2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Inner-Approximation of Manipulable and Reachable Regions using Bilinear Matrix Inequalities

Zherong Pan[1], Liang He[1], and Xifeng Gao[2]

***Abstract*— Given an articulated robot arm, we present a method to identify two regions with non-empty interiors. The first region is a subset of the configuration space where every point in the region is manipulable. The second region is a subset of the workspace where every point in the region is reachable by the end-effector. Our method expresses the kinematic state of the robot arm using the maximal coordinates, so that the kinematic constraints take polynomial forms. We then reformulate the optimization-based inverse kinematics (IK) algorithm as gradient flows. Finally, we use sum-of-squares (SOS) programming to certify the convergence of each gradient flow. Our main result shows that the feasibility of an SOS programming problem is a sufficient condition for the manipulability and reachability of the sublevel sets of polynomial functions. Our method can be used to certify manipulable or reachable regions by solving a set of linear matrix inequalities (LMIs) or to maximize the volume of a region by solving a set of bilinear matrix inequalities (BMIs). These identified regions can then be used in various motion planning problems as hard safety constraints.**

I. Introduction

When computing a continuous trajectory for a robot arm in the Cartesian space [27], it is important to make sure that every point on the trajectory can be reached by the robot's end-effector. Further, a "safe" trajectory should also be robust to uncertainties [44], so that an arbitrary perturbation to the trajectory can be compensated by a change in the configuration. These two requirements are known as reachability [26] and manipulability [44]. A manipulable point is a point in the robot's configuration space, from which a robot can move its end-effector along any directions. A reachable point is a point in the workspace, which belongs to the image of the forward kinematic function.

For a typical robot arm with redundant degrees of freedom, the set of manipulable points will form manipulable regions with non-empty interiors in the configuration space. Similarly, reachable points will form regions in the workspace. However, these regions have irregular boundaries that are difficult to represent discretely. To approximate the shape of these regions, previous works [26], [10], [33] propose to first find a discrete set of manipulable or reachable points and then connect these points. However, these methods are not guaranteed to produce inner-approximation, i.e. the approximations might not preserve manipulability or reachability. In this paper, we address the problems of inner-approximations of manipulable and reachable regions for a given kinematic system.

Unlike the reachability of a kinematic system, forward and backward reachability problems of a dynamic system have been vastly studied in previous works, i.e. [9], [29], [11], [17], [37], [8], [4], [42], [43]. Some of these works [29] derive outer-approximations of reachable regions, others like [42] propose inner-approximations. These works require that a dynamic system be represented in the explicit form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \tag{1}$$

where $\mathbf{x}$ is the configuration, and all entries of $\mathbf{f}$ are polynomial functions. Unfortunately, the kinematic reachability problem takes a different mathematical form, and cannot be analyzed as a dynamic system.

Main Result: We present a method to derive inner-approximations for regions of manipulability and reachability. Our main idea is that, although the kinematic system cannot be represented in the form of Equation 1, the optimization-based IK algorithms can be understood as gradient flows, which in turn take the form of Equation 1. As a result, we can use similar ideas as [30] to identify the convergent region of an IK algorithm, which corresponds to manipulable and reachable regions. In other words, we show that an IK algorithm will converge if 4 conditions hold and these 4 conditions can be certified by checking the feasibility of a set of matrix inequalities.

Our method proceeds as follows. First, we consider a robot's kinematic constraints under the maximal coordinates [5], so that the forward kinematic function is reformulated as a set of polynomial constraints (Section III). Second, we analyze various forms of optimization-based IK algorithms and derive their corresponding gradient flows (Section IV). We then present the conditions under which the trajectory of a gradient flow will converge, and we transform the conditions into matrix inequalities by applying the S-procedure (Section V). We highlight that our method can identify non-trivial manipulable and reachable regions with non-empty interiors for different 2D robot arms (Section VI).

II. Related Work

We review related topics in IK, Lyapunov theory, sum-of-squares programming, and reachable set computation.

A. Inverse Kinematics

Inverse kinematic algorithms recover a robot's configuration, given the end-effector's desired position in the workspace. IK problems can be solved using numerical methods [36], [39], [41] or machine learning methods [14], [23]. In particular, an optimization-based IK algorithm updates the robot's configuration under the minimal coordinates and move the end-effector towards its goal position, but there is no guarantee that the goal position can be reached. Methods have been proposed to improve the robustness of IK algorithms by performing approximate reachability analysis [26], [43]. These methods are based on sampling and discretization and they do not guarantee that the approximation is inner. By comparison, algebraic methods [18], [28] can check exact

[1]Zherong Pan & Liang He are with Department of Computer Science, University of North Carolina at Chapel Hill. {zherong@cs.unc.edu,lianghe.hust@gmail.com} [2]Xifeng Gao is with Department of Computer Science, Florida State University. {gao@cs.fsu.edu}

reachability or detect infeasibility. Both algebraic methods and our method are based on the maximal coordinates of the robot. The main difference is that algebraic methods check reachability for a single point, while our method can identify an entire reachable region. The notion of manipulability [34], [26] is closely related to reachability: a robot's end-effector can move from a manipulable point to any point in a small neighborhood. The manipulability ellipsoid [44] is an indication of the manipulable neighborhood size, while our method provides an inner-approximation of this neighborhood.

B. Lyapunov Theory

A Lyapunov function can be used to identify a region of attraction for a general dynamic system [30]. A time-varying Lyapunov function can be understood as funnel [6], such that a trajectory starting inside the funnel will always stay inside. It is only after sum-of-squares programming is invented in [24] that Lyapunov functions can be determined computationally. In particular, previous works [31], [25], [25] have used Lyapunov functions to analyze numerical optimization algorithms. For example, authors of [31] use special Lyapunov functions to provide convergence guarantee or prove the convergence speed of first-order methods. We also use Lyapunov functions to analyze optimization-based IK algorithms, but we are not interested in the speed of convergence. Instead, we use Lyapunov functions to restrict all the points generated by the optimizer inside a closed, viable region, by using the Nagumo's theorem:

***Lemma** 2.1 (Nagumo 's Theorem [3]):* If $\mathbf{f}$ in Equation 1 is continuous on a compact set $\mathcal{S}$, and if we have $\mathbf{f} \in \mathcal{T}(\mathcal{S}, \mathbf{x})$ for every $\mathbf{x} \in \partial\mathcal{S}$, then $\mathcal{S}$ is forward invariant for at least one solution to Equation 1. Here $\mathcal{T}(\mathcal{S}, \mathbf{x})$ is the tangent cone of $\mathcal{S}$ at $\mathbf{x}$.

C. Sum-of-Squares Programming

Sum-of-squares programming can be used to approximate (from below) the minimal value of a polynomial function, when restricted to a (semi-)algebraic set. A semi-algebraic set $\mathcal{A}$ is defined by some equality constraints $\mathbf{F}$ and inequality constraints $\mathbf{G}$, where each equation in $\mathbf{F}, \mathbf{G}$ is a polynomial:

$$\mathcal{A} = \{\mathbf{x} | \mathbf{F}(\mathbf{x}) = 0, \mathbf{G}(\mathbf{x}) \geq 0\}.$$

If another polynomial $P(\mathbf{x})$ is positive on $\mathcal{A}$, then a sufficient condition can be derived by the generalized S-procedure:

$$\begin{aligned} &P(\mathbf{x}) - L_{eq}(\mathbf{x})\mathbf{F}(\mathbf{x}) - L_{neq}(\mathbf{x})\mathbf{G}(\mathbf{x}) \in \text{SOS} \\ &L_{neq}(\mathbf{x}) \in \text{SOS}, \end{aligned} \tag{2}$$

where SOS is the set of sum-of-squares polynomials and $\mathbf{x}$ is a set of polynomial variables. If some polynomial $P(\mathbf{x}) \in$ SOS, then $P_R(\mathbf{x}) = \mathcal{M}(\mathbf{x})^T \mathbf{H} \mathcal{M}(\mathbf{x})$, where $\mathcal{M}$ contains monomials of $\mathbf{x}$ and $\mathbf{H}$ is a positive semi-definite (PSD) matrix. In this way, polynomial variables can be eliminated and the constraint $P(\mathbf{x}) \in$ SOS is transformed in a PSD-cone: $\mathbf{H} \in$ PSD. The sufficient condition is also necessary when the semi-algebraic set is Archimedean [13], which is usually the case in our problem because $\mathbf{x}$ is bounded in our definition of $\mathcal{A}$. Although Equation 2 is infinite dimensional in the polynomial variables $\mathbf{x}$, verifying that a polynomial belongs to SOS can be transformed into a matrix inequality in the polynomial coefficients, which is finite dimensional [24].

Sum-of-squares programming finds applications in stability analysis [30], reachable set computation [42], and collision detection [1]. However, when the semi-algebraic set is not fixed and contains decision variables, the SDP constraints become bilinear matrix inequalities (BMIs). These BMI constraints arise when a controller is optimized to maximize the stability region [15] or the size of a funnel is minimized to resist disturbance [16]. Sum-of-squares programming can only solve small problems because it induces semidefinite programming (SDP) problems, whose size is combinatorial in the number of polynomial variables. Some recent advances propose alternative optimization formulations based on linear programming (LP), second-order cone programming (SOCP) [2], and a mixture of LP and SDP [12], [38], that are more efficient to solve. However, LP/SOCP-based methods are too restrictive in the solution space and the LP+SDP-based formulation is not convenient when handling BMI constraints.

D. Reachable Set Computation

The reachable set problem of a dynamic system has been studied for decades [9], [29], [11], [17], [37], [8], [4], [42], [43]. Some of these techniques [9], [11], [4] are only applicable to linear systems. For nonlinear dynamics, both forward [20] and backward [42] reachable sets are identified based on the HJB equation. Our work considers the reachability of a kinematic problem, instead of a dynamic system, while we borrow these techniques by formulating IK algorithms as gradient flows of a numerical optimization. A similar approach is used in [40] to analyze the convergence of first order moment methods.

III. Problem Statement

In this section, we formulate a 2D robot's forward kinematic problem under both minimal and maximal coordinates [5], and we give the formal definitions of the manipulable and reachable regions. We assume that the robot is an articulated body with N links and the consecutive links are either connected by hinge joints or prismatic joints.

A. Minimal Coordinates

We denote the minimal coordinate vector as $\mathbf{x}^{min}$, which is a N-dimensional vector consisting of joint angles or translational distances. We further assume that $\mathbf{x}^{min}$ is under the joint limits:

$$\mathbf{L} \leq \mathbf{x}^{min} \leq \mathbf{U}, \tag{3}$$

where $\mathbf{L}, \mathbf{U}$ are the lower and upper bounds, respectively. The forward kinematic function maps $\mathbf{x}^{min}$ to the links' global rotation and translation by a recursive rule. We denote the ith link's global rotation by $\mathbf{R}_i$ and translation by $\mathbf{t}_i$ and we assume $\mathbf{R}_0 = \mathbf{I}$ and $\mathbf{t}_0 = \mathbf{0}$. If the ith link and the $i-1$th

link are connected using a hinge joint, then we have:

$$\begin{aligned}\mathbf{R}_i &= \mathbf{R}_{i-1}\begin{pmatrix}\mathbf{cos}(\mathbf{x}_i^{min}) & -\mathbf{sin}(\mathbf{x}_i^{min})\\ \mathbf{sin}(\mathbf{x}_i^{min}) & \mathbf{cos}(\mathbf{x}_i^{min})\end{pmatrix}\\ \mathbf{t}_i &= \mathbf{R}_{i-1}\mathbf{l}_{i-1} + \mathbf{t}_{i-1} - \mathbf{R}_i\mathbf{l}_i.\end{aligned} \tag{4}$$

Here $\mathbf{x}_i^{min}$ is the ith component of $\mathbf{x}^{min}$, $\mathbf{l}_{i-1}, \mathbf{l}_i$ are the local translations from the $i-1$th and ith link to the joint position, as illustrated in Figure 1 (a). If the ith link and the $i-1$th link are connected using a prismatic joint, we have:

$$\begin{aligned}\mathbf{R}_i &= \Delta\mathbf{R}_i\mathbf{R}_{i-1}\\ \mathbf{t}_i &= \mathbf{R}_{i-1}\mathbf{l}_{i-1} + \mathbf{t}_{i-1} - \mathbf{R}_i\mathbf{d}_i\mathbf{x}_i^{min}.\end{aligned} \tag{5}$$

Here $\Delta\mathbf{R}_i$ is the relative rotation from the $i-1$th link to the ith link, and $\mathbf{d}_i$ is the translational direction in the local frame of reference, as illustrated in Figure 1 (b).

The end-effector's position is the global position of a point on the Nth link. If the local coordinates of this point is $\mathbf{X}_N$, then we define the end-effector as $\mathbf{E}^{min}(\mathbf{x}^{min}) \triangleq \mathbf{R}_N\mathbf{X}_N + \mathbf{t}_N$. Note that Equation 4 and Equation 5 can be eliminated recursively and incorporated into the function $\mathbf{E}^{min}$. Under these definitions, the set of manipulable points is defined as:

$$\mathcal{S}_M^{min} = \left\{\mathbf{x}^{min}\middle|\forall\Delta\mathbf{E}, \exists\Delta\mathbf{x}^{min}, \Delta\mathbf{E} = \frac{\partial\mathbf{E}(\mathbf{x}^{min})}{\partial\mathbf{x}^{min}}\Delta\mathbf{x}^{min}\right\},$$

and the set of reachable points is defined as:

$$\mathcal{S}_R^{min} = \{\mathbf{E}^{min}(\mathbf{x}^{min})|\mathbf{L} \le \mathbf{x}^{min} \le \mathbf{U}\}.$$

B. Maximal Coordinates

We use $\mathbf{x}$ without subscript to denote the maximal coordinates. In 2D cases, the maximal coordinates are represented as a $4N$-dimensional vector with four scalar variables $\begin{pmatrix}s_i & c_i & x_i & y_i\end{pmatrix}$ to represent the ith link's global rotation and translation as follows:

$$\mathbf{R}_i = \begin{pmatrix}c_i & -s_i\\ s_i & c_i\end{pmatrix} \quad \mathbf{t}_i = \begin{pmatrix}x_i\\ y_i\end{pmatrix}.$$

To ensure the rigidity of the link, we introduce an additional constraint:

$$c_i^2 + s_i^2 = 1. \tag{6}$$

If the ith link is connected to the $i-1$th link via hinge joint, then we have the following constraints to model their connectivity:

$$\begin{aligned}&\mathbf{R}_i\mathbf{l}_i + \mathbf{t}_i = \mathbf{R}_{i-1}\mathbf{l}_{i-1} + \mathbf{t}_{i-1}\\ &\begin{pmatrix}c_i & s_i\end{pmatrix}\mathbf{R}_{i-1}\begin{pmatrix}\mathbf{cos}(\frac{\mathbf{U}_i+\mathbf{L}_i}{2})\\ \mathbf{sin}(\frac{\mathbf{U}_i+\mathbf{L}_i}{2})\end{pmatrix} \ge \mathbf{cos}(\frac{\mathbf{U}_i-\mathbf{L}_i}{2}),\end{aligned} \tag{7}$$

where $\mathbf{L}_i, \mathbf{U}_i$ are the ith component of $\mathbf{L}, \mathbf{U}$, respectively. If the ith link is connected to the $i-1$th link via prismatic joint, then we have the following constraints:

$$\begin{aligned}&\mathbf{R}_i = \Delta\mathbf{R}_i\mathbf{R}_{i-1}\\ &\mathbf{R}_i\mathbf{d}_i \times (\mathbf{t}_i - \mathbf{R}_{i-1}\mathbf{l}_{i-1} - \mathbf{t}_{i-1}) = 0\\ &\mathbf{L}_i \le \mathbf{t}_i^T\mathbf{R}_i\mathbf{d}_i \le \mathbf{U}_i.\end{aligned} \tag{8}$$

Unlike the minimal coordinates, Equation 6, Equation 7, and Equation 8 cannot be eliminated and we arrange them into two sets of equality and inequality polynomial constraints:

$$\mathbf{C}_{eq}(\mathbf{x}) = 0 \quad \mathbf{C}_{neq}(\mathbf{x}) \ge 0.$$

The end-effector of the robot under the maximal coordinates takes the same form as that under the minimal coordinates, denoted as: $\mathbf{E}(\mathbf{x}) = \mathbf{R}_N\mathbf{X}_N + \mathbf{t}_N$. Under these definitions, the set of manipulable points is:

$$\mathcal{S}_M = \left\{\mathbf{x}\middle|\forall\Delta\mathbf{e}, \exists\Delta\mathbf{x}, \frac{\partial\mathbf{C}_{eq}(\mathbf{x})}{\partial\mathbf{x}}\Delta\mathbf{x} = 0, \Delta\mathbf{e} = \frac{\partial\mathbf{E}(\mathbf{x})}{\partial\mathbf{x}}\Delta\mathbf{x}\right\},$$

and the set of reachable points is:

$$\mathcal{S}_R = \{\mathbf{E}(\mathbf{x})|\mathbf{C}_{eq}(\mathbf{x}) = 0, \mathbf{C}_{neq}(\mathbf{x}) \ge 0\}.$$

It is trivial to verify that the reachable and manipulable sets defined under the minimal and the maximal coordinates are identical, i.e. $\mathcal{S}_{R,M} = \mathcal{S}_{R,M}^{min}$. Our problem is to find two strict subsets: $\bar{\mathcal{S}}_M \subset \mathcal{S}_M$ and $\bar{\mathcal{S}}_R \subset \mathcal{S}_R$. Previous methods [26], [10], [33] have shown that the manipulability and reachability of a certain point $\mathbf{x}$ can be checked. However, our goal is to identify subsets $\bar{\mathcal{S}}_M$ and $\bar{\mathcal{S}}_R$ with non-empty interiors, containing infinite number of points.

Remark 1: We use the symbol $\mathbf{x}$ to denote the maximal coordinates of a robot arm, the configuration of a dynamic system in Equation 1, and the polynomial variables in a SOS constraints (Equation 2). This will not cause ambiguity as $\mathbf{x}$ plays these three roles under different contexts.

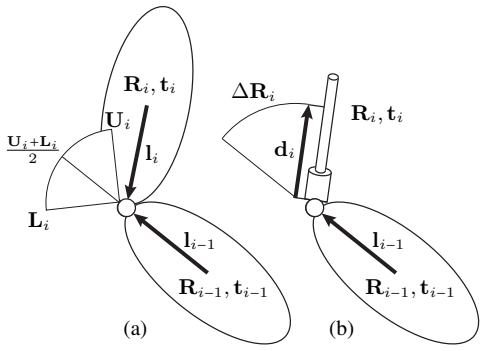

Fig. 1: Connectivity constraints between the ith link and the $i-1$th link can be hinge joints (a) and prismatic joints (b).

IV. Gradient-Flow of IK Algorithm

In this section, we first review conventional optimization-based IK algorithms under the minimal coordinates [7]. we then extend this idea to the maximal coordinates. Finally, we derive the corresponding gradient flow of each optimization algorithm.

A. Minimal Coordinates

The optimization problem under the minimal coordinates takes the following form:

$$\begin{aligned}\underset{\mathbf{x}^{min}}{\mathbf{argmin}}\ &\frac{1}{2}\|\mathbf{E}^{min}(\mathbf{x}^{min}) - \mathbf{e}\|^2\\ \mathbf{s.t.}\quad &\mathbf{L} \le \mathbf{x}^{min} \le \mathbf{U},\end{aligned} \tag{9}$$

where $\mathbf{e}$ is the desired position of the end-effector. This optimization can be solved using the Levenberg-Marquardt or damped least square method, but there is no guarantee that $\mathbf{E}^{min}(\mathbf{x}^{min}) = \mathbf{e}$ on convergence. We notice that Equation 9 can also be solved by formulating $\mathbf{x}^{min}$ as the configuration of a fictitious dynamic system and time-integrating the gradient flow:

$$\dot{\mathbf{x}}^{min} = \left[\frac{\partial\mathbf{E}^{min}}{\partial\mathbf{x}}\right]^T(\mathbf{e} - \mathbf{E}^{min}(\mathbf{x}^{min})). \tag{10}$$

As long as the joint limits are not violated during the time integration, Equation 10 will monotonically reduce the objective function and converge to a local minima. The trajectory generated by Equation 10 can be understood as

the convergence history of the least square algorithm when infinitesimal step sizes are taken.

B. Maximal Coordinates

We can extend optimization-based IK algorithms to the maximal coordinates $\mathbf{x}$. However, since $\mathbf{x}$ is not guaranteed to satisfy the rigidity and link connectivity constraints, we consider two different forms of optimization problems. First, we can formulate $\mathbf{C}_{eq}$ as soft constraints and solve the following problem:

$$\begin{aligned} \underset{\mathbf{x}}{\mathbf{argmin}}\ & \frac{1}{2}\|\mathbf{E}(\mathbf{x})-\mathbf{e}\|^2+\frac{1}{2}\|\mathbf{C}_{eq}(\mathbf{x})\|^2 \\ \mathbf{s.t.}\quad & \mathbf{C}_{neq}(\mathbf{x})\geq 0. \end{aligned} \tag{11}$$

Since $\mathbf{C}_{neq}$ are generally nonlinear in $\mathbf{x}$, typical algorithms that can solve Equation 11 involve augmented Lagrangian methods and interior point methods [22]. A major drawback of Equation 11 is that the optimization algorithm might not return a valid solution that satisfies $\mathbf{C}_{eq}(\mathbf{x}) = 0$. By comparison, solving Equation 9 will always result in a valid solution even if $\mathbf{E}^{min}(\mathbf{x}^{min}) \neq \mathbf{e}$. The gradient flow corresponding to Equation 11 is:

$$\dot{\mathbf{x}} = \frac{\partial \mathbf{E}}{\partial \mathbf{x}}^T(\mathbf{e}-\mathbf{E}(\mathbf{x}))-\frac{\partial \mathbf{C}_{eq}(\mathbf{x})}{\partial \mathbf{x}}^T\mathbf{C}_{eq}(\mathbf{x}). \tag{12}$$

A second method is to formulate $\mathbf{C}_{eq}$ as hard constraints and solve the following optimization problem:

$$\begin{aligned} \underset{\mathbf{x}}{\mathbf{argmin}}\ & \frac{1}{2}\|\mathbf{E}(\mathbf{x})-\mathbf{e}\|^2 \\ \mathbf{s.t.}\quad & \mathbf{C}_{eq}(\mathbf{x})=0\quad \mathbf{C}_{neq}(\mathbf{x})\geq 0. \end{aligned} \tag{13}$$

If the initial guess satisfies the polynomial constraints $\mathbf{C}_{eq}$ and $\mathbf{C}_{neq}$, Equation 13 will always return a valid solution in theory. However, if too large step sizes are taken by an optimizer, the solution might be stuck in an infeasible area with a rank-deficient constraint Jacobian, from which the optimizer will not be able to recover feasibility. To guarantee the satisfaction of $\mathbf{C}_{eq}$, we can take infinitesimal step sizes, resulting in the following projected gradient flow:

$$\begin{aligned} \dot{\mathbf{x}} &= \frac{\partial \mathbf{E}}{\partial \mathbf{x}}^T(\mathbf{e}-\mathbf{E}(\mathbf{x}))+\frac{\partial \mathbf{C}_{eq}(\mathbf{x})}{\partial \mathbf{x}}^T\lambda \\ \mathbf{s.t.}\quad \dot{\mathbf{C}}_{eq} &= \frac{\partial \mathbf{C}_{eq}(\mathbf{x})}{\partial \mathbf{x}}\dot{\mathbf{x}}=0, \end{aligned} \tag{14}$$

where λ is the Lagrangian multiplier that projects out any changes to $\mathbf{C}_{eq}$.

Remark 2: Both the gradient flow (Equation 12) and the projected gradient flow (Equation 14) take the form of Equation 1, where $\mathbf{x}$ can be interpreted as the state of a dynamic system. To see this for the projected gradient flow, we can compare Equation 14 and Equation 1, which immediately leads to the following linear system:

$$\begin{pmatrix} \mathbf{I} & \frac{\partial \mathbf{C}_{eq}(\mathbf{x})}{\partial \mathbf{x}}^T \\ \frac{\partial \mathbf{C}_{eq}(\mathbf{x})}{\partial \mathbf{x}} & 0 \end{pmatrix}\begin{pmatrix} \mathbf{f} \\ -\lambda \end{pmatrix}=\begin{pmatrix} \frac{\partial \mathbf{E}}{\partial \mathbf{x}}^T(\mathbf{e}-\mathbf{E}(\mathbf{x})) \\ 0 \end{pmatrix}.$$

This linear system is guaranteed to have a solution that defines $\mathbf{f}$, even when $\partial \mathbf{C}_{eq}(\mathbf{x})/\partial \mathbf{x}$ is rank-deficient.

V. Region Identification

In this section, we analyze the properties of the fictitious dynamic systems, Equation 12 and Equation 14, using sum-of-squares (SOS) programming. Similar to previous works [16], [37], we assume that $\bar{\mathcal{S}}_R$ and $\bar{\mathcal{S}}_M$ correspond to the sublevel sets of two polynomials P_M, P_R:

$$\bar{\mathcal{S}}_R = \{\mathbf{e}|P_R(\mathbf{e})\leq 1\}\quad \bar{\mathcal{S}}_M = \{\mathbf{x}|P_M(\mathbf{x})\leq 1\},$$

where the two polynomials $P_{M,R}$ are complete polynomials in $\mathbf{x}, \mathbf{e}$, whose coefficients are decision variables. We now show that the manipulability and reachability can be achieved if the following 4 conditions hold for a gradient flow:

1) The gradient flow starts from a valid initial guess.
2) One solution to the gradient flow will stay inside $\bar{\mathcal{S}}_M$.
3) Objective function is zero when the solution converges.
4) The joint limit will not be violated inside $\bar{\mathcal{S}}_M$.

Each of these conditions can be transformed into polynomial positivity constraints on some semi-algebraic set, for which the $\mathcal{S}$-procedure [24] can be used to derive a sufficient condition. We use condition 4 as an example.

Condition 4: To ensure that the joint limits will not be violated in $\bar{\mathcal{S}}_M$, we can define $\mathbf{F} = \varnothing$ and $\mathbf{G} = 1 - P_M(\mathbf{x})$ in Equation 2 and derive:

$$\begin{aligned} \mathbf{C}_{neq}(\mathbf{x})-L_M^{neq}(\mathbf{x})(1-P_M(\mathbf{x}))\in \text{SOS} \\ L_M^{neq}(\mathbf{x})\in \text{SOS}, \end{aligned} \tag{15}$$

where L_M^{neq} is the polynomial Lagrangian multipliers. In the following subsections, we certify the first 3 conditions first for Equation 12 and then for Equation 14.

A. Case with Soft Constraints

Condition 1: The gradient flow will reduce infeasibility by minimizing $\mathbf{C}_{eq}$, so it does not require the initial guess, denoted as $\mathbf{x}_{init}$, to satisfy $\mathbf{C}_{eq}(\mathbf{x}_{init}) = 0$. In other words, any $\mathbf{x}_{init}$ that falls inside $\bar{\mathcal{S}}_M$ is valid, i.e. the following constraint implies condition 1:

$$P_M(\mathbf{x}_{init})\leq 1. \tag{16}$$

Our formulation requires $\mathbf{x}_{init}$ to be a known constant. If $\mathbf{x}_{init}$ is unknown, we can treat $\mathbf{x}_{init}$ as additional decision variables, but the constraints of Equation 16 take a general nonlinear form, which requires constrained optimization with mixed nonlinear, LMI constraints. Optimization algorithms for this kind of problem are not well-studied.

Condition 2: We propose two forms of constraints, such that satisfying either one of these constraints will imply condition 2. First, since Equation 12 always reduces the objective function value, then the flow starting inside $\bar{\mathcal{S}}_M$ will never reach $\partial\bar{\mathcal{S}}_M$ if objective function values on $\partial\bar{\mathcal{S}}_M$ are always larger than that on $\mathbf{x}_{init}$. In other words, our sufficient constraint is:

$$\begin{aligned} &\frac{1}{2}\|\mathbf{E}(\mathbf{x})-\mathbf{e}\|^2+\frac{1}{2}\|\mathbf{C}_{eq}(\mathbf{x})\|^2 > \\ &\frac{1}{2}\|\mathbf{E}(\mathbf{x}_{init})-\mathbf{e}\|^2+\frac{1}{2}\|\mathbf{C}_{eq}(\mathbf{x}_{init})\|^2\quad \forall \mathbf{x}\in\partial\bar{\mathcal{S}}_M, \mathbf{e}\in\bar{\mathcal{S}}_R, \end{aligned}$$

which takes the following form by applying the S-procedure:

$$\begin{aligned} &\left[\frac{1}{2}\|\mathbf{E}(\mathbf{x})-\mathbf{e}\|^2+\frac{1}{2}\|\mathbf{C}_{eq}(\mathbf{x})\|^2\right]- \\ &\left[\frac{1}{2}\|\mathbf{E}(\mathbf{x}_{init})-\mathbf{e}\|^2+\frac{1}{2}\|\mathbf{C}_{eq}(\mathbf{x}_{init})\|^2\right]-\epsilon- \\ &L_M^{bd}(\mathbf{x},\mathbf{e})(1-P_M(\mathbf{x}))-L_R^{bd}(\mathbf{x},\mathbf{e})(1-P_R(\mathbf{e}))\in\text{SOS} \\ &L_R^{bd}(\mathbf{x},\mathbf{e})\in\text{SOS}, \end{aligned} \tag{17}$$

where $L_{M,R}^{bd}$ are the Lagrangian multipliers and ϵ is a small constant to ensure the strictness of inequality. Our second

constraint treats P_M as a Lyapunov function that restricts the trajectory in $\partial\bar{\mathcal{S}}_M$ via Lemma 2.1. Lemma 2.1 requires that $\mathbf{f}$ lies in the tangent cone of $\bar{\mathcal{S}}_M$ when $\mathbf{x}$ is on the boundary of $\bar{\mathcal{S}}_M$, i.e:

$$\frac{\partial P_M(\mathbf{x})}{\partial \mathbf{x}}\dot{\mathbf{x}} \leq 0 \quad \forall \mathbf{x} \in \partial\bar{\mathcal{S}}_M, \mathbf{e} \in \bar{\mathcal{S}}_R,$$

which takes the following form by applying the S-procedure:

$$\begin{aligned} &-\frac{\partial P_M(\mathbf{x})}{\partial \mathbf{x}}\left[\frac{\partial \mathbf{E}}{\partial \mathbf{x}}^T(\mathbf{e}-\mathbf{E}(\mathbf{x})) - \frac{\partial \mathbf{C}_{eq}(\mathbf{x})}{\partial \mathbf{x}}^T \mathbf{C}_{eq}(\mathbf{x})\right] - \\ &L_M^{bd}(\mathbf{x},\mathbf{e})(1-P_M(\mathbf{x})) - L_R^{bd}(\mathbf{x},\mathbf{e})(1-P_R(\mathbf{e})) \in \text{SOS} \\ &L_R^{bd}(\mathbf{x},\mathbf{e}) \in \text{SOS}. \end{aligned} \tag{18}$$

Condition 3: We need to show that the gradient flow always converges to zero. In other words, if the current objective function is not zero, there is always a local direction to reduce it, for which a sufficient condition is for the following Jacobian matrix to have a full rank:

$$\mathbf{J}(\mathbf{x}) = \left(\frac{\partial \mathbf{E}(\mathbf{x})}{\partial \mathbf{x}}^T \frac{\partial \mathbf{C}_{eq}(\mathbf{x})}{\partial \mathbf{x}}^T\right)^T \quad \forall \mathbf{x} \in \bar{\mathcal{S}}_M.$$

This is a polynomial matrix inequality (PMI) constraint, which takes the following form by applying the S-procedure:

$$\begin{aligned} &\mathbf{y}^T(\mathbf{J}\mathbf{J}^T - \epsilon\mathbf{I})\mathbf{y} - L_M^{conv}(\mathbf{x},\mathbf{y})(1-P_M(\mathbf{x})) \in \text{SOS} \\ &L_M^{conv}(\mathbf{x},\mathbf{y}) \in \text{SOS}, \end{aligned} \tag{19}$$

where $\mathbf{y}$ are auxiliary polynomial variables and L_M^{conv} are the Lagrangian multipliers. We add a small constant ϵ to ensure the strictness of the rank condition.

We have transformed the 3 properties of the gradient flow in Equation 12 and we summarize our main results below:

***Lemma** 5.1:* If the following optimization problem is feasible, then $\bar{\mathcal{S}}_R \subseteq \mathcal{S}_R \cap \mathbf{E}(\mathcal{S}_M)$:

$$\underset{P_M, P_R, L_M^{neq}, L_M^{bd}, L_R^{bd}, L_M^{conv}}{\textbf{argmin}} \quad 0$$

$\mathbf{s.t.}$ Equation 15,16,17,19 or Equation 15,16,18,19.

Proof: **Reachability:** Consider the trajectory generated by time-integrating gradient flow of Equation 12 and let $\mathbf{x}(t=0) = \mathbf{x}_{init}$ so $\mathbf{x}(t=0) \in \bar{\mathcal{S}}_M$ by Equation 16. The solution $\mathbf{x}(t)$ will monotonically reduce the objective function value, so $\mathbf{x}(t) \in \bar{\mathcal{S}}_M$ if Equation 17 is satisfied. Or if Equation 18 is satisfied, then we have $\mathbf{x}(t) \in \bar{\mathcal{S}}_M$ for at least one solution due to Lemma 2.1. Let $t \to \infty$, we have:

$$0 = \|\dot{\mathbf{x}}\|^2 = \|\mathbf{J}^T\begin{pmatrix}\mathbf{e}-\mathbf{E}(\mathbf{x}) \\ \mathbf{C}_{eq}(\mathbf{x})\end{pmatrix}\|^2 \geq \|\begin{pmatrix}\mathbf{e}-\mathbf{E}(\mathbf{x}) \\ \mathbf{C}_{eq}(\mathbf{x})\end{pmatrix}\|^2\epsilon,$$

due to Equation 19. Therefore, the trajectory will converge to the zero-level set of the objective function and will not violate $\mathbf{C}_{neq}$ (due to Equation 15). Therefore, for every point $\mathbf{e} \in \bar{\mathcal{S}}_R$, there exists a gradient flow that converges to a valid robot configuration with $\mathbf{e} = \mathbf{E}(\mathbf{x})$, so $\bar{\mathcal{S}}_R$ is reachable. **Manipulability:** For any $\Delta\mathbf{e}$, we can find $\Delta\mathbf{x}$ by solving:

$$\begin{pmatrix}\Delta\mathbf{e}^T & 0^T\end{pmatrix}^T = \mathbf{J}(\mathbf{x}(\infty))\Delta\mathbf{x},$$

which must be solvable as $\mathbf{J}(\mathbf{x}(\infty))$ has a full row-rank. The resulting $\Delta\mathbf{x}$ satisfies the condition of $\mathcal{S}_M$ showing that $\bar{\mathcal{S}}_R \subseteq \mathbf{E}(\mathcal{S}_M)$. ∎

B. Case with Hard Constraints

Condition 1: The projected gradient flow needs to start from an initial guess that satisfy $\mathbf{C}_{eq}$ and $\mathbf{C}_{neq}$. Therefore, a valid initial guess needs to satisfy the following constraints:

$$\begin{aligned} &P_M(\mathbf{x}_{init}) \leq 1 \\ &\mathbf{C}_{eq}(\mathbf{x}_{init}) = 0 \quad \mathbf{C}_{neq}(\mathbf{x}_{init}) \geq 0, \end{aligned} \tag{20}$$

which imply condition 1.

Condition 2: We derive two constraints similar to those in Section V-A, where the difference is that we can further restrict the semi-algebraic set by adding the equality $\mathbf{C}_{eq} = 0$ to $\mathbf{F}$. Adding more equations to the semi-algebraic set will introduce more decision variables to the SOS constraints and result in a larger feasible domain. As the first sufficient constraint, we can provide guarantee that objective function values on $\partial\bar{\mathcal{S}}_M$ are always larger than that on $\mathbf{x}_{init}$:

$$\begin{aligned} &\left[\frac{1}{2}\|\mathbf{E}(\mathbf{x})-\mathbf{e}\|^2\right] - \left[\frac{1}{2}\|\mathbf{E}(\mathbf{x}_{init})-\mathbf{e}\|^2\right] - \epsilon - \\ &L_{\mathbf{C}}^{bd}(\mathbf{x},\mathbf{e})\mathbf{C}_{eq}(\mathbf{x}) - \\ &L_M^{bd}(\mathbf{x},\mathbf{e})(1-P_M(\mathbf{x})) - L_R^{bd}(\mathbf{x},\mathbf{e})(1-P_R(\mathbf{e})) \in \text{SOS} \\ &L_R^{bd}(\mathbf{x},\mathbf{e}) \in \text{SOS}, \end{aligned} \tag{21}$$

where $L_{\mathbf{C}}^{bd}$ is the additional polynomial Lagrangian multiplier to restrict $\mathbf{C}_{eq} = 0$. As the second sufficient constraint, we treat P_M as a Lyapunov function that restricts the projected gradient flow in $\bar{\mathcal{S}}_M$ using Lemma 2.1. However, we can further restrict the semi-algebraic set using the identity $\dot{\mathbf{C}}_{eq} = 0$, leading to the combined sufficient condition:

$$\frac{\partial P_M(\mathbf{x})}{\partial \mathbf{x}}\dot{\mathbf{x}} \leq 0 \quad \forall \dot{\mathbf{C}}_{eq} = 0, \mathbf{x} \in \partial\bar{\mathcal{S}}_M, \mathbf{e} \in \bar{\mathcal{S}}_R,$$

which takes the following form after applying the S-procedure:

$$\begin{aligned} &-\frac{\partial P_M(\mathbf{x})}{\partial \mathbf{x}}\left[\frac{\partial \mathbf{E}}{\partial \mathbf{x}}^T(\mathbf{e}-\mathbf{E}(\mathbf{x})) + \frac{\partial \mathbf{C}_{eq}(\mathbf{x})}{\partial \mathbf{x}}^T\lambda\right] - \\ &L_\lambda^{bd}(\mathbf{x},\mathbf{e},\lambda)\frac{\partial \mathbf{C}_{eq}(\mathbf{x})}{\partial \mathbf{x}}\left[\frac{\partial \mathbf{E}}{\partial \mathbf{x}}^T(\mathbf{e}-\mathbf{E}(\mathbf{x})) + \frac{\partial \mathbf{C}_{eq}(\mathbf{x})}{\partial \mathbf{x}}^T\lambda\right] - \\ &L_{\mathbf{C}}^{bd}(\mathbf{x},\mathbf{e},\lambda)\mathbf{C}_{eq}(\mathbf{x}) - L_M^{bd}(\mathbf{x},\mathbf{e},\lambda)(1-P_M(\mathbf{x})) - \\ &L_R^{bd}(\mathbf{x},\mathbf{e},\lambda)(1-P_R(\mathbf{e})) \in \text{SOS} \\ &L_R^{bd}(\mathbf{x},\mathbf{e},\lambda) \in \text{SOS}, \end{aligned} \tag{22}$$

where $L_{\lambda,\mathbf{C}}^{bd}$ are additional polynomial Lagrangian multipliers to restrict $\dot{\mathbf{C}}_{eq} = 0, \mathbf{C}_{eq} = 0$ and λ are treated as auxiliary polynomial variables.

Condition 3: We have to ensure that there is always a direction along which we can reduce the objective function value and that the direction is not orthogonal to the tangent bundle of the constrained manifold of $\mathbf{C}_{eq}$. A sufficient condition for this argument is that the direction after projection has a length of at least ϵ of its original length. Mathematically, this constraint is:

$$\|\dot{\mathbf{x}}\|^2 \geq \epsilon\|\mathbf{e}-\mathbf{E}(\mathbf{x})\|^2 \quad \forall \dot{\mathbf{C}}_{eq} = 0, \mathbf{x} \in \partial\bar{\mathcal{S}}_M, \mathbf{e} \in \bar{\mathcal{S}}_R,$$

which takes the following form after applying the S-procedure:

$$\begin{aligned} &\|\frac{\partial \mathbf{E}}{\partial \mathbf{x}}^T(\mathbf{e}-\mathbf{E}(\mathbf{x})) + \frac{\partial \mathbf{C}_{eq}(\mathbf{x})}{\partial \mathbf{x}}^T\lambda\|^2 - \epsilon\|\mathbf{e}-\mathbf{E}(\mathbf{x})\|^2 - \\ &L_\lambda^{conv}(\mathbf{x},\mathbf{e},\lambda)\frac{\partial \mathbf{C}_{eq}(\mathbf{x})}{\partial \mathbf{x}}\left[\frac{\partial \mathbf{E}}{\partial \mathbf{x}}^T(\mathbf{e}-\mathbf{E}(\mathbf{x})) + \frac{\partial \mathbf{C}_{eq}(\mathbf{x})}{\partial \mathbf{x}}^T\lambda\right] - \\ &L_{\mathbf{C}}^{conv}(\mathbf{x},\mathbf{e},\lambda)\mathbf{C}_{eq}(\mathbf{x}) - L_M^{conv}(\mathbf{x},\mathbf{e},\lambda)(1-P_M(\mathbf{x})) - \\ &L_R^{conv}(\mathbf{x},\mathbf{e},\lambda)(1-P_R(\mathbf{e})) \in \text{SOS} \\ &L_M^{conv}(\mathbf{x},\mathbf{e},\lambda), L_R^{conv}(\mathbf{x},\mathbf{e},\lambda) \in \text{SOS}, \end{aligned} \tag{23}$$

where $L_{\lambda,\mathbf{C}}^{conv}$ are additional polynomial Lagrangian multipliers. Below, we summarize our main results for the projected gradient flow.

***Lemma** 5.2:* If the following optimization problem is feasible, then $\bar{\mathcal{S}}_R \subseteq \mathcal{S}_R$:

$$\begin{aligned} &\underset{\substack{P_M, P_R, L_M^{neq}, L_M^{bd}, L_R^{bd}, L_{\mathbf{C}}^{bd}, L_\lambda^{bd}, \\ L_M^{conv}, L_R^{conv}, L_{\mathbf{C}}^{conv}, L_\lambda^{conv}}}{\mathbf{argmin}} \quad 0 \\ &\mathbf{s.t.} \quad \text{Equation 15,20,21,23 or Equation 15,20,22,23.} \end{aligned}$$

Proof: **Reachability:** Consider the trajectory generated by time-integrating gradient flow of Equation 14 and let $\mathbf{x}(t=0) = \mathbf{x}_{init}$ so $\mathbf{x}(t=0) \in \bar{\mathcal{S}}_M$ by Equation 16 and $\dot{\mathbf{C}}_{eq}(\mathbf{x}(t)) = \mathbf{C}_{eq}(\mathbf{x}(t)) = 0$ for all $t \geq 0$ by the definition of the projected gradient flow. The solution $\mathbf{x}(t)$ will monotonically reduce the objective function value, so $\mathbf{x}(t) \in \bar{\mathcal{S}}_M$ if Equation 21 is satisfied. Or if Equation 22 is satisfied, then we have $\mathbf{x}(t) \in \bar{\mathcal{S}}_M$ for at least one solution due to Lemma 2.1. Let $t \to \infty$, we have $0 = \|\mathbf{f}\|^2 \geq \|\mathbf{e} - \mathbf{E}(\mathbf{x})\|^2 \epsilon$ due to Equation 23. Therefore, the trajectory will converge to the zero-level set of the objective function and will not violate $\mathbf{C}_{neq}$ (due to Equation 15) or $\mathbf{C}_{eq}$. ■

Remark 3: As compared with the gradient flow, the projected gradient flow leads to more relaxed conditions to identify a reachable subset. In the gradient flow, we assume that $\mathbf{J}$ has a full-rank, which is used to pull an infeasible $\mathbf{x}$ back to the constraint manifold. This is not needed in the projected gradient flow as $\mathbf{x}$ never leaves the constraint manifold. This relaxed condition potentially allows the identification of a larger reachable set. On the other hand, the full-rank condition in the gradient flow always leads to the identification of the reachable and the manipulable subset simultaneously. For the projected gradient flow, we can only prove reachability but not manipulability in Lemma 5.2.

C. Applications

If $\mathbf{x}_{init}$ is known, the optimization problems defined in Lemma 5.1 and Lemma 5.2 are under BMI constraints that are linear in both $P_{M,R}$ and polynomial Lagrangian multipliers $L^{\bullet}_{M,R}$. As a typical application, users want to verify whether the known $P_{M,R}$ are strict inner-approximations of the manipulable and reachable sets. In this case, $P_{M,R}$ are fixed and BMI constraints are further reduced to LMI constraints. Feasibility of LMI constraints can be verified in polynomial time using interior point methods. As another application, users want to find $P_{M,R}$ that are as-large-as-possible. In this case, we follow [1] and approximate the reciprocal of volume of $\bar{\mathcal{S}}_R$ using the trace of SOS matrix. In other words, if $P_R(\mathbf{x})$ is an SOS polynomial, then $P_R(\mathbf{x}) = \mathcal{M}(\mathbf{x})^T \mathbf{H} \mathcal{M}(\mathbf{x})$ and the volume of $\bar{\mathcal{S}}_R$ can then be approximated by $\mathbf{Tr}(\mathbf{H})$. A common, additional requirement from users is that the region $\bar{\mathcal{S}}_R$ is convex, which can be further restricted to SOS-convexity. This is because verifying function convexity cannot be performed using SOS programming, but verifying SOS-convexity can be transformed into a PSD-cone [1]. In summary, to maximize the identified region, we solve the following optimization for the gradient flow:

$$\begin{aligned} &\underset{P_M, P_R, L_M^{neq}, L_M^{bd}, L_R^{bd}, L_M^{conv}, L_R^{conv}}{\mathbf{argmin}} \quad \mathbf{Tr}(\mathbf{H}) \\ &\mathbf{s.t.} \quad \text{Equation 15,16,17,19 or Equation 15,16,18,19} \\ &\qquad P_R(\mathbf{x}) \in \text{SOS} \cap \text{SOS-convex}. \end{aligned} \tag{24}$$

And we solve the following optimization for the projected gradient flow:

$$\begin{aligned} &\underset{\substack{P_M, P_R, L_M^{neq}, L_M^{bd}, L_R^{bd}, L_{\mathbf{C}}^{bd}, L_\lambda^{bd}, \\ L_M^{conv}, L_R^{conv}, L_{\mathbf{C}}^{conv}, L_\lambda^{conv}}}{\mathbf{argmin}} \quad \mathbf{Tr}(\mathbf{H}) \\ &\mathbf{s.t.} \quad \text{Equation 15,20,21,23 or Equation 15,20,22,23} \\ &\qquad P_R(\mathbf{x}) \in \text{SOS} \cap \text{SOS-convex}. \end{aligned} \tag{25}$$

We can only find local minima for BMI-constrained optimizations. Similar to [16], we use an alternating direction solver. Within each outer loop, we first fix $P_{M,R}$ and optimize $L^{\bullet}_{M,R}$, which is a LMI-constrained convex optimization. We then fix $L^{\bullet}_{M,R}$ and optimize $P_{M,R}$, which is another LMI-constrained convex optimization. The outer loop terminates when the relative function value change is smaller than a threshold.

VI. Results and Analysis

We implement our algorithm using C++ and solve all the LMI-constrained optimization problems using Mosek [21] on a desktop machine with 16GB memory and Intel Xeon W-2155 processor. We use polynomials of degree $d = 2$ to relax all the SOS constraints. In other words, if some polynomial $P(\mathbf{x}) \in \text{SOS}$, then a degree d relaxation is denoted as $P(\mathbf{x}) \in \text{SOS}_d$, which dictates that $P(\mathbf{x}) = \mathcal{M}_d(\mathbf{x}) \mathbf{H}_d \mathcal{M}_d(\mathbf{x})$. Here $\mathcal{M}_d$ is a vector of monomials in $\mathbf{x}$ up to degree d and $\mathbf{H}_d$ is the matrix of decision variables. Note that $d = 2$ is enough for relaxing all the SOS constraints, because every polynomial optimization problem (POP) can be reduced to an equivalent quadratic-constrained optimization problem (QCQP) by introducing auxiliary variables [19]. Below, we illustrate our method on different simplified robot models and highlight the influence of different factors and parameters. We always use Equation 24 to maximize the regions with Equation 18 for condition 2, unless otherwise stated.

As our first evaluation, we use a 2D robot arm with 2 links connected by 2 hinge joints, both with non-trivial joint limits, in which case we have $|\mathbf{x}^{min}| = 2$, $|\mathbf{x}| = 8$, $|\mathbf{C}_{eq}(\mathbf{x})| = 6$, $|\mathbf{C}_{neq}(\mathbf{x})| = 2$, and $|\mathbf{e}| = 2$. We solve Equation 24 from an initial guess $\mathbf{x}_{init}$ that is derived by randomly sampling a minimal coordinates vector $\mathbf{x}_{min}$ within the joint limits and then converting to the maximal coordinates. The polynomials $P_{M,R}$ are initialized to small unit balls around $\mathbf{x}_{init}$ and $\mathbf{E}(\mathbf{x}_{init})$, respectively. The convergence history of the outer loop of the alternating direction solver is illustrated in Figure 2. In all examples, the outer loop converges within 20 iterations (each involving 2 LMI-constrained optimizations). On convergence, the identified reachable subset $\bar{\mathcal{S}}_R$ is locally maximized, but the region is small compared with the true reachable set $\mathcal{S}_R$. To identify larger subsets of $\mathcal{S}_R$, we can randomly sample multiple $\mathbf{x}_{init}$, solve for $\bar{\mathcal{S}}_R$ around each $\mathbf{x}_{init}$, and then take union of all $\bar{\mathcal{S}}_R$. As illustrated in Figure 3, we can identify a large subset of $\mathcal{S}_R$ by randomly sampling 260 initial guess. The computational time of solving Equation 24 for 260 times is 25200 seconds.

In our second evaluation, we use 2D robot arm with 2 links connected by 1 hinge joint and 1 prismatic joint, both with non-trivial joint limits, in which case we have $|\mathbf{x}^{min}| = 2$,

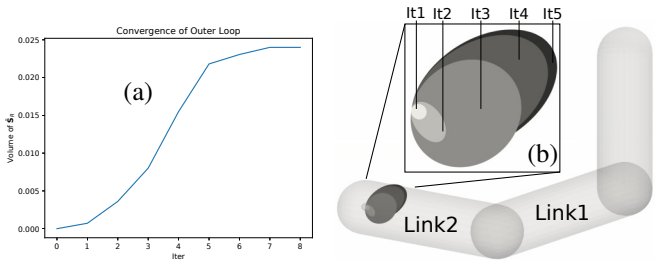

Fig. 2: (a): Convergence of the outer loop of the alternating direction solver. (b): The identified region $\bar{\mathcal{S}}_R$ of each iteration.

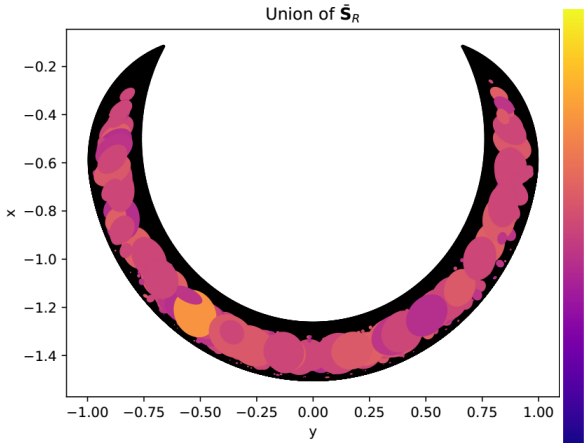

Fig. 3: For a 2-link robot with 2 hinge joints, we show the true reachable set $\mathcal{S}_R$ (black) and the 260 randomly sampled $\bar{\mathcal{S}}_R$ (color coded by size of each $\bar{\mathcal{S}}_R$).

$|\mathbf{x}| = 8$, $|\mathbf{C}_{eq}(\mathbf{x})| = 7$, $|\mathbf{C}_{neq}(\mathbf{x})| = 3$, and $|\mathbf{e}| = 2$. Again, we solve Equation 24 for 260 times and take union of all $\bar{\mathcal{S}}_R$, which takes 54100 seconds of computational time. The result is illustrated in Figure 4.

In our third evaluation, we compare the formulation based on gradient flow (Equation 24 with Equation 18 for condition 2) and the formulation based on projected gradient flow (Equation 25 with Equation 22 for condition 2). We solve for $\bar{\mathcal{S}}_R$ 260 times using both formulations and profile the distribution of identified subset size in Figure 5. Some of the subsets identified using Equation 25 has larger volumes, but the two distributions of volumes are comparable on average. However, the total computational time using Equation 24 is 25200 seconds and the computational time using Equation 25 is longer than 100000 seconds, i.e. computational cost of Equation 25 is at least $4\times$ higher. This is because we need to introduce λ as additional polynomial variables in Equation 25, leading to much larger optimization problems.

Finally, we compare the two different constraints for condition 2 in both formulations. As illustrated in Figure 6, we solve Equation 24 first with Equation 17 and then with Equation 18 and compare the volume of identified $\bar{\mathcal{S}}_R$. Clearly, the constraint based on Lemma 2.1 (Equation 18,22) generates larger volumes than the constraint based on objective function values (Equation 17,21).

VII. Conclusion and Limitations

Our work applies the sum-of-squares programming technique to identify the convergent region of an optimization algorithm. We apply this technique to analyze the optimization-based IK solver and show that the identified convergent regions correspond to manipulable and reachable sets of an articulated robot arm. Our method can be applied to both soft-constrained and hard-constrained problems, and the resulting BMI problem allows users to either certify a fixed region or to maximize the volume of an unknown region from an initial guess. We have evaluated our method on 2D robot arms and we show that manipulable/reachable regions of non-trivial sizes can be identified using low-order polynomial relaxations ($d = 2$), in both soft-constrained and hard-constrained cases. Finally, we highlight that, although we apply our method to analyze IK algorithms, our method is more general and can be applied to any optimization algorithms, as long as the functions $\mathbf{C}_{eq}, \mathbf{C}_{neq}, \mathbf{E}$ take polynomial forms.

Our major limitation inherited from general SOS programming is that the method only handles a small number of polynomial variables. Indeed, the size of PSD-cone constraints for a problem with N polynomial variables grow like $\binom{N+d}{d}$. To make computation practical on a desktop machine, our method is restricted to robots with at most 2 links. A potential method to reduce computational cost is to exploit sparsity in polynomial variables [35]. In robot arms without closed loops, the dependency between polynomial variables for different links has banded sparse pattern, which

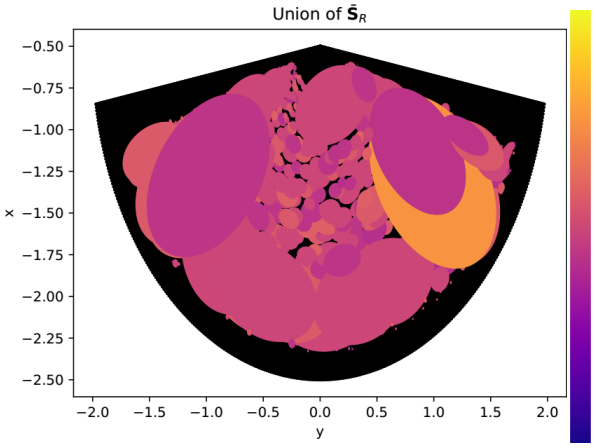

Fig. 4: For a 2-link robot with 1 hinge and 1 prismatic joint, we show the true reachable set $\mathcal{S}_R$ (black) and the 260 randomly sampled $\bar{\mathcal{S}}_R$ (color coded by size of each $\bar{\mathcal{S}}_R$).

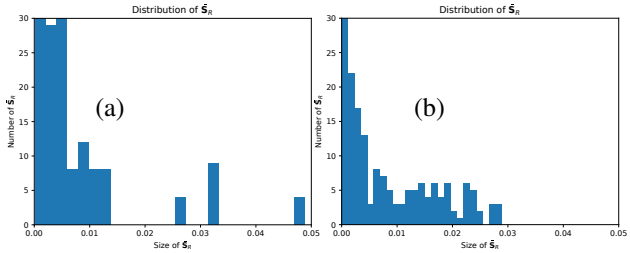

Fig. 5: We compare $\bar{\mathcal{S}}_R$ identified using Equation 25 (a) and Equation 24 (b). Some of the $\bar{\mathcal{S}}_R$ identified using Equation 25 has larger volumes (around 0.05), while the maximal volume of $\bar{\mathcal{S}}_R$ under Equation 24 is only 0.03. However, the computational cost of Equation 25 is $4\times$ higher than Equation 24 at degree $d = 2$.

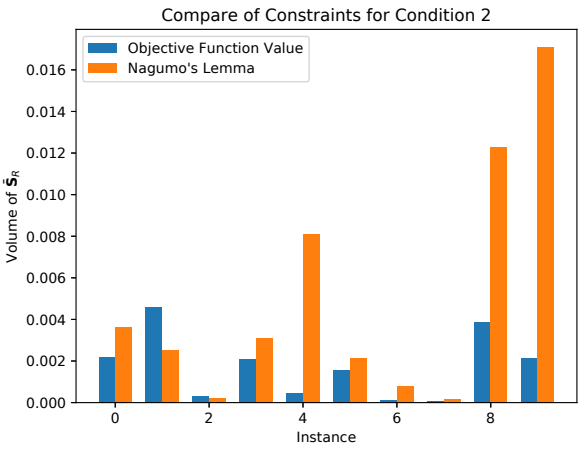

Fig. 6: We compare the two constraints that imply condition 2. We solve Equation 25 for 10 times from same $\mathbf{x}_{init}$ using Equation 18,22 and compare the volume of identified $\bar{S}_R$.

is benign to sparse SOS formulations. Another method to reduce cost is to use other theorems of Positivstellensatz [12], [38], [2]. However, some alternative theorems [12], [38] lead to more general, multi-linear problems, for which more complex, local optimization techniques are required.

As a second limitation, we restrict the notion of reachability to the set of end-effector positions that can be reached by local optimization from a common initial guess. As a result, we can only find a small, reachable subset around a nominal solution $\mathbf{x}_{init}$. However, we have shown that we can identify larger subsets by randomly sampling nominal solutions, identifying many local reachable subsets around each solution and finally taking union of these subsets. This technique is similar to previous work [32]. Another avenue for future work is to analyze non-optimization-based IK algorithms such as the algebraic algorithm [28].

REFERENCES

[1] A. A. Ahmadi, G. Hall, A. Makadia, and V. Sindhwani, "Geometry of 3d environments and sum of squares polynomials," in *2017 Robotics: Science and Systems, RSS 2017*. MIT Press Journals, 2017.

[2] A. A. Ahmadi and A. Majumdar, "Dsos and sdsos optimization: more tractable alternatives to sum of squares and semidefinite optimization," *SIAM Journal on Applied Algebra and Geometry*, vol. 3, no. 2, pp. 193–230, 2019.

[3] J.-P. Aubin, *Viability Theorems for Ordinary and Stochastic Differential Equations*. Boston, MA: Birkhäuser Boston, 2009, pp. 19–52.

[4] S. Baldi and W. Xiang, "Reachable set estimation for switched linear systems with dwell-time switching," *Nonlinear Analysis: Hybrid Systems*, vol. 29, pp. 20–33, 2018.

[5] D. Baraff, "Linear-time dynamics using lagrange multipliers," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 1996, pp. 137–146.

[6] R. R. Burridge, A. A. Rizzi, and D. E. Koditschek, "Sequential composition of dynamically dexterous robot behaviors," *The International Journal of Robotics Research*, vol. 18, no. 6, pp. 534–555, 1999.

[7] S. R. Buss and J.-S. Kim, "Selectively damped least squares for inverse kinematics," *Journal of Graphics tools*, vol. 10, no. 3, pp. 37–49, 2005.

[8] A. El-Guindy, D. Han, and M. Althoff, "Estimating the region of attraction via forward reachable sets," in *2017 American Control Conference (ACC)*. IEEE, 2017, pp. 1263–1270.

[9] E. Fridman and U. Shaked, "On reachable sets for linear systems with delay and bounded peak inputs," *Automatica*, vol. 39, no. 11, pp. 2005–2010, 2003.

[10] K. Hauser and S. Emmons, "Global redundancy resolution via continuous pseudoinversion of the forward kinematic map," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 3, pp. 932–944, July 2018.

[11] I. Hwang, D. M. Stipanović, and C. J. Tomlin, "Polytopic approximations of reachable sets applied to linear dynamic games and a class of nonlinear systems," in *Advances in control, communication networks, and transportation systems*. Springer, 2005, pp. 3–19.

[12] J. B. Lasserre, K.-C. Toh, and S. Yang, "A bounded degree sos hierarchy for polynomial optimization," *EURO Journal on Computational Optimization*, vol. 5, no. 1-2, pp. 87–117, 2017.

[13] M. Laurent, "Sums of squares, moment matrices and optimization over polynomials," in *Emerging applications of algebraic geometry*. Springer, 2009, pp. 157–270.

[14] G. G. Lendaris, K. Mathia, and R. Saeks, "Linear hopfield networks and constrained optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 29, no. 1, pp. 114–118, Feb 1999.

[15] A. Majumdar, A. A. Ahmadi, and R. Tedrake, "Control design along trajectories with sums of squares programming," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 4054–4061.

[16] A. Majumdar and R. Tedrake, "Funnel libraries for real-time robust feedback motion planning," *The International Journal of Robotics Research*, vol. 36, no. 8, pp. 947–982, 2017.

[17] O. Maler, "Computing reachable sets: An introduction," *Tech. rep. French National Center of Scientific Research*, 2008.

[18] D. Manocha and J. F. Canny, "Efficient inverse kinematics for general 6r manipulators," *IEEE transactions on robotics and automation*, vol. 10, no. 5, pp. 648–657, 1994.

[19] M. Mevissen and M. Kojima, "Sdp relaxations for quadratic optimization problems derived from polynomial optimization problems," *Asia-Pacific Journal of Operational Research*, vol. 27, no. 01, pp. 15–38, 2010.

[20] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin, "A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games," *IEEE Transactions on automatic control*, vol. 50, no. 7, pp. 947–957, 2005.

[21] A. Mosek, "The mosek optimization software," *Online at http://www. mosek. com*, vol. 54, no. 2-1, p. 5, 2010.

[22] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.

[23] E. Oyama, N. Y. Chong, A. Agah, and T. Maeda, "Inverse kinematics learning by modular architecture neural networks with performance prediction networks," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, vol. 1. IEEE, 2001, pp. 1006–1012.

[24] P. A. Parrilo, "Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization," Ph.D. dissertation, California Institute of Technology, 2000.

[25] B. Polyak and P. Shcherbakov, "Lyapunov functions: An optimization theory perspective**this work was supported by the russian scientific foundation, project no. 16-11-10015," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 7456 – 7461, 2017, 20th IFAC World Congress. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405896317320955

[26] O. Porges, T. Stouraitis, C. Borst, and M. A. Roa, "Reachability and capability analysis for manipulation tasks," in *ROBOT2013: First Iberian robotics conference*. Springer, 2014, pp. 703–718.

[27] S. Seereeram and J. T. Wen, "A global approach to path planning for redundant manipulators," *IEEE Transactions on Robotics and Automation*, vol. 11, no. 1, pp. 152–160, 1995.

[28] S. Stifter, "Algebraic methods for computing inverse kinematics," *Journal of Intelligent and Robotic Systems*, vol. 11, no. 1-2, pp. 79–89, 1994.

[29] D. M. Stipanović, I. Hwang, and C. J. Tomlin, "Computation of an over-approximation of the backward reachable set using subsystem level set functions," in *2003 European Control Conference (ECC)*. IEEE, 2003, pp. 300–305.

[30] W. Tan and A. Packard, "Stability region analysis using sum of squares programming," in *2006 American Control Conference*. IEEE, 2006, pp. 6–pp.

[31] A. Taylor, B. Van Scoy, and L. Lessard, "Lyapunov functions for first-order methods: Tight automated convergence guarantees," in *International Conference on Machine Learning*, 2018, pp. 4897–4906.

[32] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, "Lqr-trees: Feedback motion planning via sums-of-squares verification," *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 1038–1052, 2010.

[33] N. Vahrenkamp and T. Asfour, "Representing the robot's workspace through constrained manipulability analysis," *Autonomous Robots*, vol. 38, no. 1, pp. 17–30, 2015.

[34] N. Vahrenkamp, T. Asfour, G. Metta, G. Sandini, and R. Dillmann, "Manipulability analysis," in *2012 12th ieee-ras international conference on humanoid robots (humanoids 2012)*. IEEE, 2012, pp. 568–573.

[35] H. Waki, S. Kim, M. Kojima, and M. Muramatsu, "Sums of squares and semidefinite program relaxations for polynomial optimization problems with structured sparsity," *SIAM Journal on Optimization*, vol. 17, no. 1, pp. 218–242, 2006.

[36] L.-C. Wang and C.-C. Chen, "A combined optimization method for solving the inverse kinematics problems of mechanical manipulators," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 4, pp. 489–499, 1991.

[37] T.-C. Wang, S. Lall, and M. West, "Polynomial level-set method for polynomial system reachable set estimation," *IEEE Transactions on Automatic Control*, vol. 58, no. 10, pp. 2508–2521, 2013.

[38] T. Weisser, J. B. Lasserre, and K.-C. Toh, "Sparse-bsos: a bounded degree sos hierarchy for large scale polynomial optimization with sparsity," *Mathematical Programming Computation*, vol. 10, no. 1, pp. 1–32, 2018.

[39] D. E. Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Transactions on man-machine systems*, vol. 10, no. 2, pp. 47–53, 1969.

[40] A. C. Wilson, B. Recht, and M. I. Jordan, "A lyapunov analysis of momentum methods in optimization," *arXiv*, pp. arXiv–1611, 2016.

[41] W. A. Wolovich and H. Elliott, "A computational technique for inverse kinematics," in *The 23rd IEEE Conference on Decision and Control*. IEEE, 1984, pp. 1359–1363.

[42] B. Xue, M. Fränzle, and N. Zhan, "Inner-approximating reachable sets for polynomial systems with time-varying uncertainties," *IEEE Transactions on Automatic Control*, 2019.

[43] H. Yin, A. Packard, M. Arcak, and P. Seiler, "Finite horizon backward reachability analysis and control synthesis for uncertain nonlinear systems," in *2019 American Control Conference (ACC)*. IEEE, 2019, pp. 5020–5026.

[44] T. Yoshikawa, "Manipulability of robotic mechanisms," *The international journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, 1985.