Advanced Robotics

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/tadr20

Fast and robust numerical method for inverse kinematics with prioritized multiple targets for redundant robots

Masanori Sekiguchi & Naoyuki Takesue

To cite this article: Masanori Sekiguchi & Naoyuki Takesue (2020) Fast and robust numerical method for inverse kinematics with prioritized multiple targets for redundant robots, Advanced Robotics, 34:16, 1068-1078, DOI: 10.1080/01691864.2020.1780151

To link to this article: https://doi.org/10.1080/01691864.2020.1780151

ADVANCED ROBOTICS
2020, VOL. 34, NO. 16, 1068–1078
https://doi.org/10.1080/01691864.2020.1780151

FULL PAPER

OPEN ACCESS Check for updates

Fast and robust numerical method for inverse kinematics with prioritized multiple targets for redundant robots

Masanori Sekiguchi and Naoyuki Takesue

Graduate School of Systems Design, Tokyo Metropolitan University, Tokyo, Japan

ABSTRACT

In this paper, a new numerical method for inverse kinematics with prioritized multiple targets is proposed. The proposed method is constructed based on the virtual spring model and joint-based damping control. The targets are prioritized by adjusting the effect of the virtual springs. The proposed method has the following three features. First, it does not require complex calculations such as a Jacobian matrix projection into the null space. Second, it can solve prioritized inverse kinematics problems in the position level without integrating the joint velocity. Third, it is robust to parameter variations and singular configurations. The second feature is motivated by the background that most industrial robots in factories are used as position-controlled robots. Simulation experiments using a 9-DOF redundant robot show that the proposed method is faster and more robust than the conventional method. The proposed method is expected to be useful for helping to avoid collisions between links and obstacles using the redundancy.

ARTICLE HISTORY
Received 7 October 2019
Revised 3 February 2020
Accepted 17 March 2020

KEYWORDS
Inverse kinematics; multiple targets; prioritization; redundant robot

1. Introduction

In redundant robots, the variety of motions that robots can perform is expanded. For example, the range of the end effector can be increased, and obstacle avoidance problem can be solved by utilizing the redundancy of joints [1,2].

Inverse kinematics methods with prioritized multiple targets can be implemented to utilize redundancy for obstacle avoidance. Such inverse kinematics methods are called prioritized inverse kinematics [3–5]. In prioritized inverse kinematics, prioritized multiple targets (position and orientation in the task space) are assigned to each part of a robot. For example, if two targets are given to the end effector and a certain link, the target given to the end effector can be set to high priority, and the target given to the link can be set to low priority. It is also possible to set each of the targets to the opposite priority. If the errors between the targets and the corresponding parts of the robot cannot be simultaneously zero, priority is given to minimizing the error corresponding to the high-priority target, and the error corresponding to the low-priority target is minimized as much as possible. Prioritized inverse kinematics is used not only for serial link robots but also for robots with multiple end effectors such as humanoid robots [6–8].

Inverse kinematics methods can be divided into two categories: either analytical methods for obtaining solutions by geometrical consideration or algebraic transformation, or numerical methods that perform iterative calculations [9]. Several analytical methods have been proposed for robots with few redundant degrees of freedom, such as a 7-DOF robot arm [10,11]. However, analytical solutions can be obtained only when the robot structure satisfies certain conditions. In other words, the analytical methods lack versatility, and the robot mechanisms that we can develop are limited.

A number of numerical methods have been proposed to solve prioritized inverse kinematics problems for the velocity as an alternative when an analytical solution cannot be obtained. Focusing on the instantaneous velocity of the end effector and each joint makes it easier to handle kinematics mathematically. The original method for prioritized inverse kinematics in the velocity level used a pseudo-inverse matrix and Jacobian matrix projection onto the null space. This method was proposed by Nakamura et al. [12] and Maciejewski *et al.* [13] and extended later for an arbitrary number of targets by Siciliano *et al.* [14]. Since this method numerically unstable in the vicinity of a singular configuration, Chiaverini et al. [15] proposed a new method to reduce this

CONTACT Masanori Sekiguchi sekiguchi-masanori@ed.tmu.ac.jp; mechatronics0816@outlook.com

instability. Furthermore, Park et al. [16] and Choi et al. [17] improved the accuracy of the numerical solutions by replacing the pseudo-inverse matrix used in Chiaverini's method with a weighted pseudo-inverse matrix. As the weighted pseudo-inverse cannot be calculated in a singular configuration, a regularization step, such as introducing a damping factor, is required [18–20]. A common feature of these methods is that they are calculating the joint velocity. However, most industrial robots in factories are used as joint position-controlled robots. While position control has the disadvantage of being potentially dangerous when unintended contact occurs, it is practical and widely used in industrial robots and elsewhere. Therefore, in terms of utility, it is more important to obtain the desired joint position than to calculate the instantaneous joint velocity.

Sugihara proposed a numerical method for prioritized inverse kinematics in the position level [21], and its application to humanoid robots has also been demonstrated in the literature [22]. Sugihara's method has the advantages that the desired joint position can be obtained without integrating the joint velocity, and the algorithm is numerically stable even in singular configurations. However, Sugihara's method has the following two problems. The first one is the speed of convergence depends on the solvability and parameter settings. The second is the addition of the angle-axis vector (orientation error) included in Sugihara's method complicates the calculation process.

In this paper, a new numerical method for prioritized inverse kinematics is proposed, which is easy to implement and converges faster and more robustly than the conventional method. By using the proposed numerical method, the desired joint position (the solution of the prioritized inverse kinematics problem in the position level) can be calculated without integrating the joint velocity. The technique proposed in this paper is expected to be applicable to motion planning algorithms for redundant robots considering collisions between links and obstacles.

In Section 2, differential kinematics and calculation methods for the error vector are described. These methods form the basis of the numerical method described later. In Section 3, a numerical method for inverse kinematics problems with non-prioritized multiple targets is constructed based on the virtual spring model and joint damping control, and a framework for the proposed method is created. In Section 4, the sufficient conditions for reducing the elastic energy of the virtual spring are considered. Moreover, based on the sufficient conditions, a numerical method for inverse kinematics is proposed. In Section 5, two methods for assigning priorities to each target are described. The first is the conventional method, and the second is the proposed method. In Section 6, the calculation time and robustness of the proposed method are compared with the conventional method based on the results of simulation experiments. In addition, it is shown that the proposed method is algorithmically stable even for singular configurations. Finally, conclusion is presented in Section 7.

2. Definition of error vector and differential kinematics

In this section, the definitions of the error vector and differential kinematics are given. This section provides the foundations for the inverse kinematics method described later. The vector of joint displacements for an n-DOF redundant robot is defined as

$$\boldsymbol{q} = [\, q_1 \; q_2 \; \cdots \; q_n \,]^{\mathrm{T}} \in \mathbb{R}^n. \tag{1}$$

As shown in Figure 1, $\boldsymbol{p}_1(\boldsymbol{q}) \in \mathbb{R}^3$ and $\boldsymbol{R}_1(\boldsymbol{q}) \in SO(3)$ are the position and orientation of the end effector, respectively. Let ${}^{\mathrm{d}}\boldsymbol{p}_1 \in \mathbb{R}^3$ and ${}^{\mathrm{d}}\boldsymbol{R}_1 \in SO(3)$ be the desired position and orientation (hereafter Target 1), respectively. Then the error vector $\boldsymbol{e}_1(\boldsymbol{q})$ is given by

$$\boldsymbol{e}_1(\boldsymbol{q}) = \begin{bmatrix} {}^{\mathrm{d}}\boldsymbol{p}_1 - \boldsymbol{p}_1(\boldsymbol{q}) \\ \boldsymbol{\phi}\left({}^{\mathrm{d}}\boldsymbol{R}_1 \boldsymbol{R}_1^{\mathrm{T}}(\boldsymbol{q})\right) \end{bmatrix} \in \mathbb{R}^6 \tag{2}$$

where $\boldsymbol{\phi}(*)$ is a function that transforms the rotation matrix $*$ into an angle-axis vector (see Appendix 1). The angle-axis vector represents the orientation error. The orientation error vector proposed by Luh et al. [23] has two problems. One is that the error vector is not computed correctly when the rotation matrix is symmetric. Second, the norm of the error vector is not proportional to the actual orientation error. However, these two problems are solved by using the angle-axis vector. The angle-axis vector is equivalent to the product of the rotation angle and the unit axis vector. The absolute value of the rotation angle is π or less. That is, the norm of the angle-axis vector is also π or less. From the viewpoint of representing the orientation error, the angle-axis vector is more effective than Euler angles. In addition, the velocity of the end effector is expressed as

$$\begin{bmatrix} \dot{\boldsymbol{p}}_1(\boldsymbol{q}) \\ \boldsymbol{\omega}_1(\boldsymbol{q}) \end{bmatrix} = \boldsymbol{J}_1(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{3}$$

where $\boldsymbol{\omega}_1(\boldsymbol{q})$ is the angular velocity vector of the end effector and $\boldsymbol{J}_1(\boldsymbol{q}) \in \mathbb{R}^{6\times n}$ is the Jacobian matrix.

Let the second desired position and orientation (hereafter Target 2) be denoted by ${}^{\mathrm{d}}\boldsymbol{p}_2 \in \mathbb{R}^3$ and ${}^{\mathrm{d}}\boldsymbol{R}_2 \in SO(3)$, respectively. The position and orientation of the link corresponding to Target 2 are defined as $\boldsymbol{p}_2(\boldsymbol{q}) \in \mathbb{R}^3$ and $\boldsymbol{R}_2(\boldsymbol{q}) \in SO(3)$, respectively. Then the error vector $\boldsymbol{e}_2(\boldsymbol{q})$

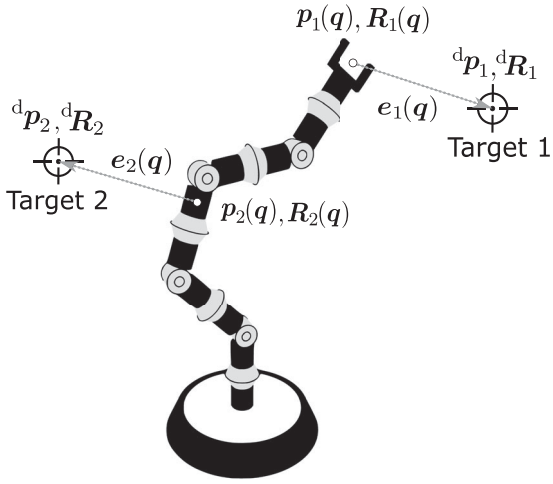


Figure 1. Error vectors when $n = 9$ and the number of targets is two. Target 1 corresponds to the end effector and Target 2 corresponds to the center of the sixth link.

is defined as

$$\boldsymbol{e}_2(\boldsymbol{q}) = \begin{bmatrix} {}^{\mathrm{d}}\boldsymbol{p}_2 - \boldsymbol{p}_2(\boldsymbol{q}) \\ \boldsymbol{\phi}\left({}^{\mathrm{d}}\boldsymbol{R}_2\boldsymbol{R}_2^{\mathrm{T}}(\boldsymbol{q})\right) \end{bmatrix} \in \mathbb{R}^6. \tag{4}$$

The velocity at position $\boldsymbol{p}_2(\boldsymbol{q})$ is expressed as

$$\begin{bmatrix} \dot{\boldsymbol{p}}_2(\boldsymbol{q}) \\ \boldsymbol{\omega}_2(\boldsymbol{q}) \end{bmatrix} = \boldsymbol{J}_2(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{5}$$

where $\boldsymbol{\omega}_2(\boldsymbol{q})$ is the angular velocity vector at position $\boldsymbol{p}_2(\boldsymbol{q})$ and $\boldsymbol{J}_2(\boldsymbol{q}) \in \mathbb{R}^{6\times n}$ is the Jacobian matrix. The Jacobian matrix is a sparse matrix and has a block structure. For example, in Figure 1, the sixth to ninth joints do not affect $\dot{\boldsymbol{p}}_2(\boldsymbol{q})$ and $\boldsymbol{\omega}_2(\boldsymbol{q})$; thus the elements from the sixth to ninth columns of $\boldsymbol{J}_2(\boldsymbol{q})$ are zero.

3. Inverse kinematics with non-prioritized multiple targets based on the virtual spring model and joint-based damping control

In this section, a numerical inverse kinematics method with non-prioritized multiple targets is constructed based on the virtual spring model and the joint-based damping control. The virtual spring hypothesis [24,25] and virtual spring-damper hypothesis [26,27] proposed by Arimoto et al. are based on robot dynamics and the desired joint torques to be calculated. Arimoto's methods are dynamically stable and have also been verified in experiments using industrial redundant robots [28]. However, these methods are difficult to apply to position-controlled robots. A method for calculating prioritized inverse kinematics based on dynamics has also been proposed [29,30], but is also difficult to apply with position-controlled robots. Therefore, in this paper, we derive a numerical method for inverse kinematics in the position level.

3.1. Virtual spring model and joint damping control

A virtual tension spring and a virtual torsion spring are installed with respect to the error vector $\boldsymbol{e}_1(\boldsymbol{q})$. The spring constants of the virtual tension spring and torsion spring are $K_{1f} \in \mathbb{R}$ and $K_{1m} \in \mathbb{R}$, respectively. The spring constant matrix is defined as

$$\boldsymbol{K}_1 = \begin{bmatrix} K_{1f}\boldsymbol{I}_3 & \boldsymbol{O} \\ \boldsymbol{O} & K_{1m}\boldsymbol{I}_3 \end{bmatrix} \in \mathbb{R}^{6\times 6} \tag{6}$$

where $\boldsymbol{I}_3$ is a 3×3 identity matrix. Similarly, a virtual tension spring and a virtual torsion spring are installed with respect to the error vector $\boldsymbol{e}_2(\boldsymbol{q})$. The spring constants of the virtual tension spring and torsion spring are $K_{2f} \in \mathbb{R}$ and $K_{2m} \in \mathbb{R}$, respectively. The spring constant matrix is defined as

$$\boldsymbol{K}_2 = \begin{bmatrix} K_{2f}\boldsymbol{I}_3 & \boldsymbol{O} \\ \boldsymbol{O} & K_{2m}\boldsymbol{I}_3 \end{bmatrix} \in \mathbb{R}^{6\times 6}. \tag{7}$$

Figure 2 shows the connected springs, $\boldsymbol{K}_1$ and $\boldsymbol{K}_2$, which link the robot links and the targets. Considering only the virtual spring $\boldsymbol{K}_1$ attached to the end effector, the virtual torque $\boldsymbol{\tau}_1$ generated in each joint is expressed as

$$\boldsymbol{\tau}_1(\boldsymbol{q}) = \boldsymbol{J}_1^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{K}_1\boldsymbol{e}_1(\boldsymbol{q}) \in \mathbb{R}^n. \tag{8}$$

Similarly, considering only the virtual spring $\boldsymbol{K}_2$, the virtual torque $\boldsymbol{\tau}_2$ generated in each joint is expressed as

$$\boldsymbol{\tau}_2(\boldsymbol{q}) = \boldsymbol{J}_2^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{K}_2\boldsymbol{e}_2(\boldsymbol{q}) \in \mathbb{R}^n. \tag{9}$$

The total virtual torque can be written as

$$\boldsymbol{\tau}(\boldsymbol{q}) = \boldsymbol{\tau}_1(\boldsymbol{q}) + \boldsymbol{\tau}_2(\boldsymbol{q}) = \boldsymbol{J}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{K}\boldsymbol{e}(\boldsymbol{q}) \in \mathbb{R}^n \tag{10}$$

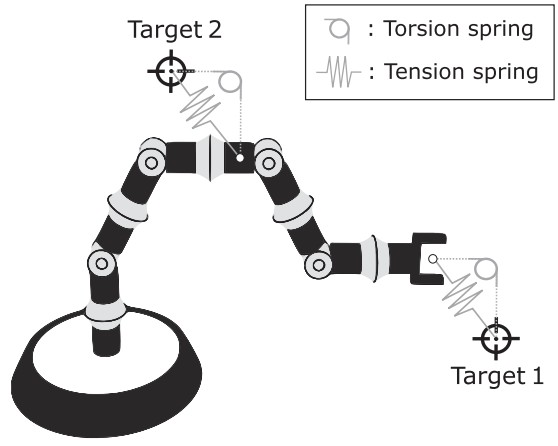


Figure 2. Virtual tension spring and virtual torsion spring.

where

$$\boldsymbol{J}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{J}_1(\boldsymbol{q}) \\ \boldsymbol{J}_2(\boldsymbol{q}) \end{bmatrix} \in \mathbb{R}^{12\times n} \tag{11}$$

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}_1 & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{K}_2 \end{bmatrix} \in \mathbb{R}^{12\times 12} \tag{12}$$

$$\boldsymbol{e}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{e}_1(\boldsymbol{q}) \\ \boldsymbol{e}_2(\boldsymbol{q}) \end{bmatrix} \in \mathbb{R}^{12} \tag{13}$$

For the case where the number of targets is 3 or more, refer to Appendix 2. Consider the case where the virtual torque $\boldsymbol{\tau}(\boldsymbol{q})$ is converted to the velocity of each joint with joint-based damping control. Using the joint damping coefficient matrix, the velocity of each joint $\dot{\boldsymbol{q}}$ can be written as follows: $\boldsymbol{D}(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$.

$$\dot{\boldsymbol{q}} = \boldsymbol{D}^{-1}(\boldsymbol{q})\boldsymbol{\tau}(\boldsymbol{q}) \tag{14}$$

The definition of $\boldsymbol{D}(\boldsymbol{q})$ is given later in this paper.

3.2. Numerical method for inverse kinematics

Consider the numerical solution for an inverse kinematics problem obtained by iterative calculations. Let $\boldsymbol{q}_k$ be the joint displacement at k iterations. In the remainder of this paper, $\boldsymbol{e}(\boldsymbol{q}_k)$, $\boldsymbol{\tau}(\boldsymbol{q}_k)$, $\boldsymbol{D}(\boldsymbol{q}_k)$, and $\boldsymbol{J}(\boldsymbol{q}_k)$ are abbreviated as $\boldsymbol{e}_k$, $\boldsymbol{\tau}_k$, $\boldsymbol{D}_k$, and $\boldsymbol{J}_k$, respectively.

The change in the joint displacement per step is $\Delta\boldsymbol{q}_k \in \mathbb{R}^n$ and is expressed as

$$\Delta\boldsymbol{q}_k = \boldsymbol{D}_k^{-1}\boldsymbol{\tau}_k = \boldsymbol{D}_k^{-1}\boldsymbol{J}_k^{\mathrm{T}}\boldsymbol{K}\boldsymbol{e}_k. \tag{15}$$

The joint displacement in the next step is expressed as

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k + \Delta\boldsymbol{q}_k. \tag{16}$$

$\boldsymbol{q}_{k+1}$ can be written as

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_0 + \sum_{i=0}^{k} \Delta\boldsymbol{q}_i. \tag{17}$$

$\boldsymbol{q}_0$ is the initial joint displacement. The above iterative calculation is summarized with the block diagram in Figure 3. In the figure, the set ${}^{\mathrm{d}}\boldsymbol{\chi}$ consisting of the targets and the set $\boldsymbol{\chi}_k$ consisting of the position and orientation of each robot part are defined as

$${}^{\mathrm{d}}\boldsymbol{\chi} = \left\{{}^{\mathrm{d}}\boldsymbol{p}_1, {}^{\mathrm{d}}\boldsymbol{R}_1, {}^{\mathrm{d}}\boldsymbol{p}_2, {}^{\mathrm{d}}\boldsymbol{R}_2\right\} \tag{18}$$

and

$$\boldsymbol{\chi}_k = \left\{\boldsymbol{p}_1(\boldsymbol{q}_k), \boldsymbol{R}_1(\boldsymbol{q}_k), \boldsymbol{p}_2(\boldsymbol{q}_k), \boldsymbol{R}_2(\boldsymbol{q}_k)\right\}, \tag{19}$$

respectively.

4. Minimizing the elastic energy of a virtual spring

In this section, sufficient conditions for $\boldsymbol{D}_k$ to minimize the elastic energy of the virtual spring are derived. In addition, a definition for $\boldsymbol{D}_k$ that satisfies the derived conditions is shown. The elastic energy V_k of the virtual spring is expressed as

$$V_k = \frac{1}{2}\boldsymbol{e}_k^{\mathrm{T}}\boldsymbol{K}\boldsymbol{e}_k \in \mathbb{R}. \tag{20}$$

The spring constant matrix $\boldsymbol{K}$ is a positive-semidefinite matrix; thus $V_k \geq 0$ holds. Here, V_k is also a candidate for quadratic Lyapunov functions in discrete-time systems.

As shown in Figure 4, the vector $\boldsymbol{J}_1(\boldsymbol{q}_k)\Delta\boldsymbol{q}_k$ is approximated by the change in the end effector per iteration; hence the following approximation holds:

$$\boldsymbol{e}_1(\boldsymbol{q}_{k+1}) \simeq \boldsymbol{e}_1(\boldsymbol{q}_k) - \boldsymbol{J}_1(\boldsymbol{q}_k)\Delta\boldsymbol{q}_k. \tag{21}$$

Similarly, the following approximation formula can be obtained regarding the error vector $\boldsymbol{e}_2(\boldsymbol{q}_k)$:

$$\boldsymbol{e}_2(\boldsymbol{q}_{k+1}) \simeq \boldsymbol{e}_2(\boldsymbol{q}_k) - \boldsymbol{J}_2(\boldsymbol{q}_k)\Delta\boldsymbol{q}_k. \tag{22}$$

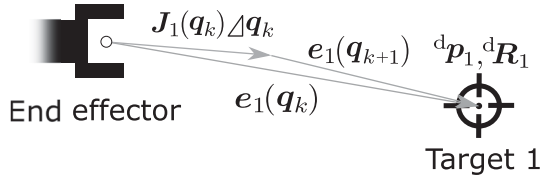


Figure 4. Change in the end effector per iteration.

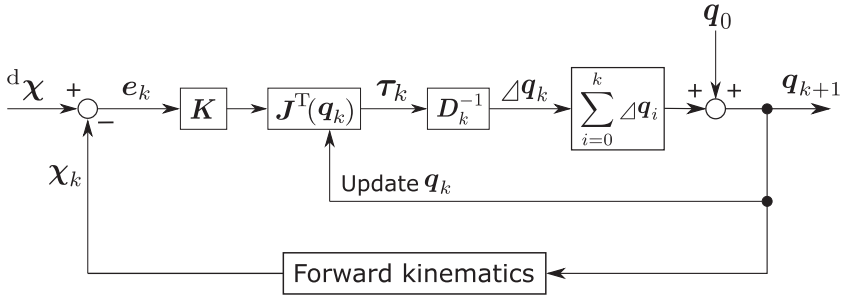


Figure 3. Numerical method for inverse kinematics based on virtual spring and joint damping control.

The following approximation formula is obtained from Equations (21) and (22):

$$\boldsymbol{e}_{k+1} \simeq \boldsymbol{e}_k - \boldsymbol{J}_k \Delta \boldsymbol{q}_k. \tag{23}$$

Hence, V_{k+1} can be transformed as follows:

$$\begin{aligned} V_{k+1} &= \frac{1}{2}\boldsymbol{e}_{k+1}^{\mathrm{T}} \boldsymbol{K}\, \boldsymbol{e}_{k+1} \\ &\simeq \frac{1}{2}\left(\boldsymbol{e}_k - \boldsymbol{J}_k \Delta \boldsymbol{q}_k\right)^{\mathrm{T}} \boldsymbol{K}\left(\boldsymbol{e}_k - \boldsymbol{J}_k \Delta \boldsymbol{q}_k\right) \\ &= \frac{1}{2}\left(\boldsymbol{e}_k^{\mathrm{T}} \boldsymbol{K} \boldsymbol{e}_k - 2\Delta \boldsymbol{q}_k^{\mathrm{T}} \boldsymbol{J}_k^{\mathrm{T}} \boldsymbol{K} \boldsymbol{e}_k + \Delta \boldsymbol{q}_k^{\mathrm{T}} \boldsymbol{J}_k^{\mathrm{T}} \boldsymbol{K} \boldsymbol{J}_k \Delta \boldsymbol{q}_k\right) \\ &= \frac{1}{2}\left(\boldsymbol{e}_k^{\mathrm{T}} \boldsymbol{K} \boldsymbol{e}_k - 2\Delta \boldsymbol{q}_k^{\mathrm{T}} \boldsymbol{D}_k \Delta \boldsymbol{q}_k + \Delta \boldsymbol{q}_k^{\mathrm{T}} \boldsymbol{J}_k^{\mathrm{T}} \boldsymbol{K} \boldsymbol{J}_k \Delta \boldsymbol{q}_k\right) \\ &= V_k - \left(\Delta \boldsymbol{q}_k^{\mathrm{T}} \boldsymbol{D}_k \Delta \boldsymbol{q}_k - \frac{1}{2}\Delta \boldsymbol{q}_k^{\mathrm{T}} \boldsymbol{J}_k^{\mathrm{T}} \boldsymbol{K} \boldsymbol{J}_k \Delta \boldsymbol{q}_k\right) \\ &= V_k - \Delta \boldsymbol{q}_k^{\mathrm{T}} \left(\boldsymbol{D}_k - \frac{1}{2}\boldsymbol{J}_k^{\mathrm{T}} \boldsymbol{K} \boldsymbol{J}_k\right) \Delta \boldsymbol{q}_k. \end{aligned} \tag{24}$$

From the above equation, the following equation is obtained:

$$V_{k+1} - V_k \simeq -\Delta \boldsymbol{q}_k^{\mathrm{T}} \left(\boldsymbol{D}_k - \frac{1}{2}\boldsymbol{J}_k^{\mathrm{T}} \boldsymbol{K} \boldsymbol{J}_k\right) \Delta \boldsymbol{q}_k. \tag{25}$$

Therefore, under conditions $\boldsymbol{J}_k^{\mathrm{T}} \boldsymbol{K} \boldsymbol{e}_k \neq \boldsymbol{0}$ and $\Delta \boldsymbol{q}_k \neq \boldsymbol{0}$, the following two sufficient conditions are derived to reduce the elastic energy of the virtual spring.

(1) Matrix $(\boldsymbol{D}_k - 1/2\boldsymbol{J}_k^{\mathrm{T}} \boldsymbol{K} \boldsymbol{J}_k)$ is a positive definite matrix, and
(2) matrix $\boldsymbol{D}_k$ is regular.

The following $\boldsymbol{D}_k$ proposed in the literature [31,32] satisfies both of the above two conditions:

$$\boldsymbol{D}_k = \boldsymbol{J}_k^{\mathrm{T}} \boldsymbol{K} \boldsymbol{J}_k + \left(\frac{1}{2} V_k + \delta\right) \boldsymbol{I}_n, \tag{26}$$

where $\boldsymbol{I}_n \in \mathbb{R}^{n \times n}$ is the identity matrix and $\delta \in \mathbb{R}$ is a small positive constant. The constant δ prevents $\boldsymbol{D}_k$ from becoming irregular when $V_k \simeq 0$ holds. The $\boldsymbol{D}_k$ shown in Equation (26) is known to be robust to singular configurations and variation of the kinematic model [31,32]. By substituting $\boldsymbol{D}_k$ into the following equation, the update equation of the joint displacement can be obtained:

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k + \boldsymbol{D}_k^{-1} \boldsymbol{J}_k^{\mathrm{T}} \boldsymbol{K} \boldsymbol{e}_k. \tag{27}$$

By using the Cholesky decomposition [33–36], the computational cost of computing the inverse matrix can be reduced, because $\boldsymbol{D}_k$ is a positive definite symmetric matrix. The procedure for computing $\boldsymbol{q}_{k+1}$ from $\boldsymbol{q}_k$ is shown in Table 1.

Table 1. Procedure for computing $\boldsymbol{q}_{k+1}$ from $\boldsymbol{q}_k$.

Step 1	Set the initial joint displacement $\boldsymbol{q}_0$ and targets ${}^{\mathrm{d}}\boldsymbol{\chi}$
Step 2	Get the position and orientation $\boldsymbol{\chi}_k$ and Jacobian matrix $\boldsymbol{J}_k$ by calculating the forward kinematics
Step 3	Calculate the error vector $\boldsymbol{e}_k$
Step 4	Calculate the virtual torque $\boldsymbol{\tau}_k = \boldsymbol{J}_k^{\mathrm{T}} \boldsymbol{K} \boldsymbol{e}_k$ and elastic energy $V_k = \frac{1}{2}\boldsymbol{e}_k^{\mathrm{T}} \boldsymbol{K} \boldsymbol{e}_k$
Step 5	Calculate the joint damping coefficient matrix $\boldsymbol{D}_k$
Step 6	Calculate the change in the joint displacement $\Delta \boldsymbol{q}_k = \boldsymbol{D}_k^{-1} \boldsymbol{\tau}_k$
Step 7	Calculate the update formula $\boldsymbol{q}_{k+1} = \boldsymbol{q}_k + \Delta \boldsymbol{q}_k$
Step 8	Add 1 to k and return to Step 2

The following minimization problem can be solved by repeating the procedure of Table 1.

$$\min_{\boldsymbol{q}} \left(V(\boldsymbol{q}) = \frac{1}{2}\boldsymbol{e}^{\mathrm{T}}(\boldsymbol{q}) \boldsymbol{K} \boldsymbol{e}(\boldsymbol{q})\right) \tag{QP1}$$

Namely, the norm of the joint velocity vector is not minimized, but the weighted sum of squared position error can be minimized. This algorithm is a local optimization procedure, because the following sufficient conditions are required for global convergence.

(1) All joints are not locked; that is, they do not exceed the joint limit during the iterative calculation.
(2) When V_k is not minimum, $\boldsymbol{J}_k^{\mathrm{T}} \boldsymbol{K} \boldsymbol{e}_k \neq \boldsymbol{0}$ holds.

However, the condition (2) is satisfied in almost all cases. Solving the minimization problem (QP1) is equivalent to solving the inverse kinematics problem with non-prioritized multiple targets.

The method derived in this section is an extended formulation of the numerical method described in [32] for application to multiple targets. However, this method cannot assign priorities to each target. In the next section, a method for assigning priorities to targets is described.

5. Multiple target prioritization

In this section, two methods for assigning priorities to multiple targets are described. One is the conventional method and the other is the proposed method.

5.1. Prioritization using the multiplier method

This section briefly outlines the numerical multiplier method for prioritized inverse kinematics. This method was proposed by Sugihara [21]. In this paper, Sugihara's method is used as a benchmarking method and compared with the proposed method described later.

If the highly prioritized constraint (Target 1) is satisfiable, the prioritized inverse kinematics problem can be regarded as the following quadratic programming problem:

$$\min_{\boldsymbol{q}} \left(V(\boldsymbol{q}) = \frac{1}{2}\boldsymbol{e}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{K}\boldsymbol{e}(\boldsymbol{q}) \right) \quad \text{subject to} \quad \boldsymbol{e}_1(\boldsymbol{q}) = \boldsymbol{0} \tag{QP2}$$

If the highly prioritized constraint is not satisfiable ($\forall \boldsymbol{q}, \boldsymbol{e}_1(\boldsymbol{q}) \neq \boldsymbol{0}$), it is necessary to convert the above problem into the following minimization problem:

$$\min_{\boldsymbol{q}} \left(V_1(\boldsymbol{q}) := \frac{1}{2}\boldsymbol{e}_1^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{K}_1\boldsymbol{e}_1(\boldsymbol{q}) \right) \tag{QP3}$$

In Sugihara's method, the problem (QP2) is solved by using multiplier method. The extended Lagrange function $L'(\boldsymbol{q}, \boldsymbol{\lambda})$ is defined as

$$L'(\boldsymbol{q}, \boldsymbol{\lambda}) = V(\boldsymbol{q}) + \boldsymbol{\lambda}^{\mathrm{T}}\boldsymbol{K}_1\boldsymbol{e}_1(\boldsymbol{q}) \in \mathbb{R} \tag{28}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^6$ is the Lagrange multiplier. In addition, the following equations are satisfied at the optimum of the problem (QP2).

$$\left(\frac{\partial L'(\boldsymbol{q}, \boldsymbol{\lambda})}{\partial \boldsymbol{q}} \right)^{\mathrm{T}} = -\boldsymbol{\tau}_2(\boldsymbol{q}) - \underbrace{\boldsymbol{J}_1^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{K}_1\Big(\boldsymbol{e}_1(\boldsymbol{q}) + \boldsymbol{\lambda}\Big)}_{\boldsymbol{\tau}_1(\boldsymbol{q})} = \boldsymbol{0} \tag{29}$$

$$\left(\frac{\partial L'(\boldsymbol{q}, \boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}} \right)^{\mathrm{T}} = \boldsymbol{e}_1(\boldsymbol{q}) = \boldsymbol{0} \tag{30}$$

Considering $\boldsymbol{J}_1^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{K}_1\boldsymbol{e}_1(\boldsymbol{q}) = \boldsymbol{\tau}_1(\boldsymbol{q})$, the torque generated by the highly prioritized target is increased by the presence of $\boldsymbol{\lambda}$. Namely, it is expected that $\boldsymbol{e}_1(\boldsymbol{q})$ can be made closer to the zero vector by increasing the $\boldsymbol{\lambda}$. Based on the above, the error vector $\boldsymbol{e}_k$ in Equation (27) is replaced with

$$\boldsymbol{e}'_k = \begin{bmatrix} \boldsymbol{e}_1(\boldsymbol{q}_k) + \boldsymbol{\lambda}_k \\ \boldsymbol{e}_2(\boldsymbol{q}_k) \end{bmatrix} \in \mathbb{R}^{12} \tag{31}$$

$$\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + \boldsymbol{e}_1(\boldsymbol{q}_k), \quad \boldsymbol{\lambda}_0 = \boldsymbol{0} \tag{32}$$

From Equation (32), the error accumulates with each step in the Lagrange multiplier. It should be noted that the fourth to sixth components of the error vector $\boldsymbol{e}_1(\boldsymbol{q}_k)$ are angle-axis vector (pseudo vector); hence the addition of the vectors is not straightforward. When calculating Equation (32), it is necessary to convert the angle-axis vector into unit quaternions or rotation matrices. Method for calculating summation of angle-axis vectors using unit quaternions is given in [21]. The difference between this prioritization method and the method shown in Table 1 is that the $\boldsymbol{e}_k$ included in Table 1 is replaced with the above $\boldsymbol{e}'_k$.

This algorithm is also a local optimization method, because the following sufficient conditions are required for global convergence.

(1) All joints are not locked.
(2) When V_k is not minimum, $\boldsymbol{J}_k^{\mathrm{T}}\boldsymbol{K}\boldsymbol{e}'_k \neq \boldsymbol{0}$ holds.

This method has two problems. One is that the convergence of the Lagrange multiplier is slow and a large calculation time is required. The second is that the calculation of $\boldsymbol{\lambda}_k$ involves the addition of angle-axis vectors, which makes the procedure more complex.

5.2. Prioritization by the proposed method

In the proposed method, the spring constant matrix $\boldsymbol{K}$ in Equation (27) is replaced with

$$\boldsymbol{K}_k = \begin{bmatrix} \boldsymbol{K}_1 & \boldsymbol{O} \\ \boldsymbol{O} & \zeta_k\boldsymbol{K}_2 \end{bmatrix} \in \mathbb{R}^{12\times 12} \tag{33}$$

$$\zeta_{k+1} = \begin{cases} \zeta_k - \dfrac{1}{4}\zeta_0 & \left(V_k \geq 0.99V_{k-1}\right) \\ 0 & \left(\zeta_k < 0\right) \\ \zeta_k & (\text{otherwise}) \end{cases} \tag{34}$$

where $\zeta_0 = 1$. If the condition $V_k \geq 0.99V_{k-1}$ holds, then the rate of decrease of V_k is 1% or less. The coefficient 0.99 was added with consideration of numerical errors. The ζ_k parameter gradually reduces the influence of $\boldsymbol{K}_2$; thus Target 1 has a higher priority than Target 2. Note, to reverse the priorities, Equation (33) can be replaced with

$$\boldsymbol{K}_k = \begin{bmatrix} \zeta_k\boldsymbol{K}_1 & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{K}_2 \end{bmatrix} \in \mathbb{R}^{12\times 12}. \tag{35}$$

The difference between this prioritization method and the method shown in Table 1 is that the $\boldsymbol{K}$ included in Table 1 is replaced with the above $\boldsymbol{K}_k$.

Considering $\boldsymbol{J}_2^{\mathrm{T}}(\boldsymbol{q})(\zeta_k\boldsymbol{K}_2)\boldsymbol{e}_2(\boldsymbol{q}) = \zeta_k\boldsymbol{\tau}_2(\boldsymbol{q})$ or $\boldsymbol{J}_1^{\mathrm{T}}(\boldsymbol{q})(\zeta_k\boldsymbol{K}_1)\boldsymbol{e}_1(\boldsymbol{q}) = \zeta_k\boldsymbol{\tau}_1(\boldsymbol{q})$, the torque generated by the lower prioritized target is reduced by the presence of ζ_k. As mentioned in [21], only changing the weight ratio of $\boldsymbol{K}$ is not enough to converge the joint displacements $\boldsymbol{q}_k$, because a small residual remains in the error vector. Therefore, this proposed prioritization method solves the problem by using the ζ_k. As the ζ_k gets smaller, the joint displacements $\boldsymbol{q}_k$ approaches the optimum value, and the error corresponding to the highly prioritized target goes to zero.

One of the advantages of this prioritization method is that it does not require the addition of angle-axis vectors (pseudo-vectors) during the computation process.

This algorithm is also a local optimization method, because the following sufficient conditions are required for global convergence.

(1) All joints are not locked.
(2) When V_k is not minimum, $\boldsymbol{J}_k^{\mathrm{T}}\boldsymbol{K}_k\boldsymbol{e}_k \neq \boldsymbol{0}$ holds.

Additionally, in the proposed method, $\zeta_k = 0$ and $V_k \geq 0.99\,\mathrm{V}_{k-1}$ can be used as the termination condition. However, this condition is not used in simulation experiments later in order to make the experimental conditions equivalent.

6. Evaluation by simulation experiment

In this section, a comparative evaluation of the conventional method and the proposed method is performed through simulation experiments. In addition, the robustness of the proposed method in the singular configuration is verified.

6.1. Experimental conditions

Figure 5 shows the 9-DOF articulated robot used in this experiment. The length of each link is 200 mm, and the total length is $L = 2000$ mm. Let $\boldsymbol{e}_1(\boldsymbol{q}_k)$ be the error vector between the end effector and Target 1, and let $\boldsymbol{e}_2(\boldsymbol{q}_k)$ be the error vector between the center of the sixth link and Target 2. Target 1 was given a higher priority than Target 2. The δ parameter and the components of $\boldsymbol{K}$ were determined as follows using the total length L of the robot:

$$K_{1f} = 1, \quad K_{1m} = \frac{K_{1f}L^2}{\pi} = \frac{L^2}{\pi} \tag{36}$$

$$K_{2f} = 1, \quad K_{2m} = 0 \tag{37}$$

$$\delta = 10^{-3}L\sqrt{K_{1f}K_{1m}} = \frac{10^{-3}L^2}{\sqrt{\pi}}. \tag{38}$$

The above equation was solved using the physical dimension of each parameter and a regulation performed by the authors. In this experiment, $K_{2m} = 0$ because the virtual torsion spring was not attached to the error vector $\boldsymbol{e}_2(\boldsymbol{q}_k)$. The target settings used in this experiment are shown in Figure 6, and a singular configuration was used for the initial posture. The target position and orientation were determined from

$${}^{\mathrm{d}}\boldsymbol{p}_1 = \begin{bmatrix} 1200 \\ 0 \\ 1000 \end{bmatrix} \mathrm{mm}, \; {}^{\mathrm{d}}\boldsymbol{R}_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \tag{39}$$

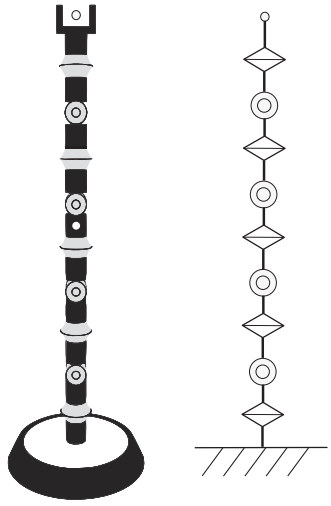


Figure 5. 9-DOF articulated robot used in the simulation experiment.

$${}^{\mathrm{d}}\boldsymbol{p}_2 = \begin{cases} \begin{bmatrix} 600 \\ 0 \\ 200 \end{bmatrix} \mathrm{mm} & \text{(Test1)} \\ \begin{bmatrix} 600 \\ 0 \\ 800 \end{bmatrix} \mathrm{mm} & \text{(Test2)} \\ \begin{bmatrix} 600 \\ 0 \\ 1200 \end{bmatrix} \mathrm{mm} & \text{(Test3)} \\ \begin{bmatrix} 600 \\ 0 \\ 1600 \end{bmatrix} \mathrm{mm} & \text{(Test4)} \end{cases} \tag{40}$$

In this experiment, the iterative calculation ends when $V_1(\boldsymbol{q}_k) < 0.1$, which means that the error corresponding to Target 1 (highly prioritized constraint) is converged.

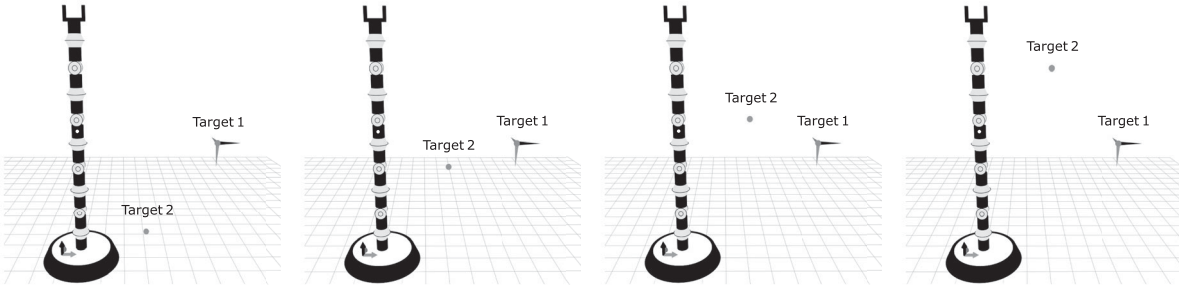
6.2. Evaluation by comparative experiment

In this section, the prioritized inverse kinematics problem shown in Figure 6 is solved by both the proposed method and the conventional method [21], and the results and calculation time for both methods are evaluated.

Figures 7 and 8 show the calculation results for the proposed method and the conventional method, respectively. In this experiment, the α term was included in Equation (32) for the conventional method as

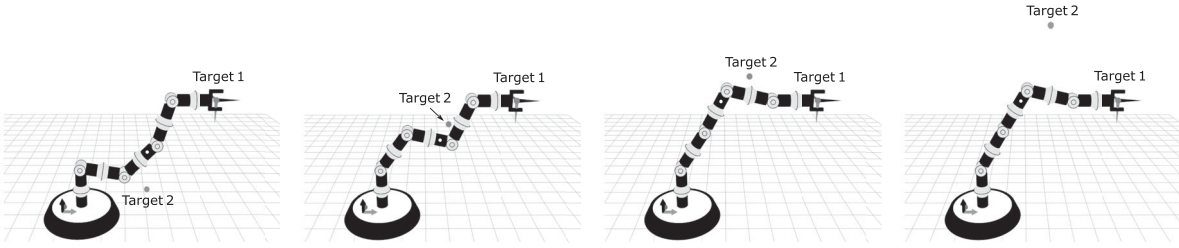
$$\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + \alpha\boldsymbol{e}_1(\boldsymbol{q}_k), \quad \boldsymbol{\lambda}_0 = \boldsymbol{0}. \tag{41}$$

In the original paper, this α term was not considered, but here the conventional method was found to converge



(a) Test 1 (b) Test 2 (c) Test 3 (d) Test 4

Figure 6. Four types of targets used in the comparison.

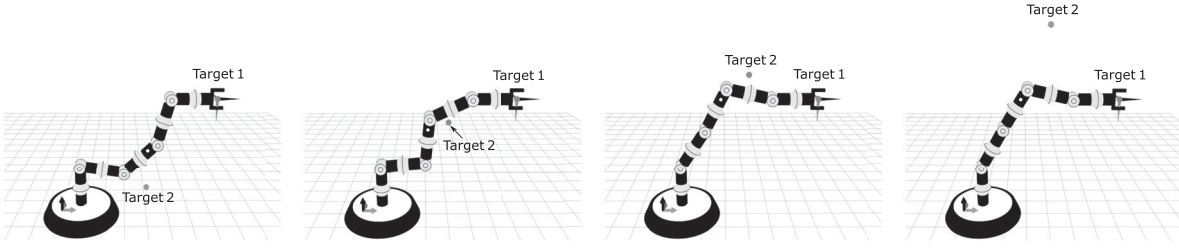


(a) Result for Test 1 (b) Result for Test 2 (c) Result for Test 3 (d) Result for Test 4

Figure 7. Calculation results for the four experimental settings obtained using the proposed method.

rapidly when α was less than 1. The results for $\alpha = 0.4$ are shown in Figure 8, although the final results were the same when $\alpha = 0.2, 0.3, \ldots, 0.7$ (not shown).

In the comparison of Figures 7 and 8, the same result was obtained in Tests 1, 3, and 4. In Test 2, the flexion direction of the joints is different; however, this difference was unintentional and did not significantly affect the results. Therefore, there is no difference between the proposed method and the conventional method in terms of the obtained joints displacement. However, the computation time (number of iterations) required for convergence and the robustness are different. The numbers of iterations required for convergence are summarized in Table 2. In the conventional method, the end effector

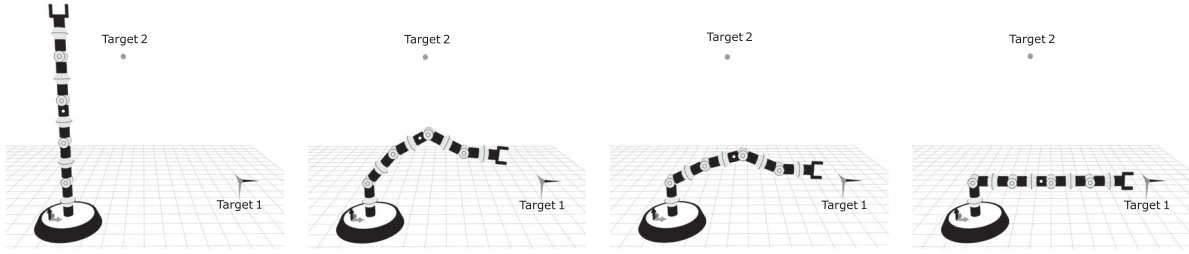


(a) Result for Test 1 (b) Result for Test 2 (c) Result for Test 3 (d) Result for Test 4

Figure 8. Calculation results for the four experimental settings obtained using the conventional method ($\alpha = 0.4$).

Table 2. Numbers of iterations required for convergence using the proposed method and the conventional method.

		Conventional method					
	Proposed method	$\alpha = 0.2$	$\alpha = 0.3$	$\alpha = 0.4$	$\alpha = 0.5$	$\alpha = 0.6$	$\alpha = 0.7$
Test1	31	64	40	29	26	28	37
Test2	21	41	58	66	77	90	105
Test3	15	58	39	32	35	72	147
Test4	15	63	45	34	28	35	45
Average	20.5	56.5	45.5	40.25	41.5	56.25	83.5



(a) $k = 0,\ \zeta_k = 1$ (b) $k = 5,\ \zeta_k = 0.75$ (c) $k = 13,\ \zeta_k = 0.25$ (d) $k = 17,\ \zeta_k = 0$

Figure 9. Calculation results of singularity robustness test.

gradually converges while vibrating around Target 1 due to the effect of the Lagrange multiplier $\boldsymbol{\lambda}_k$. The larger α is, the larger the vibration becomes, and conversely, the smaller α is, the smaller the vibration becomes. When α is larger than the proper value, convergence is slow due to the influence of the vibration, and many iterations are required. Conversely, if α is lower than the optimal range of values, there is almost no delay due to vibration, but the diminution of $V_1(\boldsymbol{q}_k)$ worsens and many iterations are required before $V_1(\boldsymbol{q}_k)$ falls below the threshold (0.1). For the average values in Table 2, the iteration count of the conventional method was smallest when $\alpha = 0.4$, but the proposed method converged faster than the conventional method. Therefore, it can be said that the proposed method converges faster than the conventional method.

In addition, when $\alpha > 0.9$, the results obtained with the conventional method were seen to diverge. In other words, in the conventional method, whether the calculation results converge or not depends on the value of α. For these reasons, the proposed method is more robust than the conventional method.

In this experiment, the joint displacement did not reach the joint limit during the iterative calculation. If the ith joint exceeds the joint limit during the iterative calculation, the following procedures are required.

(1) Reset the joint displacement at the limit of the joint range.
(2) In the next step, replace the ith column of the Jacobian matrix $\boldsymbol{J}_k$ with a zero vector.

In addition, the authors have proposed a numerical method for prioritized inverse kinematics using the method of section [37], but in this experiment, it cannot be used, because Target 2 does not include the position and orientation simultaneously.

6.3. Robustness verification of proposed method in singular configurations

In this section, the robustness of the proposed method in a singular configuration is verified. Targets 1 and 2 were set as

$$ {}^{\mathrm{d}}\boldsymbol{p}_1 = \begin{bmatrix} 1800 \\ 0 \\ 400 \end{bmatrix} \text{mm},\ {}^{\mathrm{d}}\boldsymbol{R}_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \tag{42} $$

$$ {}^{\mathrm{d}}\boldsymbol{p}_2 = \begin{bmatrix} 600 \\ 0 \\ 1600 \end{bmatrix} \text{mm}. \tag{43} $$

When V_{1k} is minimized under this condition, the robot takes a singular configuration. Figure 9 shows the result of solving the prioritized inverse kinematics problem using the proposed method. The diagonal matrix $\delta \boldsymbol{I}_n$ included in matrix $\boldsymbol{D}_k$ alleviates the ill-conditioning of $\boldsymbol{D}_k$ [31]. Therefore, the joint displacements (inverse kinematic solution) which minimize $V_1(\boldsymbol{q})$ can be obtained even in a singular configuration.

7. Conclusion

In this paper, a new numerical method for inverse kinematics with prioritized multiple targets was proposed. The proposed method is easier to implement than the conventional method because the addition of angle-axis vectors is not required in the calculation process. In addition, the proposed method has a lower calculation time and greater robustness to parameter variation. The technique proposed in this paper is expected to be applicable to redundant robots, to help avoid collisions between links and obstacles using the redundancy.

Acknowledgements

This paper is based on results obtained from a project, JPNP18002, subsidized by the New Energy and Industrial Technology Development Organization (NEDO).

Disclosure statement

No potential conflict of interest was reported by the authors.

Notes on contributors

Masanori Sekiguchi received a B.E. and M.E. degrees from Tokyo Metropolitan University in 2017 and 2019, respectively. Since 2019, he has been a Ph.D. student at the Department of Systems Design, Tokyo Metropolitan University. His research interests include kinematics, motion planning and industrial applications.

Naoyuki Takesue received B.E. and M.E degrees from The University of Electro-Communications, respectively, in 1995 and 1997. In 2000, he received a Ph.D. in Engineering from Osaka University. He joined Osaka University in 2000 and then joined Nagoya Institute of Technology in 2003. He joined Tokyo Metropolitan University in 2008 as an associate professor. His research interests include mechanism design, motion control of manipulator, industrial applications, physically-assistive robots, and aerial/aquatic/mobile robots.

ORCID


Naoyuki Takesue http://orcid.org/0000-0002-8029-5480

References

[1] Sciavicco L, Siciliano B. A solution algorithm to the inverse kinematic problem for redundant manipulators. IEEE J Rob Autom. 1988;4(4):403–410.
[2] Nakamura Y, Hanafusa H. Optimal redundancy control of robot manipulators. Int J Rob Res. 1987;6(1):32–42.
[3] Chiacchio P, Chiaverini S, Sciavicco L, et al. Closed-Loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy. Int J Rob Res. 1991;10(4):410–425.
[4] An S-I, Lee D. Prioritized inverse kinematics with multiple task definitions. IEEE Int Conf Robot Autom. 2015:1423–1430.
[5] An S-I, Lee D. Prioritized inverse kinematics: generalization. IEEE Rob Autom Lett. 2019;4(4):3537–3544.
[6] Baerlocher P, Boulic R. Task-priority formulations for the kinematic control of highly redundant articulated structures. IEEE/RSJ Int Conf Intell Rob Syst. 1998;1:323–329.
[7] Del Prete A, Romano F, Natale L, et al. Prioritized optimal control. IEEE Int Conf Robot Autom. 2014:2540–2545.
[8] Romano F, Del Prete A, Mansard N, et al. Prioritized optimal control: a hierarchical differential dynamic programming approach. IEEE Int Conf Robot Autom. 2015:3590–3595.
[9] Sugihara T. Numerical solution of inverse kinematics. J Rob Soc Jpn. 2016;34(3):167–173, (in Japanese).
[10] Asfour T, Dillmann R. Human-like motion of a humanoid robot arm based on a closed-form solution of the inverse kinematics problem. IEEE/RSJ Int Conf Intell Rob Syst. 2003;2:1407–1412.
[11] Shimizu M, Kakuya H, Yoon WK, et al. Analytical inverse kinematics for 7 DOF redundant manipulators with joint limits. J Rob Soc Jpn. 2007;25(4):606–617, (in Japanese).
[12] Nakamura Y, Hanafusa H, Yoshikawa T. Task-priority based redundancy control of robot manipulators. Int J Rob Res. 1987;6(2):3–15.
[13] Maciejewski AA, Klein CA. Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. Int J Rob Res. 1985;4(3):109–117.
[14] Siciliano B, Slotine JE. A general framework for managing multiple tasks in highly redundant robotic systems. Int Conf Adv Rob. 1991;2:1211–1216.
[15] Chiaverini S. Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. IEEE Trans Rob Autom. 1997;13(3):398–410.
[16] Park J, Choi Y, Chung WK. Multiple tasks kinematics using weighted pseudo-inverse for kinematically redundant manipulators. IEEE Int Conf Robot Autom. 2001;4:4041–4047.
[17] Choi Y, Oh Y, Oh SR, et al. Multiple tasks manipulation for a robotic manipulator. Adv Rob. 2004;18(6):637–653.
[18] Wampler CW. Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods. IEEE Trans Syst Man Cybern. 1986;16(1):93–101.
[19] Nakamura Y, Hanafusa H. Inverse kinematic solutions with singularity robustness for robot manipulator control. J Dyn Syst Meas Control. 1986;108(3):163–171.
[20] Deo AS, Walker ID. Overview of damped least-squares methods for inverse kinematics of robot manipulators. J Intell Rob Syst. 1995;14(1):43–68.
[21] Sugihara T. Robust solution of prioritized inverse kinematics based on hestenes-powell multiplier method. IEEE/RSJ Int Conf Intell Rob Syst. 2014:510–515.
[22] Tanaka K, Sugihara T. Dynamically consistent motion design of a humanoid robot even at the limit of kinematics. IEEE-RAS Int Conf Humanoid Rob. 2014:1007–1012.
[23] Luh J, Walker M, Paul R. Resolved-acceleration control of mechanical manipulators. IEEE Trans Automat Contr. 1980;25(3):468–474.
[24] Arimoto S, Sekimoto M, Hashiguchi H, et al. Natural resolution of ill-posedness of inverse kinematics for redundant robots: a challenge to Bernstein's degrees-of-freedom problem. Adv Rob. 2005;19(4):401–434.
[25] Arimoto S, Sekimoto M, Ozawa R. A challenge to Bernstein's degrees-of-freedom problem in both cases of human and robotic multi-joint movements. IEICE Trans Fundam Electron Commun Comput Sci. 2005;E88-A(10):2484–2495.
[26] Arimoto S, Hashiguchi H, Sekimoto M, et al. Generation of natural motions for redundant multi-joint system: a differential-geometric approach based upon the principle of least actions. J Rob Syst. 2005;22(11):583–605.
[27] Arimoto S, Sekimoto M. Human-like movements of robotic arms with redundant DOFs: virtual spring-damper hypothesis to tackle the Bernstein problem. IEEE Int Conf Rob Autom. 2006:1860–1866.
[28] Sekimoto M, Arimoto S. Experimental study on control method for robot arms with redundant joints based upon virtual spring-Damper hypothesis. J Rob Soc Jpn. 2007;25(5):785–791, (in Japanese).
[29] Sadeghian H, Villani L, Keshmiri M, et al. Dynamic multi-priority control in redundant robotic systems. Robotica. 2013;31(7):1155–1167.
[30] Ott C, Dietrich A, Albu-Schaffer A. Prioritized multi-task compliance control of redundant manipulators. Automatica. 2015;53:416–423.
[31] Sugihara T. Solvability-unconcerned inverse kinematics by Levenberg–Marquardt method. IEEE Trans Rob. 2011;27(5):984–991.

[32] Sekiguchi M, Takesue N. Stable numerical solution of inverse kinematics in singular posture and unsolvable problem based on minimization of elastic energy of virtual spring. J Rob Soc Jpn. 2018;36(9):645–653, (in Japanese).
[33] Krishnamoorthy A, Menon D. Matrix inversion using cholesky decomposition. 2013 Signal Process: Algorithms, Architectures, Arrangements, Appl. 2013: 70–72.
[34] Bojanczyk AW, Brent RP, van Dooren P, et al. A note on downdating the cholesky factorization. SIAM J Sci Stat Comput. 1987;8(3):210–221.
[35] Higham NJ. Cholesky factorization. WIREs Comput Stat. 2009;1(2):251–254.
[36] Li J, Widlund OB. FETI-DP, BDDC, and block Cholesky methods. Int J Numer Methods Eng. 2006;66(2):250–271.
[37] Sekiguchi M, Takesue N. Numerical solution of prioritized inverse kinematics using the method of sections. J Rob Soc Jpn. 2019;37(8):711–717, (in Japanese).
[38] Sekiguchi M, Takesue N. A method for calculating angle-axis vector at singular points and a proposal of extended angle-axis vector. J Rob Soc Jpn. 2019;37(8):726–734, (in Japanese).

Appendices

Appendix 1. Angle-axis vector

For any rotation matrix $\boldsymbol{R} = [r_{ij}] \in SO(3)$, the vector $\boldsymbol{\ell}$ is defined as

$$\boldsymbol{\ell} := \begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix} \in \mathbb{R}^3. \tag{A1}$$

Then the angle-axis vector is expressed as

$$\boldsymbol{\phi}(\boldsymbol{R}) = \begin{cases} \dfrac{\operatorname{atan2}\left(\|\boldsymbol{\ell}\|,\ \operatorname{tr}\boldsymbol{R} - 1\right)}{\|\boldsymbol{\ell}\|}\boldsymbol{\ell} & (\|\boldsymbol{\ell}\| \neq 0) \\ \boldsymbol{0} & (\|\boldsymbol{\ell}\| = 0 \text{ and } \operatorname{tr}\boldsymbol{R} = 3) \\ \text{see below} & (\|\boldsymbol{\ell}\| = 0 \text{ and } \operatorname{tr}\boldsymbol{R} = -1) \end{cases} \tag{A2}$$

where $\operatorname{tr}\boldsymbol{R} = r_{11} + r_{22} + r_{33}$. If $\|\boldsymbol{\ell}\| = 0$ and $\operatorname{tr}\boldsymbol{R} = -1$, the rotation matrix $\boldsymbol{R}$ is a symmetric matrix. The angle-axis vector where $\|\boldsymbol{\ell}\| = 0$ and $\operatorname{tr}\boldsymbol{R} = -1$ can be calculated as follows according to the sign of the non-diagonal component of the matrix $\boldsymbol{R}$.

(1) $r_{21} = r_{12} \geq 0, r_{32} = r_{23} \geq 0, r_{13} = r_{31} \geq 0$

$$\boldsymbol{\phi}(\boldsymbol{R}) = \pm\frac{\sqrt{2}\pi}{2}\begin{bmatrix} \sqrt{r_{11}+1} \\ \sqrt{r_{22}+1} \\ \sqrt{r_{33}+1} \end{bmatrix}, \tag{A3}$$

(2) $r_{21} = r_{12} \geq 0, r_{32} = r_{23} < 0, r_{13} = r_{31} \leq 0$

$$\boldsymbol{\phi}(\boldsymbol{R}) = \pm\frac{\sqrt{2}\pi}{2}\begin{bmatrix} \sqrt{r_{11}+1} \\ \sqrt{r_{22}+1} \\ -\sqrt{r_{33}+1} \end{bmatrix}, \tag{A4}$$

(3) $r_{21} = r_{12} < 0, r_{32} = r_{23} \leq 0, r_{13} = r_{31} \geq 0$

$$\boldsymbol{\phi}(\boldsymbol{R}) = \pm\frac{\sqrt{2}\pi}{2}\begin{bmatrix} \sqrt{r_{11}+1} \\ -\sqrt{r_{22}+1} \\ \sqrt{r_{33}+1} \end{bmatrix}, \tag{A5}$$

(4) $r_{21} = r_{12} \leq 0, r_{32} = r_{23} \geq 0, r_{13} = r_{31} < 0$

$$\boldsymbol{\phi}(\boldsymbol{R}) = \pm\frac{\sqrt{2}\pi}{2}\begin{bmatrix} -\sqrt{r_{11}+1} \\ \sqrt{r_{22}+1} \\ \sqrt{r_{33}+1} \end{bmatrix}. \tag{A6}$$

The derivation of the angle-axis vector is described in [38].

Appendix 2. Generalization of target number

Let the number of targets be m, and let the error vector, Jacobian matrix, and spring constant matrix corresponding to each target be $\boldsymbol{e}_i(\boldsymbol{q})$, $\boldsymbol{J}_i(\boldsymbol{q})$, and $\boldsymbol{K}_i$ $(i = 1, 2, \ldots, m)$, respectively. These are summarized as

$$\boldsymbol{e}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{e}_1(\boldsymbol{q}) \\ \vdots \\ \boldsymbol{e}_m(\boldsymbol{q}) \end{bmatrix} \in \mathbb{R}^{6m} \tag{A7}$$

$$\boldsymbol{J}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{J}_1(\boldsymbol{q}) \\ \vdots \\ \boldsymbol{J}_m(\boldsymbol{q}) \end{bmatrix} \in \mathbb{R}^{6m\times n} \tag{A8}$$

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}_1 & \cdots & \boldsymbol{O} \\ \vdots & \ddots & \vdots \\ \boldsymbol{O} & \cdots & \boldsymbol{K}_m \end{bmatrix} \in \mathbb{R}^{6m\times 6m}. \tag{A9}$$

The virtual torque $\boldsymbol{\tau}(\boldsymbol{q})$ generated at each joint is expressed as

$$\boldsymbol{\tau}(\boldsymbol{q}) = \boldsymbol{J}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{K}\boldsymbol{e}(\boldsymbol{q}) \in \mathbb{R}^n. \tag{A10}$$