EFFICIENT HIERARCHICAL GLOBAL MOTION PLANNING FOR AUTONOMOUS VEHICLES

BY

WILLIAM TODD CERVEN

B.S., University of Illinois at Urbana-Champaign, 1997
M.S., University of Illinois at Urbana-Champaign, 1999

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Aeronautical and Astronautical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2003

Urbana, Illinois

Abstract

Motion planning has become an increasingly important task as autonomy in mechanical systems has gained in popularity. Such systems must be able to plan new trajectories and controls reliably and rapidly in response to inputs. The challenges in creating such planners are decidedly non-trivial, including issues such as algorithm convergence, its correspondence to system controllability, optimality of solution, computational complexity, and dynamic environments. In response to these challenges, a hierarchical algorithm will be introduced that provides a) decreased computational complexity though symmetry and a hybrid systems representation of the dynamics, b) utilization of local controllability through a local planning algorithm, c) minimization of a cost functional, d) a randomized planner for obstacle avoidance, and e) convergence guarantees. Applications include autonomous vehicles, sensor-based planning, and multiple vehicle coordination in ground-based, underwater, atmospheric, and orbital environments.

To my grandfather, James R. McGhee,
my parents,
my wife,
and to God, from Whom all blessings flow.

Acknowledgements

I would like to express my appreciation to my co-advisors, Prof. Francesco Bullo and Prof. Victoria Coverstone. Both have provided valuable advice and encouragement to me in this process. In particular, I would like to thank Prof. Bullo for his loyalty in funding, humility and consideration in advising, and incredible insight into differential geometry and dynamical systems theory. To Prof. Coverstone, I owe thanks for her support of my exploration of varied topics in my studies as well as her willingness to meet with me even at a moment's notice to discuss anything from technical subjects to my latest trip.

I would be remiss if I did not also thank the other members of my committee, Prof. Emilio Frazzoli and Prof. John Prussing. Prof. Frazzoli's advice and suggestions have been pivotal in formulating and carrying out the research documented here. Prof. Prussing has always left his door open to me and partially provided the inspiration for the orbital examples given in the text.

My experience, both technical and otherwise, has been enriched by interactions with other faculty as well. While I am indebted to all of my professors for their teaching and encouragement over the years, a few stand out for their particular contributions. Prof. Steven Lavalle and his students, Peng Cheng and Steve Lindemann, were stimulating sources of conversation and advice on algorithmic issues. I owe Prof. Harry Hilton thanks for his friendship, advice, and support on many levels over the years. Another professor for whose advice I am indebted is Prof. N. Sri Namachchivaya, whose consistent faith in me helped when my enthusiasm would flag. Last of these, I would like to thank Prof. Wayne Solomon, who took me on as naive and unprepared undergrad and got me started in research.

Appreciation is also due to my cadre of cohorts in Prof. Bullo's and Prof. Coverstone's research groups, notably Timur Karatas, Jorge Cortes, Bill Hartmann, Bill Mason, Jen

Hargens, Aaron Trask, and Byoungsam Woo. Their support, comraderie, and humor helped me keep my (in?)sanity. Of course, I cannot ever forget the ladies of the department administrative staff, without whom no one would be able to accomplish anything. Of these, Diane Jeffers is most dear, as I have wasted a ridiculous amount of her time for no better reason than to share a joke.

Although many people have had a hand in my success, I truly owe it all to my parents, who have shown love, commitment, wisdom, and Godliness to me all of my life. Without their support and encouragement, I would not be where I am today. Thanks also goes to my overgrown little brother, who has always looked up to me and inspired me more often than not.

Last of all, I would like to thank my wife, Marie, for being willing to marry a punny graduate student, support him through his studies, and still love him after two years.

This research was supported under a National Science Foundation Graduate Fellowship, the U.S. Army Research Office under Grant DAAD-190110716 and by the National Science Foundation under Grant IIS-0118146. Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the author and do not necessarily reflect the views of the National Science Foundation.

Contents

List of Figures xi

List of Tables xiii

List of Symbols xiv

1 Introduction 1

1.1 Motivating Applications 1

1.1.1 Autonomous vehicles 2

1.1.2 Multiple Vehicle Coordination 2

1.1.3 Sensor-Based Planning 2

1.2 Background 3

1.3 Approach and Contribution 6

1.4 Overview 7

2 Mechanical System Dynamics 8

2.1 Mechanical Control Systems 8

2.2 Symmetry 10

2.3 Trajectory Primitives 11

2.4 Vehicle Models 13

2.4.1 Generalized Vehicle Model 13

2.4.2 Systems with a negligible gravity term 17

2.4.3 Systems with a constant gravity term 20

2.4.4 Systems with an inverse square gravity term 21

3 Series-based Local Planning **26**
3.1 Introduction . . . 26
3.2 A class of polynomial control systems . . . 27
3.2.1 Operator and function norms . . . 28
3.2.2 Evolution as series expansion . . . 29
3.2.3 Accessibility and nilpotency . . . 30
3.2.4 Series expansion about a trajectory . . . 31
3.3 Formulation of motion planning and minimum energy planning problems 32
3.3.1 Base functions for the control inputs . . . 32
3.3.2 Motion planning with base functions . . . 33
3.3.3 Minimum energy planning (without base functions) . . . 34
3.4 Solving the planning problems via inversion . . . 36
3.4.1 Existence of solution for linearly controllable systems . . . 38
3.4.2 Existence of solution for linearly uncontrollable systems . . . 39
3.4.3 Computational approaches . . . 42
3.5 Simulation . . . 44
3.5.1 PVTOL with Damping Example . . . 44
3.5.2 Implementation . . . 46
3.5.3 Results . . . 48
3.6 Conclusions . . . 53

4 Maneuver Automaton **54**
4.1 Preliminaries . . . 55
4.1.1 Graphs and Automata . . . 55
4.1.2 Trajectory Primitives . . . 56
4.2 Maneuver Automaton Framework . . . 57
4.3 Well-posedness and Controllability . . . 61
4.3.1 Discrete System Properties . . . 62
4.3.2 Continuous System Properties . . . 63
4.3.3 Maneuver Automaton Properties . . . 67
4.4 Optimal Control on the Maneuver Space . . . 71
4.5 Computational Search Methodology . . . 74

4.5.1 Obstacle-Free Dynamic Programming . . . 74
4.5.2 Convergence of the Dynamic Program . . . 77
4.5.3 Optimality of the Dynamic Program . . . 81
4.6 Algorithm Description . . . 81
4.6.1 Preprocessing . . . 82
4.6.2 Online computation . . . 82
4.7 Earth to Mars Transfer Example . . . 83
4.7.1 Mathematical Model . . . 83
4.7.2 Implementation . . . 87
4.7.3 Results . . . 88
4.8 Conclusions . . . 91
4.9 Introduction . . . 92
4.10 Rapidly Exploring Random Trees . . . 94
4.10.1 Background . . . 94
4.10.2 A New Approach . . . 96
4.10.3 RRT Convergence . . . 98
4.11 Error Correction and the RRT . . . 100
4.12 Algorithm description . . . 101
4.13 Asteroid Landing Example . . . 102
4.13.1 Problem Set-up . . . 102
4.13.2 Results . . . 103
4.14 Conclusion . . . 107

5 Conclusions **109**
5.1 Summary . . . 109
5.2 Future Directions . . . 110
5.2.1 Addressing safety and real-time issues . . . 110
5.2.2 Expansion of Series-Based Planning . . . 111
5.2.3 Optimal Primitive Choice . . . 111
5.2.4 Optimal Trajectory Generation . . . 111
5.2.5 Incorporation of Sensors into Open Loop Control . . . 111
5.2.6 Multiple Vehicle Coordination and Cooperation . . . 113

Bibliography **114**

A Local planner supplementary development and proofs **128**

A.1 Minimum energy planning with base functions 128

A.2 Solution of polynomial system as series expansion 130

A.3 Convergence of iterative algorithm 130

A.4 Convergence of power series inversion 134

A.5 Local Planner Control Bounds 136

Vita **138**

List of Figures

2.1 Symmetry 11

3.1 Diagram of the PVTOL model. 44
3.2 Final position error for motion planning algorithm (left), minimum energy planning algorithm (right). 48
3.3 Cost comparison between motion planning with base functions and minimum energy planning algorithms for series truncated at order 6. 50
3.4 Trajectory comparison of the motion planning with base functions and minimum energy planning algorithms for series truncated at order 6 with their linear counterpart. 51
3.5 Resulting motions from minimum energy planning algorithm for PVTOL series truncated at orders 1 (top) and 2 (bottom), starting at rest at the origin with desired final condition of a small negative velocity at the circled location. 52

4.1 The Maneuver Automaton: On left, evolution in the discrete space is in the horizontal plane and evolution along the symmetry group is shown on the vertical axis. At right is a projection of the evolution onto the discrete space. 61
4.2 Local Planner-based Maneuver 62
4.3 Lie bracket 65
4.4 Orbital Elements 85
4.5 Grid representation of $\mathcal{H}$. 87
4.6 Optimal cost distribution. 88
4.7 Earth-to-Mars Setup and Final Trajectory 89

4.8 Earth-to-Mars control history in orbit-referenced frame// u_1: tangential thrust, u_2: radial thrust, u_3: normal thrust 90
4.9 RRT Exploration 95
4.10 New RRT Variation 98
4.11 Error Correction 101
4.12 Coding Structure 102
4.13 Ida Landing Setup 103
4.14 Precalculated Approximate Cost 104
4.15 Ida Random Tree and Final Path, static relative frame (left), inertial frame with time variation (right) 105
4.16 Ida Results: Snapshots of Trajectory 106
4.17 Ida Control History in Orbit-referenced Frame 106
4.18 Ida Histogram of 50 Cases 108

List of Tables

4.1 Discretized Space 87
4.2 Discrete Steps 89
4.3 Ida Discretized Space 104
4.4 Randomized Planner Iterations: Initial and Target points 107
4.5 Randomized Planner Iterations: $EXTEND$ Results 108

List of Symbols

Symbol	: Description and page(s) when applicable
$\dot{\cdot}$	: Augmented System quantities/Relative motion variables, 35
$\hat{\cdot}$	: Skew-symmetric matrix operator, 14
$Ad_{u_k \ldots u_1} \mathbb{Y}_i(x)$, $\mathbb{Y}_i$	: Discrete-time vector fields, 66
α	: Integrable function, 63
$A(t)$	: Linear time-varying EOM term, 13
$\mathcal{A} = \{A_0, A_1, \ldots, A_k\}$:	Attraction sequence, 99
B	: Linear control matrix, 13
$B_1,\ B_2$	: Submatrices of B, 15
B_i	: Ball in $\mathbb{R}^n$ of volume ϵ_i, 63
$B_\delta(0) \subset \Xi$	: Ball of radius δ about the origin, 61
$(B_\rho(x^0))$	: Ball of radius ρ centered at x_0, 63
β_k	: Arbitrary positive constant, 59
$\binom{a}{k}$	: Binomial coefficient, 29
$B_i^r b$	: Matrix of the orthonormal basis of B_i, 15
B_v, B_ω	: Constant linear input matrices, 14
$\mathcal{C}$	: Catalan generating function, 29
χ	: Variable in $\mathbb{R}^n$, 42
C^∞	: Smooth Class, 8
c_k	: Catalan numbers, 29
clos U	: The closure of U, 66

$\overline{\text{Lie}}(\{Y_1 \dots Y_m\})$: Involutive closure, 66

$\cdot_c$: Controllable subspace, 40

$\mathcal{C}_\Xi$: Obstacle-free configuration space of Ξ, 94

$d_1,\ d_2$: Series convergence constants, 30

$D_1,\ D_2$: Modified series convergence constants, 36

$\delta\tau,\ \delta\Delta h$: Perturbations on continuous control variables, 69

Δh_k : Starting continuous state of $k+1$th time step relative to propagated state on kth reference trajectory, 59

$d_{\mathcal{H}},\ d_{\mathcal{U}}$: Maximum distances between any two grid element nodes in state and control spaces, 78

$\mathcal{D}$: Distribution on a open subset of Q, 65

$d\varphi_f(q)$: Differential of φ_f at point $q \in Q$, 64

$D_v,\ D_\omega$: Translational and rotational drag, 14

δx : Symmetry group variable on $\mathbb{R}^3$, 18

e : Relative variation variable, 31

$\epsilon,\ \epsilon_i$: Nonzero positive scalar constants, usually assumed to be small, 63

η : Time scaling factor, 11

e^{At} : Matrix exponential, 40

$f(y), f_k$: Generalized series representation of a function, 36, 60

$\mathcal{F} : \mathbb{H} \times \mathcal{I}_\tau(\gamma) \to \Xi$: Mapping of the hybrid state to the full state space, 60

$f^*(\cdot, t^*)$: Time-scaled transformation of vector field $f(\cdot, t)$, 11

$f^{[2]}(x, x)$: Quadratic EOM term, 13

$\mathcal{F}_\delta$: Maps the hybrid state and controls to the perturbed state about the reference trajectory, 60

f_i : Piecewise constant functions, 63

F_{nc} : Map describing non-conservative forces, 8

$\cdot_{\text{free}}$: Space defined as being free of obstacles, 99

f : Vector valued function, also represented as a vector field in a dynamical system, 63

g : (n_u)-dimensional distribution of input directions, 9, 20, 58

G : Acceleration due to gravity/Gravitational Constant, 14, 53

γ : Initial hybrid state, 58

γ^c : Projection of γ onto $\mathbb{H}_c$, 69

Γ^k_{ij} : Christoffel symbols, 9

γ^c_{lbound} : Lower bound on coordinates of the grid cell, 77

γ^c_{rel} : Location relative to center of grid cell, 77

$\gamma^d = \ell, \gamma^c$: Discrete and continuous components of hybrid state, 58

$\mathbb{G}$: Gravitational constant, 21

$g(x),\ g_k(x)$: Generalized inverse function series, 43

G_T : Acceleration due to gravity torque, 14

H : Symmetry group, 10, 58, 59

$H_A(\cdot)$: State transition matrix convolution integral, 28

$H(\cdot)$: Hamiltonian function, 34

$\mathcal{H}$: Compact subset of $\mathbb{H}_c$, 75

$\mathbb{H}$: Discrete and continuous subspaces of maneuver space, 58

$\mathbb{I}$: Inertia matrix, 13

int U : The interior of U, 66

$\mathcal{I}_0$: Time state interval, 58

$\mathcal{I}_f$: Closed and bounded interval in $\mathcal{I}_0$, 75

I : Generic interval on $\mathbb{R}$, 28

$\mathcal{I}_\tau$: Coast time interval for maneuver automaton, 59

J : PVTOL rotational inertia, 45, 72

J^* : Optimal cost-to-go, 75

J_0 : Upper bound on optimal cost-to-go, 76

J_i : Component of the full cost J, 74

J^*_k : Stepwise component of optimal cost-to-go, 76

$\tilde{J}$: Approximate cost, 77

$J_T,\ J_M$: Cost functions of reference trajectory coast and maneuver, respectively, 72

κ : Distance scaling factor, 11

ker : Kernel, 20

k_i : PVTOL damping constants, 45

K_u : Control transformation matrix, 15

k_u : PVTOL control factor, 45

K_{vv}, $K_{\omega v}$, $K_{\omega\omega}$, K_O :
Constant damping matrices, 18

ℓ_{next_k} : reference trajectory index of $k+1$th time step, 59

$L\left(\xi(t), u(t), t\right)$: Optimal Control Lagrangian, 72

$L(\dot{q}, q)$: Dynamical systems Lagrangian, 9

λ : Costates of optimal control problem, 35, 73

$\Lambda_1,\ \Lambda_2$: Convergence radii, 43

$[X, Y]$: Lie bracket of vector fields X and Y, 65

$\text{LinReach}_A()$: Linear reachability subspace, 30

$L_X \cdot \varphi_f(q)$: *Directional* or *Lie derivative* of φ_f along X at q, 64

M : Differential manifold, 10, 45

m : Input space dimension, 40

$m_1,\ m_2$: Dimensions of $U_1,\ U_2$, 14

$\mathbb{M}$: Kinetic energy metric, 8

μ : Automaton word, i.e., sequence of control inputs $\{\nu_{n_T}, \nu_{n_T-1}, \ldots \nu_1\}$, 60

$\bar{\mu}$: Fixed maneuver sequence, 68

$\bar{\mu}^N$: Fixed control sequence μ iterated N times, 70

n : State space dimension, 40

N : Iterations of fixed control sequence μ, 70

$n_1,\ n_2$: Dimensions of $\Xi_1,\ \Xi_2$, 14

$\cdot_{nc}$: Uncontrollable subspace, 40

$\mathbb{N}_\delta(\gamma) \subseteq \mathbb{H}_d$: Set of discrete maneuver automaton inputs, 59

$N_{\mathbb{H}}$: Dimension of $\mathbb{H}$, 70

N_i : Nodes per dimension in grid, 76

n_T : Total number of discrete time steps in μ, 60

$Node_A,\ Node'_A$: Nodes in data tree, 94

$\|\cdot\|$: Generic norm, 63

$\|\cdot\|_{\mathcal{L}_\infty}$: Function infinity norm, 28

$\|\cdot\|_{\mathcal{L}_1}$: Function one norm, 28

$\|\cdot\|_\infty$: Infinity norm, 49

$\|\cdot\|_2$: Vector 2-norm, 34

$\|\cdot\|_\infty$: Vector/Matrix infinity norm, 28

n_p : Dimension of p, 36

$n_T^{\max}$: Maximum number of discrete-time steps allowed to complete the motion, 77

N_{tot} : Total nodes in grid, 76

ν_k : maneuver automaton control variable for kth time step, 58, 59

ν^* : Optimal single step control, 75

N_T : Number of Reference Trajectories, 58

ω : Angular velocity, 13

p : Base function parameter, 34

P : Controllability decomposing transformation matrix, 40

$\phi(\xi_0, t, u) = \phi_u(\xi_0, t)$: State flow, 9

ϕ_u^H : Projection of ϕ_u onto H, 59

$\Phi_k(t)$: kth order series tensor, 33

$\Phi_t^f(x_0, u_{\text{ref}}(t))$: State flow due to $u_{\text{ref}}(t)$, 31

φ_f : Function mapping Q to a Banach space, 64

ϕ_X^t : Flow along vector fields X for (small) time t, 65

$\mathcal{P}_{\bar{\mu}}$: Mapping due to perturbations on a fixed maneuver sequence, 69

$\psi(x, h) = \psi_h(x)$: Symmetry action, 10

$\psi^i(t)$: Bounded piecewise continuous base functions, 33

$\Psi(t, \tau)$: State transition matrix, 30

$.^p$: Pseudoinverse of a matrix, 42

Q : Configuration manifold, 8

q : Coordinates on configuration manifold Q, 8

$\mathbb{N}_\delta(\ell_k, h_k)$: Set of reference trajectories accessible from (ℓ_k, h_k), 59

R : Rotation matrix, 13

$\mathcal{R}_k$: Accessibility subspaces, 31

R_δ : Symmetry group variable on $SO(3)$, 18

$\mathcal{R}_\mathcal{X}(x)$: Reachable set in $\mathcal{X}$ in x, 64

$\mathcal{R}^{FULL}(S)$: Reachable set of S under the full set of feasible motions of the system., 95

$\mathcal{R}_k$: Reachable set after k steps of maneuver automaton, 75

$\mathcal{R}_H^{\bar{\mu}}$: Reachable set on $\mathbb{H}_c$ for this system under perturbations on the maneuver sequence $\bar{\mu}$, 69

$\mathcal{R}^\mu(S)$: μ-reachable set of S, 95

$\mathcal{R}_Q(q_0)$: Reachable set in Q from q_0, 67

$\mathbb{R}^k$: k-dimensional space of real numbers, 54

$\mathbb{R}^+$: All real numbers greater than zero, 11

$\text{Remainder}_K(f)(\cdot)$: Arbitrary smooth function $f(\cdot)$ Taylor remainder of order K, 29

ρ : Metric for RRT, 99

ρ_η : Symmetry group variable on $\mathbb{R}^+$, 18

ρ_κ : Symmetry group variable on $\mathbb{R}^+$, 18

$\text{round}_{\text{up}}(\cdot)$: Round up a scalar argument, 80

$\cdot^r$: Relative variables, 16

S^k : k-dimensional circle space, 54

S : A generic system space, subset of $\mathbb{R}^n$, 61

$(s, c, x, z, \omega, v_x, v_z)$: PVTOL State Variables, 45

$SE(k)$: Special Euclidean group of order k, 11

Σ : Mechanical Control System, 8

S_j : Interpolation function, 77

$SO(k)$: Special orthogonal group of order k, 13

$(\phi(u,\xi_0,t))$, $(\phi_u(\xi_0,t))$:
State Flow of Differential Equations, 12, 57

sup : Supremum, 28

T : Local planner final time, 33

t^* : Scaled time value, 11

τ_k : Coasting time on reference trajectory in kth time step, defined in the infinite closed and bounded interval $\mathcal{I}_\tau \subset \mathbb{R}$, 59

T_ϵ : Lower bound on maneuver time, 80

TF, TF_{ij} : Linear system transfer function and submatrices, 14

ϑ : Flat output, 45

T_ℓ : Maneuver time corresponding to reference trajectory ℓ, 59

$t_{\max}$: Upper time bound, 80

TQ : Tangent bundle of Q, 8

T^*Q : Cotangent bundle of Q, 8

V^T, V' : Transpose of vector or matrix V, 73

T_u : Decoupling transformation matrix, 15

$U \subset \mathbb{R}^{n_u}$: Compact input set, 9, 59

u, u_{ref}, $\bar{u}$: Continuous-Time Controls, general, reference trajectory, and relative, respectively, 13, 58

U_1, U_2 : Decoupled input subspaces of U, 14

$V : Q \to \mathbb{R}$: Potential energy function, 8, 13, 86

φ_μ : Action of the maneuver automaton due to the control sequence μ, 60

$\varphi(\gamma,\nu) = \varphi_\nu(\gamma)$: Action of the maneuver automaton, 60

$\varphi^H(\gamma_k,\nu_k)$: Projection of maneuver automaton action onto H, 60

W_T : Controllability Grammian, 39

x : Position, 13

Ξ : State Space of System, 9

ξ_0, ξ_f : Initial, final states, 94

Ξ_1, Ξ_2 : Decoupled state subspaces of Ξ, 14

ξ_r : Random state, 94

$xi_1(t), \xi_2(t)$: Curves mapping $[t_0, t_f] \mapsto TQ$, 12

$x(t),\ x_k(t)$: Series expansion terms, 30

$\mathcal{X}$: Subspace of $\mathbb{R}^n$, 63

x_{target} : Desired target state, 32

$X(q),\ Y_j(q)$: Time invariant vector fields, 64

y : Generalized series parameter, 36

$\tilde{\cdot}$: Controllability-decomposed system/Scaled variables, 40

Chapter 1

Introduction

Motion planning can be described as the act of planning a trajectory and its corresponding controls to guide a system from one state to another without violating given constraints on the system. Motion planning is one of the oldest problems known to mankind, dating back to the navigation of early hunters and explorers. As we have developed new and more complicated tools and vehicles, these challenges have become far more complicated. Today, most of the machines we rely upon have at least some degree of autonomy, as it has become inefficient if not impossible to control them manually. Due to recent advances in technology, vehicles that are entirely unmanned are gaining prevalence, and the development of such vehicles and robots has become a priority in many industries. Once again, motion planning is a key problem to be addressed, as a truly autonomous mechanical system must be able to plan new trajectories and controls quickly in response to changing inputs.

1.1 Motivating Applications

These vehicular and robotic systems under development provide a plethora of new opportunities and challenges for dynamics and control engineers. Listed below are just some of the growing areas of work.

1.1.1 Autonomous vehicles

Autonomous vehicles are gradually becoming more common, with numerous new programs in development. With the advances in microelectronics and the success of vehicles such as the Predator, the government has focused on unmanned aerial vehicles (UAVs) as a key part of its future plans and has actively encouraged control development for such platforms. In space, a plethora of current activities in interplanetary exploration [116, 6] have accentuated the need for effective online motion planning, notably in the NEAR, Mars rover, and Mars sample return programs. These, together with Earth orbiting missions such as those involving the international space station (ISS), highlight the need for global motion planning algorithms to handle issues from orbital transfers [113, 80, 35, 67] to obstacle avoidance [107] to rendezvous and docking [60, 109, 45]. Here, a global motion planning algorithm is one in which motion planning is not restricted to a local neighborhood, but possible between points spanning the state space. Current trajectory optimization techniques have difficulty addressing obstacles, while algorithms providing maneuvering relative to another object are often local in scope, not allowing for large scale trajectories.

1.1.2 Multiple Vehicle Coordination

The coordination of multiple agents jointly performing a specific task is a classic problem in robotics, but current algorithms fall short of addressing differential constraints, obstacles, and optimality. Air traffic management systems and collision avoidance [57, 61, 124] are important examples of this, requiring the capability for fast replanning of feasible and efficient maneuvers. This is a capability that can be provided by a global planner. Other relevant problems include clusters of spacecraft, aircraft, watercraft, and/or land-based vehicles performing activities such as inferometry [76], surveillance, communication services, and exploration.

1.1.3 Sensor-Based Planning

Along with these other topics, sensor-based planning is not to be neglected. As robotic craft are sent into difficult environments, they are being asked to navigate based upon

their sensory inputs. The personal satellite assistant (PSA) [132] and Aercam [34] have been designed to navigate and inspect the inside and outside of the ISS, respectively. Other applications in development requiring rapid planning to incorporate sensor data include submersibles inspecting wrecks and caves, robotics in search and rescue operations, and microscopic robots in medical applications. While research in this area has focused heavily on BUG algorithms [86, 123, 49, 73, 98] and kinematic systems [10], a general obstacle avoiding planner for dynamical systems with the ability to replan quickly carries the potential to be a more powerful tool.

1.2 Background

To be able to address the aforementioned applications effectively, motion planning algorithms must meet the following challenges:

Completeness and Controllability This speaks directly to the reliability of the algorithm. Completeness can be roughly described as the property that, if a solution to the algorithm exists, a solution will always be found. Controllability is an issue in that one would like an algorithm to be able to take advantage of the controllability properties of a system, i.e., being able to provide local control when the system is locally controllable.

Optimality As limited resources (time, energy, fuel, etc.) are available to systems, one does not want to waste them unnecessarily. Thus, providing some measure of cost reduction, if not optimality, is desired.

Computational Complexity and Speed Although computational resources have grown exponentially in recent years, our demands on systems have also been growing. Thus, the need to decrease computational complexity of algorithms and increase the speed has not declined, especially as systems are being asked to develop controls online.

Dynamic Environment Last is the issue of operating in a dynamic environment, where constraints and obstacles may exist and change with time. These factors and others might be understood, but could also be relatively unanticipated and unknown.

Many different approaches have been developed to try to address the motion planning problem, each addressing a subset of these challenges. The following is an overview of the major tools currently in use.

Differential flatness, as presented in [40], uses analytic understanding of the system to reduce it to an algebraic problem. While a catalog of flat systems has been described by [96], there is no algorithmic procedure to cast systems into the nonlinear form required, limiting its generality. Furthermore, limited attention [8] has gone towards characterizing approximately flat systems.

Lie brackets based planners are another set of algorithms for open loop control design; see [16, 17]. The typical planner relies on oscillations in order to move, in a way similar to how one parks a car or how an animal changes its shape to locomote. These methods have been applied to chained form systems by [97], driftless systems in [122, 81], locomotion systems in [103, 133], and addressed by the author's earlier work in [29]. The classic limitation of Lie bracket methods is their local nature, as only small amplitude motions can be planned satisfactorily.

In *numerical optimal control*, [18], trajectories are obtained through a numerical optimization. Because the problem is infinite dimensional, various forms of transcription (discretization/parameterization) are used to cast the variational problem into a nonlinear program. Collocation [53] uses base functions to parameterize both the states and the controls, while differential inclusion [115] avoids using the discretized controls by explicitly solving for the controls in terms of the states and their derivatives. Shooting methods [106, 39] to solve the optimal two-point boundary value problem have also been used, but the repeated numerical integration required has made this less attractive than transcription-based methods. Although useful and powerful, the high dimension, complexity, and lack of convergence guarantees limit the speed and reliability of these algorithms.

Although non-optimal, other techniques based on *heuristics and randomization* have been developed that promise fast execution in complex environments. These algorithms focus more on obstacle avoidance than on the nonlinear dynamics of the system, thus falling in the realm of "path planning" rather than "trajectory design." Popular among these are solutions based upon roadmaps [65] and incremental searches [79]. In roadmap methods, path planning is accomplished in two steps: a collection of sample con-

figurations is selected, and then trajectories connecting all sample points are computed. The second step, however, is a local controllability problem in itself.

Recently, efforts have been made to combine aspects of numerical optimization and randomized incremental searches. Karatas and Bullo [64] developed a version of the Rapidly Exploring Random Tree [79] using collocation as a local planner. Frazzoli [41] also used dynamic programming in his local planner, but used motion primitives to discretize the systems in question. In his methodology, he was able to reduce the continuous system into a finite set of possible *trim trajectories* that could be connected by a finite set of precomputed maneuvers. Both methods have been shown to find fast and attractive solutions. The latter even provides a controllability criteria for its hybrid system and a guaranteed solution for the local planning problem when that criteria is satisfied. However, its complete reliance on a precomputed finite set of maneuvers do not allow for small-time local controllability (STLC).

While often overlooked in motion planning, power series are another important tool in nonlinear control. They have been used widely, notably in the nonlinear regulator problem. Al'brekht [5] used power series to solve the Hamilton-Jacobi-Bellman (HJB) partial differential equations to obtain an optimal stabilizing control. For the same problem, Halme and coworkers [50, 51, 102, 52] developed a polynomial power series and a local inverse to generalized power series. Krener and coworkers [70, 99] took Al'brekht's method and used it to solve for the nonlinear regulator corresponding to the Francis-Byrnes-Isidori equations and, using level set methods and power series about extremal trajectories, proposed a method to extend Al'brekht's solutions to the HJB equations well beyond the neighborhood of the origin. Of course, power series methods are invariably local, but that does not make them any less relevant or useful. As mentioned beforehand, the randomized method requires a local motion planner. Likewise, the maneuvers of Frazzoli [41] also require computation.

Lastly, there is the field of *hybrid systems*, which, although not directly related to solving the motion planning problem, has been shown to provide an efficient modeling framework within which motion planning can take place. These frameworks are often referred to as hybrid automatons, in which the system is modeled via predefined classes of motion and discrete transitions between them [59]. This approach has been used by several researchers to reduce the computational complexity of the control problem.

Manikonda and coauthors [87] defined a motion description language, MDLe, that provides a general way to cast robotic motion planning problems into a suitable hybrid architecture. Similar hybrid system approaches have been used on the air traffic control problem [23] and autonomous helicopters [41].

1.3 Approach and Contribution

This research builds on the aforementioned areas of research to develop an efficient global motion planning algorithm for linearly controllable dynamical systems. This algorithm is structured in a hierarchical methodology combining, at the lowest level, online local planning about a trajectory using a novel power series-based approach. It then uses a hybrid system representation of the full continuous dynamics to simplify computation of controls to provide global motion planning. The local complete constructive trajectory generation and optimization algorithms are utilized to provide online maneuver planning for a trajectory primitive-based global planning algorithm reminiscent of Frazzoli's maneuver automaton [41]. This provides not only a dynamic set of motions available to the vehicle, but also allows for open loop control to a goal position regardless of the discretization mesh. In addition, this research proposes to introduce new trajectory primitives developed around symmetries on the state space and time. This provides a means to decrease the number of trajectory primitives needed while also representing a greater range of motions. Among other byproducts, this will extend the capability of the primitive-based algorithm beyond vehicles with dynamics invariant in translation and yaw. Because these components provide reliable solutions in an obstacle-free space, another layer, that of a randomized planning algorithm provides for avoidance of obstacles and constraints in the state space, both static and time-varying. In addition to the qualities already mentioned, this approach also provides cost reduction and convergence guarantees.

More specifically, the following contributions are outlined in the following chapters:

(i) Creation of a flexible local planner with convergence guarantees.

(ii) Construction of a global planner able to take advantage of local controllability properties in addition to time-varying final conditions.

(iii) Expansion of the Maneuver Automaton to include new symmetries and convergence proofs.

(iv) Application of a primitive-based planner with randomization to an orbital problem.

1.4 Overview

This dissertation is organized to follow the hierarchical structure defined above. Chapter 2 discusses the mechanical system dynamics on which much of the rest of this research is based. Chapter 3 develops the power series based local planning algorithms. The expanded maneuver automaton is introduced in Chapter 4 along with a dynamic programming solution method. The randomized planner providing time-varying obstacle avoidance in then addressed in Chapter 5. Chapter 6 summarizes the results of this research and comments on future research directions. The appendix provides supporting material for Chapter 3.

Chapter 2

Mechanical System Dynamics

2.1 Mechanical Control Systems

The mathematics describing the dynamics of a mechanical control system can be derived in several ways, of which the Lagrangian and Hamiltonian formulations are well-known examples. For our development, we have chosen the Lagrangian method, of which [47, 48, 2, 89] are but a few references on the subject.

Definition 2.1.1 (Mechanical Control System) *A mechanical control system Σ is defined by the following objects:*

(i) An n-dimensional C^∞ manifold Q, the configuration manifold, *with local coordinates $q = (q_1, q_2, \ldots, q_n)$. This manifold describes the space of all possible inertial positions of each particle of the mechanical system.*

(ii) A Riemannian metric $\mathbb{M} : TQ \times TQ \to \mathbb{R}$ on Q, describing kinetic energy $\left(\frac{1}{2}\mathbb{M}(q)(\dot{q}, \dot{q})\right)$, where TQ is the tangent bundle to Q.

(iii) A function $V : Q \to \mathbb{R}$, describing the potential energy.

(iv) A map $F_{nc} : TQ \to T^*Q$, describing the action of non-conservative forces on the mechanical system, where T^*Q is the cotangent bundle to Q, the dual of TQ.

(v) An (n_u)-dimensional distribution $g = span\{g^1(q,t), g^2(q,t), \ldots, g^{(n_u)}(q,t)\}$, defining the input directions.

(vi) A compact input set $U \subset \mathbb{R}^{n_u}$. The control input are partitioned into a set of control inputs $u \in U$.

Representing $\mathbb{M}(q)$ as a positive definite matrix in $\mathbb{R}^{n \times n}$, the Lagrangian can thus be written as

$$L(\dot{q}, q) = \frac{1}{2}\dot{q}^T \mathbb{M}(q)\dot{q} - V(q). \tag{2.1}$$

Lagrange's equations of motion are then described by

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = F_{nc}(q, \dot{q}) + \sum_{i=1}^{m} g(q)^i u_i, \tag{2.2}$$

which, in coordinates, simplify to

$$\ddot{q}_i + \Gamma^i_{jk}\dot{q}_j\dot{q}_k = \mathbb{M}^{-1}_{ij}\left(-\frac{\partial V}{\partial q_j} + F_{nc_j}(q, \dot{q}) + \sum_{k=1}^{m} g^k_j u_k\right), \tag{2.3}$$

where $\Gamma^i_{jk}\dot{q}_j\dot{q}_k = \sum_{j=1..n,\ k=1..n} \Gamma^i_{jk}\dot{q}_j\dot{q}_k = \Gamma^i\dot{q}\dot{q}$ and the *Christoffel symbols* Γ^k_{ij} are defined as

$$\Gamma^k_{ij} = \frac{1}{2}\sum_m (\mathbb{M}^{-1})_{mk}\left(\frac{\partial \mathbb{M}_{mj}}{\partial q_i} + \frac{\partial \mathbb{M}_{mi}}{\partial q_j} - \frac{\partial \mathbb{M}_{ij}}{\partial q_m}\right). \tag{2.4}$$

Defining v_q in terms of q, one can recast this second order differential equation as a first order system $\dot{\xi} = [\dot{q}, \dot{v_q}]^T = f(\xi, t)$. For example, when $\dot{q} = v_q$,

$$\dot{\xi} = \begin{bmatrix} \dot{q} \\ \dot{v_q} \end{bmatrix} = \begin{bmatrix} v_q \\ -\Gamma v_q v_q + \mathbb{M}^{-1}\left(-\frac{\partial V}{\partial q} + F_{nc}(q, v_q) + \sum_{k=1}^{m} g^k u_k\right) \end{bmatrix} = f(\xi, t) \tag{2.5}$$

The solution, or *state flow* of the equation of motion can be written as $\xi = \phi(\xi_0, t, u) = \phi_u(\xi_0, t)$. Note that ξ is defined on the state space $\Xi = TQ$.

2.2 Symmetry

One key property used in the analysis and control of mechanical systems is that of *symmetry*. To understand this, the concept of an action must first be introduced [129].

Definition 2.2.1 (Action) *Let M be a differential manifold and H be a Lie Group. A C^∞ map $\psi : M \times H \to M$ taking a pair $(x, h) \in M \times H$ to $\psi(x, h) = \psi_h(x)$ and satisfying the following conditions*

(i) $\psi_e(x) = x$, $\forall x \in M$ *and e is the identity element of H*

(ii) $\psi_{h_1}(\psi_{h_2}(x)) = \psi_{h_1 h_2}(x)$

is called an action *of H on M.*

In the context of mechanical systems, the action is most commonly on the configuration manifold Q. A group action on Q induces corresponding maps on scalar functions over Q, tangent vectors, and one-forms (examples include include the system's potential energy, velocity, and external forces respectively) [101].

Given the definition of an action, the notion of symmetry can be defined.

Definition 2.2.2 (Symmetry) *Let x be a point in M. The group action $\psi_h : M \to M$ is then a* symmetry *of the differential equations of the system $\dot{x} = f(x, t)$ if, given the state flow $\phi(x_0, t)$, the action and the state flow commute, i.e.,*

$$\psi_h(\phi(x_0, t)) = \phi\left(\psi_h(x_0), t\right), \tag{2.6}$$

Definition 2.2.3 (Symmetry Group) *The group H consisting of all h for which the action ψ_h is a symmetry is called a symmetry group.*

Frazzoli [41] used the symmetry group $SE(2) \times \mathbb{R}$ in his work, where the manifold M is defined as configuration manifold Q (or its tangent bundle TQ). However, more general symmetries exist when one is not limited to the configuration manifold. Consider the mechanical control system in equation (2.5). Let us add another variable $\eta \in \mathbb{R}^+$ to account for a scaling in the progression of time ($\mathbb{R}^+$ denotes all real numbers greater than

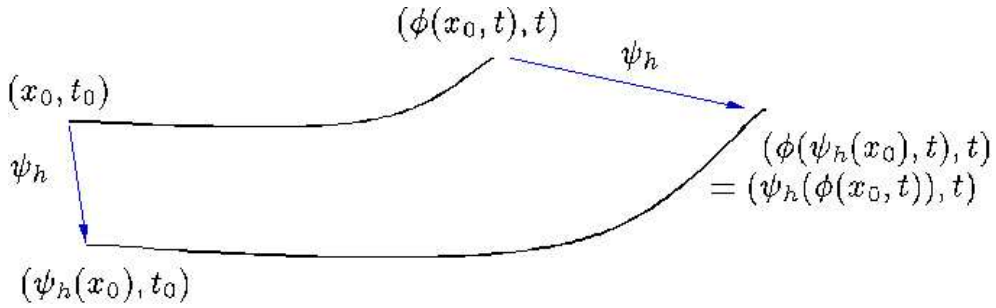

Figure 2.1: Symmetry

zero). The variable t^* can then be defined such that $t^* = \eta t$ and the function $f^*(\xi, t^*)$ is defined such that $\eta f^*(\xi, t^*) = f(\xi, t)$.

When the scaling is taken into account with the derivative, the system of equation (2.5) can be rewritten as

$$\begin{aligned} \frac{d\xi}{dt^*} &= f^*(\xi, \eta t) & \xi(t_0^*) &= \xi_0 \\ \frac{d\eta}{dt^*} &= 0 & \eta(t_0^*) &= \eta_0, \end{aligned} \tag{2.7}$$

In this case, the manifold M is defined as $TQ \times \mathbb{R}^+$ instead of Q (or TQ). Thus, the symmetry group H of this system may allow for action on the time scaling factor as well as on the configuration manifold. It should be noted that distance scaling symmetries [1] can be incorporated in a similar fashion by adding another variable $\kappa \in \mathbb{R}^+$.

2.3 Trajectory Primitives

Let us define a trajectory as the state flow for a given control history such that $\xi : [t_0, t_f] \mapsto TQ$. The symmetry property provides an equivalence relation between trajectories.

Definition 2.3.1 (Trajectory Equivalence) *Consider two trajectories, $\xi_1, \xi_2 : [t_0, t_f] \mapsto TQ$. These trajectories are said to be equivalent on the interval $[t_0, t_f]$ if there exists*

[1] This same statement could be said for scaling with respect to mass, although such a symmetry is not specifically used in this thesis.

an element $h \in H$ such that $\xi_1(t) = \psi_h(\xi_2(t))$.

Definition 2.3.2 (Trajectory Primitive) *We define a trajectory primitive as an equivalence class of trajectories, under the trajectory equivalence definition.*

Each trajectory in this class can then be uniquely identified by a point on the symmetry group. One class of a trajectory primitive occurring commonly in the setting of mechanical systems is known as relative equilibria and has been the focus of recent research [41, 20, 19].

Definition 2.3.3 (Relative Equilibrium) *Mechanical system motion evolving along a symmetry group where an initial condition ξ_0, coordinate frame, and a constant control input u can be set such that the velocity is constant.*

Frazzoli [41] noted that, for aerospace vehicles, these are commonly described as *trim trajectories.*

Relative equilibria have been popular because of their useful properties, particularly their ease of manipulation and the intuitive correlation to vehicle motion, but other primitive types also exist which provide useful theoretical properties. In this work, we are interested in a broader class of primitives defined as follows:

Definition 2.3.4 (Reference Trajectory) *A trajectory satisfying the equations of motion of the system Σ for which*

(i) The state flow ($\phi_{u_{ref}}(\xi_0, t)$) is known and is unique for a given initial state $\xi_0 \in \Xi$.

(ii) The control $u_{ref}(t)$ is a continuous function of time only

(iii) The dynamical system Σ can be approximated as a perturbation about the trajectory primitive with the form

$$\begin{aligned} \dot{x} &= A(t)x + f^{[2]}(x, x) + Bu \\ x(0) &= 0, \end{aligned} \tag{2.8}$$

where $x(t)$ and $A(t)$ are defined as continuous functions on the interval I, and the $\mathcal{L}_1$-norm of the corresponding linear state transition matrix $\|\Psi(t,0)\|_{\mathcal{L}_1}$ is bounded [2].

This property will play an important role when motion off of this class is considered. We will refer to these as *reference trajectories*. Each can be uniquely defined by its control history and symmetry. Many vehicle systems can be cast into this form, either naturally through transformation or through series expansion. The reference trajectories of such systems include relative equilibria but are not necessarily restricted to them. In addition, Kang and Krener [63] showed that any nonlinear system of the form $\dot{\xi} = f(\xi) + g(\xi)u$ can be represented in a time-invariant version of the aforementioned quadratic form (plus higher order terms) by a change of coordinates and state feedback.

2.4 Vehicle Models

2.4.1 Generalized Vehicle Model

As this thesis is focused primarily around the dynamics and control of vehicles, we will restrict our mechanical control system to such a case. Here, $(q_1, \ldots q_n) = (x, R)$ correspond to the inertial translational and rotational coordinates of the vehicle and $(v_{q_1}, \ldots v_{q_n}) = (v, \omega)$ correspond to its body-fixed velocities. For full dynamics in three dimensions, $x, v, \omega \in \mathbb{R}^3$ and $R \in SO(3)$. In two dimensions, $x, v \in \mathbb{R}^2$, $\omega \in \mathbb{R}$ and $R \in SO(2)$. The Riemannian metric $\mathbb{M} : TQ \times TQ \to \mathbb{R}$ describing the kinetic energy is then defined as the inertia matrix $\mathbb{I}$. If the vehicle is rigid and the fuel usage is negligible in comparison to the overall mass distribution, the inertia matrix can be assumed to be constant. The potential energy of interest is the gravity potential, which can be modelled as a constant (for neutrally buoyant vehicles such as water vehicles, airships, and hovercraft), proportional to height (for aerial vehicles such as helicopters and airplanes), and inverse proportional to radius (for spacecraft). Damping terms such

[2] Here, $\|\Psi(t,0)\|_{\mathcal{L}_1} = \max_{i=1,\ldots,n} \sum_{j=1}^{n} \int_I |(\Psi(t,0)_{ij}| dt$. This statement implies that the system $\dot{x} = A(t)x$ is stable in the sense of Lyapunov (for the time-invariant system, it is sufficient the matrix A is Hurwitz) or that the interval I over which $A(t)$ is defined is finite and the escape time of $\dot{x} = A(t)x$ (if it exists) is not in the interval.

as lift and drag provide nonconservative forces and can be modelled as linear and/or quadratic in velocity. For simplicity, we will consider only vehicles for which the control input directions are fixed with respect to the body frame. The dynamical equations of such vehicles can then be written as:

$$\dot{\xi} = \begin{bmatrix} \dot{x} \\ \dot{R} \\ \dot{v} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} Rv \\ R\hat{\omega} \\ -\hat{\omega}v + G(R,x) + D_v(v) + B_v u \\ -\mathbb{I}^{-1}\hat{\omega}\mathbb{I}\omega + G_T(R,x) + D_\omega(v,\omega) + B_\omega u \end{bmatrix} = f(\xi, t), \tag{2.9}$$

where G and G_T are acceleration due to gravity and the torque acceleration due to gravity, D_v and D_ω are the damping terms, B_v and B_ω are constant matrices, and $\hat{}$ is the skew-symmetric matrix operator.

Driftlessly Decoupled System

Before proceeding further, let us introduce the concept of a *decoupled* system with regards to inputs and states. Let Ξ be the state space of a dynamical system and Ξ_1, Ξ_2 be defined such that $\Xi_1 \times \Xi_2 = \Xi$. Likewise, let U be the input space and U_1, U_2 be defined such that $U_1 \times U_2 = U$. It can then be said that the system is *input-state decoupled* between $\{\Xi_1, U_1\}$ and $\{\Xi_2, U_2\}$ if the state flow along Ξ_1 is invariant for all inputs in U_2 and the state flow along Ξ_2 is invariant for all inputs in U_1. Otherwise, $\{\Xi_1, U_1\}$ and $\{\Xi_2, U_2\}$ are *input-state coupled*. Note that, for system (2.5), $\Xi = TQ$.

For linear time-invariant systems ($\dot{\xi} = A\xi + Bu$), this property can be related to the transfer function. Let $\xi = (\xi_1, \xi_2) \in \Xi_1 \times \Xi_2$ and $u = (u_1, u_2) \in U_1 \times U_2$, where $\dim(U_i) = m_i$, $\dim(\Xi_i) = n_i$, $m_1 + m_2 = dim(U) = m$ and $n_1 + n_2 = dim(\Xi) = n_{full}$. The transfer function TF is defined as follows:

$$TF = [sI - A]^{-1}B \begin{bmatrix} TF_{11} & TF_{12} \\ TF_{21} & TF_{22} \end{bmatrix} \rightarrow \begin{matrix} x_1 = TF_{11}u_1 + TF_{12}u_2 \\ x_2 = TF_{21}u_1 + TF_{22}u_2 \end{matrix}$$

Then the system is *input-state decoupled* between $\{\Xi_1, U_1\}$ and $\{\Xi_2, U_2\}$ if and only if TF_{12} and TF_{21} are zero matrices.

From this background, let us introduce the concept of a *driftlessly decoupled system.*

Let us use the same nomenclature of ξ, Ξ, u, and U as given previously. We will also define the system such that $B = [B_1^T \ B_2^T]^T$ where B_i is an $n_i \times m$ matrix and $rank(B) = m$.

Definition 2.4.1 (Driftlessly Decoupled) *Given the system* (2.9) *Let T_u be a non-singular orthogonal square matrix of dimension m by m, where m is the dimension of u. The system* (2.9) *is* driftlessly decoupled *between $\{\Xi_1, U_1\}$ and $\{\Xi_2, U_2\}$ by the control $u = T_u\tilde{u}$ if the matrix BT_u evaluates to the form:*

$$BT_u = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} T_u = \begin{bmatrix} T_{11} & 0_{n_1 \times m_2} \\ 0_{n_2 \times m_1} & T_{22} \end{bmatrix} \tag{2.10}$$

Note that this is analogous to the condition for input-state decoupling of the linear system if the drift term $(A\xi)$ is zero for all ξ. We can find the transformation T_u from the following lemma:

Lemma 2.4.2 *If B_1 and B_2 have mutually orthogonal bases, there exists a driftlessly decoupling transformation $T_u = [(B_1^{rb})^T \ (B_2^{rb})^T]$, where $B_i^r b$ is a matrix of the orthonormal basis of B_i with dimension m_i by m.*

Proof: The calculation is straightforward:

$$BT_u = \begin{bmatrix} B_1(B_1^r b)^T & B_1(B_2^r b)^T \\ B_2(B_1^r b)^T & B_2(B_2^r b)^T \end{bmatrix} = \begin{bmatrix} T_{11} & 0_{n_1 \times m_2} \\ 0_{n_2 \times m_1} & T_{22} \end{bmatrix}$$

■

For driftless decoupling of rotation and translation, $B_1 = B_v$ and $B_2 = B_\omega$. Physically, this decoupling of rotation and translation means that the inputs can be linearly combined into m torque-only and force-only inputs. Thus, a transformation that allows for driftless decoupling of rotation and translation also allows for incorporation of both time and distance scaling symmetries. Provided such a transformation can be calculated by Lemma 2.4.2, the substitution $u = K_u\tilde{u}$ can be defined where K_u is an invertible m by m matrix defined as follows:

$$K_u = T_u \begin{bmatrix} \eta^2\kappa I_{m_1 \times m_1} & 0_{m_1 \times m_2} \\ 0_{m_2 \times m_1} & \eta^2\kappa I_{m_2 \times m_2} \end{bmatrix} T_u^T \tag{2.11}$$

This substitution allows for the quantities η and κ to be factored out of the control vector, as

$$BK_u = \begin{bmatrix} B_1(B_1^r b)^T & 0_{n_1 \times m_2} \\ 0_{n_2 \times m_1} & B_2(B_2^r b)^T \end{bmatrix} \begin{bmatrix} \eta^2 \kappa I_{m_1 \times m_1} & 0_{m_1 \times m_2} \\ 0_{m_2 \times m_1} & \eta^2 \kappa I_{m_2 \times m_2} \end{bmatrix} \begin{bmatrix} (B_1^r b) \\ (B_2^r b) \end{bmatrix} = \begin{bmatrix} \eta^2 \kappa B_1 \\ \eta^2 B_2 \end{bmatrix}$$

A system driftlessly coupled between rotation and translation does not allow for κ to be factored out of the control vector and thus does not allow a distance scaling symmetry. For the sake of brevity, systems driftlessly coupled between rotation and translation will be referred to as *coupled* and systems driftlessly decoupled between rotation and translation will be referred to as *decoupled*.

Equations of Relative Motion

Let $\xi^r(t) = [x^r(t), R^r(t), v^r(t), \omega^r(t)]^T$ denote a known trajectory due to a given control. Dropping the (t) suffix for brevity, let us then define a perturbation to the system via the coordinates $\bar{\xi} = [\bar{x}, \bar{R}, \bar{v}, \bar{\omega}]^T$ such that

$$\xi^r = \left[x^r + R^r(\bar{R} + I_3)\bar{x}, R^r(\bar{R} + I_3), v^r + \bar{v}, \omega^r + \bar{\omega}\right]^T \tag{2.12}$$

$$u = K_u(u^r + \bar{u}), \tag{2.13}$$

yielding the equations of relative motion

$$\dot{\bar{\xi}} = \begin{bmatrix} \dot{\bar{x}} \\ \dot{\bar{R}} \\ \dot{\bar{v}} \\ \\ \dot{\bar{\omega}} \\ \\ \end{bmatrix} = \begin{bmatrix} -\bar{R}^T v^r + \bar{v} - \left(\widehat{\omega}^r + \widehat{\bar{\omega}}\right)\bar{x} \\ \bar{R}\left(\widehat{\omega}^r + \widehat{\bar{\omega}}\right) + \widehat{\omega}^r \bar{R} + \widehat{\bar{\omega}} \\ -\widehat{\bar{\omega}}\bar{v} - \left(\widehat{\omega}^r \bar{v} + \widehat{\bar{\omega}} v^r\right) + (G(R,x) - G(R^r, x^r)) \\ + (D_v(v) - D_v(v^r)) + \eta^2 \kappa B_v \bar{u} \\ -\mathbb{I}^{-1}\widehat{\bar{\omega}}\mathbb{I}\bar{\omega} - \mathbb{I}^{-1}\widehat{\bar{\omega}}\mathbb{I}(\omega^r + \bar{\omega}) + (G_T(R,x) - G_T(R^r, x^r)) \\ + (D_\omega(v,\omega) - D_\omega(v^r, \omega^r)) + \eta^2 \bar{u} \end{bmatrix}, \tag{2.14}$$

where $\dot{\eta} = 0$ and $\dot{\kappa} = 0$ represent the augmented states of the system. K_u, in the case of a coupled system, is defined as in (2.11), otherwise, it is assumed to be defined as

$K_u = \eta^2 I_m$. It should be noted that, in the three dimensional case, while $\bar{x}, \bar{v}, \bar{\omega} \in \mathbb{R}^3$ lie on the manifold of their general counterparts, $\bar{R} + I_n \in SO(3)$ where $R \in SO(3)$. As this allows for the boundary condition $\bar{\xi}(0) = 0$ if the equations of relative motion are of the quadratic form of (2.8), any trajectory satisfying the equations of motion could be defined as a reference trajectory. This quadratic property is inherent in many mechanical systems, and many others can be approximated by a quadratic system when motion about a trajectory is assumed to be sufficiently local. Several representative systems are addressed in the following sections.

2.4.2 Systems with a negligible gravity term

For a system to have a negligible gravity term does not mean that gravity does not act on the vehicle, rather it means that gravity is countered by another force or its effect is simply small enough to be ignored for the problem at hand. Examples of this include hovercraft, airships, neutrally buoyant underwater vehicles, self-propelled boats, and small spacecraft operating within a limited time frame, where the time frame and smallness of the spacecraft are dependent upon the mass and relative location of the dominating gravity source.

No damping, decoupled controls

In this case, equation (2.14) is reduced to

$$\begin{bmatrix} \dot{\bar{\tilde{x}}} \\ \dot{\bar{R}} \\ \dot{\bar{\tilde{v}}} \\ \dot{\bar{\tilde{\omega}}} \end{bmatrix} = \eta \begin{bmatrix} -\bar{R}^T \tilde{v}^r + \bar{\tilde{v}} - \left(\hat{\tilde{\omega}}^r + \hat{\bar{\tilde{\omega}}}\right) \bar{\tilde{x}} \\ \bar{R}\left(\hat{\tilde{\omega}}^r + \hat{\bar{\tilde{\omega}}}\right) + \hat{\tilde{\omega}}^r \bar{R} + \hat{\bar{\tilde{\omega}}} \\ -\hat{\bar{\tilde{\omega}}}\bar{\tilde{v}} - \left(\hat{\tilde{\omega}}^r \bar{\tilde{v}} + \hat{\bar{\tilde{\omega}}}\tilde{v}^r\right) + B_v \bar{u} \\ -\mathbb{I}^{-1}\hat{\bar{\tilde{\omega}}}\mathbb{I}\bar{\tilde{\omega}} - \mathbb{I}^{-1}\hat{\bar{\tilde{\omega}}}\mathbb{I}(\tilde{\omega}^r + \bar{\tilde{\omega}}) + B_\omega \bar{u} \end{bmatrix} \tag{2.15}$$

when $x = \kappa\tilde{x}$, and $v = \kappa\eta\tilde{v}$, and $\omega = \eta\tilde{\omega}$. Here, $\tilde{x}$, $\tilde{v}$, and $\tilde{\omega}$ represent nondimensionalized quantities. From the above derivation, it can thus be seen that the system in (2.14), under a zero drag and damping, with decoupled controls, admits the symmetry action:

$$\begin{aligned}\psi : &\left(\mathbb{R}^3 \times SO(3) \times \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}^+ \times \mathbb{R}^+\right) \times \left(\mathbb{R}^3 \times SO(3) \times \mathbb{R}^+ \times \mathbb{R}^+\right) \\ &\to \left(\mathbb{R}^3 \times SO(3) \times \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}^+ \times \mathbb{R}^+\right) \\ &(x, R, v, \omega, \eta, \kappa) \times (\delta x, R_\delta, \rho_\kappa, \rho_\eta) \\ &\mapsto (\rho_\kappa R_\delta(x + \delta x), R_\delta R, \rho_\kappa \rho_\eta v, \rho_\eta \omega, \rho_\eta \eta, \rho_\kappa \kappa)\,.\end{aligned}$$

When the controls are coupled, κ is unitary and no longer arbitrary, breaking the distance scaling symmetry and limiting the symmetry action to

$$\psi : (x, R, v, \omega, \eta, \kappa) \times (\delta x, R_\delta, \rho_\eta) \mapsto (R_\delta(x + \delta x), R_\delta R, \rho_\eta v, \rho_\eta \omega, \rho_\eta \eta, \kappa)\,. \quad (2.16)$$

Linear damping

In certain problems, it is appropriate to represent the damping forces as a linear function of the system velocities such that $D_v(v) = -K_{vv}v$ and $D_\omega(v, \omega) = -K_{\omega v}v - K_{\omega\omega}\omega$, where K_{vv}, $K_{\omega v}$, and $K_{\omega\omega}$ are constant matrices, resulting in the following damping contribution to the equations of relative motion:

$$\begin{aligned}(D_v(v) - D_v(v^r)) &= -K_{vv}\bar{v} \\ (D_\omega(v, \omega) - D_\omega(v^r, \omega^r)) &= -K_{\omega v}\bar{v} - K_{\omega\omega}\bar{\omega}\end{aligned}$$

When the controls are decoupled and $K_{\omega v} = 0$, this term can be directly inserted into (2.15), provided η is fixed. Thus, linear damping breaks the time scaling symmetry of the system, yielding the symmetry action:

$$\psi : (x, R, v, \omega, \eta, \kappa) \times (\delta x, R_\delta, \rho_\kappa) \mapsto (R_\delta \rho_\kappa(x + \delta x), R_\delta R, \rho_\kappa v, \omega, \eta, \rho_\kappa \kappa)$$

When the assumption on the controls or $K_{\omega v}$ is not true, κ must be unity to be substituted directly into (2.15), breaking the distance scaling symmetry of the system as well. Thus, the symmetry action is further reduced to

$$\psi : (x, R, v, \omega, \eta, \kappa) \times (\delta x, R_\delta) \mapsto (R_\delta(x + \delta x), R_\delta R, v, \omega, \eta, \kappa) \tag{2.17}$$

It should be noted that this symmetry action applies for all forms of damping dependent only on the system velocities.

Second-order damping

Second-order damping terms are also quite common, particularly in aerodynamic problems, often taking the form $D_v(v) = -K_{vv}\|v\|_2 v$ and $D_\omega(v, \omega) = -K_o\|K_{\omega v}v + K_{\omega\omega}\omega\|_2(K_{\omega v}v + K_{\omega\omega}\omega)$, where K_o, K_{vv}, $K_{\omega v}$, and $K_{\omega\omega}$ are constant matrices. The damping force in the equations of relative motion then appears as

$$\begin{aligned}
(D_v(v) - D_v(v^r)) &= -K_{vv}\left[(\|v^r + \bar{v}\|_2 - \|v^r\|_2)v^r\right] + \|v^r + \bar{v}\|_2(\bar{v})] \\
(D_\omega(v, \omega) - D_\omega(v^r, \omega^r)) &= -K_o(\|K_{\omega v}v^r + K_{\omega\omega}\omega^r + K_{\omega v}\bar{v} + K_{\omega\omega}\bar{\omega}\|_2 \\
&\quad - \|K_{\omega v}v^r + K_{\omega\omega}\omega^r\|_2)(K_{\omega v}v^r + K_{\omega\omega}\omega^r) \\
&\quad + \|K_{\omega v}v^r + K_{\omega\omega}\omega^r + K_{\omega v}\bar{v} + K_{\omega\omega}\bar{\omega}\|_2(K_{\omega v}\bar{v} + K_{\omega\omega}\bar{\omega})
\end{aligned}$$

When the $K_{\omega v} = 0$, this term can be directly inserted into (2.15), provided κ is fixed. Thus, quadratic damping breaks the distance scaling symmetry of the system, yielding the symmetry action of (2.16). Such a condition might occur when the center of pressure and center of mass of the vehicle are collocated.

When $K_{\omega v}$ is nonzero, the time scaling symmetry is also broken, resulting in the symmetry action of (2.17).

Unlike the previous formulations, systems with a second-order drag such as that mentioned above do not directly satisfy the quadratic form of (2.8), as the drag term is nonsmooth at zero angular and linear velocity. Simplifying assumptions

such as a linear drag or, for reference velocities well away from zero, a quadratic drag approximation would have to be used.

All of these systems can thus be cast or approximated by the quadratic form of (2.8). The time varying nature of the linear term is due to the reference vectors v^r and ω^r. When these are constant, the linear term is constant. The reference trajectories resulting in the time invariant system thus correspond to helices and are relative equilibria of the system.

2.4.3 Systems with a constant gravity term

Almost all non-buoyant aircraft, assuming non-drastic altitude changes, fit this description. It also includes non-neutrally buoyant underwater vehicles, also assuming non-drastic depth changes. The acceleration of gravity can be written as $G(R, x) = R^T g$, where g is the inertial constant gravity acceleration vector. The torque due to gravity is zero, because the center of gravity and the center of mass are collocated under constant gravity. The gravity component of the equations of relative motion then evaluates to

$$G(R, x) - G(R^r, x^r) = \bar{R}^T {R^r}^T g$$

This term fits the framework of (2.15) only when η and κ are defined such that $\eta^2 \kappa = \|g\|_2$, thus coupling the time scaling and distance scaling symmetries. In addition, the rotational symmetry $R_\delta \in SO(3)$ is limited such that $g \in \ker R_\delta$. Thus, without damping, the symmetry group H is $\mathbb{R}^3 \times \mathbb{R}^+ \times (SO(3) \cap \ker g)$ and the symmetry action is

$$\psi : (x, R, v, \omega, \eta, \kappa) \times (\delta x, R_\delta) \mapsto \left(R_\delta \rho_\kappa (x + \delta x), R_\delta R, \rho_\kappa^{\frac{1}{2}} v, \omega, \rho_\kappa^{-\frac{1}{2}} \eta, \rho_\kappa \kappa \right),$$

With damping, the scaling symmetries are broken, so the symmetry group H reduces to $\mathbb{R}^3 \times (SO(3) \cap \ker g)$ and the symmetry action is

$$\psi : (x, R, v, \omega, \eta, \kappa) \times (\delta x, R_\delta) \mapsto (R_\delta (x + \delta x), R_\delta R, v, \omega, \eta, \kappa),$$

which is equivalent to (2.17), except that $R_\delta \in (SO(3) \cap \ker g)$. Like the case of negligible gravity, this class of systems is of the quadratic form of (2.8). The time-invariant case (when v^r and ω^r are constant) once again corresponds to motion about helices, but the rotation of the helix is constrained to be about the gravity vector, i.e., ω^r is parallel to g.

2.4.4 Systems with an inverse square gravity term

The inverse square relationship corresponds to Newton's law of gravity for point masses, where one body is substantially more massive than the other. Thus, this category includes every vehicle for which gravity can be regarded as emanating from a single point source at the origin. It encompasses all spacecraft for which planetary (or solar) mass distribution and third body effects are negligible [3]. The acceleration due to gravity can then be written as $G(x) = -\mu / \left(\|x\|_2^3\right) R^T x$, where $\mu = \mathbb{G}M$, the gravitational constant $\mathbb{G}$, multiplied by the mass of the attracting body M. When cast into the relative frame,

$$(G(R,x) - G(R^r,x^r)) = -\mu \left[\left(\bar{R} + I_3\right)^T R^{rT} \frac{x^r + R^r(\bar{R} + I_3)\bar{x}}{\|x^r + R^r(\bar{R} + I_3)\bar{x}\|_2^3} - R^{rT} \frac{x^r}{\|x^r\|_2^3} \right]$$

Once again, let us use the substitution $x = \kappa\tilde{x}$.

$$(G(R,x) - G(R^r,x^r)) = -\frac{\mu}{\kappa^3}\kappa \left[\frac{\left(\bar{R} + I_3\right)^T R^{rT}\tilde{x}^r + \bar{\tilde{x}}}{\|\left(\bar{R} + I_3\right)^T R^{rT}\tilde{x}^r + \bar{\tilde{x}}\|_2^3} - R^{rT} \frac{\tilde{x}^r}{\|\tilde{x}^r\|_2^3} \right] \tag{2.18}$$

In order to fit this term to (2.15), the time and distance scaling factors must be constrained such that $\eta^2 = \mu/(\kappa^3)$. In addition, the translational symmetry is broken, as the position relative to the point source is not invariant with arbitrary translation.

Because the gravity is not constant, the gravitational torque felt by the spacecraft is dependent on its mass distribution. The ensuing acceleration due to the gravitational torque is expressed by the function

[3] For more information on rigid-body dynamics in an orbital environment, see [47, 108].

$$G_T(R,x) = \frac{3\mu}{\|x\|_2^5}\mathbb{I}^{-1}\widehat{[R^T x]}\mathbb{I}\left[R^T x\right].$$

Assuming that $x = \kappa\tilde{x}$, the gravity angular acceleration component of the equations of relative motion reduces to

$$G_T(R,x) - G_T(R^r,x^r) = \frac{3\mu}{\kappa^3}\mathbb{I}^{-1}\left[\frac{\left[\widehat{(\bar{R}+I_3)^T R^{rT}}\tilde{x}^r + \bar{\tilde{x}}\right]\mathbb{I}\left[(\bar{R}+I_3)^T R^{rT}\tilde{x}^r + \bar{\tilde{x}}\right]}{\|(\bar{R}+I_3)^T R^{rT}\tilde{x}^r + \bar{\tilde{x}}\|_2^5} - \frac{\left[\widehat{R^{rT}\tilde{x}^r}\right]\mathbb{I}\left[R^{rT}\tilde{x}^r\right]}{\|\tilde{x}^r\|_2^5}\right]$$

This term is thus also compatible with (2.15) when $\eta^2 = \mu/\kappa^3$ and preserves a scaling symmetry. Therefore, when the controls are decoupled, the symmetry group H is $SO(3)\times\mathbb{R}^+$ and the action is

$$\psi : (x,R,v,\omega,\eta,\kappa)\times(R_\delta,\rho_\kappa) \mapsto \left(R_\delta\rho_\kappa x, R_\delta R, \rho_\kappa^{-\frac{1}{2}}v, \rho_\kappa^{-\frac{3}{2}}\omega, \rho_\kappa^{-\frac{3}{2}}\eta, \rho_\kappa\kappa\right).$$

In all other cases, the symmetry group is reduced to $H = SO(3)$ and the action is

$$\psi : (x,R,v,\omega,\eta,\kappa)\times(R_\delta) \mapsto (R_\delta x, R_\delta R, v, \omega, \eta, \kappa).$$

Physically, this means that the orbital behavior of a spacecraft with a given radius and control history is equivalent to its behavior at any other point where the radius is the same and the orientation and velocity vectors do not change in relation to the radius vector. When the scaling symmetry is intact, it is also equivalent to motion at other radii and velocity magnitudes provided the quantity $v^T v\sqrt{x^T x}$ is invariant.

For an elliptical reference orbit, when the scaling symmetry exists, κ can be considered its semimajor axis and η then corresponds to its mean motion. Regardless, while position on the ellipse does not deviate with a symmetry action, the orientation of the ellipse can be arbitrarily changed, i.e., the unit normal vector can be reoriented and the periapse can be rotated about that vector. The scaling symmetry equates any two ellipses with the same eccentricity.

Unfortunately, although drag can be modelled effectively as a linear function at orbital altitudes due to rarefied gas flow, the equations of motion using the inverse-square model

of gravity do not follow the quadratic form of (2.8). Thus, a series expansion is necessary. Let $R^{r^T}\tilde{x}^r = \|\tilde{x}^r\|_2 \mathbf{e}^r$ and $\bar{\tilde{x}} = \|\tilde{x}^r\|_2 z$. Substituting into (2.18), we get

$$(G(R,x) - G(R^r,x^r)) = -\frac{\mu}{\kappa^3}\kappa\frac{1}{\|\tilde{x}^r\|_2^2}\left[\frac{(\bar{R}+I_3)^T\mathbf{e}^r + z}{\|(\bar{R}+I_3)^T\mathbf{e}^r + z\|_2^3} - \mathbf{e}^r\right] \tag{2.19}$$

It can be seen that $\|(\bar{R}+I_3)^T\mathbf{e}^r + z\|_2 = \sqrt{1 + 2\mathbf{e}^{r^T}(\bar{R}+I_3)z + z^Tz}$. Let b be defined as $2\mathbf{e}^{r^T}(\bar{R}+I_3)z + z^Tz$. The Taylor series expansion for $(1+b)^{3/2}$ is then

$$\begin{aligned}(1+b)^{3/2} &= 1 - \frac{3}{2}b + \frac{15}{8}b^2 - \frac{35}{16}b^3 + \frac{315}{128}b^4 + O(b^5) \\ &= 1 - \frac{3}{2}\left(2\mathbf{e}^{r^T}(\bar{R}+I_3)z + z^Tz\right) + \frac{15}{8}(2\mathbf{e}^{r^T}z)^2 + O\left(\|\begin{bmatrix} z \\ \bar{R}\end{bmatrix}\|^3\right)\end{aligned} \tag{2.20}$$

So,

$$\begin{aligned}&\frac{(\bar{R}+I_3)^T\mathbf{e}^r + z}{\|(\bar{R}+I_3)^T\mathbf{e}^r + z\|_2^3} - \mathbf{e}^r = -\mathbf{e}^r + \left((\bar{R}+I_3)^T\mathbf{e}^r + z\right) \\ &\quad - \frac{3}{2}\left(2\mathbf{e}^{r^T}(\bar{R}+I_3)z + z^Tz\right)\left((\bar{R}+I_3)^T\mathbf{e}^r + z\right) \\ &\quad + \frac{15}{8}(2\mathbf{e}^{r^T}z)^2\left((\bar{R}+I_3)^T\mathbf{e}^r + z\right) + O\left(\|\begin{bmatrix} z \\ \bar{R}\end{bmatrix}\|^3\right) \\ &\quad = \left(\bar{R}^T\mathbf{e}^r + z\right) - \frac{3}{2}\left(2\mathbf{e}^{r^T}\bar{R}z + z^Tz\right)\left(\bar{R}^T\mathbf{e}^r + \mathbf{e}^r + z\right) \\ &\quad - 3\mathbf{e}^{r^T}z\left(\bar{R}^T\mathbf{e}^r + \mathbf{e}^r + z\right) + \frac{15}{2}(\mathbf{e}^{r^T}z)^2\mathbf{e}^r + O\left(\|\begin{bmatrix} z \\ \bar{R}\end{bmatrix}\|^3\right) \\ &\quad = \left[\bar{R}^T - \frac{3}{2}\left(2\mathbf{e}^{r^T}\bar{R}z + z^Tz\right) - 3\mathbf{e}^{r^T}z\left(\bar{R}^T + I_3\right) + \frac{15}{2}(\mathbf{e}^{r^T}z)^2\right]\mathbf{e}^r \\ &\quad + \left(I_3 - 3\mathbf{e}^{r^T}z\right)z + O\left(\|\begin{bmatrix} z \\ \bar{R}\end{bmatrix}\|^3\right) \\ &\quad = \left[\left(\bar{R}^T - 3\mathbf{e}^{r^T}z\right)\mathbf{e}^r + z\right] + \left(-\frac{3}{2}\left(2\mathbf{e}^{r^T}\bar{R}z + z^Tz\right) - 3\mathbf{e}^{r^T}z\bar{R}^T + \frac{15}{2}(\mathbf{e}^{r^T}z)^2\right)\mathbf{e}^r \\ &\quad - 3(\mathbf{e}^{r^T}z)z + O\left(\|\begin{bmatrix} z \\ \bar{R}\end{bmatrix}\|^3\right)\end{aligned}$$

Substituting into (2.19),

$$
\begin{aligned}
G(R,x) - G(R^r,x^r) = -\frac{\mu}{\kappa^3}\kappa\frac{1}{\|\tilde{x}^r\|_2^2}(&\left[\left(\bar{R}^T - 3\mathrm{e}^{rT}z\right)\mathrm{e}^r + z\right] \\
&+ \left[\left(-\frac{3}{2}\left(2\mathrm{e}^{rT}\bar{R}z + z^Tz\right) - 3\mathrm{e}^{rT}z\bar{R}^T + \frac{15}{2}(\mathrm{e}^{rT}z)^2\right)\mathrm{e}^r - 3(\mathrm{e}^{rT}z)z\right] \\
&+ O\left(\| \begin{bmatrix} z \\ \bar{R} \end{bmatrix} \|^3\right)) \\
= -\eta^2\kappa(&\frac{1}{\|\tilde{x}^r\|_2}\left[\left(\frac{1}{\|\tilde{x}^r\|_2}\bar{R}^T - 3\mathrm{e}^{rT}\bar{\tilde{x}}\right)\mathrm{e}^r + \bar{\tilde{x}}\right] \\
&+ \frac{1}{\|\tilde{x}^r\|_2}\left[-3\mathrm{e}^{rT}\bar{R}\bar{\tilde{x}} - 3\mathrm{e}^{rT}\bar{\tilde{x}}\bar{R}^T\right]\mathrm{e}^r \\
&+ \left[\left(-\frac{3}{2}\left(\bar{\tilde{x}}^T\bar{\tilde{x}}\right) + \frac{15}{2}(\mathrm{e}^{rT}\bar{\tilde{x}})^2\right)\mathrm{e}^r - 3(\mathrm{e}^{rT}\bar{\tilde{x}})\bar{\tilde{x}}\right] + O\left(\frac{1}{\|\tilde{x}^r\|_2^2}\begin{bmatrix}\bar{\tilde{x}} \\ \bar{R}\end{bmatrix}^3\right))
\end{aligned}
$$

While this gravity approximation is quadratic, it does not (in general) fit the form of (2.8) due to the presence of $\|\tilde{x}^r\|_2$ in the quadratic terms containing $\bar{R}$. This leaves two options to make the equations of relative motion compliant, either assuming $\|\tilde{x}^r\|_2$ is constant or reducing the problem to translation only. It can be seen from this force approximation and the torque formulation that the variables causing the time variance of the linear term are x^r, v^r, and ω^r. When these are constant, the resultant reference trajectory is a circular orbit, about which both the rotation and translation dynamics can be addressed.

Note that reduction of the problem to a translation only case results in a much simpler form of the equations of relative motion:

$$
\dot{\xi} = \begin{bmatrix}\dot{\bar{x}} \\ \dot{\bar{v}}\end{bmatrix} = \begin{bmatrix}\bar{v} - \widehat{\omega}^r\bar{x} \\ -\widehat{\omega}^r\bar{v} + (G(R,x) - G(R^r,x^r)) - K_{vv}\bar{v} + \eta^2\kappa B_v\bar{u}\end{bmatrix},
$$

where

$$G(R,x) - G(R^r,x^r) = -\eta^2\kappa(\frac{1}{\|\tilde{x}^r\|_2}\left[-3\mathbf{e}^{rT}\bar{\tilde{x}}\mathbf{e}^r + \bar{\tilde{x}}\right] + \left[\left(-\frac{3}{2}\left(\bar{\tilde{x}}^T\bar{\tilde{x}}\right) + \frac{15}{2}(\mathbf{e}^{rT}\bar{\tilde{x}})^2\right)\mathbf{e}^r - 3(\mathbf{e}^{rT}\bar{\tilde{x}})\bar{\tilde{x}}\right] + O\left(\frac{1}{\|\tilde{x}^r\|_2^2}\begin{bmatrix}\bar{\tilde{x}}\\ \bar{R}\end{bmatrix}^3\right))$$

Note that, as mentioned earlier, drag at orbital altitudes is linear.

If we restrict ourselves to motion about a circular trajectory, assume no drag, and let $\mathbf{e}^r = [0,1,0]^T$ and $\omega^r = [0,0,\eta]^T$, we get a form of London's equations [82]:

$$\dot{\bar{\xi}} == \begin{bmatrix}\dot{\bar{\tilde{x}}}_1\\ \dot{\bar{\tilde{x}}}_2\\ \dot{\bar{\tilde{x}}}_3\\ \dot{\bar{\tilde{v}}}_1\\ \dot{\bar{\tilde{v}}}_2\\ \dot{\bar{\tilde{v}}}_3\end{bmatrix} = \eta\begin{bmatrix}\begin{bmatrix}\bar{\tilde{v}}_1 + \bar{\tilde{x}}_2\\ \bar{\tilde{v}}_2 - \bar{\tilde{x}}_1\\ \bar{\tilde{v}}_3\end{bmatrix}\\ \begin{bmatrix}\bar{\tilde{v}}_2 - \bar{\tilde{x}}_1 + 3\bar{\tilde{x}}_2\bar{\tilde{x}}_1\\ -\bar{\tilde{v}}_1 + 2\bar{\tilde{x}}_2 - 3\bar{\tilde{x}}_2{}^2 + \frac{3}{2}\left(\bar{\tilde{x}}_1{}^2 + \bar{\tilde{x}}_3{}^2\right)\\ -\bar{\tilde{x}}_3 + 3\bar{\tilde{x}}_2\bar{\tilde{x}}_3\end{bmatrix} + \quad B_v\bar{u}\end{bmatrix} \tag{2.21}$$

The acceleration due to gravitational torque term can be cast into the necessary form in a similar way by using the expansion in (2.20).

Chapter 3

Series-based Local Planning

3.1 Introduction

This chapter[1] develops local complete constructive trajectory generation and optimization algorithms for a class of low order polynomial systems which is representative of a large array of dynamical systems. These algorithms are complete in that they guarantee a solution and are "constructive" in the sense that they rely directly on the controllability properties of the system.

This chapter presents two algorithms to generate a feasible path using base functions and a minimum energy path parameterized by the initial values of the costates of the system. For a linearly controllable system, we can show that there exists a neighborhood about the origin in which the algorithms are guaranteed to find a solution. To find these parameterized controls, we develop iterative as well as series inversion methods, both of which have convergence guarantees. We provide proofs to this along with computation of explicit neighborhoods that, even if conservative, provide a lower bound on region of validity, or the region over which these algorithms are guaranteed to converge. Additionally, we investigate the behavior of the algorithms for a one dimensional system and for a planar vertical takeoff and landing vehicle (PVTOL) with damping. This includes an

[1] This chapter and the appendices are an extension of the paper [28]. Portions of this chapter and the appendices are taken directly from the paper and are © 2003 IEEE, reprinted with permission, from W. T. Cerven and F. Bullo. Constructive controllability algorithms for motion planning and optimization. *IEEE Transactions on Automatic Control*, 48(4):575–589, 2003.

illustration of the level of conservativeness of the lower bounds on the neighborhoods of convergence for the example systems.

This chapter is organized as follows. In Section 3.2, we introduce the polynomial systems of interest and define the norms and series expansions upon which these algorithms are based. We discuss the accessibility and the nilpotency of the polynomial system as well as the formulation required to apply the series expansion about a trajectory. Next, in Section 3.3, we present trajectory generation and optimization problems and we cast both of them into the form of a function inversion problem. In Section 3.4, we proceed to show how a unique solution to the inversion problem exists locally, and define two numerical approaches to compute it. A lower bound to the radius of convergence is provided for both methods. Lastly, in Section 3.5, we apply these algorithms to a simple one dimensional example and the PVTOL with damping. Appendices A.2, A.3, and A.4 contain various proofs.

3.2 A class of polynomial control systems

Throughout the chapter we shall concern ourselves with n-dimensional second order polynomial systems of the form

$$\begin{aligned} \dot{x} &= A(t)x + f^{[2]}(x,x) + Bu \\ x(0) &= x_0, \end{aligned} \tag{3.1}$$

where $f^{[2]} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ is a symmetric tensor,[2] A is an $n \times n$ matrix, and B is an $n \times m$ matrix. While the approach advocated in this work can be extended to address more general systems, we focus on this class of polynomial systems for simplicity of exposition. This class is representative of a large array of dynamical systems, as any smooth system linear in controls not fitting this form naturally can be approximated as such by a Taylor expansion. Classical dynamical systems such as the Lorentz, Lotka-Volterra, and Euler equations are characterized by second order polynomial vector fields. In addition, Kang and Krener [63] showed that any nonlinear system of the form $\dot{\xi} = f(\xi) + g(\xi)u$ can

[2] Any vector field with components homogeneous polynomials of degree 2 can be written in terms of a symmetric tensor $f^{[2]}$.

be represented as such (plus higher order terms) by a change of coordinates and state feedback. Note that this class of polynomial systems is not contained in the class of systems in chained form, driftless systems, and feedback linearizable systems.

3.2.1 Operator and function norms

In defining mapping and norms we follow the notation in [68, Chapter 6].

Let $\mathbb{N}$ be the set of strictly positive integers. Over the linear space $\mathbb{R}^n$ we will use the norms $\|x\|_2 = \sqrt{x'x}$, and $\|x\|_\infty = \max_{i\in\{1,\dots,n\}} |x_i|$. Consider the normed linear space $\mathcal{L}^n_\infty$ of piecewise continuous, uniformly bounded functions over the interval I

$$\begin{aligned} x : I \subset \mathbb{R}_+ &\to \mathbb{R}^n \\ t &\mapsto x(t), \end{aligned}$$

with norm

$$\|x\|_{\mathcal{L}_\infty} = \sup_{t\in I} \|x(t)\|_\infty = \sup_{t\in I} \max_{i\in\{1,\dots,n\}} |x_i(t)| < +\infty.$$

Assume the system $\dot{x} = A(t)x$ is stable in the sense of Lyapunov [3] or that the interval I is finite and the escape time of $\dot{x} = A(t)x$ (if it exists) is not in the interval, and let H_A be the mapping

$$\begin{aligned} H_A : \mathcal{L}^n_\infty &\to \mathcal{L}^n_\infty \\ x(t) &\mapsto \int_0^t \Psi(t,\tau)x(\tau)d\tau. \end{aligned}$$

where $\Psi(t,\tau)$ is the linear state transition matrix of the system $\dot{x} = A(t)x$. The $\mathcal{L}^n_\infty$ induced norm for H_A is

$$\|H_A\|_{\mathcal{L}_\infty} = \|\Psi(t,0)\|_{\mathcal{L}_1} = \max_{i=1,\dots,n} \sum_{j=1}^{n} \int_I |(\Psi(t,0)_{ij}|dt.$$

[3] For the time-invariant system, it is sufficient the matrix A is Hurwitz.

Next, we consider 2-tensor $f^{[2]} : \mathcal{L}_\infty^n \times \mathcal{L}_\infty^n \to \mathcal{L}_\infty^n$ defined via

$$(x(t), y(t)) \mapsto f^{[2]}(x(t), y(t)),$$

and define its induced norm $\|f^{[2]}\|_{\mathcal{L}_\infty}$ via

$$\|f^{[2]}\|_{\mathcal{L}_\infty} = \|f^{[2]}\|_\infty = \max_{\substack{\|y_1\|_\infty = 1 \\ \|y_2\|_\infty = 1}} \|f^{[2]}(y_1, y_2)\|_\infty.$$

3.2.2 Evolution as series expansion

We present a series expansion for the solution of the initial value problem in equation (3.1). The result is an extension of the results in [21] and is proven in Appendix A.2. Before proceeding, it is useful to introduce a few preliminary concepts. The Catalan numbers are an infinite sequence of integers discovered by Euler as the solution to the question *"How many ways can a convex polygon be divided into triangles via non-intersecting diagonals?"*. The result is a sequence of numbers corresponding to polygons with increasing numbers of vertices, starting with a triangle. We define the Catalan numbers $\{c_k \in \mathbb{R}, k \in \mathbb{N}\}$ as in [130, Section 2.3]; our definition differs from Euler's more standard sequence by a scaling factor. Define $\mathcal{C} : [0, 1] \to [0, 1]$ as $\mathcal{C}(\eta) = 1 - \sqrt{1 - \eta}$, and let $\text{Remainder}_K(\mathcal{C})(\eta)$ be its Taylor remainder of order K. If we develop $\mathcal{C}$ in power series

$$\mathcal{C}(\eta) = \sum_{k=1}^{+\infty} c_k \eta^k,$$

then the following equivalent conditions hold

$$c_k = \frac{1}{k\, 2^{2k-1}} \binom{2k-2}{k-1}, \qquad \text{and} \qquad c_1 = \frac{1}{2}, \;\; c_k = \frac{1}{2} \sum_{i=1}^{k-1} c_i c_{k-i}. \tag{3.2}$$

We are now able to characterize the flow of the differential equation (3.1).

Lemma 3.2.1 *The solution of the system in equation* (3.1) *is written as a series*

$x(t) = \sum_{k=1}^{+\infty} x_k(t)$ *where*

$$\begin{aligned} x_1(t) &= \Psi(t,0)x_0 + \int_0^t \Psi(t,\tau)Bu(\tau)d\tau \\ x_k(t) &= \sum_{a=1}^{k-1} \int_0^t \Psi(t,\tau) f^{[2]}(x_a(\tau), x_{k-a}(\tau))d\tau, \quad \forall k > 1, \end{aligned} \tag{3.3}$$

$\Psi(t,\tau)$ is the linear system state transition matrix and satisfies the relation $\dot{\Psi}(t,\tau) = A(t)\Psi(t,\tau)$, $\Psi(\tau,\tau) = I_n$, and I_n is the $n \times n$ identity matrix.

Let $d_1 = 2\left(\|\Psi(t,0)x_0\|_{\mathcal{L}_\infty} + \|\Psi(t,0)\|_{\mathcal{L}_1}\|Bu\|_{\mathcal{L}_\infty}\right)$ and $d_2 = 2\|\Psi(t,0)\|_{\mathcal{L}_1}\|f^{[2]}\|_{\mathcal{L}_\infty}$. Provided $d_1 d_2 \leq 1$, a solution exists over the interval I and the series converges absolutely and uniformly in $t \in I$, and the following upper bounds hold:

$$\|x_k\|_{\mathcal{L}_\infty} \leq c_k\, d_1^k d_2^{k-1},$$
$$\|x - \sum_{k=1}^{K} x_k\|_{\mathcal{L}_\infty} \leq \frac{1}{d_2}\,\text{Remainder}_K(C)(d_1 d_2).$$

3.2.3 Accessibility and nilpotency

Consider the polynomial control system in equation (3.1) and described by the tensors $A(t)$, B, and $f^{[2]}$. Let the subspace $\mathcal{B} \subset \mathbb{R}^n$ be the image of the matrix B. Given two linear subspaces V and W of $\mathbb{R}^n$, let

$$f^{[2]}(V,W) = \{f^{[2]}(v,w) \in \mathbb{R}^n \mid v \in V, w \in W\} \subset \mathbb{R}^n.$$

Let $\text{LinReach}_A(\mathcal{B})$ be the subspace generated by the reachability Grammian

$$\int_0^t \Psi(t,s)BB'\Psi(t,s)'ds,$$

where Ψ is the linear system state transition matrix, and define the *accessibility subspaces* $\{\mathcal{R}_k \subset \mathbb{R}^n, k \in \mathbb{N}\}$ as follows

$$\mathcal{R}_1 = \text{LinReach}_A(\mathcal{B}) = \text{range}(\int_0^t \Psi(t,s)BB'\Psi(t,s)'ds)\mathcal{R}_2 \quad = \text{LinReach}_A(f^{[2]}(\mathcal{R}_1, \mathcal{R}_1))$$
$$\vdots$$
$$\mathcal{R}_k = \text{LinReach}_A(\cup_{a=1}^{k-1}\{f^{[2]}(\mathcal{R}_a, \mathcal{R}_{k-a})\}).$$

The subspaces $\{\mathcal{R}_k \subset \mathbb{R}^n, k \in \mathbb{N}\}$ play a key role in studying controllability and nilpotency of the time-invariant form of system (3.1). In particular, we state the following facts:

(i) the kth order component $x_k(t)$ is in $\mathcal{R}_k$ for all $t \in I$ and for all inputs $u : I \to \mathbb{R}^m$,

(ii) the system is *linearly controllable* if and only if $\mathcal{R}_1$ is full rank,

(iii) the accessibility subspace $\mathcal{R}_k$ of the time-invariant system is generated by all the Lie brackets evaluated at the origin — between an arbitrary number of the vector field $Ax + f^{[2]}(x,x)$ and k vector fields of the form B_i,

(iv) the system is *locally accessible* if $\sum_{k=1}^{+\infty} \mathcal{R}_k = \mathbb{R}^n$, and

(v) the system is *nilpotent* if there exists an integer k such that $\mathcal{R}_i = 0$ for all $i \geq k$.

Note that $\mathcal{R}_1$ is full rank if and only if the controllability grammian $\int_0^t \Psi(0,s)BB'\Psi(0,s)'ds$ is also full rank.

3.2.4 Series expansion about a trajectory

As described in Lemma 3.2.1 the series expansion in equation (3.3) converges under the assumption of small initial condition $x(0)$. There is a second setting in which a similar expansion can be easily written. Assume that a reference trajectory satisfying $\dot{x} = A(t)x + f^{[2]}(x,x) + Bu_{\text{ref}}(t)$, $x(0) = x_0$ is known analytically as $x(t) = \Phi_t^f(x_0, u_{\text{ref}}(t))$. Define a relative variation variable e

$$e = x - \Phi_t^f(x_0, u_{\text{ref}}(t)),$$

and compute the differential equation regulating its evolution

$$\begin{aligned}
\dot{e} &= \dot{x} - \dot{\Phi}_t^f(x_0, u_{\text{ref}}(t)) \\
&= \Big(A(t)x + f^{[2]}(x,x) + Bu\Big) \\
&- \Big(A(t)\Phi_t^f(x_0, u_{\text{ref}}(t)) + f^{[2]}(\Phi_t^f(x_0, u_{\text{ref}}(t)), \Phi_t^f(x_0, u_{\text{ref}}(t))) + Bu_{\text{ref}}(t)\Big) \\
&= A(t)e + f^{[2]}(x + \Phi_t^f(x_0, u_{\text{ref}}(t)), x - \Phi_t^f(x_0, u_{\text{ref}}(t))) + B(u - u_{\text{ref}}(t)) \\
&= A(t)e + f^{[2]}(2\Phi_t^f(x_0, u_{\text{ref}}(t)) + e, e) + B(u - u_{\text{ref}}(t)) \\
&= \Big(A(t) + 2f^{[2]}(\Phi_t^f(x_0, u_{\text{ref}}(t)))\Big)\, e + f^{[2]}(e,e) + B(u - u_{\text{ref}}(t)),
\end{aligned}$$

where we define the matrix $f^{[2]}(x)$ according to $\Big(f^{[2]}(x)\Big)\, y = f^{[2]}(x,y)$ for all $x, y \in \mathbb{R}^n$.

In the new variable e, the system is again in second order polynomial form and the initial condition is $e(0) = 0$.

3.3 Formulation of motion planning and minimum energy planning problems

This section describes two interesting planning problems [4]. We transform these problems into inverse function problems exploiting the series expansion described above.

Consider the control system in equation (3.1), let the initial condition be the origin $x(0) = 0$, and let $x_{\text{target}} \in \mathbb{R}^n$ be the desired target location. We shall require that $d_1 d_2 \leq 1$, i.e., we restrict our investigation to the convergence radius of the series in equation (3.3).

3.3.1 Base functions for the control inputs

It is often useful to introduce a collection of bounded piecewise continuous base functions $\{\psi^i(t) : [0,T] \mapsto \mathbb{R}^m, i \in \{1, \ldots, n_p\}\}$ to parametrize the input functions u. This is the case for example when magnitude and rate constraints or binary actuators are present.

[4] A third approach is presented in Appendix A

We write

$$u(t) = \sum_{i=1}^{n_p} \psi^i(t) p_i = \psi(t) p.$$

A wide variety of base functions are possible including splines, Hermite polynomials, sinusoidal functions, piecewise constant functions, and wavelets. Define the tensors $\{\Phi_k : \mathbb{R}^{k \times n_p} \to \mathbb{R}^n, k \in \mathbb{N}\}$ as:

$$\begin{aligned}
\Phi_1^i(t) &= \int_0^t \Psi(t,\tau) B \psi^i(\tau) d\tau \\
\Phi_2^{i_1 i_2}(t) &= \int_0^t \Psi(t,\tau) f^{[2]}\left(\Phi_1^{i_1}(\tau), \Phi_1^{i_2}(\tau)\right) d\tau, \\
\Phi_3^{i_1 i_2 i_3}(t) &= \int_0^t \Psi(t,\tau) \Big(f^{[2]}(\Phi_1^{i_1}(\tau), \Phi_2^{i_2 i_3}(\tau)) + f^{[2]}(\Phi_2^{i_1 i_2}(\tau), \Phi_1^{i_3}(\tau)) \Big) d\tau \\
&\vdots \\
\Phi_k^{i_1 \dots i_k}(t) &= \sum_{a=1}^{k-1} \int_0^t \Psi(t,\tau) f^{[2]}(\Phi_a^{i_1 \dots i_a}(\tau), \Phi_{k-a}^{i_1 \dots i_{k-1}}(\tau)) d\tau.
\end{aligned} \tag{3.4}$$

Assuming $x(0) = 0$, the kth term of the series in equation (3.3) can now be rewritten as

$$x_k(t) = \Phi_k(t)(\underbrace{p, \dots, p}_{k \text{ times}}).$$

In what follows, we will only need $\Phi_k(t)$ evaluated at final time T, therefore we introduce the abbreviation $\Phi_k = \Phi_k(T)$.

3.3.2 Motion planning with base functions

Consider the following design problem: find a control input $u : [0, T] \mapsto R^m$ such that

$$\begin{aligned}
\dot{x} &= A(t)x + f^{[2]}(x, x) + Bu \\
x(0) &= 0, \;\; x(T) = x_{\text{target}}.
\end{aligned}$$

Using the series expansion characterization in Section 3.2.2, the problem becomes finding a control input $u : [0, T] \mapsto R^m$ that solves

$$x_{\text{target}} = \sum_{k=1}^{+\infty} x_k(T).$$

This equation is a constraint on the input functions u since all the terms x_k depend on it. This constraint can be discretized into a finite dimensional equation via a collection of bounded piecewise continuous base functions $\{\psi^i(t) : [0, T] \mapsto \mathbb{R}^m, i \in \{1, \ldots, n_p\}\}$. Using the notation in Section 3.3.1, the design problem is to find a vector $p \in \mathbb{R}^{n_p}$ such that

$$x_{\text{target}} = \Phi_1 p + \sum_{k=2}^{+\infty} \Phi_k(p, \ldots, p). \tag{3.5}$$

3.3.3 Minimum energy planning (without base functions)

Consider the following design problem: find a control input $u : [0, T] \mapsto R^m$ that solves

$$\begin{aligned} \min \ & \int_0^T \|u(t)\|_2^2 dt \\ \text{subject to } & \dot{x} = A(t)x + f^{[2]}(x, x) + Bu \\ & x(0) = 0, \ \ x(T) = x_{\text{target}}. \end{aligned} \tag{3.6}$$

Thus, the Hamiltonian associated with the optimal control problem in equation (3.6) is:

$$H(x, \lambda, u) = \frac{1}{2}\|u\|_2^2 + \lambda' \left(A(t)x + f^{[2]}(x, x) + Bu\right).$$

As known from optimal control theory, we let u extremize H, that is, we let $u(t) = -B'\lambda(t)$, where B' denotes the transpose of B and we write necessary conditions

$$\begin{aligned} \dot{x} &= A(t)x + f^{[2]}(x, x) - BB'\lambda \\ \dot{\lambda} &= -A(t)'\lambda - 2f^{[2]}(x)'\lambda \\ x(0) &= 0, \ \ x(T) = x_{\text{target}}. \end{aligned} \tag{3.7}$$

The design problem is to find the initial value $\lambda(0) = \lambda_0$ compatible with problem (3.7) that uniquely determines the optimal control law.

The two point boundary value problem has the same polynomial structure of the initial value problem in equation (3.1). We let $\overline{x} = (x, \lambda) \in \mathbb{R}^{2n}$, and

$$\overline{A}(t) = \begin{bmatrix} A(t) & -BB' \\ 0 & -A(t)' \end{bmatrix}, \quad \overline{f}^{[2]}(\overline{x}, \overline{x}) = \begin{bmatrix} f^{[2]}(x, x) \\ -2f^{[2]}(x)'\lambda \end{bmatrix},$$

so that the first order term in the solution to (3.7) is

$$\overline{x}_1(t) = \begin{bmatrix} x_1(t) \\ \lambda_1(t) \end{bmatrix} = \overline{\Psi}(t, 0) \begin{bmatrix} 0 \\ \lambda_0 \end{bmatrix} = \overline{\Phi}_1(t) \begin{bmatrix} 0 \\ \lambda_0 \end{bmatrix},$$

where $\overline{\Psi}(t, \tau)$ is the state transition matrix of the system $\dot{\overline{x}} = \overline{A}\overline{x}$ and $\overline{\Phi}_1$ now maps $\mathbb{R}^{2n} \to \mathbb{R}^{2n}$. The higher order terms $\{\overline{\Phi}_k : \mathbb{R}^{k \times 2n} \to \mathbb{R}^{2n}, k > 1\}$ can be recursively defined following equation (3.4) in Section 3.3.1. Using the series expansion characterization in Section 3.2.2, we have

$$\begin{bmatrix} x(T) \\ \lambda(T) \end{bmatrix} = \overline{\Phi}_1 \begin{bmatrix} 0 \\ \lambda_0 \end{bmatrix} + \sum_{k=2}^{+\infty} \overline{\Phi}_k \left(\begin{bmatrix} 0 \\ \lambda_0 \end{bmatrix}, \ldots, \begin{bmatrix} 0 \\ \lambda_0 \end{bmatrix} \right),$$

where we drop the T argument as usual. This expression can be rewritten as

$$x_{\text{target}} = \overline{\Phi}_{1,x\lambda}\lambda_0 + \sum_{k=2}^{+\infty} \overline{\Phi}_{k,x\lambda}(\lambda_0, \ldots, \lambda_0), \tag{3.8}$$

where we project the image and restrict the domain of the tensor $\{\overline{\Phi}_k, k \in \mathbb{N}\}$ as

$$\overline{\Phi}_k \left(\begin{bmatrix} 0 \\ \lambda_0 \end{bmatrix}, \ldots, \begin{bmatrix} 0 \\ \lambda_0 \end{bmatrix} \right) = \left(\begin{bmatrix} \overline{\Phi}_{k,x\lambda}(\lambda_0, \ldots, \lambda_0) \\ \overline{\Phi}_{k,\lambda\lambda}(\lambda_0, \ldots, \lambda_0) \end{bmatrix} \right).$$

In summary, the design problem is to find a vector $\lambda_0 \in \mathbb{R}^n$ solution to equation (3.8). Once an appropriate value of λ_0 is found, the optimal control law can be computed as a series expansion.

3.4 Solving the planning problems via inversion

In this section, we treat both motion planning and minimum energy planning as a function inversion problem for an appropriate function f characterized via a power series. We study conditions that guarantee that the function f and its Jacobian are invertible. Finally, we describe two approaches to inverting f and to bound the neighborhood over which the function is invertible.

Equations (3.5) and (3.8) are equivalent to the solution of an equation of the form

$$x_{\text{target}} = f(y) = f_1 y + \sum_{k=2}^{+\infty} f_k(y, \ldots, y), \tag{3.9}$$

where $y = p \in \mathbb{R}^{n_p}$ for motion planning and $y = \lambda_0 \in \mathbb{R}^n$ for minimum energy planning. Additionally, $x_{\text{target}} \in \mathbb{R}^n$, and the tensors f_k live in linear spaces of appropriate dimensions. Next, we transcribe the bounds known from Lemma 3.2.1 into the new setting. Let the sequence $\{c_k, k \in \mathbb{N}\}$ and the function $\mathcal{C}$ be defined as in Section 3.2.2.

Lemma 3.4.1 *Define*

$$D_1 = \begin{cases} 2\|\Psi(t,0)\|_{\mathcal{L}_1}\|B\psi(t)\|_{\mathcal{L}_\infty} \\ 2\|\overline{\Psi}(t,0)\|_{\mathcal{L}_\infty} \end{cases} \qquad \text{and} \qquad D_2 = \begin{cases} 2\|\Psi(t,0)\|_{\mathcal{L}_1}\|f^{[2]}\|_{\mathcal{L}_\infty} \\ 2\|\overline{\Psi}(t,0)\|_{\mathcal{L}_1}\|\overline{f}^{[2]}\|_{\mathcal{L}_\infty} \end{cases}.$$

Provided $D_1 D_2 \|y\|_\infty \leq 1$, the series converges absolutely, and the following upper bounds hold:

$$\|f_k(y, \ldots, y)\|_\infty \leq c_k \, D_1^k D_2^{k-1} \|y\|_\infty^k,$$
$$\|f(y) - \sum_{k=1}^{K} f_k(y, \ldots, y)\|_\infty \leq \frac{1}{D_2} \, \text{Remainder}_K(\mathcal{C}) \, (D_1 D_2 \|y\|_\infty).$$

Proof: We relate the coefficients $\{d_1, d_2\}$ for each settings to $\{D_1, D_2\}$ via

$$\begin{aligned} d_1 &= 2\left(\|\Psi(t,0)x_0\|_{\mathcal{L}_\infty} + \|\Psi(t,0)\|_{\mathcal{L}_1}\|Bu\|_{\mathcal{L}_\infty}\right) \\ &= \begin{cases} 2\|\Psi(t,0)\|_{\mathcal{L}_1}\|B\psi(t)c\|_{\mathcal{L}_\infty} \\ 2\left\|\overline{\Psi}(t,0)\begin{bmatrix}0\\ \lambda_0\end{bmatrix}\right\|_{\mathcal{L}_\infty} \end{cases} \leq D_1\|y\|_\infty \\ d_2 &= D_2. \end{aligned}$$

From Lemma 3.2.1 we transcribe the first of the two bounds

$$\|f_k(y,\ldots,y)\|_\infty \leq \begin{cases} \|\Phi_k(c,\ldots,c)\|_\infty \\ \|\overline{\Phi}_k(\lambda_0,\ldots,\lambda_0)\|_\infty \end{cases} = \begin{cases} \|x_k\|_\infty \\ \|\overline{x}_k\|_\infty \end{cases} \leq c_k\, d_1^k d_2^{k-1} \leq c_k\, D_1^k D_2^{k-1}\|y\|_\infty^k.$$

The second bound can be proven using the definition of remainder:

$$\begin{aligned} \|f(y) - \sum_{k=1}^{K} f_k(y,\ldots,y)\|_\infty &\leq \frac{1}{d_2}\,\text{Remainder}_K(\mathcal{C})(d_1 d_2) \\ &\leq \frac{1}{d_2}\sum_{k=K+1}^{+\infty} c_k (d_1 d_2)^k \leq \frac{1}{D_2}\sum_{k=K+1}^{+\infty} c_k \left(D_1 D_2\|y\|_\infty\right)^k \\ &\leq \frac{1}{D_2}\,\text{Remainder}_K(\mathcal{C})\left(D_1 D_2\|y\|_\infty\right). \end{aligned}$$

■

Theorem 3.4.2 (A generalized inverse function theorem) *Assume the power series in equation* (3.9) *converges absolutely over* $V_a = \{y \in \mathbb{R}^{n_p} |\ D_1 D_2\|y\|_\infty \leq 1\}$, *and let* $f : \mathbb{R}^{n_p} \to \mathbb{R}^n$, $n_p \geq n$ *be the function defined by the series. Furthermore, assume that the tensor* f_1 *is full rank. Then there exists a neighborhood* $V_b \subseteq V_a$ *such that, for all* $x_{\text{target}} \in f(V_b)$, *there exists a smooth right inverse* $f^{-1} : f(V_b) \to V_b$. *Furthermore, if* $n_p = n$, f^{-1} *is unique.*

Proof: It can be seen that $f_1 = \frac{\partial f}{\partial y}(0)$ is the Jacobian of f evaluated at $y = 0$. Since f_1 is full rank, we can compute its pseudo-inverse f_1^p. Let $\chi \in \mathbb{R}^n$ and let $y = f_1^p \chi$.

Then equation (3.9) becomes

$$x_{\text{target}} = \chi + \sum_{k=2}^{+\infty} f_k(f_1^p\chi, \ldots, f_1^p\chi) = h(\chi).$$

The Jacobian of the function $h : \mathbb{R}^n \to \mathbb{R}^n$ evaluated at the origin is

$$\frac{\partial h}{\partial \chi}(0) = I_n.$$

Therefore, the function h has a unique inverse in a neighborhood of the origin because of Theorem 2.5.2 in [2]. This implies that f^{-1} exists in a neighborhood of the origin. Furthermore, when $n_p = n$, the inverse function f^{-1} is unique since the pseudo-inverse f_1^p equals f_1^{-1}. ■

3.4.1 Existence of solution for linearly controllable systems

Motivated by the previous theorem, we investigate necessary and sufficient conditions in order for the tensor f_1 to be full rank. It turns out that in both settings the property of linear controllability plays a central role.

Lemma 3.4.3 *There exist smooth base functions $\{\psi^i(t) : i \in \{1, \ldots, n_p\}\}$ such that the tensor Φ_1 is invertible if and only if the system in equation (3.1) is linearly controllable.*

Proof: If the tensor Φ_1 is full rank, then the linear systems obtained by setting $f^{[2]}$ to zero is controllable, and therefore the system in equation (3.1) is linearly controllable. Vice-versa, let $n_p = n$ and define the functions

$$\psi^i(t) = B'\Psi(0,t)'W_T^{-1}\mathrm{e}_i \tag{3.10}$$
$$W_T = \int_0^T \Psi(0,s)BB'\Psi(0,s)'ds = W_T' > 0,$$

where $\{\mathrm{e}_1, \ldots, \mathrm{e}_n\}$ is the standard base for $\mathbb{R}^n$ and W_T is the controllability Grammian. As this system is linearly controllable by assumption, W_T is full rank. Given these input base functions, it is easy to see that $\Phi_1 = I_n$. ■

The base functions in equation (3.10) are selected according to the classic *minimum energy design* for point to point planning of linear control systems; see [31, page 557].

Lemma 3.4.4 *The tensor $\overline{\Phi}_{1,x\lambda}$ is invertible if and only if the system in equation* (3.1) *is linearly controllable.*

Proof: The tensor $\overline{\Phi}_{1,x\lambda}$ can be found by solving the differential equation

$$\dot{\overline{\Phi}}_1 = \begin{bmatrix} \dot{\overline{\Phi}}_{1,xx} & \dot{\overline{\Phi}}_{1,x\lambda} \\ \dot{\overline{\Phi}}_{1,\lambda x} & \dot{\overline{\Phi}}_{1,\lambda\lambda} \end{bmatrix} = \begin{bmatrix} A(t) & -BB' \\ 0 & -A(t)' \end{bmatrix} \begin{bmatrix} \overline{\Phi}_{1,xx} & \overline{\Phi}_{1,x\lambda} \\ \overline{\Phi}_{1,\lambda x} & \overline{\Phi}_{1,\lambda\lambda} \end{bmatrix}, \quad \overline{\Phi}_i(0) = I_n,$$

which simplifies to

$$\dot{\overline{\Phi}}_{1,x\lambda} = A(t)\overline{\Phi}_{1,x\lambda} - BB'\Psi(0,t)', \quad \overline{\Phi}_{1,x\lambda}(0) = 0_n.$$

The solution to the last equation is the convolution integral

$$\overline{\Phi}_{1,x\lambda} = -\Psi(T,0)\left(\int_0^T \Psi(0,s)BB'\Psi(0,s)'ds\right).$$

Since $\overline{\Phi}_{1,x\lambda}$ is the negative of the product of an invertible matrix $\Psi(T,0)$ and the controllability Grammian of the system $(A(t),B)$, $\overline{\Phi}_{1,x\lambda}$ is full rank and invertible if and only if the system $(A(t),B)$ is controllable. ■

3.4.2 Existence of solution for linearly uncontrollable systems

Now let us consider time-invariant systems that are not linearly controllable.

Theorem 3.4.5 *Given the n-dimensional time invariant dynamical equation* (3.1), *if its linear controllability matrix has rank $n_c < n$, then there exists a transformation $\tilde{x} = Px$, where P is a constant nonsingular matrix, which transforms* (3.1) *into*

$$\dot{\tilde{x}} = \begin{bmatrix} \tilde{A}_c & \tilde{A}_{12} \\ \tilde{A}_{nc} & 0 \end{bmatrix} \tilde{x} + \tilde{f}^{[2]}(\tilde{x},\tilde{x}) + \begin{bmatrix} \tilde{B}_c \\ 0 \end{bmatrix} u \tag{3.11}$$

with the controllable n_c-dimensional subsystem

$$\dot{\tilde{x}}_c = \tilde{A}_c \tilde{x}_c + \tilde{B}_c u. \tag{3.12}$$

This transformation is called the system's *canonical decomposition*; see [31]. The matrix P can be defined such that P^{-1} is composed of first n_c independent columns of the controllability matrix $[B \quad AB \ldots A^{n-1}B]$ augmented by arbitrary vectors that make the matrix nonsingular. The linear controllability Grammian $\widetilde{W}_t$ for the canonical decomposition (3.11) reduces to

$$\begin{bmatrix} \widetilde{W}_{t_c} & 0 \\ 0 & 0_{n-n_c} \end{bmatrix}, \tag{3.13}$$

where $\widetilde{W}_{t_c} = \int_0^t e^{\tilde{A}_c(t-s)} \tilde{B}_c \tilde{B}_c' e^{\tilde{A}_c'(t-s)} ds$ is the controllability Grammian for the system (3.12). Using the canonical decomposition, the inversion problem (3.9) can be recast as

$$\begin{bmatrix} \tilde{x}_{c_{\text{target}}} \\ \tilde{x}_{nc_{\text{target}}} \end{bmatrix} = \tilde{f}(\tilde{y}) = \begin{bmatrix} \tilde{f}_{c_1} \\ \tilde{f}_{nc_1} \end{bmatrix} \tilde{y} + \sum_{k=2}^{+\infty} \begin{bmatrix} \tilde{f}_{c_k} \\ \tilde{f}_{nc_k} \end{bmatrix} (\tilde{y}, \ldots, \tilde{y}), \tag{3.14}$$

where $\tilde{x}_{c_{\text{target}}} \in \mathbb{R}^{n_c}$ and $\tilde{x}_{nc_{\text{target}}} \in \mathbb{R}^{n-n_c}$. Let us then ignore motion on the linearly uncontrollable space and reduce our planning problem to that on the linearly controllable space $\mathbb{R}^{n_c}$

$$\tilde{x}_{c_{\text{target}}} = \tilde{f}_c(\tilde{y}_c) = \tilde{f}_{c_1} \tilde{y}_c + \sum_{k=2}^{+\infty} \tilde{f}_{c_k}(\tilde{y}_c, \ldots, \tilde{y}_c), \tag{3.15}$$

where $\tilde{y} = [\tilde{y}_c' \quad \tilde{y}_{nc}']' \in \mathbb{R}^{n_p}$ and $\tilde{y}_c \in \mathbb{R}^{n_c}$.

This problem is now in a form where the inverse function theorem can be applied, so we again investigate necessary and sufficient conditions in order for the tensor $\tilde{f}_{c_1}$ to be full rank.

Lemma 3.4.6 *There exist smooth base functions $\{\psi^i(t) : i \in \{1, \ldots, n_p\}\}$ such that the tensor $\tilde{f}_{c_1} = \tilde{\Phi}_{c_1}$ is invertible if and only if the system in equation* (3.1) *is linearly controllable on the space $\mathbb{R}^{n_c}$.*

Proof: This proof follows that of Lemma 3.4.3, where $n_p = n$ and the base functions

are defined as

$$\psi^i(t) = \tilde{B}' e^{\tilde{A}'(T-t)} \begin{bmatrix} \widetilde{W}_{T_c}^{-1} & 0 \\ 0 & 0_{n-n_c} \end{bmatrix} \mathrm{e}_i,$$

where $\{\mathrm{e}_1, \ldots, e_n\}$ is the standard base for $\mathbb{R}^n$. It is then straightforward to find that

$$\tilde{f}_1 = \begin{bmatrix} I_{n_c} & 0 \\ 0 & 0_{n-n_c} \end{bmatrix}.$$

and $\tilde{f}_{c_1} = I_{n_c}$ is invertible. ■

Lemma 3.4.7 *The tensor $\tilde{f}_{c_1} = \overline{\overline{\Phi}}_{c_{1,x\lambda}}$ is full rank and is invertible if and only if the system in equation* (3.1) *is linearly controllable on the space $\mathbb{R}^{n_c}$.*

Proof: This proof follows that of Lemma 3.4.4. The linear term $\tilde{f}_1$ is then defined as

$$\tilde{f}_1 = \overline{\overline{\Phi}}_{1,x\lambda} = -e^{\tilde{A}T} \left(\int_0^T e^{-\tilde{A}s} \tilde{B} \tilde{B}' e^{-\tilde{A}'s} ds \right) = \begin{bmatrix} \widetilde{W}_{T_c} & 0 \\ 0 & 0_{n-n_c} \end{bmatrix}$$

It can then be seen that $\tilde{y} = \tilde{\lambda} \in \mathbb{R}^n$ and $\tilde{f}_{c_1} = \widetilde{W}_{T_c}$ has inverse $(\widetilde{W}_{T_c}^{-1})$ if and only if the system (3.11) is linearly controllable. ■

Remark 3.4.8 *The treatment of linearly uncontrollable systems is particularly important when considering nonminimum or redundant coordinate representations. Here, a* nonminimum coordinate representation *is a coordinate parametrization of a configuration space for which the number of coordinates exceeds the dimension of the space. Such representations are often important to avoid singularities and write certain dynamical system in quadratic form* (3.1)*. For example, unit quaternions or rotation matrices are very common to model planar and 3D orientations. Furthermore, they are naturally associated to quadratic vector fields; see the discussion on the model in equation* (2.9) *and the PVTOL with damping example below.*

Remark 3.4.9 *It should be possible to extend this method to the time-varying case as a transformation can be based on a time-varying controllability matrix. This approach is alluded to by Loukianov and Utkin [83] in their reference to [118]. However, as this approach is nontrivial and its investigation would require significant effort beyond the scope of this work, it is not included here.*

3.4.3 Computational approaches

This section presents two methodologies to solve the inverse function problem under the linear controllability assumption. First, we note that equation (3.9) can be solved numerically by a root-finding method such as the classic Newton's method. This type of routine is well-documented in books such as [106] and its implementation is relatively straightforward. Along these lines, we present here a provably convergent iterative method based upon the contraction mapping. We provide an explicit lower bound on the region of convergence of the algorithm. Second, we provide an explicit inverse function written in power series expansion. The closed form expressions here are taken from [51, 102, 52]. Again, we provide an explicit lower bound on the region of convergence of the algorithm.

Iterative contraction algorithm

Define the pseudo-inverse f_1^p and let $y = f_1^p \chi$, where χ is the new free variable living in $\mathbb{R}^n$. We rewrite equation (3.9) into the equivalent expression:

$$x_{\text{target}} = \chi + \sum_{k=2}^{+\infty} f_k(f_1^p \chi, \ldots, f_1^p \chi). \tag{3.16}$$

Define the map $\mathcal{M} : \mathbb{R}^n \to \mathbb{R}^n$

$$\mathcal{M}(\chi) = x_{\text{target}} - \sum_{k=2}^{+\infty} f_k(f_1^p \chi, \ldots, f_1^p \chi),$$

and set up the iteration

$$\begin{aligned} \chi_1 &= x_{\text{target}} \\ \chi_{n+1} &= x_{\text{target}} - \sum_{k=2}^{+\infty} f_k(f_1^p \chi_n, \ldots, f_1^p \chi_n) = \mathcal{M}(\chi_n). \end{aligned}$$

We shall prove convergence of this iteration starting from any initial condition inside the set

$$S = \{\chi : \|\chi - x_{\text{target}}\|_\infty \leq \|x_{\text{target}}\|_\infty\}.$$

Theorem 3.4.10 *If*

$$\|x_{\text{target}}\|_\infty < \Lambda_1 = \frac{1}{2\|f_1^p\|_\infty D_1 D_2} \min\left\{\frac{1}{D_1\|f_1^p\|_\infty}, 1 - \frac{(D_1\|f_1^p\|_\infty)^2}{(1+D_1\|f_1^p\|_\infty)^2}\right\},$$

there exists a unique χ^* belonging to the set S and satisfying $\chi^* = \mathcal{M}(\chi^*)$. Furthermore, the unique solution can be computed by iterating the map $\mathcal{M}$ starting from any initial condition in S.

The proof to this theorem can be found in Appendix A.3. By this theorem, the set V_b in Theorem 3.4.2 contains a ball of radius Λ_1 about the origin.

Power series inversion algorithm

Next, we present an explicit inverse to the function. We borrow the result from [102, 52]. Consider the power series in equation (3.9)

$$x_{\text{target}} = f(y) = f_1 y + \sum_{k=2}^{+\infty} f_k(y, \ldots, y).$$

Let $m = n$, and assume that f_1 is invertible. Define the function $g : \mathbb{R}^n \mapsto \mathbb{R}^n$ via the power series

$$g(x) = g_1 x + \sum_{k=2}^{+\infty} g_k(x, \ldots, x), \tag{3.17}$$

where we let

$$g_1 = f_1^{-1}, \qquad g_k(x, \ldots, x) = -g_1 \sum_{m=2}^{k} \sum_{\substack{i_1+\cdots+i_l=k \\ i_1,\cdots,i_l<k}} f_l\Big(g_{i_1}(x, \cdots, x), \cdots, g_{i_l}(x, \cdots, x)\Big).$$

Theorem 3.4.11 *The function g is the inverse of f, and it converges provided*

$$\|x_{\text{target}}\|_\infty \leq \Lambda_2 = \frac{1}{4(D_1\|f_1^p\|_\infty + 1)\|f_1^p\|_\infty D_1 D_2} < \Lambda_1.$$

The proof to this theorem can be found in Appendix A.4. By this theorem, V_b in Theorem 3.4.2 contains a ball of radius Λ_2 about the origin.

3.5 Simulation

Two models were used to illustrate the algorithms. First, a one dimensional nonlinear system $\dot{x} = -x^2 + u$ was used to study the convergence properties of these algorithms. Second, a planar vertical takeoff and landing aircraft model was chosen to test the minimum-energy planning algorithm performance on a more complicated system.

3.5.1 PVTOL with Damping Example

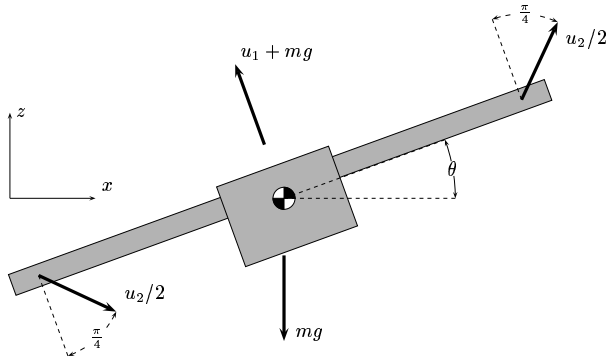

Figure 3.1: Diagram of the PVTOL model.

We consider the model of a simple planar vertical takeoff and landing aircraft model based upon that of [54, 90], but with added viscous damping forces; see Figure 3.1. In other words, we consider the classic PVTOL model subject to a linear drag force. We parameterize its configuration and velocity space via the state variables $(s, c, x, z, \omega, v_x, v_z)$. We let x and z be the inertial coordinates of the aircraft and s and c represent its roll angle θ such that $s = \sin\theta$ and $c = \cos\theta - 1$. The angular velocity is ω and the linear velocities in the body-fixed x (respectively z) axis are v_x (respectively v_z). Explicitly separating the linear from the homogeneous polynomial component, the equations are

written as

$$\begin{bmatrix} \dot{s} \\ \dot{c} \\ \dot{x} \\ \dot{z} \\ \dot{\omega} \\ \dot{v}_x \\ \dot{v}_z \end{bmatrix} = \begin{bmatrix} \omega \\ 0 \\ v_x \\ v_z \\ \frac{-k_1}{J}\omega \\ \frac{-k_2}{M}v_x - gs \\ \frac{-k_3}{M}v_z - gc \end{bmatrix} + \begin{bmatrix} c\omega \\ -s\omega \\ cv_x - sv_z \\ sv_x + cv_z \\ 0 \\ v_z\omega \\ -v_x\omega \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & \frac{h}{J}k_u \\ 0 & \frac{1}{M}k_u \\ \frac{1}{M}k_u & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}.$$

As stated in Section 3.2, the quadratic term can be represented via a symmetric tensor $f^{[2]}$. In components, let us write the ith component of $f^{[2]}(x,x)$ as $(f^{[2]})_i^{jk}x_jx_k$. All components of $f^{[2]}$ vanish except for $(f^{[2]})_i^{jk} = (f^{[2]})_i^{kj} = 1/2$ at indices $(1,2,5)$, $(3,2,6)$, $(4,1,6)$, $(4,2,7)$, $(6,5,7)$ and $(f^{[2]})_i^{jk} = (f^{[2]})_i^{kj} = -1/2$ at indices $(2,1,5)$, $(3,1,7)$, $(7,5,6)$. The control u_1 corresponds to the body vertical force minus gravity, while u_2 corresponds to coupled forces on the wingtips with a net horizontal component. The other forces depend upon the constants k_i, which parameterize some damping force, and g, the gravity constant. The constant h is the distance from the center of mass to the wingtip, while M and J are mass and moment of inertia, respectively. The constant k_u is a control gain.

Remark 3.5.1 *Although motion planning for the classic PVTOL example has been done effectively using flatness [90], the PVTOL with damping model appears not to be flat. A system with state x and control u is differentially flat if there exists an output function $\vartheta(x,u,\dot{u},\ddot{u},\dots)$ such that the states and controls can be expressed solely as a function of the output and its derivatives. The PVTOL equations with damping can be rewritten as follows:*

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \frac{M}{k_u}\begin{bmatrix} -\sin\theta & \cos\theta \\ \cos\theta & \sin\theta \end{bmatrix}\left(\begin{bmatrix} \ddot{x} \\ \ddot{z} \end{bmatrix} + \begin{bmatrix} 0 \\ g \end{bmatrix}\right. \tag{3.18}$$

$$\left. + \begin{bmatrix} \frac{k_2}{M}\cos^2\theta + \frac{k_3}{M}\sin^2\theta & (\frac{k_2}{M} - \frac{k_3}{M})\sin\theta\cos\theta \\ (\frac{k_2}{M} - \frac{k_3}{M})\sin\theta\cos\theta & \frac{k_2}{M}\sin^2\theta + \frac{k_3}{M}\cos^2\theta \end{bmatrix}\begin{bmatrix} \dot{x} \\ \dot{z} \end{bmatrix}\right) \tag{3.19}$$

$$u_2 = \frac{J}{k_u h}\left(\ddot{\theta} + \frac{k_1}{J}\dot{\theta}\right).$$

Equating the second and third equations, we obtain

$$\frac{J}{h}\left(\ddot{\theta}+\frac{k_1}{J}\dot{\theta}\right)-M\left(\ddot{x}\cos\theta+\ddot{z}\sin\theta+\frac{k_2}{M}\left(\dot{x}\cos\theta+\dot{z}\sin\theta\right)+g\sin\theta\right)=0. \qquad (3.20)$$

For the classical PVTOL (when the damping coefficients are zero), the flat output is

$$\vartheta=\begin{bmatrix}x\\z\end{bmatrix}+\frac{J}{hM}\begin{bmatrix}-\sin\theta\\\cos\theta\end{bmatrix}.$$

This is also known as the Huygens center of oscillation. *Inserting the flat output into* (3.20)*, the angle θ is found [96, 90] to be related to the flat output via $\ddot{\vartheta}_1\cos\theta+(\ddot{\vartheta}_2+g)\sin\theta=0$. Once θ is derived, the states and controls can then be calculated by using the output relation and equations of motion, respectively. However, when the damping coefficients are nonzero,*

$$(\ddot{\vartheta}_1+\frac{k_2}{M}\dot{\vartheta}_1)\cos\theta+(\ddot{\vartheta}_2+\frac{k_2}{M}\dot{\vartheta}_2+g)\sin\theta=\frac{J}{hM}\left(\frac{k_1}{J}+\frac{k_2}{M}\right)\dot{\theta}.$$

Thus, θ can no longer be written in terms of ϑ and its derivatives, so that the classical PVTOL flat output is no longer a flat output of the system with damping. It is unclear whether a flat output still exists.

3.5.2 Implementation

The two algorithms were divided into two implementation steps: preprocessing and control derivation. Preprocessing includes the system definition and the calculation of the corresponding tensors in the series expansion. The resulting expansion can be saved to memory for use by the control derivation. The control derivation includes solution of the inverse problem using the contraction method and the calculation of the controls with respect to that solution. The contraction method was chosen because it both has a larger lower bound on its radius of convergence as well as a more straightforward implementation. The simulation was carried out by numerical solution of the ordinary differential equations. Each of these tasks was implemented in Maple 5.4, primarily due to the nontrivial nature of the calculation of the required tensors. As this involves computation of

a series of tensors of increasing dimension, each defined by lower order tensors, it necessitates a data type with expandable structure. This is not straightforward in programming languages such as C, nor in numerical software such as Matlab. Another disadvantage of Matlab is that its tensor manipulation routines are not as comprehensive as its matrix routines, thus requiring nested loops to carry out tensor calculations. While Maple is less computationally efficient than either of the aforementioned methods (documentation [30] suggests floating point computations in Maple can be 50 to 500 times slower than in equivalent Fortran programs), its tensor package accommodates tensor products as well as calculation of the tensors using the recursive functions, avoiding data structure issues.[5] Another computational challenge was posed by the PVTOL system itself. Its controllability Grammian is ill-conditioned (using the parameters described below, its condition number [55, page 56] is in the order of $1e+5$), thus requiring careful treatment and high accuracy. Fortunately, these issues take place in the preprocessing stage and can be tackled offline. These tensor calculations dominate the preprocessing and require, at most, $O(n_{tot}^{k+3})$ multiplications and integration of $O(n_{tot}^{2k+4})$ terms, where n_{tot} and k are the total dimension[6] of the system and the order of the series expansion, respectively (assuming $n \geq k > 1$). The integration then proves to be the primary factor in run time. The control derivation is far less computationally intensive, as it involves primarily floating point computations. Yet, because the number of recursions needed to find a solution for a given accuracy is variable, the number of online calculations is less predictable. This, too, was implemented in Maple, although any programming language would work as well. For the PVTOL example, using a second order series approximation, on an 800 Mhz Windows ME PC using 128 megabytes of RAM, the algorithm took 98.5 seconds in preprocessing and 2.8 seconds (7 iterations) in solving for the control. Third order series calculations took 13,173 seconds in preprocessing and 20.9 seconds (23 iterations) in solving for the control online, corresponding to the computational estimate above. All of the necessary series data stored for the control derivation stage amounted to 27 and 168 kilobytes for the second and third order expansions, respectively.

[5] An implementation in Mathematica was found to encounter similar features as in Maple.

[6] For the base function algorithm, $n_{tot} = n$, while $n_{tot} = 2n$ for the minimum energy algorithm

3.5.3 Results

Convergence Study

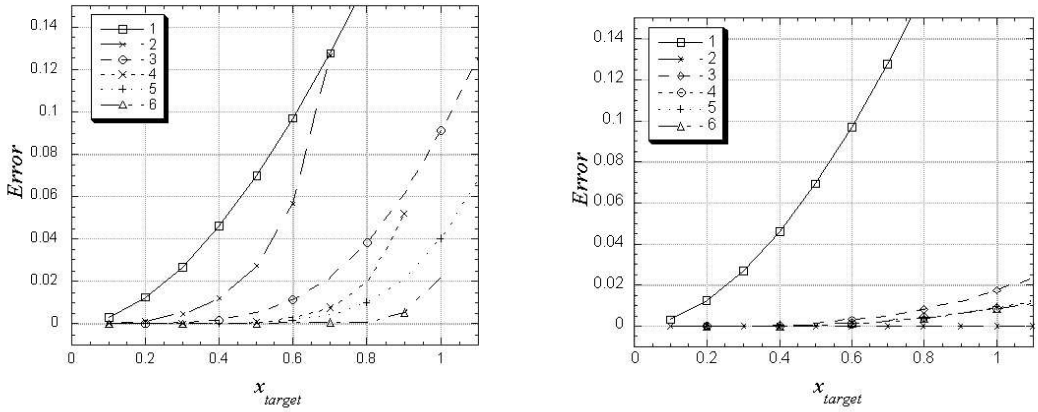

Figure 3.2: Final position error for motion planning algorithm (left), minimum energy planning algorithm (right).

The one dimensional system was used to show the solution convergence properties of both algorithms. For the motion planning algorithm, the inverse problem simplifies to the following root finding problem and control definition:

$$x_{\text{target}} = p - \frac{1}{3}p^2 + \frac{2}{15}p^3 - \frac{16}{315}p^4 + \frac{58}{2835}p^5 - \frac{1262}{155925}p^6 + \ldots, \quad u = p.$$

The lower bound of the neighborhood of convergence of the algorithm, as defined in Theorem 3.4.10, is $\|x_{\text{target}}\|_\infty < .0625$. As the control is a constant, the solution to the differential equation, for a positive coefficient p, can be written as $x = \sqrt{p}\tanh(\sqrt{p}t)$. Truncating the series at orders one through six, the corresponding coefficients and controls were found. Figure 3.2 shows the comparative error among the levels of truncation for a range of x_{target}. This shows a general decrease in error as the order of the truncation increases. The x_{target} at which the even truncated series cease to have a solution corresponds to the maximums of the truncated polynomials. It can therefore be seen that

the actual convergence radius of the algorithm is orders of magnitude greater than the minimum described in Theorem 3.4.10.

For the minimum energy planning algorithm, the inverse problem simplifies to the following root finding problem and control definition:

$$x_{\text{target}} = -\lambda_0 + 0\,{\lambda_0}^2 - \frac{1}{10}\,{\lambda_0}^3 + 0\,{\lambda_0}^4 - \frac{1}{180}\,{\lambda_0}^5 + 0\,{\lambda_0}^6 + \dots$$
$$u = -\lambda_0 + t^2{\lambda_0}^2 - \frac{1}{2}\,t^4{\lambda_0}^3 + \frac{1}{10}\,t^6{\lambda_0}^4 - \frac{1}{20}\,t^8{\lambda_0}^5 + \frac{7}{450}\,t^{10}{\lambda_0}^6 + \dots\,.$$

The lower bound of the neighborhood of convergence of the algorithm, as defined in Theorem 3.4.10, is $\|x_{\text{target}}\|_\infty < .0023$. The control input is computed via a series expansion on the initial value of the costate λ_0. Figure 3.2 shows the comparative error among the levels of truncation for a range of x_{target}. This shows a general decrease in error as the order of the truncation increases, although this is not true uniformly. This is not unexpected, as the error reflects the accuracy of the solution of x only, ignoring λ. For example, the second order expansion solves the differential equation and constraints on x exactly, but does not solve as accurately for λ. Thus, a feasible trajectory is generated that is suboptimal. Despite this apparent non-uniformity, the algorithm behaves very well at x_{target}, orders of magnitude beyond the conservative minimum provided by Theorem 3.4.10.

Figure 3.3 provides a cost comparison between the two techniques for series truncated at order 6. Understandably, as the target distance increases, the control is active longer and the cost differential is more apparent, with a difference of 18% of the optimal cost at $x_{\text{target}} = 1$.

Figure 3.4 compares the state histories of the linear case as well as the motion and minimum energy planning algorithms of order six. As in the previous figures, the optimal algorithm consistently reaches x_{target} with greater accuracy. Both methods significantly outperform their linear counterpart.

PVTOL

The minimum energy planning algorithm, having showed good performance for the one dimensional case, was applied to that of the PVTOL. For this case, the aforementioned

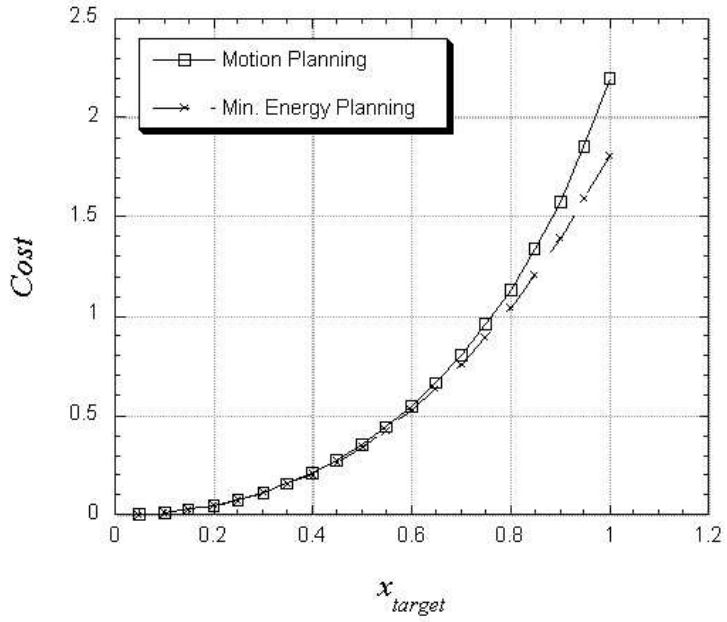

Figure 3.3: Cost comparison between motion planning with base functions and minimum energy planning algorithms for series truncated at order 6.

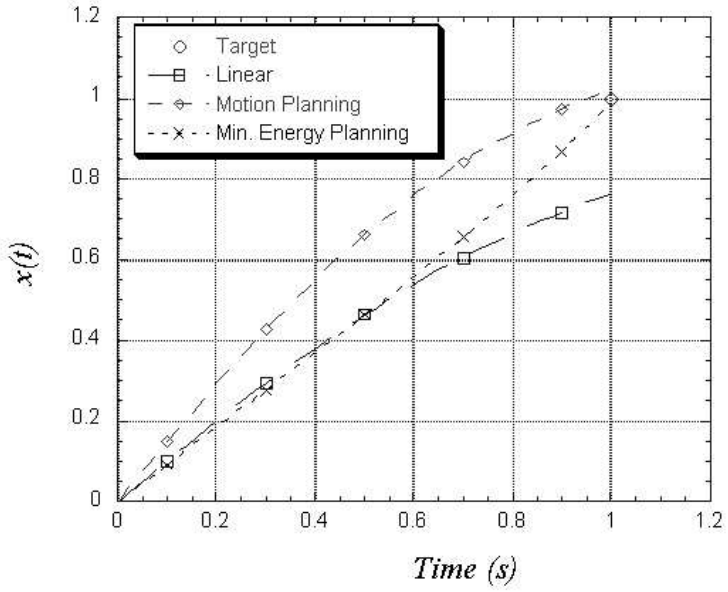

Figure 3.4: Trajectory comparison of the motion planning with base functions and minimum energy planning algorithms for series truncated at order 6 with their linear counterpart.

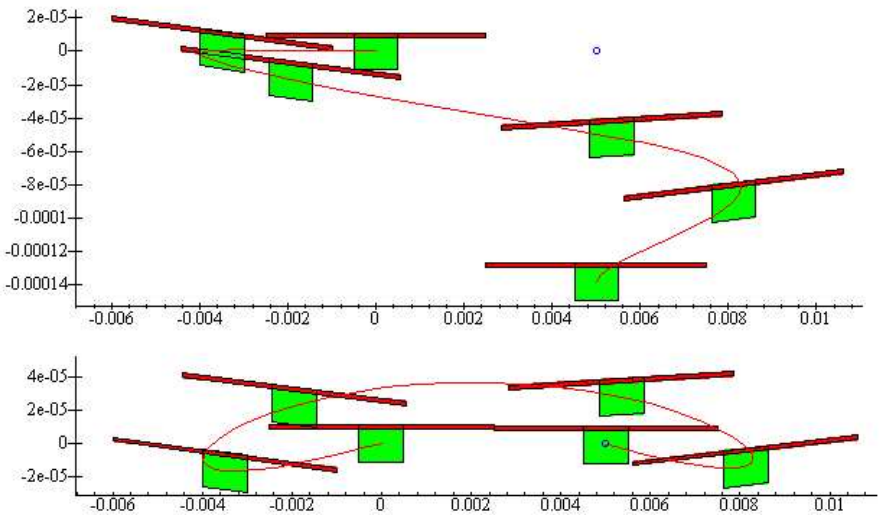

Figure 3.5: Resulting motions from minimum energy planning algorithm for PVTOL series truncated at orders 1 (top) and 2 (bottom), starting at rest at the origin with desired final condition of a small negative velocity at the circled location.

model was chosen with the constants chosen such that k_u/M, hk_u/J, k_1/J, k_2/M, and k_3/M are all normalized to 1, $g = 10$, and $T = 1$. Note that these numbers follow the convention of meters-kilograms-seconds. As defined in Theorem 3.4.10, the lower bound of the neighborhood of convergence of the algorithm is $\|x_{\text{target}}\|_\infty < 1.6 \times 10^{-13}$. As with the one dimensional case, this was over-conservative, as solutions could be found over 10^{10} times greater than the bound. One such example is shown in Figure 3.5, where a positive x displacement of 0.005 was requested with an x component of velocity of -0.0005. Using the series truncated at second order shows a clear improvement over the linear solution, as the error in the final second order state is negligible in comparison to the 3 percent error in the final position of the linear case.

Remark 3.5.2 *Preliminary numerical investigations shows that the convergence radius has bounds near .01 meters, which, although much larger than the lower bound analytical estimate, is still quite small (even over a one second time interval). This small value appears to be due in part to the ill-conditioning of the controllability Grammian, as the Grammian must be inverted to find the solution. This small*

convergence radius makes the PVTOL an unattractive application for practical implementation of this algorithm. It should be noted, however, that the convergence radius varies with choice of the unit systems and corresponding scaling of the constants in such a way that the effective region of convergence (when mapped back to the original units) also changes. Thus, it may be possible to find unit transformation that significantly increases the effective convergence region of the PVTOL. Also note that this convergence limitation on the PVTOL is not necessarily typical of other systems, as the next chapter will demonstrate that the effective convergence region of orbital motions is many kilometers in size.

Remark 3.5.3 *The assumptions on the mass and dimensional constants correspond to estimates of these values for the British harrier FRS-1 jet. As for the damping constants, these have been artificially inflated to increase the non-flat effect. In practice, these numbers would be negligible, but the region of convergence is relatively invariant to this difference.*

3.6 Conclusions

We have presented a variety of constructive controllability and minimum energy control algorithms. The results are local in nature but constructive: existence, uniqueness and optimality are guaranteed for a class of polynomial systems. Lower bounds on the region of validity of these algorithms are presented, and evaluated with respect to algorithm performance in specific examples.

Chapter 4

Maneuver Automaton

While the behavior of a modern mechanical system is complicated with many degrees of freedom, the full range of motions available is often not necessary for control. In fact, given a finite subset of motions having certain symmetry properties and the ability to transition between them, the system's motion can be controlled from one point or trajectory to another easily. Reducing the full class of motions available to the mechanical system to a finite subset reduces the dimension of the problem while also enabling it to be converted to a hybrid system. This provides a means of decreasing computation time and complexity. This approach was pioneered by Frazzoli [41], who used trim trajectories of aircraft connected by predetermined maneuvers on a symmetry group of $\mathbb{R}^3 \times S^1$, or 3-D translation and yaw rotation. While this symmetry is located on the configuration manifold, other symmetries also exist and provide other options for trajectory planning. It is this framework that the development of this chapter is based.

This chapter addresses the problem of global motion planning in obstacle free environment for a nonlinear time-invariant dynamical system. Rather than attack the nonlinear high-dimensional system outright, evolution on the full dynamical system is limited to a set of *trajectory primitives*, as defined in Chapter 2. The reduced dynamical system of all possible interconnections of trajectory primitives is called a *maneuver automaton* and it forms the basis for rapid obstacle-free motion planning. The concept of *maneuver automaton* was first introduced by Frazzoli [41] where the trajectory primitives used are relative equilibria or "trim trajectories" and precomputed transitions in between them. Much of the strength of this approach comes from the existence of symmetries in the

dynamics of the system that allows a single trajectory primitive to represent numerous vehicle motions. The maneuver automaton defined in this chapter expands this structure to include

(i) a larger set of primitives than the aforementioned relative equilibria,

(ii) transitions with an infinite set of possible final conditions,

(iii) incorporation of time-dependent final conditions, and

(iv) use of scaling symmetries.

as well as an online local planner such as in Chapter 3 which provides

(i) Overall solution convergence guarantees.

(ii) Controllability with fewer trajectory primitives.

(iii) Feasible solutions with a coarser grid.

This chapter begins with an explanation of the structure of the maneuver automaton and background material on continuous and discrete systems before discussing the properties of the maneuver automaton itself. From this, the basis for optimal control of the maneuver automaton is developed. Dynamic programming is then presented as a solution method and convergence criteria with proofs are given. The rest of the chapter is devoted to a brief algorithm description and an Earth to Mars orbital transfer example to which this methodology is applied.

4.1 Preliminaries

4.1.1 Graphs and Automata

The development in this chapter has a strong basis in computer science and data structures. A key concept is that of the automaton, defined as follows [74]:

Definition 4.1.1 (Automaton) *An* automaton *is an abstract computational model which consists of a set of states, a set of input symbols (an alphabet), stored start states, and state transition that maps input symbols and the current state to the next state.*

An automaton with a finite number of states is known as a finite state machine or finite state automaton [15]. Well known variants of these include Turing machines and Moore machines. The automaton itself can be represented as a *directed graph*[1] [37].

Definition 4.1.2 (Directed Graph) *A* directed graph *is defined as a set of vertices, a set of edges, and a mapping that maps every edge onto an ordered pair or vertices.*

Note that, when represented as a graph, the vertices correspond to the states and the edges correspond to the inputs. The sequence of input symbols used to traverse the automaton graph is called a *word*.

4.1.2 Trajectory Primitives

The trajectory primitive, as defined in Chapter 2, provides a way of describing a set of motions as a one equivalent motion, where equivalence is dependent upon the symmetry properties of the system. As any motion satisfying the differential equations could be described as belonging to a trajectory primitive, describing the dynamics of a system with a full set of primitives (where any evolution on the system could be described as a sequence of primitives) does nothing to reduce the generality of the method. Furthermore, vehicles (and other systems) tend to be controlled primarily in a subset of these primitives for various reasons such as simplicity and/or efficiency. Thus, a much smaller set of primitives can be used to represent most motions of a system. Classes of primitives that have been used to represent common vehicle motions include the quantized-input results of Marigo [88, 13, 14], sinusoids [97], relative equilibria [41], and decoupling vector fields [22]. Another such class is the *reference trajectory*, defined in Chapter 2 as

Definition 4.1.3 (Reference Trajectory) *A trajectory satisfying the equations of motion of the system Σ for which*

[1] A directed graph is also known as an oriented graph, although the latter term is also sometimes used to denote a subset of directed graphs for which there is, at most, one edge between vertices.

(i) The state flow ($\phi_{u_{ref}}(\xi_0, t)$) is known and is unique for a given initial state $\xi_0 \in \Xi$.

(ii) The control $u_{ref}(t)$ is a continuous function of time only

(iii) The dynamical system Σ can be approximated as a perturbation about the trajectory primitive with the form

$$\begin{aligned} \dot{x} &= A(t)x + f^{[2]}(x, x) + Bu \\ x(0) &= 0, \end{aligned} \tag{4.1}$$

where $x(t)$ and $A(t)$ are defined as continuous functions on the interval I, and the $\mathcal{L}_1$-norm of the corresponding linear state transition matrix $\|\Psi(t, 0)\|_{\mathcal{L}_1}$ is bounded [2].

A reference trajectory on Σ can then be identified by the sextuple $\{\phi_{u_{ref}}, u_{ref}, A, B, f^{[2]}, I\}$.

4.2 Maneuver Automaton Framework

For our application, we are concerned with two classes of trajectory primitives, namely

(i) Reference trajectories

(ii) Maneuvers

The framework under consideration would include a finite number of reference trajectories connected by a finite number of maneuvers, creating a *maneuver library.*

This system is thus a *hybrid system*, where the discrete dynamics of the finite reference trajectories and their maneuvers connecting them correspond to a directed graph. The continuous component of this structure is the time-dependent evolution of the continuous states along the reference trajectory. This framework is called a *maneuver*

[2] Here, $\|\Psi(t, 0)\|_{\mathcal{L}_1} = \max_{i=1,\ldots,n} \sum_{j=1}^{n} \int_I |(\Psi(t, 0)_{ij}| dt$. This statement implies that the system $\dot{x} = A(t)x$ is stable in the sense of Lyapunov (for the time-invariant system, it is sufficient the matrix A is Hurwitz) or that the interval I over which $A(t)$ is defined is finite and the escape time of $\dot{x} = A(t)x$ (if it exists) is not in the interval.

automaton. This maneuver automaton can be described as a finite state machine for which words can be mapped to motion on the state space of the system [43] or as an extension of the finite state machine to include a continuous state [41, 42], analogous to the hybrid automata of literature [58, 59]. The latter is given below.

Definition 4.2.1 (Maneuver Automaton) *A maneuver automaton MA over a mechanical control system Σ, with symmetry group H, is described by the following objects:*

(i) Hybrid state: *Let γ be the state of the system on the hybrid state, or maneuver, space of the system $\mathbb{H}$. This maneuver space is divided into discrete ($\mathbb{H}_d$) and continuous ($\mathbb{H}_c$) subspaces where $\gamma = (\gamma^d, \gamma^c) \in \mathbb{H}_d \times \mathbb{H}_c = \mathbb{H}$.*

(ii) Input alphabet: *Let ν be an input symbol (or control input) in the alphabet (or control space) $\mathbb{U}(\gamma)$.*

(iii) Initial states: *Let γ_0 be the initial state of the system,*

(iv) State transition map: *Let $\varphi : \mathbb{H} \times \mathbb{U}(\gamma) \to \mathbb{H}$ map the transition from one state to the next.*

Hybrid State

As the hybrid system is composed of discrete and continuous parts, so too is the hybrid state γ.

(i) A discrete state $\ell = \gamma^d \in \{1, \dots N_T\} = \mathbb{H}_d \subset \mathbb{N}$, indexing the N_T reference trajectories $\{{}^{\ell}\phi_{u_{ref}}, {}^{\ell}u_{ref}, {}^{\ell}A, {}^{\ell}B, {}^{\ell}f^{[2]}, {}^{\ell}I\}$ and a base state ${}^{\ell}\xi_b$

(ii) A continuous state $h \in H$, defined as a position on the symmetry group;

For problems in which time plays a part in the definition of the terminal condition of the system, it is beneficial to add the following state

(iii) The time state $t \in \{z \in \mathbb{R} | z \geq 0\} = \mathcal{I}_0 \subset \mathbb{R}$, evolving with $\dot{t} = 1$ along each trajectory and jumping with each maneuver.

Thus, the continuous subspace of the maneuver space $\mathbb{H}_c$ is defined as either H or $H \times \mathcal{I}_0$.

Input Alphabet

The input symbols, or control variables, are defined as $\nu = (\tau, \ell_{next}, \Delta h)$, where

(i) τ_k is the *coasting time* on the reference trajectory, or time on the reference trajectory before a maneuver is initiated, defined in the closed and bounded interval $\mathcal{I}_\tau(\gamma) \subset \mathbb{R}$. Note that $\mathcal{I}_\tau(\gamma)$ assumed to be Lipschitz continuous (with respect to γ^c) in a Hausdorff metric sense [38], i.e.,

$$\mathcal{I}_\tau(\gamma^d, \gamma^c) \subset \mathcal{I}_\tau(\gamma^d, \gamma^{c'}) + \left\{ \tau \middle| \|\tau\| \leq \beta \|\gamma^c - \gamma^{c'}\| \forall \gamma^c, \gamma^{c'} \in \mathcal{H} \right\},$$

where β is a positive constant and $\mathcal{H}$ is a compact subset of $\mathbb{H}_c$.,

(ii) ℓ_{next} is the next reference trajectory index, defined in $\mathbb{N}_\delta(\gamma) \subseteq \mathbb{H}_d$, and

(iii) Δh is the starting state of reference trajectory ℓ_{next} relative to the original, defined in the set $\mathcal{U}(\gamma)$. Note that $\mathcal{U}(\gamma)$ may be characterized as a) a finite set of inputs as in [41] [3], or b) an infinite compact set on Euclidean space for which $\mathcal{U}(\gamma^d, .) : \gamma^c \mapsto \mathcal{U}(\gamma^d, \gamma^c)$ is Lipschitz continuous in a Hausdorff metric sense [38].

Thus, the input alphabet, or control space, of the maneuver automaton is defined as $\mathbb{U}(\gamma) = \mathbb{N}_\delta(\gamma) \times \mathcal{U}(\gamma) \times \mathcal{I}_\tau(\gamma)$.

Initial States

The initial state of the system γ_0 can be represented, in the nominal case, by the couple (ℓ_0, h_0), and, in the time-varying final condition case, by the triple (ℓ_0, h_0, t_0).

State Transition Map

Evolution of the hybrid state can then be broken down into that due to "coasting" on a reference trajectory and that due to maneuvers.

We can then indicate this evolution using the state transition map $\varphi : \mathbb{H} \times \mathbb{U} \to \mathbb{H}$, with the notation $\varphi(\gamma, \nu) = \varphi_\nu(\gamma)$. The stepwise evolution of the maneuver automaton thus

[3] Frazzoli used the controls $\nu = (\tau, \text{Maneuver}_{\text{next}})$, for which $\text{Maneuver}_{\text{next}}$ was in a finite set of precomputed maneuvers, each identifiable by the original trajectory ℓ and the pair $\ell_{next}, \Delta h$

follows $\gamma_{k+1} = \varphi(\gamma_k, \nu_k)$ and can be described by the following discrete time transition equations:

$$\begin{aligned} \ell_{k+1} &= \ell_{next} \\ h_{k+1} &= h_{next} = \phi^H_{u^{\text{ref}}}(h_k, \tau_k + T_{\ell_k}) \cdot \Delta h_k \\ \{t_{k+1} &= (\tau_k + T_{\ell_k}(\Delta h_k))/\eta(h_k)\} \end{aligned} \tag{4.2}$$

where $\phi^H_{u^{\text{ref}}}$ is the projection of $\phi_{u^{\text{ref}}}$ onto H, $\eta(h)$ is the time scaling factor as defined in Chapter 2 and T_ℓ is the maneuver time corresponding to reference trajectory ℓ.

Because of the assumptions on the controls, the system (4.2) can be described as *stationary*. This means that the control policy and the state evolution equations only have state dependence and are invariant to the discrete "time".

The evolution on the maneuver automaton from initial to final states is then composed of motion along reference trajectories and the maneuvers connecting them and can be mathematically described as:

$$\gamma_{n_T} = \varphi_\mu(\gamma_0), \quad \varphi_\mu = \varphi_{\nu_{n_T}} \circ \varphi_{\nu_{n_T-1}} \circ \cdots \circ \varphi_{\nu_1} \tag{4.3}$$

where $\gamma_{k+1} = \varphi_{\nu_k}(\gamma_k)$ is defined as in (4.2), μ is the automaton word, i.e., the sequence of control inputs $\{\nu_{n_T}, \nu_{n_T-1}, \ldots \nu_1\}$, and n_T is the total number of discrete time steps in the final motion.

As seen in Figure 4.1, the projection of motion along the automaton corresponds to movement on a directed graph when projected upon the discrete space $\mathbb{H}_d$.

Mapping from MA to Σ

While on a reference trajectory, the state of the full system on Ξ is completely determined by the *hybrid state* γ and the coasting time τ. Let $\mathcal{F} : \mathbb{H} \times \mathcal{I}_\tau(\gamma) \to \Xi$ then be this mapping of the hybrid state to the space of the full system, where $(\gamma, \tau) = (\ell, h, \{t\}, \tau) \mapsto \psi_h({}^\ell\phi_{u^{\text{ref}}}(\xi_b, \tau/\eta(h)))$.

Let $\mathcal{F}_\delta : \mathbb{H} \times \mathbb{U}(\gamma) \to \Xi$ then map the hybrid state and controls to the perturbed state about the reference trajectory. When the maneuver is defined using a planner such

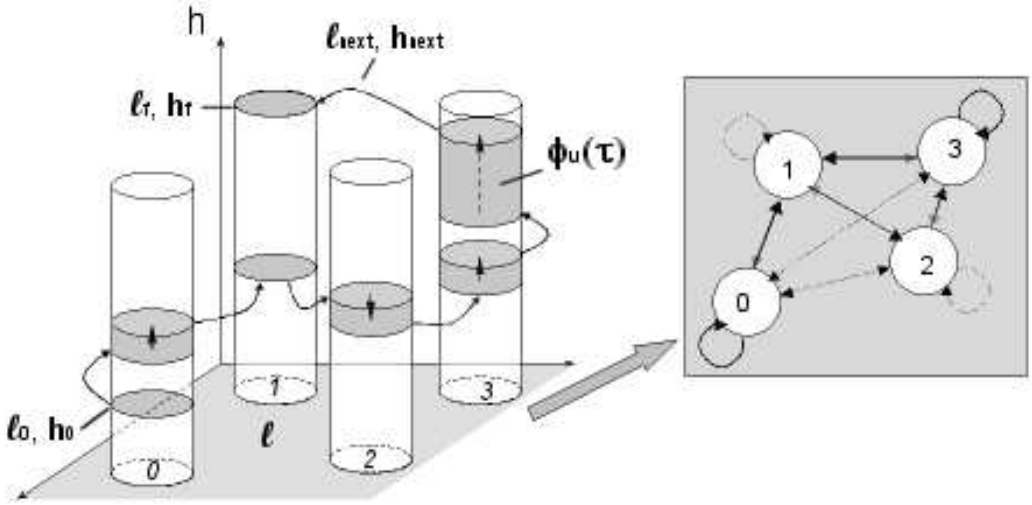

Figure 4.1: The Maneuver Automaton: On left, evolution in the discrete space is in the horizontal plane and evolution along the symmetry group is shown on the vertical axis. At right is a projection of the evolution onto the discrete space.

as in Chapter 3, transitions on the maneuver automaton occur between two reference trajectories and positions on the symmetry group that are local to one another, i.e., transitions whose relative difference when mapped to Ξ fall within the open set of points for which the planner is valid $\mathcal{F}_\delta(\gamma, \nu) \in B_\delta(0) \quad \forall (\gamma, \nu) \in \mathbb{H} \times \mathbb{U}$. Here, $B_\delta(0) \subset \Xi$ is a ball of radius δ about the origin. Note that using this type of planner also allows $\mathcal{U}(\gamma)$ to be defined such that its interior is nonempty. This type of maneuver is represented in Figure 4.2.

4.3 Well-posedness and Controllability

In addressing motion planning on the Maneuver Automaton, we rely on two key dynamical system properties, *well-posedness* and *controllability*. A system is said to be *well-posed* [127] if trajectories both exist and are unique for any given control and initial conditions. A system defined on a space S is *controllable from an initial state* $s_0 \in S$ *and time* $t_0 \in \mathbb{R}$ *to a final state* $s_f \in S$ [119] if there exists an input such that the evo-

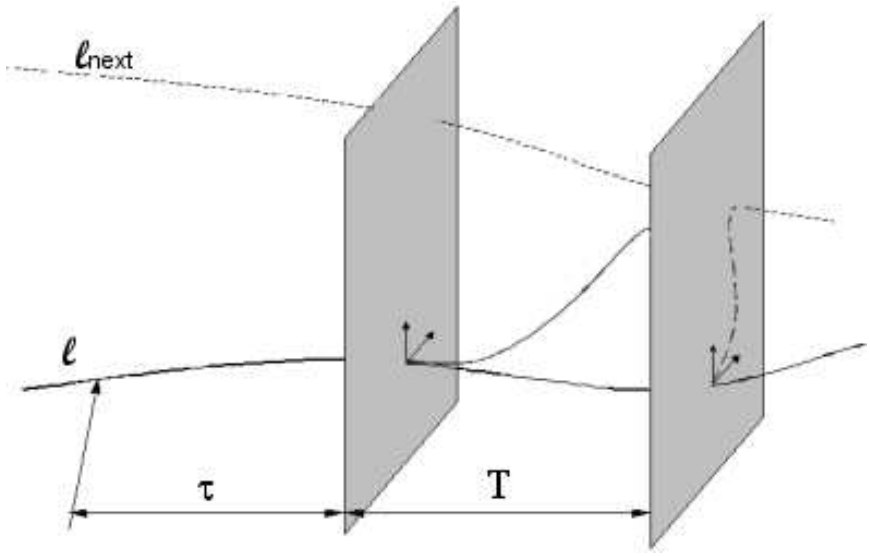

Figure 4.2: Local Planner-based Maneuver

lution of the system following the input from the initial condition results in the system having the final state at some time t_{final}. Alternatively, it can be said that s_f is *reachable* from s_0 at t_0. A system defined on a space S is *(completely) controllable*, if, for all $s_0, s_f \in S$ and $t_0 \in \mathbb{R}$, the system is controllable from s_0 at t_0 to s_f.

As the Maneuver Automaton is a combination of discrete and continuous systems, we will examine the properties of each before returning to the hybrid system.

4.3.1 Discrete System Properties

As previously noted, the projection of the automaton on the discrete space forms a directed graph.

Here, the vertices correspond to motion along the reference trajectories and the edges correspond to maneuvers among them. The concept of controllability is closely related to the concept of accessibility. For a directed graph, a vertex b is *accessible* from vertex a if there is a directed path (a series of contiguous consistently directed edges) from a to b. Another important concept is that of connectedness. A directed graph is *strongly connected* if there exists a directed path between every pair of vertices in the graph. It

is *weakly connected* if a directed path does not exist, but an undirected path (a series of contiguous but not necessarily consistently directed edges) does. By definition, a directed graph is strongly connected if and only if every vertex in the graph is accessible from every other vertex. Also by definition, a discrete graph is controllable if and only if it is strongly connected.

4.3.2 Continuous System Properties

Before we address the properties of continuous systems,i.e., systems for which the state variables are defined on an open set, several key mathematical concepts must be defined. For more detail one should refer to [119]. A subset S of $\mathbb{R}^n$ is said to have *zero measure* if, for each $\epsilon > 0$, there exists a countable union of balls $(B_1, B_2, \dots)$ of volume ϵ_i such that $\sum_{i=1}^{\infty} \epsilon_i < \epsilon$ and $S \subseteq \cup_{i=1}^{\infty} B_i$. A property is said to *hold almost everywhere* if it fails only in a set of zero measure. A function mapping an interval $\mathcal{I}$ to a metric space $\mathcal{U}$ is *piecewise constant* if it is constant over each element of a finite partition of the interval. A function $f : \mathcal{X} \subseteq \mathbb{R}^n \to \mathcal{U}$ is *measurable* if there exists some sequence of piecewise constant functions $f_i, i \in \mathbb{N}$ such that f_i converges to f almost everywhere. The function f is then *locally integrable* if the integral $\int_a^b \|f(\tau)\| d\tau < \infty$ for each $a < b$ where $a, b \in \mathcal{I}$. The function f is *locally Lipschitz* if, there are, for each $x_0 \in \mathcal{X}$, a real number $\rho > 0$ and a locally integrable function $\alpha : \mathcal{I} \to \mathbb{R}^+$ such that a ball $(B_\rho(x^0))$ of radius ρ centered at x_0 is contained in $\mathcal{X}$ and $\|f(t,x) - f(t,y)\| \leq \alpha(t)\|x - y\|$ for all $t \in \mathcal{I}$ and $x, y \in B_\rho(x^0)$. The *well-posedness* of a continuous system is equivalent to the existence and uniqueness of a solution to the initial value problem of the system equations of motion. For continuous-time systems, these correspond to ordinary differential equations, for which the following theorem holds [119]:

Theorem 4.3.1 (Existence and Uniqueness of solutions to ODEs) *Given a system $\dot{x} = f(t,x)$, where $x \in \mathcal{X} \subseteq \mathbb{R}^n$, $t \in \mathcal{I} \subseteq \mathbb{R}$ and $\mathcal{X}$ is open, for every $(t_0, x_0) \in \mathcal{I} \times \mathcal{X}$, there exists a unique solution to the initial value problem if f is measurable and locally integrable on $t \in \mathcal{I}$ as well as continuous and locally Lipschitz on $x \in \mathcal{X}$.*

For the rest of this section, unless otherwise noted, we will assume "system" implies a well-posed system.

For nonlinear continuous-time systems, *accessibility* [119] is the property that, from any given state $x \in \mathcal{X}$, the reachable set $\mathcal{R}_{\mathcal{X}}(x)$ is of full dimension, i.e., non-empty. This is sometimes also called *weak controllability*. A system is *(small time) locally accessible* from x if, for every $t_{\text{final}} > 0$, the reachable set is non-empty. Discrete time systems are called *forward accessible* if the reachable set from a given state is non-empty. [62]

When considering nonlinear systems, complete controllability is difficult to establish. Thus, one must define the concept of *local controllability*.

Definition 4.3.2 (Local Controllability) *Given a (well-posed) system $\dot{x} = f(t, x)$ or $x(t_{k+1}) = f(t_k, x)$ with $x \in \mathcal{X}$ and letting $x^r(t)$ be any path on the interval $[t_0, t_f]$, the system is* locally controllable *about $x^r(t)$ if, $\forall \epsilon > 0$, there exists a $\delta > 0$ such that for every $x_0, x_f \in \mathcal{X}$ with $d(x_0, x^r(t_0)) < \delta$, $d(x_f, x^r(t_f)) < \delta$, there is a path $x^*(t)$ with $t \in [t_0, t_f]$ for which $x^*(t_0) = x_0$, $x^*(t_f) = x_f$ and $d(x^*(t), x^r(t)) < \epsilon$ for all $t \in [t_0, t_f]$.*

Time Invariant Systems

Let us now address time invariant systems of the form:

$$\dot{q} = X(q) + \sum_{j=1}^{m} Y_j(q) u_j, \tag{4.4}$$

where q is in the n-dimensional manifold Q, u is bounded and measurable, and $q(0) = q_0$ is an equilibrium point, i.e., $X(q_0) = 0$. While controllability and accessibility are difficult to quantify for general nonlinear systems, tests for certain controllability and accessibility properties have been developed for systems of this form. First, however, let us define some necessary mathematical constructs (A more rigorous definition is given in [2, 119]):

Lie derivative:
Let φ_f be an arbitrary function mapping Q to a Banach space and let $d\varphi_f(q)$ be the differential of φ_f at point $q \in Q$. Furthermore, let X be an arbitrary vector field on Q. The *directional* or *Lie derivative* of φ_f along X is

$$L_X \cdot \varphi_f(q) = X[\varphi_f](q) = d\varphi_f(q) \cdot X(q). \tag{4.5}$$

Lie bracket:

Let Y also be an arbitrary vector field on Q. The *Lie derivative* of Y with respect to X, also known as the *Lie bracket* of X and Y, written as $L_X Y = [X, Y] = ad_X Y$ is the unique vector field satisfying the relation

$$L_{[X,Y]} = [L_X, L_Y] \tag{4.6}$$

This quantity is also uniquely defined by the relation

$$[X, Y][\varphi_f] = X[Y[\varphi_f]] - X[Y[\varphi_f]]. \tag{4.7}$$

Geometrically, the Lie bracket's meaning can be expressed in terms of the flow of the system along vector fields X and Y. Let ϕ_X^t and ϕ_Y^s be the flows along vector fields X and Y for (small) times t and s, respectively. Furthermore, as in Figure 4.3, let us define $qf = \phi_{-Y}^s \circ \phi_{-X}^t \circ \phi_Y^s \circ \phi_X^t(q0)$, where the resulting displacement $\phi_{-Y}^s \circ \phi_{-X}^t \circ \phi_Y^s \circ \phi_X^t$ is approximately $s \cdot t \cdot [X, Y]$. It can be shown that, if $[X, Y] = 0$, the flows commute. This property is important because non-commutative vector fields can allow for motion independent of the basis formed by the vector fields themselves. Such properties impact the dimension of the accessible region of a system.

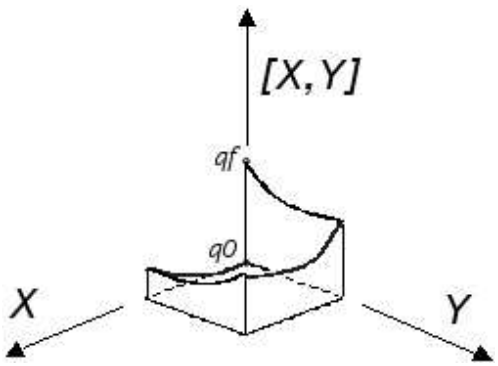

Figure 4.3: Lie bracket

Distribution:

A *distribution* on the open subset $\mathcal{O} \subseteq Q$ is a map $\mathcal{D}$ which assigns, to each $q \in \mathcal{O}$, a subspace $\mathcal{D}(q) \subseteq TQ$.

The following development is derived from Kawski [66]. Define the *involutive closure* $\overline{\text{Lie}}(\{Y_1 \dots Y_m\})$ as the distribution generated by all iterated Lie brackets of the vector fields $\{Y_1 \dots Y_m\}$. A necessary and sufficient condition for local accessibility can then be stated as:

Theorem 4.3.3 (Chow) *The system* (4.4) *is* locally accessible *from* q_0 *if and only if it satisfies the* Lie Algebra Rank Condition *(LARC):*

$$rank(\overline{Lie}(\{X, Y_1 \dots Y_m\})(q_0) = n \tag{4.8}$$

Jakubcyk and Sontag [62] provided a generalization of this to time invariant discrete time systems of the form

$$x(t_{k+1}) = f(x, u(t_k)), \tag{4.9}$$

where $f_u = f(\cdot, u(t_k)) : \mathcal{X} \to \mathcal{X}$ is a global diffeomorphism and $x \in \mathcal{X}$, where $\mathcal{X}$ is a manifold of class C^k, Hausdorff, and second countable[4].

Let us define the vector fields

$$\begin{aligned} \mathbb{Y}_i &= \frac{\partial}{\partial v^i}\Big|_{v=0} f_u^{-1} \circ f_{u+v}(x) \\ Ad_{u_k \dots u_1} \mathbb{Y}_i(x) &= \frac{\partial}{\partial v^i}\Big|_{v=0} (f_{u_k} \circ \dots \circ f_{u_1})^{-1} \circ f_{u_i}^{-1} \circ f_{u_i+v}(x) \circ f_{u_k} \circ \dots \circ f_{u_1}, \end{aligned}$$

for which we can then state the theorem

Theorem 4.3.4 *The discrete-time system* (4.9) *is forward accessible from equilibrium state* x_0 *if and only if*

$$rank(\overline{Lie}(\{Ad_{u_k \dots u_1} \mathbb{Y}_i(x) | k \geq 0, 1 \leq i \leq m, u_0, \dots u_k \in U\})(q_0) = n, \tag{4.10}$$

where $U \subset clos\ int\ U$ *(the closure of the interior of* U*) is a subset of* $\mathbb{R}^m$ *and any two points in the same connected component of* U *can be joined by a smooth curve lying entirely in the interior of* U *(excepting, possibly, the endpoints).*

While local controllability is not as easy to establish, there is a way to check *small-time local controllability*, where any state close to the equilibrium point q_0 is reachable

[4] A topological space is second countable if it has a countable basis

from q_0. Alternatively, it can be defined as:

Definition 4.3.5 (small-time local controllability) *A system* (4.4) *is* small-time locally controllable (STLC) *from q_0 if it is locally accessible from q_0 and q_0 is in the interior of the reachable set $\mathcal{R}_Q(q_0)$ from q_0.*

Only sufficiency conditions for STLC are known and require the following definitions:

(i) Given the $(m+1)$-tuple of non-negative integers $(k, l) = (k, l_1, \ldots l_m)$, let $\overline{\text{Lie}}^{(k,l)}(\{X, Y_1 \ldots Y_m\})$ be the distribution generated by all iterated Lie brackets containing the vector X k times and the vector Y_j l_j times.

(ii) Given a weight $\theta \in [0, 1]$ and two $(m+1)$-tuples (k, l) and (k', l'), define the ordering $(k, l) \prec_\theta (k', l')$ as whenever $\theta k + \sum_j l_j = \theta k' + \sum_j l'_j$.

Theorem 4.3.6 (Sussmann [121], Kawski [66]) *Suppose the system* (4.4) *is locally accessible from q_0. The system is STLC from q_0 if there exists $\theta \in [0, 1]$ such that, whenever k is odd and $l_1, \ldots l_m$ are all even,*

$$\overline{Lie}^{(k,l)}(\{X, Y_1 \ldots Y_m\})(q_0) \subseteq \cup_{(k,l) \prec_\theta (k',l')} \overline{Lie}^{(k',l')}(\{X, Y_1 \ldots Y_m\})(q_0). \tag{4.11}$$

This is known as the "good/bad bracket condition." For driftless control systems ($X = 0$), local accessibility is equivalent to STLC, as $k = 0$. Accessibility and controllability are likewise equivalent for linear systems, where the LARC reduces to the controllability matrix rank condition.

4.3.3 Maneuver Automaton Properties

The maneuver automaton is a hybrid system, but can be viewed as a consistent hierarchical abstraction [46] of the underlying system's continuous dynamics, as a sequence of primitives and maneuvers on the automaton corresponds to a trajectory on the full continuous system.

Well-posedness

To ensure that the automaton is hierarchically consistent with respect to the existence and uniqueness of the initial value problem, we must define well-posedness of the automaton such that the well-posedness of the underlying system is enforced. As motion along the reference trajectories follows the continuous dynamics of the underlying system, such motion is well-posed iff the underlying system is well-posed. When addressing maneuvers, assuming a maneuver is unique [5] for a given discrete-time control input ν, the same is true, as they are also defined as finite-time trajectories on the continuous system. With this in mind, a given initial condition and discrete-time control history on the maneuver automaton ensures that a solution exists and is unique for the corresponding initial value problem on the underlying continuous-time system, provided that the transitions cannot form an accumulation point (i.e., no Zeno executions). This is not possible because there is a nonzero minimum time for all maneuvers, insured by the fact that a finite set of fixed-time maneuvers exist and that the time scaling factor (if applicable) is assumed to be bounded away from zero. Thus, because the well-posedness of the underlying system is enforced, the maneuver automaton is well-posed by definition.

Accessibility

As the automaton is a hybrid of continuous and discrete systems, accessibility of the automaton requires accessibility of the system projected on both the continuous and discrete systems. The *accessibility* of the states on the discrete space $\mathbb{H}_d$ depends solely on the connectedness of the automaton. To characterize accessibility on the continuous state, we must examine the evolution on the symmetry group H, paralleling the development in [41].

Let us consider a fixed maneuver sequence $\bar{\mu}$, for which the evolution of the system is written as

$$\gamma_{n_T} = \varphi_{\bar{\mu}}(\gamma_0), \quad \varphi_{\bar{\mu}} = \varphi_{\bar{\nu}_{n_T}} \circ \varphi_{\bar{\nu}_{n_T-1}} \circ \cdots \circ \varphi_{\bar{\nu}_1}.$$

As we are interested in motion on the continuous part of the maneuver space $\mathbb{H}_c$, let us analyze the effects of perturbing the coasting times τ_i and, in the case of Δh in an

[5] For a finite set of precomputed maneuvers, this can be easily enforced. In addition, maneuvers based on the local planner of Chapter 3 also are guaranteed to be unique.

infinite closed set, the symmetry group offset Δh. As this will not affect the discrete state, we will describe the resulting change as follows:

$$\gamma^c_{n_T} = \mathcal{P}_{\bar{\mu}}(\gamma^c_0, \delta\tau, \{\delta\Delta h\}, \tag{4.12}$$

where γ^c is the projection of γ onto $\mathbb{H}_c$ and $\delta\tau$ as well as $\delta\Delta h$ correspond to the sequence of perturbations on the coasting times and symmetry group offsets. The parentheses about $\delta\Delta h$ denotes that this term is only applicable when Δh is in a set which is open locally about the origin in at least one dimension. Note that $\delta\tau \in \mathcal{I}_{\delta\tau}(\tau) = \{\delta\tau | \forall(\tau + \delta\tau) \in \mathcal{I}_\tau\}$ and $\delta\Delta h \in \mathcal{U}_\delta(\Delta h, \ell, h) = \{\delta\Delta h | \forall(\Delta h + \delta\Delta h) \in \mathcal{U}(\ell, h)\}$. The mapping $\mathcal{P}_{\bar{\mu}}$ is continuous in each of its arguments because evolution on the trajectory primitives is continuous with respect to time and the symmetry group. Additionally, $\mathcal{P}_{\bar{\mu}}(., 0, 0)$ is a diffeomorphism, as it is simply a translation on $\mathbb{H}_c$. The relation (4.12) can thus be considered a discrete time system. Note that, if $\mathcal{P}_{\bar{\mu}}$ is a fixed-point map, i.e., $\mathcal{P}_{\bar{\mu}}(\gamma^c, 0, 0) = \gamma^c$, then the system is driftless. The *reachable set* on $\mathbb{H}_c$ for this system under perturbations on the maneuver sequence $\bar{\mu}$ is then defined as:

$$\mathcal{R}^{\bar{\mu}}_{H_c}(\gamma^c_0, \leq t_{\text{final}}) = \{\gamma^c \in \mathbb{H}_c : \gamma^c = \mathcal{P}_{\bar{\mu}}(\gamma^c_0, \delta\tau, \{\delta\Delta h\}), \sum_i (T_i + \tau_i + \delta\tau_i) \leq t_{\text{final}}\} \tag{4.13}$$

A first order test to see if $\mathcal{R}^{\bar{\mu}}_{H_c}$ has a non-empty interior would be to check if the rank of the Jacobian of the system (4.12) is full:

$$\frac{\partial \mathcal{P}_{\bar{\mu}}}{\partial(\delta\tau, \{\delta\Delta h\})}$$

where $N_{\mathbb{H}_c} = dim\mathbb{H}_c$.

Note that, when Δh is in the closure of an open set (but not on the boundary) {and τ is in the closure of an open interval (but not on the boundary)} [6], the Jacobian is always full rank because the open set must be of full dimension. However, if this is not the case, one could define a new sequence $\bar{\mu}^N$ by iterating the control sequence $\bar{\mu}$ a sufficiently large number of times N. Thus, one can get a version of the LARC for the maneuver

[6] This condition is only necessary when time is included as a state

automaton (where perturbations on Δh are not possible or neglected):

Theorem 4.3.7 (Reachability-Frazzoli) *The system* (4.12) *is (forward) accessible (the reachable set has a non-empty interior) from all* $\gamma^c \in \mathbb{H}_c$ *if, for all* $\sum_i (T_i + \tau_i + \delta\tau_i) \leq t_{final}$ *and some* $N > 0$, *if* $\tau_i \in int\mathcal{I}_\tau; \forall i = 1 \dots n_T$ *and*

$$\overline{Lie}\left(\left\{\frac{\partial \mathcal{P}_{\bar{\mu}^N}}{\partial \delta\tau_i} \forall i = 1 \dots n_T\right\}\right) = N_{\mathbb{H}}.$$

Proof: As $\mathcal{P}_{\bar{\mu}}(., \delta\tau, \delta\Delta h) : H \to H$ is a diffeomorphism, $\mathcal{P}_{\bar{\mu}}$ is a fixed-point map, i.e., is in equilibrium, $Ad_{u_{n_T} \dots u_1} \mathbb{Y}_i(x) = \frac{\partial \mathcal{P}_{\bar{\mu}}}{\partial \delta\tau_i} \forall i = 1 \dots n_T$, and $\delta\tau_I \in \mathcal{I}_{\delta\tau} \forall i = 1 \dots n_T$, the conditions for theorem 4.3.4 are satisfied and the system is (forward) accessible. ■

Controllability

Maneuver Automaton controllability is defined as follows:

Definition 4.3.8 (Controllability) *A maneuver automaton is controllable if, given any initial condition* (l_i, h_i), *and any compact set* $Y \subset H$, *there exists a time* T_F *such that it is possible to find an admissible sequence of primitives steering the system to any desired goal configuration* (l_f, h_f), *with* $h_f \in Y$, *in time* $t_f \leq T_F$.

From what we understand of the automaton projected onto the continuous and discrete states, we can state necessary and sufficient conditions for automaton controllability:

Theorem 4.3.9 (Controllability-Frazzoli) *The maneuver automaton is controllable if and only if*

(i) The directed graph of discrete states is strongly connected.

(ii) There exists a fixed point maneuver sequence $\bar{\mu}$ *such that* $\gamma^c \in int\mathcal{R}^{\bar{\mu}}_{\mathbb{H}_c}(\gamma^c, \leq t_{final})$.

Proof: Proof follows from definition of controllability on discrete and continuous systems as well as use of symmetry. For a complete proof, see [41]. ■

Frazzoli [41] applied this to planar vehicles with no gravity and vehicles in three dimensions with constant gravity, constraining Δh to a finite discrete set. For such systems, he showed controllability on the maneuver automaton required no more or less than two reference trajectories and two maneuvers (total) connecting them back and forth from one another, provided that they can be combined to form a fixed point maneuver. There is more flexibility when dealing with Δh in an set with a nonempty interior.

Lemma 4.3.10 *When a Δh_i of a control sequence μ is in the interior of its input set and maneuvers can connect to equivalent reference trajectories, the maneuver automaton is controllable on H with a single reference trajectory primitive if and only if there exists a fixed point maneuver (with respect to H) $\left(\mathcal{P}_{\vec{\mu}}^{H}(\gamma^c, 0, 0) = h\right)$ that can be created from it.*

Proof: By assumption, a reference trajectory can always connect to an equivalent trajectory through a single maneuver. If there exists a fixed point maneuver, then, as any φ_{μ}^{H} is a simple group action and Δh_i is in an open set by definition, so the original point must also be within an open set. If the system is controllable with a single reference trajectory primitive, then a fixed point maneuver must exist by 4.3.9. ■

It then follows that other reference trajectories and maneuvers can be added without altering the controllability of the automaton provided that the discrete graph representation of the system remains strongly connected. In the most trivial case, an equilibrium point reference trajectory could provide controllability.

4.4 Optimal Control on the Maneuver Space

While the hybrid system lends itself to more computationally efficient solutions than the corresponding continuous system, the cost of the maneuver automaton system lies in the fact that a subset of motions are used to approximate the set of all feasible motions admitted by the dynamical system. As a result, any optimal control on the maneuver space is only an approximation of the true optimal control. This difference is bound by

the maneuver library's coverage of the space of all possible motions of the continuous system.

To impose an optimal control on this hybrid system, one must first define the cost functional to be minimized (maximized). This cost functional can be defined as

$$J = \int_{t_0}^{t_f} L\left(\xi(t), u(t), t\right) dt. \tag{4.14}$$

For the maneuver automaton, this reduces to

$$J = \sum_{i=1}^{n_T - 1} \left(J_{T[i]} + J_{M[i]} \right), \tag{4.15}$$

where the subscripts T and M denote reference trajectory and maneuver, respectively. We have:

$$J_T = \int_0^{\tau} L\left(\phi_u(\xi_0(t), t), u(t), t\right) \frac{1}{\eta} d\sigma$$

$$J_M = \int_{\tau}^{\tau + T} L\left(\phi_u(\xi_0(t), t), u(t) + u_{ref}, t\right) \frac{1}{\eta} d\sigma$$

For $L = 1$, or a minimum time problem,

$$J_T = \frac{1}{\eta}\tau \quad J_M = \frac{1}{\eta}T;$$

$$\begin{aligned} minJ &= \min_{\nu} \sum_{i=1}^{n_T - 1} \left(J_{T[i]} + J_{M[i]} \right) \\ &= \min_{\nu} \sum_{i=1}^{n_T - 1} \left(\frac{1}{\eta[i]} (\tau[i] + T[i]) \right). \end{aligned}$$

This cost is invariant to all motion on the vehicle symmetry group excepting a time-scaling symmetry, if it exists. In such a case, the symmetry action on the cost would be

described as:

$$\psi_J : (J, \rho_\eta) \mapsto \left(\rho_\eta^{-1} J\right).$$

Let us now address $L = u^{tot}(t)^T u^{tot}(t)$, or minimum "energy" (control effort) problem, where $u^{tot} = K_u(\bar{u}(t) + u_{ref}(t))$:

$$\begin{aligned}
J_T &= \int_0^\tau u_{ref}(\sigma)^T K_u^T K_u u_{ref}(\sigma) \frac{1}{\eta} d\sigma \\
J_M &= \int_\tau^{\tau+T} (\bar{u}(\sigma) + u_{ref}(\sigma))^T K_u^T K_u (\bar{u}(\sigma) + u_{ref}(\sigma)) \frac{1}{\eta} d\sigma \\
&= \int_\tau^{\tau+T} u_{ref}(\sigma)^T K_u^T K_u u_{ref}(\sigma) \frac{1}{\eta} d\sigma + 2 \int_\tau^{\tau+T} u_{ref}(\sigma)^T K_u^T K_u \bar{u}(\sigma) \frac{1}{\eta} d\sigma \\
&+ \min_u \left[\int_\tau^{\tau+T} \bar{u}(\sigma)^T K_u^T K_u \bar{u}(\sigma) \frac{1}{\eta} d\sigma \right].
\end{aligned}$$

Assuming the maneuvers are defined as the minimum energy controls from Chapter 3, $\bar{u}(t) = -B^T \lambda = -B^T \sum_{k=1}^{+\infty} \Phi_{k,\lambda\lambda}(t)(\lambda_0)$, thus

$$\begin{aligned}
J_M = &\int_\tau^{\tau+T} u_{ref}(t)^T K_u^T K_u u_{ref}(t) dt + 2 \int_\tau^{\tau+T} u_{ref}(t)^T K_u^T K_u \left(-B^T \sum_{k=1}^{+\infty} \Phi_{k,\lambda\lambda}(t)(\lambda_0) \right) dt \\
&+ \min_u \left[\int_\tau^{\tau+T} \left(-B^T \sum_{k=1}^{+\infty} \Phi_{k,\lambda\lambda}(t)(\lambda_0) \right)^T K_u^T K_u \left(-B^T \sum_{k=1}^{+\infty} \Phi_{k,\lambda\lambda}(t)(\lambda_0) \right) dt \right].
\end{aligned}$$

From (2.11), we know that

$$K_u = T_u \begin{bmatrix} \eta^2 \kappa I_{m_1 \times m_1} & 0_{m_1 \times m_2} \\ 0_{m_2 \times m_1} & \eta^2 \kappa I_{m_2 \times m_2} \end{bmatrix} T_u^T$$

where T_u is an orthogonal matrix. Thus,

$$\begin{aligned}
K_u^T K_u &= \eta^4 T_u \begin{bmatrix} \kappa^2 I_{m_1} & 0 \\ 0 & I_{m_2} \end{bmatrix} T_u^T \\
&= \eta^4 T_u \begin{bmatrix} 0_{m_1} & 0 \\ 0 & I_{m_2} \end{bmatrix} T_u^T + \eta^4 \kappa^2 T_u \begin{bmatrix} I_{m_1} & 0 \\ 0 & 0_{m_2} \end{bmatrix} T_u^T
\end{aligned}$$

Thus, the cost can be divided such that

$$\begin{aligned} J &= \int_{t_0}^{t_f} u(t)^T K_u^T K_u u(t) \frac{1}{\eta} dt \\ &= \eta^4 \int_{t_0}^{t_f} u(t)^T T_u \begin{bmatrix} 0_{m_1} & 0 \\ 0 & I_{m_2} \end{bmatrix} T_u^T u(t) \frac{1}{\eta} dt + \eta^4 \kappa^2 \int_{t_0}^{t_f} u(t)^T \begin{bmatrix} I_{m_1} & 0 \\ 0 & 0_{m2} \end{bmatrix} u(t) \frac{1}{\eta} dt \\ &= J_1 + J_2 \end{aligned}$$

The "energy" cost is invariant to all motion on the vehicle symmetry group excepting scaling symmetries, if they exist. In such a case, the symmetry action on the cost would be described as:

$$\psi_J : (J = J_1 + J_2, \rho_\eta, \rho_\kappa) \mapsto \left(\rho_\eta^3 J_1 + \rho_\eta^3 \rho_\kappa^2 J_2 \right).$$

4.5 Computational Search Methodology

4.5.1 Obstacle-Free Dynamic Programming

The object of the computation is to minimize the cost functional for the Maneuver Automaton, subject to specified initial and final conditions

$$min_{\mu, n_T} J_{n_T} \qquad \text{where} \quad J_0 = 0 \tag{4.16}$$

$$J_{k+1} = J_{\Delta_k}(\gamma_k, \nu_k) + J_k = \left(J_{T[i]} + J_{M[i]} \right)(\gamma_k, \nu_k) + J_k \tag{4.17}$$

subject to

$$\gamma_{k+1} = \left\{ \begin{matrix} \ell_{k+1}, h_{k+1} \\ \ell_{k+1}, h_{k+1}, t_{k+1} \end{matrix} \right\} = \varphi(\gamma_k, \nu_k)$$

$$\gamma_0 = \left\{ \begin{matrix} \ell_0, h_0 \\ \ell_0, h_0, t_0 \end{matrix} \right\} \quad \gamma_{n_T} = \left\{ \begin{matrix} \ell_{n_T}, h_{n_T} \\ \ell_{n_T}, h_{n_T}(t_{n_T}), t_{n_T} \in \mathcal{I}_f \end{matrix} \right\}$$

$$h_{min} \leq h_k \leq h_{\max} \forall k \in \{1 ... n_T\},$$

where n_T is unknown, $\mathcal{I}_f$ is a closed and bounded interval in $\mathcal{I}_0$ and $\|h_{min}\| < \infty$, $\|h_{\max} < \infty\|$. The state transition relation φ is defined as in (4.2). Because of the bounds on h and, when included as a state, t ($t \geq 0$ by definition and $t \leq \sup(I_f)$), the projection of the hybrid state onto the continuous space γ^c is limited to the compact set $\mathcal{H} \subset \mathbb{H}_c$. Furthermore, let $\mathcal{R}_k \subseteq \mathcal{H}$ be the reachable set after k steps. Let $\bar{\mathcal{R}}_k$ then be the set defined as $\cup_{i=1}^{k} \mathcal{R}_i \subseteq \mathcal{H}$, the set of all reachable points in k or less steps. Note that, for a finite number of steps, $\mathcal{R}_k$ can be shown to be compact by induction, as each set from a given point has a compact reachable set and the initial condition is a single given point. If the system is controllable over all $\mathcal{H}$, then, when time is not a state, $\lim_{k\to\infty} \mathcal{R}_k = \mathcal{H}$. Because each maneuver has a finite time with a lower bound $\epsilon > 0$, there exists a finite integer $k_{\max}$ for which $\mathcal{R}_k = \text{NULL}\ \forall k > k_{\max}$ and $\lim_{k\to\infty} \bar{\mathcal{R}}_k = \bar{\mathcal{R}}_{k_{\max}}$, where $k_{\max}$ is defined as the smallest integer for which $k_{\max} \leq t_{\max}/\epsilon$. Thus, $\mathcal{R}_k$ is compact for all $k \in \mathbb{N}$, as is $\bar{\mathcal{R}}_k$.

Let $J^*(\gamma)$ be the optimal cost to reach the final condition (the cost-to-go) for the initial condition γ, i.e., the cost due to the optimal choice of reference trajectories and maneuvers. It should be noted that, due to symmetry, transformation on H can allow this function to be used for *any* final condition with the same final reference trajectory. By Bellman's optimality principle, given an initial control $\nu = (\tau, \ell_{next}, h_{next})$, the following relation must be satisfied:

$$J^*(\gamma) = min_\nu \left(J_T + J_M + J^*(\varphi_\nu(\gamma))\right) \tag{4.18}$$

This is called Bellman's equation. Thus, once the optimal cost function is known, the optimal control ν^* can be computed by solving

$$\nu^* = (\tau^*, \ell^*_{next}, h^*_{next}) = argmin_\nu \left(J_T + J_M + J^*(\varphi_\nu(\gamma))\right) \tag{4.19}$$

The full set of discrete controls can be found by simply iterating this procedure from the initial conditions through each (ℓ_{next}, h_{next}) to the final conditions without any prior knowledge of the required number of maneuvers. We also can derive the optimal cost-to-go through an iterative procedure, which can be seen to be monotonically decreasing with J^* and Bellman's equation defining the final stationary point. As the number of steps n_T is unknown, we will set up the problem as an infinite horizon optimization:

$$J^*_{k+1}(\ell, h) = min_\nu \left(J_T + J_M + J^*_k(\varphi_\nu(\gamma)) \right) \tag{4.20}$$
$$\lim_{k \to \infty} J^*_k = J^*_\infty,$$

where $J_0(\gamma)$ is an upper bound on $J^*(\gamma)$ possibly derived from initial feasible trajectories or $J^*(\ell, h) = \infty$ except at (γ_f) where $J^*(\gamma_f) = 0$. Where the final condition is time-dependent (i.e., a fixed final time or a condition changing with time), the cost-to-go can be defined as a function of γ and time t, where $J^*(\gamma_f(t_f), t_f) = 0$ for all $t_f \in \mathcal{I}_f \subseteq \mathbb{R}$ and $\mathcal{I}_f$ is the interval over which the final condition acts ($\mathcal{I}_f$ is a single point for fixed final time problems).

To implement this dynamic programming routine, the continuous state space $\mathbb{H}_c$ and the continuous control space $\mathcal{I}_\tau\{\times\mathcal{U}\}$[7] must be discretized, where $\mathcal{U} = \cup_{\forall \gamma^c \in \mathcal{H}} \mathcal{U}(\gamma^c)$. For this purpose, we propose to represent $\mathcal{H}$ as a $N_{\mathbb{H}_c}$-dimensional uniform grid and τ as a sequence of equally spaced points. For this development, we will focus on H discretization, as the process for τ is the same when $N_{\mathbb{H}_c} = 1$. Let there be N_i nodes per dimension $\{i = 1 \dots N_{\mathbb{H}_c}\}$ resulting in $N_{\text{tot}} = \prod_{i=1}^{N_{\mathbb{H}_c}} N_i$ nodes in the discretized space. This space is then divided into $\prod_{i=1}^{N_{\mathbb{H}_c}} (N_i - 1)$ $N_{\mathbb{H}_c}$-dimensional cells with $2^{N_{\mathbb{H}_c}}$ nodes per cell and the vector $d\gamma^c$ defining the length of each "side" of the cell. Thus, the discretized version of $\mathbb{H}_c$ is expressed as

$$\left(\gamma^{c1}, \gamma^{c2}, \dots \gamma^{cN_{\text{tot}}}\right) = \left(\begin{bmatrix} \gamma^c_1 \\ \gamma^c_2 \\ \vdots \\ \gamma^c_k \end{bmatrix}^1, \dots, \begin{bmatrix} \gamma^c_1 \\ \gamma^c_2 \\ \vdots \\ \gamma^c_k \end{bmatrix}^{N_{\text{tot}}} \right), \tag{4.21}$$

where the location within the cell can be described with respect to the nodes of the cell $i_1, i_2, \dots i_{2^{N_{\mathbb{H}_c}}}$ by the function

[7] Only applies when Δh is in a closed set which is open locally about the origin in at least one dimension.

$$\gamma^c = \begin{bmatrix} \gamma_1^c \\ \gamma_2^c \\ \vdots \\ \gamma_k^c \end{bmatrix} = \sum_{j=1}^{2^{N_{\mathbb{H}_c}}} S_j(\gamma_{rel}^c)\gamma^{ci_j} \tag{4.22}$$

$$\gamma_{rel}^c = \gamma^c - \gamma_{lbound}^c - d\gamma^c/2 \tag{4.23}$$

The location within the cell is defined relative to the center of the cell and is denoted by γ_{rel}^c, while the lower bound on the coordinates of the cell are described by the vector γ_{lbound}^c. The interpolation, or shape, functions S_j are defined as

$$S_j(\gamma^c) = \prod_{p=1}^{k} (\gamma_{rel}^c - \gamma_{rel}^{ci_j})_p, \tag{4.24}$$

where $\gamma_{rel}^{ci_j}$ are the relative positions of each cell node and p is the index of the vector.

Based upon (4.22), the cost can be approximated by

$$\tilde{J} = \sum_{j=1}^{2^{N_{\mathbb{H}_c}}} S_j(\gamma_{rel}^c) J^{i_j}.$$

The optimal cost approximation is then found by setting the cost at all $N_{\text{tot}} \times max(\mathbb{H}_d)$ points in $\mathcal{H} \times \mathbb{H}_d$ to an upper bound and iterating (4.20), where, at every step, all discretized controls in $\mathbb{U}(\gamma)$ are tried at every discretized point $\gamma \in \mathcal{H} \times \mathbb{H}_d$. These iterations are then carried out $n_T^{\max}$ times until the cost is sufficiently converged ($\tilde{J}_{n_T^{\max}}^*$). Thus, we are now faced with an $n_T^{\max}$ stage problem, where the number of stages n_T required to reach the terminal condition are less than or equal to $n_T^{\max}$. The control policy is then found as discussed previously.

4.5.2 Convergence of the Dynamic Program

Let $\tilde{J}^*$ be the linear interpolated approximation of J^* on the continuous state space, where $\tilde{\mu}^*$ is the "optimal" control history (policy) on $\tilde{J}^*$ and μ^* is the optimal control

history on J^*. Let n_{nodes} is the number of nodes used to discretize the continuous state space (in a uniformly distributed pattern). The property that $\lim_{n_{nodes}\to\infty} \tilde{J}^* = J^*$ and $\lim_{n_{nodes}\to\infty} \tilde{\mu}^* = \mu^*$ is referred to as *consistency*. Consistency is typically guaranteed if there is a "sufficient amount of continuity" in the problem [12]. As this is a non-trivial problem, let us restrict ourselves to the case where only a single reference trajectory is used, i.e., $\ell_k = 1 \ \forall k \in 1 \dots n_T$. In this case, the discrete state can be ignored and $\mathbb{H} = \mathbb{H}_c$ ($\gamma = \gamma^c$). Extension of these results to multiple reference trajectories is left for future publication. For the following development, let $\mathcal{R}_k$ denote the reachable set of the automaton at stage k.

Theorem 4.5.1 *Given the system* (4.16) *restricted to only one reference trajectory and defining an arbitrary mesh over the state space $\mathcal{H}$ and the continuous control space $\mathcal{I}_\tau\{\times\mathcal{U}\}$[8], where $d_\mathcal{H}$ and $d_\mathcal{U}$ are equal to the maximum distances between any two grid element nodes over all the grid elements in the discretized state and control spaces, respectively. Furthermore, let $\tilde{J}(\gamma^c)$ be defined as a linear cost interpolation on each grid element. Then, if the "energy" cost is used (minimum of the integral of the dot product of the controls) and the control $u = K_u(u_{ref} + \bar{u})$ is defined such that u_{ref} is a continuous function of time only and*

(i) $\mathcal{U}(\gamma)$ is a finite set
or

(ii) $\mathcal{U}(\gamma)$ is Lipschitz continuous in a Hausdorff metric sense and compact for every $\gamma \in \mathcal{H}$ and $\bar{u}$ is continuous in t and Lipschitz continuous $\forall \nu \in \mathcal{U}(\gamma)$, $\gamma \in \mathcal{R}_k$

for any $n_T^{\max} \in \mathbb{N}$, there exists a constant $\alpha_{n_T^{\max}}$ such that

(i) $\left|J^*_{n_T^{\max}} - \tilde{J}_{n_T^{\max}}\right| < \alpha_{n_T^{\max}} d_\mathcal{H}$
or

(ii) $\left|J^*_{n_T^{\max}} - \tilde{J}_{n_T^{\max}}\right| < \alpha_{n_T^{\max}} (d_\mathcal{H} + d_\mathcal{U})$,

respectively, for all $h \in \widetilde{\mathcal{R}}_{n_T^{\max}}$. Thus, as $d_\mathcal{H}\{and\ d_\mathcal{U}\} \to 0$, $\tilde{J}_{n_T^{\max}}(\gamma^c) \to J^*_{n_T^{\max}}(\gamma^c)$ and $\tilde{\mu}_{n_T^{\max}}(\gamma^c) \to \mu^*_{n_T^{\max}}(\gamma^c)$.

[8] Only applies when Δh is in a closed set which is open locally about the origin in at least one dimension.

Proof: The proof for this theorem is based upon the proof of convergence of the truncation of the infinite horizon problem in [11]. For the sake of brevity, it will be cited in this proof rather than reiterated. Once again, the system is stationary. In this case, however, the discount factor of the cost functional is assumed to be one and the stepwise interpolation scheme is replaced by a linear interpolation scheme as described in previous sections. Following the reasoning in [11], the discount factor in the cost functional is irrelevant to the results stated in the theorem. Additionally, the linear interpolation scheme can be accommodated as follows: Proposition 1 of [11] is mesh-independent and thus automatically applies. To address Proposition 2 of [11] and the development in Section IV of [11], we need to look at certain properties of the cost approximation. For every grid element, let $\tilde{J}_{n_T^{\max}}(\gamma^{ci^{\max}})$ denote the maximal cost of each grid element and $h^{i^{\max}}$ denote the vertex at which it occurs. Likewise, let $\tilde{J}_{n_T^{\max}}(\gamma^{ci^{min}})$ denote the minimal cost of each grid element and $h^{i^{min}}$ denote the vertex at which it occurs. Thus, at any point h on the grid element:

$$\tilde{J}_{n_T^{\max}}(\gamma^{ci^{min}}) \leq \tilde{J}_{n_T^{\max}}(\gamma^{c}) \leq \tilde{J}_{n_T^{\max}}(\gamma^{ci^{\max}}) \tag{4.25}$$

and, providing assumptions A' or B' of [11] are satisfied, the following is also true:

$$\left|J^*_{n_T^{\max}}(\gamma^c) - \tilde{J}_{n_T^{\max}}(\gamma^c)\right| \leq \max_{\gamma^{ci}\in\{\gamma^{ci^{\min}},\gamma^{ci^{\max}}\}} \left|J^*_{n_T^{\max}}(\gamma^c) - \tilde{J}_{n_T^{\max}}(\gamma^{ci})\right| \tag{4.26}$$

$$\leq \alpha_{n_T^{\max}} d_{\mathcal{H}} \tag{4.27}$$

$$\left|J^*_{n_T^{\max}-1}(\gamma^c) - \tilde{J}_{n_T^{\max}-1}(\gamma^c)\right| \leq \max_{\gamma^{ci}\in\{\gamma^{ci^{\min}},\gamma^{ci^{\max}}\}} \left|J^*_{n_T^{\max}-1}(\gamma^c) - \tilde{J}_{n_T^{\max}-1}(\gamma^{ci})\right| \tag{4.28}$$

$$\leq (A_{n_T^{\max}-1} + \alpha_{n_T^{\max}}) d_{\mathcal{H}} \tag{4.29}$$

$$\left|\tilde{J}_{n_T^{\max}-1}(\gamma^c) - \hat{J}_{n_T^{\max}-1}(\gamma^c)\right| \leq \max_{\gamma^{ci}\in\{\gamma^{ci^{\min}},\gamma^{ci^{\max}}\}} \left|\tilde{J}_{n_T^{\max}-1}(\gamma^{ci}) - \hat{J}_{n_T^{\max}-1}(\gamma^c)\right| \tag{4.30}$$

$$\leq \alpha_{n_T^{\max}-1} d_{\mathcal{H}} \tag{4.31}$$

Note that the notations $\tilde{J}$ and $\hat{J}$ are reversely defined in [11]. These relations extend the corresponding development in Proposition 2 and Section IV of [11] to

fit our system. The first equation replaces (40) in [11], while the second allows for the equation following (41) to be satisfied. The third accommodates for the relation in the equation preceding (43) of [11].

Thus, it only remains to show that assumptions A' or B' are satisfied. This is relatively simple, as the compactness and continuity conditions on the input spaces and reachable sets are satisfied by definition. Furthermore, $\varphi_k(\gamma, \nu)$ is Lipschitz continuous for all $\gamma \in \mathcal{R}_k, \nu \in \mathcal{U}(\gamma)$ by equation (4.2) and the reference trajectory definition. Furthermore, J_{Δ_k} is lipschitz continuous for all $\gamma \in \mathcal{R}_k, \nu \in \mathcal{U}(\gamma)$ because the cost is defined as the time integral of the dot product of the controls, where reference controls are continuous in time (by definition) and independent of γ and ν, whereas the maneuver control is Lipschitz continuous for all $\gamma \in \mathcal{R}_k, \nu \in \mathcal{U}(\gamma)$ by assumption. Thus, by Proposition 2 and Section IV of [11], conditions i) and ii) are satisfied respective to assumptions i) and ii).

■

Note that this theorem does not prove convergence to the infinite stage problem. Rather, it shows convergence to a truncation of the infinite stage problem. However, if, for a given $\epsilon > 0$, one can choose $n_T^{\max}$ such that $|J^*_{n_T^{\max}} - J^*_\infty| < \epsilon$, one can ensure convergence to the optimal solution of the infinite stage problem.

Lemma 4.5.2 *Given the same system and assumptions as in 4.5.1 with an upper time bound of* $t_{\max}$. *Then there exists a lower bound* $T_\epsilon > 0$ *on the maneuver time and a finite* $n_T^{\max} = round_{up}(t_{\max}/T_\epsilon)$ *for which* $J^*_{n_T^{\max}} = J^*_\infty$ *over all points in* $\bar{\mathcal{R}}_\infty$.

Proof: As each maneuver has a fixed time length by definition and the time scaling factor must be bounded away from zero because the available states on the symmetry group are closed and bounded, the lower bound $T_\epsilon > 0$ on the maneuver time must exist. Furthermore, any state requiring $> \text{round}_{\text{up}}(t_{\max}/T_\epsilon)$ stages would exceed $t_{\max}$ and thus be inadmissible. Thus, $\mathcal{R}_k = \text{NULL}\ \forall k > \text{round}_{\text{up}}(t_{\max}/T_\epsilon)$ and $J^*_{n_T^{\max}} = J^*_\infty$ over all points in $\bar{\mathcal{R}}_{\text{round}_{\text{up}}(t_{\max}/T_\epsilon)} = \bar{\mathcal{R}}_\infty$. ■

Corollary 4.5.3 *Given that $\mathcal{F}_\delta$ is Lipschitz continuous in the continuous components of γ and ν, base functions (if used) are continuous functions of time, and a compact set of inputs $\mathcal{U}(\gamma)$ (Lipschitz continuous in a Hausdorff metric sense) is defined such that $\mathcal{F}_\delta(\gamma, \nu) \in \mathcal{B}_\delta(0) \forall \nu \in \mathcal{U}(\gamma), \gamma \in \mathcal{H}$, then the local planner in Chapter 3 can be used to satisfy Theorem 4.5.1.*

Proof: Only need to prove assumption ii) of the theorem is satisfied. The conditions on the input set are satisfied by definition and also meet the convergence criterion of the local planner. As $\bar{u}$ is a function of x_{target} by nested polynomial series, it is a smooth function of x_{target}. As $x_{\text{target}} = \mathcal{F}_\delta(\gamma, \nu)$, $\bar{u}$ must be Lipschitz continuous in the continuous components of γ and ν. Also, because $A(t)$ is continuous in time and the base functions (if used) are continuous in time, the control must be continuous in time also. Thus, the assumptions of the theorem are satisfied. ■

4.5.3 Optimality of the Dynamic Program

As noted in the previous section, under certain assumptions, the solution to the dynamic programming routine on the maneuver automaton converges to the optimal control policy on the maneuver automaton. This, however, leaves the question of how the optimal control of the maneuver automaton approaches the true optimal for the full continuous system. While a theoretical comparison between the two is beyond the scope of this dissertation, there are good reasons to believe that the solution to the maneuver automaton will converge to the true optimal under an increase in the density of reference trajectory primitives. A study of a simple double integrator system was carried out by Frazzoli [41] in which he showed this to be true.

4.6 Algorithm Description

The algorithm solving the overall motion planning problem using the aforementioned approaches must be broken into two segments: preprocessing and online computation, with the most time-consuming aspects addressed in the preprocessing.

4.6.1 Preprocessing

Finding the Maneuver Data The maneuvers can be precomputed and stored into memory and/or the necessary precalculations completed for an online planner. For the local planner in Chapter 3, one must find the local planner equations (3.3) for every reference trajectory and store them into memory.

Discretizing H Second, the symmetry set must be discretized (4.21) for calculation of the optimal cost-to-go distribution.

Generate set of discrete inputs, $\nu = (\tau, \ell_{next}, \Delta h)$ Third, a discrete set of inputs is also needed to compute the dynamic programming routine. For precomputed trajectories, this corresponds to a finite set of ℓ_{next} and Δh for each reference trajectory. When the set of possible inputs $\mathcal{U}(\gamma)$ is an infinite compact set, one can discretize the continuous portion in the fashion mentioned earlier. If the maneuvers can be calculated online, one can also define an input ν_f controlling the system to the final state when such an input falls within the set of possible inputs. This allows for control precisely to a final condition even with a coarse grid. It is important to note that the final state on the symmetry group can be defined arbitrarily in the preprocessing, as the preprocessed inputs and cost distribution can be transformed to meet the actual final states using the symmetry property.

Calculate J^* distribution Fourth, calculate the optimal cost-to-go distribution as in equation (4.20).

4.6.2 Online computation

Solve for discrete controls Given the final state, the set of inputs, and the optimal cost-to-go distribution, Bellman's optimality principle can once again be used to find the set of optimal discrete inputs (4.19).

Solve for Continuous Controls Once the discrete controls have been found, the continuous controls are derived from the maneuver data compiled in preprocessing, either through precomputed maneuver control profiles or an online planner's result. For the local planner in Chapter 3, the transformation $\mathcal{F}_\delta$ can be used to find

the x_{final} for each maneuver. This yields the information needed to calculate the continuous controls from the local planner equations (3.3).

Simulation The discrete control segments can then be joined into a full control history, whose performance can be shown using a simple ordinary differential equation integration routine.

4.7 Earth to Mars Transfer Example

The object of this exercise is to find a variable very low thrust transfer from Earth to Mars. This is a translation-only problem with roational dynamics of the spacecraft neglected. It should be noted that, as this is an orbital transfer problem, the final condition is time dependent and time must be considered in the maneuver space. For simplicity, we will assume that all motion will occur in the orbital plane.

Only a single reference trajectory primitive will be used, that of a circular orbit, and the local planner from Chapter 3 is used to plan the maneuvers online. This structure is significant in that this problem would only be solvable by Frazzoli's automaton if multiple circular orbits of fixed radius were defined and transfers between them had been precalculated.

4.7.1 Mathematical Model

In this example we choose a mechanical control system modeling motion in a Newtonian gravity field. The configuration manifold is defined as $Q = \mathbb{R}^3$, where we choose the coordinate system $q = [q_1, q_2, q_3]^T$ to have an origin at the point source of the three-dimensional gravity well. The kinetic and potential energy are $\mathbb{M}(v_q, v_q) = m v_q^T v_q / 2$ and $V(q) = -m\mu / ||q||_2$, where m is the mass of the vehicle and $\mu = GM$, the gravitational constant G, multiplied by the mass of the attracting body M. Nonconservative forces such as drag are neglected. The controls $u = [u_1, u_2, u_3]^T$ are defined to be nondimensionalized control accelerations with the units $(1/s^2)$, where the matrix $g(q,t)$ both defines the control directions as well as the dimensionalizing factor. The equations of motion can then be written as:

$$\dot{\xi} = \begin{bmatrix} \dot{q} \\ \dot{v}_q \end{bmatrix} = \begin{bmatrix} v_q \\ -\frac{\mu}{\|q\|_2^3} q + g(q,t)u(t) \end{bmatrix} \tag{4.32}$$

For the augmented system, $\eta = \sqrt{(\mu/a^3)}$ corresponds to the mean motion of a Keplerian elliptical orbit with semimajor axis a. The distance scaling factor κ is equivalent to the semimajor axis.

The symmetry group H is defined as $SO(3) \times \mathbb{R}^+$ with an action that can be expressed as

$$\psi_h : (\xi, \eta) \mapsto \left(\begin{bmatrix} \rho_\kappa R_\delta & 0 \\ 0 & \sqrt{\frac{1}{\rho_\kappa}} R_\delta \end{bmatrix} \xi, \sqrt{\frac{1}{\rho_\kappa^3}} \eta \right), \tag{4.33}$$

where $\rho_\kappa = \frac{r_{new}}{r_{old}} \in \mathbb{R}^+$ is the ratio of the new radius $r_{new} = \|q_{new}\|_2$ to the old radius $r_{old} = \|q_{old}\|_2$ and $R_\delta = R_{new} R_{old}^T \in SO(3)$ is the rotation from one circular orbit and orbital position to another (of equal radius).

This group can be represented using the coordinates $[h_1, h_2, h_3, h_4]^T$ where $(h_1, h_2, h_3) = (\psi, \theta, \phi + \frac{\pi}{2})$ are defined in terms of the $3-1-3$ Euler angles $(\psi, \theta, \phi = \phi_\omega + \phi_f)$ and h_4 is the semimajor axis of the orbit. It should be noted that these quantities correspond to orbital elements (see Figure 4.4,), as ϕ_ω is the argument of periapsis, ϕ_f is the true anomaly, θ is the inclination, and ψ is the longitude of the ascending node.

Each conical reference trajectory can then be defined by a unique eccentricity. For this example, the only class of reference trajectories are the set of all circular orbits (e $= 0$), which happen to also be relative equilibria, thus $\ell = 0$ for all time. The relation between a position on the symmetry group for this reference trajectory and the vehicle's position in Ξ is then:

$$\xi = \mathcal{F}(h) = \begin{bmatrix} \mathcal{F}_x(h) \\ \mathcal{F}_v(h) \end{bmatrix} = \begin{bmatrix} \kappa(h) R(h_1, h_2, h_3) \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \\ \kappa(h)\eta(h) R(h_1, h_2, h_3) \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \end{bmatrix}$$

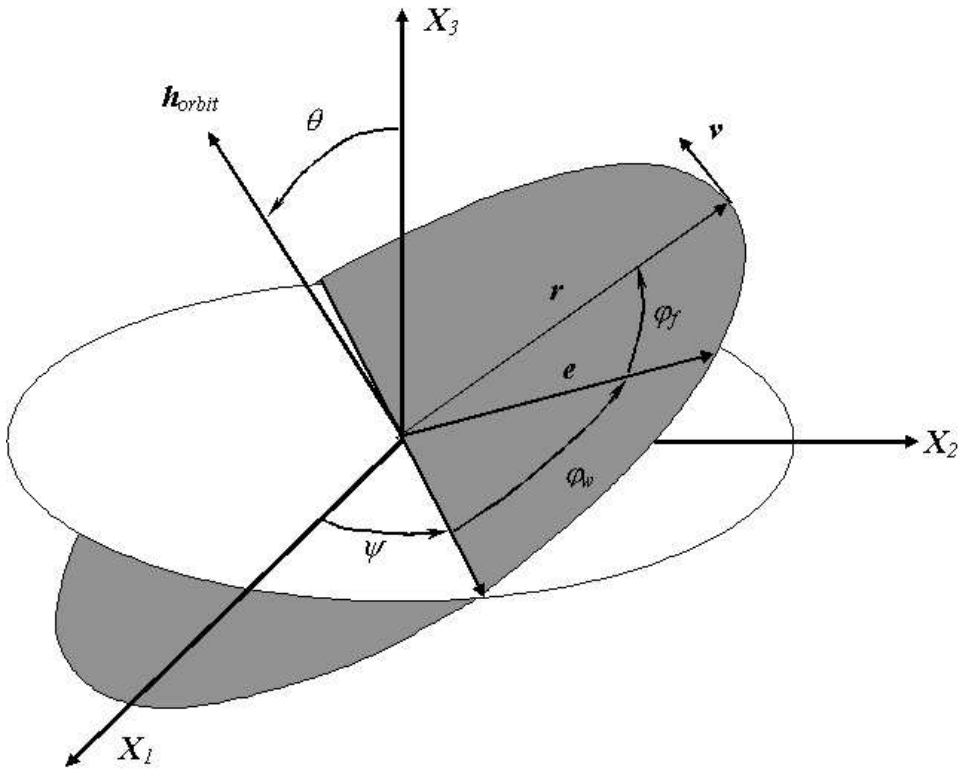

Figure 4.4: Orbital Elements

where $R(h_1, h_2, h_3)$ is the rotation matrix corresponding to the Euler angles, $\kappa(h) = h_4$ and $\eta(h) = \sqrt{\frac{\mu}{h_4^3}}$. With regards to the Euler angle representation of $SO(3)$, it should be noted that the only mappings required in the code were $S^1 \times S^1 \times S^1 \to SO(3)$, and not the inverse, thus avoiding singularity issues.

Rewriting the equations of motion relative to a circular orbit and using a series expansion as in Section 2.4.4, one gets a form of London's equations [82],

$$\frac{dx}{d\gamma} = \frac{d}{d\gamma}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ 2v_2 + 3x_1x_2 + u_1 \\ -2v_1 + 3x_2 - 3x_2^2 + \frac{3}{2}(x_1^2 + x_3^2) + u_2 \\ -x_3 + 3x_2x_3 + u_3 \end{bmatrix}$$

which satisfies the linear controllability and quadratic expression required by the local planner. Here, the x_1 direction is opposite of the orbital velocity vector, x_2 corresponds to the radial vector, and x_3 corresponds to the orbit normal direction. After some investigation, the convergence radius δ of the local planner was found to be .2. The coasting time τ and the maneuver time T are implemented as nondimensionalized times, corresponding to the mean angular anomaly.

The transformation $\xi = (x, v) = \mathcal{F}_\delta(\gamma, \nu)$ then can map the maneuver automaton states and controls into these relative equations, where

$$\begin{aligned}\mathcal{F}_\delta : (\gamma, \nu) \mapsto \Big(&\kappa(h)^{-1}R^T(h_{prop})(\mathcal{F}_x(h_{next}) - \mathcal{F}_x(h_{prop})), \\ &\kappa(h)^{-1}R^T(h_{prop})\left[(\eta(h)^{-1}\mathcal{F}_v(h_{next}) - \mathcal{F}_v(h_{prop})) - \widehat{\mathbf{e}}_{x_3}(\mathcal{F}_x(h_{next}) - \mathcal{F}_x(h_{prop}))\right]\Big)\end{aligned}$$

where $\mathbf{e}_{x_3}$ is the unit vector in the x_3 direction, $\widehat{\mathbf{e}}_{x_3}$ represents the skew-symmetric operator on the unit vector, $h_{prop} = h + \mathbf{e}_{x_3} \cdot (\tau + T)$ is the propagation on the symmetry group due to motion on the reference trajectory, and $h_{next} = h_{prop} + \Delta h$.

4.7.2 Implementation

The algorithm was implemented in C++ on a 850 Mhz Pentium 3 computer running Redhat Linux 7.2.[9] As previously mentioned, the continuous portion of $\mathcal{H}$ and $\mathcal{U}(\gamma)$ must be discretized. As shown in Figure 4.5 and in Table 4.1, the continuous "state" space is approximated by 1800 nodes and the continuous input space is approximated by 130 possible nodes. Note that $K_{hf} = h_{f_4}/1.5$ in Table 4.1.

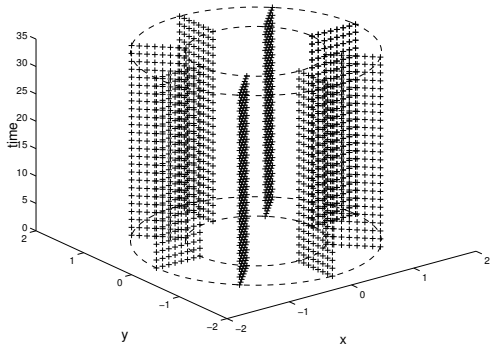

Figure 4.5: Grid representation of $\mathcal{H}$.

$\mathcal{H}$:		$\mathcal{U}(\gamma)$:	
$\mathbb{N}_T = \{0\}$		$\ell_{next} = 0$	
$h_1 = 0$	1 node	$\Delta h_1 = 0$	1 node
$h_2 = 0$	1 node	$\Delta h_2 = 0$	1 node
$0 \leq h_3 \leq 2\pi$	9 nodes	$-.1 \cdot K_{hf} \leq \Delta h_3 \leq .1 \cdot K_{hf}$	3 nodes
$.9 \leq h_4 \leq 1.6$	8 nodes	$-.1 \cdot K_{hf} \leq \Delta h_4 \leq .1 \cdot K_{hf}$	3 nodes
		$+\Delta h_f$ (when applicable)	1 node
0 d $\leq t \leq$ 1374 d	25 nodes	$0 \leq \tau \leq 1.32$	13 nodes

Table 4.1: Discretized Space

Note that one modification was made to the dynamic programming routine to compensate for coarse grids. This modification involved using the final state input when

[9] This is excepting the derivation of the local planner equations, which are derived through a Maple code and imported into the C++ implementation.

available if the alternative is to get stuck at a point within a grid element's spacing of the final condition. When implemented correctly, this does not affect the convergence of the dynamic program, but allows for feasible solutions (solutions with little final error) to be found with a coarse grid.

4.7.3 Results

Preprocessing

The series calculation was completed to second order and took 210 seconds using Maple 5.4 under Windows ME on the aforementioned computer. The cost distribution was calculated using the nominal final state $h_f = [0, 0, \pi/2, 1.5]$ at $t = 0$ and propagating with time. The result is seen in Figure 4.6, which took 24 iterations in 9 hours to compute.

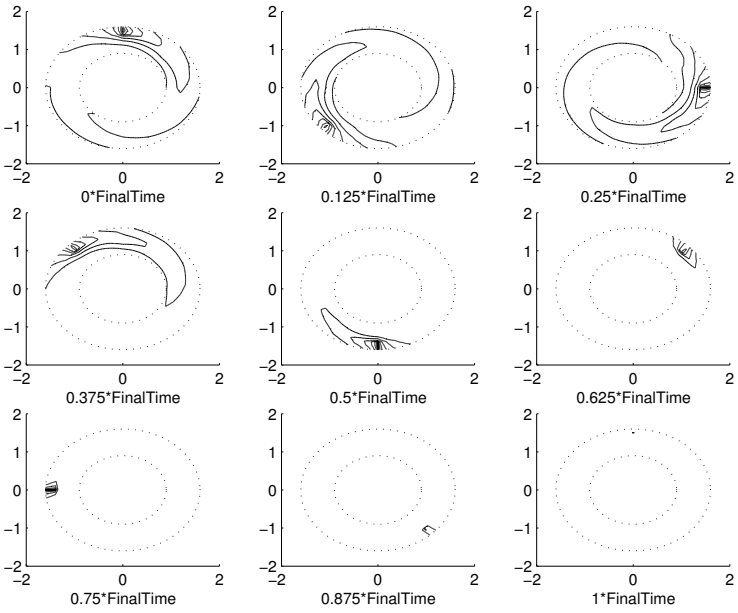

Figure 4.6: Optimal cost distribution.

Online Calculations

The actual problem was set up as in Figure 4.7a, where Earth and Mars are in circular coplanar orbits of radii 1 au and 1.5237 au, respectively. In this example, Mars leads Earth initially by $\pi/2$ and an upper bound of 626 days was put on the transfer. The algorithm took 3.52 seconds to find a 5 step (5 circle-to-circle maneuver) solution, with the coast times and final maneuver locations defined by the discrete controls in Table 4.2.

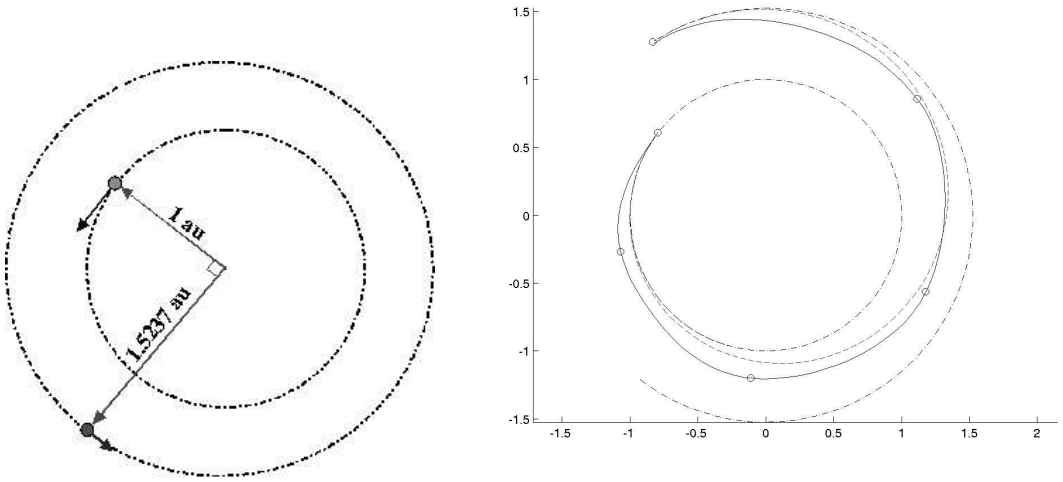

Figure 4.7: Earth-to-Mars Setup and Final Trajectory

Step (k)	τ_k	ℓ_k	$h_{next_{k1}}$	$h_{next_{k2}}$	$h_{next_{k3}}$	$h_{next_{k4}}$	t_k
0	n/a	0	0	0	0.918	1	n/a
1	0	0	0	0	1.818	1.10	0 d
2	0.33	0	0	0	3.048	1.20	58 d
3	0.22	0	0	0	4.268	1.30	148 d
4	0	0	0	0	5.368	1.41	241 d
5	0.55	0	0	0	0.577	1.52	327 d

Table 4.2: Discrete Steps

From a start point on the symmetry group of $h = [0, 0, .918, 1]^T$, the discrete controls direct a series of coasts and transitions to traverse 5.94 radians in 478 days to rendezvous

with Mars. The solid line in Figure 4.7b shows the final trajectory with the position at each step marked by a circle. The final error in position and velocity was only 0.9%. As it can be seen that the discrete controls directly target the final position, the error is entirely due to the truncation error in the local planner series expansion. If the series were extended beyond second order, the error would be lessened. The total control profile can be seen in Figure 4.8, where the coasting and maneuvering segments can be clearly seen. These controls, corresponding to the acceleration due to thrust in the orbital frame, are clearly very low thrust, as the acceleration of gravity at the Earth's surface is $9.8 \times 10^{-3}\ km/s^2$, orders of magnitude higher than the actual control acceleration.

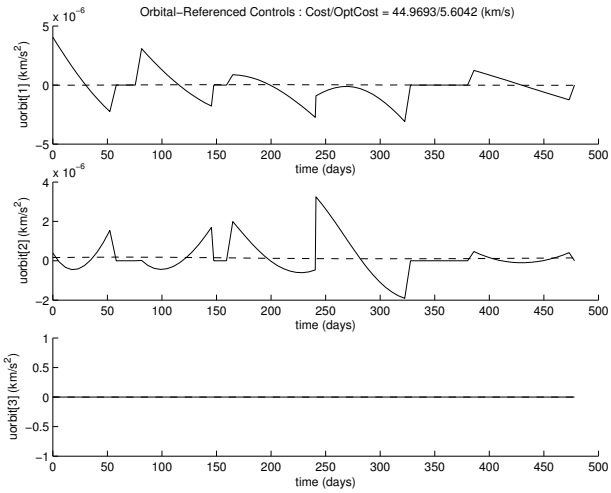

Figure 4.8: Earth-to-Mars control history in orbit-referenced frame// u_1: tangential thrust, u_2: radial thrust, u_3: normal thrust

Comparison to Optimal Control

In Figure 4.7b another transfer can be seen as a dashed line. This transfer corresponds to a locally optimal trajectory found by the optimization software Varitop [131, 93], developed by the Jet Propulsion Laboratory. It can be seen, that, even when limited to circular trajectories only and a rough input discretization, our algorithm provides a rough shape approximation of the optimal route. Unfortunately, the same discretization and use of circular orbits alone results in a suboptimal control cost of 45.0 km/s in comparison to 5.6

km/s [10]. The Varitop control history, seen as the dashed line in Figure 4.8, attests to this, showing a much smaller control magnitude. The discrepancy is due to two factors. First, the input set in the dynamic programming routine used a very coarse mesh, limiting the maneuvering options of the vehicle on the MA and impairing optimality as a result. Second, the MA only used circular reference trajectories. Inclusion of other reference trajectories could enable the MA to model a larger range of motions and thus allow for an MA-optimal trajectory to better approximate an optimal solution on Σ. While the computation time of the given Varitop solution was of the same order as that of our algorithm, such run times are attainable only with a good guess of the initial values of the costates. When this guess is poor, it is unlikely any solution will be found.

4.8 Conclusions

This chapter has greatly expanded on the framework of the maneuver automaton first introduced by Frazzoli through use of reference trajectories rather than relative equilibria, scaling symmetries, and time dependent final conditions. This new structure allows for maneuvers with an infinite set of possible final conditions. When implementing this new type of maneuver in the form of the online local planner of Chapter 3, we have been able to, for the first time, provide an overall solution with convergence guarantees, controllability with a single trajectory primitive, and feasible solutions with a coarser grid. While applied to an orbital example, this technique has been generally formulated to be applicable to a wide variety of systems, whether vehicular in nature or otherwise.

[10] The cost given corresponds to $\int \sqrt{u^T u} dt$

chapterPlanning with Obstacles

4.9 Introduction

Planning a trajectory in an environment with obstacles has long been an interesting research problem to roboticists and computer scientists, with a primary focus on kinematic motion problems [75]. These methods fall into two main categories, incremental searching methods and roadmap methods, both of which find collision-free paths on the configuration space (the configuration manifold modulo all points for which collisions exist). Incremental searching methods perform an iterative search to try to connect the initial configuration to the goal configuration. Examples of these include dynamic programming, such as that introduced in Chapter 4 and potential field methods. Potential field methods are based on an artificial potential field that provides and attraction to the goal and repulsion from obstacles. The solution is then found by a gradient descent through the field. This is a feedback control in the sense that the control is recomputed in light of the state at every step in time, as compared to the open-loop formulations most common in motion planning problems. The difficulty in this method lies in the possible existence of local minima. One method for addressing this is to generate a "random walk" to escape from such cases [9]. Another method is specifically defined a potential function without local minima called a "navigation function," but computing such a function in the general case is as difficult as solving the motion planning problem for all initial conditions [111]. On the other hand, roadmap methods solve the path planning problem by generating a graph of collision-free connecting paths spanning the configuration space. These "roadmaps" can be created through cell decomposition [26], Voronoi graphs [32], visibility graphs [84], and probabilistic methods [65]. The online problem then reduces to a graph traversal along with planning on and off the roadmap. These motion planning algorithms are evaluated in terms of completeness and computational complexity. An algorithm is said to be *complete* if it returns a valid solution to the motion planning problem if and only if a feasible solution exists. Complexity is also a major issue in path and motion planning. The "generalized mover's problem" involving path planning for a 3-D linkage of polyhedral parts among polyhedral obstacles was shown by Reif [110] to be PSPACE-hard, where PSPACE is the complexity class includ-

ing decision problems for which answers can be found with resources, such as memory, which are polynomial in the size of the input. His analysis provides evidence that any complete planner will run in exponential time in the number of degrees of freedom. Since then, Schwartz and Sharir [114] as well as Canny [26] have developed general purpose path planning algorithms with execution time doubly and singly exponential in degrees of freedom, respectively [75].

The high computational cost of these deterministic complete path planners has motivated the development of randomized path planning algorithms that are *probabilistically complete*, i.e., that the probability of finding a path from the initial to final conditions converges to one if a feasible path exists. One successful example of this is the probabilistic roadmap (PRM), which generates a graph based on random points in the configuration space in an offline phase and plans motions locally to, from, and traversing the roadmap online. Some disadvantages to this approach lay in the reliance of the method based upon precalculation of the roadmap and the lack of ability of the basic roadmap to incorporate system dynamics. These issues were addressed by the introduction of the rapidly exploring random trees (RRTs) of LaValle and Kuffner [78, 79, 71]. The RRT grows a tree of feasible trajectories from the initial condition, or root node. Each node, or waypoint, on the tree has possible trajectories branching from it. The tree incrementally builds itself in random directions, node by node, until the final condition is met (within accuracy bounds).

Frazzoli [41, 44] demonstrated that the Maneuver Automation structure could be used with randomized planners as well. By using the cost as a metric, modifying the method of node choice in the expansion, and using a dynamic programming algorithm for his base planner, he provided a means to address planning in the presence of both static and dynamic obstacles. The structure of the planner developed in this dissertation also has a great potential for use in a randomized scheme. This scheme is based on the aforementioned RRTs and is the focus of this chapter. This chapter is organized such that Section 2 discusses the RRT and its variations relating to the maneuver automaton in detail as well as its convergence properties. Here, a new variation of the RRT utilizing the maneuver automaton and addressing time dependent final conditions is presented. Section 3 discusses error correction in the RRT and a novel way for dealing with errors passed on from the lower-level obstacle-free planner on the maneuver automaton. The

next two sections then provide an algorithm description detailing the way the code is organized and an example solving for a spacecraft landing on the asteroid Ida.

4.10 Rapidly Exploring Random Trees

4.10.1 Background

The idea of this method is to incrementally build a *tree* of feasible trajectories to efficiently explore a reachable space, where a *tree* is defined as follows:

Definition 4.10.1 (Tree) *A tree is a directed graph with no cycles in which all nodes have one incoming edge (excluding the root), originating from the* parent*, and an arbitrary number of outgoing edges leading to* child *nodes.*

The basic RRT algorithm [78] acts as described below, where $\mathcal{C}_\Xi$ is the (obstacle-free) configuration space of Ξ.

(i) *INITIALIZE* tree root at the initial state $\xi \in \mathcal{C}_\Xi$

(ii) *CHOOSE* a random point ξ_r in $\mathcal{C}_\Xi$ toward which the tree will be expanded.

(iii) *EXTEND* the tree from the closest node (relative to a given metric) toward the random point. The node added at the end of this step, if one is added at all, is a *milestone.*

(iv) *REPEAT ii*) and *iii*) until the final condition is met.

In its basic incarnation, *EXTEND* takes the following steps

(i) Pick the closest node $Node_A$ by a standard Euclidean distance metric.

(ii) Pick a constant input among several that takes the system closest to ξ_r after a given time δ_t.

(iii) Propagate the system equations of motion by δ_t using the chosen input. If no collision is found at the new point, add a new child node to the $Node_A$ with the propagated state.

This method is shown to be probabilistically complete as the number of milestones goes to infinity. Figure 4.9[11] demonstrates how a single RRT for a simple holonomic 2-D system can efficiently explore a space.

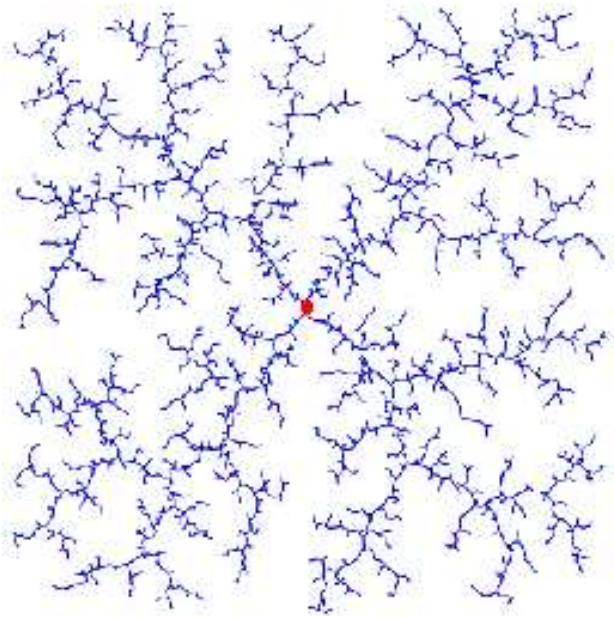

Figure 4.9: RRT Exploration

A slight twist on the basic RRT [71] is the RRT-Connect algorithm, which is more "greedy" by two modifications. First, it puts a bias on the random point generator such that, for a given percentage of the time, it targets the final condition. Second, it iterates step *iii*) of $EXTEND$, letting the child be the new $Node_A$, until either a collision occurs or the new child is further away from ξ_r than $Node_A$. This also is shown to be probabilistically complete in the same manner.

We can now take this concept and apply it to motion planning using the maneuver automaton structure defined in Chapter 4. Remember, from every point $\gamma \in \mathcal{H}$, there exists a reachable set $\mathcal{R}_{n_T^{max}}$ for which a control profile μ exists that can control the system from the state γ to any state in $\mathcal{R}_{n_T^{max}}(\gamma)$. Thus, for every $\xi = \mathcal{F}(\gamma) \in \mathcal{F}(\mathcal{H})$, the μ-reachable set is $\mathcal{F}(\mathcal{R}_{n_T^{max}})$. Thus, given a set $S \subset \mathcal{H} \times \mathbb{R}$, the μ-reachable set of S is $\mathcal{R}^{\mu}(S) = \cup_{(\xi,t)=\mathcal{F}(\gamma)\in S}\mathcal{F}(\mathcal{R}_{n_T^{max}}(\gamma))$. As the maneuver automaton represents a subset of

[11] Figure courtesy of *The RRT Page*, http://msl.cs.uiuc.edu/rrt

the feasible motions of the system, $\mathcal{R}^{\mu}(S) \subseteq \mathcal{R}^{FULL}(S)$, where $\mathcal{R}^{FULL}(S)$ is the reachable set under the full set of feasible motions of the system. Thus, unless it can be established that $\mathcal{R}^{\mu}(S) = \mathcal{R}^{FULL}(S)$, the RRT, when applied using μ over the maneuver automaton structure, can only be complete over $\mathcal{R}^{\mu}(S)$.

Frazzoli's approach [41, 44] addressed only time-independent final states. He replaced the Euclidean metric with the cost-to-go to the random point ξ_r and restricted $\xi_0, \xi_f, \xi_r \in \mathcal{R}^{\mu}(S) \subseteq \mathcal{F}(\mathcal{H})$. The only other changes were to the $EXTEND$ routine, which is given below:

(i) Pick the closest node $Node_A$ by cost-to-go metric.

(ii) Get the appropriate μ from the obstacle-free planner to control the system from $Node_A$ to ξ_r

(iii) Propagate the system equations of motion according to μ, checking for collisions along the way. If no collisions are found, add a new child node to the $Node_A$ with the new final state.

(iv) If collision found, let $Node_A$ be the next closest node and repeat (Unless no nodes left in tree). If no collision found and final condition not met, let $Node_A$ be the last node added to the tree and $\xi_r = \xi_f$ and repeat ii) through iv).

The process of cycling through the nodes is necessary to accommodate for obstacles evolving with time when time is not part of the state. A completeness proof of this approach is in [41]. Note that this method is "greedy" in that the original ξ_r chosen is forced to be ξ_f and, from every successfully connected random point, the $EXTEND$ routine will automatically try to connect to ξ_f.

4.10.2 A New Approach

In order to accommodate a time-dependent final condition, time must also be considered a state in the RRT implementation. One advantage in doing this is that it alleviates the need for cycling through the nodes in the tree to address time-varying obstacles. With slight modification of the "greedy" steps, the proposed algorithm is as follows:

(i) *INITIALIZE* tree root at the initial state $\xi_0 \in \mathcal{C}_\Xi$

(ii) *CHOOSE*:

- 1st iteration - Let $\xi_r = \xi_f$, the goal state, propagating with time.
- After 1st iteration - a random (with a small Goal Bias) point ξ_r in $\mathcal{C}_\Xi$ toward which the tree will be expanded.

(iii) *EXTEND* the tree from the closest node (relative to a given metric) toward the random point. The node added at the end of this step, if one is added at all, is a *milestone*.

(iv) *REPEAT* *ii*) and *iii*) until the final condition is met.

with the *EXTEND* routine

(i) Pick the closest node $Node_A$ by cost-to-go metric. Let $Node_{A'} = Node_A$.

(ii) Get the appropriate μ from the obstacle-free planner to control the system from $Node_A$ to ξ_r

(iii) Propagate the system equations of motion according to μ. Add a new child nodes successively at given intervals along the path (provided they are on a reference trajectory) until μ complete or a collision found.

(iv) If no collision found and the final condition is not met:

(a) Let $\xi_r = \xi_f$, propagating with time.

(b) Let $Node_A$ be the last descendant of $Node_{A'}$ that has not been used for replanning, if it exists, and repeat *ii*) through *iv*). If it does not exist, end *EXTEND* routine.

This procedure is reflected in steps a through f of Figure 4.10, where ξ_0 is the left point, ξ_f is the point at the right, and the rectangle is an obstacle. In the first step, the algorithm tries to use the obstacle-free planner to connect to the final point. The tree is then extended along this path as it is collision-checked. A collision is found, so a random point is generated. The obstacle-free routine is then called again to connect the closest

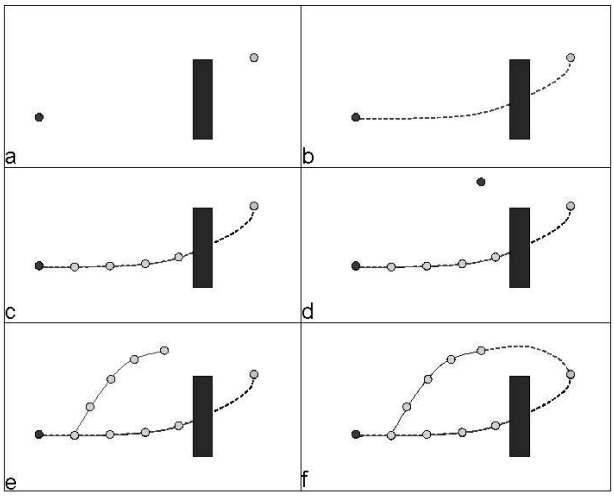

Figure 4.10: New RRT Variation

node to the random point. New nodes are added along this path as collision checking is done. No obstacles were encountered, so $EXTEND$ tried to connect to the final state. No obstacles are in this path, so after the tree is extended along this path (step iii) of $EXTEND$), the algorithm would terminate.

4.10.3 RRT Convergence

All of the aforementioned methods are probabilistically complete over the reachable space of the embedded planner. The key differences in establishing this completeness lay in the assumptions on the obstacles and the definition of the state. In the nominal RRT, obstacles are assumed fixed in the state space, where the state space may or may not include time. In both the variants incorporating the maneuver automaton, time-varying obstacles are addressed, but Frazzoli's version did not include time as a state whereas the proposed variant does. This distinction is important, as , when time is included as a state, the time-varying obstacles become fixed in the state space and the basic RRT analysis applies. That analysis is the focus of this section. The lack of time as a state in the Frazzoli planner [41, 44] necessitated cycling through all of the nodes in the tree as well as a different analysis.

To show the proposed method's convergence properties, we follow the analysis of the

RRT in [78], where the proofs of Theorems 4.10.2 and 4.10.3 can be found. Assume no two RRT milestones lie within a specified $\epsilon > 0$ of one another for the given metric (i.e., the cost-to-go) and the time between nodes is bounded away from zero. Suppose further that (ξ_0, t_0) and (ξ_f, t_f) lie in the same connected component of a bounded, open, p-dimensional connected component of a p-dimensional state space. In addition, stipulate that there exists a sequence of obstacle-free controls $\mu_1, \mu_2, \ldots \mu_k$, that, when applied to (ξ_0, t_0), yield a sequence of states $(\xi_0, t_0), (\xi_1, t_1), \ldots (\xi_{k+1}, t_{k+1}) = (\xi_f, t_f)$, all of which lie in the same open connected component of the space. Under these assumptions, we can state the following theorem regarding probabilistic completeness of the algorithm:

Theorem 4.10.2 *The probability that the RRT initialized at (ξ_0, t_0) and will contain (ξ_f, t_f) as a node approaches one as the number of random point milestones*[12] *approaches infinity.*

Let $\mathcal{A} = \{A_0, A_1, \ldots, A_k\}$ be a sequence of subsets of $\mathcal{F}(\mathcal{H})$, referred to as an *attraction sequence*. Let $A_0 = \{(\xi_0, t_0)\}$. The remaining sets must be chosen with the following rules. For each A_i in $\mathcal{A}$, there exists a basin $B_i \subseteq \mathcal{F}(\mathcal{H})$ such that the following hold:

(i) For all $x \in A_{i-1}$, $y \in A_i$, and $z \in \mathcal{F}(\mathcal{H})/B_i$, the metric, ρ, yields $\rho(x, y) < \rho(x, z)$.

(ii) For all $x \in B_i$, there exists an m such that the sequence of controls $\{\mu_1, \mu_2, \ldots, \mu_k\}$ selected by the $EXTEND$ routine will bring the state into $A_i \subseteq B_i$

(iii) $A_k = \{(\xi_f, t_f)\}$

Furthermore, let p be defined such that

$$p = \min_i \frac{\text{measure}(A_i)}{\text{measure}(\mathcal{F}(\mathcal{H})_{free}))}$$

where $\mathcal{F}(\mathcal{H})_{\text{free}}$ is $\mathcal{F}(\mathcal{H})$ modulo any obstacles. This corresponds to a lower bound on the probability that any particular state will fall in a particular A_i. Given these definitions, a criterion for exponential convergence in the number of random point milestones can be established.

[12] Random point milestones are milestones for which ξ_r was generated using a random point generator.

Theorem 4.10.3 *If an attraction sequence of length k exists, for a constant $\delta \in (0, 1]$, the probability that the RRT finds a path after n random point milestones is at least $1 - e^{-np\delta^2/2}$, in which $\delta = 1 - k/(np)$.*

4.11 Error Correction and the RRT

The framework of the RRT also allows for correction of errors from the underlying planning algorithm. For the initial incarnation of the RRT, a constant input was chosen from a finite set and was highly unlikely to control the system to the intended final state. As a result, the actual final state (as found by integrating the system under the input) is that which is stored as the new node state rather than the targeted state. Thus, replanning from that node takes into account the error correction. For implementation on the maneuver automaton, the characteristics of these errors become important, as replanning can only occur from nodes on a reference trajectory. Thus, when integrating along the trajectory, nodes would only be added to the tree where the state matched the reference trajectory within an acceptable error. Figure 4.11 demonstrates how this correction would work. In this figure, horizontal lines represent reference trajectories, the dashed lines and $\times$'s represent the planned trajectory and node locations, while the solid lines and circles represent the actual trajectory and node locations. In the figure, a trajectory is planned from node 0, with a new trajectory planned at node 1. There is an error correction e between the planned trajectory and node 1. This error correction, when coupled with the last step of *EXTEND*, has an effect similar to a receding horizon planner, causing that step to iterate until the final condition is reached or no untried nodes are left. While this methodology is useful in practice (it was consistently effective for the example in this chapter), it is not a complete error correction, as e is only the projection of the total error onto the reference trajectory. Thus, there exists an uncorrected component of the error that is propagated further. Further analysis is needed to better understand the overall effectiveness of this method. Of course, other methods of error mitigation exist as well. Frazzoli [44] addressed the issue by assuming use of a feedback controller that forces the system to a reference trajectory, while error due to the local planner as in Chapters 3 and 4 can be reduced by simply increasing the terms in the series expansion.

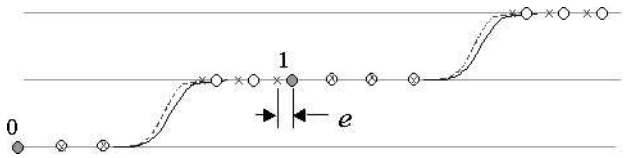

Figure 4.11: Error Correction

4.12 Algorithm description

The algorithm is implemented in a C++ object-oriented fashion using the motion strategy library [77]. The overall structure is depicted in Figure 4.12, where the functionality is broken into three major objects: the RRT-based planner, the underlying obstacle-free planning algorithm, and the model class. Their descriptions are given below:

RRT-based planner

This is the highest level object and acts as the user interface. Given constraint geometry as well as initial and final conditions, it outputs feasible trajectories and control profiles. Using model information such as state transformations and constraints, it sends initial and final conditions to the obstacle-free planning algorithm for solutions. Collision-checking is done using the Proximity Query Package [125].

Obstacle-free planning algorithm

This object contains two algorithms, the dynamic programming routine for finding the discrete-time controls and the root-finding method used to define the continuous-time control inputs. The only portion of this object devoted to preprocessing calculates the cost distribution.

Model class

Contains all information relating to the dynamics of the system as well as the state transformations and symmetry actions. The local planner series, calculated in Maple analytically, is stored here, as are the incremental cost functionals.

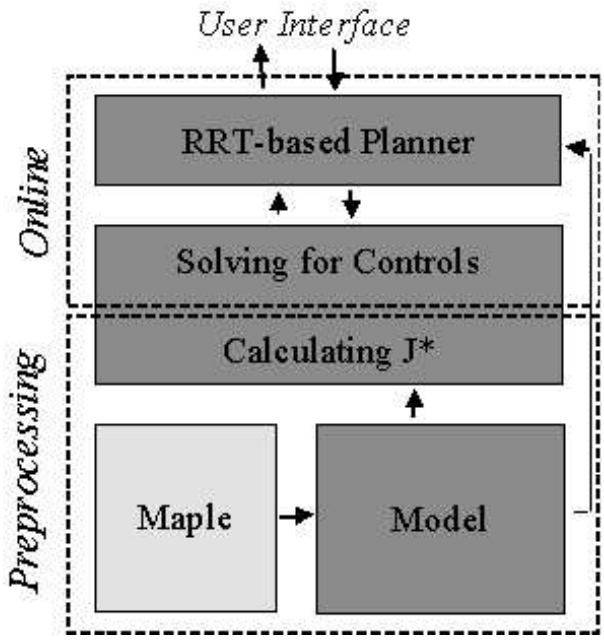

Figure 4.12: Coding Structure

4.13 Asteroid Landing Example

4.13.1 Problem Set-up

The example chosen here is that of a spacecraft landing on the asteroid Ida. This example is inspired by the February 12, 2001 landing of the NEAR-Shoemaker spacecraft on the asteroid Eros and the Rosetta mission expected to land a craft on the comet Wirtanen in 2012. Ida is an S-class (stony or stony-iron) asteroid in the asteroid belt orbiting at 2.86 au from the sun. Discovered in 1884, it falls within the class of large asteroids, having a mass of approximately 5×10^{16} kilograms, having dimensions of $60 \times 25 \times 19$ kilometers (comparable to the size of Cape Cod) and a rotational period of 4 hours, 38 minutes. Ida is unique in that it is the only known asteroid to have a natural satellite (Dactyl, discovered by the Galileo spacecraft in 1993). For the purpose of this example, we made simplifying assumptions by geometrically modeling the asteroid as a 60 km long cylinder

with radius 12 km and modeling the gravity as a Newtonian point source [13] The setup of the problem is seen in Figure 4.13, where the initial position of the spacecraft is 18 km above the surface of the asteroid with the final condition of "landing" at a point just off the surface on the other side of the asteroid. Additionally, artificial constraints are imposed to limit motion to between 19.1 and 38.3 km of the center of mass. The goal is once again a minimum energy transfer, limiting the motion to Ida's plane of rotation and circular reference trajectories. Thus, the setup of the dynamics and the transformations are equivalent to that in Section 4.7, modulo a different central body mass. The discretization of the automaton space follows Table 4.3.

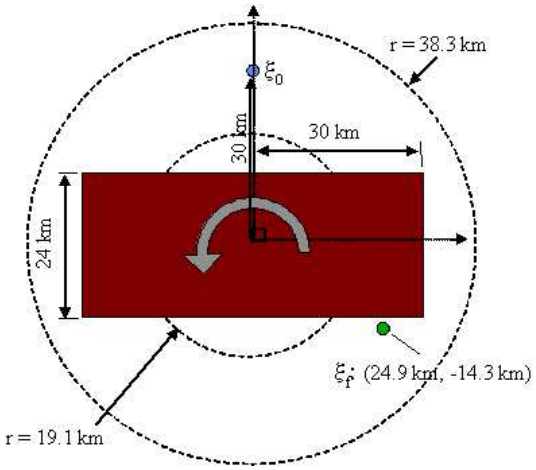

Figure 4.13: Ida Landing Setup

Note that, rather than sample the reachable set $\mathcal{R}^u(S)$, which is computationally impractical to define, sampling was done over $\mathcal{F}(\mathcal{H})$, where $\mathcal{F}(\mathcal{H})/\mathcal{R}^u(S)$ is comprised of obstacles and unreachable space.

4.13.2 Results

The preprocessing was carried out as in Section 4.7, with the cost distribution (see Figure 4.14 calculated in 24 iterations as before but in roughly twice the time due to

[13] This is not an accurate model of such an irregularly shaped object. Rather, it is a simplification mirroring the example in Chapter 4. A more accurate model, once developed, could also be incorporated by this chapter's framework, albeit with a smaller symmetry group.

$\mathcal{H}$:		$\mathcal{U}(\gamma)$:	
$\mathbb{N}_T = \{0\}$		$\ell_{next} = 0$	
$h_1 = 0$	1 node	$\Delta h_1 = 0$	1 node
$h_2 = 0$	1 node	$\Delta h_2 = 0$	1 node
$0 \leq h_3 \leq 2\pi$	9 nodes	$-.1 \cdot K_{hf} \leq \Delta h_3 \leq .1 \cdot K_{hf}$	3 nodes
$1 \leq h_4 \leq 2$	11 nodes	$-.1 \cdot K_{hf} \leq \Delta h_4 \leq .1 \cdot K_{hf}$	3 nodes
		$+\Delta h_f$ (when applicable)	1 node
0 hr $\leq t \leq$ 14.8 h	25 nodes	$0 \leq \tau \leq 0.785$	9 nodes

Table 4.3: Ida Discretized Space

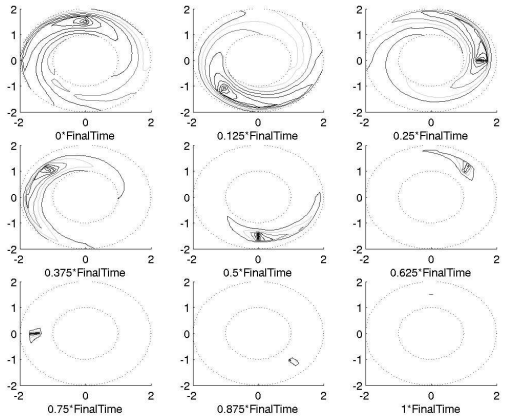

Figure 4.14: Precalculated Approximate Cost

the larger grid. For the online calculation, an upper bound of 46 hours was placed on the transfer time. From a start point at $\xi = [0, 1.5, 0, -0.8165, 0, 0]^T$, where the units are referenced to a circular orbit at 20 km, the randomized algorithm grows a tree as seen in Figure 4.15, the result of 11 randomized planner iterations. The results of these iterations are given in Tables 4.4 and 4.5 .

Not surprisingly, iteration 0 (the collision-free planner result) had a collision. While 6 of the random samples fell in the unreachable space, the others expanded the tree, with the final state reached at iteration 11. The "Greedy Loops" column in the table refers to the fact that the error correction mentioned in section 4.11 can cause a loop in the *EXTEND* routine. This occurred in step 11, where the *EXTEND* algorithm

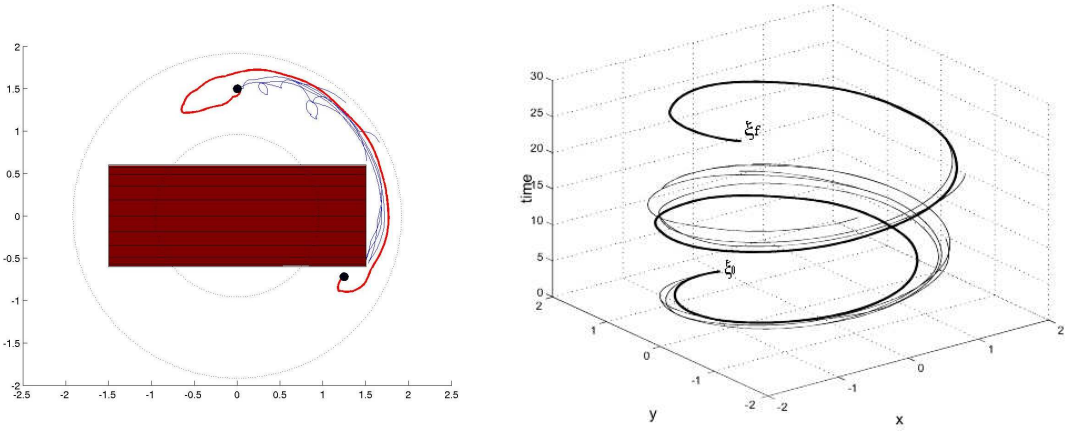

Figure 4.15: Ida Random Tree and Final Path, static relative frame (left), inertial frame with time variation (right)

looped an extra time to reach the final state. The overall solution was computed in 64.2 seconds, with the final path composed of 12 circle-to-circle maneuvers and a final state of $[-1.31, -0.60, 0, 0.34, -0.76]$ at 12.3 hours. This is an error of .03 percent in position and .2 percent in velocity from the final condition. Snapshots of the trajectory are shown in Figure 4.16. It can be seen that the spacecraft starts on an inner trajectory and then switches to an outer trajectory and to let the asteroid rotate underneath it before returning to an inner trajectory to reach its final destination.

The control history is shown in Figure 4.17. Note that this trajectory once again corresponds to a very low thrust mission as the acceleration of gravity at the Earth's surface is $9.8 \times 10^{-3}\ km/s^2$.

Of course, this is a planner based on randomized methods, and, as such, every solution to this algorithm will be slightly different with different run times. A batch of 50 runs of the aforementioned example were completed and histograms of the randomized planner iterations and run times are seen in Figure 4.18. The median run time was 95.9 s with 78% of the cases taking less than 200 s. The median number of randomized planner iterations was 11, with 90% taking less than 40 iterations. Interestingly, the execution time does not directly correspond to the number of iterations. This discrepancy is due to the number of "greedy loops" executed. The case with the maximum run time took 79 iterations with a total of 74 greedy loops in 1484 seconds, while the next longest case

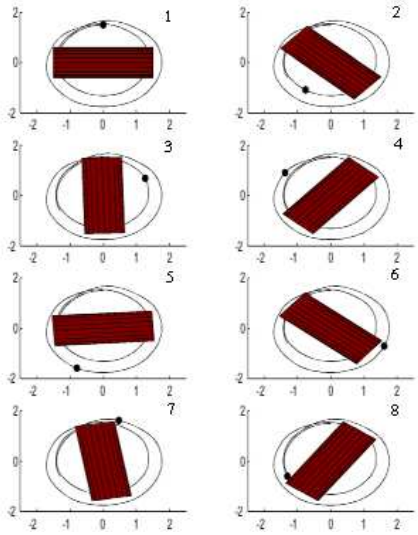

Figure 4.16: Ida Results: Snapshots of Trajectory

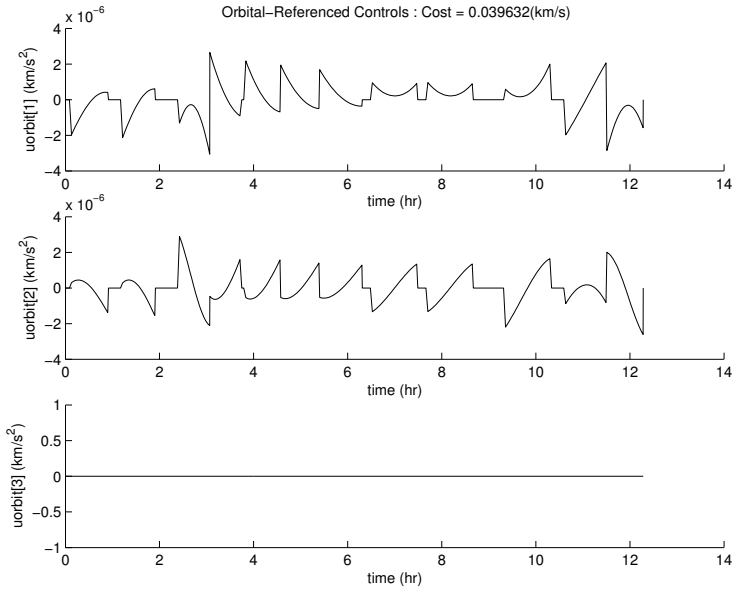

Figure 4.17: Ida Control History in Orbit-referenced Frame

Iteration	Start point	Start time	Final Point ($t = 0$)	Upper time bound
0	$[0, 1.5, 0, -0.82, 0, 0]$	0 h	$[1.25, -0.72, 0, 0.42, 0.72, 0]$	14.8 h
1	$[0, 1.5, 0, -0.82, 0, 0]$	0 h	$[0.94, -0.75, 00.57, 0.71, 0]$	5.3 h
2	$[-1.60, -0.53, 0, 0.24, -0.73, 0]$	1.7 h	$[-1.46, 0.31, 0, -0.17, -0.80, 0]$	8.7 h
3	$[1.68, 0.13, 0, -0.06, 0.77, 0]$	4.5 h	$[1.25, -0.72, 0, 0.42, 0.72, 0]$	14.8 h
4	$[-1.34, 0.86, 0, -0.43, -0.67, 0]$	.9 h	$[-1.13, 0.78, 0, -0.48, -0.70, 0]$	4.3 h
5	$[0, 1.5, 0, -0.82, 0, 0]$	0 h	$[0.76, 0.77, 0, -0.68, 0.68, 0]$	3.8 h
6	$[-1.34, 0.86, 0, -0.43, -0.67, 0]$	.9 h	$[-1.78, 0.45, 0, -0.18, -0.72, 0]$	11.4 h
7	$[0, 1.5, 0, -0.82, 0, 0]$	0 h	$[-0.54, 1.64, 0, -0.72, -0.24, 0]$	.8 h
8	$[-1.27, -0.74, 0, 0.42, -0.71, 0]$	1.7 h	$[-1.19, 0.51, 0, -0.34, -0.81, 0]$	9.8 h
9	$[-1.27, -0.74, 0, 0.42, -0.71, 0]$	1.7 h	$[-0.49, -1.04, 0, 0.84, -0.40, 0]$	4.1 h
10	$[-1.01, -1.40, 0, 0.62, -0.45, 0]$	2.1 h	$[-1.11, -1.48, 0, 0.594, -0.44, 0]$	11.2 h
11	$[-0.24, 1.48, 0, -0.81, -0.13, 0]$	.1 h	$[-0.17, 1.35, 0, -0.85, -0.11, 0]$	10.8 h

Table 4.4: Randomized Planner Iterations: Initial and Target points

took 67 iterations in only 853 seconds because only 16 greedy loops were necessary. These times should be viewed in the context that a single call to the obstacle-free planner took roughly 3 seconds. In every case tested, the algorithm successfully found a solution.

4.14 Conclusion

This chapter presented a new variant of the RRT for use with a maneuver automaton-based planner. By including time as a state it is able to accommodate time-varying obstacles and final conditions. This method is shown to be probabilistically complete, finding a solution with a probability of one as the number of iterations goes to infinity. This method was then applied to the example of a spacecraft landing on the asteroid Ida, for which analysis of a large batch or runs was completed. This method showed itself to be reliable with typical run times of less than 3 minutes. While the randomized method shown is not optimal, there exist methods to refine the tree to increase optimality [41].

Iteration	End Condition	Greedy Loops
0	Collision	0
1	Not Reachable	0
2	Not Reachable	0
3	Collision	0
4	Not Reachable	0
5	Not Reachable	0
6	Collision	1
7	Collision	1
8	Not Reachable	0
9	Not Reachable	0
10	Collision	1
11	Connected	2

Table 4.5: Randomized Planner Iterations: *EXTEND* Results

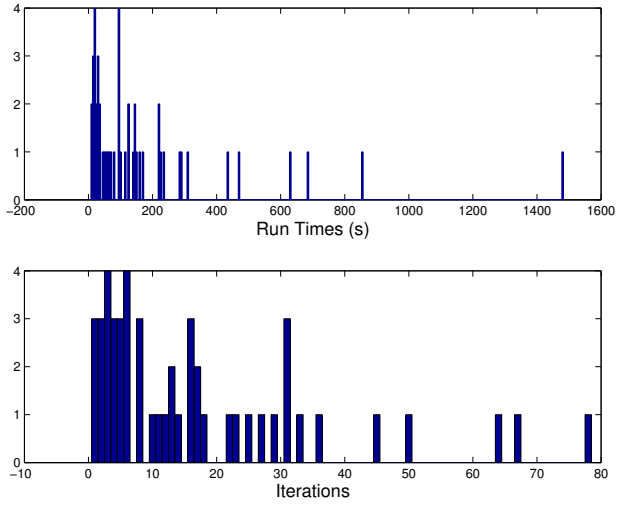

Figure 4.18: Ida Histogram of 50 Cases

Chapter 5

Conclusions

This thesis has presented a hierarchical global motion planning algorithm applicable to a large class of systems that addresses some of the key challenges in motion planning. The algorithm is complete in environments both cluttered and uncluttered with fixed and moving obstacles. It also provides for local control of systems when linearly controllable as well as global control. Gains in speed and computational efficiency are obtained through the hybrid system representation of the dynamics and the use of power series in local planning. Efficiency with regards to cost was obtained through optimization on the maneuver automaton. Finally, a dynamic environment was addressed by incorporating moving targets and obstacles.

5.1 Summary

Chapter 2 introduced the framework of mechanical control systems and some of the key properties enabling this motion planning framework to operate effectively. In particular, the concepts of symmetry, trajectory primitives, and reference trajectories were introduced. We then showed how the dynamics of various vehicle systems could be cast into these concepts.

Chapter 3 provided one of the key developments of this dissertation, presenting novel local complete planning algorithms based on power series. These algorithms provide guaranteed solutions for motion locally about reference trajectories. Variants included

series-based solutions to the minimum energy problem as well as feasible solutions using base functions. Convergence and performance of this method are presented using an academic one dimensional example and a PVTOL (Planar Vertical Takeoff and Landing) aircraft example.

Chapter 4 introduces a method for providing obstacle-free global motion planning, which included the other primary developments of this thesis. The computational complexity of the full dynamical system is avoided by limiting system evolution to motion along trajectory primitives. This framework, the Maneuver Automaton, is extended from its original incarnation [41] to include new scaling symmetries in time and time-dependent final conditions. In addition, the primitives used in the automaton have also been generalized, as motion now is allowed along reference trajectories (solutions to the differential equations of motion with certain other properties) and transitions which may or may not be predefined. When these transitions are defined using the local planner in Chapter 3, other useful properties of the planner can be shown. This includes controllability with a single reference trajectory, proof of convergence for that case, and the opportunity to obtain feasible solutions with a coarse grid by targeting the final condition. These properties were shown in an Earth to Mars orbital transfer example.

Chapter 5 introduces a randomized algorithm to address time-varying obstacles and time-dependent final conditions. As in [41] a variant of the Rapidly-Exploring Random Tree is proposed. This variant differs from that version, however, in its accommodation of time-dependent final conditions. This capability is included by adding time as a state, resulting in a simpler overall algorithm. Other elements introduced in this chapter are some greedy steps in the routine as well as an error correction scheme. The entire algorithm is then applied to the problem of landing a spacecraft on the asteroid Ida.

5.2 Future Directions

5.2.1 Addressing safety and real-time issues

In order for this work to be extended for use in an online vehicle guidance and control system, the issues of real-time computation [128] and safety [24, 25, 95] must be addressed. Frazzoli [41] has laid down groundwork in which the maneuver automaton

can be implemented to address safety and real-time issues. Speeding the computation of solutions without losing reliability is another challenge for which approaches other than dynamic programming must be investigated.

5.2.2 Expansion of Series-Based Planning

Other opportunities are available through further development of the series-based planning of Chapter 3. One way the usefulness of this method can be extended is by allowing for nonzero initial conditions. Another is in the investigation of the higher order terms in the series expansions. If a "nilpotency"-type condition can be established in which the higher order terms disappear, this planning method would become globally valid.

5.2.3 Optimal Primitive Choice

As mentioned in [41], choice of the best primitives to use in motion planning via the Maneuver Automaton is an open question. In our expanded version of the Automaton the number of these choices is drastically increased. This is through the use of reference trajectories over relative equilibria and the ability to use preprocessed maneuvers, those computed through an online local planner, or some combination of the two. The overall goal is to maximize the speed and flexibility of the planner while minimizing cost.

5.2.4 Optimal Trajectory Generation

Rather than sacrificing the benefits in complexity of using an automaton with few primitives, one could simply be content to find a "satisfactory" feasible solution. A method could then be developed to turn this feasible solution into an initial guess for an optimization routine. For instance, the feasible solution could be segmented and cast via collocation [53] into a nonlinear programming problem.

5.2.5 Incorporation of Sensors into Open Loop Control

One drawback to this development is that it is entirely open loop. The system dynamics are assumed to be completely known as are the geometry and motions of all obstacles

as well as other bounds on the state and velocity. While even under these assumptions the problem has not been simple, the motivating applications of this effort do not meet those assumptions. Therefore, sensor feedback is critical, and incorporating sensors in a more effective way than trajectory tracking feedback is an area of particular interest.

Replanning with new state estimates This is an extension of the error correction in Section 4.11 in which the updated state estimates are derived from sensor data rather than a forward integration of an accurate model. This approach is then akin to that of receding horizon control, also known as model predictive control. One key aspect of this problem is to provide stability and convergence guarantees [91, 94]. This approach has been applied to hybrid systems previously in [1] and also has been applied to the obstacle avoidance problem in a potential field-based method [100]. These developments provide some of the groundwork necessary to include receding horizon control into the control framework presented in this thesis.

Sensor-Based Motion Planning This is the problem of motion planning in unknown or only partially known environments. Thus, as sensor data updates the model of the environment, the motion planning algorithm must adapt. Attempts to solve this problem have included potential field methods [112, 69, 120], wall-following BUG algorithms [86, 85, 117], or roadmap methods [33, 34]. Unfortunately, these algorithms have difficulties addressing dynamics, 3-D environments, and convergence guarantees. As the random tree method presented in this thesis inherently addresses dynamic and 3-D environments, it provides an attractive alternative to aforementioned approaches. In using the random tree, choices must be made about which branches to explore without knowing if the environment beyond the sensor range will allow for a feasible motion [126]. The issue then expands to not only including sensor updates into the motion planning problem, but also providing guarantees of completeness of the algorithm.

Sensor-Dependent Goals/Final Conditions This algorithm was able to address time-dependent final conditions. However, the objective of many systems may not be time-dependent, but, rather, sensor dependent. The role of autonomous vehicles is, more often than not, as a sensor platform. In these cases, the final condition or goal of the motion planning algorithm should correspond to the goal of the sensing

mission. This goal may change depending upon the sensor data, which may reveal new priorities and targets as the algorithm is executed. Examples of this include exploration [4, 3], pursuit [72, 36], data-gathering [7], and changing priorities to follow "targets of opportunity."

5.2.6 Multiple Vehicle Coordination and Cooperation

As mentioned in the introduction, this is one of the motivating applications of this work. The structure presented in this thesis provides the basis for investigation of multiple vehicle collaborations as well. The Maneuver Automaton is constructed around symmetries, and there exist discrete symmetries [92, 105] among multiple vehicles that can be utilized. Reference trajectories can then be defined about individual vehicle motions in formations or periodically evolving swarming motion, that, when combined via discrete symmetries, allows for group formations and swarming to be defined and maintained. Developing decentralized online maneuvers to transition into and out of these formations or swarms would allow for decentralized multiple vehicle coordination and cooperation as well as changes to the number and configuration of vehicles. A key challenge in this is in efficient collaboration, where individual vehicles may act in a non-optimal way to obtain an optimal result for the entire group [56, 104, 27].

Bibliography

[1] W. P. M. H. Heemels A. Bemporad and B. DeSchutter. On hybrid systems and closed-loop mpc systems. *IEEE Transactions on Automatic Control*, 47(5):863–869, 2002.

[2] R. Abraham, J. E. Marsden, and T. S. Ratiu. *Manifolds, Tensor Analysis, and Applications*, volume 75 of *AMS*. Springer Verlag, New York, NY, second edition, 1988.

[3] S. Albers and M. Henzinger. Exploring unknown environments. *SIAM Journal of Computing*, 29(4):1164–1188, 2000.

[4] S. Albers, K. Kursawe, and S. Schuierer. Exploring unknown environments with obstacles. *Algorithmica*, 32:123–143, 2002.

[5] E. G. Al'brekht. On the optimal stabilization of nonlinear systems. *PMM - Journal of Applied Mathematics and Mechanics*, 25:1254–1266, 1961.

[6] M. Hofbaur J. How J. Kennell J. Loy R. Ragno J. Stedl B. Williams, P. Kim and A. Walcott. Model-based reactive programming of cooperative vehicles for Mars exploration. In *Proceedings of the International Symposium on Artificial Intelligence, Robotics, and Automation in Space*, St-Hubert, Canada, June 2001.

[7] R. Bachmayer and N. E. Leonard. Vehicle networks for gradient descent in a sampled environment. In *IEEE Conf. on Decision and Control*, pages 112–117, Las Vegas, NV, December 2002.

[8] A. Banaszuk and J. Hauser. Approximate feedback linearization: A homotopy operator approach. *SIAM Journal on Control and Optimization*, 34(5):1533–1554, 1996.

[9] J. Barraquand and J-C. Latombe. Robot motion planning: A distributed representation approach. *International Journal of Robotics Research*, 10(6):628–649, 1991.

[10] P. Berman. *Online Algorithms: the State of the Art*, chapter On-line searching and navigation, pages 232–241. Springer-Verlag, 1998.

[11] D. Bertsekas. Convergence of discretization procedures in dynamic programming. *IEEE Transactions on Automatic Control*, 20(6):415–419, June 1975.

[12] D. Bertsekas. *Dynamic Programming and Optimal Control*, volume 1. Athena Scientific, Belmont, Massachusetts, 1995.

[13] A. Bicchi, A. Marigo, and B. Piccoli. Quantized control systems and discrete nonholonomy. In *IFAC Workshop on Lagrangian and Hamiltonian Methods for Nonlinear Control*, pages 19–26, March 2000.

[14] A. Bicchi, A. Marigo, and B. Piccoli. On the reachability of quantized control systems. *IEEE Transactions on Automatic Control*, 47(4):546–63, 2002.

[15] P. E. Black. www.nist.gov/dads, 2003.

[16] R. W. Brockett. Control theory and singular Riemannian geometry. In P. Hilton and G. Young, editors, *New Directions in Applied Mathematics*, pages 11–27, New York, NY, 1982. Springer Verlag.

[17] R. W. Brockett. Asymptotic stability and feedback stabilization. In R. W. Brockett, R. S. Millman, and H. J. Sussmann, editors, *Geometric Control Theory*, pages 181–191, Boston, MA, 1983. Birkhäuser.

[18] A. E. Bryson and Y.-C. Ho. *Applied Optimal Control: Optimization, Estimation, and Control.* Taylor & Francis, Bristol, PA, 1981.

[19] F. Bullo. Exponential stabilization of relative equilibria for mechanical systems with symmetries. In *Mathematical Theory of Networks and Systems*, pages 987–990, Padova, Italy, July 1998.

[20] F. Bullo. Stabilization of relative equilibria for underactuated systems on Riemannian manifolds. *IFAC Automatica*, 36(12):1819–1834, 2000.

[21] F. Bullo. Series expansions for analytic systems linear in controls. *IFAC Automatica*, 38(9):1425–1432, 2002.

[22] F. Bullo and K. M. Lynch. Kinematic controllability and decoupled trajectory planning for underactuated mechanical systems. In *IEEE Int. Conf. on Robotics and Automation*, pages 3300–3307, Seoul, Korea, April 2001.

[23] G. J. Pappas C. Tomlin and S. Sastry. Conflict resolution in air traffic management: A study in multi-agent hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):509–521, April 1998.

[24] I. Mitchell C. Tomlin and R. Ghosh. Safety verification of conflict resolution maneuvers. *IEEE Transactions on Intelligent Transportation Systems*, 2(2):110–120, June 2001.

[25] J. Lygeros C. Tomlin and S. Sastry. Synthesizing controllers for nonlinear hybrid systems. In T. Henzinger and S. Sastry, editors, *Hybrid Systems: Computation and Control I*, Lecture Notes in Computer Science, pages 360–373. Springer Verlag, 1998.

[26] J. F. Canny. *The Complexity of Robot Motion Planning*. MIT Press, Cambridge, MA, 1988.

[27] C. G. Cassandras and W. Li. A receding horizon approach for solving some cooperative control problems. In *Proceedings of the 41st IEEE Conference on Decision and Control*, pages 3760–3765, Las Vegas, Nevada, December 2002.

[28] W. T. Cerven and F. Bullo. Constructive controllability algorithms for motion planning and optimization. *IEEE Transactions on Automatic Control*, 48(4):575–589, 2003.

[29] W. T. Cerven and V. L. Coverstone-Carroll. Optimal reorientation of multibody spacecraft through joint motion using averaging theory. *AIAA Journal of Guidance, Control, and Dynamics*, 24(4):788–795, 2001.

[30] B. Char et al. *Maple V Library Reference Manual*. Springer Verlag, 1991.

[31] C.-T. Chen. *Linear System Theory and Design*. Holt, Rinehart, and Winston, New York, NY, 1984.

[32] H. Choset and J. Burdick. Sensor-based exploration: the hierarchical generalized voronoi graph. *International Journal of Robotics Research*, 19(2):96–126, 2000.

[33] H. Choset and J. Burdick. Sensor based motion planning: The hierarchical generalized voronoi graph. *International Journal of Robotics Research*, 19(2):96–125, 2000.

[34] H. Choset and D. Kortkenkamp. Path planning and control for aercam, a free-flying inspection robot in space. *ASCE Journal of Aerospace Engineering*, 12:74–81, 1999.

[35] V. Coverstone-Carroll. Near-optimal low-thrust trajectories via micro-genetic algorithms. *AIAA Journal of Guidance, Control, and Dynamics*, 20(1):196–198, 1996.

[36] I. Suzuki D. Crass and M. Yamashita. Searching for a mobile intruder in a corridor - the open edge variant of the polygon search problem. *Int'l Journal of Computational Geometry and Applications*, 5(4):397–412, 1995.

[37] Narsingh Deo. *Graph Theory with Applications to Engineering and Computer Science*. Prentice Hall, Inc., 1974.

[38] J. Dugundji. *Topology*. Allyn and Bacon, Boston, Massachusetts, 1966.

[39] C. Fernandes, L. Gurvits, and Z. Li. Near optimal nonholonomic motion planning for a system of coupled rigid bodies. *IEEE Transactions on Automatic Control*, 39(3):450–463, 1994.

[40] M. Fliess, J. Lévine, P. Martin, and P. Rouchon. Flatness and defect of non-linear systems: Introductory theory and examples. *International Journal of Control*, 61(6):1327–1361, 1995.

[41] E. Frazzoli. *Robust Hybrid Control for Autonomous Vehicle Motion Planning*. PhD thesis, MIT, Cambridge, MA, June 2001.

[42] E. Frazzoli, M. A. Dahleh, and E. Feron. Robust hybrid control for autonomous vehicle motion planning. In *IEEE Conf. on Decision and Control*, pages 821–826, Sydney, Australia, December 2000.

[43] E. Frazzoli, M. A. Dahleh, and E. Feron. Maneuver-based motion planning for nonlinear systems with symmetries. *IEEE Transactions on Automatic Control*, 2003. To be submitted.

[44] E. Frazzoli, M. A. Daleh, and E. Feron. Real-time motion planning for agile autonomous vehicles. *AIAA Journal of Guidance, Control, and Dynamics*, 25(1):116–129, 2002.

[45] V. Prudkoglyad V. Semenchenko K. Yolkin G. Uspensky, V. Lukiashchenko and V. Kozlov. The unmanned multi-functional free-flying spacecraft for microgravity researches, serviced during periodic docking with the international space station. In *Proceedings of the 52nd International Astronautical Congress*, Toulouse, France, October 2001.

[46] Gerardo Laferriere George Pappas and Shankar Sastry. Hierarchically consistent control systems. *IEEE Transactions on Automatic Control*, 45(6):1144–1160, June 2000.

[47] D. T. Greenwood. *Principles of Dynamics*. Prentice Hall, 1988.

[48] D. T. Greenwood. *Classical Dynamics*. Dover, 1997.

[49] T. Komaki H. Noborio, I. Yamamoto. Sensor-based path-planning algorithms for a nonholonomic mobile robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 917–924, 2000.

[50] A. Halme. Polynomial operators for nonlinear systems analysis. *Acta Polytechnica Scandinavica*, Ma24:7–63, 1972.

[51] A. Halme and J. Orava. Generalized polynomial operators for nonlinear systems analysis. *IEEE Transactions on Automatic Control*, 17(2):226–8, 1972.

[52] R. P. Hamalainen and A. Halme. A solution of nonlinear TPBVP's occurring in optimal control. *IFAC Automatica*, 12(5):403–15, 1976.

[53] C. R. Hargraves and S. W. Paris. Direct trajectory optimization using nonlinear programming and collocation. *AIAA Journal of Guidance, Control, and Dynamics*, 10(4):338–342, 1987.

[54] J. E. Hauser, S. S. Sastry, and G. Meyer. Nonlinear control design for slightly nonminimum phase systems: application to V/STOL aircraft. *IFAC Automatica*, 28(4):665–679, 1992.

[55] Michael T. Heath. *Scientific Computing: an Introductory Survey*. McGraw-Hill, New York, 2 edition, 2002.

[56] Y. Ho and K. Chu. Team decision theory and information structures in optimal control problems - part 1. *IEEE Transactions on Automatic Control*, AC-17(1):15–22, 1972.

[57] J. Hu, M. Prandini, and S. S. Sastry. Optimal maneuver for multiple aircraft conflict resolution: A braid point of view. In *IEEE Conf. on Decision and Control*, pages 4164–4170, Sydney, Australia, December 2000.

[58] C. Tomlin J. Lygeros and S. Sastry. Controllers for reachability specifications for hybrid systems. *Automatica*, 35(3):349–370, 1999.

[59] S. Sastry J. Lygeros, K. H. Johansson and M. Egerstedt. On the existence of executions of hybrid automata. In *Proceedings of the 38th IEEE Conference on Decision and Control*, volume 3, pages 2249–2254, Phoenix, Arizona, December 1999.

[60] D. King L. Gregoris J. Middleton, H. Jones and J-C Piedboeuf. Innovative space servicing concepts. In *Proceedings of the 52nd International Astronautical Congress*, Toulouse, France, October 2001.

[61] O. Jae-Hyuk and E. Feron. Safety certification of air traffic conflict resolution algorithms involving more than two aircraft. In *IEEE American Control Conference*, pages 2807–11, Philadelphia, PA, June 1998.

[62] B. Jakubczyk and E. D. Sontag. Controllability of nonlinear discrete-time systems: A lie algebraic approach. *SIAM J. Control and Optimization*, 28(1):1–33, 1990.

[63] W. Kang and A. J. Krener. Extended quadratic controller form and dynamic state feedback linearization of nonlinear systems. *SIAM Journal on Control and Optimization*, 30(6):1319–1337, 1992.

[64] T. Karatas and F. Bullo. Randomized searches and nonlinear programming in trajectory planning. In *IEEE Conf. on Decision and Control*, pages 5032–5037, Orlando, FL, December 2001.

[65] L. E. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional space. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[66] M. Kawski. Nonlinear control and combinatorics of words. In B. Jakubczyk and W. Respondek, editors, *Geometry of Feedback and Optimal Control*, pages 305–346. Dekker, New York, NY, 1998.

[67] J. A. Kechichian. Optimal low-thrust transfer using variable bounded thrust. *Acta Astronautica*, 36(7):357–365, 1995.

[68] H. K. Khalil. *Nonlinear Systems*. Prentice Hall, Englewood Cliffs, NJ, second edition, 1995.

[69] K. Konolige. A gradient method for realtime robot control. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*.

[70] A. J. Krener. The construction of optimal linear and nonlinear regulators. In A. Isidori and T. J. Tarn, editors, *Systems, Models and Feedback: Theory and Applications*, pages 301–322. Birkhäuser, Boston, MA, 1992.

[71] J. J. Kuffner and S. M. LaValle. RRT-Connect: An efficient approach to single-query path planning. In *IEEE Int. Conf. on Robotics and Automation*, pages 995–1001, San Francisco, CA, April 2000.

[72] S.M. LaValle D. Lin L. J. Guibas, J.-C. Latombe and R. Motwani. Visibility-based pursuit-evasion in a polygonal environment. *Int'l Journal of Computational Geometry and Applications*, 9(5):471–494, 1999.

[73] E. Morales L. Rommero and E. Sucar. An exploration and navigation approach for indoor mobile robotics. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3092–3806, 2001.

[74] P. A. Laplante, editor. *Dictionary of Computer Science, Engineering, and Technology*. CRC Press, 2001.

[75] J.-C. Latombe. Motion planning: A journey of robots, molecules, digital actors, and other artifacts. *International Journal of Robotics Research*, 18(11):1119–1128, 1999.

[76] K. Lau, M. Colavita, G. Blackwood, R. Linfield, M. Shao, and D. Gallagher. The new millennium formation flying optical interferometer. In *AIAA Conf. on Guidance, Navigation and Control*, pages 650–656, 1997.

[77] S. M. Lavalle. Motion strategy library. http://msl.cs.uiuc.edu/msl/, 1 2003.

[78] S. M. LaValle and J. J. Kuffner. Rapidly-exploring random trees: Progress and prospects. In *Workshop on Algorithmic Foundations of Robotics*, pages 293–308, Dartmouth, NH, March 2000.

[79] S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. *International Journal of Robotics Research*, 20(5):378–400, 2001.

[80] D. Lawden. Optimal transfers between coplanar elliptical orbits. *AIAA Journal of Guidance, Control, and Dynamics*, 15(3):788–791, 1991.

[81] N. E. Leonard and P. S. Krishnaprasad. Motion control of drift-free, left-invariant systems on Lie groups. *IEEE Transactions on Automatic Control*, 40(9):1539–1554, 1995.

[82] H. S. London. Second approximation to the solution of the rendezvous equation. *AIAA Journal*, pages 1691–1693, July 1963.

[83] A. Loukianov and Vadim Utkin. Time-varying linear system decompozed control. In *Proceedings of the American Control Conference*, pages 2884–2888, Philadelphia, Pennsylvania, June 1998. IEEE.

[84] T. Lozano-Perez and M. Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 22(10):560–570, 1979.

[85] V. J. Lumelsky and T. Skewis. Incorporating range sensing in the robot navigation function. *IEEE Transactions on Systems, Man, and Cybernetics*, 20(5):1058–1069, 1990.

[86] Vladimir J. Lumelsky and Alexander A. Stepanov. Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape. *Algorithmica*, 2:403–430, 1987.

[87] V. Manikonda, P. S. Krishnaprasad, and J. Hendler. Languages, behaviors, hybrid architectures and motion control. In J. Baillieul and J. C. Willems, editors, *Mathematical Control Theory*. Springer Verlag, New York, NY, 1998.

[88] A. Marigo and A. Bicchi. Steering driftless nonholonomic systems by control quanta. In *IEEE Conf. on Decision and Control*, pages 4164–9, Tampa, FL, December 1998.

[89] J. E. Marsden, R. Montgomery, and T. S. Ratiu. Reduction, symmetry and phases in mechanics. *Mem. Amer. Math. Soc.*, 436, 1990.

[90] P. Martin, S. Devasia, and B. Paden. A different look at output tracking: Control of a VTOL aircraft. *IFAC Automatica*, 32(1):101–107, 1996.

[91] D. Q. Mayne and H. Michalska. Receding horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 35(7):814–824, 1990.

[92] M. Brett McMickell and B. Goodwine. Reduction and nonlinear controllability of symmetric distributed robotic systems with drift. In *Proceedings of the 2002 IEEE Int'l Conference on Robotics and Automation*, pages 3454–3460, Washington, DC, May 2002.

[93] W. Melbourne and C. Sauer. Optimum thrust programs for power-limited propulsion systems. Technical Report 32-118, Jet Propulsion Laboratory, Pasadena, CA, 1961.

[94] H. Michalska and D. Q. Mayne. Robust receding horizon control of constrained nonlinear systems. *IEEE Transactions on Automatic Control*, 38(11):1623–1633, 1993.

[95] I. Mitchell and C. Tomlin. Level set methods for computation in hybrid systems. In B. Krogh and N. Lynch, editors, *Hybrid Systems: Computation and Control III*, Lecture Notes in Computer Science, pages 310–323. Springer Verlag, 2000.

[96] R. M. Murray, M. Rathinam, and W. Sluis. Differential flatness of mechanical control systems: A catalog of prototype systems. In *ASME International Mechanical Engineering Congress and Exposition*, San Francisco, CA, November 1995.

[97] R. M. Murray and S. S. Sastry. Nonholonomic motion planning: Steering using sinusoids. *IEEE Transactions on Automatic Control*, 38(5):700–726, 1993.

[98] W. Xunzhang G. Seet M. Lau N. Ying, L. Eicher. Real-time 3d path planning for sensor-based underwater robotics vehicles in unknown environment. *Oceans Conference Record (IEEE)*, 3:2051–2058, 2000.

[99] C. L. Navasca and A. J. Krener. Solution of Hamilton-Jacobi-Bellman equations. In *IEEE Conf. on Decision and Control*, pages 570–574, Sydney, Australia, December 2000.

[100] P. Ogren and N. E. Leonard. A convergent dynamic window approach to obstacle avoidance. *IEEE Transactions on Robotics and Automation*, 2003. Submitted.

[101] P. Olver. *Equivalence, Invariants, and Symmetry*. Cambridge University Press, 1995.

[102] J. Orava and A. Halme. Inversion of generalized power series representations. *Journal of Mathematical Analysis and Applications*, 45:136–141, 1974.

[103] J. P. Ostrowski. Steering for a class of dynamic nonholonomic systems. *IEEE Transactions on Automatic Control*, 45(8):1492–1497, 2000.

[104] M. Pachter P. Chandler and S. Rasmussen. Uav cooperative control. In *Proceedings of American Control Conference*, pages 50–55, Arlington, VA, 2001.

[105] E. Fiorelli P. Ogren and N. Leonard. Formations with a mission: Stable coordination of vehicle group maneuvers. In *Proceedings of the Symposium on Mathematical Theory of Networks and Systems*, August 2002.

[106] W.H. Press, W.T. Vetterling, S.A. Teukolsky, and B.P. Flannery. *Numerical Recipes in C*. Cambridge University Press, New York, NY, 1992.

[107] J. E. Prussing and R. S. Clifton. Optimal multiple-impulse satellite evasive maneuvers. *AIAA Journal of Guidance, Control, and Dynamics*, 17(3):599–606, 1994.

[108] J. E. Prussing and B. A. Conway. *Orbital Mechanics*. Oxford University Press, Inc., New York, 1993.

[109] Q. P. Chu R. Bennis and J. A. Mulder. Adaptive fuzzy control by reinforcement learning for rendezvoux and docking. In *Proceedings of the 52nd International Astronautical Congress*, Toulouse, France, October 2001.

[110] J. H. Reif. Complexity of the mover's problem and generalizations. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 421–427, 1979.

[111] E. Rimon and D. E. Koditschek. Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation*, 8(5):501–518, 1992.

[112] E. Rimon and D. E. Koditschek. Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation*, 8(5):501–518, 1992.

[113] W. A. Scheel and B. A. Conway. Optimization of very-low-thrust, many-revolution spacecraft trajectories. *AIAA Journal of Guidance, Control, and Dynamics*, 17(6):1185–1192, November-December 1994.

[114] J. T. Schwartz and M. Sharir. On the 'piano movers' problem: Ii. general techniques for computing topological properties of real algebraic manifolds. *Advances in Applied Mathematics*, 4:298–351, 1983.

[115] H. Seywald. Trajectory optimization based on differential inclusion. *AIAA Journal of Guidance, Control, and Dynamics*, 17(3):480–487, 1994.

[116] Z. Shiller. Motion planning for mars rover. In *Proc. First Workshop on Robot Motion and Control*, Kiekrz, Poland, June 1999.

[117] A. Shkel and V. Lumelsky. The jogger's problem : Control of dynamics in real-time motion planning. *Automatica*, 33(7):1219–1233, July 1997.

[118] Ye. Ya. Smirnov. *Some mathematical theory control problems.* Leningrad University Press, 1982. in Russian.

[119] E. D. Sontag. *Mathematical Control Theory: Deterministic Finite Dimensional Systems*, volume 6 of *TAM*. Springer Verlag, New York, NY, second edition, 1998.

[120] S. Sundar and Z. Shiller. Optimal obstacle avoidance based on the hamilton-jacobi-bellman equation. *IEEE Transactions on Automatic Control*, 13(2):305–310, April 1997.

[121] H. J. Sussmann. A general theorem on local controllability. *SIAM Journal on Control and Optimization*, 25(1):158–194, 1987.

[122] H. J. Sussmann. New differential geometric methods in nonholonomic path finding. In A. Isidori and T. J. Tarn, editors, *Systems, Models, and Feedback: Theory and Applications*, pages 365–384. Birkhäuser, Boston, MA, 1992.

[123] C. Taylor and D. Kriegman. *Algorithmic Foundations of Robotics*, chapter Vison-Based Motion Planning and Exploration Algorithms for Mobile Robots, pages 69–83. A. K. Peters Ltd., 1995.

[124] C. Tomlin, G. J. Pappas, and S. S. Sastry. Conflict resolution for air traffic management: a study in multiagent hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):509–21, 1998.

[125] Physically-Based Simulation UNC Research Group on Modeling and Applications. Proximity query package. http://www.cs.unc.edu/ geom/SSV/, 1 2003.

[126] C. Urmson. Locally randomized kinodynamic motion planning for robots in extreme terrain. http://www-2.cs.cmu.edu/ curmson/Research/ThesisProposalUrmson.pdf, May 2002.

[127] A. J. van der Schaft and B. M. Maschke. On the Hamiltonian formulation of nonholonomic mechanical systems. *Reports in Mathematical Physics*, 34(2):225–233, 1994.

[128] M. J. van Nieuwstadt and R. M. Murray. Real time trajectory generation for differentially flat systems. In *IFAC World Congress*, June 1996.

[129] F. W. Warner. *Foundations of Differential Manifolds and Lie Groups*. Springer-Verlag, 1983.

[130] H. S. Wilf. *Generatingfunctionology*. Academic Press, New York, NY, second edition, 1994.

[131] S. Williams. *An introduction to the use of varitop, a general purpose low-thrust trajectory optimization program*. Jet Propulsion Laboratory, Pasadena, CA, January 1994.

[132] B. Williams Y. Gawdiak, J. Bradshaw and H. Thomas. R2d2 in a softball: the portable satellite assistant. In *Proceedings of the International Conference on Intelligent User Interfaces*, pages 125–128, New Orleans, Louisiana, 2000.

[133] H. Zhang and J. P. Ostrowski. Control algorithms using affine connections on principal fiber bundles. In *IFAC Workshop on Lagrangian and Hamiltonian Methods for Nonlinear Control*, pages 129–34, Princeton, NJ, March 2000.

Appendix A

Local planner supplementary development and proofs

A.1 Minimum energy planning with base functions

Section 3.3 presents a motion planning algorithm using base functions and a minimum energy planning algorithm that requires no base functions. This appendix presents a third algorithm that solves the minimum energy planning problem using base functions.

Consider the following design problem: find a control input $u : [0, T] \mapsto R^m$ that solves

$$\begin{aligned} \min \ & \int_0^T \|u(t)\|_2^2 dt \\ \text{subject to } & \dot{x} = A(t)x + f^{[2]}(x, x) + Bu \\ & x(0) = 0, \ \ x(T) = x_{\text{target}}. \end{aligned}$$

Using the series expansion characterization in Section 3.2.2, the problem becomes finding a control input $u : [0, T] \mapsto R^m$ that solves

$$\begin{aligned} \text{minimize} \quad & \int_0^T \|u(t)\|_2^2 dt \\ \text{subject to} \quad & x_{\text{target}} = \sum_{k=1}^{+\infty} x_k(T). \end{aligned}$$

Using the base functions $\{\psi_i(t) : i \in \{1, \ldots, m\}\}$ introduced in Section 3.3.1, the design problem is to find a vector $c \in \mathbb{R}^m$ that solves

$$\begin{aligned} \text{minimize} \quad & \|c\|_Q^2 = c'Qc \\ \text{subject to} \quad & x_{\text{target}} = \sum_{k=1}^{+\infty} \Phi_k(c, \ldots, c), \end{aligned}$$

where we define the symmetric positive definite matrix Q according to

$$Q_{ij} = \int_0^T \psi^i(t)'\psi^j(t)dt.$$

Next, we introduce the Lagrange multiplier $\lambda \in \mathbb{R}^n$ and write the Hamiltonian as

$$H(c, \lambda) = \frac{1}{2}c'Qc + \lambda' \left(-x_{\text{target}} + \sum_{k=1}^{+\infty} \Phi_k(c, \ldots, c) \right).$$

Since the tensors $\{\Phi_k, k \in \mathbb{N}\}$ are symmetric, the necessary conditions for optimality are

$$\begin{aligned} x_{\text{target}} &= \sum_{k=1}^{+\infty} \Phi_k(c, \ldots, c) \\ 0 &= Qc + \sum_{k=1}^{+\infty} k\Phi_k(\underbrace{c, \ldots, c}_{k-1 \text{ times}})'\lambda. \end{aligned}$$

In the second equation the tensor Φ_k is contracted with $(k-1)$ arguments and it is therefore an $(n \times m)$ matrix. The first equation has n components, while the second equation has m components. Rewriting the previous conditions in vector format, the design problem is to find a vector $(c, \lambda) \in \mathbb{R}^{m+n}$ such that

$$\begin{bmatrix} x_{\text{target}} \\ 0 \end{bmatrix} = \begin{bmatrix} \Phi_1 & 0 \\ Q & \Phi_1' \end{bmatrix} \begin{bmatrix} c \\ \lambda \end{bmatrix} + \begin{bmatrix} \Phi_2(c, c) \\ 2\Phi_2(c)'\lambda \end{bmatrix} + \sum_{k=3}^{+\infty} \begin{bmatrix} \Phi_k(\underbrace{c, \ldots, c}_{k \text{ times}}) \\ k\Phi_k(\underbrace{c, \ldots, c}_{k-1 \text{ times}})'\lambda \end{bmatrix}. \tag{A.1}$$

As in Section 3.3, this trajectory optimization setting is cast as an inverse function problem.

A.2 Solution of polynomial system as series expansion

We prove here Lemma 3.2.1.

Proof: The proof is a direct extension of the treatment in [21]. We present here only the convergence proof as the derivation of the formal expansion is straightforward. We start with the bounds

$$\|x_1\|_{\mathcal{L}_\infty} \leq \|\Psi(t,0)x_0\|_{\mathcal{L}_\infty} + \|\Psi(t,0)\|_{\mathcal{L}_1}\|Bu\|_{\mathcal{L}_\infty} = (1/2)d_1$$
$$\|x_k\|_{\mathcal{L}_\infty} \leq \|\Psi(t,0)\|_{\mathcal{L}_1}\|f^{[2]}\|_{\mathcal{L}_\infty}\sum_{i=1}^{k-1}\|x_i\|_{\mathcal{L}_\infty}\|x_{k-i}\|_{\mathcal{L}_\infty} = (1/2)d_2\sum_{i=1}^{k-1}\|x_i\|_{\mathcal{L}_\infty}\|x_{k-i}\|_{\mathcal{L}_\infty}.$$

Provided $d_1 d_2 \leq 1$, and assuming $\|x_k\|_{\mathcal{L}_\infty} \leq c_k d_1^k d_2^{k-1}$,

$$\begin{aligned}
\|x_{k+1}\|_{\mathcal{L}_\infty} &\leq (1/2)d_2\sum_{i=1}^{k}\|x_i\|_{\mathcal{L}_\infty}\|x_{k+1-i}\|_{\mathcal{L}_\infty} \\
&\leq (1/2)d_2\sum_{i=1}^{k} c_i c_{k+1-i} d_1^{k+1} d_2^{k-1} = c_{k+1} d_1^{k+1} d_2^k \\
\|x - \sum_{k=1}^{K} x_k\|_{\mathcal{L}_\infty} &\leq \sum_{k=K+1}^{+\infty}\|x_k\|_{\mathcal{L}_\infty} \leq \sum_{k=K+1}^{+\infty} c_k d_1^k d_2^{k-1} = \frac{1}{d_2}\sum_{k=K+1}^{+\infty} c_k d_1^k d_2^k \\
&\leq \frac{1}{d_2}\,\text{Remainder}_K(C)(d_1 d_2).
\end{aligned}$$

Thus, Lemma 3.2.1 is true by induction. ∎

A.3 Convergence of iterative algorithm

We start with some preliminary results. Let Symm$(\cdot)$ be the symmetrization operator [50] defined as

$$\text{Symm}(F)(y_1, y_2, \ldots, y_k) = \frac{1}{k!}\sum_{\alpha_1,\ldots,\alpha_k \in \{0,1\}} (-1)^{(k-\sum_{i=1}^k \alpha_i)} F\left(\sum_{i=1}^k \alpha_i y_i, \ldots, \sum_{i=1}^k \alpha_i y_i\right). \tag{A.2}$$

Lemma A.3.1 *Let F be a tensor, i.e., a multi-linear map, from k copies of $\mathbb{R}^n$ to $\mathbb{R}^n$. For all $y_1, y_2 \in \mathbb{R}^n$ we have*

$$F(y_2,\ldots,y_2) - F(y_1,\ldots,y_1) = \sum_{j=0}^{k-1} \mathrm{Symm}(F)(y_2 - y_1, \underbrace{y_2,\ldots,y_2}_{k-1-j \ \ times}, \underbrace{y_1,\ldots,y_1}_{j \ \ times}). \qquad (A.3)$$

Proof: Consider the following chain of equalities

$$\begin{aligned}
&\sum_{j=0}^{k-1} \mathrm{Symm}(F)(y_2 - y_1, \underbrace{y_2,\ldots,y_2}_{k-1-j \text{ times}}, \underbrace{y_1,\ldots,y_1}_{j \text{ times}}) = \\
&\quad \sum_{j=0}^{k-1} \left(\mathrm{Symm}(F)(\underbrace{y_2,\ldots,y_2}_{k-j \text{ times}}, \underbrace{y_1,\ldots,y_1}_{j \text{ times}}) - \mathrm{Symm}(F)(\underbrace{y_2,\ldots,y_2}_{k-1-j \text{ times}}, \underbrace{y_1,\ldots,y_1}_{j+1 \text{ times}}) \right) \\
&\quad = \sum_{j=0}^{k-1} \mathrm{Symm}(F)(\underbrace{y_2,\ldots,y_2}_{k-j \text{ times}}, \underbrace{y_1,\ldots,y_1}_{j \text{ times}}) - \sum_{i=1}^{k} \mathrm{Symm}(F)(\underbrace{y_2,\ldots,y_2}_{k-i \text{ times}}, \underbrace{y_1,\ldots,y_1}_{i \text{ times}}) \\
&\quad = \mathrm{Symm}(F)(\underbrace{y_2,\ldots,y_2}_{k-j \text{ times}}, \underbrace{y_1,\ldots,y_1}_{j \text{ times}})\Big|_{j=0} - \mathrm{Symm}(F)(\underbrace{y_2,\ldots,y_2}_{k-i \text{ times}}, \underbrace{y_1,\ldots,y_1}_{i \text{ times}})\Big|_{i=k} \\
&= \mathrm{Symm}(F)(\underbrace{y_2,\ldots,y_2}_{k \text{ times}}) - \mathrm{Symm}(F)(\underbrace{y_1,\ldots,y_1}_{k \text{ times}}) = F(\underbrace{y_2,\ldots,y_2}_{k \text{ times}}) - F(\underbrace{y_1,\ldots,y_1}_{k \text{ times}}).
\end{aligned}$$

■

Lemma A.3.2 *For $\eta \in [0,1]$, the remainder of the Catalan function $C(\eta) = 1 - \sqrt{1-\eta}$ satisfies*

$$\mathrm{Remainder}_1\left(1 - \sqrt{1-\eta}\right) = (1 - \sqrt{1-\eta}) - \frac{\eta}{2} \leq \frac{\eta^2}{2}.$$

Proof: Let $\eta \in [0,1]$. The following chain of inequalities holds

$$\begin{aligned}
&-\frac{3}{4} + \frac{1}{2}\eta + \frac{1}{4}\eta^2 < 0 \quad \Rightarrow \quad -\frac{3}{4}\eta^2 + \frac{1}{2}\eta^3 + \frac{1}{4}\eta^4 + (1-\eta) \leq (1-\eta) \\
&\Rightarrow \quad \left(1 - \frac{1}{2}\eta - \frac{1}{2}\eta^2\right)^2 \leq (1-\eta) \quad \Rightarrow \quad \left(1 - \frac{1}{2}\eta - \frac{1}{2}\eta^2\right) \leq \sqrt{1-\eta} \\
&\Rightarrow \quad 1 - \sqrt{1-\eta} - \frac{1}{2}\eta \leq \frac{1}{2}\eta^2.
\end{aligned}$$

■

We are finally ready to prove Theorem 3.4.10, that we restate for convenience.

Theorem A.3.3 *Let* $z = 2\|f_1^p\|_\infty D_1 D_2 \|x_{\text{target}}\|_\infty$. *If*

$$z < \min\left\{\frac{1}{D_1\|f_1^p\|_\infty}, 1 - \frac{(D_1\|f_1^p\|_\infty)^2}{(1+D_1\|f_1^p\|_\infty)^2}\right\},$$

there exists a unique χ^* *belonging to the set* S *and satisfying* $\chi^* = \mathcal{M}(\chi^*)$. *Furthermore, the unique solution can be computed by iterating the map* $\mathcal{M}$ *starting from any initial condition in* S.

Proof: We prove the theorem in three steps. We show first that the series converges for any input in S, then that S is invariant under the map $\mathcal{M}$, and finally that $\mathcal{M}$ is a contraction over S.

First, note that $y = f_1^p \chi$ implies $\|y\|_\infty \leq \|f_1^p\|_\infty \|\chi\|_\infty$, and $\chi \in S$ implies $\|\chi\|_\infty \leq 2\|x_{\text{target}}\|_\infty$. Hence we compose the bounds to obtain $D_1 D_2 \|y\|_\infty \leq D_1 D_2 \|f_1^p\|_\infty \|\chi\|_\infty \leq D_1 D_2 \|f_1^p\|_\infty 2\|x_{\text{target}}\|_\infty = z < 1 - \frac{(D_1\|f_1^p\|_\infty)^2}{(1+D_1\|f_1^p\|_\infty)^2} \leq 1$, which guarantees series convergence according to Lemma 3.4.1 and establishes that the contraction bounds are more conservative than the series bounds.

Second, we show that if $\chi \in S$, then $\mathcal{M}(\chi)$ also belongs to S, i.e., $\|\mathcal{M}(\chi) - x_{\text{target}}\|_\infty < \|x_{\text{target}}\|_\infty$. We compute

$$\begin{aligned}\|\mathcal{M}(\chi) - x_{\text{target}}\|_\infty &= \|\sum_{k=2}^{+\infty} f_k(f_1^p\chi, \ldots, f_1^p\chi)\|_\infty = \|f(f_1^p\chi) - f_1(f_1^p\chi)\|_\infty \\ &= \frac{1}{D_2}\,\text{Remainder}_1(\mathcal{C})(D_1 D_2\|f_1^p\chi\|_\infty).\end{aligned}$$

From the bound in Lemma A.3.2

$$\begin{aligned}&= \frac{1}{D_2}\,\text{Remainder}_1(\mathcal{C})(D_1 D_2\|f_1^p\chi\|_\infty) \leq \frac{(D_1 D_2\|f_1^p\chi\|_\infty)^2}{2D_2} \\ &\leq \frac{(2D_1 D_2\|f_1^p\|_\infty\|x_{\text{target}}\|_\infty)^2}{2D_2} = \frac{z^2}{2D_2}.\end{aligned}$$

From the second bound on z we have

$$\|\mathcal{M}(\chi) - x_{\text{target}}\|_\infty \leq \frac{z^2}{2D_2} = z(D_1\|f_1^p\|_\infty\|x_{\text{target}}\|_\infty) \leq \|x_{\text{target}}\|_\infty.$$

Finally, we show that $\|\mathcal{M}(\chi_2) - \mathcal{M}(\chi_1)\|_\infty \leq \rho\|\chi_2 - \chi_1\|_\infty$, where $0 \leq \rho < 1$. Applying the equality (A.3) from Lemma A.3.1:

$$\begin{aligned}
&\|\mathcal{M}(\chi_2) - \mathcal{M}(\chi_1)\|_\infty \qquad\qquad (A.4)\\
&\quad \|\sum_{k=2}^{+\infty}\sum_{j=0}^{k-1} \text{Symm}(f_k)(f_1^p(\chi_2 - \chi_1), \underbrace{f_1^p\chi_2, \ldots, f_1^p\chi_2}_{k-1-j \text{ times}}, \underbrace{f_1^p\chi_1, \ldots, f_1^p\chi_1}_{j \text{ times}})\|_\infty \\
&\quad \leq \sum_{k=2}^{+\infty}\sum_{j=0}^{k-1} \left(\|\text{Symm}(f_k)\|_\infty\|f_1^p\|_\infty^k\|\chi_2\|_\infty^{k-1-j}\|\chi_1\|_\infty^j\|\chi_2 - \chi_1\|_\infty\right) \\
&\quad \leq \|\chi_2 - \chi_1\|_\infty \sum_{k=2}^{+\infty}\sum_{j=0}^{k-1} \left(\|\text{Symm}(f_k)\|_\infty\|f_1^p\|_\infty^k 2^{k-1}\|x_{\text{target}}\|_\infty^{k-1}\right) \\
&\quad \leq \|\chi_2 - \chi_1\|_\infty \sum_{k=2}^{+\infty} \left(k2^{k-1}\|\text{Symm}(f_k)\|_\infty\|f_1^p\|_\infty^k\|x_{\text{target}}\|_\infty^{k-1}\right). \qquad (A.5)
\end{aligned}$$

We now upper bound $\|\text{Symm}(f_k)\|_\infty$ for $k > 1$

$$\|\text{Symm}(f_k)\|_\infty \leq (k!)^{-1}2^k\|f_k\|_\infty \leq 2^{1-k}2^k\|f_k\|_\infty = 2\|f_k\|_\infty.$$

Plugging the bound on f_k from Lemma 3.4.1 into equation (A.5), we obtain

$$\begin{aligned}
\|\mathcal{M}(\chi_2) - \mathcal{M}(\chi_1)\|_\infty &\leq \|\chi_2 - \chi_1\|_\infty \sum_{k=2}^{+\infty} \left(k2^k c_k D_1^k D_2^{k-1}\|f_1^p\|_\infty^k\|x_{\text{target}}\|_\infty^{k-1}\right) \\
&\leq \|\chi_2 - \chi_1\|_\infty 2D_1\|f_1^p\|_\infty^k \sum_{k=2}^{+\infty} \left(kc_k z^{k-1}\right).
\end{aligned}$$

The power series $\left(\sum_{k=1}^{+\infty} ka_k z^{k-1}\right)$ is the derivative of the generating function C (note the initial index), and can be shown to be convergent for $z < 1$. Using these

facts, we can write

$$\|\mathcal{M}(\chi_2) - \mathcal{M}(\chi_1)\|_\infty \leq \|\chi_2 - \chi_1\|_\infty D_1 \|f_1^p\|_\infty^k \left(\frac{1}{\sqrt{1-z}} - 1\right) = \rho \|\chi_2 - \chi_1\|_\infty,$$

where we set $\rho = D_1\|f_1^p\|_\infty^k \left(\frac{1}{\sqrt{1-z}} - 1\right)$. A few algebraic equalities based on last bound on z prove the bound $\rho < 1$. In summary, the map $\mathcal{M}$ is well-defined and is a contraction over the set S. The statement in the theorem follows from an application of the contraction mapping theorem. ■

A.4 Convergence of power series inversion

We start with some useful facts about a series.

Lemma A.4.1 *Let $\beta \in \mathbb{R}_+$, consider the series of positive numbers*

$$a_1 = 1, \qquad a_k = \beta \sum_{m=2}^{k} \sum_{\substack{i_1+\cdots+i_m=k \\ i_1,\cdots,i_m<k}} a_{i_1} \cdots a_{i_m}, \tag{A.6}$$

and define its generating function $h(\eta) = \sum_{k=1}^{+\infty} a_k \eta^k$. The following results hold:

(i) $h(\eta) = (1 + \eta - \sqrt{1 - 2(1+2\beta)\eta + \eta^2})/(2\beta + 2)$,

(ii) the function h is defined real, or in other words, the series $\sum_{k=1}^{+\infty} a_k \eta^k$ converges absolutely, provided $0 \leq \eta \leq (4(\beta+1))^{-1}$, and

(iii) the series $c_1 = \delta$, $c_k = \beta \sum_{m=2}^{k} \sum_{\substack{i_1+\cdots+i_m=k \\ i_1,\cdots,i_m<k}} \alpha^{m-1} c_{i_1} \cdots c_{i_m}$, can be bounded as $c_k \leq \delta^k \alpha^{k-1} a_k$.

We refer to [21] for the proof of most results in the lemma. Next, we prove Theorem 3.4.11.

Proof: We start by showing $\Lambda_2 \leq \Lambda_1$. First, we have

$$\begin{aligned}\Lambda_2 &= \frac{1}{4(D_1\|f_1^p\|_\infty + 1)\|f_1^p\|_\infty D_1 D_2} < \frac{1}{2\|f_1^p\|_\infty D_1 D_2}\left(\frac{1}{(D_1\|f_1^p\|_\infty + 1)}\right) \\ &\leq \frac{1}{2\|f_1^p\|_\infty D_1 D_2}\left(\frac{1}{D_1\|f_1^p\|_\infty}\right)\end{aligned}$$

and furthermore

$$\begin{aligned}\Lambda_2 &< \frac{1}{2\|f_1^p\|_\infty D_1 D_2}\left(\frac{1}{(D_1\|f_1^p\|_\infty+1)}\right)\\ &< \frac{1}{2\|f_1^p\|_\infty D_1 D_2}\left(\frac{1}{(D_1\|f_1^p\|_\infty+1)}\right)\left(1+\frac{D_1\|f_1^p\|_\infty}{(D_1\|f_1^p\|_\infty+1)}\right)\\ &< \frac{1}{2\|f_1^p\|_\infty D_1 D_2}\left(\frac{1+2D_1\|f_1^p\|_\infty}{(1+D_1\|f_1^p\|_\infty)^2}\right) < \frac{1}{2\|f_1^p\|_\infty D_1 D_2}\left(1-\frac{(D_1\|f_1^p\|_\infty)^2}{(1+D_1\|f_1^p\|_\infty)^2}\right).\end{aligned}$$

As seen in the proof of Theorem 3.4.10, when $\|x_{\text{target}}\|_\infty < \Lambda_1$, f is analytic (i.e., its series converges). Knowing this, we prove that the series defining the inverse function g in equation (3.17) converges uniformly in a neighborhood of x_{target}. From the Theorem 3.4.11, one can see that

$$\|g_1\|_\infty = \|f_1^{-1}\|_\infty\,, \qquad \|g_k\|_\infty \le \|g_1\|_\infty \sum_{m=2}^{k} \sum_{\substack{i_1+\cdots+i_m=k \\ i_1,\cdots,i_m<k}} \|f_m\|_\infty \|g_{i_1}\|_\infty \cdots \|g_{i_m}\|_\infty.$$

Plugging in the bound on $\|f_m\|_\infty$ from Lemma 3.4.1, we have

$$\begin{aligned}\|g_k\|_\infty &\le \|g_1\|_\infty \sum_{m=2}^{k} \sum_{\substack{i_1+\cdots+i_m=k \\ i_1,\cdots,i_m<k}} \left(c_m D_1^m D_2^{m-1}\right) \|g_{i_1}\|_\infty \cdots \|g_{i_m}\|_\infty\\ &\le \left(D_1\|f_1^{-1}\|_\infty\right) \sum_{m=2}^{k} \sum_{\substack{i_1+\cdots+i_m=k \\ i_1,\cdots,i_m<k}} (D_1 D_2)^{m-1} \|g_{i_1}\|_\infty \cdots \|g_{i_m}\|_\infty\,,\end{aligned}$$

where we used the bound $c_k \le 1$, for all $k > 1$. Let $\beta = D_1\|f_1^{-1}\|_\infty$, define the series $\{a_k \in \mathbb{R}, k \in \mathbb{N}\}$ as in equation (A.6), and following induction from the last statement above:

$$\|g_k\|_\infty \le \|f_1^{-1}\|_\infty^k (D_1 D_2)^{k-1} a_k.$$

In summary, we have

$$\begin{aligned}\|g(x_{\text{target}})\|_\infty &\leq \|\sum_{k=1}^{+\infty} g_k(x_{\text{target}},\dots,x_{\text{target}})\|_\infty \\ &\leq \frac{1}{D_1 D_2}\sum_{k=1}^{+\infty} a_k \left(\|f_1^{-1}\|_\infty D_1 D_2 \|x_{\text{target}}\|_\infty\right)^k,\end{aligned}$$

and, by Lemma A.4.1, convergence is ensured provided
$4(D_1\|f_1^{-1}\|_\infty + 1)\|f_1^{-1}\|_\infty D_1 D_2\|x_{\text{target}}\|_\infty \leq 1$.

Next, we prove that g is the inverse of f. The following proof is borrowed from [51] and we report it here for completeness. Evaluating the following expression,

$$\begin{aligned}f_1^{-1}\left(f\left(g(x)\right)\right) - g(x) &= f_1^{-1}\left(f - f_1\right)\left(g(x)\right) = f_1^{-1}\sum_{k=2}^{+\infty} f_k\left(\sum_{i_1=1}^{+\infty} g_{i_1},\cdots\sum_{i_k=1}^{+\infty} g_{i_k}\right) \\ &= f_1^{-1}\sum_{k=2}^{+\infty}\sum_{i_1,\cdots i_k=1}^{+\infty} f_k\left(g_{i_1},\cdots g_{i_k}\right) = \sum_{k=2}^{+\infty} f_1^{-1}\sum_{\substack{i_1+\cdots+i_m=k \\ i_1,\cdots,i_m<k}} f_m\left(g_{i_1},\cdots g_{i_m}\right) \\ &= \sum_{k=2}^{+\infty} g_k = f_1^{-1}(x) - g(x).\end{aligned}$$

Therefore, $f_1^{-1}\left(f\left(g(x)\right)\right) = f_1^{-1}(x)$, and $\left(f\left(g(x)\right)\right) = x$. ■

A.5 Local Planner Control Bounds

The control generated by the local planner in chapter 3 (in conjunction with the iterative contraction algorithm) has an inherent upper bound dependent upon the desired state of the system x_{target}. Mathematically, this derives from the series definition and the nature of the contraction algorithm itself. The contraction method works when the property $\|\chi\|_\infty \leq 2\|x_{\text{target}}\|_\infty$ is satisfied, where the variable χ is related to the parameter y_0 by $y_0 = f_1^{-1}\chi$. Let $\bar{u}$ be the control history of the local planner, which is then, in turn, related to the parameter by the function $\gamma(t) : y_0 \mapsto \bar{u}$. Let us now define $\gamma(t)$ such that, for the motion planning problem with control inputs, $\gamma(t) = \psi(t)$, whereas

$\gamma(t) = -B^T\overline{\Phi}_{\lambda\lambda}(t)$ for the minimum energy planning problem. Note that $\Phi_{\lambda\lambda}(t)$ is defined such that $\overline{\Phi}_{\lambda\lambda}(t)(y) = \sum_{k=1}^{N_{\text{trunc}}} \overline{\Phi}_{k,,\lambda\lambda}(t)(y,\ldots,y)$. When the overall control is defined as in (2.12), the control vector component relating to the local planner is expressed as $(Bu)_{\text{local planner}} = BK_u\bar{u}$. In that context, the bound on the control vector's local planner component evaluates to

$$\|(Bu)_{\text{local planner}}\|_{\mathcal{L}_\infty} \leq 2\|BK_u\gamma(t)(f_1^{-1})\|_{\mathcal{L}_\infty}\|x_{\text{target}}\|_\infty. \tag{A.7}$$

Vita

William Todd Cerven

Education:

University of Illinois Ph.D. in Aero./Astro. Eng. October 2003 GPA: 3.92/4.00
University of Illinois M.S. in Aero./Astro. Eng. May 1999 GPA: 3.96/4.00
University of Illinois B.S. in Aero./Astro. Eng. May 1997 GPA: 4.00/4.00

Academic Experience:

UNIVERSITY OF ILLINOIS, Urbana, Illinois
RESEARCH ASSISTANT, Coordinated Sciences Laboratory 8/01 to 6/03
- Developed motion planning theory and software

INSTRUCTOR, WYSE Program 6/02 to 7/02
- Taught basic rocketry concepts to high school students

TEACHING ASSISTANT, Dept. of Aero. and Astro. Eng. 1/00 to 5/00
- Head teaching assistant for undergraduate structures and controls laboratory

RESEARCH ASSISTANT, Dept. of Aero. and Astro. Eng. 1/97 to 5/97
- Helped design chemical laser nozzle

UNDERGRAD. RESEARCHER, Dept. of Aero. and Astro. Eng. 8/94 to 12/96
- Investigated hydrocarbon scramjet fuel

JOHN A. LOGAN COLLEGE, Carterville, Illinois 6/94 to 8/94
MATH AND PHYSICS TUTOR

Professional Experience:

BOEING SATELLITE SYSTEMS COMPANY, El Segundo, California 6/97 to 6/02
MTS-3
-Control Systems Development, TDRS, Odin, Spaceway (Summers '99, '00, '01)
-General Systems Eng. for HS601HP line, HS702 AMRC S/C (Summers '97, '98)

MCDONNELL DOUGLAS CORPORATION, St. Louis, Missouri 5/96 to 8/96
SUPPLIER QUALITY MANAGEMENT REP./PROCESS CONSULTANT
-Evaluated supplier processes, approved several for ship-to-stock program

Journal Publications:

- Cerven, W.T. and Bullo, F., "Constructive Controllability Algorithms for Motion Planning and Optimization," IEEE Transactions on Automatic Control, April 2003.
- Cerven, W.T. and Coverstone,V., "Optimal Reorientation of a Multibody Spacecraft Through Joint Motion Using Averaging Theory," Journal of Guidance, Control, and Dynamics, Vol. 24, No.4, 2001.

Conference Papers:

- Bullo, F. and Cerven, W.T., On Trajectory Optimization for Polynomial Systems via Series Expansions, IEEE Conference on Decision and Control, Sydney, Australia, 2000.
- Cerven, W.T. and Coverstone,V., Optimal Reorientation of a Multibody Spacecraft Through Joint Motion Using Averaging Theory, AAS 00-203, AAS/AIAA Space Flight Mechanics Meeting, Clearwater, FL, 2000.

- Davis, K., Cerven, W.T., and Solomon,W., "The Use of Methane as a Fuel for Hypersonic Prop.," AIAA Paper 95-2769, 31st AIAA Joint Propulsion Conference, San Diego, CA, 1995.

Leadership Training (at Univ. of Illinois):

- Engineering Council Leadership Conference ('95 Part., '96 Planner)
- LeaderShape Institute ('95 Participant)
- Student Eng. Leadership Program ('95 Part., '96 Facilitator, '97 Planner)

Awards/Honors:

- Aerospace Illinois Space Grant Consortium Fellowship (2003)
- Mavis Memorial Fund Scholarship (2002)
- AIAA Guidance, Navigation, and Control Graduate Award (2001)
- National Science Foundation Fellowship Recipient (1997-2001)
- University of Illinois Departmental Fellowship (1999)
- Ammon S. Andes National Sigma Gamma Tau Award (1997)
- University Honors, University of Illinois (1997)
- AIAA Outstanding Undergraduate Student (1997), U of I AIAA chapter
- Chancellor's Scholar, University of Illinois (1993-1997)
- James Scholar, University of Illinois (1993-1997)
- H. H. Jordan Award Finalist (1997), U of I Coll. of Eng.
- Knights of St. Patrick Finalist (1996, 1997), U of I Coll. of Eng.
- Robert W. McCloy Award (1996), U of I AAE Dept.
- Sigma Gamma Tau Award (Chapter-1995, Regional-1997)

- Aerospace Illinois Space Grant Consortium Scholar (1995)
- Dads Association Book Plate Award (1995), U of I Dad's Assoc.
- Dean's List, University of Illinois (1993-1996)
- Eggers Engineering Scholarship (1996-1997), University of Illinois
- Dwight Glasscock Scholarship (1995-1996), University of Illinois
- Illinois General Assembly Scholarship (1995-1996)
- Hendrick House Scholarship Award (1994-1995), Hendrick House
- Undergraduate Development Award (1994-1995), University of Illinois
- Tylenol Leadership Scholarship (1993-1994)
- Robert C. Byrd Honors Scholarship (1993-1997), U. S. Dept. of Education
- Hamill Engineering Scholarship (1993-1994), University of Illinois
- Illinois Merit Scholarship (1993-1994), Illinois Dept. of Education

Professional Memberships:

- American Institute of Aeronautics and Astronautics
- Sigma Gamma Tau (Aerospace Engineering Honor Society)
  99-02 V. Pres., Tutoring Chair; 95-Pres.; 94,96-Eng. Council Rep.
- Golden Key (National Honor Society)
- Phi Kappa Phi (National Honor Society)
- Tau Beta Pi (Engineering Honor Society)
- Order of the Engineer (Organizing Committee 1999)

Reviewer of Publications:

- AIAA Journal
- IEEE Transactions on Robotics and Automation
- Journal of Robotic Systems
- Journal of Spacecraft and Rockets
- American Control Conference
- International Journal of Nonlinear Mechanics

Computer Programming Experience:

- C, C++, Matlab, Maple, Adsim