DISCRETE GEOMETRIC MOTION CONTROL OF AUTONOMOUS VEHICLES

by

Marin Kobilarov

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(COMPUTER SCIENCE)

August 2008

Acknowledgments

This work would not have been possible were I not surrounded by many great people who have helped and encouraged me along the way. I would like to thank my family for their faith in me and constant support. I am deeply grateful to my adviser, Gaurav Sukhatme, who created an environment with unlimited academic possibilities that gave me complete freedom to explore interesting areas that I had not imagined before. Part of this environment were also the students, many of which good friends, at the USC robotics labs. I would especially like to thank Srikanth Saripalli and Boyoon Jung for their help during my first year at USC.

Towards the end of my doctoral work I had the unique chance to work closely with Mathieu Desbrun and Jerry Marsden at Caltech. I am grateful to Jerry and Mathieu for their guidance and source of knowledge that helped me shape this thesis and identify my future goals.

I would like to thank Dr. David Ahlgren for introducing me to robotics in 1999. His dedication to learning was one of the reasons why I continued pursuing my dream and passion for robotics with the hope to contribute to science one day.

Table of Contents

List Of Figures

Abstract

The goal of this work is to develop methods to optimally control autonomous robotic vehicles in natural environments. The main contribution is the derivation of state-space structure respecting integration and optimization schemes for mechanical systems with *symmetries*, controllable shape dynamics, and *nonholonomic constraints* based on the theory of *discrete mechanics*. At the core of this approach lies the discretization of variational principles of mechanics that results in various numerical benefits previously unexplored in the area. The resulting framework is then used as a basis for developing optimal control methods applicable to various systems. Developed examples include simplified models of a car, a helicopter, a snakeboard, and a boat. The resulting algorithms are numerically stable, preserve the mechanical geometric structure, and are numerically competitive to existing methods. In addition, two important extensions with view towards practical applications are proposed. First, complex constraints are handled more robustly using *homotopy continuation* – the process of relaxing nontrivial motion constraints arising either from complicated dynamics or from obstacles in the environment and then smoothly transforming the solution of such easier problem into the original one by deforming the constraints back to their original shape. Second, the optimality and

computational efficiency of solution trajectories is addressed by combining discrete mechanics and optimal control (*DMOC*) with *sampling-based roadmaps*–a motion planning method focused on global exploration of the state-space. This allows the composition of simple locally optimal DMOC solution trajectories into near globally optimal motions that can handle complex, cluttered environments.

Chapter 1

Introduction

1.1 Problem setup

A *vehicle* is an actuated mechanical systems that moves, senses and performs a given task in the environment. The vehicle is equipped with sensors such as motor encoders, cameras, lasers or GPS. It uses these sensors to build a model of its surrounding, e.g. a map of the terrain, and to estimate its own state, e.g. its joint angles and its global position with respect to the environment. The following figure gives simplistic view of how a robotic vehicle operates. The robot moves around and *senses* the environment, then uses the sensor information to *make decisions* about the task to be executed, and then it *controls* itself in order to in order to complete the task. Consequently, its state

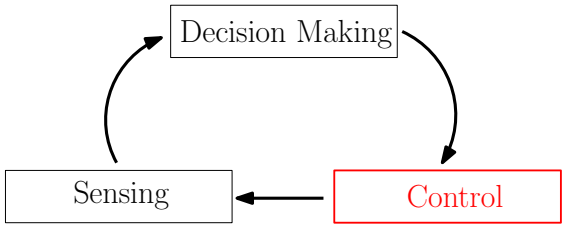


Figure 1.1: Vehicle operation

Figure 1.2: Segway RMP robot equipped with laser sensor, GPS, and omnidirectional camera (upper left). A typical scenario of following a fast moving person (upper right) and a view of the camera image of another scenario as the robot moves around the environment avoiding trees and tracking the correct person of interest

changes, it senses again, and so on. See Fig. 1.2 for an example of a vehicle performing person tracking and following outdoors.

This thesis focuses on the *control* part of the vehicle operation. The main control framework developed in this work relies on two major assumptions:

- the vehicle has perfect knowledge of the environment and its state,
- the vehicle dynamical model is known in advance.

Equivalently, it is assumed that accurate sensing is available and, and sensing uncertainty is not fully considered at the control stage. Nevertheless, sensing is extremely important and it is critical to account for the uncertainty in sensing and acting. Therefore, preliminary results in considering the effects of uncertainty at the control stage are included as an extension to our general framework.

The goal of this work is to develop numerical methods to *optimally* control vehicles to achieve a given task. The task could be basic such moving to a goal state with minimum fuel or more complex such as finding a target by maximizing the area covered by the vehicle sensors in minimum time.

In general, finding a solution to this types of problems is a difficult task. The main challenges lie in the fact that the vehicles can have complicated dynamics, are subject to constraints and underactuation, and must avoid colliding with complex obstacles. In contrast to local (or reactive) decision making and control, searching for an optimal solution requires the computation of a complete global trajectory from the vehicle current state to a desired state. This turns into the problem of searching among an infinite number of possible paths. In practice, the trajectory of the vehicle is approximated using some

finite-dimensional representation or discretization. A major issue in this work is how to construct such a *discretization* in order to properly account for the underlying dynamics and to provide a numerically stable and efficient numerical framework.

1.2 General approach and summary of results

Our general approach is based on a combination of standard optimal control techniques, and classical search and dynamic programming methods. These methods stand on top of a robust numerical representation of the underlying vehicle dynamics derived using the theory of *discrete mechanics*. The resulting formulation is general and applies to a variety of models widely used in robotics.

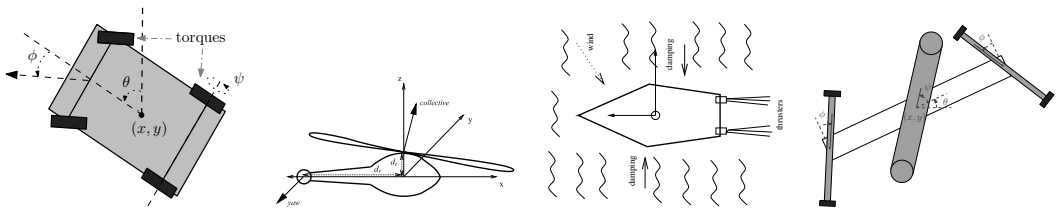
In essence, this thesis develops a general control framework through a principled approach that is applicable to various type of vehicles and environments. While general optimal control frameworks exist, the proposed framework possesses benefits previously unexplored in robotics. These can be summarizes as follows:

- structure-respecting geometric discretization of mechanical systems with symmetries, internal actuated shape, and nonholonomic constraints
- discrete optimal control formulation that respects the geometric structure
- more robust constraint handling through homotopy-continuation techniques
- combining the derived *local* optimal control techniques with global search methods in order to guarantee *near-globally* optimal solutions

- extending the basic motion planning framework to handle more specific tasks such as time-varying goal state, maximizing sensor coverage, deployment under goal motion uncertainty, planning for multiple vehicles

The theoretical framework is implemented in software and applied to several problems. Some of the developed simple models and simulated scenarios include (see Fig. 1.3)

- a helicopter flying to a goal state in a digital terrain with minimum control effort (e.g., related to fuel consumption)
- a car moving optimally while avoiding other cars and static obstacles
- a boat performing optimal station keeping subject to currents
- a snakeboard generating optimal maneuvers
- computing a statically stable reference trajectory for the LittleDog robot
- computing optimal trajectories for cars and helicopters in cluttered urban environment
- computing trajectories to moving targets with known dynamics
- computing a trajectory that maximizes the information gained about the uncertain position of a moving target in an environment with obstacles
- computing trajectory that maximizes the total area covered by a vehicle with a limited circular sensing radius among obstacles



Car Helicopter Boat Snakeboard

Figure 1.3: Examples of developed vehicle models.

Real Vehicles We must acknowledge the important fact that this thesis does not report any real-world experimental results. While this is unfortunate it does not mean that the proposed methods are not applicable to realistic applications. In fact, this whole work was motivated by the previous work of the author with real vehicles and the realization that a more unified, more optimal, and more principled numerical framework would be beneficial for the proper development of autonomous control techniques. Hence, the algorithms developed here are aimed at and could soon be applied to useful robotics applications. Yet, one of the biggest issues that must be addressed before the proposed methods can be successfully used in practice the handling of uncertainty. Sensing uncertainty and model/actuation uncertainty are inherent in real vehicles and at the end of the thesis we would touch upon these issues and mention preliminary results and ongoing work that would enable the application of the proposed discrete geometric control framework on real systems.

1.3 Outline

Ch. 2 deals with a simpler, restricted class of vehicles, i.e. systems with symmetries evolving on Lie groups, and develops their discretization and optimal control. Ch. 3

extends the discrete mechanical approach to systems with symmetries, shape variables, and constraints described in terms of a principle bundle and *connection* that geometrically encodes conservation laws and/or nonholonomic constraints. Ch. 4 proposes ways to extend DMOC for finding near globally optimal trajectories efficiently by combining optimal control solutions with classical probabilistic motion planning techniques. Ch. 5 deals exclusively with methods that increase the robustness of control methods in dealing with complex constraints such as obstacles in the environment or mechanical constraints. We propose novel (in the context of motion control) *homotopy-continuation* methods that increase the efficiency and radius of convergence of optimization-based methods through appropriate constraint deformation and embedding techniques.

Chapter 2

A Discrete Geometric Framework for Optimal Control on Lie groups

Summary

We consider the optimal control of mechanical systems on Lie groups that are left-invariant with respect to group actions. Our approach is based on the discretization of a Lagrange-d'Alembert-Pontryagin variational principle from which we derive structure-preserving discrete equations of motion.

We first apply a direct approach which uses the equations of motion as constraints in the optimization of a cost function (such as minimum control effort or minimum time) and results in a nonlinear programming problem. Then we propose a different formulation through the derivation of necessary conditions for optimality that results in a root-finding problem. In this indirect approach we use geometric methods to reduce the problem and transform it into numerically convenient form. Both approaches yield equivalent solutions but have different structure which motivates our numerical comparison.

We address the robustness and efficiency of the optimization algorithms by proposing strategies for trajectory initialization and trajectory refinement.

The optimal control framework is demonstrated with several examples: rigid body minimum control effort reorientation, boat navigation subject to external disturbances, optimal trajectory computation of a simple helicopter in a digital terrain map.

2.1 Introduction

We consider the optimal control of a mechanical system on a finite dimensional Lie group with Lagrangian that is left invariant under group actions. The goal is to move the system from its current state to a desired state in an optimal way, e.g. by minimizing distance, control effort, or time.

The standard way to solve such problems is to first derive the continuous equations of motion of the system. Among the trajectories satisfying these equations one can find extremal (cost function minimizing) ones by solving a variational problem. Either *direct* methods using nonlinear programming or *indirect* methods based on the derivation of necessary conditions for optimality and root-finding are used to find a solution. Both approaches require the discretization of the equations of motion into equality constraints suitable for numerical optimization. In contrast, we solve the optimal control problem by discretizing a variational principle, called Lagrange-d'Alembert-Pontryagin Principle. The principle yields a set of discrete trajectories that approximately satisfy the dynamics and that respect the state space structure. Among these trajectories we find the extremal ones without any further discretization. This allows important properties of the

mechanical system to be preserved and results in algorithms with provable accuracy and stability.

We develop a general optimization framework for systems on Lie groups and apply it to the computation of optimal rigid body motions on $SO(3)$, $SE(2)$ and $SE(3)$, as well as to general real matrix subgroups. In addition, we introduce techniques for trajectory initialization and refinement that increase the algorithms performance.

The Lagrangian Mechanical System

Let the configuration space be a Lie group G with algebra $\mathfrak{g}$ and Lagrangian $L : TG \to \mathbb{R}$ that is left invariant under the action of G. Using the invariance we can left-trivialize such systems by introducing the *body-fixed* velocity $\xi \in \mathfrak{g}$ defined by translation to the origin $\xi = TL_{g^{-1}}\dot{g}$ and the reduced Lagrangian $\ell : TG/G \to \mathbb{R}$ such that $\ell(\xi) = L(g^{-1}g, g^{-1}\dot{g}) = L(e, \xi)$.

The Optimization problem

The system is required to move from a fixed initial state $(g(0), \xi(0))$ to a fixed final state $(g(T), \xi(T))$ during a time interval $[0, T]$ under the influence of a body-fixed control force[1] $f(t) \in \mathfrak{g}^*$ (i.e. an internal force produced by actuators in the body reference frame) while minimizing:

$$J(g, \xi, f) = \int_0^T C(g(t), \xi(t), f(t))\mathrm{dt}, \tag{2.1}$$

[1] In the Lagrangian setting a force is an element of the Lie algebra dual $\mathfrak{g}^*$, i.e. a one-form $\langle f, \cdot \rangle$ that pairs with velocity vectors to produce the total work $\int_0^T \langle f, \xi \rangle dt$ done by the force along a path between $g(0)$ and $g(T)$.

where $C : G \times \mathfrak{g} \times \mathfrak{g}^* \to \mathbb{R}$ is a given cost function. For example, in case of minimum control effort $C(g(t), \xi(t), f(t)) = \frac{1}{2}\|f(t)\|^2$ or for minimum time problems $C(g(t), \xi(t), f(t)) = 1$.

2.1.1 Related work

Trajectory design and motion control of robotic systems have been studied from many different perspectives. Of particular interest are geometric approaches [31, 46, 51] that use symmetry and reduction techniques [43, 6]. Reduction by symmetry can be used to greatly simplify the optimal control problem and provide a framework to compute motions for general nonholonomic systems [47]. A related approach, applied to an underwater eel-like robot, involves finding approximate solutions using truncated basis of cyclic input functions [12]. There are a number of successful methods for motion planning with obstacles—see [35] for references.

While standard optimization methods are based on shooting, multiple shooting, or collocation techniques, recent work on Discrete Mechanics and Optimal Control (DMOC, see [28, 30, 36]) proposes a different discretization strategy. At the core of DMOC is the use of *variational integrators* [41] that are derived from the discretization of variational principles such as Hamilton's principle for conservative systems or Lagrange-D'Alembert for dissipative systems. Unlike other existing variational approaches [47, 15] where the continuous equations of motion are enforced as constraints and *subsequently* discretized, DMOC *first* discretizes the variational principles underlying the mechanical system dynamics; the resulting discrete equations are then used as constraints along with a discrete cost function to form the control problem. Because the discrete equations of motion result from a discrete variational principle, momenta preservation and symplecticity are

automatically enforced, avoiding numerical issues (like numerical dissipation) that generic algorithms often possess.

2.1.2 Contributions

In this chapter, we extend the generalized variational principle of [32, 5, 53] to the DMOC framework to derive optimization algorithms based on structure-preserving, discrete-mechanical integrators. In particular, we employ a discrete *Lagrange-d'Alembert-Pontryagin* principle to characterize mechanical systems with symmetries and external forces. We use this new discrete geometric optimal control framework for holonomic systems (possibly underactuated and/or with symmetries) and illustrate the implemented algorithms with a simulated example of a simplified helicopter flying through a canyon.

The numerical benefits of our discrete geometric approach are numerous. First, it automatically preserves motion invariants and geometric structures of the continuous system, exhibits good energy behavior, and respects the work-energy balance due to its variational nature. Such properties are often crucial for numerical accuracy and stability, in particular for holonomic systems such as underwater gliders traveling at low energies along ocean currents. Second, it benefits from an exact reconstruction of curves in the Lie group configuration space from curves in its Lie algebra. Thus, numerical drift, for example associated with enforcing rotation frame orthogonality constraints, is avoided. Third, the simplicity of the variational principle allows flexibility of implementation. Finally, this framework is flexible enough to strike a balance between a desired order of accuracy and runtime efficiency.

In addition to these well-documented advantages of discrete variational methods, there is growing evidence that DMOC methods are especially well suited for optimization problems. In particular, their discrete variational nature seems to offer very good trajectory approximation even at low temporal resolutions. This stability vis-a-vis resolution is particularly suitable for design and exploration purposes as well as for hierarchical optimizations, as it leads to faster convergence towards optimal solutions.

It is also important to note that non-holonomic constraints can also be imposed in our framework. While in this chapter we focus solely on holonomic systems with symmetries Ch. 3 contains initial results in the structure-preserving discretization of nonholonomic systems with symmetries and internal variables.

The derivation of an indirect method for solving the optimal control for systems on Lie groups is another contribution described in this chapter. An indirect formulation commonly requires the use of additional state variables, i.e. Lagrange multipliers, that enforce the constraints. The main disadvantage of using an indirect method lies in the increased dimension of the problem, in the lack of systematic way to initialize these extra variables, and in increased problem sensitivity. These negative features reduce the radius of convergence and stability of the optimization problem. Our discrete geometric indirect formulation, on the other hand, possesses certain structure that permits the removal of the multipliers and the option to redefine the problem in terms of its original optimization variables alone. This results in higher order optimality conditions with analogues form the continuous case [3] but with simpler and numerically convenient structure upon discretization. Standard nonlinear root-finding is then sufficient to find a

solution. Numerical evidence suggests that such a reduced problem does not suffer from the aforementioned issues and exhibits faster and more stable convergence.

2.2 Overview of Mechanical Integrators

A mechanical integrator integrates a dynamical system forward in time. The construction of such numerical algorithms usually involves some form of discretization or Taylor expansion that results in either implicit or explicit equations to compute the next state in time. In an optimal control setting, these equations are then used as constraints.

Instead, the integrators employed in this work are based on the discretization of variational principles, i.e. *variational integrators*. In essence, they ensure the optimality (in the sense of Hamilton's principle, for instance) of the discrete path of the mechanical system in space-time. In addition, certain systems have group structure and symmetries that can be factored out directly in order to obtain more accurate and efficient integrators, e.g. *Lie group integrators*. After giving a brief overview of such integrators below we present a variational principle to derive more general integrators that account for symmetries.

2.2.1 Variational Integrators

Variational integrators [41] are derived from a variational principle (e.g., Hamilton's principle) using a discrete Lagrangian. Unlike standard integration methods, variational integrators can preserve momenta, energy, and symplectic structure (i.e., a symplectic 2-form in phase space) for conservative systems; in the presence of forces and/or dissipation, they compute the change in these quantities with remarkable accuracy. Such features are obviously desirable for accurate dynamics simulation. The underlying theory has discrete

analogs of Noether's theorem and the Legendre transform, and a Lagrange-d'Alembert principle to handle non-conservative forces and constraints. *Discrete mechanics*, therefore, stands as a self-contained theory similar to Hamiltonian or Lagrangian mechanics [5] and has already been applied to several domains: nonsmooth variational collision integration [20], elasticity simulation in computer graphics [32], satellite formation trajectory design [29], optimal control of rigid bodies [36], of articulated bodies in fluid [30, 50], and optimal control of wheeled robots [33].

In the variational integration setting, the state space TQ is replaced by a product of two manifolds $Q \times Q$ [41]. Thus, a velocity vector $(q, \dot{q}) \in TQ$ is represented by a pair of points $(q_0, q_1) \in Q \times Q$. A path $q : [0, T] \to Q$ is replaced by a discrete path $q_d : \{kh\}_{k=0}^{N} \to Q$ ($q_d = \{q_0, ..., q_N\}$, $q_k = q(kh)$), $Nh = T$. One formulates a discrete version of Hamilton's principle (i.e. $\delta \int_0^T L(q, \dot{q})dt = 0$) by approximating the integral of the Lagrangian $L : TQ \to \mathbb{R}$ between q_k and q_{k+1} by a discrete Lagrangian $L_d : Q \times Q \to \mathbb{R}$

$$L_d(q_k, q_{k+1}) \approx \int_{kh}^{(k+1)h} L(q(t), \dot{q}(t))\mathrm{d}t.$$

The discrete principle then requires that

$$\delta \sum_{k=0}^{N-1} L_d(q_k, q_{k+1}) = 0,$$

where variations are taken with respect to each position q_k along the path, and the resulting equations of motion become

$$D_2 L_d(q_{k-1}, q_k) + D_1 L_d(q_k, q_{k+1}) = 0.$$

Example

For example, consider a Lagrangian of the form $L(q,\dot{q}) = \frac{1}{2}\dot{q}^T M\dot{q} - V(q)$ and define the discrete Lagrangian $L_d(q_k, q_{k-1}) = hL\left(q_{k+\frac{1}{2}}, (q_{k+1} - q_k)/h\right)$, using the notation $q_{k+\frac{1}{2}} := (q_k + q_{k+1})/2$. The resulting equations are

$$M\frac{q_{k+1} - 2q_k + q_{k-1}}{h^2} = -\frac{1}{2}(\nabla V(q_{k-\frac{1}{2}}) + \nabla V(q_{k+\frac{1}{2}})),$$

which is a discrete analog of Newton's law $M\ddot{q} = -\nabla V(q)$. For controlled (i.e., non conservative) systems, forces can be added using a discrete version of Lagrange-d'Alembert principle and discrete virtual work in a similar manner.

2.2.2 Lie Group Integrators

Lie group integrators preserve symmetry and group structure for systems with motion invariants. Consider a system on configuration manifold $Q = G \times M$ where G is a Lie group (with Lie algebra $\mathfrak{g}$) whose action leaves the system invariant, i.e., it preserves the induced momentum map. For example, $G = SE(3)$ can represent the group of rigid body motions of a free-floating articulated body while M is a space of internal variables describing the joints of the body. The idea is to transform the system equations from the original state space TQ into equations on the *reduced* space $\mathfrak{g} \times TM$ (elements of TG are translated to the origin and expressed in the algebra $\mathfrak{g}$) which is a linear space where standard integration methods can be used. The inverse of this transformation is then used to map curves in the algebra variables back to the group. Two standards maps have been commonly used to achieve this transformation for any Lie group G:

- Exponential map $\exp : \mathfrak{g} \to G$, defined by $\exp(\xi) = \gamma(1)$, with $\gamma : \mathbb{R} \to G$ is the integral curve through the identity of the left invariant vector field associated with $\xi \in \mathfrak{g}$ (hence, with $\dot{\gamma}(0) = \xi$);

- Canonical coordinates of the second kind $\text{ccsk} : \mathfrak{g} \to G$, $\text{ccsk}(\xi) = \exp(\xi^1 e_1) \cdot \exp(\xi^2 e_2) \cdot ... \cdot \exp(\xi^n e_n)$, where $\{e_i\}$ is the Lie algebra basis.

A third choice, valid only for certain *quadratic* matrix groups [7] (which include the rigid motion groups $SO(3)$, $SE(2)$, and $SE(3)$), is the Cayley map $\text{cay} : \mathfrak{g} \to G$, $\text{cay}(\xi) = (e - \xi/2)^{-1}(e + \xi/2)$. Although this last map provides only an approximation to the integral curve defined by exp, we include it as one of our choices since it is very easy to compute and thus results in a more efficient implementation. Other approaches are also possible, e.g., using retraction and other commutator-free methods; we will however limit our exposition to the three aforementioned maps in the formulation of the discrete reduced principle presented in the next section.

2.3 Lagrange-d'Alembert-Pontryagin Principle

The Lagrange-d'Alembert-Pontryagin (LDAP) principle is a generalization of the Lagrange-d'Alembert variational principle and yields equivalent solution trajectories. The LDAP principle, however, provides additional freedom in the choice of variations which turns out to be crucial for obtaining symmetry and group structure preserving integrators and solving optimal control problems based on such integrators.

Continuous Reduced LDAP Principle

Define the reduced path $(g, \xi, \mu) : [0, T] \to G \times \mathfrak{g} \times \mathfrak{g}^*$, and the control force $f : [0, T] \to \mathfrak{g}^*$. The principle requires that

$$\delta \int_0^T \left[\ell(\xi) + \langle \mu, g^{-1}\dot{g} - \xi \rangle \right] dt + \int_0^T \left[TL^*_{g^{-1}} f \cdot \delta g \right] dt = 0, \tag{2.2}$$

for variations $\delta g(t)$, $\delta \xi(t)$, $\delta \mu(t)$ that vanish at the endpoints. The curve $\xi(t)$ describes the velocity determined from the dynamics of the system. In view of the formulation, ξ does not necessary correspond to the left-trivialized rate of change of the configuration g. The additional variables μ, though, indirectly enforce this dependence and correspond to both Lagrange multipliers and the momenta of the system. Thus, the principle generalizes Lagrange-d'Alembert principle and is linked to Pontryagin maximum principle of optimal control.

After taking variations the continuous equations of motion become

$$\begin{aligned} \dot{\mu} - \mathrm{ad}^*_\xi \mu &= f, \\ \mu &= \ell'(\xi), \\ \dot{g} &= g\xi \end{aligned} \tag{2.3}$$

These equations[2] are called the Euler-Poincaré equations and μ denotes the system momentum.

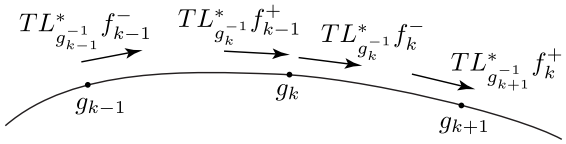
[2] $\mathrm{ad}^*_\xi \mu$ is defined by $\langle \mathrm{ad}^*_\xi \mu, \eta \rangle = \langle \mu, \mathrm{ad}_\xi \eta \rangle$, where $\mathrm{ad}_\xi \eta = [\xi, \eta]$, $\eta \in \mathfrak{g}$.

Discrete Reduced LDAP Principle

The *discrete* reduced Hamilton-Pontryagin principle for conservative systems was introduced in [5]. We make an elementary extension to these results for systems with internal forces. The *discrete reduced path*[3] is denoted by $(g, \xi, \mu)_{0:N} : \{t_k\}_{k=0}^{N} \to G \times \mathfrak{g} \times \mathfrak{g}^*$. Define the discrete control force $f_{0:N} : \{t_k\}_{k=0}^{N} \to \mathfrak{g}^*$ which approximates a continuous control force. The discrete reduced LDAP principle is formulated as:

$$\begin{aligned} &\delta \sum_{k=0}^{N-1} h \left[\ell(\xi_k) + \left\langle \mu_k, \tau^{-1}(g_k^{-1} g_{k+1})/h - \xi_k \right\rangle \right] \\ &+ \sum_{k=0}^{N-1} \left[TL^*_{g_k^{-1}} f_k^- \cdot \delta g_k + TL^*_{g_{k+1}^{-1}} f_k^+ \cdot \delta g_{k+1} \right] = 0, \end{aligned} \tag{2.4}$$

where the map $\tau : \mathfrak{g} \to G$ relates Lie algebra elements to *discrete changes*[4] in the group configuration. τ is selected as a local diffeomorphism such that $\tau(\xi) \cdot \tau(-\xi) = e$ [5]. The *left* and *right* discrete forces $f_k^- \in \mathfrak{g}^*$ and $f_k^+ \in \mathfrak{g}^*$ (as shown below)



are such that the work done by f along each discrete segment is approximated using the following quadrature (see [41] for more details):

$$TL^*_{g_k^{-1}} f_k^- \cdot \delta g_k + TL^*_{g_{k+1}^{-1}} f_k^+ \cdot \delta g_{k+1} \approx \int_{kh}^{(k+1)h} TL^*_{g(t)^{-1}} f(t) \cdot \delta g(t) dt$$

[3]The term *discrete path* $x_{0:N}$ is also used to denote the set of all discrete configurations, i.e. $x_{0:N} = \{x_0, ..., x_N\}$

[4]Essentially, the choice of τ answers the question: How do we define the *difference* between two configurations g_k and g_{k+1} on the curved space, and based on that how do we compute the velocity along the segment between them.

The total work is therefore approximated by

$$\int_0^T TL^*_{g(t)^{-1}} f(t) \cdot \delta g(t) dt \approx \sum_{k=0}^{N} TL^*_{g_k^{-1}} \widetilde{f}_k \cdot \delta g_k, \tag{2.5}$$

where $\widetilde{f}_{0:N} : \{t_k\}_{k=0}^N \to \mathfrak{g}^*$ is the resulting force at each discrete point and is defined by

$$\begin{aligned} \widetilde{f}_0 &:= f_0^-, \\ \widetilde{f}_k &:= f_{k-1}^+ + f_k^-, \quad k = 1, .., N-1, \\ \widetilde{f}_N &:= f_N^+. \end{aligned} \tag{2.6}$$

After taking variations in (2.4) we obtain the following discrete equations of motion (see Sec. 4.2 in [5]).

$$(\mathrm{d}\tau^{-1}_{h\xi_k})^* \mu_k - (\mathrm{d}\tau^{-1}_{-h\xi_{k-1}})^* \mu_{k-1} = \widetilde{f}_k, \qquad k = 1, ..., N-1, \tag{2.7}$$

$$\mu_k = \ell'(\xi_k), \qquad k = 0, ..., N-1, \tag{2.8}$$

$$g_k^{-1} g_{k+1} = \tau(h\xi_k), \qquad k = 0, ..., N-1, \tag{2.9}$$

where $\mathrm{d}\tau_\xi : \mathfrak{g} \to \mathfrak{g}$ is the *right-trivialized tangent* of $\tau(\xi)$ defined by

$$\mathrm{D}\,\tau(\xi) \cdot \delta = TR_{\tau(\xi)}(\mathrm{d}\tau_\xi \cdot \delta), \tag{2.10}$$

and $\mathrm{d}\tau_\xi^{-1} : \mathfrak{g} \to \mathfrak{g}$ is its inverse[5]. Equations (2.7)-(2.9) can be considered as a discrete approximation to equations (2.3).

[5] D is the standard derivative in the direction δ

In addition, using arguments similar to Sec.3.2.3 in [41] the discrete forced Noether theorem yields the following boundary conditions

$$(\mathrm{d}\tau^{-1}_{h\xi_0})^*\mu_0 - \ell'(\xi(0)) = \widetilde{f}_0, \tag{2.11}$$

$$\ell'(\xi(T)) - (\mathrm{d}\tau^{-1}_{-h\xi_{N-1}})^*\mu_{N-1} = \widetilde{f}_N, \tag{2.12}$$

where $\xi(0)$ and $\xi(T)$ are the initial and final velocities. Note the distinction between ξ_0 and $\xi(0)$ and ξ_{N+1} and $\xi(T)$. These quantities are not necessary the same since $\xi(t)$ refers to the point on the continuous curve at time t while ξ_k can be thought of as an average velocity along the k-th trajectory segment resulting from the discretization.

The exact form of (2.7), (2.11), and (2.12) depends on the choice of τ. It is important to point out that this choice will influence the computational efficiency of the optimization framework when the equalities above are enforced as constraints. There are several choices commonly used for integration on Lie groups: the exponential map, canonical coordinates of the second kind (CCSK), and the Cayley map. In this work we employ the first and the third types of methods (CCSK methods are based on the exponential map and can be considered closely related).

Exponential map

The exponential map $\exp : \mathfrak{g} \to G$ is defined by $\exp(\xi) = \gamma(1)$, where $\gamma : \mathbb{R} \to G$ is a one-parameter subgroup of G such that $\dot{\gamma}(0) = \xi$. The right-trivialized derivative of the map exp and its inverse are defined as

$$\operatorname{dexp}_x y = \sum_{j=0}^{\infty} \frac{1}{(j+1)!} \operatorname{ad}_x^j y, \quad \operatorname{dexp}_x^{-1} y = \sum_{j=0}^{\infty} \frac{B_j}{j!} \operatorname{ad}_x^j y, \tag{2.13}$$

where B_j are the Bernoulli numbers. Typically, these expressions are truncated in order to achieve a desired order of accuracy. The first few Bernoulli numbers are $B_0 = 1$, $B_1 = -1/2$, $B_2 = 1/6$, $B_3 = 0$ (see [7, 24] for more details). Setting $\tau = \exp$, (2.7) becomes

$$(\operatorname{dexp}_{h\xi_k}^{-1})^* \mu_k - (\operatorname{dexp}_{-h\xi_{k-1}}^{-1})^* \mu_{k-1} = \widetilde{f}_k.$$

Cayley map

The Cayley map $\operatorname{cay} : \mathfrak{g} \to \mathbb{R}$ is defined by $\operatorname{cay}(\xi) = (\mathrm{I} - \xi/2)^{-1}(\mathrm{I} + \xi/2)$. Based on this simple form, the derivative maps become [6]

$$\operatorname{dcay}_x y = \left(\mathrm{I} - \frac{x}{2}\right)^{-1} y \left(\mathrm{I} + \frac{x}{2}\right)^{-1}, \quad \operatorname{dcay}_x^{-1} y = \left(\mathrm{I} - \frac{x}{2}\right) y \left(\mathrm{I} + \frac{x}{2}\right). \tag{2.14}$$

Using $\tau = \operatorname{cay}$ (see also [5]) (2.7) simplifies to

$$\mu_k - \mu_{k-1} - \frac{h}{2}\left(\operatorname{ad}_{\xi_k}^* \mu_k + \operatorname{ad}_{\xi_{k-1}}^* \mu_{k-1}\right) - \frac{h^2}{4}\left(\xi_k^* \mu_k \xi_k^* - \xi_{k-1}^* \mu_{k-1} \xi_{k-1}^*\right) = \widetilde{f}_k. \tag{2.15}$$

[6] see Sec.IV.8.3 in [24] for derivation

The Cayley map provides only an approximation to the geodesic flow on the group (or to the exponential map).

2.4 Direct Optimal Control Formulation

The most straightforward way to find a numerical solution to the optimal control problem is to formulate a nonlinear program that minimizes the cost function over all discrete configurations, velocities, and forces, while satisfying the boundary conditions and the discrete equations of motion enforced as equality constraints.

2.4.1 Problem Formulation

The optimal control problem for a system with reduced Lagrangian $\ell : \mathfrak{g} \to \mathbb{R}$ and fixed initial and final states $(g(0), \xi(0))$ and $(g(T), \xi(T))$ respectively can be directly formulated as

$$
\begin{aligned}
&\textbf{Compute:} \quad \xi_{0:N-1}, f_{0:N}, h \\
&\textbf{minimizing } J_d(g_{0:N}, \xi_{0:N-1}, f_{0:N}, h) = \sum_{k=0}^{N-1} C_d(g_k, \xi_k, f_k, h) \\
&\textbf{subject to:} \\
&\begin{cases}
(\mathrm{d}\tau_{h\xi_0}^{-1})^* \mu_0 - \ell'(\xi(0)) = \widetilde{f}_0, \\
(\mathrm{d}\tau_{h\xi_k}^{-1})^* \mu_k - (\mathrm{d}\tau_{-h\xi_{k-1}}^{-1})^* \mu_{k-1} = \widetilde{f}_k, \quad k = 1, ..., N-1, \\
\ell'(\xi(T)) - (\mathrm{d}\tau_{-h\xi_{N-1}}^{-1})^* \mu_{N-1} = \widetilde{f}_N, \\
\mu_k = \ell'(\xi_k), \\
g_0 = g(0), \\
g_{k+1} = g_k \tau(h\xi_k), \quad k = 0, ..., N-1, \\
\tau^{-1}(g_N^{-1} g(T)) = 0, \\
\xi_k \in [\xi_l, \xi_u], f_k \in [f_l, f_u], h \in [h_l, h_u]
\end{cases}
\end{aligned}
\tag{2.16}
$$

where C_d is a discrete approximation of C defined in (2.1). The formulation allows time to vary and the last constraint places bounds on the time variation as well as bounds on all other variables.

Average velocity

The variables denoted ξ_N and μ_N have no effect on the trajectory $g_{0:N}$ so we can treat these last points as irrelevant to the optimization. This is coherent with thinking of the velocities ξ_k as the average body-fixed velocity along the k-th path segment between configurations g_k and g_{k+1}.

2.4.2 Algorithm Construction

Midpoint rule

We choose the midpoint rule as a natural way to construct $f^{\pm}$ and C_d. According to the midpoint rule any discrete quantities defined over a trajectory segment are approximated at the segment midpoint. Therefore, the left and right discrete forces at each segment are equal:

$$f_k^- = f_k^+ = \frac{h}{2}\frac{1}{2}(f_k + f_{k+1}),$$

The discrete cost function then becomes

$$C_d(g_k, \xi_k, f_k, h) = hC\left(g_{k+\frac{1}{2}}, \xi_k, \frac{1}{2}(f_k + f_{k+1})\right)$$

where $g_{k+\frac{1}{2}} = g_k\tau(\frac{h}{2}\xi_k)$, i.e. the midpoint along the flow defined by τ.

Remark

There are other choices besides the Midpoint rule that can lead to integrators of arbitrary high order of accuracy, such as composition methods or symplectic Runga-Kutta

methods [41]. The midpoint rule is an appropriate choice for optimization problem since it provides a balance between accuracy and efficiency.

Implementation

The optimal control formulation (2.16) can be solved using a standard constrained optimization technique such as sequential quadratic programming (SQP). One should choose a sparse SQP implementation to achieve scalability in the number of time steps. Sec. 2.10.1 provides empirical results and analysis of our implementation for an example rigid body control problem.

2.5 Indirect Approach

In contrast to the direct approach, the optimal control problem can be solved by further analysis of solution trajectories through the derivation of conditions for optimality. An optimal solution is then the root of a system of nonlinear equations corresponding to these optimality conditions.

In this section we derive necessary conditions for optimality for the constrained nonlinear optimization problem (2.16) under some restrictions. The first restriction is to assume that the reduced Lagrangian is of the form

$$\ell(\xi) = \frac{1}{2}\langle \mathbb{I}\,\xi, \xi \rangle, \tag{2.17}$$

where $\mathbb{I} : \mathfrak{g} \to \mathfrak{g}^*$ is a symmetric positive definite linear map[7]. The second restriction is to specialize our derivation to systems with fixed time-step h, and with minimum control effort cost function defined by

$$J_d(g_{0:N}, \xi_{0:N-1}, f_{0:N}) = \sum_{k=0}^{N} \frac{1}{2} \|\widetilde{f}_k\|^2, \tag{2.18}$$

where $\widetilde{f}_k$ was defined in (2.6). Such cost function is physically meaningful since it corresponds to the sum of the squared norm of the approximating force evaluated at each discrete point (see (2.5)).

Since the evolution of the Lie algebra variables (2.7) is *decoupled* from the evolution on the group (2.9) we can remove the discrete group path $g_{0:N}$ from the optimization state vector and express the boundary conditions $g_0 = g(0)$ and $g_N = g(T)$ in terms of $\xi_{0:N-1}$.

Define the variables

$$\begin{aligned}
&\mathrm{DEP}_0 := (\mathrm{d}\tau^{-1}_{h\xi_0})^* \ell'(\xi_0) - \ell'(\xi(0)), && \% \text{ initial momentum} \\
&\mathrm{DEP}_k := (\mathrm{d}\tau^{-1}_{h\xi_k})^* \ell'(\xi_k) - (\mathrm{d}\tau^{-1}_{-h\xi_{k-1}})^* \ell'(\xi_{k-1}), && \% \text{ momentum equation} \\
&\mathrm{DEP}_N := \ell'(\xi(T)) - (\mathrm{d}\tau^{-1}_{-h\xi_{N-1}})^* \ell'(\xi_{N-1}), && \% \text{ final momentum} \\
&\mathrm{REC} := \tau^{\text{-}1} \left(\tau(h\xi_0) \cdots \tau(h\xi_{N-1}) \cdot (g(0)^{-1} g(T))^{-1} \right), && \% \text{ reconstruction}
\end{aligned}$$

[7] $\langle , \rangle$ is the standard pairing between covectors and vectors of coordinates in the chosen Lie algebra basis

for $k = 1, ..., N-1$. The optimization problem is to compute $\xi_{0:N-1}$ and $f_{0:N}$ that minimize (2.18) subject to

$$\begin{aligned} &\mathrm{DEP}_k = \widetilde{f}_k, \quad k = 0, ..., N, \\ &\mathrm{REC} = 0. \end{aligned} \tag{2.19}$$

Clearly, the quantity DEP_0 is used to define the initial velocity boundary condition, DEP–the discrete Euler-Poincaré equations or momentum balance equations along the path, DEP_N–the final velocity boundary condition, and REC–to define the reconstruction equation.

Proposition 1. *The gradients of the constraints* (2.19) *are linearly independent.*

Proof. The proof follows directly from the fact that the DEP_k constraints have full rank dependence on different variables for each k (i.e. each equation is a function of different pairs ξ_k, ξ_{k+1}, and f_k, f_{k+1}) and that the maps $\tau(\xi)$, $\mathrm{d}\tau_\xi$, and $\mathrm{d}\tau_\xi^{-1}$ are non-singular for any ξ. □

This proposition guarantees linear independence constraint qualification (LICQ) condition. Therefore, we can assume that necessary conditions for optimality do not include *abnormal* solutions (i.e. solutions that satisfy the constraints but do not account for the cost function).

Furthermore, based on the above definitions, the cost function (2.18) can be re-expressed as

$$J_d(g_{0:N}, \xi_{0:N-1}, f_{0:N}) = \frac{1}{2} \sum_{k=0}^{N} \| \mathrm{DEP}_k \|^2,$$

and since now J_d is independent of the configurations and forces, the optimization problem can be stated as minimizing $J_d(\xi_{0:N-1}) := J_d(g_{0:N}, \xi_{0:N-1}, f_{0:N})$ subject to $\mathrm{REC} = 0$. Thus, the optimization can be performed with respect to the velocity variables alone.

Define the Lagrangian multipliers $\lambda \in \mathfrak{g}^*$ and the Hamiltonian function

$$H(\xi_{0:N-1}, \lambda) = J_d(\xi_{0:N-1}) + \langle \lambda, \mathrm{REC}(\xi_{0:N-1}) \rangle \tag{2.20}$$

By Prop. (1) any optimal solution is normal and, therefore, must satisfy $\delta H = 0$. Hence, we have the following *necessary conditions* for an optimal solution

$$\begin{aligned} &\mathrm{D}_{\xi_k} H \cdot \delta\xi_k = 0, \qquad k = 0, ..., N-1, \\ &\mathrm{D}_{\lambda} H \cdot \delta\lambda = 0, \end{aligned} \tag{2.21}$$

for arbitrary variations $\delta\xi_k \in \mathfrak{g}$, $\delta\lambda \in \mathfrak{g}^*$. These conditions are necessary but not sufficient because, in general, the constraints are not affine and global optimality is not guaranteed.

Further analysis of (2.21) requires the following lemmas.

Lemma 1. *The following relation holds for any* $\xi, \nu, \delta \in \mathfrak{g}$

$$\mathrm{D}_{\xi} \left(\mathrm{Ad}_{\tau(\xi)} \nu \right) \cdot \delta = \mathrm{ad}_{(\mathrm{d}\tau_{\xi}(\delta))} \mathrm{Ad}_{\tau(\xi)} \nu$$

Proof. Using the identity $\mathrm{d}\tau_{-\xi}(\delta) = \mathrm{Ad}_{\tau(-\xi)} \mathrm{d}\tau_{\xi}(\delta)$ (see [5]) we get

$$\begin{aligned} \mathrm{D}_{\xi} \left(\mathrm{Ad}_{\tau(\xi)} \nu \right) \cdot \delta &= (\mathrm{d}\tau_{\xi}(\delta))\tau(\xi) \cdot \nu \cdot \tau(-\xi) - \tau(\xi) \cdot \nu \cdot (\mathrm{d}\tau_{-\xi}(\delta))\tau(-\xi) \\ &= (\mathrm{d}\tau_{\xi}(\delta)) \mathrm{Ad}_{\tau(\xi)} \nu - \tau(\xi) \cdot \nu \cdot \tau(-\xi)(\mathrm{d}\tau_{\xi}(\delta)) = [\mathrm{d}\tau_{\xi}(\delta), \mathrm{Ad}_{\tau(\xi)} \nu] \end{aligned}$$

□

Lemma 2. *The differential of the reconstruction operator* REC_τ *along a path* $\xi_{0:N-1}$ *in the direction of variations* $\delta\xi_k$ *is*

$$\mathrm{D}_{\xi_k}\,\mathrm{REC}_\tau(\xi_{0:N-1})\cdot\delta\xi_k = \mathrm{d}\tau_{\tau^{-1}(\Delta g)}(h\,\mathrm{Ad}_{\tau(h\xi_0)\cdots\tau(h\xi_{k-1})}\,\mathrm{d}\tau_{h\xi_k}(\delta\xi_k)),\quad k=1,...,N-2,$$

where $\Delta g = \tau(h\xi_0)\cdots\tau(h\xi_{N-1})\cdot(g_i^{-1}g_f)^{-1}$

Proof. Differentiating

$$\begin{aligned}
&\mathrm{D}_{\xi_k}\,\mathrm{REC}_\tau(\xi_{0:N-1})\cdot\delta\xi_k\\
&=\mathrm{D}\,\tau^{-1}(\Delta g)\cdot\tau(h\xi_0)\cdots\tau(h\xi_{k-1})\cdot\mathrm{d}\tau_{h\xi_k}(h\delta\xi_k)\cdot\tau(h\xi_k)\cdots\tau(h\xi_{N-1})\cdot(g_i^{-1}g_f)^{-1}\\
&=\mathrm{d}\tau^{-1}_{\tau^{-1}(\Delta g)}(h\,\mathrm{Ad}_{\tau(h\xi_0)\cdots\tau(h\xi_{k-1})}\,\mathrm{d}\tau_{h\xi_k}(\delta\xi_k))
\end{aligned}$$

□

The following proposition is the main result in this section.

Proposition 2. *The trajectory of a discrete mechanical system on a Lie group* G *with algebra* $\mathfrak{g}$ *and Lagrangian* $\ell(\xi_k)=\frac{1}{2}\langle\mathbb{I}\,\xi_k,\xi_k\rangle$*, with fixed initial and final configurations and velocities* $g(0)\in G$*,* $\xi(0)\in\mathfrak{g}$ *and* $g(T)\in G$*,* $\xi(T)\in\mathfrak{g}$*, minimizes the total control effort* $\sum_{k=0}^{N}\frac{1}{2}\|\widetilde{f}_k\|^2$ *only if the discrete body-fixed velocity curve* $\xi_{0:N-1}$ *satisfies the following*

Necessary Conditions for Optimality

$$(\mathrm{d}\tau_{h\xi_k}^{-1})^*\eta_k - (\mathrm{d}\tau_{-h\xi_{k-1}}^{-1})^*\eta_{k-1} = 0, \qquad k = 1, ..., N-1 \tag{2.22}$$

$$\tau^{-1}(\tau(h\xi_0)\cdots\tau(h\xi_{N-1})\cdot(g(0)^{-1}g(T))^{-1}) = 0, \tag{2.23}$$

where $\eta_k \in \mathfrak{g}^*$ *is defined by*

$$\begin{aligned}\langle \eta_k, \delta\rangle = &\left\langle \mathbb{I}(\mathrm{d}\tau_{h\xi_k}^{-1}(\nu_k)) - h(\mathrm{d}\tau_{h\xi_k})^* \operatorname{ad}^*_{\left(\mathrm{Ad}_{\tau(h\xi_k)}\, \widetilde{f}_{k+1}^{\,\sharp}\right)} (\mathrm{d}\tau_{h\xi_k}^{-1})^*\,\mathbb{I}(\xi_k), \delta \right\rangle \\ &+ \left\langle \mathbb{I}(\xi_k), h\left(\mathrm{D}\,\mathrm{d}\tau_{h\xi_k}^{-1}\cdot\delta\right)(\nu_k)\right\rangle,\end{aligned} \tag{2.24}$$

$$\nu_k = \mathrm{Ad}_{\tau(h\xi_k)}\, \widetilde{f}_{k+1}^{\,\sharp} - \widetilde{f}_k^{\,\sharp}, \tag{2.25}$$

$$\widetilde{f}_0 = (\mathrm{d}\tau_{h\xi_0}^{-1})^*\,\mathbb{I}(\xi_0) - \mathbb{I}(\xi(0)), \tag{2.26}$$

$$\widetilde{f}_k = (\mathrm{d}\tau_{h\xi_k}^{-1})^*\,\mathbb{I}(\xi_k) - (\mathrm{d}\tau_{-h\xi_{k-1}}^{-1})^*\,\mathbb{I}(\xi_{k-1}), \qquad k = 1, ..., N-1 \tag{2.27}$$

$$\widetilde{f}_N = \mathbb{I}(\xi(T)) - (\mathrm{d}\tau_{-h\xi_{N-1}}^{-1})^*\,\mathbb{I}(\xi_{N-1}), \tag{2.28}$$

Note: The proposition defines Nn *equations* (2.22)-(2.23) *in the* Nn *unknowns* $\xi_0, ..., \xi_{N-1}$.

A solution can be found using standard nonlinear root finding.

Proof. The condition $\mathrm{D}_{\xi_k} H\cdot\delta = 0$ for an arbitrary $\delta \in \mathfrak{g}$ is equivalent to

$$\mathrm{D}_{\xi_k}\left\langle \mathbb{I}(\xi_k), \mathrm{d}\tau_{h\xi_k}^{-1}(\mathrm{Ad}_{\tau(h\xi_k)}\, \widetilde{f}_{k+1}^{\,\sharp} - \widetilde{f}_k^{\,\sharp})\right\rangle\cdot\delta = \left\langle \lambda, \mathrm{d}\tau_{\tau^{-1}(\Delta g)}^{-1}(h\,\mathrm{Ad}_{\tau(h\xi_0)\cdots\tau(h\xi_{k-1})}\,\mathrm{d}\tau_{h\xi_k}(\delta))\right\rangle,$$

for $k = 0, ..., N-1$ using the relation $\mathrm{d}\tau^{-1}_{-\xi}(\delta) = \mathrm{d}\tau^{-1}_{\xi}(\mathrm{Ad}_{\tau(\xi)}\,\delta)$ [5] and Lemma (2). Setting $\nu_k = \mathrm{Ad}_{\tau(h\xi_k)}\,\widetilde{f}^{\,\sharp}_{k+1} - \widetilde{f}^{\,\sharp}_{k}$ and applying the derivative on the left hand side in the above we get

$$\begin{aligned}
&\left\langle \mathbb{I}(\mathrm{d}\tau^{-1}_{h\xi_k}(\widetilde{\nu}_k)), \delta \right\rangle + \left\langle \mathbb{I}(\xi_k), h\left(\mathrm{D}\,\mathrm{d}\tau^{-1}_{h\xi_k}\cdot\delta\right)(\nu_k) + h\,\mathrm{d}\tau^{-1}_{h\xi_k}(\mathrm{ad}_{(\mathrm{d}\tau_{h\xi_k}(\delta))}\,\mathrm{Ad}_{\tau(h\xi_k)}\,\widetilde{f}^{\,\sharp}_{k+1}) \right\rangle \\
&= \left\langle h(\mathrm{d}\tau_{h\xi_k})^*\,\mathrm{Ad}^*_{\tau(h\xi_0)\cdots\tau(h\xi_{k-1})}(\mathrm{d}\tau^{-1}_{\tau^{-1}(\Delta g)})^*\lambda, \delta \right\rangle,
\end{aligned} \tag{2.29}$$

where have used the symmetry of $\mathbb{I}$ in the first pairing and Lemma (1) in the second. Now define the *k-th reconstruction force* $\eta_k = h(\mathrm{d}\tau_{h\xi_k})^*\,\mathrm{Ad}^*_{\tau(h\xi_0)\cdots\tau(h\xi_{k-1})}(\mathrm{d}\tau^{-1}_{\tau^{-1}(\Delta g)})^*\lambda$. Substituting it in (2.29) we find the quantity $\langle \eta_k, \delta\rangle$ as defined by (2.24). The components of η_k can be directly computed using $(\eta_k)_\alpha = \langle \eta_k, E_\alpha\rangle$, where $\{E_\alpha\}_{\alpha=1}^n$ is the chosen Lie algebra basis. By the definition of η_k it is true that $\eta_k = \mathrm{Ad}^*_{\tau(h\xi_{k-1})}\,\eta_{k-1}$ and this relation is added as an optimality condition (2.22).

The condition $\mathrm{D}_\lambda H = 0$ simply enforces the constraint (2.23). □

Geometric Interpretation We can think of the whole discrete trajectory as a single mechanical system consisting of a chain of $N+1$ bodies, with the last body rigidly fixed at configuration $g(T)$ and a force λ applied at configuration $g(0)$ to keep the first body fixed there. If the first body detaches from $g(0)$ the instantaneous force that is pulling it back to $g(0)$ is $(\mathrm{d}\tau^{-1}_{\tau^{-1}(\Delta g)})^*\lambda$. This force has different representation on each of the remaining N bodies along the chain, i.e. through the $\mathrm{Ad}^*_{\tau(h\xi_0)\cdots\tau(h\xi_k)}$ transformation. The dynamics of the interaction between the k-th and $(k+1)$-th bodies is encoded by the velocities ξ_{k-1}, ξ_k, and ξ_{k+1} and the reconstruction force η_k (which is simply the transformed λ) and we

are able to express this force in terms of these three velocities through (2.24). When we do this for every segment along the chain we arrive at the relations (2.22) since it is the same force, just transformed along the chain. Thus, the original force λ is implicitly accounted for through the interactions of the bodies and need not be part of the optimization.

Implementing the necessary conditions Implementation of equations (2.22)-(2.23) in Prop. (2) requires defining the maps τ, $\mathrm{d}\tau^{-1}$, and $D\,\mathrm{d}\tau^{-1}$. Two choices for τ discussed in Sec. 2.3 were the exponential map exp and the Cayley map cay and formulas for $\mathrm{d}\tau^{-1}$ were provided in (2.13) and (2.14) respectively. $\mathrm{D}\,\mathrm{d}\tau^{-1}$ can be computed as

$$(D\,\mathrm{dcay}_{\xi}^{-1}\cdot\delta')(\delta) = -\frac{1}{2}\,\mathrm{ad}_{\delta'}\,\delta - \frac{1}{4}(\xi\delta\delta' + \delta'\delta\xi)$$

$$(D\,\mathrm{d}\exp_{\xi}^{-1}\cdot\delta')(\delta) = -\frac{1}{2}\,\mathrm{ad}_{\delta'}\,\delta + \frac{1}{12}\,(\mathrm{ad}_{\delta'}\,\mathrm{ad}_{\xi}\,\delta + \mathrm{ad}_{\xi}\,\mathrm{ad}_{\delta'}\,\delta) + \ldots,$$

where the expression for $\mathrm{D}\,d\exp^{-1}$ can either be truncated to achieve a desired order of accuracy, or for groups such as rigid body motions can be redefined in closed form. In the next section we provide more details about the algorithm implementation for the matrix groups $SO(3)$, $SE(2)$, $SE(3)$ whose algebra can be identified with $\mathbb{R}^3$ for the first two groups and $\mathbb{R}^3 \times \mathbb{R}^3$ for the third. In these cases the operators Ad, ad, $\mathrm{d}\tau^{-1}$ and $\mathrm{D}\,\mathrm{d}\tau^{-1}$ become matrices suitable for efficient implementation.

Comparison to the Continuous Case Necessary conditions for the continuous analog of the optimal control problem, i.e. minimizing $\int \frac{1}{2}\|f\|^2 dt$ subject to the Euler-Poincaré equations (2.3) and boundary conditions have been derived (e.g. [3]). For example, in the case of semi-simple compact groups the necessary conditions become

$$\frac{D^2}{\partial t^2}\mathbb{I}\frac{D\xi}{\partial t} + R\left(\mathbb{I}\frac{D\xi}{\partial t}, \xi\right)\xi + \frac{1}{2}\left[\frac{D\xi}{\partial t}, \mathbb{I}\frac{D\xi}{\partial t}\right] + \frac{1}{2}\mathbb{I}^{-1}\left[\mathbb{I}^2\frac{D\xi}{\partial t}, \frac{D\xi}{\partial t}\right] = 0, \tag{2.30}$$

where $\frac{D}{\partial t}$ is the covariant time derivative and R is the curvature. The discrete version of these equations shows up as (2.22) in Prop. 2. Numerical discretization of (2.30), even in this simple case, would result into equations that would be no less complex than the ones in Prop. 2. Furthermore, our formulation remains unchanged for non-compact groups and, in fact, is readily applicable to general matrix subgroups as discussed below.

2.6 Computation on Matrix Groups

2.6.1 $SO(3)$

Define the map $\widehat{\cdot}: \mathbb{R}^3 \to \mathfrak{so}(3)$ by

$$\widehat{\omega} = \begin{bmatrix} 0 & -w_3 & w_3 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix} \tag{2.31}$$

A Lie algebra basis for $SO(3)$ can be defined as $\{\widehat{e}_1, \widehat{e}_2, \widehat{e}_3\}$, $\widehat{e}_i \in \mathfrak{so}(3)$ where $\{e_1, e_2, e_3\}$ is the standard basis for $\mathbb{R}^3$. Elements $\xi \in \mathfrak{so}(3)$ can be identified with the basis coordinates

$\omega \in \mathbb{R}^3$, i.e. $\xi = \omega^\alpha \widehat{e}_\alpha$, or $\xi = \widehat{\omega}$. The commutator is identified with the cross product through $[\widehat{\omega}, \widehat{\rho}] = \omega \times \rho$, $\rho \in \mathbb{R}^3$. Using this identification we have

$$\operatorname{cay}(\widehat{\omega}) = \mathbf{I}_3 + \frac{4}{4 + \|\omega\|^2} \left(\widehat{\omega} + \frac{\widehat{\omega}^2}{2} \right). \tag{2.32}$$

The linear maps $\mathrm{d}\tau_\xi$, $\mathrm{d}\tau_\xi^{-1}$, and $(D\,\mathrm{d}\tau_\xi^{-1} \cdot \delta)$ can be expressed as 3×3 matrices. In the Cayley map case these are

$$\operatorname{dcay}_\omega = \frac{2}{4 + \|\omega\|^2} \left(2\mathbf{I}_3 + \widehat{\omega} \right), \qquad \operatorname{dcay}_\omega^{-1} = \mathbf{I}_3 - \frac{\widehat{\omega}}{2} + \frac{\omega\omega^T}{4} \tag{2.33}$$

$$D \operatorname{dcay}_\omega^{-1} \cdot \delta = -\frac{\widehat{\delta}}{2} + \frac{\delta\omega^T}{4} + \frac{\omega\delta^T}{4} \tag{2.34}$$

2.6.2 $SE(2)$

The coordinates of $SE(2)$ are (θ, x, y) with matrix representation $g \in SE(2)$ given by:

$$g = \begin{bmatrix} \cos\theta & -\sin\theta & x \\ \sin\theta & \cos\theta & y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.35}$$

Using the isomorphic map $\widehat{\cdot} \colon \mathbb{R}^3 \to \mathfrak{se}(2)$ given by:

$$\widehat{v} = \begin{bmatrix} 0 & -v^1 & v^2 \\ v^1 & 0 & v^3 \\ 0 & 0 & 0 \end{bmatrix} \text{ for } v = \begin{pmatrix} v^1 \\ v^2 \\ v^3 \end{pmatrix} \in \mathbb{R}^3,$$

$\{\widehat{e}_1, \widehat{e}_2, \widehat{e}_3\}$ can be used as a basis for $\mathfrak{se}(2)$, where $\{e_1, e_2, e_3\}$ is the standard basis of $\mathbb{R}^3$.

The two maps $\tau : \mathfrak{se}(2) \to SE(2)$ are given by

$$\exp(\widehat{v}) = \begin{cases} \begin{bmatrix} \cos v^1 & -\sin v^1 & \frac{v^2 \sin v^1 - v^3(1-\cos v^1)}{v^1} \\ \sin v^1 & \cos v^1 & \frac{v^2(1-\cos v^1)+v^3 \sin v^1}{v^1} \\ 0 & 0 & 1 \end{bmatrix} & \text{if } v^1 \neq 0 \\ \begin{bmatrix} 1 & 0 & v^2 \\ 0 & 1 & v^3 \\ 0 & 0 & 1 \end{bmatrix} & \text{if } v^1 = 0 \end{cases}$$

$$\mathrm{cay}(\widehat{v}) = \begin{bmatrix} \frac{1}{4+(v^1)^2} \begin{bmatrix} (v^1)^2 - 4 & -4v^1 & -2v^1v^3 + 4v^2 \\ 4v^1 & (v^1)^2 - 4 & 2v^1v^2 + 4v^3 \end{bmatrix} \\ 0 \quad 0 \quad 1 \end{bmatrix}$$

The maps $[\mathrm{d}\tau_\xi^{-1}]$ can be expressed as the 3×3 matrices:

$$[\mathrm{dexp}_{\widehat{v}}^{-1}] \approx \mathbf{I}_3 - \frac{1}{2}[\mathrm{ad}_v] + \frac{1}{12}[\mathrm{ad}_v]^2, \tag{2.36}$$

$$[\mathrm{dcay}_{\widehat{v}}^{-1}] = \mathbf{I}_3 - \frac{1}{2}[\mathrm{ad}_v] + \frac{1}{4}\begin{bmatrix} v^1 \cdot v & \mathbf{0}_{3\times 2} \end{bmatrix}, \tag{2.37}$$

where

$$[\mathrm{ad}_v] = \begin{bmatrix} 0 & 0 & 0 \\ v^3 & 0 & -v^1 \\ -v^2 & v^1 & 0 \end{bmatrix}.$$

2.6.3 $SE(3)$

We make the identification $SE(3) \approx SO(3) \times \mathbb{R}^3$ using elements $R \in SO(3)$ and $x \in \mathbb{R}^3$ through

$$g = \begin{bmatrix} R & x \\ \mathbf{0} & 1 \end{bmatrix}, \qquad g^{-1} = \begin{bmatrix} R^T & -R^T x \\ \mathbf{0} & 1 \end{bmatrix}$$

Elements of the Lie algebra $\xi \in \mathfrak{se}(3)$ are identified with *body-fixed* angular and linear velocities denoted $\omega \in \mathbb{R}^3$ and $v \in \mathbb{R}^3$, respectively, through

$$\xi = \begin{bmatrix} \widehat{\omega} & v \\ \mathbf{0} & 0 \end{bmatrix},$$

where the map $\widehat{\cdot} : \mathbb{R}^3 \to \mathfrak{so}(3)$ is defined in (2.31).

Using this identification we have

$$\tau(\xi) = \begin{bmatrix} \tau(h\widehat{\omega}_k) & h\, \mathrm{d}\tau_{h\omega_k}\, v_k \\ 0 & 1 \end{bmatrix},$$

where, choosing $\tau = \mathrm{cay}$, $\tau : \mathfrak{so}(3) \to SO(3)$ is given by (2.32) and $\mathrm{d}\tau_\omega : \mathbb{R}^3 \to \mathbb{R}^3$ by (2.33).

The matrix representation of the right-trivialized tangent inverses $\mathrm{d}\tau^{-1}_{(\omega,v)} : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^3 \times \mathbb{R}^3$ are computed as

$$[\mathrm{dcay}^{-1}_{(\omega,v)}] = \begin{bmatrix} \mathbf{I}_3 - \frac{1}{2}\widehat{\omega} + \frac{1}{4}\omega\omega^T & \mathbf{0}_3 \\ -\frac{1}{2}\left(\mathbf{I}_3 - \frac{1}{2}\widehat{\omega}\right)\widehat{v} & \mathbf{I}_3 - \frac{1}{2}\widehat{\omega} \end{bmatrix}, \tag{2.38}$$

Note We should point out that the Cayley map yields a more computationally efficient integrator. In addition, it is suitable for iterative integration and optimization problems since the map and its derivatives do not have any singularities that might otherwise affect gradient based methods.

2.6.4 General matrix subgroups

The Lie algebra of a matrix Lie group coincides with the one-parameter subgroup generators of the group. Assume that we are given a k-dimensional lie subalgebra $\mathfrak{g} \subset \mathfrak{gl}(n, \mathbb{R})$. This subalgebra is isomorphic to the space of generators of a unique connected k-dimensional matrix subgroup $G \subset GL(n, \mathbb{R})$. Therefore, a subalgebra $\mathfrak{g}$ determines the subgroup G in a one-to-one fashion

$$\mathfrak{g} \subset \mathfrak{gl}(n, \mathbb{R}) \Longleftrightarrow G \subset GL(n, \mathbb{R}).$$

The two ingredients necessary to convert the necessary conditions in Prop. (2) into algebraic equalities are: a choice of basis for $\mathfrak{g}$; and an appropriate choice of inner product (metric) that satisfies the invariance assumptions.

Assume that the Lie algebra basis elements are $\{E_\alpha\}_{\alpha=1}^k$, $E_\alpha \in \mathfrak{g}$, i.e. that every element $\xi \in \mathfrak{g}$ can be written as $\xi = \xi^\alpha E_\alpha$. The following inner product for any $\xi, \eta \in \mathfrak{g}$ would be useful

$$\langle\langle \xi, \eta \rangle\rangle = \mathrm{tr}(K\xi^T \eta),$$

where $K \in GL(n,\mathbb{R})$ is such that $\langle\langle E_\alpha, E_\beta\rangle\rangle = \delta_\alpha^\beta$ and tr is the matrix trace. Correspondingly, a pairing between any $\mu \in \mathfrak{g}^*$ and $\xi \in \mathfrak{g}$ can be defined by

$$\langle \mu, \xi \rangle = \mathrm{tr}(K\mu\xi),$$

since the dual basis for $\mathfrak{g}^*$ is $\{[E_\alpha]^T\}_{\alpha=1}^k$ in matrix form.

Example If $\mathfrak{g} = \mathfrak{so}(3)$ with basis then setting $K = \mathrm{diag}(1/2, 1/2, 1/2)$ the pairing yields the standard inner product if we identify $\mathfrak{so}(3)$ with $\mathbb{R}^3$, i.e. $\langle \mu, \xi \rangle = \mu_\alpha \xi^\alpha$.

Example If $\mathfrak{g} = \mathfrak{se}(3)$ with basis then setting $K = \mathrm{diag}(1/2, 1/2, 1/2, 1)$ the pairing yields the standard inner product if we identify $\mathfrak{se}(3)$ with $\mathbb{R}^3 \times \mathbb{R}^3$.

Kinetic Energy-Type Metric After having defined a metric pairing, a kinetic energy tensor $\mathbb{I}$ metric (such as the one used in 2.17) can be be expressed as

$$\langle \mathbb{I}(\xi), \eta \rangle = \mathrm{tr}(KI_d\xi^T\eta),$$

where $I_d \in GL(n,\mathbb{R})$ is a symmetric matrix.

Example Consider a rigid body on $SO(3)$ with principal moments of inertia I_1, I_2, I_3 and Lagrangian $\ell(\xi) = \frac{1}{2} I_i \xi_i^2$ where the ξ_i are the velocity components in the Lie algebra basis ?. The matrix I_d must have the form

$$I_d = \mathrm{diag}(-I_1 + I_2 + I_3, -I_2 + I_1 + I_3, -I_3 + I_1 + I_2)$$

Example Consider a rigid body on $SE(3)$ with principal moments of inertia I_1, I_2, I_3, mass m, and Lagrangian $\ell(\omega, v) = \frac{1}{2}\left(I_i\omega_i^2 + mv_i^2\right)$, where $(\omega, v) \in \mathfrak{se}(3)$ are the body-fixed angular and linear velocities and their components (v_i, w_i) are with respect to the Lie algebra basis ?. The Lagrangian in this case can be equivalently expressed as $\ell(\xi) = \frac{1}{2}tr(KI_d\xi^T\xi)$, where $\xi \in \mathfrak{se}(3)$ and

$$I_d = \text{diag}(-I_1 + I_2 + I_3, -I_2 + I_1 + I_3, -I_3 + I_1 + I_2, m).$$

2.7 Underactuated Systems with Control Parameters

The direct and indirect methods for mechanical systems on Lie groups can be generalized to underactuated systems. Assume that the control forces are applied along body-fixed directions defined by the *control covectors* $\{f^1(\phi), ..., f^c(\phi)\}$, $c \leq n$, $f^i : \mathbb{M} \to \mathfrak{g}^*$ which depend on *control parameters* $\phi : [0, T] \to \mathbb{M}$, where $\mathbb{M} \subset \mathbb{R}^m$ is the control parameter set. One can think of these extra parameters as the shape variables of the control basis, i.e. parameters that do not affect the inertial properties of the systems but which determine the control directions. Assume that the system is controlled using *control input* $u : [0, T] \to \mathbb{U}$, where $\mathbb{U} \subset \mathbb{R}^c$, is the set of controls applied with respect to the basis $\{f^i(\phi)\}$. In addition, assume that the system is subject to position-dependent *external force* $f_{\text{ext}} : G \to \mathfrak{g}^*$. This general definition accounts for various types of forces. For instance, forces arising from a potential $V : G \to \mathbb{R}$ take the form $f_{\text{ext}}(g) = TL_g^*DV(g)$.

The discrete equations of motion of such system are

$$(\mathrm{d}\tau^{-1}_{h\xi_k})^* \mathbb{I}\,\xi_k - (\mathrm{d}\tau^{-1}_{-h\xi_{k-1}})^* \mathbb{I}\,\xi_{k-1} - h\sum_{i=0}^{c} u_k^i f^i(\phi_k) - h f_{\mathrm{ext}}(g_k) = 0 \tag{2.39}$$

Define the Lagrangian multipliers $\lambda \in \mathfrak{g}^*$, $\zeta_k \in \mathfrak{g}$ and the Hamiltonian function

$$\begin{aligned} &H(\xi_{0:N-1}, u_{0:N}, \phi_{0:N}, \zeta_{0:N}, \lambda) \\ &= \sum_{k=0}^{N} \left(\frac{h}{2} u_k^T u_k + \langle \mathrm{DEP}_k - h\sum_{i=0}^{c} u_k^i f^i(\phi_k) - h f_{\mathrm{ext}}(g_k), \zeta_k \rangle \right) + \langle \lambda, \mathrm{REC}(\xi_{0:N-1}) \rangle, \end{aligned}$$

where DEP is defined in Sec. 2.5.

The necessary conditions for an optimal trajectory are defined in the following proposition (which extends Prop. 2).

Proposition 3. *The trajectory of a discrete mechanical system on a Lie group G with algebra $\mathfrak{g}$ and Lagrangian $\ell(\xi_k) = \frac{1}{2}\langle \mathbb{I}\,\xi_k, \xi_k \rangle$, with fixed initial and final configurations and velocities $g(0) \in G$, $\xi(0) \in \mathfrak{g}$ and $g(T) \in G$, $\xi(T) \in \mathfrak{g}$, minimizes the total control effort $\sum_{k=0}^{N} \frac{h}{2} u_k^T u_k$ only if the discrete body-fixed velocity curve $\xi_{0:N-1}$ and control parameters $\phi_{0:N}$ satisfies the following*

Necessary Conditions for Optimality

$$(\mathrm{d}\tau_{h\xi_k}^{-1})^*\eta_k - (\mathrm{d}\tau_{-h\xi_{k-1}}^{-1})^*\eta_{k-1} = 0, \qquad k = 1, ..., N-1 \tag{2.40}$$

$$\tau^{-1}(\tau(h\xi_0)\cdots\tau(h\xi_{N-1})\cdot(g(0)^{-1}g(T))^{-1}) = 0, \tag{2.41}$$

$$\mathrm{DEP}_k \; -h\sum_{i=0}^{c}\langle f^i(\phi_k),\zeta_k\rangle f^i(\phi_k) - hf_{\mathrm{ext}}(g_k) = 0,, \qquad k = 0, ..., N \tag{2.42}$$

$$\sum_{i=0}^{c}\langle f^i(\phi_k),\zeta_k\rangle Df^i(\phi_k)^T\zeta_k = 0, \qquad k = 0, ..., N \tag{2.43}$$

where $\eta_k \in \mathfrak{g}^*$ *is defined by*

$$\begin{aligned}\langle \eta_k, \delta\rangle = &\left\langle \mathbb{I}(\mathrm{d}\tau_{h\xi_k}^{-1}(\nu_k)) - h(\mathrm{d}\tau_{h\xi_k})^*\,\mathrm{ad}^*_{\left(\mathrm{Ad}_{\tau(h\xi_k)}\,\zeta_{k+1}\right)}(\mathrm{d}\tau_{h\xi_k}^{-1})^*\,\mathbb{I}(\xi_k), \delta\right\rangle \\ &+\left\langle \mathbb{I}(\xi_k), h\left(\mathrm{D}\,\mathrm{d}\tau_{h\xi_k}^{-1}\cdot\delta\right)(\nu_k)\right\rangle \\ &+\left\langle h\,(\mathrm{d}\tau_{h\xi_k})^*\sum_{i=k+1}^{N}\mathrm{Ad}^*_{\tau(h\xi_k)\cdots\tau(h\xi_{i-1})}(D_gf_{\mathrm{ext}}(g_i)\cdot g)^T\zeta_i, \delta\right\rangle,\end{aligned} \tag{2.44}$$

$$\nu_k = \mathrm{Ad}_{\tau(h\xi_k)}\,\zeta_{k+1} - \zeta_k. \tag{2.45}$$

Note: The proposition defines $(Nn+(N+1)n+(N+1)m)$ *equations* (2.40)-(2.43) *in the* $(Nn+(N+1)n+(N+1)m)$ *unknowns* $(\xi_{0:N}, \zeta_{0:N}, \phi_{0:N})$*. A solution can be found using standard nonlinear root finding.*

2.8 The sub-Riemannian case

Next we consider the case in which the body-fixed velocity ξ is restricted to lie in a m-dimensional distribution $\mathfrak{h} \subset \mathfrak{g}$, where $m < n$. This problem is known as the the sub-Riemannian optimal control problem. The continuous version has been studied by many authors with the general case on Riemannian manifolds and compact Lie groups first developed in [48], and extended in [23],[37] to the more general *elastic sub-Riemannian*

interpolation problem and more recently used in an interesting imaging application [25]. In this section we derive necessary conditions for optimality for discrete solution trajectories of a discrete dynamic nonholonomic principle.

Assume that a basis for $\mathfrak{g}$ can be constructed using the constant elements $X_i \in \mathfrak{g}$, $i = 1, ..., n$ in such a way that the first m elements span $\mathfrak{h}$, i.e.

$$\mathfrak{h} = \text{span}\{X_1, ..., X_m\},$$

Assume that $\mathfrak{h}$ is not a proper subalgebra of $\mathfrak{g}$ but the iterated Lie brackets of $X_1, ..., X_m$ span $\mathfrak{g}$ so that the system is controllable. In order to make the derivations more clear, let $X_1, ..., X_n$ be orthonormal. The more general case can be treated with slight modification by the addition of normalizing terms and would not greatly affect the final results. Let the orthogonal complement to $\mathfrak{h}$ be defined by $\mathfrak{h}^\perp$, i.e. $\mathfrak{g} = \mathfrak{h} \oplus \mathfrak{h}^\perp$. The nonholonomic constraint $\xi \in \mathfrak{h}$ can be written as

$$\langle\langle \xi, X_c \rangle\rangle = 0, \quad c = m+1, ..., n, \tag{2.46}$$

since $\mathfrak{h}^\perp = \text{span}\{X_{m+1}, ..., X_n\}$. If we denote the velocity coordinates with respect to the basis $\{X_i\}$ by ξ^i, i.e. such that $\xi = \xi^i X_i$, then the constraint is simply

$$\xi^c = 0.$$

2.8.1 Discrete Equations of Motion

The discrete LDAP principle (2.4) can be easily extended to the sub-Riemannian case. The discrete path is formally denoted $(g, \xi, \mu, \rho)_{0:N} : \{t_k\}_{k=0}^{N} \to G \times \mathfrak{h} \times \mathfrak{g}^* \times \mathfrak{h}^{\perp *}$. The only difference with the holonomic case is the addition of the constraint (2.46) as well as restriction the allowable variations $g_k^{-1}\delta g_k$. The dynamic sub-Riemannian LDAP principle is formulated as:

$$\begin{aligned} &\delta \sum_{k=0}^{N-1} h \left[\frac{1}{2} \langle \mathbb{I}\, \xi_k, \xi_k \rangle + \left\langle \mu_k, \tau^{-1}(g_k^{-1} g_{k+1})/h - \xi_k \right\rangle \right] + \sum_{k=0}^{N} \langle T L^*_{g_k^{-1}} \widetilde{f}_k, \delta g_k \rangle = 0, \\ &\text{where } g_k^{-1}\delta g_k \in \mathfrak{h}, \quad \xi_k \in \mathfrak{h} \end{aligned} \tag{2.47}$$

where the discrete forces $\widetilde{f}_k \in \mathfrak{h}^*$ are also aligned with the constraints. After taking variations in (2.47) we obtain the following discrete equations of motion.

$$\langle (\mathrm{d}\tau^{-1}_{h\xi_k})^* \mu_k - (\mathrm{d}\tau^{-1}_{-h\xi_{k-1}})^* \mu_{k-1} - \widetilde{f}_k, X_i \rangle, i = 1, ..., m, \qquad k = 1, ..., N-1, \tag{2.48}$$

$$\langle\!\langle \xi_k, X_i \rangle\!\rangle = 0, \qquad i = m+1, ..., n, \quad k = 0, ..., N-1, \tag{2.49}$$

$$\langle \mu_k, X_i \rangle = \begin{cases} \langle \mathbb{I}\, \xi_k, X_i \rangle, & i = 1, ..., m \\ 0, & i = m+1, ..., n \end{cases} \qquad k = 0, ..., N-1, \tag{2.50}$$

$$g_k^{-1} g_{k+1} = \tau(h\xi_k), \qquad k = 0, ..., N-1, \tag{2.51}$$

Following eq. (2.50)-(2.49) the momentum and velocity can be written written in coordinates according to

$$\begin{aligned} \mu_k &:= \left(\mathbb{I}_1\, \xi_k^1, ..., \mathbb{I}_m\, \xi_k^m, 0, ..., 0\right), \\ \xi_k &:= \left(\xi_k^1, ..., \xi_k^m, 0, ..., 0\right). \end{aligned} \tag{2.52}$$

2.8.2 Necessary Conditions for Optimality

Our goal is again to compute a discrete trajectory $(g_{0:N}, \xi_{0:N-1})$ the minimizes the total control effort. The indirect method can be extended to the sub-Riemannian case by introducing additional multipliers $\rho_k \in \mathfrak{h}^{\perp *}$ that enforce the constraint $\xi_k \in \mathfrak{h}$ without restricting variations on ξ_k. Define the Lagrangian multipliers $\lambda \in \mathfrak{g}^*$, $\zeta_k \in \mathfrak{h}$ and the Hamiltonian function

$$\begin{aligned} &H(\xi_{0:N-1}, f_{0:N}, \zeta_{0:N}, \rho_{0:N-1}, \lambda) \\ &= \sum_{k=0}^{N} \left(\frac{1}{2} \|\widetilde{f}_k\|^2 + \langle \mathrm{DEP}_k - \widetilde{f}_k, \zeta_k \rangle \right) + \sum_{k=0}^{N-1} \langle \rho_k, \xi_k \rangle + \langle \lambda, \mathrm{REC}(\xi_{0:N-1}) \rangle, \end{aligned}$$

where the quantities DEP and REC were defined in Sec. 2.5. The necessary conditions for an optimal trajectory are defined in the following proposition (which extends Prop. 2).

Proposition 4. *The trajectory of a discrete mechanical system on a Lie group G with algebra $\mathfrak{g}$ and Lagrangian $\ell(\xi_k) = \frac{1}{2}\langle \mathbb{I}\, \xi_k, \xi_k \rangle$, with fixed initial and final configurations and velocities $g(0) \in G$, $\xi(0) \in \mathfrak{h}$ and $g(T) \in G$, $\xi(T) \in \mathfrak{h}$, minimizes the total control effort $\sum_{k=0}^{N} \frac{1}{2} \|\widetilde{f}_k\|^2$ only if the discrete body-fixed velocity curve $\xi_{0:N-1}$ satisfies the following*

Necessary Conditions for Optimality

$$(\mathrm{d}\tau_{h\xi_k}^{-1})^* \eta_k - (\mathrm{d}\tau_{-h\xi_{k-1}}^{-1})^* \eta_{k-1} = 0, \qquad k = 1, ..., N-1 \tag{2.53}$$

$$\tau^{-1}(\tau(h\xi_0) \cdots \tau(h\xi_{N-1}) \cdot (g(0)^{-1} g(T))^{-1}) = 0, \tag{2.54}$$

where $\eta_k \in \mathfrak{g}^*$ *is defined by*

$$\begin{aligned} \langle \eta_k, \delta \rangle = &\left\langle \mathbb{I}(\mathrm{d}\tau_{h\xi_k}^{-1}(\nu_k)) - h(\mathrm{d}\tau_{h\xi_k})^* \operatorname{ad}^*_{\left(\mathrm{Ad}_{\tau(h\xi_k)} \widetilde{f}_{k+1}^{\,\sharp}\right)} (\mathrm{d}\tau_{h\xi_k}^{-1})^* \mathbb{I}(\xi_k) + \rho_k, \delta \right\rangle \\ &+ \left\langle \mathbb{I}(\xi_k), h \left(\mathrm{D}\, \mathrm{d}\tau_{h\xi_k}^{-1} \cdot \delta \right) (\nu_k) \right\rangle, \end{aligned} \tag{2.55}$$

$$\nu_k = \mathrm{Ad}_{\tau(h\xi_k)} \widetilde{f}_{k+1}^{\,\sharp} - \widetilde{f}_k^{\,\sharp}, \tag{2.56}$$

$$\widetilde{f}_0 = (\mathrm{d}\tau_{h\xi_0}^{-1})^* \mathbb{I}(\xi_0) - \mathbb{I}(\xi(0)), \tag{2.57}$$

$$\widetilde{f}_k = (\mathrm{d}\tau_{h\xi_k}^{-1})^* \mathbb{I}(\xi_k) - (\mathrm{d}\tau_{-h\xi_{k-1}}^{-1})^* \mathbb{I}(\xi_{k-1}), \qquad k = 1, ..., N-1 \tag{2.58}$$

$$\widetilde{f}_N = \mathbb{I}(\xi(T)) - (\mathrm{d}\tau_{-h\xi_{N-1}}^{-1})^* \mathbb{I}(\xi_{N-1}), \tag{2.59}$$

Note: The proposition defines Nn *equations* (2.53)-(2.54) *in the* Nn *unknown coordinates* $[(\xi_0^1, ..., \xi_0^m, (\rho_0)_1, ..., (\rho_0)_c), ..., (\xi_{N-1}^1, ..., \xi_{N-1}^m, (\rho_{N-1})_1, ..., (\rho_{N-1})_c))$, $c = n - m$. *A solution can be found using standard nonlinear root finding.*

2.9 Efficiency Techniques

There are several issues that are critical to the success of the optimization. One is the question of trajectory initialization and region of convergence. Another is the issue of the trajectory resolution and the balance between accuracy and efficiency for optimization problems.

2.9.1 Trajectory Initialization

We propose two ways to initialize a trajectory. None of these methods start with a near-optimal trajectory but the initial guesses satisfy the system dynamics and the imposed boundary conditions. The first method is based on turning the controls on only in the beginning and end of the trajectory, and using constant velocity (geodesic) motion everywhere else. The second is based on a quadratic interpolation in the Lie algebra to produce trajectories whose reconstruction to the group satisfies the boundary conditions exactly.

2.9.1.1 Geodesic Interpolation

In the geodesic interpolation we simply find a velocity that is constant everywhere but the first and last segment and such that the boundary conditions are satisfied.

$$\xi_k = \frac{\tau^{-1}\left(\tau(-h\xi_i)(g_i^{-1}g_f)\tau(-h\xi_f)\right)}{h(N-2)}, \quad k = 1, ..., N-2$$

After computing the velocities ξ_i the dynamics equations can be satisfied by adjusting the forces which can be done by straightforward computation. Ideally, the map τ should be the exponential map since it "reconstructs" the trajectory exactly. In general, there is no closed form for the logarithm (since in this case $\tau^{-1} = \log$) of a matrix and approximations can be used (in the case of $SO(3)$ and $SE(3)$ there is a closed form and this is not an

issue). Alternatively, one can use the Cayley map but the initial trajectory would satisfy the boundary conditions only approximately. The map $\text{cay}^{-1} : G \to \mathfrak{g}$ is defined by

$$\text{cay}^{-1}(g) = -2(\text{I}+g)^{-1}(\text{I}-g) \tag{2.60}$$

The Cayley map is not a good choice if the initial and final configurations g_i and g_f are far apart. Instead, one should use a higher order approximation to the exponential map and its inverse.

2.9.1.2 Lie Algebra Quadratic Interpolation

Continuous Interpolation

For the moment assume that we are working in a continuous setting and need to find a trajectory that interpolates the boundary conditions at times $t = 0$ and $t = T$ respectively. Define the curve $r : [0, T] \to \mathfrak{g}$ by

$$r(t) = \xi_0 t + c_1 t^2/2 + c_2 t^3/3$$

for some constant $c_1, c_2 \in \mathfrak{g}$. Let the continuous trajectory have the form

$$g(t) = g_0 \tau\left(r(t)\right),$$

Then, since $\xi(t) = g(t)^{-1}\dot{g}(t) \in \mathfrak{g}$ and using the right tangent trivialization the velocity is

$$\xi(t) = \mathrm{d}\tau_{-r(t)}\,\dot{r}(t)$$

Therefore, the constants c_1 and c_2 determine a family of curves. Among these curves it now becomes interpolating curves.

Proposition 5. *A curve* $(g, \xi) : [0, T] \to G \times \mathfrak{g}$ *of the form* $(g(t), \xi(t)) = (g_0 \tau(r(t)), d\tau_{-r(t)} \dot{r}(t))$, *where* $r(t) = \xi_0 t + c_1 t^2/2 + c_2 t^3/3$, $t \in [0, T]$, *with* $c_1, c_2 \in \mathfrak{g}$ *defined by*

$$c_1 = \frac{6}{T^2} \left(\tau^{-1}(g_0^{-1} g_T) - \mathrm{d}\tau^{-1}_{-r(T)} \, \xi_T \right) - \frac{4}{T} \xi_0$$

$$c_2 = \frac{3}{T^3} \left(\tau^{-1}(g_0^{-1} g_T) - \xi_0 T \right) - \frac{3}{2T} c_1$$

satisfies the given boundary conditions $(g(0), \xi(0)) = (g_0, \xi_0)$ *and* $(g(T), \xi(T)) = (g_T, \xi_T)$.

Proof. The boundary configuration condition can be expressed as $g_0^{-1} g_T = \tau(r(T))$ which is equivalent to

$$\tau^{-1}(g_0^{-1} g_T) = \xi_0 T + c_1 T^2/2 + c_2 T^3/3.$$

The final velocity boundary condition is equivalent to

$$\xi_0 + c_1 T + c_2 T^2 = \mathrm{d}\tau^{-1}_{-r(T)} \, \xi_T$$

The above two conditions are combined to give the closed form solution for c_1 and c_2. □

Corollary 1. *If* $\xi_0 = \xi_T = 0$ *then*

$$c_1 = \frac{6}{T^2} \tau^{-1}(g_0^{-1} g_T), \quad c_2 = -\frac{1}{T} c_1$$

and

$$g(t) = g(0)\tau(c_1 t^2/2 + c_2 t^3/3) = g_0 \tau \left(\left(\frac{3t^2}{T^2} - \frac{2t^3}{T^3} \right) \tau^{-1}(g_0^{-1} g_T) \right)$$

Discrete Interpolation

The discrete interpolation follows the continuous one very closely with the exception that the initial and final segments with fixed velocities (ξ_i and ξ_f) have to be treated specially.

The trajectory is then constructed using

$$\xi_k = \tau^{-1} \left(g(kh)^{-1} g((k+1)h) \right), \quad k = 1, ..., N-2,$$

and the dynamics equations satisfied by adjusting the forces correspondingly.

2.9.2 Trajectory Refinement

The efficiency of the optimization depends on choosing an initial trajectory guess that is close to the optimal one. An obvious way to choose such an initial guess is by computing a lower resolution trajectory that approximates the desired solution. Such trajectory is easier to compute and at the same time can be used to construct a good initial guess for successively refined trajectories. This process is repeated until the desired resolution is reached.

For example, Fig. 2.1 shows the refinement of an initially coarse trajectory on SO(3). Each point along the trajectory is represented by its rotation axis frame. While we do not have detailed results on convergence and runtime efficiency comparisons, such type of incremental refinement experimentally results into tremendous computational savings.

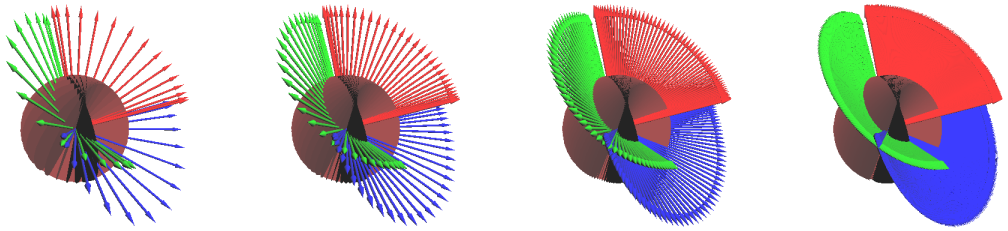


Figure 2.1: Successive refinement of an optimal trajectory on SO(3)

2.10 Applications

2.10.1 Rigid Body Reorientation on $SO(3)$

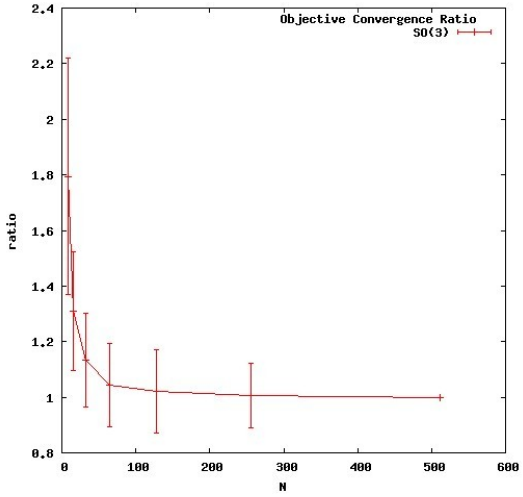


Figure 2.2: Objective function convergence ratio vs. trajectory resolution. The graph shows average results from 3000 Monte Carlo simulations.

We have implemented both the direct optimization method (defined by (2.16)) and the indirect one (defined by Prop. (2)). The direct method is implemented using SQP while the indirect using Newton's root finding method.

Here we present results only from the direct formulation applied to a simple problem of reorienting a fully-actuated asymmetric rigid body on SO(3) by applying torques around its principal axes of inertia. Each solution is obtained using the Lie algebra quadratic

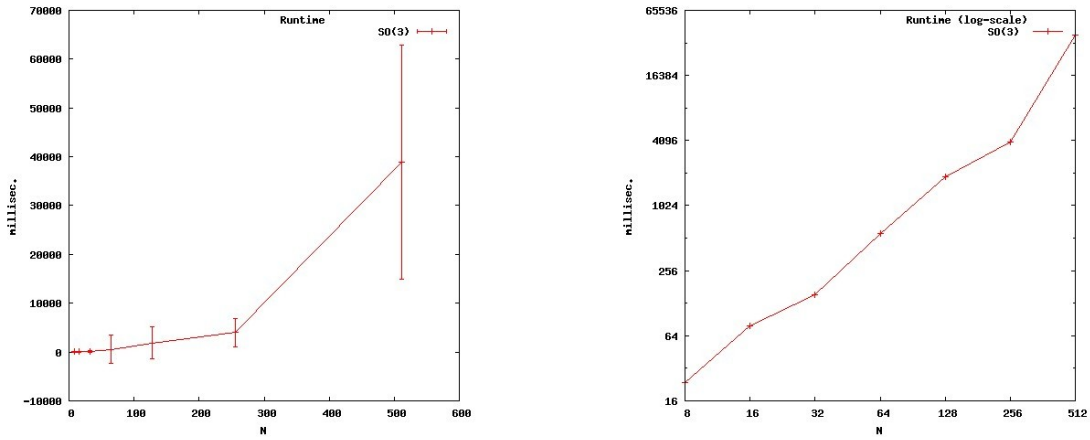


Figure 2.3: Runtime vs. trajectory resolution (normal scale on left and log scale on the right). The graph shows average results from 3000 Monte Carlo simulations.

interpolation initialization and successive refinement techniques described in Sec.2.9.1.2 and 2.9.2.

Fig. 2.2 shows how well a trajectory approximates (converges to) an optimal one as the time step (or resolution) is increased. The results are averaged from 3000 Monte Carlo runs with random boundary conditions. Overall, 3% of the trials fail to converge. Fig. 2.3 shows the average runtime in milliseconds as a function of the resolution (the tests were performed on an average desktop PC with 1.8 Pentium IV CPU.)

From both graphs one can conclude that a nearly optimal trajectory can be achieved on average with at least 100 time steps. It takes on average 1 second of computation time for a trajectory with such resolution.

2.10.2 Simple Helicopter in a Digital Terrain

Consider the following model of a helicopter depicted in Fig. 2.4. The vehicle is modeled as a single underactuated rigid body on $SE(3)$ with mass m and principal moments of rotational inertia J_1, J_2, J_3. The inertia tensors are $\mathbb{J} = \text{diag}(J_1, J_2, J_3)$ and $\mathbb{M} = m\mathbf{I}_3$.

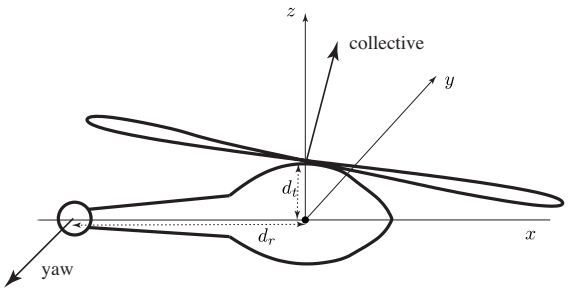


Figure 2.4: Simplified helicopter model used our tests.

The system configuration and velocity are described in standard notation defined in Sec. 2.6.3. The system is subject to external force due to gravity

$$f_{\text{ext}}((R,x),(\omega,v)) = (0,0,0,R^T(0,0,-9.81m)).$$

The vehicle is controlled through a *collective* u_c (lift produced by the main rotor) and a *yaw* u_ψ (force produced by the rear rotor), while the direction of the lift is controlled by tilting the main blades forward or backward through a *pitch* ϕ_p and a sideways *roll* ϕ_r. The shape variables are $\phi = (\phi_p, \phi_r)$, the controls are $u = (u_c, u_\psi)$. The resulting control covectors are

$$f^1(\phi) = (d_t \sin\phi_r, d_t \sin\phi_p \cos\phi_r, 0, \sin\phi_p, \cos\phi_p \cos\phi_r),$$
$$f^2(\phi) = (0,0,d_r,0,-1,0).$$

The discrete equations of motion are then obtained by substituting these specific expressions into the general integrator derived in Sec. 2.7 using the SE(3) Lie group operators defined in Sec. 2.6.3. An optimal trajectory is obtained by solving the general necessary conditions for optimality defined in Prop. (3) using nonlinear root finding. Fig. 2.5

shows a control-effort minimizing trajectory for the simulated helicopter between two zero-velocity states in an artificial canyon. The resulting state and control curves are shown in Fig. 2.8.

Controllability In order to establish the controllability of the system one can use the good symmetric products [6] of the two vectors $\mathbb{I}^{-1} f^1$ and $\mathbb{I}^{-1} f^2$ where $\mathbb{I}$ is the SE(3) inertial tensor

$$[\mathbb{I}] = \begin{bmatrix} \mathbb{J} & 0 \\ 0 & \mathbb{M} \end{bmatrix},$$

and show that the system is locally configuration controllable at zero velocity. Since the actuator inputs have bounds, one has to design an algorithm which allows time to vary in order to accommodate these limits. In our implementation, the time-step h (and hence the final time T) is part of the optimization state vector and is allowed to vary within some prescribed bounds.

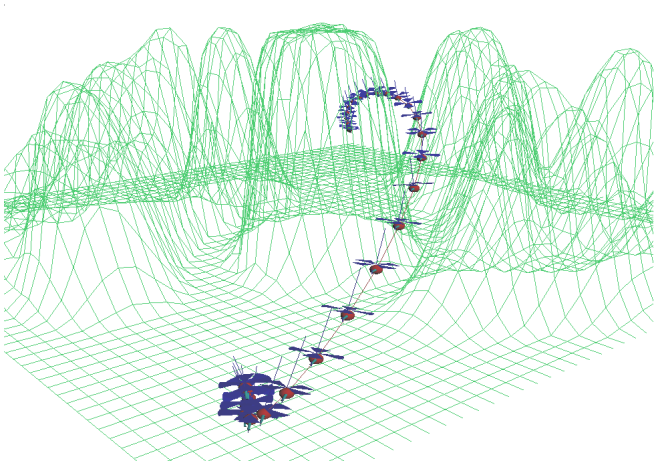


Figure 2.5: Example of an optimized trajectory in a complex environment: a helicopter path through an outdoor canyon.

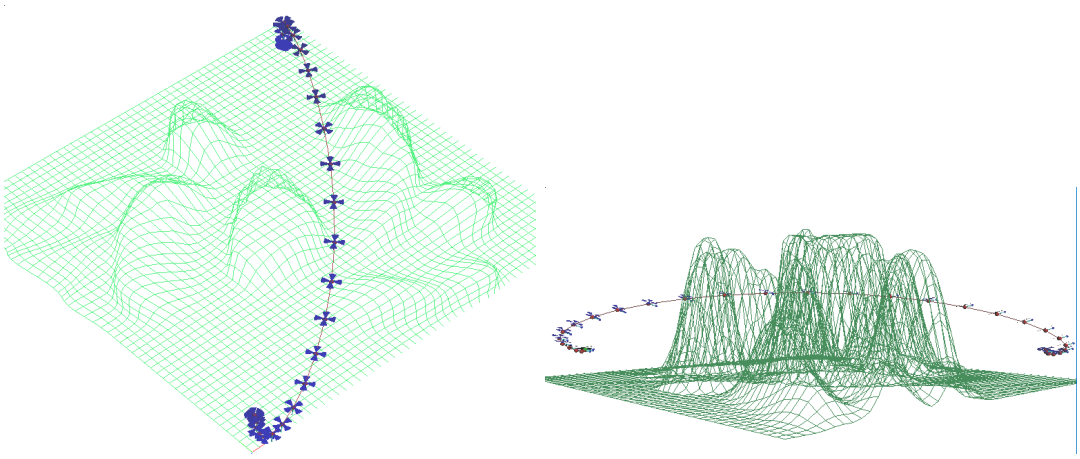


Figure 2.6: Top and side views of the helicopter trajectory shown in Fig. 2.5

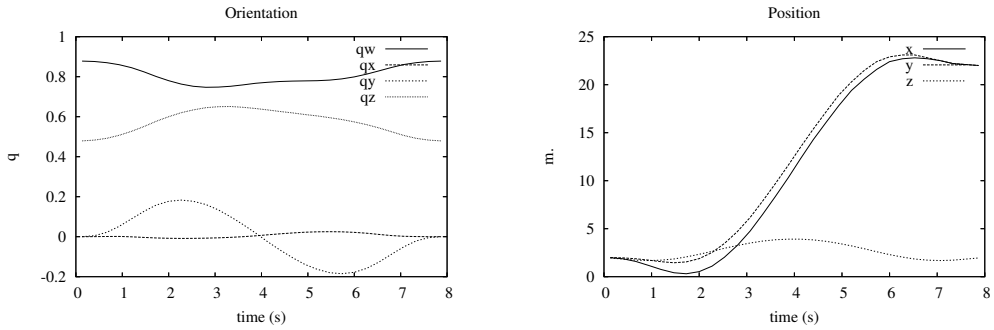


Figure 2.7: The orientation and position as functions of time for the helicopter trajectory shown in Fig. 2.5.

Obstacles The vehicle is required to stay away from obstacles by enforcing inequality constraints $H_i(R,p) = \text{dist}(\mathcal{A}(R,p), \mathcal{O}_i) - d_s > 0$, where $\mathcal{A} \subset \mathbb{R}^3$ is the region occupied by the robot, $\mathcal{O}_i \subset \mathbb{R}^3$ represent the static obstacles, and d_s is some safety distance. The function dist computes the minimum distance between two rigid objects. In our implementation both the canyon and the helicopter are triangulated surfaces and we use the Proximity Query Package (PQP) to compute dist.

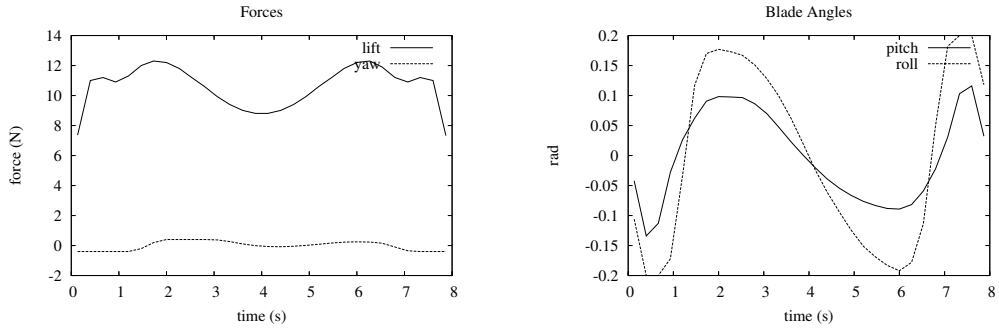


Figure 2.8: The control forces and blade angles as functions of time for the helicopter trajectory shown in Fig. 2.5.

The optimization runs efficiently in the absence of obstacles or with simple (smooth and convex) obstacles (taking in the order of a few seconds for most tests). On the other hand, complex rough terrains can slow down the system significantly. Note also that our simple implementation faces the same robustness issues as many optimization procedures: a bad choice for the initial trajectory may lead to a local minima. One way to speedup convergence is to start with a good initial obstacle-free path computed, for example, using a local obstacle avoidance method (e.g. [17]). Nevertheless, as already pointed out by several similar works cited in our references, this approach should be used to produce small-scale local solutions that are combined by more *global* methods in a hierarchical fashion (e.g. see [35] regarding incremental and roadmap planners, as well as [22, 16] for planning using primitives). We explore such ideas in Ch. 4. An obvious application is also the refinement of trajectories produced by discrete or sampling-based planners. Finally, in order to assess the exact numerical benefits of this method, a detailed performance analysis and comparison to related methods is needed and is currently under way.

2.10.3 Boat subject to External Disturbances

Consider a planar boat model shown in Fig. 2.9. The configuration space of the system is $SE(2)$ with coordinates $q = (\theta, x, y)$ denoting orientation and position with respect to a fixed global frame. Define the body fixed velocity $\xi \in \mathfrak{se}(2)$ by

$$\xi = [\omega, v, v^{\perp}]^T,$$

where ω is the angular velocity (*yaw*), v is the forward velocity (*surge*), $v^{\perp}$ is the sideways velocity (*sway*).

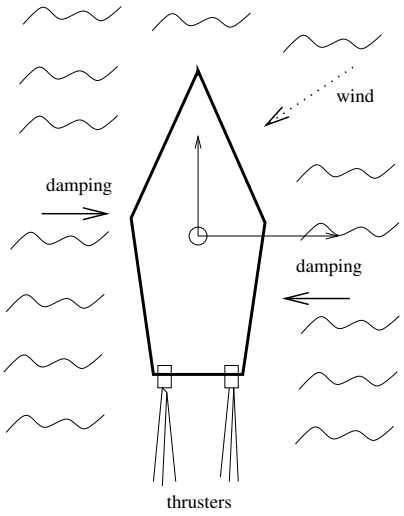


Figure 2.9: Planar boat controlled with two thrusters, and subject to hydrodynamic damping and wind forces.

The Lagrangian is $l(\xi) = \langle M\xi, \xi \rangle$, where M is the combined mass matrix of the boat's inertia and added mas of the surrounding fluid. The system is controlled with two fixed thrusters placed at the rear of the boat at distance $\pm c$ from the long axis of the boat

Figure 2.10: The USC RoboDuck2 boat used for experiments.

producing forces u_r and u_l, respectively. These two forces result in the propulsion force $f_{prop} \in \mathfrak{se}(2)^*$ with respect to the boat-fixed frame through the transformation

$$f_{prop} = B \begin{bmatrix} u_l \\ \\ u_r \end{bmatrix},$$

where B has the nominal form

$$B = \begin{bmatrix} -c & c \\ \\ 1 & 1 \\ \\ 0 & 0 \end{bmatrix}.$$

For a more accurate model, B can be determined experimentally as a function of velocity to account for unmodeled center of mass shift.

The boat is subject to hydrodynamic damping forces $f_{diss} : SE(2) \times \mathfrak{se}(2) \to \mathfrak{se}(2)^*$ in the form

$$f_{diss}(g, \xi) = -R(g, \xi)\xi,$$

where $R : SE(2) \times \mathfrak{se}(2) \to L(\mathfrak{se}(2); \mathfrak{se}(2)^*)$ is a positive definite symmetric linear map. Other external forces such as wind and currents are denoted by $f_{ext} : SE(2) \times \mathfrak{se}(2) \to \mathfrak{se}(2)^*$. For example, the USC boat has an onboard wind sensor providing wind direction and effective force. The combined force acting on the boat is

$$f = f_{prop} + f_{diss} + f_{ext}.$$

The continuous equations of motion are the Euler-Poincaré equations (2.3) on $SE(2)$ in the form

$$M \begin{bmatrix} \dot{\omega} \\ \dot{v} \\ \dot{v}^{\perp} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ v^{\perp} & 0 & -\omega \\ -v & \omega & 0 \end{bmatrix}^T M \begin{bmatrix} \omega \\ v \\ v^{\perp} \end{bmatrix} + B \begin{bmatrix} u_l \\ \\ u_r \end{bmatrix} - R \begin{bmatrix} \omega \\ v \\ v^{\perp} \end{bmatrix} + f_{ext}.$$

The discrete equations are (2.7)-(2.9) with the corresponding operations on $SE(2)$ defined in Sec. 2.6.2.

We illustrate the optimal control of the boat in two scenarios with different wind forces and final desired states. The boat parameters were chosen as

$$M = \text{diag}(3, 1, 5), \quad R = \text{diag}(1, .5, 10)$$

with R_{33} begin relatively high corresponding to the high lateral damping.

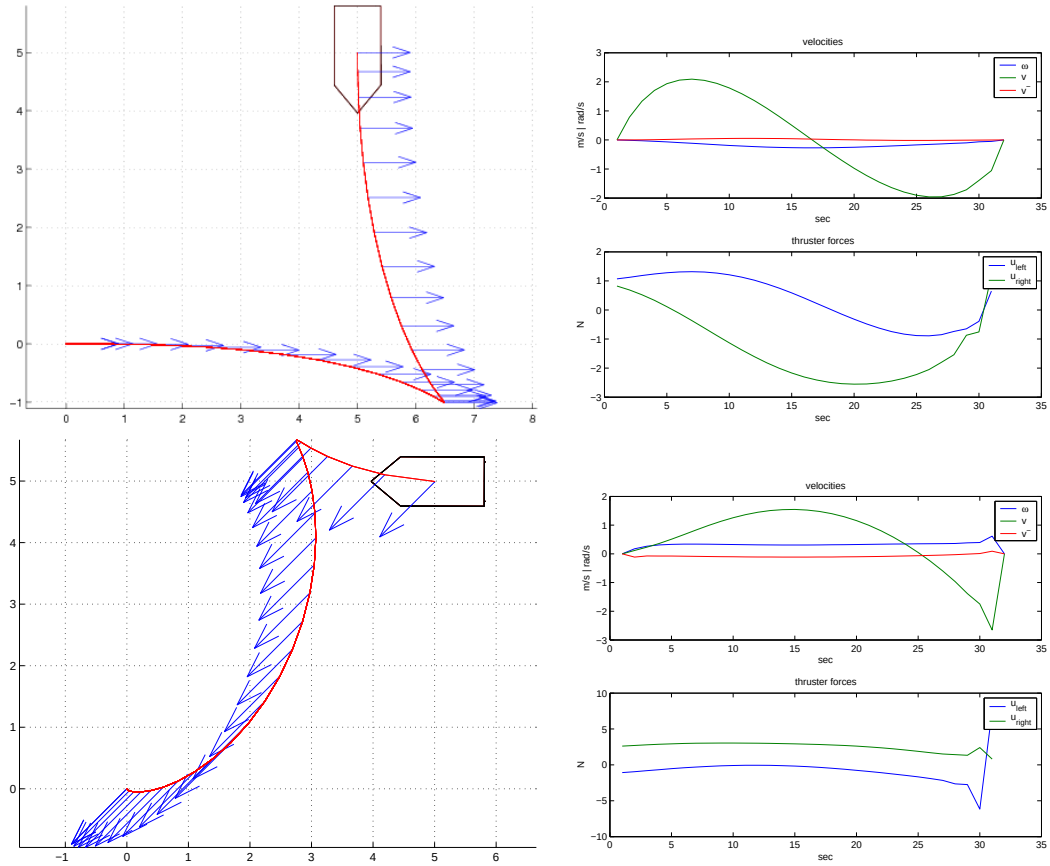


Figure 2.11: Two optimal control scenarios with different boundary conditions and wind forces (shown as arrows along the path). The boat always starts at the origin $(\theta, x, y) = (0, 0, 0)$ with zero velocity and must arrive at the designated positions with zero velocity.

Fig. 2.11 shows the path taken by the boat and the resulting velocity and control curves. The paths were discretized using $N = 32$ segments which results in real-time control performance.

Chapter 3

Nonholonomic Integrators

Summary

The chapter develops structure-preserving integrators for nonholonomic mechanical systems through a discrete geometric approach. At the core of the formulation lies a discrete Lagrange-d'Alembert-Pontryagin variational principle. The focus of this work is on systems with symmetries, controllable shape (internal variables), and nonholonomic constraints. Our discrete approach is especially well suited for such systems since it respects the structure of the state space and provides a framework for constructing more accurate and numerically stable integrators. The dynamics of the systems we study is derived by vertical and horizontal splitting of the variational principle with respect to a nonholonomic connection encoding the kinematic constraints and allowed symmetry directions. We formulate a discrete analog of this principle and derive discrete Euler-Poincaré and Euler-Lagrange equations resulting from the splitting of the principle with respect to a discrete approximation of the connection. A family of variational Euler-type integrators are then derived and applied to two examples–the steered robotic car and the snakeboard.

3.1 Introduction

The goal of this chapter is to develop integrators for mechanical systems subject to non-integrable constraints on the velocities, i.e., *nonholonomic constraints*. We study systems that evolve on a configuration manifold $Q = M \times G$ constructed from a Lie group G whose action leaves the kinetic energy invariant (and so G is a group of *symmetries*) and a vector space M that describes the system *internal shape*. This general configuration space applies to systems from several domains, e.g., locomotion systems found in nature [31, 39, 18], vehicles used in robotics and aerospace [46, 47, 3, 6], systems in molecular dynamics [26, 52].

Their dynamics is derived by explicitly factoring out the group invariance through reduction by symmetry, and consequently splitting the equations of motion into vertical–corresponding to symmetries aligned with the constraints and defining the evolution of a momentum, and horizontal–defining the dynamics of the shape space. This has proven not only computationally beneficial, from reducing the dimension and avoiding numerical ill-conditioning, but also crucial in studying the stability, controllability, and motion generation of such systems. In this thesis we focus on their proper discretization and propose geometric integrators that respect the state-space structure of the symmetries and constraints, preserve any invariants exhibited by the continuous system, and result in stable and accurate numerical schemes.

We follow the approach of *discrete mechanics* [41] and derive discrete equations of motion of the system through the discretization of the underlying variational principles governing the dynamics. In particular, we employ a Lagrange-d'Alembert-Pontryagin

(LDAP) variational principle [53] that differs from a standard variational principle, such as Lagrange-d'Alembert's, by the presence of a new additional velocity variable $v \in T_qQ$ at each point $q \in Q$ which by definition does not correspond to the rate of change of the configuration but this dependence is indirectly enforced using a kinematic constraint of the form $\dot{q} - v = 0$ and a multiplier $p \in T_q^*Q$ corresponding to the momentum. We formulate a discrete version of this principle which, in addition to retaining all the desirable preservation properties inherent to the discrete variational approach, also simplifies the construction of integrators. First, the Pontryagin viewpoint obviates the need to use a discrete Lagrangian $L_d : Q \times Q \to \mathbb{R}$ (which is the standard way to approximate the action integral in discrete mechanics, e.g. as formulated by Marsden and West[41]) as well as a discrete nonholonomic distribution $\mathcal{D}_d \subset Q \times Q$ (introduced by Cortes [13]) since now a state along the discrete trajectory is not represented by a pair of points $(q_k, q_{k+1}) \in Q \times Q$ but by $(q_k, v_k) \in TQ$ and both the original continuous Lagrangian and the constraints can be evaluated directly at such points to achieve the desired approximation. Second, the additional variables v and p give freedom in designing higher-order integrators by combining standard quadrature rules with higher order approximation of the kinematic constraint (as shown by Bou-Rabee and Marsden [5]). Further details about the principle are given in Sec. 2.3, and Sec. 3.2.3.

The results presented here build upon previous work on the variational discretization of systems with symmetries, as well as systems with nonholonomic constraints. Bobenko and Suris [4] and Marsden et al. [42] first studied the discrete Euler-Poincaré equations

for systems on Lie groups; Bou-Rabee and Marsden [5] extended those ideas in the framework of the Hamilton-Pontryagin principle in order to design more general and higher-order integrators; Jalnapurkar et al. [27] considered the discretization of the more general principle bundle case with abelian symmetry and its reduction using Routh's method. Nonholonomic constraints from a discrete variational viewpoint were first studied by Cortes [13] who also considered the invariance of such systems with respect to Lie group actions and derived a momentum equation with properties consistent with the continuous case. M. de Leon et al. [14, 38] considered an alternative discretization of nonholonomic systems in terms of generating functions. Fedorov and Zenkov [19] extended the reduced discrete approach to systems on a Lie group to include nonholonomic constraints on the group and derived the so called Euler-Poincaré-Suslov equations. McLachlan and Perlmutter [44] studied the general case of systems on vector spaces as well as on a group with nonholonomic constraints focusing on the time-reversibility and the importance of the preservation of invariants.

Contributions. Our work extends the recently-introduced geometric Lie group integrators [5] to provide a systematic approach to the design of structure-respecting integrators for nonholonomic mechanical systems. While related to the work of several authors noted above, our discrete approach to nonholonomic systems with symmetries captures the geometry of general systems (defined in terms of principle bundle and nonholonomic connections) with arbitrary group structure, constraints, and shape dynamics, and is not restricted to a configuration space that is either solely a group or has a Chaplygin-type symmetry. As a result our formulation contains a discrete momentum equation and a set

of discrete reduced Euler-Lagrange equations analogous to the continuous case (e.g. as described in [2]) that explicitly account for and respect the interaction between symmetries and constraints in the vehicle dynamics.

Organization. We start by recalling the geometry of nonholonomic systems with symmetries. Then we introduce the nonholonomic LDAP principle for systems with symmetries and derive the standard nonholonomic equations [2] that result from the splitting of the variational principle into vertical and horizontal parts with respect to the nonholonomic connection. The discrete analog of this principle is then formulated and the corresponding equations of motion derived. The vertical equation is equivalent to the evolution of a discrete momentum map whose properties are examined. Similar to a previous study by Cortes [13] we also consider the case of linear connection revealing an interesting link to the continuous dynamics. We apply the resulting algorithms to two examples: the steered car with simple dynamics and the snakeboard, and analyze the integrators accuracy and efficiency. Finally, we use the discrete equations as constraints in an optimal control problem to generate optimal maneuvers useful for motion planning.

3.2 Nonholonomic Systems with Symmetry

This section considers nonholonomic systems with symmetries in the continuous setting. We start by recalling standard concepts used to define the state space geometry. Then we formulate the nonholonomic LDAP principle and obtain the continuous nonholonomic equations of motion.

Assume that $\pi : Q \to Q/G$ is a principle bundle on the manifold Q with group G. The system has Lagrangian $L : TQ \to \mathbb{R}$ and is subject to nonholonomic constraints defined by the regular distribution $\mathcal{D} \subset TQ$. A group orbit is a submanifold denoted by $\mathrm{Orb}(q) := \{gq \mid g \in G\}$. If $\mathfrak{g}$ is the Lie algebra of G, then $\mathrm{T}_q \mathrm{Orb}(q) = \{\xi_Q(q) \mid \xi \in \mathfrak{g}\}$, where ξ_Q is the infinitesimal generator corresponding to the Lie algebra element ξ defined by

$$\xi_Q(q) = \frac{d}{ds}\bigg|_{s=0} \exp(s\xi)q.$$

Define the subspaces $\mathcal{V}_q, \mathcal{S}_q, \mathcal{H}_q$ according to

$$\mathcal{V}_q = T_q \mathrm{Orb}(q), \qquad \mathcal{S}_q = \mathcal{D}_q \cap \mathcal{V}_q, \qquad \mathcal{D}_q = \mathcal{S}_q \oplus \mathcal{H}_q.$$

These definitions have the following physical meaning (see [3] for a more detailed description):

- $\mathcal{V}_q$ – space of tangent vectors parallel to symmetry directions, i.e. the vertical space;
- $\mathcal{S}_q$ – space of symmetry directions that satisfy the constraints;
- $\mathcal{H}_q$ – space of tangent vectors that satisfy the constraints but are not aligned with any directions of symmetry, i.e. the horizontal space.

We make the following additional assumptions that are standard in the literature (see [3, 10])

- Dimension Assumption: *For each $q \in Q$, we have $T_qQ = \mathcal{D}_q + \mathcal{V}_q$.*
- Invariance of L: *The Lagrangian L is G-invariant.*

- Invariance of $\mathcal{D}$: *The distribution $\mathcal{D}$ is G-invariant.*

Define the vector space $\mathfrak{s}_q \subset T_qQ/G$ to be the set of Lie algebra element whose infinitesimal generators lie in $\mathcal{S}_q$, i.e. the space of symmetry directions that satisfy the constraints, by

$$\mathfrak{s}_q = \{\xi(q) \mid \xi_Q(q) \in \mathcal{S}_q\}.$$

The bundle with fibers $\mathfrak{s}_q$ at all $q \in Q$ is denoted $\mathfrak{s}$.

Since our main interest is in a configuration space that is by construction of the form $Q = M \times G$ we will restrict any further derivations to the *trivial bundle* case. While the more general case (introduced so far) can be treated in an analogous manner with slight modification we stick to the trivial case for clarity without loosing the general applicability of our results. Using (global) trivial bundle coordinates $(r, g) \in M \times G$ we have $\xi_Q(r, g) = (0, \xi(r, g)g) \in \mathcal{S}_q$. If we denote the basis for $\mathfrak{s}_q$ by $\{e_b(r, g)\}$, for $b = 1, ..., \dim(\mathcal{S})$ then since $\mathcal{D}$ is G-invariant g can be factored out from this basis, i.e. $e_b(r, g) = \mathrm{Ad}_g\, e_b(r)$, where $\{e_b(r)\}$ is the body-fixed basis. We denote $\mathfrak{s}_r$ the space spanned by $\{e_b(r)\}$.

Lastly, the system is subject to control force $f : [0, T] \to T^*M$ restricted to the shape space.

3.2.1 Nonholonomic Connection

With these definitions we can define a principle connection $\mathcal{A} : TQ \to \mathfrak{g}$ with horizontal distribution that coincides with $\mathcal{H}_q$ at point q. This connection is called the *nonholonomic connection* and satisfies

$$\mathcal{A}(r, g) \cdot (\dot{r}, \dot{g}) = \mathrm{Ad}_g(g^{-1}\dot{g} + \mathcal{A}(r)\dot{r}), \tag{3.1}$$

where $\mathcal{A}(r)$ is its local form.

The nonholonomic connection is constructed according to $\mathcal{A} = \mathcal{A}^{\mathrm{kin}} + \mathcal{A}^{\mathrm{sym}}$, where $\mathcal{A}^{\mathrm{kin}}$ is the kinematic connection enforcing the nonholonomic constraints and $\mathcal{A}^{\mathrm{sym}}$ is the mechanical connection corresponding to symmetries in the constrained directions (i.e. the group orbit directions satisfying the constraints). These maps satisfy

$$\begin{aligned} g^{-1}\dot{g} + \mathcal{A}^{\mathrm{kin}}(r)\dot{r} &= 0, \\ \mathcal{A}^{\mathrm{sym}}(r)\dot{r} &= \Omega, \\ g^{-1}\dot{g} + \mathcal{A}(r)\dot{r} &= \Omega. \end{aligned} \tag{3.2}$$

where $\Omega \in \mathfrak{s}_r$ is called the *locked angular velocity*, i.e. the velocity resulting from instantaneously locking the joints described by the variables r. Intuitively, when the joints stop moving the system continues its motion uniformly around a curve (with tangent vectors in $\mathcal{S}$) with body-fixed velocity Ω and a corresponding spatial momentum that is conserved.

The vector $\mathrm{ver}_r\,\dot{q} = (0,\Omega) \in (TM\times\mathfrak{s})_r$ is the *vertical* component relative the combined connection $\mathcal{A}$ and $\mathrm{hor}_r\,\dot{q} = (\dot{r}, -\mathcal{A}(r)\dot{r}) \in (TM\times\mathfrak{g})_r$ is the *horizontal* component. Velocity vectors on $TQ/G \sim TM \times \mathfrak{g}$ are split according to

$$(\dot{r}, g^{-1}\dot{g})_r = \mathrm{ver}_r\,\dot{q} + \mathrm{hor}_r\,\dot{q} = (0,\Omega) + (\dot{r}, -\mathcal{A}(r)\dot{r}).$$

3.2.2 Vertical and Horizontal Variations

We now consider variations of the configuration variables in the vertical and horizontal directions. Following [9, 3] define the following

Definition 1. *Vertical variations* Consider variations $(\delta r, \delta g)$ such that $\delta r = 0$ and $\delta g g^{-1} = \mathcal{A}(r,g)\cdot(\delta r, \delta g) \in \mathfrak{s}_{(r,g)}$. Then $(0, \delta g g^{-1}) \in T_rM \times \mathfrak{s}_{(r,g)}$ is clearly vertical and so is $(0, g^{-1}\delta g) \in T_rM \times \mathfrak{s}_r$. This can be easily checked considering that for trivial bundles $\mathfrak{s}$ can be constructed using a basis $\{e_b : Q \to \mathfrak{s}\}$ such that $e_b(r,g) = \mathrm{Ad}_g\, e_b(r)$, where $e_b(r)$ is a *body-fixed* basis in a left-trivialization.

Definition 2. *Horizontal variations* Variations satisfying $\mathcal{A}(r,g)\cdot(\delta r, \delta g) = 0$, or equivalently $g^{-1}\delta g + \mathcal{A}(r)\delta r = 0$, result in variations $(\delta r, g^{-1}\delta g) = (\delta r, -A(r)\delta r) \in (TM\times\mathfrak{g})_r$ that are horizontal.

Since vertical and horizontal variations can be taken independently we can consider variational principles based on vertical and horizontal variations separately. The next section presents these principles and the derivation of the resulting nonholonomic equations of motion.

3.2.3 Lagrange-D'Alembert-Pontryagin Nonholonomic Principle

Define the *reduced Lagrangian* $\ell : TM \times \mathfrak{g} \to \mathbb{R}$ according to

$$\ell(r, \dot{r}, \xi) = L(r, \dot{r}, e, g^{-1}\dot{g}),$$

and the *constrained reduced Lagrangian* $l^c : TM \times \mathfrak{s} \to \mathbb{R}$ such that

$$l^c(r, \dot{r}, \Omega) = \ell(r, \dot{r}, \Omega - \mathcal{A}(r)\dot{r}).$$

While both reduced Lagrangians capture the group invariance of the system, using the constrained reduced Lagrangian l^c has several advantages. One is that, unlike ξ, the locked angular velocity Ω diagonalizes the kinetic energy which has important implications in studying the stability of the system [40]. Another is that in the resulting equations of motion the rate of change of the generalized momentum decouples from that of the shape variables which is key in exploiting the holonomy of the system for locomotion and motion planning purposes.

Next, we begin from the general formulation of the LDAP principle (ref) and extend it to the principle bundle setting introduced earlier. Then we formulate two equivalent reduced principles, first in terms of the reduced Lagrangian ℓ and then in terms of the constrained reduced Lagrangian l^c.

Using bundle coordinates $(r, g) \in M \times G$ and denoting tangent vectors $(u, w) \in TM \times TG$ and corresponding momenta $(p, \hat{p}) \in T^*M \times T^*G$ we can rewrite the nonholonomic principle (2.4) as:

Definition 3. *Nonholonomic LDAP Principle (unreduced)*

$$\delta \int_0^T \{L(r,g,u,w) + \langle p, \dot{r} - u\rangle + \langle \hat{p}, \dot{g} - w\rangle\}\ dt + \int_0^T \langle f, \delta r\rangle dt = 0$$

subject to:

$$\begin{aligned} &\textit{nonholonomic constraint}\ \ \mathcal{A}(r,g)\cdot(u,w) = \mathrm{Ad}_g\,\Omega, \\ &\textit{vertical variations s.t.}\ \ (\delta r, g^{-1}\delta g) = (0,\eta),\ \ \textit{where } \eta \in \mathfrak{s}_r\ \ \textit{and,} \\ &\textit{horizontal variations s.t.}\ \ (\delta r, g^{-1}\delta g) = (\delta r, -\mathcal{A}(r)\delta r). \end{aligned} \tag{3.3}$$

After substituting the reduced Lagrangian ℓ and the constraint (eq. 3.1), and defining the local momentum $\mu = TL_g^*\hat{p} \in \mathfrak{g}^*$ the principle can be equivalently expressed as

Definition 4. *Reduced Nonholonomic LDAP Principle (reduced Lagrangian ℓ). Using the notation $\xi := \Omega - \mathcal{A}(r)\cdot u$, the principle requires that*

$$\delta \int_0^T \{\ell(r,u,\xi) + \langle p, \dot{r} - u\rangle + \langle \mu, g^{-1}\dot{g} - \xi\rangle\}\ dt + \int_0^T \langle f, \delta r\rangle dt = 0$$

subject to:

$$\begin{aligned} &\textit{vertical variations s.t.}\ \ (\delta r, g^{-1}\delta g) = (0,\eta),\ \ \textit{where } \eta \in \mathfrak{s}_r\ \ \textit{and,} \\ &\textit{horizontal variations s.t.}\ \ (\delta r, g^{-1}\delta g) = (\delta r, -\mathcal{A}(r)\delta r). \end{aligned} \tag{3.4}$$

In the above formulation variations δu, $\delta\Omega$, δp, $\delta\mu$ are free. The version of the principle in terms of the constrained Lagrangian is then

Definition 5. *Reduced Nonholonomic LDAP Principle (constrained reduced Lagrangian l^c)*

$$\delta \int_0^T \{l^c(r,u,\Omega) + \langle p, \dot{r} - u\rangle + \langle \mu, g^{-1}\dot{g} + \mathcal{A}(r)u - \Omega\rangle\}\ dt + \int_0^T \langle f, \delta r\rangle dt = 0$$

subject to:

$$\text{(3.5)}$$

vertical variations s.t. $(\delta r, g^{-1}\delta g) = (0,\eta)$, *where* $\eta \in \mathfrak{s}_r$ *and,*

horizontal variations s.t. $(\delta r, g^{-1}\delta g) = (\delta r, -\mathcal{A}(r)\delta r)$.

The principle (3.5) now contains all information necessary to derive the equations of motion that explicitly account for the symmetries, nonholonomic constraints, and their interaction.

Lie algebra basis In practice, the constrained symmetry space $\mathcal{S}_{(r,g)}$ often must be generated from Lie algebra basis that depends on the shape $r \in M$. Therefore, we assume that the basis $\{e_a(r) \mid a = 1, ..., \dim(G)\}$ spans $\mathfrak{g}_r$, in such a way that $\{e_b(r) \mid b = 1, ..., \dim(\mathfrak{s})\}$ is an orthogonal basis for $\mathfrak{s}_r$ at each r. Then $\Omega \in \mathfrak{s}_r$ in this basis is $\Omega = \Omega^b e_b(r)$.

Derivation Taking arbitrary variations δr, δg, δu, $\delta\Omega$, δp, $\delta\mu$ in (3.5) and noting that

$$\delta(g^{-1}\dot{g}) = \dot{\eta} + \left[g^{-1}\dot{g}, \eta\right], \text{ where } \eta = g^{-1}\delta g,$$

we obtain respectively

$$\delta r \Rightarrow \qquad \left\langle \frac{\partial l^c}{\partial r}, \delta r \right\rangle + \left\langle p, \dot{\delta r} \right\rangle + \langle \mu, D\mathcal{A}(r)(\delta r, u) \rangle + \langle f, \delta r \rangle \tag{3.6}$$

$$\delta g \Rightarrow \qquad + \left\langle \mu, \dot{\eta} + \left[g^{-1}\dot{g}, \eta \right] \right\rangle \tag{3.7}$$

$$\delta u \Rightarrow \qquad + \langle \frac{\partial l^c}{\partial u}, \delta u \rangle + \langle -p, \delta u \rangle + \langle \mu, \mathcal{A}(r)\delta u \rangle \tag{3.8}$$

$$\delta \Omega \Rightarrow \qquad + \left\langle \frac{\partial l^c}{\partial \Omega}, \delta \Omega \right\rangle - \langle \mu, \delta \Omega \rangle \tag{3.9}$$

$$\delta p \Rightarrow \qquad + \langle \delta p, \dot{r} - u \rangle \tag{3.10}$$

$$\delta \mu \Rightarrow \qquad + \left\langle \delta \mu, g^{-1}\dot{g} + \mathcal{A}(r)u - \Omega \right\rangle = 0 \tag{3.11}$$

Since δu, $\delta \Omega$, δp, and $\delta \mu$ are free we immediately obtain from (3.8)-(3.11)

$$\left\langle \frac{\partial l^c}{\partial u} - p, \delta r \right\rangle + \langle \mu, \mathcal{A}(r)\delta r \rangle = 0 \tag{3.12}$$

$$\frac{\partial l^c}{\partial \Omega} - \mu = 0 \tag{3.13}$$

$$\dot{r} - u = 0 \tag{3.14}$$

$$g^{-1}\dot{g} + \mathcal{A}(r)u - \Omega = 0 \tag{3.15}$$

Next we consider vertical and horizontal variations of $(\delta r, \delta g)$ separately.

3.2.3.1 Vertical Equations

Vertical variations have the form $\delta r = 0$, $g^{-1}\delta g = \eta,$ where $\eta \in \mathfrak{s}_r$, or in the previously defined basis $\eta = \eta^b e_b(r)$. Therefore, after substituting the constraint (3.15), (3.7) gives

$$\left\langle \mu, \frac{d}{dt}(\eta^b e_b(r)) + \left[\Omega - \mathcal{A}(r)u, \eta^b e_b(r)\right] \right\rangle = 0. \tag{3.16}$$

Next, we define the *momentum components* $\mu_b = \langle \mu, e_b(r) \rangle$ and, after integrating by parts, and using the boundary conditions $\delta g(0) = \delta g(T) = 0$ we obtain their time-derivative according to

$$\dot{\mu}_b = \left\langle \mu, \frac{\partial e_b(r)}{\partial r}\dot{r} + \mathrm{ad}_{\Omega - \mathcal{A}(r)u}\, e_b(r) \right\rangle,$$

since η^b are arbitrary.

3.2.3.2 Horizontal Equations

Horizontal variations are constrained according to $g^{-1}\delta g = \eta$, where $\eta = -\mathcal{A}(r)\delta r$ for variations δr in the base. Differentiating (3.12) with respect to time we get

$$\langle \dot{p}, \delta r \rangle = \left\langle \frac{d}{dt}\frac{\partial l^c}{\partial u}, \delta r \right\rangle + \langle \dot{\mu}, \mathcal{A}(r)\delta r \rangle + \langle \mu, D\mathcal{A}(r)(\dot{r}, \delta r) \rangle$$

Therefore, after integrating by parts (3.6) and (3.7), using the boundary conditions $\delta r(0) = \delta r(T) = 0$, and substituting $\eta = -\mathcal{A}(r)\delta r$ we get

$$\left\langle \frac{\partial l^c}{\partial r}, \delta r \right\rangle - \left\langle \frac{d}{dt}\frac{\partial l^c}{\partial u}, \delta r \right\rangle - \langle \dot{\mu}, \mathcal{A}(r)\delta r\rangle - \langle \mu, D\mathcal{A}(r)(\dot{r}, \delta r)\rangle$$
$$+ \langle \mu, D\mathcal{A}(r)(\delta r, u)\rangle + \langle f, \delta r\rangle$$
$$+ \langle \dot{\mu}, \mathcal{A}(r)\delta r\rangle - \langle [\Omega - \mathcal{A}(r)u, \mathcal{A}(r)\delta r]\rangle = 0$$

Simplifying, the horizontal (reduced Euler-Lagrange) equations of motion become

$$\left\langle \frac{\partial l^c}{\partial r} - \frac{d}{dt}\frac{\partial l^c}{\partial u} + f, \delta r \right\rangle = \langle \mu, \mathcal{B}(r)(u, \delta r) + \mathrm{ad}_\Omega\, \mathcal{A}(r)\delta r\rangle\,, \tag{3.17}$$

where

$$\mathcal{B}(r)(u, \delta r) = D\mathcal{A}(r)(u, \delta r) - D\mathcal{A}(r)(\delta r, u) - [\mathcal{A}(r)u, \mathcal{A}(r)\delta r]$$

is the curvature of the nonholonomic connection $\mathcal{A}$.

Summary

The equations of motion due to the splitting of the variational principle are

$$g^{-1}\dot{g} = \Omega - \mathcal{A}(r)u, \tag{3.18}$$

$$\dot{r} = u, \tag{3.19}$$

$$\mu = \frac{\partial l^c}{\partial \Omega}, \tag{3.20}$$

$$\dot{\mu}_b = \left\langle \mu, \frac{\partial e_b(r)}{\partial r}u + \mathrm{ad}_{\Omega - \mathcal{A}(r)u}\, e_b(r) \right\rangle, \tag{3.21}$$

$$\left\langle \frac{\partial l^c}{\partial r} - \frac{d}{dt}\frac{\partial l^c}{\partial u} + f, \delta r \right\rangle = \langle \mu, \mathcal{B}(r)(u, \delta r) + \mathrm{ad}_\Omega\, \mathcal{A}(r)\delta r\rangle \tag{3.22}$$

for $b = 1, ..., \dim(\mathfrak{s})$.

These equations are standard in the literature on nonholonomic systems with symmetries (e.g. [2, 34, 10, 3]) and we have managed to obtain them here directly from a reduced variational principle by restricting the variations on the configuration variables only. This is in contrast to the approach of separately studying the evolution of a momentum map (e.g. as taken in [2]) or by additionally restricting the allowable variations on the velocity variables ξ or Ω (explored in [10, 8]). Our main motivation for this alternative intrinsic formulation is that such a self-contained principle can be easily cast in a discrete framework and we expect that the resulting discrete equations of motion would most closely preserve the variational, geometric structure of the original system. We develop the discrete framework in the next section.

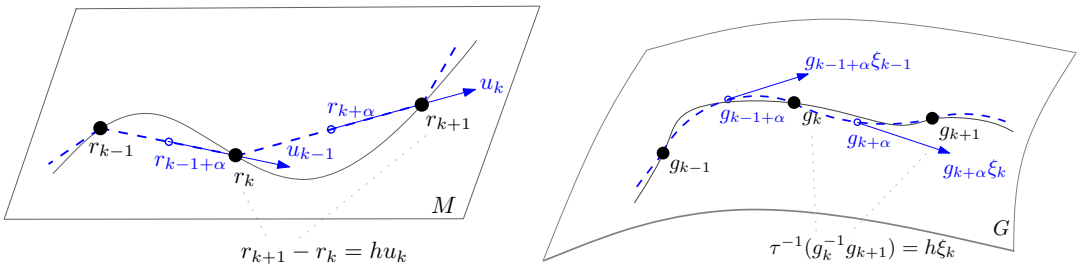


Figure 3.1: Discrete approximation (dashed) of continuous trajectories (solid) in the shape space (left) using linear interpolation, and in the group (right) using local geodesics defined by the flow of the map τ. The discrete velocity vectors shown approximate the *average* velocity along the segment and satisfy the constraint as defined underneath the figures. These velocity vectors are attached at quadrature points determined by the choice of $\alpha \in [0, 1]$.

3.3 Geometric Discretization of Nonholonomic Systems

In this section we formulate a discrete variational principle and derive a family of simple *nonholonomic integrators* that account for the group structure and constraint distribution, respect the work-energy balance, and have discrete momentum equation and a corresponding momentum map with properties analogous to the continuous case.

3.3.1 Discrete Approximation

As we noted earlier the discrete mechanics approach is based on varying discrete trajectories in order to find critical values of an action integral approximated through quadrature. The approximation scheme can be simple, i.e. by joining the discrete points along the path with simple local interpolation and few quadrature evaluations along the segments; or they can be higher-order by further discretizing each segment and performing multiple quadrature computations. Here, for clarity we focus on one simple type of discretization termed *variational Euler* [5] and provide the complete higher-order formulation in Appendix (ref).

Discrete Trajectory The discrete LDAP framework is defined in terms of a discrete trajectory whose states are elements of the tangent spaces in the reduced bundle of velocities and momenta. The trajectory is formally defined as follows (see also Fig. 3.1)

Definition 6. *The discrete reduced path is denoted*

$$(r, u, p, g, \Omega, \mu)_d : \{t_k\}_{k=0}^N \to (TM \oplus T^*M) \times G \times \mathfrak{s} \times \mathfrak{g}^*$$

and is subject to the constraints

$$r_{k+1} - r_k = hu_k, \qquad \tau^{-1}(g_k^{-1} g_{k+1}) = h\xi_k,$$

where $\xi_k = \Omega_k - \mathcal{A}(r_{k+\alpha})u_k$*, with* $r_{k+\alpha} := (1-\alpha)r_k + \alpha r_{k+1}$ *for a chosen* $\alpha \in [0,1]$ *and the map* $\tau : \mathfrak{g} \to G$ *represents the* difference *between two configurations in the group by an element in its algebra (see Sec. 2.3)*

The discrete control force is $f_{0:N} : \{t_k\}_{k=0}^N \to T^*M$ approximating a force controlling the shape.

Based on this simple approximation, the continuous and discrete state variables are related through (Fig. 3.1):

$$\begin{aligned} (r(t_{k+\alpha}), \dot{r}(t_{k+\alpha})) &\approx (r_{k+\alpha}, (r_{k+1} - r_k)/h) \\ (g(t_{k+\alpha}), \dot{g}(t_{k+\alpha})) &\approx \left(g_{k+\alpha}, g_{k+\alpha}\tau^{-1}(g_k^{-1} g_{k+1})/h\right), \end{aligned} \tag{3.23}$$

where $t_{k+\alpha} := t_0 + h(k+\alpha)$, $g_{k+\alpha} = g_k \tau(\alpha \tau^{-1}(g_k^{-1} g_{k+1}))$ for each $k = 0, ..., N-1$ and $\alpha \in [0,1]$.

Constraint Consistency It is important to note that the discretization constraints between configurations and velocities from Dfn. 6 are invariant to left translations of the discrete trajectory. Left-translating a pair of configurations (g_k, g_{k+1}) used to define velocity ξ_k is equivalent to applying the lifted left action to ξ_k itself, i.e.

$$(g_k, g_{k+1}) \Longrightarrow g_{k+\alpha}\tau^{-1}(g_k^{-1}g_{k-1}) = g_{k+\alpha}\xi_k,$$
$$(g'g_k, g'g_{k+1}) \Longrightarrow g'g_{k+\alpha}\tau^{-1}((g'g_k)^{-1}(g'g_{k-1})) = g'g_{k+\alpha}\xi_k.$$

Therefore, the approximation (3.23) remains valid in a left trivialization

$$\left(e, g^{-1}(t_{k+\alpha})\dot{g}(t_{k+\alpha})\right) \approx \left(e, \tau^{-1}(g_k^{-1}g_{k+1})/h\right),$$

and the left-invariant *discrete body-fixed velocity* ξ_k can be used for discrete reduction analogous to the continuous case.

Connection Equivariance Similarly to (3.23) the nonholonomic connection is approximated as

$$\begin{aligned}\mathcal{A}(r(t_{k+\alpha}), g(t_{k+\alpha})) \cdot (\dot{r}(t_{k+\alpha}), \dot{g}(t_{k+\alpha})) &\approx \mathcal{A}(r_{k+\alpha}, g_{k+\alpha}) \cdot (u_k, g_{k+\alpha}\xi_k) \\ &= \mathrm{Ad}_{g_{k+\alpha}}(\xi_k + \mathcal{A}(r_{k+\alpha})u_k) = \mathrm{Ad}_{g_{k+\alpha}}\Omega_k,\end{aligned}$$

for all $k = 0, ..., N-1$ and $\alpha \in [0,1]$. Note that the discretization constraint is also consistent with the required equivariance of the connection:

$$\mathrm{Ad}_{g'}\, \mathcal{A}(r_{k+\alpha}, g_{k+\alpha}) \cdot (u_k, g_{k+\alpha}\xi_k) = \mathrm{Ad}_{g'}\, \mathrm{Ad}_{g_{k+\alpha}}(\tau^{-1}(g_k^{-1}g'^{-1}g'g_{k+1}) + \mathcal{A}(r_{k+\alpha})u_k)$$

$$= \mathrm{Ad}_{g'g_{k+\alpha}}(\tau^{-1}((g'g_k)^{-1}g'g_{k+1}) + \mathcal{A}(r_{k+\alpha})u_k) = \mathcal{A}(r_{k+\alpha}, g'g_{k+\alpha}) \cdot (u_k, g'g_{k+\alpha}\xi_k).$$

3.3.2 Discrete Reduced LDAP Nonholonomic Principle

We formulate the discrete version of the LDAP principle (3.5) by approximating the action integral along each discrete segment using a single evaluation determined by the choice of $\alpha \in [0,1]$.

Definition 7. *Discrete Reduced LDAP Principle*

$$\delta \sum_{k=0}^{N-1} h \left[\{ l^c(r_{k+\alpha}, u_k, \Omega_k) + \langle p_k, (r_{k+1} - r_k)/h - u_k \rangle \right.$$
$$\left. + \langle \mu_k, \tau^{-1}(g_k^{-1}g_{k+1})/h + \mathcal{A}(r_{k+\alpha})u_k - \Omega_k \rangle \} \right] + \sum_{k=0}^{N-1} [h \langle f_{k+\alpha}, \delta r_{k+\alpha} \rangle] = 0 \tag{3.24}$$

subject to:

vertical variations s.t. $(\delta r_k, g_k^{-1}\delta g_k) = (0, \eta_k)$, *where* $\eta_k \in \mathfrak{s}_{r_k}$ *and,*

horizontal variations s.t. $(\delta r_k, g_k^{-1}\delta g_k) = (\delta r_k, -\mathcal{A}(r_k)\delta r_k)$,

In the above formulation variations δu_k, $\delta \Omega_k$, δp_k, $\delta \mu_k$ are free. Allowing the Lagrangian and the connection to be evaluated at $r_{k+\alpha}$ gives design freedom. Standard values of α are 0, 0.5, 1. Setting $\alpha = 0.5$ provides more accurate approximation of the base dynamics [1]

[1] Note that setting $\alpha = 0.5$ is not equivalent to unreduced *midpoint rule* since in left trivialization there is no notion of midpoint on the group

while $\alpha = 0, 1$ results in less accurate, but simpler integrators. As we mentioned, there are more general ways to define the discrete principle that allows arbitrary high approximation order but here we limit the exposition to lower order integrators for clarity.

The discrete force $f_{k+\alpha} = (1-\alpha)f_k + \alpha f_{k+1}$ is used to approximate the work done by f in a manner consistent with the rest of the discretization, i.e. through

$$\int_{kh}^{(k+1)h} \langle f, \delta r \rangle dt \approx h \langle f_{k+\alpha}, \delta r_{k+\alpha} \rangle.$$

Taking variations δr_k, δg_k, δu_k, $\delta \Omega_k$, δp_k, $\delta \mu_k$ in (3.24) and noting that

$$\delta(\tau^{-1}(g_k^{-1} g_{k+1})) = \mathrm{d}\tau^{-1}_{h\xi_k}(-\eta_k + \mathrm{Ad}_{g_k^{-1} g_{k+1}} \eta_{k+1}),$$

where $\eta_k = g_k^{-1}\delta g_k$, $\xi_k = \tau^{-1}(g_k^{-1} g_{k+1})/h$, and $\mathrm{d}\tau_\xi : \mathfrak{g} \to \mathfrak{g}$ is the *right-trivialized tangent* of $\tau(\xi)$ defined by $\mathrm{D}\,\tau(\xi) \cdot \delta = TR_{\tau(\xi)}(\mathrm{d}\tau_\xi \cdot \delta)$ and $d\tau_\xi^{-1} : \mathfrak{g} \to \mathfrak{g}$ is its inverse, we obtain respectively

$$\begin{aligned}
\delta r_k \Rightarrow \quad & h\left\langle \alpha\frac{\partial l^c_{k-1+\alpha}}{\partial r} + (1-\alpha)\frac{\partial l^c_{k+\alpha}}{\partial r}, \delta r_k \right\rangle + \langle -p_k + p_{k-1}, \delta r_k\rangle \\
& + h\,\langle \mu_{k-1}, \alpha D\mathcal{A}_{k-1+\alpha}(\delta r_k, u_{k-1})\rangle + h\,\langle \mu_k, (1-\alpha)D\mathcal{A}_{k+\alpha}(\delta r_k, u_k)\rangle \qquad (3.25)\\
& + h\,\langle (1-\alpha)f_{k-1+\alpha} + \alpha f_{k+\alpha}, \delta r_k\rangle \\
\delta g_k \Rightarrow \quad & + \left\langle -(\mathrm{d}\tau^{-1}_{h\xi_k})^*\mu_k + (\mathrm{d}\tau^{-1}_{-h\xi_{k-1}})^*\mu_{k-1}, \eta_k \right\rangle \qquad (3.26)\\
\delta u_k \Rightarrow \quad & + h\left\langle \frac{\partial l^c_{k+\alpha}}{\partial u}, \delta u_k \right\rangle + h\,\langle -p, \delta u_k\rangle + h\,\langle \mu_k, \mathcal{A}(r_{k+\alpha})\delta u_k\rangle \qquad (3.27)\\
\delta\Omega_k \Rightarrow \quad & + h\left\langle \frac{\partial l^c_{k+\alpha}}{\partial \Omega}, \delta\Omega_k \right\rangle - h\langle \mu_k, \delta\Omega_k\rangle \qquad (3.28)\\
\delta p_k \Rightarrow \quad & + h\,\langle \delta p_k, (r_{k+1}-r_k)/h - u_k\rangle \qquad (3.29)\\
\delta\mu_k \Rightarrow \quad & + h\left\langle \delta\mu_k, \tau^{-1}(g_k^{-1}g_{k+1})/h + \mathcal{A}(r_{k+\alpha})u_k - \Omega_k \right\rangle = 0, \qquad (3.30)
\end{aligned}$$

where $l^c_{k+\alpha} := l^c(r_{k+\alpha}, u_k, \Omega_k)$.

Since δu_k, $\delta\Omega_k$, δp_k, and $\delta\mu_k$ are free we immediately obtain from (3.27)-(3.30)

$$\begin{aligned}
& \frac{\partial l^c_{k+\alpha}}{\partial u} - p_k + \mathcal{A}(r_{k+\alpha})^*\mu_k = 0 \qquad (3.31)\\
& \frac{\partial l^c_{k+\alpha}}{\partial \Omega} - \mu_k = 0 \qquad (3.32)\\
& (r_{k+1}-r_k)/h - u_k = 0 \qquad (3.33)\\
& \tau^{-1}(g_k^{-1}g_{k+1})/h + \mathcal{A}(r_{k+\alpha})u_k - \Omega_k = 0 \qquad (3.34)
\end{aligned}$$

Next we consider vertical and horizontal variations of $(\delta r_k, \delta g_k)$ separately.

3.3.2.1 Vertical Equations

Vertical variations (Sec. 3.2.2) are of the form $\delta r_k = 0$, $g_k^{-1}\delta g_k = \eta_k$, where $\eta_k \in \mathfrak{s}_{r_k}$, or in the previously defined basis $\eta_k = \eta_k^b e_b(r_k)$. Therefore, after substituting the constraint (3.34), (3.26) gives

$$\left\langle -(\mathrm{d}\tau^{-1}_{h(\Omega_k - \mathcal{A}(r_{k+\alpha})u_k)})^*\mu_k + (\mathrm{d}\tau^{-1}_{-h(\Omega_{k-1} - \mathcal{A}(r_{k-1+\alpha})u_{k-1})})^*\mu_{k-1}, \eta_k^b e_b(r_k) \right\rangle = 0$$

If we define the *discrete Euler-Poincaré* operator DEP_τ according to

$$\mathrm{DEP}_\tau(k) := (\mathrm{d}\tau^{-1}_{h(\Omega_k - \mathcal{A}(r_{k+\alpha})u_k)})^*\mu_k - (\mathrm{d}\tau^{-1}_{-h(\Omega_{k-1} - \mathcal{A}(r_{k-1+\alpha})u_{k-1})})^*\mu_{k-1}, \tag{3.35}$$

then since η_k^b are arbitrary, the vertical equations become

$$\langle \mathrm{DEP}_\tau(k), e_b(r_k) \rangle = 0, \tag{3.36}$$

for $b = 1, ..., \dim(\mathcal{S})$, and $k = 1, ..., N-1$.

3.3.2.2 The Discrete Momentum Map

Next, we define a discrete momentum map, examine its properties and compare it to its continuous analog. It is well-known that for the nonholonomic systems that we consider the momentum, even in the direction of constrained symmetries, is not conserved in general. Instead, the *momentum equation* defines how the momentum components evolve in time. In the discrete setting the vertical equation (or the *discrete momentum equation*) is its analog.

Similar to the continuous setting the discrete vertical equation can be viewed as defining the evolution of a discrete momentum map that we define next.

Definition 8. *Discrete Nonholonomic Momentum Map*

Define the local *discrete momentum map* $\mathrm{J}^{\mathrm{loc}} : TM \times \mathfrak{g} \to \mathfrak{g}^*$ *by*

$$\mathrm{J}^{\mathrm{loc}}(r_k, u_k, \xi_k) = (\mathrm{d}\tau^{-1}_{h\xi_k})^* \mu_k, \qquad \textit{where } \mu_k = \frac{\partial \ell}{\partial \xi}(r_k, u_k, \xi_k),$$

and the spatial *discrete momentum map* $\mathrm{J} : TQ \to \mathfrak{g}^*$ *through*

$$\mathrm{J}(r_k, u_k, g_k, v_k) := \mathrm{Ad}^*_{g_k^{-1}} \mathrm{J}^{\mathrm{loc}}(r_k, u_k, g_k^{-1} v_k),$$

where $(r_k, u_k) \in TM$ *and* $(g_k, v_k) \in TG$.

With these definitions we can compute the evolution of the discrete momentum map along symmetry directions that are allowed by the constraints, i.e. along the elements of the basis $\{e_b(r, g)\}$ at point $(r, g) \in Q$, for $b = 1, ..., \dim(\mathcal{S})$. Note that this basis is constructed from a body-fixed basis $\{e_b(r)\}$ according to $e_b(r, g) = \mathrm{Ad}_g\, e_b(r)$. For all such $e_b : Q \to \mathfrak{s}$ we define the momentum map components $\mathrm{J}^{\mathrm{nh}}_b(r_k, u_k, g_k, v_k)$ at point k by

$$\mathrm{J}^{\mathrm{nh}}_b(r_k, u_k, g_k, v_k) := \langle \mathrm{J}(r_k, u_k, g_k, v_k), e_b(r_k, g_k) \rangle = \langle \mathrm{J}^{\mathrm{loc}}(r_k, u_k, g_k^{-1} v_k), e_b(r_k) \rangle \rangle.$$

Proposition 6. *Discrete Momentum Map Change.*

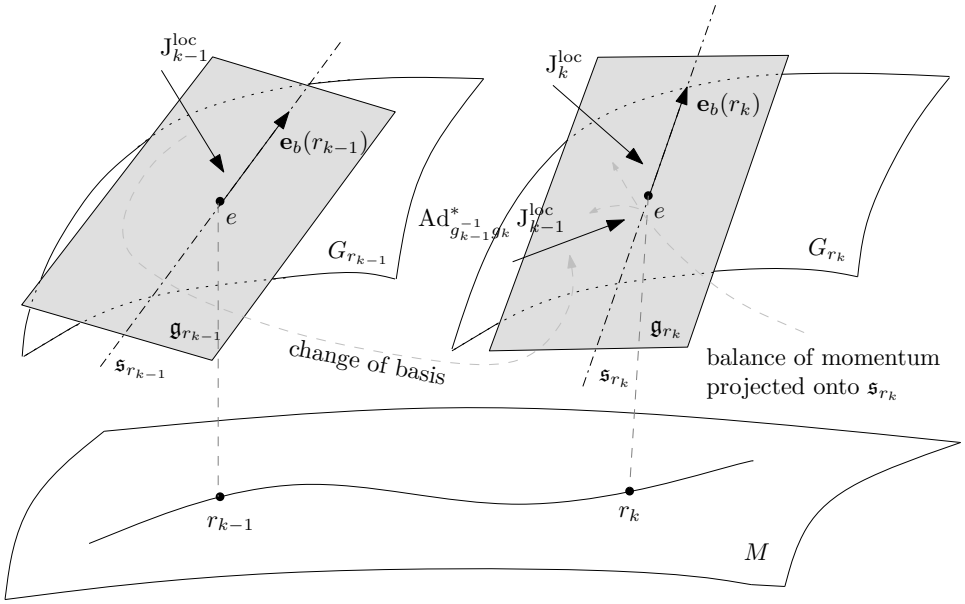


Figure 3.2: Evolution of the discrete momentum map. At point r_{k-1} the map is computed by projecting the covector $\mathrm{J}^{\mathrm{loc}}_{k-1}$ onto $\mathfrak{s}_{r_{k-1}}$ defined by the basis $e_b(r_{k-1})$; then in the Lie algebra basis attached at r_k the covector $\mathrm{J}^{\mathrm{loc}}_{k-1}$ transforms by $\mathrm{Ad}^*_{g_k^{-1}g_{k+1}}$ and the change in the momentum map is computed by subtracting it from the next momentum $\mathrm{J}^{\mathrm{loc}}_k$ and projecting onto $\mathfrak{s}_{r_k}$ (the notation $\mathrm{J}^{\mathrm{loc}}_k := \mathrm{J}^{\mathrm{loc}}(r_k, u_k, \xi_k)$ was used with covectors drawn pointing towards the vectors that they act on).

The momentum components $\mathrm{J}_b^{\mathrm{nh}}$ *evolve along discrete LDAP solution trajectories according to*

$$\begin{aligned}&\mathrm{J}_b^{\mathrm{nh}}(r_k, u_k, g_k, v_k) - \mathrm{J}_b^{\mathrm{nh}}(r_{k-1}, u_{k-1}, g_{k-1}, v_{k-1})\\&= \langle \mathrm{J}(r_{k-1}, u_{k-1}, g_{k-1}, v_{k-1}), e_b(r_k, g_k) - e_b(r_{k-1}, g_{k-1})\rangle.\end{aligned}$$

Proof. Rewriting the momentum equation (3.36), derived form the discrete LDAP principle, in terms of the momentum map we obtain

$$\langle \mathrm{J}^{\mathrm{loc}}(r_k, u_k, \xi_k) - \mathrm{Ad}^*_{\tau(h\xi_{k-1})}\, \mathrm{J}^{\mathrm{loc}}(r_{k-1}, u_{k-1}, \xi_{k-1}), e_b(r_k)\rangle = 0,$$

which is a momentum map balance equation depicted in Fig. 3.2. In spatial frame it reads

$$\langle \mathrm{J}(r_k, u_k, g_k, g_k\xi_k) - \mathrm{J}(r_{k-1}, u_{k-1}, g_{k-1}, g_{k-1}\xi_{k-1}), e_b(r_k, g_k)\rangle = 0$$

which yields the component difference. □

Corollary 2. *Properties of the momentum map*

1. *The map components* $\mathrm{J}_b^{\mathrm{nh}}$ *are not conserved in general.*

2. *If* $e_b(r)$ *are independent of* r*, then the discrete momentum equations are the discrete Euler-Poincaré equations projected onto the constraint symmetry space* $\mathfrak{s}$*.*

3. *If* $e_b(r)$ *are independent of* r *and if* G *is abelian then* $\mathrm{J}_b^{\mathrm{nh}}$ *are constant along the discrete trajectory. This follows from the equality* $e(r_k, g_k) = e(r_{k-1}, g_{k-1})$ *since in this special case* $e_b(r_k) = e_b(r_{k+1})$ *and* $\mathrm{Ad}_g = \mathrm{Id}$*.*

3.3.2.3 Horizontal Equations

Horizontal variations (Sec. 3.2.2) are constrained according to $g_k^{-1}\delta g_k = \eta_k$, where $\eta_k = -\mathcal{A}(r_k)\delta r_k$ for variations δr_k in the base. With this definition of η_k and after substituting p_k from (3.31) into (3.25) we get

$$\begin{aligned}
&\left\langle h\left(\alpha\frac{\partial l^c_{k-1+\alpha}}{\partial r} + (1-\alpha)\frac{\partial l^c_{k+\alpha}}{\partial r}\right) - \left(\frac{\partial l^c_{k+\alpha}}{\partial u} + \frac{\partial l^c_{k-1+\alpha}}{\partial u}\right)\right. \\
&\left. + h\left(\alpha f_{k-1+\alpha} + (1-\alpha)f_{k+\alpha}\right), \delta r_k\right\rangle \\
&= \langle \mu_k, \mathcal{A}(r_{k+\alpha})\delta r_k\rangle - \langle \mu_{k-1}, \mathcal{A}(r_{k-1+\alpha})\delta r_k\rangle \\
&\quad - h\left\langle \mu_{k-1}, \alpha D\mathcal{A}_{k-1+\alpha}(\delta r_k, u_{k-1})\right\rangle - h\left\langle \mu_k, (1-\alpha)D\mathcal{A}_{k+\alpha}(\delta r_k, u_k)\right\rangle \\
&\quad - \langle \mathrm{DEP}_\tau(k), \mathcal{A}(r_k)\delta r_k\rangle
\end{aligned} \tag{3.37}$$

for $k = 1, ..., N-1$.

3.3.2.4 The case of linear connection

Next we study the special case when the connection $\mathcal{A}(r)$ is linear in the base point r. This case is useful in comparing the resulting integrator to the continuous case in order to gain insight into the effect of discretization.

Assume that $\mathcal{A}(r)$ is linear. The following expressions then trivially hold

$$\mathcal{A}(r_{k+\alpha})\delta r_k = \mathcal{A}(r_k)\delta r_k + h(1-\alpha)D\mathcal{A}_{k+\alpha}(u_k, \delta r_k),$$

$$\mathcal{A}(r_{k-1+\alpha})\delta r_k = \mathcal{A}(r_k)\delta r_k - h\alpha D\mathcal{A}_{k-1+\alpha}(u_{k-1}, \delta r_k),$$

and after substituting them into (3.37) and using $\tau = \exp$ the horizontal equations become

$$\begin{aligned}
&\left\langle \alpha \frac{\partial l^c_{k-1+\alpha}}{\partial r} + (1-\alpha)\frac{\partial l^c_{k+\alpha}}{\partial r} - \frac{1}{h}\left(\frac{\partial l^c_{k+\alpha}}{\partial u} - \frac{\partial l^c_{k-1+\alpha}}{\partial u}\right) \right. \\
&\left. + \alpha f_{k-1+\alpha} + (1-\alpha) f_{k+\alpha}, \delta r_k \right\rangle \\
&= \alpha \left\langle \mu_{k-1}, d\mathcal{A}_{k-1+\alpha}(u_{k-1}, \delta r_k) - \sum_{i=1}^{\infty} \frac{B_i}{i!} \operatorname{ad}^i_{\Omega_{k-1} - \mathcal{A}(r_{k-1+\alpha})u_{k-1}} \mathcal{A}(r_k)\delta r_k \right\rangle \\
&\quad + (1-\alpha) \left\langle \mu_k, d\mathcal{A}_{k+\alpha}(u_k, \delta r_k) - \sum_{i=1}^{\infty} \frac{B_i}{i!} \operatorname{ad}^i_{\Omega_k - \mathcal{A}(r_{k+\alpha})u_k} \mathcal{A}(r_k)\delta r_k \right\rangle
\end{aligned} \tag{3.38}$$

where the curvature covariant derivative $d\mathcal{A}$ is defined by

$$d\mathcal{A}(u, \delta r) = D\mathcal{A}(u, \delta r) - D\mathcal{A}(\delta r, u),$$

and B_i are the Bernoulli numbers with the first few given by $B_1 = -1/2$, $B_2 = 1/6$, $B_3 = 0$, ...

There are several special cases that lead to further simplification of the horizontal equations. The Lie bracket in (3.38) vanishes, for instance, when G is abelian; when $\mathfrak{g}$ is one-dimensional; or whenever Ω and $\mathcal{A}(r)u$ lie in the same one-dimensional vector space for all Ω, r, and u (as in the snakeboard example from Sec. 3.4.2).

Proposition 7. *If the connection $\mathcal{A}(r)$ is linear and the* ad *operator in* (3.38) *maps to* 0 *along the path, then the non-conservative forces on the right-hand side of the reduced discrete Euler-Lagrange equations* (3.37) *match the continuous case exactly.*

Proof. If the ad operator maps to 0 then the curvature equals the covariant derivative, i.e. $\mathcal{B} = d\mathcal{A}$. Then if we denote the continuous gyroscopic force by $F^{\mathcal{A}}(r, u, \mu)_\beta = \langle \mu, d\mathcal{A}(u, \delta r^\beta) \rangle$, the discrete forces on the right-hand side of (3.38) become

$$\alpha F^{\mathcal{A}}(r_{k-1+\alpha}, u_{k-1}, \mu_{k-1}) + (1-\alpha) F^{\mathcal{A}}(r_{k+\alpha}, u_k, \mu_k)$$

exactly representing the continuous force acting on the left and the right (depending on the value of α) of the fiber at r_k. □

This claim is analogous to the result obtained by Cortes [13] for Chaplygin-type symmetries and, as noted by the same author, if the gyroscopic forces vanish then the horizontal equations become a decoupled variational integrator on their own. Prop. 7 asserts that under similar conditions (linearity of the connection and vanishing of the bracket) this is also true for systems with nontrivial intersection between the constraints and symmetries.

3.3.2.5 Summary

The discrete equations of motion are

$$
\begin{aligned}
& g_k^{-1} g_{k+1} = \tau(h(\Omega_k - \mathcal{A}(r_{k+\alpha}) u_k)), \\
& r_{k+1} - r_k = h u_k, \\
& \mu_k = \frac{\partial l^c_{k+\alpha}}{\partial \Omega}, \\
& \langle \mathrm{DEP}_\tau(k), e_b(r_k) \rangle = 0, \\
& \Big\langle h \left(\alpha \frac{\partial l^c_{k-1+\alpha}}{\partial r} + (1-\alpha) \frac{\partial l^c_{k+\alpha}}{\partial r} \right) - \left(\frac{\partial l^c_{k+\alpha}}{\partial u} + \frac{\partial l^c_{k-1+\alpha}}{\partial u} \right) \\
& + h \left(\alpha f_{k-1+\alpha} + (1-\alpha) f_{k+\alpha} \right), \delta r_k \Big\rangle \\
& = \langle \mu_k, \mathcal{A}(r_{k+\alpha}) \delta r_k \rangle - \langle \mu_{k-1}, \mathcal{A}(r_{k-1+\alpha}) \delta r_k \rangle \\
& \quad - h \langle \mu_{k-1}, \alpha D\mathcal{A}_{k-1+\alpha}(\delta r_k, u_{k-1}) \rangle - h \langle \mu_k, (1-\alpha) D\mathcal{A}_{k+\alpha}(\delta r_k, u_k) \rangle \\
& \quad - \langle \mathrm{DEP}_\tau(k), \mathcal{A}(r_k) \delta r_k \rangle
\end{aligned}
\tag{3.39}
$$

for $b = 1, ..., \dim(\mathcal{S})$ and $k = 1, ..., N-1$.

Horizontal Equations (reduced Lagrangian ℓ) It is also useful to express the base equations in terms of the reduced Lagrangian ℓ as well since our discrete formulation in general does not decouple the instantaneous change in the momentum components from that of the shape variables. While deriving the integrator in terms of l^c is more

appropriate for analysis often working with the reduced Lagrangian ℓ results in simpler implementation. In that case the horizontal equations are

$$\begin{aligned}&\left(\frac{\partial \ell_{k+\alpha}}{\partial u} - \frac{\partial \ell_{k-1+\alpha}}{\partial u}\right) - h\left(\alpha \frac{\partial \ell_{k-1+\alpha}}{\partial r} + (1-\alpha)\frac{\partial \ell_{k+\alpha}}{\partial r}\right) \\ &= \mathcal{A}(r_k)^* \,\mathrm{DEP}_\tau(k) + h\left(\alpha f_{k-1+\alpha} + (1-\alpha) f_{k+\alpha}\right).\end{aligned} \tag{3.40}$$

3.4 Examples

3.4.1 Car with simple dynamics

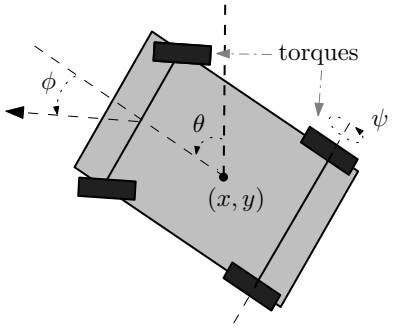


Figure 3.3: Car: pose & shape space variables.

We study the kinematic car model defined in [31] with added simple dynamics (Fig. 3.3). The configuration space is $Q = S^1 \times S^1 \times SE(2)$ with coordinates $q = (\psi, \sigma, \theta, x, y)$, where (θ, x, y) are the orientation and position of the car, ψ is the rolling angle of the rear wheels, and σ is defined by $\sigma = \tan(\phi)$ where ϕ is the steering angle. The car has mass m, rear wheel inertia I, rotational inertia K, and we assume that the steering inertia is negligible.

The car is controlled by rear wheels torque f^{ψ} and steering velocity u^{σ}. The Lagrangian is then expressed as:

$$L(q,\dot{q}) = \frac{1}{2}\left(I\dot{\psi}^2 + K\dot{\theta}^2 + m(\dot{x}^2 + \dot{y}^2)\right),$$

and the constraints (see [31]) are

$$\cos\theta dx + \sin\theta dy = \rho d\psi,$$

$$-\sin\theta dx + \cos\theta dy = 0,$$

$$d\theta = \frac{\rho}{l}\sigma d\psi,$$

where l is the distance between front and rear wheel axles, and ρ is the radius of the wheels. These constraints simply encode the fact that the car must turn in the direction in which the front wheels are pointing, that the car cannot slide sideways, and that the change in orientation is proportional to the steering angle and turning rate of the wheels.

Note now that for any element $g = (\alpha, a, b)$ of $SE(2)$, the action $\Phi_g(q) = (\phi_\rho, \phi_L, \theta + \alpha, a + \cos(\alpha)x - \sin(\alpha)y, b + \sin(\alpha)x + \cos(\alpha)y)$ leaves the Lagrangian and constraints invariant. As the shape coordinates are $r = (\psi, \sigma)$, the *reduced Lagrangian* thus becomes

$$\ell(r,u,\xi) = \frac{1}{2}\left(u^T \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} u + \check{\xi}^T \begin{bmatrix} K & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & m \end{bmatrix} \check{\xi}\right).$$

The matrix representation of the connection $\mathcal{A}$ dependent on r becomes:

$$[\mathcal{A}(r)] = \begin{bmatrix} -\frac{\rho}{l}\sigma & 0 \\ -\rho & 0 \\ 0 & 0 \end{bmatrix} \tag{3.41}$$

This model is an example of the *principle kinematic case* in which the constraint distribution complements the space tangent to the group orbits. This is easily seen noting that

$$\mathcal{D}_q = \text{span}\left\{\frac{\partial}{\partial\psi}, \frac{\partial}{\partial\sigma}\right\}, \qquad \mathcal{V}_q = \text{span}\left\{\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial\theta}\right\}$$

Thus, $\mathcal{S} = \emptyset$ and there is no momentum equation.

Continuous equations of motion The resulting continuous equations of motion are

$$\begin{aligned}
&\dot{x} = \rho\cos\theta\dot{\psi}, \\
&\dot{y} = \rho\sin\theta\dot{\psi}, \\
&\dot{\theta} = \frac{\rho\sigma}{l}\dot{\psi}, \\
&\left(I + m\rho^2 + \frac{K\rho^2\sigma^2}{l^2}\right)\dot{u}^{\psi} = -\frac{K\rho^2\sigma\dot{\psi}\dot{\sigma}}{l^2} + f^{\psi}, \\
&\dot{\sigma} = u^{\sigma},
\end{aligned}$$

Car Integrator The discrete equations of motion will be derived by substituting the Lagrangian and the connection of the steered car into (3.39). Define $u = (u^{\psi}, u^{\sigma})$ and $\xi = -\mathcal{A}(r)\cdot u$ and pick $\tau = \exp$. The expression $\langle \text{DEP}_{\exp}(k), \mathcal{A}(r_k)\cdot\delta r_k\rangle$ involves the term

dexp^{-1} defined in (2.13). Observing that in the case of the car $\langle \mathrm{ad}^*_{\mathcal{A}(r)\cdot u}\,\mu, \mathcal{A}(r)\cdot\delta\rangle = 0$ for any $\mu \in \mathfrak{h}^*$ and $u, \delta \in TM$ and therefore

$$\langle (\mathrm{dexp}^{-1}_{\xi})^*\mu, \mathcal{A}(r)\cdot\delta\rangle = \langle \mu, \mathcal{A}(r)\cdot\delta\rangle$$

and since the operator $\mathrm{DEP}_{\exp}$ simplifies to

$$\langle \mathrm{DEP}_{\exp}(k), \mathcal{A}(r_k)\cdot\delta\rangle := \langle \mu_k - \mu_{k-1}, \mathcal{A}(r_k)\cdot\delta\rangle,$$

the equations of motion simplify to

$$\begin{aligned}
g_{k+1} &= g_k \exp(-h\mathcal{A}(r_{k+\alpha})\cdot u_k),\\
r_{k+1} &= r_k + hu_k,\\
\frac{\partial \ell_{k+\alpha}}{\partial u} - \frac{\partial \ell_{k-1+\alpha}}{\partial u} &= A(r_k)^T(\check{\mu}_k - \check{\mu}_{k-1}) + h\left(\alpha f_{k-1+\alpha} + (1-\alpha) f_{k+\alpha}\right),
\end{aligned}$$

for $k = 1, ..., N-1$. The exponential and Cayley maps for $G = SE(2)$ are given in Sec. 2.6.2.

The equations of motion can now be derived by substituting

$$\mu = \frac{\partial \ell}{\partial \xi} = (K\xi^1, m\xi^2, 0), \qquad \frac{\partial \ell}{\partial r} = (0,0), \qquad \frac{\partial \ell}{\partial u} = (Iu^{\psi}, 0),$$

The exact equations are

$$x_{k+1} - x_k = \begin{cases} \frac{v_k}{\omega_k}(\sin(\theta_k + h\omega_k) - \sin\theta_k) & \text{if } \omega \neq 0; \\ \cos\theta_k h v_k & \text{if } \omega = 0. \end{cases}$$

$$y_{k+1} - y_k = \begin{cases} \frac{v_k}{\omega_k}(-\cos(\theta_k + h\omega_k) + \cos\theta_k) & \text{if } \omega \neq 0; \\ \sin\theta_k h v_k & \text{if } \omega = 0. \end{cases}$$

$$\theta_{k+1} = \theta_k + h\omega_k,$$

$$\sigma_{k+1} = \sigma_k + h u_k^{\sigma},$$

$$(I + \rho^2 m)(u_k^{\psi} - u_{k-1}^{\psi}) + \frac{\rho^2 K}{l^2}\sigma_k(\sigma_{k+\alpha}u_k^{\psi} - \sigma_{k-1+\alpha}u_{k-1}^{\psi}) = h\left(\alpha f_{k-1+\alpha}^{\psi} + (1-\alpha)f_{k+\alpha}^{\psi}\right),$$

where $v_k = \rho u_k^{\psi}$, $\omega_k = (\rho/l)\sigma_{k+\alpha}u_k^{\psi}$. Thus, the integrator is easily computed as it is *fully explicit* for any choice of quadrature point α.

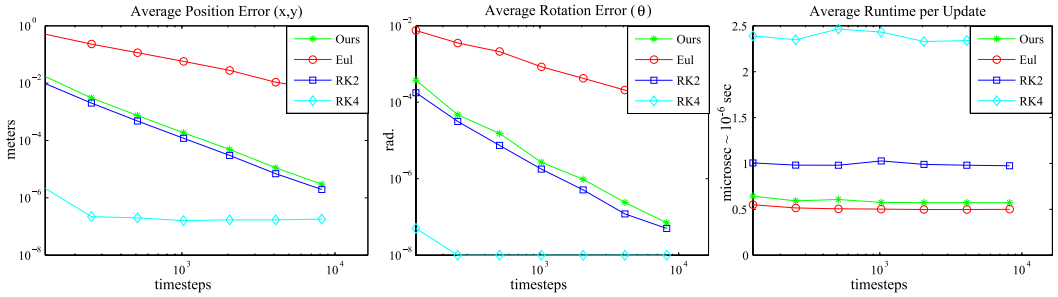


Figure 3.4: Stability and efficiency of our nonholonomic integrator for car trajectories: averaged over 50 runs using a large range of initial conditions and steering commands, our nonholonomic integrator remains as accurate as RK2, at a fraction of the computational complexity.

Numerical Comparisons Our numerical comparisons for nonholonomic motions are based on one-minute trajectories of the car with simple dynamics. The vehicle is controlled using sinusoidal inputs of frequency and amplitude designed to produce nontrivial paths such as parallel parking, sharp turns, and winding maneuvers. Since the trajectory

is relatively short all RK methods up to fourth order that we test (Fig. 3.4) are stable due in part to the simpler group structure of $SE(2)$. Yet, our integrator performs almost as equally well as $RK2$, at *half* the computational time—due to its explicit update scheme.

3.4.2 The Snakeboard

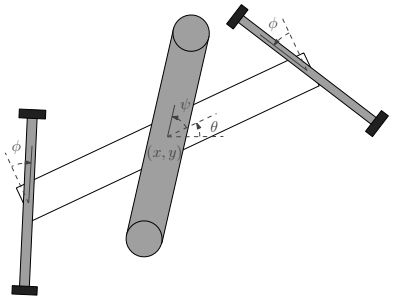


Figure 3.5: Snakeboard: pose & shape space variables.

The snakeboard is a wheeled board closely resembling the popular skateboard with the main difference that both the front and the rear wheels can be steered *independently*. This feature causes an interesting interplay between momentum conservation and the nonholonomic constraints: the rider is able build up velocity *without pushing off the ground* by transferring the momentum generated by a twist of the torso into motion of the board coupled with steering of the wheels through pivoting of the feet. When the steering wheels stop turning the systems moves along a circular arc and the momentum around the center of this rotation is conserved. A robotic version of the snakeboard also exists, equipped with a momentum-generating rotor and steering servos [45].

The shape space variables of the snakeboard are $r = (\psi, \phi) \in S \times S$ denoting the rotor angle and the steering wheels angle, while its configuration is defined by (θ, x, y) denoting orientation and position of the board (see Figure 3.5). This corresponds to

a configuration space $Q = S \times S \times SE(2)$ with shape space $M = S \times S$ and group $G = SE(2)$. Additional parameters are its mass m, distance l from its center to the wheels, and moments of inertia I and J of the board and the steering. The kinematic constraints of the snakeboard are:

$$-l\cos\phi d\theta - \sin(\theta+\phi)dx + \cos(\theta+\phi)dy = 0,$$

$$l\cos\phi d\theta - \sin(\theta-\phi)dx + \cos(\theta-\phi)dy = 0,$$

enforcing the fact that the system must move in the direction in which the wheels are pointing and spinning. The constraint distribution is spanned by three covectors:

$$\mathcal{D}_q = \text{span}\left\{\frac{\partial}{\partial\psi}, \frac{\partial}{\partial\phi}, c\frac{\partial}{\partial\theta} + a\frac{\partial}{\partial x} + b\frac{\partial}{\partial y}\right\},$$

where $a = -2l\cos\theta\cos^2\phi,\ b = -2l\sin\theta\cos^2\phi,\ c = \sin 2\phi$. The group directions defining the vertical space are:

$$\mathcal{V}_q = \text{span}\left\{\frac{\partial}{\partial\theta}, \frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right\},$$

and therefore the constrained symmetry space becomes:

$$\mathcal{S}_q = \mathcal{V}_q \cap \mathcal{D}_q = \text{span}\left\{c\frac{\partial}{\partial\theta} + a\frac{\partial}{\partial x} + b\frac{\partial}{\partial y}\right\}. \tag{3.42}$$

Since $\mathcal{D}_q = \mathcal{S}_q \oplus \mathcal{H}_q$, we have $\mathcal{H}_q = \text{span}\left\{\frac{\partial}{\partial\psi}, \frac{\partial}{\partial\phi}\right\}$. Finally, the Lagrangian of the system is $\mathcal{L}(q,\dot{q}) = \frac{1}{2}\dot{q}^T M \dot{q}$ where

$$\mathrm{M} = \begin{bmatrix} I & 0 & I & 0 & 0 \\ 0 & 2J & 0 & 0 & 0 \\ I & 0 & ml^2 & 0 & 0 \\ 0 & 0 & 0 & m & 0 \\ 0 & 0 & 0 & 0 & m \end{bmatrix}.$$

The reduced Lagrangian is, therefore: $\ell(r,u,\xi) = (u,\check{\xi})^T \,\mathrm{M}\, (u,\check{\xi})$. There is only one direction along which snakeboard motions lead to momentum conservation: it is defined by the basis vector

$$\check{e}_1(r) = 2l\cos^2\phi \begin{bmatrix} \frac{\tan\phi}{l} \\ -1 \\ 0 \end{bmatrix},$$

and, hence, there is only one momentum variable $\mu_1 = \left\langle \frac{\partial \ell}{\partial \xi}, e_1(r) \right\rangle$. Using this variable we can derive the connection according to [45, 2] as

$$[\mathcal{A}] = \begin{bmatrix} \frac{I}{ml^2}\sin^2\phi & 0 \\ -\frac{I}{2ml}\sin 2\phi & 0 \\ 0 & 0 \end{bmatrix}, \text{and} \quad \Omega = \frac{\mu_1}{4ml^2\cos^2\phi} e_1(r).$$

Continuous Equations of Motion The dynamics of the system can be derived either in terms of Ω or in terms of μ as unknown variables. Here, we provide the resulting

equations of motion based on μ since this has been the choice in previous work and will be easier to compare against. The continuous equations of motion (Sec. 3.2.3.2) can be derived as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \left(\frac{1}{2ml}(\mu_1 - \sin 2\phi\dot{\psi}) \begin{bmatrix} -1 \\ 0 \\ \frac{\tan\phi}{l} \end{bmatrix} \right), \tag{3.43}$$

$$\dot{\mu}_1 = 2I\cos^2\phi\dot{\psi}\dot{\phi} - \mu_1\tan\phi\dot{\phi}, \tag{3.44}$$

$$\left(1 - \frac{I}{ml^2}\sin^2\phi\right)\ddot{\psi} = \frac{I}{2ml^2}\sin 2\phi\dot{\psi}\dot{\phi} - \frac{1}{2ml^2}\dot{\phi}p + \frac{1}{I}f_\psi, \tag{3.45}$$

$$\ddot{\phi} = \frac{1}{2J}f_\phi. \tag{3.46}$$

A nonholonomic integrator The discrete equations of motion will be derived by substituting the Lagrangian and the connection of the snakeboard into (3.40). It seems more convenient from implementation point of view to express the Euler-Lagrange equations of the base dynamics using the reduced Lagrangian ℓ rather than the constrained reduced Lagrangian l^c so we use (3.40) instead of (3.39). We also need to choose the map τ and in the remainder of this section we set $\tau = \exp$, i.e. the exponential map.

The expression $\langle \mathrm{DEP}_{\exp}(k), e_b(r_k)\rangle$ involves the term dexp^{-1} defined in (2.13). Observing that in the case of the snakeboard $\langle \mathrm{ad}^*_\xi \mu, \eta\rangle = 0$ for any $\mu \in \mathfrak{h}^*$ and $\xi, \eta \in \mathfrak{s}$ and therefore

$$\langle (\mathrm{dexp}^{-1}_\xi)^*\mu, e_1(r)\rangle = \langle \mu, e_1(r)\rangle$$

and the operator $\mathrm{DEP}_{\exp}$ simplifies to

$$\langle \mathrm{DEP}_{\exp}(k), e_1(r_k) \rangle := \langle \mu_k - \mu_{k-1}, e_1(r_k) \rangle$$

Note that since $\langle \mathrm{DEP}_{\exp}(k), e_1(r_k) \rangle = 0$ then $\langle \mathrm{DEP}_{\exp}(k), \mathcal{A}(r_k)\cdot \rangle = 0$ since $\mathcal{A}(r_k)\cdot\delta \in \mathfrak{s}_r$. The equations of motion become

$$\begin{aligned}
&g_k^{-1} g_{k+1} = \exp(h(\Omega_k - \mathcal{A}(r_{k+\alpha}) \cdot u_k)), \\
&r_{k+1} - r_k = h u_k, \\
&\mu_k = \frac{\partial l_{k+\alpha}}{\partial \xi}, \\
&\langle \mu_k - \mu_{k-1}, e_1(r_k) \rangle = 0, \\
&\frac{\partial \ell_{k+\alpha}}{\partial u} - \frac{\partial \ell_{k-1+\alpha}}{\partial u} = h\left(\alpha f_{k-1+\alpha} + (1-\alpha) f_{k+\alpha}\right)
\end{aligned}$$

for $k = 1, ..., N-1$.

Defining $\xi = \Omega - \mathcal{A}(r)\cdot u$ and $u = (u^\psi, u^\phi)$ the equations are derived by substituting

$$\mu = (ml^2\xi^1 + Iu^\phi, m\xi^2, 0), \qquad \frac{\partial \ell}{\partial u} = (I(u^\psi + \xi^1), 2Ju^\phi).$$

Numerical Comparisons Our numerical comparisons for nonholonomic motions are based on one-minute trajectories of the snakeboard. The vehicle is controlled using sinusoidal inputs of frequency and amplitude designed to produce nontrivial paths such as parallel parking, sharp turns, and winding maneuvers. Since the trajectory is relatively short all RK methods up to fourth order that we test (Fig. 3.6) are stable due in part

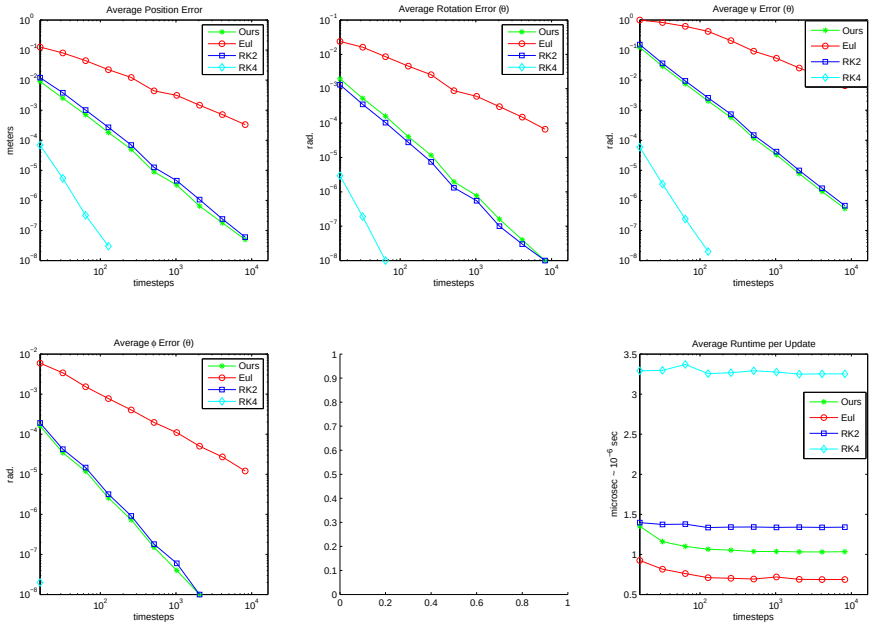


Figure 3.6: Stability and efficiency of our nonholonomic integrator for snakeboard trajectories: averaged over 50 runs using a large range of initial conditions and steering commands, our nonholonomic integrator remains as accurate as RK2, at a fraction of the computational complexity.

to the simpler group structure of $SE(2)$. Yet, our integrator performs better $RK2$, at a fraction of the computational time—due to its explicit update scheme.

Chapter 4

Global Motion Planning

4.1 Introduction

There are two main disadvantages in using optimal control methods in practical applications directly. First, if numerous complicated nonlinear constraints were imposed then finding a feasible solution would be extremely difficult and costly. Second, even if a feasible solution is found it is in general only locally optimal and there is no way of knowing how "good" it is unless there is specific prior knowledge. In this chapter we propose a way to overcome these limitations by combining DMOC with a family of methods known as *sampling-based roadmaps* in order to efficiently compute near globally optimal solution trajectories even in very complex scenarios.

The main idea is to precompute many simple and versatile optimal motions using DMOC offline, that can then be combined into longer more complicated trajectories that can accomplish the given task. Furthermore, these motions can be combined to form not only a single trajectory but also a tree or a graph whose edges are the motions. This graph can be expanded towards parts of the space that might lead to more optimal trajectories

and it would also contain edges reaching the goal state–therefore, such a graph, termed *roadmap*, contains many possible trajectories to the goal. Finding the optimal trajectory then amounts to finding the shortest path in the graph. Thus, a roadmap representation has two main advantages: it provides a compact way to encode many different paths from the start state; it lends itself to well-established *dynamic programming* methods for finding an optimal solution satisfying the dynamics by construction. In essence, the roadmap framework can be used to transform a very high-dimensional differential problem into a lower dimensional algebraic problem (figuring out how to sequence simple motions) and graph search problem (finding the best path). The solution is as good as the approximation–the bigger and denser the space the roadmap covers and the richer the set of primitive motions used, the more optimal the solution would be. Fortunately, this is a very active research area in robotics and various strategies exist for constructing roadmaps in order to solve motion planning problems with many degrees of freedom, from coordinating multiple humanoid robots in urban environments to docking molecules in protein design.

Challenges When the system dynamics is nonlinear or when the state space has complicated boundary and is non-simply connected, e.g. arising from joint limits, environment obstacles or velocity bounds, then there would be many locally optimal solution trajectories. For instance, even in the absence of physical obstacles, the computation of minimum-acceleration trajectory for a non-spherical rigid-body could lead to multiple locally optimal solutions based on how the variational method was initialized. This is because the necessary conditions for optimality are not affine and could have multiple roots.

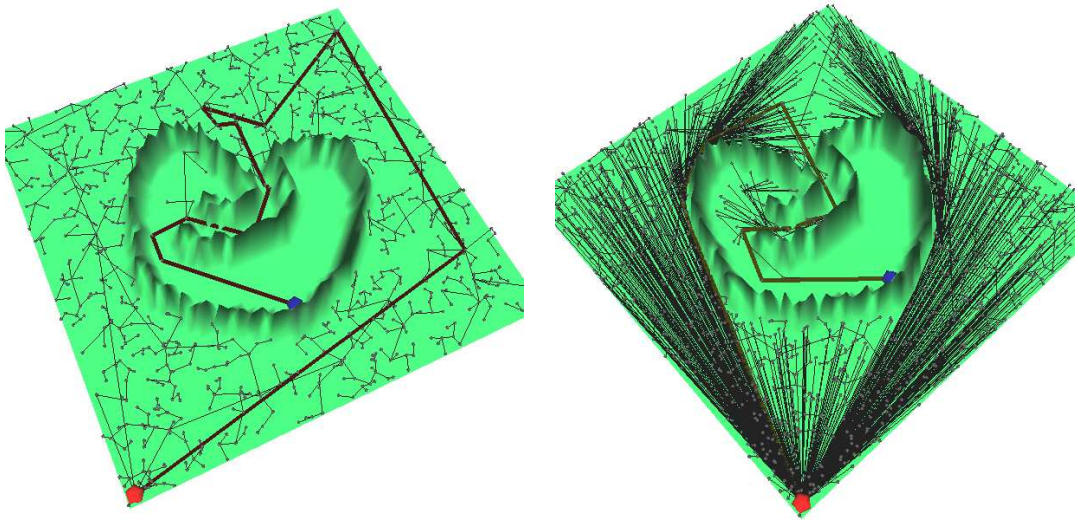
When obstacle are present and must be avoided the situation becomes more complicated since they immediately induce multiple homotopy classes of paths in the configuration space and an even more complicated topology of the state space (the tangent bundle) when velocity limits are present.

The importance of exploration The only way to guarantee that a trajectory is optimal is to generate trajectories that 1) visit every region in the reachable state space, and 2) then reach the goal state. This would guarantee that all possible routes to the state goal have been taken and the best one chosen. Since it is impossible to produce such infinite number of trajectories a realistic approach would be to produce trajectories that visit only a subset of the state space that is likely to lead to good solutions. This gives rise to a trade-off known as *exploration vs. exploitation* known in robotics and reinforcement learning, where *exploration* refers to controlling trajectories to visit unexplored but potentially leading to a more optimal solution regions, while *exploitation* amounts to then guiding this trajectory to achieving the goal or task. Often, a complicated state space would require more extensive exploration while a simple motion planning problem can be solved solely with exploitation (e.g. the simple case of stabilizing a linear system without obstacles is a convex optimal control problem that does not need exploration).

Related Work Sampling-based motion planning has been a very active research area during the past decade. Good summary of current methods can be found in the recent books [35, 11]. Multiple research groups focus on the development of new sampling-based techniques motivated by wide variety of applications not only in robotics, but also in areas such as molecular biology and graphics. More prominent research teams include

the group lead by Latombe at Stanford, Kavraki at Rice, LaValle at the University of Illinois at Urbana-Champagne, Amato at Texas AM, and others. These researchers have developed various methods and techniques applicable to a broad spectrum of examples, from planning for multiple digital actors in virtual worlds to computing optimal docking maneuvers for complex protein structures. A full review of the planning literature is beyond the scope of this thesis and in the following sections we would only briefly describe the methods applicable to the particular problems considered.

4.2 Sampling-based Roadmaps



Rapidly-exploring random tree incremental roadmap

Figure 4.1: A simple example: finding a trajectory from the start (lower-left corner) to a goal state inside the "bug-trap". A rapidly-exploring random tree (RRT) is used (left) to quickly explore the environment and find any path (usually far from optimal). Once a path is found using RRT, the expansion switches to an incremental roadmap method (right) that focuses on finding a more optimal solution.

The term roadmap can be associated with several different underlying representation based on the type of system and motion planning task required. Its most widely used

representation is an undirected graph which is applicable to kinematic systems without differential constraints and offers the ability to find a motion between any two configurations in the graph [35]. Methods in this category are called *multiple-query probabilistic roadmaps* (PRM). When computing a motion plan for a mechanical systems with differential constraints a roadmap is constructed as a directed graph or a tree rooted at the starting state. Different strategies exists for guiding the roadmap expansion. A family of methods called *rapidly-exploring dense trees* (RDT) which include the more well known *rapidly-exploring random trees* (RRT) are designed to quickly explore the state space to find *any* feasible path to the goal (see Fig.. There is no notion of optimality in these methods since they involve cost metrics aimed at fast expansion rather than considering the full cost of achieving the motion task. In contrast, *incremental PRM* methods for mechanical systems have also been proposed (e.g. [21]) that are similar to RDT but focus on reaching the goal while minimizing a trajectory cost metric and producing optimal trajectories. In general, the RDT method finds any feasibly path quickly, while the PRM approach finds a more optimal path with more computational effort. Since we are interested in motion planning for mechanical systems under differential constraints, we use the term "PRM" (possibly conflicting with terminology used by other authors) to indicate incremental planners that also include the RDT types. In fact, similar to [21] our implementations use a hybrid method between RDT and single-query PRM that initially focuses solely on exploring the space and then interleaves exploration with narrowing down a near-optimal solution.

Problems of Interest There are two types of problems that we are interested in. The first are typical motion planning problems of moving from a given start state to a goal state minimizing a given cost function defined as a metric on the state space. The second are problems of finding a trajectory that minimizes a more general cost function subject to constraints but does not need to end at a predefined goal state. An example of the second kind is computing a trajectory that achieves maximum sensor coverage of the environment subject within a given time. The second type of problems are much harder since the cost function often depends on the whole trajectory and is not directly related to a metric on the original state space.

Our approach to achieving global exploration of the state space is based on the RRT model [35] and the incremental PRM method [21]. The underlying structure is a tree $\mathcal{T}$ with nodes $\mathcal{N}$ and edges $\mathcal{E}$. A node is denoted $\mathbf{n} \in \mathcal{N}$ and an edge between two nodes $\mathbf{n}_a$ and $\mathbf{n}_b$ is denoted $\mathbf{e}_{(a,b)} \in \mathcal{E}$.

Random tree construction The RRT exploration algorithm is given in Fig. 4.2. The RRT expands by sampling a new node from the state space, finding the closest existing node in the tree, and then extending the tree from that node towards the sampled node. Either a partial or complete (reaching the node) trajectory can be produced based on which strategy is used.

Incremental roadmap construction The incremental roadmap planning algorithm is depicted in Fig. 4.2. The algorithm is given a root node $\mathbf{n}_r$ and goal node $\mathbf{n}_g$ containing the initial and final desired states. A given minimum desired trajectory cost c_{des} serves as a termination condition. At every iteration a new node $\mathbf{n}_s$ is sampled from the state space.

```
RRTExplore(n_r)
- - - - - - - - - - - - - - - - - - - - -
  N = {n_r}
  WHILE (! Terminate(N))
    n_s := Sample()                          (sample new node)
    n_b := NearestNeighbor(N, n_s)           (find nearest neighbor from tree)
    e_(b,s') := Extend(n_b, n_s)             (extend it towards sample)
    IF (e_(b,s'))                            (if successfully extended)
      N := N ∪ n_s'
      E := E ∪ e_(b,s')
```

Figure 4.2: Rapidly-exploring random tree (RRT) exploration pseudocode

The tree expansion relies on a *distance function* between the nodes. The ideal distance function is the same of the objective function to be minimized. Since newly sampled nodes are not yet connected to the tree, there is no way of knowing what this cost is and therefore a heuristic distance is used that underestimates the true distance. All existing nodes are then sorted using this cost in an array $\mathcal{N}_b$. The tree is then *extended* from the best node (with lowest cost) in this array $\mathbf{n}_b$ to the sample. The function Extend relies on a *local planner* that usually reaches the sample node $\mathbf{n}_s$ only approximately resulting in a new node $\mathbf{n}_{s'}$. If the new resulting cost at $\mathbf{n}_{s'}$ has a chance of improving the current best cost c_{best} then the edge is added. Then, an attempt is made to reach the goal from the newly added node. If this succeeds and the best cost is improved, then the tree can be pruned and the algorithm continues.

Local planner Local planning between nodes can be accomplished in several different ways. In some cases, e.g. when considering unconstrained kinematic systems, a trajectory can be computed simply using interpolation. A more general method applicable to systems with dynamics or underactuation is to use a controller that stabilizes the system

PrmPlan($\mathbf{n}_r, \mathbf{n}_g, c_{des}$)

$\mathcal{N} = \{\mathbf{n}_r\}$
$c_{best} = \infty$
WHILE ($c_{best} > c_{des}$)
  $\mathbf{n}_s = \text{Sample}()$ *(sample new node)*
  $\mathcal{N}_b = \text{Sort}(\mathcal{N}, \mathbf{n}_s)$ *(sort by distance to $\mathbf{n}_s$)*
  FOR $i = 1 : \text{size}(\mathcal{N}_b)$
    $\mathbf{n}_b = \mathcal{N}_b(i)$
    $\mathbf{e}_{(b,s')} = \text{Extend}(\mathbf{n}_b, \mathbf{n}_s)$ *(extend towards sample)*
    IF $\left(\mathbf{e}_{(b,s')} \text{ AND } \text{ImproveCost}(\mathbf{e}_{(b,s')})\right)$ *(if cost can be improved)*
      $\mathcal{N} = \mathcal{N} \cup \mathbf{n}_{s'}, \quad \mathcal{E} = \mathcal{E} \cup \mathbf{e}_{(b,s')}$ *(add node to roadmap)*
      $\mathbf{e}_{(s',g')} = \text{Extend}(\mathbf{n}_{s'}, \mathbf{n}_g)$ *(extend towards goal)*
      IF $\left(\mathbf{e}_{(s',g')} \text{ AND } \text{ImproveCost}(\mathbf{e}_{(s',g')})\right)$
        $\mathcal{N} = \mathcal{N} \cup \mathbf{n}_{g'}, \quad \mathcal{E} = \mathcal{E} \cup \mathbf{e}_{(s',g')}$
        $c_{best} =$ cost-to-come at $\mathbf{n}_{g'}$ *(update best cost)*
        Prune($\mathcal{N}$) *(prune nodes with higher cost)*

Figure 4.3: Incremental probabilistic roadmap (PRM) planning pseudocode

to the state of the new node. A drawback in such an approach is that the resulting path could be far from optimal and that often times it is necessary to stabilize to non-zero velocity states which raises issues of stability. A third approach is to use optimal control and nonlinear programming for computing an optimal trajectory. The drawback of that approach is that its convergence is very sensitive to the choice of initial guess and the computation is extremely costly. An alternative approach that we rely on in this work is to use a set of simple precomputed optimal control motions called *primitives* in order to construct more complex, suboptimal trajectories that approximately achieve the local plan very efficiently.

4.3 Planning using primitives

4.3.1 Primitives

In this section we recall the formulation of primitives for motion planning of dynamical systems introduced in [22]. Assume that a control system $\mathcal{S}$ is defined on a state-space X and has a set of allowable controls inputs denoted U. Usually for mechanical systems X is the tangent or cotangent bundle, TQ or T^*Q respectively, where Q is the configuration space. Let $x \in X$ denote the state, $u \in U$ the controls.

A trajectory from time t_i to time t_f is denoted $\pi : t \in [t_i, t_f] \rightarrow (x(t), u(t))$. Planning using primitives relies on the notion of *trajectory invariance*. Assume that the Lie group G acts on X with action on x denoted $\Phi(g, x)$, where $g \in G$. An invariant trajectory is such that the dynamics of the system is G-invariant. More formally, denoting the flow (or integral curve) of the system by $\varphi : X \times \mathbb{R} \rightarrow X$ then the invariant flow from the start state $x_0 \in X$ is $\varphi(x_0, t)$ satisfies

$$\Phi_g(\varphi(x_0, t)) = \varphi(\Phi_g(x_0), t).$$

Two trajectories π_1 and π_2 are *equivalent* if they differ by a group action and a time translation, i.e. if there exists a $g \in G$ and a time T such that

$$(x_1(t), u_1(t)) = (\Phi_g(x_2(t - T)), u_2(t - T)), \forall t \in [t_{i,1}, t_{f,1}]$$

The concatenation of two trajectories requires the definition of *compatibility*. Two trajectories π_1 and π_2 are compatible, denoted $\pi_1 C \pi_2$, if there exists $g_{12} \in G$ s.t.

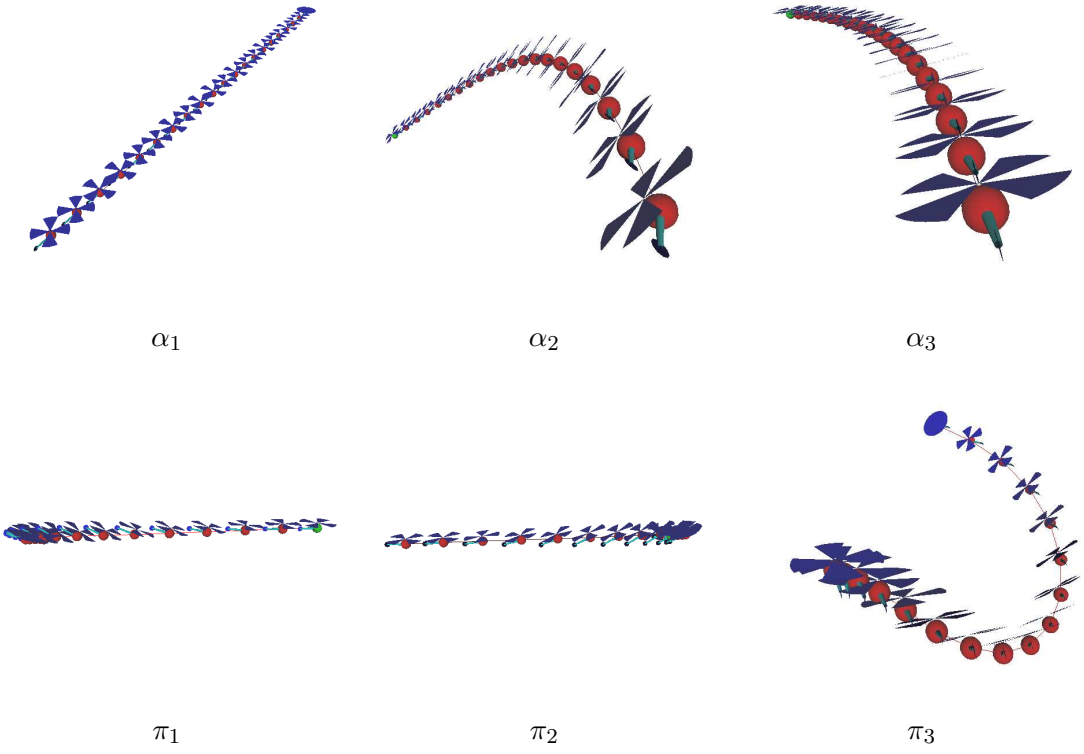


Figure 4.4: Examples of optimal helicopter primitives (computed using DMOC) invariant with respect to the action of group $G' = \mathrm{SO}(2) \times \mathbb{R}^3$. The top three trajectories are trim primitives of constant forward motions with translational velocity $v = 15$m/s and rotational velocity $\omega = 0°$/s (α_1); $v = 15$m/s and $\omega = 30°$/s (α_2); $v = 10$m/s and $\omega = 15°$/s (α_3). The bottom three trajectories are maneuvers that start from rest and achieve $v = 15$m/s and $\omega = 0°$/s (π_1); start with $v = 15$m/s and $\omega = 0°$/s and stop at rest (π_2); start from rest and achieve $v = 15$m/s and $\omega = 30°$/s (π_3). Various concatenations of these primitives are then possible, e.g. $\pi_1\alpha_1\pi_2$ or $\pi_3\alpha_2$.

$x_1(T_1) = \Phi(g_{12}, x_2(0))$ Any invariant trajectory can be regarded as a primitive. The set of all G-invariant primitives is denoted $\mathcal{P}(\mathcal{S}, G)$ and is *closed* [22]. Therefore, new more complicated primitives can be constructed by concatenating simpler ones as long as they are compatible. Fig. 4.4 shows examples of helicopter primitives.

There are two main classes of primitives useful for motion planning. They are discussed next.

4.3.1.1 Trim Primitives

Trim trajectories correspond to continuously parametrized steady-state motions. Formally, $\alpha : t \in [0, T] \rightarrow (x_\alpha(t), u_a(t))$ is a *trim primitive* if:

$$x_\alpha(t) = \Phi(\exp(t\xi_\alpha), x_\alpha(0)), u_\alpha(t) = u_\alpha, \forall t \in [0, T]$$

or if the motion is a finite flow along left invariant vector fields with constant control inputs. The set of trim primitives is denoted $\mathcal{T}(\mathcal{S}, G)$.

Once a trim primitive vector field has been identified it can be used to construct a one-parameter family of trim primitives

$$\alpha(\tau) : t \in [0, T] \rightarrow (\Phi(\exp(t\xi_\alpha), x_\alpha(0)), u_\alpha),$$

where τ is called coasting time, and the set of all such primitives defined by $\mathcal{T}_\alpha = \{\alpha(\tau), \tau \geq 0\}$.

The displacement of a trim primitive α with coasting time τ is simply $g_\alpha = \exp(\tau\xi_\alpha)$.

4.3.1.2 Maneuvers

Maneuvers are designed to efficiently switch from one steady-state motion to another. Therefore they are defined to be compatible form left and right with trim primitives. The set of maneuvers is denoted $\mathcal{M}(\mathcal{S}, G) \subseteq \mathcal{P}(\mathcal{S}, G)$. Formally, a maneuver π satisfies

$$\pi \in \mathcal{M}(\mathcal{S}, G) \Leftrightarrow \exists \alpha, \beta \in \mathcal{T}(\mathcal{S}, G) : \alpha\pi\beta \in \mathcal{P}(\mathcal{S}, G).$$

The displacement as a result of executing the maneuver is denoted g_π.

4.3.2 Computing motion plans

The motion planning problem can now be solved by finding a sequence of primitives ω consisting of trim primitives $\alpha_1, ..., \alpha_{N+1}$, with coasting times $\tau = (\tau_1, ..., \tau_{N+1})$, connected by maneuvers $\pi_1, ..., \pi_N$. The pair (primitives, coasting times) is denoted $(\omega, \tau) \in \mathcal{P}(\mathcal{S}, G)$ and is defined by

$$(\omega, \tau) = \alpha_1(\tau_1)\pi_1\alpha_2(\tau_2)\pi_2...\pi_N\alpha_{N+1}(\tau_{N+1})$$

The task is to find (ω, τ) that takes the system from an initial state $x_i \in X$ to a final goal state $x_f \in X$ or to a set $X_f \subset X$.

Controllability Although by assumption the original control system is controllable, a system that is forced to move only along primitives might not be controllable to an arbitrary state $x_f \in X$. Intuitively, this limitation arises since the trim trajectories are of particular form that could constrain the reachable space around x_f. Fortunately, the fact that the trim primitives are continuously parametrized makes it possible to consider infinitesimal motions by perturbing their coasting times near the origin. The composition of the resulting infinitesimal flows can generate new directions of motion to increase the dimension of the reachable set around x_f. Similar to the Lie algebra rank condition (LARC) for nonintegrable systems, if the number of the linearly independent iterated Lie brackets of the trim vector fields ξ_{α_i} equals the dimension of the lie algebra $\mathfrak{g}$, then the system is controllable to the configuration of x_f [22].

The total group displacement after executing trajectory $p = (\omega, \tau)$ would be

$$g_p = \left[\prod_{i=1}^{N} \exp(\tau_i \xi_{\alpha_i}) g_{\pi_1} \right] \exp(\tau_{N+1} \xi_{\alpha_{N+1}})$$

Therefore, as the authors of [22] point out, the original differential control system was transformed into a purely *kinematic system*, with the transformation being exact without any approximations.

It is reasonable to assume that the initial and final conditions are given in terms of steady states (or trim primitives) $x_\alpha(0)$ and $x_\beta(0)$ respectively, i.e. there are group elements g_0, g_f such that $x_0 = \Phi(g_0, x_\alpha(0))$ and $x_f = \Phi(g_f, x_\beta(0))$. Then computing a motion from x_0 to x_f amounts to finding a motion plan $p = (\omega, \tau)$ such that

$$g_p = g_0^{-1} g_f$$

Solving this equation is equivalent to the problem of *kinematic inversion* in robotics.

Furthermore, any cost function to be minimized can also be reexpressed in terms of cost of executing the individual primitives. The resulting optimal control problem is solved algebraically without the need of finite-difference approximation of the original control system derivatives.

4.4 Motion Planning using DMOC and Roadmaps

The main idea is to start by computing many simple short but optimal motions through DMOC optimization. These motions are guaranteed to be optimal since they are selected

as the best solution from multiple optimization runs with varying initial conditions. The trajectories are then used to build a library of primitives. A cost function minimizing motion plan can then be formulated as a kinematic optimal control problem as described in Sec. 4.3.2. A feasible solution to this problem might still be difficult to compute and far from optimal when that state-space is complex. Therefore, a global dynamic-programming approach such as a sampling-based roadmap would be appropriate and is explored in this section.

4.4.1 Trajectory Discretization

Time Discretization There are different types of DMOC discretization depending on the type of systems studied. This results into different ways of defining the state space for roadmap planning using primitives:

- The case of vehicles on Lie groups (2.4) is defined using a discrete state $(g_k, \xi_k, \mu_k) \in G \times \mathfrak{g} \times \mathfrak{g}^*$ where μ_k is computed from ξ_k using the Legendre transform. The state space is

$$X := G \times \mathfrak{g}, \quad x_k := (g_k, \xi_k).$$

- The more general case of vehicles with symmetries and nonholonomic constraints (3.24) is defined using a discrete state $((r_k, u_k, p_k), (g_k, \Omega_k, \mu_k)) \in (TM \oplus T^*M) \times G \times \mathfrak{s} \times \mathfrak{g}^*$ where p_k and μ_k are computed from u_k and ξ_k using the Legendre transform and $\xi_k = \Omega_k - \mathcal{A}(r_k + \alpha h u_k)u_k$. The minimal state is

$$X := TM \times G \times \mathfrak{s}, \quad x_k := (r_k, u_k, g_k, \Omega_k).$$

Discrete Evolution The evolution of the state in DMOC is defined using the *discrete flow*

$$\varphi_d : X \times \mathbb{N} \to X$$

$$(x_k, i) \to x_{k+i}, \quad i \geq 0$$

The discrete flow is computed recursively using an integrator. The integrator can be either *explicit*, of the general form:

$$x_k = f(x_{0:k-1}, u_{0:k-1}),$$

or *implicit*, taking the form

$$f(x_{0:k}, u_{0:k}) = 0,$$

where $x_{0:k}$ denotes the sequence of states $x_0, ..., x_k$. Normally, when first order discretizations are used, f only depends on the last state x_{k-1} and last control u_{k-1}. Higher order discretization schemes, though, result in dependency on previous states and hence, for generality, f is defined to depend on the whole trajectory and control history.

In particular, the following integrators are used for time-stepping update $x_k \to x_{k+1}$ based on the type of system:

- systems on Lie groups: (2.7)-(2.9)
- nonholonomic systems with symmetries: (3.39)

The discrete flow approximates the continuous flow $\varphi : X \times \mathbb{R} \to X$ (see Sec.4.3.1), i.e.

$$\varphi_d(x_k, i) \approx \varphi(x(kh), ih), \qquad x_k \approx x(kh)$$

Using this relationship the notions of invariance, equivalence, and compatibility defined in Sec. 4.3.1 extend trivially to DMOC trajectories when restricted to their discrete set of states.

4.4.2 Sequencing Primitives

Naive Sequencing The sampling based planners rely on a function Extend that must efficiently produce a trajectory connecting two nodes. In the primitives planning framework this amounts to first selecting a sequence of trim primitives and maneuvers. Such a selection is a combinatorial problem which can be combined with an optimization problem (usually affine) in order to determine optimal coasting times for trim primitives (Sec. 4.3.2). Although the resulting numerical problem is much easier than the original optimal control problem it still requires combinatorial search and iterative optimization. Instead, we use a simple sequencing strategy by replacing the set $\mathcal{T}_\alpha$ of continuously parametrized trim primitives along some vector field ξ_α by the discrete set $\{\alpha(\tau_{min}^i)\}_{i=1}^c$, i.e. by trim primitives of exponentially increasing times. The smallest coasting time τ_{min} is chosen so that $\|g_{\alpha(\tau_{min})}\| < d_{min}$, where $\|\cdot\| : G \to \mathbb{R}$ is distance function determining how close two configurations are and d_{min} is a minimum allowable error tolerance on achieving a motion plan. The maximum exponent c is chosen so that τ_{min}^c does not exceed the reasonable time in which we believe the while motion can be achieved. Such

an approximation is chosen since it allows the quick construction of a trim primitive with coasting time equal to any factor of τ_{min}. In essence, the optimization part of the problem is one more time replaced by a combinatorial one at the expense of slight loss of controllability.

Sequencing primitives then proceeds in the most simplistic and greedy manner:

- go through all primitives compatible with the current one
- select the one that takes the vehicle closest to goal
- keep iterating until distance to goal is less than d_{min}.

This is by no means optimal but it is very efficient, and since our roadmap has the ability to prune bad subtrees then, in practice, bad sequences are quickly replaced by better ones. The constantly decreasing upper bound on the current optimal cost guarantees that only cost improving sequences are considered.

Proper Sequencing Proper sequencing involves selecting a set of primitives and coasting times that would result in a more optimal trajectory that reaches the sampled node exactly. Such a solution can be computed using nonlinear programming since in general it has no closed-form. We choose to use the naive strategy explained above in order to avoid the extra cost of iterative optimization. In addition, the automatic cost bounding and trajectory pruning at the higher level roadmap construction assures that the resulting path is nearly optimal even when using a naive local strategy. In addition, for for certain simpler problems, it is worth exploring exact inverse kinematics solutions.

4.4.3 Examples

Helicopter We apply the roadmap planning with primitives to the helicopter model introduced in Sec. 2.10.2. One can view this model as an idealized version of the systems used in [22] with the main difference that we have chosen an exact form of control input vector fields instead of using a control input transformation determined empirically. While not as closely realistic as the form used in [22] our formulation preserves the high underactuation of the system and allows us to show how to compute requirements for trim primitives in closed form.

The helicopter dynamics is invariant under transformations by the group $G' = \mathrm{SO}(2) \times \mathbb{R}^3$, i.e. rotations around z-axis and translations. The primitives that we use are in the form of the vector field $\xi_\alpha \in \mathfrak{se}(3)$

$$\xi_\alpha = \begin{bmatrix} 0 & -\omega_z & 0 & v_x \\ \omega_z & 0 & 0 & v_y \\ 0 & 0 & 0 & v_z \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

where the angular velocity (around the body-fixed z-axis) ω_z and the linear velocity $v = (v_x, v_y, v_z) \in \mathbb{R}^3$ are chosen in a special way to preserve the dynamics invariance. Based on our model, the conditions for invariance can be computed as:

rotor blade angles: $\gamma_p = 0, \gamma_r = 0$

rudder input: $u_\psi = 0$

The remaining variables satisfy:

$$\begin{aligned} v_y\omega_z &= -g\sin\theta \\ -v_x\omega_z &= g\cos\theta\sin\phi \\ u_c &= g\cos\theta\cos\phi, \end{aligned}$$

where $g = 9.81$ from gravity, θ and ϕ are the *pitch* and the *roll* of the helicopter body computed from its rotation matrix $R \in SO(3)$. These conditions ensure a constant velocity, i.e. $\dot{\xi} = 0$ along the whole trajectory.

A general primitive with constant velocity in the form ξ_α can be then constructed by setting:

$$\theta = -\arcsin\left(\frac{v_y\omega_z}{g}\right), \qquad \text{subject to} \quad |v_y\omega_z| < g,$$

$$\phi = -\arcsin\left(\frac{v_x\omega_z}{g\cos\theta}\right), \qquad \text{subject to} \quad |v_x\omega_z| < g\cos\theta$$

$$u_c = g\cos\theta\cos\phi.$$

In other words once v and ω_z are chosen then invariance is maintained by requiring that the starting configuration of the primitive has pitch θ and roll ϕ (and hence any configuration along the primitive since these two variables would remain invariant), that

the blade angles and the rudder input be set to zero, while the collective u_c is held constant as defined above.

For example, a trim primitive corresponding to sharp right turn with velocities $v = (15, 0, 0)$ in m/s and $\omega_z = 30°/s$ requires that $\theta = 0$, $\phi = -53.19°$, $u_c = 5.89$ N. A less aggressive turn with $v_x = 10$ m/s and $\omega_z = 15°$/s requires $\phi = -15.48°$ and $u_c = 9.45$ N. See Fig. 4.4 for examples of some of these primitives.

Our analytical derivation of the invariance conditions is consistent with the experimentally determined primitives used in [22] for real helicopter. Similar to this work we can also include non-zero sideways velocity v_y and hence non-zero pitch θ in the primitives construction but that is not required since our simple model does not account for air resistance and constant forward velocity can be maintained without pitching forward. The vertical component v_z can be chosen freely within the vehicle envelope and does not affect invariance.

Fig. 4.5 shows the constructed roadmap and gives a closer view of the resulting least cost trajectory. One of the nice feature of the sampling-based methods is the ability to quickly explore the state space and find any solution, which initially is far from optimal. The algorithm can then quickly improve that solution as the roadmap covers the environment more densely.

Car In the car example (with dynamics described in Sec. 3.4), trim primitives consist of straight forward and backward motions of constant velocity; as well as left and right, forward and reverse turns with constant steering angle and constant velocity. Maneuvers are solutions to optimal control problems with boundary conditions compatible with a

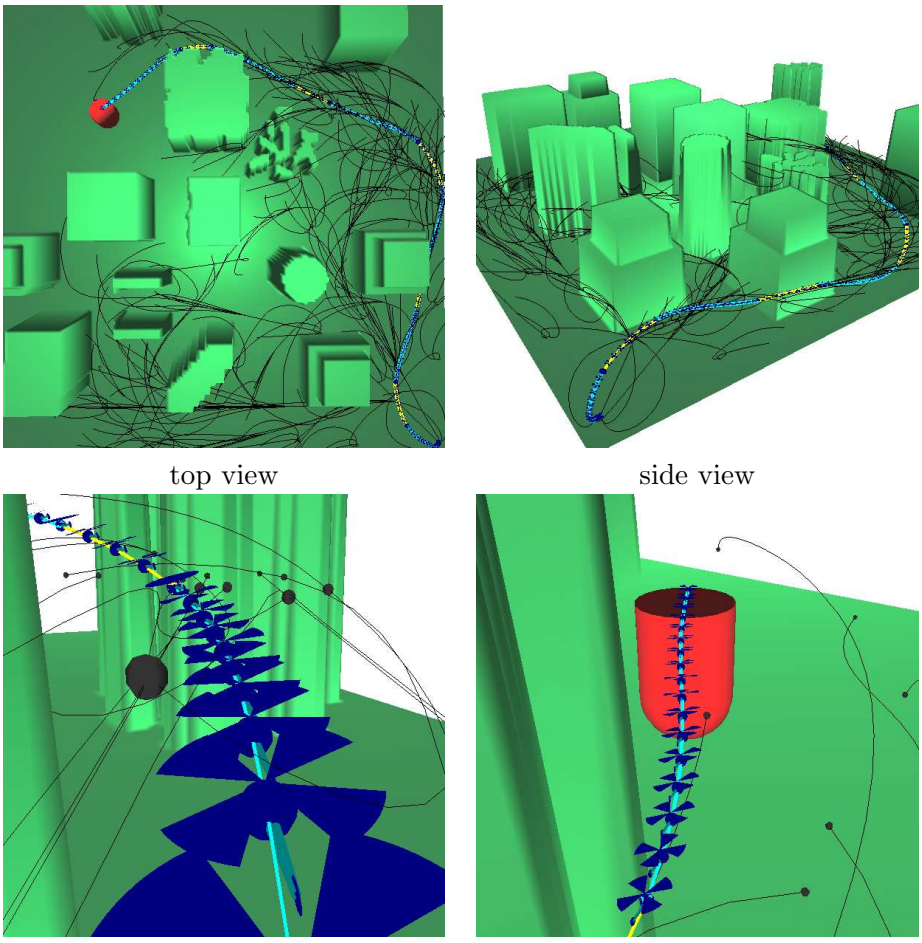


Figure 4.5: An example of planning using DMOC primitives and an incremental roadmap global search. A helicopter must traverse a cluttered urban environment and arrive inside the cylinder with minimum fuel. The different views show the resulting roadmap and least cost trajectory to goal after 1 second of computation.

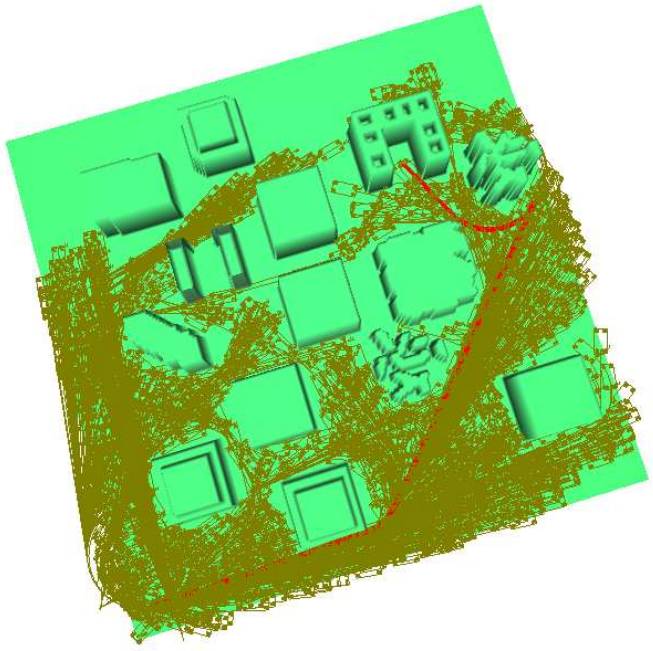


Figure 4.6: Incremental PRM for a car-like robot. The goal is to compute a trajectory that parks the car inside the Π-shaped parking structure.

chosen pair of trim primitives. Fig. 4.6 shows an example of optimal motion of a car into a parking structure. For simplicity, in this example the distance metric is the standard euclidean metric on the (x, y)-position subspace.

4.5 Extensions

In this section we consider several important extensions to the general motion planning framework. For simplicity, the simulation results are based on vehicles with simple unconstrained dynamics subject to velocity limits. Since the roadmap method is not specific to any particular dynamics model the developed algorithms are applicable to vehicles with realistic dynamics as well.

4.5.1 Moving Goal State

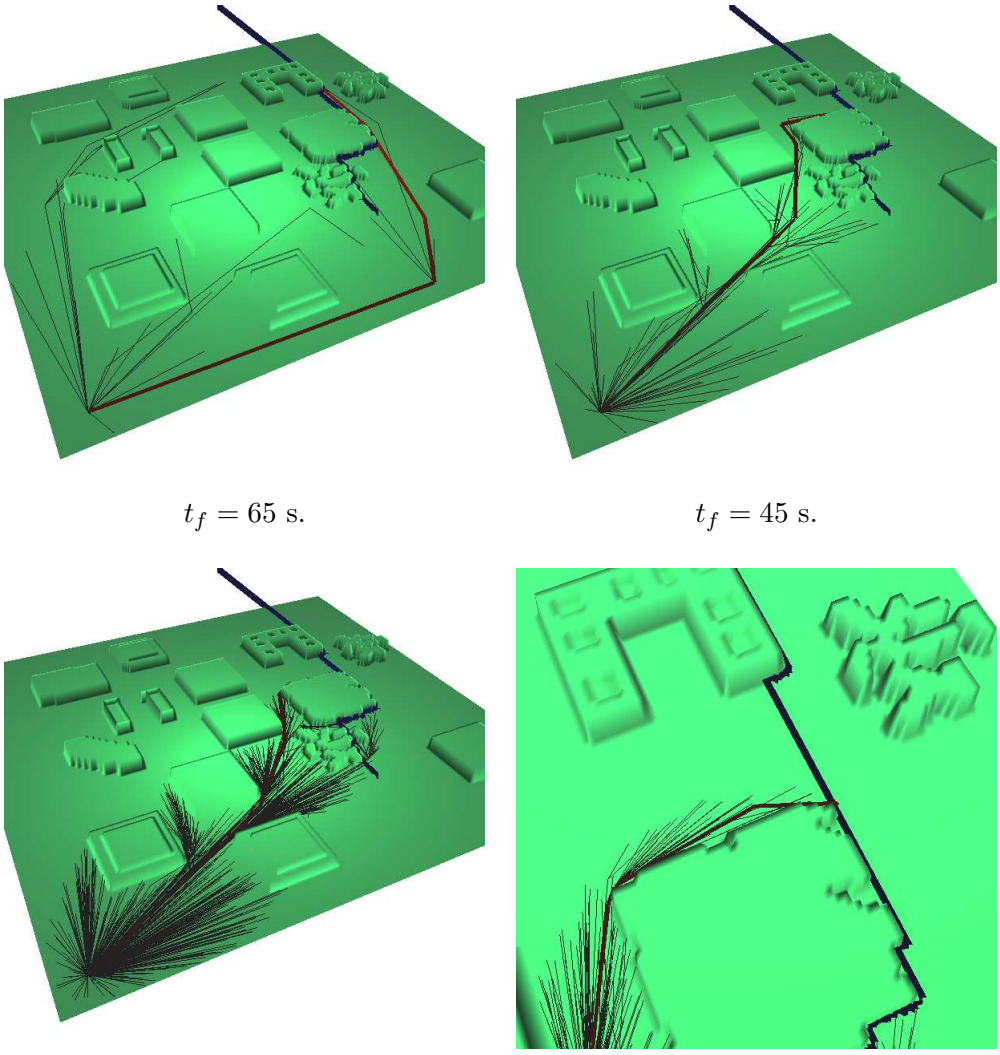
Assume that the goal state is moving in time and the vehicle has complete knowledge of its dynamical model. For clarity denote the goal state at the k-th time stage by $y_k \in X$. The task is still the same: to reach the goal state with minimum cost. For computational purposes, the trajectory of the goal can be precomputed up to the time horizon of K stages as $y_{0:K}$.

The roadmap algorithm can be used with slight modification when extending the roadmap towards the goal. Let the time and state of the roadmap node to be extended by denoted by (t_k, x_k). In order to connect this node to the goal trajectory one can simply sample an integer l, $k < l < K$, and extend towards the pair (t_l, y_l). Depending on the cost function, it might be possible to choose a heuristic for better sampling. For example, if the cost function is minimum time, then l can be chosen such that $(l - k)$ time steps is a lower bound on time required to move from x_l to y_l, and if such extension fails to continue with increasing l until the minimum l is found. Fig. 4.7 shows results for reaching the goal in minimum time using a point-mass vehicle with limits on the velocity.

4.5.2 Optimal Coverage

Consider the case of a vehicle with circular sensing area of fixed radius. Obstacles in the environment restrict the sensor coverage, e.g. the sensor such as an omnidirectional camera covers only the physically visible area. The task is to compute a vehicle path that maximizes the union of all area covered within some prescribed time horizon K.

This is a very difficult, computationally intractable problem. It has been shown that it can be reduced to a high-dimensional traveling salesman problem (TSP) which itself is



$t_f = 65$ s. $t_f = 45$ s.

$t_f = 39$ s. closer view

Figure 4.7: Finding a time optimal trajectory to reach a time-varying goal state with *known* dynamics. The goal state is the lower right at time $t = 0$ and starts to move north going around obstacles–its projected trajectory is drawn leaving the environment at time $t = 100$ s. Consecutive stages of the roadmap expansion show how the current best solution improves (the cost function is the time t_f of reaching the goal).

NP-hard and at best has exponential complexity if solved through dynamic programming. Our goal is to compute a good (but not optimal) solution through the roadmap framework (which in essence combines dynamic programming with randomization).

Applying the roadmap approach is straightforward. The only modification is to the cost function. The heuristic cost used during expansion is based on assuming there are no obstacles and can be computed quickly. The actual cost is then computed using standard computational geometry methods for polygon area (in 2-D) and polyhedra volume(in 3-D) and intersections.

In particular the cost used is the total area covered divided by the total trajectory time, i.e. the *coverage rate*. Denote the workspace to be covered by $\mathcal{W}$. Depending on the sensing application $\mathcal{W} \subset \mathbb{R}^2$ or $\mathcal{W} \subset \mathbb{R}^3$. Let the vehicle cover a region $\mathcal{A}(x_k) \subset \mathcal{W}$ while it is at state x_k at time t_k. The cost function at x_k is then

$$\text{cost}(x_{0:k}) = \frac{\text{area}(\mathcal{A}(x_0) \cup ... \cup \mathcal{A}(x_k))}{t_k - t_0}$$

It is evident that the change in the cost from one node to the next depends on the whole trajectory and not only on the new segment. Therefore, in the standard planning framework a system state at time t_k must be not the state x_k but the whole trajectory up to that time $x_{0:k}$. Only then the cost function is a distance metric satisfying the triangle inequality and useful for dynamic programming in a graph structure. Since constructing and using such states is impractical we only consider a tree structure in which nodes can still be described by the original states x_k given that the cost is recomputed after the

trajectory is extended. This can be done efficiently by storing the region covered by the whole trajectory at each node. See Fig. 4.8.

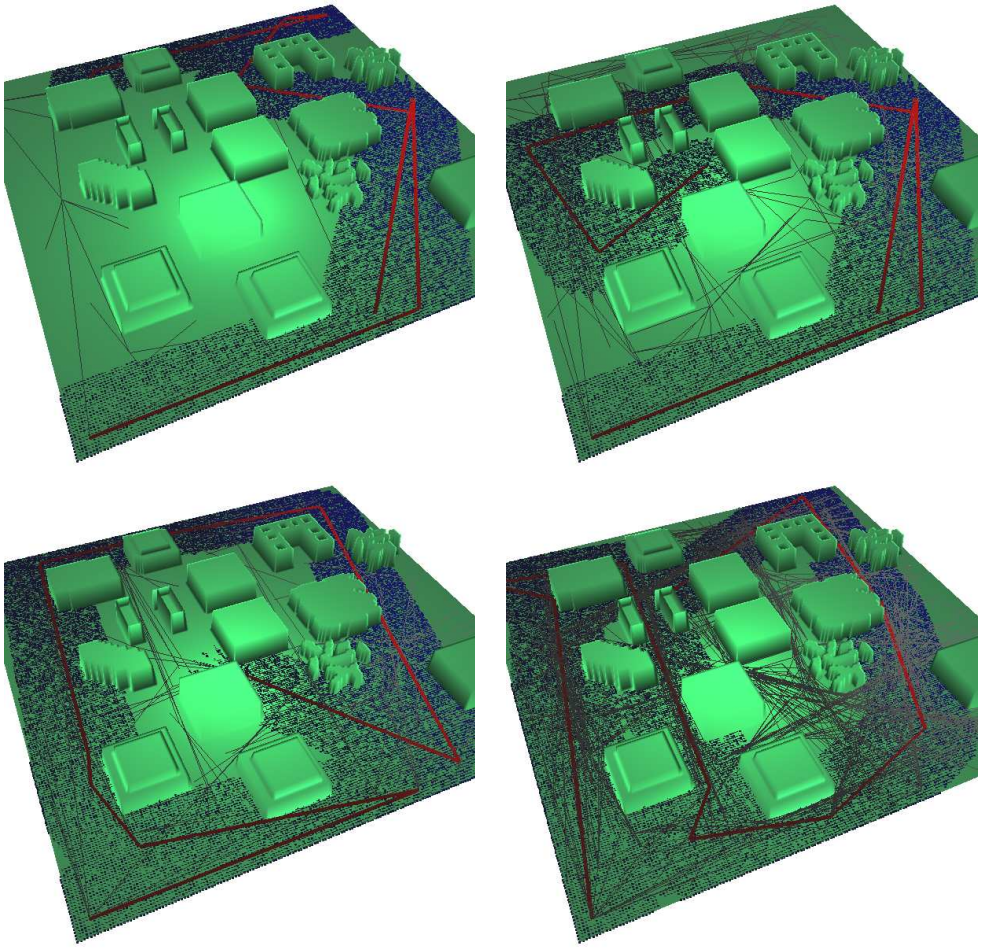


Figure 4.8: A vehicle that can sense everything outside the obstacles up to a fixed radius. Maximizing the total area searched (shaded) in fixed time horizon of of 100s. Consecutive stages of the roadmap expansion show how the current best solution improves (the coverage cost is the total shaded area shown).

4.5.3 Uncertain Target Detection with Multiple Vehicles

Imagine that some interesting phenomenon is taking place in the environment. Assume that we have multiple vehicles equipped with sensors that can make noisy measurements of this phenomenon. For example, this phenomenon could be a single object of interest

that is moving in a cluttered environment with some uncertainty, e.g. we might know that it is heading north with speed around 5 m/s and that it might slow down, avoid obstacles, speed up, etc... In this example we consider the problem of vehicle deployment in order to maximize the information gained about the target.

In this section we only briefly describe some experimental results without describing the probabilistic framework in detail. For instance, Fig. 4.9 gives a few consecutive snapshots of the vehicle and target states during deployment.

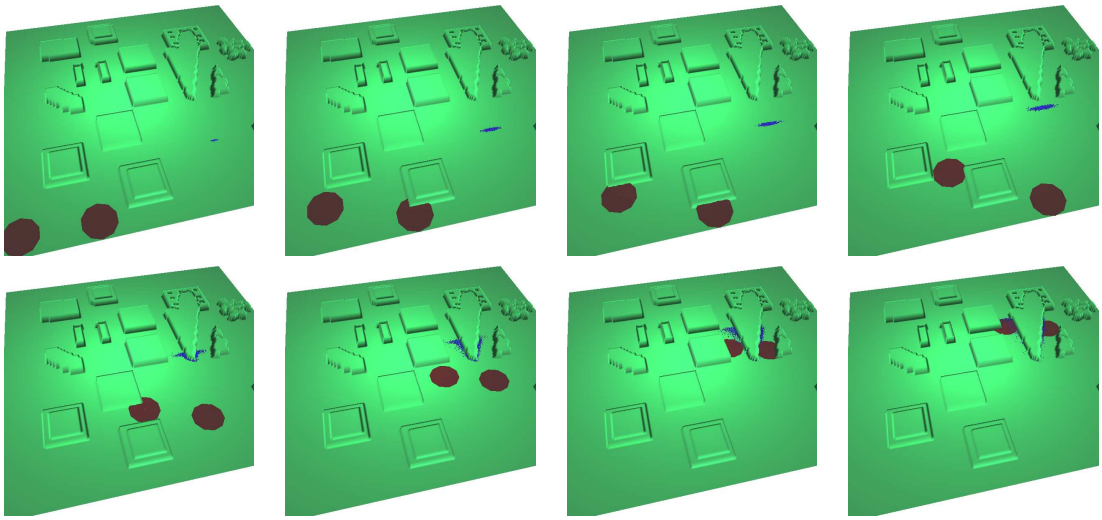


Figure 4.9: An example scenario: a target with uncertain dynamics is moving north avoiding obstacles. Its possible motion is represented by a finite set of particle trajectories. These trajectories are used to simulate future measurements taken by the vehicles. Two vehicles with circular sensing field-of-view (with initial position in the lower left corner) are deployed in order to maximize the probability of detecting the target.

In order to solve the problem we use the RRT exploration algorithm (Fig. 4.2). A node contains the states of all vehicles and a probabilistic estimate of the target state computed from sensor measurements. Since there is no preexisting target motion the measurements must be simulated. Since the target is moving whole trajectories of measurements must be simulated. Furthermore, we represent the uncertainty in target motion by generating

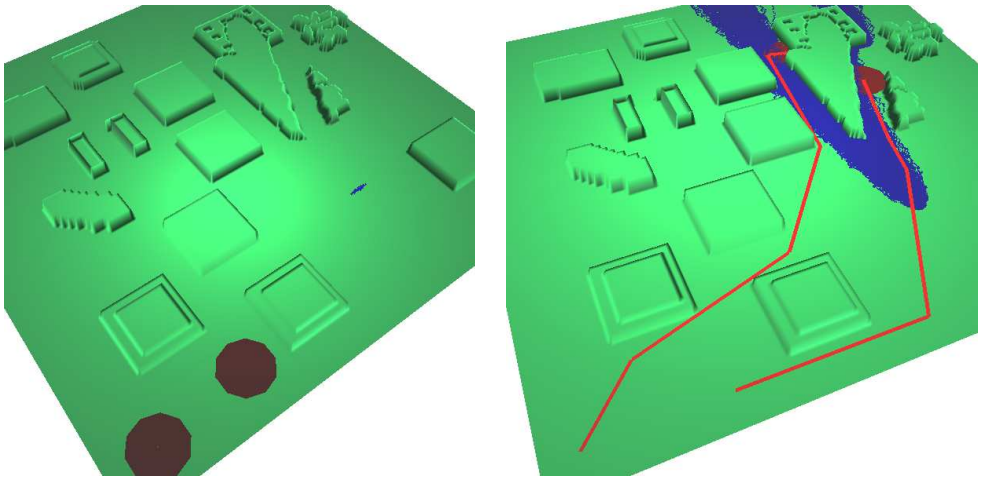
multiple target trajectories (and hence use multiple target trajectories measurements) in a Monte-Carlo fashion. Consequently, any measure of uncertainty or information is computed by taking its expected value over these possible target motions. Therefore, each node stores a set of N probabilistic filters (where N is the number of Monte-Carlo target trajectories) each filter updated by assuming that the true target motion is a particular trajectory from that set. In our implementation we use Kalman filters of constant velocity and white noise acceleration based on relative position measurements assuming Gaussian error sensor model.

A proper implementation would also involve importance resampling of the Kalman filter states in order to avoid estimates with very low probability. We leave that for future work.

The distance metric used for planning is a combination of time and the inverse of the expected variance (i.e. the norm of the filter covariance) of the target state estimate. The tree expansion is designed to achieve global exploration while also attempting to connect with areas with high expected target probability mass. This is achieved by drawing a small fraction of the tree samples from the actual simulated measurement particle set treated as a probability distribution. A near-optimal solution is depicted on Fig. 4.10.

4.6 Conclusion

This section presented a way to combine optimal control and global state space search in order to solve the problem of deploying one or multiple vehicles to achieve a certain



initial conditions least cost vehicles trajectories

Figure 4.10: A single target with time-varying uncertain dynamics (with distribution in time represented by particle trajectories) must be detection with maximum probability and in minimum time. Two vehicles with circular sensing field-of-view and limited velocity are deployed based on sampling-based roadmap computation.

goal such as reaching a goal state or minimizing a task specific cost function. Our approach is based on combining DMOC with motion planning methods developed in the robotics community and extending the framework to interesting problems involving real dynamics, uncertainty, and complex cost metrics. Some of the resulting algorithms (such as maximizing coverage from Sec. 4.5.2) are not guaranteed to be near optimal because of the exploding dimension of the search space inherent to the specific problem. Yet, even in such difficult problems employing a roadmap representation build from optimal primitives proves useful since it encodes many possible motions in a compact and efficient manner.

Proper treatment of uncertainty in our framework is still an open problem. An example of deployment with uncertain target dynamics and simple noisy sensing was given in Sec. 4.5.3. Some initial progress has been made in incorporating uncertainty in more realistic dynamical models in order to achieve the task at hand while minimizing the chance

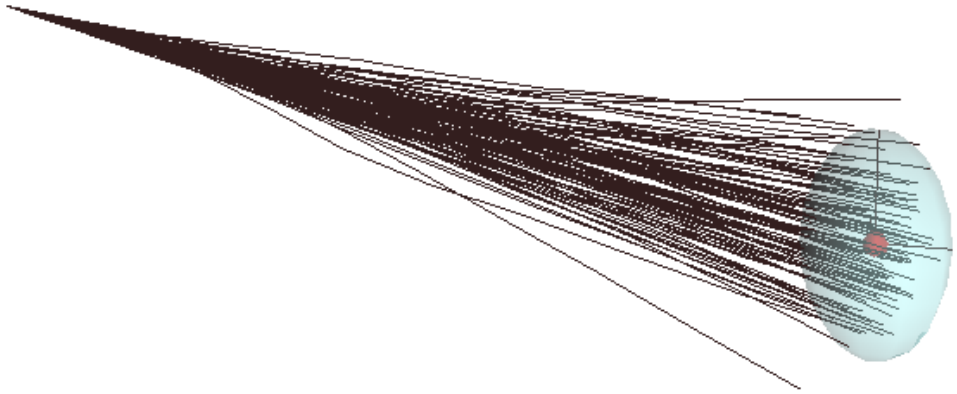
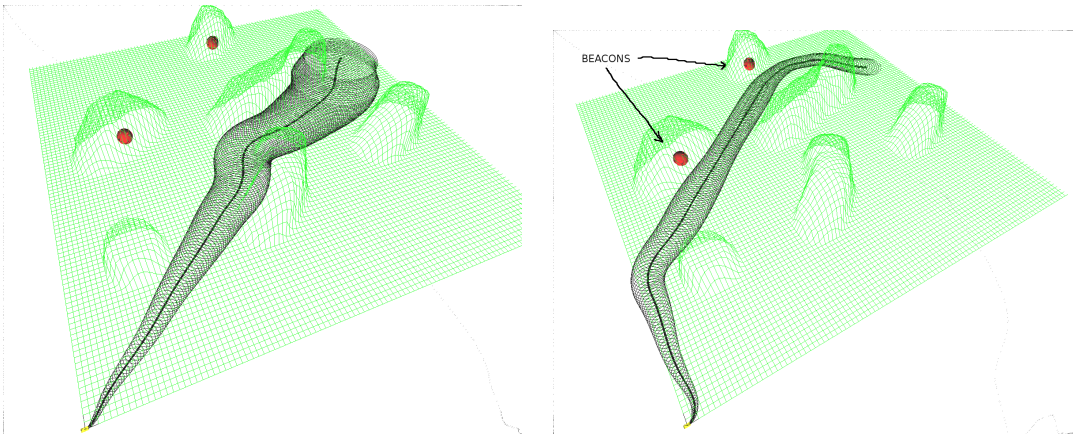


Figure 4.11: Uncertainty in the helicopter position. Different motions resulting from uncertain external disturbance forces, e.g. arising from wind. The trajectories and the resulting position uncertainty ellipsoid are generated by sampling from a Gaussian external force error model.



shortest path safest path

Figure 4.12: A helicopter with uncertain dynamics (as shown in Fig. 4.11) must fly to a goal location across a digital terrain. Its position estimate uncertainty norm is drawn as a tube along the path. It has no GPS and can only measure its own position by observing static beacons in the environment. The path on the left shows the shortest path to the goal but clearly demonstrates that the resulting uncertainty would increase the chance of collision with the terrain. The path on the right is aimed at minimizing the chance of collision by minimizing the position uncertainty along the path in the vicinity of obstacles.

of violating certain constraints such as colliding with obstacles. An example of ongoing work with a helicopter model (described in Sec. 2.10.2) with uncertainty in control inputs and external disturbances is depicted in Fig. 4.11. A computed scenario inspired by the related work [49] is described in Fig. 4.12.

The main challenges in incorporating uncertainty lie in choosing a numerical representation that would result in a balance between accurate noise propagation and efficient computation suitable for the kinds of deployment problems discussed in this section. Ultimately, one needs to examine and deal with sources of uncertainty not only in the vehicle sensing and actuation but also in connectivity and communication that are extremely important in multi-vehicle and distribution applications.

Chapter 5

Homotopy Continuation of Motion Planning Constraints

5.1 Introduction

The goal of this chapter is to apply homotopy continuation methods to motion planning of mechanical systems such as autonomous robots that are subject to complex constraints. Such constraints can arise either from the environment, such as obstacles; from the dynamics of the system, such as nonholonomic constraints or underactuation; or from the given task, such as maintaining visibility to a target of interest during motion. The main idea behind the homotopy methods that we propose is to initially solve an easier (usually trivial) problem and then continuously deform it into the original problem arriving at the true solution. This deformation could have physical meaning such as shrinking or smoothing of obstacles, or gradually adding/removing degrees of freedom of motion, or could be a based on nonphysical numerical homotopy such as convexification of the equality relation describing the constraint. The advantage of the homotopy approach is that one can start with an initial solution that is easy to compute and repeatedly solve a slightly more complex problem using the solution of the previous one as an initial

guess until the original problem is solved. If slight variations of the problem result in slight variations of the solutions, then the method offers a stable and efficient way to solve otherwise complicated problems. The theory of homotopy continuation establishes conditions under which such deformation is possible and provides numerical methods for tracing a solution.

The problems that we study involve constraints that are described by algebraic equalities or inequalities. Such relations are derived, for example, from the discretization of the systems dynamics or from the approximation of obstacle boundaries using algebraic sets. The homotopy methods described here apply to motion planning methods that are based on such constraint representation. Optimal control problems such as minimizing the control effort while satisfying boundary conditions are our main application. Nevertheless, the homotopy approach applies to other control problems such as point stabilization or trajectory tracking as well.

Next we present the general concept behind homotopy and describe two standard numerical continuation methods that apply to systems evolving on continuous configuration spaces. Then we give several applications: relaxation of environment obstacles through physical and non-physical deformations; smoothing of rough terrain to relax path planning constraints for legged robots; relaxing the underactuation of simple control systems.

5.2 Homotopy Continuation

We recall the basic formulation of the homotopy method, following [1], applied to the problem of finding a solution to the N nonlinear equations in N unknowns

$$F(x) = 0,$$

where $F : \mathbb{R}^N \to \mathbb{R}^N$ is a smooth map. Without prior knowledge of the problem a standard root-finding technique might fail because of poorly chosen initial guess. This can be avoided by using a *homotopy* (or deformation) $H : \mathbb{R}^{N+1} \to \mathbb{R}^N$ defined by

$$H(x, 1) = G(x), \quad H(x, 0) = F(x),$$

where the smooth map $G : \mathbb{R}^N \to \mathbb{R}^N$ has known or easy to compute solution. For example, a commonly used homotopy is

$$H(x, \lambda) = \lambda G(x) + (1 - \lambda)F(x).$$

The goal is to trace the implicitly defined curve $c(s) \in H^{-1}(0)$ from the initial solution $(x_1, 1)$ to the actual solution $(\bar{x}, 0)$.

If zero is a *regular value* of H then such curve exists and is diffeomorphic to the real line or the circle. This is equivalent to having the Jacobian DH have full rank for all values of x and λ. The curve $c(s)$ is then a submanifold of dimension 1, where s is usually arc-length parameter.

Bifurcation points can occur whenever the Jacobian loses rank. The set $H^{-1}(0)$ could then have isolated points and branches that need to be handled in special way in order to reach $(\bar{x}, 0)$. Such abnormal conditions can be detected whenever the determinant of the *augmented Jacobian*

$$\det \begin{bmatrix} DH(c(s)) \\ \dot{c}(s)^T \end{bmatrix}$$

becomes zero or changes sign.

The goal solution $(\bar{x}, 0)$ can be reached from the initial solution $(x_1, 1)$ using various methods. The simplest way is to gradually change the value of λ from 1 to 0 using small decrements and solve the resulting problems. This is called *embedding*. Often λ is not the most appropriate parameter for tracing the solution because it might require very small increments to guarantee convergence or the solution curve might have turning points past which embedding cannot continue. In such cases it is more natural to use the arclength parametrization of the combined curve (x, λ). The family of continuous methods based on such parametrization are called *Predictor-Corrector* (PC) methods.

5.2.1 Embedding Method

The embedding method follows the general scheme

Embedding Algorithm

$$\begin{aligned}
&\text{Select} \begin{cases} x_1 \in \mathbb{R}^N \text{ such that } H(x_1, 1) = 0, \\ \text{integer } m \end{cases} \\
&x = x_1, \quad \Delta\lambda = 1/m \\
&\text{for } \lambda = 1, 1 - \Delta\lambda, ..., \Delta\lambda, 0 \\
&\quad \text{solve } H(x, \lambda) = 0 \\
&\text{end}
\end{aligned} \tag{5.1}$$

5.2.2 Predictor-Corrector Method

The PC method is formulated as an initial value problem of integration along the curve $c(s)$ from $\lambda = 1$ to $\lambda = 0$. The problem is obtained, following [1], by differentiating $H(c(s)) = 0$ with respect to s, so that the following conditions are satisfied

$$DH(c)\dot{c} = 0, \quad \|\dot{c}\| = 1, \quad c(0) = (x_1, 1).$$

These sets of equations can be transformed into the ODE

$$c(0) = (x_1, 1), \quad \dot{c} = t(c),$$

that can be integrated until $\lambda = 0$ is reached. If DH is nonsingular then the *tangent vector* $t : \mathbb{R}^{N+1} \to \mathbb{R}^{N+1}$ can be easily obtained by QR decomposition of DH since if

$$DH^T = Q \begin{bmatrix} R \\ 0^T \end{bmatrix},$$

then since $DHt = 0$, we have

$$\left[DH^T, t\right] = Q \begin{bmatrix} R & 0 \\ 0^T & 1 \end{bmatrix},$$

and t can be taken as the last row of the orthonormal matrix Q.

The PC method then follows the general scheme:

PC Algorithm

Select $x_1 \in \mathbb{R}^N$ such that $H(x_1, 1) = 0$,

$u = (x_1, 1)$

while $\lambda \neq 0$

  determine stepsize h

  $u = u + ht(u)$ *(prediction)*

  solve $H(u) = 0$ *(correction)*

end (5.2)

Several issues determine the success of the PC method: selecting the right stepsize h; making sure that t does not change direction accomplished by monitoring the determinant of the augmented Jacobian; handling singularities and resulting bifurcations; implementing an efficient corrector solver. Here we have given only the very basic notion behind the method without any of the intricate details. We refer the reader to [1] for further information.

5.3 Applications

We apply the homotopy techniques to several optimal control problems. The solution of these problems can be obtained either by direct methods based on sequential quadratic programming (SQP) or indirect methods based on root-finding. In either case we apply the embedding algorithm (5.1) to the resulting system of equations to be solved. We have some preliminary results with PC methods (5.2) as well but we postpone them for future publications.

The problems involving the constraints described below are difficult to solve without a good initial guess. Finding a good initial guess is almost as hard as the original problem which motivates the use of homotopy continuation.

5.3.1 Obstacle Constraints

Assume that we are given a mechanical system on a configuration space with complex inaccessible regions induced, for example, by physical obstacles in the environment that must be avoided. If the environment is cluttered with obstacles with complex boundaries

it is often easier to find a solution by relaxing the obstacle avoidance requirement. This can be achieved in several ways that we discuss next.

In the following subsections assume that the configuration space of the system is Q, and a configuration is denoted $q \in Q$.

5.3.1.1 Distance Function

Obstacle avoidance is measured by the distance between the two closest points from the set of points occupied by the mechanical system and the set of points defining the obstacle. Assume that the mechanical system occupies a region $\mathcal{A}(q) \subset Q$ that is a function of its configuration q. The obstacles in the environment are denoted $\mathcal{O}_i \subset Q$ and the set of all obstacles is $\mathcal{O} = \{\mathcal{O}_i\}$. Assume that we are given a *distance function* $\rho : Q \times \mathcal{O} \to \mathbb{R}$ that computes the distance

$$\rho(q, \mathcal{O}_i)$$

between the vehicle at configuration q and obstacle $\mathcal{O}_i$. By convention, the distance is positive when the vehicle and the obstacle are not colliding and negative otherwise.

The algebraic constraint enforcing obstacle avoidance is then

$$\rho(q, \mathcal{O}_i) > 0$$

If the obstacle or the vehicle have complex nonsmooth boundary then the distance function would also be nonsmooth and would affect the convergence of nearby trajectories especially if gradient-based methods such as optimal control or potential field navigation

are used to compute a motion plan. This can be remedied by introducing the following homotopy

$$\rho^{\lambda}(q, \mathcal{O}_i) = \lambda \|q - q_1\| + (1 - \lambda)\rho(q, \mathcal{O}_i),$$

where $q_1 \in Q$ is an appropriately chosen configuration. For example, one physically meaningful choice for q_1 is the centroid of the obstacle. In this case, ρ^{λ} convexifies ρ by enclosing the obstacle in a ball that smooths its boundary while gradually growing it back to its original form as λ goes from 1 to 0.

Example: Helicopter in a digital elevation map Fig. 2.5 shows the trajectory of a simple helicopter that must fly optimally in a digital terrain map. The trajectory was computed using the distance function homotopy with q_1 chosen to lie approximately at the center of the tall mountain in the middle.

5.3.1.2 Algebraic Obstacles

It is common to represent objects as the intersection of half-spaces defined by smooth algebraic hypersurfaces in Q. For example, registration or parametric shape fitting of sensor data "point clouds" in 3-D are standard methods for producing such object representation. When the surfaces are planes the obstacle regions are polytopes, i.e. polygons in 2-D, polyhedra in 3-D, etc...

Example: Disk Obstacle A simple example is when both the vehicle and the obstacle are disks in $\mathbb{R}^2$. Let the vehicle radius be r and denote the obstacle by $\mathcal{O}_d := \{q_d, r_d\}$, where $q_d \in \mathbb{R}^2$ is its center and r_d its radius. The obstacle function is

$$\rho_d(q, \mathcal{O}_d) = \|q - q_d\| - r_d - r,$$

so that $\rho_d(q, \mathcal{O}_d) > 0$ and a homotopy that shrinks the disk can be defined by

$$\rho_d^\lambda(q, \mathcal{O}_d) = \|q - q_c\| - (1 - \lambda)r_d - r$$

Example: Polygonal Obstacle Polygonal obstacles can be treated in a similar manner. Let $\mathcal{O}_p := \{q_i\}_{i=0}^k$ denote a polygonal obstacle with $k + 1$ vertices $q_0, ..., q_k \in \mathbb{R}^2$. Let $\rho_p(q, \mathcal{O}_p)$ be the distance function between the vehicle at configuration q and polygon $\mathcal{O}_p$. If the center of mass of the polygon is q_p then a homotopy can be constructed by shrinking the polygon towards its center of mass through

$$\rho_p^\lambda(q, \mathcal{O}_p) = \rho_p(q, \mathcal{O}_p^\lambda), \quad \mathcal{O}_p^\lambda := \{q_p + (1 - \lambda)(q_i - q_p)\}_{i=0}^k$$

The disk and polygonal homotopies are illustrated in motion planning for a car with simple dynamics. Fig. 5.1 shows different stages of the obstacle growth and its affect on the current car trajectory. In the absence of obstacles it is relatively easy to compute an initial path, which can then by quickly deformed to account for the growing obstacles.

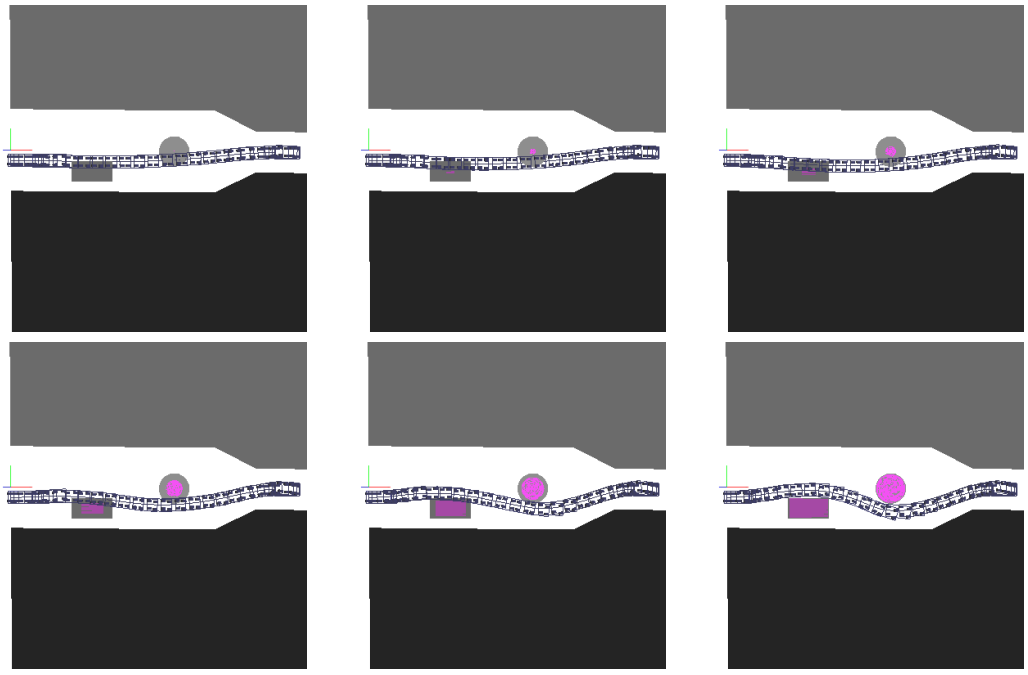


Figure 5.1: A car traveling in a tunnel. The initial trajectory is quickly computed by shrinking the obstacles to points. The obstacles are then grown and the trajectory repeatedly modified.

5.3.1.3 Rough Terrain

Assume that we have a robotic dog (Fig. 5.2) that must optimally traverse a very rough terrain. One approach is to represent the dog trajectory as a sequence of discrete poses each of which is statically stable in the sense that the center of mass of the dog projects vertically inside the support triangle formed by the set of three feet touching the ground while the fourth one is lifted to move forward. At each pose the dog kinematics must conform to a statically stable touchdown with one leg and statically stable take-off with another leg, in addition to keeping contact with the ground with the remaining legs. For example, Fig. 5.3 shows a few poses along a computed path the traverses a sample terrain. Additionally, the trajectory of the dog must end close to a particular point on the other side of the terrain. The problem is setup as a constrained optimization where

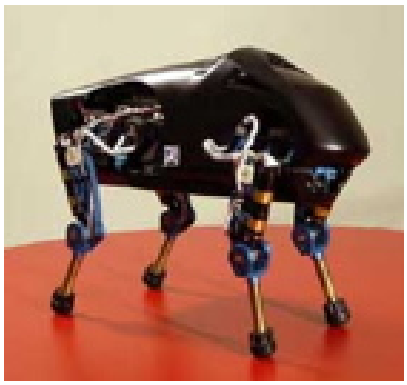


Figure 5.2: LittleDog manufactured by Boston Dynamics Inc.. The robot is used in the DARPA "Learning Locomotion" program with the goal to enable an unmanned vehicle to successfully traverse challenging terrains. One such terrain digitized from a real terrain patch is shown on Fig. 5.3. We use it as one of our test environments.

the constraints describe the motion and the objective function is the distance to that goal point.

The homotopy presented next is designed to increase the chances of successful traversal of rough terrains. It is motivated by the lack of a robust way to find an initial trajectory that satisfies the stability constraints and makes progress towards the goal. The idea is to smooth the terrain to a surface that is easy to cross and then to gradually deform that surface into the original terrain with the hope that the trajectory can be corrected to account for the increased difficulty of motion. Based on our experiments this can be successfully accomplished, or whenever failure occurs, another easy to compute starting guess can be tried until the task is achieved successfully.

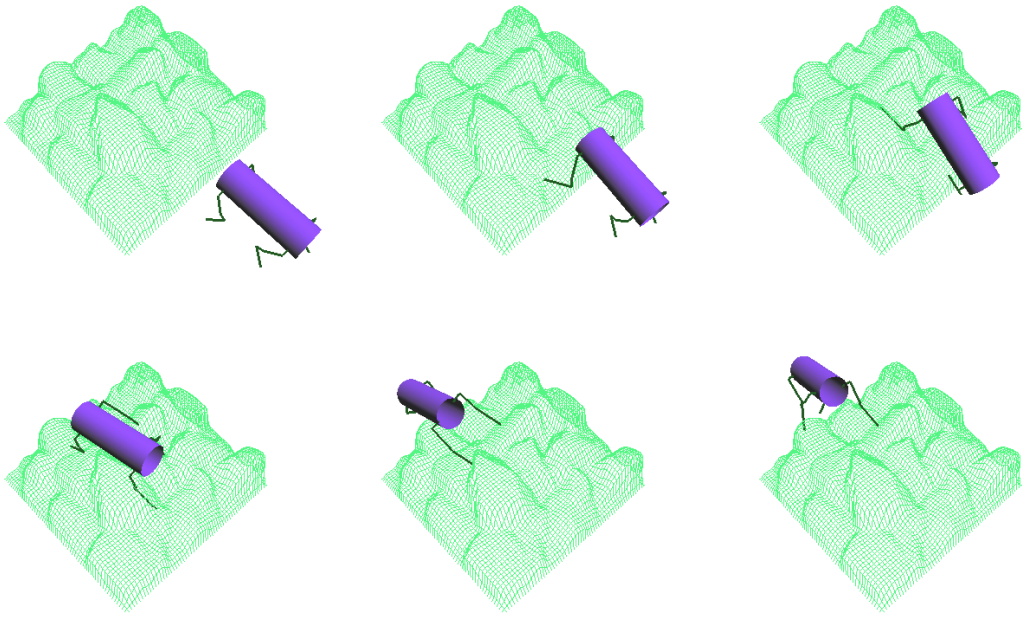


Figure 5.3: LittleDog on a digital terrain corresponding to a real laboratory terrain patch. The graphs show a few poses along an optimal path that successfully traverses the terrain.

Assume that the terrain height is described by the map $z : \mathbb{R}^2 \to \mathbb{R}$, i.e. the coordinates of each point on the terrain are $(x, y, z(x, y))$ for $(x, y) \in \mathbb{R}^2$. One way to smooth this surface is to use Gaussian filtering. Define the following homotopy

$$z^\lambda(x, y) = \frac{1}{\eta} \sum_{i=-m}^{m} \sum_{j=-m}^{m} G(\lambda\sigma, i\Delta, j\Delta) z(x + i\Delta, y + j\Delta), \tag{5.3}$$

where the scalar Δ is the terrain map resolution, σ is the Gaussian filter standard deviation, $m = \text{round}(2\lambda\sigma/\Delta)$, and

$$G(\sigma, v, w) = \begin{cases} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{v^2+w^2}{2\sigma^2}}, & \text{for } \sigma \neq 0; \\ 1, & \text{for } \sigma = 0; \end{cases}, \quad \eta = \sum_{i=-m}^{m} \sum_{j=-m}^{m} G(\lambda\sigma, i\Delta, j\Delta).$$

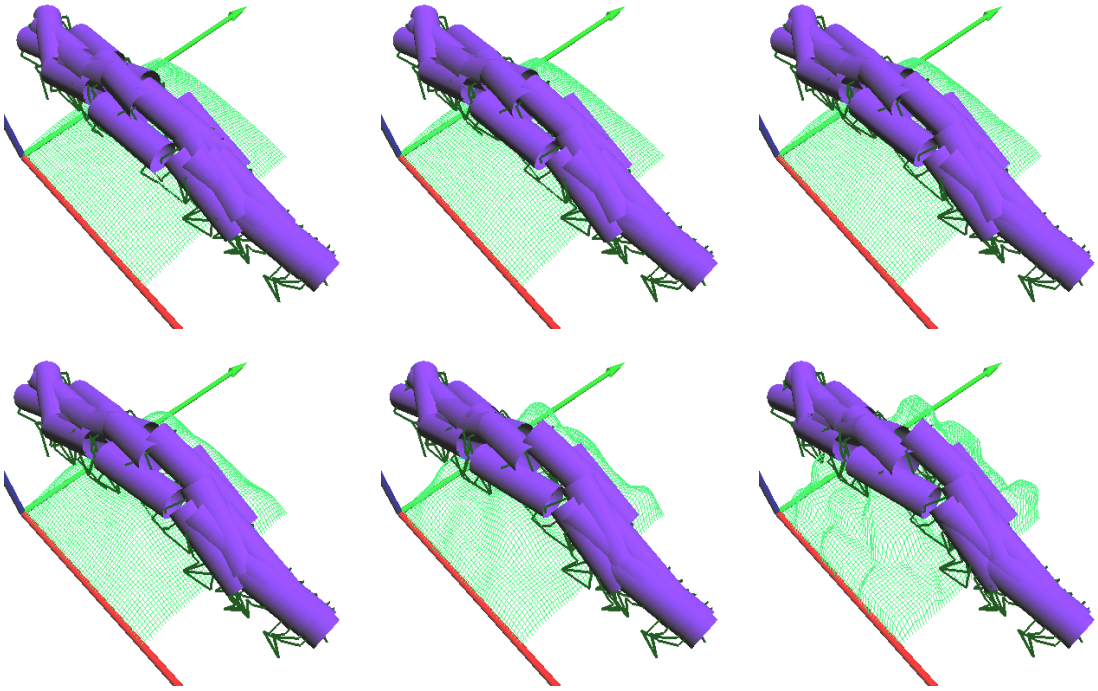


Figure 5.4: LittleDog's discrete trajectory on the terrain from Fig. 5.3. The terrain surface is initially smoothed and gradually deformed to its original. As the terrain changes the computed trajectory is adjusted accordingly to satisfy all stability and contact constraints. This type of "embedded" constrained optimization successfully yields an optimal solution, while optimization performed on the original terrain does not converge because of the high irregularity of the terrain.

Fig. 5.4 shows several stages of the terrain deformation and the resulting trajectory deformation to account for the changing stability and contact constraints. The final trajectory satisfies the contact and stability constraints and could potentially be used as a reference trajectory for successful terrain crossing.

Note The map resolution Δ can also be made part of the homotopy since the discrete convolution (5.3) should be performed at low resolution when the Gaussian standard deviation is high. As the standard deviation goes to 0, i.e. when the terrain returns to

its original form, the resolution should be increased to its highest value. This can be achieved by replacing Δ in (5.3) with Δ^λ defined by

$$\Delta^\lambda = (1-\lambda)\Delta_{high} + \lambda\Delta_{low},$$

where Δ_{high} is the original resolution of the terrain map, and Δ_{low} is the lowest resolution that would approximate the terrain.

5.3.2 Nonholonomic constraints

Assume that we are given a control system on n-dimensional manifold Q with regular distribution $\mathcal{D}_q = \text{span}\{f_i(q), i = 1, ..., c\}$ at $q \in Q$, where $f_i : Q \to TQ$ are linearly independent smooth vector fields. Velocity vectors can then be described locally by coordinates $v_q \in \mathcal{D}_q \sim \mathbb{R}^c$ so that the system has the form

$$\dot{q} = \sum_{i=1}^{c} v^i f_i(q)$$

$$\dot{v} = u,$$

where $u \in \mathbb{R}^c$ is the control input.

Assume that the task is to drive the system from its current state $(q(0), v(0))$ to a goal state $(q(T), v(T))$ after some finite time T, and that the system is controllable given this setup. Let $f_j : Q \to TQ$, $j = c+1, ..., n$ be $n-c$ linearly independent vector fields orthogonal to $\mathcal{D}$, i.e. such that $\ll f_i, f_j \gg = 0$, for all $i = 1, ..., c$ and $j = c+1, ..., n$. Then $T_qQ = \text{span}\{f_i(q), i = 1, ..., n\}$ for all $q \in Q$.

Let us imagine for a moment that the system was not restricted to move along directions in $\mathcal{D}$ only, so that it is fully holonomic. Assume that in this case we can easily find a solution to the motion planning problem and denote this solution trajectory by $(\bar{q}(t), \bar{v}(t))$, $t \in [0, T]$. Then, one way to compute a solution to the real nonholonomic problem is to continuously deform $(\bar{q}(t), \bar{v}(t))$ by gradually removing the degrees of freedom corresponding to f_j, $j = c+1, ..., n$ but keeping the boundary conditions fixed. This can be accomplished using the homotopy

$$\dot{q}^\lambda = (1-\lambda)\sum_{i=1}^{c} v^i f_i(q) + \lambda \sum_{i=1}^{n} \bar{v}^i f_i(q)$$

$$\dot{v} = u,$$

Clearly, $q^1(t) = \bar{q}(t)$ and $q^0(t) = q(t)$. The modified system can be used to find a trajectory satisfying the boundary conditions through a path-lifting method or it could serve as a constraint in an optimal control problem, i.e. minimizing the control effort $\int_0^T \|u\|^2 dt$.

Initial experiments with such homotopy are preformed on a simple nonholonomic unicycle model. Fig. 5.5 shows several stages of the homotopic computation of a control-effort minimizing trajectory.

Currently, this type of control system deformation is only an idea and need to be explored in much more depth.

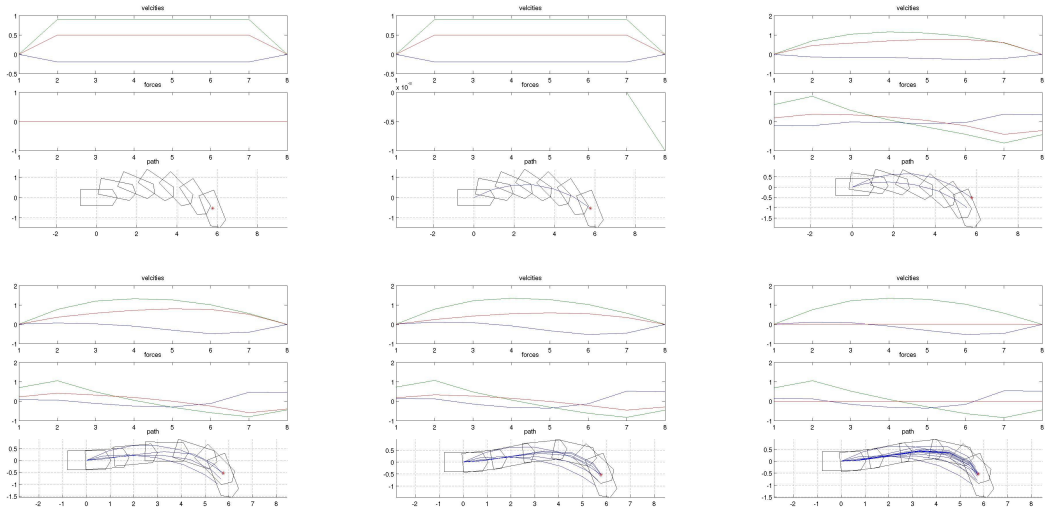


Figure 5.5: Homotopy stages of computing an optimal trajectory for a nonholonomic robot. Initially the robot is fully holonomic and a path can be computed very quickly. That path is then gradually deformed to account for the nonholonomic constraint as well as the optimality conditions.

5.4 Conclusion

The simple idea behind the methods described in this section was to solve motion planning problems more robustly by relaxing difficult constraints. The developed algorithms apply to iterative numerical schemes arising from root-finding or nonlinear programming motion planning formulation. The approach facilitates the computation of the roots of nonlinear equations or of the minimum of a cost function subject to complex constraints by initially removing, smoothing, or convexifying the constraints and then smoothly transforming them back to their original form and adjusting the solution accordingly. Applying the approach to environmental obstacles and to kinematic constraints proved promising in terms of increased efficiency (as in the car and helicopter examples) and solution convergence when the standard approach fails (as in the rough terrain and dog robot example).

Yet, it is a difficult task to provide theoretical performance guarantees for general constraints. This is an important research direction requiring both local and global analysis of the space of solution trajectories, subject to varying, possibly non-smooth constraints, and optimality conditions.

Reference List

[1] Eugene Allgower and Kurt Georg. *Introduction to Numerical Continuation Methods*. SIAM Wiley and Sons, 2003.

[2] A. M. Bloch, P. S. Krishnaprasad, J. E. Marsden, and R. Murray. Nonholonomic mechanical systems with symmetry. *Arch. Rational Mech. Anal.*, (136):21–99, 1996.

[3] Anthony Bloch. *Nonholonomic Mechanics and Control*. Springer, 2003.

[4] Alexander I. Bobenko and Yuri B. Suris. Discrete lagrangian reduction, discrete euler-poincare equations, and semidirect products. *Letters in Mathematical Physics*, 49:79, 1999.

[5] Nawaf Bou-Rabee and Jerrold E. Marsden. Reduced hamilton-pontryagin variational integrators. preprint, 2007.

[6] Francesco Bullo and Andrew Lewis. *Geometric Control of Mechanical Systems*. Springer, 2004.

[7] Elena Celledoni and Brynjulf Owren. Lie group methods for rigid body dynamics and time integration on manifolds. *Comput. meth. in Appl. Mech. and Eng.*, 19(3,4):421–438, 2003.

[8] H. Cendra, J. E. Marsden, S. Pekarsky, and T. S. Ratiu. Variational principles for lie-poisson and hamilton-poincar equations. *Moscow Mathematical Journal*, 3:833–867, 2003.

[9] H. Cendra, J. E. Marsden, and T. S. Ratiu. Lagrangian reduction by stages. *Mem. Amer. Math. Soc.*, 152(722):108, 2001.

[10] H. Cendra, J.E. Marsden, and T.S. Ratiu. Geometric mechanics, lagrangian reduction, and nonholonomic systems. In B. Engquist and W. Schmid, editors, *Mathematics Unlimited-2001 and Beyond*, pages 221–273. Springer-Verlag, New York, 2001.

[11] Howie Choset, Kevin M. Lynch, Seth Hutchinson, George A Kantor, Wolfram Burgard, Lydia E. Kavraki, and Sebastian Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, June 2005. ISBN 0-262-03327-5.

[12] J. Cortés, S. Martinez, J. P. Ostrowski, and K. A. McIsaac. Optimal gaits for dynamic robotic locomotion. *The International Journal of Robotics Research*, 20(9):707–728, 2001.

[13] Jorge Cortés. *Geometric, Control and Numerical Aspects of Nonholonomic Cystems*. Springer, 2002.

[14] M. de Leon, D. Martin de Diego, and A. Santamaria-Merino. Geometric integrators and nonholonomic mechanics. *Journal of Mathematical Physics*, 45(3):1042–1062, 2004.

[15] Jaydev P. Desai and Vijay Kumar. Motion planning for cooperating mobile manipulators. *Journal of Robotic Systems*, 16(10):557–579, 1999.

[16] Chris Dever, Bernard Mettler, Eric Feron, and Jovan Popovic. Nonlinear trajectory generation for autonomous vehicles via parameterized maneuver classes. *Journal of Guidance, Control, and Dynamics*, 29(2):289–302, 2006.

[17] Chang D. E., S. Shadden, J. E. Marsden, and R. Olfati-Saber. Collision avoidance for multiple agent systems. In *IEEE Conference on Decision and Control*, volume 42, pages 539–543, 2003.

[18] Kanso E., J.E. Marsden, C.W. Rowley, and J. Melli-Huber. Locomotion of articulated bodies in a perfect fluid. *Journal of Nonlinear Science*, (15):255–289, 2005.

[19] Yuri N. Fedorov and Dmitry V. Zenkov. Discrete nonholonomic ll systems on lie groups. *Nonlinearity*, 18:2211–2241, 2005.

[20] R.C. Fetecau, J.E. Marsden, M. Ortiz, and M. West. Nonsmooth lagrangian mechanics and variational collision integrators. *SIAM Journal on Applied Dynamical Systems*, 2(3):381–416, 2003.

[21] E. Frazzoli, M. A. Dahleh, and E. Feron. Real-time motion planning for agile autonomous vehicles. *AIAA Journal of Guidance, Control, and Dynamics*, 25(1):116–129, 2002.

[22] E. Frazzoli, M. A. Dahleh, and E. Feron. Maneuver-based motion planning for nonlinear systems with symmetries. *IEEE Transactions on Robotics*, 21(6):1077–1091, dec 2005.

[23] Roberto Giambo, Fabio Giannoni, and Paolo Piccionez. Optimal control on riemannian manifolds by interpolation. *Math. Control Signals System*, 16:278–296, 2003.

[24] E. Hairer, Ch. Lubich, and G. Wanner. *Geometric Numerical Integration*. Number 31 in Springer Series in Computational Mathematics. Springer-Verlag, 2006.

[25] I.I. Hussein and A.M. Bloch. Dynamic interpolation on riemannian manifolds: an application to interferometric imaging. In *American Control Conference*, number 1, pages 685– 690, 2004.

[26] Toshihiro Iwai and Akitomo Tachibana. The geometry and mechanics of multi-particle systems. *Annales de l'institut Henri Poincar (A) Physique thorique*, 70(5):525–559, 1999.

[27] Sameer M. Jalnapurkar, Melvin Leok, Jerrold E. Marsden, and Matthew West. Discrete routh reduction. *MATH.GEN.*, 39:5521, 2006.

[28] O. Junge, J.E. Marsden, and S Ober-Blöbaum. Discrete mechanics and optimal control. In *Proccedings of the 16th IFAC World Congress*, 2005.

[29] O. Junge, J.E. Marsden, and S. Ober-Blöbaum. Optimal reconfiguration of formation flying spacecraft - a decentralized approach. In *45th IEEE Conference on Decision and Control*, pages 5210–5215, 2006.

[30] Eva Kanso and Jerrold Marsden. Optimal motion of an articulated body in a perfect fluid. In *IEEE Conference on Decision and Control*, pages 2511–2516, 2005.

[31] Scott Kelly and Richard Murray. Geometric phases and robotic locomotion. *Journal of Robotic Systems*, 12(6):417–431, 1995.

[32] L. Kharevych, Weiwei, Y. Tong, E. Kanso, J.E. Marsden, P. Schroder, and M. Desbrun. Geometric, variational integrators for computer animation. In *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, pages 1–9, 2006.

[33] Marin Kobilarov and Gaurav Sukhatme. Reference redacted for author anonymity. 2006.

[34] Wang-Sang Koon and Jerrold E. Marsden. Optimal control for holonomic and nonholonomic mechanical systems with symmetry and lagrangian reduction. *SIAM Journal on Control and Optimization*, 35(3):901–929, 1997.

[35] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006.

[36] T. Lee, N.H. McClamroch, and M. Leok. Optimal control of a rigid body using geometrically exact computations on SE(3). In *Proc. IEEE Conf. on Decision and Control*, 2006.

[37] F. Silva Leite, M. Camarinha, and P. Crouch. Elastic curves as solutions of riemannian and sub-riemannian control problems. *Math. Control Signals Systems*, (13):140–155, 2000.

[38] A. Santamaria Merino M. de Leon, D. Martin de Diego. Geometric numerical integration of nonholonomic systems and optimal control problems. *European Journal of Control*, 10:520–526, 2004.

[39] J. E. Marsden and J. Ostrowski. Symmetries in motion: Geometric foundations of motion control. *Nonlinear Sci. Today*, 1998.

[40] J. E. Marsden and J. Scheurle. The reduced euler-lagrange equations. *Fields Inst. Commun.*, (1):139–164, 1993.

[41] J.E. Marsden and M. West. Discrete mechanics and variational integrators. *Acta Numerica*, 10:357–514, 2001.

[42] Jerrold E. Marsden, Sergey Pekarsky, and Steve Shkoller. Discrete euler-poincare and lie-poisson equations. *Nonlinearity*, 12:1647–1662, 1999.

[43] Jerrold E. Marsden and Tudor S. Ratiu. *Introduction to Mechanics and Symmetry*. Springer, 1999.

[44] R. McLachlan and M. Perlmutter. Integrators for nonholonomic mechanical systems. *Journal of NonLinear Science*, 16:283–328, aug 2006.

[45] James Ostrowski. *The Mechanics and Control of Undulatory Robotic Locomotion*. PhD thesis, California Institute of Technology, 1996.

[46] James Ostrowski. Computing reduced equations for robotic systems with constraints and symmetries. *IEEE Transactions on Robotics and Automation*, pages 111–123, 1999.

[47] James P. Ostrowski, Jaydev P. Desai, and Vijay Kumar. Optimal gait selection for nonholonomic locomotion systems. *The International Journal of Robotics Research*, 19(3):225–237, 2000.

[48] Crouch P. and Leite F. S. The dynamic interpolation problem: on riemannian manifolds, lie groups, and symmetric spaces. *Journal of Dynamical and Control Systems*, 1(2):177–202, 1995.

[49] Sam Prentice and Nicholas Roy. The belief roadmap: Efficient planning in linear pomdps by factoring the covariance. In *Proceedings of the 13th International Symposium of Robotics Research (ISRR)*, Hiroshima, Japan, November 2007.

[50] Shane Ross. Optimal flapping strokes for self-propulsion in a perfect fluid. In *American Control Conference*, 2006.

[51] Elie A. Shammas, Howie Choset, and Alfred A. Rizzi. Motion planning for dynamic variable inertia mechanical systems with non-holonomic constraints. In *International Workshop on the Algorithmic Foundations of Robotics*, 2006.

[52] T. Yanao, W. S. Koon, J. E. Marsden, and I. G. Kevrekidis. Gyration-radius dynamics in structural transitions of atomic clusters. *J. Chem. Physics.*, (126):1–17, 2007.

[53] H. Yoshimura and J.E. Marsden. Dirac structures in lagrangian mechanics part ii: Variational structures. *Journal of Geometry and Physics*, 57:209–250, dec 2006.