Discrete Optimal Control on Lie groups and Applications to Robotic Vehicles

Marin Kobilarov[1]

***Abstract*— This paper is concerned with optimal trajectory generation for robotic multi-body systems. The focus is on discrete optimal control methods which operate intrinsically in the state space system manifold and do not require coordinate charts or projections. This is accomplished by defining both the dynamics and the optimal control solution as sequences of vector fields mapping to curves on the Lie group through retraction maps, and defining variations and differentiation with respect to such vector fields. As a result, standard trajectory optimization methods can be easily extended to the Lie group setting without loss of efficiency. The methods are illustrated with three numerical examples: a quadrotor, an aerial vehicle with manipulators, and a simple nonholonomic system.**

I. INTRODUCTION

This paper considers the optimal control of robotic vehicles modeled as multi-body systems subject to holonomic or nonholonomic constraints. Such systems are typically described by their pose g and joint angles r, where g is typically an element of the Euclidean motion group $SE(2)$ or $SE(3)$. The configuration space Q as well as the state space X of such systems can be regarded as Lie groups which enables the development of coordinate-invariant algorithms. The paper adopts this point of view to develop practical optimal control methods for any general Lie group with focus on efficient implementation. Differential-geometric and Lie group structure are naturally present in multi-body dynamics simulation and a number of methods have been developed to take advantage of it [1], [2], [3], [4]. On the other hand, optimal control methods for Lie groups have been mostly limited to theoretical developments seeking analytical solutions e.g. by exploiting symmetries [5], [6] and reduction [7], [8] with application mostly limited to single rigid bodies [9] or specific systems. Several recent methods focusing on the discrete [10], [11] and continuous [12] setting were aimed at providing general numerical optimal control formulation for any Lie group. This paper builds upon these works to provide a practical optimal control approach leading to efficient algorithms for general systems. In particular, we first specify three general ways to formulate the dynamics intrinsically by regarding Q and X as general Lie groups equipped with *retraction* maps for evolving the system intrinsically. We then specify differential Lie group operators which enable the application of standard optimal control methods such as sequential quadratic programming, stage-wise Newton, and differential dynamic programming, to complex dynamics. Finally, the developed algorithms are illustrated with three numerical examples: a quadrotor, an aerial robot with two manipulator arms, and a car-like robot, which are performing non-trivial optimized maneuvers.

*This work was partially supported by NSF ISS:1302360.

[1]Marin Kobilarov is is with Faculty of Mechanical Engineering, Johns Hopkins University, Baltimore, MD, USA marin@jhu.edu

II. THE LIE GROUP SETTING

The configuration space Q as a Lie group: The configuration space of robotic multi-body systems is defined as $Q = G \times \mathbb{M}$, where G denotes the Euclidean group which is $G = SE(2)$ in the planar or $G = SE(3)$ in the 3-D case, and the vector space $\mathbb{M} \subset \mathbb{R}^\ell$ denotes the *joint space* assuming there are ℓ joints. A given configuration $q \in Q$ denotes the system *posture*, i.e. its overall pose as well as its shape. Since Q is a direct product of a Lie group G and vector space $\mathbb{M}$ then Q is a Lie group itself, also referred to as a trivial fiber bundle (i.e. there is a fiber G attached at each base point $r \in \mathbb{M}$). The configuration space dimension is denoted by $m = \dim(Q)$, where $m = 6 + \ell$ when $G = SE(3)$ or $m = 3 + \ell$ when $G = SE(2)$.

The fundamental property of Lie groups is that each tangent vector on the manifold can be generated by translating a unique tangent vector at the identity using the group operation. More formally, each vector $\dot{q} \in T_q Q$ at configuration $q \in Q$ corresponds to a unique vector $\xi \in \mathbb{R}^m$ through $\dot{q} = q\widehat{\xi}$ where the "hat" operator $\widehat{\cdot} : \mathbb{R}^m \to \mathfrak{q}$ identifies ξ with a Lie algebra element matrix $\widehat{\xi} \in \mathfrak{q}$. Here $\mathfrak{q}$ denotes the *Lie algebra* and $e \in Q$ denotes the group identity [13]. The "vee" map $(\cdot)^\vee : T_e Q \to \mathbb{R}^m$ is defined as the inverse of $\widehat{\cdot}$, so that $(\widehat{\xi})^\vee = \xi$.

Nonholonomic Constraints and Reduced Velocities: We consider multi-body systems that can be subject to nonholonomic constraints, e.g. from rolling, sliding, or from conservation laws. Assume that there are $k < m$ constraints specified by the vectors $w_i(q) \in \mathbb{R}^m$ through the equalities

$$w_i^T(q)\xi = 0, \quad i = 1, \dots, k,$$

which can be combined in matrix form using $W(q) = [w_1(q), \cdots, w_k(q)]$ as $W(q)^T\xi = 0$. One can associate null space basis vectors $g_1(q), ..., g_{m-k}(q) \in \mathbb{R}^n$ such that

$$w_i(q)^T g_j(q) = 0, \forall i = 1, \dots, k, \; j = 1, \dots, m-k$$

or in matrix form, using $S(q) = [g_1(q), \cdots, g_{m-k}(q)]$, as $W^T(q)S(q) = 0$. The constraints can be directly satisfied by replacing the body-fixed velocities ξ with *reduced* or *pseudo-velocities* $v \in \mathbb{R}^{m-k}$ which satisfy $\xi = S(q)v$ and hence the configuration evolves according to

$$\dot{q} = q \; [S(q)v]\widehat{\ }.$$

Note that in the absence of velocity constraints we simply have $S = I$ and $\xi = v$.

Continuous Equations of Motion: The continuous equations of motion describing the dynamics for robotic systems (assuming $G = SE(3)$ for generality) are then expressed as

$$\begin{aligned} \dot{q} &= q \cdot [S(q)v]^{\widehat{}}, && : \text{kinematics} \quad (1) \\ M(q)\dot{v} + b(q,v) &= B(q)u, && : \text{dynamics} \quad (2) \end{aligned}$$

where $q = (g, r) \in Q = SE(3) \times \mathbb{R}^\ell$ and $v = (V, \dot{r}) \in \mathbb{R}^{6+\ell}$. The group multiplication $q_1 q_2$ and tangent operation $q \cdot \hat{v}$ are defined, respectively, by

$$q_1 q_2 = (g_1 g_2, r_1 + r_2), \qquad q \cdot \hat{v} = (g\hat{V}, \dot{r}),$$

with the hat operator $\hat{\cdot} : \mathbb{R}^{6+\ell} \to \mathfrak{se}(3) \times \mathbb{R}^\ell$ in this case is $\hat{v} = (\hat{V}, \dot{r})$, where $\hat{V}$ for a given $V = (\omega, \nu)$ is defined by

$$\hat{V} = \begin{bmatrix} \hat{\omega} & \nu \\ 0_{1\times 3} & 0 \end{bmatrix}, \quad \hat{\omega} = \begin{bmatrix} 0 & -w_3 & w_3 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix}. \quad (3)$$

For holonomic systems the mass matrix $M(q)$, *bias* terms $b(q,v)$ and control matrix $B(q)$ employed in (1)–(2) are computed using standard methods such as the articulated composite body algorithm [14] or the more general spatial operator theory [15]. With our coordinate choice the mass matrix in fact only depends on r rather than q and for tree-structured systems can be computed readily according to

$$M'(r) = \left[\begin{array}{c|c} \mathbb{I}_0 + \sum_{i=1}^{n} A_i^T \mathbb{I}_i A_i & \sum_{i=1}^{n} A_i^T \mathbb{I}_i J_i \\ \hline \sum_{i=1}^{n} J_i^T \mathbb{I}_i A_i & \sum_{i=1}^{n} J_i^T \mathbb{I}_i J_i \end{array} \right] \quad (4)$$

using the adjoint notation $A_i := \mathrm{Ad}_{g_{0i}^{-1}(r)}$, and Jacobian $J_i := \sum_{j=1}^{n} [g_{0i}^{-1}(r)\partial_{r_j} g_{0i}(r)]^\vee$,where $g_{0i}(r)$ is the relative transformation from the base body to body $\#i$ and $\mathbb{I}_i$ is the inertia tensor of body $\#i$ [16]. In the nonholonomic case, the terms are computed according to

$$\begin{aligned} &M(q) = S^T(q)M'(q)S(q), \quad B(q) = S^T(q)B'(q), \\ &b(q,v) = S^T(q)M'(q)\dot{S}(q)v + S^T(q)b'(q, S(q)v), \end{aligned} \quad (5)$$

where $M'(q)$, $b'(q,v)$, and $B'(q)$ denote the mass matrix, bias term, and control matrix, computed in the standard unconstrained setting.

The state-space X as a Lie group: With these definitions, the state space of the system is denoted by $X = Q \times \mathbb{R}^{m-k}$ and its dimension is

$$\dim(X) = 2m - k \equiv n$$

The algorithms developed in this work perform optimization over state trajectories $x : [0,T] \to X$. This motivates us to regard X as an abstract n-dimensional Lie group to enable a straightforward extension of existing optimal control algorithms to X based on a set of general Lie group operations that will be defined.

III. Geometric Integration on Lie groups

Let $x \in X$ denote the state of the system. When X is a coordinate vector space the equations of motion have the standard state-space form $\dot{x} = f(t,x,u)$ and can be integrated numerically according to

$$x_{k+1} = x_k + f_k, \quad (6)$$

where f_k encodes the update of a one-step method (explicit or implicit). For instance, the simplest Euler update gives $f_k = hf(t_k, x_k, u_k)$ where h is the time-step. In contrast, when X is a manifold, it is more convenient to define

$$\dot{x} = x \cdot f(t,x,u), \quad (7)$$

which can be regarded as generalized version of the equations of motion (1)–(2). The Lie algebra element $f(t,x,u) \in T_eX$ is interpreted as the "body-fixed" state velocity and the product $x\,f$ denotes the tangent group action of x on f, which for our purposes is typically just a body-fixed to spatial rotation. Since x is not a vector, an integrator such as (6) is not directly applicable. Instead, the time-update $x_k \to x_{x+1}$ is performed by evolving a geodesic motion on X, i.e. as a curve with constant velocity which equals precisely $f_k \in T_eX$. In practice, the geodesic flow along f_k is computed either exactly or approximately using a *retraction map*

$$\varphi : \mathbb{R}^n \to X$$

which serves as an approximation of the standard exponential map on X.

Geometric Lie group integrators [17], [18], [19] are special integrators which often use the map φ to expresses changes in the group in terms of elements in its Lie algebra. The well-known *exponential map* was the first such map proposed for integration purposes in [20]. Retaining the Lie group structure and motion invariants under discretization has, since then, been proven to be not only a nice mathematical property, but also key to improved numerics, as they capture the right dynamics (even in long-time integration) and exhibit increased accuracy [21].

The resulting geometric integrator can be generally expressed as

$$x_{k+1} = x_k\, \varphi(f_k), \quad (8)$$

where f_k can be constructed either explicitly from x_k, u_k or could also be an implicit function of x_{k+1} as well. Examples of each case will be provided in §IV.

Retraction Maps: In this work we will employ two types of retractions, the exponential map and the Cayley map. While these maps apply to a large class of Lie groups (see [10] for general definitions), for our purposes we will be concerned with retractions for the Lie groups $Q = G \times \mathbb{R}^\ell$ and $X = G \times \mathbb{R}^{n-d}$, where $d = \dim(G)$. To construct $\varphi : \mathbb{R}^m \to Q$ and $\varphi : \mathbb{R}^n \to X$.

The exponential map for Euclidean groups is standard [22], hence we focus on the Cayley map $\mathrm{cay} : \mathbb{R}^m \to G$ defined by

$$\mathrm{cay}(V) = \left(I - \hat{V}/2\right)^{-1}\left(I + \hat{V}/2\right).$$

For a given state state $x=(g,r,v)\in X$ and algebra element $\eta=(V,\Delta r,\Delta v)\in\mathbb{R}^n$ the retraction is defined by

$$x\,\varphi(\eta)\equiv(g\,\mathrm{cay}(V),r+\Delta r,v+\Delta v)\,.$$

In particular, for $G=SE(3)$ the Cayley map $\mathrm{cay}:\mathbb{R}^6\to SE(3)$ (see [10]) is

$$\mathrm{cay}(\omega,\nu)=\begin{bmatrix} I_3+\frac{4}{4+\|\omega\|^2}\left(\widehat{\omega}+\frac{\widehat{\omega}^2}{2}\right) & \frac{2}{4+\|\omega\|^2}(2I_3+\widehat{\omega})\,\nu \\ 0 & 1\end{bmatrix}. \quad (9)$$

while its inverse $\mathrm{cay}^{-1}:SE(3)\to\mathbb{R}^6$ is defined for a given $g=(R,p)$, with $R\neq -I$, by

$$\mathrm{cay}^{-1}(g)=\begin{bmatrix}[-2(I+R)^{-1}(I-R)]^\vee \\ (I+R)^{-1}p\end{bmatrix}. \quad (10)$$

IV. Discrete Equations of Motion

We consider three general ways for discretizing the equations of motion for numerical optimal control purposes.

Semi-implicit First-order Integrator: One of the simplest first-order geometric integration methods is obtained through an Euler discretization of the dynamics:

$$q_{k+1}=q_k\varphi\left(hS(q_k)v_{k+1}\right), \quad (11)$$

$$M(q_k)\frac{v_{k+1}-v_k}{h}+b(q_k,v_k)=B(q_k)u_k,. \quad (12)$$

This method is implicit since one first updates the velocity using the dynamics (12) and then updates the configuration using the kinematics (11). The method is not recommended for highly nonlinear systems or for systems that are naturally unstable.

Implicit Second-order Integrator: An integrator with improved numerical stability can be obtained through a symmetric trapezoidal discretization of the dynamics and takes the form

$$q_{k+1}=q_k\varphi\left(\frac{h}{2}\left[S(q_k)+S(q_{k+1})\right]v_{k+1}\right), \quad (13)$$

$$M(q_k)\frac{v_{k+1}-v_k}{h}+\frac{1}{2}\left[b(q_k,v_k)+b(q_k,v_{k+1})\right]=B(q_k)u_k. \quad (14)$$

The method is implicit and requires a gradient-based algorithm during integration. The basic requirement is that the gradient of the equality constraint (14) with respect to v_{k+1} is invertible. This condition can be satisfied through a proper choice of the time-step h assuming that the matrix $M(q_k)$ is full rank. For further discussion see e.g. [23].

Implicit Second-order Integrator with variable step-size: Finally, it is possible to gain significant computational efficiency by employing variable time steps, i.e. by taking larger h during steady-state motions and smaller h during maneuvering. This can be accomplished by modifying the equations (11)– (12) according to

$$q_{k+1}=q_k\varphi\left(\frac{h_{k+1}}{2}\left[S(q_k)+S(q_{k+1})\right]v_{k+1}\right) \quad (15)$$

$$\begin{aligned}&M(q_k)[v_{k+1}-v_k]+\frac{1}{2}\left[h_k b(q_k,v_k)+h_{k+1}b(q_k,v_{k+1})\right]\\ &=h_{k+\frac{1}{2}}B(q_k)u_k,\end{aligned} \quad (16)$$

where

$$h_{k+1}=t_{k+1}-t_k, \qquad h_{k+\frac{1}{2}}=\frac{h_k+h_{k+1}}{2}.$$

V. Variations of Functions on Lie Groups

Developing optimal control algorithms on the Lie group X is based on taking variations $\delta x\in T_xX$ along a given trajectory. In our setting the elements δx cannot be represented as vectors and cannot be directly handled by standard numerical methods. We will instead employ *left-trivialized variations*

$$\bar{\delta}x=(x^{-1}\delta x)^\vee\in\mathbb{R}^n$$

which have a minimal vector representation. Furthermore, the numerical solution to our optimal control problem will be performed through directional (or Lie) derivative along such reduced variations.

Definition 5.1: The *left-trivialized gradient* $\bar{\nabla}_x f(x)\in\mathbb{R}^{m\times n}$ of a function $f:X\to\mathbb{R}^m$, where X is an n-dimensional Lie group equipped with a retraction map $\varphi:\mathbb{R}^n\to X$, is defined by

$$\bar{\nabla}_x f(x)=\nabla_\xi\Big|_{\xi=0}f(x\,\varphi(\xi))$$

or in coordinates using the standard basis $\{e_1,\dots,e_n\}$ of $\mathbb{R}^n$ by

$$\bar{\nabla}_x f(x)=\left[\frac{\partial f}{\partial s}\Big|_{s=0}(x\,\varphi(se_1)),\ \cdots,\ \frac{\partial f}{\partial s}\Big|_{s=0}(x\,\varphi(se_n))\right]^T.$$

The Taylor series expansion of scalar-valued function $L(x)$ can now be written compactly according to

$$L\left(x\,\varphi(d)\right)=L(x)+\bar{\nabla}L(x)^Td+\frac{1}{2}d^T\bar{\nabla}^2L(x)d+o(\|d\|^2),$$

where the Hessian $\bar{\nabla}^2L$ is computed by applying the trivialized gradient twice which provides sufficient accuracy for obtaining quadratic convergence in Newton-type methods [24], [22]. Note that this is only one among several ways to compute second-order terms and corresponds to the '0'-Cartan-Shouten connection (connection defining zero-acceleration geodesics on manifolds). A more detailed discussion of the geometric importance of this choice is given in [12] in the context of continuous trajectory optimization on Lie groups.

Definition 5.2: The *Jacobian* $\mathrm{d}\varphi:\mathbb{R}^n\to\mathbb{R}^{n\times n}$ of a retraction map $\varphi:\mathbb{R}^n\to X$ is such that, for any $\xi,\eta\in\mathbb{R}^n$,

$$\left[\mathrm{d}\varphi(\xi)\cdot\eta\right]^\wedge=D_\xi\varphi(\xi)\cdot\widehat{\eta}\cdot\varphi(-\xi),$$

or in coordinates

$$\mathrm{d}\varphi(\xi)=\left[\left(\frac{\partial\varphi}{\partial\xi_1}(\xi)\cdot\varphi(-\xi)\right)^\vee,\ \cdots,\ \left(\frac{\partial\varphi}{\partial\xi_n}(\xi)\cdot\varphi(-\xi)\right)^\vee\right].$$

In particular for the case $x=(g,r,v)$ where $g\in SE(3)$ is updated using the Cayley map we have

$$\mathrm{d}\varphi(V,\Delta r,\Delta v)=\begin{bmatrix}\mathrm{dcay}(V) & & \\ & I & \\ & & I\end{bmatrix},$$

where I denotes the identity map. The Cayley tangents for a given $V=(\omega,\nu)\in\mathbb{R}^6$ have the simple form [10]

$$\mathrm{dcay}(\omega,\nu)=\begin{bmatrix}\frac{2}{4+\|\omega\|^2}(2I_3+\widehat{\omega}) & 0_3\\ \frac{1}{4+\|\omega\|^2}\widehat{\nu}(2I_3+\widehat{\omega}) & I_3+\frac{1}{4+\|\omega\|^2}(2\widehat{\omega}+\widehat{\omega}^2)\end{bmatrix}. \quad (17)$$

It can be shown that the Jacobian is invertible with inverse

$$\mathrm{dcay}^{-1}(\omega,\nu)=\begin{bmatrix} I_3-\frac{1}{2}\widehat{\omega}+\frac{1}{4}\omega\omega^T & 0_3\\ -\frac{1}{2}\left(I_3-\frac{1}{2}\widehat{\omega}\right)\widehat{\nu} & I_3-\frac{1}{2}\widehat{\omega}\end{bmatrix}. \quad (18)$$

These Jacobians will be used in the linearization of the dynamics required for trajectory optimization.

VI. Discrete Optimal Control

The general problem is formulated in terms of discrete trajectories $x_{0:N}\triangleq\{x_0,...,x_N\}$ and controls $u_{0:N-1}\triangleq\{u_0,...,u_{N-1}\}$ as:

$$\begin{aligned}&\text{minimize} \quad J(x_{0:N},u_{0:N-1})=\sum_{k=0}^{N-1}L_k(x_k,u_k)+L_N(x_N),\\ &\text{subject to:}\ \ F_k(x_{k+1},x_k,u_k)=0, \qquad k=0,\dots,N-1,\\ &\qquad\qquad G_k(x_k,u_k)\leqslant 0,\ \ G_N(x_N)\leqslant 0,\ \ k=0,\dots,N-1,\end{aligned}$$

where L_k are stage-wise costs and L_N is the terminal cost. The function F_k encodes the discrete dynamics and G_k encodes any additional constraints including control bounds and path constraints. In particular, the function F_k could encode any numerical *one-step* scheme defined as a Lie group update of the form $x_{k+1}=x_k\varphi(f_k(x_k,u_k))$. For instance, in the semi-implicit case (11)-(12) we have

$$f_k(x,u)=h\begin{bmatrix}S(q)(v+ha)\\ a\end{bmatrix},\ \text{where}$$

$$a=M(q)^{-1}\left[-b(q,v)+B(q)u\right],$$

while in the fully implicit case it will encode an iterative gradient-based solution.

Linearization

The variational solution of the problem will required linearization of the dynamics based on the differential calculus developed in §V. It is possible to perform linearization numerically through the dynamics $x_{k+1}=x_k\,\varphi(f_k(x_k,u_k))$ where the function f_k might or might not be available analytically. Such a *black-box linearization* takes the form

$$\bar{\delta}x_{k+1}=A_k\cdot\bar{\delta}x_k+B_k\cdot\delta u,$$

with state and control matrices A_k and B_k are defined by

$$A_k=\mathrm{Ad}_{\varphi(-f_k)}+\mathrm{d}\varphi(-f_k)\cdot\bar{\nabla}_x f_k, \qquad B_k=\mathrm{d}\varphi(-f_k)\cdot\nabla_u f_k.$$

The derivatives of f can be computed analytically or numerically. When dealing with an implicit formulation directly the linearization is given by

$$A_k=(\bar{D}_1F_k)^{-1}(\bar{D}_2F_k),\quad B_k=(\bar{D}_1F_k)^{-1}(D_3F_k), \quad (19)$$

where $\bar{D}_1F(x,y,z)\equiv\bar{\nabla}_xF$.

VII. Numerical Solution Methods

We distinguish two types of numerical methods for solving the optimal control problems. The first is based on standard nonlinear programming with a specific implementation to take advantage of problem sparsity. The second is based on recursive or stage-wise second-order approach exploiting the iterative nature of the dynamical constraints. The key point in both methods is to perform the optimization through reduced variations (expressed as Lie algebra vectors $\bar{\delta}x\in\mathbb{R}^n$) rather than directly optimizing the states $x\in X$, e.g. using a single coordinate chart fixed a-priori.

Sparse Nonlinear Programming

Denote the optimization parameter by $y=(x_{1:N},u_{0:N-1})\in X^N\times\mathbb{R}^{Nc}$ and its variation by $\bar{\delta}y=(\bar{\delta}x_{1:N},\delta u_{0:N-1})\in\mathbb{R}^{N(n+c)}$. The dynamics constraints are encoded through the equality

$$F(y)=\begin{bmatrix}F_0(x_1,x_0,u_0)\\ \vdots\\ F_{N-1}(x_N,x_{N-1},u_{N-1})\end{bmatrix}=0,$$

while all inequality constraints are similarly combined in a function $G(y)<0$. Nonlinear programming problems are typically formulated by adjoining the constraint to the cost $J(y)$ using the Lagrangian

$$L(y,\lambda)=J(y)-\lambda^TC(y)-\mu^TG(y) \quad (20)$$

and solving the quadratic programming subproblem:

$$\begin{aligned}&\min_{\bar{\delta}y}\ \bar{\nabla}J(y)^T\bar{\delta}y+\frac{1}{2}\bar{\delta}y^T\bar{\nabla}_{yy}L(y,\lambda)\bar{\delta}y,\\ &\text{subject to: } C(y)+\bar{\nabla}C(y)^T\bar{\delta}y=0,\\ &\qquad\qquad G(y)+\bar{\nabla}G(y)^T\bar{\delta}y\geqslant 0,\end{aligned} \quad (21)$$

iteratively by computing the direction $\bar{\delta}y$ which is then used to update the next iterate y' either according to $x'_k=x'_k\,\varphi(\bar{\delta}x)$ or using the updated controls $u'_{0:N-1}$ to update $x'_{0:N}$ using the nonlinear dynamics.

Stage-wise Newton and Differential Dynamic Programming

Instead of formulating an $N(n+c)$-dimensional monolithic program such as (21) it is possible to explicitly factor out the trajectory constraints in a recursive manner and solve N smaller problems of dimension $n+c$. *Stage-wise Newton (SN) method* [26] and *differential dynamic programming (DDP)* [27], [28] are the two standard methods for this purpose. The *pure* SN method [26] sequentially optimizes a second-order model of the Hamiltonian $H_k(x_k,u_k,\lambda_{k+1})=L_k(x_k,u_k)+\lambda_{k+1}^Tf_k(x_k,u_k)$ while DDP sequentially optimizes a local model of the Hamilton-Jacobi-Bellman *value function* $V_k(x_k,u_k)$ defined recursively by

$$V_k^*(x_k)=\min_{u_k\in\{u|G_k(x_k,u)\leqslant 0\}}\Big\{\underbrace{V_{k+1}^*(x_k\varphi(f_k(x_k,u_k)))+L_k(x_k,u_k)}_{V_k(x_k,u_k)}\Big\}.$$

At each iteration, such sweep methods adjust the control according to $u'_k = u_k + \delta u_k$ with

$$\delta u_k = -(R_k + B_k^T P_{k+1} B_k)^{-1} \cdot ((M_k + B_k^T P_k A_k)\bar{\delta} x_k + \nabla_u L_k + B_k^T \lambda_k),$$

where the matrix P_k and multiplier λ_k are defined recursively, starting with $P_N = Q_N$ and $\lambda_N = \bar{\nabla}_x L_N$, by

$$\begin{aligned} P_k =& A_k^T P_{k+1} A_k + Q_k - (B_k^T P_{k+1} A_k + M_k)^T \\ & \cdot (R_k + B_k^T P_{k+1} B_k)^{-1} (B_k^T P_{k+1} A_k + M_k), \\ \lambda_k =& \bar{\nabla}_x L_k + A_k^T \lambda_{k+1} - A_k^T P_{k+1} B_k \\ & \cdot (R_k + B_k^T P_{k+1} B_k)^{-1} (\nabla_u L_k + B_k^T \lambda_{k+1}), \end{aligned}$$

with variations $\bar{\delta} x_k = \varphi^{-1}(x_k^{-1} x'_k)$ where the new state x'_k is updated using the dynamics $x'_k = x'_{i-1} \varphi(f_{k-1}(x'_{i-1}, u'_{i-1}))$. The terms Q_k, R_k, M_k form the Hessian, i.e. for stage-wise Newton we have $Q_k \triangleq \bar{\nabla}_x^2 H$, $R_k \triangleq \nabla_u^2 H$, $M_k \triangleq \nabla_u \bar{\nabla}_x H$ and for DDP the term H_k is replaced by V_k. In particular, to guarantee local quadratic convergence it is required that $R_k + B_k^T P_{k+1} B_k > 0$. Standard regularization and step-size selection (e.g. using Armijo rule) are applied to ensure that the resulting controls $u'_{0:N-1}$ yield a sufficient decrease in the cost.

In summary, the components necessary to apply stage-wise methods to systems on Lie groups are: 1) left-trivialized gradients $\bar{\nabla}$, 2) linearization which includes the retraction Jacobians $\mathrm{d}\varphi$, 3) the retraction and its inverse during sweep method updates.

The Gauss-Newton case: Convergence conditions are greatly relaxed when the stage-wise costs L_k are in a separable least-squares form $L_k(x, u) = \frac{1}{2}\left(\|q_k(x)\|^2 + \|r_k(u)\|^2\right)$, where $q_k(x)$ and $r_k(u)$ are given nonlinear functions such that $Q_k \triangleq \bar{\nabla} q_k^T \bar{\nabla} q_k \geqslant 0$ (positive semidefinite) and $R_k \triangleq \nabla r_k^T \nabla r_k > 0$ (positive definite). Using these matrices instead of the complete second order terms corresponds to a Gauss-Newton (GN) approximation to the optimal solution and reduces the computation to a standard time-varying LQR Riccati iteration, which converges under standard controllability conditions.

The Gauss-Newton approach is a common approximation, and has been previously suggested under the name *iterative LQR* (iLQR) by [29] with evidence that efficiency can be gained while still computing accurate solutions. Iterative LQR methods have since been applied to general trajectory optimization methods for complex systems such as humanoid robots interacting with a physical world [30]. In a similar context, DDP methods have also been applied to stochastic control problems in robotics [31], [32], [33]. Note that iLQR is appropriate only when the cost is based on residuals that can be truly minimized to small values, otherwise second-order convergence would be lost.

The state-deviation cost: Many practical problems involve computing costs penalizing deviation from a desired state or desired reference path. For instance, assume that the terminal cost is designed to achieve a desired state x_f. It can be generally defined by

$$L_N(x) = \frac{1}{2}\|\varphi^{-1}(x_f^{-1} x)\|_{Q_f}^2,$$

where x_f is a desired final state and $Q_f \geqslant 0$ is a given matrix. The gradient is computed according to

$$\bar{\nabla}_x L_N(x) = (\mathrm{d}\varphi^{-1}(-\Delta))^T Q_f \Delta \tag{22}$$

where $\Delta \triangleq \varphi^{-1}(x_f^{-1} x)$ while the Gauss-Newton approximation to the Hessian takes the form

$$\bar{\nabla}_x^2 L_N(x) \approx Q_N \triangleq (\mathrm{d}\varphi^{-1}(-\Delta))^T Q_f \mathrm{d}\varphi^{-1}(-\Delta). \tag{23}$$

Quadratic control constraints: An important part of the optimal control formulation are control constraints. In addition to linear box bounds, quadratic bounds of the form

$$G_k(x, u) = u^T C_k u - 1 \leqslant 0, \tag{24}$$

for a given matrix $C_k > 0$ are straightforward to handle. When an update $u' = u + \delta u$ violates the constraint, one simply sets $u' = u + \beta \delta u$ instead, where β is the solution of the quadratic equation

$$\beta^2 \delta u^T C_k \delta u + 2\beta \delta u^T C_k u + u^T C_k u - 1 = 0$$

that corresponds to the intersection of $u + \beta \delta u$ and the ellipsoid (24).

VIII. Numerical Examples

The optimal control methods are applied to three systems, a quadrotor, a car-like robot, and an aerial vehicle with two manipulator arms.

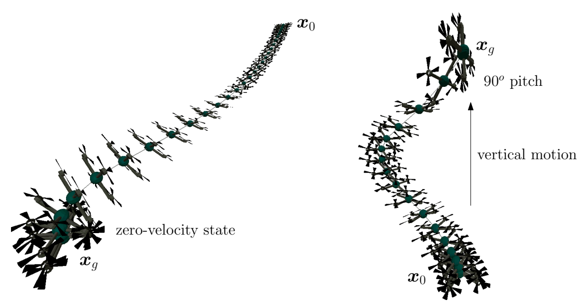

Fig. 1. Optimized quadrotor trajectories: a 10 meter long trajectory (left), and a vertical motion ending at 90^o pitch (right).

Quadrotor: It is useful to illustrate the method with the simple case of a single-rigid body system, i.e. $Q = G = SE(3)$, such as a quadrotor (Figure (1)). The body is described by its rotation R, position p, body-fixed angular velocity ω and linear velocity ν. The state $x = (q, v)$ consists of configuration and velocity given, respectively, by

$$q \equiv g = \begin{bmatrix} R & p \\ 0_{1\times3} & 1 \end{bmatrix}, \quad v \equiv V = (\omega, v).$$

There are no velocity constraints so trivially $S = I_6$ and the dynamics is defined using the standard terms:

$$M(q) = \begin{bmatrix} \mathbb{J} & 0 \\ 0 & mI_3 \end{bmatrix}, \quad b(q,v) = \begin{bmatrix} \omega \times \mathbb{J}\omega \\ m\omega \times \nu \end{bmatrix},$$

$$B(q) = \begin{bmatrix} 0 & -lk_t & 0 & lk_t \\ -lk_t & 0 & lk_t & 0 \\ k_m & -k_m & k_m & -k_m \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ k_t & k_t & k_t & k_t \end{bmatrix},$$

where $\mathbb{J}$ denotes the inertia tensor and m is the mass, and k_t, k_m, l are constants. There are four control inputs corresponding to the squared magnitude of propeller rotor speeds. For given $x = (q, v)$ and $\eta = (V, a)$, the retraction is

$$x\varphi(\eta) = (q\text{cay}(V), v + a),$$

while its inverse becomes

$$\varphi^{-1}(x_1^{-1}x_2) = \left(\text{cay}^{-1}(q_1^{-1}q_2), v_2 - v_1\right).$$

The cost function is constructed using

$$L_k(x,u) = \frac{1}{2}v^T Q_v v + \frac{1}{2}u^T R_u u,$$
$$L_N(x) = \frac{1}{2}\|\varphi^{-1}(x_f^{-1}x)\|^2_{Q_f},$$
$$G_k(x,u) = u - u_{\max},$$

where $Q_v \geqslant 0$ is a constant matrix penalizing high velocity, $R_u > 0$ is a constant matrix penalizing control, $Q_f = \text{diag}(Q_{q_f}, Q_{v_f}) \geqslant 0$ is a constant matrix penalizing deviation from a desired final state $x_f = (q_f, v_f)$. The inequality constraint encode control bounds.

To implement either SQP, SN, or DDP the state gradients (22) and (23) are computed according to

$$\bar{\nabla}_x L_k(x,u) = \begin{bmatrix} 0 \\ Q_v v \end{bmatrix}, \quad \bar{\nabla}_x^2 L_k(x,u) = \text{diag}(0, Q_v),$$
$$\bar{\nabla}_x L_N(x) = \begin{bmatrix} (\text{dcay}^{-1}(-\Delta_q))^T Q_{q_f}\Delta_q \\ Q_{v_f}(v_N - v_f) \end{bmatrix},$$
$$\bar{\nabla}_x^2 L_N(x) \approx \begin{bmatrix} (\text{dcay}^{-1}(-\Delta_q))^T Q_{q_f} \text{dcay}^{-1}(-\Delta_q) & 0 \\ 0 & Q_{v_f} \end{bmatrix},$$

where $\Delta_q \triangleq \text{cay}^{-1}(q_f^{-1}q)$ with the Cayley map and its derivatives given by (9), (10), (18).

Airbot: Motivated by recent progress in aerial robotics we consider trajectory generation of an articulated flying mechanisms capable of performing aerial manipulation tasks [34], [35], [36], [37], [38]. The vehicle is termed "airbot" in the comparison table. It is modeled using the holonomic multi-body model with inertia matrix computed according to (4). The aerial robot shown in Figure 2 has three pairs of propellers fixed onto three spokes at 120 degrees. Two three-link manipulators are suspended from the vehicle and can extend forward and sideways. Such an arrangement enables the manipulator tips to extend beyond the vehicle perimeter which enables interesting reaching and grabbing maneuvers. The setup is similar to the quadrotor with additional cost terms achieving desired joint angles. No contact forces were simulated.

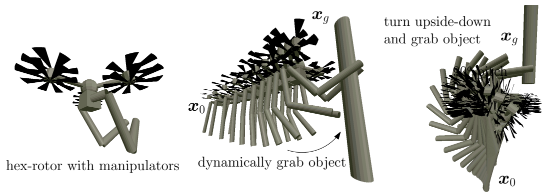

Fig. 2. Optimized aerial manipulating robot trajectories.

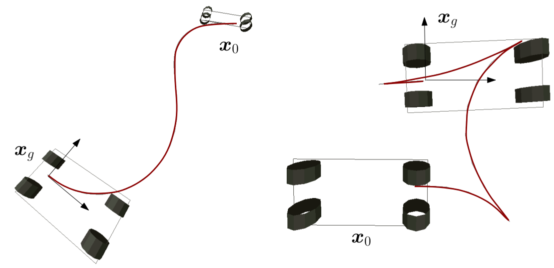

Fig. 3. Optimized car trajectories: a 10 meter long trajectory (left), and a parallel motion maneuver (right).

Simple car: Consider a simple planar second-order model with $Q = SE(2) \times \mathbb{R}^1$ and configuration $q = (g, \phi)$ where $g \in SE(2)$ is the pose and ϕ denotes the steering angle. A basic rear-wheel-drive no-sideslip wheeled vehicle model is obtained using the velocities $v = (\nu, \dot{\phi})$ where ν is the forward body-fixed velocity and ϕ is the steering rate using

$$S(q) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ \tan\phi/\ell & 0 \\ 0 & 1 \end{bmatrix},$$

where ℓ is the distance between the front and rear wheel axles. While the real system could have very complicated dynamics it is common for control purposes to either ignore the dynamics [39], [40] or use the simple dynamic extension

$$\dot{\nu} = u_1, \qquad \dot{\phi} = u_2.$$

For the purpose of illustrating optimal control algorithms we employ such simplified dynamics. Similar cost function to the quadrotor model was used with bounds on the controls.

The following table summarizes the required computational times per iteration, number of iterations til reasonable convergence, and the number of time-steps N employed. The quadrotor and car model have the ability to run in real time and converge in less than a few milliseconds. The multi-body system model requires finer resolution since it has unstable dynamics and currently requires a few seconds to converge.

System	CPU time / iter.	total iter.	time-steps N
Quadrotor	500 us	20	64
Airbot	159 ms	50	256
Car	89 us	20	64

IX. Conclusions

This paper developed optimal control methods for systems evolving on Lie groups. By specifying general Lie differential operations it was possible to apply existing methods such as stage-wise Newton and differential dynamic programming intrinsically on the Lie group manifold. Several examples illustrate the algorithm operation and its computational efficiency.

ACKNOWLEDGEMENT

The author would like to thank the anonymous reviewers for their useful feedback.

REFERENCES

[1] A. Mueller and Z. Terze, "Differential-geometric modelling and dynamic simulation of multibody systems," *Journal for Theory and Application in Mechanical Engineering*, vol. 51, no. 6, 2009.

[2] F. Park, J. Bobrow, and S. Ploen, "A lie group formulation of robot dynamics," *The International Journal of Robotics Research*, vol. 14, no. 6, pp. 609–618, 1995. [Online]. Available: http://ijr.sagepub.com/content/14/6/609.abstract

[3] J. Park and W.-K. Chung, "Geometric integration on euclidean group with application to articulated multibody systems," *Robotics, IEEE Transactions on*, vol. 21, no. 5, pp. 850 – 863, oct. 2005.

[4] A. Mueller and P. Maisser, "A lie-group formulation of kinematics and dynamics of constrained mbs and its application to analytical mechanics," *Multibody System Dynamics*, vol. 9, pp. 311–352, 2003.

[5] C. Altafini, "Reduction by group symmetry of second order variational problems on a semidirect product of Lie groups with positive definite Riemannian metric," *ESAIM: Control, Optimisation and Calculus of Variations*, vol. 10, pp. 526–548, 2004.

[6] A. M. Bloch, P. E. Crouch, J. E. Marsden, and A. K. Sanyal, "Optimal control and geodesics on quadratic matrix lie groups," *Found. Comput. Math.*, vol. 8, no. 4, pp. 469–500, Jul. 2008. [Online]. Available: http://dx.doi.org/10.1007/s10208-008-9025-1

[7] P. Crouch, G. Kun, and F. S. Leite, "The de casteljau algorithm on Lie groups and spheres," *Journal of Dynamical and Control Systems*, vol. 5, no. 3, pp. 387–429, 1999.

[8] L. Noakes, "Null cubics and Lie quadratics," *Journal of Mathematical Physics*, vol. 44, no. 3, pp. 1436–1448, 2003.

[9] A. M. Bloch, I. I. Hussein, M. Leok, and A. K. Sanyal, "Geometric structure-preserving optimal control of a rigid body," *Journal of Dynamical and Control Systems*, vol. 15, no. 3, pp. 307–330, 2009.

[10] M. Kobilarov and J. Marsden, "Discrete geometric optimal control on Lie groups," *IEEE Transactions on Robotics*, vol. 27(4), pp. 641–655, 2011.

[11] F. Jimenez, M. Kobilarov, and D. Martin de Diego, "Discrete variational optimal control," *Journal of Nonlinear Science*, pp. 1–34, 2012. [Online]. Available: http://dx.doi.org/10.1007/s00332-012-9156-z

[12] A. Saccon, J. Hauser, and A. Aguiar, "Optimal control on lie groups: The projection operator approach," *Automatic Control, IEEE Transactions on*, vol. 58, no. 9, pp. 2230–2245, 2013.

[13] J. E. Marsden and T. S. Ratiu, *Introduction to Mechanics and Symmetry*. Springer, 1999.

[14] R. Featherstone, *Rigid Body Dynamics Algorithms*. Springer, 2008.

[15] A. Jain, *Robot and Multibody Dynamics: Analysis and Algorithms*. Springer, 2011.

[16] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. CRC, 1994.

[17] A. Iserles, H. Z. Munthe-Kaas, S. P. Nørsett, and A. Zanna, "Lie group methods," *Acta Numerica*, vol. 9, pp. 215–365, 2000.

[18] E. Celledoni and B. Owren, "Lie group methods for rigid body dynamics and time integration on manifolds," *Comput. meth. in Appl. Mech. and Eng.*, vol. 19, no. 3,4, pp. 421–438, 2003.

[19] M. Leok, "Foundations of computational geometric mechanics," Ph.D. dissertation, California Institute of Technology, 2004.

[20] J. C. Simo, N. Tarnow, and K. K. Wong, "Exact energy-momentum conserving algorithms and symplectic schemes for nonlinear dynamics," *Computer Methods in Applied Mechanics and Engineering*, vol. 100, pp. 63–116, 1992.

[21] E. Hairer, C. Lubich, and G. Wanner, *Geometric Numerical Integration*, ser. Springer Series in Computational Mathematics. Springer-Verlag, 2006, no. 31.

[22] G. S. Chirikjian, *Stochastic models, information theory, and Lie groups. Volume II: Analytic methods and modern applications (to appear).* Applied and Numerical Harmonic Analysis. Basel: Birkhäuser. xxiv, 436 p., 2012.

[23] F. A. Potra, M. Anitescu, B. Gavrea, and J. Trinkle, "A linearly implicit trapezoidal method for integrating stiff multibody dynamics with contact, joints, and friction," *International Journal for Numerical Methods in Engineering*, vol. 66, no. 7, pp. 1079–1124, 2006. [Online]. Available: http://dx.doi.org/10.1002/nme.1582

[24] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds*. Princeton, NJ: Princeton University Press, 2008.

[25] A. Saccon, A. Aguiar, and J. Hauser, "Lie group projection operator approach: Optimal control on t so(3)," in *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, Dec 2011, pp. 6973–6978.

[26] D. P. Bertsekas, *Nonlinear Programming, 2nd ed.* Belmont, MA: Athena Scientific, 2003.

[27] D. H. Jacobson and D. Q. Mayne, *Differential dynamic programming*, ser. Modern analytic and computational methods in science and mathematics. New York: Elsevier, 1970. [Online]. Available: http://opac.inria.fr/record=b1078358

[28] E. Mizutani and H. L. Dreyfus, "Stagewise newton, differential dynamic programming, and neighboring optimum control for neural-network learning," in *American Control Conference, 2005. Proceedings of the 2005*, 2005, pp. 1331–1336 vol. 2.

[29] E. Todorov and W. Li, "A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems," in *American Control Conference, 2005. Proceedings of the 2005*, 2005, pp. 300–306 vol. 1.

[30] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, 2012, pp. 4906–4913.

[31] J. Morimioto, G. Zeglin, and C. Atkeson, "Minimax differential dynamic programming: application to a biped walking robot," in *SICE 2003 Annual Conference*, vol. 3, 2003, pp. 2310–2315 Vol.3.

[32] E. Theodorou, Y. Tassa, and E. Todorov, "Stochastic differential dynamic programming," in *American Control Conference (ACC), 2010*, 2010, pp. 1125–1132.

[33] J. Van Den Berg, S. Patil, and R. Alterovitz, "Motion planning under uncertainty using iterative local optimization in belief space," *Int. J. Rob. Res.*, vol. 31, no. 11, pp. 1263–1278, Sep. 2012. [Online]. Available: http://dx.doi.org/10.1177/0278364912456319

[34] P. E. I. Pounds, D. R. Bersak, and A. M. Dollar, "Stability of small-scale uav helicopters and quadrotors with added payload mass under pid control," *Auton. Robots*, vol. 33, no. 1-2, pp. 129–142, 2012.

[35] C. Korpela, M. Orsag, T. Danko, B. Kobe, C. McNeil, R. Pisch, and P. Oh, "Flight stability in aerial redundant manipulators," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, may 2012, pp. 3529 –3530.

[36] J. Thomas, J. Polin, K. Sreenath, and V. Kumar, "Avian-inspired grasping for quadrotor micro uavs," in *ASME International Design Engineering Technical Conference (IDETC)*, to appear, 2013.

[37] A. Torre, D. Mengoli, R. Naldi, F. Forte, A. Macchelli, and L. Marconi, "A prototype of aerial manipulator," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, 2012, pp. 2653–2654.

[38] A. Jimenez-Cano, J. Martin, G. Heredia, R. Cano, and A. Ollero, "Control of an aerial robot with multi-link arm for assembly tasks," in *International Conference on Robotics and Automation*, 2013.

[39] B. Nagy and A. Kelly, "Trajectory Generation for Car-Like Robots Using Cubic Curvature Polynomials," 2001.

[40] A. Luca, G. Oriolo, and C. Samson, "Feedback control of a nonholonomic car-like robot," in *Robot Motion Planning and Control*, ser. Lecture Notes in Control and Information Sciences, J.-P. Laumond, Ed. Springer Berlin Heidelberg, 1998, vol. 229, pp. 171–253. [Online]. Available: http://dx.doi.org/10.1007/BFb0036073