Invariant Configuration-Space Bubbles for Revolute Serial-Chain Robots

Claus Danielson, *Senior Member, IEEE*

***Abstract*—This letter adapts the invariant-set motion planner (ISMP) for robot motion planning. We derive control invariant subsets of configuration-space bubbles lifted into the state-space. The resulting sets guarantee collision avoidance since they are both constraint admissible and control invariant. We present a command governor that enforces the positive invariance of the constraints in closed-loop. This governor can be used to transform any nominal tracking controller into a constraint enforcing controller. We use these control invariant sets to quantify a relationship between velocity and control authority that enables collision avoidance. We demonstrate our invariant-sets through an illustrative numerical example.**

***Index Terms*—Autonomous robots, robot control, optimization, constrained control, reachability analysis.**

I. Introduction

THE INVARIANT-SET motion-planner (ISMP) is an algorithm for generating dynamically feasible trajectories from an initial state to a target equilibrium through an obstacle-filled environment [1], [2], [3], [4], [5], [6], [7], [8], [9]. The invariant-set motion-planner (ISMP) has several advantageous properties. It allows for aggressive, but safe maneuvers since, by definition, the system state will never leave the constraint admissible positive invariant (CAPI) sets. It is quantifiably robust since it incorporates feedback into the design and the CAPI sets provide a natural buffer that can absorb tracking errors due to model uncertainty and disturbances [9]. It reduces the curse-of-dimensionality by sampling from the output-space instead of the state-space. Furthermore, it does not require dense sampling since the CAPI sets can cover large volumes of the state/output-space. It typically has low online computational costs since the CAPI sets can be pre-computed as they only depend on the time-invariant closed-loop dynamics, rather than the time varying environment. The ISMP has been successfully applied to spacecraft [1], [2], [3], [4] and autonomous driving [5], [6], [7], [8]. This letter applies the ISMP for robot motion planning.

Manuscript received 29 August 2022; revised 31 October 2022; accepted 18 November 2022. Date of publication 24 November 2022; date of current version 2 December 2022. This work was supported in part by the National Science Foundation (NSF) under Grant CMMI-2105631, and in part by the Air Force Office of Scientific Research under Award FA9550-22-1-0093. Recommended by Senior Editor L. Zhang.

The author is with the Department of Mechanical Engineering, University of New Mexico, Albuquerque, NM 87131 USA (e-mail: cdanielson@unm.edu).

Digital Object Identifier 10.1109/LCSYS.2022.3224685

The defining feature of the ISMP is that knowledge of the closed-loop system dynamics is incorporated into the motion plan using CAPI sets (also called viable sets [10]). CAPI sets have two important properties; they are constraint admissible meaning that they do not collide with any obstacles (or more generally they satisfy constraints) and they are positive invariant meaning that the closed-loop dynamics will remain inside the set. Thus, these CAPI sets describe regions of the state-space where the system can safely track the corresponding references without collision. The ISMP uses random-sampling [11] and graph-searches to find a corridor of CAPI sets that safely guides the system through the obstacle filled environment to the target equilibrium.

For space and automotive applications, the positive invariant (PI) sets were known from the literature. Thus, adapting the ISMP for these applications required developing methods for certifying that these PI sets are constraint admissible. For instance, autonomous driving required certifying that a PI set for a vehicle does not intersect obstacles (e.g., other vehicles). In contrast, for robotics constraint admissible sets are known in the form of configuration-space bubbles [12], [13], [14], [15] which describe a neighborhood of configurations for which the robot does not intersect an obstacle. This letter addresses the problem of rendering these constraint admissible sets invariant.

Invariant sets are important in motion planning since they describe sets of states for which constraint enforcement is possible. Many motion planning algorithms tacitly incorporate invariant sets. For instance, kino-dynamic [11] motion planners use dynamic trajectories, which are 1-dimensional invariant sets. LQR-trees [16] expand an invariant tube around these linear invariant sets, however, this post-hoc synthesis can be computationally expensive. Invariance is also implicitly incorporated into motion planning by forcing the robot to move slowly and cautiously. This heuristically provides invariance by ensuring that the limited control authority of the joint actuators can overcome the momentum of the robot to prevent collisions. This letter quantifies this heuristic relationship between velocity and control authority using invariance.

The main contribution of this letter is the derivation of control invariant (CI) subsets of configuration-space bubbles. We lift the bubbles from the configuration-space into the state-space and describe a subset of joint positions and velocities for which it is possible to remain inside the bubble and thus avoid collisions. Another contribution of this letter is the development of a command governor (CG) for which the CI subset becomes PI, i.e., the closed-loop dynamics will not leave the configuration-space bubble. One advantage of this approach is that the CG can transform any nominal controller into a control that renders a configuration-space bubble PI. This means that an existing robot controller does not need to be redesigned to apply the ISMP. Our final contribution is using these CI sets to quantify a relationship between velocity and control authority that guarantees the joint-motors can stop the robot before its momentum carries it out of the configuration-space bubble.

The remainder of this letter is organized as follows. In Section II, we describe the robot motion planning problem and the ISMP algorithm. In Section III, we describe configuration-space bubbles and their invariant subsets. We present a formal proof that configuration-space bubbles are output-admissible. We then derive a CI subset and a CG that renders this set PI. Finally, we present an illustrative example that demonstrates the advantages of our proposed approach.

Notation and Definitions: $\mathbb{R}$ are the real numbers and $\mathbb{S} = \mathbb{R}/2\pi\mathbb{Z} \cong [0, 2\pi)$ is the circle of radians. The p-norm of a vector $x \in \mathbb{R}^n$ is $\|x\|_p = (\sum_{i=1}^n |x_i|^p)^{1/p}$. A p-norm ball $\mathbb{B}_p$ is the set $\mathbb{B}_p = \{x : \|x\|_p \leq 1\}$. The distance between sets is $d(\mathcal{X}, \mathcal{Y}) = \inf_{x\in\mathcal{X}, y\in\mathcal{Y}} \|x - y\|$ where $\|\cdot\|$ is the 2-norm. The convex-hull of a finite-set $\{x_1, \dots, x_m\}$ is the set $\{\sum_{i=1}^m \lambda_i x_i : \sum_{i=1}^m \lambda_i = 1, \lambda_i \geq 0\}$. The closure and interior of a set $\mathcal{X}$ are denoted by $\mathrm{cl}(\mathcal{X})$ and $\mathrm{int}(\mathcal{X})$, respectively. The pseudo-cross product matrix $k^\times \in \mathbb{R}^{3\times3}$ satisfies $k^\times v = k \times v$. A set $\mathcal{O}$ is positive invariant if $x(t_0) \in \mathcal{O} \Rightarrow x(t) \in \mathcal{O}\ \forall t > t_0$. A set $\mathcal{O}$ is control invariant if for all $x(t_0) \in \mathcal{O}$ there exists $u(t) \in \mathcal{U}$ such that $x(t) \in \mathcal{O}$ for all $t \geq t_0$, i.e., there exists a controller for which the set is PI. A directed graph $\mathfrak{G} = (\mathfrak{I}, \mathfrak{E})$ is a set of nodes $\mathfrak{I}$ together with a set of ordered pairs $\mathfrak{E} \subseteq \mathfrak{I} \times \mathfrak{I}$ called edges. Nodes $i, j \in \mathfrak{I}$ are called adjacent if $(i, j) \in \mathfrak{E}$ is an edge. A path is a sequence of adjacent vertices. A graph search is an algorithm for finding a path through a graph. A graph $\mathfrak{T}$ is a tree if every pair of nodes $(i, j) \in \mathfrak{I}$ is connected by exactly one path. $p_{ij}(\theta) \in \mathbb{R}^3$ denotes the position of reference-frame j with respect to frame i in configuration $\theta \in \mathbb{S}^n$.

II. Robot Motion Planning Problem

A. Robot Dynamics

We consider an n-link revolute serial-chain robot whose dynamics are modeled by the Lagrange-Euler equations

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} = \tau \tag{1}$$

where $\tau(t) \in \mathbb{R}^n$ are the torques, $\theta(t) \in \mathbb{S}^n$ are positions, and $\dot{\theta}(t), \ddot{\theta}(t) \in \mathbb{R}^n$ are, respectively, the velocity and acceleration of the n revolute joints. The configuration-dependent mass-inertia and coriolis-centripetal matrices are denoted by $M(\theta)$ and $C(\theta, \dot{\theta})$, respectively. The nonlinearity of the robot dynamics (1) is one of the reasons the robot motion planning problem is challenging.

B. Input and Output Constraints

The objective of the robot motion-planning problem is to move the robot from an initial state $(\theta(0), \dot{\theta}(0))$ to a desired configuration $\bar{\theta}_\infty$ without colliding with any obstacles in the environment. The spatial extents of the obstacles are described by the collection of sets $\{\mathcal{B}_k\}_{k=1}^m$ where $\mathcal{B} = \bigcup_{k=1}^m \mathcal{B}_k \subset \mathbb{R}^3$.

Algorithm 1 Invariant-Set Motion-Planner

```
1: Input: Search-tree $\mathfrak{T} = (\mathfrak{I}, \mathfrak{E})$
2: Find path $\{\sigma_0, \dots, \sigma_N\}$ from $\bar{\theta}_{\sigma_0} = \theta(0)$ to $\bar{\theta}_{\sigma_N} = \theta_\infty$
3: repeat
4:     if $(\theta(t), \dot{\theta}(t)) \in \mathcal{O}_{\sigma_{k+1}}$ then
5:         $k \leftarrow k + 1$
6:     end if
7:     Use controller $\tau(t) = \kappa_{\sigma_k}\big(\theta(t), \dot{\theta}(t)\big)$
8: until $\theta(t) = \bar{\theta}_\infty$
```

Likewise, the configuration-dependent spatial extent of the robot is described by the set

$$\mathcal{R}(\theta) = \big\{p_{0s}(\theta) : s \in \mathcal{R}\big\} \tag{2}$$

where $\mathcal{R} \subseteq \mathbb{R}^3$ indexes all points s on the robot in the home-configuration and $p_{0s}(\theta)$ is the forward-kinematics of the point $s \in \mathcal{R}$ relative to the base-frame 0 in configuration θ. To avoid collisions, the distance $d(\mathcal{R}(\theta), \mathcal{B})$ between the obstacles $\mathcal{B}$ and robot $\mathcal{R}(\theta)$ should be positive $d(\mathcal{R}(\theta), \mathcal{B}) > 0$. Thus, we have the following set of constraints on the admissible configurations θ for the robot

$$\Theta = \mathrm{cl}\Big(\Big\{\theta \in \mathbb{S}^n : d\big(\mathcal{R}(\theta), \mathcal{B}\big) > 0\Big\}\Big). \tag{3}$$

Here we take the closure of the admissible set since configurations where the robot barely touches the obstacle are not considered collisions. This is consistent with the definition of configuration-space bubbles [12], [13], [14], [15] and will allow the derivation of closed invariant sets. The non-convexity of the output constraint set (3) is another reason the robot motion planning problem is challenging. Our approach will exploit existing algorithms and software for collision-detection, i.e., computing the distance $d(\mathcal{R}(\theta), \mathcal{B})$.

The joint-motor torques $\tau(t) \in \mathbb{R}^n$ are subject to polytopic input constraints $\tau \in \mathcal{T}$ where

$$\mathcal{T} = \big\{\tau \in \mathbb{R}^n : h_j\tau \leq k_j \text{ for } j = 1, \dots, N_\tau\big\} \tag{4}$$

where N_τ is the number of torque constraints. For instance, bounds $\underline{\tau} \leq \tau(t) \leq \bar{\tau}$ will produce a box $\mathcal{T}$ with $N_\tau = 2n$ constraints. We assume that the set $\mathcal{T} \subseteq \mathbb{R}^n$ contain the origin in its interior. This assumption means that each of the joint-motors can produce both positive and negative torques.

C. Robot Motion Planning Problem and Algorithm

The robot motion planning problem is stated below.

Problem 1: Find a feasible torque trajectory $\tau(t) \in \mathcal{T}$ such that robot converges $\theta(t) \to \bar{\theta}_\infty$ to the target configuration $\bar{\theta}_\infty$ while remaining in the safe region $\theta(t) \in \Theta$.

The robot motion planning problem can be solved using the ISMP described by Algorithm 1. The ISMP searches an appropriately constructed search-tree $\mathfrak{T}$ for a sequence $\{\bar{\theta}_i\}_{i=1}^N$ of intermediate references $\bar{\theta}_i \in \mathbb{S}^n$ that guide the robot (1) state $x(t) = (\theta(t), \dot{\theta}(t))$ from an initial state $x(0) = x_0$ to a target equilibrium configuration $\bar{\theta}_\infty \in \mathbb{S}^n$ while avoiding obstacles $\theta(t) \in \Theta$. Associated with each node $i \in \mathfrak{I}$ is a controller κ_i which drives the robot to the reference configuration $\bar{\theta}_i$. In addition, each node has an associated CAPI set $\mathcal{O}_i$, which is output admissible $[I, 0]\mathcal{O}_i \subseteq \Theta$, input admissible $\kappa_i(\theta, \dot{\theta}) \in \mathcal{T}$ for all $(\theta, \dot{\theta}) \in \mathcal{O}_i$, and positive invariant $x(t) \in \mathcal{O}_i$ under

Algorithm 2 Search Tree $\mathfrak{T}$ Construction

```
1:  Input: State $(\theta(0), \omega(0))$, target $\theta_\infty$, obstacles $\mathcal{B}$
2:  Output: Search Tree $\mathfrak{T} = (\mathfrak{I}, \mathfrak{E})$
3:  Construct CAPI set $\mathcal{O}_\infty$ and controller $\kappa_\infty(\theta, \dot{\theta})$
4:  Initialize tree $\mathfrak{T}$.add-node $= (\bar{\theta}_\infty, \mathcal{O}_\infty, \kappa_\infty)$
5:  repeat
6:      Sample random configuration $\bar{\theta}_i \in \mathrm{int}(\Theta)$
7:      Connect $\bar{\theta}_i \in \mathrm{int}([I, 0]\mathcal{O}_j)$ to closest configuration $\bar{\theta}_j$
8:      Construct CAPI set $\mathcal{O}_i$ and controller $\kappa_i(\theta, \dot{\theta})$
9:      Update search-tree
            $\mathfrak{T}$.add-node $= \{\bar{\theta}_i, \kappa_i, \mathcal{O}_i\}$
            $\mathfrak{T}$.add-edge $= (i, j)$
10: until $(\theta(0), \omega(0)) \in \mathcal{O}_i$
```

the controller κ_i. The edges $(i, j) \in \mathfrak{E}$ of the tree $\mathfrak{T} = (\mathfrak{I}, \mathfrak{E})$ indicate that the state (1) will enter the j-th CAPI set $\mathcal{O}_j$ while tracking the i-th reference $\bar{\theta}_i$ without leaving the current CAPI set $\mathcal{O}_i$. Thus, the ISMP avoids obstacles by moving the robot state through a sequence $\{\sigma_i\}_{i=0}^N$ of CAPI sets $\mathcal{O}_{\sigma_i}$.

The construction of the search-tree is described by Algorithm 2. Since our proposed approach will render configuration-space bubble PI, the search-tree construction Algorithm 2 for the rapid-ISMP [2] is similar to the configuration-space planner [14]. See those papers for details. We will describe the synthesis of the CI sets $\mathcal{O}_i$ and controller κ_i in the subsequent section.

III. Constraint Admissible Invariant Sets

The distinguishing feature of the ISMP is the use of CAPI sets to encode the interactions between the closed-loop dynamics and constraints into the motion plan. In this section, we describe CAPI sets for robot motion planning.

A. Output Admissibility

In this section, we review the concept of configuration-space bubbles [12], [13], [14], [15]. Configuration-space bubbles provide a computationally efficient method for describing a subset of the configuration-space for which the output constraints (3) are satisfied (including self-collsions [15]). In the terminology of the ISMP, configuration-space bubbles are output-admissible sets. Our main contribution is the derivation of invariant subsets of these configuration-space bubbles in the next subsection.

For a reference configuration $\bar{\theta} \in \mathbb{S}^n$, the configuration-space bubble is given by

$$\mathcal{C}(\bar{\theta}) = \left\{\theta : \left\|P(\bar{\theta})\big(\theta - \bar{\theta}\big)\right\|_1 \leq 1\right\} \tag{5}$$

where the diagonal matrix $P(\bar{\theta}) = \mathrm{diag}(\rho_1(\bar{\theta}), \ldots, \rho_n(\bar{\theta}))$ has entries

$$\rho_i(\bar{\theta}) = \max_{s \in \mathcal{R}} \frac{\|k_i \times p_{is}(\bar{\theta})\|_2}{d\big(p_{0s}(\bar{\theta}), \mathcal{B}\big)} \tag{6}$$

where $k_i \in \mathbb{R}^3$ is the i-th axis of rotation and $p_{is}(\bar{\theta}) \in \mathbb{R}^3$ is the forward-kinematics to a point $s \in \mathcal{R}$ on the robot in configuration $\bar{\theta}$. Here we use the more recent and flexible definition [14] for configuration-space bubbles. Note that the configuration-space bubbles (5) are not defined for $\bar{\theta} \notin \mathrm{int}(\Theta)$ since this would cause a division by zero in (6).

Although configuration-space bubbles (5) have a long history in robotics [15], we have not found a rigorous proof of their output admissibility. Therefore, we provide the following proposition as a foundation for our results.

Proposition 1 (Output Admissibility): Let $\bar{\theta} \in \mathrm{int}(\Theta)$. Then, the robot does not collide with an obstacle for all configurations $\theta \in \mathcal{C}(\bar{\theta})$ in the configuration-space bubble (5), i.e., $\mathcal{C}(\bar{\theta}) \subseteq \Theta$.

Proof: For a point $s \in \mathcal{R}$ on the robot, the distance to the obstacles $\mathcal{B}$ is defined as $d(p_{0s}(\bar{\theta}), \mathcal{B}) = \min_{y \in \mathcal{B}} \|p_{0s}(\bar{\theta}) - y\|$ where all norms in this proof are 2-norms unless otherwise stated. By the triangle inequality, we have

$$\begin{aligned} d(p_{0s}(\bar{\theta}), \mathcal{B}) &\leq \|p_{0s}(\bar{\theta}) - p_{0s}(\theta)\| + \min_{y \in \mathcal{B}} \|p_{0s}(\theta) - y\| \\ &= \|p_{0s}(\bar{\theta}) - p_{0s}(\theta)\| + d(p_{0s}(\theta), \mathcal{B}) \end{aligned} \tag{7}$$

Using the forward-kinematics $p_{is} = p_{ij} + R(\theta_j)p_{js}$ where p_{is} is the position of s relative to the base-frame $i = 0$ and p_{js} is its position relative to joint-frame $j = 1$, we can bound the first term, $\|p_{is}(\theta) - p_{is}(\bar{\theta})\| = \|R(\theta_j)p_{js}(\theta) - R(\bar{\theta}_j)p_{js}(\bar{\theta})\|$ since p_{ij} does not depend on the configuration. By the triangle inequality, we have $\|p_{is}(\theta) - p_{is}(\bar{\theta})\| \leq \|R(\theta_j)p_{js}(\theta) - R(\theta_j)p_{js}(\bar{\theta})\| + \|R(\theta_j)p_{js}(\bar{\theta}) - R(\bar{\theta}_j)p_{js}(\bar{\theta})\|$. Since the 2-norm is invariant under rotations, we can simplify

$$\begin{aligned} \|p_{is}(\theta) - p_{is}(\bar{\theta})\| \leq \|p_{js}(\theta) - p_{js}(\bar{\theta})\| &+ \\ + \|p_{js}(\bar{\theta}) - R(\tilde{\theta})p_{js}(\bar{\theta})\|&. \end{aligned} \tag{8}$$

where $\tilde{\theta}_j = \bar{\theta}_j - \theta_j$ and $R(\theta_j)^{-1}R(\bar{\theta}_j) = R(\bar{\theta}_j - \theta_j) = R(\tilde{\theta}_j)$ since the rotations about the same axis k_j. Applying the Rodriguez formula $R(\tilde{\theta}_j) = I + \sin(\tilde{\theta}_j)k_j^\times + (1 - \cos(\tilde{\theta}_j))k_j^\times k_j^\times$ to the square of the second-term yields $\|p_{js}(\bar{\theta}) - R(\tilde{\theta}_j)p_{js}(\bar{\theta})\|^2 = s_{\tilde{\theta}}^2\|k_j \times p_{js}(\bar{\theta})\|^2 + (1 - c_{\tilde{\theta}})^2\|k_j \times k_j \times p_{js}(\bar{\theta})\|^2$ where the cross-term disappears since $k_j \times p_{js} \perp k_j \times k_j \times p_{js}$. Since $k_j \perp k_j \times p_{js}$, we have $\|k_j \times k_j \times p_{js}(\bar{\theta})\| = \|k_j \times p_{js}(\bar{\theta})\|\|k_j\|$ where $\|k_j\| = 1$ since k_j is a unit-vector. Therefore, $\|p_{js}(\bar{\theta}) - R(\tilde{\theta}_j)p_{js}(\bar{\theta})\|^2 = (2 - 2\cos\tilde{\theta}_j)\|k_j \times p_{js}(\bar{\theta})\|^2$. Substituting into (8), yields $\|p_{is}(\theta) - p_{is}(\bar{\theta})\| \leq \|p_{js}(\theta) - p_{js}(\bar{\theta})\| + \|k_j \times p_{js}(\bar{\theta})\||\tilde{\theta}_j|$ where $2 - 2\cos\tilde{\theta}_j \leq \tilde{\theta}_j^2$. Following the same arguments, this inequality applies for frames $i = 1, \ldots, n-1$ and $j = i+1$. Thus, we can obtain

$$\|p_{0s}(\theta) - p_{0s}(\bar{\theta})\| \leq \sum\nolimits_{j=1}^{n} \|k_j \times p_{js}(\bar{\theta})\||\tilde{\theta}_j|$$

where $p_{js}(\theta) = p_{js}(\bar{\theta})$ when s is fixed in the j-th reference-frame. Plugging into (7) and rearranging terms, we obtain

$$d(p_{0s}(\theta), \mathcal{B}) \geq d(p_{0s}(\bar{\theta}), \mathcal{B}) - \sum\nolimits_{j=1}^{n} \|k_j \times p_{js}(\bar{\theta})\||\tilde{\theta}_j|$$

Using (6), the sum simplifies to the weighted 1-norm in (5)

$$d(p_{0s}(\theta), \mathcal{B}) \geq d(p_{0s}(\bar{\theta}), \mathcal{B})\big(1 - \|P\tilde{\theta}\|_1\big).$$

For any configurations $\theta \in \mathrm{int}(\mathcal{C}(\bar{\theta}))$ in the interior configuration-space bubble (5), we have $\|P\tilde{\theta}\|_1 < 1$. Hence $d(p_{0s}(\theta), \mathcal{B}) > 0$, $\forall s \in \mathcal{R}$ where $d(p_{0s}(\bar{\theta}), \mathcal{B}) > 0$ since $\bar{\theta} \in \mathrm{int}(\Theta)$. Thus, by the definition (2) of $\mathcal{R}(\theta)$, we have $d(\mathcal{R}(\theta), \mathcal{B}) > 0$ for all $\theta \in \mathrm{int}(\mathcal{C}(\bar{\theta}))$, i.e., $\mathrm{int}(\mathcal{C}(\bar{\theta})) \subseteq \mathrm{int}(\Theta)$. Therefore, $\mathcal{C}(\bar{\theta}) = \mathrm{cl}(\mathrm{int}(\mathcal{C}(\bar{\theta}))) \subseteq \mathrm{cl}(\mathrm{int}(\Theta)) = \Theta$. ■

B. Control and Positive Invariance

In this section, we present our main contribution; a PI subset $\mathcal{O}$ of the configuration-space bubble (5). In the next section, we will discuss how to scale these PI sets to ensure that they are input admissible. The following theorem describes a CI subset of the configuration-space bubble (5).

Theorem 1 (Control Invariance): Consider the set

$$\mathcal{O} = \text{conv}\left\{ \begin{bmatrix} \bar{\theta} \pm e_i/\rho_i \\ 0 \end{bmatrix}, \begin{bmatrix} \bar{\theta} \pm e_i/\rho_i \\ \mp \nu e_i \end{bmatrix} : i = 1, \dots, n \right\} \quad (9)$$

where e_i are the standard basis-vectors and ρ_i were defined in (6). If the tuning-parameter $\nu > 0$ satisfies

$$\nu^2 \mathbb{B}_1 \subseteq P(\bar{\theta}) M(\theta)^{-1} \big(\mathcal{T} - C(\theta, \dot{\theta})\dot{\theta}\big), \ \forall (\theta, \dot{\theta}) \in \mathcal{O} \quad (10)$$

then (9) is a CI subset $[I, 0]\mathcal{O} \subseteq \mathcal{C}(\bar{\theta})$ of the configuration-space bubble (5) under the robot dynamics (1).

Proof: First, we show $[I, 0]\mathcal{O} \subseteq \mathcal{C}(\bar{\theta})$. By definition (9), any $\theta \in [I, 0]\mathcal{O}$ can be expressed as the convex combination

$$\theta = \bar{\theta} + P^{-1}\Big(\sum\nolimits_{i=1}^{n} \lambda_i^+ e_i - \lambda_i^- e_i\Big)$$

where $\lambda_i^+, \lambda_i^- \in [0, 1]$ and $\sum_{i=1}^n \lambda_i^+ + \lambda_i^- = 1$. Thus, $\|P(\theta - \bar{\theta})\|_1 = \|\sum_{i=1}^n \lambda_i^+ e_i - \lambda_i^- e_i\|_1 = \sum_{i=1}^n |\lambda_i^+ - \lambda_i^-| \le \sum_{i=1}^n \lambda_i^+ + \lambda_i^- = 1$. Therefore, $\theta \in \mathcal{C}(\bar{\theta})$ by definition of the bubble (5).

Next, we show that $\mathcal{O}$ is CI by showing the existence of a feasible control input $\tau \in \mathcal{T}$ that keeps the state $(\theta, \dot{\theta}) \in \mathcal{O}$ inside $\mathcal{O}$. Consider the nonlinear change-of-variables $\varepsilon = P(\theta - \bar{\theta})$ and feedback-linearizing controller $\tau = M(\theta)Pu + C(\theta, \dot{\theta})\dot{\theta}$. Under this change-of-variables, the dynamics (1) become a linear double-integrator

$$\frac{d}{dt}\begin{bmatrix} \varepsilon \\ \dot{\varepsilon} \end{bmatrix} = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \varepsilon \\ \dot{\varepsilon} \end{bmatrix} + \begin{bmatrix} 0 \\ I \end{bmatrix} u. \quad (11)$$

Furthermore, the candidate CI set (9) is transformed into

$$\hat{\mathcal{O}} = \text{conv}\left\{ \begin{bmatrix} \pm e_i \\ 0 \end{bmatrix}, \begin{bmatrix} \pm e_i \\ \mp \nu e_i \end{bmatrix} : i = 1, \dots, n \right\}$$

By condition (10), we can conservative under-approximate the constraints $\mathcal{U} \subseteq \mathbb{R}^n$ on the fictitious input $u \in \mathbb{R}^n$ as a 1-norm ball $\mathcal{U} = \nu^2 \mathbb{B}_1$. Since the transformed dynamics (11) are linear and the transformed constraints $\hat{\mathcal{O}}$ and $\hat{\mathcal{U}}$ are polytopic, we can prove that $\hat{\mathcal{O}}$ is CI by showing that for each of its vertices there exists a feasible control input $u \in \mathcal{U}$ such that the resulting vector-field $\dot{x}$ points into the set $\hat{\mathcal{O}}$ [17].

Consider the vertex $x_i = (e_i, 0) \in \hat{\mathcal{O}}$ and the control input $u_i = -\nu^2 e_i \in \mathcal{U}$. Then, $x_i + \frac{1}{\nu}\dot{x}_i = (e_i, -\nu e_i) \in \hat{\mathcal{O}}$ By the Minkowski-Weyl theorem, the polytope $\hat{\mathcal{O}}$ has a half-space representation $\hat{\mathcal{O}} = \{x : h_j x \le k_j, j = 1, \dots, N\}$ Thus, $\frac{1}{\nu} h_j \dot{x}_i \le k_j - h_j x_i$ since $x_i + \frac{1}{\nu}\dot{x}_i \in \hat{\mathcal{O}}$. For all half-spaces incidence to the vertex x_i, we have $h_j x_i = k_j$. Therefore, the vector-field $\dot{x}$ points $h_j \dot{x} \le 0$ into $\hat{\mathcal{O}}$ at x_i.

Next, consider the vertex $x_i = (e_i, -\nu e_i)$ and control input $u = \nu^2 e_i \in \mathcal{U}$. Then, $x_i + \frac{1}{\nu}\dot{x}_i = 0 \in \hat{\mathcal{O}}$. Following a similar argument as above, we conclude that the vector-field $\dot{x}$ points $\frac{1}{\nu} h_j \dot{x} \le 0$ into $\hat{\mathcal{O}}$ at x_i where $\nu > 0$.

Thus, the set $\hat{\mathcal{O}}$ is CI, i.e., for all $x(t) \in \hat{\mathcal{O}}$ there exists $u(t) \in \mathcal{U}$ that keeps the state inside $\hat{\mathcal{O}}$. Inverting the nonlinear transformation, this means for all $(\theta, \dot{\theta}) \in \mathcal{O}$ there exists $\tau(t) \in \mathcal{T}$ that keeps the state inside $\hat{\mathcal{O}}$. ∎

Since the transformed constraint sets $\hat{\mathcal{O}}$ and $\hat{\mathcal{U}}$ are polytopic and the feedback linearized dynamics (11) are linear, we can use the standard methods [17] to compute the maximal control/positive invariant set for this problem. However, computing this set for a multi-link robot $n \gg 1$ may be computationally intractable. Furthermore, the resulting invariant set may require an excessive amount of memory to store either its half-space representation or vertex representation. In contrast, the CI set (9) is pre-computed, and the number of vertices depends linearly $2n$ on the number of links n.

The tuning-parameter $\nu > 0$ limits the joint-velocities $\dot{\theta}_i$ inside the CI set $\mathcal{O}$, i.e., $|\dot{\theta}_i| \le \nu$. Intuitively, it is clear that if the robot moves too quickly then its momentum can carry it out of the configuration-space bubble (5). This intuition is quantified by condition (10) which provides a relationship betwen the maximum joint-velocities ν and the torque authority (4) of the joint-motors. Condition (10) requires that the coriolis and centripetal torques $C(\theta, \dot{\theta})\dot{\theta}$ do not saturate the joint-motors, allowing a deceleration of at least $|\ddot{\theta}_i| = \nu^2$ for each joint $i = 1, \dots, n$ to stop the robot before it leaves the CI set. In the next section, we discuss how to compute this bound to ensure that the CI set $\mathcal{O}$ is input admissible.

According to Theorem 1, the set (9) is CI meaning that *there exists* a feasible controller which can keep the robot state $(\theta, \dot{\theta})$ inside the set $\mathcal{O}$. The ISMP will require a controller that renders the CI set (9) *positive* invariant. Although the proof of Theorem 1 suggests a vertex controller [17], it would be overly conservative and inflexible. Instead, we propose a CG [18] which can transform any controller into a constraint enforcing controller. The CG uses the following input-admissible set

$$\mathcal{U}(\theta, \dot{\theta}) = \left\{ \tau \in \mathcal{T} : \begin{array}{c} \exists \delta > 0 \text{ s.t.} \\ \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} + \delta \begin{bmatrix} \dot{\theta} \\ M^{-1}(\tau + C\dot{\theta}) \end{bmatrix} \in \mathcal{O} \end{array} \right\} \quad (12)$$

where the parameter $\delta > 0$ ensures that a point $x + \delta\dot{x}$ along the ray from x passing through $x + \dot{x}$ is contained inside $\mathcal{O}$. This is equivalent to showing that $\dot{x} = (\dot{\theta}, \ddot{\theta})$ points into $\mathcal{O}$. In practice, we use a fixed $0 < \delta \ll 1$. This set describes all feasible torques $\tau \in \mathcal{T}$ that ensure that the CI set (9) becomes PI. The CG computes the admissible torque $\tau = \kappa(\theta, \dot{\theta}) \in \mathcal{U}(\theta, \dot{\theta})$ closest to the desired torque $\bar{\kappa}(\theta, \bar{\theta}, \dot{\theta})$

$$3\kappa(\theta, \dot{\theta}) = \arg\min_{\tau} \ \|\tau - \bar{\kappa}(\theta, \bar{\theta}, \dot{\theta})\| \quad (13a)$$

$$\text{s.t.} \quad \tau \in \mathcal{U}(\theta, \dot{\theta}) \quad (13b)$$

where $\bar{\kappa}(\theta, \bar{\theta}, \dot{\theta})$ is any tracking $\theta \to \bar{\theta}$ controller. The CG only interferes with the nominal controller when necessary to enforce constraints. If the requested torque $\tau = \bar{\kappa}(\theta, \dot{\theta}, \bar{\theta})$ is admissible $\tau \in \mathcal{U}(\theta, \dot{\theta})$ then it is implemented $\tau^\star = \bar{\kappa}(\theta, \bar{\theta}, \dot{\theta})$. For the i-th reference $\bar{\theta}_i$ and PI set $\mathcal{O}_i$, we denote the resulting controller (13) as $\kappa_i(\theta, \dot{\theta})$. Alternatively, the CG could be implemented with a control barrier function [19] whose level-sets are PI.

Note that θ and $\dot{\theta}$ are parameters, not decision variables in the optimization problem (13). Likewise, for $M(\theta)$ and $C(\theta, \dot{\theta})\dot{\theta}$. Thus, the constraints (13b) are polytopic since $\mathcal{O}$ is a polytope. Therefore, if the cost (13a) is a 2-norm then the CG (13) can be implemented as a quadratic program. Likewise, if the cost (13a) is a 1-norm or ∞-norm then the CG (13) can be implemented as a linear program. Thus, the CG (13) is computationally tractable.

C. Input Admissibility

In this section, we provide a joint-velocity bound $\nu > 0$ which satisfies (10) ensuring that the set (9) is CI. The velocity bound is given by the following proposition.

Proposition 2 (Velocity Bound): Condition (10) holds for

$$\nu \leq \sqrt{\frac{\bar{\rho}}{m + c\bar{\rho}} \min_{j=1,\dots,N_\tau} \frac{k_j}{\|h_j\|}} \tag{14}$$

where $\bar{\rho} = \max_{i=1,\dots,n} \rho_i$, and m and c are uniform bounds on the mass-inertia matrix $\|M(\theta)\| \leq m$ and corilois-centripetal matrix $\|C(\theta,\dot{\theta})\| \leq c\|\dot{\theta}\|^2$, respectively.

Proof: Condition (10) is equivalent to

$$\bigcup_{(\theta,\dot{\theta})\in\mathcal{O},\ \ddot{\theta}\in\nu^2 P^{-1}\mathcal{B}_1} M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} \subseteq \mathcal{T}$$

We will outer-approximate the non-convex set on the left-hand side. For $(\theta,\dot{\theta}) \in \mathcal{O}$, the following bound holds $\|C(\theta,\dot{\theta})\dot{\theta}\| \leq \max_{(\theta,\dot{\theta})\in\mathcal{O}} c\|\dot{\theta}\|^2 = c\nu^2$ where the inequality follows from the uniform bound on the corilois/centripetal matrix [20] and the equality follows from $\dot{\theta} = \nu e_i$ being the largest magnitude velocity $\|\dot{\theta}\|_2 \leq \|\dot{\theta}\|_1 \leq \nu$ for velocity in (9). Likewise, the following bound on the mass-inertia term holds $\max_{\ddot{\theta}\in\nu^2 P^{-1}\mathcal{B}_1} \|M(\theta)\ddot{\theta}\| \leq \max_{\ddot{\theta}\in\nu^2 P^{-1}\mathcal{B}_1} m\|\ddot{\theta}\| = m\nu^2/\bar{\rho}$ where $\bar{\rho} = \max_{i=1,\dots,n} \rho_i(\bar{\theta})$ and the inequality follows from the uniform bound on the mass-inertia matrix [21]. Thus, $M(\theta)\ddot{\theta} \in \nu^2 m/\bar{\rho}\mathbb{B}_2$. Therefore, the following set inclusion holds

$$\bigcup_{(\theta,\dot{\theta})\in\mathcal{O},\ \ddot{\theta}\in\nu^2 P^{-1}\mathcal{B}_1} M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} \subseteq (m/\bar{\rho} + c)\nu^2\mathbb{B}_2.$$

Hence, (10) holds if $\mathcal{T}$ includes the ball $(m/\bar{\rho} + c)\ \nu^2\mathbb{B}_2$ of radius $(m/\bar{\rho} + c)\ \nu^2$. Or equivalently, $h_j^\top h_j(m/\bar{\rho} + c)\ \nu^2 \leq k_j^2$ for $j = 1,\dots,N_\tau$. Solving for ν yields (14). ■

The velocity bound $\nu > 0$ in Proposition 2 ensures that the joint-motors can produce sufficient torque to keep the robot configuration inside the invariant set (9). For serial revolute robots, uniform bounds on the mass-inertia matrix $\|M(\theta)\| \leq m$ [21] and the coriolis/centripetal matrix $\|C(\theta,\dot{\theta})\dot{\theta}\| \leq c\|\dot{\theta}\|^2$ [20] are guaranteed to exist. Note that a non-negative velocity bound (14) exists since $\mathcal{T}$ is assumed to contain the origin in its interior and therefore $k_j > 0$. The velocity-bound is tighter for smaller $\bar{\rho}$ configuration-space bubbles (5). Likewise, larger inertia bounds m or corilois/centripetal bounds c will tighten the bound (14) on the maximum velocity.

IV. Illustrative Example

In this section, we demonstrate our path planning algorithms through an illustrative numerical example. For illustrative purpose, we consider a planar 2-link robot so that both the configuration-space and work-space can be shown in the plane. The robot is shown in Fig. 1. The dynamics (1) and spatial-extent (2) were modeled using MATLABs robotics toolbox [22]. The objective is to move the robot from the home configuration $\bar{\theta}_0 = (0,0)$ to the vertical position $\bar{\theta}_\infty = (\frac{\pi}{2}, 0)$ while avoiding the single obstacle $\mathcal{B}$ shown in Fig. 1. The obstacles $\mathcal{B}$ in configuration-space is shown in Fig. 2.

Algorithm 2 was used to construct a search-tree $\mathfrak{T} = (\mathfrak{I}, \mathfrak{E})$ for the Algorithm 1. We uniformly sampled configurations

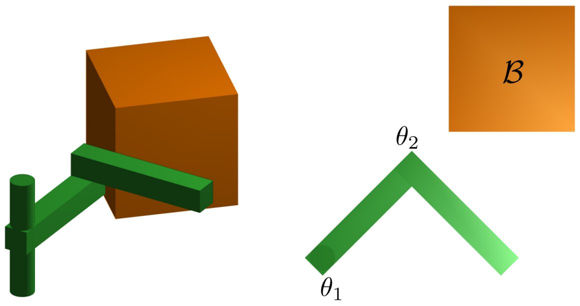

Fig. 1. The robot considered in this illustrative example.

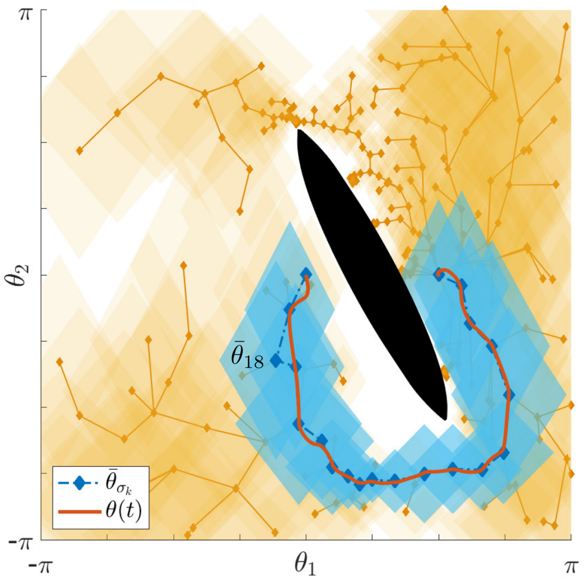

Fig. 2. The obstacle $\mathcal{B}$ is shown in configuration-space $\mathbb{S}^2$. Algorithm 2 planned a path $\bar{\theta}_{\sigma_k}$ around the obstacle. The robot trajectory $\theta(t)$ tracks the path $\bar{\theta}_{\sigma_k}$ without leaving the corridor $\mathcal{O}_{\sigma_k}$ of CAPI sets. Video at https://youtu.be/EDCwnRzLiVg.

$\bar{\theta} \in [-\pi,\pi]^2$. The sampled configuration $\bar{\theta}$ was moved to connect $(i,j) \in \mathfrak{E}$ to the nearest node $j \in \mathfrak{I}$ using the convex combination

$$\bar{\theta}_i = \bar{\theta}_j + \frac{\lambda}{\|P(\bar{\theta}_j)(\bar{\theta} - \bar{\theta}_j)\|_1}(\bar{\theta} - \bar{\theta}_j) \in \mathrm{int}([I, 0]\mathcal{O}_j)$$

where the tuning-parameter $\lambda \in [0,1]$ determines how deeply $\bar{\theta}_i$ lies inside $[I, 0]\mathcal{O}_j$. The weighting matrix $P(\bar{\theta}_i)$ for the new node $i \in \mathfrak{I}$ was generated using (5) where $d(p_{s0}(\bar{\theta}), \mathcal{B})$ was computed using the collision checker in MATLABs robotic toolbox. Fig. 2 shows the references $\bar{\theta}_i$ and CAPI sets $\mathcal{O}_i$ associated with the nodes $i \in \mathfrak{I}$ of the search-tree $\mathfrak{T} = (\mathfrak{I}, \mathfrak{E})$ as well as the edges $(i,j) \in \mathfrak{E}$ connecting these nodes. The search-tree $\mathfrak{T}$ has $|\mathfrak{I}| = 261$ nodes and $|\mathfrak{E}| = 260$ edges. Algorithm 2 required 2.45 seconds to construct the tree $\mathfrak{T}$. The tree was re-wired using an RRT*-like scheme to produce more efficient paths [23].

The simulation results for the robot (1) in closed-loop with Algorithm 1 are shown in Fig. 3. Algorithm 1 searched the tree $\mathfrak{T}$ for the shortest path $\{\sigma_k\}_{k=1}^{20} \subset \mathfrak{I}$ from $\bar{\theta}_{\sigma_1} = \theta(0)$ to $\bar{\theta}_{\sigma_{20}} = \theta_\infty$ as shown Fig. 2. The nominal controller $\kappa(\theta,\bar{\theta},\dot{\theta}) = -M(\theta)F(\theta - \bar{\theta},\dot{\theta}) + C(\theta,\dot{\theta})\dot{\theta}$ used in Algorithm 1 is a feedback-linearized linear quadratic regulator (LQR) where the LQR-gain $F \in \mathbb{R}^{2\times4}$ was computed for

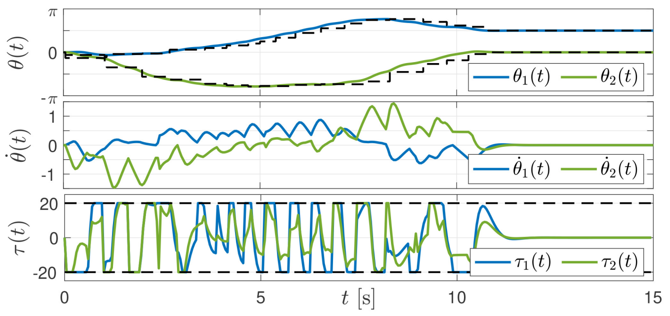

Fig. 3. The angular position $\boldsymbol{\theta}(t)$, velocity $\dot{\boldsymbol{\theta}}(t)$, and torque $\boldsymbol{\tau}(t)$ for the two revolute joints. The aggressive controller saturates the motors, but the CG (13) prevents collisions.

the double-integrator (11) with $Q = \text{diag}(1, 0.1, 0, 0) \in \mathbb{R}^{4\times 4}$ and $R = 10^{-3}I \in \mathbb{R}^{2\times 2}$. Algorithm 1 modifies the nominal controller using the CG (13). Algorithm 1 is executed in discrete-time with a sample-time of $\delta t = 50$ millisecond (20 hertz) and $\delta = \delta t$ was used in (12). The CG (13) was posed as a quadratic program which was solved using YALMIP toolbox [24] and `quadprog`. The CG required 15 milliseconds on average and 40 milliseconds maximum to execute. Between samples, the robot dynamics were simulated in continuous-time.

Without the results of this letter, the nominal controller fails the motion planning problem. First, the asymmetric state cost $Q = \text{diag}(1, 0.1, 0, 0)$ produces nonlinear closed-loop trajectories between references. The nonlinear trajectories can leave *non-invariant* configuration-space bubbles (5) even though the end-points of the trajectory satisfy the constraints. Second, the low input cost $R = 10^{-3}I$ produces an aggressive controller that saturates the joint-motors. Without the forethought provided by an invariant set, the weak motors cannot stop the momentum of the robot from carrying it out of the configuration-space bubbles (5).

The CAPI subsets (9) of the configuration-space bubbles (5) presented in this letter alleviate these issues. For each node $\sigma_k \in \mathfrak{I}$ along the path $\{\sigma_k\}_{k=1}^{20} \subset \mathfrak{I}$, the CG produces a nonlinear controller (13) that enforces the CAPI set $\mathcal{O}_{\sigma_k}$. Thus, the trajectory of the robot (1) in closed-loop with Algorithm 1 remains inside the safe corridor of CAPI sets $\mathcal{O}_{\sigma_k}$ as shown in Fig. 2. This is despite the saturation of the torque constraints shown in Fig. 3. One of the main advantages of the ISMP is that collision avoidance is guaranteed despite imperfect tracking. This is demonstrated in Fig. 2 where the trajectory effectively ignored the reference $\bar{\theta}_{18}$. Despite this poor tracking the trajectory does not leave the corridor of CAPI sets and therefore we can guarantee that collisions are avoided.

Acknowledgment

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation nor the United States Air Force.

References

[1] C. Danielson, J. Kloeppel, and C. Petersen, "Spacecraft attitude control using the invariant-set motion-planner," *IEEE Control Syst. Lett.*, vol. 6, pp. 1700–1705, 2022.
[2] A. Weiss, C. Danielson, K. Berntorp, I. Kolmanovsky, and S. Di Cairano, "Motion planning with invariant set trees," in *Proc. Conf. Control Technol. Appl.*, 2017, pp. 1625–1630.
[3] A. Weiss, F. Leve, M. Baldwin, J. R. Forbes, and I. Kolmanovsky, "Spacecraft constrained attitude control using positively invariant constraint admissible sets on SO(3) × R3," in *Proc. Amer. Control Conf.*, 2014, pp. 4955–4960.
[4] G. R. Frey, C. D. Petersen, F. A. Leve, I. V. Kolmanovsky, and A. R. Girard, "Constrained spacecraft relative motion planning exploiting periodic natural motion trajectories and invariance," *J. Guid., Control, Dyn.*, vol. 40, no. 12, pp. 3100–3115, 2017.
[5] K. Berntorp, A. Weiss, C. Danielson, S. Di Cairano, and I. Kolmanovsky, "Automated driving: Safe motion planning using positive-invariant sets," in *Proc. Intell. Transp. Syst. Conf.*, 2017, pp. 1–6.
[6] K. Berntorp, R. Bai, K. F. Erliksson, C. Danielson, A. Weiss, and S. D. Cairano, "Positive invariant sets for safe integrated vehicle motion planning and control," *IEEE Trans. Intell. Veh.*, vol. 5, no. 1, pp. 112–126, Mar. 2020.
[7] C. Danielson, A. Weiss, K. Berntorp, and S. D. Cairano, "Path planning using positive invariant sets," in *Proc. Conf. Decis. Control*, 2016, pp. 5986–5991.
[8] C. Danielson, K. Berntorp, S. D. Cairano, and A. Weiss, "Motion-planning for unicycles using the invariant-set motion-planner," in *Proc. Amer. Control Conf.*, 2020, pp. 1235–1240.
[9] C. Danielson, K. Berntorp, A. Weiss, and S. D. Cairano, "Robust motion planning for uncertain systems with disturbances using the invariant-set motion planner," *IEEE Trans. Autom. Control*, vol. 65, no. 10, pp. 4456–4463, Oct. 2020.
[10] J.-P. Aubin, *Viability Theory*. Boston, MA, USA: Birkhäuser, Inc., 1991.
[11] S. LaValle and J. Kuffner, "Randomized kinodynamic planning," *Int. J. Robot. Res.*, vol. 20, no. 5, pp. 1–37, 2001.
[12] B. Lacevic and D. Osmankovic, "Improved C-space exploration and path planning for robotic manipulators using distance information," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2020, pp. 1176–1182.
[13] X. Zhang, R. Belfer, P. G. Kry, and E. Vouga, "C-space tunnel discovery for puzzle path planning," *ACM Trans. Graph.*, vol. 39, no. 4, p. 104, Jul. 2020.
[14] A. Ademovic and B. Lacevic, "Path planning for robotic manipulators using expanded bubbles of free C-space," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2016, pp. 77–82.
[15] S. Quinlan, *Real-Time Modification of Collision-Free Paths*. Stanford, CA, USA: Stanford Univ., 1994.
[16] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, "LQR-trees: Feedback motion planning via sums-of-squares verification," *Int. J. Robot. Res.*, vol. 29, no. 8, pp. 1038–1052, 2010.
[17] F. Blanchini and S. Miani, *Set-Theoretic Methods in Control*. Cham, Switzerland: Birkhäuser, 2008.
[18] E. Garone, S. Di Cairano, and I. Kolmanovsky, "Reference and command governors for systems with constraints: A survey on theory and applications," *Automatica*, vol. 75, pp. 306–328, Jan. 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109816303715
[19] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *Proc. 18th Eur. Control Conf. (ECC)*, 2019, pp. 3420–3431.
[20] J. I. Mulero-Martínez, "Uniform bounds of the coriolis/centripetal matrix of serial robot manipulators," *IEEE Trans. Robot.*, vol. 23, no. 5, pp. 1083–1089, Oct. 2007.
[21] F. Ghorbel, B. Srinivasan, and M. W. Spong, "On the uniform boundedness of the inertia matrix of serial robot manipulators," *J. Robot. Syst.*, vol. 15, no. 1, pp. 17–28, 1998.
[22] *MATLAB, Version 9.9.0 (R2020b)*, MathWorks, Inc., Natick, NA, USA, 2020.
[23] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 846–894, 2011.
[24] J. Löfberg, "YALMIP: A toolbox for modeling and optimization in MATLAB," in *Proc. CACSD Conf.*, Taipei, Taiwan, 2004, pp. 284–289.