{p}_{a_5,d_d}), \rho_d < \delta_i \text{ or } d_d = b_d \\ \mathbf{p}_{i,d_d}, \text{ otherwise} \end{cases} \quad (2.243)$$

4. DE/current-to-rand/1:

$$\mathbf{u}_i = \mathbf{p}_i + G_i(\mathbf{p}_{a_1} - \mathbf{p}_i) + F_i(\mathbf{p}_{a_2} - \mathbf{p}_{a_3}) \quad (2.244)$$

2.7 Chapter Summary

The uncooperative spacecraft problem involves very many areas of active research. This chapter served to introduce the dynamics of spacecraft flight, of spacecraft relative motion, linear and nonlinear dynamical systems, linear and adaptive control, and the state of mathematical optimization theory. All of these fields are necessary in order to determine what can be done to improve adaptive control formulations,

and thereby ensure that spacecraft autonomy can be improved.

The foundation presented in this section will be used to develop adaptive control techniques. Where possible improvements to SAC design methodologies and techniques will be made. A thorough understanding of each of these domains will be leveraged to improve the performance and autonomy of space systems.

The equations of motion for spacecraft are used to determine how to manage position and orientation in the uncontrolled spacecraft problem. The dynamics of relative spacecraft motion are leveraged to ensure that a chaser spacecraft can successfully rendezvous with the target. Approximations of the system dynamics can be made using linear dynamics systems, with the full system dynamics being described by nonlinear system dynamics. Linear controllers provide sufficient control when the linear approximation of a system is known, and ample tools exist to ensure that control system designers can develop good linear controllers. Optimal linear controllers have been developed to maximize performance in a known linear system, or the linear approximation of a nonlinear system. Adaptive controllers have been developed to ensure flexibility when there is a large amount of system uncertainty. Simple adaptive control is a direct adaptive controller methodology that has been tested and is well suited to the uncontrolled spacecraft problem. Various changes have been made to the SAC architecture to ensure that it can be used with a large variety of control system. However, a number of questions still remain around implementations of SAC.

Optimization techniques have been used previously to determine optimal controllers such as the linear quadratic regulator. In order to determine how SAC can be optimized, mathematical optimization techniques must be understood. Optimization problems can be broadly divided into three classes: convex problems, nonlinear problems, and hard problems. Convex problems are the easiest to solve, but have the most restrictive definition. Nonlinear optimization techniques cover a larger variety of systems, but may not be able to solve all nonlinear problems. The solutions provided by nonlinear optimization techniques are not necessarily globally optimal. Candidate solutions to nonlinear and hard optimization problems can be found using metaheuristics, which trade solution optimality for computational efficiency using stochastic methods. Metaheuristics make use of multiple agents that semi-randomly

wander the parameter space to determine good solutions to the optimization problem. There is no guarantee that a metaheuristic search will find the global optima, however they are more suited to multimodal problems than nonlinear search techniques.

In the following chapters, knowledge of spacecraft dynamics, nonlinear system dynamics, and optimization techniques will be leveraged to improve implementations and performance of SAC.

Chapter 3

Simple Adaptive Controller Implementations

3.1 Introduction

This work will present the many efforts undertaken to determine how Simple Adaptive Controller (SAC) implementations can be improved. The continuous SAC stability proof shown in Theorem 2.5.1 shows that for any ASPR system SAC can match the system response to the ideal model response. Furthermore, Sec. 2.5.5 showed that any stabilizable continuous-time non-ASPR plant can be rendered ASPR through feedforward parallelization, meaning that any system with existing linear control can be modified to include adaptive control. The goal of this work is to determine how implementations of SAC can be improved.

The analysis techniques, theories, and outcomes of research into the formulation and performance of SACs are discussed and developed here. Nonlinear and metaheuristic optimization techniques are used to determine improved SAC designs for spacecraft proximity operations, which are experimentally verified in Chapter 4. Heuristics and design techniques are developed for selection of the ideal model, integral adaptation parameters, and proportional adaptation parameters. The effects of disturbances on a feedforward parallelized SAC are explored, and previously described disturbance accommodation techniques are leveraged to minimize tracking error. Finally, accumulated knowledge on the design and implementation of SACs is collected for quick reference at the end of the chapter.

3.2 Nonlinear Optimization and Metaheuristics in SAC Design

It is rare for researchers to present reasoning for the selection of SAC design parameters. Manual “trial-and-error” determination of parameters is fraught with frustration, as the nonlinear system response changes nonlinearly with the design parameters. The

design space of acceptable parameters for any one SAC is also extremely large; it is not uncommon for stable choices of design parameter for a SAC implementation to range from 10^{-6} to 10^{10} , a huge parameter space to manually search. Even worse, a SAC can appear to have poor performance when a design parameter is either too small *or* if that design parameter is too large, with no clear indication of when a parameter is one or the other. It is apparent that optimal SAC parameter selection is not a trivial problem.

Attempts to determine optimal SAC formulation were previously made by Takagi et al. [66] through the use of DE on a single-input-single-output (SISO) SAC simulation. Many metaheuristic optimization techniques exist apart from DE, however. A review of metaheuristic techniques by Ab Wahab et al. [34] highlights that DE, SaDE, PSO, and SPSO searches are also useful for multimodal optimization problems. Each of these optimization techniques is probed in a simulation of spacecraft trajectory tracking in Predmyrskyy and Ulrich [72], the results of which are summarized here.

Although metaheuristics were shown by Takagi et al. [66] to be useful for lowering the cost of a SAC design, it was still unclear if other optimization techniques could also be used to improve SAC design. The SAC design equations are *not* convex, and as such convex techniques cannot be used. Nonlinear optimization techniques are tested for solutions to the design problem.

The metaheuristic and nonlinear optimization techniques developed here for spacecraft trajectory tracking in simulation are repeated for the SRCL SPOT hardware and experimentally verified later in Chapter 4.

3.2.1 Preliminary Metaheuristic Optimization of a SAC

To begin, a simulation of a spacecraft position controller is made to evaluate the applicability of various metaheuristic searches to SAC design. Metaheuristic optimization was used to minimize the value O of a linear quadratic cost function over the simulation period. The cost to be minimized is given by the equation

$$O(t_f) = \int_0^{t_f} \mathbf{e}_y^T(t) \mathcal{Q} \mathbf{e}_y(t) + \mathbf{u}_p^T(t) \mathcal{R} \mathbf{u}_p(t) dt \quad (3.1)$$

The weighing matrices \mathcal{Q} and \mathcal{R} determine the relative importance of minimizing

the tracking error or control inputs, respectively. The linear-least-squares cost function is used specifically to simplify comparison between the optimized SAC designs and a reference LQR controller for the same system.

Each of the DE, SaDE, PSO, and SPSO searches are implemented to determine SAC design parameters $\mathbf{\Gamma}_I$, $\mathbf{\Gamma}_P$ and $\boldsymbol{\sigma}$ for a nonlinear simulation of a spacecraft with thruster saturation of 0.425 N, and mass of 16.95 kg.

The LQR controller was designed using a linear approximation of the system, with LTI state-space matrices $\{\mathbf{A}_{LQR}, \mathbf{B}_{LQR}, \mathbf{C}_{LQR}, \mathbf{0}\}$, of

$$\mathbf{A}_{LQR} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \mathbf{B}_{LQR} = \begin{bmatrix} 0 \\ 1/16.95 \end{bmatrix}, \mathbf{C}_{LQR} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (3.2)$$

for the state vector

$$\mathbf{x} = \begin{bmatrix} x_p \\ \dot{x}_p \end{bmatrix} \quad (3.3)$$

being the position and velocity of the spacecraft. Since there is no difference between the x and y position axes, the controller is repeated for both directions.

The weighing matrices

$$\mathcal{Q} = \begin{bmatrix} 1000 & 0 \\ 0 & 1000 \end{bmatrix}, \mathcal{R} = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix} \quad (3.4)$$

were used for both the SAC and LQR optimizations. From the LQR system equations described in Eqs. (2.60) and (2.61) the gain for the LQR system can be determined as

$$\mathbf{K}_{LQR} = \begin{bmatrix} 31.6 & 45.5 \end{bmatrix} \quad (3.5)$$

which multiplies measurements of the state vector \mathbf{x} to determine the thruster control power.

The SAC ideal model is a second order transfer function with a damping ratio ζ of 1 and a large natural frequency ω_n of 40 rad/s, which corresponds to a 2% settling time of $t_s = 0.15$ seconds. The ideal model used is particularly fast to simplify comparison between the SAC, which follows the ideal model, and the LQR, which

tracks the input command. The ideal model state-space matrices are given by

$$\mathbf{A}_m = \begin{bmatrix} -0.25 & -0.0156 \\ 1 & 0 \end{bmatrix}, \quad \mathbf{B}_m = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad (3.6)$$

$$\mathbf{C}_m = \begin{bmatrix} 0 & 0.0156 \end{bmatrix}, \quad \mathbf{D}_m = \begin{bmatrix} 0 \end{bmatrix} \quad (3.7)$$

Since a second order model was used, two adaptation parameter $\mathbf{\Gamma}$ values are present for the state adaptation parameter values. The final adaptation parameter matrices are in the form

$$\mathbf{\Gamma}_{xP} = \text{diag}(\mathbf{I}_2\mathbf{\Gamma}_{xP1}, \mathbf{I}_2\mathbf{\Gamma}_{xP2}) \quad (3.8)$$

$$\mathbf{\Gamma}_{xI} = \text{diag}(\mathbf{I}_2\mathbf{\Gamma}_{xI1}, \mathbf{I}_2\mathbf{\Gamma}_{xI2}) \quad (3.9)$$

The remaining adaptation parameters follow the formulation in Eqs. (2.100) through (2.105), with the sigma modification as described in Eq. (2.111) being applied to only the tracking error adaptation \mathbf{K}_e .

Each metaheuristic search was implemented in MATLAB[®], with Simulink[®] calls to the controller simulation determining the cost of any given set of design parameters. The metaheuristic search would determine a set of parameters to test, these parameters would then be used to run a simulation of the system that returned the tracking error \mathbf{e}_y and control activation \mathbf{u}_p that then determined the cost of that SAC design. At the end of the search the final solution would be the controller design determined by the metaheuristic search to yield the lowest cost. A manually designed SAC was created for comparison to the metaheuristic optimized controllers, with each controller being tested and optimized for a reference circular trajectory of angular velocity 0.035 rad/s and radius of 1 m.

The true plant dynamics in the simulation are identical to the LQR dynamics, that is to say

$$\mathbf{A}_p = \mathbf{A}_{LQR}, \quad \mathbf{B}_p = \mathbf{B}_{LQR}, \quad \mathbf{C}_p = \mathbf{C}_{LQR}, \quad \mathbf{D}_p = \mathbf{D}_{LQR} = \mathbf{0} \quad (3.10)$$

Table 3.1: Metaheuristic Optimized SAC Parameters

Variable	Designed SAC	DE	PSO	SaDE	SPSO
Γ_{eP}	100	4.394×10^5	7.007×10^4	4.938×10^5	1.225×10^5
Γ_{eI}	1×10^5	0	1.277×10^5	0	4.513×10^5
Γ_{uP}	100	614.5712	1.582×10^5	0	5.345×10^5
Γ_{uI}	1	0	0	0	0
Γ_{xP1}	100	0	2.079×10^4	867.9	2.580×10^5
Γ_{xP2}	100	1.000×10^6	1.672×10^4	4.851×10^5	3.833×10^5
Γ_{xI1}	1	0	0	0	0
Γ_{xI2}	1	1.000×10^6	5.827×10^4	2.421×10^5	9.998×10^5
σ	0.4	0.93	0.21	0.54	0.02
Cost	3338.6	1283.5	3337.6	1405.9	3184.0

All four metaheuristic optimization algorithms were run for 100 iterations with 100 agents. Values of $\omega_p = 0.2$, $\phi_g = 0.1$, and $\phi_r = 1.0$ were chosen for the PSO search. The DE search was performed using values of 0.9 and 0.8 for the crossover rate and scaling factor, respectively. For SPSO, 50% of the agents were used to provide updated positions of the other 50% of agents. The designed SAC controller gains and their optimized counterparts are found in Table 3.1, along with the final cost of each controller in the tested circular simulation.

Each of these controllers was then verified on a cycloid trajectory of the form

$$x_{cmd}(t) = 1.428 \cos(0.035t) + 1 \sin(3 \cdot 0.035t) \quad (3.11)$$

$$y_{cmd}(t) = 1.428 \sin(0.035t) + 1 \cos(3 \cdot 0.035t) \quad (3.12)$$

The trajectory tracking results for the designed LQR and SAC controllers are shown in Figs. 3.1 through 3.3.

In all cases the SAC controller was able to improve on the command trajectory tracking of the LQR controller. The DE and SaDE metaheuristics were able to determine lower cost than the PSO and SPSO controllers. The lack of disturbances or noise in the simulation environment likely affected the final parameters chosen by metaheuristic optimization.

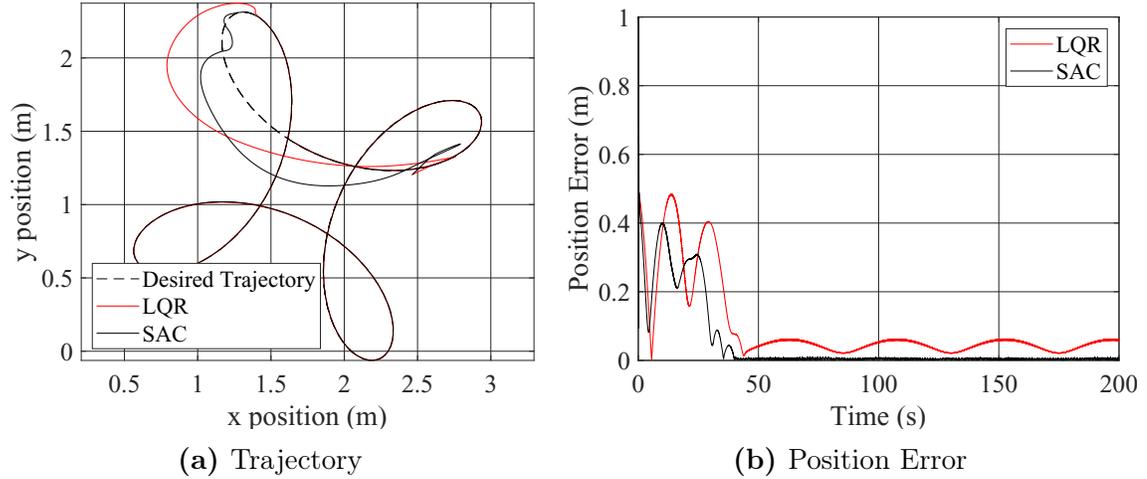


Figure 3.1: LQR and Manually Tuned SAC Performance

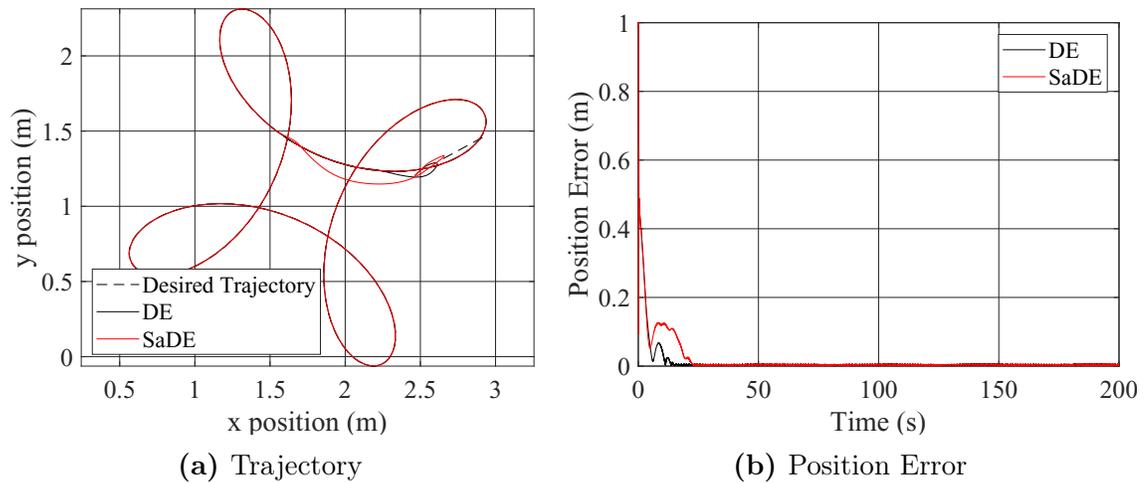


Figure 3.2: DE and SaDE Optimized SAC Performance

The designed LQR controller achieved similar transient behaviour as the manually designed SAC, but was unable to reach zero error.

It can be seen that all optimization methods were able to find parameters that improve the response when compared with a manually designed SAC. The DE and SaDE searches were able to determine significantly lower cost controllers than the PSO and SPSO searches. While the designed controller contains mostly feedback error adaptation, The DE determined controller contains mostly state adaptation. The DE controller converges on the model response very quickly, which is reflected in the cost. Optimization significantly improved the controller cost. It is likely

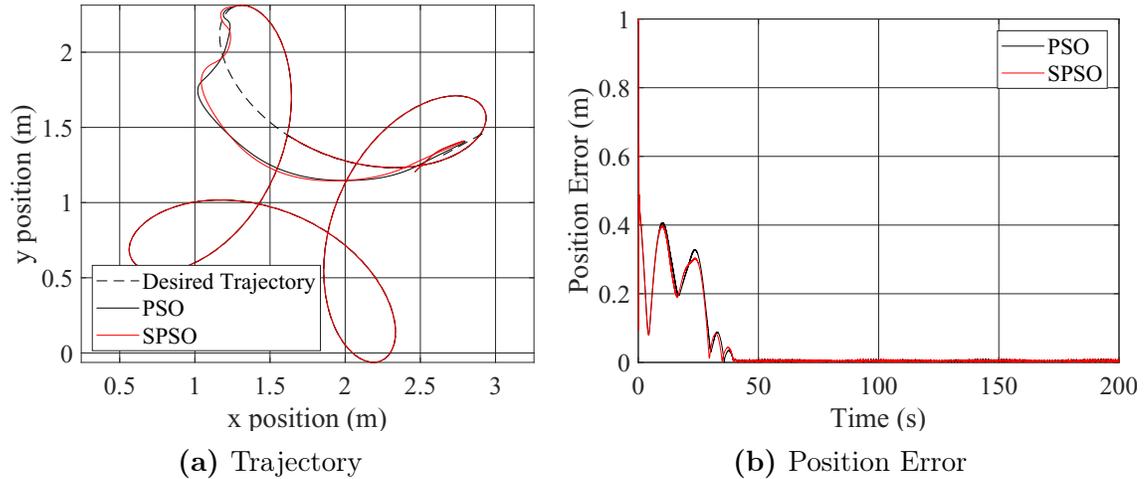


Figure 3.3: PSO and SPSO Optimized SAC Performance

that the absence of disturbances and measurement noise caused the optimization to ignore error adaptation Γ_{eI} since adequate control could be achieved with only state adaptation. The SPSO search produced very similar results to the PSO search, using high error and state adaptations to improve the tracking of the hand-designed controller. The use of selection in SPSO likely increased convergence by moving high cost agents closer to the lower cost agents and more quickly refining the determined control parameters. The faster convergence of PSO and SPSO techniques compared to DE and SaDE may have contributed to the higher final cost of their designs, due to lower exploration of the design space. Generally it was found that convergence and exploration behaviour of the techniques matched those presented in Wahab et al. [34], with SPSO having the fastest convergence time, at the expense of exploration, followed by PSO, SaDE, and finally DE which more thoroughly explored the search space at the expense of convergence. The explicit attraction of PSO to the determined minimum increases convergence when compared to DE, with selection pressure further increasing convergence in SPSO. The mutations present in DE encourage exploration but reduce convergence behaviour, and the use of multiple strategies in SaDE increase the convergence behaviour of DE somewhat. The simulation used in this survey did not include any measurement noise or disturbances. It is possible that due to the lack of complicating factors in simulation that some of the determined controller parameters may not be useful in practice. For example, although the DE controller

has the lowest cost, it does not make use of error adaptation at all, and may have a higher cost response than a controller including these terms when implemented in hardware. Similarly, all controllers used very large values for many of the adaptation terms, which may cause instability in slower hardware implementations.

Finally, since several parameter configurations were able to produce similar costs for the controller, it is likely that even for quadratic cost functions there exists a complex cost landscape for parameter selection in SACs. Wahab et al. [34] suggests that DE has the best performance of swarm-based techniques for multimodal cost functions, which may be the case for many SAC designs.

It is clear from simulation that application of metaheuristics to SAC design can yield controllers that improve on the performance of traditional linear controllers, as well as improving the performance of manually tuned SACs.

Metaheuristics are applied and experimentally verified on SRCL's SPOT platform in Chapter 4, after several critical elements of SAC implementation are clarified in the following sections.

3.3 Simple Adaptive Control Design and Heuristics

The aim of this section is to simplify the process of SAC design by presenting clarifications and design heuristics on the development of SAC architectures for physical control systems.

First, the role of the ideal model in SAC implementations is explored and clarified. The ideal model is developed to ensure that limitations of physical systems do not affect SAC's ability to adapt and match the ideal response. Then, information on the control gains and adaptation parameters are leveraged to simplify selection of design parameters. Heuristics are developed to determine appropriate values of the integral and proportional adaptations of a system. The effect of disturbances and disturbance compensation for a typical SAC is discussed. Finally, a short summary of the recommended SAC design process is included to simplify future implementations.

3.3.1 Ideal Model as Guidance

The ideal model used in SAC provides the template that the adaptation scheme will attempt to match. The SAC stability proof provided in previous sections ensures that, when sufficient control power is present in an ASPR system, SAC will always match the model. Furthermore, it is possible for SAC to match the ideal model even when ideal gains do not exist, which is discussed in Barkana [73]. When ideal gains are present, they can be determined using the matching conditions detailed in Sec. 2.5.3 Eqs. (2.115) through (2.118).

When implementing a SAC into a control system, the designer has the choice of ideal model for the adaptive controller. The theoretical framework for SAC suggests that any choice of ideal model, even a very simple or very fast one, is correct. A great deal of effort has been put into verifying and reverifying that it is indeed true that, with sufficient adaptation time and a truly ASPR system, that SAC will converge to the ideal response.

When designing an ideal model the designer must decide what constitutes an appropriate model, instead. The model matching conditions in Eqs. (2.115) through (2.118) clarify that only the form of the model determines if ideal gains exist; if a SAC can match the plant response to a second-order reference model, then it does not matter if that model has a response time of two seconds or two hundred seconds. The only difference that model selection has on SAC response is to affect the magnitude of the final gains, and in effect the final control output.

For example, the ASPR plant characterized by the transfer function

$$G(s) = \frac{1}{s + 1} \quad (3.13)$$

can be made to match a second-order model with transfer function and state-space representation

$$M(s) = \frac{1}{s^2 + s + 1} \quad (3.14)$$

$$\mathbf{A}_m = \begin{bmatrix} -2 & -1 \\ 1 & 0 \end{bmatrix}, \quad \mathbf{B}_m = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad (3.15)$$

$$\mathbf{C}_m = \begin{bmatrix} 0 & 1 \end{bmatrix}, \quad \mathbf{D}_m = \begin{bmatrix} 0 \end{bmatrix} \quad (3.16)$$

with ideal gains

$$\mathbf{K}_x^* = \begin{bmatrix} 1 & 1 \end{bmatrix} \quad (3.17)$$

$$\mathbf{K}_u^* = \begin{bmatrix} 0 \end{bmatrix} \quad (3.18)$$

from the matching conditions in Eqs. (2.115) through (2.118).

For the same plant, but ideal model of

$$M(s) = \frac{4}{s^2 + 2s + 4} \quad (3.19)$$

$$\mathbf{A}_m = \begin{bmatrix} -4 & -4 \\ 1 & 0 \end{bmatrix}, \quad \mathbf{B}_m = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad (3.20)$$

$$\mathbf{C}_m = \begin{bmatrix} 0 & 2 \end{bmatrix}, \quad \mathbf{D}_m = \begin{bmatrix} 0 \end{bmatrix} \quad (3.21)$$

the matching conditions now provide the gains

$$\mathbf{K}_x^* = \begin{bmatrix} 2 & 2 \end{bmatrix} \quad (3.22)$$

$$\mathbf{K}_u^* = \begin{bmatrix} 0 \end{bmatrix} \quad (3.23)$$

In effect, the gains have doubled because the ideal model is faster, but the ideal gains still exist. A step command to the faster ideal model produces larger control activations for the same command. A physical system, however, will always have a maximum control activation that must be taken into account when choosing the ideal model. The choice of system commands and system responses that ensure control activations stay under a maximum is one of the fundamental guidance problems.

The role of the ideal model, then, should be understood as choosing the guidance of the adaptive system. If a control system has a maximum control output, then it is unwise to choose an ideal model that would require more than the maximum control effort to track the model. Further efforts should be made to determine what elements of guidance theory can be leveraged to improve SAC formulations.

3.3.2 Ideal Models for Nonlinear Systems; Feasible Models

The development of SAC so far has focused completely on linear systems. For SAC to be useful it must be able to control systems with nonlinearities, similarly to how linear control techniques can still control some classes of nonlinear systems. If SAC cannot be applied to systems with small nonlinearities, then it will be impossible to apply it to any real system without fear of the system going unstable whenever the nonlinearity is encountered. Therefore some time will be taken to understand how nonlinearities in the plant can affect the performance of SAC when applied to these systems.

Nonlinearities that are managed in implementation of linear controllers can still be managed in implementations of SAC. For example, in attitude control, the system dynamics in the body reference frame are governed by the nonlinear equation

$$\mathbf{J}\dot{\boldsymbol{\omega}} = \mathbf{T} - \boldsymbol{\omega}^\times(\mathbf{J}\boldsymbol{\omega}) \quad (3.24)$$

for the angular velocity $\boldsymbol{\omega}$, Torque \mathbf{T} , and moment of inertia \mathbf{J} . Typically the nonlinear term $\boldsymbol{\omega}^\times(\mathbf{J}\boldsymbol{\omega})$ is compensated through measurement of the angular velocity and measurements of the nominal moment of inertia. For systems where these values are already known, the rotational nonlinearity can be addressed through dynamic inversion in SAC the same way it is addressed in linear controllers.

Adaptive control can expand the control envelope outside the traditional linear control envelope, though. By expanding the control envelope to systems with unknown moments of inertia, adaptive control introduces a new control problem. The issue for adaptive controller implementations in this case, then, is that their expansion of the linear control envelope cannot be met with a similar expansion of the nonlinear control envelope, which cannot be managed by linear controllers to begin

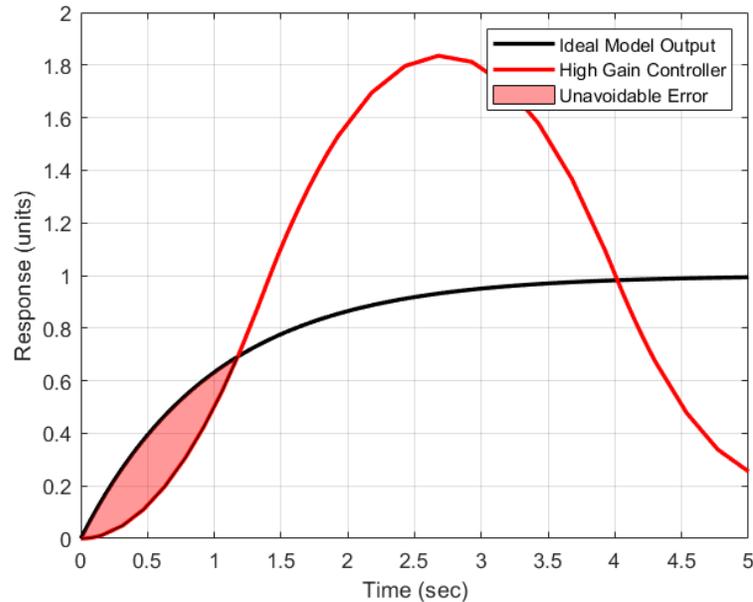


Figure 3.4: Ideal Model Requires More Thrust Than System Can Deliver

with. When SAC cannot inherently compensate for additional nonlinearities, additional disturbance compensation techniques such as nonlinear dynamic observers or real-time estimation techniques, to name but two, may be necessary to maintain performance or ensure stability.

Two techniques are recommended for the designer to avoid unwanted behaviour around nonlinearities: adaptation should be halted when no additional control can take place and/or the ideal model presented to the SAC should always represent an achievable response.

An example of nonlinearities creating an unachievable ideal model response is demonstrated in Fig. 3.4, where a control saturation makes it impossible for a high gain controller to match the ideal model response. When the ideal model is unachievable, there is an unavoidable amount of tracking error, which results in unavoidable adaptation. Stability of the system may be compromised as the system attempts to match an ideal model that cannot be matched.

Whenever the SAC outputs commands beyond the control saturation of the system, if it can be measured, then adaptation should be halted. Unfortunately, halting

adaptation can also be undesirable; if a SAC is poorly tuned when a sustained saturated command is required, especially if disturbances or changes in the plant occur during this saturation, then it is possible that poor tracking will occur during command saturation, and that sudden adaptation will occur once the saturation ends. Halting adaptation during command saturation may be a safer alternative, to ensure adaptation does not “wind up” similar to linear integrator control, however stopping adaptation altogether may be undesirable in systems where fast adaptation is always required, or if commands are frequently saturated.

An alternative to stopping gain adaptation outside of the linear region is the introduction of a *feasible model*. One of the fundamental assumptions of the SAC architecture is that the ideal model represents an achievable system response. When nonlinearities are present it is important that the model continues to represent an achievable response. The concept of a feasible model follows naturally from attempting to provide an achievable system response.

Consider the example of an ideal model that gives a position command under limited thrust. The ideal response given to the SAC must be achievable even under thrust saturation. When the saturation is known, an achievable response might be constructed as in the first block diagram of Fig. 3.5.

By limiting the ideal response to an achievable response, the adaptive controller does not attempt to adapt when the thrust is limited. The initial block diagram shown in Fig. 3.5 serves to present an achievable response, however, the output will not match the position command input into the model. The achievable response does not match the command because the limiting nonlinearity stops the achievable response from reaching the command.

Unavoidably, by ensuring that a feasible model output is used to adapt the SAC, a new control problem is created. The feasible model output should approach the ideal model output when there is no saturation. It is therefore necessary to develop a new controller to ensure that the feasible model output matches the ideal model output when the nonlinearity is not present, while minimizing the effect on command tracking of the conversion from ideal to feasible model. The second and third diagrams in Fig. 3.5 shows how the limiting nonlinearity is now managed by a new controller.

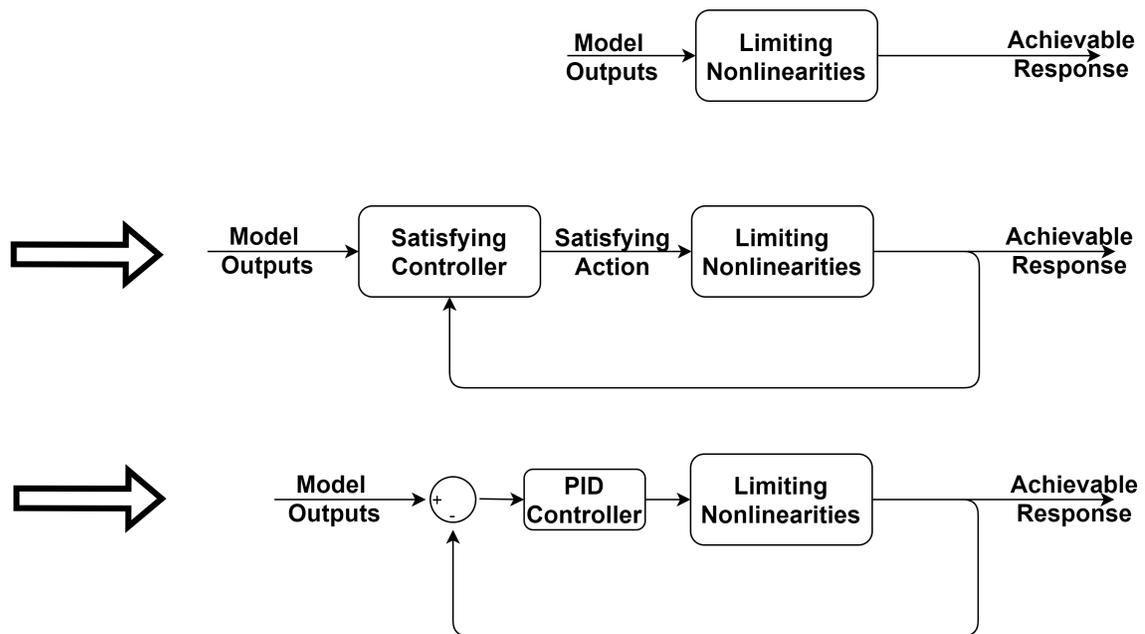


Figure 3.5: Origin of the Feasible Model Architecture. A Satisfying Controller Ensures the Feasible Response Approaches the Ideal Response.

An example of the new control problem is shown in Fig. 3.6, where a *satisfying controller* has been added to the feasible model to ensure the feasible model output matches the ideal model output as much as possible.

Even if the feasible model matches the system nonlinearity perfectly, only a SAC with ideal gains will be able to match the feasible response. Any additional error between the SAC gains and the ideal gains will lead to more unavoidable tracking error and adaptation. One final consideration might be to ensure the SAC always has sufficient control power to match the feasible response. Increasing the available control power might be done by slowing down the feasible model even further, so that the SAC might be able to continue adapting even if it does not have the ideal gains at the beginning of the response. This final act, of increasing the available control power to the SAC, is the same as decreasing the system bandwidth. Managing system bandwidth is a fundamental guidance problem, and it once again becomes clear that the problem of providing a feasible model for the SAC corresponds to a nonlinear guidance problem. In effect, the issue of developing a model response for SAC, in nonlinear and linear systems, corresponds to a guidance problem for the system.

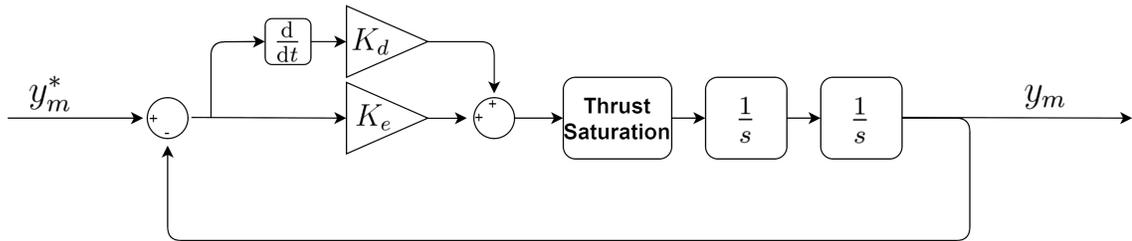


Figure 3.6: Feasible Model and Satisfying Controller for Example Thruster Saturation

An example of the concept of a feasible model can also be seen in Ulrich and deLaFontaine [28] where nonlinear guidance techniques are used to create a nonlinear ideal model for a Mars atmospheric re-entry vehicle. In order to manage system nonlinearities it is required that some characterization of the nonlinearities be available to the designer.

Additional research should be made into the applicability of nonlinear guidance techniques to the SAC architecture.

3.3.3 Adaptation Parameter Selection and Design Heuristics

Thorough analysis of the SAC design equations has clarified that no choice of proportional or integral adaptation will lead to instability for linear and discrete ASPR systems [63]. Furthermore, the stability equations suggest that larger values of the integral and proportional adaptation will always lead to faster convergence or smaller tracking errors.

The development of these theoretical results rest on three critical assumptions that follow from the ASPR condition, and are mentioned in Corollary 2.3.0.1:

1. The controlled system is linear, and therefore any desired control input can be achieved. A doubling of the input doubles the output response,
2. The controlled system is stable for all feedback gain values \mathbf{K}_e such that

$$\mathbf{K}_{\min} \leq \mathbf{K}_e \leq \infty = \mathbf{K}_{\max} \quad (3.25)$$

3. The controlled system, with feedback, has frequency response such that for any input, the output is never shifted more than $\pm 90^\circ$.

Although these assumptions are correct for ASPR systems, it is nonetheless true that very few examples of physical ASPR systems exist. Augmentation of a plant through feedforward parallelization and signal fusion are discussed in Sec. 2.5.5, however plant augmentations and signal synthesis will never be able to address the contradiction at the heart of SAC design; any system, no matter how idealized, will eventually fail to be accurately described by an ASPR plant.

There are numerous ways that any of the three ASPR assumptions above can be broken by implementations of SAC: system noise is amplified by the control gains, reducing stability at higher gain values; measurement dynamics introduce delay, breaking the ASPR phase requirement at higher frequencies; all systems have some form of control limiting, breaking the linear system requirement; no system is perfectly linear, even without control saturation; and discrete implementations will eventually face numerical issues. Any one of these or various other nonlinear effects are a reality of physical systems and threaten instability in implementations of SAC.

The same way linear control law can be applied to nonlinear systems, SAC can still be applied to systems that are not perfectly ASPR, so long as control can stay within an approximately ASPR region. That is to say, the system may be stable in operation even if the analytical stability proof no longer holds. For example, when designing a linear control system that is theoretically stable for all gain values, instability at large gains does not suggest that the linear system analysis was incorrect, simply that the linear approximation no longer holds for the current gains. Similarly, instability in SAC for large adaptation parameters should not be understood as a failure of analysis, simply that any one of the critical ASPR requirements is failing to be met for the large adaptations in use. In linear design, gains are reduced to ensure that linear system approximations holds, SAC designs must similarly determine what constitutes stable gain selection and choose adaptation parameters that keep the ASPR approximation applicable.

Within this and the following sections, an example default system will be used to elaborate design principles relevant to parameter selection when implementing a

SAC. First, a default SAC system is created. A plant with transfer function

$$G(s) = \frac{1}{s + 1} \quad (3.26)$$

is created, and converted to a discrete state space system with timestep 0.01 seconds, or 100 Hz. The ideal model is a second-order system with transfer function

$$M(s) = \frac{1}{s^2 + s + 1} \quad (3.27)$$

that is also converted to a discrete state-space model with the same timestep. The ideal gains for the SAC are found using the algorithm in Eqs. (2.115) through (2.118). The ideal gains are found to be

$$\mathbf{K}^* = \begin{bmatrix} K_e^* & K_{x1}^* & K_{x2}^* & K_u^* \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 0 \end{bmatrix} \quad (3.28)$$

where K_{x1}^* and K_{x2}^* can also be defined as elements of the ideal state gain vector

$$\mathbf{K}_x^* = \begin{bmatrix} K_{x1}^* \\ K_{x2}^* \end{bmatrix} \quad (3.29)$$

The test SAC attempts to find these ideal gains throughout simulation. The test SAC will begin with initial gains

$$\mathbf{K}_0 = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix} \quad (3.30)$$

The SAC described here, called the “default SAC design”, is used in future discussions to outline different behaviours of the SAC architecture.

The command to the default SAC is composed of four step commands, each separated by 10 seconds, and beginning with an initial step. The final simulation is $t_f = 40$ seconds long, and its Simulink® diagram is demonstrated in Fig. 3.7.

Linear Output and Maximum Acceptable Gains

When adaptation is omitted, the SAC output equation is a form of linear control law.

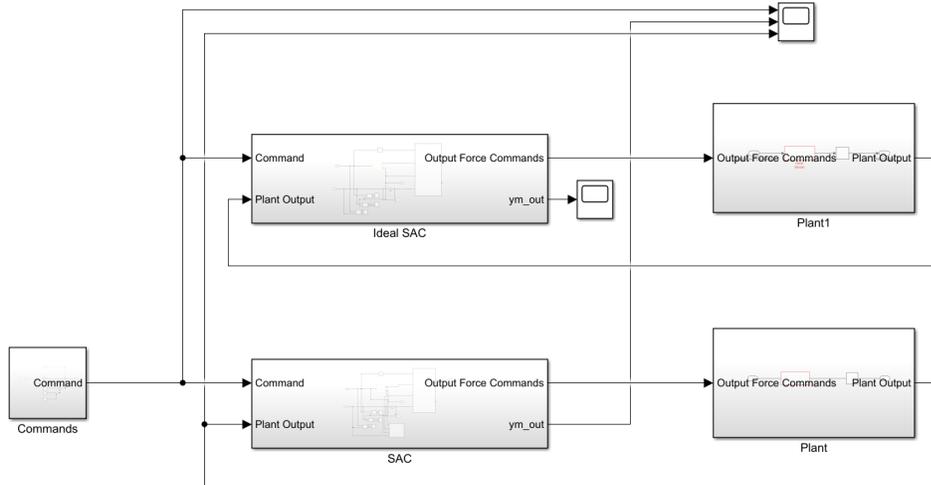


Figure 3.7: Default SAC Simulation

$$\mathbf{u}_c(t) = \mathbf{K}_e \mathbf{e}_y(t) + \mathbf{K}_x \mathbf{x}_m(t) + \mathbf{K}_u \mathbf{u}_m(t) \quad (3.31)$$

Designers can use linear and nonlinear design tools to determine performance of their system for different outputs of SAC gains. Designers could, if desired, make a linear controller with this architecture, and find the set of gains that deliver good performance. Instead, it is more likely that a linear controller has already been designed for their system, and the gains of that linear controller inform the magnitudes of SAC output gains that deliver acceptable performance for a given system.

For example, the presence of noise in system measurements typically affects the magnitude of feedback gain that can be used before noise appears in the command response. Pre-existing linear controller formulations provide an estimate of the magnitude of feedback gain that is acceptable for the noise in that system. If a set of feedforward gains exists for the system, the signals used in those feedforward gains should correspond to the states in the ideal model. Similarly, the magnitude of pre-existing feedforward gains informs acceptable magnitudes of SAC feedforward gains. If the ideal gain matching conditions Eqs. (2.115) through (2.118) have a solution for the system, these can be used as a baseline linear controller for the system.

The existence of linear analysis techniques allows a designer to determine what values of SAC gains are acceptable for their system, as well as allowing them to

determine an acceptable upper-bound. Acceptable upper bounds for the gains will be leveraged to determine sets of appropriate adaptation parameters for the system.

Call

$$\mathbf{K}_m = [\mathbf{K}_{max,e}, \mathbf{K}_{max,x}, \mathbf{K}_{max,u}] \quad (3.32)$$

the matrix of acceptable maximum gains for Eq. (3.31). The values of these maximum gains are chosen by the designer, and can be determined through experiment, analysis or according to design specifications. The choice of these maximum values is a design decision, and will vary with the system in consideration. The maximum gains \mathbf{K}_m represent the highest linear gains that a designer would implement in their system, and will be leveraged in future sections to determine how the design parameters $\mathbf{\Gamma}_I$, $\mathbf{\Gamma}_P$ and $\boldsymbol{\sigma}$ should be chosen to ensure that adaptation stays beneath this value.

The default SAC presented in Sec. 3.3.3 is directly implemented and the values of its parameters are chosen to ensure stability. Note that the default plant is continuous time ASPR, since it meets the order and phase requirements, however the discrete time system corresponding to the continuous time plant is *not* ASPR. A stability analysis of the discrete system yields that the system becomes unstable for feedback gains above a value of

$$K_{max,e} = 200 \quad (3.33)$$

and as such cannot be ASPR. The implementation of a continuous time ASPR plant into a non-ASPR discrete plant occurs frequently. However, this naive implementation is useful for discussions of adaptation parameter selection, where the maximum feedback gain in the example system may correspond to a gain limitation that a designer wishes to avoid in their SAC implementation. At present it is useful to show that proper selection of design parameters can avoid instability in the discrete system even though it is not ASPR for all values of feedback gain.

For now, a set of maximum values for the default SAC are chosen by the designer from the available knowledge of the system. The maximum allowable SAC gains for the default system are chosen as

$$\mathbf{K}_m = [0.75\mathbf{K}_{max,e}, 10\mathbf{K}_x^*, 10\mathbf{K}_u^* + 1] = [150, 10, 10, 1]; \quad (3.34)$$

Since the ideal gains are known for the feedforward terms, a multiple of those terms are used for the maximum gains. The ideal gain for the command input is 0, and an additional unit is added to create a positive maximum gain value. The maximum feedback gain is decreased to 75% of the unstable value to ensure that adaptation parameter estimates do not approach the maximum values of the system.

Integral Adaptation Parameter Selection

In order to choose good values of integral adaptation parameters, effort is made to understand how the magnitude of the output SAC gains are affected by the choice of integral adaptation parameter. By choosing appropriate values of $\mathbf{\Gamma}_I$ it can be ensured that the output SAC gains do not increase outside of the ASPR region.

A design heuristic will be developed here that ensures that the designer determined maximum values of the SAC gains are not exceeded during operation.

The value of the SAC integral gain at any time $t \in \mathbb{R}^+$ follows the equation

$$\mathbf{K}_I(t) = \int_0^t \mathbf{e}_y(\tau)\mathbf{r}^T(\tau)\mathbf{\Gamma}_I d\tau + \mathbf{K}_0 \quad (3.35)$$

for an initial gain \mathbf{K}_0 . Notice that the adaptation parameter is constant throughout the integration, and as such can be extracted from the integral. The integral of the tracking error with the reference signal, effectively the contribution of the system response to the gain adaptation, is shortened to the variable

$$\mathbf{K}_F = \int_0^t \mathbf{e}_y(\tau)\mathbf{r}^T(\tau)d\tau = \int_0^t \mathbf{I}_K d\tau \quad (3.36)$$

Where \mathbf{K}_F is called the *forcing gain*, and \mathbf{I}_K the *gain impulse*, due to their resemblance to impulses and forces in classical mechanics.

If a gain impulse or forcing gain can be determined, then the remaining equation for the maximum acceptable adaptation parameter becomes

$$\mathbf{K}_I = \mathbf{K}_F \mathbf{\Gamma}_I + \mathbf{K}_0 \quad (3.37)$$

$$\mathbf{K}_m \geq \mathbf{K}_F \mathbf{\Gamma}_I + \mathbf{K}_0 \quad (3.38)$$

At equality, Eq. (3.38) is a matrix equation for $\mathbf{\Gamma}_I$. The gains \mathbf{K}_m and \mathbf{K}_I are of dimension $m \times (2m + q)$ and as such Eq. (3.38) is a set of $m \times (2m + q)$ equations. If $\mathbf{\Gamma}_I$ is taken to be a positive semi-definite diagonal matrix of unknowns, then $\mathbf{\Gamma}_I$ must be a square matrix of $(2m + q)$ variables. Equation (3.38) thus corresponds to a set of $m \times (2m + q)$ equations and $(2m + q)$ unknowns. In the SISO case this is a set of $(2 + q)$ equations and $(2 + q)$ unknowns, which can always be solved. If $\mathbf{\Gamma}_I$ is instead a symmetric positive semi-definite matrix, the number of unknowns can be increased to $((2m + q) \times (2m + q + 1))/2$, which is always greater than the $m \times (2m + q)$ outputs. Further analysis should be undertaken for adaptation parameter estimates in MIMO systems. For now only the SISO case is considered.

The estimate for the magnitude of the adaptation parameter through the solutions of Eq. (3.38) for the SISO case is called $\hat{\mathbf{\Gamma}}_I \in \mathbb{R}^{(2m+q) \times (2m+q)}$.

Several increasingly accurate estimations of the gain impulse will be used to determine what values of the integral adaptation parameters $\mathbf{\Gamma}_I$ ensure that \mathbf{K}_I remains smaller in magnitude, component-wise, than the matrix of maximum acceptable gains \mathbf{K}_m .

Bounding the Gain Impulse

To ensure that $\hat{\mathbf{\Gamma}}_I$ estimates do not result in responses that exceed \mathbf{K}_m , estimates of the forcing gain must be larger than the true forcing gain. To ensure that the maximum gains are not exceeded, the magnitude of the forcing gain estimates will be taken to be larger than the forcing gain that is expected during operations. To do so, all adaptation are assumed to work towards increasing the gains. The real forcing gain must be bounded, and the estimate taken as the higher bound.

First, it is noted that the integral of a signal must be less than or equal to the integral of the absolute value of that signal

$$\int_0^t \mathbf{e}_y(\tau) \mathbf{r}^T(\tau) d\tau \leq \int_0^t |\mathbf{e}_y(\tau)| |\mathbf{r}^T(\tau)| d\tau \quad (3.39)$$

The forcing gain estimate is defined to be

$$\hat{\mathbf{K}}_F(t) = \int_0^t |\hat{\mathbf{e}}_y(\tau)| |\hat{\mathbf{r}}^T(\tau)| d\tau \geq \int_0^t |\mathbf{e}_y(\tau)| |\mathbf{r}^T(\tau)| d\tau \quad (3.40)$$

for estimated tracking error $\hat{\mathbf{e}}_y$ and estimated reference signal $\hat{\mathbf{r}}$. By ensuring the forcing gain estimate is greater than or equal to the true forcing gain, the maximum gain reached in operation by the adaptation parameter estimate is guaranteed to be lower than the designed maximum gain value.

Uninformed Estimate

First, an initial uninformed estimate of the gain impulse is made, from which initial parameters can be selected and used in future estimates.

The worst-case system response is assumed over the course of a representative simulation to determine the worst case adaptation parameters. It should be noted that the SAC control response is magnitude dependant, and as such the largest expected commands and errors should be used for this estimate. The length of the simulation will affect the magnitude of the adaptation parameter estimate, however longer simulations will simply scale down the adaptation parameter. Future estimates will greatly decrease the effect of the simulation time on parameter estimation. The incorporation of σ in future estimates will further mitigate the effect of the representative simulation on the choice of suitable adaptation parameters.

Assuming that a unit step command is the largest command that will be sent to the system. The largest gain impulse that can occur is if no control occurs at all and all elements of the reference signal are at their largest magnitude for the entirety of the simulation time. The maximum error is the value of the input command, and the maximum value of the states can be determined from the response of the ideal model. For the default SAC, the maximum value of each signal is

$$\hat{\mathbf{e}}_y = \max(|\mathbf{e}_y(t)|) = |u_c| = 1$$

$$\hat{\mathbf{r}} = \begin{bmatrix} \max(|\mathbf{e}_y(t)|) \\ \max(|\mathbf{x}_m(t)|) \\ \max(|\mathbf{u}_m(t)|) \end{bmatrix} = \begin{bmatrix} 1 \\ 0.37 \\ 1 \\ 1 \end{bmatrix}$$

The bounded forcing gain estimate is therefore

$$\hat{\mathbf{K}}_F = \int_0^t |\hat{\mathbf{e}}_y(\tau)| |\hat{\mathbf{r}}^T(\tau)| d\tau \quad (3.41)$$

$$\hat{\mathbf{K}}_F = t_f \hat{\mathbf{e}}_y \hat{\mathbf{r}}^T \quad (3.42)$$

which, when used in conjunction with Eq. (3.38) yields an adaptation parameter estimate of

$$\hat{\mathbf{\Gamma}}_I = \begin{bmatrix} 1.5939 & 0 & 0 & 0 \\ 0 & 0.0295 & 0 & 0 \\ 0 & 0 & 0.0797 & 0 \\ 0 & 0 & 0 & 0.008 \end{bmatrix} \quad (3.43)$$

The system output and gain response for this parameter estimate are shown in Fig. 3.8.

The system response is somewhat slow, however adaptation is still occurring. The significant overestimate of the gain impulse has also ensured that the actual gain response stays very small. Over the simulation, the feedback gain stays well below the maximum of 150, and none of the other gains increase above 0.5. More refined estimates of the forcing gain will increase the ultimate gain and improve on the performance seen with this worst-case estimate. The gain impulse estimate and SISO adaptation parameter equation has been able to determine a set of parameters that stay well below the maximum value in simulation.

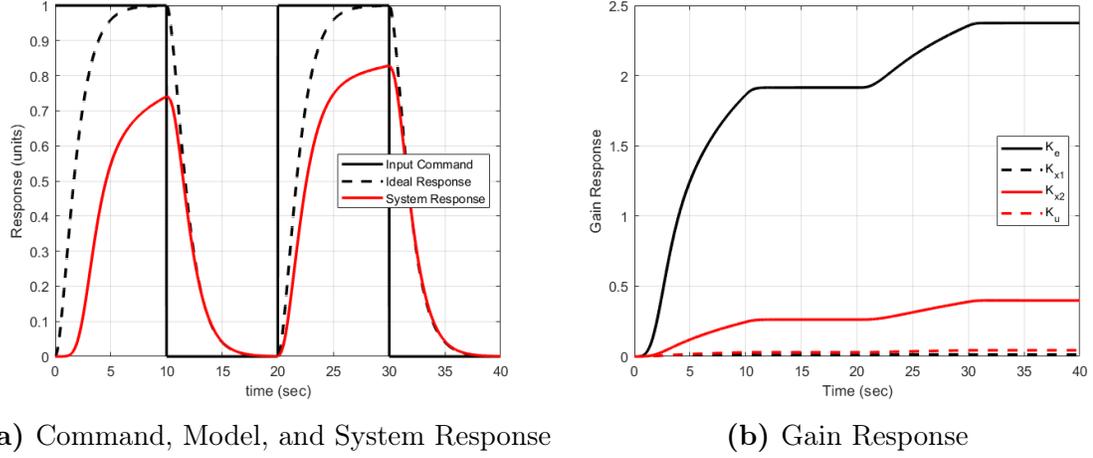


Figure 3.8: Uninformed Estimate Results

Maximum Reference Estimate

If a simulation is available to verify the response of the system under SAC architecture, a slightly less conservative estimate can be used.

The values of $\hat{\mathbf{e}}_y$ and $\hat{\mathbf{r}}$ in the previous section are large overestimates of the true maximum values of these parameters. If a simulation is available, then the magnitudes of these signals during simulation can be used to update the gain impulse estimate. We now call $\mathbf{e}_y(t, \mathbf{\Gamma}_I)$ the tracking error during a simulation for a SAC using parameters $\mathbf{\Gamma}_I$, and $\mathbf{r}(t, \mathbf{\Gamma}_I)$ the reference signal under the same parameters. An estimate for the new maximum signals can be found as

$$\hat{\mathbf{e}}_y(\mathbf{\Gamma}_I) = \max(|\mathbf{e}_y(t, \mathbf{\Gamma}_I)|) \quad (3.44)$$

$$\hat{\mathbf{r}}(\mathbf{\Gamma}_I) = \begin{bmatrix} \max(|\mathbf{e}_y(t, \mathbf{\Gamma}_I)|) \\ \max(|\mathbf{x}_m(t, \mathbf{\Gamma}_I)|) \\ \max(|\mathbf{u}_m(t, \mathbf{\Gamma}_I)|) \end{bmatrix} \quad (3.45)$$

which yields a new forcing gain $\hat{\mathbf{K}}_f$ through

$$\hat{\mathbf{K}}_F = t_f \hat{\mathbf{e}}(\mathbf{\Gamma}_I) \hat{\mathbf{r}}^T(\mathbf{\Gamma}_I) \quad (3.46)$$

A simulation of the new SAC parameters can then be done, and the maximum value estimate repeated until satisfactory convergence in the forcing gain has been achieved.

Stability of the maximum value estimate over successive simulations and reevaluations of $\hat{\mathbf{K}}_F$ is likely, however difficult to prove.

Each estimate of $\hat{\mathbf{\Gamma}}_I$ is guaranteed to have a set of final gains lower than \mathbf{K}_m , by nature of forcing the estimated forcing gain to be larger than the true forcing gain. If the simulated system is stable for all gain values lower than \mathbf{K}_m , then the use of stable maximum gains \mathbf{K}_m implies that adaptation parameter estimates are also stable. Since the time response of a SAC design is still unknown, it is unlikely that a closed-form proof of the existence of a stable set of parameters for this method can be determined; if a method of proving the stability of this method existed, then information on the magnitude of $\mathbf{e}_y(t, \mathbf{\Gamma}_I)$ could be determined without a simulation, and the form of the SAC response for changing $\mathbf{\Gamma}_I$ would be known. Very little can be said about the stability of this estimation technique without additional methods to estimate the nonlinear system response.

However, the inverse relationship between the forcing gain and adaptation parameter estimate work antagonistically to help the parameter estimate converge on a result. If the forcing gain is increased by the adaptation parameter estimate, then the updated adaptation parameter estimate must decrease. If the gain impulse is decreased by the adaptation parameter estimate, then the updated adaptation parameter estimate will increase. The relationship between the gain impulse and the adaptation parameter estimate therefore constitutes a negative feedback response for the adaptation parameter estimate.

The maximum value estimate is applied to the default system and repeated for thirty iterations to ensure convergence. The final controller determined by the estimate is shown in Fig. 3.9, alongside its gain response.

The final response determined by this estimate is still somewhat slower to converge than a manual design might achieve, however each of the gains continue to stay well below their maximum values. The evolution of the forcing gain and adaptation parameter estimates over subsequent iterations are shown in Fig. 3.10. The estimate

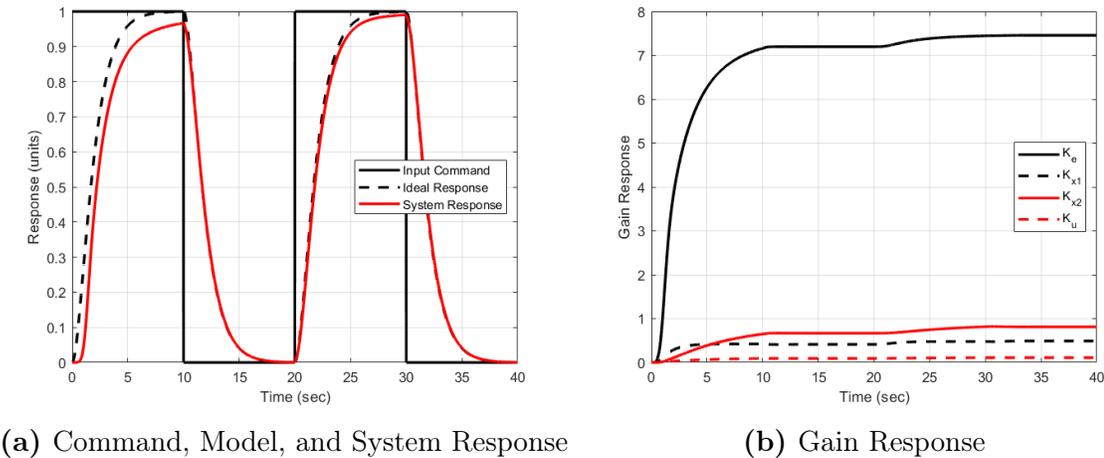


Figure 3.9: Maximum Value Estimate Results

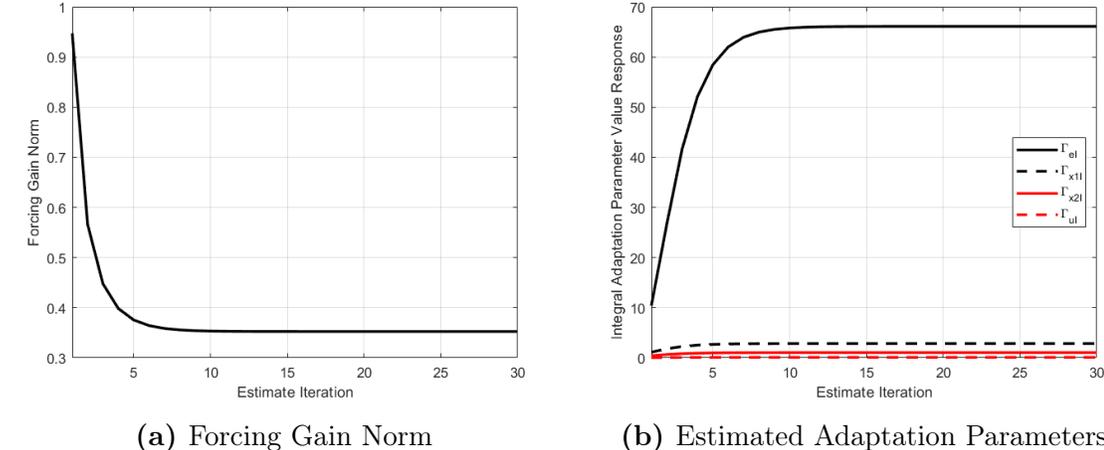


Figure 3.10: Maximum Value Estimate Iteration Results

converges after ten iterations. The final adaptation parameter estimate given by the maximum value technique is found to be

$$\hat{\Gamma}_I = \begin{bmatrix} 66.1 & 0 & 0 & 0 \\ 0 & 2.85 & 0 & 0 \\ 0 & 0 & 1.05 & 0 \\ 0 & 0 & 0 & 0.105 \end{bmatrix} \tag{3.47}$$

Simulated Forcing Gain

If an accurate simulation is available, a reasonable estimate of the forcing may be found by simply simulating the system and determining the bounded gain impulse.

After completing a simulation, the forcing gain estimate for that simulation is found through direct evaluation of

$$\hat{\mathbf{K}}_F = \int_0^t |\mathbf{e}_y(\tau, \mathbf{\Gamma}_I)| |\mathbf{r}^T(\tau, \mathbf{\Gamma}_I)| d\tau \quad (3.48)$$

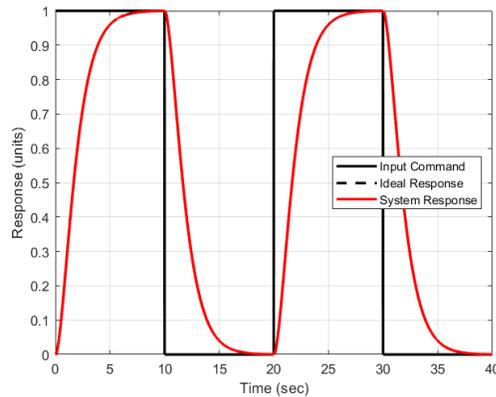
The gain impulse estimate is then used to find a new set of adaptation parameters through Eq. (3.38). This process can then be iterated to determine a new gain impulse estimate and maximum adaptation parameters.

Similar to the maximum value estimate, the convergence of this estimate through multiple iterations cannot be easily proven, however the nature of the relationship between the gain impulse estimate and the adaptation parameter estimate works to close in on a stable result.

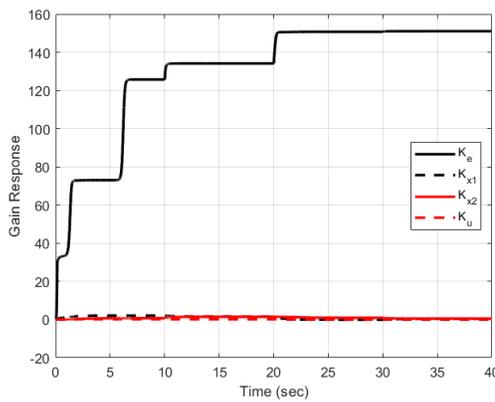
The simulated gain impulse estimate is demonstrated on the default system, and repeated for thirty iterations. Figure 3.11 shows the final system response and the final gain response. The system response converges quickly on the ideal model, with the signals for the ideal model output and the system response overlapping in Fig. 3.11. The model state and model command gains stay small over the course of the simulation. A zoom in on the final adapted gain values is shown in Fig. 3.11c. Due to the structure of the simulated forcing gain estimate, the final feedback error response approaches the maximum value $\mathbf{K}_{max,e}$ exactly. Since the forcing gain estimate is equal to the true forcing gain for the term for the feedback error, that is to say

$$|\mathbf{e}_y(t)| |\mathbf{e}_y^T(t)| = \mathbf{e}_y(t) \mathbf{e}_y^T(t) \quad (3.49)$$

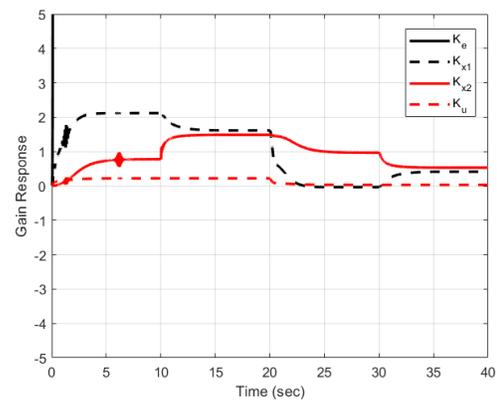
it is clear that the simulated forcing gain estimate must cause the feedback error to approach the maximum allowable value at the end of the simulation. Figure 3.12 shows the behaviour of the estimate over thirty iterations. The forcing gain norm decreases quickly over the first five iterations. The adaptation parameter estimates similarly do not stabilize until the 20th iteration, with small variations continuing



(a) Command, Model, and System Response



(b) Gain Response



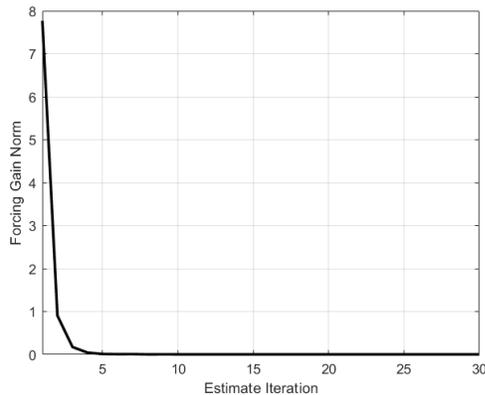
(c) Gain Response, Cropped

Figure 3.11: Simulated Forcing Gain Estimate Results

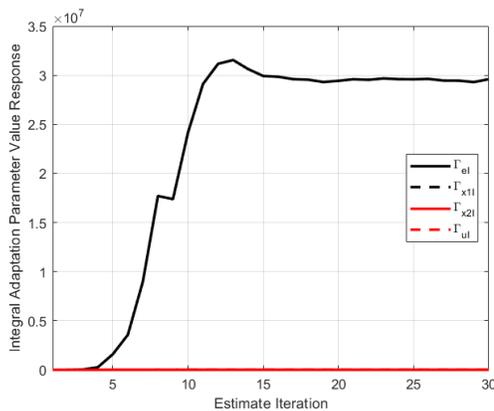
throughout. The small value of the forcing gain results in large variations in the final adaptation parameter between iterations. A zoom in of the adaptation parameter response over subsequent iterations is demonstrated in Fig. 3.12c, where it can be seen that all of the adaptation parameter estimates respond similarly to the feedback error adaptation estimate.

The simulated gain impulse estimate is able to drastically increase the system response when compared with similar estimates. Furthermore, the estimate is able to demonstrate that stability can be maintained for adaptation parameters on the order of 1×10^4 for this system.

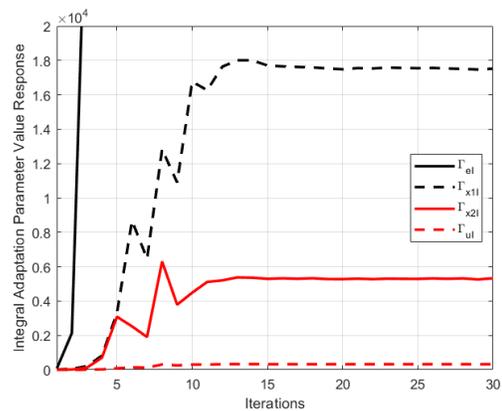
The final adaptation parameters determined by the estimate are found to be



(a) Forcing Gain Norm



(b) Adaptation Parameter Estimates



(c) Cropped Adaptation Parameter Estimates

Figure 3.12: Simulated Forcing Gain Estimate Iteration Results

$$\hat{\Gamma}_I = \begin{bmatrix} 2.1 \times 10^7 & 0 & 0 & 0 \\ 0 & 1.7 \times 10^4 & 0 & 0 \\ 0 & 0 & 5.3 \times 10^3 & 0 \\ 0 & 0 & 0 & 325 \end{bmatrix} \quad (3.50)$$

3.3.4 Steady State Gains Under Noise and Signal Covariance

Knowledge of the previous forcing gain estimates, and system noise can be used to determine appropriate values of the forgetting factor σ for the integral adaptation.

The forgetting factor can be applied to the gain update in Eq. (3.38) to yield a new set of conditions

$$\mathbf{K}_m \geq \int_0^t (\mathbf{e}_y(\tau)\mathbf{r}^T(\tau)\mathbf{\Gamma}_I - \mathbf{K}_I(\tau)\boldsymbol{\sigma}) d\tau + \mathbf{K}_0 \quad (3.51)$$

The equation for the feedback gain derivative is similarly useful

$$\dot{\mathbf{K}}_I = \mathbf{e}_y(t)\mathbf{r}^T(t)\mathbf{\Gamma}_I - \mathbf{K}_I(t)\boldsymbol{\sigma} \quad (3.52)$$

Both of these equations can be leveraged to determine values of $\boldsymbol{\sigma}$ that ensure that \mathbf{K}_m is not exceeded. For example, if a gain impulse estimate $\hat{\mathbf{I}}_K$ is certain to be larger than the gain impulse experienced in normal operations, then all that is required is to find the degradation parameter that ensures no gain adaptation for that impulse. With the inclusion of an additional safety factor $\varsigma > 1$, the condition becomes

$$\hat{\mathbf{I}}_K\mathbf{\Gamma}_I = \mathbf{K}_m\boldsymbol{\sigma}\varsigma \quad (3.53)$$

which has similar properties to the integral adaptation estimate. The estimate of the gain impulse can be found through any of the techniques presented in the previous section. Furthermore, if degradation is only necessary to negate a forcing gain over some period of time Δt , then setting the integral in Eq. (3.52) to zero and solving over Δt gives an equation for $\boldsymbol{\sigma}$. For starting and ending gains \mathbf{K}_0

$$\mathbf{K}_0 = \int_0^{\Delta t} (\mathbf{e}_y(\tau)\mathbf{r}^T(\tau)\mathbf{\Gamma}_I - \mathbf{K}_I(\tau)\boldsymbol{\sigma}) d\tau + \mathbf{K}_0 \quad (3.54)$$

$$\int_0^{\Delta t} \mathbf{e}_y(\tau)\mathbf{r}^T(\tau)\mathbf{\Gamma}_I d\tau = \int_0^{\Delta t} \mathbf{K}_I(\tau)\boldsymbol{\sigma} d\tau \quad (3.55)$$

If an estimate for the forcing gain can be made, and if values of the integral adaptations have already been chosen, then an approximation for the value of sigma, called $\hat{\boldsymbol{\sigma}}$, that cancels the forcing gain over time Δt is given by

$$\hat{\mathbf{K}}_F\mathbf{\Gamma}_I = \Delta t\mathbf{K}_0\hat{\boldsymbol{\sigma}}\varsigma \quad (3.56)$$

Notice as well that the safety factor ς is the same as decreasing the maximum

gain that the degradation factor affects. Similar to the integral adaptation parameter estimate, $\hat{\boldsymbol{\sigma}}$ is of size $(2m+q) \times (2m+q)$, and has only $(2m+q)$ parameters if diagonal. The degradation parameter estimate $\hat{\boldsymbol{\sigma}}$ is always uniquely defined if diagonal and in the SISO case.

Knowledge of the system noise can also be leveraged by the designer to specify a steady state value for the tracking error gain \mathbf{K}_{ss} through the use of the degradation parameter $\boldsymbol{\sigma}$.

Consider that the measured signal \mathbf{y}_p includes uncorrelated zero-mean noise with variance

$$\text{Var}(\mathbf{y}_p) = \mathbf{E}[(\mathbf{y}_p - \mathbf{y}_{avg})^2] = \boldsymbol{\delta}^2 \quad (3.57)$$

where the random variable following the variance $\boldsymbol{\delta}$ is called \mathbf{w} . Assuming either a steady state system or a converged and zero error system, the tracking error gain adaptation due to the noise is found to be

$$\mathbf{e}_n(t) = \mathbf{e}_y(t) + \mathbf{w}(t) \approx \mathbf{w}(t) \quad (3.58)$$

$$\mathbf{r}_n(t) = \left[\mathbf{e}_n^T(t) \quad \mathbf{x}_m^T(t) \quad \mathbf{u}_m^T(t) \right]^T \quad (3.59)$$

$$\dot{\mathbf{K}}_{ss}(t) = \mathbf{0} = \mathbf{e}_n(t) \mathbf{r}_n^T(t) \boldsymbol{\Gamma}_I - \mathbf{K}_{ss} \hat{\boldsymbol{\sigma}} \quad (3.60)$$

$$\approx \mathbf{w}(t) \mathbf{w}^T(t) \boldsymbol{\Gamma}_I - \mathbf{K}_{ss} \hat{\boldsymbol{\sigma}} \quad (3.61)$$

$$\mathbf{K}_{ss} \hat{\boldsymbol{\sigma}} = \boldsymbol{\delta}^2 \boldsymbol{\Gamma}_I \quad (3.62)$$

allowing the designer to choose a value of $\boldsymbol{\sigma}$ that corresponds to the tracking error gain that they would like to achieve at steady state. It quickly becomes apparent that unless there is correlation between the noise on the tracking error and the other reference signals that no additional adaptation will be present due to noise on either the tracking error or the other reference signals. If correlation exists between the system measurement noise and any of the signals, steady state values of those gains can be similarly calculated to the steady state model gain \mathbf{K}_{ss} in Eq. (3.62).

Noise of variance $\delta = 1$ was applied to the default system, and the effects observed.

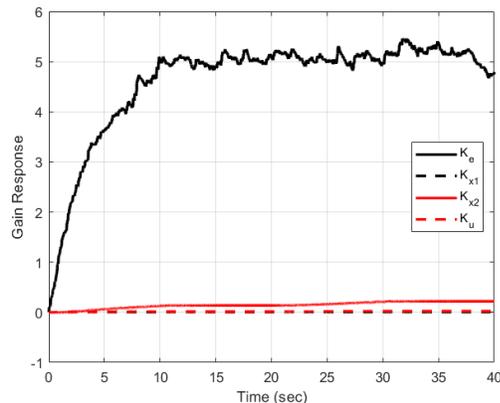


Figure 3.13: Gains Under Noise and Appropriate σ Selection

For a desired steady-state feedback gain of $\mathbf{K}_{ss} = 5$ with the adaptation parameters

$$\mathbf{\Gamma}_I = \begin{bmatrix} 1.5939 & 0 & 0 & 0 \\ 0 & 0.0295 & 0 & 0 \\ 0 & 0 & 0.0797 & 0 \\ 0 & 0 & 0 & 0.008 \end{bmatrix} \quad (3.63)$$

Eq. (3.62) suggests a value of $\sigma_e = 0.2391$. The gain impulse imparted by the noise in this system is also larger than any other major sources of forcing gain or gain impulse, and as such the degradation suggested by Eq. (3.62) is taken to be the parameter during implementation. The resultant gain response under noise is demonstrated in Fig. 3.13. It can be seen that the estimate of the degradation parameter does indeed cause the feedback gain to converge on the designed steady-state value.

Notice that the assumption in Eq. (3.58) does not hold if the steady state feedback gain chosen amplifies noise sufficiently to introduce additional tracking error due to noise feedback. If noise is being sufficiently amplified the assumption in Eq. (3.58) will underestimate the expectation of $\mathbf{e}_n \mathbf{e}_n^T$ and therefore underestimate the steady state gain.

3.3.5 Proportional Adaptation as Quadratic or Boundary Control

The integral adaptation and the proportional adaptation equations have several similarities. Their structure is almost identical, save for the presence of a derivative between the two. The presence of design tools for the integral adaptations might suggest that similar design tools are present for the proportional adaptations.

The use of the proportional adaptations Γ_{eP} , Γ_{xP} , and Γ_{uP} can also be simply leveraged in two other ways. Assuming a SAC with only proportional adaptations yields a control equation of

$$\mathbf{u}_p(t) = \mathbf{e}_y(t)\mathbf{e}_y^T(t)\Gamma_{eP}\mathbf{e}_y(t) + \mathbf{e}_y(t)\mathbf{x}_m^T(t)\Gamma_{xP}\mathbf{x}_m(t) + \mathbf{e}_y(t)\mathbf{u}_m^T(t)\Gamma_{uP}(t)\mathbf{u}_m \quad (3.64)$$

$$\mathbf{u}_p(t) = \mathbf{e}_y(t)\mathbf{r}^T(t)\Gamma_P\mathbf{r}(t) \quad (3.65)$$

which resembles a negative feedback controller with adaptive gains of $\mathbf{r}^T\Gamma_P\mathbf{r}$, which must be quadratic due to the choice of a positive definite Γ_P . In this sense, Eq. (3.64) represents a controller that follows a quadratic activation in the signal \mathbf{r} , and additional linear activation in the signal \mathbf{e}_y . When considering the full controller, the output \mathbf{u}_p increases cubically with \mathbf{e}_y . The structure of this control law can be leveraged.

By choosing appropriate values of the proportional adaptations Γ_P , a design gain at a predetermined tracking error can be chosen. For tracking errors above the design value, gains will be larger and the proportional adaptations will force the controller back towards zero tracking error. For tracking errors smaller than the design value, smaller proportional adaptations will have less impact, and integral adaptation can take control in these regions.

The designer simply determines appropriate solutions to the equation

$$\mathbf{e}_{y,b}\mathbf{r}_b^T\Gamma_P\mathbf{r}_b = \mathbf{u}_{p,b} \quad (3.66)$$

such that \mathbf{r}_b and $\mathbf{e}_{y,b}$ are the reference signal and tracking error that result in a given control input at the boundary $\mathbf{u}_{p,b}$. Through appropriate selection, proportional adaptation can be used to ensure the tracking error does not increase beyond a certain

value, creating a boundary within which the integral adaptation is free to control the system, and beyond which proportional adaptation ensures the system is able to maintain stability.

An example of the use of proportional adaptation as boundary control is briefly highlighted with the default system. The default system is given a set of poorly chosen, yet stable integral adaptations of

$$\mathbf{\Gamma}_I = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 \\ 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (3.67)$$

with no degradation σ or proportional control $\mathbf{\Gamma}_P$. The results of the control are seen in Fig. 3.14a, where significant bursting is present in the system due to inappropriate selection of the state adaptations. The magnitude of the bursts can be decreased by direct application of Eq. (3.66). Boundary conditions of

$$\mathbf{e}_{y,b} = 0.1 \quad (3.68)$$

$$\mathbf{u}_{p,b} = 3 \quad (3.69)$$

$$\mathbf{r}_b = \begin{bmatrix} \mathbf{e}_{y,b} & 0 & 0 & 0 \end{bmatrix}^T \quad (3.70)$$

yield a proportional adaptation of $\Gamma_{eP} = 3000$. Note that different selections of $\mathbf{e}_{y,b}$ and $\mathbf{u}_{p,b}$ will scale the quadratic activation of the gain. Since the gain activation is quadratic, solutions of Eq. (3.66) only guarantee accuracy of the force output at the boundary condition, and does not suggest that the system will not exceed that boundary during operation, or that a sufficient restoring force will not occur within the boundary condition. Nevertheless, the ability to choose a force at a given plant condition is useful for system design. The effect of including the proportional adaptation $\Gamma_{eP} = 3000$ on the default system is illustrated in Fig. 3.14b, where the choice of proportional adaptation has successfully limited the bursting to occur between ± 0.1 units of the ideal model response.

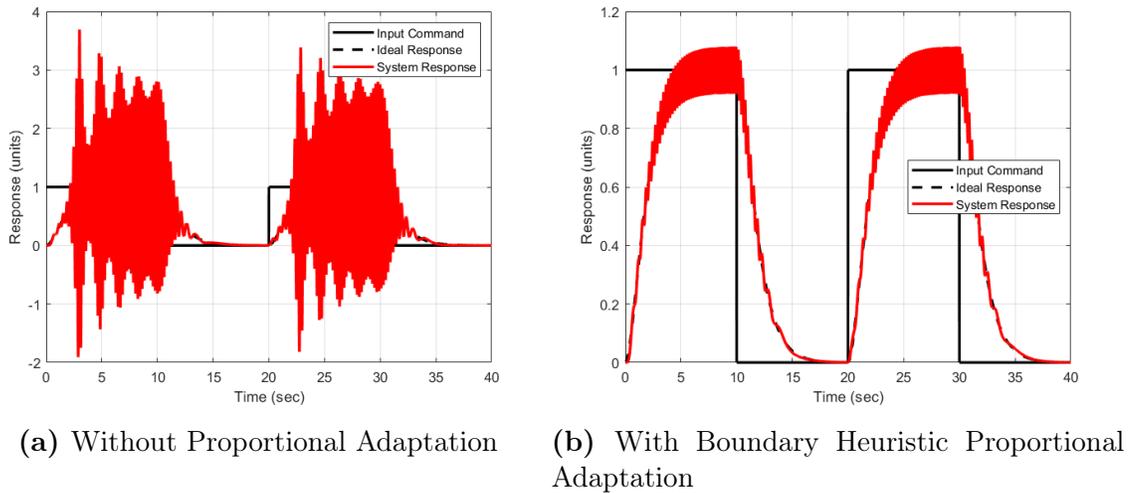


Figure 3.14: Poorly Tuned System With and Without Proportional Adaptation

This concept can also be used to understand how a system will behave under only proportional adaptations.

Consider the default system under only proportional adaptation, solving for the equilibrium of the system in the SISO case yields:

$$\mathbf{u}_p = \mathbf{e}_y \mathbf{r}^T \mathbf{\Gamma}_P \mathbf{r} \quad (3.71)$$

$$\mathbf{e}_y = \mathbf{y}_m - \mathbf{y}_p \quad (3.72)$$

Since the plant $P(s)$ is a unitary gain system, under steady-state input the output value is stable to the input value. At equilibrium the system response becomes

$$\mathbf{e}_y = \mathbf{y}_m - \mathbf{e}_y \mathbf{r}^T \mathbf{\Gamma}_P \mathbf{r} \quad (3.73)$$

rearranging for the tracking error yields

$$\mathbf{e}_y(\mathbf{I}_m + \mathbf{r}^T \mathbf{\Gamma}_P \mathbf{r}) = \mathbf{y}_m \quad (3.74)$$

$$\mathbf{e}_y = \mathbf{y}_m (\mathbf{I}_m + \mathbf{r}^T \mathbf{\Gamma}_P \mathbf{r})^{-1} \quad (3.75)$$

The equilibrium tracking error, therefore, is related to a cubic polynomial of the ideal model output for the default system.

The default system is given proportional adaptation of magnitude

$$\mathbf{\Gamma}_{eP} = \begin{bmatrix} 1000 \end{bmatrix} \quad (3.76)$$

$$\mathbf{\Gamma}_{xP} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad (3.77)$$

$$\mathbf{\Gamma}_{uP} = \begin{bmatrix} 0 \end{bmatrix} \quad (3.78)$$

Solving Eq. (3.75) for the steady-state tracking error under unit step commands for the given proportional adaptations becomes

$$e_y = 1(1 + e_y \mathbf{\Gamma}_{eP} e_y)^{-1} \quad (3.79)$$

$$0 = e_y^3 \mathbf{\Gamma}_{eP} + e_y - 1 \quad (3.80)$$

which is a cubic equation with solution of $e_y = 0.0967$ (and two imaginary roots). The final system response should therefore reach a value of 0.9033 for a unit step command. The results of the system response and tracking error are shown in Fig. 3.15.

The system response in Fig. 3.15a does meet the steady state tracking value suggested in Eq. (3.80).

By understanding the proportional adaptation as a form of boundary control or a quadratic gain control, designers can leverage their choice of proportional adaptation to ensure stability, keep the benefits of integral adaptation, and make meaningful predictions about the outputs of proportional adaptation.

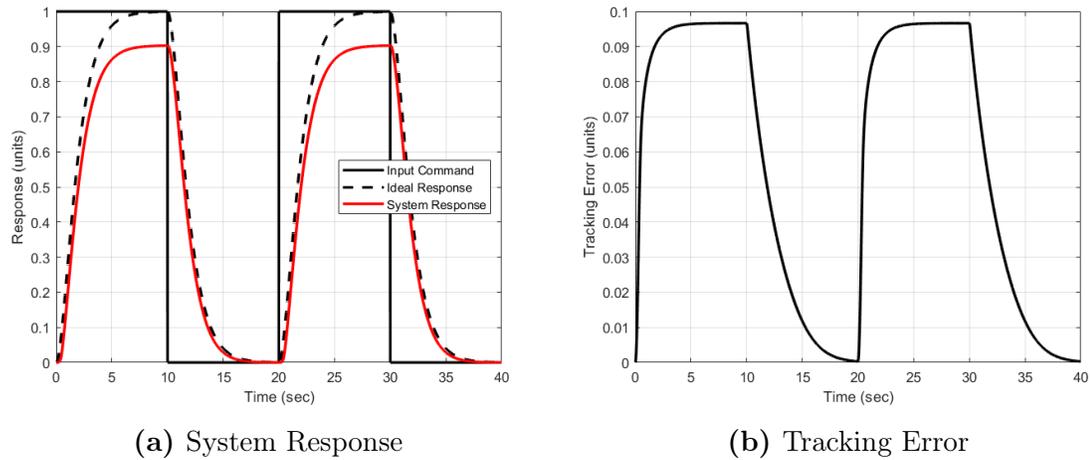


Figure 3.15: Default System Under Only Proportional Adaptation

3.4 SAC Disturbance Compensation in Feedforward Parallelized Plants

The design of SAC allows for quick numerical solutions to questions of magnitude. By varying the magnitude of the gains associated with signals that affect the error SAC is able to meaningfully determine the relationships between the tracking error and the signals that affect it.

Prabhakar et al. [29] provides a setup for a disturbance compensation component for the standard SAC architecture. The disturbance compensating SAC component was presented in Sec. 2.5.4. Prabhakar et al. augment their system to be ASPR using sensor blending, and shows that a SAC with the additional disturbance compensation component is able to control an UAV in simulation when presented with a linear disturbance, specifically an oscillation with known frequency content.

Prabhakar’s results are herein extended in three notable ways. Firstly, the effects of linear disturbance adaptive compensation are considered for a system with feedforward parallelization, phase uncertainty of the disturbance is included, and the disturbance compensation is experimentally verified in Carleton’s SRCL SPOT, the results of which are discussed in Chapter 4. For now, the theory behind implementing linear disturbance compensation in a feedforward parallelized system will be summarized.

Consider a disturbance that is characterized by the linear disturbance generator

model

$$\dot{\mathbf{z}}_d(t) = \mathbf{F}_d \mathbf{z}_d(t) \quad (3.81)$$

$$\mathbf{u}_d(t) = \mathbf{\Theta}_d \mathbf{z}_d(t) \quad (3.82)$$

Notably, due to the solutions to linear systems presented previously, it is known that the only dynamic behaviour such a system can provide without an external input is of the form

$$\mathbf{u}_d(t) = \mathbf{\Theta}_d e^{\mathbf{F}_d t} \quad (3.83)$$

which means that the solutions must either be exponential or sinusoidal in nature. The original proposal considered that the disturbance model would be well known, that is to say that the disturbance model $\hat{\mathbf{F}}_d$, $\hat{\mathbf{\Theta}}_d$, and $\hat{\mathbf{z}}_d$ approximate \mathbf{F}_d , $\mathbf{\Theta}_d$, and \mathbf{z}_d , respectively. Luckily, the requirements for frequency information can be relaxed somewhat due to trigonometric identities and the ability for SAC to quickly and accurately determine relevant signal magnitudes. The phase of a sinusoid can be described through the superposition of a sin and cos wave through the identity

$$\sin(\omega t + \phi) = a \sin(\omega t) + b \cos(\omega t) \quad (3.84)$$

for the angular frequency ω in rad/s, phase shift ϕ in rad, and superposition magnitudes a and b . So long as the angular frequency of the sinusoid is known the phase information can be determined by the disturbance accommodating SAC. Additionally, since the derivative of sin is $\frac{d}{dt} \sin(\omega t) = \omega \cos(\omega t)$, simply ensuring the disturbance accommodating SAC has access to all the states when adapting will ensure that it can compensate for the phase shift in the signal on its own, the final gains will simply compensate for the scaling in the derivative.

Some additional consideration must be taken when using feedforward parallelization to stabilize a non-ASPR plant under disturbances. If a SAC encounters a disturbance while controlling a plant that is naturally ASPR, the block diagram for the system would resemble the diagram in Fig. 3.16 and the SAC would manage the

disturbance through observations of its effects on the tracking error e_y . Disturbances can be rejected so long as additional gain adaptation can be used to manage the disturbance. Ideally, in the case of a non-ASPR plant the disturbance rejection of the system would be similarly strong. The block scheme diagram of a feedforward parallelized plant under ideal disturbance is shown in Fig. 3.17.

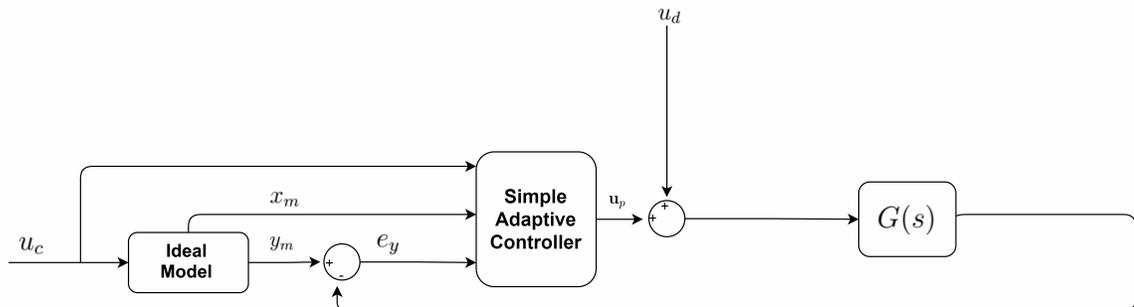


Figure 3.16: Disturbance Acting on ASPR Plant

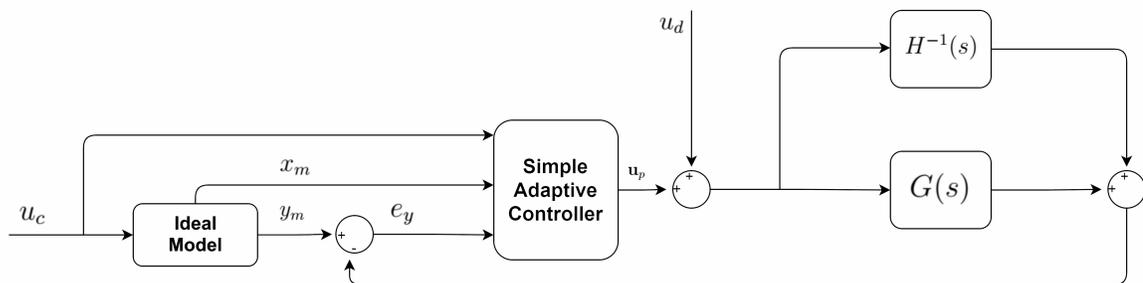


Figure 3.17: Disturbance Acting on Augmented ASPR Plant

However, in practice the disturbance being applied to the system is unknown. Since the disturbance is unknown it is not applied to the system augmentation. If the disturbance were known, it could also be applied to the system augmentation, and the disturbance would be countered as in Fig. 3.17. Instead, the block scheme diagram for a real disturbance is similar to Fig. 3.18.

The disturbance that the SAC attempts to control is equivalent to the one in Fig. 3.19.

The outcome is that the SAC successfully manages the disturbance on the augmented plant by increasing the output of the stabilizing controller. This is the same

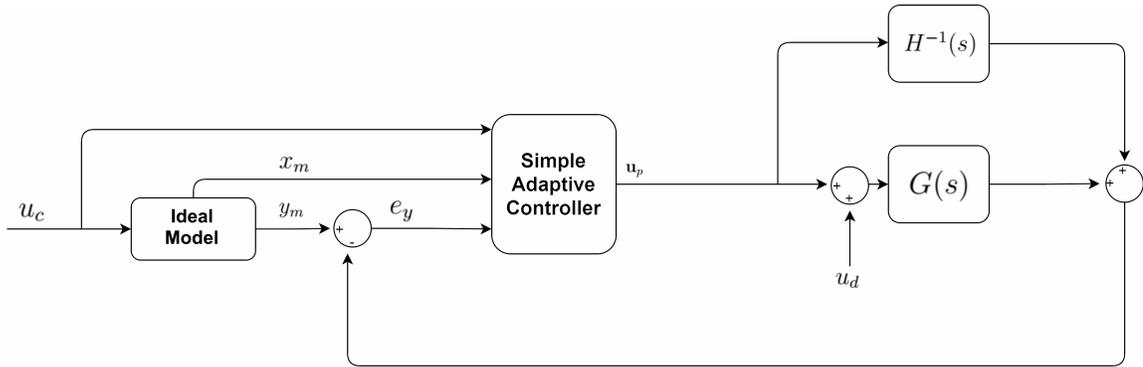


Figure 3.18: Disturbance Acting on non-ASPR Plant

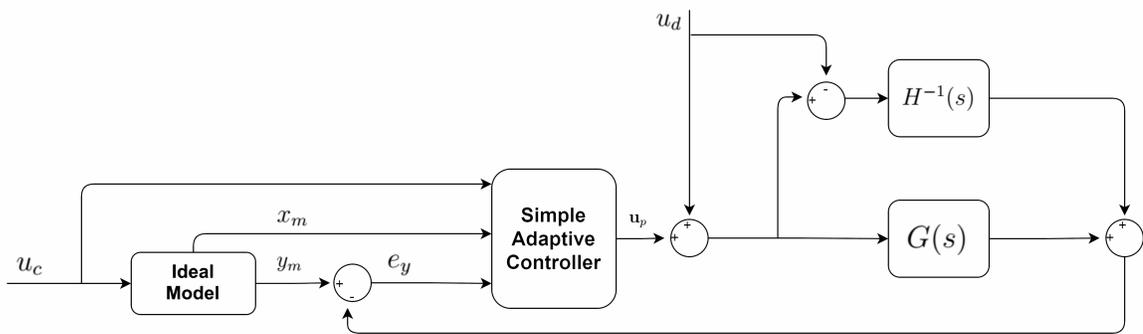


Figure 3.19: Disturbance Managed by SAC with Feedforward Parallelization

as decreasing the augmented system’s tracking error by increasing the tracking error of the true system. The additional tracking error added to the system follows

$$\frac{\mathbf{y}_s(s)}{\mathbf{u}_d(s)} = -H^{-1}(s) \tag{3.85}$$

The disturbance rejection of a SAC with feedforward parallelization is dependant on the quality of the stabilizing controller, and should motivate research into the benefits and drawbacks of different schemas to stabilize non-ASPR plants for use with SAC. The effects of uncompensated disturbances on SAC tracking accuracy will be experimentally verified in chapter 4.

Although using feedforward parallelization with SAC must lead to additional tracking error when a disturbance is introduced, there are still methods to mitigate the effects of disturbances. Disturbance accomodation can be introduced, and the

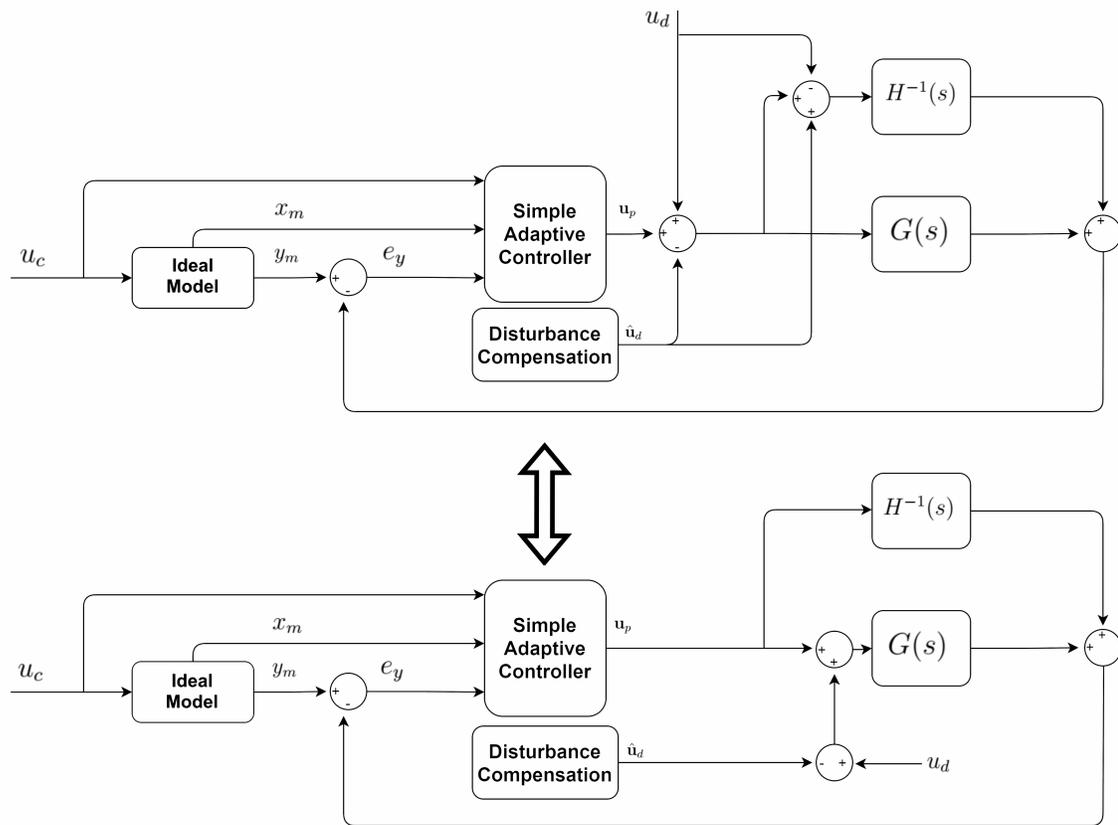


Figure 3.20: Equivalent Disturbance Compensation SAC Block Diagrams. To Improve Tracking, Disturbance Compensation Cannot be Passed to the Stabilizing Controller

additional commands passed directly to the plant without presenting it to the stabilizing controller. Allowing the disturbance compensation to bypass the feedforward parallelization allows the system with disturbance and compensation to resemble the one in Fig. 3.20. Now, the more the disturbance is managed by the disturbance compensation, the smaller the effect on the true system tracking.

3.5 SAC Design Summary

Despite its apparent complexity, it can be simple to design a SAC that matches or improves on linear controller performance under nominal conditions, and improves on control performance under uncertainty. The following section acts as a summary and overview of the design principles discussed in this chapter, as well as a guide to the

method of creating a SAC for a given physical system. By taking a few precautions SAC implementation becomes simple, while ensuring that stability and performance of the system are maintained when compared to an equivalent linear control system.

1. Determine if the system $\mathbf{G}(s)$ to be controlled is ASPR. The properties and requirements of ASPR systems in continuous and discrete time are discussed in Secs. 2.3.1 and 2.3.2.

If the system is ASPR, SAC can be implemented directly using Eqs. (2.100) through (2.105).

If the system is not ASPR, determine if sensor blending or feedforward parallelization is preferred to augment the system to an ASPR one. Discussion of ASPR system augmentation techniques is found in Sec. 2.5.5.

- If feedforward parallelization will be used to augment the system, determine if a stabilizing controller $\mathbf{H}(s)$ already exists for the system. If a controller exists, determine if its inverse can be computed. If the inverse of a stabilizing controller exists for the system, then the system can be converted into a similar ASPR system through the use of the inverse of the stabilizing controller $\mathbf{H}^{-1}(s)$. The augmented plant output \mathbf{y}_a will now be

$$\mathbf{y}_a(s) = (\mathbf{G}(s) + \mathbf{H}^{-1}(s))\mathbf{u}_p(s) \quad (3.86)$$

The stabilizing controller must stabilize the plant across the entire design space. When implemented with the SAC equations, the new system will have the block diagram seen in Fig. 3.21. Consider using the *augmentation of feedback* system, where the equation for augmentation is instead

$$\mathbf{y}_a(s) = \mathbf{G}(s)\mathbf{u}_p(s) + \mathbf{H}^{-1}(s)(\mathbf{K}_e(s)\mathbf{e}_y(s)) \quad (3.87)$$

- If sensor blending is used instead, determine if there exists a matrix of the

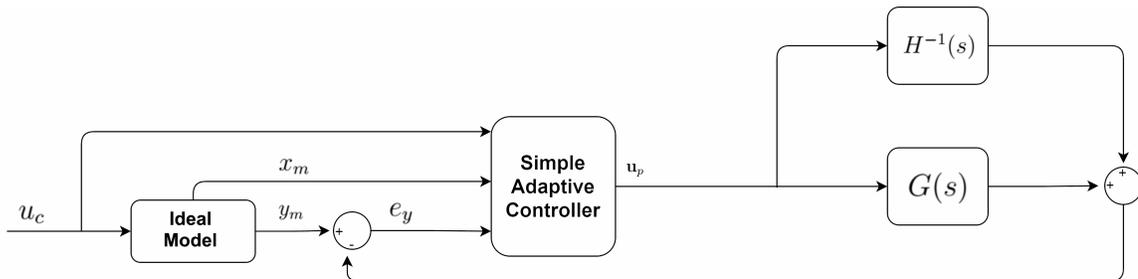


Figure 3.21: SAC Design with Feedforward Parallelization Using Stabilizing Controller $H(s)$.

system output measurements \mathbf{C}_Δ such that

$$\mathbf{y}_a = \mathbf{C}_\Delta \mathbf{y}_p \quad (3.88)$$

is the output of an ASPR system. If so, implement \mathbf{C}_Δ to create the augmented output for your system. Further discussion of sensor blending techniques is found in Balas and Frost [64].

2. Once the (augmented) plant is ASPR, determine the ideal model for the system. The ideal model can be a transfer function of any form, however a transfer function with sufficiently large degree will be able to meet the matching conditions seen in Eqs. (2.115) through (2.118), ensuring that a set of ideal gains exist when the plant structure is known. Selection of the ideal model should follow from guidance considerations, and the performance that must be achieved by the controller. If possible, ensure that ideal gains exist across the design space.
3. After the ideal model has been chosen, identify all possible nonlinearities in the system, and determine if they can be compensated. If the nonlinearities can be compensated through traditional techniques such as feedback linearization, incorporate that compensation. If not, ensure that nonlinear guidance can be created for the SAC such that the model output now represents an achievable system output. The nonlinear guidance may be in the form of the feasible model developed in Fig. 3.5, or any other nonlinear guidance. If ideal gains exist under the matching conditions, ensure that the states and ideal gains together

still correspond to a system with zero tracking error. If the nonlinear guidance provided can not be tracked by constant gains, then the SAC will continue adapting during the system response, which may reduce tracking during these periods. Ensure that at no point across the envelope does the feasible model correspond to an unachievable system response. Further discussion of ideal model and feasible model development is done in Secs. 3.3.1 and 3.3.2.

4. The normal SAC equations can now be implemented for the system with ASPR plant and feasible model reference. When running, each signal in \mathbf{r} represents a value that can be correlated to the tracking error \mathbf{e}_y . Ensure that no sources of uncorrelated error exist in the signal \mathbf{e}_y . Introducing uncorrelated tracking error will cause adaptation windup, and will affect the response.
5. Once the SAC architecture has been implemented, values for the design parameters can be determined. If the adaptive response does not need to be optimized, heuristics can be used to find an appropriate set of gains. Since heuristics rely on some amount of knowledge of the system response, the designer must first decide on suitable maximum linear gains \mathbf{K}_m through linear system analysis, from which suitable adaptation parameters can be determined. The heuristics provided in Sec. 3.3 can be used to determine sets of integral adaptation parameters that remain below predefined maximum gain values for a SISO system. If a reference simulation can be made, that simulation can be further used with the heuristics in Sec. 3.3.3 to determine accurate values for the integral adaptation parameters that remain within the selected maximum gains. Since the SAC adaptation equations are magnitude dependant use the largest expected commands. The value of the degradation parameter can be found based on estimates of the forcing gain or gain impulse of the system during operation, and is discussed in Sec. 3.3.4. The proportional adaptation parameters can also be chosen to provide guaranteed control power for some boundary values, and is discussed in Sec. 3.3.5.
6. If performance of the system must be optimized to reduce convergence time or control activation, metaheuristic searches of a representative simulation can

be used to determine appropriate design parameters for the simulation. Due to the robustness of direct adaptive methods, results for reasonably accurate simulations will transfer to real systems. Nonetheless, ensure that the nonlinear simulation used for optimization includes all representative behaviours of the plant, including delay, noise, and any other nonlinearities. The applicability of optimizing nonlinear simulations for selection of SAC design parameters is further explored through experimental verification in Chapter 4.

7. If known linear disturbances are present in the system, consider incorporating a disturbance compensator as seen in Prabhakar et al. [29].
8. Attempt to eliminate all sources of uncorrelated error. Many unexpected sources of uncorrelated error may be present and degrade the final system response. A common example is initializing the ideal model to a set of states that do not match the current system. It will take time for the erroneous adaptation caused by the initial mismatch to dissipate, and as such it is highly beneficial to eliminate sources of uncorrelated error before they occur.
9. The first design iteration has been completed. With the feasible model determined, an ASPR plant, and a set of designed adaptation parameters, adaptive control can be tested for performance. If design parameters are not sufficient, determine whether the cause is due to effects of the system architecture, the signal inputs, the design parameters, or other conflicting elements. Once the cause has been found, iterate the design process until system requirements can be met.

3.6 Chapter Summary

The body of knowledge on SAC design has been collected and developed to determine the aspects that can be used to improve the design and implementation of SACs in real world systems. Simple matching conditions allow for ideal gains for a given SAC to be determined. Heuristics for integral adaptation, degradation parameter, and proportional adaptation have been developed to simplify parameter selection. The problem of determining optimal SAC design parameters is formulated for nonlinear

and metaheuristic optimization searches. The performance of the output of these optimization searches is experimentally verified in the following Chapter. The design of a disturbance compensating SAC component is extended to a non-ASPR system stabilized through feedforward parallelization, with the consequences of disturbances on the system explained. The concept of a feasible model is created to ensure that adaptation windup does not occur for adaptive control of nonlinear systems, and more generally the problem of ideal model selection is understood to be a guidance problem. Design concepts and heuristics are collected into a short description of the process of completing an iteration of SAC design.

By collecting and analyzing the elements of SAC design, the autonomy of systems can be easily improved through the implementation of adaptation.

Chapter 4

Simple Adaptive Controller Design Experimental Verification

4.1 Introduction

The current chapter details the methods by which accumulated knowledge on the design of SAC is used to develop an implementation of a SAC for position control of the Spacecraft Robotics Control Laboratory's (SRCL) three degree of freedom frictionless testbed, called SPOT.

Nonlinear and metaheuristic optimization of SAC designs for the position control of a spacecraft are found for a representative simulation, and experimentally verified in Carleton's SRCL SPOT. A linear disturbance compensator is developed and tested for two virtual sinusoidal disturbances. The performance of a feedforward parallelized SAC is compared with and without adaptive disturbance compensation when the amplitude and phase are unknown but the frequency is known. The code and other resources used in designing and creating the experiments in this chapter are in a public GitHub repository at <https://github.com/AndriyPredmyrskyy/PredmyrskyyThesisCode2021>, or by contacting the author at AndriyPredmyrskyy@cmail.carleton.ca.

Nonlinear optimization was unable to improve SAC design to a noticeable degree. Metaheuristic searches are able to drastically improve selection of SAC design parameters. Virtual disturbances cause noticeable tracking error that is not detected in the SAC, as was predicted in Sec. 3.4. The inclusion of adaptive disturbance compensation is able to eliminate the tracking error due to the unknown virtual disturbance.

4.2 System Formulation

The SRCL's 3DOF frictionless laboratory, named the Spacecraft Proximity Operations Testbed or SPOT, is used to determine the control accuracy and viability for

the adaptive control techniques described in Chapter 2 and 3. An image of the SRLC is available in Fig. 4.1. The performance of metaheuristic and nonlinearly optimized SAC is tested for and compared to linear control for managing the position of a spacecraft. The performance of SAC disturbance compensation for linear disturbances is also tested.

4.2.1 Experimental Apparatus

The SPOT makes use of *platforms* which act as representative small spacecraft with onboard computation, measurement, and control capabilities to test various guidance, navigation, and control systems. Each platform is cubic with edges 1 ft long, and holding a compressed air system alongside onboard computation. Only the “Black” platform, so called because it typically is covered by black acrylic panelling, is used in these experiments. All modelling was done for the “Red” platform, typically covered in red acrylic panelling, which has a different mass and moment of inertia from the Black platform. The Red and Black platforms can be seen during operation in Fig. 4.2.

The platforms make use of three bottom-mounted air-bearings for frictionless contact between it and the level test surface. The test surface is an almost perfectly planar and level granite slab with surface dimensions 2.4 m by 3.7 m. The platforms are actuated by eight air nozzles, two on the bottom of each side of the platform. Each nozzle expels compressed air that is downregulated to 550 kPa to allow for acceleration in two directions. Nozzles can also be opened to create a torque around its one rotational axis. Each nozzle is opened and closed at a frequency of 500 Hz by pulsewidth modulated solenoid valves. Independently the thrusters are able to produce approximately 0.2 N of thrust, allowing a maximum planar force of 0.4 N in any one body-axis direction. Compressed air is used to provide the air-bearings and thrusters with the pressurized air that is required. A compressed air tank is filled between and 7 MPa and 27 MPa, which is then downregulated to provide the thrusters and air-bearing with the required pressure during experiments.

The platform structure is composed of an aluminum frame that holds three decks



Figure 4.1: Three-Degree-Of-Freedom Frictionless Testbed, the SPOT.

for storage of the various digital computation and pneumatic components. The platform is typically covered with semi-transparent acrylic panelling. The nominal mass of the Black platform with panelling installed is 13.6 Kg.

The position and rotation information of the platform is measured through the use of Light Emitting Diodes (LED) on the top corners of the platform. The PhaseSpace[©] motion capture system installed in the laboratory makes use of eight stereoscopic cameras to determine the position of each LED, which are then used to determine the position and rotation of the platform. The measurements of the PhaseSpace[©] system are highly accurate, with standard deviations of approximately $1e-7$ m during experiments, suggesting an average noise of less than 1 mm per measurement being common.

The PhaseSpace[©] measurements are processed by a server in the lab and are sent to the platform to be used for processing using the onboard Raspberry Pi-3 system.

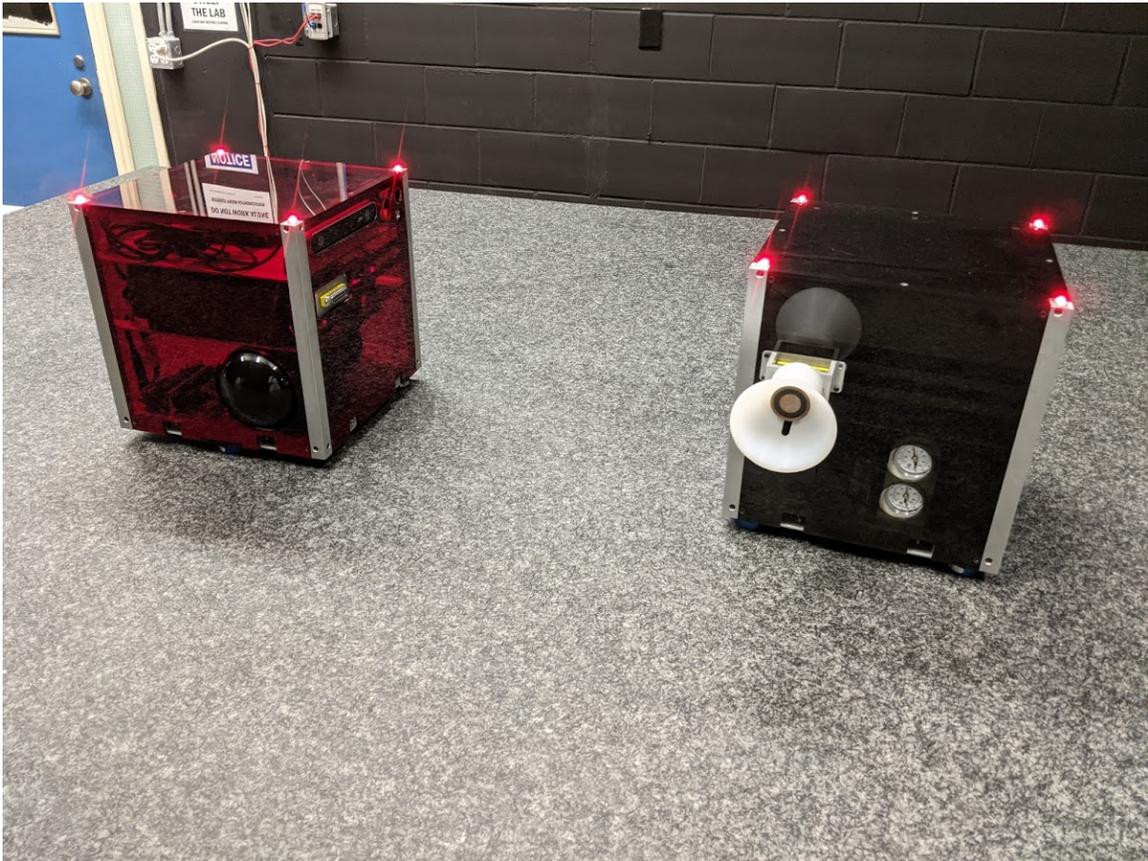


Figure 4.2: The “Red” Test Platform, Beside “Black” Test Platform

Guidance, navigation, and control algorithms for the platforms are deployed in C++ and run during experiments. The deployed C++ code is automatically generated from MATLAB®Simulink®.slx block diagrams, which allows for analysis of system performance in the MATLAB®Simulink®environment.

During experiments, measurements of the system’s position and attitude are sent to the onboard computers which pass the measurements to code running the onboard control procedures, which then produce thrusts to be used by the onboard pneumatic actuation system to provide positional and attitude control of the platform.

4.2.2 Software Implementation and Simulation

Previous experimental efforts have created a detailed and accurate simulation of the platform dynamics. The available SPOT simulation was used to prototype and develop experiments without the need to undergo lengthy setup and teardown of the

that the experiment occurs as desired. Once the experiment is complete the platform continues on to the fifth phase, where the platform is once again moved into a final position, the platform is allowed to float in the sixth phase, and finally systems are stopped and data is logged in the seventh phase. If at any time the experiment is deemed unsafe, if for example a control system goes unstable, measurements are not reaching the platform, or low-air is detected, the platforms can be made to cease all thruster actuations and simply float on the table whenever the emergency stop button, seen in the bottom right of Fig. 4.1, is pressed.

The SPOT 3.0 software was able to simulate measurement noise that was accurate to the real system performance, as well as control allocation that was sufficiently accurate to the real system control allocation. The experiments crafted to test the positional tracking of SAC for various control formulations, as well as the disturbance compensation of SAC to a linear disturbance generator are developed in simulation before being experimentally verified.

4.2.3 Nonlinear SAC Parameter Search

Although it can be shown that metaheuristic search techniques can determine SAC design parameters that lower the linear quadratic cost of the design, metaheuristic searches can be time and computationally intensive. Additionally, the use of a simulation in metaheuristic searches guarantees that a search must be rerun whenever the simulation changes. An attempt is made to determine if nonlinear search techniques can determine candidate SAC designs that lower the linear-quadratic cost of a SISO simulation of spacecraft proximity operations, and if those designs can compete with the outputs of metaheuristic searches. The nonlinearly optimized SAC design determined in this subsection is compared to metaheuristic designs and tested in the SPOT.

The SAC design equations outlined in Eqs. (2.96) through (2.100) are implemented into a nonlinear optimizer and optimized using the SQP method outlined in Sec. 2.6.3. Since both position axes have identical behaviour, only the nonlinear optimization of a SISO controller for position is performed.

A linear model approximation of the SPOT hardware is used during optimization.

The LTI state-space approximation $\{\mathbf{A}_{SPOT}, \mathbf{B}_{SPOT}, \mathbf{C}_{SPOT}, \mathbf{0}\}$ of the SPOT “Red” platform of mass $m_r = 16.45$ kg is formulated as

$$\mathbf{A}_{SPOT} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \mathbf{B}_{SPOT} = \begin{bmatrix} 0 \\ 1/16.95 \end{bmatrix}, \mathbf{C}_{SPOT} = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad (4.1)$$

Although the Black platform is used for experiments, by optimizing the SAC design for the Red platform the robustness of the design technique can also be probed. Metaheuristic optimization of the SAC design are also conducted for the Red platform and tested on the Black platform.

The system plant $\{\mathbf{A}_{SPOT}, \mathbf{B}_{SPOT}, \mathbf{C}_{SPOT}, \mathbf{0}\}$ is not ASPR, and is augmented through feedforward parallelization by a stabilizing controller. A PD controller with proportional gain $K_p = 0.1$ and derivative gain $K_d = 0.1$ stabilizes the system. The augmented plant output is then created following the feedforward parallelization technique

$$H^{-1}(s) = \frac{1}{K_d s + K_p} \quad (4.2)$$

$$y_a(s) = y_p(s) + y_s(s) \quad (4.3)$$

$$y_s(s) = H^{-1}(s)u_p(s) \quad (4.4)$$

From which a state-space system can be determined. The final discrete system is characterized by $\{\mathbf{A}_{SPOTd}, \mathbf{B}_{SPOTd}, \mathbf{C}_{SPOTd}, \mathbf{D}_{SPOTd}\}$ with

$$\mathbf{A}_{SPOTd} = \begin{bmatrix} -1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad (4.5)$$

$$\mathbf{B}_{SPOTd} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad (4.6)$$

$$\mathbf{C}_{SPOTd} = \begin{bmatrix} 0.1 & 0.0811 & 0.0811 \end{bmatrix}, \quad (4.7)$$

$$\mathbf{D}_{SPOTd} = \begin{bmatrix} 0 \end{bmatrix} \quad (4.8)$$

A timestep of $h = 0.1$ s was chosen to match the SPOT hardware update rate of 10 Hz.

Each of the SAC design equations represents an equality constraint for either a system variable or its derivative over time. Similar to the creation of system dynamics constraints through equality constraints in Liu and Lu [68], the gain adaptations are constructed through Euler integration of the nonlinear update equations. For timestep $k \in \mathbb{Z}^+$, the equality constraint for the gain updates are

$$\mathbf{K}_I(k) = \mathbf{K}_I(k-1) + \frac{h}{2} \mathbf{e}_y(k) \mathbf{r}^T(k) \mathbf{\Gamma}_I + \frac{h}{2} \mathbf{e}_y(k-1) \mathbf{r}^T(k-1) \mathbf{\Gamma}_I \quad (4.9)$$

The discrete plant update equations can then be included in the equality constraints following the method outlined by Lu and Liu [68]. Now it is true for all timesteps k that

$$\begin{aligned} \left(\mathbf{I}_3 - \frac{h}{2} \mathbf{A}_{SPOTd}(k) \right) \mathbf{x}_p(k) &= \left(\mathbf{I}_3 + \frac{h}{2} \mathbf{A}_{SPOTd}(k-1) \right) \mathbf{x}_p(k-1) \\ &+ \frac{h}{2} \mathbf{B}_{SPOTd} \mathbf{u}_p(k) + \frac{h}{2} \mathbf{B}_{SPOTd} \mathbf{u}_p(k-1) \end{aligned} \quad (4.10)$$

Since the ideal model outputs and states do not depend on the SAC design parameters, they can be calculated beforehand and do not need to be incorporated as

constraints. For a second-order ideal model of natural frequency $\omega_n = 1$ rad/s and damping coefficient $\zeta = 1$, the ideal model output signals is given by $y_m \in \mathbb{R}$ and state signals $\mathbf{x}_m \in \mathbb{R}^2$ for model inputs $u_c \in \mathbb{R}$. The specific ideal model used during optimization is given by

$$\mathbf{A}_m = \begin{bmatrix} 0 & 1 \\ -11.11 & -4 \end{bmatrix}, \quad \mathbf{B}_m = \begin{bmatrix} 0 \\ 11.11 \end{bmatrix} \quad (4.11)$$

$$\mathbf{C}_m = \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad \mathbf{D}_m = \begin{bmatrix} 0 \end{bmatrix} \quad (4.12)$$

The remainder of the SAC design equations are implemented as equality constraints across each of the timesteps k .

$$\mathbf{u}_p(k) = \mathbf{K}(k)\mathbf{r}(k) \quad (4.13)$$

$$\mathbf{K}(k) = \mathbf{K}_P(k) + \mathbf{K}_I(k) \quad (4.14)$$

$$\mathbf{K}_P(k) = \mathbf{e}_y(k)\mathbf{r}^T(k)\mathbf{\Gamma}_P \quad (4.15)$$

$$\mathbf{r}(k) = \begin{bmatrix} y_m(k) - (\mathbf{C}_{SPOTd}\mathbf{x}_p(k) + \mathbf{D}_{SPOTd}u_c(k)) \\ \mathbf{x}_m \\ u_c \end{bmatrix} \quad (4.16)$$

The final nonlinear optimization problem is written as

$$\underset{x \in \mathcal{V}}{\text{minimize}} \quad O = \int_0^T \mathbf{e}_y^T(k) \mathbf{Q} \mathbf{e}_y(k) + u_p^T(k) \mathcal{R} u_p(k) dk \quad (4.17)$$

$$\text{subject to} \quad \left(\mathbf{I}_3 - \frac{h}{2} \mathbf{A}_{SPOTd}(k) \right) \mathbf{x}(k) = \left(\mathbf{I}_3 + \frac{h}{2} \mathbf{A}_{SPOTd}(k-1) \right) \mathbf{x}(k-1) \quad (4.18)$$

$$+ \frac{h}{2} \mathbf{B}_{SPOTd} \mathbf{u}_p(k)$$

$$+ \frac{h}{2} \mathbf{B}_{SPOTd} \mathbf{u}_p(k-1)$$

$$\mathbf{K}_I(k) = \mathbf{K}_I(k-1) + \frac{h}{2} \mathbf{e}_y(k) \mathbf{r}^T(k) \mathbf{\Gamma}_I + \frac{h}{2} \mathbf{e}_y(k-1) \mathbf{r}^T(k-1) \mathbf{\Gamma}_I \quad (4.19)$$

$$\mathbf{u}_p(k) = \mathbf{K}(k) \mathbf{r}(k) \quad (4.20)$$

$$\mathbf{K}(k) = \mathbf{K}_P(k) + \mathbf{K}_I(k) \quad (4.21)$$

$$\mathbf{K}_P(k) = \mathbf{e}_y(k) \mathbf{r}^T(k) \mathbf{\Gamma}_P \quad (4.22)$$

$$\mathbf{r}(k) = \begin{bmatrix} y_m(k) - (\mathbf{C}_{SPOTd} x(k) + \mathbf{D}_{SPOTd} u_c(k)) \\ \mathbf{x}_m(k) \\ u_c(k) \end{bmatrix} \quad (4.23)$$

where t_f simply represents the final iteration instead of the final time in seconds. The nonlinear update in Eq. (4.9) is not compatible with convex optimization techniques, and so nonlinear optimization techniques must be used. The nonlinear SQP optimizer available in MATLAB[®]'s optimization toolbox is used alongside the YALMIP toolbox to solve the nonlinear optimization problem. Nonlinear SQP determined an improved SAC design for a step input of 1 meter over a command time of 100 seconds.

In all cases tested, nonlinear optimization was able to determine adaptation parameters that decreased the initial system cost, however decreases to the cost of the final solutions were never as drastic as those that occurred in metaheuristic searches. The most reliable method to improve the result of a nonlinear search was to decrease the cost of the initial design parameters.

The output controller determined by nonlinear optimization of the SAC design equations for a linear approximation of the SPOT hardware is summarized in Table 4.1. The performance of the nonlinearly optimized SAC will be compared to the metaheuristic optimized controllers during experiments.

Table 4.1: Nonlinearly Optimized SAC Parameters

Variable	Nonlinearly Optimized SAC
Γ_{eP}	12.8
Γ_{eI}	6.63
Γ_{uP}	3.25
Γ_{uI}	2.63×10^{-4}
Γ_{x1P}	1.8×10^{-5}
Γ_{x2P}	0.8120
Γ_{x1I}	4.77×10^{-5}
Γ_{x2I}	160
σ	0

4.2.4 SAC Positional Tracking Experimental Setup and Controller Development

Previous work by Ulrich et al. [74] has determined that SAC can be implemented to successfully control the position of a spacecraft during spacecraft proximity operations under significant mass uncertainty. The goal of current positional tracking experiments are to determine if nonlinear optimization and/or metaheuristics are able to determine improved controller formulations for SAC in the SPOT.

For the SPOT no orbital perturbations or primary gravitational body affects the planar translational or rotational dynamics. Additionally, to simplify results, only positional tracking is considered. Rotational tracking of the system is managed by a PD control scheme that keeps the platform's rotation fixed. As such, the system dynamics are simply described by a double-integrator form

$$F = ma = m\ddot{x} \quad (4.24)$$

which is not an ASPR system.

The SAC response will be compared to the response and tracking of an LQR controller for the same system as a baseline. To allow for comparison of system responses, the same weighing matrices \mathcal{Q} and \mathcal{R} will be used for both the LQR and metaheuristic evaluation of the optimal control parameters, with values of

$$\mathcal{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (4.25)$$

$$\mathcal{R} = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix} \quad (4.26)$$

The LQR controller formulation for this system suggests the following proportional and derivative gains

$$\mathbf{e}(t) = \mathbf{y}_{cmd}(t) - \mathbf{y}_p(t) \quad (4.27)$$

$$\mathbf{u}_p(t) = \dot{\mathbf{e}}(t)2.8108 + \mathbf{e}(t)0.3162 \quad (4.28)$$

For the adaptive controller, feedforward parallelization by a stabilizing controller is added to the system to create an ASPR augmented plant. The stabilizing controller is chosen to minimize differences between the augmented and true plant. The PD position controller used for the stabilizing controller $H(s)$ is found to be

$$H(s) = K_{dSAC}s + K_{pSAC} = 1s + 0.1 \quad (4.29)$$

The inverse of this stabilizing controller is found and used to create an augmented plant following

$$\mathbf{H}^{-1}(s) = \frac{1}{\mathbf{K}_{dSAC}s + \mathbf{K}_{pSAC}} = \mathbf{I}_2 \frac{1}{K_{dSAC}s + K_{pSAC}} \quad (4.30)$$

$$\mathbf{y}_s(s) = \mathbf{u}_p(s)\mathbf{H}^{-1}(s) \quad (4.31)$$

$$\mathbf{y}_a(s) = \mathbf{y}_p(s) + \mathbf{y}_s(s) \quad (4.32)$$

with conversion of the Laplace space stabilizing controller inverse $\mathbf{H}^{-1}(s)$ being done through MATLAB[®]'s `tf2ss` function to create the state-space equivalent system for the filter and `c2d` to convert that filter into a discrete form that can be used by

the onboard calculations to determine the augmented system tracking error

$$\mathbf{e}_{ya}(t) = \mathbf{y}_m(t) - \mathbf{y}_a(t) \quad (4.33)$$

The ideal model used by the SAC to determine the ideal response was defined by the second-order system and associated state-space model

$$\frac{\mathbf{y}_m(s)}{\mathbf{u}_c(s)} = \frac{11.1}{s^2 + 4s + 11.1} \quad (4.34)$$

$$\mathbf{A}_m = \begin{bmatrix} \mathbf{0} & \mathbf{1}\mathbf{I}_2 \\ -11.1\mathbf{I}_2 & -4\mathbf{I}_2 \end{bmatrix}, \quad \mathbf{B}_m = \begin{bmatrix} \mathbf{0} \\ 11.1\mathbf{I}_2 \end{bmatrix}, \quad (4.35)$$

$$\mathbf{C}_m = \begin{bmatrix} \mathbf{I}_2 & \mathbf{0} \end{bmatrix}, \quad \mathbf{D}_m = \mathbf{0} \quad (4.36)$$

A second-order system with dampening of only 0.6 was intentionally chosen to provide an ideal response that was comparable, yet somewhat faster than the LQR response for the same system.

The corresponding system design parameters were thus encoded as

$$\mathbf{\Gamma}_P = \begin{bmatrix} \Gamma_{eP}\mathbf{I}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \Gamma_{x1P}\mathbf{I}_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \Gamma_{x2P}\mathbf{I}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \Gamma_{uP}\mathbf{I}_2 \end{bmatrix} \quad (4.37)$$

$$\mathbf{\Gamma}_I = \begin{bmatrix} \Gamma_{eI}\mathbf{I}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \Gamma_{x1I}\mathbf{I}_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \Gamma_{x2I}\mathbf{I}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \Gamma_{uI}\mathbf{I}_2 \end{bmatrix} \quad (4.38)$$

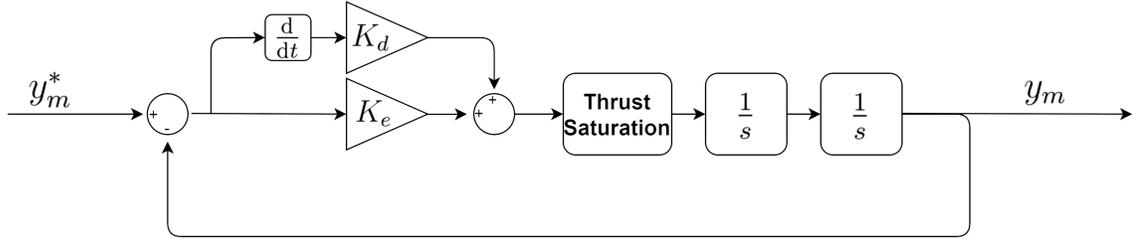


Figure 4.4: Feasible Model and Satisfying Controller for Example Thruster Saturation

$$\sigma = \begin{bmatrix} \sigma \mathbf{I}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \quad (4.39)$$

In order to manage system nonlinearities and ensure performance despite the possibility of adaptation windup, the ideal model was converted to a feasible model through the process described in Sec. 3.3.2. The Black platform has a maximum thrust of 0.4 N, and due to the memoryless nature of the LQR controller, it is not possible for a SAC implementation to match or surpass the saturated LQR response without also being capable of reaching thruster saturation without compromising the stability of the system. As such, the feasible model was constructed using the knowledge of the thrust saturation, alongside a satisfying controller which was used to ensure that, for commands that did not meet saturation, the feasible model closely matched the ideal model. Figure 4.4 is a reminder of the satisfying controller structure to produce a feasible model output, with Fig.4.5 demonstrating the satisfying controller used in the SPOT simulation, and subsequently the experiments run using that simulation.

The satisfying controller used in the SPOT simulation had gains of $\mathbf{K}_{p,sat} = 3.1623$ and $\mathbf{K}_{d,sat} = 10$. The states used for the SAC \mathbf{x}_m signal were also pulled from the feasible model.

Finally, SAC was implemented using the original SAC described in Eqs. (2.88) through (2.96). The forgetting factor term was included due to the presence of noise in the system position measurements. The system was initialized with zero initial gains. The SAC gains therefore developed without any knowledge of the system structure.

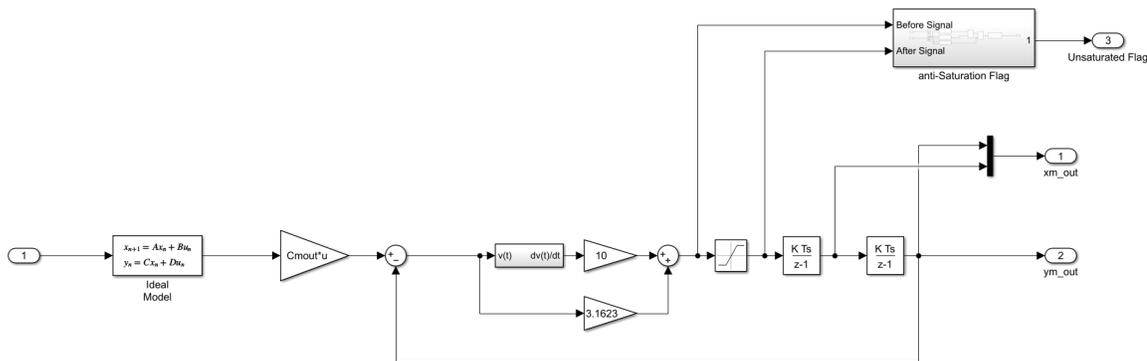


Figure 4.5: Satisfying Controller in SPOT Simulation and Experimental Verification

Different sets of adaptation rates, proportional and integral, were determined by the nonlinear optimization and metaheuristic searches.

A manual SAC design was created using the heuristics outlined in Sec. 3.3. A proportional adaptation was chosen to ensure the platform would not deviate more than 0.1 m from the ideal model, and the forcing gain was computed directly from the SPOT nonlinear simulation to determine good integral adaptation parameters. The forgetting factor was chosen using a steady-state feedback gain of 100 for the known noise variance of $\delta^2 = 1 \times 10^{-7}$.

Nonlinear optimization for the SAC gains was done using the nonlinear optimization equations outlined in Sec. 4.2.3. The optimization trajectory was composed of a step command for 100 seconds and an update rate of 100 Hz. The system constraints were encoded in the YALMIP open-source optimization toolbox for use with MATLAB®2021a. The nonlinear equations were solved using the SQP technique described in Sec. 2.6.3.

Metaheuristic searches for SAC design parameters were done using calls to the SPOT 3.0 software directly, thereby incorporating any nonlinearities and system behaviours into the search. Of the four metaheuristic techniques, (DE, SaDE, PSO, and SPSO) all four were used to determine optimized designs, but only two were used for experiments. The SAC designs determined by SaDE and SPSO were used for experiments not only because they determined lower cost controllers than the DE and PSO techniques, but also because the output SaDE controller relies on large Γ_{xI} , while the SPSO controller lowered its cost through Γ_{eI} , highlighting that multiple

Name	Γ_{eP}	Γ_{x1P}	Γ_{x2P}	Γ_{uP}	Γ_{eI}	Γ_{x1I}	Γ_{x2I}	Γ_{uI}	σ
Heuristic Design	200	0	0	0	5.64×10^4	6.66	679.9	6.66	0.0011
Nonlinear Optimization	12.8	1.8×10^{-5}	0.8120	3.25	6.63	4.77×10^{-5}	160	2.63×10^{-4}	0
SaDE	0	0	0	0	7.04×10^3	0	4.43×10^3	9.18	0
SPSO	0	0	0	0	1×10^4	0	0.183×10^3	0	0.02

Table 4.2: Nonlinear Search and Metaheuristic Optimized SAC Parameters

SAC controller formulations can achieve similar cost using different approaches.

The manually designed, nonlinearly optimized, SaDE optimized, and SPSO optimized SAC designs are shown in Table 4.2. Metaheuristic optimization was conducted in only the integral adaptation parameters, while nonlinear optimization could not determine good controllers for the system without proportional adaptation.

Optimization searches were conducted on a circular test trajectory with angular velocity 0.03 rad/s and radius 1 m. The experimental test trajectory is different from the optimization trajectory to ensure that optimization does not overfit for trajectory. The experimental test trajectory is composed of two frequency components, and is governed by the equation

$$x_{cmd}(t) = 0.5 \cos(0.03t) + 0.35 \sin(0.09t) \quad (4.40)$$

$$y_{cmd}(t) = 0.5 \sin(0.03t) + 0.35 \cos(0.09t) \quad (4.41)$$

Experiments are conducted to determine the trajectory tracking performance of each of the LQR linear controller, nonlinearly optimized SAC design, SaDE optimized SAC design, and SPSO optimized SAC design. Each controller is commanded to follow the experimental trajectory through one full rotation of the command trajectory, corresponding to a 200 second experiment. Trajectories are repeated for each of the tested controllers 10 times to ensure that performance is repeatable for each of the controllers.

4.2.5 SAC Disturbance Compensation Experimental Setup

Disturbance compensation for a linear disturbance is tested on the SPOT to determine the applicability of linear disturbance compensation to SAC implementations.

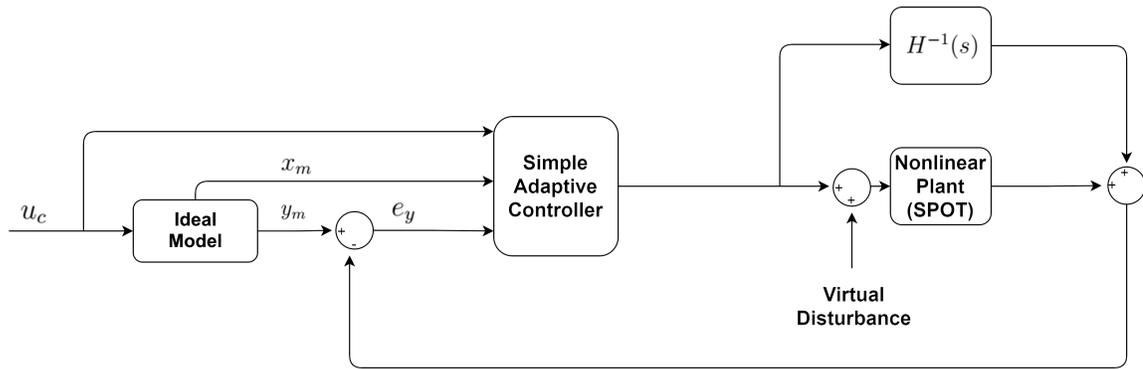


Figure 4.6: Virtual Disturbances Act Similarly to Real Disturbances.

Tests are performed for two controllers: one controller with manually chosen SAC parameters, without disturbance compensation, and under virtual linear disturbance; and another controller using the same SAC design under the same virtual linear disturbance, but now including adaptive disturbance compensation. The disturbance compensation follows the linear disturbance generator model shown in Sec. 2.5.4 and 3.4, where the frequency of the disturbances, but not the phase or amplitude, are known.

The disturbance is provided to the system virtually. No easily accessible and linear disturbances could be generated on the system directly, so a virtual disturbance was created for the SAC disturbance accommodation to combat. A virtual disturbance was created by including an unknown force command to the thruster output. The disturbance command is passed to the thrusters which creates the force disturbance. If the SAC is able to compensate for the force disturbance perfectly then the final thruster output will be the same as though the disturbance were not there, however, if the SAC is not able to compensate for the virtual disturbance the platform will include additional uncompensated thrusts in its command to the thruster actuation system. The inclusion of virtual disturbances is similar to a true disturbance, with the only difference being that the disturbance occurs before nonlinear system effects, such as control allocation and thruster saturation. The virtual disturbance is visualised in Fig. 4.6, where the virtual disturbance appears identical to the real disturbance present in Fig. 3.18.

Despite being produced internally, the virtual disturbance still constitutes a real

Name	Γ_{zI}	$\mathbf{\Gamma}_P$	Γ_{eI}	Γ_{x1I}	Γ_{x2I}	Γ_{uI}	σ
SAC without DA	0	$\mathbf{0}$	1×10^4	1×10^{-1}	1×10^{-1}	1×10^{-2}	0.001
SAC with DA	1	$\mathbf{0}$	1×10^4	1×10^{-1}	1×10^{-1}	1×10^{-2}	0.001

Table 4.3: Disturbance Compensation Testing Gains

signal in the system that must be combated by the SAC in order to achieve ideal model following.

The SAC architecture used during these tests and the position tracking experiments, with the addition of Prabhakar et al.'s disturbance compensation component for two sinusoidal disturbances of known frequency. The new form of the design parameters is

$$\mathbf{\Gamma}_P = \begin{bmatrix} \Gamma_{eP}\mathbf{I}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \Gamma_{x1P}\mathbf{I}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \Gamma_{x2P}\mathbf{I}_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \Gamma_{uP}\mathbf{I}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \Gamma_{zP}\mathbf{I}_2 \end{bmatrix} \quad (4.42)$$

$$\mathbf{\Gamma}_I = \begin{bmatrix} \Gamma_{eI}\mathbf{I}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \Gamma_{x1I}\mathbf{I}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \Gamma_{x2I}\mathbf{I}_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \Gamma_{uI}\mathbf{I}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \Gamma_{zI}\mathbf{I}_4 \end{bmatrix} \quad (4.43)$$

$$\sigma = \begin{bmatrix} \sigma\mathbf{I}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \quad (4.44)$$

A manually designed SAC position controller is used to clarify the effects of linear disturbances on a typical SAC implementation. The manual SAC design parameters with and without disturbance accommodation are outlined in Table 4.3.

The virtual disturbance used to negatively affect SAC performance was composed of two sinusoids of random phase and amplitude less than the thruster saturation. The choice of disturbance included one large-amplitude lower-frequency disturbance, and one low-amplitude higher-frequency disturbance. The frequency of these disturbances are known by the SAC linear disturbance compensator, while the amplitude and phase are unknown.

The disturbances are provided by the signals

$$u_{d1}(t) = 0.2 \sin(0.03t + 2.06) \quad (4.45)$$

$$u_{d2}(t) = 0.1 \sin(0.06t + 5.87) \quad (4.46)$$

which create the disturbance force vector

$$\mathbf{u}_d(t) = \begin{bmatrix} u_{d1}(t) \\ u_{d2}(t) \end{bmatrix} \quad (4.47)$$

The SAC disturbance accommodation is composed of the linear disturbance model described by

$$\hat{\mathbf{z}}_d(t) = \begin{bmatrix} \sin(0.03t) \\ \cos(0.03t) \\ \sin(0.06t) \\ \cos(0.06t) \end{bmatrix} \quad (4.48)$$

The disturbance generator is implemented in Simulink[®] using sinusoid sources for simplicity. The disturbance generator is used alongside the disturbance accommodation adaptation Eqs. (2.138) through (2.142) to produce the force to be made to compensate for the measured disturbance.

As was discussed in Sec. 3.4, to remove tracking error in a feedforward parallelized non-ASPR SAC implementation, the force commanded by the disturbance compensation component does not pass to the feedforward parallelization, instead passing

directly to the final output command.

Both the compensated and uncompensated controllers are made to track the two-frequency cycloid used in the SAC positional tracking experiment described in Sec. 4.2.4 under the virtual disturbance. Five experiments were performed for both the uncompensated and compensated cases.

4.3 SAC Position Tracking Experimental Results

The experiment described in Sec. 4.2.4 is performed in SRCL's SPOT. Each of the nonlinearly optimized, SaDE optimized, and SPSO optimized SAC formulations were compared with a baseline LQR controller.

The position response, force output, tracking error, and position command error are compared for each controller. The graphs for each of the pertinent system responses is demonstrated in Figs. 4.7 through 4.25.

The cycloid standard deviations used in Figs. 4.7, 4.10, 4.18, 4.22, 4.26, and 4.30 are calculated using the cross-track error of each controller during the command. Since position deviations in the on-track direction align with the cycloid command, representing the on-track deviations in the cycloid figures would unfairly represent the system response.

The on-track direction for a given sample is found through the change in the position command \mathbf{r}_c . That is to say, at iteration k the difference of the position (the on-track difference vector) is given by

$$\mathbf{v}_c(k) = \mathbf{r}_c(k) - \mathbf{r}_c(k-1) = \begin{bmatrix} \mathbf{v}_{c1}(k) \\ \mathbf{v}_{c2}(k) \end{bmatrix} \quad (4.49)$$

A unit vector that is perpendicular to the velocity is then found and is called the cross-track direction $\mathbf{v}_\perp(k)$

$$\mathbf{v}_\perp(k) = \begin{bmatrix} \mathbf{v}_{c2}(k) \\ -\mathbf{v}_{c1}(k) \end{bmatrix} \cdot \frac{1}{\|\mathbf{v}_c(k)\|} \quad (4.50)$$

The projection of the standard deviation at iteration k is then taken in the cross-track direction $\mathbf{v}_\perp(k)$ to determine the cross-track standard deviation.

	LQR	Manually Designed	Nonlinearly Optimized	SaDE Optimized	SPSO Optimized
Cost	7.51	29.60	41.11	27.07	19.94

Table 4.4: Linear-Quadratic Cost of SAC Designs

The final cost of each controller, according to the linear-quadratic cost function, is presented in Tab. 4.4. The cost displayed is the average of the costs of each individual run.

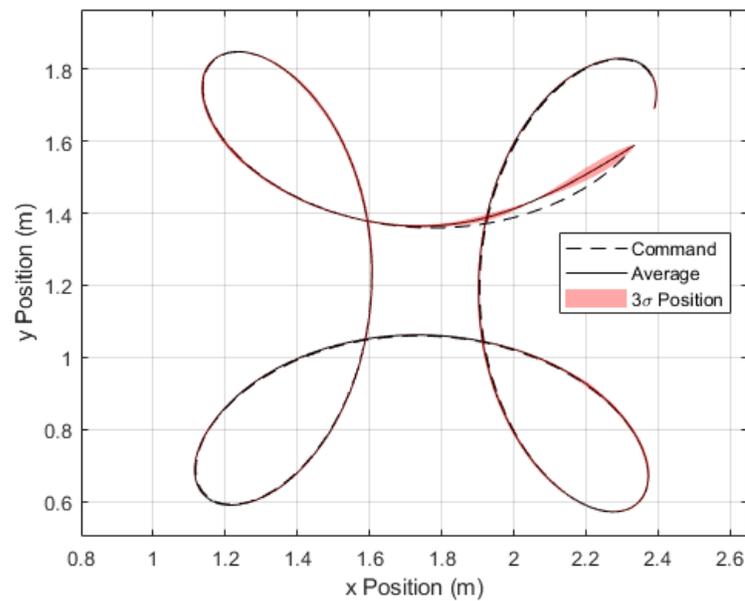


Figure 4.7: LQR Controller Position Graph and 3σ Standard Deviation

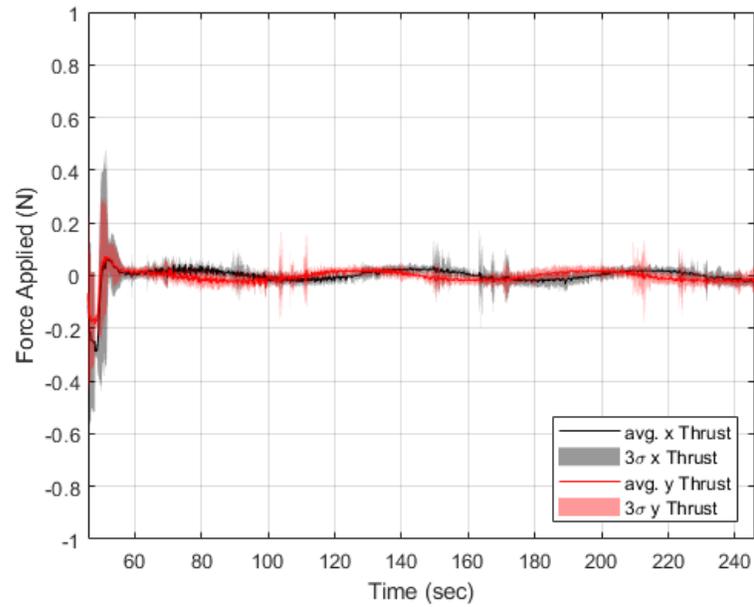


Figure 4.8: LQR Controller Force Output, with 3σ Standard Deviation

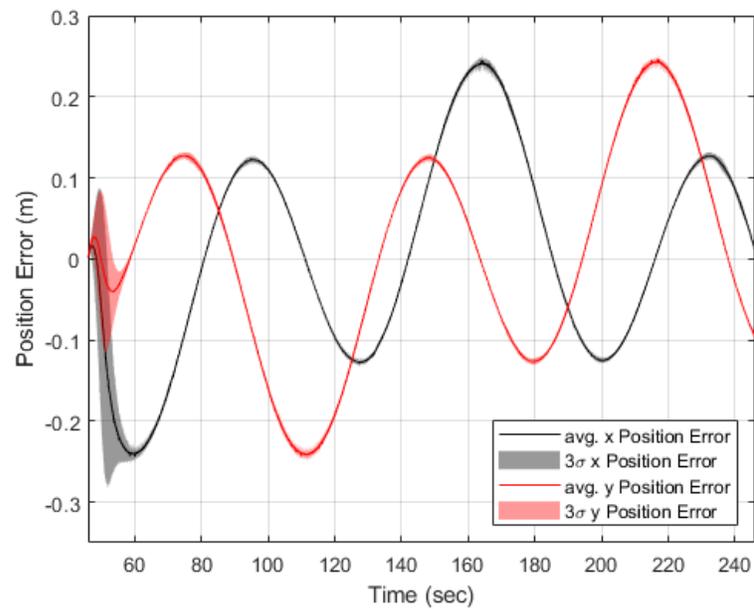


Figure 4.9: LQR Controller Position Error to the Command, with 3σ Standard Deviation

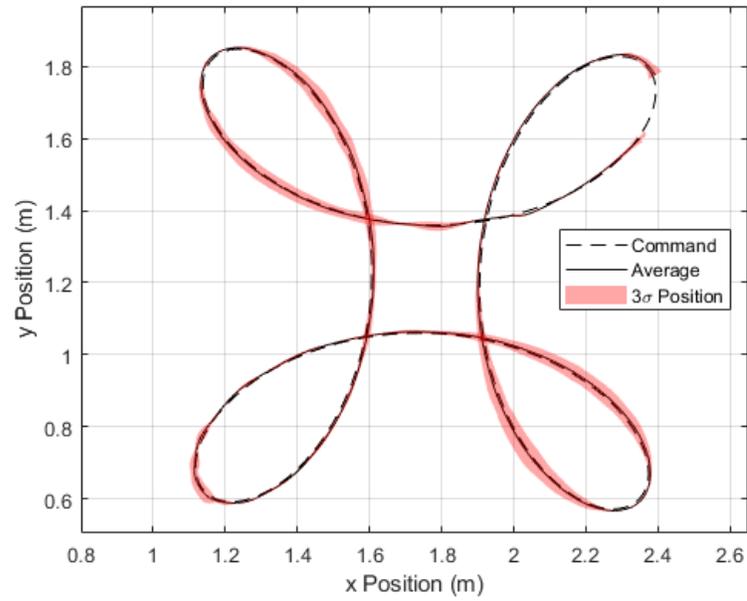


Figure 4.10: Manually Designed Controller Position Graph and 3σ Standard Deviation

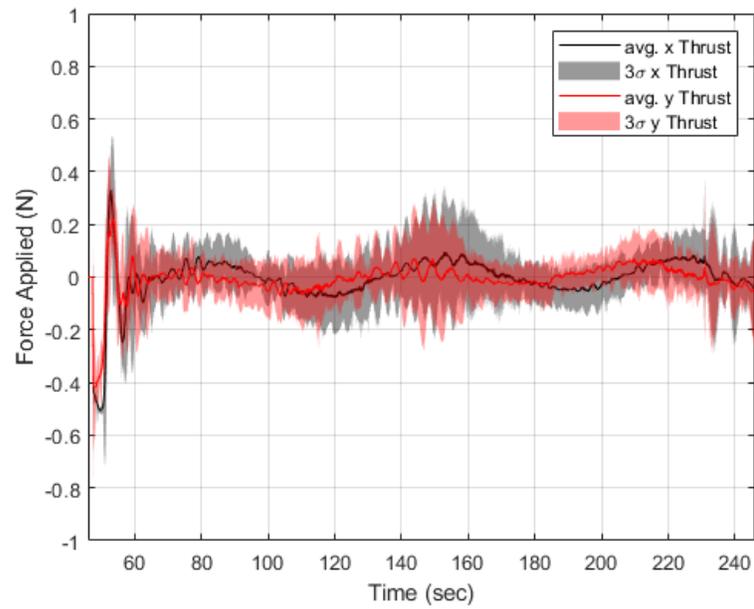


Figure 4.11: Manually Designed Controller Force Output, with 3σ Standard Deviation

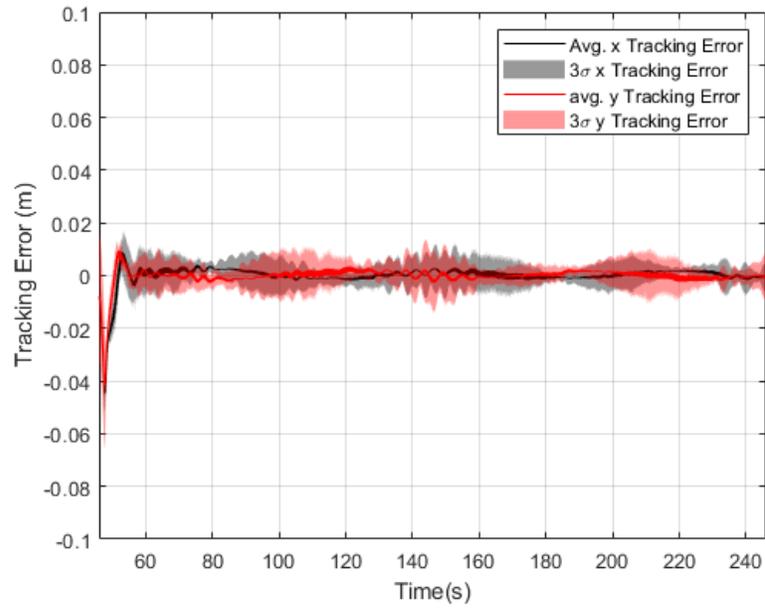


Figure 4.12: Manually Designed Controller Tracking Error, with 3σ Standard Deviation

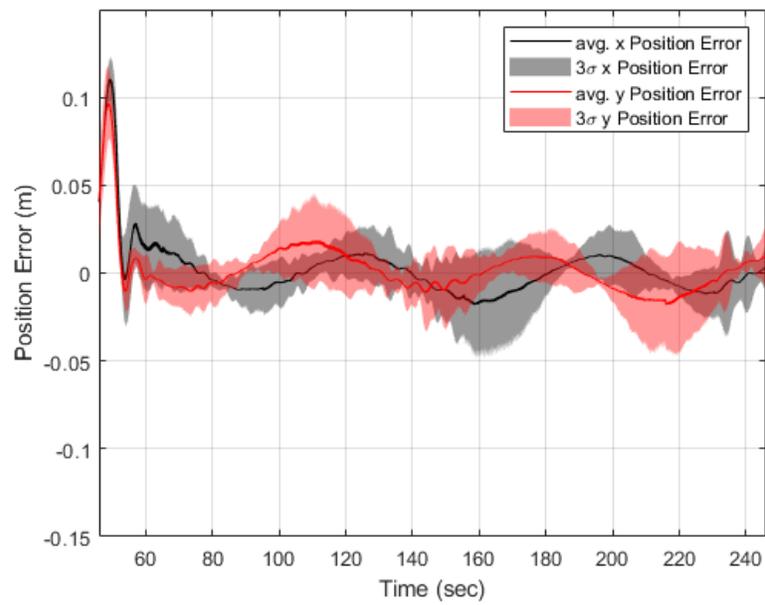


Figure 4.13: Manually Designed Controller Position Error to the Command, with 3σ Standard Deviation

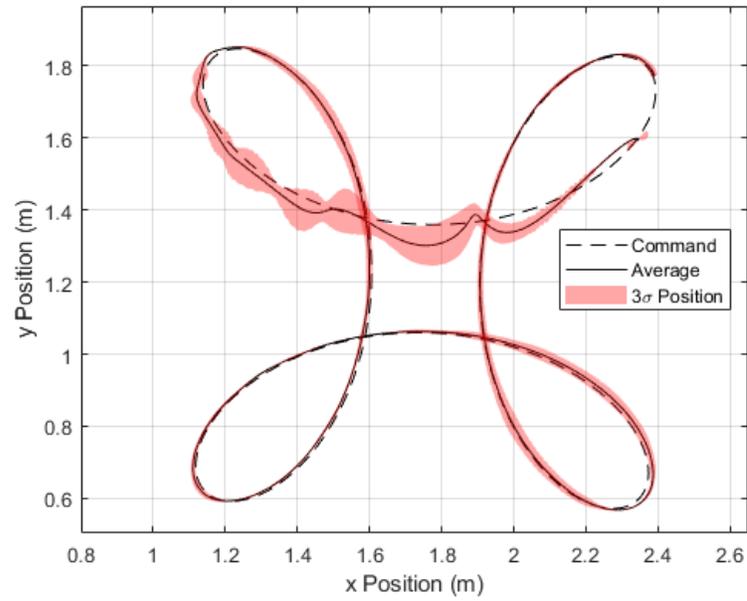


Figure 4.14: Nonlinearly Optimized Controller Position Graph and 3σ Standard Deviation

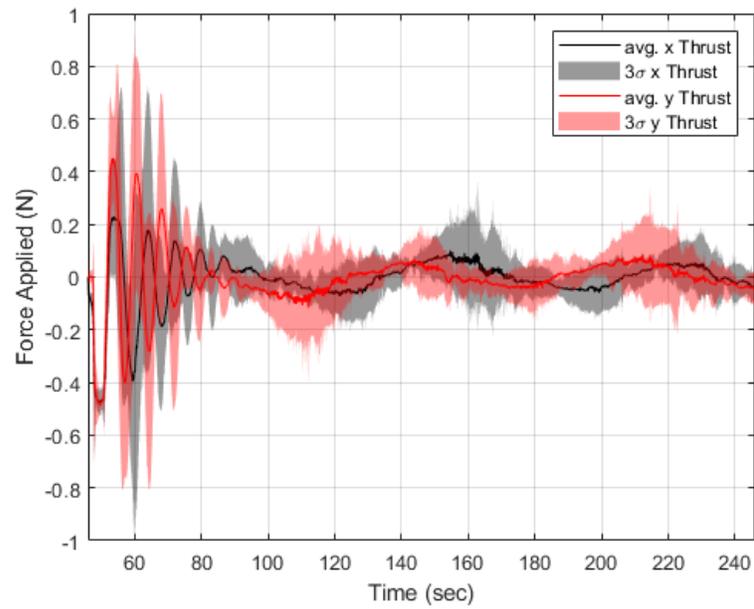


Figure 4.15: Nonlinearly Optimized Controller Force Output, with 3σ Standard Deviation

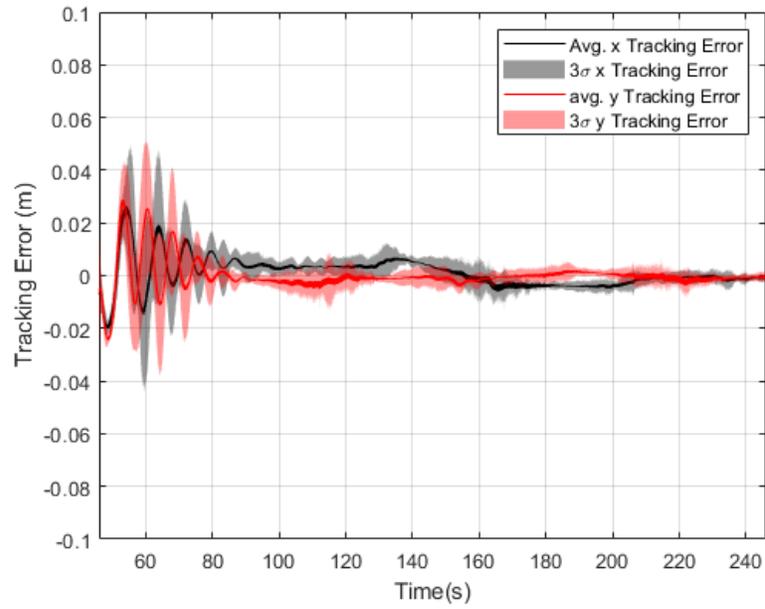


Figure 4.16: Nonlinearly Optimized Controller Tracking Error, with 3σ Standard Deviation

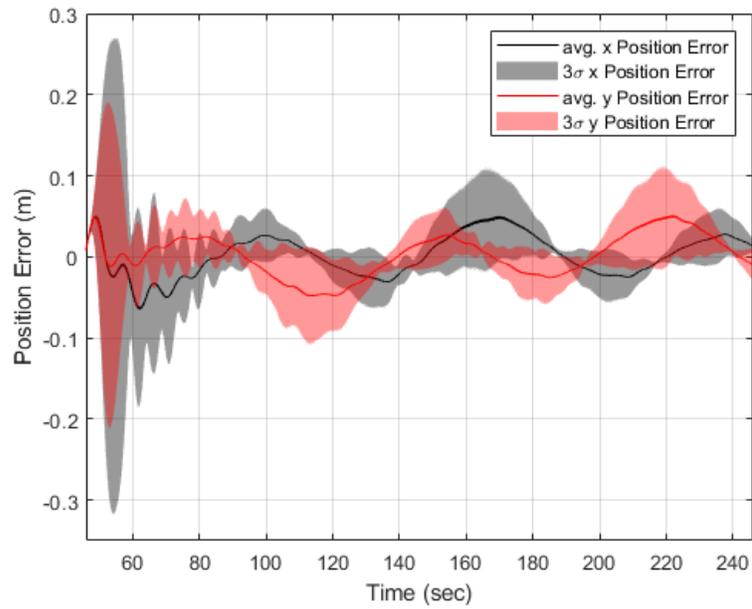


Figure 4.17: Nonlinearly Optimized Controller Position Error to the Command, with 3σ Standard Deviation

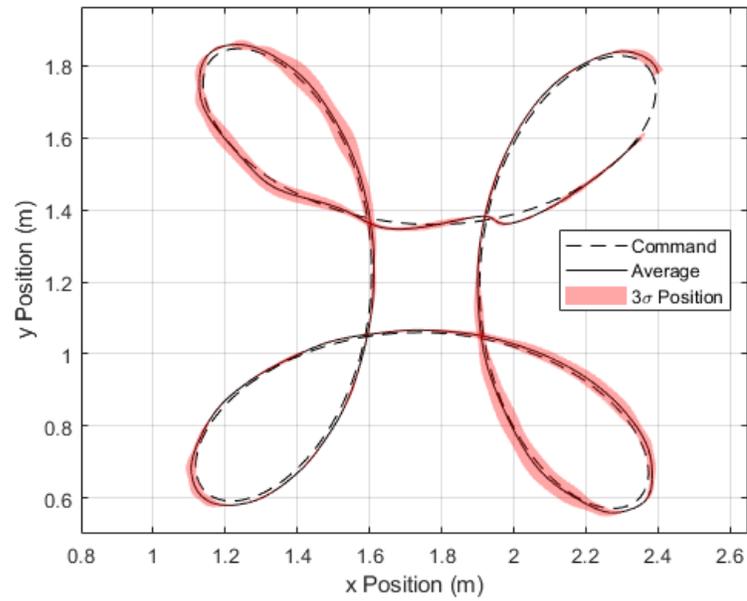


Figure 4.18: SaDE Optimized Controller Position Graph and 3σ Standard Deviation

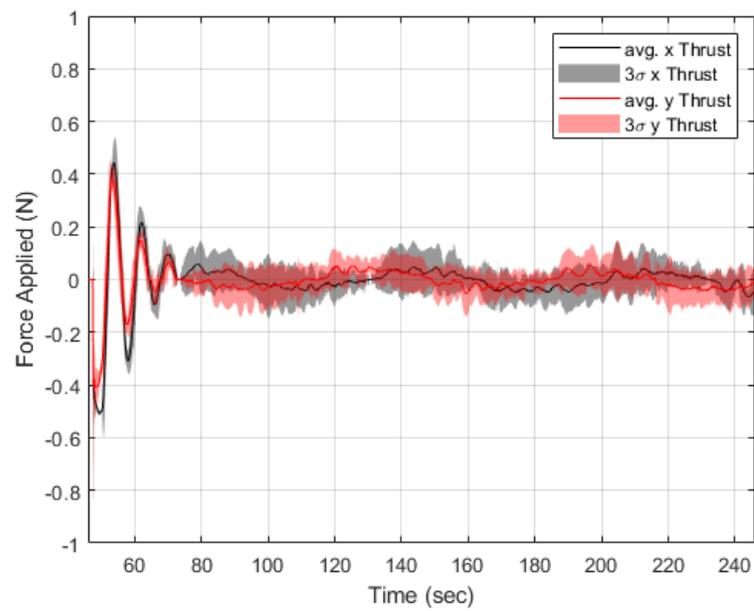


Figure 4.19: SaDE Optimized Controller Force Output, with 3σ Standard Deviation

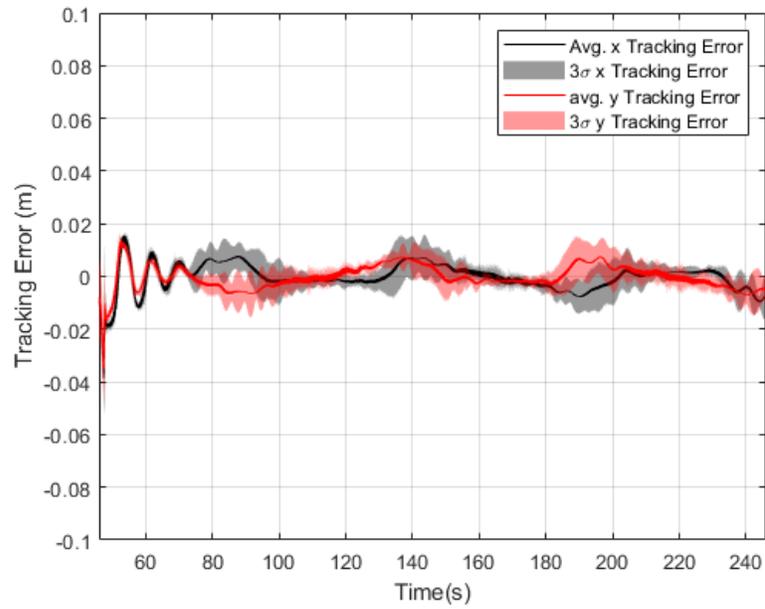


Figure 4.20: SaDE Optimized Controller Tracking Error, with 3σ Standard Deviation

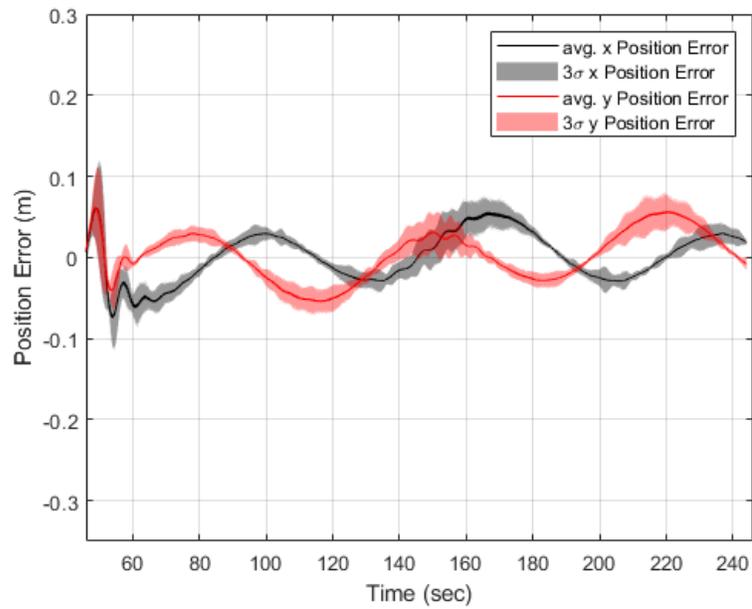


Figure 4.21: SaDE Optimized Controller Position Error to the Command, with 3σ Standard Deviation

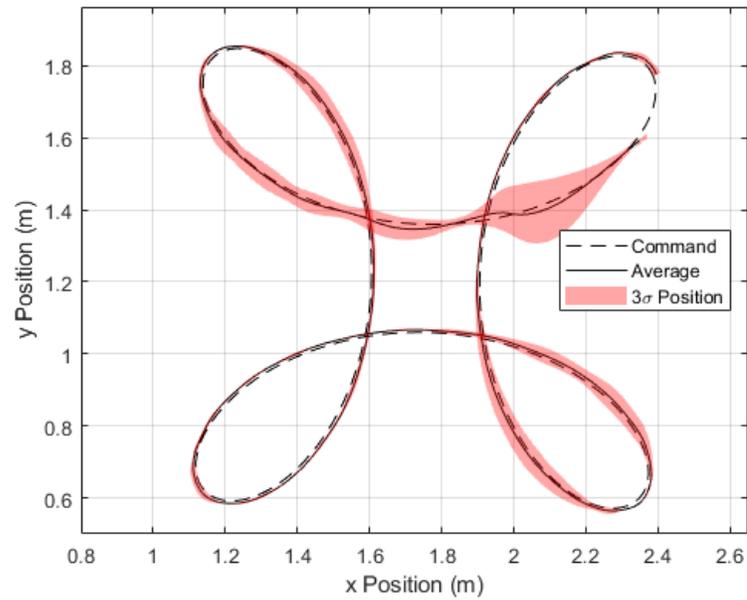


Figure 4.22: SPSO Optimized Controller Position Graph and 3σ Standard Deviation

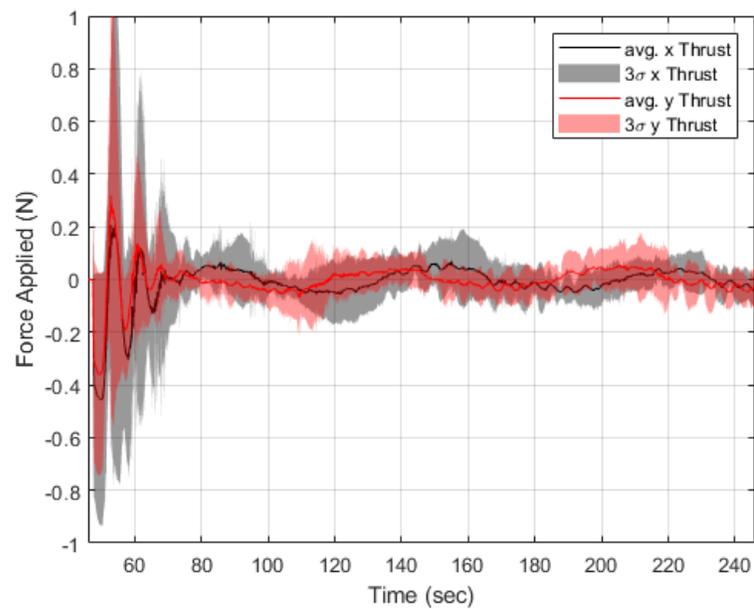


Figure 4.23: SPSO Optimized Controller Force Output, with 3σ Standard Deviation

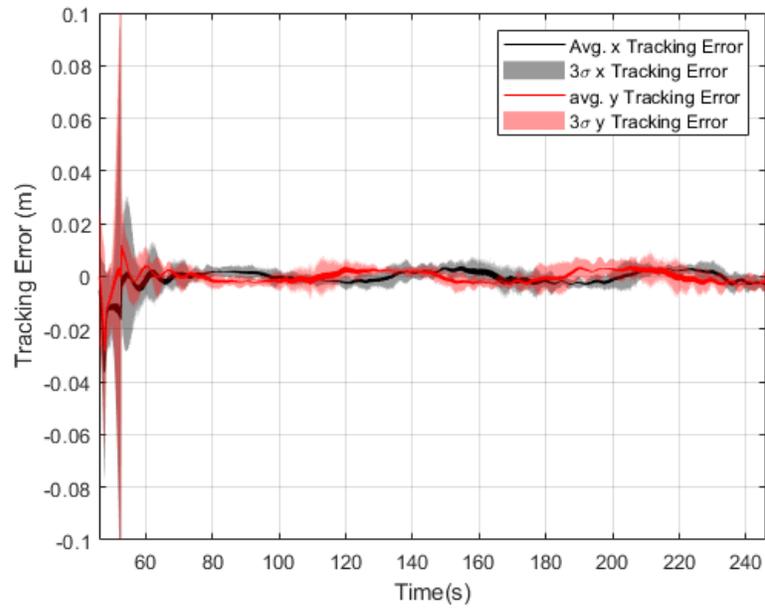


Figure 4.24: SPSO Optimized Controller Tracking Error, with 3σ Standard Deviation

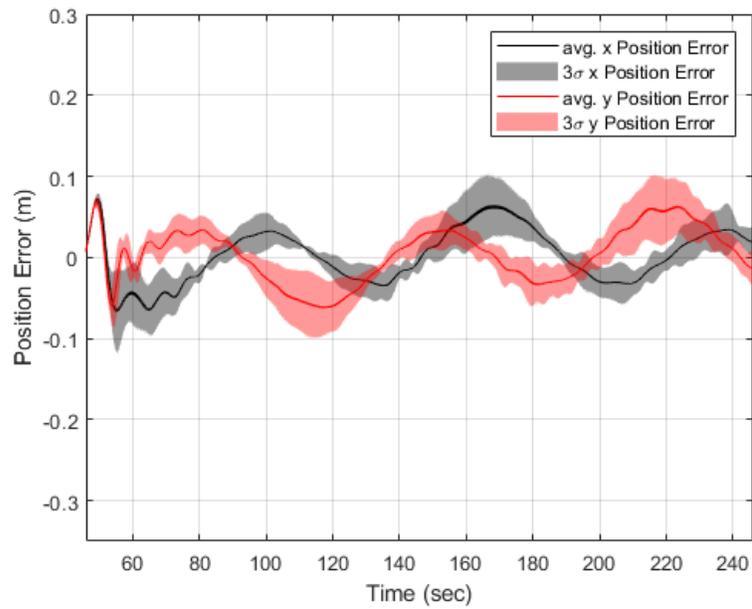


Figure 4.25: SPSO Optimized Controller Position Error to the Command, with 3σ Standard Deviation

4.4 SAC Disturbance Compensation Experimental Results

Disturbance compensation of a SAC with feedforward parallelization was compared to a similar SAC with disturbance compensation. The graphs for each of the pertinent system responses is demonstrated in Figs. 4.26 through 4.33.

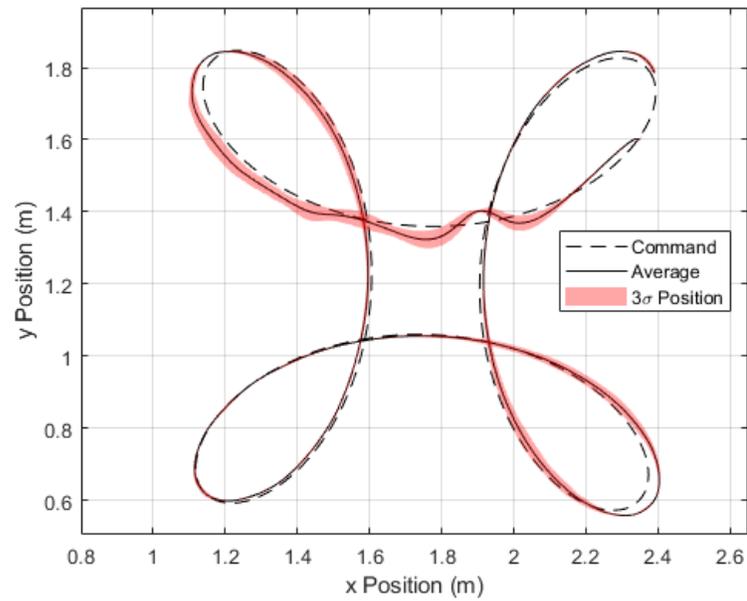


Figure 4.26: SAC Position Response Graph and 3σ Standard Deviation Under Linear Disturbance

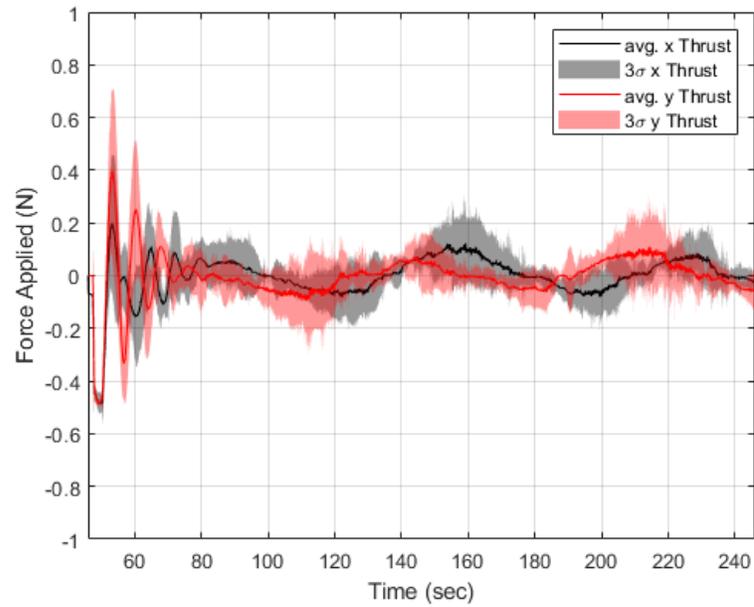


Figure 4.27: SAC Force Output, with 3σ Standard Deviation Under Linear Disturbance

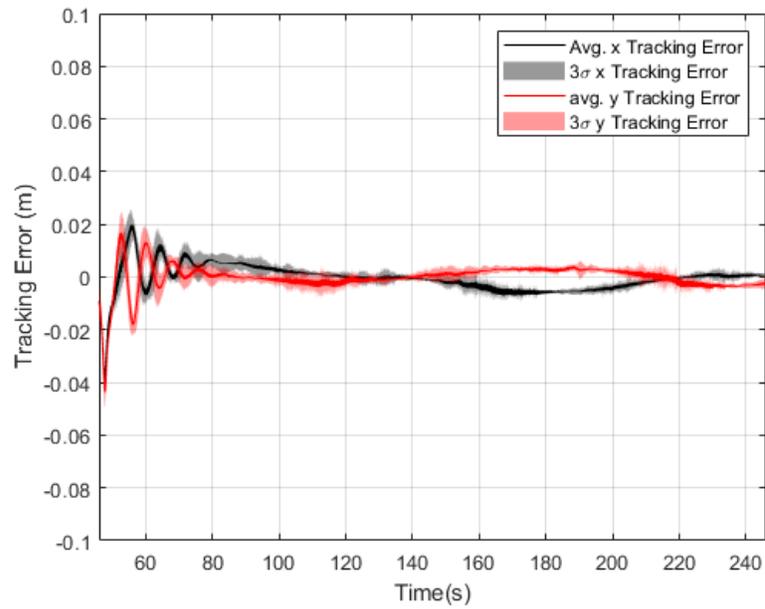


Figure 4.28: SAC Tracking Error, with 3σ Standard Deviation Under Linear Disturbance

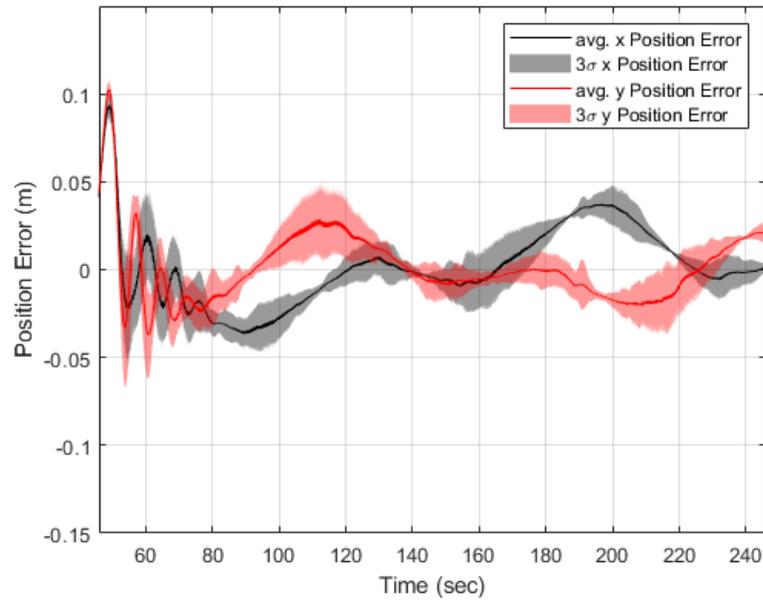


Figure 4.29: SAC Position Error to the Command, with 3σ Standard Deviation Under Linear Disturbance

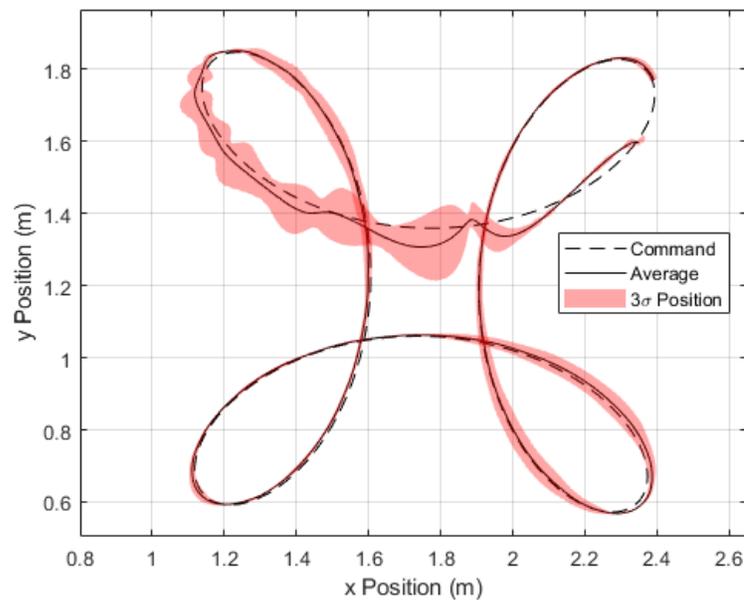


Figure 4.30: SAC Position Response Graph and 3σ Standard Deviation Under Linear Disturbance with Compensation

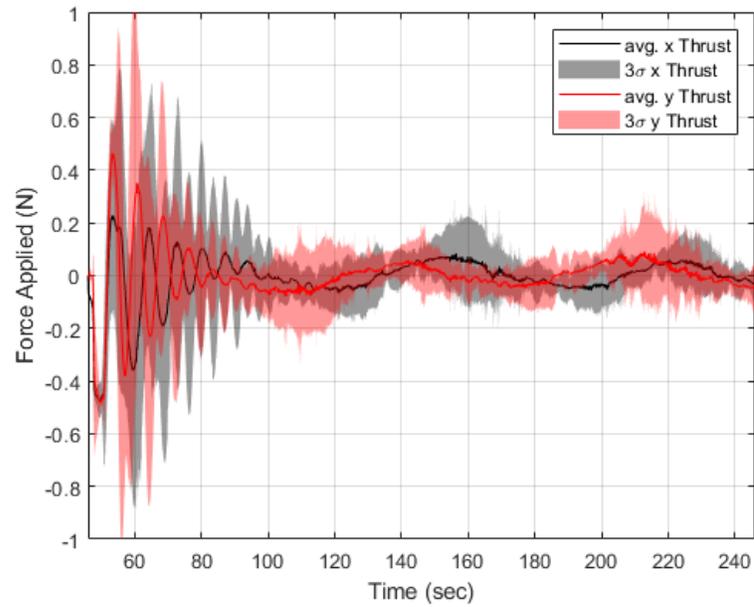


Figure 4.31: SAC Force Output, with 3σ Standard Deviation Under Linear Disturbance With Compensation

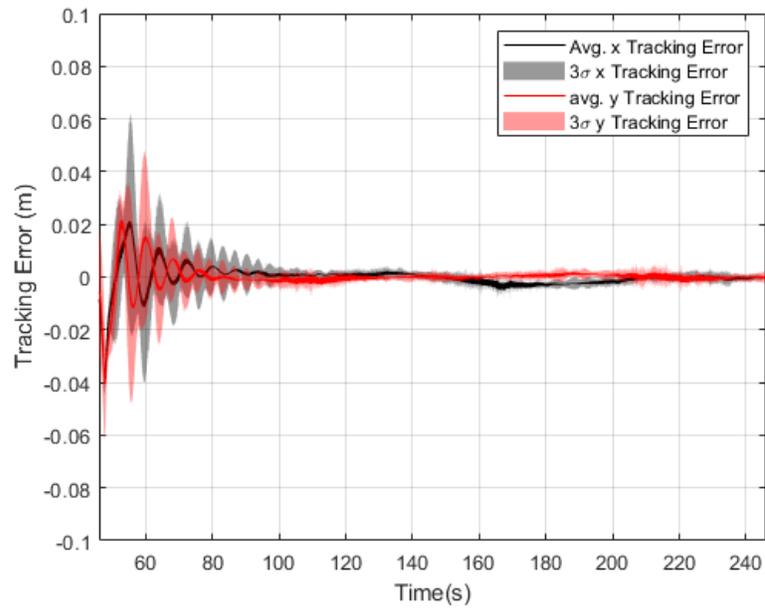


Figure 4.32: SAC Tracking Error to the Command, with 3σ Standard Deviation Under Linear Disturbance with Compensation

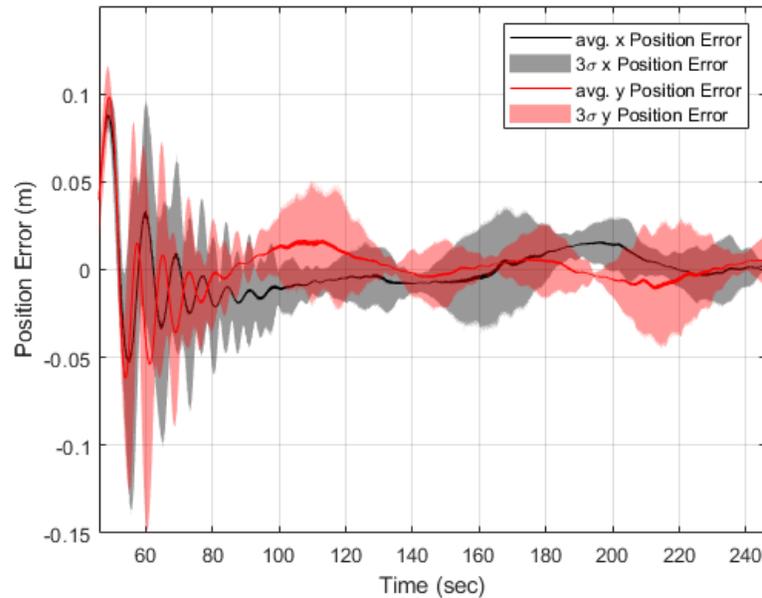


Figure 4.33: SAC Position Error, with 3σ Standard Deviation Under Linear Disturbance with Compensation

4.5 Discussion of Optimized SAC Results

The results of position tracking between LQR, manual SAC design, and optimized SAC controllers helps to quickly and clearly clarify the advantages and drawbacks of both LQR and SAC in the context of spacecraft proximity operations.

The positional tracking in Fig. 4.7 appears to be very close to the position command, alongside very tight 3σ position deviations along the track. In general the crosstrack error of the LQR controller appears very small. The force outputs of the LQR controller in Fig. 4.8 is small, between $\pm 0.05\text{N}$, with very few deviations. Small spikes in the force are present, likely due to noise and occasional position processing difficulties in the measurement apparatus. The command tracking of the LQR controller in Fig. 4.9 is quite small, but consistent. The command tracking is never less than 10 cm, but is sometimes as large as 22.5 cm. The LQR controller is consistently behind the cycloid command.

The manually designed SAC converges on the ideal response the fastest of any of the presented controllers. Its response in Fig. 4.10 shows that it stays close to the

command trajectory, even while adapting its gains. Its fast convergence is obvious in the system force output, shown in Fig. 4.11, which settles before 60 seconds. Small bursts are also visible in Fig. 4.11, which mirror the locations of increased deviation in Fig. 4.10. The tracking error in Fig. 4.12 stays small throughout. The error to the command, shown in Fig. 4.13, is considerably smaller than the LQR, and stays between ± 2 cm on average, with consistent deviation from that value due to bursts. In all runs, bursting increased the command tracking error by no more than 5 mm. The force commanded by the system is larger than in the optimized controllers, discussed below, which is likely the cause of the slightly higher cost of the manual design. It is possible that the decreased convergence time of the manual design may be similar to metaheuristic designs for cost functions that emphasize minimizing the command error over the control activation. The presented heuristics allow for the rapid development of an adaptive controller with desirable characteristics.

The cycloid position response of the nonlinearly optimized controller is visible in Fig. 4.14. Similar to what was observed in Sec. 2.6.3, nonlinear optimization was not able to determine SAC parameters that perform well compared to other controller designs. The use of the 3σ deviation on a set of 10 test runs means that the 96th percentile of runs must be included in the deviation. The red band in Fig. 4.14 coincides with the cross-track error of the worst run in the experiment. The response of the worst run was somewhat more oscillatory than other runs, which may have been caused by excessive noise, or delayed measurements. Convergence of the system to the ideal parameters was slow, and noise disproportionately affected the system response. Since the nonlinear parameter optimization was done without noise, it is likely that large gain adaptations were found for the system, resulting in the susceptibility to noise.

The slow convergence of the nonlinearly optimized SAC design is clear in the thrust response in Fig. 4.15, where oscillations in both the average and 3σ standard deviation thrust response take until 80 seconds to reach some sort of convergence. The nonlinearly optimized SAC design commands larger thrusts than the LQR controller, likely due to the ideal model chosen specifically to closely track the command input.

The tracking error of the nonlinearly optimized SAC in Fig. 4.16 remains between

± 2 cm once converged, suggesting a small error in the final signal. The command error is seen in Fig. 4.17, which is slightly larger than ± 2 cm once converged, however the difference is slight. More concerning is the increase in deviation up to ± 5 cm when approaching the lobes of the cycloid, denoted by maximal values of the x and y command error. Even the slow cycloid being performed by the system, taking a full three minutes to complete, the nonlinearly optimized controller is not able to consistently maintain small errors. An error of 10 cm can be enough to cause a collision during spacecraft proximity operations.

The SaDE and SPSO optimized controllers behave much better when compared to the nonlinearly optimized SAC. In all cases the metaheuristic optimized controllers had less deviation than the other SAC designs. The SaDE optimized SAC design cycloid performance is shown in Fig. 4.18. The SaDE optimized controller has a consistent response, with the typical performance often overlapping the command position. The SaDE response is still, however, rarely as consistent as the LQR response. The convergence time of the SaDE optimized SAC is much shorter than the convergence time for the nonlinearly optimized SAC, having fully converged by 75 seconds. Noise, however, is still increasing the deviation of the output force somewhat. The tracking error of the SaDE optimized SAC is never more than ± 2 cm, with deviations often less than ± 1 cm. The position command error of the SaDE optimized SAC shown in Fig. 4.21 is similar in magnitude to the nonlinearly optimized SAC response, with the significantly shorter convergence time and tighter deviation being distinct. The SaDE position error from the command is never more than ± 2.5 cm.

The SPSO optimized SAC achieves similar performance to the SaDE optimized SAC using different adaptation rates. Where SaDE prioritized the model state gains, SPSO prioritizes the tracking error feedback gain. The difference in control style is clear in the control responses. Figure 4.22 shows that the SPSO optimized SAC has larger cross-track deviations than the SaDE optimized SAC, however converges faster than the SaDE design. The reason for the performance becomes clear when observing the force graph in Fig. 4.23; SPSO maintains similar output thrusts to the SaDE design while also decreasing the magnitude of oscillations during convergence, alongside a minor decrease in convergence time. The SPSO optimized SAC takes

slightly shorter to converge, with oscillations ending after approximately 70 seconds. The force output of the SPSO optimized SAC is the most consistent of the SAC designs, with the lowest activations. The tracking error response in Fig. 4.24 has much lower deviations than the SaDE ones in Fig. 4.20, however this does not correspond to lower position error in the SPSO command tracking in Fig. 4.25. Slightly larger deviations are present in the SPSO command tracking. The SPSO metaheuristic optimization has designed a controller that has larger tracking error with lower control activations when compared with its SaDE optimized counterpart.

All of the optimized controllers made use of larger command forces than the LQR design, and consistently had larger deviations for both the command force and position error. The increase in force of the adaptive controllers is consistent with the increased command following demanded by the ideal model. All of the adaptive controller designs were able to achieve smaller command errors than the LQR controller. The LQR controller has the lowest cost of any of the controllers, as presented in Tab. 4.4, since it is the optimal linear controller for this system, and the SAC targets a different system response from the LQR response. Both of the metaheuristic optimization techniques were able to decrease the linear-quadratic cost, while nonlinear optimization was unable to do so.

4.6 Discussion of Disturbance Compensation Results

The results of the disturbance compensation experiments help to reinforce the theoretical results of disturbance rejection of a feedforward parallelized SAC.

The frequencies of the virtual disturbances used during experiments were explicitly chosen to be multiples of the cycloid angular frequency, so as to demonstrate the effects of the disturbance on the scale of the position command. Figure 4.26 clearly demonstrates the effect that the virtual disturbances have had on the trajectory tracking of the uncompensated SAC. The virtual disturbances have caused the position response of the uncompensated SAC to consistently miss the top left and bottom right cycloid lobe by a visible and significant margin. Other small deviations throughout the trajectory are visible where the disturbance has directly affected the ability of the SAC to track the command. The virtual disturbance has created a

visible and consistent effect on the trajectory tracking.

The 3σ deviations of the uncompensated SAC response are significantly smaller than the SPSO, SaDE, or nonlinearly optimized SAC designs. The effects of the virtual disturbance on the SAC response are almost invisible in the other graphs. Figure 4.27 appears similar to the force graphs for any of the other tests, and Fig. 4.28 appears similar to the other tracking error graphs, albeit with a slightly different shape but similar behaviour and lower deviations. It is not until observing the commanded position error in Fig. 4.29 that it becomes clear that the disturbance has had an effect on the response. As described in Sec. 3.4, the presence of a disturbance on an uncompensated SAC with feedforward parallelization has created a noticeable and consistent disturbance on the command tracking.

The deviations seen on the uncompensated SAC response are much smaller than the optimized SAC responses of the previous experiment. The smaller deviations are consistent with smaller gain adaptations, as well as longer adaptation times. During the 90 seconds it takes the uncompensated SAC to converge, it uses larger force commands and has larger tracking errors than the optimized controllers presented in the previous experiments. One of the side-effects of optimizing the SAC response is the larger deviations that come with larger SAC adaptation parameters, and similarly, with larger gains.

Disturbance compensation was included in the SAC design seen in Figs. 4.30 through 4.33. The position tracking seen in Fig. 4.30 is noticeably improved from the uncompensated case. Small deviations from the commanded path still exist, however it is clear that no systematic deviations in the position response exist once the controller gains have converged. Notably, the gain convergence time has increased, with the position response in Fig. 4.30 remaining oscillatory until the top left lobe is entered at almost 100 seconds. In the previous experiment, and in Fig. 4.26 oscillations had settled before entering the top left lobe. The force output of the compensated controller in Fig. 4.31 supports that gains take a longer time to converge in the disturbance compensated controller, with the force oscillations taking 100 seconds to converge. Once again the tracking error of the compensated case, demonstrated in Fig. 4.32, stays low. The disturbance compensated SAC has a somewhat smaller

tracking error than the uncompensated case.

The experiments of disturbance compensation for a SAC under feedforward parallelization have been able to show that disturbances acting on a compensated non-ASPR SAC lead directly to disturbances in the command tracking without similarly affecting the model tracking error. Uncompensated disturbances on a non-ASPR feedforward parallelized SAC will affect system performance in a way that cannot be managed by the adaptive controller. Disturbance compensation in SAC allows for disturbances to continue to be compensated by the SAC without loss of performance, even without amplitude or phase information of the disturbance, so long as the frequency information of the disturbance is known.

4.7 Summary of Experimental Results

The SAC design techniques inherited from previous research and described in Chapter 2, alongside the design techniques developed in Chapter 3 were implemented into two experimental setups to show the advantages and capabilities of SAC design techniques when compared to other controller designs. The ability of SAC to improve on command tracking using feedforward gain adaptation is demonstrated in contrast to traditional feedback control systems. The variability of SAC performance when large adaptation rates are implemented are also highlighted, alongside the tradeoffs an optimization method will make when attempting to maximize performance in a representative nonlinear simulation model. Disturbance rejection of a compensated and uncompensated non-ASPR SAC is also explored, where it is found that disturbances acting on a non-ASPR SAC design creates consistent command tracking performance loss, while disturbance compensation is able to successfully reject a disturbance of known frequency and unknown magnitude and phase without any loss of command tracking at the expense of longer convergence time.

Chapter 5

Implementation of SAC for Uncontrolled Spacecraft Capture and Control

Having been reviewed, developed, and experimentally verified, the various methods of SAC implementation are collected to present one cohesive motivating example; the uncontrolled spacecraft problem for a target of unknown mass properties is managed by a linear controller, and its performance compared to a SAC for the same system. The SAC is able to dramatically decrease cross-coupled disturbances due to the unknown system dynamics that cannot be compensated with traditional design techniques.

5.1 Introduction

Previous sections have outlined the theory behind simple adaptive controllers, applied that theory to improve implementations of SAC, and experimentally verified theoretical results in Carleton's SRCL SPOT. The presented research has been able to show that not only can adaptive control offer a more flexible controller formulation that can operate in the presence of plant uncertainty and variation, but also that the available tools allow adaptive controllers to outperform traditional linear design techniques while doing so.

The strong design foundations presented previously clarify many of the elements that are necessary for a designer to move from a theoretical adaptive controller for their system to an implemented adaptive controller with demonstrable performance. The developed SAC design techniques are leveraged in a presentation of their applicability through a simulation of a full six-degree-of-freedom uncooperative spacecraft rendezvous, docking, and post-docking control scenario.

A simulation of active control of an uncooperative and unknown spacecraft is undertaken here to demonstrate the possible future uses of adaptive controllers. The

uncontrolled spacecraft problem, fully outlined in Chapter 2, is made up of a controlled chaser and uncontrolled target spacecraft that are commanded to rendezvous and dock. After rigidly docking, the combined target-chaser system is controlled to change its position or orientation. In a future scenario, control of this system might be leveraged to actively deorbit debris without knowledge of the debris mass properties. A linear controller tuned for the problem is compared to a SAC optimized with meta-heuristics to determine if one is more suited to the task than the other. Along the way, several areas of interest for adaptive control research and performance optimization of adaptive control are highlighted.

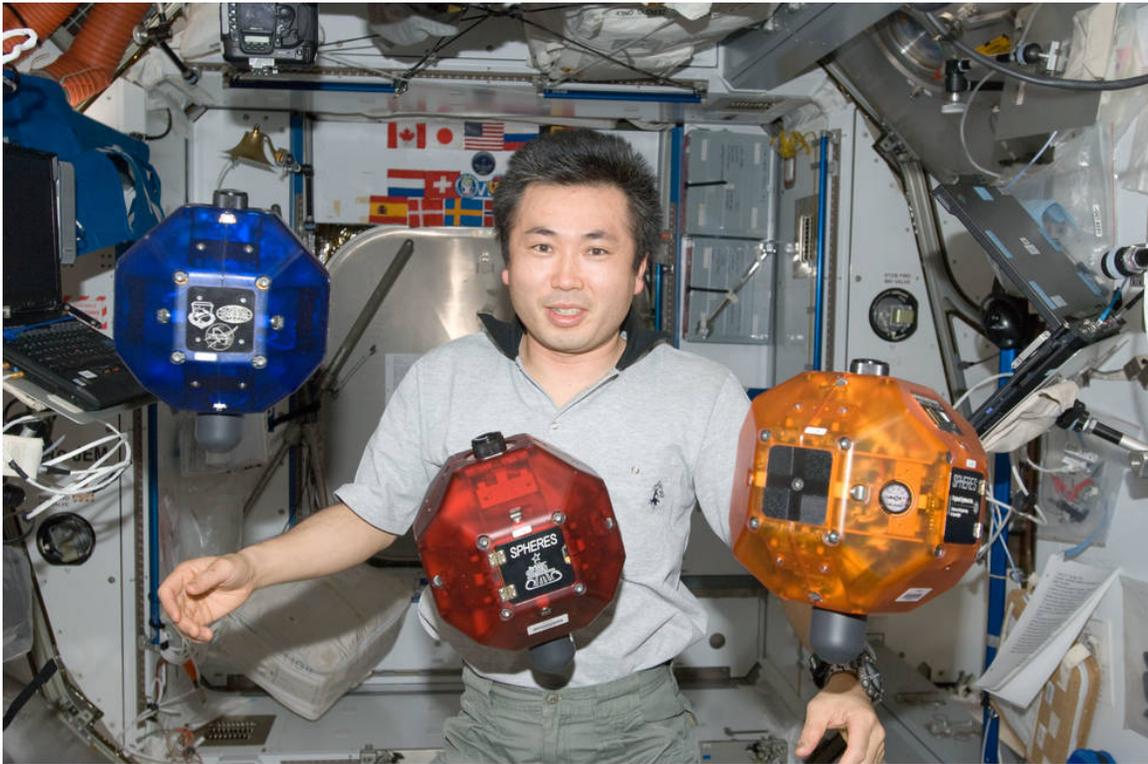
5.2 Problem Formulation

The equations of motion for the problem are described and detailed from Sec. 2.2 onwards. Section 2.2.3 describes the the formation flying equations that govern how two spacecraft orbit with respect to each other, Sec. 2.2.4 describes the equations of motion of the chaser spacecraft, and Sec. 2.2.5 derives and characterizes the equations of motion for the combined target-chaser system.

The primary purpose of simulating a debris deorbiting mission is to identify the strengths, weaknesses, and points of improvement of the SAC architectures and designs to a test mission profile. Several of the parameters used may be inaccurate for a mission profile with currently available spacecraft, however they are still representative of the challenges that will face a future active debris deorbit mission, as well as many other future missions that will rely on the ability to control time-varying systems with great accuracy using adaptive controllers.

5.2.1 Physical Parameters

The physical parameters of the target and chaser spacecraft are modelled on the SPHERES (Synchronized Position Hold Engage Reorient Experiment Satellites) spacecraft, which are used in the Massachusetts Institute of Technology space systems laboratory. The SPHERES spacecraft were successfully deployed to the ISS for use in spacecraft proximity operations testing and formation flying research and have also been used to test post-docking manoeuvres in experiments by James [75]. A reference



ISS020E019069

Figure 5.1: SPHERES Platforms on the ISS, Courtesy NASA¹

image of the SPHERES spacecraft is shown in Fig. 5.1.

The target and chaser have similar masses and moments of inertia. The target spacecraft has mass $m_t = 5.8450$ kg and moment of inertia

$$\mathbf{J}_t = \begin{bmatrix} 0.0289 & -0.0003 & 0.0011 \\ -0.0003 & 0.0560 & 0.0002 \\ 0.0011 & 0.0002 & 0.0588 \end{bmatrix} \text{ kg m}^2 \quad (5.1)$$

measured from the target center of mass. The outer shape of the target is roughly an 18-sided polyhedra that fits within a cube of edges 0.2m long. The docking port of the target is centered on one of the faces of that cube, and some additional hardware is included to allow for docking of the two SPHERES spacecraft, extending the docking port farther than 0.2m from the center of mass. The docking point of the SPHERES spacecraft measured relative to its center of mass in the body-fixed reference frame

¹<https://www.nasa.gov/spheres/the-history-of-spheres>

is given by James to be

$$\mathbf{r}_{a,t} = [-0.2097, 0.0062, 0.0178]^T \text{m} \quad (5.2)$$

The chaser spacecraft is identical to the target, with mass $m_c = 5.8450$ kg and moment of inertia

$$\mathbf{J}_c = \begin{bmatrix} 0.0289 & -0.0003 & 0.0011 \\ -0.0003 & 0.0560 & 0.0002 \\ 0.0011 & 0.0002 & 0.0588 \end{bmatrix} \text{kg m}^2 \quad (5.3)$$

measured from the chaser center of mass and identical docking point of

$$\mathbf{bor}_{a,c} = [-0.2097, 0.0062, 0.0178]^T \text{m} \quad (5.4)$$

The target spacecraft is uncontrolled, and begins with an angular velocity about one of its body frame axes. Many possible scenarios exist for the evolution of angular velocity of a spacecraft, the most common of which are well covered in Seweryn and Sasiadek [76]. For this scenario, only the simplest and most common scenario is used, where a target has been tumbling for some time without significant perturbation torques. The initial angular velocity of the target is

$$\boldsymbol{\omega}_{t0} = \begin{bmatrix} 0 \\ 0.01 \\ 0 \end{bmatrix} \quad (5.5)$$

whereas the initial angular velocity of the chaser is $\boldsymbol{\omega}_{c0} = \mathbf{0}$. The target begins in a 370 km altitude circular LEO, as such its initial position is

$$\mathbf{r}_{t0} = \begin{bmatrix} 6347000 + 370000 \\ 0 \\ 0 \end{bmatrix} \quad (5.6)$$

in meters in the ECI reference frame, whereas its initial velocity corresponds to the circular orbit velocity at that altitude, as

$$\mathbf{v}_{t0} = \begin{bmatrix} 0 \\ 7685.7 \\ 0 \end{bmatrix} \quad (5.7)$$

in meters per second. The chaser begins with the same velocity at a separation of 15 km in the inertial z direction, and 30 km in the inertial y direction, given by the initial positions of

$$\mathbf{r}_{c0} = \begin{bmatrix} 6347000 + 100000 \\ -30000 \\ -15000 \end{bmatrix} \quad (5.8)$$

in meters in the ECI reference frame. The initial velocity of the chaser is identical to the target velocity, given by

$$\mathbf{v}_{c0} = \begin{bmatrix} 0 \\ 7685.7 \\ 0 \end{bmatrix} \quad (5.9)$$

in meters per second in the ECI frame.

The orbit of the target is kept in the inertial x-y plane in order to simplify analysis. The LVLH cross-track direction thus corresponds to the inertial z direction, and the LVLH reference frame is a simple z rotation by the angular difference between the target velocity and the inertial x axis.

5.2.2 Combined Target-Chaser System Parameters

The change in system properties due to docking of a controlled target and uncontrolled chaser spacecraft are discussed in Sec. 2.2.5. In the derivations of Sec. 2.2.5 it is shown that the combined system has different mass and moment of inertia properties from the chaser-only system, as well as cross coupling between the torque and position responses for the combined system.

The mass of the combined system is equal to the sum of the individual target and chaser mass, while the combined moment of inertia is found through application of

the parallel axis theorem and moment of inertia rotation found in Eqs. (2.28) through (2.32). The moment of inertia of the combined system is found to be

$$m_{cmb} = 11.69\text{kg}, \quad \mathbf{J}_{cmb} = \begin{bmatrix} 0.0620 & -0.0158 & -0.0414 \\ -0.0158 & 0.6298 & -0.0009 \\ -0.0414 & -0.0009 & 0.6321 \end{bmatrix} \text{kg m}^2 \quad (5.10)$$

which are consistent with the value quoted in James [75].

Additionally, the cross-coupling between axes is due to the offset from the center of mass location and the original chaser center of mass. The position offset after docking is found through Eq. (2.34) to be

$$\bar{\mathbf{x}} = \begin{bmatrix} -0.2097 \\ 0.0062 \\ 0.0178 \end{bmatrix} = \mathbf{r}_{a,t} = \mathbf{r}_{a,c} \quad (5.11)$$

The relevant torque due to positional force commands being the cross product of the COM offset detailed above with the force command. Similarly the acceleration disturbance due to torque commands is a product of the angular acceleration with the COM offset. The linear transfer function approximation of the cross-coupling effects is given in Sec. 2.2.5 by Eq. (2.38).

Thrust and torque saturations are present on the system. Although James [75] cites lower values, a thrust saturation of ± 0.76 N will be used in this simulation to allow for reasonable rendezvous of the spacecraft, for which the original SPHERES platforms were not intended. A saturation of 0.0736 Nm is used for the maximum torque.

5.3 Measurement and Command Signals

A linear controller is implemented for the rendezvous and docking of the chaser with the target. Once the target and chaser are docked, the performance of the linear controller to various positional and attitude commands can then be observed. An

adaptive controller using the SAC architecture will also be created to handle rendezvous and docking of the target and chaser. Since adaptive control is able to improve performance through time, a series of “convergence commands” are given to the controller to allow the adaptive system to converge. Once the controller has converged on a final controller, the adaptive controller will then be provided the same commands that were provided to the linear controller to compare the performance of both.

Both the linear and adaptive controller make use of the LVLH reference frame to manage relative position errors. All orientation commands are given in terms of quaternions representing a rotation between the ECI and desired body-fixed orientation. Any user-generated commands are input in terms of Euler angles between the ECI and body-fixed reference frame, then converted to their equivalent quaternion. The error quaternion is used to determine the smallest angular arc in terms of Euler angles between the current spacecraft rotation and the desired rotation. The values of the Euler angle errors can then be used to generate body-frame torques to control orientation. For the desired quaternion \mathbf{q}_{cmd} and current measured quaternion \mathbf{q}_m , the difference quaternion \mathbf{q}_e is determined through

$$\mathbf{q}_e(t) = \mathbf{q}_{cmd}(t)\mathbf{q}_m^{-1}(t) \quad (5.12)$$

where quaternion multiplication is used, and the superscript “ -1 ” denotes the quaternion inverse. The error quaternion can be converted into a set of Euler angles for use in determining angular error. The Euler angular errors are called ϕ_e for roll error, θ_e for pitch error, and ψ_e for yaw error, all in rad. Rotational rates are determined from a quaternion through

$$\begin{bmatrix} 0 \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = 2\dot{\mathbf{q}}_m(t)\mathbf{q}_m^*(t) \quad (5.13)$$

where dot notation is used to denote an elementwise derivative, and the superscript $*$ denotes the quaternion conjugate.

It is assumed that guidance and navigation are able to measure the LVLH position of the chaser, along with the rotation of the target, inertial Euler angles of the chaser, and location of the target docking point.

5.4 Path Planning

The mission begins with the target in a circular orbit at an altitude of 370 km, while the chaser begins with a separation of -15 km in the off-track direction and -30 km in the on-track direction. A position command is generated for the spacecraft to follow to rendezvous with the target. Once the chaser has reached a separation of less than 200m from the target, the docking phase will begin. During the docking phase the chaser will approach the target, eventually matching its rotation and meeting the docking point. Once the docking point of the target and chaser spacecraft are within 1 mm of separation, and have relative velocities of less than 5 mm/s, the two spacecraft are considered docked and the dynamics will shift to the combined rigid body system, with mass m_{cmb} , moment of inertia \mathbf{J}_{cmb} , disturbance torques described in Eq. (2.33), and acceleration due to angular accelerations described in Eq. (2.36). After docking, position and orientation commands can now be sent to the system to determine the controller performance in the presence of large uncertainty in the target mass properties.

5.4.1 Rendezvous Path-Planning

Commands that result in efficient rendezvous are produced using the path optimization described by Lu and Liu [68], and detailed in Sec. 2.6.2. The trajectory optimization script determines thrust profiles to move a chaser towards a target spacecraft with maximum thrust limitations for a given rendezvous time and approach cone. The output optimized LVLH trajectory is used as the command for the rendezvous sequence of the simulation.

Optimal trajectory planning is used to reduce the control problem under the nonlinear formation flying Eqs. (2.17) through (2.19) to simple double-integrator position control.

The YALMIP optimizer was used in MATLAB[®]2021a to solve the rendezvous

and docking problem through the second-order conic optimization problem presented in Sec. 2.6.2 for the proposed mission. The final inertial trajectories are used as rendezvous and docking commands.

The spacecraft dock at 3000 seconds, during which the chaser reaches a target separation of 200 m after 2000 seconds under thruster saturation of 0.1 N. During docking, the chaser approaches the target at a maximum velocity of 0.3 m/s, in an approach cone of 10 degrees, and thruster contamination cone of 10 degrees. The final optimized LVLH position trajectory is shown in Fig. 5.2. Rotations of the target are not considered during optimization, and are instead managed by command mixing during the docking phase. The trajectory solution of this optimization is called $\mathbf{r}_{cmd,r}(t)$, the time-varying position command during rendezvous.

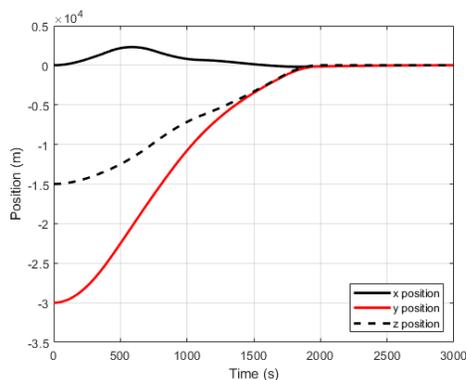


Figure 5.2: LVLH Optimized Rendezvous Trajectory

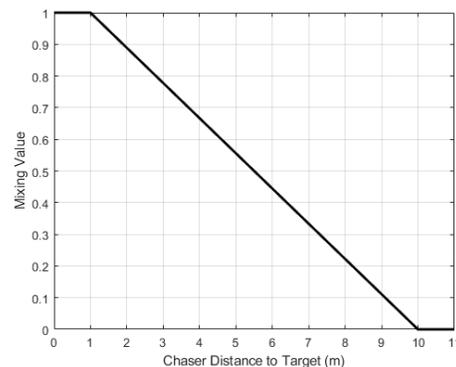


Figure 5.3: Docking Command Mixing

5.4.2 Docking Phase

At the moment of docking the chaser docking axis must be collinear with the target docking axis. If the rotation of the target is given by the quaternion \mathbf{q}_t , then the chaser orientation quaternion \mathbf{q}_c at the moment of docking is found to be

$$\mathbf{q}_{cmd,d}(t) = (0 + 0i_q + 1j_q + 0k_q)\mathbf{q}_t(t) \quad (5.14)$$

where i_q , j_q , and k_q are the three basic quaternions, and where $\mathbf{q}_{cmd,d}$ now points the chaser docking axis in the opposite direction of the target docking axis. The chaser

COM will also need to be positioned along the target docking axis to ensure that the chaser docking point approaches the target docking point. In the case of a rotating target, call ϕ , θ , and ψ the Euler angles for the roll, pitch, and yaw of the target body-fixed reference frame relative to the LVLH reference frame. That is to say

$$\vec{\mathcal{F}}_t = \mathbf{R}(\phi, \theta, \psi)\vec{\mathcal{F}}_L \quad (5.15)$$

For the rotation matrix $\mathbf{R}(\phi, \theta, \psi) \in \mathbb{R}^{3 \times 3}$ between the LVLH and target body-fixed reference frame. For the target docking point displacement in LVLH of $\mathbf{r}_{a,t}$, and chaser docking point displacement $\mathbf{r}_{a,c}$, the chaser position command during docking \mathbf{r}_{cmd} corresponds to

$$\mathbf{r}_{cmd,d}(t) = \mathbf{r}_{a,t} - \mathbf{r}_{a,c} + \mathbf{R}(\phi, \theta, \psi)\mathbf{r}_{cmd,r}(t) \quad (5.16)$$

which is simply a rotation of the original docking command \mathbf{r}_{cmd} to match the current rotation of the target, given by \mathbf{R} , and shifted by the docking axis displacements to ensure that the docking points, and not the spacecraft COMs, meet at the final time.

Finally, the rendezvous orientation command $\mathbf{q}_{cmd,r}$ and position commands $\mathbf{r}_{cmd,r}$ are transitioned between the rendezvous and docking behaviours $\mathbf{q}_{cmd,d}$ and $\mathbf{r}_{cmd,r}$ using a linear mixing function that goes from a value of 0 at 10 m separation, and 1 at 1 m separation. The mixing variable and mixed commands are found using the equation

$$z_m(t) = \max\left(\min\left(\frac{\|\boldsymbol{\rho}(t)\|}{9} + \frac{8}{9}, 1\right), 0\right) \quad (5.17)$$

$$\mathbf{q}_{cmd}(t) = z_m(t)\mathbf{q}_{cmd,d}(t) + (1 - z_m(t))\mathbf{q}_{cmd,r}(t) \quad (5.18)$$

$$\mathbf{r}_{cmd}(t) = z_m(t)\mathbf{r}_{cmd,d}(t) + (1 - z_m(t))\mathbf{r}_{cmd,r}(t) \quad (5.19)$$

The graph for the transition variable z_m as a function of distance is demonstrated in Fig. 5.3

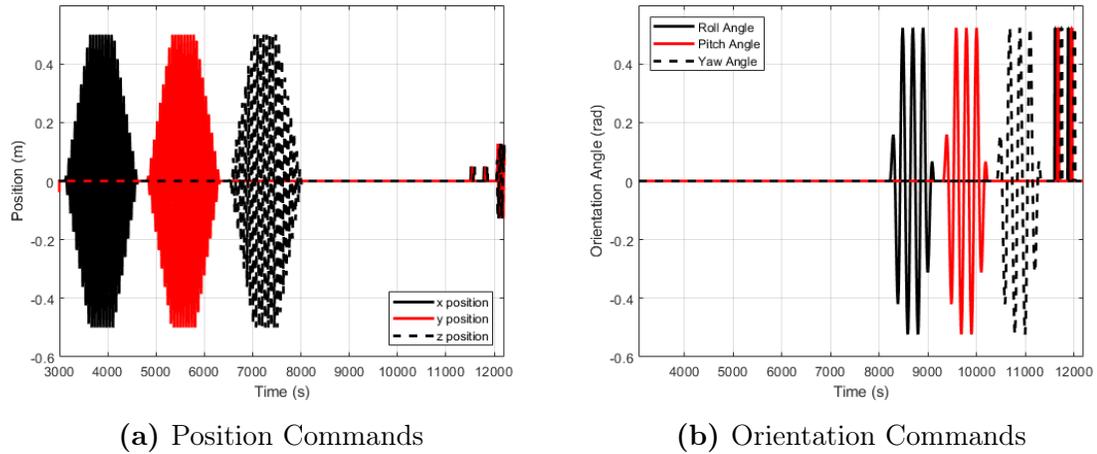


Figure 5.4: Position and Orientation Commands After Docking

5.4.3 Post-Docked Behaviour

After docking has occurred, simple sinusoidal commands are sent to each of the control channels to allow for SAC gains to update. An oscillation of frequency 0.1 rad/s and amplitude 0.5 m is sent to each of the position controllers, and a sinusoid of frequency 0.03 rad/s and amplitude $\pi/6$ rad is sent to each of the orientation controllers. Once gains have converged, step commands of 0.05 m and $\pi/6$ rad are sent to the position and orientation controllers, followed by a cycloid position command governed by the equations

$$\mathbf{r}_{cmd}(t) = \begin{bmatrix} 0 \\ 0.5 \cos(0.03t) + 0.35 \sin(3 \cdot 0.03t) \\ 0.5 \sin(0.03t) + 0.35 \cos(3 \cdot 0.03t) \end{bmatrix} \quad (5.20)$$

The command timeseries sent to both the position and orientation controllers are demonstrated in Fig. 5.4.

5.5 Linear Controller Design

To maintain performance through the critical docking phase of flight, and since mass properties of the target are unknown, linear controller design must emphasize the

chaser-only system. In particular, larger gains that might correspond to improved performance in the combined system will also degrade performance during rendezvous and docking. Independent controllers are designed for the position and orientation control of the chaser, with any performance degradation of the combined system being unavoidable due to the unknown combined system properties.

Position control is achieved through a PD controller with gain matrices

$$\mathbf{K}_{p,p} = 9.4869\mathbf{I}_3, \mathbf{K}_{d,p} = 19.3548\mathbf{I}_3 \quad (5.21)$$

such that

$$\mathbf{u}_{p,p}(t) = \mathbf{K}_{p,p}\mathbf{e}_p(t) + \mathbf{K}_{d,p}\dot{\mathbf{e}}_p(t) \quad (5.22)$$

where \mathbf{e}_p is the positional error in the LVLH reference frame, and $\mathbf{u}_{p,p}$ is the thrust command for each of the three linear directions in the LVLH reference frame.

A PD controller is designed for orientation control. The error quaternion is calculated following Eq. (5.12). The quaternion error is then converted to Euler angles to provide a linear representation of the orientation error. A proportional gain of $K_{p,\theta} = 0.5 \text{ Nm/rad}$ is used alongside a derivative gain of $K_{d,\theta} = 1 \text{ Nms/rad}$.

Formatted as a three-degree-of-freedom controller for the orientation, the gains on the linear angular error controller become the matrices

$$\mathbf{K}_{p,\theta} = 0.5\mathbf{I}_3, \mathbf{K}_{d,\theta} = \mathbf{I}_3 \quad (5.23)$$

such that

$$\mathbf{e}_\theta(t) = \begin{bmatrix} \phi_e(t) \\ \theta_e(t) \\ \psi_e(t) \end{bmatrix} \quad (5.24)$$

$$\mathbf{u}_{p,\theta}(t) = \mathbf{K}_{p,\theta}\mathbf{e}_\theta(t) + \mathbf{K}_{d,\theta}\dot{\mathbf{e}}_\theta(t) \quad (5.25)$$

The linear controller design separates the position and orientation errors, and as

such the linear controller design corresponds to a diagonal MIMO system with inputs of the position and angular errors, and outputs of the body forces and moments to be actuated by the thrusters.

5.6 SAC Design

The recommendations for SAC design described in Chapter 3 will be followed here to produce a SAC design that is able to significantly improve on the performance of the linear controller when managing the unknown combined system. Not only can SAC determine feedforward terms to compensate for command responses before they happen, SAC can also determine gains for cross-coupled system responses. By implementing the SAC as a MIMO controller for both position and orientation, the adaptation mechanism can be used to determine the cross-coupling between the position and orientation response of the combined system automatically. The SAC can determine which set of gains minimize the disturbance from position to orientation and vice-versa without any additional knowledge of the system.

5.6.1 Feedforward Parallelization

To begin the design, it is noted that neither the position nor orientation response is ASPR. The inverse of a stabilizing linear controller for position and orientation is placed diagonally into state-space form to create

$$\mathbf{H}^{-1}(s) = \begin{bmatrix} \frac{1}{38.71s+18.97}\mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \frac{1}{s+0.5}\mathbf{I}_3 \end{bmatrix} \quad (5.26)$$

which renders the augmented system ASPR. With the inclusion of the feedforward parallelization the augmented plant will be ASPR and a SAC can be used to stabilize the system.

5.6.2 Feasible Model Development

An ideal model is created for the system, which is then modified to produce a feasible model for the SAC to match. Since independent position and orientation responses are desired, the two models will be run independently, corresponding to a diagonal MIMO system for inputs of position and orientation command, alongside outputs of desired position response and desired orientation response.

The ideal model for the position response will correspond to a second-order system response. The ideal model used for the position response is characterized by the second-order transfer function, and equivalent state-space system

$$\mathbf{M}_p(s) = \frac{26.6^2}{s^2 + 2 \cdot 0.6 \cdot 26.6s + 26.6^2} \mathbf{I}_3 \quad (5.27)$$

$$\mathbf{A}_{m,p} = \begin{bmatrix} \mathbf{0}_3 & \mathbf{I}_3 \\ -711.1\mathbf{I}_3 & -32\mathbf{I}_3 \end{bmatrix}, \quad \mathbf{B}_{m,p} = \begin{bmatrix} \mathbf{0}_3 \\ 711.1\mathbf{I}_3 \end{bmatrix}, \quad (5.28)$$

$$\mathbf{C}_{m,p} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0}_3 \end{bmatrix}, \quad \mathbf{D}_{m,p} = \mathbf{0}_3 \quad (5.29)$$

Since the ideal model is a linear system, it follows that for large position displacement commands large thrust commands would be required. To minimize the tracking error a feasible model is created using knowledge of the system nonlinearities.

The chaser system maximum thrust is 0.76 N. It follows then that a feasible model reference should have a maximum acceleration of 0.76 N divided by the maximum system mass, which at worst case is rounded up to be 15 kg. A satisfying controller is created following the suggestion in Sec. 3.3.2 to create a simple nonlinear guidance that incorporates the maximum chaser acceleration.

The satisfying controller that ensures the feasible model approaches the ideal model is a simple PD controller with $\mathbf{K}_{p,s,p} = 12.5\mathbf{I}_3$ and $\mathbf{K}_{d,s,p} = 20\mathbf{I}_3$. The position satisfying controller is shown in Fig. 5.5.

Construction of an ideal and feasible model must then be repeated for the orientation system response. Similar to the position model, a second-order response is used for the orientation ideal model. The parameters used for the orientation response

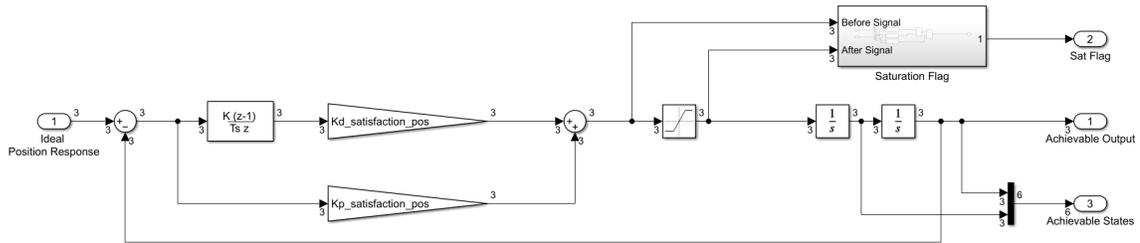


Figure 5.5: Position Satisfying Controller

ideal model are an angular frequency of $\omega = 4$ rad/s, and a damping coefficient of $\zeta = 1$. There are, however, some issues in implementing the orientation ideal model.

For the controller to work, it is necessary that the ideal model be composed of a linear state-space system, which in turn must be composed of linear inputs, outputs, and states. That is to say, we require an ideal model that is $\mathbf{M} : \mathbb{R}^6 \rightarrow \mathbb{R}^6$. However, orientations occur in the circular 3-space, denoted \mathbb{S}^3 . Similar to the orientation PD controller, the measured signal for the adaptive orientation controller will be the angular errors θ_e , ϕ_e , and ψ_e , which will be made to follow the ideal model angular errors. By making the output of the ideal model angular errors, the tracking error calculation stays linear and always corresponds to the shortest arc between the orientation measurement and the command. The input of the ideal model must also be linear, however. The ideal model will provide a conversion from angular rate commands (which are linear) to the ideal angular error.

To create the ideal model from angular rates to angular errors, a second-order linear transfer function of “linear angle command to desired linear angle” will be converted into a linear transfer function of “angular rate commands to desired angular error”.

First, a second order function from linear angle commands $\Theta_c(s)$ to desired angular response $\Theta_m(s)$ is created.

$$\frac{\Theta_m(s)}{\Theta_c(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (5.30)$$

The second-order feedback system is then rearranged such that the error signal is now the output, to create the transfer function from angle commands $\Theta_c(s)$ to

angular error $\Theta_{e,m}(s)$

$$\frac{\Theta_{e,m}(s)}{\Theta_c(s)} = \frac{s^2 + 2\zeta\omega_n s}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (5.31)$$

and finally, the angle command is substituted with an angular rate command $\Omega_c(s)$, to create a transfer function $M(s) : \mathbb{R} \rightarrow \mathbb{R}$ of

$$\frac{\Theta_{e,m}(s)}{\Omega_c(s)} = \frac{s + 2\zeta\omega_n}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (5.32)$$

The state-space system corresponding to this model, repeated for each axis of rotation, is given by

$$\mathbf{A}_{m,\theta} = \begin{bmatrix} -8\mathbf{I}_3 & -16\mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 \end{bmatrix}, \quad \mathbf{B}_{m,\theta} = \begin{bmatrix} \mathbf{I}_3 \\ \mathbf{0}_3 \\ \mathbf{0}_3 \end{bmatrix}, \quad (5.33)$$

$$\mathbf{C}_{m,\theta} = \begin{bmatrix} \mathbf{I}_3 & 8\mathbf{I}_3 & \mathbf{0}_3 \end{bmatrix}, \quad \mathbf{D}_{m,\theta} = \mathbf{0}_3 \quad (5.34)$$

A feasible model is constructed for the orientation response similarly to the feasible model for position response. The nonlinearity for this response is the maximum spacecraft torque of 0.0736 Nm. A maximum angular acceleration for the system was calculated using the worst case inertia

$$J_{worstcase} = \mathbf{I}_3 \quad (5.35)$$

The satisfying controller for the feasible model is a PD controller with proportional gain of $\mathbf{K}_{p,s,\theta} = 10\mathbf{I}_3$ and derivative gain $\mathbf{K}_{d,s,\theta} = 20\mathbf{I}_3$. The orientation satisfying controller is shown in Fig. 5.6

The final full linear model is constructed from the independent position and rotation ideal models as

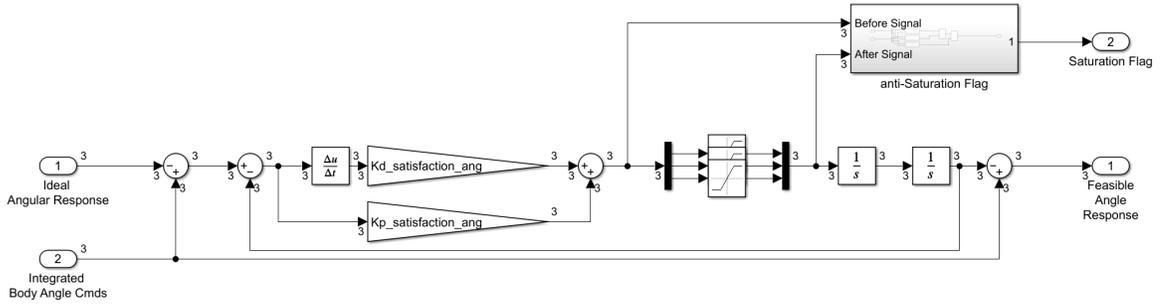


Figure 5.6: Orientation Satisfying Controller

$$\mathbf{A}_m = \begin{bmatrix} \mathbf{A}_{m,p} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{m,\theta} \end{bmatrix}, \quad \mathbf{B}_m = \begin{bmatrix} \mathbf{B}_{m,p} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_{m,\theta} \end{bmatrix}, \quad (5.36)$$

$$\mathbf{C}_m = \begin{bmatrix} \mathbf{C}_{m,p} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{m,\theta} \end{bmatrix}, \quad \mathbf{D}_m = \mathbf{0} \quad (5.37)$$

5.6.3 MIMO SAC Design

With the feedforward parallelization and feasible controller models for both the position and orientation response implemented, it remains to choose the adaptation rates for the SAC implementation. The final SAC is a MIMO system that controls the 6DOF system using three degrees of position control and three degrees of orientation control.

When implemented as a controller for a mission, there is no reason not to improve performance by using good initial gains when they are available. The gains used at the beginning of the simulation are

$$\mathbf{K}_{eI0} = \begin{bmatrix} 58.1124 & -0.0388 & 0.0811 & -0.7189 & -0.0303 & -3.0966 \\ -0.0388 & 58.2002 & -0.0232 & -1.0205 & -0.0454 & -2.5112 \\ 0.0811 & -0.0232 & 58.3125 & 1.3434 & 0.5394 & 0.1538 \\ -0.0070 & -0.0099 & 0.0130 & 10.8446 & -0.0239 & 0.1018 \\ -0.0003 & -0.0004 & 0.0052 & -0.0239 & 10.9315 & -0.1335 \\ -0.0300 & -0.0243 & 0.0015 & 0.1018 & -0.1335 & 11.0765 \end{bmatrix}$$

$$\mathbf{K}_{xI0} = \begin{bmatrix} 0 & -0.0195 & 0.0003 & -0.3084 & 0.0416 & 0.0287 & 8.9853 & -0.0021 & 0.0019 & 0.3123 & 0.5365 & -0.2771 & 0 & 0 & 0 \\ 0 & 0.0249 & -0.0004 & 0.2796 & -0.1297 & 0.0238 & -0.0168 & -0.0064 & -0.0062 & -0.1630 & -0.3922 & -0.3723 & 0 & 0 & 0 \\ 0 & 0.0472 & -0.0004 & 0.3982 & 0.0825 & -0.2265 & 0.0338 & 0.0100 & 0.0153 & 0.3891 & 0.6437 & 0.7615 & 0 & 0 & 0 \\ 0 & 1.6761 & -0.0141 & -0.0075 & 0.0035 & 0.0015 & 0.0001 & 0 & -0.0001 & 0.0008 & 0.0128 & -0.0058 & 0 & 0 & -0.0001 \\ 0 & -0.9652 & 0.0082 & 0.0035 & -0.0012 & -0.0009 & 0 & 0 & 0 & 0.0006 & 0.0012 & -0.0021 & 0 & 0 & 0.0001 \\ 0 & 0.3306 & -0.0028 & -0.0013 & -0.0002 & 0.0005 & -0.0002 & 0 & 0 & 0 & -0.0085 & 0.0032 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{K}_{uI0} = \begin{bmatrix} 0 & 0 & 0 & -0.0337 & 0.0141 & -0.0046 \\ 0 & 0 & 0 & -0.0561 & -0.0372 & -0.0334 \\ 0 & 0 & 0 & 0.1212 & 0.0608 & 0.0718 \\ 0.0002 & 0 & 0 & -0.0028 & 0.0003 & -0.0001 \\ -0.0002 & 0 & 0 & -0.0003 & 0 & 0 \\ 0 & 0 & 0 & -0.0004 & -0.0004 & -0.0002 \end{bmatrix}$$

which were found by running the rendezvous portion of the scenario with a hand-tuned SAC until the control response converges. In this way, artefacts of the adaptation process that might be present in a real mission are also preserved.

Tuning of the SAC parameters is done through metaheuristic search, as described in Sec. 2.6.4. Only a single metaheuristic search was done on this simulation, for simplicity. A DE search was done using the cost function

$$J = \int_0^t \mathbf{e}_y^T(t) \mathcal{Q} \mathbf{e}_y(t) + \mathbf{u}_p^T(t) \mathcal{R} \mathbf{u}_p(t) dt \quad (5.38)$$

with values of $\mathcal{Q} = 1000\mathbf{I}_6$ and $R = \text{diag}([0.1, 0.1, 0.1, 1, 1, 1])$. The metaheuristic search used simulations of the full mission profile with a dummy target mass and moment of inertia of

$$m_{cmb,d} = 6\text{kg}, \quad \mathbf{J}_{cmb,d} = \begin{bmatrix} 0.0588 & -0.0044 & -0.0087 \\ -0.0044 & 0.2414 & 0.0001 \\ -0.0087 & 0.0001 & 0.2462 \end{bmatrix} \text{kg m}^2 \quad (5.39)$$

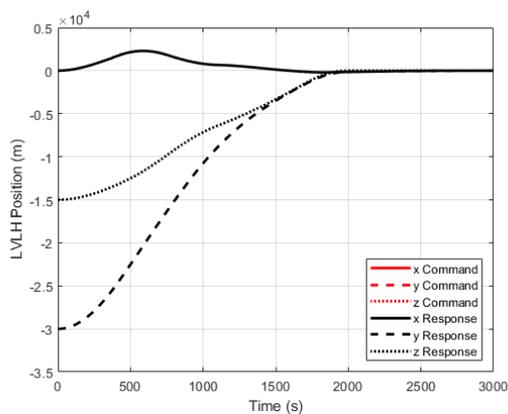
The optimized plant, therefore, had approximately half of the mass of the full plant, and significantly smaller moments of inertia, and therefore, smaller cross-coupled disturbances according to Eq. (2.38). The resulting adaptation rates output by the DE search are found to be

$$\mathbf{\Gamma}_P = \begin{bmatrix} 3.1157\mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & 2.0829\mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 1.19 \times 10^9 \mathbf{I}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & 3.8572 \times 10^9 \mathbf{I}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & 5.739\mathbf{I}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & 8.3457\mathbf{I}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & 0.1541\mathbf{I}_3 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 1.0086\mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & 1.6649\mathbf{I}_3 \end{bmatrix} \quad (5.40)$$

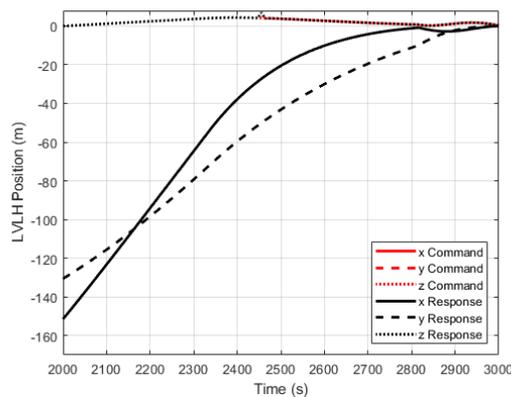
$$\mathbf{\Gamma}_I = \begin{bmatrix} 1.6157 \times 10^9 \mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & 5.6154 \times 10^8 \mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 1.0728 \times 10^3 \mathbf{I}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & 1.9287 \times 10^5 \mathbf{I}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & 580 \mathbf{I}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & 10 \mathbf{I}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & 0.3918 \mathbf{I}_3 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 1.07 \times 10^4 \mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & 2.0 \times 10^4 \mathbf{I}_3 \end{bmatrix} \quad (5.41)$$

$$\boldsymbol{\sigma} = \begin{bmatrix} \mathbf{0}_{21} & \mathbf{0} \\ \mathbf{0} & 121.4834 \mathbf{I}_6 \end{bmatrix} \quad (5.42)$$

These adaptation rates are then used to control the chaser and combined system during the mission. The SAC is run at the same update rate as the simulation, at 400 Hz, or one update every 0.0025 s.

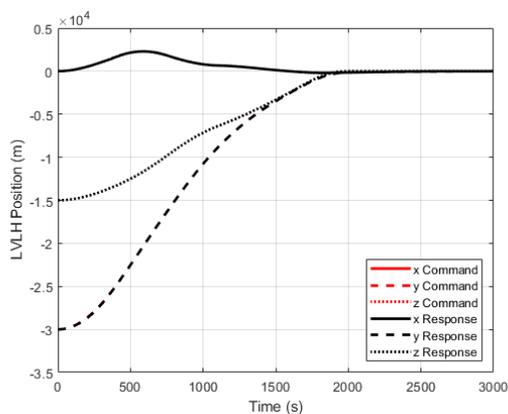


(a) Rendezvous

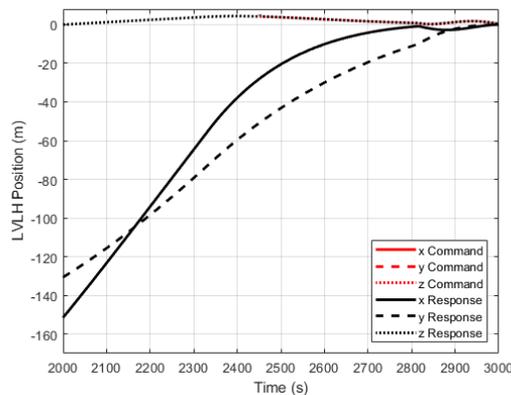


(b) Docking

Figure 5.7: PID Position Response During Rendezvous and Docking



(a) Rendezvous



(b) Docking

Figure 5.8: SAC Position Response During Rendezvous and Docking

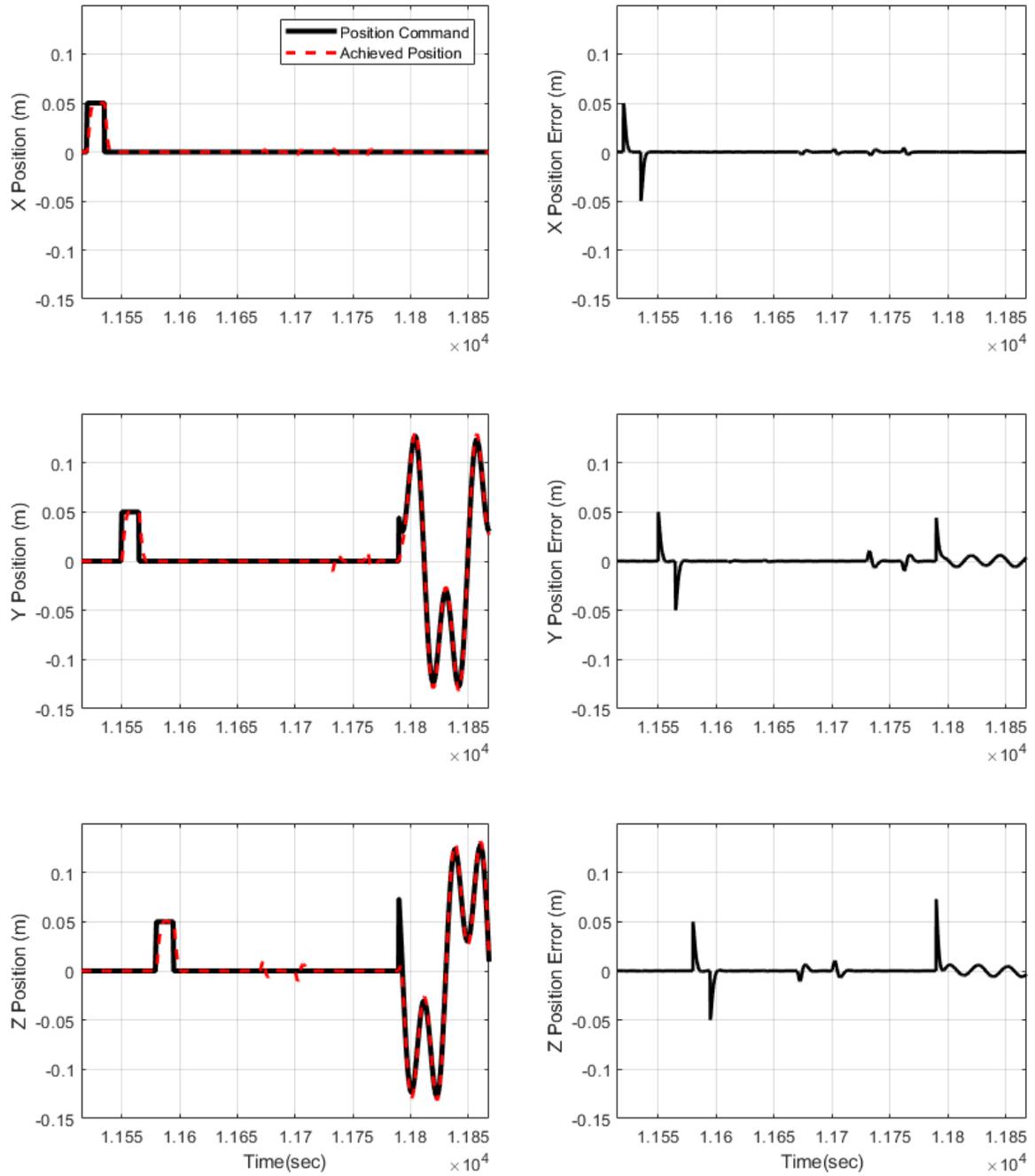


Figure 5.9: PID Position Responses Post-Dock

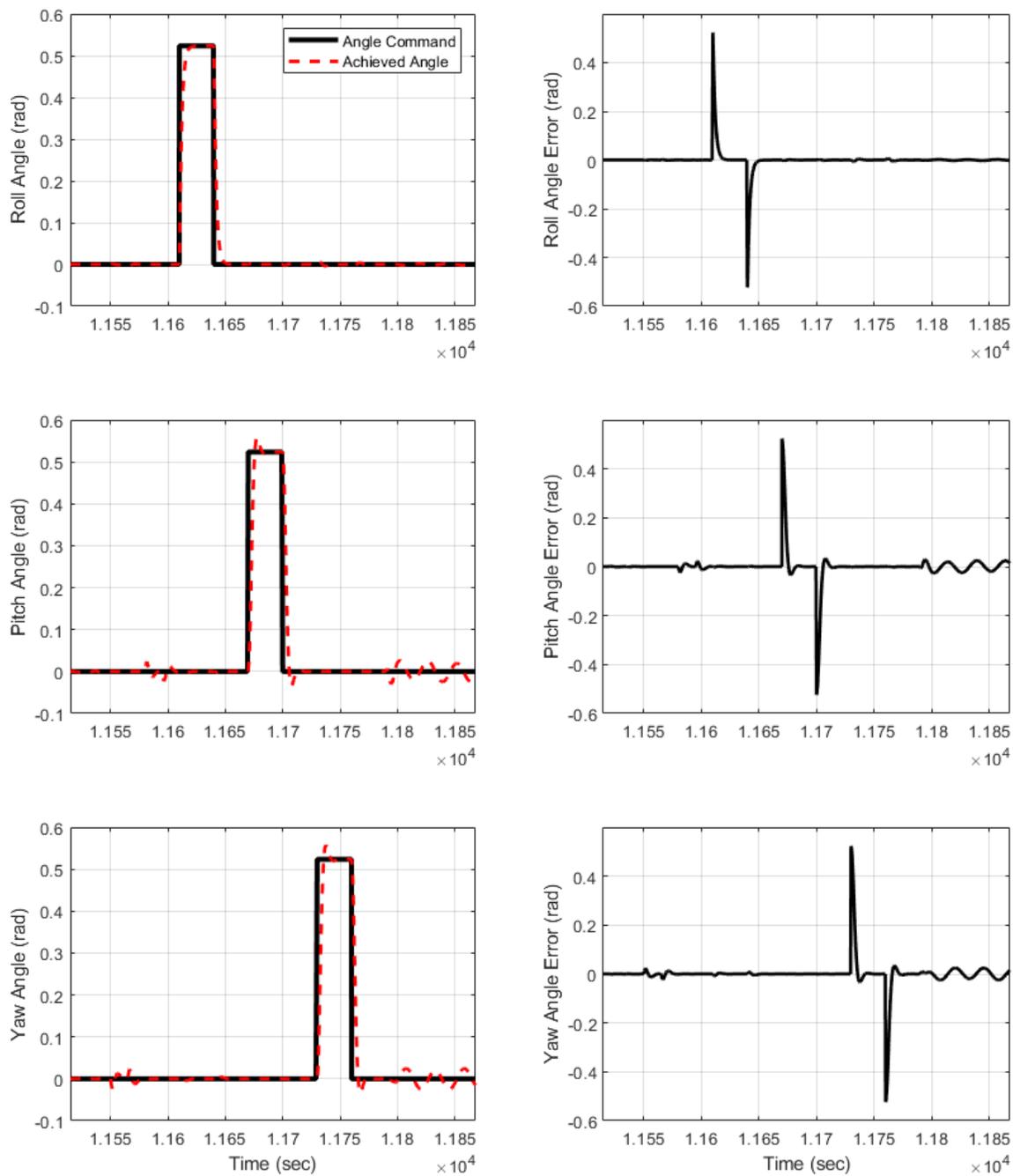


Figure 5.10: PID Orientation Responses Post-Dock

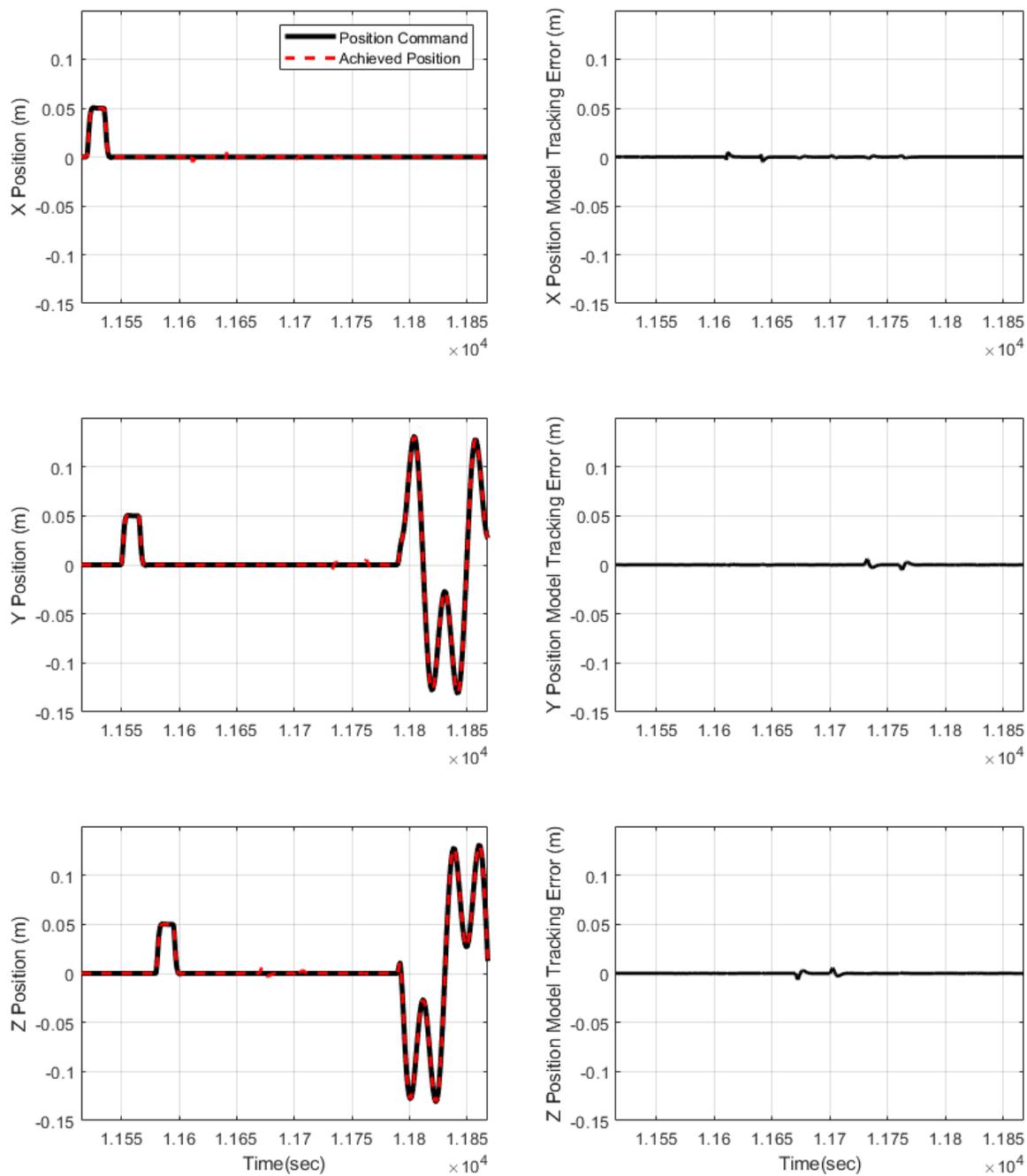


Figure 5.11: SAC Position Responses Post-Dock

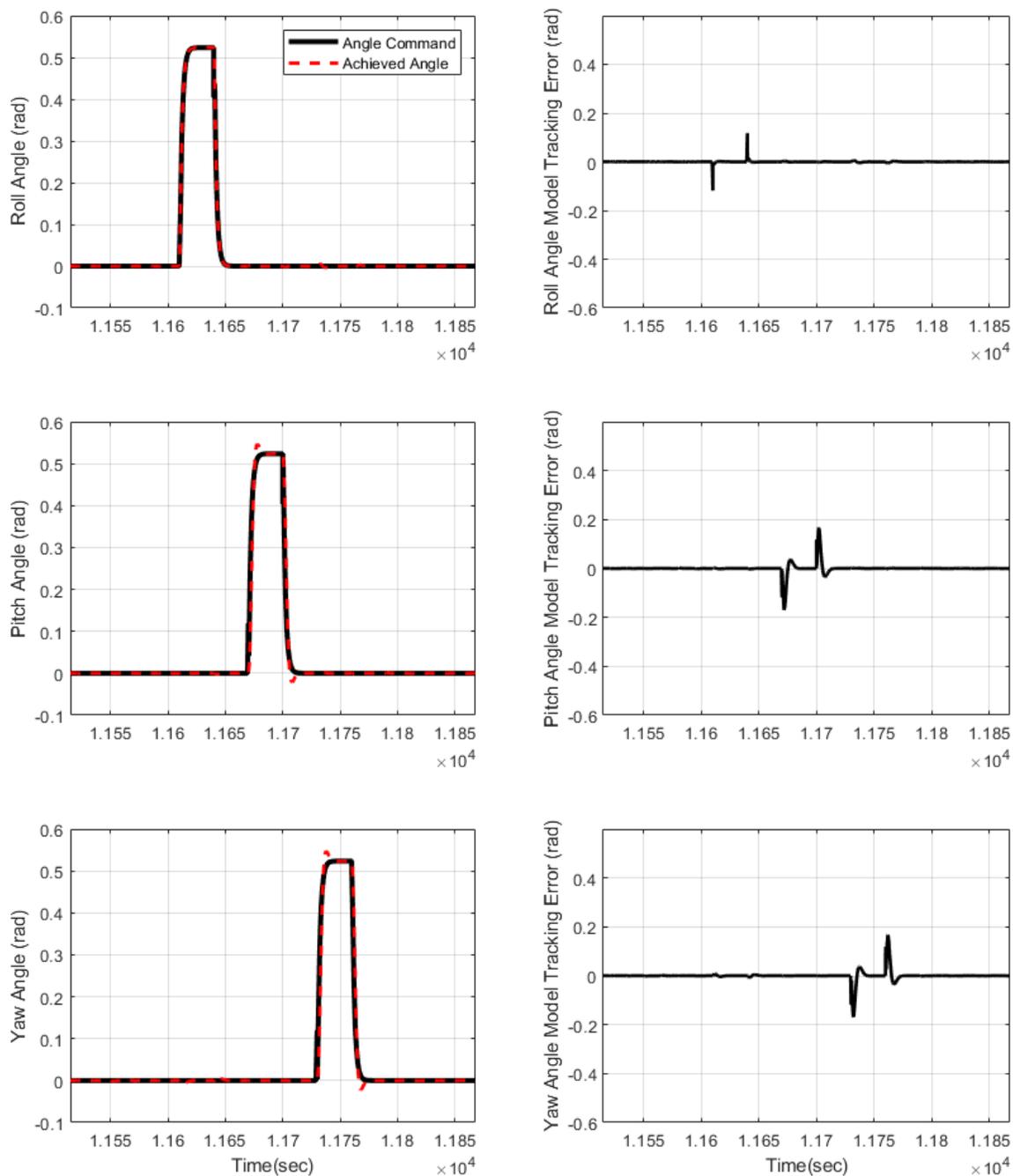


Figure 5.12: SAC Orientation Responses Post-Dock

5.7 Simulation and Results

The performance of the linear and adaptive controllers during rendezvous and docking are demonstrated in Figs. 5.7 and 5.8. It should be noted that for most of each response the command and the response overlap. The largest trajectory errors occur during the transition to the docking phase of flight, which occurs at a separation of 200 m, or at about 2450 seconds into the mission.

The response of the linear feedback controller to position and orientation commands in the combined system is shown in Fig. 5.9 and Fig. 5.10. Since the nature of the control problem requires an uncertain combined system, it is not possible for a linear control design to compensate for the cross-coupling of the position and orientation response. Any attempt to compensate for cross-coupling in the combined system would affect the chaser response during rendezvous and docking, a critical phase of flight. Severe position disturbances are present during orientation commands, and orientation disturbances are present during position commands.

The adaptive controller response for the same commands in the combined system are shown in Fig. 5.11 and 5.12. The adaptive controller is able to identify and separate the cross-coupled position and orientation dynamics, whereas the linear controller cannot. Orientation disturbances due to position commands are negligible during step commands, and invisible during the cycloid command. The combined position-orientation SAC controller is able to identify and eliminate cross coupling between the position and orientation response, while also maintaining the accuracy necessary for rendezvous and docking.

5.8 Discussion of Simulation

Both controllers are able to achieve adequate performance during the rendezvous stage of flight to meet with the target. Similarly, both the linear and adaptive controller are able to align themselves with the rotating target, approach within the rendezvous cone, and meet the successful docking requirement of 1 mm and 5 mm/s relative velocity between the two docking points despite rotation of the target, as well as bringing the combined system to a halt before beginning post-dock manoeuvres.

The linear controller position step responses seen in Fig. 5.9 are slightly underdamped, with a time constant of 2 seconds in all three axes. The y and z step responses are slightly underdamped compared to the x axis. The steady-state position error during the y-z cycloid oscillates between ± 0.005 m. Very notable are the large disturbances to the position during orientation commands. as much as 0.01 m of displacement occurs in the x and y positions during pitch and yaw step commands. The linear controller orientation step response becomes underdamped in the pitch and yaw axes after docking, overshooting to 0.56 rad. Significant orientation errors are present during position responses of up to 0.02 rad (1.15°), whereas the orientation error oscillates between ± 0.02 rad during the entirety of the cycloid position command.

The adaptive control implementation is able to maintain a critically damped position response despite the large increase in mass in the combined system. The SAC step command reaches 2% settling in 8 seconds, while the PD does not meet the 2% settling requirement during the 15s command. Although disturbances are present in the SAC position due to orientation commands, the magnitude of the largest of these disturbances are half the magnitude of the same linear disturbances, reaching ± 0.005 m at most, and are not easily visible in Fig. 5.11. Similarly, orientation disturbances due to position commands are an order of magnitude smaller than the linear case, with disturbances only reaching ± 0.003 rad during position commands. The orientation step response for the SAC is underdamped as well in the y and z axes, however a smaller overshoot of 0.55 rad is reached. Additionally, the steady-state orientation tracking during the y-z cycloid command are an order of magnitude smaller than in the linear case, never reaching more than ± 0.001 rad.

Adaptive control, by merging the previously independent position and orientation control, is able to identify and mitigate the presence of disturbances due to cross-coupling of the position and orientation response. Post-docking performance, despite the presence of large uncertainty in the final system, is similar to the chaser-only performance.

Chapter 6

Conclusions

6.1 Thesis Summary

There is great opportunity in space activities: NASA and ESA are partnering on sample-return missions from Mars to teach us more about the geology of Earth's cousin than could ever be done before, all without risking a single human life¹; spacecraft repair is opening up the possibility of permanent communication, observation, or infrastructure benefits from spacecraft in orbit; and megaconstellations being launched now will be able to provide services to people anywhere on the surface of the Earth. If done right, space activities have the ability to improve the quality of life of everyone on the planet, and they can only do so by improving spacecraft autonomy. The consequences of failing to increase spacecraft autonomy, however, could be catastrophic; tens of pieces of massive debris will need to be removed from the LEO environment every year to maintain access to the scientific knowledge of our solar system, our spacecraft, or even our telecommunications. In order to keep access to space, control systems will need to decrease their dependence on fixed design, and evolve to compensate for unknown or unforeseeable consequences of their mission. Adaptive control is one prominent avenue for increasing spacecraft autonomy, and that is why even older systems like the Curiosity rover have incorporated adaptive controller designs; the simplest way to make a system autonomous without risking human life or wasting valuable mission time is to create a system that can adapt to its mission.

The work presented here addresses the need for improving spacecraft autonomy through research into direct adaptive control technologies and their application in

¹https://www.esa.int/Science_Exploration/Human_and_Robotic_Exploration/Exploration/Mars_sample_return

spacecraft. Surveys of debris deorbiting techniques have determined that active techniques, in which physical contact is made with the target, are the best candidate for safe, fast, and efficient debris removal [1]. Research has been done on the dynamics for both this system, as well as alternative methods of spacecraft capture and control [77]. Previous work had already identified SAC as a good candidate for adaptive control in spacecraft proximity operations [74], further suggesting that proper implementation of SAC in space systems could increase the reliability and autonomy of space systems. However, the high-performance, high-risk environment of space activities does not pair well with the relatively undertested direct adaptive control concept. This work has endeavoured to increase confidence in direct adaptive techniques, while countering elements of the design that lead to uncertainty in operation, since spacecraft require anything but.

Although theoretical results suggested stability in any stabilizable system, experimental and simulation results suggested this was not always the case. In order to learn more about SAC, countless simulations, SAC designs, and days worth of parameter selection were undertaken and could not be included in this text. The result of experimentation is ample knowledge of the effects of the SAC architecture and how the various signals relate to one another even under adaptation and feedback. Most importantly, relationships that underpin the design of a SAC have been determined and generalized.

The performance of the ideal model in a SAC is understood to be a guidance task first and foremost. The new understanding of the ideal model as a guidance task should spur research into the selection of the ideal model, as well as increasing its scope to systems with large nonlinearities.

The issue of SAC stability in implementations is addressed by bounding the maximum system gain. A heuristic is designed that allows the maximum gain reached by the SAC to be bounded by appropriate and algorithmic selection of the adaptation parameters, ensuring that so long as a set of maximum gains exists that a set of acceptable parameters can also be determined.

The proportional adaptation of SAC is similarly reinterpreted as a feedback controller with a gain that grows quadratically with the reference signal, explaining at

once why it improves system convergence, decreases integral adaptation, and maintains stability. The reimagining of the proportional adaptation as a “quadratic controller” allows designers to craft a control boundary for the system within which adaptation can occur, and beyond which stability is guaranteed. The choice of this boundary can be made, and the effects of a set of proportional adaptation parameters can be both qualitatively and quantitatively determined with the technique.

A disturbance accommodating adaptation scheme, first proposed by Prabhakar et al. [29] is extended to include a feedforward parallelized system, and the effects of disturbance on that system are described. It is found that systems that are stabilized through feedforward parallelization, if the disturbance is not also augmented, will increase the real tracking error to minimize the tracking error seen by the SAC. The disturbance accommodating adaptation is experimentally verified for a set of virtual disturbances with known frequency, unknown phase, and unknown amplitude, where it is found that the technique is able to significantly counter the disturbances.

Principles of mathematical optimization, in the form of nonlinear and metaheuristic optimization, are applied to the design of SAC. Nonlinear optimization is unable to determine good controller designs for SAC, likely due to the strong nonlinearity present in the controller response. Metaheuristic searches are able to determine good SAC designs if they are supplied with a simulation that captures those effects. The application of metaheuristic searches to the control of spacecraft position through SAC is verified experimentally, in which metaheuristic searches are able to tightly control the position of a spacecraft.

Finally, the techniques for implementation of SAC in physical systems are applied to a simulation of the rendezvous, docking, and post-docking control of a target in orbit; the full uncontrolled spacecraft problem can be solved by SAC design alone, which is able to not only control the spacecraft during and throughout rendezvous and docking, but is also able to significantly compensate for unknown and unavoidable cross-coupling in the final system. A linear controller designed for the system is unable to compensate for the new system dynamics.

The work presented here has successfully demonstrated a myriad of methods and

techniques to implement SAC into any physical system, addressing several of the major concerns of SAC implementation along the way, and verifying their suitability for use in space activities. Whether it be determining a set of stable adaptation parameters, compensating for disturbances, or optimizing the adaptive control response, the presented work ensures that there remains few reasons not to include adaptation in systems with uncertainty.

6.2 Significance of Work

The work presented here is the most comprehensive review available for SAC design techniques, but also represents the most complete collection of considerations for SAC implementation in physical systems. Any control engineer who hopes to include adaptation in their system can skim the contents of this reference, whether in the design summary of Chapter 3 to see a broad overview of the process of SAC implementation, or any element of SAC theory in Chapter 2, to get a quick and useful grasp of direct adaptive control with links for further reading. The work has benefitted the author's grasp of multivariable systems immensely, and the results of the research have been shared with the wider control systems community as both conference proceedings and a journal publication. Below is the list of published and submitted papers.

6.2.1 Conference Proceedings

Predmyrskyy, A. and Ulrich S., "Swarm Optimized Simple Adaptive Controller for Spacecraft Proximity Operations Trajectory Tracking," *21st IFAC World Congress*, Berlin, Germany, 12-17 July 2020, pp. 3851-3856

Predmyrskyy, A. and Ulrich S., "Spacecraft Rendezvous, Docking, and Post-Docking Maneuvers Under Large Uncertainties Via Swarm Optimized Simple Adaptive Control," *AAS/AIAA Astrodynamics Specialist Conference*, Virtual, 9-12 August 2021, Accepted for Publication

6.2.2 Journal Articles

Predmyrskyy, A. and Ulrich S., “Design Methodologies for Simple Adaptive Controllers with Applications to Spacecraft Proximity Operations,” *IEEE transactions on control systems technologies*, under review

6.3 Recommendations for Future Work

Despite what has already been written on the topic in this work alone, the most exciting work on adaptive control and spacecraft autonomy has yet to come. During the course of conducting this work a tidal wave of research avenues became apparent. At its most fundamental, tools for analysing SAC design, and nonlinear controller design more generally, are desperately needed. The SAC response changes nonlinearly with design parameters, and the response itself can vary wildly with parameter selection, signal content, and samplerate. The most immediate benefit to adaptive controller designs is determining methods to increase the samplerate of these systems, minimize noise, and improve stability; although this is an objective of GNC research more generally. There are, however, a few more specific and possibly impactful areas of research that deserve their own description in the sections that follow.

6.3.1 Spacecraft Implementation

The currently developed docking simulation does not incorporate all aspects of the docking phase of target control. Having understood the role of adaptive control in post-dock performance, the accuracy of the docking portion of simulation should be expanded upon. In general confidence of SAC in spacecraft applications should be increased by higher fidelity modelling and implementation.

6.3.2 Bursting Phenomena

The bursting phenomena is a poorly characterized and highly-impactful part of adaptive controller design. Some work has already been taken to understand the bursting phenomena, although it is typically in relation to neural networks. Masaud and Macnab [78] consider some of the effects of bursting in direct adaptive control systems and

suggests counteracting them with an introspective neural network. Masaud and Macnab mention the common methods of combatting bursting, namely techniques called *leakage*, *e-modification*, *dead-zones*, and *weight projection*, and notices that they all reduce performance in order to minimize bursting. More recently Macnab [79] has recommended a technique to minimize bursting using weight smoothing. Some descriptions of how bursting might occur have been posited by Inoue and Kaneko [80], although their description is for multicellular and networked systems, noting that bursting appears as some sort of oscillation between system states, adaptation, and tracking.

Some of what the author has seen to cause bursting are encompassed in part or in whole by the descriptions above. There is a sense, in the discrete SAC update equations, in which bursting manifests as a literal tradeoff between ideal gain error and tracking error.

Research into how bursting behaviour occurs to begin with, and how it generalizes to other systems will be invaluable for any and all systems that make use of adaptation.

6.3.3 Comparison or Equivalence of Augmentation Techniques

In the original development of SAC theory, augmentation of SAC for non-ASPR systems was completed with the description of feedforward parallelization, sometimes called a *shunt*. Since then, sensor blending has also been developed as a method to augment SAC and other direct adaptive control methodologies for use in non-ASPR systems.

Research into the effects of different augmentation techniques should be further explored. It may also be that unexpected similarities or differences exist between several classes of ASPR augmentation techniques. Regardless of the outcome, a survey of augmentation techniques should be undertaken to clarify which methods are suitable for real-time applications.

6.3.4 Guidance for Linear and Nonlinear Ideal Models

Section 3.3.1 covered the paradigm shift of considering the ideal model in SAC less as a control decision, and more as a guidance decision. Numerous guidance techniques and methodologies already exist in the world of linear system guidance, before taking into consideration nonlinear guidance, and adaptive guidance. The applicability of currently existing guidance techniques in improving SAC should be explored.

6.3.5 Extended Disturbance Accommodation

The linear disturbance accommodation presented by Prabhakar [29] that was experimentally verified in Chapter 4 suggests incredible applications in the world of disturbance accommodation, and perhaps even control.

The first necessity is a proof of the stability of the disturbance accommodation control. Prabhakar and the author have not provided a proof of stability for the given controller in an ASPR system. Time should be taken to derive this proof, and supplement the experimental verification with theoretical understanding.

The ability for adaptive disturbance accommodation to determine the magnitude and phase of a signal makes it resemble a clumsy real-time Fourier transform. The relationship between frequency identification techniques and SAC should be explored.

6.3.6 Kalman Filters: Incorporating Navigation Techniques

Following from the discussion of ideal models as guidance and the understanding of SAC adaptation as covariance identifications, elements of navigation theory should be implemented into the SAC architecture. Kalman filters, for example, already deduce statistical information about incoming datastreams and synthesize ideal output signals. It may be possible to link the statistical elements of the SAC design with other techniques, improving both.

6.3.7 Real-Time Correlation Engines

Direct adaptive controllers in general aim to identify system responses and determine what actions need to be taken to improve that response. What is clear from this work

is that SAC determines the linear correlation between a gain and the tracking error. It may be that other direct adaptive control techniques attempt to create a “real-time correlation engine” and that some aspects are shared between them.

If the SAC represents a first order “correlation engine”, research should be made to determine what higher orders of such an engine might resemble. It may further be possible to borrow techniques from system identification and modelling to improve adaptive controllers. The effects of including too many signals, not enough signals, or uncorrelated signals to this system should be observed and characterized. It may be possible to leverage elements of all GNC techniques in bulk to develop what is essentially a machine for relating signals to one another.

Bibliography

- [1] Liou, J. C. and Johnson, N. L., “A sensitivity study of the effectiveness of active debris removal in LEO,” *Acta Astronautica*, Vol. 64, No. 2-3, jan 2009, pp. 236–243.
- [2] Montenbruck, O., Gill, E., and Lutze, F., “Satellite Orbits: Models, Methods, and Applications,” *Applied Mechanics Reviews*, Vol. 55, No. 2, mar 2002, pp. B27.
- [3] Welch, R., Limonadi, D., and Manning, R., “Systems engineering the Curiosity rover: A retrospective,” *Proceedings of 2013 8th International Conference on System of Systems Engineering: SoSE in Cloud Computing and Emerging Information Technology Applications, SoSE 2013*, 2013, pp. 70–75.
- [4] Bahr, N. J. and DePalo, S. V., “Making the Hubble Space Telescope servicing mission safe,” *Acta Astronautica*, Vol. 29, No. 10-11, oct 1993, pp. 757–763.
- [5] Juillard, M., Richard-Noca, M., and Kneib, J.-P., “Dedicated On-Board Computer for Active Debris Removal Mission,” *Small Satellite Conference*, aug 2020.
- [6] Hakima, H. and Emami, M. R., “Assessment of active methods for removal of LEO debris,” *Acta Astronautica*, Vol. 144, mar 2018, pp. 225–243.
- [7] Pelton, J. N., *New Solutions for the Space Debris Problem*, SpringerBriefs in Space Development, Springer International Publishing, Cham, 2015.
- [8] Johnson, N. L., Stansbery, E., Liou, J. C., Horstman, M., Stokely, C., and Whitlock, D., “The characteristics and consequences of the break-up of the Fengyun-1C spacecraft,” *Acta Astronautica*, Vol. 63, No. 1-4, jul 2008, pp. 128–135.
- [9] Petro, A. J., “Techniques for orbital debris control,” *Journal of Spacecraft and Rockets*, Vol. 29, No. 2, may 1992, pp. 260–263.
- [10] Toupet, O., Biesiadecki, J., Rankin, A., Steffy, A., Meirion-Griffith, G., Levine, D., Schadeegg, M., and Maimone, M., “Terrain-adaptive wheel speed control on the Curiosity Mars rover: Algorithm and flight results,” *Journal of Field Robotics*, Vol. 37, No. 5, aug 2020, pp. 699–728.
- [11] Samad, T., “A survey on industry impact and challenges thereof,” *IEEE Control Systems*, Vol. 37, No. 1, feb 2017, pp. 17–18.
- [12] Maxwell, J. C., “On Governors,” Tech. rep., Proceedings of the Royal Society of London, 1867.

- [13] Routh, E. J., *A treatise on the stability of a given state of motion, particularly steady motion*, Macmillan and co., 1877.
- [14] Porter, A., *Theory of Servomechanisms*, Vol. 161, McGraw-Hill, 1948.
- [15] Jury, E. I., *Sampled-Data Control System*, Wiley, 1958.
- [16] Truxal, J. G., *Automatic Feedback Control System Synthesis*, McGraw-Hill, 1955.
- [17] Chipman, J. C., Houtz, W., and Shillor, M., “Simulations of a thermostat model I: Approach to steady states,” *Mathematical and Computer Modelling*, Vol. 32, feb 2000, pp. 765 – 790.
- [18] Kiencke, U. and Nielsen, L., *Automotive control systems: For engine, driveline, and vehicle: Second edition*, Springer Berlin Heidelberg, 2005.
- [19] Carvalho, A., Lefèvre, S., Schildbach, G., Kong, J., and Borrelli, F., “Automated driving: The role of forecasts and uncertainty - A control perspective,” *European Journal of Control*, Vol. 24, European Control Association, jul 2015, pp. 14–32.
- [20] P. J. Antsaklis, A. N. M., *A Linear Systems Primer*, Birkhauser Boston 2007, 2007.
- [21] Stengel, R. F., *Flight Dynamics*, No. 17, Princeton University Press, oct 2004.
- [22] Leith, D. J. and Leithead, W. E., “Survey of gain-scheduling analysis and design,” *International Journal of Control*, Vol. 73, No. 11, jul 2000, pp. 1001–1025.
- [23] Langton, R., *Stability and control of aircraft systems : introduction to classical feedback control*, Wiley, 2006.
- [24] Lyapunov, A. M., “The general problem of the stability of motion,” *International Journal of Control*, Vol. 55, No. 3, 1992, pp. 531–534.
- [25] Krafes, S., Chalh, Z., and Saka, A., “Review: Linear, nonlinear and intelligent controllers for the inverted pendulum problem,” *Proceedings of 2016 International Conference on Electrical and Information Technologies, ICEIT 2016*, Institute of Electrical and Electronics Engineers Inc., jul 2016, pp. 136–141.
- [26] Barkana, I., *Simple adaptive control: The optimal model reference-short tutorial*, Vol. 11, IFAC, 2013.
- [27] Bar-Kana, I., “On Parallel Feedforward and Simplified Adaptive Control,” *IFAC Proceedings Volumes*, Vol. 20, No. 2, jul 1987, pp. 99–104.
- [28] Ulrich, S. and De Lafontaine, J., “Autonomous atmospheric entry on mars: Performance improvement using a novel adaptive control algorithm,” *Journal of the Astronautical Sciences*, Vol. 55, No. 4, aug 2007, pp. 431–449.

- [29] Prabhakar, N., Painter, A., Prazenica, R., and Balas, M., “Trajectory-driven adaptive control of autonomous unmanned aerial vehicles with disturbance accommodation,” *Journal of Guidance, Control, and Dynamics*, Vol. 41, No. 9, jul 2018, pp. 1976–1989.
- [30] Rusnak, I., Weiss, H., and Barkana, I., “Improving the performance of existing missile autopilot using simple adaptive control,” *International Journal of Adaptive Control and Signal Processing*, Vol. 28, No. 7-8, 2014, pp. 732–749.
- [31] Boyd, S. P., *Convex optimization*, Cambridge, 2004.
- [32] Berkovitz, L. D., *Nonlinear optimal control theory*, CRC Press, 2012.
- [33] Siarry, P., editor, *Metaheuristics*, Springer International Publishing, 2016.
- [34] Ab Wahab, M. N., Nefti-Meziani, S., and Atyabi, A., “A Comprehensive Review of Swarm Optimization Algorithms,” *PLOS ONE*, Vol. 10, No. 5, may 2015, pp. e0122827.
- [35] Ouyang, P. and Pano, V., “Comparative Study of DE, PSO, and GA for Position Domain PID Controller Tuning,” *Algorithms*, Vol. 8, No. 3, 2015, pp. 697–711.
- [36] Ito, T. T. M. and Mizumoto, I., “Parameter optimization of simple adaptive control via differential evolution,” *2017 6th International Symposium on Advanced Control of Industrial Processes (AdCONIP)*, Taipei, Taiwan, 2017.
- [37] Qin, A., “Differential Evolution Algorithms With Strategy Adaptation for Global Numerical Optimization,” *IEEE Transactions on Evolutionary Computation*, Vol. 13, 2009, pp. 398 – 417.
- [38] Olsson, A. E., editor, *Particle swarm optimization theory, techniques and applications*, Nova Science Publishers, 2011.
- [39] Qing, A., *Differential Evolution: Fundamentals and Applications in Electrical Engineering*, John Wiley and Sons, sep 2009.
- [40] Yousefi, H., Handroos, H., and Soleymani, A., “Application of Differential Evolution in system identification of a servo-hydraulic system with a flexible load,” *Mechatronics*, Vol. 18, No. 9, nov 2008, pp. 513–528.
- [41] Hull, D. G., *Optimal Control Theory for Applications*, Mechanical Engineering Series, Springer New York, New York, NY, 2003.
- [42] Hu, W., *Fundamental spacecraft dynamics and control*, Wiley, 2015.
- [43] Park, R. S., Folkner, W. M., Williams, J. G., and Boggs, D. H., “The JPL Planetary and Lunar Ephemerides DE440 and DE441,” *The Astronomical Journal*, Vol. 161, No. 3, feb 2021, pp. 105.

- [44] Alfriend, K. T., *Spacecraft formation flying dynamics, control and navigation*, Butterworth-Heinemann, 2010.
- [45] Efroimsky, M., “Relaxation of Wobbling Asteroids and Comets. Theoretical Problems. Perspectives of Experimental Observation,” Tech. rep., Planetary and Space Science, 2001.
- [46] Chow, T. L., “Classical mechanics,” 2013.
- [47] Barkana, I., “Gain conditions and convergence of simple adaptive control,” *International Journal of Adaptive Control and Signal Processing*, Vol. 19, No. 1, 2005, pp. 13–40.
- [48] Hakimi-Moghaddam, M., “Positive real and strictly positive real MIMO systems: theory and application,” *International Journal of Dynamics and Control*, Vol. 8, No. 2, jun 2020, pp. 448–458.
- [49] Hoagg, J. B., Lacyt, S. L., Erwint, R. S., and Bemstein, D. S., “First-Order-Hold Sampling of Positive Real Systems And Subspace Identification of Positive Real Models,” Tech. rep.
- [50] Mehrmann, V., editor, *The Autonomous linear quadratic control problem : theory and numerical solution*, Springer-Verlag, 1991.
- [51] Lancaster, P., *Algebraic Riccati equations*, Clarendon Press, 1995.
- [52] Willems, J. C., “Least Squares Stationary Optimal Control and the Algebraic Riccati Equation,” *IEEE Transactions on Automatic Control*, Vol. 16, No. 6, 1971, pp. 621–634.
- [53] Laub, A. J., “A Schur Method for Solving Algebraic Riccati Equations,” *IEEE Transactions on Automatic Control*, Vol. 24, No. 6, 1979, pp. 913–921.
- [54] Grad, J. R. and Morris, K. A., “Solving the linear quadratic optimal control problem for infinite-dimensional systems,” *Computers and Mathematics with Applications*, Vol. 32, No. 9, nov 1996, pp. 99–119.
- [55] Benner, P., Li, J. R., and Penzl, T., “Numerical solution of large-scale Lyapunov equations, Riccati equations, and linear-quadratic optimal control problems,” *Numerical Linear Algebra with Applications*, Vol. 15, No. 9, nov 2008, pp. 755–777.
- [56] Khalil, H. K., *Nonlinear systems*, Macmillan Pub. Co., 1992.
- [57] Barkana, I., “Adaptive Control? But is so Simple!: A Tribute to the Efficiency, Simplicity and Beauty of Adaptive Control,” *Journal of Intelligent and Robotic Systems: Theory and Applications*, Vol. 83, No. 1, 2016, pp. 3–34.

- [58] Bar-Kana, I. and Kaufman, H., “Global stability and performance of a simplified adaptive algorithm,” *International Journal of Control*, Vol. 42, No. 6, 1985, pp. 1491–1505.
- [59] Bar-Kana, I., “POSITIVE REALNESS IN DISCRETE-TIME ADAPTIVE CONTROL SYSTEMS.” *Proceedings of the American Control Conference*, IEEE, 1986, pp. 1440–1443.
- [60] H. Kaufman, I. Barkana, K. S., *Direct Adaptive Control Algorithms: Theory and Applications*, Springer-Verlag New York, 1998.
- [61] Fradkov, A. L., “Quadratic Lyapunov functions in the adaptive stability problem of a linear dynamic target,” *Siberian Mathematical Journal*, Vol. 17, No. 2, mar 1976, pp. 341–348.
- [62] Golub, G. H., Nash, S., and van Loan, C., “A Hessenberg-Schur Method for the Problem $AX+XB=C$,” *IEEE Transactions on Automatic Control*, Vol. 24, No. 6, 1979, pp. 909–913.
- [63] Barkana, I., Rusnak, I., and Weiss, H., “Almost passivity and simple adaptive control in discrete-time systems,” *Asian Journal of Control*, Vol. 16, No. 4, 2014, pp. 947–958.
- [64] Balas, M. J. and Frost, S. A., “Sensor blending for direct adaptive control of non-minimum phase linear infinite-dimensional systems in Hilbert space,” *Proceedings of the American Control Conference*, Institute of Electrical and Electronics Engineers Inc., jun 2017, pp. 474–480.
- [65] Shibata, H., Fujinaka, T., and Sun, Y., “A Discrete-Time Algorithm for Simple Adaptive Control,” *IFAC Proceedings Volumes*, Vol. 30, No. 3, apr 1997, pp. 349–354.
- [66] Takagi, T., Ito, M., and Mizumoto, I., “Parameter optimization of simple adaptive control via differential evolution,” *2017 6th International Symposium on Advanced Control of Industrial Processes, AdCONIP 2017*, 2017, pp. 318–323.
- [67] Andersen, E. D. and Roos, . C., “On implementing a primal-dual interior-point method for conic quadratic optimization,” *Math. Program., Ser. B*, Vol. 95, 2003, pp. 249–277.
- [68] Lu, P. and Liu, X., “Autonomous trajectory planning for rendezvous and proximity operations by conic optimization,” *Journal of Guidance, Control, and Dynamics*, Vol. 36, No. 2, 2013, pp. 375–389.
- [69] Schittkowski, K., “NLPQL: A FORTRAN SUBROUTINE SOLVING CONSTRAINED NONLINEAR PROGRAMMING PROBLEMS,” Tech. Rep. 6, *Annals of Operations Research*, 1985.

- [70] *Numerical Optimization*, Springer Series in Operations Research and Financial Engineering, Springer New York, 2006.
- [71] Angeline, P., “Using Selection to Improve Particle Swarm Optimization,” *IEEE World Conference on Computational Intelligence*, Anchorage, USA, 1998.
- [72] Predmyrskyy, A. and Ulrich, S., “Swarm Optimized Simple Adaptive Controller for Spacecraft Proximity Operations Trajectory Tracking,” *IFAC-PapersOnLine*, Vol. 53, No. 2, jan 2020, pp. 3785–3790.
- [73] Barkana, I., “Robustness and perfect tracking in simple adaptive control,” *International Journal of Adaptive Control and Signal Processing*, Vol. 30, No. 8-10, aug 2016, pp. 1118–1151.
- [74] Ulrich, S., Hayhurst, D. L., Saenz-Otero, A., Miller, D. W., and Barkana, I., “Simple adaptive control for spacecraft proximity operations,” *AIAA Guidance, Navigation, and Control Conference*, 2014.
- [75] James, J., “Adaptive control for post-dock maneuvers with an unknown semi-cooperative object,” *IEEE Aerospace Conference Proceedings*, Vol. 2016-June, IEEE Computer Society, jun 2016.
- [76] Seweryn, K. and Sasiadek, J. Z., “Satellite angular motion classification for active on-orbit debris removal using robots,” *Aircraft Engineering and Aerospace Technology*, Vol. 91, No. 2, mar 2019, pp. 317–332.
- [77] Hovell, K. and Ulrich, S., “Postcapture Dynamics and Experimental Validation of Subtethered Space Debris,” *Journal of Guidance, Control, and Dynamics*, Vol. 41, No. 2, feb 2018, pp. 519–525.
- [78] Masaud, K. and Macnab, C. J. B., “Preventing bursting in adaptive control using an introspective neural network algorithm,” *Neurocomputing*, 2014.
- [79] Macnab, C. J. B., “Modifying CMAC adaptive control with weight smoothing in order to avoid overlearning and bursting,” *Neural Computing and Applications*.
- [80] Inoue, M. and Kaneko, K., “Dynamics of Coupled Adaptive Elements : Bursting and Intermittent Oscillations Generated by Frustration in Networks,” Tech. rep., 2009.
- [81] A. H. deRuiter, C. Damaren, J. R. F., *Spacecraft Dynamics and Control: An Introduction*, Wiley, jan 2013.

Appendix A

Essential Operations and Notation

Several dependencies are assumed of the reader in the area of matrix notation, operations, and reference frame transformations. The relevant background to understand mathematical concepts relevant to the thesis material and astrodynamics are presented here for the reader's convenience.

A.1 Matrix Operations

Several common matrix operations are used that should be understood by the reader.

For the column matrix

$$\mathbf{x} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad (\text{A.1})$$

the skew symmetric form of that matrix is given by

$$\mathbf{x}^\times \triangleq \begin{bmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{bmatrix} \quad (\text{A.2})$$

The skew symmetric operation here produces a skew-symmetrix matrix from the input. Notice that

$$(\mathbf{x}^\times)^T = -\mathbf{x}^\times \quad (\text{A.3})$$

and furthermore the product of the skew symmetric matrix with another column matrix $\mathbf{y} \in \mathbb{R}^{3 \times 1}$ gives

$$\mathbf{x}^\times \mathbf{y} = (\mathbf{y}^\times)^T \mathbf{x} = -\mathbf{y}^\times \mathbf{x} \quad (\text{A.4})$$

which is similar to identities of the cross product. In this way the skew-symmetric operation resembles a cross product of column matrices.

The two norm of a column matrix is given by

$$\|\mathbf{x}\| \triangleq \sqrt{\mathbf{x}^T \mathbf{x}} = \sqrt{a^2 + b^2 + c^2} \quad (\text{A.5})$$

And finally, the trace is used in Lyapunov stability equations. For a square matrix of size n called $\mathbf{A} \in \mathbb{R}^{n \times n}$ with i^{th} row element and j^{th} column element called $a_{i,j}$

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix} \quad (\text{A.6})$$

then the trace of that matrix is defined as

$$\text{Tr}(\mathbf{A}) \triangleq \sum_{k=1}^n a_{k,k} \quad (\text{A.7})$$

The trace has several properties that are useful for Lyapunov stability analysis. Firstly, it is a linear mapping, namely that for a second similar matrix \mathbf{B} , and multiplier b

$$\text{Tr}(\mathbf{A} + \mathbf{B}) = \text{Tr}(\mathbf{A}) + \text{Tr}(\mathbf{B}) \quad (\text{A.8})$$

$$\text{Tr}(b\mathbf{A}) = b\text{Tr}(\mathbf{A}) \quad (\text{A.9})$$

Second, the contents of a trace can be transposed or cycled without affecting the result. With a third matrix \mathbf{C} now, it is also true that

$$\text{Tr}(\mathbf{A}) = \text{Tr}(\mathbf{A}^T) \quad (\text{A.10})$$

$$\text{Tr}(\mathbf{ABC}) = \text{Tr}(\mathbf{CAB}) \quad (\text{A.11})$$

The requirement for the trace to be taken for square matrices ensures that cycling the elements of a trace keeps the argument square.

A.2 Reference Frame Construction and Conversion

Matrices necessarily represent physical quantities. It will be important to identify how we relate the matrices used in mathematical representations to the physical quantities they represent. Consider the three-dimensional Cartesian space \mathbb{R}^3 . In order to quantify an element of this space, it must have a representation denoting one of each of the individual dimensions of that space. Each dimension is independent of the other dimensions, and as such any element of that set can be described as an independent combination of the unit lengths of each dimension.

Call the unit vectors in \mathbb{R}^3 : \hat{x} , \hat{y} , and \hat{z} . Then the coefficients $a, b, c \in \mathbb{R}$ can be used alongside the unit vectors to construct a vector in that space.

$$\vec{x} = a\hat{x} + b\hat{y} + c\hat{z} = \begin{bmatrix} \hat{x} & \hat{y} & \hat{z} \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad (\text{A.12})$$

It is now plain to see that the vector \vec{x} is composed of the unit vectors of the space, alongside the magnitude of each of those components. We can formalize this relationship through the definitions

$$\vec{\mathcal{F}}_{\mathbb{R}} \triangleq \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{bmatrix} \quad (\text{A.13})$$

$$\mathbf{x} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad (\text{A.14})$$

where now the unit vectors of the reference frame are denoted by $\vec{\mathcal{F}}_{\mathbb{R}}$, called the *vectorix* and the components of the vector in the reference frame, called \mathbf{x} . Together

the relationship between the vector \vec{x} and the column matrix of components of \vec{x} is succinctly described through

$$\vec{x} = \vec{\mathcal{F}}_{\mathbb{R}}^T \mathbf{x} \quad (\text{A.15})$$

A more thorough overview of vectrix notation is present in de Ruiter's *Spacecraft Dynamics and Control* [81]. Notice that the vector representing a quantity can now be described independent of its reference frame. The vector \vec{r} can describe a length in ECI, LVLH, or any other reference frame, and the vector still exists despite those transformations. If operations are being done within a reference frame, the vectrix notation can be dropped and only the components need to be used.

Other useful vector operations can be reimaged in vectrix notation. A dot product between vectors \vec{x} and \vec{y} is given by

$$\vec{x} \cdot \vec{y} = \vec{\mathcal{F}}_{\mathbb{R}}^T \mathbf{x}^T \mathbf{y} \quad (\text{A.16})$$

Cross products are given by

$$\vec{x} \times \vec{y} = \vec{\mathcal{F}}_{\mathbb{R}}^T \mathbf{x} \times \mathbf{y} \quad (\text{A.17})$$

Consider now that vectors \vec{x} and \vec{y} are in two different reference frames $\vec{\mathcal{F}}_x^T$ and $\vec{\mathcal{F}}_y^T$, respectively. The rotation matrix from reference frame $\vec{\mathcal{F}}_y^T$ to reference frame $\vec{\mathcal{F}}_x^T$ is now succinctly written as

$$\mathbf{R}_{xy} = \vec{\mathcal{F}}_x \vec{\mathcal{F}}_y^T \quad (\text{A.18})$$

and is called the *direction cosine matrix*, or sometimes just a rotation matrix. The properties of the rotation come about due to properties of the unit vectors in each frame. One useful side effect is that the inverse of a direction cosine matrix is its transpose, and the inverse of a rotation with itself is the identity matrix, which is expected. By notation we also define the subscripts of a rotation matrix to correspond to the reference frames it rotates from and to. Together these make the description

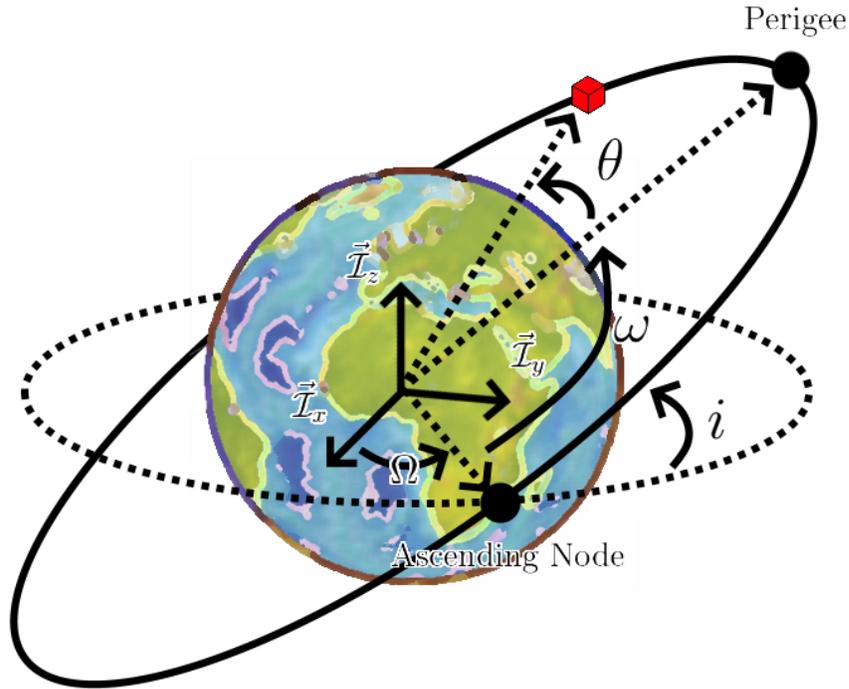


Figure A.1: Orbital Elements $\{a, e, i, \Omega, \omega, \theta\}$ Describe an Individual Orbit

$$\mathbf{R}_{xy}^{-1} = \mathbf{R}_{xy}^T \triangleq \mathbf{R}_{yx} \quad (\text{A.19})$$

$$\mathbf{R}_{xy}\mathbf{R}_{yx} = \mathbf{I}_3 = \mathbf{R}_{xy}\mathbf{R}_{xy}^T \quad (\text{A.20})$$

A.3 Classical Orbital Elements

The development and subsequent verification of Newton's law of gravitation showed that all orbits could be concisely described by a single simple equation of motion. Further theoretical application of Newton's law of gravitation determined the solutions for the motion of two massive bodies in orbit around one another. It is not always the case that both objects are large enough to influence each others' motion, however. Typically, any object in orbit has its motion governed by one massive body (such as a satellite's orbit around the Earth) while accelerations due to gravitational effects from other bodies are negligible. The gravity of the Sun does not dramatically

affect the orbit of a spacecraft in LEO, for example, due to the difference in scale between the gravitational pull of the Earth and the Sun on that spacecraft. Similarly, the Earth is not affected by the mass of spacecraft in orbit around it due to their small mass. The problem that corresponds to the motion of the spacecraft is called the *restricted two-body problem*, which has been solved.

The motion of the satellite in the restricted two-body problem, and by consequence the motion of any spacecraft around the Earth, can be determined from the initial conditions of the spacecraft or through observations of its motion. Without external perturbations acting on the spacecraft, the orbit is fixed relative to the inertial reference frame and can be described through the *classical orbital elements*, which are described here.

All orbits about a parent body are elliptical. The eccentricity e , borrowed from mathematical definitions of the ellipse, describes the shape of the ellipse. An eccentricity of 0 denotes a perfectly circular orbit, while increasing values up to 1 indicate increasingly elliptical orbits. Orbits with eccentricities near 1 are highly elliptical, and become parabolic when $e = 1$. Any value of e above 1 denotes a hyperbolic orbit. Unbounded parabolic and hyperbolic orbits denote orbits that leave their parent body, which are useful when planning interplanetary journeys that leave the influence of their parent body.

With the eccentricity known, the size of the orbit must also be described. The semi-major axis of an ellipse, denoted a following the mathematical convention, describes the distance between the center of the ellipse (the halfway point between each foci) and the edge of the long axis of the ellipse. For circular orbits, the semi-major axis is simply the radius of the orbit, and can also be understood as the average between the perigee and apogee radii. The semi-major axis is typically on the order of kilometers, and fixes the size of the orbit. Furthermore, since the spacecraft is assumed to have negligible mass, the center of mass of the parent body must be located at one of the foci of the orbital ellipse.

With the eccentricity and semi-major axis constrained, the orbit is fully described in the orbital plane. That is to say that without external perturbations, the orbit must occur within a single plane, and the size and shape of the orbit within that

plane has been fully described. What remains is to constrain the orientation of the orbit with respect to the parent body, as well as the position of the spacecraft along the orbit; this will be done using the inclination of the orbit i , the right ascension of the ascending node Ω , the argument of perigee ω , and the true anomaly θ .

The inclination i of the orbit describes the angle between the orbital plane and the equatorial plane of the parent body. An inclination of 0 means that the orbital plane is aligned with the equatorial plane, an inclination of 90° means that the orbital plane is perpendicular to the equatorial plane, while values above 90° denote a “retrograde” orbit, that is to say the orbit moves in the opposite direction of the parent body’s rotation. The inclination is always measured at the ascending node, and is therefore restricted between 0 and 180° .

The Right Ascension of the Ascending Node (RAAN), denoted by Ω , fixes the location of the ascending node, and by consequence the rotation of the orbit about the planet’s rotational axis. The ascending node is the location where the spacecraft’s orbit transitions from the southern hemisphere to the northern hemisphere. In casual parlance it is the location where the orbit “ascends” from below the equator to above the equator. The right ascension of the ascending node, then, is the angle between the ECI x axis and the ascending node. Variation of the RAAN appears to rotate the orbit about the ECI z axis.

The orbital plane itself is fixed through the argument of perigee ω , which describes the angle between the ascending node and the perigee of the orbit. The perigee being the location of the orbit that appears closest to the parent body’s center of mass. Varying the argument of perigee varies the rotation of the orbit within the orbital plane, effectively rotating it about the LVLH z axis.

Finally, with the orbit fully described in 3D space, the position of the spacecraft along the orbit is described by the true anomaly θ , which describes, in degrees, the angular displacement of the spacecraft from the perigee, with a true anomaly of $\theta = 0$ denoting that the spacecraft is currently at the perigee, and a true anomaly of $\theta = 180^\circ$ denoting that the spacecraft is at the apogee. The true anomaly increases from 0° to 360° throughout one orbit.

Now, the orbital diagram in Fig. [A.1](#) can be understood through the classical

orbital elements $\{a, e, i, \Omega, \omega, \theta\}$. The semi-major axis a describes the size of the orbit, the eccentricity e describes the shape of the orbital ellipse, the inclination i describes the angle between the orbital plane and the equatorial plane, the RAAN Ω describes the position of the ascending node along the equatorial plane, the argument of perigee ω describes the position of the perigee in the orbital plane, and the true anomaly θ describes the location of the spacecraft along the orbit. With the classical orbital elements any orbit around a parent body can be succinctly described and understood.

A.4 Orbital Perturbations

Newton’s universal law of gravitation, and the numerous mathematical corollaries that follow it provide a simple theory that captures a great deal of the phenomena associated with the orbital motion. The assumptions made while pursuing this theory, however, ignore details that are present in practice. In order to include these perturbations in mathematical descriptions of the system, they are appended to the dynamical equations as perturbing forces. Most of the forces present in orbit are significantly smaller than the acceleration caused by the parent body, but over the course of multiple orbits can nevertheless cause significant changes in a spacecraft’s final orbit if unaccounted for.

The greatest of these perturbations are three orders of magnitude smaller than the planetary acceleration, marked “GM” in Figure A.2, with diminishing effects as orbital radius increases.

Orbital perturbations will not be considered for the present analyses, since perturbations are of small magnitude, and active control techniques with adequate control power can easily compensate for their effects. Orbital perturbations can nonetheless have a sizable impact on spacecraft dynamics over large timescales. Some of the perturbations that act on spacecraft include:

- Orbital motion due to the oblateness of the Earth, called the J2 harmonic,
- Orbital motion due to other variations in the shape of the Earth, called higher order spherical harmonics,

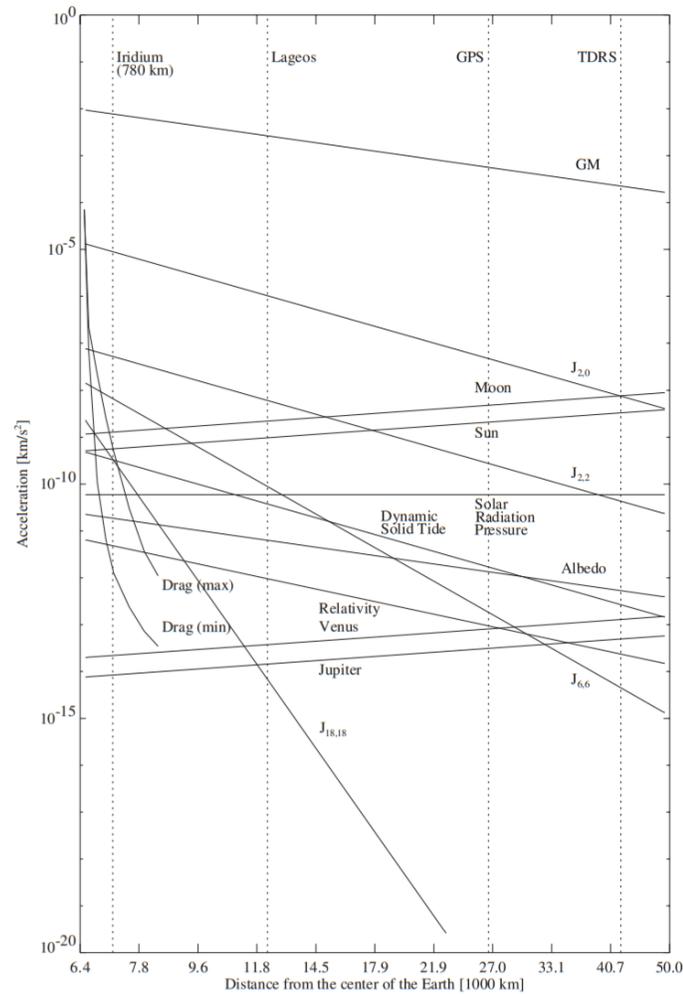


Figure A.2: Orbital Perturbation as compiled by Montenbruck and Gill in *Satellite Orbits: Models, Methods, and Applications* [2]

- Drag force on the spacecraft due to the presence of small numbers of atmospheric molecules even at high altitude,
- Momentum transfer from photons of light to the spacecraft, called solar radiation pressure,
- Torque acting on the spacecraft due to its shape and the unevenness of gravitation forces along the body of the spacecraft, called gravity gradient torque,
- and many more forces and torques, too small or numerous to mention.

Appendix B

Linear Systems Theory and Results

B.1 Introduction

The field of linear system analysis is very well matured. Results from the 1880s stand beside results from the 1960s as useful tools for understanding linear system responses, and the linear ordinary differential equations that describe them.

Over time, several conventions have also become common in the field. This appendix aims to characterize some of the most common results of linear and nonlinear systems theory that are useful to someone unfamiliar with the field. In this section, some preliminary results and tools of linear system analysis are covered.

B.1.1 Time-Varying Signal

Systems evolve over time, and as such the notion of how time affects a system must be precisely defined.

DEFINITION 8 (TIME-VARYING SIGNAL) A time-varying signal is composed of the input parameter, time $t \in \mathbb{R}$, and the output space \mathcal{U} . The time varying signal is the map from the parameter space to the output space. The mapping is written using function notation as

$$x : \mathbb{R} \rightarrow \mathcal{U} \tag{B.1}$$

pronounced “ x is the map from \mathbb{R} to \mathcal{U} ”. The function for this map, then, is written

$$x(t) \tag{B.2}$$

where the input to the mapping is t , and the output of the mapping is $x(t)$.

Due to the common usage of time-varying signals, the definition is shortened for

brevity. For the variable

$$x \in \mathcal{U} \tag{B.3}$$

the associated time-varying signal for x is defined as simply $x(t)$, or sometimes,

$$x(t) \in \mathcal{U} \tag{B.4}$$

Referring to the “time-varying” version of a signal is understood to mean the mapping from t to \mathcal{U} that corresponds to the size of x and holds a similar meaning to the variable x .

REMARK The purpose of clarifying the role of time-varying signals here is to simplify future descriptions of system dynamics. Whenever a variable is defined, such as

$$x \in \mathbb{R} \tag{B.5}$$

the reader knows that this variable can at any time be represented by a time-varying signal for input t . The transformation from static variable to time-varying signal is a simple one to understand in abstract, but must be done to many signals when considering dynamic systems. It becomes cumbersome to write

$$x : \mathbb{R} \rightarrow \mathcal{U} \tag{B.6}$$

every time a parameter goes from representing a single variable in an equation to becoming a time-varying signal, but otherwise must be done to stay rigorous. The natural understanding of how x relates to $x(t)$ is so ingrained that many texts omit the definition of how a variable x relates to its time-varying counterpart $x(t)$. To maintain precision in this text, the relationship between a variable and its time-varying counterpart is clarified. Furthermore, the relationship becomes important when considering certain types of operations on dynamic systems.

B.1.2 Linear Differential Equations and State-Space Representations

A great deal of physical systems can be characterized by linear differential equations, which are developed here and leveraged later to develop linear and adaptive controllers.

DEFINITION 9 (LINEAR DIFFERENTIAL EQUATION) A linear differential equation is a polynomial equation such that, for any constants a_0, \dots, a_n , $a_i \in \mathbb{R}$, $\forall i = 1, \dots, n$ and b_0, \dots, b_n , $b_i \in \mathbb{R} \forall i = 1, \dots, m$, and values $x(t), y(t) \in \mathbb{R}$

$$a_0 + a_1x(t) + a_2\dot{x}(t) + \dots + a_nx^{(n)}(t) = b_0 + b_1y(t) + b_2\dot{y}(t) + \dots + b_ny^{(n)}(t) \quad (\text{B.7})$$

with the superscript (n) denoting the n^{th} derivative in time of a variable.

A large body of research surrounding linear state-space system has been conducted, with collections of the results published in linear analyses textbooks, an example of which is Antsaklis [20]. From mathematically rigorous studies of the properties of these systems, several valuable results have been obtained that will be used to understand linear dynamic systems, and to probe changes to those systems.

Theorem B.1.1 (*State-Space Realization [20]*)

Any linear differential equation can be represented by a linear state-space system.

For example, the simple Newtonian double integrator system

$$F = ma = m\ddot{x}_p \quad (\text{B.8})$$

can be represented by an state-space system for the position variable $x_p \in \mathbb{R}$. The double integrator system can be written as

$$\ddot{x}_p = \frac{F}{m} \quad (\text{B.9})$$

which follows the form given by Eq. (2.39). The full representation of the double-integrator system is described as

$$\begin{bmatrix} \dot{x}_p \\ \ddot{x}_p \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_p \\ \dot{x}_p \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} F \quad (\text{B.10})$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_p \\ \dot{x}_p \end{bmatrix} + [0]F \quad (\text{B.11})$$

The form of linear state-space systems can be leveraged to determine the form of their solutions. Notice that the initial states of the system are used for the initial values of the state vector and that scaling of the input or states leads to scaling of the state derivatives. Since a large variety of physical systems can be represented by solutions of linear state-space systems, it is useful to determine the form of the solutions of these equations.

B.1.3 Solutions to Linear State-Space Systems

The evolution of the system states can be derived when no control input is present to produce the “zero-input response”. For the state vector \mathbf{x} the system becomes

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x}(t) \quad (\text{B.12})$$

$$\frac{1}{\mathbf{x}} d\mathbf{x} = \mathbf{A}dt \quad (\text{B.13})$$

integrating both sides yields

$$\mathbf{x}(t) = e^{\mathbf{A}(t-t_0)} \mathbf{x}_0(t_0) \quad (\text{B.14})$$

where the additional t_0 and $\mathbf{x}_0(t_0)$ are due to the constants of integration of the system. Equation (B.14) depends on the initial conditions, and the time varying term

$$\Phi(t, t_0) = e^{\mathbf{A}(t-t_0)} \quad (\text{B.15})$$

The characteristics of the state-transition matrix Φ , and thus the time-varying solution of the system $\mathbf{x}(t)$, is determined by the properties of \mathbf{A} under matrix exponentiation. Antsaklis [20] shows how the zero-input solution can be extended to include the control input and show that the full system response is given by

$$\mathbf{x}(t) = \Phi(t, t_0) \left(\mathbf{x}_0 + \int_{t_0}^t \Phi(\tau, t_0) \mathbf{B} \mathbf{u}(\tau) d\tau \right) \quad (\text{B.16})$$

Despite the ability of the control input to affect the final trajectory, the system dynamics are ultimately reliant on the properties of the state-transition matrix Φ , which affects both the response due to control inputs, states, and is directly related to the properties of \mathbf{A} under matrix exponentiation.

B.1.4 Laplace Transform and Linear System Analysis Tools

The Laplace transform is one tool that is commonly used in linear system analysis and controller design which simplifies representation of exponential behaviours.

DEFINITION 10 (LAPLACE TRANSFORM) The Laplace transform $\mathcal{L} : \mathbb{R} \rightarrow \mathbb{C}$ of a function $f : \mathbb{R} \rightarrow \mathbb{R}$ is given by

$$F(s) = \int_0^{\infty} f(t) e^{-st} dt = \mathcal{L}(f(t)) \quad (\text{B.17})$$

which creates the function $F(s)$ as a transformation of the function $f(t)$ into the Laplace space with complex Laplace variable $s \in \mathbb{C}$.

REMARK This is where a clear understanding of time-varying signals becomes practical. By definition the Laplace transform must take as its input a function. As such the Laplace transform of a signal $x(t) : \mathbb{R} \rightarrow \mathcal{U}$, written $\mathcal{L}(x(t))$, is equivalent to the Laplace transform of a function. This is how the Laplace transform of a time-varying

signal can be taken, since it is a mapping from time to output. In future, the Laplace transform will be used to understand and exploit system dynamics.

To see how the Laplace transform can be a useful tool, consider the Laplace transform of a simple exponential, e^{2t} .

$$\mathcal{L}(e^{2t}) = \int_0^{\infty} e^{2t} e^{-st} dt \quad (\text{B.18})$$

$$= \int_0^{\infty} e^{2t-st} dt \quad (\text{B.19})$$

$$F(s) = \frac{1}{s-2} \quad (\text{B.20})$$

The value of the Laplace transform can be taken at any point s , however the individual values of the function in Eq. (B.20) are only interesting at one point; at $s = 2$ the function increases to infinity, a feature which is called a pole of the Laplace transform of the function. If $F(s)$ had instead decreased to zero, the location would be called a zero of the Laplace function. The location of the pole in this example also corresponds to the value of the constant applied to the exponential input of the transformed function e^{2t} , and this is no coincidence. Notice that this pole at a positive s value corresponds to a positive exponential, and it is also true that poles at negative s values correspond to negative exponentials.

Much like how a Fourier transform clarifies the fundamental frequencies of a signal, the Laplace transform allows for the exponential behaviour of a signal to be determined. In some sense, the Laplace function is similar to a Fourier transform for exponential signals. Furthermore, due to Euler's identity

$$e^{(\alpha+\beta i)} = e^{\alpha}(\cos(\beta) + i \sin(\beta)) \quad (\text{B.21})$$

for $\alpha, \beta \in \mathbb{R}$, and the imaginary number $i = \sqrt{-1}$, there is a direct link between exponential behaviour, sinusoidal behaviour, and the form of the Laplace transform of a function.

Indeed, much like a Fourier transform returns large magnitudes at frequency values that correspond to frequencies that are present in the original signal, the Laplace

transform returns poles and zeros at complex values that correspond to the exponential and sinusoidal behaviour of the input signal.

Since the solutions of a linear system are given by an exponentiation of the state matrix \mathbf{A} , it follows that important aspects of the linear system solution are given by the Laplace transform of a state-space system. Indeed, Antsaklis [20] derives the relationship between a state-space system and its Laplace transform.

Theorem B.1.2 (Laplace Transform of an LTI State-Space System [20]) *For the LTI state-space system $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$, the associated Laplace transfer function is given by*

$$\mathbf{H}(s) = \frac{\mathbf{Y}(s)}{\mathbf{U}(s)} = \mathbf{C}(s\mathbf{I}_n - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D} \quad (\text{B.22})$$

where $\mathbf{H}(s)$ is the transfer function of the system, that takes the input $\mathbf{U}(s)$ and “transfers” it to the output $\mathbf{Y}(s)$, all in Laplace space, while \mathbf{I}_n is the $n \times n$ identity matrix.

For a system to be stable, each of the individual states should converge on a single value over time. Since the solutions of LTI state-space systems must be of the form of a matrix exponential, convergence only occurs when there are negative exponentials which converge to zero. Negative exponential behaviour occurs in the system whenever the system poles are negative, so it follows that the dynamic system can only be stable if all poles are in the left-hand plane. Hurwitz stability describes the usual notion of stability for linear state-space systems.

DEFINITION 11 (HURWITZ STABILITY [20]) A system $H(s)$ with pole-zero form

$$H(s) = \frac{(s + \lambda_{a1})(s + \lambda_{a2}) \dots (s + \lambda_{az})}{(s + \lambda_{b1})(s + \lambda_{b2}) \dots (s + \lambda_{bp})} \quad (\text{B.23})$$

is Hurwitz stable if all poles $\lambda_{bi} < 0 \forall i = 1, \dots, p$ for p poles. A system is Hurwitz marginally stable if $\lambda_{bk} \leq 0 \forall k = 1, \dots, p$, and $\lambda_{bk} = a \neq \lambda_{bj} \forall k = 1, \dots, p$, and $j = 1, \dots, p$ and $a \in \mathbb{C}$

REMARK Hurwitz stable transfer functions have all of their poles in the left-hand plane, while Hurwitz marginally stable transfer functions have a pole on the imaginary

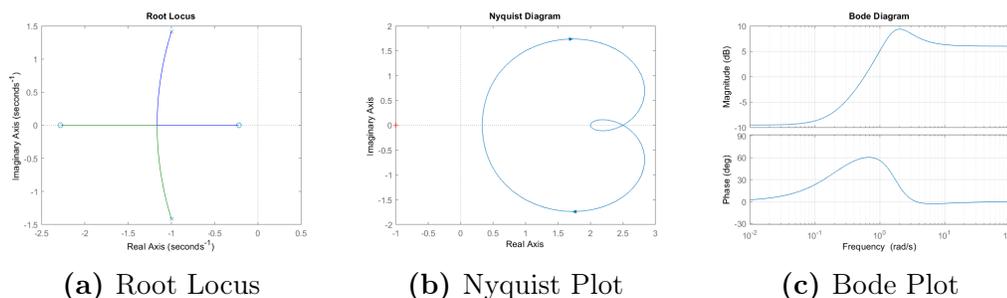


Figure B.1: Examples of Linear System Analysis Tools For $H(s) = \frac{2s^2+5s+1}{s^2+2s+3}$

axis, but no repeated poles on the imaginary axis. Repeated poles on the imaginary axis are not stable, such as in the double integrator system, which has two poles at the origin.

In addition to probing the zeros and poles of a transfer function through the Laplace transform, several other tools exist for determining the stability of a linear system. Demonstrations of the visual nature of some of these methods is demonstrated in Fig. B.1.

The various design tools available drastically simplify understanding of linear dynamic systems. Additionally, the compact representation of dynamic systems in Laplace space, as well as their clear input/output nature allow systems to be described in terms of block diagrams of transfer functions. Fig. B.2 shows how the general transfer function $H(s)$ described in Eq. (B.22) is denoted in a block diagram.

B.1.5 Block-Scheme Diagram Analysis

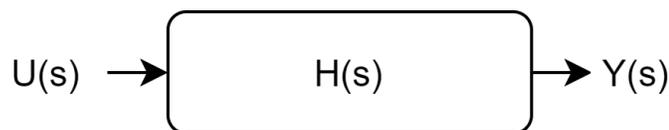


Figure B.2: A Single Transfer Function $H(s)$ with Inputs $U(s)$ and Outputs $Y(s)$

Lines in a block diagram denote individual signals, while blocks denote transformations from the input signal to an output signal. Multiple transfer functions can

also be chained together to create larger systems, whose representations can be simplified using standard operations. Figure B.3 shows how two transfer functions can be simplified into a single larger transfer functions when they are placed in parallel, series, or when feedback of a signal is used [20].

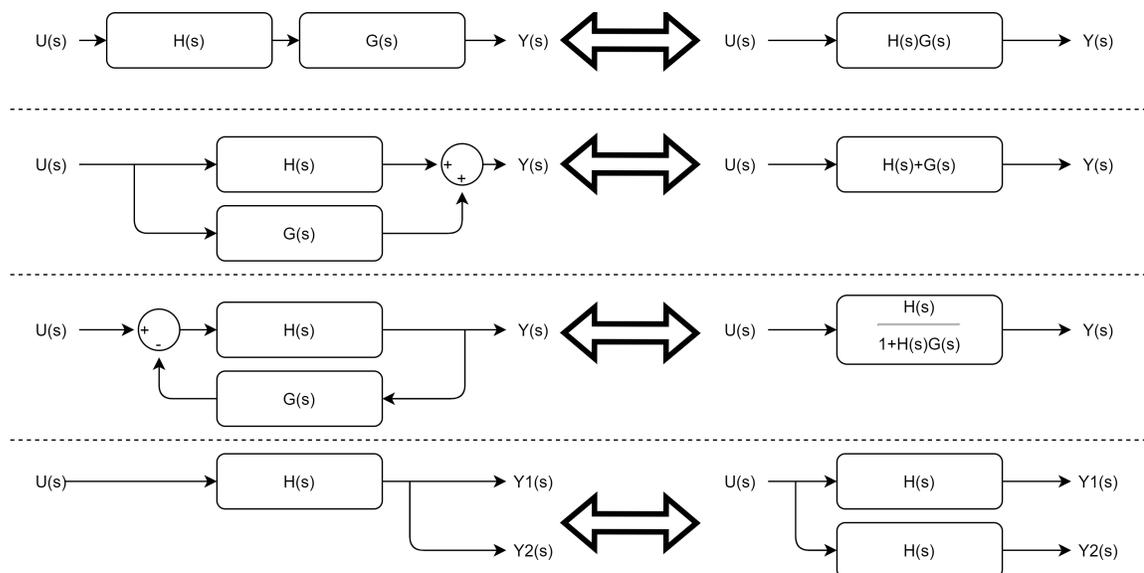


Figure B.3: The Block Diagram Operations of Multiplication, Addition, Feedback, and Shifting

B.2 Discrete Systems

The discrete state-space form is defined in the body of the text, where it is mentioned that discrete state-space systems can exactly describe continuous systems with zero-order hold inputs. A zero-order hold input simply refers to an input that is held to the same value between samples. The equations that describe the discrete counterpart to a linear state-space system are described below.

Theorem B.2.1 (Continuous to Discrete Realization [20]) *Any system that can be described by the continuous differential system $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$, when subjected to inputs $\mathbf{u}_d \in \mathbb{R}^m$ under zero-order hold of timestep $h \in \mathbb{R}$ and with output measurements $\mathbf{y}_d \in \mathbb{R}^p$ polled at the same timestep, is equivalently described by the discrete transition matrices*

$$\mathbf{A}_d = e^{\mathbf{A}h} \quad (\text{B.24})$$

$$\mathbf{B}_d = \left(\int_{\tau=0}^h e^{\mathbf{A}\tau} dt \right) \mathbf{B} \quad (\text{B.25})$$

$$\mathbf{C}_d = \mathbf{C} \quad (\text{B.26})$$

$$\mathbf{D}_d = \mathbf{D} \quad (\text{B.27})$$

and forms the new discrete state-space system $\{\mathbf{A}_d, \mathbf{B}_d, \mathbf{C}_d, \mathbf{D}_d\}$.

REMARK When the timestep h is taken to be very small, it should be noted that

$$\lim_{h \rightarrow 0} \mathbf{A}_d = \mathbf{I}_{n \times n} \quad (\text{B.28})$$

$$\lim_{h \rightarrow 0} \mathbf{B}_d = \mathbf{0}_{n \times m} \quad (\text{B.29})$$

for the zero matrix $\mathbf{0}_{n \times m} \in \mathbb{R}^{n \times m}$. This matches with the understanding that discrete state-space systems provide an update to the states, instead of a difference. When the timestep becomes sufficiently small, there is no noticeable difference in the states between timesteps, and commands that affect the derivative do not cause a noticeable difference in the state update. For sufficiently small timesteps, the discrete state-space update reverts to the continuous case.

B.3 Additional Properties of Linear Systems

Two more definitions of the properties of linear state-space matrices are useful for analyzing the trajectories and observations of a system.

DEFINITION 12 A system $\{\mathbf{A} \in \mathbb{R}^{n \times n}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$ is completely output controllable if $\forall \mathbf{x}_1(t_1), \mathbf{x}_2(t_2), t_2 > t_1 \exists \mathbf{u}(t)$ s.t.:

$$\mathbf{x}_2(t_2) = \Phi(t_2, t_0) \left(\mathbf{x}_1(t_1) + \int_{t_1}^{t_2} \Phi(t_0, \tau) \mathbf{B} \mathbf{u}(\tau) d\tau \right) \quad (\text{B.30})$$

That is to say that $\mathbf{x}_2(t_2)$ can be achieved through a some command timehistory $\mathbf{u}(t)$. Equivalently, a system is completely output controllable if the matrix

$$\mathcal{C} = \begin{bmatrix} CB & CAB & \dots & CA^{n-1}B & D \end{bmatrix} \quad (\text{B.31})$$

is of rank n .

REMARK A controllability matrix of rank n implies that the control vector is able to change each state independently through the input.

DEFINITION 13 A system $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$ is completely state observable if the system states \mathbf{x} can be determined from only inputs $\mathbf{u}(t)$ and outputs $\mathbf{y}(t)$, or equivalently the matrix

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix} \quad (\text{B.32})$$

has rank n .

REMARK An observability matrix of rank n implies that each state is independently described by combinations of the state inputs and outputs.

B.3.1 Minimal Realization

The equation for the Laplace space representation of a continuous time linear state-space system is given in Eq. (B.22). A side-effect of the Laplace representation of a system dynamics is that only controllable and observable states can be represented by a transfer function. Any state-space system that is put into a transfer function by Eq. (B.22) will not include any unobservable or uncontrollable states, and any realization of a transfer function will not include any unobservable or uncontrollable states. The state-space representation of a transfer function is therefore called the *minimal realization* of that transfer function.

The process for converting a transfer function to a state-space system is algorithmic and not illuminating on the attributes of either transfer functions or state-space systems. The reader can find a description of the process in the *A Linear Systems Primer* [20].

Appendix C

Additional Linear System Analysis and Controller Design

The development of simple adaptive control has continued for the better part of four decades now, with some of the earliest results occurring in the 1980s [58]. In that time, several key results have been found in the field. For reasons of brevity, the most important of these were not covered in the text. For the curious reader, the continuous time SAC stability proof, the discrete time SAC stability proof, and an intuitive understanding for the feedforward parallelization scheme are covered in this appendix.

C.1 Continuous Time SAC Stability Proof

A bulk of the work on SAC and its stability was undertaken by I. Barkana, and this stability proof is reproduced from his excellent tutorial on the fundamentals of SAC formulation *Simple Adaptive Control: The Optimal Model Reference - Short tutorial*. The proof uses Lyapunov stability theory in order to show that the plant state errors must go to zero. For those unfamiliar with the intricacies of Lyapunov stability theory, an excellent example is present in Sec. 2.5.3.

First, some necessary equations are developed that are used in the final Lyapunov equation. In order for the controller to achieve the ideal model response, it is necessary for the system to input control actions without any tracking error present. It is assumed that ideal gains exist for the system. Ideal gains are the set of gains that produce no tracking error for any set of commands. An algorithm for producing the ideal gains is mentioned in Sec. 2.5.3, however here their simple existence is sufficient. The ideal controller has control action

$$\mathbf{u}^*(t) = \mathbf{K}_x^* \mathbf{x}_m(t) + \mathbf{K}_u^* \mathbf{u}_m(t) = \mathbf{K}^* \mathbf{r}(t) \quad (\text{C.1})$$

where \mathbf{K}_x^* and \mathbf{K}_u^* are the ideal gains that convert the ideal model signals into the control actions necessary to make the plant response match the ideal response.

Although during operation the true plant state will not be known by the controller, the trajectories of the plant states under ideal inputs are necessary to show that the true system converges to the ideal system.

The plant is given by the system $\{\mathbf{A}_p, \mathbf{B}_p, \mathbf{C}_d, \mathbf{0}\}$, with states \mathbf{x}_p and output \mathbf{y}_p . The plant is given by the system $\{\mathbf{A}_m, \mathbf{B}_m, \mathbf{C}_m, \mathbf{0}\}$, with states \mathbf{x}_m and output \mathbf{y}_m . The input to the ideal model is \mathbf{u}_c .

It is known beforehand that the ideal plant will follow the ideal model trajectory, which provides the state-space system

$$\dot{\mathbf{x}}_p^*(t) = \mathbf{A}_p \mathbf{x}_p^*(t) + \mathbf{B}_p \mathbf{u}_p^*(t) \quad (\text{C.2})$$

$$\mathbf{y}^*(t) = \mathbf{y}_m(t) = \mathbf{C}_m \mathbf{x}_m(t) = \mathbf{C}_p \mathbf{x}_p^*(t) \quad (\text{C.3})$$

Given by the ideal states \mathbf{x}_p^* and the ideal output \mathbf{y}_p^* . What will be shown by the stability proof is that the true plant states approach the ideal plant states. The difference between the states of the ideal and non-ideal systems can be considered, which is found to be

$$\mathbf{e}_x(t) = \mathbf{x}_p^*(t) - \mathbf{x}_p(t) \quad (\text{C.4})$$

and the tracking error can be described in terms of this new state error

$$\mathbf{e}_y(t) = \mathbf{y}_m(t) - \mathbf{y}_p(t) = \mathbf{C}_m \mathbf{x}_m(t) - \mathbf{C}_p \mathbf{x}_p(t) \quad (\text{C.5})$$

$$= \mathbf{C}_p \mathbf{x}_p^*(t) - \mathbf{C}_p \mathbf{x}_p(t) \quad (\text{C.6})$$

$$= \mathbf{C}_p \mathbf{e}_x(t) \quad (\text{C.7})$$

The dynamics of the real system can now be considered, for which the plant updates follow

$$\dot{\mathbf{x}}_p = \mathbf{A}_p \mathbf{x}_p(t) + \mathbf{B}_p \mathbf{K}(t) \mathbf{r}(t) \quad (\text{C.8})$$

When the ideal trajectories are subtracted from the real trajectories, the state error dynamic equation that results is

$$\dot{\mathbf{e}}_x(t) = (\mathbf{A}_p - \mathbf{B}_p \mathbf{K}_e^* \mathbf{C}_p) \mathbf{e}_x(t) - \mathbf{B}_p (\mathbf{K}(t) - \mathbf{K}^*) \mathbf{r}(t) \quad (\text{C.9})$$

$$= \mathbf{A}_c \mathbf{e}_x(t) - \mathbf{B}_p (\mathbf{K}(t) - \mathbf{K}^*) \mathbf{r}(t) \quad (\text{C.10})$$

The stability proof can now be performed. Consider a Lyapunov function of

$$V(t) = V_1(t) + V_2(t) \quad (\text{C.11})$$

$$V_1(t) = \mathbf{e}_x^T(t) \mathbf{P} \mathbf{e}_x(t) \quad (\text{C.12})$$

$$V_2(t) = \text{Tr} \left[(\mathbf{K}(t) - \mathbf{K}^*) \mathbf{\Gamma}_I^{-1} (\mathbf{K}(t) - \mathbf{K}^*)^T \right] \quad (\text{C.13})$$

where \mathbf{P} is the Lyapunov stability matrix introduced in Eq. (2.42) that meets the ASPR requirement of the system. The first Lyapunov term, $V_1(t)$ is considered. The time derivative of $V_1(t)$ for an ASPR system is found to be

$$\dot{V}_1(t) = \dot{\mathbf{e}}_x^T \mathbf{P} \mathbf{e}_x + \mathbf{e}_x^T \mathbf{P} \dot{\mathbf{e}}_x \quad (\text{C.14})$$

$$\dot{V}_1(t) = \mathbf{e}_x^T(t) (\mathbf{P} \mathbf{A}_c + \mathbf{A}_c^T \mathbf{P}) \mathbf{e}_x(t) - 2 \mathbf{e}_y^T(t) (\mathbf{K}(t) - \mathbf{K}^*) \mathbf{r}(t) \quad (\text{C.15})$$

However, due to the ASPR requirements, we know that $\mathbf{P} \mathbf{A}_c + \mathbf{A}_c^T \mathbf{P}$ must also be a negative semi-definite matrix $-\mathbf{Q}$. There remains, however, the final term based on tracking error which stops us from proving stability. Differentiation of the second term of the Lyapunov function yields, when using definitions for the derivatives of the gains

$$\dot{V}_2(t) = 2\text{Tr}[(\mathbf{K}(t) - \mathbf{K}^*(t))\mathbf{\Gamma}_I^{-1}\dot{\mathbf{K}}_I(t)] \quad (\text{C.16})$$

$$\dot{V}_2(t) = 2\text{Tr}[(\mathbf{K}(t) - \mathbf{K}^*(t))\mathbf{\Gamma}_I^{-1}(\mathbf{e}_y(t)\mathbf{r}^T(t)\mathbf{\Gamma}_I)^T] \quad (\text{C.17})$$

$$\dot{V}_2(t) = 2\mathbf{e}_y^T(t)[\mathbf{K}(t) - \mathbf{K}^*]\mathbf{r}(t) \quad (\text{C.18})$$

Where now the derivative $\dot{V}_2(t)$ cancels out the additional term in $\dot{V}_1(t)$ to leave only the negative semi-definite portion of the Lyapunov derivative. The full Lyapunov derivative is now

$$\dot{V}(t) = -\mathbf{e}_x^T(t)\mathbf{Q}\mathbf{e}_x(t) \quad (\text{C.19})$$

with positive definite matrix \mathbf{Q} , implying $\mathbf{e}_x^T(t)\mathbf{Q}\mathbf{e}_x(t)$ must always be positive, with the negative finally implying that $\dot{V}(t)$ must always be negative or equal to zero. The SAC architecture and gain adaptation scheme must therefore make the state errors approach zero, implying that the ASPR plant output approaches the model reference output.

Notably, the Lyapunov derivative does not contain the adaptation parameter $\mathbf{\Gamma}_I$ or the adaptive gains $\mathbf{K}(t)$, suggesting that the choice of adaptation parameter does not affect the final system stability, while also suggesting that the adaptive gains are not necessarily stable to any one value and there are multiple combinations of gains that achieve perfect model following. Some of the implications of the stability are covered in the main text.

Stability for SAC can be repeated while including the proportional adaptations, however the analysis is similar, without particular insight. It is clear from the stability analysis that the proportional adaptations do not contribute to instability. For a full derivation of the proportional adaptation stability proof, consider reading *Direct Adaptive Control Algorithms: Theory and Applications* by H. Kaufman, I. Barkana, and K. Sobel [60].

C.2 Discrete Time SAC Stability Proof

Critically, a bounded stability proof for SAC in discrete systems has been found. Some of the issues with SAC implementations sprang from the presence of a continuous-time stability proof that insinuated that all values of adaptation parameter would be stable. Since it is plainly clear that SAC can go unstable when improper adaptation parameters are chosen, the question remains what could be causing the issue. Although it is possible discretization plays a part, the Lyapunov difference suggests it is not a significant factor.

The discrete proof for SAC stability is reproduced in part here. The discrete SAC stability proof uses the same assumptions as the linear stability proof, and comes to similar conclusions. First, signals are defined before being used to calculate the discrete Lyapunov function and its difference.

The discrete ideal model at iteration k is characterized by the state-space system

$$\mathbf{x}_m(k+1) = \mathbf{A}_m \mathbf{x}_m(k) + \mathbf{B}_m \mathbf{u}_m(k) \quad (\text{C.20})$$

$$\mathbf{y}_m(k) = \mathbf{C}_m \mathbf{x}_m(k) + \mathbf{D}_m \mathbf{u}_m(k) \quad (\text{C.21})$$

for $\mathbf{x}_m \in \mathbb{R}^q$, and $\mathbf{y}_m(k), \mathbf{u}_m(k) \in \mathbb{R}^m$. Similarly, the ASPR discrete plant response is given by

$$\mathbf{x}_p(k+1) = \mathbf{A}_p \mathbf{x}_p(k) + \mathbf{B}_p \mathbf{u}_p(k) \quad (\text{C.22})$$

$$\mathbf{y}_p(k) = \mathbf{C}_p \mathbf{x}_p(k) + \mathbf{D}_p \mathbf{u}_p(k) \quad (\text{C.23})$$

for $\mathbf{x}_p \in \mathbb{R}^q$, and $\mathbf{y}_p(k), \mathbf{u}_p(k) \in \mathbb{R}^m$.

The tracking error is still given by

$$\mathbf{e}_y(k) = \mathbf{y}_m(k) - \mathbf{y}_p(k) \quad (\text{C.24})$$

for augmented discrete ASPR system output \mathbf{y}_p . It is not typical for discrete systems to be ASPR, so it is likely that the ASPR plant in the stability proof has been

augmented somehow.

The ideal output is constructed as

$$\mathbf{u}_p^*(k) = \mathbf{K}^* \mathbf{r}(k) \quad (\text{C.25})$$

where now

$$\mathbf{K}^* = \begin{bmatrix} \mathbf{K}_e^* & \mathbf{K}_x^* & \mathbf{K}_u^* \end{bmatrix} \quad (\text{C.26})$$

$$\mathbf{r}(k) = \begin{bmatrix} \mathbf{e}_y(k) \\ \mathbf{x}_m(k) \\ \mathbf{u}_c(k) \end{bmatrix} \quad (\text{C.27})$$

Similar to the linearity condition in the SAC gain matching conditions, in order for ideal model following to occur, the plant states must be some multiple of the ideal model outputs. Recall for the matching conditions

$$\begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} \\ \mathbf{K}_x^* & \mathbf{K}_u^* \end{bmatrix} \begin{bmatrix} \mathbf{C}_p \\ \mathbf{D}_p \end{bmatrix} = \begin{bmatrix} \mathbf{C}_m \\ \mathbf{D}_m \end{bmatrix} \quad (\text{C.28})$$

The state error for the discrete system is now

$$\mathbf{e}_x(k) = \mathbf{x}_p^*(k) - \mathbf{x}_p(k) \quad (\text{C.29})$$

The choice of adaptation parameter $\mathbf{\Gamma}_I$ in the discrete case is made so as to include the constant of proportionality for discrete Euler integration. For this reason the discrete update equation for SAC gains is simply

$$\mathbf{K}_I(k+1) = \mathbf{K}_I(k) + \mathbf{e}_y(k) \mathbf{r}^T(k) \mathbf{\Gamma}_I \quad (\text{C.30})$$

Similar to how there is an equivalent feedback system for the continuous ASPR condition, there is an equivalent feedback system for the discrete ASPR condition. Using the discrete ASPR conditions, the equivalent system matrices are

$$\mathbf{A}_c = \mathbf{A}_p - \mathbf{B}_p \mathbf{K}_e^* \mathbf{C}_p \quad (\text{C.31})$$

$$\mathbf{B}_c = \mathbf{B}_p (\mathbf{I}_m + \mathbf{D}_p \mathbf{K}_e^*)^{-1} \quad (\text{C.32})$$

$$\mathbf{C}_c = (\mathbf{I}_m + \mathbf{K}_e^* \mathbf{D}_p)^{-1} \mathbf{C}_p \quad (\text{C.33})$$

$$\mathbf{F}_c(k) = (\mathbf{A}_p \mathbf{S}_{11} - \mathbf{S}_{11} \mathbf{A}_m + \mathbf{B}_p \mathbf{K}_x^*) \mathbf{x}_m(k) \quad (\text{C.34})$$

$$+ (\mathbf{A}_p \mathbf{S}_{12} - \mathbf{S}_{11} \mathbf{B}_m + \mathbf{B}_p \mathbf{K}_u^*) \mathbf{u}_c(k) \quad (\text{C.35})$$

where $(\mathbf{I}_m + \mathbf{D}_p \mathbf{K}_e^*)$ is provably nonsingular for an ASPR system. The tracking error can now be rearranged to be

$$\mathbf{e}_y(k) = \mathbf{C}_c \mathbf{e}_x(k) - \mathbf{D}_c (\mathbf{K}(k) - \mathbf{K}^*) \mathbf{r}(k) \quad (\text{C.36})$$

The difference equation for the state error can now be found, after some manipulation to be

$$\mathbf{e}_x(k+1) = \mathbf{A}_p \mathbf{e}_x(k) - \mathbf{B}_p (\mathbf{K}(k) - \mathbf{K}^*) \mathbf{r}(k) - \mathbf{F}_c(k) \quad (\text{C.37})$$

Finally, the Lyapunov stability analysis can be done.

For the Lyapunov function, similar to the continuous case

$$V(k) = \mathbf{e}_x^T(k) \mathbf{P} \mathbf{e}_x(k) + \text{Tr} \left[(\mathbf{K}(k) - \mathbf{K}^*) \mathbf{\Gamma}_I^{-1} (\mathbf{K}(k) - \mathbf{K}^*)^T \right] \quad (\text{C.38})$$

A slight change of variables is done here for clarity. Where the original discrete ASPR conditions were phrased as

$$\mathbf{P}_d - \delta \mathbf{P}_d - \mathbf{A}_{c,d}^T \mathbf{P}_d \mathbf{A}_{c,d} = \mathbf{L}^T \mathbf{L} \quad (\text{C.39})$$

$$\mathbf{C}_d - \mathbf{A}_{c,d}^T \mathbf{P}_d \mathbf{B}_d = \mathbf{L}^T \mathbf{W} \quad (\text{C.40})$$

$$\mathbf{D}_d^T + \mathbf{D}_d - \mathbf{B}_d^T \mathbf{P}_d \mathbf{B}_d = \mathbf{W}^T \mathbf{W} \quad (\text{C.41})$$

The first condition is equivalently stated with the variable \mathbf{Q}

$$\mathbf{P}_d - \mathbf{Q} - \mathbf{A}_{c,d}^T \mathbf{P}_d \mathbf{A}_{c,d} = \mathbf{L}^T \mathbf{L} \quad (\text{C.42})$$

After significant algebra, the Lyapunov difference becomes

$$\begin{aligned} \Delta \mathbf{V} = & -\mathbf{e}_x^T(k) \mathbf{Q} \mathbf{e}_x & (\text{C.43}) \\ & - [\mathbf{e}_x^T(k) \mathbf{L}^T - \mathbf{r}^T(k) (\mathbf{K}(k) - \mathbf{K}^*)^T \mathbf{W}^T] [\mathbf{L} \mathbf{e}_x(k) - \mathbf{W} (\mathbf{K}(k) - \mathbf{K}^*) \mathbf{r}(t)] \\ & - 2\text{Tr}[(\mathbf{e}_y(k) \mathbf{r}^T(k) \mathbf{\Gamma}_I) \mathbf{\Gamma}^{-1} (\mathbf{e}_y(k) \mathbf{r}^T(k) \mathbf{\Gamma}_I)^T] \\ & - 2\mathbf{e}_x(k+1) \mathbf{P} \mathbf{F}_c(k) - \mathbf{F}_c(k)^T \mathbf{P} \mathbf{F}_c(k) \end{aligned}$$

which is not negative semi-definite. Notably, the discrete ASPR stability conditions are all present in the final difference. Nothing can be said about the stability of the system, then. However, the system does typically have a negative Lyapunov difference.

The most full description of the above stability proof and its results can be found in *Direct Adaptive Control Algorithms: Theory and Applications* by H. Kaufman, I. Barkana, and K. Sobel [60]. Although the initial result in that text is that the discrete difference of the Lyapunov function is not negative semi-definite, further results have been able to improve on that result somewhat, with *Almost Passivity and Simple Adaptive Control in Discrete-Time Systems* [63] by Barkana and Ruslan proving bounded stability of the discrete system.

Notably, in Shibata, Fujinaka and Sun [65], a slightly different controller formulation (the *augmentation of adaptation* formulation seen in Sec. 2.5.5) claims to yield a fully negative semi-definite Lyapunov function difference. That claim has been repeated in at least one other paper by Fujinaka and Sun, however the (current) author has not been able to reproduce the Lyapunov difference achieved in that paper. Results by Barkana stating a fully negative semi-definite Lyapunov difference have not been found. If Barkana had found a stability proof for discrete SAC it is likely that it would be celebrated in other works. It is currently unclear if the discrete stability of SAC has been sufficiently proven by existing literature.

C.3 Parallel Feedforward Compensation of non-ASPR systems

The result by Barkana in the mid 1980s was that feedforward parallelization could be used by continuous and discrete systems to meet the (discrete or continuous) ASPR requirements [59]. The result can be rederived through basic linear algebra identities, which are long and tedious. Instead, an intuitive understanding of why feedforward parallelization works is developed for the reader here.

For non-ASPR system $G(s)$

$$G(s) = \frac{B(s)}{A(s)} \quad (\text{C.44})$$

and stabilizing controller

$$H(s) = \frac{Q(s)}{P(s)} \quad (\text{C.45})$$

then the closed loop system which is asymptotically stable is

$$G_c(s) = \frac{B(s)Q(s)}{A(s)P(s) + B(s)Q(s)} \quad (\text{C.46})$$

meaning that all of its poles are in the left-hand plane.

The stabilizing controller $H(s)$ can be used to create a parallel feedforward compensator and augmented plant $G_a(s)$ to be controlled by a SAC following

$$G_a(s) = G(s) + H^{-1}(s) = \frac{B(s)}{A(s)} + \frac{P(s)}{Q(s)} = \frac{A(s)P(s) + B(s)Q(s)}{A(s)Q(s)} \quad (\text{C.47})$$

which is the same as

$$G_a(s) = \frac{A(s)P(s) + B(s)Q(s)}{A(s)Q(s)} = G_c^{-1}(s)G(s) \quad (\text{C.48})$$

Notice that the stabilized closed loop system $G_c(s)$ is inverted and multiplied by $G(s)$, implying $G_a(s)$ has zeroes where the poles of the closed loop plant originally were while retaining the poles of the original transfer function $G(s)$. The augmented system, therefore, has zero and pole positions that are very similar, with the only

distinction being caused by the difference between the closed-loop and open-loop pole-zero pairs. Additionally, due to the requirement for the stabilizing controller $H(s)$ to be minimum phase, no zeros have been added to the closed-loop response or unstable poles to the inverse response $G_c^{-1}(s)$. A root locus analysis of the augmented system would show that the response poles are now accompanied by zeros nearby. Finally, closing the loop for some negative feedback gain K_e gives the closed loop transfer function

$$G_s(s) = \frac{K_e(A(s)P(s) + B(s)Q(s))}{K_e(A(s)P(s) + B(s)Q(s)) + A(s)Q(s)} = \frac{K_eG(s)}{K_eG(s) + G_c(s)} \quad (\text{C.49})$$

which now must have poles and zeros that are very close to one another. Indeed, in the limit, as K_e reaches a sufficiently large value, the contribution due to the stabilizing controller approaches zero while the poles and zeros of the plant transfer function approach one another, allowing the frequency SPR conditions to be met.

Since the augmented system becomes SPR when a sufficiently large negative feedback gain is applied it meets the definition for an ASPR system, and can thus be controlled by a SAC. Similar thinking applies to the discrete case.