An SDP Optimization Formulation for the Inverse Kinematics Problem

Liangting Wu and Roberto Tron

Abstract— **Inverse kinematics (IK) is an important problem in robot control and motion planning; however, the nonlinearity of the map from joint angles to robot configurations makes the problem nonconvex. In this paper, we propose an inverse kinematics solver that works in the space of rotation matrices of the link reference frames rather than joint angles. To overcome the nonlinearity of the manifold of rotation matrices $\mathbf{SO}(3)$, we propose a semidefinite programming (SDP) relaxation of the kinematic constraints followed by a fixed-trace rank minimization via maximization of a convex function. Along the way, we show that the feasible set of an IK problem is exactly the intersection of a convex set and fixed-trace rank-1 matrices. Thanks to the use of matrices with fixed trace, our algorithm to obtain rank-1 solutions has guaranteed local convergence. Unlike some traditional solvers, our method does not require an initial guess, and can be applied to robots with closed kinematic chains without ad-hoc modifications such as splitting the kinematic chain. Compared to other work that performs SDP relaxation for IK problems, our formulation is simpler, and uses variables with smaller sizes. We validate our approach via simulations on a closed kinematic chain constituted by two robotic arms holding a box, comparing against a standard IK method.**

I. Introduction

Robot manipulators play a pervasive role in fields such as manufacturing, education, medicine, and aerospace. A fundamental problem for robots in these settings is inverse kinematics (IK) [19], where one needs to determine the values of the joint configurations that result in a given desired position and orientation of the end-effector.

Despite its importance, solving such problem is difficult for multiple reasons: *1)* the kinematic map from joint angles to end-effector poses is generally nonlinear; *2)* different numbers of solutions (zero, multiple distinct, or infinite) may exist depending on the structure of the robot and the query pose; *3)* typically there are nonlinear equality and nonlinear constraints deriving from joint limits, self-intersection constraints, and closed kinematic chains. Previous work has shown that a finite number of analytical solutions for manipulators with up to 6-DOF exist [10], and can be derived in algebraic form [7], [17]. The popular solver IKFast [5] generalizes this method and automatically computes IK solutions in closed form. However, analytical methods are generally unavailable for robots with higher DOFs. On the other hand, numerical methods have been successful in solving the IK problem, producing numerous efficient inverse-kinematics solvers such as CCD [8], triangulation [15] and FABRIK [1]. These solvers often perturb joint angles iteratively to decrease a distance between the end-effector and the target. Despite their efficiency, kinematic constraints such as self-collision, multiple end-effectors, and closed chains are either ignored or require ad-hoc modifications.

Other approaches to IK are based on formulations as nonlinear optimization problems which are then solved numerically, such as in [3], [9]. The main weakness of these methods is that, in general, there is no guarantee that the global optimum can be reached from arbitrary initial conditions. Moreover, it becomes difficult to enforce constraints deriving from closed kinematic chains.

Instead of solving nonlinear problems directly, some other work relaxes the nonlinear constraints and solves an approximated IK problem. To name a few, Dai et al. [4] introduce an IK solver based on mixed-integer convex optimization and can certify global infeasibility. Maric et al. [13] use a Riemannian manifold parameterization to enable solutions with mature Riemannian optimization methods. Similar to our appoach, [24] and [6] both relax the IK problem into semidefinite problems (SDPs) each with an additional low-rank constraint. Different parameterizations are used in these two work. The former uses global rigid body transformations while the latter uses a "distance-geometric" formulation. In these two papers, the rank constraints are treated differently, where [24] simply drops the constraint while [6] provides a rank minimization algorithm. Compared to [24], our approach uses a simpler formulation and has smaller dimension. Compared to [6], our method parameterizes the robot differently and has a more principled rank minimization algorithm based on the maximization of the eigenvalues of positive semidefinite matrices with constant traces. In addition, we show that our method converges locally to feasible solutions.

Some other work also investigates semidefinite relaxation of problems not limited to IK, but general problems with rotations. For instance, [11], [21]–[23] propose SDP relaxations to different estimation problems involving rotations. In [2], [16], [18], the semidefinite relaxation techniques for such problems are evaluated for their tightness.

Our proposed approach provides the following novel contributions:

- We parametrize the problem exclusively as a function of the rotation of reference frames of each link, allowing us to easily incorporate a variety of constraints and arrangements of links, such as those arising in parallel robots.
- We develop a relaxation of the manifold of robot configurations as a combination of linear and semidefinite constraints. We show that our relaxation has some interesting properties: it is convex, it contains every kinematically feasible solution, and every kinematically

The authors are with Department of Mechanical Engineering, Boston University, 110 Cummington Mall, Boston, MA 02215, USA. Emails: tomwu@bu.edu, tron@bu.edu. The authors gratefully acknowledge the support by NSF award FRR-2212051.

feasible solution is on the boundary of the relaxed set.
- We can use the relaxation as a sound method to check for kinematic feasibility.
- We show local convergence to solutions that satisfy the manifold constraints with a novel rank minimization algorithm that is based on the maximization of a convex function on the relaxed set.

II. Preliminaries

In this section we define some notation and derivations to be used in the rest of the paper.

A. General notation

We use $\mathbf{I}_d$ to denote the identity matrix of dimension d, and we use $\mathbf{e}_i$ to define a standard basis column vector with 1 in the i-th entry and zero elsewhere. We denote the set of $n \times n$ symmetric matrices as $\mathbb{S}^n$ and the set of $n \times n$ positive semidefinite matrices as $\mathbb{S}^n_+$. The angle between two vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ is denoted as $\angle(\mathbf{v}_1, \mathbf{v}_2)$.

We make use of the vectorization property of the Kronecker product $\otimes$: $(\mathbf{B}^\mathsf{T} \otimes \mathbf{A})\text{vec}(\mathbf{X}) = \text{vec}(\mathbf{AXB})$, for any $\mathbf{A}, \mathbf{B}$ and $\mathbf{X}$ of appropriate dimensions.

B. Differentiating eigenvalues

Section VI-C introduced a rank minimization that uses the gradient of an eigenvalue with respect to the entries of the corresponding matrix. Specifically, consider a matrix $\mathbf{A} \in \mathbb{S}^n$, and let the eigenvalues of $\mathbf{A}$ be $\lambda_1 \geq \cdots \geq \lambda_n$. We are interested in finding $\frac{\partial \lambda_k}{\partial A}$, for the k-th largest eigenvalue λ_k. Lemma 1 summarizes a result from [12, Theorem 1].

Lemma 1 (Gradient of eigenvalues [12]): Given $\mathbf{X}_0 \in \mathbb{S}^n$, let $\mathbf{v}_k$ and λ_k be a pair of *normalized* eigenvector and eigenvalue of $\mathbf{X}_0$. For functions λ and v defined for all $\mathbf{X}$ in some neighborhood $N(\mathbf{X}_0) \subset \mathbb{R}^{n \times n}$ of $\mathbf{X}_0$ such that $\lambda(\mathbf{X}_0) = \lambda_k$ and $v(\mathbf{X}_0) = \mathbf{v}_k$, the gradient of $\lambda(\mathbf{X})$ is given by

$$\frac{\partial \lambda}{\partial \mathbf{X}} = \mathbf{v}_k \otimes \mathbf{v}_k. \tag{1}$$

III. Parameterization

This section discusses how to model general kinematic chains using rotations and translations.

A. Kinematic chains

To create a model for the IK problem, we define a world inertial frame $\mathcal{W}$, and we associate a reference frame $\mathcal{B}_i$, $i \in \{1, \ldots, n\}$ in correspondence of each joint. For revolute joints, the z-axis of each reference frame is aligned with the revolute axis.

We construct a graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of indices for the links and $\mathcal{E}$ is a set of ordered pairs that indicates the relations among the links. In particular, we have $(i, j) \in \mathcal{E}, i, j \in \mathcal{V}$ if the link i is the parent of link j. For robot with only spherical and revolute joints, $\mathcal{E}$ has two disjoint subsets, $\mathcal{E}_s$ and $\mathcal{E}_r$, respectively, representing the relations among spherical and revolute joints. We denote n_l and n_j as the number of links and joints.

We define respectively $\mathcal{V}_t$ and $\mathcal{V}_r$ the subsets of $\mathcal{V}$ whose translations and rotations are to be determined. We denote some special indices, including $base$ to represent a base frame (i.e., a frame rigidly attached to $\mathcal{W}$) and ee to represent the end-effector. We assume that there exists a path from any $base = p_1$ to the end effector $ee = p_n$, given by $\mathcal{P}_{fk} = \{p_1, p_2, \ldots, p_{n_{fk}}\} \subseteq \mathcal{V}$, where $(p_i, p_{i+1}) \in \mathcal{E}, \forall p_i, p_{i+1} \in \mathcal{P}_{fk}$.

B. Modeling kinematic chains using rotations and translations

The poses $\{\mathbf{R}_i, \mathbf{T}_i\}$ represent the rigid body transformation (rotation and translation) from the reference frame $\mathcal{B}_i$ to the world frame, i.e., $\mathbf{R}_i = {}^{\mathcal{W}}\mathbf{R}_{\mathcal{B}_i}$ and $\mathbf{T}_i = {}^{\mathcal{W}}\mathbf{T}_{\mathcal{B}_i}$. To simplify the notation, we denote the relative rotation from $\mathcal{B}_j$ to $\mathcal{B}_i$, ${}^{\mathcal{B}_i}\mathbf{R}_{\mathcal{B}_j}$ as ${}^{i}\mathbf{R}_j$, and the relative translation ${}^{\mathcal{B}_i}\mathbf{T}_{\mathcal{B}_j}$ is as ${}^{i}\mathbf{T}_j$.

In this paper, we assign a reference frame $\mathcal{B}_i$ to each link and parameterize our problem on subsets of $\{\mathbf{R}_i, \mathbf{T}_i\}$. This parameterization allows us to derive linear or semidefinite constraints, as we discuss in Section VI.

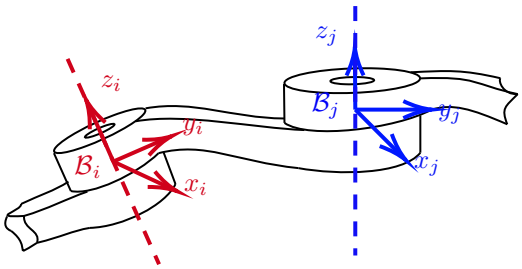


Fig. 1: The reference frames of two connected links $(i, j) \in \mathcal{E}_r$, each associated with a reference frame ($\mathcal{B}_i$ and $\mathcal{B}_j$).

A general set of variables to be solved for the IK problem is

$$\mathbf{x} = \{\mathbf{T}_{i_t} \in \mathbb{R}^3, \mathbf{R}_{i_r} \in \mathbf{SO}(3) | i_t \in \mathcal{V}_t, i_r \in \mathcal{V}_r\}. \tag{2}$$

We denote as n_t and n_r, respectively, the number of free translations and rotations in $\mathbf{x}$. For convenience, we define $\mathbf{u}$ as the vectorization of all the free rotations

$$\mathbf{u} = \text{stack}(\{\text{vec}(\mathbf{R}_{i_r})\}_{i_r \in \mathcal{V}_r}). \tag{3}$$

IV. Kinematic constraints

In this section we discuss how to model translation constraints on connected links as *linear* equality constraints and then investigate revolute joints and develop *linear* constraints on the joint axes and angle limits.

A. Joint translations

Given the rigid structure of the robot, the relative translation between two reference frames on connected links (i, j) is fixed. For $(i, j) \in \mathcal{E}$, the following relation holds:

$$\mathbf{T}_j - \mathbf{T}_i = \mathbf{R}_i {}^{i}\mathbf{T}_j. \tag{4}$$

In this equation, the translation ${}^{i}\mathbf{T}_j$ is given by the structure of the robot, while all the others are variables to be determined through the IK process. Using this relation, we can write the

translation of the end-effector as a function of the rotations along a path of links $\mathcal{P}_{fk}$. For $\forall i, j \in \mathcal{P}_{fk}$, we have

$$\begin{aligned}\mathbf{T}_{ee} &= \mathbf{T}_{base} + \sum_i \mathbf{R}_i {}^i\mathbf{T}_j \\ &= \mathbf{T}_{base} + \sum_i ({}^i\mathbf{T}_j^\mathsf{T} \otimes \mathbf{I}_3)\mathrm{vec}(\mathbf{R}_i) \\ &= \mathbf{T}_{base} + (\mathbf{M}_{te} \otimes \mathbf{I}_3)\mathrm{vec}(\mathbf{R}),\end{aligned} \tag{5}$$

where $\mathbf{R} = \begin{bmatrix}\mathbf{R}_1, \mathbf{R}_2, \ldots, \mathbf{R}_n\end{bmatrix}$ and $\mathbf{M}_{te} \in \mathbb{R}^{1\times 3n}$ is constructed with ${}^i\mathbf{T}_j$. Equation (5) provides a linear expression for end-effector location from the rotations. This expression allows us to algebraically eliminate the translations from our problem, and solve for the rotations alone (this is discussed more in detail in Section VI-A). Assuming the links are connected in a chain, and that at least one $\mathbf{T}_i$ is known (e.g., the robot base is fixed), then all the translations can be recovered using (4) once the rotations are determined.

Remark 1: Observe that (5) enables us to impose additional structural constraints on the robot. For example, consider a situation where two manipulators are working collaboratively with their end-effectors rigidly attached. For each of them, the end-effector position is a function of its rotations $\mathbf{R}$. To fulfill the cooperation requirements we can simply let these two functions equal to each other, resulting in a linear constraint on the closed chain.

B. Revolute joint axis constraints

For each pair of links $(i, j) \in \mathcal{E}_r$ that are connected with a revolute joint, the orientations $\mathbf{R}_i$ and $\mathbf{R}_j$ are limited by the equation

$$\mathbf{R}_j = \mathbf{R}_i \mathbf{R}_e \mathbf{R}_\theta \tag{6}$$

where $\mathbf{R}_\theta : \mathbb{R} \mapsto \mathbf{SO}(3)$ is a function of the joint angle θ defined such that $\mathbf{R}_\theta = \mathbf{I}$ when $\theta = 0$, and $\mathbf{R}_e$ is a parameter defined as the rotation from $\mathcal{B}_j$ to $\mathcal{B}_i$ when $\theta = 0$. Without loss of generality, we assume that $\mathbf{R}_\theta$ is a rotation about the $z-$axis, meaning that frames $\mathcal{B}_i$ and $\mathcal{B}_j$ share the same $z-$axis, that is:

$$\mathbf{R}_i \mathbf{R}_e \mathbf{e}_3 - \mathbf{R}_j \mathbf{e}_3 = \mathbf{0}. \tag{7}$$

We propose the following proposition for the constraint on revolute joint axis.

Proposition 1: For a robot with links $\mathcal{V}$ connected by relation $\mathcal{E}$ and variables defined by (2), the revolute joint common axis constraint (7) is satisfied for all $(i, j) \in \mathcal{E}_r$ if the rotations $\mathbf{R}_{i_r} \in \mathbf{SO}(3), \forall i_r \in \mathcal{V}_r$ and for all $(i, j) \in \mathcal{E}_r$, $\mathbf{R}_i$, $\mathbf{R}_j$ satisfy

$$(\mathbf{e}_3^\mathsf{T}\mathbf{R}_e^\mathsf{T} \otimes \mathbf{I})\mathrm{vec}(\mathbf{R}_i) - (\mathbf{e}_3^\mathsf{T} \otimes \mathbf{I})\mathrm{vec}(\mathbf{R}_j) = \mathbf{0}. \tag{8}$$

Proof: Equation (8) is equivalent to (7) after using the vectorization property of Kronecker product. ■

C. Revolute joint angle limits

In physical systems, the joint angle θ in (6) is limited to an interval $[-\phi_1, \phi_2]$, $\phi_1, \phi_2 > 0$. Without loss of generality, we can assume this interval to be symmetric, i.e., $\phi_1 = \phi_2 = \alpha$ (if not, we can translate the origin of the angle θ so that it is in the middle of the interval). With this assumption, the joint angle constraint becomes $|\theta| \leq \alpha$.

We are interested in developing angle limit constraints on the rotations. We define the following sets:

$$\begin{aligned}\mathcal{A}_{ij} &= \{\mathbf{R}_i, \mathbf{R}_j : \mathbf{R}_i\mathbf{R}_e\mathbf{e}_1 - \mathbf{R}_j\mathbf{e}_1 \in \mathcal{S}\big(\sqrt{2 - 2\cos(\alpha_{ij})}\big)\} \\ \mathcal{A} &= \{\{\mathbf{R}_i\} : (\mathbf{R}_i, \mathbf{R}_j) \in \mathcal{A}_{ij}, \forall (i, j) \in \mathcal{E}\}\end{aligned} \tag{9}$$

where $\mathcal{S}(r)$ is a ball with radius r and centered at the origin. These sets exactly capture the joint limit constraints, as shown in the following proposition.

Proposition 2: For a robot with links $\mathcal{V}$ connected by relation $\mathcal{E}$ and variables defined by (2), the revolute joint angle limit constraint is satisfied if $\{\mathbf{R}_i\}$ are rotations, and $\{\mathbf{R}_i\} \in \mathcal{A}$.

Proof: Observe that in (6) the rotations $\mathbf{R}_j$ and $\mathbf{R}_i\mathbf{R}_e$ share the same z-axis and their x- and y-axes are on the same plane. The angle θ can be seen as the angular displacement from the x-axis of $\mathbf{R}_i\mathbf{R}_e$ to that of $\mathbf{R}_j$. Therefore, for two vectors $\mathbf{w}_i = \mathbf{R}_i\mathbf{R}_e\mathbf{e}_1$ and $\mathbf{w}_j = \mathbf{R}_j\mathbf{e}_1$, we have $\theta = \angle(\mathbf{w}_i, \mathbf{w}_j)$ and $-\alpha_{ij} < \angle(\mathbf{w}_i, \mathbf{w}_j) < \alpha_{ij}$. Substituting (6), and the expressions for $\mathbf{w}_i, \mathbf{w}_j$ into $\|\mathbf{w}_i - \mathbf{w}_j\|^2$ we have $\|\mathbf{w}_i - \mathbf{w}_j\|^2 = 2 - 2\cos(\theta) \leq 2 - 2\cos(\alpha_{ij})$ for $\theta \in [-\alpha_{ij}, \alpha_{ij}]$, which gives the bound (see Fig. 2 for a visualization)

$$\mathbf{w}_i - \mathbf{w}_j \in \mathcal{S}\big(\sqrt{2 - 2\cos(\alpha_{ij})}\big). \tag{10}$$

This is equivalent to $(\mathbf{R}_i, \mathbf{R}_j) \in \mathcal{A}_{ij}$ and $\{\mathbf{R}_i\} \in \mathcal{A}$, from which the claim follows. ■

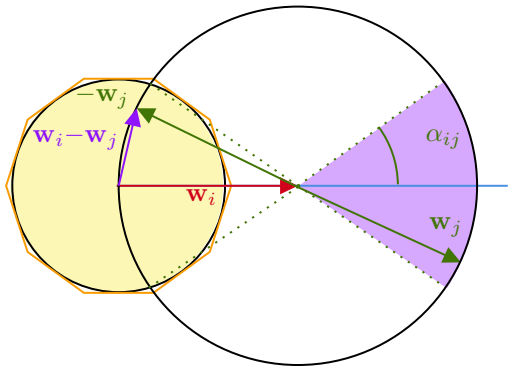


Fig. 2: The joint limit between two links $(i, j) \in \mathcal{E}_r$ can be written as an angle limit between two unit vectors $\mathbf{w}_i = \mathbf{R}_i\mathbf{R}_e\mathbf{e}_1$ and $\mathbf{w}_j = \mathbf{R}_j\mathbf{e}_1$ (purple sector), which can be further bounded by a ball (painted yellow) on $\mathbf{w}_i - \mathbf{w}_j$.

V. Modeling and Relaxation of the Feasible Set

In this section we introduce how to model and relax the feasible set defined by the kinematic constraints in the previous section.

A. Relaxation of the feasible set for revolute joints

We are interested in developing constraints that are linear or semidefinite Linear Matrix Inequalities (LMIs). First, we drop the manifold constraint $\mathbf{R}_{i_r} \in \mathbf{SO}(3)$; by doing so, we can

treat $\mathbf{R}_{i_r}$ as a simple real matrix; this constraint is revisited in Section V-B. Observe that the condition for joint axis in Proposition 1 is linear in the vectorized rotations $\mathbf{u}$ defined in (3). We therefore concatenate (8) for each $(i,j) \in \mathcal{E}_r$ as

$$\mathbf{A}_{\text{axis}}\mathbf{u} = \mathbf{b}_{\text{axis}}. \tag{11}$$

Next, from Proposition 2 the joint angle limit constraint requires that for every pair $(i,j) \in \mathcal{E}_r$, $(\mathbf{R}_i, \mathbf{R}_j)$ satisfies the ball bound (10), which can be approximated using linear inequalities, namely, a polyhedron. Specifically, for each $(i,j) \in \mathcal{E}_r$, multiple points are uniformly chosen on the ball that bounds $\mathbf{w}_i - \mathbf{w}_j$ in (10), and the polyhedron is defined as the hull formed by all faces tangent to the ball at the selected points. Because $\mathbf{w}_i$ and $\mathbf{w}_j$ are linear in $\mathbf{u}$, the linear inequalities for all $(i,j) \in \mathcal{E}_r$ can then be concatenated as

$$\mathbf{A}_{\text{angle}}\mathbf{u} \leq \mathbf{b}_{\text{angle}}. \tag{12}$$

B. Relaxation of $\mathbf{SO}(3)$

Our definition of $\mathbf{u}$ requires that each $\mathbf{R}_{i_r} \in \mathbf{SO}(3)$, i.e.,

$$\mathbf{R}_{i_r}^{\mathsf{T}}\mathbf{R}_{i_r} = \mathbf{I}_3 \text{ and } \det(\mathbf{R}_{i_r}) = +1. \tag{13}$$

These constraints are nonconvex in $\mathbf{u}$. We propose a novel way to relax $\mathbf{SO}(3)$ using convex constraints. For $\mathbf{R}_{i_r} = \begin{bmatrix}\mathbf{R}_{i_r}^{(1)} & \mathbf{R}_{i_r}^{(2)} & \mathbf{R}_{i_r}^{(3)}\end{bmatrix}$, equation (13) is equivalent to

$$\begin{aligned} \|\mathbf{R}_{i_r}^{(1)}\| &= 1 \\ \|\mathbf{R}_{i_r}^{(2)}\| &= 1 \\ \mathbf{R}_{i_r}^{(1)} \cdot \mathbf{R}_{i_r}^{(2)} &= 0 \\ \mathbf{R}_{i_r}^{(1)} \times \mathbf{R}_{i_r}^{(2)} &= \mathbf{R}_{i_r}^{(3)} \end{aligned} \tag{14}$$

We define a new variable $\mathbf{Y} = \begin{bmatrix}\mathbf{Y}_1 & \mathbf{Y}_2 & \dots & \mathbf{Y}_{i_r} & \dots & \mathbf{Y}_{n_r}\end{bmatrix} \in \mathbb{R}^{7\times 7n_r}, i_r \in \mathcal{V}_r$ with

$$\mathbf{Y}_{i_r} = \begin{bmatrix}\mathbf{R}_{i_r}^{(1)} \\ \mathbf{R}_{i_r}^{(2)} \\ 1\end{bmatrix}\begin{bmatrix}\mathbf{R}_{i_r}^{(1)} \\ \mathbf{R}_{i_r}^{(2)} \\ 1\end{bmatrix}^{\mathsf{T}} \in \mathbb{R}^{7\times 7}. \tag{15}$$

Observe that $\mathbf{Y}_{i_r}$ is a symmetric rank-1 matrix with the top left 6×6 block containing all the element-wise multiplication of $(\mathbf{R}_{i_r}^{(1)}, \mathbf{R}_{i_r}^{(1)}), (\mathbf{R}_{i_r}^{(1)}, \mathbf{R}_{i_r}^{(2)})$, and $(\mathbf{R}_{i_r}^{(2)}, \mathbf{R}_{i_r}^{(2)})$. The last column of $\mathbf{Y}_{i_r}$ consists of $\mathbf{R}_{i_r}^{(1)}$, $\mathbf{R}_{i_r}^{(2)}$, and 1. The advantages of choosing this structure are that:

1) the first three equations of (14) are linear in $\mathbf{Y}_{i_r}$ and for all $i_r \in \mathcal{V}_r$, these three equations can be concatenated as a linear equality constraint;
2) $\mathbf{R}_{i_r}^{(1)} \times \mathbf{R}_{i_r}^{(2)}$ is linear in $\mathbf{Y}_{i_r}$, meaning that we can represent $\mathbf{R}_{i_r}^{(3)}$ as a linear function of $\mathbf{Y}_{i_r}$;
3) since each column of $\mathbf{R}_{i_r}$ is linear in $\mathbf{Y}_{i_r}$, there exists a linear transformation g such that $\mathbf{u} = g(\mathbf{Y})$, which makes the revolute joint axis and angle limit constraints linear in $\mathbf{Y}_{i_r}$ as well.

Definition 1: We define explicitly a linear transformation $g(\mathbf{Y}) : \mathbb{R}^{7\times 7n_r} \mapsto \mathbb{R}^{9n_r}$ which is given by the composition of the following operations.

1) For each $\mathbf{Y}_{i_r}$, extract the first and second 3×1 vectors of the last column $\mathbf{y}_1$ and $\mathbf{y}_2$.
2) Compute $\mathbf{y}_3 = \mathbf{y}_1 \times \mathbf{y}_2$ using the needed elements from the top left 6×6 block of $\mathbf{Y}_{i_r}$. (Note that this is a linear operation and is different from directly computing the cross product.)
3) Concatenate vertically $\mathbf{y}_1$,$\mathbf{y}_2$, and $\mathbf{y}_3$ in sequence for all $i_r \in \mathcal{V}_r$.

Definition 2: We define $\mathcal{U}$ as the set of the vectors $\mathbf{u} = \text{stack}(\{\text{vec}(\mathbf{R}_{i_r})\}) \in \mathbb{R}^{9n_r}$ such that

$$\mathbf{A}_{\text{axis}}\mathbf{u} = \mathbf{b}_{\text{axis}}, \tag{16a}$$
$$\mathbf{A}_{\text{angle}}\mathbf{u} \leq \mathbf{b}_{\text{angle}}, \tag{16b}$$
$$\mathbf{R}_{i_r} \in \mathbf{SO}(3), \forall i_r \in \mathcal{V}_r \tag{16c}$$

and the set $\bar{\mathcal{U}}$ of $\bar{\mathbf{u}} = g(\mathbf{Y}) \in \mathbb{R}^{9n_r}$ such that

$$\mathbf{A}_{\text{axis}}\bar{\mathbf{u}} = \mathbf{b}_{\text{axis}}, \tag{17a}$$
$$\mathbf{A}_{\text{angle}}\bar{\mathbf{u}} \leq \mathbf{b}_{\text{angle}}, \tag{17b}$$
$$\mathbf{A}_{\text{structure}}\text{vec}(\mathbf{Y}) = \mathbf{b}_{\text{structure}} \tag{17c}$$
$$\mathbf{Y}_{i_r} \succeq 0, \forall i_r \in \mathcal{V}_r \tag{17d}$$

where (17c) is a constraint that imposes the following structure on $\mathbf{Y}$: for each $\mathbf{Y}_{i_r}$,

1) the sum of the first three diagonal entries, equals 1 and so does that of the next three;
2) the trace of the 3×3 block including on rows 1 to 3, and columns 4 to 6 of $\mathbf{Y}_{i_r}$ equals 0;
3) the bottom right entry of $\mathbf{Y}_{i_r}$ equals 1.

The structure constraint restrict the structure of $\mathbf{Y}_{i_r}$ in order to enforce the relations in (14).

We show some useful results about $\mathcal{U}$ and $\bar{\mathcal{U}}$.

Proposition 3: The set $\bar{\mathcal{U}}$ is compact.

Proof: Any $\mathbf{Y}_i \succeq 0$ can be decomposed as $\mathbf{Y}_i = U\Sigma U^{\mathsf{T}}$ where U is an orthonormal matrix and Σ is a diagonal matrix containing all the eigenvalues of $\mathbf{Y}_i$. The set of all $\mathbf{Y}_i$ defined in (17) is closed and bounded because every element of U belongs to $[0,1]$, while (17c) and (17d) restrict each elements of Σ to the interval $[0,3]$. Therefore $\bar{\mathcal{U}}$ is compact, as it is the image of a product of compact sets under the linear transformation $g(\cdot)$. ■

Proposition 4: The relaxed set $\bar{\mathcal{U}}$ contains every element of $\mathcal{U}$, i.e., $\mathcal{U} \subset \bar{\mathcal{U}}$.

Proof: Any $\mathbf{u} \in \mathcal{U}$ also satisfies (17a) and (17b). For every $\mathbf{u} \in \mathcal{U}$, we can construct a $\mathbf{Y}$ with rank-1 matrices $\mathbf{Y}_{i_r}$ as discussed in (15), which satisfies (17c). The only non-zero eigenvalue of $\mathbf{Y}_{i_r}$ equals 3 because $\text{tr}(\mathbf{Y}_{i_r}) = 3$. Therefore $\mathbf{Y}_{i_r} \succeq 0$ and $\bar{\mathbf{u}} = g(\mathbf{Y}) \in \bar{\mathcal{U}}$. ■

Proposition 5: The set $\mathcal{U}$ exactly matches the intersection of $\bar{\mathcal{U}}$ and $\mathcal{R}_1$, i.e., $\mathcal{U} = \bar{\mathcal{U}} \cap \mathcal{R}_1$, where $\mathcal{R}_1$ is the set of $\mathbf{u} = g(\mathbf{Y}) \in \mathbb{R}^{9n_r}$ such that each $\mathbf{Y}_{i_r} \in \mathbb{R}^{7\times 7}$ of $\mathbf{Y}$ is rank-1.

Proof: For any $\bar{\mathbf{u}} = g(\mathbf{Y}) \in \bar{\mathcal{U}} \cap \mathcal{R}_1$, because $\mathbf{Y}_{i_r} \succeq 0$ and $\text{rank}(\mathbf{Y}_{i_r}) = 1$, we can write each $\mathbf{Y}_{i_r}$ as

$$\mathbf{Y}_{i_r} = \begin{bmatrix}\mathbf{y}_1 \\ \mathbf{y}_2 \\ 1\end{bmatrix}\begin{bmatrix}\mathbf{y}_1 \\ \mathbf{y}_2 \\ 1\end{bmatrix}^{\mathsf{T}}. \tag{18}$$

The structural constraint (17c) acts on the entries of $\mathbf{Y}_{i_r}$ such that

$$\begin{cases} \mathrm{tr}(\mathbf{y}_1\mathbf{y}_1^\mathsf{T}) = \mathrm{tr}(\mathbf{y}_2\mathbf{y}_2^\mathsf{T}) = 1 \\ \mathrm{tr}(\mathbf{y}_1\mathbf{y}_2^\mathsf{T}) = 0 \end{cases} \tag{19}$$

which is equivalent to $\|\mathbf{y}_1\| = \|\mathbf{y}_2\| = 1$ and $\mathbf{y}_1^\mathsf{T}\mathbf{y}_2 = 0$. We form a new matrix $\tilde{\mathbf{R}}_{i_r} = \begin{bmatrix}\mathbf{y}_1 & \mathbf{y}_2 & \mathbf{y}_1 \times \mathbf{y}_2\end{bmatrix}$. It is clear that $\tilde{\mathbf{R}}_{i_r}$ satisfies (14) and thus (13) and (16c). Observe that the transformation g is defined such that $\tilde{\mathbf{u}} = g(\mathbf{Y})$ is the concatenation of all the $\mathrm{vec}(\tilde{\mathbf{R}}_{i_r})$. We also have $\tilde{\mathbf{u}}$ satisfies (16a) and (16b) because $g(\mathbf{Y})$ satisfies (17a) and (17b).

On the other hand, for any solution $\mathbf{u} \in \mathcal{U}$, we can always construct the columns of each $\mathbf{R}_{i_r}$ and form a corresponding rank-1 $\tilde{\mathbf{Y}}_{i_r}$ such that $\tilde{\mathbf{Y}}$ contains all $\tilde{\mathbf{Y}}_{i_r}$ satisfies (17). ■

Proposition 6: The set $\mathcal{U}$ is a subset of the boundary of $\bar{\mathcal{U}}$, i.e., $\mathcal{U} \subset \partial\bar{\mathcal{U}}$.

Proof: From Proposition 5 we have $\mathcal{U} = \bar{\mathcal{U}} \cap \mathcal{R}_1$. Thus for any $\mathbf{u} = g(\mathbf{Y}) \in \mathcal{U}$, each $\mathbf{Y}_{i_r}$ of $\mathbf{Y}$ is rank-1 and has a zero eigenvalue and therefore for any $t > 0$ there exists a matrix $\mathbf{M}$ such that $\mathbf{Y}_{i_r} + t\mathbf{M} \notin \mathbb{S}_+$. Thus $\mathbf{u} = g(\mathbf{Y})$ is on the boundary of $\bar{\mathcal{U}}$. ■

VI. Optimization

This section first introduces how to build an inverse kinematics optimization problem, then discusses how to relax this problem by dropping a rank constraint, and finally introduces a rank minimization algorithm to find low-rank solutions.

A. Inverse kinematics problem

The inverse kinematics problem aims to find the optimal and feasible $\mathbf{x}^*$ such that the end-effector, ee, reaches a desired location $\mathbf{T}_{goal}$ with an orientation $\mathbf{R}_{goal}$. This can be encoded as the cost

$$\|\mathrm{vec}(\mathbf{R}_{ee}) - \mathrm{vec}(\mathbf{R}_{goal})\|_2^2 + \|\mathbf{T}_{ee} - \mathbf{T}_{goal}\|_2^2. \tag{20}$$

We define a selction matrix $\mathbf{E}_{ee} \in \mathbb{R}^{3n\times 3}$ such that $\mathbf{R}\mathbf{E}_{ee} = \mathbf{R}_{ee}$ then substituting it along with (5) into the cost to get

$$\begin{aligned} f_0(\mathbf{R}) = &\|(\mathbf{E}_{ee}^\mathsf{T} \otimes \mathbf{I}_3)\mathrm{vec}(\mathbf{R}) - \mathrm{vec}(\mathbf{R}_{goal})\|_2^2 \\ &+ \|(\mathbf{M}_{te} \otimes \mathbf{I}_3)\mathrm{vec}(\mathbf{R}) + \mathbf{T}_{base} - \mathbf{T}_{goal}\|_2^2 \end{aligned} \tag{21}$$

Vectorizing (21), we can define an equivalent quadratic function $f(\mathbf{u})$ such that $f(\mathbf{u}) = f_0(\mathbf{R})$. Our inverse kinematics problem is then defined as

Problem 1 (Inverse kinematics):

$$\min_{\mathbf{u}=\mathrm{stack}(\{\mathrm{vec}(\mathbf{R}_{i_r})\})} \quad f(\mathbf{u}) \tag{22a}$$

$$\text{subject to} \quad \mathbf{u} \in \mathcal{U} \tag{22b}$$

As defined above, the set $\mathcal{U}$ is the feasible set such that any $\mathbf{u} \in \mathcal{U}$ satisfies the joint axis and angle limit constraints. When $f(\mathbf{u}) = 0$, from (20) we know that the pose of the end-effector matches the target.

B. Rank constrained problem

Using Proposition 5 we can equivalently write the inverse kinematics problem as

Problem 2a (Rank constrained inverse kinematics):

$$\min_{\mathbf{Y}\in\mathbb{R}^{7\times 7n_r}} \quad h(\mathbf{Y}) \tag{23a}$$

$$\text{subject to} \quad \mathbf{u} = g(\mathbf{Y}) \in \bar{\mathcal{U}} \cap \mathcal{R}_1 \tag{23b}$$

where $h(\mathbf{Y}) = f \circ g(\mathbf{Y})$. This problem has a quadratic objective function and a convex constraint except for $\mathbf{u} \in \mathcal{R}_1$, which requires the rank of each $\mathbf{Y}_{i_r}$ to be one. We define the following relaxed problem which is obtained from Problem 2a but with the omission of the rank constraint $\mathbf{u} \in \mathcal{R}_1$.

Problem 2b (Relaxed inverse kinematics):

$$\min_{\mathbf{Y}\in\mathbb{R}^{7\times 7n_r}} \quad h(\mathbf{Y}) \tag{24a}$$

$$\text{subject to} \quad \mathbf{u} = g(\mathbf{Y}) \in \bar{\mathcal{U}} \tag{24b}$$

From Proposition 5 we know that the feasible set of Problem 2a equals that of Problem 1. Additionally, from Propositions 4 and 6, the feasible set of Problem 1 is a subset of that of Problem 2b and every feasible solution to Problem 1 lies on the boundary $\partial\bar{\mathcal{U}}$.

C. Rank minimization via eigenvalue maximization

We propose a way to solve for rank-1 matrices by manipulating their eigenvalues by using the following.

Proposition 7: Consider a matrix $\mathbf{M}_0 \in \mathbb{S}_+^m$, whose eigenvalues are $\lambda_1 \geq \cdots \geq \lambda_m$. Consider the function $\lambda_1(\mathbf{M}) = \lambda_1$ defined for all $\mathbf{M}$ in a neighborhood $\mathbf{N}(\mathbf{M}_0) \subset \mathbb{S}_+^m$ of $\mathbf{M}_0$. When $\mathrm{tr}(\mathbf{M}) = c$ and $c \in \mathbb{R}$, $\mathbf{M}_0$ is a rank-1 matrix if and only if $\mathbf{M}_0 = \mathrm{argmax}(\lambda_1(\mathbf{M}))$.

Proof: The trace of a matrix is also the sum of all of its eigenvalues, which are all non-negative when the matrix is positive semidefinite. When $\mathrm{tr}(\mathbf{M}) = c$, the condition $\mathbf{M}_0 = \mathrm{argmax}(\lambda_1(\mathbf{M}))$ is achieved when $\lambda_1(\mathbf{M}_0) = c$ and $\lambda_2(\mathbf{M}_0), \ldots, \lambda_m(\mathbf{M}_0) = 0$, meaning that $\mathrm{rank}(\mathbf{M}_0) = 1$. On the other hand, when $\mathrm{tr}(\mathbf{M}) = c$, $\mathrm{rank}(\mathbf{M}_0) = 1$ implies that the only positive eigenvalue $\lambda_1(\mathbf{M}_0)$ equals c and since $0 \leq \lambda_1(\mathbf{M}) \leq c$ we have $\mathbf{M}_0 = \mathrm{argmax}(\lambda_1(\mathbf{M}))$. ■

Observe that the structure constraint (17c) of Problem 2a restricts that the trace of each $\mathbf{Y}_{k,i_r}$ always equals 3. Using Proposition 7 we can rewrite Problem 2a as the following problem:

Problem 2c (Eigenvalue maximization):

$$\max_{\mathbf{Y}\in\mathbb{R}^{7\times 7n_r}} \quad \sum_{i_r\in\mathcal{V}_r} \lambda_1(\mathbf{Y}_{i_r}) \tag{25a}$$

$$\text{subject to} \quad \mathbf{Y} = \mathrm{argmin}(h(\mathbf{Y})) \tag{25b}$$

$$\mathbf{u} = g(\mathbf{Y}) \in \bar{\mathcal{U}} \tag{25c}$$

We propose a gradient-based approach to Problem 2c, in which we first find a solution to Problem 2b and then minimize the rank by iteratively solving the following SDP

Problem 3 (Sequential problem):

$$\max_{\mathbf{U}_k\in\mathbb{R}^{7\times 7n_r}} \quad \sum_{i_r\in\mathcal{V}_r} \mathrm{vec}(\mathbf{U}_{k,i_r})^\mathsf{T}(\mathbf{V}_{k-1,i_r}^{(1)} \otimes \mathbf{V}_{k-1,i_r}^{(1)}) \tag{26a}$$

$$\text{subject to} \quad \nabla h(\mathbf{U}_k) = \mathbf{0} \tag{26b}$$

$$g(\mathbf{Y}_{k-1} + \mathbf{U}_k) \in \bar{\mathcal{U}} \tag{26c}$$

The variable $\mathbf{U}_k$ is an update of $\mathbf{Y}_k$, i.e., $\mathbf{Y}_{k,i_r} = \mathbf{Y}_{k-1,i_r} + \mathbf{U}_{k,i_r}$. The objective function (26a) is the inner product of $\mathrm{vec}(\mathbf{U}_{k,i_r})$ and the gradient of the largest eigenvalue λ_1 of $\mathbf{Y}_{k-1,i_r}$ with respect to $\mathbf{Y}_{k-1,i_r}$ itself, which can be computed using Lemma 1:

$$\begin{aligned}&\sum_{i_r\in\mathcal{V}_r} \mathrm{vec}(\mathbf{U}_{k,i_r})^\mathsf{T} \frac{\partial \lambda_1(\mathbf{Y}_{k-1,i_r})}{\partial \mathbf{Y}_{k-1,i_r}} \\ &= \sum_{i_r\in\mathcal{V}_r} \mathrm{vec}(\mathbf{U}_{k,i_r})^\mathsf{T} (\mathbf{V}^{(1)}_{k-1,i_r} \otimes \mathbf{V}^{(1)}_{k-1,i_r}),\end{aligned} \tag{27}$$

where $\mathbf{V}^{(1)}_{k-1,i_r}$ is the normalized eigenvector of $\mathbf{Y}_{k-1,i_r}$ corresponding to λ_1. With this, (26a) ensures that each $\mathbf{U}_{k,i_r}$ moves in the direction of the largest possible improvement in terms of increasing the sum of largest eigenvalues of the matrices $\mathbf{Y}_{k,i_r}$. Since the trace of $\mathbf{Y}_{k,i_r}$ is fixed, when the largest eigenvalue is increased, the other eigenvalues will decrease and eventually make each $\mathbf{Y}_{k,i_r}$ a rank-1 matrix. The constraint (26b) ensures that $h(\mathbf{Y}_k) = h(\mathbf{Y}_0)$ throughout the iterations, where $\mathbf{Y}_0$ is the solution of Problem 2b. The constraint (26c) makes sure that the updated $\mathbf{u} = g(\mathbf{Y}_k)$ remains on the feasible set $\bar{\mathcal{U}}$.

The complete algorithm that solves inverse kinematics Problem 1 is summarized in Algorithm 1.

Algorithm 1 Iterative SDP Inverse Kinematics Solver

Input $\mathbf{T}_{goal}, \mathbf{R}_{goal}, \mu, \epsilon_1, \epsilon_2, k_{max}$
Output $\mathbf{x}^*$

1: Solve Problem 2b to get an initial solution $\mathbf{Y}_0$ and set $k = 1$.
2: **while** $\exists \lambda_{1,i_r} \leq 3 - \epsilon_1$ & $\|\mathbf{U}_k\|_F \geq \epsilon_2$ & $k \leq k_{max}$ **do**
3: For each $\mathbf{Y}_{k-1,i_r}$, compute the largest eigenvalue λ_{1,i_r} and the corresponding normalized eigenvector $\mathbf{V}^{(1)}_{k-1,i_r}$.
4: Solve Problem 3 to get $\mathbf{U}_k$.
5: Update $\mathbf{Y}_{k,i_r} = \mathbf{Y}_{k-1,i_r} + \mathbf{U}_{k,i_r}$ for all $i_r \in \mathcal{V}_r$ and set $k = k + 1$.
6: **end while**
7: Recover the rotations $\{\mathbf{R}_{i_r}\}$ by reshaping $g(\mathbf{Y}_{k-1})$.
8: Recover the translations $\{\mathbf{T}_{i_t}\}$ using (4).
9: **return** $\mathbf{x}^*$ defined in (2).

D. Convergence analysis

We provide below a convergence analysis for proposed algorithm. We start with the following assumption.

Assumption 1: Problem 1 is feasible and there exists a $\mathbf{u}^* \in \mathcal{U}$ such that $f(\mathbf{u}^*) = 0$.

Proposition 8: When Assumption 1 holds, the corresponding $\mathbf{Y}^* = g^{-1}(\mathbf{u}^*)$ is a global minimizer of Problem 2b.

Proof: By Proposition 4, $\mathbf{Y}^*$ satisfies all the constraints in Problem 2b. The cost satisfies $h(\mathbf{Y}^*) = f \circ g(\mathbf{Y}^*) = f(\mathbf{u}^*) = 0$. Since Problem 2b is convex, and $\min(h(\mathbf{Y})) = 0$, the solution $\mathbf{Y}^*$ is a global minimizer of Problem 2b. ■

Proposition 9: Every globally optimal solution $\mathbf{Y}^*$ of Problem 2c is also an optimal solution of Problem 2a, and $\mathbf{u}^* = g(\mathbf{Y}^*) \in \partial\bar{\mathcal{U}}$.

Proof: For a maximizer $\mathbf{Y}^*$ of Problem 2c, by Proposition 7 it holds that $\mathbf{u}^* = g(\mathbf{Y}^*) \in \mathcal{R}_1$ and since $\mathbf{Y}^* = \mathrm{argmin}(h(\mathbf{Y}))$ it is an optimal solution to Problem 2a. Using Proposition 5 and 6 we have $\mathbf{u}^* \in \bar{\mathcal{U}} \cap \mathcal{R}_1 = \mathcal{U} \in \partial\bar{\mathcal{U}}$. ■

Proposition 10: When Assumption 1 holds and $k \to +\infty$, it holds that $\mathbf{Y}_k \to \tilde{\mathbf{Y}}^*$ and $\tilde{\mathbf{u}}^* = g(\tilde{\mathbf{Y}}^*) \in \partial\bar{\mathcal{U}}$, where $g(\tilde{\mathbf{Y}}^*)$ is a local maximizer of Problem 2c.

Proof: Consider another version of Problem 2c (we refer it as Problem 2d) where the constraint (25b) is replaced with $\nabla h(\mathbf{Y}) = \mathbf{0}$. This new constraint can be seen as a convex relaxation of (25b) because $\nabla h(\mathbf{Y}) = \mathbf{0}$ implies that $\mathbf{Y}$ is the minimizer of the convex function h. The objective function of this problem is convex in $\mathbf{Y}$ because $\lambda_1(\mathbf{Y}_{i_r}) = \sup \mathbf{V}^{(1)\mathsf{T}}_{i_r} \mathbf{Y}_{i_r} \mathbf{V}^{(1)}_{i_r}$ is convex in $\mathbf{Y}_{i_r}$. As a result, Problem 2d is a maximization of a convex function over a convex set. Algorithm 1 can be seen as a gradient approach to Problem 2d. Since $\bar{\mathcal{U}}$ is compact, when $k \to +\infty$, $\mathbf{u} = g(\mathbf{Y}) \to g(\tilde{\mathbf{Y}}^*) \in \partial\bar{\mathcal{U}}$ and $g(\tilde{\mathbf{Y}}^*)$ is a local maximizer. To see why, for any point $\mathbf{Y}$ in the neighborhood $N(\tilde{\mathbf{Y}}^*)$ such that $g(\mathbf{Y}) \in \bar{\mathcal{U}} \cup \{\nabla h(\mathbf{Y}) = 0\}$, it holds that $\lambda_1(\tilde{\mathbf{Y}}^*) \geq \lambda_1(\mathbf{Y})$ because by contradiction if there were a $\bar{\mathbf{Y}} \in N(\tilde{\mathbf{Y}}^*)$ and $\bar{\mathbf{Y}} = \mathbf{Y}_{k-1} + \bar{\mathbf{U}}_k$ such that $\lambda_1(\bar{\mathbf{Y}}) \geq \lambda_1(\tilde{\mathbf{Y}}^*)$, then the fact that $\mathbf{U}_k = \mathrm{argmax}(\sum_{i_r\in\mathcal{V}_r} \mathrm{vec}(\mathbf{U}_{k,i_r})^\mathsf{T} \frac{\partial \lambda_1(\mathbf{Y}_{k,ir})}{\partial \mathbf{Y}_{k,i_r}})$ would not hold.

■

VII. Simulation Results

To check if our IK solver can find postures for complicated constraints such as closed kinematic chain and joint limits, the proposed method is implemented on a humanoid dual-arm Rethink Robotics Baxter robot. The robot possesses two arms mounted on a fixed torso. Each arm has 7 revolute joints with angle limits. We model the arms of the robot as one closed kinematic chain by rigidly aligning the two grippers on a common line with a fixed distance to simulate the task of collaboratively holding a box. The objective is to solve for postures of the robot given some predetermined goal poses of the end-effector.

Unlike most traditional IK solvers, which can only deal with open kinematic chains, our solver can evaluate the kinematic chain as a whole without cutting it into separate sub-trees. This can be done by adding a linear constraint discussed in Remark 1 to Problems 2b and 3.

Figure 3 shows a solution to a given $\mathbf{T}_{goal}$ and $\mathbf{R}_{goal}$ using our IK solver. In this example, the total number of free links is $n_r = 15$, which defines the size of variables $\mathbf{Y} \in \mathbb{R}^{7\times 7n_r}$. The total numbers of rows are 77 for equality constraints and 6112 for inequality constraints. The end-effector is set as the midpoint of the two grippers and is treated as a link of the robot asigned with the reference frame $\{\mathbf{R}_{ee}, \mathbf{T}_{ee}\}$. The errors of the end-effector pose, $err(\mathbf{R}_{ee}) = \|\mathrm{vec}(\mathbf{R}_{ee}) - \mathrm{vec}(\mathbf{R}_{goal})\|_2$ and $err(\mathbf{T}_{ee}) = \|\mathbf{T}_{ee} - \mathbf{T}_{goal}\|_2$ are $1.59 \cdot 10^{-8}$ and $4.61 \cdot 10^{-9}$, respectively. We verified that all the poses satisfy the imposed constraints in Problem 1 along with the translation relation (4). Figure 4 shows some results regarding the computation process where 4a shows

the change of the largest eigenvalue, λ_1 of each $\mathbf{Y}_{k,i_r}$ during the rank minimization process. We observed that values of λ_1 increase iteratively, eventually reaching the maximum value of 3 (given by the trace constraint) at $k = 5$. Figure 4b presents the 7 eigenvalues of every $\mathbf{Y}_{i_r}$ in the final solution, where all eigenvalues except λ_1 are below the tolerance ϵ_1. This shows that each $\mathbf{Y}_{i_r}$ in the solution is approximately a rank-1 matrix. With the above results, we can say that in this example, the solver successfully solves the IK problem.

To test the performance of our solver on multiple different targets, we implement it on a set of random end-effector poses. We build this set by randomly sampling 500 points in a space $\mathbf{T}_{goal} = [x, y, z]^\mathsf{T} \in \mathcal{T}_{goal}$, where $x \in [0.4, 0.75]$, $y \in [-0.2, 0.2]$, and $z \in [0.2, 0.7]$. For each point, we asign a randomly generated orientation $\mathbf{R}_{goal} = \mathbf{R}_z(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_x(\gamma) \in \mathcal{R}_{goal}$, where $\alpha \in [0, \pi/2]$, $\beta \in [0, \pi]$, and $\gamma \in [\pi/2, 0]$. These poses are selected based on the mutual reachable space of the arms but are not guaranteed to have feasible IK solutions. For comparison, we applied a BFGS IK solver (available with the MATLAB `generalizedInverseKinematics` class) to the same problem set. It is worth-mentioning that this solver can only find solutions to open kinematic chains. Therefore, for a mutual end-effector pose of the two arms, the BFGS solver is applied twice, one for each arm, which is different from our method. Moreover, unlike our method, the BFGS solver requires an initial guess every time, which is set to be the zero joint angles in this simulation. The MOSEK [14] SDP solver is employed to solve the SDP problems within our method. Eventually, the results are visualized in Figure 5, where the sampled goals are colored in terms of which of the methods succeed. It is seen that in the tested 500 problems, in 355 times both methods succeed (71%, green dots); in 28 times only our method succeeds (5.6%, blue dots); in 33 times only BFGS succeeds (6.6%, purple dots); and in 84 times both solvers fail (16.8%, red dots). The solvers are compared for their performance in Table I including success rates and for successful solutions: the average time covering only the time consumed in the SDP solver and the average errors of the end-effector poses. Some other results of our method are also listed. This includes maximal $\|\mathbf{R}_{i_r} - P(\mathbf{R}_{i_r})\|_F$, which is the maximal value of all Frobenius norms of the difference between computed $\mathbf{R}_i$ and its projection $P(\mathbf{R}_{i_r})$ on $\mathbf{SO}(3)$ (see [20]), for all $i_r \in \mathcal{V}_r$ in the successful solutions. This shows how close to the $\mathbf{SO}(3)$ manifold the computed rotations are. Another result is the maximal value within all of the second largest eigenvalues of every $\mathbf{Y}_{i_r}$ in the successful solutions. This shows how close to rank-1 matrix each $\mathbf{Y}_{i_r}$ is.

Looking at the results in Table I, we see that although having a similar success rate and precision, our method computes the results slower than the BFGS solver. However, the last two columns of Table I shows that our method finds valid solutions with rank-1 $\mathbf{Y}_{i_r}$ and the recovered rotations are on $\mathbf{SO}(3)$, which verifies that the proposed rank minimization algorithm works. From the results in Figure 5 we see that the proposed method can find solutions in problems that the BFGS solver fails. Another interesting finding is that, in all of the problems that BFGS solver succeeds on, none of them failed with "infeasible" status in step 1 of Algorithm 1 when our method is applied, meaning that at least in the tested problems there does not exist a feasible IK problem that is infeasible to Problem 2b. This would otherwise contradict Proposition 4, which enables us to certify infeasibility by solving Problem 2b as a feasibility problem and we are sure that the problem is infeasible when no feasible solution can be found.

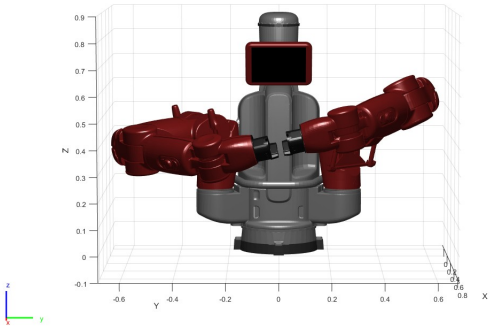
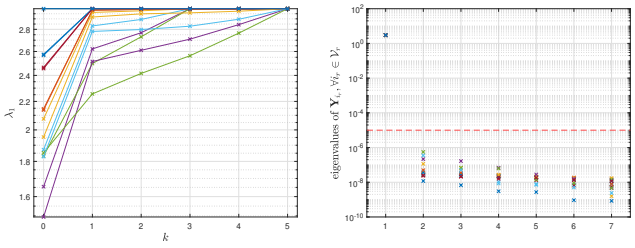


Fig. 3: An example posture solved for Baxter, where the two arms are modeled as one closed kinematic chain.



(a) The largest eigenvalues λ_1 of each $\mathbf{Y}_{k,i_r}$ over iteration k, where each line corresponds to one matrix.

(b) The eigenvalues of each $\mathbf{Y}_{i_r}$ in the solution, where all eigenvalues except the largest one are below the tolerance ϵ_1 (red dashed line)

Fig. 4: Computational results of the solution in Fig. 3

Method	Success rate	Avg. time	$\overline{err}(\mathbf{R}_{ee})$
SDP	76.6%	1.2629 s	1.0308e-08
BFGS	77.6%	0.1966 s	1.3452e-08

Method	$\overline{err}(\mathbf{T}_{ee})$	$\max(\|\mathbf{R}_{i_r} - P(\mathbf{R}_{i_r})\|_F)$	maximal e_2
SDP	3.7495e-09	6.9394e-06	6.7919e-06
BFGS	5.9359e-09		

TABLE I: Performance of the proposed IK solver on 500 different goals compared against the BFGS solver

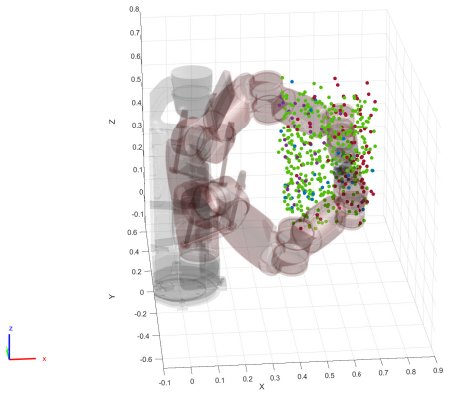


Fig. 5: Implementations of the proposed method and a traditional BFGS Gradient Projection method on a dual-arm Baxter for 500 different end-effector poses with the results: both of the solver succeed (green dots); only our solver succeeds (blue dots); only BFGS succeeds (purple dots); both solvers fail (red dots).

VIII. Conclusions

In this paper we offer a new relaxation of the feasible sets in inverse kinematics problems. The relaxed set contains every feasible solutions, meaning that we can use it to certify infeasibility. We show through simulations that the proposed method is applicable to closed kinematic chain and can serve as an alternative approach for existing solvers. The objective costs developed in this work is defined separately from the constraints, meaning that we can replace it with other formulations designed for different kinematic problems. For instance, we will in the future include costs for forces and grasping tasks, which will take into consideration entire kinematic chains and better demonstrate advantage of the rotation parameterization. To further hone the performance we can also try to algebraically remove the equality constraints to reduce the number of variables in the SDP solver. Moreover, we are interested in including more kinematic constraints such as prismatic joints to extend the capability of our method.

References

[1] A. Aristidou and J. Lasenby. Fabrik: A fast, iterative solver for the inverse kinematics problem. *Graphical Models*, 73(5):243–260, 2011.

[2] A. S. Bandeira, N. Boumal, and A. Singer. Tightness of the maximum likelihood semidefinite relaxation for angular synchronization. *Mathematical Programming*, 163:145–167, 2017.

[3] P. Beeson and B. Ames. Trac-ik: An open-source library for improved solving of generic inverse kinematics. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 928–935, 2015.

[4] H. Dai, G. Izatt, and R. Tedrake. Global inverse kinematics via mixed-integer convex optimization. *The International Journal of Robotics Research*, 38(12-13):1420–1441, 2019.

[5] R. Diankov. Automated construction of robotic manipulation programs. 2010.

[6] M. Giamou, F. Marić, D. M. Rosen, V. Peretroukhin, N. Roy, I. Petrović, and J. Kelly. Convex iteration for distance-geometric inverse kinematics. *IEEE Robotics and Automation Letters*, 7(2):1952–1959, 2022.

[7] M. L. Husty, M. Pfurner, and H.-P. Schröcker. A new and efficient algorithm for the inverse kinematics of a general serial 6r manipulator. *Mechanism and machine theory*, 42(1):66–81, 2007.

[8] B. Kenwright. Inverse kinematics–cyclic coordinate descent (ccd). *Journal of Graphics Tools*, 16(4):177–217, 2012.

[9] T. Le Naour, N. Courty, and S. Gibet. Kinematics in the metric space. *Computers & Graphics*, 84:13–23, 2019.

[10] H.-Y. Lee and C.-G. Liang. Displacement analysis of the general spatial 7-link 7r mechanism. *Mechanism and machine theory*, 23(3):219–226, 1988.

[11] M. Li, G. Liang, H. Luo, H. Qian, and T. L. Lam. Robot-to-robot relative pose estimation based on semidefinite relaxation optimization. pages 4491–4498, 2020.

[12] J. R. Magnus. On differentiating eigenvalues and eigenvectors. *Econometric Theory*, 1:179–191, 1985.

[13] F. Marić, M. Giamou, A. W. Hall, S. Khoubyarian, I. Petrović, and J. Kelly. Riemannian optimization for distance-geometric inverse kinematics. *IEEE Transactions on Robotics*, 38(3):1703–1722, 2021.

[14] MOSEK ApS. *The MOSEK optimization toolbox for MATLAB manual. Version 10.0.*, 2022.

[15] R. Muller-Cajar and R. Mukundan. Triangualation-a new algorithm for inverse kinematics. 2007.

[16] L. Peng, M. Fazlyab, and R. Vidal. Semidefinite relaxations of Truncated Least-Squares in robust rotation search: Tight or not. 2022.

[17] M. Raghavan and B. Roth. Inverse kinematics of the general 6r manipulator and related linkages. 1993.

[18] J. Saunderson, P. A. Parrilo, and A. S. Willsky. Semidefinite descriptions of the convex hull of rotation matrices. *SIAM Journal on Optimization*, 25(3):1314–1343, 2015.

[19] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. Modelling, planning and control. *Advanced Textbooks in Control and Signal Processing. Springer,*, 2009.

[20] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 13(04):376–380, 1991.

[21] H. Yang. *Certifiable Outlier-Robust Geometric Perception*. PhD thesis, Massachusetts Institute of Technology, 2022.

[22] H. Yang and L. Carlone. A quaternion-based certifiably optimal solution to the Wahba problem with outliers. pages 1665–1674, 2019.

[23] H. Yang, J. Shi, and L. Carlone. TEASER: Fast and certifiable point cloud registration. 37(2):314–333, 2020.

[24] T. Yenamandra, F. Bernard, J. Wang, F. Mueller, and C. Theobalt. Convex optimisation for inverse kinematics. In *2019 International Conference on 3D Vision (3DV)*, pages 318–327. IEEE, 2019.