Journal of Intelligent & Robotic Systems (2021) 101: 30
https://doi.org/10.1007/s10846-020-01279-w

Solving the Inverse Kinematics Problem of Multiple Redundant Manipulators with Collision Avoidance in Dynamic Environments

Liangliang Zhao[1] · Jingdong Zhao[1] · Hong Liu[1]

Received: 13 June 2020 / Accepted: 21 October 2020 / Published online: 22 January 2021

Abstract

This article presents an approach for collision-free kinematics of multiple redundant manipulators in complex environments. The approach describes a representation of task space and joint limit constraints for redundant manipulators and handles collision-free constraints by micromanipulator dynamic model and velocity obstacles. A new algorithm based on Newton-based and first-order techniques is proposed to generate collision-free inverse kinematics solutions. The present approach is applied in simulation for the redundant manipulators in a various working environments with dynamic obstacles. The physical experiments using a Baxter robot in a various working environments with dynamic obstacles are also performed. The results demonstrate the effectiveness of the proposed approach compared with existing methods regarding working environment and computational cost.

Keywords Inverse kinematics · Multiple manipulators · Collision avoidance · Dynamic obstacles

1 Introduction

One current trend in robotics is to use the manipulators in dynamic environments for different tasks. Moreover, the need for spatial and temporal coordination of multiple manipulators work cooperatively in complex environments arises in industrial and service robotics alike. For instance, many complex tasks in industrial processes that cannot be accomplished by a single manipulator are still performed in cooperation with human workers or other robots. Furthermore, cooperative human-robot and multi-robot working environments have been extensively applied in various related fields [1, 2]. In such systems, collision avoidance is a major problem in the application of the manipulators. To enhance human safety and work efficiency, the manipulator must compute an inverse kinematics (IK) solution based on the desired pose of its end-effector and local observations of the workspace, such that it stays free of collisions with the human workers or other moving robots.

The redundant manipulators are designed with more degrees of freedom (DOF) that have been used in many complex environments to achieve the tasks. However, the complex environments provide significant challenges in inverse kinematics problem of the redundant manipulates:

1. Task Space Constraints: Computing the joint coordinates that satisfy kinematics constraints based on the given each end-effector pose [3, 4].
2. Collision-free Constraints: A redundant manipulator should not collide with other manipulators and/or moving obstacles [5].
3. Joint Limits Constraints: Joints have limited range of movement so the manipulator cannot move if a joint move to a position outside its limits [6, 7].

To compute efficient IK solutions for redundant manipulators, the numerical and optimization methods have been used in several approaches in literature [8–12]. However, most methods do not focus on collisions with other manipulators and/or moving obstacles in a shared working environment. Moreover, there are many studies work on the problem of collision-avoidance for multi-robot systems or moving obstacles [25–35]. The limitations of these studies though are that the robots are assumed to rigid bodies and only take dynamics constraints into account; they can not be directly applied to redundant manipulators.

✉ Jingdong Zhao
zhaojingdong@hit.edu.cn

[1] Harbin Institute of Technology, 92 West Dazhi Street, Nan Gang District, Harbin, Heilongjiang Province, 150001, China

In this paper, a real-time collision-free IK (CFIK) algorithm is presented to compute efficient IK solutions for a redundant manipulator, which operates in a three-dimensional (3-D) working environment with other manipulators and/or moving obstacles. The algorithm uses a novel combination of the Jacobian matrix and velocity obstacles (VO). In such a case, the formulation of the VO is extended to redundant manipulators and the IK problem is defined as a new high dimensional optimization problem by task constraints. Our algorithm initially relates IK solutions to task space and joint limit constraints for each redundant manipulator and updates joint state at each iteration to reduce the task error by a Newton-based method. In contrast to existing IK algorithm, the CFIK algorithm continues to search for the collision-free IK solutions using a first-order method. The basic idea is that the movements of each joint are guided by collision-free constraints, which are formulated using linear inequalities and taken into account to compute the constraint factor of the VO in the given time window. Based on a brief analysis of convergence rate and stability of the collision-free constraints, it can be concluded that the proposed algorithm provides collision-avoidance guarantee for all redundant manipulators when working in a shared space.

The CFIK algorithm has been implemented in simulation and experiment for a variety of working environments with a Baxter robot and computes IK solutions that satisfy the task space, joint limit and collision-free constraints. The algorithm also considers the dynamic obstacles moving in the environments. To guarantee that there is no collision, the simulations use Gazebo simulator, which supports physics engines and offers the ability to efficiently and accurately simulate robots in dynamic environments. In general, the run time to generate an IK solution depends on the kinematic solver and the number of manipulators and dynamic obstacles. In our benchmarks, it takes less than a millisecond for a 7-DOF arm on a single CPU core.

The rest of this paper is organized as follows. Section 2 reviews the related work in IK and VO algorithms. In Section 3, the formulation of task space, joint limit and collision-free constraints are applied to the redundant manipulators. In Section 4, we relate IK solutions to all three constraints, present a novel CFIK algorithm and discuss how to address the problem of collision avoidance with static obstacles in this framework. Section 5 presents a convergence rate and stability analysis for the collision-free constraints. Section 6 highlights the performance on both arms of Baxter robot in dynamic environments with the human model and the Kuka LBR iiwa articulated robot arm. Our method solves a broader class of problems and performs much faster than existing methods. Finally, the paper is concluded in Section 7.

2 Previous Work

In this section, the overview of prior work on inverse kinematics of redundant manipulators and collision-free navigation of multiple robots will be given.

The most popular way to generate valid IK solutions is to use closed-loop algorithms based on the Inverse Jacobian matrix [8, 9]. As in [10], the relevant closed-loop IK (CLIK) methods for redundant robots are compared based on the main points of kinematic issue and two enhancements are proposed. In [11], the problem of false-negative failures of Kinematics and Dynamics Library (KDL)'s IK implementation is considered. An IK method with a variety of open source algorithms is presented to generate a better IK solution, without those failures. In [12], the authors present a method to find accurate solutions for high DOF robots using a hybrid evolutionary approach. A variety of shape trajectory control methods can be used to find an analytical IK solution of the redundant or hyper-redundant manipulators. These methods include the classical planar curve called tractrix [13], the shape trajectory of rattlesnakes [14], the mechanics-based model [15], the plain and extended spline [16], the elastic backbone [17], the physical curves [18], the backbone curves [19], and the inchworm steps [20].

To find collision-free IK solutions of redundant and soft manipulators, the method proposed in [21] is based on the natural cyclic coordinate descent (CCD) to solve this problem. In [22], the authors explore three algorithms for task space constrained of redundant robots by Tangent Space Sampling and First-Order Retraction. The algorithm described in [23] combines the closed-loop pseudo-inverse (CLP) method with a multi-objective genetic algorithm (GA) for the trajectory planning of redundant robots. Other techniques are based on algebraic solvers [24]. None of these methods consider collision-free constraints or dynamic obstacles.

Many researches have been focused on effective methods to compute collision-free IK solutions for multiple robots in a shared workspace. One of the early developments to solve this issue is the VO [25–27]. In [28], a new concept named reciprocal velocity obstacle (RVO) is proposed for collision-free and oscillation-free navigation among dynamic obstacles in the same environment. Besides, the RVO algorithm has been extended to deal with bounds on acceleration [29] or flying robots in 3-D workspace [30]. Other VO-based methods have been generalized to apply to handle the dynamic constraints. These include multiple differential-drive robots [31], double or arbitrary integrator [32, 33], car-like robots [34], linear quadratic regulator (LQR) controllers [35], non-linear equations of motion [36], etc. As in [32], a n-vehicle collision avoidance strategy is presented that can account for delays and higher order dynamics. However, these methods also did not consider the IK problem for redundant manipulators in a dynamic environment.

3 Specifying Constraints

3.1 Mathematical Statement

A redundant manipulator is considered as consisting of a finite number of rigid-links, which is a concatenation of n identical modules. Let $\xi \in \mathbb{R}^c$ represents the constraint variable to be handled. Let $\boldsymbol{q} = [q_1, \ldots, q_n]$ be the joint configuration refer to a vector of the joint variable. Moreover, it can be assumed that the constraint ξ can be derived from the joint configuration $\boldsymbol{q}$ by function

$$\xi = f(\boldsymbol{q}). \tag{1}$$

For redundant systems (i.e., when $n > c$), it is especially useful to add other constraints, which can be used to keep the manipulator from restricted region of the joint configuration. For simplicity, an additional constraint function can be expressed as

$$\psi \triangleq \psi(\boldsymbol{q}) \leq 0. \tag{2}$$

Now considering the functional form of those constraints for the manipulator, one must solve the problem with equality constraints, gives

$$\begin{array}{l} \text{Given constraint functions } \xi \text{ and } \psi \\ \qquad \text{Compute } \boldsymbol{q} \text{ subject to:} \\ \qquad\qquad \xi = f(\boldsymbol{q}) \text{ and} \\ \qquad\qquad\quad \psi \leq 0. \end{array} \tag{3}$$

The following subsections will describe the definition of many common constraints including task space, joint limit and collision-free constraints.

3.2 Task Space and Joint Limit Constraints

The forward kinematics of a redundant manipulator is described by the equation

$$\xi^E = f_E(\boldsymbol{q}), \tag{4}$$

where $\xi^E \in \mathbb{R}^6$ is the end-effector frame. Let $\mathbf{T}$ represents the rigid transformation between two frames. Then, the matrix $\mathbf{T}_i^{i-1}$ is the transformation from link coordinate frame ξ^i to frame $\xi^{i-1} (i = 1, \ldots, n)$. The matrix $\mathbf{T}_E^0$ is also employed for translations and rotations of the end-effector frame ξ^E with respect to the base frame ξ^0. For any joint configuration $\boldsymbol{q}$, the homogeneous transformations can be used to compute the individual link transformation matrices of an n-DOF manipulator:

$$\xi^E \sim \mathbf{T}_E^0(\boldsymbol{q}) = \mathbf{T}_1^0\mathbf{T}_2^1 \cdots \mathbf{T}_n^{n-1}\mathbf{T}_E^n. \tag{5}$$

The task constraints of a manipulator can be described by a restriction on the motion freedom of the end-effector [37]. Giving the desired position and orientation of the end-effector $\mathbf{T}_t^0$ (transformation from the task frame ξ^t to the base frame ξ^0), the algorithm needs to compute a joint configuration $\boldsymbol{q}$, which satisfies task constraints based on the forward kinematics. The functional form of the task constraints is given by

$$\mathbf{T}_E^t(\boldsymbol{q}) = (\mathbf{T}_t^0)^{-1}\mathbf{T}_E^0(\boldsymbol{q}), \tag{6}$$

where the matrix $\mathbf{T}_E^t$ is the transformation from frame ξ^t to frame ξ^E. Then, the positioning error $\boldsymbol{e} \in \mathbb{R}^6$ between frame ξ^t and frame ξ^E is defined as

$$\boldsymbol{e} \equiv \mathbf{T}_E^t(\boldsymbol{q}). \tag{7}$$

In addition, the joint limit constraints for the joint position q_i can be defined as

$$q_{i,min} \leq q_i \leq q_{i,max}. \tag{8}$$

3.3 Collision-free Constraints

The collision-free constraints begin with decomposing each movable link of the kinematic chain into a series of spheres. The number of spheres and their relative frames and radii are dynamically determined by the size of the links to make sure that there are no collisions. The bounding sphere is used to generate a micromanipulator dynamic model based on the specifications of the manipulator. Moreover, each dynamic obstacle in the environment is also assumed to be a sphere or a series of spheres. In practice, our collision avoidance method tends to be conservative because of these bounding sphere approximations.

In the Cartesian space, let S_m ($m = 1, \ldots, M$, where M is the number of spheres used for this manipulator's decomposition) be one of the spheres on link i with radius r_m, and let A be an obstacle with radii r_A. The sphere frame ξ^{S_m} and the obstacle frame ξ^A are derived from the base frame ξ^0, based on the rigid transformations of the base axes. The matrixes $\mathbf{T}_{S_m}^0$ and $\mathbf{T}_A^0$ specify the position and orientation of ξ^{S_m} and ξ^A, with respect to ξ^0. In a micromanipulator dynamic model, all spheres are rigidly attached to the manipulator after motion. Let $\mathbf{T}_{S_m}^i$ represents the position and orientation of ξ^{S_m} with respect to ξ^i, the transform $\mathbf{T}_{S_m}^0$ is given as follows:

$$\mathbf{T}_{S_m}^0 = \mathbf{T}_i^0\mathbf{T}_{S_m}^i = \mathbf{T}_1^0 \cdots \mathbf{T}_i^{i-1}\mathbf{T}_{S_m}^i. \tag{9}$$

During the iteration process (Algorithms 1 and 2 in Section 4), the matrix $\mathbf{T}_{S_m}^0$ transforms into a new matrix $\mathbf{T}_{S_{m'}}^0$, which represents the position and orientation of frame ξ^0 with respect to frame $\xi^{S_{m'}}$. The transform $\mathbf{T}^{S_{m'}}_{S_m}$ is given as follows:

$$\mathbf{T}^{S_{m'}}_{S_m} = \mathbf{T}^{S_{m'}}_{0}\mathbf{T}^{0}_{S_m} = \left(\mathbf{T}^{0}_{S_{m'}}\right)^{-1}\mathbf{T}^{0}_{S_m}. \tag{10}$$

As shown in Fig. 1, the distance d_{Am} specifies the Euclidean distance between the obstacle frame ξ^A and the sphere frame ξ^{S_m}. If $d_{Am} \le r_m + r_A$, it can be deduced that sphere S_m and sphere A are colliding. The Minkowski sum of these spheres S_m and A can be described by the equation

$$S_m \oplus A = \{s_m + a | s_m \in CH(S_m), a \in CH(A)\}, \tag{11}$$

where $CH(S_m)$ and $CH(A)$ are the convex hulls of sphere S_m and sphere A, respectively. The VO for sphere S_m induced by sphere A for time window τ is shown in Fig. 2. It can be given as

$$VO^{\tau}_{S_m|A} = \{\boldsymbol{v} | \lambda^{\tau}(\boldsymbol{p}_m, \boldsymbol{v} - \boldsymbol{v}_A) \cap A \oplus -S_m \neq \emptyset\}, \tag{12}$$

where $-S_m = \{-s_m | s_m \in CH(S_m)\}$, $\boldsymbol{v}_A$ is the velocity vector of sphere A, and $\lambda^{\tau}(\boldsymbol{p}_m, \boldsymbol{v} - \boldsymbol{v}_A)$ is a ray starting at the position of $\mathbf{T}^0_{S_m}$ with direction $\boldsymbol{v}$,

$$\lambda^{\tau}(\boldsymbol{p}_m, \boldsymbol{v} - \boldsymbol{v}_A) = \{\boldsymbol{p}_m + t(\boldsymbol{v} - \boldsymbol{v}_A) | t \in [0, \tau]\}. \tag{13}$$

As shown in Fig. 3, if sphere S_m has velocity $\boldsymbol{v}_m$, it can be observed that sphere S_m may collide with sphere A during the time interval $[0, \tau]$ if the relative velocity vector of $\boldsymbol{v}_m - \boldsymbol{v}_A$ is inside the region $VO^{\tau}_{S_m|A}$. To avoid a possible collision before time τ, $\boldsymbol{v}_m - \boldsymbol{v}_A$ must be outside $VO^{\tau}_{S_m|A}$. To select a velocity $\boldsymbol{v}_m$, we use a constraint defined with respect to $\boldsymbol{v}_m - \boldsymbol{v}_A$ and $VO^{\tau}_{S_m|A}$. The vector, $\boldsymbol{\omega}_m$, from $\boldsymbol{v}_m - \boldsymbol{v}_A$ to the closest point of the boundary of $VO^{\tau}_{S_m|A}$ is defined as follows:

$$\boldsymbol{\omega}_m = (\arg\min ||\boldsymbol{v} - (\boldsymbol{v}_m - \boldsymbol{v}_A)||_2) - (\boldsymbol{v}_m - \boldsymbol{v}_A),\ \boldsymbol{v} \in \partial VO^{\tau}_{S_m|A}, \tag{14}$$

where $\partial VO^{\tau}_{S_m|A}$ is the boundary of $VO^{\tau}_{S_m|A}$. Let $\boldsymbol{\varphi}_m$ represents the vector from point $(\boldsymbol{p}_m - \boldsymbol{p}_A)/\tau$ to $\boldsymbol{v}_m - \boldsymbol{v}_A$.

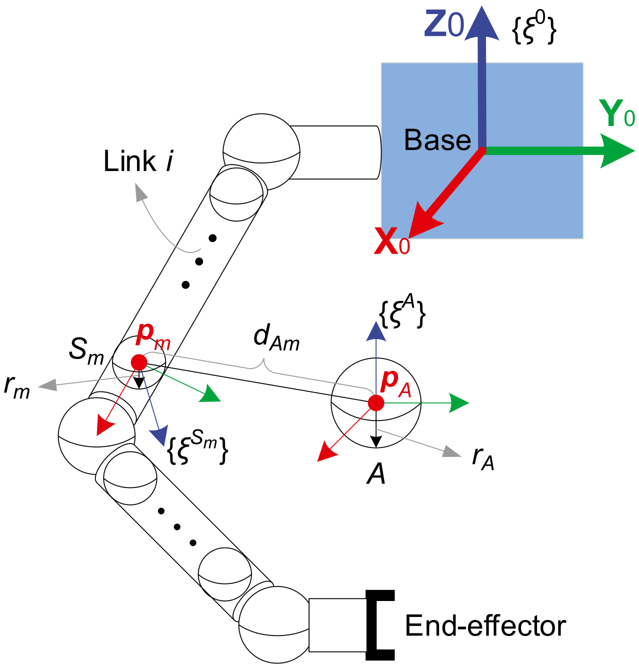


Fig. 1 In the XYZ-dimensional, one of the spheres on the link i and a dynamic obstacle are bounded by spheres whose transforms are $\mathbf{T}^0_{S_m}$ and $\mathbf{T}^0_A$ and radii are r_m and r_A, respectively

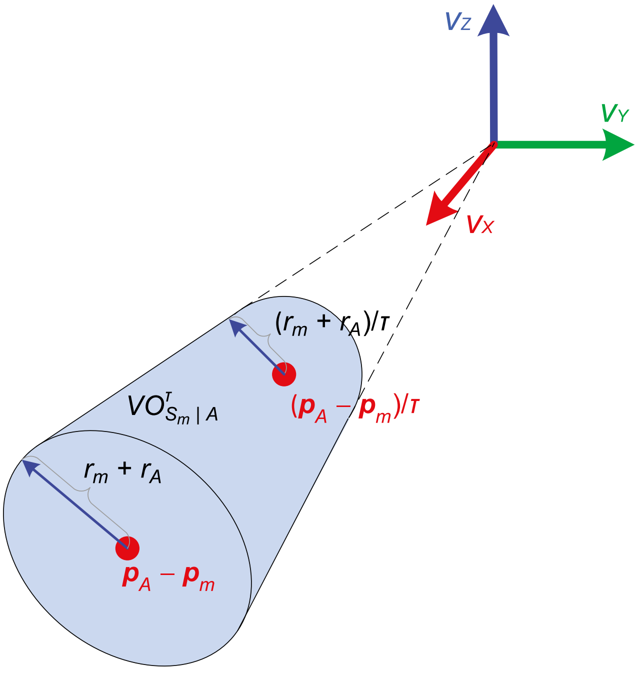


Fig. 2 In the $v_X v_Y v_Z$-dimensional, the shaded area represents the velocity obstacles for sphere A induced by sphere S_m in time window τ

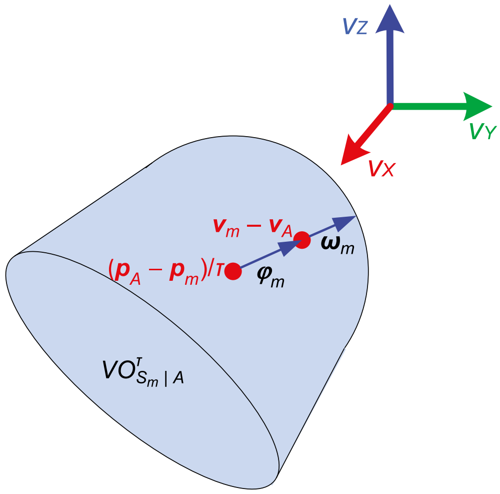


Fig. 3 In the $v_X v_Y v_Z$-dimensional, the vector $\boldsymbol{\varphi}_m$ determines if there is a collision, and the vector $\boldsymbol{\omega}_m$ informally represents the amount of load the sphere takes to avoid collisions

As with previous formulations, the constraint factor ψ_m is defined as follows:

$$\psi_m = \frac{\boldsymbol{\varphi}_m}{\|\boldsymbol{\varphi}_m\|} \cdot \boldsymbol{\omega}_m. \tag{15}$$

If $\psi_m > 0$, then sphere S_m will collide with sphere A. If $\psi_m \leq 0$, there will be no such collision between those two spheres during the time interval $[0, \tau]$.

4 Algorithmic Solution

4.1 Relating IK Solutions to Task Space and Joint Limit Constraints

As introduced in Section 3.2, the positioning error $\boldsymbol{e}$ is computed using the forward kinematics of a redundant manipulator.

For each given task frame ξ^t (i.e., the task space constraints), the analytical solution to IK problem

$$\arg\min \|\boldsymbol{e}\|_2^2 \tag{16}$$

is the set defined as

$$Q \triangleq \left\{\boldsymbol{q}^* \in \mathbb{R}^n : \xi^t = f_E\left(\boldsymbol{q}^*\right)\right\}. \tag{17}$$

Due to kinematic redundancy, let us consider a numerical method to generate valid IK solutions. We use the Newton-based convergence algorithm, which applying a singular value decomposition (SVD) to compute the pseudoinverse of the Jacobian $\mathbf{J}^\dagger$ and reducing the task space constraints by iterating the function

$$\boldsymbol{q}^{k+1} = \boldsymbol{q}^k + \Delta\boldsymbol{q}^k, \tag{18}$$

where $\boldsymbol{q}^{k+1}$ is the joint configuration at step $k+1$, $\boldsymbol{q}^k$ is the previous joint configuration. The most common values of a seed joint configuration $\boldsymbol{q}^0$ for this algorithm are the current joint values. $\Delta\boldsymbol{q}^k$ is the joint state variation that can be used to map $\|\boldsymbol{e}\|_2^2$ into a stopping criteria ε, and it is defined as

$$\Delta\boldsymbol{q}^k = \mathbf{J}^\dagger\boldsymbol{e}. \tag{19}$$

Using the SVD method, $\mathbf{J}^\dagger \in \mathbb{R}^{6\times n}$ can be expressed as follow:

$$\mathbf{J}^\dagger = \mathbf{U}\boldsymbol{\Sigma}^\dagger\mathbf{V}^T. \tag{20}$$

More specifically, $\mathbf{U}$ and $\mathbf{V}$ are the $n \times n$ and 6×6 unitary orthogonal matrix, respectively. $\boldsymbol{\Sigma}$ is an $6 \times n$ rectangular diagonal matrix [38]. When the redundant manipulator gets close to singularities, the joint state variation $\Delta\boldsymbol{q}^k$ will be very large. To prevent the redundant manipulator getting stuck in this situation, we discard the joint configuration $\boldsymbol{q}^k$ and find a better initial joint configuration through a random joint values over a standard uniform distribution number on the open interval $(q_{i,min}, q_{i,max})$.

In addition to keep joint values with in the valid range (i.e., joint limits) in each iteration, it is recommended to add a second adjustment to Problem 16.

The functional form of Problem 16 with joint limit constraints is

$$\begin{aligned} &\arg\min && \|\boldsymbol{e}\|_2^2 \\ &\ \ s.t. && q_{i,min} \leq q_i \leq q_{i,max}. \end{aligned} \tag{21}$$

Furthermore, the set of the analytical solution to Problem 21 can be defined similar to before as

$$Q \triangleq \left\{\boldsymbol{q}^* \in \mathbb{R}^n : \xi^t = f_E\left(\boldsymbol{q}^*\right),\ q_{i,min} \leq q_i^* \leq q_{i,max}\right\}. \tag{22}$$

The approach of adjust the joint values by the joint limit constraints is summarized in Table 1.

As a baseline, the Pseudo-code for the Newton-based convergence algorithm is presented in Algorithm 1. Moreover, Algorithm 1 terminates after $\|\boldsymbol{e}\|_2^2 \leq \varepsilon$ or a maximum computation time t_{max}, see line 3. If the transforms $\mathbf{T}_t^0$ and $\mathbf{T}_E^0(\boldsymbol{q})$ are found, $\boldsymbol{e}$ can be computed using Eq. 7 within function COMPUTE_ERROR(), see lines 2 and 9. $\mathbf{J}^\dagger$ can be computed using Eq. 20 within function JACOBIAN_INVERSE(), see line 5.

4.2 Relating IK Solutions to Collision-free Constraints

Algorithm 1 can compute the joint configuration $\boldsymbol{q}^{k+1}$ and positioning error $\boldsymbol{e}$ after each iteration. To find a collision-free IK solution, our method continues to update the joint configuration if $\psi_m(\boldsymbol{q}^k) > 0$. The constraint factor ψ_m can be computed using the joint configuration $\boldsymbol{q}^{k+1}$ and $\boldsymbol{q}^0$.

Table 1 Joint limit constraints

```
JOINT_LIMIT_CONSTRAINT ($\boldsymbol{q}$);

for $i = 1$ to $n$
if $q_i \leq q_{i,min}$
    $q_i^r \leftarrow *\mathrm{REMAINDER}*\,(\frac{q_i - q_{i,min}}{2\pi})$
    $q_i = q_{i,min} - q_i^r + 2\pi$
     if $q_i \geq q_{i,max}$
        then $q_i = q_{i,min}$
else if $q_i \geq q_{i,max}$
    $q_i^r \leftarrow \mathrm{REMAINDER}(\frac{q_{i,max} - q_i}{2\pi})$
    $q_i = q_{i,max} + q_i^r - 2\pi$
     if $q_i \leq q_{i,min}$
        then $q_i = q_{i,max}$
return $\boldsymbol{q}$
```

[1] The word *REMAINDER* represents it will return the floating-point remainder of division

Algorithm 1 Newton-based convergence.

```
Input: $q^0 = currentjoint, k = 0, t_{left} = 0$.
Output: IK solution of the redundant manipulator.
Initialize: $t_{max} \leftarrow$ GET_MAXTIME().
Initialize: $e \leftarrow$ COMPUTE_ERROR($\mathbf{T}_E^t(q^0)$).
while $t_{left} \leq t_{max}$ and $\|e\|_2^2 > \varepsilon$ do
    $k \leftarrow k + 1$
    $\mathbf{J}^\dagger \leftarrow$ JACOBIAN_INVERSE($q^k$)
    $\Delta q^k = \mathbf{J}^\dagger e$
    $q^{k+1} = q^k + \Delta q^k$
    $q^{k+1} \leftarrow$ JOINT_LIMIT_CONSTRAINT($q^{k+1}$)
    $e \leftarrow$ COMPUTE_ERROR($\mathbf{T}_E^t(q^{k+1})$)
    if $\|e\|_2^2 \leq \varepsilon$ then
        return $q^{k+1}$
    else
        $t_{left} \leftarrow t_{left} +$ GET_DURATION()
    end if
end while
return $q^0$
```

Then, a secondary constraint is formulated as a gradient of ψ_m, which is used to find a better joint configuration for the IK algorithm to perform obstacle avoidance.

Now, Problem 21 can be rewritten as

$$\begin{aligned} &\arg\min && \|e\|_2^2 \\ &s.t. && \begin{cases} q_{i,min} \leq q_i \leq q_{i,max} \\ \psi_m \leq 0 \end{cases} \end{aligned} \tag{23}$$

With this definition, it is possible to generate the set of the analytical solution, which is the deterministic function of Problem 23, gives

$$\begin{aligned} Q \triangleq \{ & q^* \in \mathbb{R}^n : \xi^t = f_E\left(q^*\right), \ q_{i,min} \leq q_i^* \\ & \leq q_{i,max}, \ \psi_m\left(q^*\right) \leq 0 \}. \end{aligned} \tag{24}$$

Since the relationship between ψ_m and q is nonlinear, the first-order iterative optimization algorithm is used to make ψ_m converges to the minimum. Let $\frac{\partial \psi_m}{\partial x_m}$ represents the function of differentials of x_m. Then, an associate function is given as

$$G(p_m) = \left[\frac{\partial \psi_m}{\partial x_m}, \frac{\partial \psi_m}{\partial y_m}, \frac{\partial \psi_m}{\partial z_m} \right]^T. \tag{25}$$

Note that $p_m = [x_m, y_m, x_m]^T$ presents the position vector of frame ξ^{S_m}, which is the translation vector in homogeneous transform $\mathbf{T}_{S_m}^0(q_I)$, where $q_I = [q_1, \cdots, q_i]$. Let $\mathbf{J}_p^m \in \mathbb{R}^{3\times i}$ represents the linear velocity Jacobian of the frame ξ^{S_m}, which can be computed as follow:

$$\mathbf{J}_p^m = \begin{bmatrix} \frac{\partial x_m}{\partial q_1} & \cdots & \frac{\partial x_m}{\partial q_i} \\ \frac{\partial y_m}{\partial q_1} & \cdots & \frac{\partial y_m}{\partial q_i} \\ \frac{\partial z_m}{\partial q_1} & \cdots & \frac{\partial z_m}{\partial q_i} \end{bmatrix}. \tag{26}$$

Given the definitions of Eqs, 25 and 26, the equation of gradient $\nabla\psi_m(q_I)$ is given as follows:

$$\nabla\psi_m(q_I) = \left[\frac{\partial \psi_m}{\partial q_1}, ..., \frac{\partial \psi_m}{\partial q_i} \right]^T = \left(\mathbf{J}_p^m(q_I)\right)^T G(p_m). \tag{27}$$

Then, the first-order iterative optimization algorithm is applied by iterating the function

$$q_I^{k+1} = q_I^k - \gamma^k \nabla\psi_m(q_I^k), \tag{28}$$

where γ^k is the step size. In the spirit of Eq. 18, a transformation from $\gamma^k\nabla\psi_m(q_I^k) \in \mathbb{R}^i$ to $\gamma^k\nabla\hat{\psi}_m(q_I^k) \in \mathbb{R}^n$ is provided as

$$\begin{aligned} \gamma^k\nabla\psi_m(q_I^k) &\hat{=} \gamma^k\nabla\hat{\psi}_m(q_I^k) \\ &= [\underbrace{\gamma^k\nabla\psi_m(q_1^k), \ldots, \gamma^k\nabla\psi_m(q_i^k)}_{i}, \underbrace{0, \ldots, 0}_{n-i}]. \end{aligned} \tag{29}$$

For this transformation, the gradient direction $\gamma^k\nabla\hat{\psi}_m(q_I^k)$ and the joint configuration q have the same dimensional $\mathbb{R}^n$. Then Eq. 18 can be rewritten as

$$q^{k+1} = q^k + \Delta q^k - \gamma^k\nabla\hat{\psi}_m(q_I^k). \tag{30}$$

4.3 CFIK Algorithm

Given the Eq. 30, the Pseudo-code for our CFIK algorithm can be given as Algorithm 2. Similar to Algorithm 1, CFIK algorithm terminates after $\|e\|_2^2 \leq \varepsilon$ and $\psi_m \leq 0$ or a maximum computation time t_{max}, see lines 4 and 17. ψ_m is computed using Eq. 15 within function COLLISION-FREE_CONSTRAINT(), see lines 3 and 16. The gradient direction $\gamma^k\nabla\hat{\psi}_m(q_I^k)$ can be computed using Eq. 29 within function COMPUTE_GRADIENT_DIRECTION(), see line 11.

4.4 Avoiding Collisions with Static Obstacles

A typical robot working environment also contain static obstacles, such as wall. Therefore, the same method as above will be applied to this situation, it would compute the velocity obstacle for sphere to avoid collisions with static obstacles. The geometric shape of the static obstacle is assumed to be a regularly shaped body, in terms of the cylinder and cuboid. As can be seen in Fig. 4a, the wall obstacle is modeled as a cuboid. Let A be the cuboid, and S_m is a sphere centered at p_m with radius r_m. Then, the velocity obstacle $VO_{S_m|A}^\tau$ can be defined as

$$VO_{S_m|A}^\tau = \{v | \lambda^\tau(p_m, v) \cap A \oplus -S_m \neq \emptyset\}, \tag{31}$$

as shown in Fig. 4b and c. If the velocity v_m is inside, sphere S_m will collide with the cuboid A within time τ. The vector ω_m is redefined as

$$\omega_m = (\arg\min ||v - v_m||_2) - v_m, v \in \partial VO_{S_m|A}^\tau, \tag{32}$$

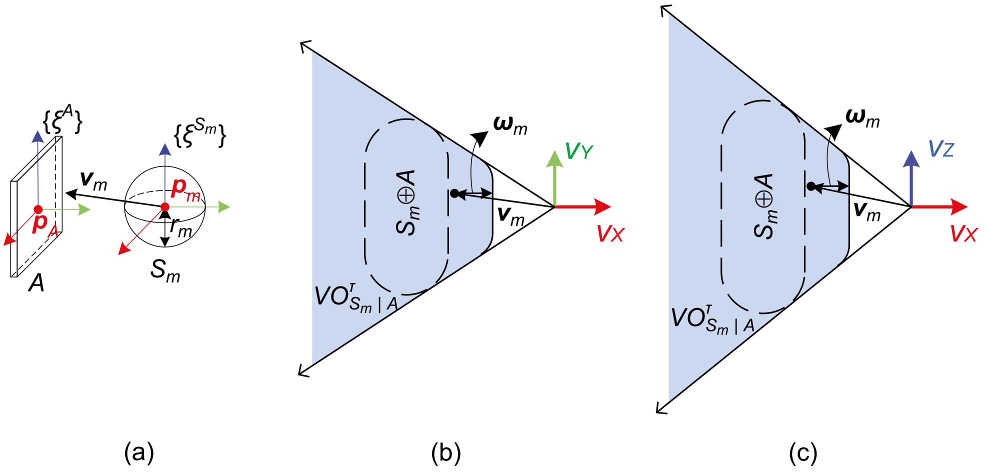


Fig. 4 **a** In the XYZ-dimensional, a sphere S_m and a static obstacle A. **b**-**c** The shaded areas represent the geometric construction of $VO^{\tau}_{S_m|A}$. The vector $\boldsymbol{\omega}_m$ is the amount of load the sphere S_m takes to avoid static obstacle A with respect to $\boldsymbol{v}_m$

then this vector will be used in the collision avoidance approach of the CFIK algorithm to redundant manipulators.

5 Convergence Rate and Stability Analysis

5.1 Convergence Rate

In this section, the convergence rate of the collision-free constraints is illustrated to ensure convergence of the defined scenarios when using our method. Equation 28 shows the update rule of the joint configuration $\boldsymbol{q}$ at each iteration to keep ψ_m moving towards the minimum. To demonstrate the function ψ_m has a convergence rate, we first show that ψ_m is Lipschitz continuous and β smooth [39]. For sake of simplicity, the convergence rate analysis will be discussed with respect to the condition of ψ_m as shown in Fig. 3.

Lemma 5.1 *The function ψ_m is Lipschitz continuous.*

Proof Equation 15 defines the function ψ_m. For any two joint configuration $\boldsymbol{q}_I^1$ and $\boldsymbol{q}_I^2$, we define

$$\begin{aligned}&\left|\psi_m(\boldsymbol{q}_I^1) - \psi_m(\boldsymbol{q}_I^2)\right| \\ &= \left|\frac{\boldsymbol{\varphi}_m(\boldsymbol{q}_I^1)}{\|\boldsymbol{\varphi}_m(\boldsymbol{q}_I^1)\|}\boldsymbol{\omega}_m(\boldsymbol{q}_I^1) - \frac{\boldsymbol{\varphi}_m(\boldsymbol{q}_I^2)}{\|\boldsymbol{\varphi}_m(\boldsymbol{q}_I^2)\|}\boldsymbol{\omega}_m(\boldsymbol{q}_I^2)\right|. \end{aligned} \tag{33}$$

The values of $\boldsymbol{\varphi}_m$ and $\boldsymbol{\omega}_m$, can be computed according to Eq. 11–14.

$$\boldsymbol{\varphi}_m = (\boldsymbol{v}_m - \boldsymbol{v}_A) - \left(\frac{\boldsymbol{p}_m + \boldsymbol{p}_A}{\tau}\right), \tag{34}$$

$$\boldsymbol{\omega}_m = \left(\frac{r_m + r_A}{\tau} - \|\boldsymbol{\varphi}_m\|\right)\frac{\boldsymbol{\varphi}_m}{\|\boldsymbol{\varphi}_m\|}. \tag{35}$$

Substituting (34) and (35) into Eq. 33 gives

$$\begin{aligned}&\left|\psi_m(\boldsymbol{q}_I^1) - \psi_m(\boldsymbol{q}_I^2)\right| \\ &= \left|\tfrac{\boldsymbol{\varphi}_m(\boldsymbol{q}_I^1)}{\|\boldsymbol{\varphi}_m(\boldsymbol{q}_I^1)\|}\left(\tfrac{r_m+r_A}{\tau} - \|\boldsymbol{\varphi}_m(\boldsymbol{q}_I^1)\|\right)\tfrac{\boldsymbol{\varphi}_m(\boldsymbol{q}_I^1)}{\|\boldsymbol{\varphi}_m(\boldsymbol{q}_I^1)\|}\right. \\ &\quad \left. - \tfrac{\boldsymbol{\varphi}_m(\boldsymbol{q}_I^2)}{\|\boldsymbol{\varphi}_m(\boldsymbol{q}_I^2)\|}\left(\tfrac{r_m+r_A}{\tau} - \|\boldsymbol{\varphi}_m(\boldsymbol{q}_I^2)\|\right)\tfrac{\boldsymbol{\varphi}_m(\boldsymbol{q}_I^2)}{\|\boldsymbol{\varphi}_m(\boldsymbol{q}_I^2)\|}\right| \\ &= \left|\|\boldsymbol{\varphi}_m(\boldsymbol{q}_I^2)\| - \|\boldsymbol{\varphi}_m(\boldsymbol{q}_I^1)\|\right| \\ &\leq \left\|\left(\boldsymbol{v}_m^{\boldsymbol{q}_I^2} - \boldsymbol{v}_A\right) - \left(\tfrac{\boldsymbol{p}_m(\boldsymbol{q}_I^2)+\boldsymbol{p}_A}{\tau}\right) - \left[\left(\boldsymbol{v}_m^{\boldsymbol{q}_I^1} - \boldsymbol{v}_A\right)\right.\right. \\ &\quad \left.\left. - \left(\tfrac{\boldsymbol{p}_m(\boldsymbol{q}_I^1)+\boldsymbol{p}_A}{\tau}\right)\right]\right\| \\ &= \left\|\boldsymbol{v}_m^{\boldsymbol{q}_I^2} - \boldsymbol{v}_m^{\boldsymbol{q}_I^1} - \tfrac{\boldsymbol{p}_m(\boldsymbol{q}_I^2)-\boldsymbol{p}_m(\boldsymbol{q}_I^1)}{\tau}\right\|. \end{aligned} \tag{36}$$

□

To construct the VO, the sphere velocity vector is obtained in each iteration, as a function of time Δt, through the translation vector $\boldsymbol{p}_m(\boldsymbol{q}_I)$ in transform $\mathbf{T}^0_{S_m}(\boldsymbol{q}_I)$

$$\boldsymbol{v}_m^{k+1} = \begin{bmatrix} v_{m_x}^{k+1} \\ v_{m_y}^{k+1} \\ v_{m_z}^{k+1} \end{bmatrix} = \frac{\boldsymbol{p}_m(\boldsymbol{q}_I^{k+1}) - \boldsymbol{p}_m(\boldsymbol{q}_I^0)}{\Delta t} = \begin{bmatrix} \frac{x_m^{k+1} - x_m^0}{\Delta t} \\ \frac{y_m^{k+1} - y_m^0}{\Delta t} \\ \frac{z_m^{k+1} - z_m^0}{\Delta t} \end{bmatrix}. \tag{37}$$

Consider Δt is the time duration of the end-effector moving between the two Cartesian poses ξ^E and ξ^t at one movement step. For simplicity, it can be assumed that the end-effector moves at constant velocity in our experiments. Substituting the velocity of sphere S_m in Eq. 37 into Eq. 36, can be rearranged as

$$\begin{aligned}&\left|\psi_m(\boldsymbol{q}_I^1) - \psi_m(\boldsymbol{q}_I^2)\right| \\ &\leq \left\|\tfrac{\boldsymbol{p}_m(\boldsymbol{q}_I^2)-\boldsymbol{p}_m(\boldsymbol{q}_I^0)}{\Delta t} - \tfrac{\boldsymbol{p}_m(\boldsymbol{q}_I^1)-\boldsymbol{p}_m(\boldsymbol{q}_I^0)}{\Delta t} - \tfrac{\boldsymbol{p}_m(\boldsymbol{q}_I^2)-\boldsymbol{p}_m(\boldsymbol{q}_I^1)}{\tau}\right\| \\ &= \left(\tfrac{1}{\Delta t} - \tfrac{1}{\tau}\right)\left\|\boldsymbol{p}_m(\boldsymbol{q}_I^2) - \boldsymbol{p}_m(\boldsymbol{q}_I^1)\right\|. \end{aligned} \tag{38}$$

Let us assume the transformation between joint configurations $\boldsymbol{q}_I$ and the position vector $\boldsymbol{p}_m$ of frame ξ_m is locally

Algorithm 2 Collision-Free IK Solution Generator (our novel CFIK algorithm that accounts for task space, joint limit and collision avoidance constraints).

```
Input: $q^0 = currentjoint, k = 0, t_{left} = 0$.
Output: Collision-Free IK solution of the redundant manipulator.
Initialize: $t_{max} \leftarrow$ GET_MAXTIME().
Initialize: $\boldsymbol{e} \leftarrow$ COMPUTE_ERROR($\mathbf{T}^t_E(\boldsymbol{q}^0)$).
Initialize: $\psi_m(\boldsymbol{q}^0) \leftarrow$COLLISION-FREE_CONSTRAINT $\boldsymbol{q}^0$.
while $t_{left} \le t_{max}$ do
    $k \leftarrow k + 1$
    if $\|\boldsymbol{e}\|_2^2 > \varepsilon$ then
        $\mathbf{J}^{\dagger} \leftarrow$ JACOBIAN_INVERSE($\boldsymbol{q}^k$)
        $\Delta \boldsymbol{q}^k = \mathbf{J}^{\dagger}\boldsymbol{e}$
    end if
    if $\psi_m(\boldsymbol{q}^k) > 0$ then
        $\gamma^k \nabla\hat{\psi}_m(\boldsymbol{q}_I^k) \leftarrow$ COMPUTE_GRADIENT_DIRECTION($\boldsymbol{q}_I^k$)
    end if
    $\boldsymbol{q}^{k+1} = \boldsymbol{q}^k + \Delta \boldsymbol{q}^k - \gamma^k \nabla\hat{\psi}_m(\boldsymbol{q}_I^k)$
    $\boldsymbol{q}^{k+1} \leftarrow$ JOINT_LIMIT_CONSTRAINT($\boldsymbol{q}^{k+1}$)
    $\boldsymbol{e} \leftarrow$ COMPUTE_ERROR($\mathbf{T}^t_E(\boldsymbol{q}^{k+1})$)
    $\psi_m(\boldsymbol{q}^{k+1}) \leftarrow$ COLLISION-FREE_CONSTRAINT($\boldsymbol{q}^{k+1}$)
    if $\|\boldsymbol{e}\|_2^2 \le \varepsilon$ and $\psi_m(\boldsymbol{q}^{k+1}) \le 0$ then
        return $\boldsymbol{q}^{k+1}$
    else
        $t_{left} \leftarrow t_{left} +$ GET_DURATION()
    end if
end while
return $\boldsymbol{q}^0$
```

linear; in other words, the Jacobian $\mathbf{J}_p^m$ is a constant for those two configurations $\boldsymbol{q}_I^1$ and $\boldsymbol{q}_I^2$, leading to the following linear relationship

$$\boldsymbol{p}_m(\boldsymbol{q}_I^1) = \mathbf{J}_p^m \boldsymbol{q}_I^1 \tag{39}$$

and

$$\boldsymbol{p}_m(\boldsymbol{q}_I^2) = \mathbf{J}_p^m \boldsymbol{q}_I^2. \tag{40}$$

Equation 38 can be rewritten as

$$\begin{aligned}\left|\psi_m(\boldsymbol{q}_I^1) - \psi_m(\boldsymbol{q}_I^2)\right| &\le \left(\tfrac{1}{\Delta t} - \tfrac{1}{\tau}\right)\left\|\mathbf{J}_p^m\left(\boldsymbol{q}_I^2 - \boldsymbol{q}_I^1\right)\right\| \\ &\le \left(\tfrac{1}{\Delta t} - \tfrac{1}{\tau}\right)\left\|\mathbf{J}_p^m\right\|\left\|\boldsymbol{q}_I^2 - \boldsymbol{q}_I^1\right\|.\end{aligned} \tag{41}$$

This proves that the function ψ_m is Lipschitz continuous [40].

Lemma 5.2 *The function ψ_m is β smooth.*

Proof Equation 27 defines the equation of gradient $\nabla\psi_m(\boldsymbol{q}_I^1)$. For any two joint configuration $\boldsymbol{q}_I^1$ and $\boldsymbol{q}_I^2$, gives

$$\begin{aligned}&\left\|\nabla\psi_m(\boldsymbol{q}_I^1) - \nabla\psi_m(\boldsymbol{q}_I^2)\right\| \\ &= \left\|\left(\mathbf{J}_p^m(\boldsymbol{q}_I^1)\right)^T G\left(\boldsymbol{p}_m(\boldsymbol{q}_I^1)\right) - \left(\mathbf{J}_p^m(\boldsymbol{q}_I^2)\right)^T G\left(\boldsymbol{p}_m(\boldsymbol{q}_I^2)\right)\right\|.\end{aligned} \tag{42}$$

According to Eq. 25 we have

$$G\left(\boldsymbol{p}_m\right) = \left[\tfrac{\partial\psi_m}{\partial x_m}, \tfrac{\partial\psi_m}{\partial y_m}, \tfrac{\partial\psi_m}{\partial z_m}\right]^T = \frac{\boldsymbol{\varphi}_m}{\left\|\boldsymbol{\varphi}_m\right\|\Delta t}. \tag{43}$$

Then, in the linear case, Eq. 42 can be rewritten as

$$\begin{aligned}&\left\|\nabla\psi_m(\boldsymbol{q}_I^1) - \nabla\psi_m(\boldsymbol{q}_I^2)\right\| \\ &= \left\|\mathbf{J}_p^m\left(\frac{\boldsymbol{\varphi}_m(\boldsymbol{q}_I^1)}{\|\boldsymbol{\varphi}_m(\boldsymbol{q}_I^1)\|\Delta t} - \frac{\boldsymbol{\varphi}_m(\boldsymbol{q}_I^2)}{\|\boldsymbol{\varphi}_m(\boldsymbol{q}_I^2)\|\Delta t}\right)\right\| \\ &\le \left\|\mathbf{J}_p^m\right\|\left\|\frac{\boldsymbol{\varphi}_m(\boldsymbol{q}_I^1)}{\|\boldsymbol{\varphi}_m(\boldsymbol{q}_I^1)\|\Delta t} - \frac{\boldsymbol{\varphi}_m(\boldsymbol{q}_I^2)}{\|\boldsymbol{\varphi}_m(\boldsymbol{q}_I^2)\|\Delta t}\right\|.\end{aligned} \tag{44}$$

□

Now let $\varphi_{m,min} = \min(\| \ \boldsymbol{\varphi}_m(\boldsymbol{q}_I^1) \ \|, \| \ \boldsymbol{\varphi}_m(\boldsymbol{q}_I^2) \ \|)$. Substituting this in the previous result gives

$$\begin{aligned}&\left\|\nabla\psi_m(\boldsymbol{q}_I^1) - \nabla\psi_m(\boldsymbol{q}_I^2)\right\| \\ &\le \frac{\left\|\mathbf{J}_p^m\right\|}{\varphi_{m,min}\Delta t}\left\|\boldsymbol{\varphi}_m(\boldsymbol{q}_I^1) - \boldsymbol{\varphi}_m(\boldsymbol{q}_I^2)\right\| \\ &= \frac{\left\|\mathbf{J}_p^m\right\|}{\varphi_{m,min}\Delta t}\left\|\boldsymbol{v}_m^{\boldsymbol{q}_I^1} - \boldsymbol{v}_m^{\boldsymbol{q}_I^2} - \frac{\boldsymbol{p}_m(\boldsymbol{q}_I^1) - \boldsymbol{p}_m(\boldsymbol{q}_I^2)}{\tau}\right\| \\ &= \frac{\left\|\mathbf{J}_p^m\right\|}{\varphi_{m,min}\Delta t}\left(\tfrac{1}{\Delta t} - \tfrac{1}{\tau}\right)\left\|\boldsymbol{p}_m(\boldsymbol{q}_I^1) - \boldsymbol{p}_m(\boldsymbol{q}_I^2)\right\| \\ &= \frac{\left\|\mathbf{J}_p^m\right\|^2}{\varphi_{m,min}\Delta t}\left(\tfrac{1}{\Delta t} - \tfrac{1}{\tau}\right)\left\|\boldsymbol{q}_I^1 - \boldsymbol{q}_I^2\right\|,\end{aligned} \tag{45}$$

and it can be concluded that ψ_m is a β-smooth function, where $\beta = \frac{\left\|\mathbf{J}_p^m\right\|^2}{\varphi_{m,min}\Delta t}\left(\frac{1}{\Delta t} - \frac{1}{\tau}\right)$.

Theorem 5.3 *Let ψ_m be a convex and β-smooth function with $\boldsymbol{q}_I$ as the current joint configuration and $\boldsymbol{q}_I^*$ as the optimal joint configuration on $\mathbb{R}^I$. Then $\|\boldsymbol{q}_I^k - \boldsymbol{q}_I^*\|$ decreases with k if $\gamma^k \le 1/\beta$.*

Proof: For the two joint configuration $\boldsymbol{q}_I^1$ and $\boldsymbol{q}_I^2$, we define

$$\begin{aligned}\left\|\boldsymbol{q}_I^{k+1} - \boldsymbol{q}_I^*\right\|^2 &= \left\|\boldsymbol{q}_I^k - \gamma^k\nabla\psi_m(\boldsymbol{q}_I^k) - \boldsymbol{q}_I^*\right\|^2 \\ &= \left\|\boldsymbol{q}_I^k - \boldsymbol{q}_I^*\right\|^2 - 2\gamma^k\nabla\psi_m(\boldsymbol{q}_I^k)^T\left(\boldsymbol{q}_I^k - \boldsymbol{q}_I^*\right) \\ &\quad + \left(\gamma^k\left\|\nabla\psi_m(\boldsymbol{q}_I^k)\right\|\right)^2.\end{aligned} \tag{46}$$

As derived in Lemma 5.2, gives

$$\psi_m(\boldsymbol{q}_I^k) - \psi_m(\boldsymbol{q}_I^*) \le \nabla\psi_m(\boldsymbol{q}_I^k)^T\left(\boldsymbol{q}_I^k - \boldsymbol{q}_I^*\right) - \frac{\left\|\nabla\psi_m(\boldsymbol{q}_I^k)\right\|^2}{2\beta}. \tag{47}$$

And knowing that

$$\psi_m(\boldsymbol{q}_I^k) - \psi_m(\boldsymbol{q}_I^*) \geq 0 \tag{48}$$

and

$$-\nabla\psi_m(\boldsymbol{q}_I^k)^T \left(\boldsymbol{q}_I^k - \boldsymbol{q}_I^*\right) \leq -\frac{\left\|\nabla\psi_m(\boldsymbol{q}_I^k)\right\|^2}{2\beta} \tag{49}$$

in Eq. 47, it can be rewritten as

$$\begin{aligned}\left\|\boldsymbol{q}_I^{k+1} - \boldsymbol{q}_I^*\right\|^2 \leq \left\|\boldsymbol{q}_I^k - \boldsymbol{q}_I^*\right\|^2 - \gamma^k \frac{\left\|\nabla\psi_m(\boldsymbol{q}_I^k)\right\|^2}{\beta} \\ +\left(\gamma^k \left\|\nabla\psi_m(\boldsymbol{q}_I^k)\right\|\right)^2 \leq \left\|\boldsymbol{q}_I^k - \boldsymbol{q}_I^*\right\|^2 \\ -\gamma^k \left(\frac{1}{\beta} - \gamma^k\right) \left\|\nabla\psi_m(\boldsymbol{q}_I^k)\right\|^2.\end{aligned} \tag{50}$$

This means that if $\gamma^k \leq 1/\beta$, then

$$\left\|\boldsymbol{q}_I^{k+1} - \boldsymbol{q}_I^*\right\|^2 \leq \left\|\boldsymbol{q}_I^k - \boldsymbol{q}_I^*\right\|^2. \tag{51}$$

5.2 Stability Analysis

The stability analysis of our CFIK algorithm will be discussed based on solely three tasks of a redundant manipulator(the DH parameters are listed in Appendix A):

1. The position of the end-effector $\xi_p^t \in \mathbb{R}^3$.
2. The position of the wrist $\xi_y^5 \in \mathbb{R}$, along the y axis of the base frame ξ^0.
3. The position of the elbow $\xi_x^4 \in \mathbb{R}$, along the x axis of the base frame ξ^0.

The task errors $\boldsymbol{e} \in \mathbb{R}^{3+1+1}$ is

$$\boldsymbol{e} = \left[\left(\boldsymbol{e}_p^E\right)^T \; \left(\boldsymbol{e}_y^5\right)^T \; \left(\boldsymbol{e}_x^4\right)^T \right]^T, \tag{52}$$

where $\boldsymbol{e}_p^E \in \mathbb{R}^3$, $\boldsymbol{e}_y^5 \in \mathbb{R}$ and $\boldsymbol{e}_x^4 \in \mathbb{R}$ are the task error of three tasks, respectively. There is a dynamic obstacle in the working environment with the position of [0.35, -0.67, 0.7] (in meters), the velocity of [0, 0.05, 0] (in m/s) and the radii of 0.1m. The initial joint configuration of the manipulator is given as

$$\boldsymbol{q} = [-0.18, -1.0, 1.19, 1.94, -0.67, 1.03, 0.50] \text{ (in rads),} \tag{53}$$

The desired positions of three tasks are assigned:

$$\xi_p^E = [0.168867, -0.010499, 0.784523]^T \text{ (in meters),} \tag{54}$$

$$\xi_y^5 = [-0.14] \text{ (in meters),} \tag{55}$$

$$\xi_x^4 = [0.078] \text{ (in meters).} \tag{56}$$

Then, the Lyapunov function candidate is defined as

$$V(\boldsymbol{e}) = \frac{1}{2}\boldsymbol{e}^T\boldsymbol{e} \tag{57}$$

whose time derivative is

$$\dot{V} = -\boldsymbol{e}^T \begin{bmatrix} \mathbf{J}_p^E \\ \mathbf{J}_y^5 \\ \mathbf{J}_x^4 \end{bmatrix} \dot{\boldsymbol{q}}. \tag{58}$$

With Eq. 26 and 30, the corresponding Jacobian $\mathbf{J}_p^E \in \mathbb{R}^{3\times n}$ of the primary task is given by

$$\mathbf{J}_p^E = \mathbf{J}_p^\dagger - \frac{\gamma\nabla\hat{\psi}_m(\boldsymbol{q}_I)}{\boldsymbol{e}_p^E}. \tag{59}$$

The Jacobians $\mathbf{J}_y^5 \in \mathbb{R}^{1\times n}$ and $\mathbf{J}_x^4 \in \mathbb{R}^{1\times n}$ can be easily computed using Eq. 26.

Based on our CFIK algorithm, the Lyapunov function V and the norm error of those three tasks versus the motion time are shown in Fig. 5. It can be seen that the Lyapunov function and the norm errors decrease to zero with the value of the time horizon increases. More specifically, we can take into account some parameters similar to the approach in [41], gives

$$\underline{\lambda}_{22} = \sigma_{\min}\left(\mathbf{J}_y^5 N_p^E \left(\mathbf{J}_y^5\right)^\dagger\right), \tag{60}$$

$$\underline{\lambda}_{33} = \sigma_{\min}\left(\mathbf{J}_x^4 \overline{N} \left(\mathbf{J}_x^4\right)^\dagger\right), \tag{61}$$

$$\lambda_y^5 = \sigma_{\min}\left(\Lambda_y^5\right), \tag{62}$$

$$\lambda_x^4 = \sigma_{\min}\left(\Lambda_x^4\right), \tag{63}$$

in which

$$N_p^E = \mathbf{I} - \left(\mathbf{J}_p^E\right)^\dagger \mathbf{J}_p^E, \; \overline{N} = N_p^E \left(\mathbf{I} - \left(\mathbf{J}_y^5\right)^\dagger \mathbf{J}_y^5\right), \tag{64}$$

$\sigma_{\min}$ ($\sigma_{\max}$) is the smallest (largest) singular value of the matrix, $\mathbf{I}$ is the identity matrix of proper size, Λ_y^5 and Λ_x^4 are the gain matrices of the tasks ξ_y^5 and ξ_x^4, respectively. In

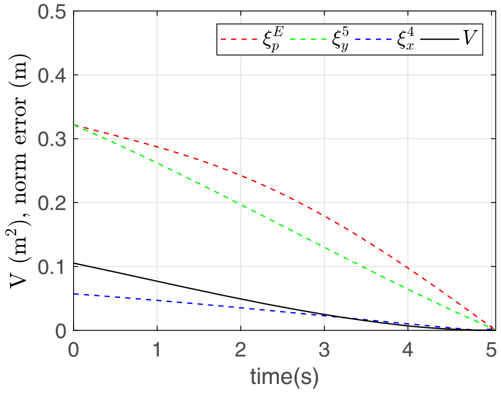
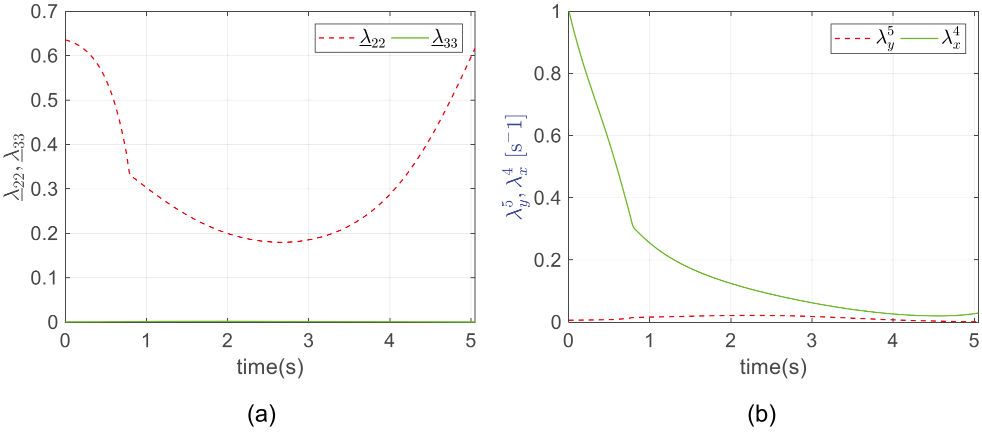


Fig. 5 Time history of the Lyapunov function and the norm error of three tasks

Fig. 6 Time history of $\underline{\lambda}_{22}$, $\underline{\lambda}_{33}$, λ_y^5 and λ_x^4



(a)

(b)

this work, the values of λ_y^5 and λ_x^4 are obtained using the estimates

$$\lambda_y^5 > \frac{\overline{\lambda}_{21}^2}{4\underline{\lambda}_{22}}, \tag{65}$$

$$\lambda_x^4 > \max\left\{\frac{\overline{\lambda}_{31}^2}{4\underline{\lambda}_{33}}, \frac{\overline{\lambda}_{32}^2}{4\underline{\lambda}_{22}\underline{\lambda}_{33}}\right\}, \tag{66}$$

where $\overline{\lambda}_{21} = \sigma_{\max}\left(\mathbf{J}_y^5\left(\mathbf{J}_p^E\right)^\dagger\right)$, $\overline{\lambda}_{31} = \sigma_{\max}\left(\mathbf{J}_x^4\left(\mathbf{J}_p^E\right)^\dagger\right)$, and $\overline{\lambda}_{32} = \sigma_{\max}\left(\mathbf{J}_x^4 N_p^E\left(\mathbf{J}_y^5\right)^\dagger\right)$. Given the definitions of Eqs. 60–66, the values of $\underline{\lambda}_{22}$, $\underline{\lambda}_{33}$, λ_y^5 and λ_x^4 with respect to time are shown in Fig. 6.

6 Results

6.1 Implementation and Performance

This section presents the results from various simulation and physical experiments that highlight the performance of our CFIK algorithm in a various working environments, which include multiple redundant manipulators and dynamic obstacles. The experiments are initialized by decomposing the right arm of Baxter robot into a series of spheres to generate a micromanipulator dynamic model. As shown in Fig. 7, the micromanipulator dynamic model contains five spheres from sphere S_1 (with frame ξ^{S_1}) to sphere S_5 (with frame ξ^{S_5}), and the base coordinate system with frame ξ^0 is fixed to the ground. In addition, the seven revolute joints of the Baxter arm are represented in the following manner: $\boldsymbol{q} = [q_1, \ldots, q_7]$, which parametrize the rotation about axis z_i $(i = 1, \ldots, 7)$ in link coordinate frame ξ^i respectively. The DH parameters of those spheres and the right arm model are listed in Appendix A, which can be used to describe the rigid transforms between successive coordinate systems.

The CFIK algorithm does not take all sphere obstacles into account, as the constraint factor ψ_m is always a negative real number when the distance between two spheres is sufficiently large. Therefore, a neighboring region NR_m around the current position of sphere S_m is defined and we only consider the sphere obstacle A_h inside this region. Furthermore, the size of NR_m is determined by the velocity of each obstacle and the size of time-step τ. Unless otherwise noted, a neighboring region of $NR_m = 5r_m$ is used during the experiments. The choice of τ must be big enough to make sure that there are no collisions in each movement step. However, a big value of τ may lead to oscillations in the joint trajectory. In these experiments, the time window is $\tau = 2$, which resulted in good solutions.

Then, the desired position and orientation of the end-effector are used to compute positioning error and use the VO to define the constraint factor for all spheres in each manipulator. A new joint configuration is computed by iterating the Eq. 30. The loop terminates as soon as the constraints of positioning error and constraint factors

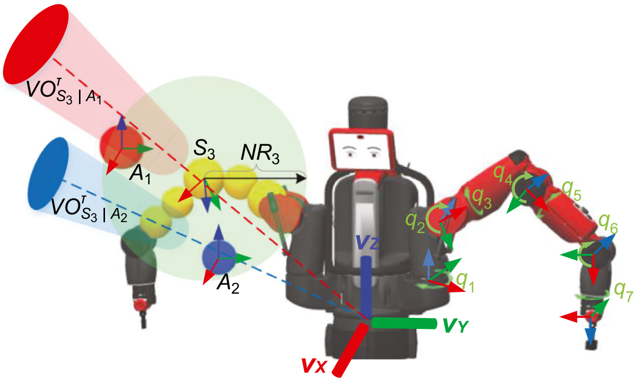


Fig. 7 The general method of combined the micromanipulator dynamic model and the VO for the right arm of Baxter robot. The 3-D environment contains two dynamic obstacles A_1 and A_2. $VO^{\tau}_{S_3|A_1}$ and $VO^{\tau}_{S_3|A_2}$ represent the velocity obstacle for sphere S_3 (orange sphere) induced by the obstacles A_1 and A_2 in the time window τ. The green sphere represents the neighbor region NR_3 of sphere S_3

Table 2 A schematic overview of our approach for finding collision-free IK solutions

Input: $\mathcal{R}$: List of redundant manipulators, $\mathcal{A}$: List of obstacles, $\varXi$:Inverse kinematics query.

```
loop
  for all $R_j \in \mathcal{R}$ and $A_h \in \mathcal{A}$
    Construct micromanipulator dynamic models
  end for
  for all $S_{j,m} \in R_j$ such that $S_{l,m} \in R_l$ and $j \neq l$ do
    Construct $VO^{\tau}_{S_{j,m}|S_{l,m}}$ and $VO^{\tau}_{S_{j,m}|A_h}$
    Compute gradient direction $\gamma^k \nabla \hat{\psi}_{j,m}(\boldsymbol{q}^k_I)$
  end for
  for all $R_j \in \mathcal{R}$ do
    Compute joint state variation $\Delta \boldsymbol{q}^k_j$
    Compute new joint configuration $\boldsymbol{q}^{k+1}_j$
    Compute positioning error $\boldsymbol{e}_j$
    Compute collision-free constraint factor $\boldsymbol{\psi}_j = (\psi_{j,1}, \dots \psi_{j,M})$
    if $\boldsymbol{q}^{k+1}_j$ fulfils stopping criterion in Algorithm 2 then
      Apply joint configuration $\boldsymbol{q}^{k+1}_j$ to $R_j$ as a collision-free IK
      solution
    end if
  end for
end loop
```

are satisfied. The general CFIK algorithm for multiple redundant manipulators is shown in Table 2.

Our approach is implemented and evaluated in three challenging scenarios of experimented redundant manipulators to demonstrate the avoidance behavior. In our experiments, a path planner of the end-effector is needed, which is out of the scope of our current formulation. The end-effector paths using this method are given by user. The joint configuration $\boldsymbol{q}^0$ and end-effector starts from a known position. The desired Cartesian error is 10^{-6} m in any of the dimensions, the step size $\gamma^k = 0.5$ and the timeout $t_{max} = 5$ ms.

A. Dynamics Obstacle (See Figs. 8, 9 and 10) One manipulator (7-DOF) avoid collisions with a human model. The human model is modeled as a series of spheres. Only that spheres are considered with respect to collision avoidance. The DH parameters of those spheres are listed in Appendix B. At this level, the human model takes the role of a dynamic obstacle, moving toward and against the Baxter robot with a planned walking path, which can be expressed by the planar position and orientation $\boldsymbol{r}(x_H, y_H, \theta_H)$. In addition, the end-effector of right arm must maintain the task constraint in the Cartesian space.

B. Dual-Arm Robot Coordination (See Figs. 11 and 12) Two robot arms (14-DOF) working independently in a shared environment and do not collide with each other. The left arm of Baxter robot is modeled as a series of spheres, which are introduced in the workspace as obstacles that the right arm has to avoid. At the beginning of the movement, two arms are in the untucked position. The DH parameters of sphere obstacles and left arm model are listed in Appendix C. The joint motion command for the left arm was generated based on traditional closed-loop IK algorithm, and the joint motion command for the right arm was generated based on our CFIK algorithm. The end-effector of both arms must maintain the task constraint in the Cartesian space. As soon as the right arm moves three boxes to the new positions on the table, the left arm can complete its motion.

C. Three Arms Coordination (See Figs. 13 and 14) Three arms (21-DOF) working independently in a shared environment and do not collide with each other. Those three arms are in the untucked position at the beginning of movement. The Kuka LBR iiwa articulated robot arm has been considered as obstacles that the Baxter's two arms have to avoid. The DH parameters of those spheres and the Kuka arm model are listed in Appendix D. The joint motion command for the Kuka arm was generated based on traditional closed-loop IK algorithm, and the joint motion command for the Baxter's two arms was generated based on our CFIK algorithm. In addition, the end-effector of three arms must maintain the task constraint in the Cartesian space.

These results show that the proposed algorithm is feasible for the redundant manipulators to perform tasks with

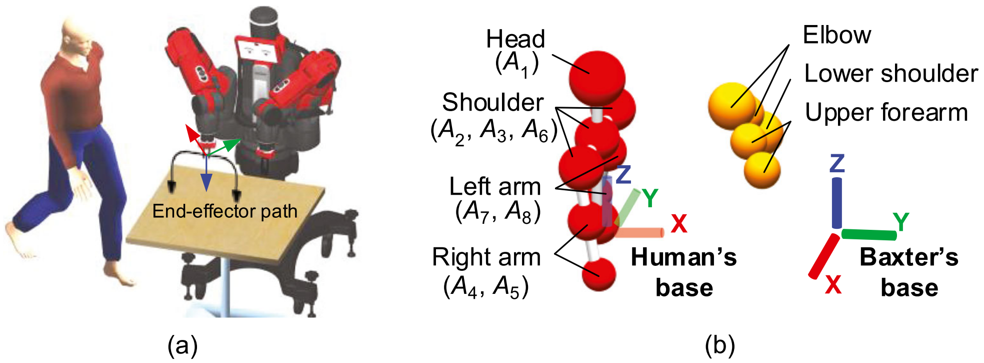


Fig. 8 **a** View of the working environment with a Baxter robot and a human model. The end-effector path is also shown. **b** The micromanipulator dynamic model for the right arm of Baxter robot (orange spheres) and the human model (red spheres)

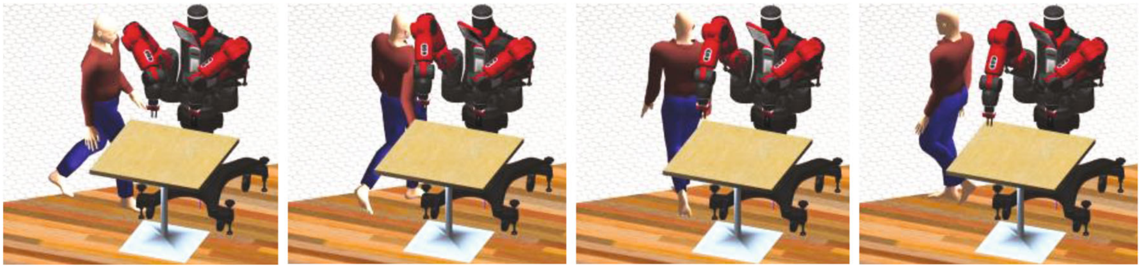


Fig. 9 Simulation of 3-D collision-free motion for picking & placing use-case by right arm. The right arm is in the untucked position before the movement. There is a human model walking in the working environment. Our method can generate collision-free IK solutions for the right arm and complete the place motion

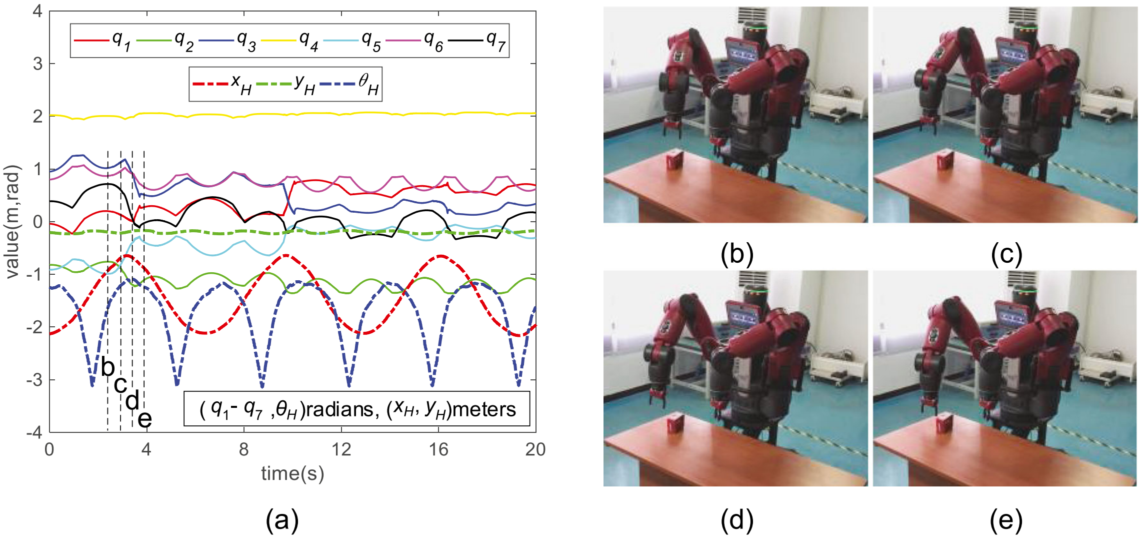
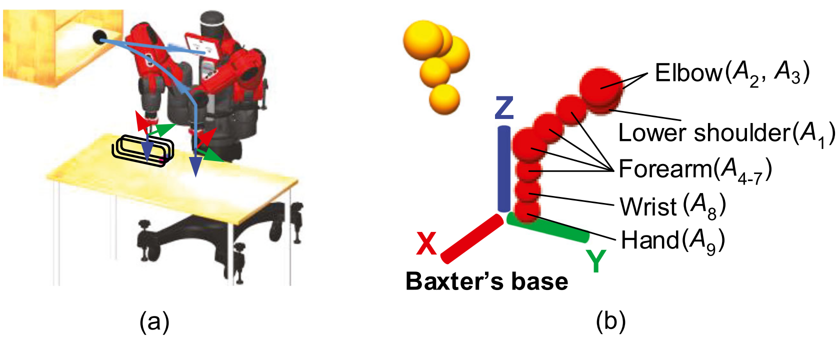


Fig. 10 Experiment of 3-D collision-free motion for picking & placing use-case. (a) Paths for right arm joints q_1-q_7 and human motion parameters x_H, y_H and θ_H. (b-e) The CFIK algorithm has been successfully performed by the real Baxter robot. The human and the right arm will meet, and joints q_1-q_3 diverge significantly to avoid human

Fig. 11 **a** View of the simulation environment with a Baxter robot. The end-effector paths of right arm (black curve) and left arm (blue curve) are also shown. **b** The micromanipulator dynamic model for the left arm (red spheres) and right arm (orange spheres)



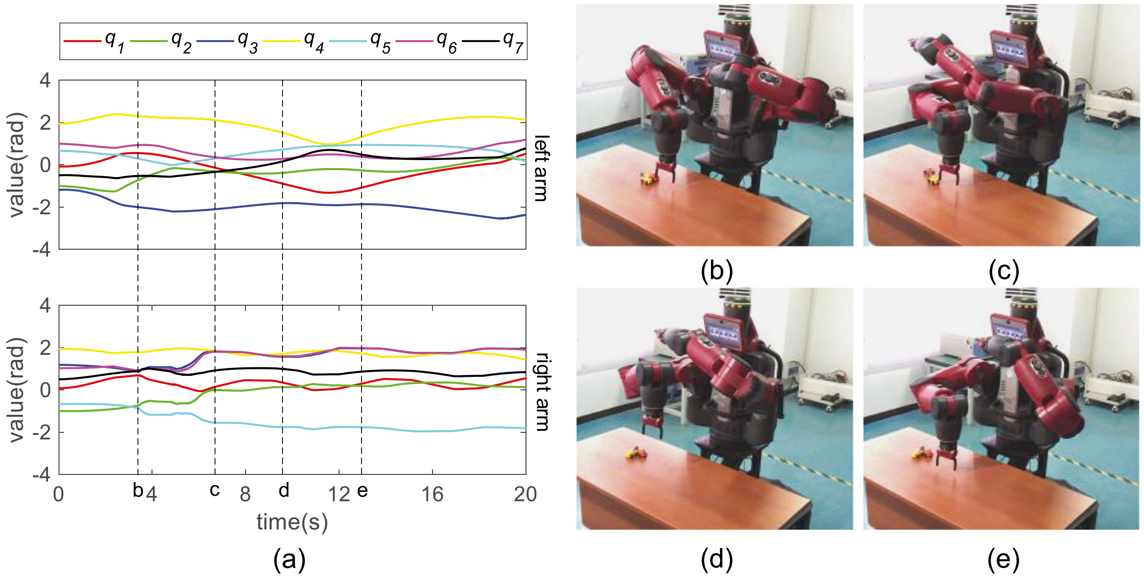
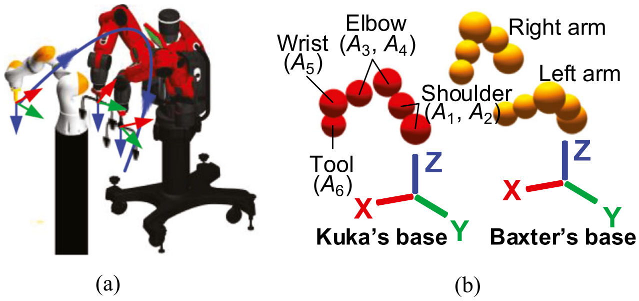


Fig. 12 Experiment of 3-D collision-free motion for moving box use-case of both arms of Baxter robot (b-e). (a) Paths for joints q_1-q_7 of the left and right arms. The arms will meet, and right arm joints q_2 and q_3 diverge significantly to avoid left arm

Fig. 13 **a** View of the simulation environment with a Baxter robot and a Kuka arm. The end-effector paths of both arms of the Baxter robot (black curves) and Kuka arm (blue curve) are also shown. **b** The micromanipulator dynamic model for the Baxter's two arms (orange spheres) and Kuka arm (red spheres)



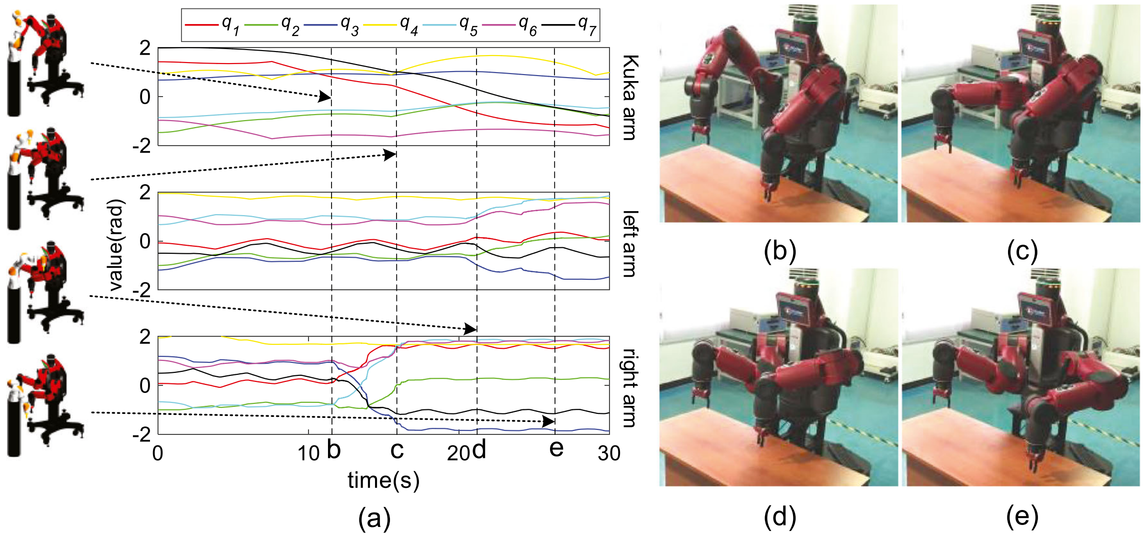


Fig. 14 Experiment of 3-D collision-free motion for picking & placing use-case of both arms of Baxter robot (**b**-**e**). **a** Paths for joints q_1-q_7 of Kuka arm, left arm and right arm. The arms will meet, and left and right arm joints q_1-q_5 diverge significantly to avoid Kuka arm

Table 3 A comparison of the computation time for the six algorithms tested across various DOF

Technique	DOF	$\boldsymbol{e}$ (Position/Rotation, m)	Environment	Avg Time(ms)
FR-RRT	$6+2$	$10^{-6}/10^{-6}$	static obstacles	≥ 2.5
TS-RRT	$6+2$	$10^{-4}/10^{-4}$	static obstacles	≥ 2.5
RGD-RRT	$6+2$	$10^{-5}/10^{-5}$	static obstacles	≥ 5
PRM	6	/	static obstacles	< 10
PSO	7	$10^{-6}/10^{-6}$	dynamic obstacles	2.01
CFIK	7	$10^{-6}/10^{-6}$	dynamic obstacles	< 1.5

collision avoidance in dynamic environments. In these simulation and physical experiments, there is no collisions occur during operations. As Fig. 8 shows, consider the right arm of Baxter robot with seven joints and the task space constraints with six DOF (i.e., $\xi^t \in \mathbb{R}^6$). This manipulator uses the joint q_1-q_5 to avoid obstacles, i.e., $i = 5$ in Eq. 29, as shown in Fig. 10a. Furthermore, all joint angles are within their specified limit constraints and there is no sudden activation of each joint, as shown by the joint space path in Figs. 12a and Fig. 14a.

6.2 Performance Analysis

As introduced in Section 4, the CFIK algorithm uses a locally optimal collision avoidance scheme, which computes the velocity vector without any collisions for each sphere independently. If this method produces a valid velocity vector for each sphere, then this proves by the VO that it is possible to work in a 3-D workspace among other manipulators and/or moving obstacles. If this method fails, we assume it is impossible to operate in dynamic environments and reject this IK solution.

The CFIK algorithm is implemented in C++ based on an ROS Kinetic Kame. The code runs on a single notebook CPU i7-7700HQ at 2.81 GHz. The computational time is a critical variable when multiple redundant manipulators work cooperatively in a shared workspace with dynamic obstacles. To compute collision-free IK solutions, some algorithms, such as RRT [42] and PRM [43], operate by the randomized motion planning algorithms. Likewise, an early version of this work using PSO and RVO is presented in [44]. However, in the case of random selection, these algorithms make the computational cost grow. Our CFIK algorithm is a computationally efficient method. The algorithm creates a gradient of the function $\psi_m(\boldsymbol{q}_I)$ as a secondary constraint, and projects it to the Newton-based convergence with the task space and joint limit constraints. This method is found to be faster than related randomized motion planning approaches; achieving computational performance similar to closed-loop IK algorithms with no avoid collisions. To further demonstrate the efficiency of our algorithm, Table 3 summarizes the average computation time of the six algorithms for standard manipulation chains. Note that our approach can generate collision-free IK solutions in dynamic environments and can handle redundant manipulators.

Furthermore, to avoid collisions, the relationship between the VO and the joint space motion is identified instead of defining the distance functions for obstacles such as [45] and [46]. This has been done for the redundant manipulators with different dynamic environments. To our knowledge, there is no state-of-the-art IK algorithm that is capable of the joint space motion planning of multiple redundant manipulators using the velocity of each sphere in a shared workspace, which is a huge advantage.

7 Conclusion and Future Work

In this paper, a new CFIK algorithm for redundant manipulators has been presented. The algorithm can be implemented in real-time for the control of multiple redundant manipulators in dynamic scenes and is based on an IK solution that consider task space, joint limit and collision-free constraints. Moreover, convergence rate and stability analysis have been discussed to demonstrate the feasibility of the proposed approach. In simulation and physical experiments, a Baxter robot (each arm has seven DOF) was used to demonstrate the efficiency of the CFIK algorithm in different dynamic workspaces, which may include the human worker or other moving robots. The results have shown that our algorithm requires less computation and is able to compute collision-free, smooth IK solutions for all manipulators in the workspace.

In future, the research would aim at evaluating the performance of our method in dynamic environment with other complex obstacles, that takes into account more than simply sphere obstacle, and apply them to the micromanipulator dynamic model formulation. In addition, our implementation considers only 7-DOF redundant manipulators, but we would like to adapt the CFIK algorithm for other kinematic configurations, in particular snake robots in [14] and hyper-redundant manipulators in [21] as they have similar kinematic parameters.

Acknowledgements This work has been supported by the National Natural Science Foundation of China [Project Number: 91848101] and the Foundation for Innovative Research Groups of the National Natural Science Foundation of China [Grant Number: 51521003].

Supplementary Information The online version contains supplementary material available at (10.1007/s10846-020-01279-w)

Compliance with Ethical Standards

Conflict of interests The authors declare that they have no conflict of interest.

Appendix A: DH Parameters of Micromanipulator Dynamic Model and Right Arm Model

i/S_m	$\alpha_{i-1}/\alpha_{m-1}$(rad)	a_{i-1}/a_{m-1}(m)	d_i/d_m(m)	θ_i/θ_m (rad)	joint limit(rad)	r_m(m)
0	0	0	0	-0.7854	/	/
1	0	0.83288	0.129626	q_1	$(-1.7016, +1.7016)$	/
2	-1.5708	0.069	0.27053	q_2	$(-2.147, +1.047)$	/
3	1.5708	0.102	0	$q_3 + 1.5708$	$(-3.0541, +3.0541)$	/
S_1	0	0	0	0	/	0.08
S_2	0	0	0.128	0	/	0.07
S_3	0	0.069	0.134	0	/	0.085
4	-1.5708	0	0	$q_4 - 1.5708$	$(-0.05, +2.618)$	/
5	1.5708	0.10359	0	$q_5 + 1.5708$	$(-3.059, +3.059)$	/
S_4	0	0	0	0	/	0.065
S_5	0	0	0.16641	0	/	0.065
6	-1.5708	0.01	0.10359	$q_6 - 1.5708$	$(-1.5707, +2.094)$	/
7	1.5708	0	0.115975	$q_7 + 1.5708$	$(-3.059, +3.059)$	/

Appendix B: DH Parameters of the Human Model

A_h	α_{h-1}(rad)	a_{h-1}(m)	d_h(m)	θ_h (rad)	r_h(m)
A_1	0	0	0.7	0	0.1
A_2	0	0	-0.25	0	0.08
$A_3(A_6)$	$+(-)1.5708$	0	0.2	$-(+)1.5708$	0.08
$A_4(A_7)$	0	0	0.2	0	0.08
$A_5(A_8)$	0	0	0.15	0	0.07

Appendix C: DH Parameters of Sphere Obstacles and Left Arm Model

i/A_h	$\alpha_{i-1}/\alpha_{h-1}$(rad)	a_{i-1}/a_{h-1}(m)	d_i/d_h(m)	θ_i/θ_h (rad)	joint limit(rad)	r_h(m)
0	0	0	0	0.7854	/	/
1	0	0.83288	0.129626	q_1	$(-1.7016, +1.7016)$	/
2	-1.5708	0.069	0.27053	q_2	$(-2.147, +1.047)$	/
3	1.5708	0.102	0	$q_3 + 1.5708$	$(-3.0541, +3.0541)$	/
A_1	0	0	0	0	/	0.08
A_2	0	0	0.128	0	/	0.07
A_3	0	0.069	0.134	0	/	0.085
4	-1.5708	0	0	$q_4 - 1.5708$	$(-0.05, +2.618)$	/
5	1.5708	0.10359	0	$q_5 + 1.5708$	$(-3.059, +3.059)$	/
A_4	0	0	0	0	/	0.065
A_5	0	0	0.16641	0	/	0.065
A_6	0	0.01	0.10359	0	/	0.075
6	-1.5708	0	0	$q_6 - 1.5708$	$(-1.5707, +2.094)$	/
7	1.5708	0	0.115975	$q_7 + 1.5708$	$(-3.059, +3.059)$	/
A_7	0	0	0.015	0	/	0.055
A_8	0	0	0.124	0	/	0.055
A_9	0	0	0.115	0	/	0.055

Appendix D: DH Parameters of Sphere Obstacles and Kuka Arm Model

i/A_h	$\alpha_{i-1}/\alpha_{h-1}$(rad)	a_{i-1}/a_{h-1}(m)	d_i/d_h(m)	θ_i/θ_h (rad)	joint limit(rad)	r_h(m)
1	1.5708	0	0.36	q_1	$(-2.967, +2.967)$	/
2	-1.5708	0	0	q_2	$(-2.094, +2.094)$	/
A_1	0	0	0	0	/	0.08
A_2	0	0	0.21	0	/	0.07
3	-1.5708	0	0.21	q_3	$(-2.967, +2.967)$	/
4	1.5708	0	0	q_4	$(-2.094, +2.094)$	/
A_3	0	0	0	0	/	0.075
A_4	0	0	0	0.21	/	0.07
5	1.5708	0	0.19	q_5	$(-2.967, +2.967)$	/
6	-1.5708	0	0	q_6	$(-2.094, +2.094)$	/
A_5	0	0	0	0	/	0.075
7	0	0	0.126	q_7	$(-2.967, +2.967)$	/
A_6	0	0	0	0.02	/	0.065

References

1. Ajoudani, A., Zanchettin, A.M., Ivaldi, S., Albu-Schäffer, A., Kosuge, K., Khatib, O.: Progress and prospects of the human–robot collaboration. Auton. Robot. **42**(5), 957–975 (2018)
2. Dogar, M., Spielberg, A., Baker, S., Rus, D.: Multi-robot grasp planning for sequential assembly operations. Auton. Robot. **43**(3), 649–664 (2019)
3. Oriolo, G., Mongillo, C.: Motion planning for mobile manipulators along given end-effector paths. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, pp. 2154–2160. IEEE (2005)
4. Yao, Z., Gupta, K.: Path planning with general end-effector constraints: Using task space to guide configuration space search. In: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1875–1880. IEEE (2005)
5. Stilman, M., Schamburek, J.U., Kuffner, J., Asfour, T.: Manipulation planning among movable obstacles. In: Proceedings 2007 IEEE international conference on robotics and automation, pp. 3327–3332. IEEE (2007)
6. Chan, T.F., Dubey, R.V.: A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators. IEEE Trans. Robot. Autom. **11**(2), 286–292 (1995)
7. Assal, S.F., Watanabe, K., Izumi, K.: Neural network-based kinematic inversion of industrial redundant robots using cooperative fuzzy hint for the joint limits avoidance. IEEE/ASME Trans. Mech. **11**(5), 593–603 (2006)
8. Orin, D.E., Schrader, W.W.: Efficient computation of the jacobian for robot manipulators. Int. J. Robot. Res. **3**(4), 66–75 (1984)
9. Ben-Israel, A., Greville, T.N.: Generalized inverses: theory and applications, vol. 15. Springer Science & Business Media, New York (2003)
10. Colomé, A., Torras, C.: Closed-loop inverse kinematics for redundant robots: Comparative assessment and two enhancements. IEEE/ASME Trans. Mech. **20**(2), 944–955 (2014)
11. Beeson, P., Ames, B.: Trac-Ik: an open-source library for improved solving of generic inverse kinematics. In: 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids), pp. 928–935. IEEE (2015)
12. Starke, S., Hendrich, N., Krupke, D., Zhang, J.: Evolutionary multi-objective inverse kinematics on highly articulated and humanoid robots. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 6959–6966. IEEE (2017)
13. Sreenivasan, S., Goel, P., Ghosal, A.: A real-time algorithm for simulation of flexible objects and hyper-redundant manipulators. Mech. Mach. Theory **45**(3), 454–466 (2010)
14. Gong, C., Travers, M.J., Astley, H.C., Li, L., Mendelson, J.R., Goldman, D.I., Choset, H.: Kinematic gait synthesis for snake robots. Int. J. Robot. Res. **35**(1-3), 100–113 (2016)
15. Camarillo, D.B., Milne, C.F., Carlson, C.R., Zinn, M.R., Salisbury, J.K.: Mechanics modeling of tendon-driven continuum manipulators. IEEE Trans. Robot. **24**(6), 1262–1273 (2008)
16. Bayram, A., Özgören, M.K.: The position control of a spatial binary hyper redundant manipulator through its inverse kinematics. Proc. Inst. Mech. Eng. C **227**(2), 359–372 (2013)
17. Ivanescu, M., Popescu, N., Popescu, D.: The shape control of tentacle arms. Robotica **33**(3), 684–703 (2015)
18. Nakagaki, K., Follmer, S., Ishii, H.: Lineform: Actuated curve interfaces for display, interaction, and constraint. In: Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology, pp. 333–339. ACM (2015)
19. Chirikjian, G.S., Burdick, J.W.: A modal approach to hyper-redundant manipulator kinematics. IEEE Trans. Robot. Autom. **10**(3), 343–354 (1994)
20. Shvalb, N., Moshe, B.B., Medina, O.: A real-time motion planning algorithm for a hyper-redundant set of mechanisms. Robotica **31**(8), 1327–1335 (2013)
21. Martín, A., Barrientos, A., del Cerro, J.: The natural-ccd algorithm, a novel method to solve the inverse kinematics of hyper-redundant and soft robots. Soft robotics **5**(3), 242–257 (2018)
22. Stilman, M.: Global manipulation planning in robot joint space with task constraints. IEEE Trans. Robot. **26**(3), 576–584 (2010)
23. da Graċa Marcos, M., Machado, J.T., Azevedo-Perdicoúlis, T.P.: A multi-objective approach for the motion planning of redundant manipulators. Appl. Soft Comput. **12**(2), 589–599 (2012)
24. Manocha, D., Canny, J.F.: Real time inverse kinematics for general 6r manipulators. In: Proceedings 1992 IEEE International Conference on Robotics and Automation, pp. 383–389. IEEE (1992)
25. Shiller, Z., Large, F., Sekhavat, S.: Motion planning in dynamic environments: Obstacles moving along arbitrary trajectories. In: Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164), vol. 4, pp. 3716–3721. IEEE (2001)
26. Kluge, B., Prassler, E.: Reflective navigation: individual behaviors and group behaviors. In: IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004, vol. 4, pp. 4172–4177. IEEE (2004)
27. Abe, Y., Yoshiki, M.: Collision avoidance method for multiple autonomous mobile agents by implicit cooperation. In: Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180), vol. 3, pp. 1207–1212. IEEE (2001)
28. Van den Berg, J., Lin, M., Manocha, D.: Reciprocal velocity obstacles for real-time multi-agent navigation. In: 2008 IEEE International Conference on Robotics and Automation, pp. 1928–1935. IEEE (2008)
29. Van Den Berg, J., Snape, J., Guy, S.J., Manocha, D.: Reciprocal collision avoidance with acceleration-velocity obstacles. In: 2011 IEEE International Conference on Robotics and Automation, pp. 3475–3482. IEEE (2011)
30. Snape, J., Manocha, D.: Navigating multiple simple-airplanes in 3D workspace. In: 2010 IEEE International Conference on Robotics and Automation, pp. 3974–3980. IEEE (2010)
31. Alonso-Mora, J., Breitenmoser, A., Rufli, M., Beardsley, P., Siegwart, R.: Optimal reciprocal collision avoidance for multiple non-holonomic robots. In: Distributed Autonomous Robotic Systems, pp. 203–216. Springer (2013)
32. Lalish, E., Morgansen, K.A.: Distributed reactive collision avoidance. Auton. Robot. **32**(3), 207–226 (2012)
33. Rufli, M., Alonso-Mora, J., Siegwart, R.: Reciprocal collision avoidance with motion continuity constraints. IEEE Trans. Robot. **29**(4), 899–912 (2013)
34. Alonso-Mora, J., Breitenmoser, A., Beardsley, P., Siegwart, R.: Reciprocal collision avoidance for multiple car-like robots. In: 2012 IEEE International Conference on Robotics and Automation, pp. 360–366. IEEE (2012)
35. Bareiss, D., Van den Berg, J.: Reciprocal collision avoidance for robots with linear dynamics using lqr-obstacles. In: 2013 IEEE International Conference on Robotics and Automation, pp. 3847–3853. IEEE (2013)
36. Bareiss, D., van den Berg, J.: Generalized reciprocal collision avoidance. Int. J. Robot. Res. **34**(12), 1501–1514 (2015)
37. Khatib, O.: A unified approach for motion and force control of robot manipulators: the operational space formulation. IEEE J. Robot. Autom. **3**(1), 43–53 (1987)
38. Aristidou, A., Lasenby, J., Chrysanthou, Y., Shamir, A.: Inverse kinematics techniques in computer graphics: a survey. In:

Computer Graphics Forum, vol. 37, pp. 35–58. Wiley Online Library (2018)
39. Shapiro, A., Wardi, Y.: Convergence analysis of gradient descent stochastic algorithms. J. Optim. Theory Appl. **91**(2), 439–454 (1996)
40. Gass, S.I., Fu, M.C. (eds.): Lipschitz Continuous. Springer, Boston (2013)
41. Antonelli, G.: Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic systems. IEEE Trans. Robot. **25**(5), 985–994 (2009)
42. LaValle, S.M.: Rapidly-Exploring Random Trees: a New Tool for Path Planning. Technical report, Iowa State University (1998)
43. Kavraki, L.E., Svestka, P., Latombe, J.C., Overmars, M.H.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Robot. Autom. **12**(4), 566–580 (1996)
44. Zhao, L., Zhao, J., Liu, H., Manocha, D.: Collision-free kinematics for redundant manipulators in dynamic scenes using optimal reciprocal velocity obstacles. arXiv:1811.00600 (2018)
45. Khatib, O., Sentis, L., Park, J., Warren, J.: Whole-body dynamic behavior and control of human-like robots. Int. J. Human. Robot. **1**(01), 29–43 (2004)
46. Mu, Z., Xu, W., Liang, B.: Avoidance of multiple moving obstacles during active debris removal using a redundant space manipulator. Int. J. Control Autom. Syst. **15**(2), 815–826 (2017)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Liangliang Zhao was born in 1987. He is currently a Ph.D. candidate in State Key Laboratory of Robotics and System at Harbin Institute of Technology in China. He received his B.S. (2010) and M.S. (2015) degrees in Mechanical Engineering and Automation at Northeastern University in China. Previously (2018–2019), he was a Visiting Student at the University of Maryland at College Park, USA. His research interests include robotics, motion planning of redundant/hyper-redundant manipulator in dynamic workspace. E-mail: zhaoliangliang0619@126.com.

Jingdong Zhao received his B.S., M.S. and Ph.D. degrees in Mechanical Engineering, in 2000, 2002 and 2006, respectively, all from Harbin Institute of Technology (HIT) in China. Now he is a professor at the school of Mechanic Engineering in HIT. His research projects include dexterous robot hands, space robotics and control of redundant/hyper-redundant manipulator, etc. E-mail: zhaojingdong@hit.edu.cn.

Hong Liu received his B.S. (1986) and Ph.D. degrees (1993) from the Harbin Institute of Technology (HIT) in China. During 1991–1993 he worked as a joint-Ph.D. candidate in the German Aerospace Center (DLR) in Germany. Since 1993 he has been a Research Fellow in this institute. He is currently a professor in HIT. His research projects include the development of a dexterous robot hand, space manipulator, etc. E-mail: hong.liu@hit.edu.cn.