IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Handling Transitions Across Singularities for UR-Like Serial Robots

Ivan Boschi, *Graduate Student Member, IEEE*, Alessandro De Toni, *Graduate Student Member, IEEE*, Roberto Di Leva, Edoardo Ida', *Member, IEEE*, and Marco Carricato, *Senior Member, IEEE*

***Abstract*—The inverse kinematics (IK) of serial robots admits multiple solutions, making the selection of the desired one potentially challenging depending on the application at hand. For robots with a non-cuspidal architecture, a suitable strategy is to partition the joint space into independent regions, known as Uniqueness Domains (UDs), which are separated by surfaces defined by singular configurations. Within each UD, a single IK-solution branch corresponds to a specific robot posture. In this case, when an assigned task-space trajectory requires the robot to transition between UDs, a singular configuration must be crossed. Despite the practical importance of this issue, the existing literature lacks straightforward techniques for enabling such transitions. This paper proposes a method that facilitates switching between IK-solution branches when a task-space trajectory requires crossing a singularity, ensuring continuity and differentiability of the joint variables. The proposed method is evaluated against competing ones, both in simulation and experimentally, showing significant advantages.**

***Index Terms*—Inverse kinematics, serial robots, singularities.**

I. Introduction

When a robot is assigned a task in Cartesian space, the inverse-kinematics (IK) problem must be solved. The IK problem generally yields multiple solutions, corresponding to distinct branches (namely, regions in the joint space). Selecting one solution over another is crucial in several applications, e.g., to avoid potential collisions between the robot links and the manipulated object and/or the surrounding environment.

Unlike serial robots equipped with a spherical wrist, where the orientation and position sub-problems within the IK are decoupled, most cobots, such as the widespread Universal Robot (UR) [1], [2], usually lack this feature, making IK inherently more complex. For non-cuspidal architectures, like the UR one, the joint space may be decomposed into independent regions known as Uniqueness Domains (UDs) [3]. These regions are separated by boundaries associated with singularities, i.e. configurations in which the robot loses some degrees of freedom. Within each UD, a unique IK-solution branch corresponds to a specific robot posture [1]. A posture is defined as a joint configuration that allows the end-effector (EE) to reach a given pose and can be classified in a non-ambiguous way according to the Jacobian determinant structure [3]. For non-cuspidal robots, transitioning from one posture to another necessarily involves passing through a singularity.

Several strategies have been proposed to address this issue, such as the damped least-squares (DLS) method, although it leads to deviations from the planned path [4]. To allow a more structured handling of singularities, a technique based on rewriting the robot kinematics into a 'normal form' was proposed in [5], [6]; however, the resulting equations can be analytically complex. In [7], the IK was formulated as an optimization problem, avoiding Jacobian inversion and related singularity instabilities. However, this approach yields approximate solutions and is mainly used for redundant robots, while alternative methods may be more efficient for non-redundant robots. Similar considerations apply to the method proposed in [8], where artificial neural networks are employed to learn a direct mapping between task space and joint space. The work presented in [9] discusses whether a singularity surface can be crossed or not, but does not provide details about how to cross it. Alternatively, modifications to the Jacobian matrix by eliminating singular components to transform it into a full-rank matrix were proposed in [10], [11], assuming that the dynamic model of the system is known; in the event of an infeasible trajectory, [11] provides a re-planning strategy technique. Likewise, a transformation of the workspace into a non-singular form was proposed in [12], although this technique does not apply to all types of singularities.

Algorithms capable of computing all IK-solutions for general 6R robots have also been developed [13], [14]. While such methods return all joint configurations, they do not address the problem of selecting a robot posture among all the available solutions. For serial robots with a UR-like architecture, a decision-based approach was introduced in [15]; however, this method does not guarantee control over the robot's posture and does not allow transitions through singularities. To ensure robot posture consistency, the authors in [1] proposed a method based on index assignment to solve the IK problem, but only hinted at how to deal with singularity transitions.

Building on the definition of UDs in [3] and the work of [1] on transitions through singularities, the current study observes that, when passing through a singularity, the joint-space solution must be able to switch between UDs to maintain continuity and differentiability of the joint variables. Specifically, this paper proposes a computationally efficient algorithm capable of specifying postures and ensuring continuous transitions through singularities, demonstrating its effectiveness through experimental validation.

Received 30 July 2025; accepted 28 November 2025. Date of publication 12 January 2026; date of current version 22 January 2026. This article was recommended for publication by Associate Editor C. Gaz and Editor L. Pallottino upon evaluation of the reviewers' comments. This work was supported by FISA2023 Project *APACHE* under Grant CUP J53C25000520001. *(Corresponding author: Marco Carricato.)*

The authors are with the Department of Industrial Engineering, University of Bologna, 40136 Bologna, Italy (e-mail: ivan.boschi3@unibo.it; a.detoni@unibo.it; roberto.dileva@unibo.it; edoardo.ida2@unibo.it; marco.carricato@unibo.it).

This article has supplementary downloadable material available at https://doi.org/10.1109/LRA.2026.3653295, provided by the authors.

Digital Object Identifier 10.1109/LRA.2026.3653295

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

TABLE I
DH PARAMETERS OF UR-LIKE ROBOTS

Joint t	1	2	3	4	5	6
a_t [m]	0	a_2	a_3	0	0	0
d_t [m]	d_1	0	0	d_4	d_5	d_6
α_t [rad]	$\pi/2$	0	0	$\pi/2$	$-\pi/2$	0

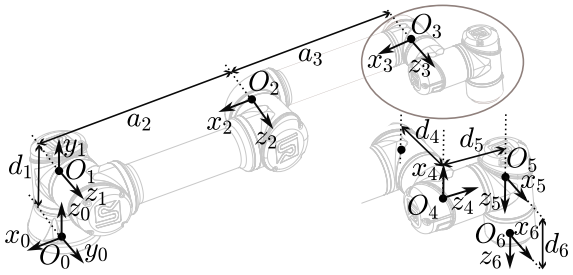

Fig. 1. UR architecture with DH parameters.

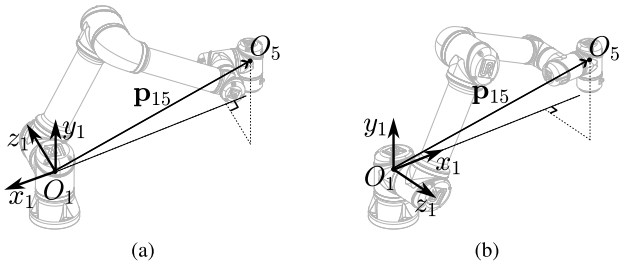

Fig. 2. Shoulder postures: (a) left ($i = 1$); (b) right ($i = -1$).

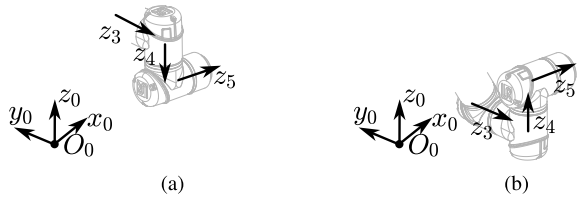

Fig. 3. Wrist postures: (a) left ($j = -1$); (b) right ($j = 1$).

The remainder of the paper is organized as follows. Section II outlines the theoretical background, while Section III details the proposed approach and compares it with existing methods. Section IV presents the experimental validation, and Section V draws conclusions.

II. THEORETICAL BACKGROUND

A. IK Resolution

In the following, the notation $\cos(q_l) = c_l$, $\sin(q_l) = s_l$, $\cos(q_l + \cdots + q_m) = c_{l\ldots m}$, $\sin(q_l + \cdots + q_m) = s_{l\ldots m}$ is adopted. The DH parameters for a UR-like robot, according to the distal variant [16], are reported in Table I and illustrated in Fig. 1. These parameters are embedded in the homogeneous transformation matrix $\mathbf{T}_{t-1,t}$, which encodes the orientation of frame t with respect to $t-1$, represented by the rotation matrix $\mathbf{R}_{t-1,t}$, and the position vector from origin O_{t-1} to origin O_t, expressed in frame $t-1$, denoted by $\{\mathbf{p}_{t,t-1}\}_{t-1}$. By considering the chain of transformations $\prod_{t=1}^{6} \mathbf{T}_{t-1,t}$, the IK problem can be approached in several ways. Here, we summarize the closed-form solution available in [1].

Given the orientation $\mathbf{R}_{06}$ and the position $\{\mathbf{p}_{06}\}_0 = \begin{bmatrix} p_{06,x} & p_{06,y} & p_{06,z} \end{bmatrix}^T$, $\{\mathbf{p}_{05}\}_0$ can be determined analytically, through the relation:

$$\{\mathbf{p}_{05}\}_0 = [p_{05,x}\, p_{05,y}\, p_{05,z}]^T = \{\mathbf{p}_{06}\}_0 - \mathbf{R}_{06}[0\,0\,d_6]^T. \quad (1)$$

This position enables the determination of joint variable q_1. Due to the robot structure, two valid configurations typically arise for q_1 (illustrated in Fig. 2), distinguished by the index $i \in \{-1, 1\}$, defined as:

$$q_1 = \beta_1 + i\beta_2 + \frac{\pi}{2}, \quad i \in \{-1, 1\}, \quad (2)$$

where

$$\beta_1 = \operatorname{atan2}(p_{05,y}, p_{05,x}), \quad \beta_2 = \cos^{-1} \frac{d_4}{\sqrt{p_{05,x}^2 + p_{05,y}^2}}. \quad (3)$$

Considering

$$\{\mathbf{p}_{16}\}_1 = \mathbf{R}_{01}^T(q_1)\{\mathbf{p}_{06}\}_0 + [0 \quad -d_1 \quad 0]^T \quad (4)$$

it can be demonstrated that

$$[0\ \ 0\ \ 1]\{\mathbf{p}_{16}\}_1 = d_4 + d_6 c_5. \quad (5)$$

Substituting (4) in (5) leads to a trigonometric equation in q_5, which admits two distinct solutions, distinguished by the index $j \in \{-1, 1\}$:

$$q_5 = j \cos^{-1} \frac{p_{06,x} s_1 - p_{06,y} c_1 - d_4}{d_6}, \quad j = \{-1, 1\}. \quad (6)$$

The corresponding robot configurations are shown in Fig. 3.

By analyzing the elements of $\mathbf{R}_{16}$, it can be observed that $\mathbf{R}_{16}(3,1)$ and $\mathbf{R}_{16}(3,2)$—which correspond to the entries in the third row and the first and second columns, respectively—are given by:

$$\mathbf{R}_{16}(3,1) = s_5 c_6, \quad \mathbf{R}_{16}(3,2) = -s_5 s_6. \quad (7)$$

Computing $\mathbf{R}_{16} = \mathbf{R}_{01}^T \mathbf{R}_{06}$, one can uniquely determine the joint variable q_6 as:

$$q_6 = \operatorname{atan2}\left(-\mathbf{R}_{16}(3,2)\operatorname{sgn}(s_5), \mathbf{R}_{16}(3,1)\operatorname{sgn}(s_5)\right). \quad (8)$$

Subsequently, since joint axes 2, 3, and 4 are mutually parallel, they form a planar 3R mechanism. Considering that $\{\mathbf{p}_{13}\}_1 = [p_{13,x_1}\ \ p_{13,y_1}\ \ p_{13,z_1}]^T$ is given by:

$$\{\mathbf{p}_{13}\}_1 = [0\ \ 0\ \ -d_4]^T + \mathbf{R}_{15}[0\ \ d_5\ \ 0]^T + \mathbf{R}_{16}[0\ \ 0\ \ -d_6]^T + \mathbf{R}_{01}^T\{\mathbf{p}_{06}\}_0 + \mathbf{R}_{01}^T[0\ \ 0\ \ -d_1]^T \quad (9)$$

and applying the cosine law to the 3R mechanism—i.e., $\|\{\mathbf{p}_{13}\}_1\|^2 = a_2^2 + a_3^2 - 2a_2a_3\cos(\pi - q_3)$— the joint variable q_3 can be computed as:

$$q_3 = k \cos^{-1} \frac{\|\{\mathbf{p}_{13}\}_1\|^2 - a_2^2 - a_3^2}{2a_2a_3}, \quad k \in \{-1, 1\}. \quad (10)$$

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

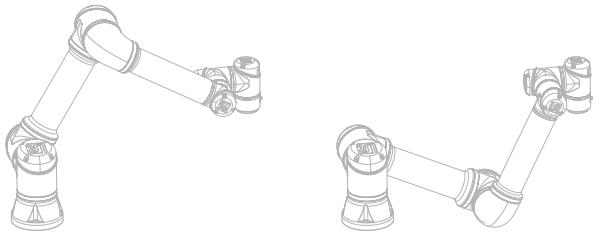

Fig. 4. Elbow postures: (a) up ($k = 1, i = 1$); (b) down ($k = -1, i = 1$).

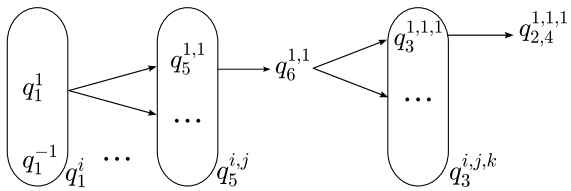

Fig. 5. Solution branches.

Like (6), (10) admits two possible solutions, obtained by setting $k \in \{-1, 1\}$. The corresponding robot configurations are illustrated in Fig. 4.

At this stage, the robot configuration is fully defined, as the remaining variables q_2 and q_4 can be uniquely determined. By projecting $\{\mathbf{p}_{13}\}_1$ on axes x_1 and y_1, q_2 can be found as:

$$q_2 = \text{atan2}\left(\frac{-a_3 s_3 p_{13,x_1} + (a_2 + a_3 c_3) p_{13,y_1}}{a_2^2 + a_3^2 + 2a_2 a_3 c_3}, \frac{(a_2 + a_3 c_3) p_{13,x_1} + a_3 s_3 p_{13,y_1}}{a_2^2 + a_3^2 + 2a_2 a_3 c_3}\right) \tag{11}$$

and, considering that $\mathbf{R}_{34} = \mathbf{R}_{03}^T \mathbf{R}_{06} \mathbf{R}_{46}^T$, q_4 is given by:

$$q_4 = \text{atan2}\left(\mathbf{R}_{34}(2,1), \mathbf{R}_{34}(1,1)\right), \tag{12}$$

where $\mathbf{R}_{34}(2,1)$ and $\mathbf{R}_{34}(1,1)$ are the elements in the first column and in the second and first rows of $\mathbf{R}_{34}$, respectively.

B. IK-Solution Branches and Postures

Based on the IK formulation presented above, the solution space is composed of multiple discrete branches. Fig. 5 illustrates a representative branch of the solution tree, corresponding to the index combination $(i, j, k) = (1, 1, 1)$. Although only one branch is shown for clarity, the complete solution tree includes 8 distinct configurations resulting from all combinations of $(i, j, k) \in \{-1, 1\}$.

The determinant of the Jacobian matrix $\mathbf{J}$ allows the characterization of each branch and its relation to singular configurations. In accordance with [15], the determinant is:

$$\det(\mathbf{J}) = |a_2| a_2 a_3 P_1 P_2 P_3, \tag{13}$$

with

$$P_1 = s_3,\ P_2 = s_5,\ P_3 = (a_2 c_2 + a_3 c_{23} + d_5 s_{234})/|a_2|. \tag{14}$$

Owing to the non-cuspidal nature of the UR-type architecture [17], the joint space can be partitioned into separate regions—referred to as UDs [3]—characterized by joint-angle sets corresponding to a single IK-solution branch. Each IK-solution branch is also associated with a so-called "posture", which can be classified according to the signs of P_1, P_2, and P_3, which in turn are directly related to the index triplet (i, j, k)[1]. The classification rules are as follows:

- shoulder-right (Fig. 2(b)) or left (Fig. 2(a)) if
$$P_3 |a_2| = [1\ \ 0\ \ 0]\{\mathbf{p}_{15}\}_1$$
is positive or negative, respectively, so that $i = \text{sgn}(-P_3)$;
- wrist-right (Fig. 3(b)) or left (Fig. 3(a)) if
$$P_2 = \{\mathbf{z}_4\}_0^T (\{\tilde{\mathbf{z}}_3\}_0 \{\mathbf{z}_5\}_0)$$
is positive or negative, respectively, so that $j = \text{sgn}(P_2)$ ($\{\tilde{\mathbf{z}}_3\}_0$ denotes the skew-symmetric matrix of $\{\mathbf{z}_3\}_0$);
- elbow-up (Fig. 4(a)) or down (Fig. 4(b)) if, respectively, $ik > 0$ or < 0, where $k = \text{sgn}(P_1)$.

The indices (i, j, k) are closely related to the signs of the determinant factors and are thus linked to singular configurations (a detailed discussion of singularities for UR-like robots can be found in [18]). When one of the factors approaches zero (corresponding to a singular configuration), the sign—and consequently the associated index—becomes undefined. This indicates that two solutions of the corresponding joint variable, which would normally be distinct, actually coincide.

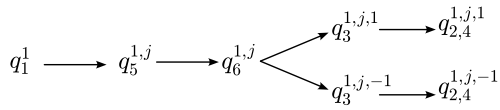

Fig. 6. A solution branch in a wrist singularity ($i = 1$, j undefined, $k = \pm 1$).

III. Transitions Across Singularities

A. Strategy

When the robot reaches a singularity, the structure of the IK-solution tree changes and some branches momentarily converge. For example, in the case of a wrist singularity (i.e., when $P_2 = 0$), the two solutions for q_5, identified by index j, numerically coincide, as shown in Fig. 6.

A key aspect in trajectory planning through singularities is determining whether the solution branch (and thus the posture), identified by the index triplet (i, j, k), should be preserved after crossing the singularity, or a transition to a different branch is necessary to ensure continuity and differentiability of the joint variables with respect to the path parameter.

The authors of [1] propose a descriptive example in which keeping the index triplet (i, j, k) constant results in a reasonably expected outcome: when the robot reaches a workspace boundary singularity ($P_1 = 0$) from a start configuration, it can go back to the latter with the same posture. However, this procedure does not generalize well to all singularities, as demonstrated in the following case, which involves a transition through a wrist singularity. Suppose that the reference point O_6, starting from position A depicted in Fig. 7(a), must follow a circular arc of radius d_6, centered at point O_5, and subtending an angle of $\frac{2}{3}\pi$ radians, in order to reach point B. The task is assigned in Cartesian space and it is straightforward to see that it can be accomplished by moving q_5 from $\frac{\pi}{3}$ to $-\frac{\pi}{3}$ radians,

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

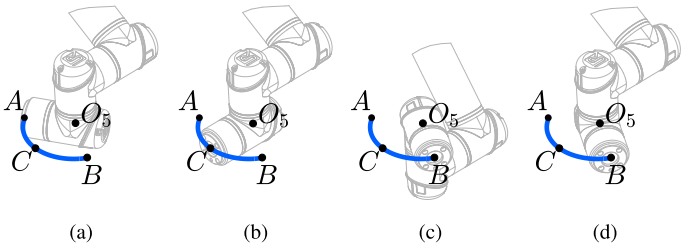

Fig. 7. Circular path followed by point O_6, centered in O_5 and having radius d_6: (a) start pose; (b) singular pose; (c) final pose with no change in solution branch; (d) final pose with a change in solution branch.

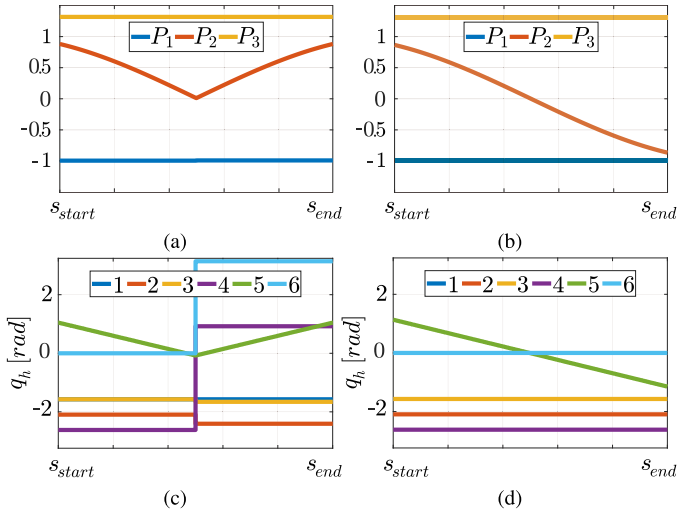

Fig. 8. Evolution of P_n $(n = 1, 2, 3)$ and q_h $(h = 1, \ldots, 6)$ along the path illustrated in Fig. 7, with s being a path parameter: (a), (c) trends with no change in solution branch; (b), (d) trends with a change in solution branch.

without encountering any physical problem. However, a wrist singularity is encountered in point C, when $q_5 = 0$, as illustrated in Fig. 7(b). When assigning this task to the robot, solving the IK problem can be approached through three distinct strategies:

- *Using the built-in solver of the Universal Robot controller:* In this case, the motion is interrupted even before reaching point C;
- *Applying the method proposed in* [1]*:* This approach allows the trajectory to pass through the singularity, reaching point B as shown in Fig. 7(c), without switching the IK-solution branch, namely keeping $j = \text{sgn}(P_2) = 1$ (Fig. 8(a)). In this case, discontinuities appear in joint variables q_2, q_3, q_4 and q_6 (Fig. 8(c)), resulting in significant tracking errors;
- *Employing the method proposed in this work:* Our strategy enables switching between solution branches upon encountering a singular configuration, achieving point B as shown in Fig. 7(d). In this case, parameters P_1, P_2, and P_3 evolve smoothly (Fig. 8(b)), and joint trajectories remain continuous (Fig. 8(d)).

B. Algorithm

From the previous example, it is evident that when the robot crosses a singular configuration, some of the indices (i, j, k) defining the IK-solution branch may need to be updated to ensure a feasible and continuous evolution of joint variables.

The factors P_1, P_2 and P_3 are linear combinations of trigonometric functions of the joint variables. Accordingly, if the latter must be continuous with all their derivatives to guarantee a smooth motion, the same property should be shared by P_1, P_2 and P_3. When one of these factors vanishes with a non-zero derivative, a sign change is mathematically required to preserve differentiability of joint trajectories, implying a transition to a different IK branch. When a factor approaches zero with a vanishing first derivative, higher-order derivatives can be used to classify the vanishing point[1]. However, in discrete-time implementations (such as industrial digital controllers, where only past and current trajectory values are available) the behavior of a vanishing factor can only be assessed from one side of the zero. This makes it unreliable to determine whether the point corresponds to a local extremum or an inflection through higher-order derivatives. For this reason, a more practical and robust approach is preferred here. When P_n $(n = 1, 2, 3)$ approaches zero (i.e. $|P_n| \leq \varepsilon_n$, with ε_n being a small positive tolerance), the joint configurations corresponding to both possible sign choices of the associated index are evaluated. The configuration that minimizes the joint displacement from the previously selected solution is retained and the corresponding index triplet (i, j, k) is updated.

The choice of ε_n directly affects algorithm performance and robustness: large values may trigger unnecessary branch switches and increase computation time, whereas small values may miss singularities and prevent required switches. To determine a suitable ε_n, let s denote the trajectory discretization parameter (representing the sequence of control steps). The first-order Taylor expansion of P_n over a step δs is:

$$P_n(s + \delta s) = P_n(s) + \frac{\partial P_n}{\partial s}\delta s = P_n(s) + \sum_{h=1}^{6} \frac{\partial P_n}{\partial q_h}\frac{\partial q_h}{\partial s}\delta s. \tag{15}$$

If a zero-crossing occurs at s $(P_n(s) = 0)$, (15) becomes:

$$P_n(s + \delta s) = \sum_{h=1}^{6} \frac{\partial P_n}{\partial q_h}\frac{\partial q_h}{\partial s}\delta s, \tag{16}$$

where $\frac{\partial q_h}{\partial s}\delta s$ represents the joint displacement over one control cycle. To ensure a conservative threshold, we consider the worst-case variation of (16), bounded by the maximum physical capability of the robot per sampling interval. Using the maximum joint velocity $\dot{q}_{\max}$ (in rad/s) and the sampling frequency f_c (in Hz) yields:

$$|P_n(s + \delta s)| \leq \sum_{h=1}^{6} \left|\frac{\partial P_n}{\partial q_h}\right|_{max} \frac{\dot{q}_{max}}{f_c} = \varepsilon_n. \tag{17}$$

Consequently, $|P_n(s + \delta s)| \leq \varepsilon_n$ is a necessary condition for $P_n(s)$ to be zero. Thus, ε_n represents a conservative lower bound to reliably trigger the branch switch procedure.

An additional consideration concerns the wrist singularity $(P_2 = s_5 = 0)$. While the elbow singularity $(P_1 = 0)$ represents a singularity at the external boundary of the reachable

[1]If the first non-zero derivative of P_n at the point where it approaches zero is of *even* order, the point is a local extremum and a sign change is generally not required. If the first non-zero derivative is of *odd* order, the point is an inflection, and a sign change may be required to preserve differentiability.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

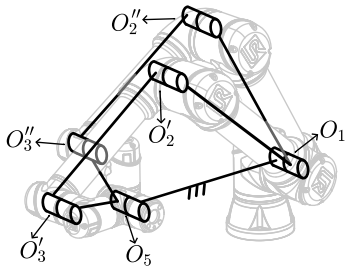

Fig. 9. 4-bar mechanism.

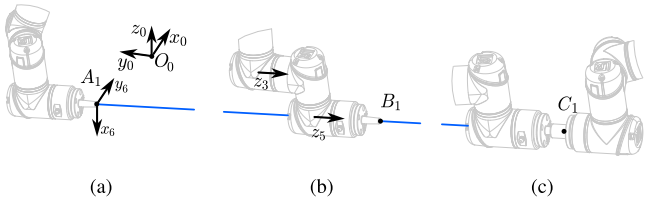

Fig. 10. Various configurations for a Peg-in-Hole task: (a) start pose with indices $(1, -1, 1)$; (b) singularity; (c) final pose with indices (1,1,1).

workspace, both the shoulder ($P_3 = 0$) and the wrist singularities occur within the workspace. However, the wrist singularity allows the robot to gain a finite internal mobility besides decreasing the EE degree of freedom, as the axes of joints 2, 3, 4, and 6 become parallel, thereby forming a 4-bar linkage within the robot's architecture (Fig. 9). This configuration enables finite motions of joints q_2, q_3, q_4 and q_6 while the EE pose remains fixed. Consequently, the system becomes kinematically indeterminate, yielding infinitely many solutions to the IK problem based on the formulas proposed in Section II. Various strategies can be adopted to handle this scenario. For instance, if at a critical configuration $|P_2| < \varepsilon'$—where ε' is a small positive tolerance accounting for the numerical precision of the computing environment (e.g., 10^{-9})—then a single joint variable of the 4-bar mechanism, such as q_6, can be computed by interpolating its values from the last $M \geq 2$ sampled instants. The remaining joint variables of the linkage (q_2, q_3, q_4) are then determined through the position analysis of the 4-bar mechanism (Solve4Bar($\texttt{robot}, q_6$)[2]), thus ensuring kinematic consistency. Among the two possible solutions to the 4-bar DK problem, the one that minimizes the difference to the previous joint configuration is chosen. The use of past trajectory data for extrapolating a single driving joint of the 4-bar mechanism, coupled with its analytical closure, ensures the continuity of the joint evolution while guaranteeing kinematic feasibility, which is the intended goal of the proposed strategy. The overall procedure is summarized in the pseudocode reported in Algorithm 1. It is important to note that by leveraging the distinct correspondence between the solution indices (i, j, k) and the factors P_n $(n = 1, 2, 3)$, the algorithm is inherently capable of finding a continuous solution even when encountering multiple singular configurations simultaneously.

[2]This function represents any method for solving the direct kinematics (DK) of the 4-bar linkage.

Algorithm 1: Path Planning With Transitions Across Singularities.

Input: Robot model `robot`; pose path $\{\mathbf{T}_{06}(s)\}_{s\in[s_{\text{start}}, s_{\text{end}}]}$; initial indices (i, j, k); $\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon'$
Init: Initialize $\mathbf{q}_{\text{prev}} \leftarrow$ null
for each discretized $s \in [s_{\text{start}}, s_{\text{end}}]$ **do**
  Compute $\mathbf{q}_{\text{init}} \leftarrow \text{IK}(\texttt{robot},\ \mathbf{T}_{06}(s),\ i,\ j,\ k)$
  Compute Jacobian factors $[P_1,\ P_2,\ P_3]$
  Normalize $\mathbf{q}_{\text{init}} \leftarrow \text{WrapToPi}(\mathbf{q}_{\text{init}})$[3]
  if $\min(|P_1|,\ |P_2|,\ |P_3|) \geq \varepsilon$ **then**
    $\mathbf{q}(s) \leftarrow \mathbf{q}_{\text{init}}$
  else
    if $|P_2| < \varepsilon'$ **then**
      Estimate $\mathbf{q}_{\text{est}}[6]$ by interpolation
      $\mathbf{q}_{\text{est}}[2, 3, 4] \leftarrow \text{Solve4Bar}(\texttt{robot},\ \mathbf{q}_{\text{est}}[6])$
      $\mathbf{q}_{\text{s}} \leftarrow [\mathbf{q}_{\text{init},1}; \mathbf{q}_{\text{est},2}; \mathbf{q}_{\text{est},3}; \mathbf{q}_{\text{est},4}; \mathbf{q}_{\text{init},5}; \mathbf{q}_{\text{est},6}]$
    else
      if $|P_3| < \varepsilon_3$ **then** $I \leftarrow \{\pm 1\}$ **else** $I \leftarrow \{i\}$
      if $|P_2| < \varepsilon_2$ **then** $J \leftarrow \{\pm 1\}$ **else** $J \leftarrow \{j\}$
      if $|P_1| < \varepsilon_1$ **then** $K \leftarrow \{\pm 1\}$ **else** $K \leftarrow \{k\}$
      Evaluate all valid IK solutions $\mathbf{q}_{\text{cand}}$ for
      $(i', j', k') \in I \times J \times K$
      Normalize $\mathbf{q}_{\text{cand}} \leftarrow \text{WrapToPi}(\mathbf{q}_{\text{cand}})$
      Get $\mathbf{q}_{\text{best}}$ minimizing $\|(\mathbf{q}_{\text{cand}} - \mathbf{q}_{\text{prev}})\|$
      Update: $(i, j, k) \leftarrow (i', j', k')$ by $\mathbf{q}_{\text{best}}$
      $\mathbf{q}(s) \leftarrow \mathbf{q}_{\text{best}}$
    end if
  end if
  $\mathbf{q}_{\text{prev}} \leftarrow \mathbf{q}(s)$
end for
Output: Joint path $\{\mathbf{q}(s)\}_{s\in[s_{\text{start}}, s_{\text{end}}]}$

IV. Experimental Validation

To evaluate the effectiveness of the proposed IK algorithm in handling singularity crossing, two[3] representative tasks are tested on a UR5e robot[4]. The DH parameters in Table I, take the following values: $a_2 = -0.4250\,\text{m}, a_3 = -0.3922\,\text{m}, d_1 = 0.1625\,\text{m}, d_4 = 0.1333\,\text{m}, d_5 = 0.0997\,\text{m}$ and $d_6 = 0.0996\,\text{m}$. Parameters f_c and $\dot{q}_{max}$, in (17) are 500 Hz and $\pi\,\text{rad/s}$, respectively, so that $\varepsilon_1 = \varepsilon_2 = 6.283 \cdot 10^{-3}$ and $\varepsilon_3 = 2.230 \cdot 10^{-2}$. Finally, $\epsilon' = 10^{-9}$. Both experiments involve trajectories that intentionally pass through singular configurations, providing a practical demonstration of the algorithm's behavior in such scenarios. For each trajectory, the IK is solved using three distinct methods: the method integrated in the UR internal controller, the approach proposed in [1], and the algorithm developed in this paper. In the first case, Cartesian poses—i.e., TCP position and orientation expressed through roll, pitch, and yaw angles, following the extrinsic XYZ rotation sequence used by the UR controller—are sent to the robot via the RTDE (Real-Time Data Exchange) interface, and the IK is computed by the internal UR solver. In contrast, the latter two methods requires transmitting joint displacements through the RTDE interface, with the robot executing the motion via the `Servoj` command in a URScript. The parameters `lookahead_time` $= 0.2\,\text{s}$ and `gain` $= 2000$ are chosen according to the users' experience.

[3]`WrapToPi` maps each joint angle to the interval $(-\pi, \pi]$ to avoid discontinuities due to 2π wrapping.

[4][Online]. Available: https://www.universal-robots.com/products/ur5e/

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

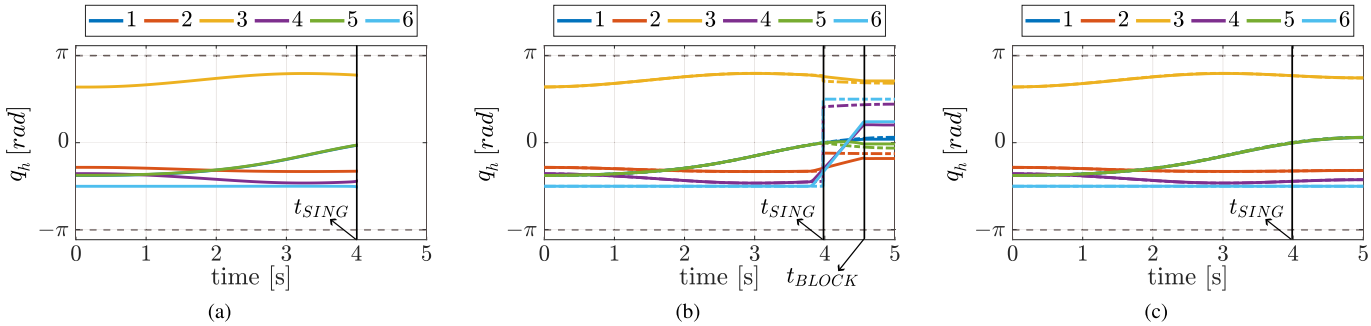

Fig. 11. Joint trajectories over time for different strategies during the Peg-in-Hole task. Continuous lines correspond to measured joint angles and dashed lines correspond to model-predicted, and therefore commanded, joint angles. (a) Trajectory provided by the robot; (b) Trajectory with no change in solution branch; (c) Trajectory with a change in solution branch.

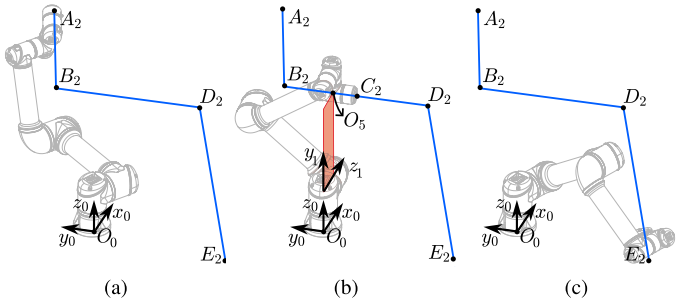

Fig. 12. Various configurations for a Pick&Place task: (a) start pose with indices $(-1, 1, 1)$; (b) singularity; (c) final pose with indices (1,1,1).

Dataset [19] contains all data needed to reproduce the simulation and experimental results, as well as additional trajectories illustrating the algorithm's effectiveness in handling simultaneous singularity crossings, which are omitted here for brevity. A video illustrating the experiments described in the article is provided as supplementary downloadable material.

A. Peg-in-Hole Task Across a Wrist Singularity

The first scenario consists of a peg-in-hole task simulating an object transfer between two adjacent robots. A pin mounted on the EE of a UR5e robot is to be inserted into a cavity realized on the EE of a second robot remaining stationary during the task. The path—with key instances shown in Fig. 10—is a straight-line segment executed over 5 s, with the EE orientation kept constant. The motion follows a trajectory governed by a third-degree polynomial motion law with zero velocity boundary conditions. Points A_1 and C_1, whose position vectors from O_0 are $(A_1 - O_0) = [-0.290 \;\; 0.204 \;\; 0.040]^T$ m and $(C_1 - O_0) = [-0.290 \;\; -0.344 \;\; 0.040]^T$ m, denote the initial and final positions of the pin tip (offset by 0.051 m from O_6 along the z_6 axis). The orientation of frame 6 relative to frame 0 is given by the rotation matrix $\mathbf{R}_{06} = \mathbf{R}_x(\pi/2)\mathbf{R}_z(-\pi/2)$. Fig. 11 shows the time evolution of the joint trajectories obtained using the three different methods introduced earlier. With the UR internal controller, upon encountering a wrist singularity—i.e., when the rotation axes of joints 4 and 6 (namely, z_3 and z_5) become parallel, as illustrated in Fig. 10(b)—the robot stops when the pin tip reaches point B_1 at t_{SING} (Fig. 11(a)) and fails to complete the assigned task. If no change in solution branch is allowed, the model predicts finite jumps in some joint variables over an infinitesimal time (Fig. 11(b)): due to the robot's acceleration limits and the way the UR controller's `ServoJ` function processes joint commands, the EE deviates from the planned trajectory, resulting in maximum deviations of $[e_x \;\; e_y \;\; e_z]^T = [0.0941 \;\; 0.0301 \;\; 0.0218]^T$ m on the position components and $[e_\phi \;\; e_\theta \;\; e_\psi]^T = [0.0355 \;\; 0.1072 \;\; 4.6441]^T$ rad on the XYZ Euler angles. At t_{BLOCK}, the robot is intentionally stopped to avoid collisions with the experimental setup, which explains why the measured joint variables remain constant after the stop instant. In contrast, when the IK is solved using Algorithm 1, the robot successfully completes the task (Fig. 10(c)), and the joint variables show a continuous and differentiable behavior without abrupt jumps (Fig. 11(c)).

B. Pick&place Task Across a Shoulder Singularity

In this experiment, the UR5e robot performs a generic pick-and-place operation, with selected instances shown in Fig. 12, moving an object from a shelf to a table within its workspace. The planned path consists of three straight-line Cartesian segments, each executed over 5 s, connecting points A_2, B_2, D_2, and E_2, whose position vectors from O_0 are $(A_2 - O_0) = [0.500 \;\; 0.300 \;\; 0.700]^T$ m, $(B_2 - O_0) = [0.133 \;\; 0.200 \;\; 0.563]^T$ m, $(D_2 - O_0) = [0.133 \;\; -0.400 \;\; 0.563]^T$ m and $(E_2 - O_0) = [-0.200 \;\; -0.600 \;\; 0.020]^T$ m. Each segment is executed following a third-degree polynomial motion law, ensuring zero velocity at both its initial and final points. Along each segment, the EE z-axis, i.e. z_6, remains aligned with the velocity of O_6. This orientation is achieved by defining the approach vector, $\mathbf{r}_{approach}$, as the unit vector from the current point to the segment end point.[5] The EE orientation is then parameterized using a ZYZ Euler-angle convention. Specifically, for the first and third segments, the first two Euler angles are calculated to align z_6 with $\mathbf{r}_{approach}$, while the third Euler angle (rotation about z_6) is set to zero. For the central segment, the Euler angles were specifically set to $[\pi/2 \;\; -\pi/2 \;\; -\pi/2]^T$ radians to accommodate task-specific orientation requirements. Between consecutive segments, reorientations are performed over 3 s by interpolating the Euler angles from the initial to the final pose according to a cubic polynomial. Fig. 13 shows the joint trajectories over time obtained using

[5]For the first segment, the unit vector is reversed, pointing from B_2 to A_2.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

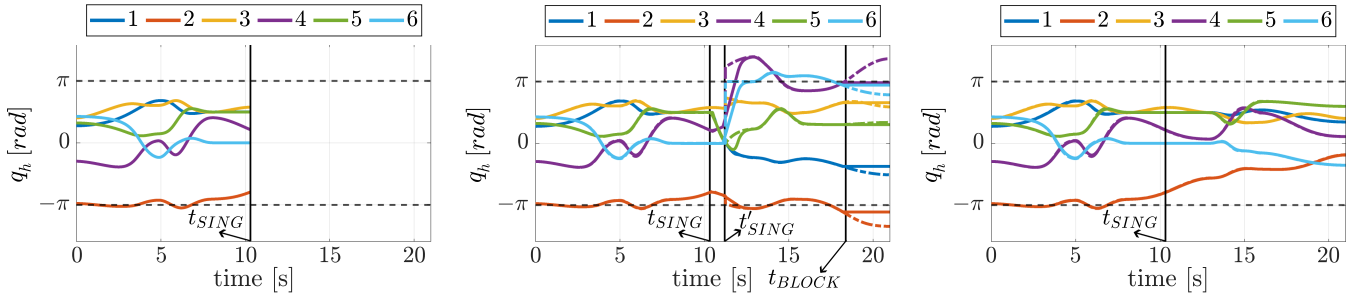

Fig. 13. Joint trajectories over time for different strategies during the Pick&Place task. Continuous lines correspond to measured joint angles and dashed lines correspond to model-predicted, and therefore commanded, joint angles. (a) Trajectory provided by the robot; (b) Trajectory with no change in solution branch; (c) Trajectory with a change in solution branch.

TABLE II
COMPARISON BETWEEN COMPLETE AND LOCAL IK SOLVERS

Solver type	Complete					Local	
	Ours	[1]	[15]	[20]	[21]	[22]	[23]
All solutions	✓	✓	✓	✓	✓	×	×
Solution at singularity	✓	✓	×	✓	✓	✓*	✓
Accuracy at singularity	✓	✓	×	✓	✓	✓*	×
Posture prediction	✓	✓	×	×	×	×	×
Joint continuity criterion	✓	×	×	×	×	✓*	×

*not always applicable

TABLE III
COMPARISON OF COMPUTATION TIME FOR DIFFERENT IK METHODS

Experiment (No. of Steps)	Peg-in-Hole (2501)		Pick&Place (10505)	
Method	Alg. 1	Alg. 2	Alg. 1	Alg. 2
$t_{\min}$ $[s]$	0.0933	0.3509	0.4392	1.6273
$t_{\max}$ $[s]$	0.3285	0.7247	0.9048	2.5493
$\bar{t}$ $[\mu s]$	43.74	147.0	45.06	158.8

the three previously discussed methods. With the UR internal method, upon encountering a shoulder singularity—i.e., when O_5 lies within the plane defined by the axes of joints 1 and 2 (namely, y_1 and z_1), highlighted in red in Fig. 12(b)—the robot stops at t_{SING} (Fig. 13(a)) and fails to complete the assigned task. The method that allows no change in solution branch shows large joint angle variations (Fig. 13(b)), deviating the EE from the planned trajectory (resulting in maximum deviations of $[e_x \;\; e_y \;\; e_z]^T = [0.0674 \;\; 0.0298 \;\; 0.0385]^T$ m on the position components and $[e_\phi \;\; e_\theta \;\; e_\psi]^T = [0.6926 \;\; 0.3072 \;\; 4.7025]^T$ rad on ZYZ Euler angles) and encounters an additional wrist singularity at t'_{SING}. Conversely, the method proposed in this paper successfully completes the assigned task (Fig. 12(c)), with smooth and continuous evolution of joint variables (Fig. 13(c)).

C. Comparison With Other IK Methods

We compared our method with complete solvers (which provide all solutions to a given IK problem) and local solvers (which provide a single solution depending on an initial guess) on the experimental trajectories. Key characteristics are summarized in Table II. Refs. [1] and [15] were re-implemented by the authors in MATLAB, IK-Fast [20] was evaluated using its open-source C++ implementation,[6] IK-Geo [21] using its open-source repository,[7] TRAC-IK [22] via its Python implementation and MoveIt2 plugin,[8] and the Levenberg-Marquardt method [23] using the MATLAB built-in `inverseKinematics` function. The comparison highlights that complete solvers provide all possible solutions, but leave the selection of the appropriate one to the user, whereas our algorithm enables deterministic posture selection and continuous path tracking through singular configurations, with the added advantage of improved computational performance (see Section IV-D). Local solvers are not always accurate or reliable in computing a solution close to a singularity. In particular, TRAC-IK may provide inconsistent results, as it may succeed or fail under identical computation conditions (this non-deterministic behavior is denoted by ✓* in Table II). To quantify this behavior, each experiment was repeated 1,000 times: TRACK-IK failed to find a continuous solution in 38 Peg-in-Hole trials and in 102 Pick&Place trials.

From an implementation standpoint, among the methods in Table II, the one most similar to ours is [15]. However, Ref. [15] does not address prescribed path tracking, as it solves the IK only at the initial and the final pose, computing all possible solutions at the latter and selecting the one minimizing total joint displacement; the motion between the start and goal poses is not considered. Moreover, solutions at singular configurations are discarded, and no posture prediction is provided.

D. Computation Time

To complement the qualitative analysis of the previous subsections, Table III compares the computational efficiency of the proposed IK strategy with an IK solver (Algorithm 2) derived from [15] by improving its capabilities. Specifically, all possible solutions are computed at each step of the prescribed trajectory, and the joint-displacement minimization approach of [15] is applied between consecutive steps rather than only at the initial and final poses; moreover, solutions corresponding to singular configurations are not discarded but treated as acceptable. The main difference between Algorithms 1 and 2 is that Algorithm 1 does not evaluate all solutions along the entire path, but considers a reduced subset only when approaching a singularity. In such

[6][Online]. Available: https://github.com/cambel/ur_ikfast
[7][Online]. Available: https://github.com/rpiRobotics/ik-geo
[8][Online]. Available: https://bitbucket.org/traclabs/trac_ik/

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

cases, the alternative solutions are selected based on which of the factors P_n falls below ε_n. All computations were performed on a computer equipped with an Intel Core i3-5005 U CPU @ 2.00 GHz, 8 GB of RAM, and running MATLAB R2023b on Windows 10. Both the Peg-in-Hole and Pick&Place trajectories were solved 100 times using both Algorithms. For each case, the minimum ($t_{\min}$) and maximum ($t_{\max}$) computation times for the entire trajectory, as well as the average computation time ($\bar{t}$) for the solution of the IK problem on a single pose, are reported. The results reported in Table III show that the proposed method achieves a significant reduction in computation time while maintaining solution continuity. Even though the set points were precomputed in MATLAB, it is expected that the same computational advantages would be retained on lower-level languages like C++ for ROS implementations. This demonstrates the attractiveness of our technique for both online and offline planning applications.

V. Conclusion

This study presented an IK resolution technique for non-redundant serial robots with UR-like architecture, which not only determines the robot posture but also enables a controlled transition through singularities, ensuring continuity and differentiability of the joint variables. A comparative study was conducted between the built-in UR IK solver, a method from literature and the algorithm developed in this work, showing the benefits introduced by the latter. The proposed methodology is expected to be applicable to any 6R non-cuspidal serial robot, extending beyond the UR-type architecture; nonetheless, additional validation on different robot models is necessary to fully confirm its generality.

It should be emphasized that the proposed method guarantees smooth and continuous transitions through singularities only when the requested trajectory is kinematically admissible. Should motion along a lost degree of freedom be required, the approach would need to be extended to allow controlled deviations or alternative strategies to maintain feasibility. Coupling the method with infeasibility detection and online re-planning would make it a viable solution for online planning. Addressing such scenarios is beyond the scope of the present study and represents an interesting direction for future research.

References

[1] L.-T. Schreiber and C. Gosselin, "Determination of the inverse kinematics branches of solution based on joint coordinates for universal robots-like serial robot architecture," *J. Mechanisms Robot.*, vol. 14, no. 3, pp. 1–7, 2021, doi: 10.1115/1.4052805.

[2] J. Villalobos, I. Y. Sanchez, and F. Martell, "Alternative inverse kinematic solution of the UR5 robotic arm," in *Advances in Automation and Robotics Research*. Berlin, Germany: Springer, 2021, pp. 200–207, doi: 10.1007/978-3-030-90033-5_22.

[3] X. Zhang, G. Li, M. Xu, D. Jiang, and J. Yun, "A novel method for selecting inverse kinematic solutions based on configuration space partition for 6R noncuspidal manipulators," *J. Intell. Robotic Syst.*, vol. 110, no. 1, pp. 1–20, 2023, doi: 10.1007/s10846-023-02029-4.

[4] C. W. Wampler, "Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods," *IEEE Trans. Syst., Man, Cybern.*, vol. TSMC-16, no. 1, pp. 93–101, Jan. 1986, doi: 10.1109/TSMC.1986.289285.

[5] K. Tchoń, W. Respondek, and J. Ratajczak, "Normal forms and configuration singularities of a space manipulator," *J. Intell. Robotic Syst.*, vol. 93, pp. 621–634, Jun. 2018, doi: 10.1007/s10846-018-0883-8.

[6] K. Tchoń and J. Ratajczak, "Singularities of holonomic and non-holonomic robotic systems: A normal form approach," *J. Franklin Inst.*, vol. 358, no. 15, pp. 7698–7713, 2021, doi: 10.1016/j.jfranklin.2021.07.028.

[7] J. Hernandez-Barragan, C. Lopez-Franco, N. Arana-Daniel, A. Y. Alanis, and A. Lopez-Franco, "A modified firefly algorithm for the inverse kinematics solutions of robotic manipulators," *Integr. Comput.-Aided Eng.*, vol. 28, no. 3, pp. 257–275, 2021, doi: 10.3233/ica-210660.

[8] A. T. Hasan and H. Al-Assadi, "Performance prediction network for serial manipulators inverse kinematics solution passing through singular configurations," *Int. J. Adv. Robotic Syst.*, vol. 7, no. 4, pp. 10–23, 2010, doi: 10.5772/10492.

[9] K. Abdel-Malek and H.-J. Yeh, "Crossable surfaces of robotic manipulators with joint limits," *J. Mech. Des.*, vol. 122, no. 1, pp. 52–60, 2000, doi: 10.1115/1.533545.

[10] K.-S. Chang and O. Khatib, "Manipulator control at kinematic singularities: A dynamically consistent strategy," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Pittsburgh, PA, USA, 1995, pp. 84–88, doi: 10.1109/IROS.1995.525866.

[11] D. Oetomo and M. H. J. Ang, "Singularity robust algorithm in serial manipulators," *Robot. Comput.- Integr. Manuf.*, vol. 25, no. 1, pp. 122–134, 2009, doi: 10.1016/j.rcim.2007.09.007.

[12] J. Lloyd, "Removing the singularities of serial manipulators by transforming the workspace," in *Proc. IEEE Int. Conf. Robot. Autom.*, Leuven, Belgium, 1998, pp. 2935–2940, doi: 10.1109/robot.1998.680733.

[13] D. Manocha and J. F. Canny, "Efficient inverse kinematics for general 6R manipulators," *IEEE Trans. Robot. Automat.*, vol. 10, no. 5, pp. 648–657, Oct. 1994, doi: 10.1109/70.326569.

[14] M. L. Husty, M. Pfurner, and H.-P. Schröcker, "A new and efficient algorithm for the inverse kinematics of a general serial 6R manipulator," *Mechanism Mach. Theory*, vol. 42, no. 1, pp. 66–81, 2007, doi: 10.1016/j.mechmachtheory.2006.02.001.

[15] J. Villalobos, I. Y. Sanchez, and F. Martell, "Singularity analysis and complete methods to compute the inverse kinematics for a 6-DOF UR/TM-type robot," *Robotics*, vol. 11, no. 6, pp. 1–14, 2022, doi: 10.3390/robotics11060137.

[16] H. Lipkin, "A note on denavit-hartenberg notation in robotics," in *Proc. ASME Int. Desi. Eng. Tech. Conf.*, Long Beach, CA, USA, 2005, pp. 921–926, doi: 10.1115/detc2005-85460.

[17] D. H. Salunkhe, T. Marauli, A. Müller, D. Chablat, and P. Wenger, "Kinematic issues in 6R cuspidal robots, guidelines for path planning and deciding cuspidality," *Int. J. Robot. Res.*, vol. 44, no. 6, pp. 1035–1054, 2025, doi: 10.1177/02783649241293481.

[18] M. H. FarzanehKaloorazi and I. A. Bonev, "Singularities of the typical collaborative robot arm," in *Proc. ASME Int. Des. Eng. Tech. Conf.*, Quebec City, QC, Canada, 2018, pp. 1–7, doi: 10.1115/detc2018-86305.

[19] I. Boschi, A. De Toni, R. Di Leva, E. Ida', and M. Carricato, "Handling transitions across singularities for UR-like serial robots," [Data set] Zenodo, 2025, doi: 10.5281/zenodo.17909620.

[20] R. Diankov, "Automated construction of robotic manipulation programs," *Ph.D. thesis*, Robot. Inst., Carnegie Mellon Univ., Pittsburgh, PA, USA, Aug. 2010.

[21] A. J. Elias and J. T. Wen, "IK-Geo: Unified robot inverse kinematics using subproblem decomposition," *Mechanism Mach. Theory*, vol. 209, pp. 105971–105996, 2025, doi: 10.1016/j.mechmachtheory.2025.105971.

[22] P. Beeson and B. Ames, "TRAC-IK: An open-source library for improved solving of generic inverse kinematics," in *Proc. IEEE-RAS 15th Int. Conf. Humanoid Robots*, Seoul, South Korea, 2015, pp. 928–935, doi: 10.1109/HUMANOIDS.2015.7363472.

[23] T. Sugihara, "Solvability-unconcerned inverse kinematics by the Levenberg–Marquardt method," *IEEE Trans. Robot.*, vol. 27, no. 5, pp. 984–991, Oct. 2011, doi: 10.1109/TRO.2011.2148230.