Online Feedback Control for Input-Saturated Robotic Systems on Lie Groups

Taosha Fan and Todd Murphey
Department of Mechanical Engineering, Northwestern University
Email: taosha.fan@u.northwestern.edu, t-murphey@northwestern.edu

Abstract—In this paper, we propose an approach to designing online feedback controllers for input-saturated robotic systems evolving on Lie groups by extending the recently developed Sequential Action Control (SAC). In contrast to existing feedback controllers, our approach poses the nonconvex constrained nonlinear optimization problem as the tracking of a desired negative mode insertion gradient on the configuration space of a Lie group. This results in a closed-form feedback control law even with input saturation and thus is well suited for online application. In extending SAC to Lie groups, the associated mode insertion gradient is derived and the switching time optimization on Lie groups is studied. We demonstrate the efficacy and scalability of our approach in the 2D kinematic car on $SE(2)$ and the 3D quadrotor on $SE(3)$. We also implement iLQG on a quadrator model and compare to SAC, demonstrating that SAC is both faster to compute and has a larger basin of attraction.

Index-Terms— sequential action control; Lie group; online feedback control; input saturation

I. INTRODUCTION

Dynamic robotic tasks require suitable feedback controllers to simultaneously manage dynamics, kinematics, input saturation etc.. In the last thirty years, a number of techniques to design stabilizing control laws have been developed in robotics and control community, such as feedback linearization and backstepping [19, 22], linearizing the dynamics around a nominal trajectory to implement a linear quadratic regulator (LQR) [2] or linear model predicative control (LMPC) [5], dynamic programming [7] and nonlinear model predicative control (NMPC) [1, 27, 32, 34], or feedback motion planning via semidefinite programming to precompute a library of control policies filling up the space of possible initial states [18, 24, 33]. However, feedback linearization and backstepping are generally inapplicable when actuation limits exist while LQR and LMPC only work in a small neighborhood of the nominal trajectory. Dynamic programming and NMPC are usually computationally expensive and feedback motion planning via semidefinite programming may need substantial time and memory to compute and store the library of control policies if the state space is high-dimensional and noncompact.

Generally speaking, it is preferable to model robot locomotion systems, e.g., vehicular systems, on Lie groups rather than in coordinates since there are no singularity problems. Despite the development of numerous feedback controllers for systems evolving on Lie groups [17, 21, 23, 31, 37], all of them remain in the category of aforementioned techniques. Furthermore, feedback controllers on Lie groups resulting from existing control techniques tend to be much more complicated than their coordinate counterparts and as a result, might not be available for time-critical tasks in the presence of actuation limits. Unfortunately, nearly all robot locomotion tasks are required to manage input saturation while being executed in real time. Hence the application of these techniques [17, 21, 23, 31, 37] is restricted in practice.

In this paper, we address the shortcomings of existing control techniques by proposing a novel approach to designing online feedback controllers for systems evolving on Lie groups. Our approach is inspired by the recently developed *Sequential Action Control* (SAC) [3]. In contrast to existing control techniques, SAC improves the long-time performance of the controller by tracking a desired negative *mode insertion gradient* and the resulting control action has a *closed-form solution* whose calculation is almost as inexpensive as simulation. More importantly, control actions in SAC can be efficiently saturated and hence it is well suited for robotic tasks in real time. SAC has solved various benchmark problems, such as cart-pendulum, pendubot, acrobot, spring-loaded inverted pendulum etc. in [3, 4, 35, 36]. Nevertheless, up till now, the application of SAC remains limited to nonlinear systems in coordinates and the purpose of this paper is to extend SAC to Lie groups so that we may synthesize control techniques directly on the Lie groups that are commonly used in robotics.

A. Relationship to Existing Control Techniques

The major difference between SAC and other optimization-based nonliner control techniques [1, 5, 7, 18, 24, 27, 32, 33] is that expensive iterative computations of gradients and Hessians are avoided. Control inputs in SAC are computed and saturated based on the mode insertion gradient (Eq. (3)), whose argument is the time duration λ instead of control inputs $u(t)$, and which can be exactly evaluated (not approximated) by Eq. (4). Moreover, most gradient-based methods such as DDP or iLQG attempt to update control inputs over the whole control horizon, whereas SAC only updates the next control input at a specified future time τ for a duration λ.

Although SAC performs well for various nonlinear systems both in coordinates and on Lie groups as these shown in [3, 4, 35, 36] and Section V, we are not intending to imply that SAC is superior to existing control techniques [1, 2, 5, 7, 18, 19, 22, 24, 27, 33]. Instead we are more interested in how SAC and existing control techniques can complement each other across a state space during online computation.

As far as we know, SAC augments existing control techniques in the following ways:

1) SAC can drive a nonlinear system to a region where feedback linearization, backstepping, LQR, LMPC etc. can be used for final stabilization even with input saturation.
2) Control actions computed by SAC can be used to initialize optimization-based techniques, e.g., NMPC, for which a reasonable initialization is of great significance.
3) In combination with feedback motion algorithms via semidefnitie programming [18, 24, 33], which requires one to precompute a library of control policies, SAC serves as an early-stage online feedback motion planner to drive the nonlinear system to the target set of initial states that are covered by the feedback motion algorithms via semidefinite programming with a library of much smaller size so that precomputational time and storage memory can be saved.

B. Contribution and Organization

The main contribution of this paper is extending the concept of sequential action control from coordinates to Lie groups so that feedback controllers can be constructed for input-saturated robotics systems evolving on Lie groups. An advantage of our approach over SAC in coordinates is that our approach does not require one to choose coordinates so that the expensive chart-switching and the associated sudden jumps are avoided. In addition to extending SAC to Lie groups, switching time optimization on Lie groups is studied and the first order derivative, i.e., the mode transition gradient, is derived.

The remainder of this paper is organized as following: Section II reviews sequential action control in coordinates and its properties. Section III considers switching time optimization for systems evolving on Lie groups and derives the associated mode insertion and transition gradients. Section IV analyzes the logarithm-constructed quadratic functions on Lie groups and presents exact expressions of $d\exp(\xi)$ and $d\exp^{-1}(\xi)$ for $SO(3)$, $SE(2)$ and $SE(3)$. Section V demonstrates the efficacy and scalability of our approach on two classic robotic systems evolving on Lie groups and Section VI concludes the paper.

II. SEQUENTIAL ACTION CONTROL

Sequential Action Control is an online model-based approach to high-quality trajectory planning and tracking for nonlinear systems [3]. Compared with general NMPC methods which minimize the objective iteratively over a finite horizon while enforcing system dynamics as constraints, SAC computes control actions by tracking a desired negative mode insertion gradient, which is a control methodology common in hybrid systems. Since control actions in SAC are assumed to be applied in an infinitesimal duration $\lambda \to 0^+$, there exists a closed-form solution for control actions with actuation limits; this makes SAC well suited for online application. In this section, we review SAC in coordinates and discuss how to extend SAC to Lie groups.

A. An Overview of Sequential Action Control

Consider the affine nonlinear system

$$f\Big(x(t), u(t); t\Big) = g\Big(x(t); t\Big) + h\Big(x(t); t\Big) \cdot u(t) \quad (1)$$

where the state $x(t) \in \mathbb{R}^n$, control action $u(t) \in \mathbb{R}^m$, $g\Big(x(t); t\Big) \in \mathbb{R}^n$ and $h\Big(x(t); t\Big) \in \mathbb{R}^{n \times m}$. Let

$$J_1 = \int_{T_0}^{T_N} L\Big(g(\tau)\Big) d\tau + \varphi\Big(g(T_N)\Big)$$

be the objective and suppose the dynamics $f\Big(x(t), u(t); t\Big)$ first switches from the nominal mode

$$f_1\Big(x(t); t\Big) = f\Big(x(t), u_1(t); t\Big)$$

to the new mode

$$f_2\Big(x(t); t\Big) = f\Big(x(t), u_2(t); t\Big)$$

at time τ_0 and then back to $f_1\Big(x(t), u_1(t); t\Big)$ again after a duration of λ, i.e.,

$$u(t) = \begin{cases} u_1(t) & t > \tau_0 + \lambda \ \text{ or } \ t \le \tau_0, \\ u_2(t) & \tau_0 \le t < \tau_0 + \lambda. \end{cases} \quad (2)$$

If the duration $\lambda \to 0^+$ is infinitesimal, the *mode insertion gradient* $\dfrac{dJ_1}{d\lambda}\Big|_{0^+}$ is defined to be

$$\frac{dJ_1}{d\lambda}\Big|_{0^+} = \lim_{\lambda \to 0^+} \frac{\Delta J_1}{\lambda} \quad (3)$$

where ΔJ_1 is the objective change yielded by this switch. The mode insertion gradient $\dfrac{dJ_1}{d\lambda}\Big|_{0^+}$ can be calculated exactly with Eq. (4) [11]

$$\frac{dJ_1}{d\lambda}\Big|_{0^+} = \rho(\tau_0)^T \cdot \Big[f_2\Big(x(\tau_0); \tau_0\Big) - f_1\Big(x(\tau_0); \tau_0\Big)\Big] \quad (4)$$

where $\rho(t)$ are costates satisfying

$$\dot{\rho}(s)^T = -\rho(s)^T \cdot D_x f_1\Big(x(s); s\Big) - D_x L\Big(x(s)\Big) \quad (5)$$

$$\text{subject to: } \rho(T_N) = D_g \varphi\Big(g(T_N)\Big).$$

By continuity, the objective change ΔJ_1 can be approximated with

$$\Delta J_1 \approx \frac{dJ_1}{d\lambda}\Big|_{0^+} \cdot \lambda$$

if the duration λ is small (but not infinitesimal). So for sufficiently small $\lambda > 0$, J_1 is reduced in the switch of Eq. (2) as long as the mode insertion gradient $\dfrac{dJ_1}{d\lambda}\Big|_{0^+}$ is negative.

The idea of SAC is to compute control actions $u^*(t)$ reducing the objective J_1 by tracking a desired negative mode insertion gradient $\dfrac{dJ_1}{d\lambda}\Big|_{0^+} = \alpha_d$ instead of optimizing J_1 iteratively. In tracking the negative mode insertion gradient α_d, SAC incorporates feedback and computes control actions

Algorithm 1 Sequential Action Control

1: Simulate states and costates $(x(t), \rho(t))$ with nominal control dynamics $f_1(t)$
2: Compute $u_2^*(t)$ from $(x(t), \rho(t))$ by

$$u_2^*(t) = u_1 + (\Lambda + R)^{-1} h\Big(x(t); t\Big)^T \rho(t) \alpha_d$$

3: Modify $u_2^*(t)$ to satisfy the control bounds $u_{\min}$ and $u_{\max}$
4: Determine the mode insertion time τ_0 to maximize the negative mode insertion gradient
5: Determine the duration λ to apply $u_2^*(t)$

---

that guarantee an optimal/near-optimal reduction of J_1, which eventually results in a long-time improvement.

The mode insertion gradient tracking can be formulated as an adjoint optimization problem

$$u_2^*(t) = \arg\min_{u_2(t)} \int_{T_0}^{T_N} \left[\left(\frac{dJ_1}{d\lambda} \Big|_{0^+} - \alpha_d \right)^2 + \\ \|u_2(s) - u_1(s)\|_R^2 \right] ds \quad (6)$$

where $R \in \mathbb{R}^{m \times m}$ is symmetric positive-definite. For affine system Eq. (1), it has been shown in [3] that Eq. (6) has a closed-form solution

$$u_2^*(t) = u_1 + (\Lambda + R)^{-1} h\Big(x(t); t\Big)^T \rho(t) \alpha_d \quad (7)$$

where $\Lambda := h\Big(x(t); t\Big)^T \rho(t) \rho(t)^T h\Big(x(t); t\Big)$.

As shown in Algorithm 1, control actions can be saturated in SAC and this is discussed in detail in Section II-B. For these not discussed, such as determining mode insertion time τ_0 and control duration λ, interested readers can refer to [3] for a complete introduction.

B. Saturating Control Actions

If actuation limits exist, the adjoint optimization problem Eq. (6) can be reformulated by quadratic programming with

$$u_{\min}(t) \leq u_2^*(t) \leq u_{\max}(t)$$

imposed as constraints. This is not preferable considering the high computational expense. Instead, if $R \in \mathbb{R}^{m \times m}$ is diagonal and let $u'^*_2(t) = u_1 + (\Lambda + R)^{-1} h\Big(x(t); t\Big)^T \rho(t) \alpha_d$, then $u_2^*(t)$ can be saturated elementwise by

$$[u_2^*(t)]_i = \begin{cases} [u_{\min}]_i & [u'^*_2(t)]_i < [u_{\min}]_i, \\ [u'^*_2(t)]_i & [u_{\min}]_i \leq [u'^*_2(t)]_i \leq [u_{\max}]_i, \\ [u_{\max}]_i & [u'^*_2(t)]_i > [u_{\max}]_i. \end{cases} \quad (8)$$

Despite that $u_2^*(t)$ determined by Eq. (8) may not be optimal to Eq. (7), it works very well in practice and guarantees a reduction of J_1 if $u_{\min} < u_1(t) < u_{\max}$. Most importantly, the computational expense introduced by Eq. (8) is almost negligible in contrast to quadratic programming.

C. Extension of Sequential Action Control to Lie Groups

As we can see from the overview above, to extend sequential action control to Lie groups, the first problem is calculating the mode insertion gradient on Lie groups. Unfortunately, Eqs. (4) and (5) may no longer hold for nonabelian Lie groups, meaning that the mode insertion gradient needs to be rederived. We study this problem in Section III, which is the fundamental contribution of this paper. Another problem, though not so obvious, is choosing appropriate objective functions. Though quadratic functions can be constructed easily on Lie groups through the logarithm map, it is difficult to evaluate their derivatives if there are no exact expressions of the trivialized tangent inverse of the exponential map $\mathrm{d}\exp^{-1}(\xi)$, which is discussed in detail in Section IV.

III. MODE INSERTION AND TRANSITION GRADIENTS FOR SYSTEMS EVOLVING ON LIE GROUPS

In this section we study the switching time optimization for systems evolving on Lie groups and derive the associated mode insertion and transition gradients. Readers are assumed to have a basic knowledge of Lie group theory, which can be found in a variety of textbooks, e.g., [16].

A. Review of Lie Group Dynamics and Linearization

To be complete, we first define the following operator that is frequently used in this paper.

Definition 1. Given a differentiable map $f : G \longrightarrow \mathbb{R}^n$ on a Lie group G, then $D_g f|_g : \mathfrak{g} \longrightarrow \mathbb{R}^n$ is defined to be the linear map such that

$$D_g f|_g \cdot \eta = \frac{d}{ds} f\Big(g \cdot \exp(s \cdot \eta)\Big)\Big|_{s=0} \quad (9)$$

for all $\eta \in \mathfrak{g}$.

If G is abelian, then $D_g f|_g$ is just the first-order derivative in multi-variable calculus. For brevity, the dependence $|_g$ is dropped and $D_g f|_g$ is simply written as $D_g f$.

Let G be the Lie group and $\mathfrak{g}$ the associated Lie algebra, the continuous system evolving on G is defined to be

$$\dot{g}(t) = g(t)\xi(t). \quad (10a)$$

$$\xi(t) = f\Big(g(t), u(t); t\Big) \quad (10b)$$

where $f : G \times \mathcal{U} \times \mathbb{R}^+ \longrightarrow \mathfrak{g}$. In the remainder of the paper, we may sometimes write $f\Big(g(t), u(t); t\Big)$ as $f(t)$ for simplicity.

If $f\Big(g(t), u(t); t\Big)$ is C^1, it is possible to linearize the corresponding system, which is as follows [29].

Proposition 1. If $f : G \times \mathcal{U} \times \mathbb{R}^+ \longrightarrow \mathfrak{g}$ is piecewise differentiable, then a linearization of continuous system Eq. (10) is

$$\dot{\eta}(t) = \Big(D_g f(t) - \mathrm{ad}_{f(t)}\Big) \cdot \eta(t) + D_u f(t) \cdot \nu(t) \quad (11)$$

where $\eta(t) = g(t)^{-1} \delta g(t) \in \mathfrak{g}$ and $\nu(t) \in \mathcal{U}$ is the variation of inputs $u(t)$.

For more details about Lie group dynamics and linearization, interested readers can refer to [8, 13, 25, 26].

B. Mode Transition Gradient

Definition 2. If the inputs $u(t) \in \mathcal{U}$ in Eq. (10) are transition times

$$\mathcal{T} = \begin{bmatrix} T_1 & T_2 & \cdots & T_{N-1} \end{bmatrix}^T,$$

such that

$$f\Big(g(t), \mathcal{T}; t\Big) = \Big[1(t - T_0) - 1(t - T_1^-)\Big] f_1\Big(g(t); t\Big) +$$
$$\Big[1(t - T_1^+) - 1(t - T_2^-)\Big] f_2\Big(g(t); t\Big) + \cdots \cdots +$$
$$\Big[1(t - T_{N-1}^+) - 1(t - T_N^-)\Big] f_N\Big(g(t); t\Big) \quad (12)$$

where $T_0 \leq T_1 \leq \cdots \leq T_{N-1} \leq T_N$ and $1(t) : \mathbb{R} \longrightarrow \{0, 1\}$ is the unit step function, then Eq. (10) is a *switched system on the Lie group* G and $f_i\Big(g(t); t\Big) : G \times \mathbb{R}^+ \longrightarrow \mathfrak{g}$ are *mode functions*.

The optimization problem to determine transition times for switched system is called *switching time optimization*. The first- and second-order derivatives of switching time optimization can be evaluated exactly for switched system in coordinates [9, 11, 12]. However, to our knowledge, no work has been done for switching time optimization on Lie groups. Though switching time optimization is not the main focus of this paper, the *mode transition gradient*, i.e. the first-order derivative of the switching time optimization, is closely related to the mode insertion gradient that SAC tracks in computing control actions $u_2^*(t)$. Here we derive the mode transition gradient for systems evolving on Lie groups that is used to derive the mode insertion gradient in Section III-C.

Lemma 1. Suppose $J(\mathcal{T})$ is an objective function of the switched system evolving on Lie group G defined by Eqs. (10) and (12)

$$J(\mathcal{T}) = \int_{T_0}^{T_N} L\Big(g(\tau)\Big) \, d\tau + \varphi\Big(g(T_N)\Big). \quad (13)$$

Provided each $f_i(\cdot; \cdot)$ in f is C^1, then the mode transition gradient $\dfrac{dJ}{d\mathcal{T}}\Big|_{\mathcal{T}}$ of $J(\mathcal{T})$ is

$$\frac{dJ}{d\mathcal{T}}\Big|_{\mathcal{T}} = \begin{bmatrix} \dfrac{dJ}{dT_1} & \dfrac{dJ}{dT_2} & \cdots & \dfrac{dJ}{dT_{N-1}} \end{bmatrix}^T\Big|_{\mathcal{T}}$$

such that

$$\frac{dJ}{dT_i}\Big|_{\mathcal{T}} = \rho(T_i)^T \Big[f_i\Big(g(T_i); T_i\Big) - f_{i+1}\Big(g(T_i), T_i\Big) \Big] \quad (14)$$

where

$$\dot{\rho}(s)^T = -\rho(s)^T \cdot \Big[D_g f\Big(g(s), \mathcal{T}; s\Big) - $$
$$\mathrm{ad}_{f(g(s), \mathcal{T}; s)} \Big] - D_g L\Big(g(s)\Big) \quad (15)$$

$$\text{subject to: } \rho(T_N) = D_g \varphi\Big(g(T_N)\Big).$$

Proof: The derivative $\dfrac{dJ}{dT_i}\Big|_{\mathcal{T}}$ of $J(\mathcal{T})$ w.r.t T_i is

$$\frac{dJ}{dT_i}\Big|_{\mathcal{T}} = \int_{T_0}^{T_N} D_g L\Big(g(\tau)\Big) \cdot D_{T_i} g(\tau) d\tau +$$
$$D_g \varphi\Big(g(T_N)\Big) \cdot D_{T_i} g(T_N) \quad (16)$$

where $D_{T_i} g(t) = g(t)^{-1} \dfrac{dg(t)}{dt} \in \mathfrak{g}$ and, according to Eq. (11), satisfies

$$\frac{\partial}{\partial t}\Big(D_{T_i} g(t)\Big) =$$
$$\Big(D_g f\Big(g(t), \mathcal{T}; t\Big) - \mathrm{ad}_{f(g(t), \mathcal{T}; t)}\Big) \cdot D_{T_i} g(t) +$$
$$D_{T_i} f\Big(g(t), \mathcal{T}; t\Big). \quad (17)$$

Note that $f_i(\cdot; \cdot)$ is C^1, thus $D_g f\Big(g(t), \mathcal{T}; t\Big)$ is well defined. Since that all initial states are fixed, we have

$$D_{T_i} g(0) = \mathbf{0}.$$

Therefore, the solution to Eq. (17) is

$$D_{T_i} g(t) = \int_{T_0}^{T_N} \Phi(t, s) \cdot D_{T_i} f\Big(g(s), \mathcal{T}; s\Big) ds, \quad (18)$$

where $\Phi(t, s)$ is the state transition matrix of Eq. (17). Substitute Eq. (18) to Eq. (16) and switch the integration order, $\dfrac{dJ}{dT_i}\Big|_{\mathcal{T}}$ becomes

$$\frac{dJ}{dT_i}\Big|_{\mathcal{T}} = \int_{T_0}^{T_N} \rho(s)^T \cdot D_{T_i} f\Big(g(s), \mathcal{T}; s\Big) ds \quad (19)$$

and the costate $\rho(s)^T$ is

$$\rho(s)^T = \int_s^{T_N} D_g L\Big(g(\tau)\Big) \Phi(\tau, s) d\tau +$$
$$D_g \varphi\Big(g(T_N)\Big) \Phi(T_N, s). \quad (20)$$

Taking derivative on both sides of Eq. (20), we have

$$\dot{\rho}(s)^T = -\rho(s)^T \Big[D_g f\Big(g(s), \mathcal{T}; s\Big) - $$
$$\mathrm{ad}_{f(g(s), \mathcal{T}; s)} \Big] - D_g L\Big(g(s)\Big), \quad (21)$$

$$\text{subject to: } \rho(T_N) = D_g \varphi\Big(g(T_N)\Big),$$

In addition, note that

$$D_{T_i} f\Big(g(t), \mathcal{T}; t\Big) = \delta(t - T_i^-) f_i\Big(g(t); t\Big) -$$
$$\delta(t - T_i^+) f_{i+1}\Big(g(t); t\Big). \quad (22)$$

Integrate δ-functions in Eq. (19) and the result is Eq. (14), which completes the proof. ∎

C. Mode Insertion Gradient

We have obtained the mode transition gradient for switching time optimization on Lie groups, with which the mode insertion gradient can be derived just as the following corollary indicates.

Corollary 1. Given a system evolving on Lie group G with $f : G \times \mathcal{U} \times \mathbb{R}^+ \longrightarrow \mathfrak{g}$ taking the form

$$f\Big(g(t), \lambda; t\Big) = \begin{cases} f_1\Big(g(t); t\Big), & t < \tau_0 \text{ or } t \geq \tau_0 + \lambda, \\ f_2\Big(g(t); t\Big), & \tau_0 \leq t < \tau_0 + \lambda \end{cases} \tag{23}$$

with the duration $\lambda \in \mathbb{R}^+$ as input and an objective function

$$J(\lambda) = \int_{T_0}^{T_N} L\Big(g(\tau)\Big) \, d\tau + \varphi\Big(g(T_N)\Big), \tag{24}$$

then the mode insertion gradient is

$$\frac{dJ}{d\lambda}\Big|_{\lambda=0^+} = \rho(\tau_0)^T \cdot \Big[f_2\Big(g(\tau_0); \tau_0\Big) - f_1\Big(g(\tau_0); \tau_0\Big) \Big] \tag{25}$$

where

$$\dot{\rho}(s)^T = -\rho(s)^T \cdot \Big[D_g f_1\Big(g(s); s\Big) - \\ \mathrm{ad}_{f_1(g(s); s)} \Big] - D_g L\Big(g(s)\Big) \tag{26}$$

$$\text{subject to: } \rho(T_N) = D_g \varphi\Big(g(T_N)\Big).$$

Proof: Let $\tau_1 = \tau_0 + \lambda$ and the dynamical system defined by Eq. (23) is a switched system with $\mathcal{T} = [\tau_1]$. Apply Lemma 1 and note $\frac{dJ}{d\lambda} = \frac{dJ}{d\tau_1}$, we have

$$\frac{dJ}{d\lambda} = \rho(\tau_0 + \lambda)^T \cdot \Big[f_2\Big(g(\tau_0 + \lambda); \tau_0 + \lambda\Big) - \\ f_1\Big(g(\tau_0 + \lambda); \tau_0 + \lambda\Big) \Big]. \tag{27}$$

Substituting $\lambda = 0^+$ into Eq. (27), the resulting equation is Eq. (25), which completes the proof. ∎

Now the mode insertion gradient can be exactly calculated for systems evolving on Lie groups by Eqs. (25) and (26), with which substituted into Eqs. (7) and (8), we have SAC extended to Lie groups.

IV. THE QUADRATIC FUNCTION ON LIE GROUPS AND ITS DERIVATIVE

The *quadratic function* in $\mathbb{R}^n$

$$f(x) = \frac{1}{2} \|x - x_d\|_M^2 \tag{28}$$

where $\|x\|_M = \sqrt{x^T M x}$ and $M \in \mathbb{R}^{n \times n}$ is symmetric positive definite defines a class of functions frequently used in control theory and application. Their essence as Lyapunov functions and easiness in numerical computation render quadratic objective functions quite popular for both SAC and optimization in $\mathbb{R}^n$. This suggests it may be necessary to reasonably extend quadratic functions to Lie groups.

A. Extending Quadratic Functions to Lie Groups

An immediate and common way in differential geometry to extend the quadratic function Eq. (28) in $\mathbb{R}^n$ to Lie groups is

$$f(g) = \frac{1}{2} \| \log(g_d^{-1} g) \|_M^2 \tag{29}$$

whose derivative is

$$D_g f = \mathrm{d} \exp^{-1} \Big(-\log(g_d^{-1} g) \Big)^T M \log(g_d^{-1} g), \tag{30}$$

where $\mathrm{d} \exp^{-1}(\xi)$ is the trivialized tangent inverse of the exponential map.

If Lie group G has a *bi-invariant pesudo-Riemannian metric*, Eq. (29) is a reasonable extension of quadratic functions to Lie groups in sense that $\log(g_d^{-1} g)$ is the initial velocity of the geodesic $\gamma : [0, 1] \longrightarrow G$ connecting g_d and g under the bi-invariant pesudo-Riemannian metric. Fortunately, there always exists such bi-invariant pesudo-Riemannian metrics for subgroups of $SE(3)$ [38], some of which actually turn out to be positive-definite, i.e., Riemannian metrics.

Though there are a number of other ways to extend quadratic functions from $\mathbb{R}^n$ to Lie groups, such as these in [20, 29], none of them have the same geometric meaning as Eq. (28) in $\mathbb{R}^n$ that captures the bi-invariant pesudo-Riemannian structure. Most importantly, according to numerical tests, when using Eq. (29) as the objective, SAC on Lie groups has a larger region of attraction and converges faster than objectives of other kinds.

B. Expressions of $\mathrm{d} \exp(\xi)$ and $\mathrm{d} \exp^{-1}(\xi)$ for Some Common Lie Groups

The trivialized tangent of the exponential map and its inverse $\mathrm{d} \exp(\xi)$ and $\mathrm{d} \exp^{-1}(\xi)$ are frequently used in the construction and linearization of Lie group integrators [13, 15, 29]. Since $\|\xi\|$ is usually small in Lie group integrators, it is possible to approximate $\mathrm{d} \exp(\xi)$ and $\mathrm{d} \exp^{-1}(\xi)$ by truncating the series expansions [15]

$$\mathrm{d} \exp(\xi) = \sum_{j=0}^{\infty} \frac{1}{(j+1)!} \mathrm{ad}_\xi^j, \quad \mathrm{d} \exp^{-1}(\xi) = \sum_{j=0}^{\infty} \frac{B_j}{j!} \mathrm{ad}_\xi^j$$

where B_j are Bernoulli numbers. However, when $\|\xi\|$ is no longer small, e.g., in an objective function Eq. (29), the approximation by truncating series expansions is invalid.

Here we give exact expressions (not series expansions) of $\mathrm{d} \exp(\xi)$ and $\mathrm{d} \exp^{-1}(\xi)$ for $SO(3)$, $SE(2)$ and $SE(3)$, all of which are commonly used in robotics.

1) $SO(3)$: Elements of Lie algebra $\hat{\omega} \in \mathfrak{so}(3)$ is usually associated with $\mathbb{R}^3$ through the hat operator $\wedge : \mathbb{R}^3 \longrightarrow \mathfrak{so}(3)$

$$\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}^{\wedge} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix},$$

then $\mathrm{d} \exp(\hat{\omega})$ and $\mathrm{d} \exp^{-1}(\hat{\omega})$ are [10]

$$\mathrm{d} \exp(\hat{\omega}) = \mathrm{I} + \frac{1 - \cos \|\omega\|}{\|\omega\|^2} \hat{\omega} + \frac{\|\omega\| - \sin \|\omega\|}{\|\omega\|^3} \hat{\omega}^2, \tag{31a}$$

$$\operatorname{d}\exp^{-1}(\hat{\omega}) = I - \frac{1}{2}\hat{\omega} + \frac{\frac{1}{2}\|\omega\|\sin\|\omega\| + \cos\|\omega\| - 1}{\|\omega\|^2(\cos\|\omega\| - 1)}\hat{\omega}^2. \tag{31b}$$

2) $SE(2)$: We represent $\xi \in \mathfrak{se}(2)$ in terms of coordinate $\begin{bmatrix} \omega \\ u \\ v \end{bmatrix} \in \mathbb{R}^3$ such that $\xi = \begin{bmatrix} 0 & -\omega & u \\ \omega & 0 & v \\ 0 & 0 & 0 \end{bmatrix}$. The exact expressions of $\operatorname{d}\exp(\hat{\omega})$ and $\operatorname{d}\exp^{-1}(\hat{\omega})$ are

$$\operatorname{d}\exp(\xi) = \begin{bmatrix} 1 & 0 & 0 \\ \#1 & \dfrac{\sin\omega}{\omega} & -\dfrac{1-\cos\omega}{\omega} \\ \#2 & \dfrac{1-\cos\omega}{\omega} & \dfrac{sin\omega}{\omega} \end{bmatrix} \tag{32a}$$

$$\operatorname{d}\exp^{-1}(\xi) = \begin{bmatrix} 1 & 0 & 0 \\ \#3 & \dfrac{\omega}{2}\dfrac{\sin\omega}{1-\cos\omega} & \dfrac{\omega}{2} \\ \#4 & -\dfrac{\omega}{2} & \dfrac{\omega}{2}\dfrac{\sin\omega}{1-\cos\omega} \end{bmatrix} \tag{32b}$$

where

$$\#1 = \frac{u(\omega - \sin\omega) + v(1 - \cos\omega)}{\omega^2},$$
$$\#2 = \frac{v(\omega - \sin\omega) - u(1 - \cos\omega)}{\omega^2},$$
$$\#3 = -\frac{v}{2} + \frac{u}{2}\frac{\omega\sin\omega + 2\cos\omega - 2}{\omega\cos\omega - \omega},$$
$$\#4 = \frac{u}{2} + \frac{v}{2}\frac{\omega\sin\omega + 2\cos\omega - 2}{\omega\cos\omega - \omega}.$$

3) $SE(3)$: It is standard to identify $\xi \in \mathfrak{se}(3)$ with the body-fixed velocity $\begin{bmatrix} \omega \\ v \end{bmatrix} \in \mathbb{R}^6$ through $\xi = \begin{bmatrix} \hat{\omega} & v \\ \mathbf{0} & 0 \end{bmatrix}$ where $\omega, v \in \mathbb{R}^3$ are respectively the body-fixed angular and linear velocities, and we have

$$\operatorname{d}\exp(\xi) = \begin{bmatrix} \#1 & \mathbf{O}_{3\times3} \\ \#2 & \#1 \end{bmatrix}, \tag{33a}$$

$$\operatorname{d}\exp^{-1}(\xi) = \begin{bmatrix} \#3 & \mathbf{O}_{3\times3} \\ \#4 & \#3 \end{bmatrix} \tag{33b}$$

where

$$\#1 = I + \frac{1 - \cos\|\omega\|}{\|\omega\|^2}\hat{\omega} + \frac{\|\omega\| - \sin\|\omega\|}{\|\omega\|^3}\hat{\omega}^2,$$

$$\#2 = \frac{2 - 2\cos\|\omega\| - \frac{1}{2}\|\omega\|\sin\|\omega\|}{\|\omega\|^2}\hat{v} +$$
$$\frac{\|\omega\| - \sin\|\omega\|}{\|\omega\|^3}(\hat{\omega}\hat{v} + \hat{v}\hat{\omega}) +$$
$$\frac{1 - \cos\|\omega\| - \frac{1}{2}\|\omega\|\sin\|\omega\|}{\|\omega\|^4}(\hat{\omega}^2\hat{v} + \hat{\omega}\hat{v}\hat{\omega} + \hat{v}\hat{\omega}^2) +$$
$$\frac{\|\omega\| - \frac{3}{2}\sin\|\omega\| + \frac{1}{2}\|\omega\|\cos\|\omega\|}{\|\omega\|^5}(\hat{\omega}^2\hat{v}\hat{\omega} + \hat{\omega}\hat{v}\hat{\omega}^2)$$

$$\#3 = I - \frac{1}{2}\hat{\omega} + \frac{\frac{1}{2}\|\omega\|\sin\|\omega\| + \cos\|\omega\| - 1}{\|\omega\|^2(\cos\|\omega\| - 1)}\hat{\omega}^2,$$

$$\#4 = -\frac{1}{2}\hat{v} + \frac{\frac{1}{2}\|\omega\|\sin\|\omega\| + \cos\|\omega\| - 1}{\|\omega\|^2(\cos\|\omega\| - 1)}(\hat{\omega}\hat{v} + \hat{v}\hat{\omega}) +$$
$$\frac{\frac{1}{4}\|\omega\|^2 + \frac{1}{4}\|\omega\|\sin\|\omega\| + \cos\|\omega\| - 1}{\|\omega\|^4(\cos\|\omega\| - 1)}(\hat{\omega}^2\hat{v}\hat{\omega} + \hat{\omega}\hat{v}\hat{\omega}^2).$$

Eq. (33) is derived based on the fact that $\hat{\omega}^3 = -\|\omega\|^2\hat{\omega}$, of which Eq. (33a) is equivalent to that presented in [30].

V. EXAMPLES

In this section, we implement our approach on the 2D kinematic car and 3D quadrotor respectively evolving on $SE(2)$ and $SE(3)$. The results indicate that our approach has a large region of attraction with input saturation. Furthermore, the computation is much faster than real time meaning that our approach can be implemented online. We also compare the performance of SAC with iLQG for a quadrotor in terms of computational time and basin of attraction. All the tests are run in C++ on a 2.7GHz Intel Core i7 Thinkpad T440p laptop.

A. Example 1: The 2D Kinematic Car

The configuration space of a kinematic car evolves on $g \in SE(2)$ and the dynamic equation can be described as

$$\dot{g} = g\begin{bmatrix} 0 & -\omega & v^{\|}\cos(\phi) \\ \omega & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{34a}$$

$$\omega = v^{\|}\sin(\phi), \quad \dot{v}^{\|} = u_1, \quad \dot{\phi} = u_2 \tag{34b}$$

where ω is the angular velocity, $v^{\|}$ is the forward velocity, ϕ is the steering angle and $u_1, u_2 \in \mathbb{R}$ are the control inputs.

Here we implement SAC for feedback motion planning of car parking. In our tests, the control bounds are $u_{\min} = \begin{bmatrix} -4 & -5 \end{bmatrix}^T$ and $u_{\max} = \begin{bmatrix} 4 & 5 \end{bmatrix}^T$, and to be more consistent with a real car, the steering angle ϕ is constrained to
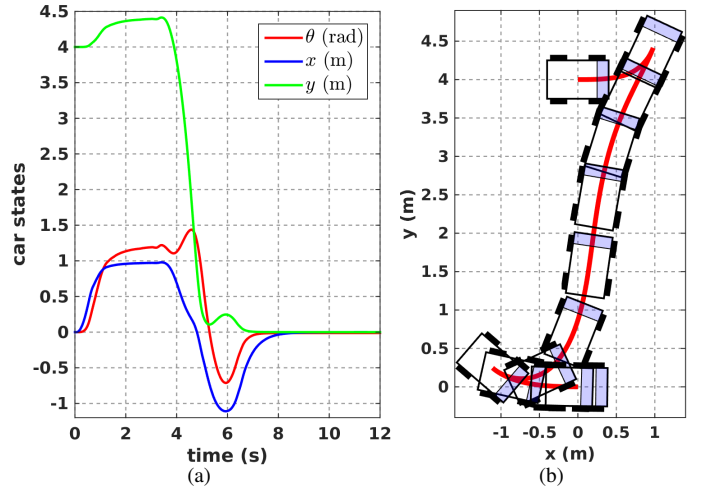
Fig. 1: Feedback motion planning for parallel parking. The initial state is $\theta_0 = 0\,\text{rad}$, $x_0 = 0\,\text{m}$, $y_0 = 4\,\text{m}$, $\phi_0 = 0\,\text{rad}$, $\omega_0 = 0\,\text{rad/s}$, $v_0 = 0\,\text{m/s}$ and the desired state is $\theta_d = 0\,\text{rad}$, $x_d = 0\,\text{m}$, $y_d = 0\,\text{m}$, $\omega_d = 0\,\text{rad/s}$, $v_d = 0\,\text{m/s}$. It takes $0.4\,\text{s}$ for SAC to successfully plan a trajectory to the desired configuration. The feedback sampling rate is 100 Hz.
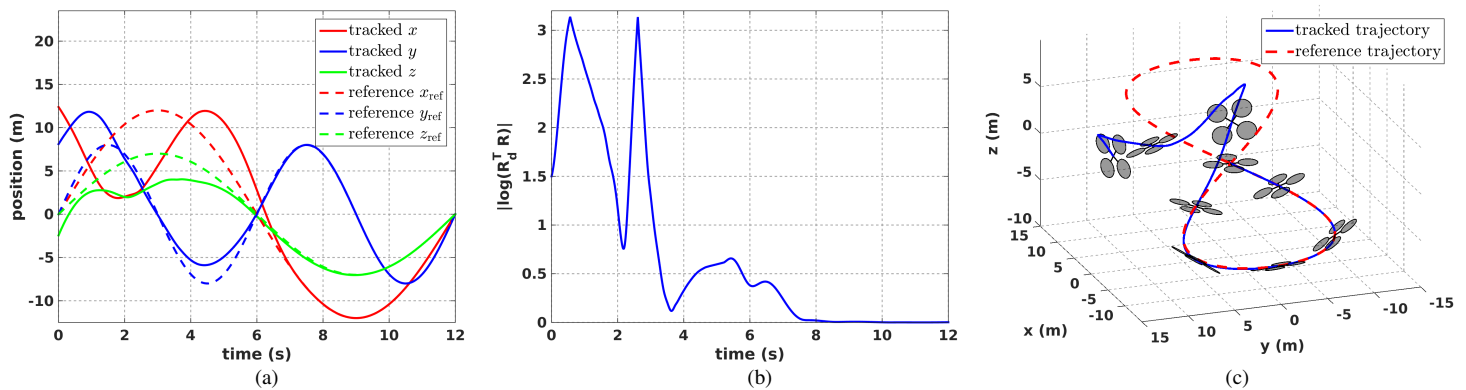
Fig. 2: Trajectory tracking of a quadrotor with large initial error. The reference initial state is $\alpha = 0$, $\beta = 0$, $\gamma = 0$, $x_q = [0\ 0\ 0]^T$ m, $\omega = [0.94\ -0.18\ 0.25]^T$ rad/s and $v_q = [6.28\ 8.38\ 3.67]^T$ m/s while the tracking initial state is $\alpha = 1.45$, $\beta = -0.92$, $\gamma = -0.70$ $x_q = [12.38\ 8.10\ -2.44]^T$ m, $\omega = [-0.56\ 0.90\ 3.80]^T$ rad/s and $v_q = [10.39\ 4.17\ 4.85]^T$ m/s, where α, β, γ are respectively yaw, pitch, roll angles. The resulting performance of position tracking is in (a), orientation tracking in (b) and trajectory tracking in (c). The LQR controller takes effect at $t = 5.20$ s. SAC is around 5.3 times faster than real time. The results are based on a feedback sampling rate of 50 Hz.
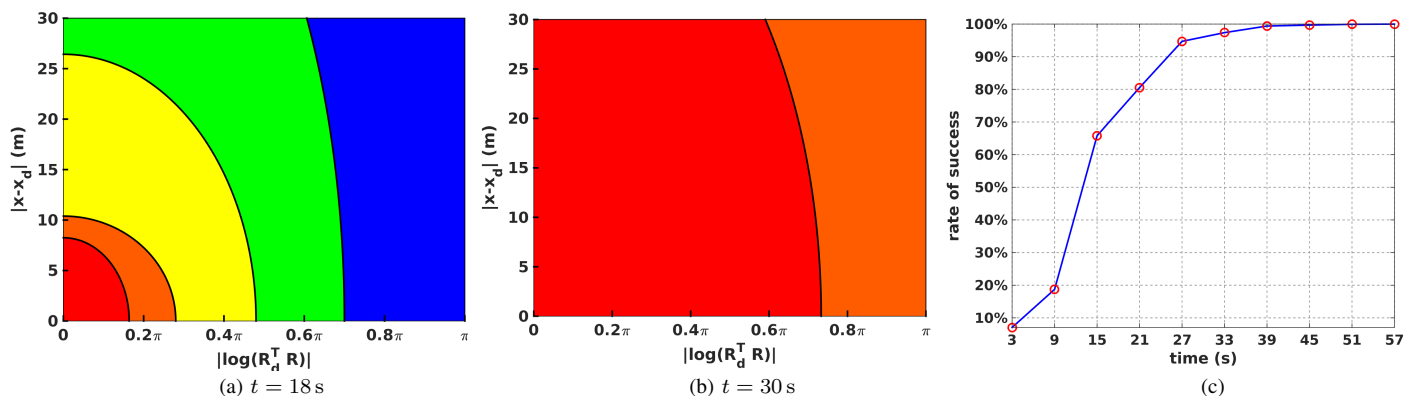

Fig. 3: Statistics of SAC for quadrotor control with 24000 random initial states. In (a) and (b) different colors denote rates of success converging to the region of attraction of LQR at simulation times of $t = 18$ s and 30 s for different initial errors where red $\geq 95\%$, orange $\geq 90\%$, yellow $\geq 80\%$, green $\geq 70\%$, blue $\geq 60\%$ while (c) is the overall rate of success w.r.t. different simulation times. At $t = 63.4$ s, SAC successfully drives the quadrotor to the region of attraction of LQR for all of the 24000 initial states. Note herein the region of attraction of LQR refers to that in the presence of actuation limits.

$[-\frac{1}{3}\pi, \frac{1}{3}\pi]$. The reference control inputs $u_1(t)$ are unknown in motion planning and thus we assume $u_1(t) \equiv \mathbf{0}$ in SAC. However, in parallel parking, it can be checked that $h\left(g(t); t\right)^T \rho(t) \equiv \mathbf{0}$ if the weighting matrix M in Eq. (29) is diagonal and the control action $u_2^*(t) \equiv \mathbf{0}$ by Eq. (7). Therefore, the car is stuck at the initial state despite the fact that it remains far away from the target. To help the car jump out of the singularity, we let off-diagonal elements of M vary slightly *by random* at intervals instead of keeping M unchanged and then $u_2^*(t)$ no longer remains $\mathbf{0}$.[1] As shown in Fig. 1, this works well for the parallel parking problem.[2]

We also test SAC with around 45000 initial states randomly sampled from $\theta_0 \in [-\pi, \pi]$ rad, $x_0, y_0 \in [-10, 10]$ m. In fact, as stated in Section I-A, rather than make SAC search feedback motion plans all by itself, we prefer to combine it with feedback motion planning algorithms via semidefinite programming. In all of our tests, SAC successfully drives the car from initial states aforementioned into a region of $|\theta_T| \leq 0.15$ rad, $\|\mathbf{x}_T\| \leq 0.35$ m, $|v_T^{\|}| \leq 0.02$ m/s after a computational time of around 0.8 s, which can be covered with significantly fewer control policies by feedback motion planning algorithms via semidefnite programming [18, 24, 33].

B. Example 2: The 3D Quadrotor

The quadrotor is an underactuated system evolving on $h = (R, p) \in SE(3)$ and the dynamics can be formulated as

$$\dot{h} = h \begin{bmatrix} \hat{\omega} & v \\ \mathbf{0} & 0 \end{bmatrix}, \tag{35a}$$

$$J\dot{\omega} = M + J\omega \times \omega, \tag{35b}$$

---

[1] M remains positive definite since off-diagonal elements merely slightly vary around 0.

[2] The desired steering angle ϕ_d is not specified as is seldom considered in motion planning.

$$\dot{v} = \frac{1}{m}Fe_3 - \omega \times v - gR^T e_3 \qquad (35c)$$

where ω, $v \in \mathbb{R}^3$ are respectively the body-fixed angular and linear velocities, F and M are forces and torques exerted on the quadrotor associated with control inputs u_i ($i = 1, 2, 3, 4$) by

$$F = k_t(u_1^2 + u_2^2 + u_3^3 + u_4^2),$$
$$M = \begin{bmatrix} k_t l(u_2^2 - u_4^2) \\ k_t l(u_3^2 - u_1^2) \\ k_m(u_1^2 - u_2^2 + u_3^3 - u_4^2) \end{bmatrix}, \qquad (36)$$

and k_t, k_m, l are model parameters. The dynamics (Eq. (35)) used herein are derived on $SE(3)$ and it is different from that on $SO(3) \times \mathbb{R}^3$ due to the distinct Lie group structure[23].

Since in Eq. (36) $u_i^2 \geq 0$ always holds, values of M and F are no longer arbitrary. Though numbers of papers develop various quadrotor controllers, most of them are either NMPC-based or assume M and F can be any values, which may not be implemented online with input saturation.

The fact that the quadrotor dynamics is an affine system with $u_i^2 \in [0, +\infty)$ means that SAC on Lie groups can be implemented to saturate control actions. In this example, we use the combination of SAC and LQR on a quadrotor with $m = 0.6\,\mathrm{kg}$, $J = \mathrm{diag}\{0.04, 0.0375, 0.0675\}\,\mathrm{kg \cdot m^2}$, $k_t = 0.6$, $k_m = 0.15$, $l = 0.2\,\mathrm{m}$ and $u_i^2 \in [0, 6]$ for $i = 1, 2, 3, 4$. The LQR is employed only when the tracking error is below a threshold of $\|\log(g_d^{-1}g)\|^2 + \|\omega - \omega_d\|^2 + \|x - x_d\|^2 \leq 6^2$ and the control inputs given by LQR satisfy the actuation limits, otherwise SAC is applied. The reference control inputs are obtained by differential flatness [28] and the flat outputs are $x(t) = 12\sin(\frac{\pi t}{6})\,\mathrm{m}$, $y(t) = 8\sin(\frac{\pi t}{3})\,\mathrm{m}$, $z(t) = 7\sin(\frac{\pi t}{6})\,\mathrm{m}$ and $\alpha = \frac{1}{2}\sin(\frac{t}{2})\,\mathrm{rad}$. Fig. 2 is a result of trajectory tracking while Fig. 3 are the statistics of 24000 trials whose initial states are randomly sampled from $\|x - x_d\| \in [0, 30]\,\mathrm{m}$, $\|\log(R_d^T R)\| \in [0, \pi]$ and in all of which the combination of SAC and LQR stabilizes the quadrotor to the reference trajectory in a simulation duration of $72\,\mathrm{s}$ with saturated control inputs.

We implement iLQG [32, 34] for trajectory tracking for the purpose of comparison. The iLQG method is one of the most efficient NMPC methods. As is shown in Figs. 3 and 4, SAC outperforms iLQG in both basin of attraction and computational efficiency for cases with large initial errors. Though iLQG achieves a relatively satisfactory performance with 20 iterations, it takes 45 ms for computation for every sample time and still has failures. As a comparison, SAC takes around 4 ms and stabilizes quadrator in all of the 24000 trials including these with large initial errors (Fig. 3).

Usually iLQG needs a quadratic optimizer to saturate control inputs. We tested both SNOPT [14] and the Projected-Newton method [6, 32] for quadratic optimization in iLQG. As noted in Table I, SAC is much faster than iLQG using both optimizers. We also find that the performance of iLQG is severely affected by the chosen optimizers while SAC saturates control inputs in closed form and no optimizers are needed.
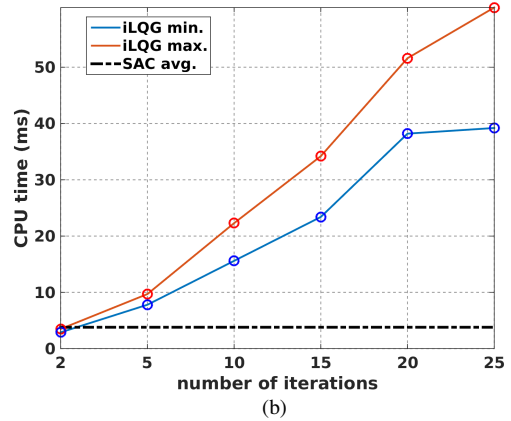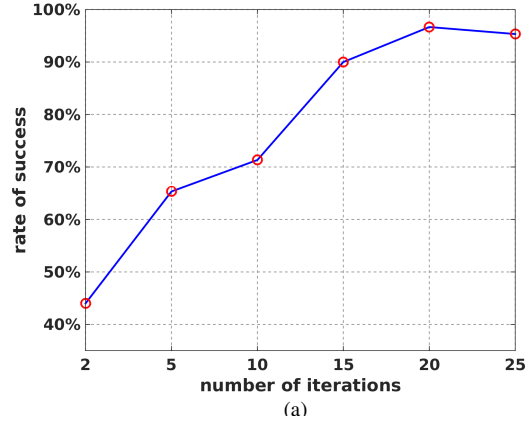

(a)

(b)

Fig. 4: The performance of iLQG under different number of iterations. The stability is tested with 150 random initial states with $\|x - x_d\| = 30$ m and $\|\log(g_d^{-1}g)\| = 3$ rad and the rate of success of iLQG is shown in (a). The computational time of iLQG with different numbers of iterations is in (b) where the black dotted line represents the average computational time of SAC. The computational time of SAC is around $3.8 \sim 4.1$ ms.

	SAC	iLQG	
		SNOPT	Projected-Newton
CPU time	$3.8 \sim 4.1$ ms	$217 \sim 236$ ms	$37 \sim 51$ ms

TABLE I: Computation time of SAC and iLQG. In our tests, the number of iterations in iLQG is set to 20 to obtain an accepted performance, whereas no iterations are needed in SAC.

VI. Conclusion

In this paper, we propose an approach to designing online feedback controllers with input saturation for nonlinear systems evolving on Lie groups by extending sequential action control, which demonstrates a large region of attraction for the kinematic car and quadrotor model though formal proofs of global stability are still needed. In addition, the associated mode insertion and transition gradients are derived and exact expressions of $\mathrm{d}\exp(\xi)$ and $\mathrm{d}\exp^{-1}(\xi)$ for some common Lie groups are given.

REFERENCES

[1] Frank Allgöwer, Rolf Findeisen, and Zoltan K Nagy. Nonlinear model predictive control: From theory to application. *Journal of the Chinese Institute of Chemical Engineers*, 35(3):299–315, 2004.

[2] Brian Anderson and John Moore. *Optimal Control: Linear Quadratic Methods*. Courier Corporation, 2007.

[3] Alex Ansari and Todd D Murphey. Sequential action control: Closed-form optimal control for nonlinear systems. *IEEE Transactions on Robotics*, Conditionally accepted.

[4] Alex Ansari, Kathrin Flaßkamp, and Todd D Murphey. Sequential action control for tracking of free invariant manifolds. In *Conference on Analysis and Design of Hybrid Systems*, pages 335–342, 2015.

[5] Alberto Bemporad, Francesco Borrelli, and Manfred Morari. Model predictive control based on linear programming – the explicit solution. *IEEE Transactions on Automatic Control*, 47(12):1974–1985, 2002.

[6] Dimitri P Bertsekas. Projected newton methods for optimization problems with simple constraints. *SIAM Journal on Control and Optimization*, 20(2):221–246, 1982.

[7] Dimitri P Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

[8] Francesco Bullo and Andrew Lewis. *Geometric control of mechanical systems*, volume 49. Springer-Verlag, 2005.

[9] Timothy M Caldwell and Todd D Murphey. Switching mode generation and optimal estimation with application to skid-steering. *Automatica*, 47(1):50–64, 2011.

[10] Elena Celledoni and Brynjulf Owren. Lie group methods for rigid body dynamics and time integration on manifolds. *Computer Methods in Applied Mechanics and Engineering*, 192(3):421–438, 2003.

[11] Magnus Egerstedt, Yorai Wardi, and Florent Delmotte. Optimal control of switching times in switched dynamical systems. In *IEEE Conference on Decision and Control*, pages 2138–2143, 2003.

[12] Magnus Egerstedt, Yorai Wardi, and Henrik Axelsson. Transition-time optimization for switched-mode dynamical systems. *IEEE Transactions on Automatic Control*, 51(1):110–115, 2006.

[13] Taosha Fan and Todd Murphey. Structured linearization of discrete mechanical systems on lie groups: A synthesis of analysis and control. In *IEEE Conference on Decision and Control (CDC)*, pages 1092–1099, 2015.

[14] Philip E Gill, Walter Murray, and Michael A Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM review*, 47(1):99–131, 2005.

[15] Ernst Hairer, Christian Lubich, and Gerhard Wanner. *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*, volume 31. Springer Science & Business Media, 2006.

[16] Brian C Hall. *Lie groups, Lie algebras, and representations*, volume 222. Springer-Verlag, 2003.

[17] Tiffany A Huang, Matanya B Horowitz, and Joel W Burdick. Convex model predictive control for vehicular systems. *arXiv preprint arXiv:1410.2792*, 2014.

[18] Zachary Jarvis-Wloszek, Ryan Feeley, Weehong Tan, Kunpeng Sun, and Andrew Packard. Some controls applications of sum of squares programming. In *IEEE Conference on Decision and Control (CDC)*, pages 4676–4681, 2003.

[19] Hassan K Khalil. *Nonlinear Systems*. Prentice Hall, 2002.

[20] Marin Kobilarov. Discrete optimal control on Lie groups and applications to robotic vehicles. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5523–5529, 2014.

[21] Marin Kobilarov. Nonlinear trajectory control of multi-body aerial manipulators. *Journal of Intelligent & Robotic Systems*, 73(1-4):679–692, 2014.

[22] Pelor V Kokolovié. The joy of feedback: nonlinear and adaptive. *IEEE Control Systems*, 12(3):7–17, 1992.

[23] Taeyoung Lee, Melvin Leok, and N Harris McClamroch. Geometric tracking control of a quadrotor UAV on SE(3). In *IEEE Conference on Decision and Control (CDC)*, pages 5420–5425, 2010.

[24] Anirudha Majumdar, Ram Vasudevan, Mark M Tobenkin, and Russ Tedrake. Convex optimization of nonlinear feedback controllers via occupation measures. *The International Journal of Robotics Research*, pages 1209–1230, 2014.

[25] Jerrold E Marsden. *Lectures on mechanics*, volume 174. Cambridge University Press, 1992.

[26] Jerrold E Marsden and Tudor S Ratiu. *Introduction to mechanics and symmetry: a basic exposition of classical mechanical systems*, volume 17. Springer-Verlag, 1999.

[27] David Mayne. A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems. *International Journal of Control*, 3(1):85–95, 1966.

[28] Daniel Mellinger and Vijay Kumar. Minimum snap trajectory generation and control for quadrotors. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[29] Alessandro Saccon, John Hauser, and A Pedro Aguiar. Optimal control on Lie groups: The projection operator approach. *IEEE Transactions on Automatic Control*, 58(9):2230–2245, 2013.

[30] Jon M Selig. Lie groups and Lie algebras in robotics. In *Computational Noncommutative Algebra and Applications*, pages 101–125. Springer, 2004.

[31] Koushil Sreenath and Vijay Kumar. Dynamics, control and planning for cooperative manipulation of payloads suspended by cables from multiple quadrotor robots. In *Robotics: Science and Systems (RSS)*, 2013.

[32] Yuval Tassa, Nicolas Mansard, and Emo Todorov. Control-limited differential dynamic programming. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1168–1175, 2014.

[33] Russ Tedrake, Ian R Manchester, Mark Tobenkin, and

John W Roberts. LQR-trees: Feedback motion planning via sums-of-squares verification. *The International Journal of Robotics Research*, 29(8):1038–1052, 2010.

[34] Emanuel Todorov and Weiwei Li. A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *American Control Conference (ACC)*, pages 300–306, 2005.

[35] Emmanouil Tzorakoleftherakis, Alexander Ansari, Andrew Wilson, Jarvis Schultz, and Todd Murphey. Model-based reactive control for hybrid and high-dimensional robotic systems. *IEEE Robotics and Automation Letters*, 1(1):431–438, 2016.

[36] Andrew D Wilson, J.A. Schultz, Alex Ansari, and Todd D Murphey. Real-time trajectory synthesis for information maximization using sequential action control and least-squares estimation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4935–4940, 2015.

[37] Guofan Wu and Koushil Sreenath. Safety-critical and constrained geometric control synthesis using control Lyapunov and control barrier functions for systems evolving on manifolds. In *American Control Conference (ACC)*, pages 2038–2044, 2015.

[38] Milos Zefran, Vijay Kumar, and Christopher Croke. Choice of Riemannian metrics for rigid body kinematics. In *ASME 24th Biennial Mechanisms Conference*, 1996.