A New Approach to Time-Optimal Path Parameterization Based on Reachability Analysis

Hung Pham and Quang-Cuong Pham

***Abstract*—Time-optimal path parameterization (TOPP) is a well-studied problem in robotics and has a wide range of applications. There are two main families of methods to address TOPP: numerical integration (NI) and convex optimization (CO). The NI-based methods are fast but difficult to implement and suffer from robustness issues, while CO-based approaches are more robust but, at the same time, significantly slower. Here, we propose a new approach to TOPP based on reachability analysis. The key insight is to recursively compute reachable and controllable sets at discretized positions on the path by solving small linear programs. The resulting algorithm is faster than NI-based methods and as robust as CO-based ones (100% success rate), as confirmed by extensive numerical evaluations. Moreover, the proposed approach offers unique additional benefits: admissible velocity propagation and robustness to parametric uncertainty can be derived from it in a simple and natural way.**

***Index Terms*—Controllability, motion planning, optimal control, robot control.**

I. Introduction

TIME-OPTIMAL path parameterization (TOPP) is the problem of finding the fastest way to traverse a path in the configuration space of a robot system while respecting the system constraints [1]. This classical problem has a wide range of applications in robotics. In many industrial processes (cutting, welding, machining, 3-D printing, etc.) or mobile robotic applications (driverless cars, warehouse unmanned ground vehicles, aircraft taxiing, etc.), the robot paths may be predefined, and optimal productivity implies tracking those paths at the highest possible speed while respecting the process and robot constraints. From a conceptual viewpoint, TOPP has been used extensively as a subroutine to kinodynamic motion planning algorithms [2], [3]. Because of its practical and theoretical importances, TOPP has received considerable attention since its inception in the 1980s.

Manuscript received July 22, 2017; revised November 21, 2017; accepted January 18, 2018. Date of publication April 16, 2018; date of current version June 6, 2018. This paper was recommended for publication by Associate Editor P.-B. Wieber and Editor T. Murphey upon evaluation of the reviewers' comments. This work was supported in part by Grant ATMRI:2014-R6-PHAM (awarded by NTU and the Civil Aviation Authority of Singapore) and in part by the Medium-Sized Centre funding scheme (awarded by the National Research Foundation, Prime Minister's Office, Singapore). *(Corresponding author: Hung Pham.)*

The authors are with the Air Traffic Management Research Institute, Singapore Centre for 3D Printing, and School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore 639798 (e-mail: pham0074@e.ntu.edu.sg; cuong.pham@normalesup.org).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2018.2819195

A. Existing Approaches to TOPP

There are two main families of methods to TOPP, based, respectively, on numerical integration (NI) and convex optimization (CO). Each approach has its strengths and weaknesses.

The NI-based approach was initiated by [1], and further improved and extended by many researchers, see [4] for a recent review. The NI-based algorithms are based on Pontryagin's maximum principle, which states that the TOPP consists of alternatively maximally accelerating and decelerating segments. The key advantage of this approach is that the optimal controls can be explicitly computed (and not searched for as in the CO approach) at each path position, allowing fast implementations. However, this requires finding the switch points between accelerating and decelerating segments, which constitutes a major implementation difficulty as well as the main cause of failure [4]–[7]. Another notable implementation difficulty is handling of velocity bounds [8].[1] The approach proposed in this paper naturally removes both difficulties.

The CO-based approach was initiated by [9], and further, extended in [10]. This approach formulates and solves TOPP as a single large CO program, whose optimization variables are the accelerations and squared velocities at discretized positions along the path. The main advantages of this approach are as follows: 1) it is simple and robust, owning to available CO libraries; and 2) other convex objectives than the traversal time can be considered. On the downside, the optimization program to solve is huge—the number of variables and constraint inequalities scales with discretization step size—resulting in implementations that are one order of magnitude slower than NI-based algorithms [4]. This makes CO-based algorithms inappropriate for online motion planning or as subroutine to kinodynamic motion planners [3].

B. Proposed New Approach Based on Reachability Analysis

In this paper, we propose a new approach to TOPP based on reachability analysis (RA), a standard concept from the control theory. The principal insight is: given an interval of squared velocities $\mathbb{I}_s$ at some position s on the path, the *reachable* set $\mathbb{I}_{s+\Delta}$ (the set of all squared velocities at the next path position that can be reached from $\mathbb{I}_s$ following admissible controls) and

[1]To account for velocity bounds, NI-based algorithms compute the direct maximum velocity curve $\mathrm{MVC}_{\mathrm{direct}}$, then find and resolve "trap points" [8]. Implementing this procedure is tricky in practice because of accumulating numerical errors. This observation comes from our own experience with the TOPP library [4].

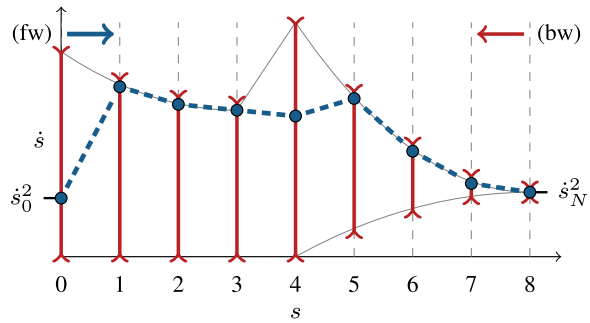

Fig. 1. Time-optimal path parameterization by reachability analysis (TOPP-RA) computes the optimal parameterization in two passes. In the first pass (backward), starting from the last grid point N, the algorithm computes controllable sets (red intervals) recursively. In the second pass (forward), starting now from grid point 0, the algorithm repeatedly selects the highest controls such that resulting velocities remain inside the respective controllable sets.

the *controllable* set $\mathbb{I}_{s-\Delta}$ (the set of all squared velocities at the previous path position such that there exists an admissible control leading to a velocity in $\mathbb{I}_s$) can be computed quickly and robustly by solving a few *small* linear programs (LPs). By recursively computing controllable sets at discretized positions on the path, one can then extract the time-optimal parameterization in time $O(mN)$, where m is the number of constraint inequalities and N the discretization grid size, see Fig. 1 for an illustration.

As compared to NI-based methods, the proposed approach has a better time complexity (actual computation time is similar for problem instances with few constraints, and becomes significantly faster for instances with > 20 constraints). More importantly, the proposed method is much easier to implement and has a success rate of 100%, while state-of-the-art NI-based implementations (e.g., [4]) comprise thousands of lines of code and still report failures on hard problem instances. As compared to CO-based methods, the proposed approach enjoys the same level of robustness and ease-of-implementation while being significantly faster.

Besides the gains in implementation robustness and performance, viewing TOPP from the proposed new perspective yields the following additional benefits.

1) Redundantly actuated systems can be easily handled: there is no need to project the constraints to the plane (path acceleration $\times$ control) at each path position, as done in [10], [11].
2) Admissible velocity propagation [3], a recent concept for kinodynamic motion planning (see Section VI-A for a brief summary), can be derived "for free."
3) Robustness to parametric uncertainty, e.g., uncertain coefficients of friction or uncertain inertia matrices, can be obtained readily.

More details regarding the benefits as well as definitions of relevant concepts will be given in Section VI.

C. Organization of This Paper

The rest of this paper is organized as follows. Section II formulates TOPP in a general setting. Section III applies RA to the path-projected dynamics. Section IV presents the algorithm to compute the time-optimal path parameterization. Section V reports extensive experimental results to demonstrate the gains in robustness and performance permitted by the new approach. Section VI discusses the additional benefits mentioned previously: Admissible velocity propagation and robustness to parametric uncertainty. Finally, Section VII offers some concluding remarks and directions for future research.

II. Problem Formulation

A. Generalized Constraints

Consider a n-dof robot system, whose configuration is denoted by a n-dimensional vector $\mathbf{q} \in \mathbb{R}^n$. A *geometric path* $\mathcal{P}$ in the configuration space is represented as a function $\mathbf{q}(s)_{s\in[0,s_{\text{end}}]}$. We assume that $\mathbf{q}(s)$ is piece-wise $\mathcal{C}^2$-continuous. A *time parameterization* is a piecewise $\mathcal{C}^2$, increasing scalar function $s : [0, T] \to [0, s_{\text{end}}]$, from which a *trajectory* is recovered as $\mathbf{q}(s(t))_{t\in[0,T]}$.

In this paper, we consider *generalized second-order constraints* of the following form [10], [11]:

$$\mathbf{A}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{q}}^\top \mathbf{B}(\mathbf{q})\dot{\mathbf{q}} + \mathbf{f}(\mathbf{q}) \in \mathscr{C}(\mathbf{q}), \text{ where} \tag{1}$$

1) $\mathbf{A}, \mathbf{B}, \mathbf{f}$ are continuous mappings from $\mathbb{R}^n$ to $\mathbb{R}^{m\times n}$, $\mathbb{R}^{n\times m\times n}$, and $\mathbb{R}^m$, respectively;
2) $\mathscr{C}(\mathbf{q})$ is a convex polytope in $\mathbb{R}^m$.

Implementation remark 1: The aforementioned form is the most general in the TOPP literature to date and can account for many types of kinodynamic constraints, including velocity and acceleration bounds, joint torque bounds for fully- or redundantly actuated robots [11], and contact stability under Coulomb friction model [10], [12], [13].

Consider the equation of motion of a fully actuated manipulator with bounded torques

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{q}}^\top \mathbf{C}(\mathbf{q})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau} \tag{2}$$

$$\tau_i^{\min} \le \tau_i(t) \le \tau_i^{\max} \; \forall i \in [1, \dots, n], \; t \in [0, T]. \tag{3}$$

This can be put in the form of (1) with $\mathbf{A} := \mathbf{M}, \mathbf{B} := \mathbf{C}, \mathbf{f} := \mathbf{g}$ and

$$\mathscr{C}(\mathbf{q}) := [\tau_1^{\min}, \tau_1^{\max}] \times \cdots \times [\tau_n^{\min}, \tau_n^{\max}].$$

For closed-chain manipulators with bounded torques, one cannot use (2) and (3) directly to form generalized second-order constraints. One method, proposed in [11], is to use the equation of motion of an open-chain manipulator obtained by cutting the closed-chain manipulator at specific joints. In particular, let $\mathbf{q}_O$ denote the configuration of the open chain. Suppose both chains have the same motion, [14] showed that τ is a feasible torque if

$$\mathbf{M}_O(\mathbf{q}_O)\ddot{\mathbf{q}}_O + \dot{\mathbf{q}}_O^\top \mathbf{C}_O(\mathbf{q}_O)\dot{\mathbf{q}}_O + \mathbf{g}_O(\mathbf{q}_O) = \boldsymbol{\tau}_O$$

$$\mathbf{S}^\top \boldsymbol{\tau} = \mathbf{W}^\top \boldsymbol{\tau}_O$$

$$\tau_i^{\min} \le \tau_i(t) \le \tau_i^{\max} \; \forall i \in [1, \dots, n], \; t \in [0, T]$$

where $\mathbf{S}$ and $\mathbf{W}$ are the *sentivity matrices*, computed from the kinematics of the closed-chain manipulator and the selection of joints. See [14] for more details.

For legged robots, TOPP under contact-stability constraints with linearized friction cones was shown to be reducible to a generalized second-order constraint (1) [10]–[12].

If the friction cones are not linearized, then the set $\mathscr{C}(\mathbf{q})$ in (1) is still convex, but not polytopic. The developments in this paper that concern reachable and controllable sets (see Section III) are still valid in the convex, nonpolytopic case. The developments on time optimality (see Section IV) is, however, only applicable to the polytopic case.

Finally, we also consider first-order constraints of the form

$$\mathbf{A}^v(\mathbf{q})\dot{\mathbf{q}} + \mathbf{f}^v(\mathbf{q}) \in \mathscr{C}^v(\mathbf{q})$$

where the coefficients are matrices of appropriate sizes and $\mathscr{C}^v(\mathbf{q})$ is a convex set. Direct velocity bounds and momentum bounds are examples of first-order constraints.

B. Projecting the Constraints on the Path

Differentiating successively $\mathbf{q}(s)$, one has

$$\dot{\mathbf{q}} = \mathbf{q}'\dot{s}, \quad \ddot{\mathbf{q}} = \mathbf{q}''\dot{s}^2 + \mathbf{q}'\ddot{s} \tag{4}$$

where $\square'$ denotes differentiation with respect to the path parameter s, while $\dot{\square}$ denotes differentiation with respect to time. From now on, we shall refer to s, $\dot{s}$, and $\ddot{s}$ as the position, velocity, and acceleration, respectively.

Substituting (4) into (1), one transforms generalized second-order constraints into constraints on s, $\dot{s}$, and $\ddot{s}$ as follows:

$$\begin{aligned} &\mathbf{a}(s)\ddot{s} + \mathbf{b}(s)\dot{s}^2 + \mathbf{c}(s) \in \mathscr{C}(s), \text{ where} \qquad (5)\\ &\mathbf{a}(s) := \mathbf{A}(\mathbf{q}(s))\mathbf{q}'(s)\\ &\mathbf{b}(s) := \mathbf{A}(\mathbf{q}(s))\mathbf{q}''(s) + \mathbf{q}'(s)^\top \mathbf{B}(\mathbf{q}(s))\mathbf{q}'(s)\\ &\mathbf{c}(s) := \mathbf{f}(\mathbf{q}(s))\\ &\mathscr{C}(s) := \mathscr{C}(\mathbf{q}(s)). \end{aligned}$$

Similarly, first-order constraints are transformed into

$$\begin{aligned} &\mathbf{a}^v(s)\dot{s} + \mathbf{b}^v(s) \in \mathscr{C}^v(s), \text{ where} \qquad (6)\\ &\mathbf{a}^v(s) := \mathbf{A}^v(\mathbf{q}(s))\mathbf{q}'(s)\\ &\mathbf{b}^v(s) := \mathbf{f}^v(\mathbf{q}(s))\\ &\mathscr{C}^v(s) := \mathscr{C}^v(\mathbf{q}(s)). \end{aligned}$$

C. Path Discretization

As in the CO-based approach, we divide the interval $[0, s_{\text{end}}]$ into N segments and $N+1$ grid points

$$0 =: s_0, s_1 \ldots s_{N-1}, s_N := s_{\text{end}}.$$

Denote by u_i the constant path acceleration over the interval $[s_i, s_{i+1}]$ and by x_i the squared velocity $\dot{s}_i^2$ at s_i. One has the following relation:

$$x_{i+1} = x_i + 2\Delta_i u_i, \quad i = 0 \ldots N-1 \tag{7}$$

where $\Delta_i := s_{i+1} - s_i$. This relation is obtained by noting that

$$\frac{\mathrm{d}\dot{s}^2}{\mathrm{d}s} = 2\dot{s}\frac{\mathrm{d}\dot{s}}{\mathrm{d}s} = 2\ddot{s} = 2u.$$

In the sequel, we refer to s_i as the i-stage, u_i and x_i as, respectively, the control and state at the i-stage. Any sequence $x_0, u_0, \ldots, x_{N-1}, u_{N-1}, x_N$ that satisfies the linear relation (7) is referred to as a path parameterization.

A parameterization is *admissible* if it satisfies the constraints at every point in $[0, s_{\text{end}}]$. One possible way to bring this requirement into the discrete setting is through a *collocation* discretization scheme: for each position s_i, one evaluates the continuous constraints and requires the control and state u_i, x_i to verify

$$\mathbf{a}_i u_i + \mathbf{b}_i x_i + \mathbf{c}_i \in \mathscr{C}_i \tag{8}$$

where $\mathbf{a}_i := \mathbf{a}(s_i), \mathbf{b}_i := \mathbf{b}(s_i), \mathbf{c}_i := \mathbf{c}(s_i)$, and $\mathscr{C}_i := \mathscr{C}(s_i)$.

Since constraints (8) are enforced only at a finite number of points, the continuous constraints (5) and (6) are not satisfied everywhere along $[0, s_{\text{end}}]$.[2] Therefore, it is important to characterize satisfaction errors. We show in Appendix D that the collocation scheme has satisfaction errors of order $O(\Delta_i)$. Appendix D also presents another discretization scheme with satisfaction errors of order $O(\Delta_i^2)$ but that involves more variables and constraints than the collocation scheme.

III. RA of the Path-Projected Dynamics

The key to our analysis is that the "path-projected dynamics" (7), (8) is a *discrete-time linear system* with *linear control-state inequality constraints*. This observation immediately allows us to take advantage of the set-membership control problem studied in the model predictive control literature [15]–[17].

A. Admissible States and Controls

We first need some definitions. Denote the i-stage set of *admissible* control-state pairs by

$$\Omega_i := \{(u, x) \mid \mathbf{a}_i u + \mathbf{b}_i x + \mathbf{c}_i \in \mathscr{C}_i\}.$$

One can see Ω_i as the projection of $\mathscr{C}_i$ on the $(\ddot{s}, \dot{s}^2)$ plane [10]. Since $\mathscr{C}_i$ is a polytope, Ω_i is a *polygon*. Algorithmically, the projection can be obtained by, e.g., the recursive expansion algorithm [18].

Next, the i-stage set of *admissible states* is the projection of Ω_i on the second axis

$$\mathcal{X}_i := \{x \mid \exists u : (u, x) \in \Omega_i\}.$$

The i-stage set of *admissible controls* given a state x is

$$\mathcal{U}_i(x) := \{u \mid (u, x) \in \Omega_i\}.$$

Note that, since Ω_i is convex, both $\mathcal{X}_i$ and $\mathcal{U}_i(x)$ are *intervals*.

Classic terminologies in the TOPP literature (e.g., maximum velocity curve, α and β acceleration fields) can be conveniently expressed using these definitions. See the first part of Appendix A for more details.

Implementation remark 2: For redundantly actuated manipulators and legged robots with contact-stability constraints, both NI-based and CO-based algorithms must compute Ω_i at each

[2] This limitation is not specific to the proposed approach as all NI-based and CO-based algorithms require discretization at some stages.

discretized position i along the path, which is costly. Our proposed approach avoids performing this 2-D projection: instead, it will only require a few 1-D projections per discretization step. Furthermore, each of these 1-D projections amounts to a pair of LPs and can be performed extremely quickly.

B. Reachable Sets

A key notion in RA is the i-stage reachable set.

Definition 1 (i-stage reachable set): Consider a set of starting states $\mathbb{I}_0$. The *i-stage reachable set* $\mathcal{L}_i(\mathbb{I}_0)$ is the set of states $x \in \mathcal{X}_i$ such that there exists a state $x_0 \in \mathbb{I}_0$ and a sequence of admissible controls $u_0, \dots, u_{i-1}$ that steers the system from x_0 to x. ◊

To compute the i-stage reachable set, one needs the following intermediate representation.

Definition 2 (Reach set): Consider a set of states $\mathbb{I}$. The *reach set* $\mathcal{R}_i(\mathbb{I})$ is the set of states $x \in \mathcal{X}_{i+1}$ such that there exists a state $\tilde{x} \in \mathbb{I}$ and an admissible control $u \in \mathcal{U}_i(\tilde{x})$ that steers the system from $\tilde{x}$ to x, i.e.,

$$x = \tilde{x} + 2\Delta_i u. \qquad \lozenge$$

Implementation remark 3: Let us note $\Omega_i(\mathbb{I}) := \{(u, \tilde{x}) \in \Omega_i \mid \tilde{x} \in \mathbb{I}\}$. If $\mathbb{I}$ is convex, then $\Omega_i(\mathbb{I})$ is convex as it is the intersection of two convex sets. Next, $\mathcal{R}_i(\mathbb{I})$ can be seen as the intersection of the projection of $\Omega_i(\mathbb{I})$ onto a line and the interval $\mathcal{X}_{i+1}$. Thus, $\mathcal{R}_i(\mathbb{I})$ is an interval, hence, defined by its lower and upper bounds (x^-, x^+), which can be computed as follows:

$$x^- := \min_{(u,\tilde{x}) \in \Omega_i(\mathbb{I}),\ x^- \in \mathcal{X}_{i+1}} (\tilde{x} + 2\Delta_i u)$$

$$x^+ := \max_{(u,\tilde{x}) \in \Omega_i(\mathbb{I}),\ x^+ \in \mathcal{X}_{i+1}} (\tilde{x} + 2\Delta_i u).$$

Since $\Omega_i(\mathbb{I})$ is a polygon, the aforementioned equations constitute two LPs. Note, finally, that there is no need to compute explicitly $\Omega_i(\mathbb{I})$, since one can write directly

$$x^+ := \max_{(u,\tilde{x}) \in \mathbb{R}^2} (\tilde{x} + 2\Delta_i u)$$

$$\text{subject to: } \mathbf{a}_i u + \mathbf{b}_i \tilde{x} + \mathbf{c}_i \in \mathscr{C}_i, \tilde{x} \in \mathbb{I} \text{ and } x^+ \in \mathcal{X}_{i+1}$$

and similarly for x^-. ◊

The i-stage reachable set can be recursively computed by

$$\begin{aligned} \mathcal{L}_0(\mathbb{I}_0) &= \mathbb{I}_0 \cap \mathcal{X}_0 \\ \mathcal{L}_i(\mathbb{I}_0) &= \mathcal{R}_{i-1}(\mathcal{L}_{i-1}(\mathbb{I}_0)). \end{aligned} \tag{9}$$

Implementation remark 4: If $\mathbb{I}_0$ is an interval, then by recursion and by application of *implementation remark 3*, all the $\mathcal{L}_i$ are intervals. Each step of the recursion requires solving two LPs for computing $\mathcal{R}_{i-1}(\mathcal{L}_{i-1}(\mathbb{I}_0))$. Therefore, $\mathcal{L}_i$ can be computed by solving $2i + 2$ LPs. ◊

The i-stage reachable set may be empty, which implies that the system cannot evolve without violating constraints: the path is not time parameterizable. One can also note that

$$\mathcal{L}_i(\mathbb{I}_0) = \emptyset \Rightarrow \forall j \geq i,\ \mathcal{L}_j(\mathbb{I}_0) = \emptyset.$$

C. Controllable Sets

Controllability is the dual notion of reachability, as made clear by the following definitions.

Definition 3 (i-stage controllable set): Consider a set of desired ending states $\mathbb{I}_N$. The *i-stage controllable set* $\mathcal{K}_i(\mathbb{I}_N)$ is the set of states $x \in \mathcal{X}_i$ such that there exists a state $x_N \in \mathbb{I}_N$ and a sequence of admissible controls $u_i, \dots, u_{N-1}$ that steers the system from x to x_N. ◊

The dual notion of "reach set" is that of "one-step" set.

Definition 4 (One-step set): Consider a set of states $\mathbb{I}$. The *one-step set* $\mathcal{Q}_i(\mathbb{I})$ is the set of states $x \in \mathcal{X}_i$ such that there exists a state $\tilde{x} \in \mathbb{I}$ and an admissible control $u \in \mathcal{U}_i(x)$ that steers the system from x to $\tilde{x}$, i.e.,

$$\tilde{x} = x + 2\Delta_i u. \qquad \lozenge$$

The i-stage controllable set can now be computed recursively by

$$\begin{aligned} \mathcal{K}_N(\mathbb{I}_N) &= \mathbb{I}_N \cap \mathcal{X}_N \\ \mathcal{K}_i(\mathbb{I}_N) &= \mathcal{Q}_i(\mathcal{K}_{i+1}(\mathbb{I}_N)). \end{aligned} \tag{10}$$

Implementation remark 5: Similar to *implementation remark 4*, every one-step set $\mathcal{Q}_i(\mathbb{I})$ is an interval, whose lower and upper bounds (x^-, x^+) can be computed as follows:

$$x^+ := \max_{(u,x) \in \mathbb{R}^2} x$$

$$\text{subject to: } \mathbf{a}_i u + \mathbf{b}_i x + \mathbf{c}_i \in \mathscr{C}_i \text{ and } x + 2\Delta_i u \in \mathbb{I}$$

and similarly for x^-. Thus, computing the i-stage controllable set will require solving $2(N - i) + 2$ LPs. ◊

The i-stage controllable set may be empty, in that case, the path is not time parameterizable. One also has

$$\mathcal{K}_i(\mathbb{I}_N) = \emptyset \Rightarrow \forall j \leq i,\ \mathcal{K}_j(\mathbb{I}_N) = \emptyset.$$

IV. TOPP BY RA

A. Algorithm

Armed with the notions of reachable and controllable sets, we can now proceed to solving the TOPP problem. The RA-based TOPP algorithm (TOPP-RA) is given in Algorithm IV-A and illustrated in Fig. 1.

The algorithm proceeds in two passes. The first pass goes backward: it recursively computes the controllable sets $\mathcal{K}_i(\{\dot{s}_N^2\})$ given the desired ending velocity $\dot{s}_N$, as described in Section III-C. If any of the controllable sets is empty or if the starting state $\dot{s}_0^2$ is not contained in the 0-stage controllable set, then the algorithm reports failure.

Otherwise, the algorithm proceeds to a second, forward, pass. Here, the optimal states and controls are constructed *greedily*: At each stage i, the highest admissible control such that the resulting next state belongs to the $(i + 1)$-stage controllable set is selected.

Note that one can construct a "dual version" of TOPP-RA as follows: 1) in a forward pass, recursively compute the i-stage reachable sets, $i \in [0, \dots, N]$; and 2) in a backward pass,

Algorithm 1: TOPP-RA.

Input : Path $\mathcal{P}$, starting and ending velocities $\dot{s}_0, \dot{s}_N$

Output: Parameterization $x_0^*, u_0^*, \ldots, u_{N-1}^*, x_N^*$

`/* Backward pass: compute the controllable sets */`

$\mathcal{K}_N := \{\dot{s}_N^2\}$

for $i \in [N-1 \ldots 0]$ **do**

  $\mathcal{K}_i := \mathcal{Q}_i(\mathcal{K}_{i+1})$

if $\mathcal{K}_0 = \emptyset$ or $\dot{s}_0^2 \notin \mathcal{K}_0$ **then**

  return `Infeasible`

`/* Forward pass: select controls greedily */`

$x_0^* := \dot{s}_0^2$

for $i \in [0 \ldots N-1]$ **do**

  $u_i^* := \max u$, subject to: $x_i^* + 2\Delta_i u \in \mathcal{K}_{i+1}$ and $(u, x_i^*) \in \Omega_i$

  $x_{i+1}^* := x_i^* + 2\Delta_i u_i^*$

greedily select, at stage i, the lowest control such that the previous state belongs to the $(i-1)$-stage reachable set.

In the following sections, we show the correctness and optimality of the algorithm and give a detailed complexity analysis.

B. Correctness of TOPP-RA

We show that TOPP-RA is correct in the sense of the following theorem.

Theorem 1: Consider a discretized TOPP instance. TOPP-RA returns an admissible parameterization solving that instance whenever one exists, and reports `Infeasible` otherwise.

Proof: 1) We first show that, if TOPP-RA reports `Infeasible`, then the instance is indeed not parameterizable. By contradiction, assume that there exists an admissible parameterization $\dot{s}_0^2 = x_0, u_0, \ldots, u_{N-1}, x_N = \dot{s}_N^2$. We now show by backward induction on i that $\mathcal{K}_i$ contains at least x_i.

Initialization: $\mathcal{K}_N$ contains x_N by construction.

Induction: Assume that $\mathcal{K}_i$ contains x_i. Since the parameterization is admissible, one has $x_i = x_{i-1} + 2\Delta_i u_{i-1}$ and $(u_{i-1}, x_{i-1}) \in \Omega_{i-1}$. By definition of the controllable sets, $x_{i-1} \in \mathcal{K}_{i-1}$.

We have thus shown that none of the $\mathcal{K}_i$ is empty and that $\mathcal{K}_0$ contains at least $x_0 = \dot{s}_0^2$, which implies that TOPP-RA cannot report `Infeasible`.

(2) Assume now that TOPP-RA returns a sequence $(x_0^*, u_0^*, \ldots, u_{N-1}^*, x_N^*)$. One can easily show by forward induction on i that the sequence indeed constitutes an admissible parameterization that solves the instance. □

C. Asymptotic Optimality of TOPP-RA

We show the following result: as the discretization step size goes to zero, the traversal time of the parameterization returned by TOPP-RA converges to the optimal value.

Unsurprisingly, the main difficulty in proving asymptotic optimality comes from the existence of zero-inertia points [4], [5], [7]. Note, however, that this difficulty does not affect the robustness or the correctness of the algorithm.

To avoid too many technicalities, we make the following assumption:

Assumption 1 (and definition): There exist piecewise $\mathcal{C}^1$-continuous functions $\tilde{\mathbf{a}}(s)_{s\in[0,1]}, \tilde{\mathbf{b}}(s)_{s\in[0,1]}$, and $\tilde{\mathbf{c}}(s)_{s\in[0,1]}$ such that for all $i \in \{0, \ldots, N\}$, the set of admissible control-state pairs is given by

$$\Omega_i = \{(u, x) \mid u\tilde{\mathbf{a}}(s_i) + x\tilde{\mathbf{b}}(s_i) + \tilde{\mathbf{c}}(s_i) \leq 0\}.$$

Augment $\tilde{\mathbf{a}}, \tilde{\mathbf{b}}$, and $\tilde{\mathbf{c}}$ into $\bar{\mathbf{a}}, \bar{\mathbf{b}}$, and $\bar{\mathbf{c}}$ by adding two inequalities that express the condition $x + 2\Delta_i u \in \mathcal{K}_{i+1}$. The set of admissible *and controllable* control-state pairs is then given by

$$\Omega_i \cap (\mathbb{R} \times \mathcal{K}_i) = \{(u, x) \mid u\bar{\mathbf{a}}(s_i) + x\bar{\mathbf{b}}(s_i) + \bar{\mathbf{c}}(s_i) \leq 0\}. \ \Diamond$$

The aforementioned assumption is easily verified in the canonical case of a fully actuated manipulator subject to torque bounds tracking a smooth path. It allows us to next conveniently define zero-inertia points.

Definition 5 (Zero-inertia points): A point $s^\bullet$ constitutes a zero-inertia point if there is a constraint k such that $\bar{\mathbf{a}}(s^\bullet)[k] = 0$. $\Diamond$

We have the following theorem, whose proof is given in Appendix B (to simplify the notations, we consider uniform step sizes $\Delta_0 = \cdots = \Delta_{N-1} = \Delta$).

Theorem 2: Consider a TOPP instance *without zero-inertia points*. There exists a Δ_{thr} such that if $\Delta < \Delta_{\text{thr}}$, then the parameterization returned by TOPP-RA is optimal.

The key hypothesis of this theorem is that there is no zero-inertia points. In practice, however, zero-inertia points are unavoidable, and in fact, constitute the most common type of switch points [4]. The next theorem, whose proof is given in Appendix C, establishes that the suboptimality gap converges to zero with step size.

Theorem 3: Consider a TOPP instance with a zero-inertia point at $s^\bullet$. Denote by J^* the cost of the parameterization returned by TOPP-RA $\sum_{i=0}^{N+1} \frac{\Delta}{\sqrt{x_i^*}}$ and by $J^\dagger$ the minimum cost at the same step size. Then, one has

$$J^* - J^\dagger = O(\Delta).$$

This theorem implies that, by reducing the step size, the cost of the parameterization returned by TOPP-RA can be made arbitrarily close to the minimum cost.

D. Complexity Analysis

We now perform a complexity analysis of TOPP-RA and compare it with NI and CO-based methods. For simplicity, we shall restrict the discussion to fully actuated manipulators (the redundantly actuated case actually brings an additional advantage to TOPP-RA, see *implementation remark 3*).

Assume that there are m constraint inequalities and that the path discretization grid size is N. As a large part of the computation time is devoted to solving LPs, we need a good estimate of the practical complexity of this operation. Consider an LP with ν optimization variables and m inequality constraints. Different LP methods (ellipsoidal, simplex, active sets, etc.) have

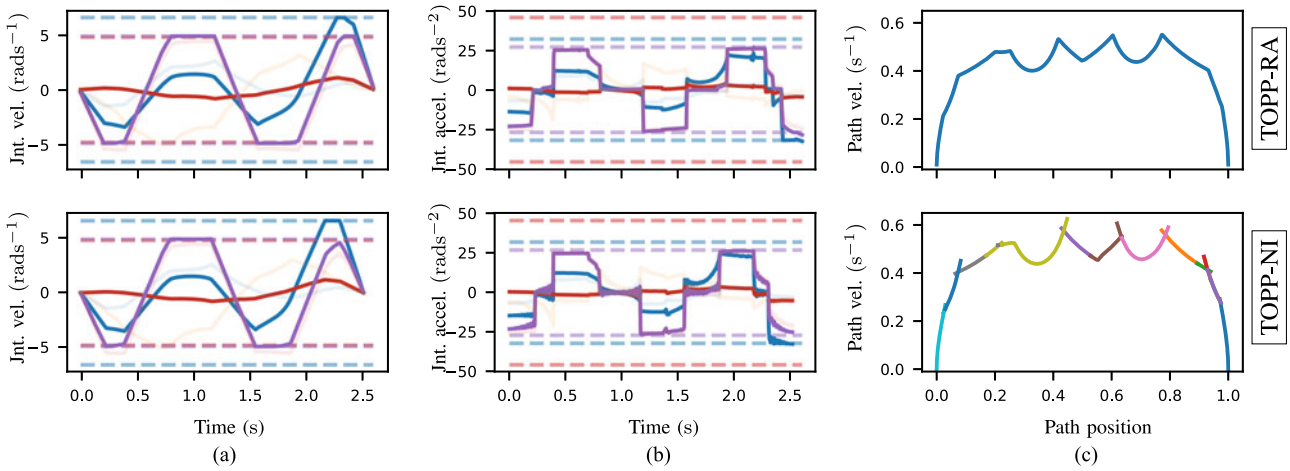

Fig. 2. Time-optimal parameterization of a 6-dof path under velocity and acceleration bounds ($m = 14$ constraint inequalities and $N = 500$ grid points). TOPP-RA and TOPP-NI produce nearly identical results. **(A)**: joint velocities. **(B)**: joint accelerations. **(C)**: velocity profiles in the $(s, \dot{s})$ plane. Note the small chattering in the joint accelerations produced by TOPP-NI, which is an artifact of the integration process. This chattering is absent from the TOPP-RA profiles.

different complexities. For the purpose of this section, we consider the best *practical* complexity, which is realized by the simplex method, in $O(\nu^2 m)$ [19].

1) *TOPP-RA:* The LPs considered here have two variables and $m + 2$ inequalities. Since one needs to solve $3N$ such LPs, the complexity of TOPP-RA is $O(mN)$.
2) *NI approach:* The dominant component of this approach, in terms of time complexity, is the computation of the maximum velocity vurve (MVC). In most TOPP-NI implementations to date, the MVC is computed, at each discretized path position, by solving $O(m^2)$ second-order polynomials [1], [4]–[7], which results in an overall complexity of $O(m^2 N)$.
3) *CO approach:* This approach formulates the TOPP problem as a single large CO program with $O(N)$ variables and $O(mN)$ inequality constraints. In the fastest implementation we know of, the author solves the CO problem by solving a sequence of linear programs (SLP) with the same number of variables and inequalities [10]. Thus, the time complexity of this approach is $O(KmN^3)$, where K is the number of SLP iterations.

This analysis shows that TOPP-RA has the best theoretical complexity. The next section experimentally assesses this observation.

V. Experiments

We implement TOPP-RA in Python. All LPs are solved with `qpOASES` [20]. Experiments were performed on a machine running Ubuntu with an Intel i7-4770(8) 3.9-GHz CPU and 8-Gb RAM. The implementation and test cases are available at https://github.com/hungpham2511/toppra.

A. Experiment 1: Pure Joint Velocity and Acceleration Bounds

In this experiment, we compare TOPP-RA against TOPP-NI—the fastest known implementation of TOPP, which is based on the NI approach [4]. For simplicity, we consider pure joint velocity and acceleration bounds, which involve the same difficulty as any other types of kinodynamic constraints, as far as TOPP is concerned.

1) Effect of the Number of Constraint Inequalities: We considered random geometric paths with varying dimensions $n \in [2, 60]$. Each path was generated as follows: Five random waypoints were sampled, then interpolated by cubic spline interpolation. For each path, velocity and acceleration bounds were also randomly chosen such that the bounds contain zero. This ensures that all generated TOPP instances are feasible. Each instance, thus, has $m = 2n + 2$ constraint inequalities: $2n$ inequalities corresponding to acceleration bounds (no pruning was applied, contrary to [10]) and two inequalities corresponding to velocity bounds (the joint velocity bounds could be immediately pruned into one lower and one upper bound on $\dot{s}^2$). According to the complexity analysis of Section IV-D, we consider the number of inequalities, rather than the dimension, as independent variable. Finally, the discretization grid size was chosen as $N = 500$.

Fig. 2 shows the time-parameterizations and the resulting trajectories produced by TOPP-RA and TOPP-NI on an instance with ($n = 6$ and $m = 14$). One can observe that the two algorithms produced nearly identical results, hinting at the correctness of TOPP-RA.

Fig. 3 shows the computation time for TOPP-RA and TOPP-NI, excluding the "setup" and "extract trajectory" steps (which takes much longer in TOPP-NI than in TOPP-RA). The experimental results confirm our theoretical analysis in that the complexity of TOPP-RA is in linear in m, while that of TOPP-NI is quadratic in m. In terms of actual computation time, TOPP-RA becomes faster than TOPP-NI as soon as $m \geq 22$.

Perhaps even more importantly than mere computation time, TOPP-RA was extremely robust: it maintained 100% success rate over all instances, while TOPP-NI struggled with instances with many inequality constraints ($m \geq 40$), see Fig. 4. Since all TOPP instances considered here were feasible, an algorithm failed when it failed to return a parameterization.

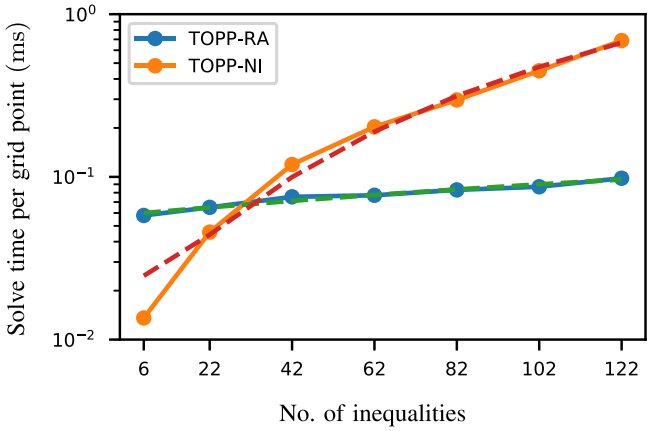

Fig. 3. Solve time per grid point of TOPP-RA (solid blue) and TOPP-NI (solid orange), excluding the "setup" and "extract trajectory" steps, as a function of the number of constraint inequalities. Confirming our theoretical complexity analysis, time complexity of TOPP-RA is linear in the number of constraint inequalities m (linear fit in dashed green), while that of TOPP-NI is quadratic in m (quadratic fit in dashed red). In terms of actual computation time, TOPP-RA becomes faster than TOPP-NI as soon as $m \geq 30$.

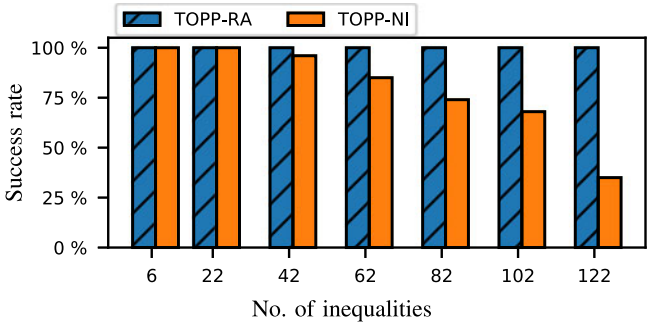

Fig. 4. Success rate for TOPP-RA and TOPP-NI. TOPP-RA enjoys consistently 100% success rate, while TOPP-NI reports failure for more complex problem instances ($m \geq 40$).

TABLE I
BREAKDOWN OF TOPP-RA, TOPP-RA-INTP AND TOPP-NI TOTAL COMPUTATION TIME TO PARAMETERIZE A PATH DISCRETIZED WITH $N = 500$ GRID POINTS, SUBJECT TO $m = 30$ INEQUALITIES

	Time (ms)		
	TOPP-RA	TOPP-RA-intp	TOPP-NI
setup	1.0	1.5	123.6
solve TOPP	26.1	29.1	28.3
backward pass	16.5	19.9	
forward pass	9.6	9.2	
extract trajectory	2.7	2.7	303.4
total	29.8	33.3	455.3

Table I reports the different components of the computation time. In addition to TOPP-RA and TOPP-NI, we considered TOPP-RA-intp. This variant of TOPP-RA employs the first-order interpolation scheme (see Appendix D) to discretize the constraints, instead of the collocation scheme introduced in Section II-C.

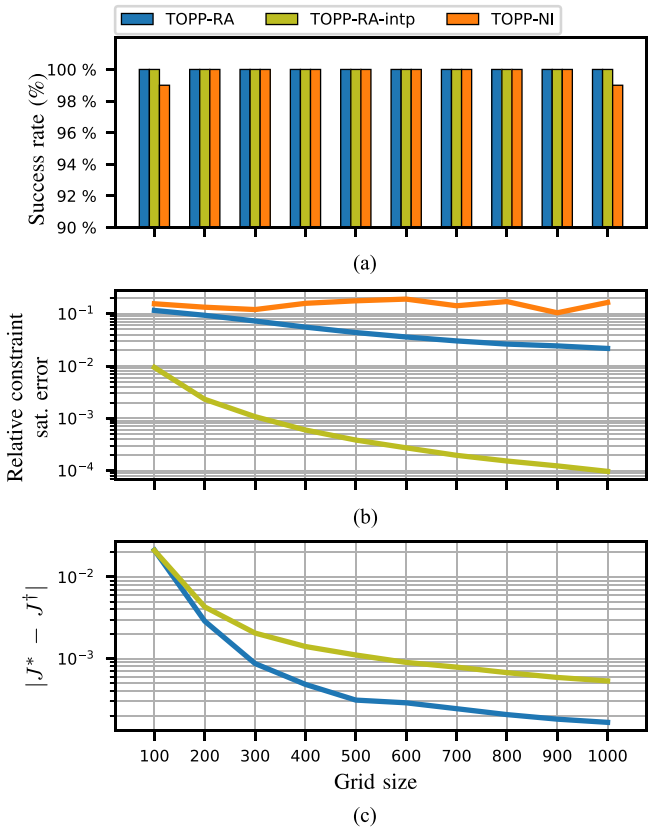

Fig. 5. (a) Effect of the grid size on the success rate. (b) Effect of the grid size on the relative constraint satisfaction error, defined as the ratio between the error and the respective bound. TOPP-RA-intp returned solutions that are orders of magnitude better than TOPP-RA and TOPP-NI. (c) Effect of the grid size on difference between the average solution's cost and the optimal cost, which is approximated by solving TOPP-RA-intp with $N = 10000$. Solutions produced by TOPP-RA-intp had higher costs than those produced by TOPP-RA as those instances were more highly constrained.

2) Effect of Discretization Grid Size: Discretization grid size (or its inverse, discretization step) is an important parameter for both TOPP-RA and TOPP-NI as it affects running time, success rate, and solution quality, as measured by constraint satisfaction error and suboptimality. Here, we assess the effect of the grid size on *success rate* and *solution quality*. Remark that, based on our complexity analysis in Section IV-D, running time depends linearly on the grid size in both algorithms.

We considered different grid sizes $N \in [100, 1000]$. For each grid size, we generated and solved 100 random TOPP instances; each instance consists of a random path with $n = 14$ subject to random kinematic constraints, as in the previous experiment. Fig. 5(a) shows success rates versus grid sizes. One can observe that TOPP-RA and TOPP-RA-intp maintained 100% success rate across all grid sizes, while TOPP-NI reported two failures at $N = 100$ and $N = 1000$.

Next, to measure the effect of the grid size on solution quality, we looked at the *relative greatest constraint satisfaction errors*, defined as the ratio between the errors, whose definition is given in Appendix D2, and the respective bounds. For each instance, we sampled the resulting trajectories at 1 ms and computed the

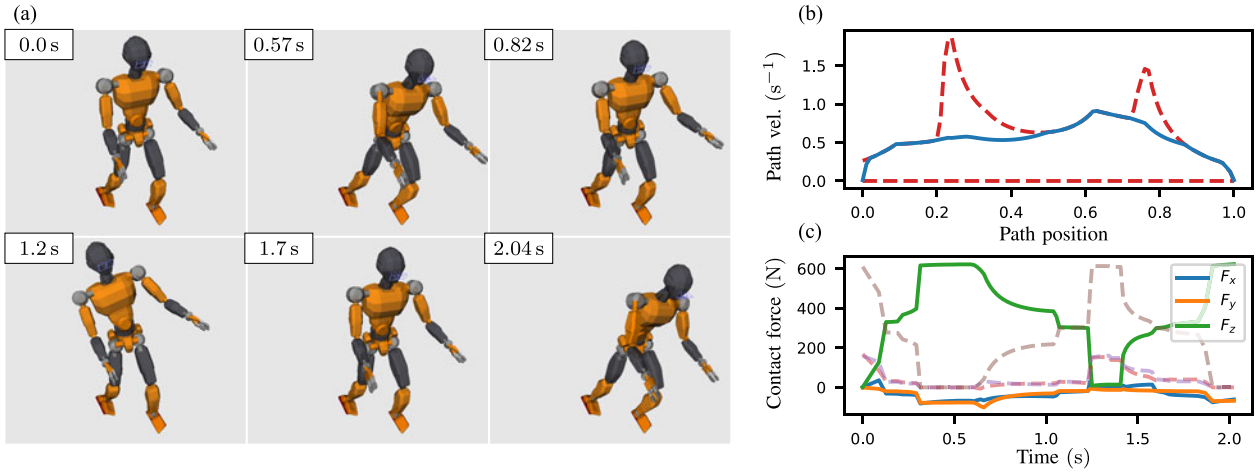

Fig. 6. Time parameterization of a legged robot trajectory under joint torque bounds and multicontact friction constraints. (a) Snapshots of the retimed motion. (b) Optimal velocity profile computed by TOPP-RA (blue) and upper and lower limits of the controllable sets (dashed red). (c) Optimal joint torques and contact forces are obtained "for free" as slack variables of the optimization programs solved in the forward pass. Net contact forces for the left foot are shown in colors and those for the right foot are shown in transparent lines.

greatest constraint satisfaction errors by comparing the sampled joint accelerations and velocities to their respective bounds. Then, we averaged instances with the same grid size to obtain the average error for each N.

Fig. 5(b) shows the average relative greatest constraint satisfaction errors of the three algorithms with respect to the grid size. One can observe that TOPP-RA and TOPP-NI had constraint satisfaction errors of the same order of magnitude for $N < 500$, while TOPP-RA demonstrated better quality for $N \geq 500$. TOPP-RA-intp produced solutions with much higher quality. This result confirms our error analysis of different discretization schemes in Appendix D and demonstrates that the interpolation discretization scheme is better than the collocation scheme whenever solution quality is concerned.

Fig. 5(c) shows the average difference between the costs of solutions returned by TOPP-RA and TOPP-RA-intp with the *true* optimal cost, which was approximated by running TOPP-RA-intp with grid size $N = 10\,000$. One can observe that both algorithms are asymptotically optimal. Even more importantly, the differences were relatively small even at coarse grid sizes.

B. Experiment 2: Legged Robot in Multicontact

Here, we consider the TOPP problem for a 50-dof legged robot subject to joint torque bounds and contact stability constraints with linearized friction cones.

1) Formulation: We now give a brief description of our formulation, for more details, refer to [10] and [11]. Let $\mathbf{w}_i$ denote the net contact wrench (force–torque pair) exerted on the robot by the i-th contact at point $\mathbf{p}_i$. Using the linearized friction cone, one obtains the set of feasible wrenches as a polyhedral cone

$$\{\mathbf{w}_i \mid \mathbf{F}_i \mathbf{w}_i \leq 0\}$$

for some matrix $\mathbf{F}_i$. This matrix can be found using the cone double description method [13], [21]. Combining with the equation governing rigid-body dynamics, one obtains the full dynamic feasibility constraint as follows:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{q}}^\top \mathbf{C}(\mathbf{q})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau} + \sum_{i=1,2} \mathbf{J}_i(\mathbf{q})^\top \mathbf{w}_i$$

$$\mathbf{F}_i \mathbf{w}_i \leq 0$$

$$\boldsymbol{\tau}_{\min} \leq \boldsymbol{\tau} \leq \tau_{\max}$$

where $\mathbf{J}_i(\mathbf{q})$ is the wrench Jacobian. The convex set $\mathscr{C}(\mathbf{q})$ in (1) can now be identified as a multidimensional polyhedron.

We considered a simple swaying motion: the robot stands with both feet lie flat on two uneven steps and shift its body back and forth, see Fig. 6. The coefficient of friction was set to $\mu = 0.5$. Start and end path velocities were set to zero. Discretization grid size was $N = 100$. The number of constraint inequalities was $m = 242$.

2) Results: Excluding computations of dynamic quantities, TOPP-RA took 267 ms to solve the TOPP instance. The parameterization is shown in Fig. 6. Computation time is given in Table II.

Compared to TOPP-NI and TOPP-CO, TOPP-RA had significantly better computation time, chiefly because both existing methods require an expensive polytopic projection step. Indeed, [10] reported projection time of 2.4 s for a similar sized problem, which is significantly more expensive than the TOPP-RA computation time. Notice that, in [10], computing the parameterization takes an addition 2.46 s, which leads to a total computation time of 4.86 s.

To make a more accurate comparison, we implement the pipeline given in [11] to solve the TOPP instance as follows:

1) project the constraint polyhedron $\mathscr{C}_i$ onto the path using Bretl's polygon recursive expansion algorithm [18];
2) parameterize the resulting problem using TOPP-NI.

This pipeline turned out to be much slower than TOPP-RA. We found that the number of LPs the projection step solved was nearly eight times the number of LPs solved by TOPP-RA (which is fixed at $3N = 300$). For a more detailed comparison

TABLE II
COMPUTATION TIME (MS) AND INTERNAL PARAMETERS COMPARISON BETWEEN TOPP-RA, TOPP-RA-INTP, AND TOPP-NI IN EXPERIMENT 2

	TOPP-RA	TOPP-RA-intp	TOPP-NI
	Time (ms)		
comp. dynamic quantities	181.6	193.6	281.6
polytopic projection	0.0	0.0	3671.8
solve TOPP	267.0	1619.0	335.0
extract trajectory	3.0	3.0	210.0
total	451.6	1815.6	4497.8
	Parameters		
joint torques / contact forces avail.	yes	yes	no
No. of LP(s) solved	300	300	2110
No. of variables	64	126	64
No. of constraints	242	476	242
Constraints sat. error	$O(\Delta)$	$O(\Delta^2)$	$O(\Delta)$

of the computation time and parameters of the LPs, refer to Table II.

3) Obtaining Joint Torques and Contact Forces "For Free": Another interesting feature of TOPP-RA is that the algorithm can optimize and obtain joint torques and contact forces "for free" without additional processing. Concretely, since joint torques and contact forces are slack variables, one can simply store the optimal slack variable at each step and obtain a trajectory of feasible torques and forces. To optimize the torques and forces, we can modify the ith step of the forward pass to solve the following quadratic program:

$$\begin{aligned} \min \quad & -u + \epsilon \|(\mathbf{w}, \boldsymbol{\tau})\|_2^2 \\ \text{s.t.} \quad & x = x_i \\ & (u, x) \in \Omega_i \\ & x + 2\Delta_i u \in K_{i+1} \end{aligned}$$

where ϵ is a positive scalar. Fig. 6 (lower plot) shows computed contact wrench for the left leg. We note that both existing approaches, TOPP-NI and TOPP-CO are not able to produce joint torques and contact forces readily as they "flatten" the constraint polygon in the projection step.

In fact, the aforementioned formulation suggests that time optimality is simply a specific objective cost function (linear) of the more general family of quadratic objectives. Therefore, one can in principle depart from time optimality in favor of more realistic objective such as minimizing torque while maintaining a certain nominal velocity x_{norm} as follows:

$$\begin{aligned} \min \quad & \|x_i + 2\Delta_i u - x_{\text{norm}}\|_2^2 + \epsilon \|(\mathbf{w}, \boldsymbol{\tau})\|_2^2 \\ \text{s.t.} \quad & x = x_i \\ & (u, x) \in \Omega_i \\ & x + 2\Delta_i u \in K_{i+1}. \end{aligned}$$

Finally, we observed that the choice of the path discretization scheme has noticeable effects on both computational cost and quality of the result. In general, TOPP-RA-intp produced smoother trajectories and better (lower) constraint satisfaction error at the cost of longer computation time. On the other hand, TOPP-RA was faster but produced trajectories with jitters[3] near dynamic singularities [4] and had higher constraint satisfaction error.

VI. ADDITIONAL BENEFITS OF TOPP BY RA

We now elaborate on the additional benefits provided by the RA approach to TOPP.

A. Admissible Velocity Propagation

Admissible velocity propagation (AVP) is a recent concept for kinodynamic motion planning [3]. Specifically, given a path and an initial interval of velocities, AVP returns exactly the interval of all the velocities the system can reach after traversing the path while respecting the system kinodynamic constraints. Combined with existing *geometric* path planners, such as RRT [22], this can be advantageously used for *kinodynamic* motion planning: at each tree extension in the configuration space, AVP can be used to guarantee the eventual existence of admissible path parameterizations.

Suppose that the initial velocity interval is $\mathbb{I}_0$. It can be immediately seen that, what is computed by AVP is exactly the reachable set $\mathcal{L}_N(\mathbb{I}_0)$ (cf., Section III-B). Furthermore, what is computed by AVP-Backward [23] given a desired final velocity interval $\mathbb{I}_N$ is exactly the controllable set $\mathcal{K}_0(\mathbb{I}_N)$ (cf., Section III-C). In terms of complexity, $\mathcal{R}_N(\mathbb{I}_0)$ and $\mathcal{K}_0(\mathbb{I}_N)$ can be found by solving, respectively, $2N+2$ and $2N+2$ LPs. We have thus rederived the concept of AVP at no cost.

B. Robustness to Parametric Uncertainty

In most works dedicated to TOPP, including the development of this paper up to this point, the parameters appearing in the dynamics equations and in the constraints are supposed to be exactly known. In reality, those parameters, which include inertia matrices or payloads in robot manipulators, or feet positions or friction coefficients in legged robots, are only known up to some precision. An admissible parameterization for the nominal values of the parameters might not be admissible for the actual values, and the probability of constraints violation is even higher in the *optimal* parameterization, which saturates at least one constraint at any moment in time.

TOPP-RA provides a natural way to handle parametric uncertainties. Assume that constraints appear in the following form:

$$\begin{aligned} & \mathbf{a}_i u + \mathbf{b}_i x + \mathbf{c}_i \in \mathscr{C}_i \\ & \forall (\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i, \mathscr{C}_i) \in \mathcal{E}_i \end{aligned} \tag{11}$$

where $\mathcal{E}_i$ contains all the possible values that the parameters might take at path position s_i.

Implementation remark 6: Consider, for instance, the manipulator with torque bounds of (2). Suppose that, at path position i, the inertia matrix is uncertain, i.e., that it might

[3]Experiments prove that singularities do not cause failures for TOPP-RA. The jitters can usually be removed easily, i.e., using cubic splines interpolation to smooth the parametrization locally around jitters.

take any values $\mathbf{M}_i \in B(\mathbf{M}_i^{\text{nominal}}, \epsilon)$, where $B(\mathbf{M}_i^{\text{nominal}}, \epsilon)$ denotes the ball of radius ϵ centered around $\mathbf{M}_i^{\text{nominal}}$ for the max norm. Then, the first component of $\mathcal{E}_i$ is given by $\{\mathbf{M}_i \mathbf{q}'(s_i) \mid \mathbf{M}_i \in B(\mathbf{M}_i^{\text{nominal}}, \epsilon)\}$, which is a convex set.

In legged robots, uncertainties on feet positions or on friction coefficients can be encoded into a "set of sets," in which $\mathscr{C}_i$ can take values.

TOPP-RA can handle this situation by suitably modifying its two passes. Before presenting the modifications, we first give some definitions. Denote the i-stage set of *robust admissible* control-state pairs by

$$\widehat{\Omega}_i := \{(u, x) \mid (11) \text{ holds}\}.$$

The sets of robust admissible states $\widehat{\mathcal{X}}_i$ and robust admissible controls $\widehat{\mathcal{U}}_i(x)$ can be defined as in Section III-A.

In the backward pass, TOPP-RA computes the *robust controllable sets*, whose definition is given as follows.

Definition 6 (i-stage: ***robust*** **controllable set)** Consider a set of desired ending states $\mathbb{I}_N$. The *i-stage robust controllable set* $\widehat{\mathcal{K}}_i(\mathbb{I}_N)$ is the set states $x \in \widehat{\mathcal{X}}_i$ such that there exists a state $x_N \in \mathbb{I}_N$ and a sequence of *robust admissible controls* $u_i, \ldots, u_{N-1}$ that steers the system from x to x_N.

To compute the robust controllable sets, one needs the robust one-step set.

Definition 7 (: ***Robust*** **one-step set)** Consider a set of states $\mathbb{I}$. The *robust one-step set* $\widehat{\mathcal{Q}}_i(\mathbb{I})$ is the set of states $x \in \widehat{\mathcal{X}}_i$ such that there exists a state $\tilde{x} \in \mathbb{I}$ and a robust admissible control $u \in \widehat{\mathcal{U}}_i(x)$ that steers the system from x to $\tilde{x}$.

Finally, in the forward pass, the algorithm selected the *greatest* robust admissible control at each stage.

Implementation remark 7: Computing the robust one-step set and the greatest robust admissible control involves solving LPs with uncertain constraints of the form (11). In the mathematical optimization literature, they are known as "robust linear programs," and specific methods have been developed to handle them efficiently, when the robust constraints are [24]

1) polyhedra;
2) ellipsoids;
3) conic quadratic representable sets.

The first case can be treated as normal LPs with appropriate slack variables, while the last two cases are explicit conic quadratic program. For more information on this conversion, refer to [24, chs. 1 and 2]. ◊

VII. Conclusion

We have presented a new approach to solve the TOPP problem based on RA (TOPP-RA). The key insight is to compute, in a first pass, the sets of controllable states, for which admissible controls allowing to reach the goal are guaranteed to exist. Time optimality can then be obtained, in a second pass, by a greedy strategy. We have shown, through theoretical analyses and extensive experiments, that the proposed algorithm is extremely robust (100% success rate) and competitive in terms of the computation time as compared to the fastest known TOPP implementation [4], and produces solutions with high quality. Finally, the new approach yields additional benefits: no need for polytopic projection in the redundantly actuated case, admissible velocity projection, and robustness to parameter uncertainty.

A recognized disadvantage of the classical TOPP formulation is that the time-optimal trajectory contains hard acceleration switches, corresponding to infinite jerks. Solving TOPP subject to jerk bounds, however, is not possible using the CO-based approach as the problem becomes nonconvex [9]. Some prior works proposed to either extend the NI-based approach [25], [26] or to represent the parameterization as a spline and optimize directly over the parameter space [27], [28]. Exploring how RA can be extended to handle jerk bounds is another direction of our future research.

Similar to the CO-based approach, RA can only be applied to instances with convex constraints [9]. Yet in practice, it is often desirable to consider, in addition, nonconvex constraints, such as joint torque bounds with viscous friction effect. Extending RA to handle nonconvex constraints is another important research question.

Appendix

A. Relation Between TOPP-RA and TOPP-NI

TOPP-RA and TOPP-NI are subtly related: they compute the same velocity profiles, but in different orders. Let us first recall some terminologies from the literature of the NI-based approach to TOPP (see [4] for more details).

1) *MVC*: Mapping from path position to the highest dynamically feasible path velocity.
2) *Integrate forward (or backward) following α (or β)*: For each tuple $(s, \dot{s})$, $\alpha(s, \dot{s})$, and $\beta(s, \dot{s})$ are the smallest and largest controls, respectively; forward and backward integrations are done following the respective controls.
3) *$\alpha \to \beta$ switch points*: There are three kinds of $\alpha \to \beta$ switch points: tangent, singular, and discontinuous.
4) $\dot{s}_{\text{beg}}, \dot{s}_{\text{end}}$: Starting and ending velocities at s_{beg} and s_{end}.

Note that α and β functions recalled previously are different from the functions defined in Definition 8 (Appendix B). The formers maximize over the set of feasible states, while the laters maximize over the set of feasible *and controllable* states.

TOPP-NI proceeds as follows:

1) determine the $\alpha \to \beta$ switch points;
2) from each $\alpha \to \beta$ switch point, integrate forward following β and backward following α to obtain the limiting curves (LCs);
3) take the lowest value of the LCs at each position to form the concatenated limiting curve (CLC);
4) from $(0, \dot{s}_{\text{beg}})$ integrate forward following β; from $(s_{\text{end}}, \dot{s}_{\text{end}})$ integrate backward following α until their intersections with the CLC; then return the combined $\beta - \text{CLC} - \alpha$ profile.

We now rearrange the aforementioned steps into a backward pass and a forward pass in order to highlight the relation with TOPP-RA.

Backward pass

1) determine the $\alpha \to \beta$ switch points;
2a) from each $\alpha \to \beta$ switch point, integrate backward following α to obtain the backward limiting curves (BLCs);

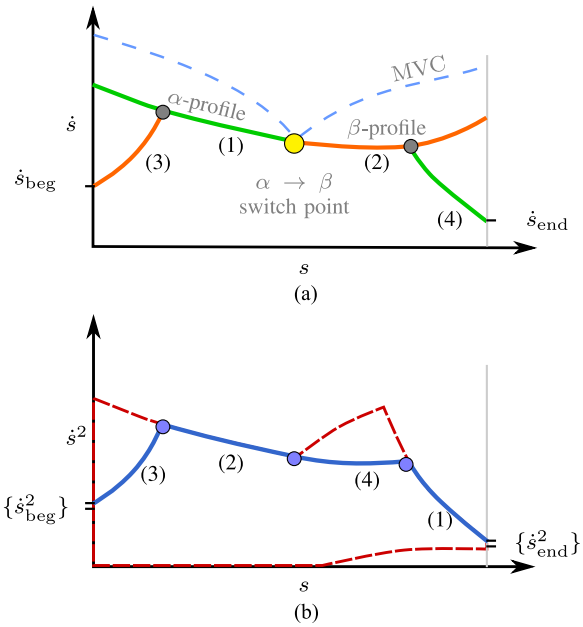

Fig. 7. (a) TOPP-NI and (b) TOPP-RA compute the time-optimal path parameterization by creating similar profiles in different ordering. (1, 2, 3, 4) is the order in which TOPP-RA computes the profiles.

2b) from the point $(s_{\text{end}}, \dot{s}_{\text{end}})$ integrate backward following α to obtain the last BLC;
3) take the lowest value of the BLC's *and the* MVC to form the upper boundary of the controllable sets.

Forward pass

4a) set the current point to the point $(s_{\text{beg}}, \dot{s}_{\text{beg}})$;
4b) repeat until the current point is the point $(s_{\text{end}}, \dot{s}_{\text{end}})$, from the current point integrate forward following β until hitting a BLC, set the corresponding switch point as the new current point.

See Fig. 7 for a visualization of the rearrangement.

The key idea in this rearrangement is to not compute β profiles immediately for each switch point, but delay until needed. The resulting algorithm is almost identical to TOPP-RA except for the following points.

1) Since TOPP-RA does not require explicit computation of the switch points (they are implicitly identified by computing the controllable sets), the algorithm avoids one of the major implementation difficulties of TOPP-NI.
2) TOPP-RA requires additional postprocessing to remove jitters. See the footnote in Section V-B3 for more details.

B. Proof of Optimality (Case With No Zero-Inertia Point)

The optimality of TOPP-RA relies on the properties of the maximal transition functions.

Definition 8: At a given stage i, the minimal and maximal controls at state x are defined by [4]

$$\alpha_i(x) := \min\{u \mid \bar{\mathbf{a}}_i u + \bar{\mathbf{b}}_i x + \bar{\mathbf{c}}_i \leq 0\}$$
$$\beta_i(x) := \max\{u \mid \bar{\mathbf{a}}_i u + \bar{\mathbf{b}}_i x + \bar{\mathbf{c}}_i \leq 0\}.$$

[4]These definitions differ from the common definitions of maximal and minimal controls. See Appendix A for more details.

The *minimal and maximal transition functions* are defined by

$$T_i^{\alpha}(x) := x + 2\Delta\alpha_i(x), \quad T_i^{\beta}(x) := x + 2\Delta\beta_i(x). \qquad \Diamond$$

The key observation is: if the maximal transition function is *nondecreasing*, then the greedy strategy of TOPP-RA is optimal. This is made precise by the following lemma.

Lemma 1: If for all i, the maximal transition function is nondecreasing, i.e.,

$$\forall x, x'' \in \mathcal{K}_i, \quad x \geq x' \Rightarrow T_i^{\beta}(x) \geq T_i^{\beta}(x')$$

then TOPP-RA produces the optimal parameterization.

Proof: Consider an arbitrary admissible parameterization $\dot{s}_0^2 = x_0, u_0, \ldots, u_{N-1}, x_N = \dot{s}_N^2$. We show by induction that, for all $i = 0, \ldots, N$, $x_i^* \geq x_i$, where the sequence (x_i^*) denotes the parameterization returned by TOPP-RA (Algorithm 1).

Initialization: One has $x_0 = \dot{s}_0^2 = x_0^*$, so the assertion is true at $i = 0$.

Induction: Steps 8 and 9 of Algorithm 1 can, in fact, be rewritten as follows:

$$x_{i+1}^* := \min\{T_i^{\beta}(x_i^*), \max(\mathcal{K}_{i+1})\}.$$

By the induction hypothesis, one has $x_i^* \geq x_i$. Since $x_i^*, x_i \in \mathcal{K}_i$, one has

$$T_i^{\beta}(x_i^*) \geq T_i^{\beta}(x_i) \geq x_{i+1}.$$

Thus

$$\min\{T_i^{\beta}(x_i^*), \max(\mathcal{K}_{i+1})\} \geq \min\{x_{i+1}, \max(\mathcal{K}_{i+1})\}, \text{ i.e.,}$$
$$x_{i+1}^* \geq x_{i+1}.$$

We have shown that at every stage the parameterization $x_0^*, \ldots, x_N^*$ has a higher velocity than that of any admissible parameterization, hence it is optimal.

Unfortunately, the maximal transition function is not always nondecreasing, as made clear by the following lemma.

Lemma 2: Consider a stage i, there exists x_i^{β} such that for all $x, x' \in \mathcal{K}_i$

$$x \leq x' \leq x_i^{\beta} \Rightarrow T_i^{\beta}(x) \leq T_i^{\beta}(x')$$
$$x_i^{\beta} \leq x \leq x' \Rightarrow T_i^{\beta}(x) \geq T_i^{\beta}(x').$$

In other words, T_i^{β} is nondecreasing below x_i^{β} and is nonincreasing above x_i^{β}.

Similarly, there exists x_i^{α} such that for all $x, x' \in \mathcal{K}_i$

$$x_i^{\alpha} \leq x \leq x' \Rightarrow T_i^{\alpha}(x) \leq T_i^{\alpha}(x')$$
$$x \leq x' \leq x_i^{\alpha} \Rightarrow T_i^{\alpha}(x) \geq T_i^{\alpha}(x').$$

Proof: Consider a state x. In the (u, x) plane, draw a horizontal line at height x. This line intersects the polygon $\Omega_i \cap (\mathbb{R} \times \mathcal{K}_i)$ at the minimal and maximal controls. See Fig 8.

Consider now the polygon $\Omega_i \cap (\mathbb{R} \times \mathcal{K}_i)$. Suppose that one enumerates the edges counter-clockwise (ccw), then the normals of the enumerated edges also rotate ccw. For example, in Fig. 8, the normal v_1 of edge 1 can be obtained by rotating ccw the normal v_2 of edge 2.

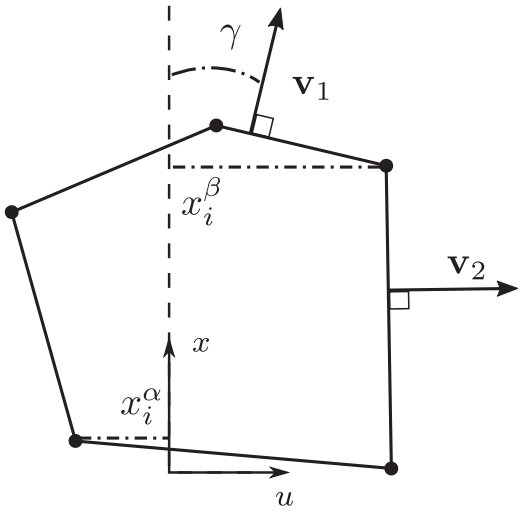

Fig. 8. At any stage, the polygon of controllable states and controls $\Omega_i \cap (\mathbb{R} \times \mathcal{K}_i)$ contains x_i^β: The highest state under which the transition function T_i^β is nondecreasing, and x_i^α: The lowest state above which the transition function T_i^α is nondecreasing.

Let γ denote the angle between the vertical axis and the normal vector of the active constraint k at $(x, \beta(x))$. One has $\cot\gamma = \bar{\mathbf{b}}_i[k]/\bar{\mathbf{a}}_i[k]$.

As x increases, γ decreases in the interval $(\pi, 0)$. Let x_i^β be the lowest x such that, for all $x > x_i^\beta$, $\gamma < \cot^{-1}(1/(2\Delta))$ ($x_i^\beta := \max \mathcal{K}_i$ if there is no such x).

Consider now a $x > x_i^\beta$, one has, by construction

$$\frac{\bar{\mathbf{b}}_i[k]}{\bar{\mathbf{a}}_i[k]} > \frac{1}{2\Delta} \tag{12}$$

where k is the active constraint at $(x, \beta_i(x))$. The maximal transition function can be written as

$$\begin{aligned} T_i^\beta(x) &= x + 2\Delta\beta_i(x) \\ &= x + 2\Delta\frac{-\bar{\mathbf{c}}_i[k] - \bar{\mathbf{b}}_i[k]x}{\bar{\mathbf{a}}_i[k]} \\ &= x\left(1 - 2\Delta\frac{\bar{\mathbf{b}}_i[k]}{\bar{\mathbf{a}}_i[k]}\right) - 2\Delta\frac{\bar{\mathbf{c}}_i[k]}{\bar{\mathbf{a}}_i[k]}. \end{aligned} \tag{13}$$

Since the coefficient of x is negative, $T_i^\beta(x)$ is nonincreasing. Similarly, for $x \le x_i^\beta$, $T_i^\beta(x)$ is nondecreasing.

We are now ready to prove Theorem 2.

Proof of Theorem 2 As there is no zero-inertia point, by uniform continuity, the $\bar{\mathbf{a}}(s)[k]$ are bounded away from 0. We can thus chose a step size Δ_{thr} such that

$$\frac{1}{2\Delta_{\text{thr}}} > \max_{s,k}\left\{\frac{\bar{\mathbf{b}}(s)[k]}{\bar{\mathbf{a}}(s)[k]} \mid \bar{\mathbf{a}}(s)[k] > 0\right\}.$$

For any step size $\Delta < \Delta_{\text{thr}}$, there is by construction no constraint that can have an angle $\gamma < \cot^{-1}(1/(2\Delta))$. Thus, for all stages i, one has $x_i^\beta = \max \mathcal{K}_i$, or in other words, that $T_i^\beta(x)$ is nondecreasing in the whole set $\mathcal{K}_i$. By Lemma 1, TOPP-RA returns the optimal parameterization.

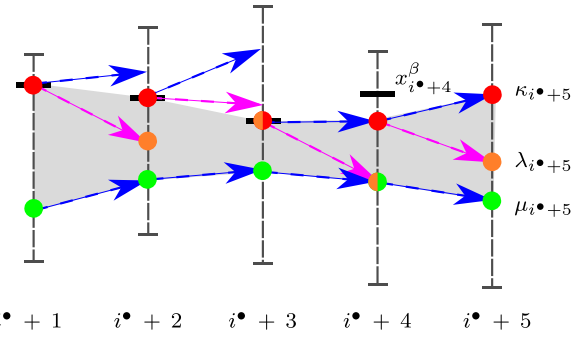

Fig. 9. "Perturbation strip" contains three vertical boundaries: (κ_i) [red dots], (λ_i) [orange dots], and (μ_i) [green dots]. The states (x_i^β) [thick horizontal black lines] and the controllable sets $(\mathcal{K}_i)$ [vertical intervals] are both shown.

C. Proof of Asymptotic Optimality (Case With Zero-Inertia Points)

In the presence of a zero-inertia point, one cannot bound the $\bar{\mathbf{a}}(s)[k]$ away from zero. Therefore, for any step size Δ, there is an interval around the zero-inertia point where the maximal transition function is *not* monotonic over the whole controllable set $\mathcal{K}_i$. Our strategy is to show that the suboptimality gap caused by that interval decreases to 0 with Δ.

We first identify a "perturbation interval." For simplicity, assume that the zero-inertia point $s^\bullet$ is exactly at $s_{i^\bullet}$.

Lemma 3: There exists an integer l such that, for small enough Δ, the maximal transition function is nondecreasing at all stages except in $[i^\bullet + 1, \dots, i^\bullet + l]$.

Proof: Consider the Taylor expansion around $s^\bullet$ of the constraint that triggers the zero-inertia point

$$\begin{aligned} \bar{\mathbf{a}}(s)[k] &= A'(s - s^\bullet) + o(s - s^\bullet) \\ \bar{\mathbf{b}}(s)[k] &= B + B'(s - s^\bullet) + o(s - s^\bullet). \end{aligned}$$

Without loss of generality, suppose $A' > 0$. Equation (12) can be written for stage $i^\bullet + r$ as follows:

$$2\Delta(B + B'r\Delta) > A'r\Delta + o(r\Delta).$$

Thus, in the limit $\Delta \to 0$, for $r > l := \text{ceil}(2B/A')$, (12) will not be fulfilled by constraint k. Using the construction of Δ_{thr} in the proof of Theorem 2, one can next rule out all the other constraints at all stages.

We now construct a "perturbation strip" by defining an upper and an lower boundaries. See, Fig. 9, for an illustration.

Definition 9: Define states $(\kappa_i)_{i\in[i^\bullet+1, i^\bullet+l+1]}$ by

$$\begin{aligned} \kappa_{i^\bullet+1} &:= x_{i^\bullet+1}^\beta \\ \kappa_i &:= \min(T_{i-1}{}^\beta(\kappa_{i-1}), x_i^\beta),\ i = i^\bullet + 2, \dots, i^\bullet + l + 1. \end{aligned}$$

Next, define $(\lambda_i)_{i\in[i^\bullet+1, i^\bullet+l+1]}$ by

$$\begin{aligned} \lambda_{i^\bullet+1} &= \kappa_{i^\bullet+1} \\ \lambda_i &= \min(T_{i-1}^\alpha(\kappa_{i-1}), x_i^\beta),\ i = i^\bullet + 2, \dots, i^\bullet + l + 1. \end{aligned}$$

Finally, define $(\mu_i)_{i\in[i^\bullet+1,i^\bullet+l+1]}$ as the highest profile that can be obtained by repeated applications of T^β and that remains below the (λ_i).

The (κ_i) and (μ_i) form, respectively, the upper and the lower boundaries of the "perturbation strip." Before going further, let us establish some estimates on the size of the strip.

Lemma 4: There exist constants C_κ and C_μ such that for all $i \in [i^\bullet+1, i^\bullet+l+1]$

$$\max \mathcal{K}_{i^\bullet+1} - \kappa_i \leq lC_\kappa\Delta \tag{14}$$

$$\max \mathcal{K}_{i^\bullet+1} - \mu_i \leq lC_\mu\Delta. \tag{15}$$

Proof: Let C be the upper bound of the absolute values of all admissible controls α and β over whole segment. One has

$$\mathcal{K}_{i^\bullet+1} - \kappa_{i^\bullet+1} = \mathcal{K}_{i^\bullet+1} - x^\beta_{i^\bullet+1}$$
$$\leq (\beta_i(x^\beta_{i^\bullet+1}) - \alpha_i(x^\beta_{i^\bullet+1}))\tan(\gamma) \leq 2C\Delta.$$

Next, by definition of κ, one can see that the difference between two consecutive κ_i, κ_{i+1} is bounded by $2C\Delta$. This shows (14).

Since (μ_i) is the highest profile below (λ_i), there exists one index p such that $\mu_p = \lambda_p$. Thus, $\kappa_p - \mu_p = \kappa_p - \lambda_p \leq 2C\Delta$, where the last inequality comes from the definition of λ. Remark, finally, that the difference between two consecutive μ_i, μ_{i+1} is also bounded by $2C\Delta$. This shows (15). ■

We now establish two properties of the "perturbation strip."

Lemma 5 (and definition): Let $J^*_i(x)$ denote TOPP-RA's cost-to-go: The cost of the profile produced by TOPP-RA starting from x at the i-stage, and $J^\dagger_i(x)$ the optimal cost-to-go.

1) In the interval $[\min \mathcal{K}_i, \mu_i]$, $J^*_i(x)$ equals $J^\dagger_i(x)$ and is nonincreasing.
2) For all $i \in [i^\bullet+1, \ldots, i^\bullet+l]$

$$x \in [\mu_i, \kappa_i] \Rightarrow T^\dagger_i(x), T^\beta_i(x) \in [\mu_{i+1}, \kappa_{i+1}] \tag{16}$$

where $T^\dagger_i(x)$ is the optimal transition.

Proof: 1) We use backward induction from $i^\bullet+l$ to $i^\bullet+1$.

Initialization: One has $x \leq \mu_{i^\bullet+l} \leq \lambda_{i^\bullet+l} \leq x^\beta_{i^\bullet+l}$. It follows that $T^\beta_{i^\bullet+l}(x)$ is nondecreasing over the interval $[\min \mathcal{K}_{i^\bullet+l}, \mu_{i^\bullet+l}]$.

As there is no constraint verifying (12) at stages $i = i^\bullet+l+1, \ldots, N$, the cost-to-go $J^*_{i^\bullet+l+1}(x)$ is nonincreasing and equals the optimal cost-to-go $J^\dagger_{i^\bullet+l+1}(x)$ by Theorem 2. Choosing the greedy control at the $i^\bullet+l$-stage is, therefore, optimal. Next, note that

$$J^*_{i^\bullet+l}(x) = \frac{\Delta}{\sqrt{x}} + J^*_{i^\bullet+l+1}(T^\beta_{i^\bullet+l}(x)) \tag{17}$$

since $J^*_{i^\bullet+l+1}(x)$ and $T^\beta_{i^\bullet+l}(x)$ are nonincreasing and nondecreasing, respectively, over $[\min \mathcal{K}_{i^\bullet+l+1}, \mu_{i^\bullet+l+1}]$ and $[\min \mathcal{K}_{i^\bullet+l}, \mu_{i^\bullet+l}]$, it follows that $J^*_{i^\bullet+l}(x)$ is nonincreasing over the interval $[\min \mathcal{K}_{i^\bullet+l}, \mu_{i^\bullet+l}]$.

Induction: Suppose the hypothesis is true for $i+1 \in \{i^\bullet+2, \ldots, i^\bullet+l\}$. Since $x \leq \mu_i \leq \lambda_i \leq x^\beta_i$, one has that $T^\beta_i(x) \leq \mu_{i+1}$ and that $T^\beta_i(x)$ is nondecreasing over the interval $[\min \mathcal{K}_i, \mu_i]$. Note that

$$J^*_i(x) = \frac{\Delta}{\sqrt{x}} + J^*_{i+1}(T^\beta_i(x)). \tag{18}$$

By the induction hypothesis, $J^*_{i+1}(T^\beta_i(x))$ is nonincreasing, it then follows that $J^*_i(x)$ is nondecreasing and that $\beta_i(x)$ is the optimal control, and thus, $J^*_i(x) = J^\dagger_i(x)$.

2) The part that $T^\dagger_i(x), T^\beta_i(x) \leq \kappa_{i+1}$ is clear from the definition of κ. We first show $\mu_{i+1} \leq T^\beta_i(x)$.

Suppose, first, $x \leq x^\beta_i$. Then, T^β_i is nondecreasing in $[\mu_i, x]$, which implies $T^\beta_i(x) \geq T^\beta_i(\mu_i) = \mu_{i+1}$.

Suppose now $x \geq x^\beta_i$. One can choose a step size Δ such that $x^\alpha_i < x^\beta_i$, which implies $x > x^\alpha_i$. One then has $T^\beta_i(x) \geq T^\alpha_i(x) \geq T^\alpha_i(x^\beta_i) \geq \lambda_{i+1} \geq \mu_{i+1}$.

Finally, to show that $\mu_{i+1} \leq T^\dagger_i(x)$, we reason by contradiction. Suppose $T^\dagger_i(x) < \mu_{i+1}$. By (a), $J^\dagger_{i+1}$ is nonincreasing below μ_{i+1}, thus $J^\dagger_{i+1}(T^\dagger_i(x)) > J^\dagger_{i+1}(\mu_{i+1})$ (*). On the other hand, since $T^\dagger_i(x) < \mu_{i+1} \leq T^\beta_i(x)$, there exists an admissible control that steers x toward μ_{i+1}. Since $T^\dagger_i$ is the true optimal transition from x, $J^\dagger_{i+1}(\mu_{i+1}) \geq J^\dagger_{i+1}(T^\dagger_i(x))$. This contradicts (*). ■

We are now ready to prove Theorem 3.

Proof of Theorem 3 Recall that $(\dot{s}^2_0 = x^*_0, \ldots, x^*_N)$ is the profile returned by TOPP-RA and $(\dot{s}^2_0 = x^\dagger_0, \ldots, x^\dagger_N)$ is the true optimal profile. By definition of the time-optimal cost functions, we can expand the initial costs $J^*_0(\dot{s}^2_0)$ and $J^\dagger_0(\dot{s}^2_0)$ into three terms as follows:

$$J^*_0(\dot{s}^2_0) = \sum_{i=0}^{i^\bullet} \frac{\Delta}{\sqrt{x^*_i}} + \sum_{i=i^\bullet+1}^{i^\bullet+l} \frac{\Delta}{\sqrt{x^*_i}} + J^*_{i^\bullet+l+1}(x^*_{i^\bullet+l+1}) \tag{19}$$

and

$$J^\dagger_0(\dot{s}^2_0) = \sum_{i=0}^{i^\bullet} \frac{\Delta}{\sqrt{x^\dagger_i}} + \sum_{i=i^\bullet+1}^{i^\bullet+l} \frac{\Delta}{\sqrt{x^\dagger_i}} + J^\dagger_{i^\bullet+l+1}(x^\dagger_{i^\bullet+l+1}) \tag{20}$$

1) Applying Theorem 2, for small enough Δ, one can show that

$$\forall i \in [0, \ldots, i^\bullet+1],\ x^*_i \geq x^\dagger_i. \tag{21}$$

Thus, the first term of $J^*_0(\dot{s}^2_0)$ is smaller than the first term of $J^\dagger_0(\dot{s}^2_0)$.

2) Suppose $x^*_{i^\bullet+1}, x^\dagger_{i^\bullet+1} \in [\mu_{i^\bullet+1}, \kappa_{i^\bullet+1}]$. From Lemma 5(b), one has for all $i \in [i^\bullet+1, \ldots, i^\bullet+l]$, $x^*_i, x^\dagger_i \in [\mu_i, \kappa_i]$. Thus, using the estimates of Lemma 4, the second terms can be bounded as follows:

$$\sum_{i=i^\bullet+1}^{i^\bullet+l} \frac{\Delta}{\sqrt{x^*_i}} \leq \frac{l\Delta}{\sqrt{\max \mathcal{K}_{i^\bullet+1} - C_\mu\Delta l}}, \text{ and}$$

$$\sum_{i=i^\bullet+1}^{i^\bullet+l} \frac{\Delta}{\sqrt{x^\dagger_i}} \geq \frac{l\Delta}{\sqrt{\max \mathcal{K}_{i^\bullet+1} + C_\kappa\Delta l}}.$$

Thus

$$\sum_{i=i^\bullet+1}^{i^\bullet+l} \frac{\Delta}{\sqrt{x_i^*}} - \sum_{i=i^\bullet+1}^{i^\bullet+l} \frac{\Delta}{\sqrt{x_i^\dagger}} \leq \frac{(C_\mu + C_\kappa)\Delta^2 l^2}{2\sqrt{\max \mathcal{K}_{i^\bullet+1} - C_\mu \Delta l}}.$$

If $x^*_{i^\bullet+1}, x^\dagger_{i^\bullet+1} < \mu_{i^\bullet+1}$, by Lemma 5(a), it is easy to see that $J_0^*(\dot{s}_0^2) = J_0^\dagger(\dot{s}_0^2)$.

If $x^*_{i^\bullet+1} \geq \mu_{i^\bullet+1} > x^\dagger_{i^\bullet+1}$, then by Lemma 5, $x^*_{i^\bullet+l+1} \geq \mu_{i^\bullet+l+1} > x^\dagger_{i^\bullet+l+1}$, which implies next that $J_0^*(\dot{s}_0^2) < J_0^\dagger(\dot{s}_0^2)$, which is impossible.

3) Regarding the third terms, observe that, by applying Theorem 2 over $[i^\bullet + l + 1, \ldots, N]$, one has $J^*_{i^\bullet+l+1}(x) = J^\dagger_{i^\bullet+l+1}(x)$ for all $x \in \mathcal{K}_{i^\bullet+l+1}$. Thus

$$\begin{aligned} &J^*_{i^\bullet+l+1}(x^*_{i^\bullet+l+1}) - J^\dagger_{i^\bullet+l+1}(x^\dagger_{i^\bullet+l+1}) \\ &= J^\dagger_{i^\bullet+l+1}(x^*_{i^\bullet+l+1}) - J^\dagger_{i^\bullet+l+1}(x^\dagger_{i^\bullet+l+1}) \\ &\leq C_{J^\dagger} |x^*_{i^\bullet+l+1} - x^\dagger_{i^\bullet+l+1}| \leq C_{J^\dagger}(C_\mu + C_\kappa)\Delta l \end{aligned}$$

where $C_{J^\dagger}$ is the Lipshitz constant of $J^\dagger$.

Grouping together the three estimates 1)–3) leads to the conclusion of the theorem. ■

D. Error Analysis of Discretization Schemes

1) First-Order Interpolation Scheme: In the main text, the collocation discretization scheme was presented to discretize the constraints over $N+1$ grid points. We now introduce another scheme: first-order interpolation scheme.

In this scheme, at stage i, we require (u_i, x_i) and $(u_i, x_i + 2\Delta_i u_i)$ to satisfy the constraints at $s = s_i$ and $s = s_{i+1}$, respectively. That is, for $i = 0, \ldots, N-1$

$$\begin{bmatrix} \mathbf{a}(s_i) \\ \mathbf{a}(s_{i+1}) + 2\Delta \mathbf{b}(s_{i+1}) \end{bmatrix} u + \begin{bmatrix} \mathbf{b}(s_i) \\ \mathbf{b}(s_{i+1}) \end{bmatrix} x + \begin{bmatrix} \mathbf{c}(s_i) \\ \mathbf{c}(s_{i+1}) \end{bmatrix} \in \begin{bmatrix} \mathscr{C}(s_i) \\ \mathscr{C}(s_{i+1}) \end{bmatrix}. \tag{22}$$

At $i = N$, one uses only the top half of the aforementioned equations. By appropriately rearranging the terms, the aforementioned equations can, finally, be rewritten as

$$\mathbf{a}_i u + \mathbf{b}_i x + \mathbf{c}_i \in \mathscr{C}_i. \tag{23}$$

2) Error Analysis: For simplicity, suppose Assumption 1 holds. That is, there exists $\tilde{\mathbf{a}}(s)_{s\in[0,1]}, \tilde{\mathbf{b}}(s)_{s\in[0,1]}, \tilde{\mathbf{c}}(s)_{s\in[0,1]}$, which define the set of admissible control-state pairs. Consider the interval $[s_0, s_1]$, the parameterization is given by

$$x(s; u_0, x_0) = x_0 + 2su_0$$

where x_0 is the state at s_0 and u_0 is the constant control along the interval. Note that $s_0 = 0, s_1 = \Delta$.

Define the constraint satisfaction *function* by

$$\boldsymbol{\epsilon}(s) := u_0 \tilde{\mathbf{a}}(s) + x(s)\tilde{\mathbf{b}}(s) + \tilde{\mathbf{c}}(s). \tag{24}$$

The *greatest* constraint satisfaction error over $[s_0, s_1]$ is

$$\max \left\{ \max_{k, s\in[s_0,s_1]} \boldsymbol{\epsilon}(s)[k], 0 \right\}.$$

Different discretization schemes enforce different conditions on $\boldsymbol{\epsilon}(s)$. In particular, one has for

1) *collocation scheme*: $\boldsymbol{\epsilon}(s_0) \leq 0$;
2) *first-order interpolation scheme*: $\boldsymbol{\epsilon}(s_0) \leq 0, \boldsymbol{\epsilon}(s_1) \leq 0$.

Using the classic result on error of polynomial interpolation [29, Th. 2.1.4.1], one obtains the following estimate of $\boldsymbol{\epsilon}(s)$ for the *collocation scheme*:

$$\boldsymbol{\epsilon}(s) = \boldsymbol{\epsilon}(s_0) + (s - s_0)\boldsymbol{\epsilon}'(\xi) = s\boldsymbol{\epsilon}'(\xi)$$

for $\xi \in [s_0, s]$. Suppose the derivatives of $\tilde{\mathbf{a}}(s), \tilde{\mathbf{b}}(s),$ and $\tilde{\mathbf{c}}(s)$ are bounded, one has

$$\max_{s\in[s_0,s_1]} \boldsymbol{\epsilon}(s) = O(\Delta).$$

Thus, the greatest constraint satisfaction error of the collocation discretization scheme has order $O(\Delta)$.

Using the same theorem, one obtains the following estimation of $\boldsymbol{\epsilon}(s)$ for the *first-order interpolation scheme*:

$$\begin{aligned} \boldsymbol{\epsilon}(s) &= \boldsymbol{\epsilon}(s_0) + (s - s_0)\frac{\boldsymbol{\epsilon}(s_1) - \boldsymbol{\epsilon}(s_0)}{s_1 - s_0} + \frac{(s - s_0)(s - s_1)\boldsymbol{\epsilon}''(\xi)}{2!} \\ &= \frac{s(s - \Delta)\boldsymbol{\epsilon}''(\xi)}{2!} \end{aligned}$$

for some $\xi \in [s_0, s_1]$. Again, since the derivatives of $\tilde{\mathbf{a}}(s), \tilde{\mathbf{b}}(s)$, and $\tilde{\mathbf{c}}(s)$ are assumed to be bounded, one has

$$\max_{s\in[s_0,s_1]} \boldsymbol{\epsilon}(s) = O(\Delta^2).$$

Thus, the greatest constraint satisfaction error of the first-order interpolation discretization scheme has order $O(\Delta^2)$.

REFERENCES

[1] J. E. Bobrow, S. Dubowsky, and J. S. Gibson, "Time-optimal control of robotic manipulators along specified paths," *Int. J. Robot. Res.*, vol. 4, no. 3, pp. 3–17, 1985.

[2] Z. Shiller and S. Dubowsky, "On computing the global time-optimal motions of robotic manipulators in the presence of obstacles," *IEEE Trans. Robot. Autom.*, vol. 7, no. 6, pp. 785–797, Dec. 1991.

[3] Q.-C. Pham, S. Caron, P. Lertkultanon, and Y. Nakamura, "Admissible velocity propagation: Beyond quasi-static path planning for high-dimensional robots," *Int. J. Robot. Res.*, vol. 36, no. 1, pp. 44–67, 2017.

[4] Q.-C. Pham, "A general, fast, and robust implementation of the time-optimal path parameterization algorithm," *IEEE Trans. Robot.*, vol. 30, no. 6, pp. 1533–1540, Dec. 2014.

[5] F. Pfeiffer and R. Johanni, "A concept for manipulator trajectory planning," *IEEE J. Robot. Autom.*, vol. 3, no. 2, pp. 115–123, Apr. 1987.

[6] J.-J. E. Slotine and H. S. Yang, "Improving the efficiency of time-optimal path-following algorithms," *IEEE Trans. Robot. Autom.*, vol. 5, no. 1, pp. 118–124, Feb. 1989.

[7] Z. Shiller and H.-H. Lu, "Computation of path constrained time optimal motions with dynamic singularities," *J. Dyn. Syst., Meas., Control*, vol. 114, no. 1, pp. 34–40, 1992.

[8] L. Zlajpah, "On time optimal path control of manipulators with bounded joint velocities and torques," in *Proc. IEEE Int. Conf. Robot. Autom.*, 1996, vol. 2, pp. 1572–1577.

[9] D. Verscheure, B. Demeulenaere, J. Swevers, J. De Schutter, and M. Diehl, "Time-optimal path tracking for robots: A convex optimization approach," *IEEE Trans. Autom. Control*, vol. 54, no. 10, pp. 2318–2327, Oct. 2009, doi: 10.1109/TAC.2009.2028959.

[10] K. Hauser, "Fast interpolation and time-optimization with contact," *Int. J. Robot. Res.*, vol. 33, no. 9, pp. 1231–1250, Aug. 2014.
[11] Q.-C. Pham and O. Stasse, "Time-Optimal path parameterization for redundantly actuated robots: A numerical integration approach," *IEEE/ASME Trans. Mechatronics*, vol. 20, no. 6, pp. 3257–3263, Dec. 2015.
[12] S. Caron, Q.-C. Pham, and Y. Nakamura, "Leveraging cone double description for multi-contact stability of humanoids with applications to statics and dynamics," in *Proc. Robot., Sci. Syst.*, 2015. [Online]. Available: http://www.roboticsproceedings.org/rss11/p28.html
[13] S. Caron, Q.-C. Pham, and Y. Nakamura, "ZMP support areas for multicontact mobility under frictional constraints," *IEEE Trans. Robot.*, vol. 33, no. 1, pp. 67–80, Feb. 2017.
[14] Y. Nakamura and H. Hanafusa, "Inverse kinematic solutions with singularity robustness for robot manipulator control," *J. Dyn. Syst., Meas., Control*, vol. 108, no. 3, pp. 163–171, 1986.
[15] E. C. Kerrigan, "Robust constraint satisfaction: Invariant sets and predictive control," Ph.D. dissertation, Univ. Cambridge, Cambridge, U.K., 2001.
[16] D. Bertsekas and I. Rhodes, "On the minimax reachability of target sets and target tubes," *Automatica*, vol. 7, no. 2, pp. 233–247, Mar. 1971.
[17] S. V. Rakovic, E. C. Kerrigan, D. Q. Mayne, and J. Lygeros, "Reachability analysis of discrete-time systems with disturbances," *IEEE Trans. Automat. Control*, vol. 51, no. 4, pp. 546–561, Apr. 2006.
[18] T. Bretl and S. Lall, "Testing static equilibrium for legged robots," *IEEE Trans. Robot.*, vol. 24, no. 4, pp. 794–807, Aug. 2008.
[19] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge Univ. Press, 2004.
[20] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Math. Program. Comput.*, vol. 6, no. 4, pp. 327–363, Dec. 2014.
[21] K. Fukuda and A. Prodon, "Double description method revisited," in *Combinatorics and Computer Science*. Berlin, Germany: Springer, 1996, pp. 91–111.
[22] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2000, vol. 2, pp. 995–1001.
[23] P. Lertkultanon and Q.-C. Pham, "Dynamic non-prehensile object transportation," in *Proc. 13th Int. Conf. Control Autom. Robot. Vis.*, 2014, pp. 1392–1397.
[24] A. Ben-Tal and A. Nemirovski, *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*. Philadelphia, PA, USA: SIAM, 2001.
[25] M. Tarkiainen and Z. Shiller, "Time optimal motions of manipulators with actuator dynamics," in *Proc. 1993 IEEE Int. Conf. Robot. Autom.*, 1993, pp. 725–730.
[26] H. Pham and Q.-C. Pham, "On the structure of the time-optimal path parameterization problem with third-order constraints," in *Proc. 2017 IEEE Int. Conf. Robot. Autom.*, 2017, pp. 679–686.
[27] D. Costantinescu and E. A. Croft, "Smooth and time-optimal trajectory planning for industrial manipulators along specified paths," *J. Robot. Syst.*, vol. 17, no. 5, pp. 233–249, 2000.
[28] M. Oberherber, H. Gattringer, and A. Müller, "Successive dynamic programming and subsequent spline optimization for smooth time optimal robot path tracking," *Mech. Sci.*, vol. 6, no. 2, pp. 245–254, 2015.
[29] J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis*. New York, NY, USA: Springer, 1982, p. 96.

Hung Pham received the Bachelor degree in mechanical engineering from Nanyang Technological University, Singapore, in 2015, where he is currently working toward the Ph.D. degree in robotics.

His current research interests include robotic motion planning and its applications to fields such as air traffic management and 3-D printing.

Quang-Cuong Pham was born in Hanoi, Vietnam. He received the graduate degree from École Normale Supérieure, Paris, France, in 2007 and the Ph.D. degree in neuroscience from Universit Paris 6 and Collège de France, Paris, France, in 2009.

In 2010, he was a Visiting Researcher with University of São Paulo, São Paulo, Brazil. From 2011 to 2013, he was a Fellow with Japan Society for the Promotion of Science, studying Robotics with University of Tokyo. He joined Nanyang Technological University, Singapore, as an Assistant Professor, in 2013.