International Core Journal of Engineering Volume 12 Issue 2, 2026
ISSN: 2414-1895 DOI: 10.6919/ICJE.202602_12(2).0005

# Research on Trajectory Planning of a 6R Robotic Manipulator based on Jacobian Iteration

Xiao Liu, Aiwen Yuan, Shiyu Chao, Chaoming Huang, and Hangze Yan

School of Aeronautic Manufacturing Engineering, Nanchang Hangkong University, Nanchang, 330063, China

## Abstract

**To meet the requirements of end-effector positioning accuracy and trajectory smoothness for laser positioning tasks of a 6R robotic manipulator, this paper investigates trajectory planning for a 6R robot based on a Jacobian iterative inverse kinematics method. First, a complete kinematic model of the 6R manipulator is established, and the mapping relationship between the end-effector pose and the joint variables is derived. On this basis, to address the limited applicability of analytical methods for manipulators with complex configurations, a Jacobian matrix–based iterative approach is introduced to solve the inverse kinematics problem, and an error-feedback mechanism is employed to achieve the gradual convergence of the end-effector pose to the desired target. Furthermore, considering the application scenario of laser positioning, straight-line trajectory planning of the end effector is performed in Cartesian space. Continuous linear motion trajectories are generated through linear interpolation between trajectory points. Simulation and experimental results demonstrate that the proposed Jacobian iterative method ensures convergence while effectively reducing the pose error between the robot end effector and the laser spot. The method also exhibits good stability and positioning accuracy within a certain range of motion.**

## Keywords

**Jacobian Iteration; Trajectory Planning; Laser Positioning; Kinematics.**

## 1. Introduction

The solutions to forward and inverse kinematics constitute core problems that must be addressed in the practical control of robotic manipulators. Compared with forward kinematics, inverse kinematics is generally more challenging, primarily because manipulator structures vary significantly and there is no unified or universally applicable solution procedure or standard methodology. At present, commonly used approaches for solving inverse kinematics include analytical methods, numerical iterative methods, and geometric methods.[1,5]

The main advantage of analytical methods lies in their ability to obtain all closed-form solutions of the inverse kinematics equations and to rigorously determine the existence of solutions. However, these methods typically involve complex derivations, substantial computational effort, and impose strict requirements on the manipulator's structural configuration. Numerical iterative methods are relatively easy to implement and have broad applicability, making them suitable for solving the inverse kinematics of most manipulators. Nevertheless, such methods are highly dependent on the initial guess and, for a given initial configuration, usually converge to only one solution. Moreover, when no solution exists for the target pose, the iterative process tends to fail to converge. Geometric methods simplify the inverse kinematics problem by introducing spatial geometric relationships and

ISSN: 2414-1895 DOI: 10.6919/ICJE.202602_12(2).0005

are applicable to manipulators with certain specific configurations; in practice, they are often used in combination with analytical methods.[1,7]

Previous studies have shown that Pieper proved the existence of closed-form solutions for the inverse kinematics of six-degree-of-freedom manipulators when three adjacent joint axes intersect at a single point. Duffy further demonstrated that closed-form solutions can also be obtained when three adjacent joint axes are mutually parallel. Most industrial robots adopt structures with three intersecting joint axes; therefore, their complete inverse kinematics solutions can generally be obtained using analytical methods, after which additional strategies are employed to select an optimal feasible solution. However, analytical approaches are strongly dependent on the manipulator's structural form. When the structure or parameters of the manipulator change, the analytical model usually needs to be re-derived, resulting in limited flexibility.

To address these issues, this paper proposes a Jacobian-based iterative numerical method for solving the inverse kinematics of a 6R manipulator. Numerical approaches typically transform the inverse kinematics problem into the solution of a set of nonlinear equations or an optimization problem and iteratively approximate the target solution. Such methods feature strong generality and minimal structural constraints, making them applicable to manipulators of arbitrary degrees of freedom and configurations. They are also relatively straightforward to implement in engineering practice and are well suited for programmatic and modular design. The manipulator studied in this work is rigidly mounted on a mobile platform and equipped with a laser emitter to determine the target position of the end-effector. By iteratively updating the manipulator configuration from an initial pose until it reaches the target laser point, the inverse kinematics is solved numerically. Finally, trajectory planning experiments are conducted to validate the effectiveness of the proposed method.[2]

## 2. Control System Architecture for a 6R Robot

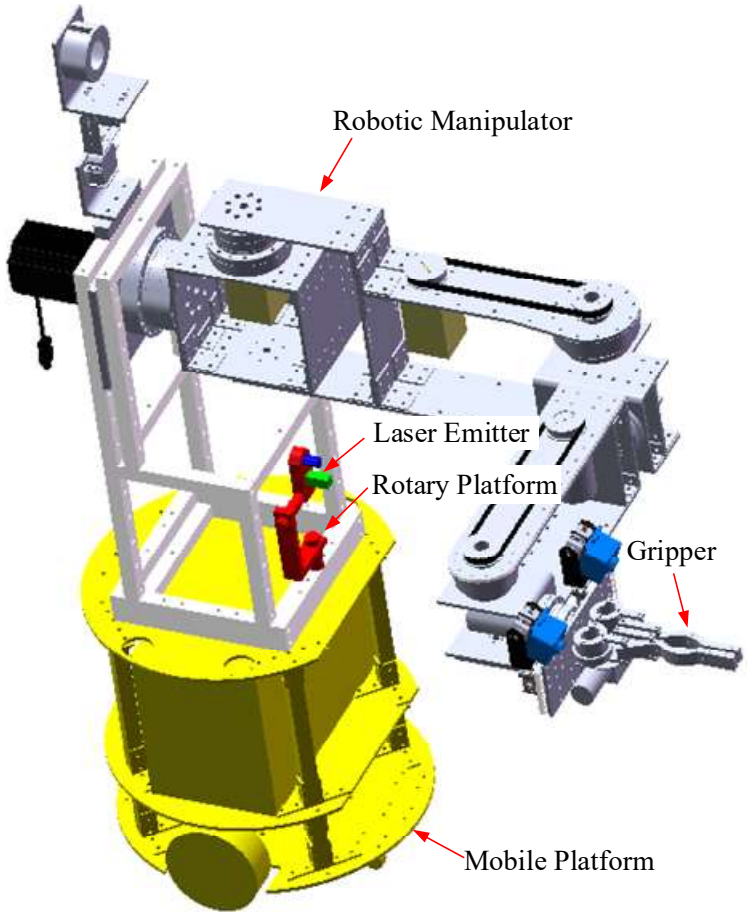


**Fig. 1** Hardware Structure of a 6R Robot

ISSN: 2414-1895 DOI: 10.6919/ICJE.202602_12(2).0005

The hardware architecture of the 6R robot is illustrated in Fig. 1. The overall system consists of a mobile platform, a main controller, and a robotic manipulator. A pan–tilt unit mounted adjacent to the manipulator is equipped with a laser emitter. The laser emitter operates at a wavelength of 650 nm, corresponding to visible red light, with an output power of 5 mW.

A gripper is installed at the end of the manipulator. When the pan–tilt unit is rotated to a desired orientation, the laser emitter projects a laser beam along the current direction toward the target point to acquire its positional information. Subsequently, the manipulator gripper performs a linear motion from its current configuration toward the target point position.

Based on the structural characteristics of the 6R robot, as shown in Fig. 2, the overall architecture of the manipulator control system can be divided into four functional layers: the laser emitter layer, the main control module, the drive module, and the application layer. These modules form a complete closed-loop control system through information exchange and the transmission of control commands.

The laser emitter serves as the positioning unit and is responsible for acquiring relevant positional information. The main control module constitutes the core of the entire control system, undertaking tasks such as processing positioning data, solving the manipulator kinematic models, performing trajectory planning, and generating control commands. The drive module mainly consists of servo drives and actuators; it receives control signals output by the main control module and converts them into the corresponding motions of each joint motor.

Through this layered architectural design, the responsibilities of each functional module are clearly defined. While operating independently, the modules also cooperate closely, which not only enhances the system's scalability and maintainability but also provides reliable system-level support for laser-based precise control and trajectory planning of the robotic manipulator.

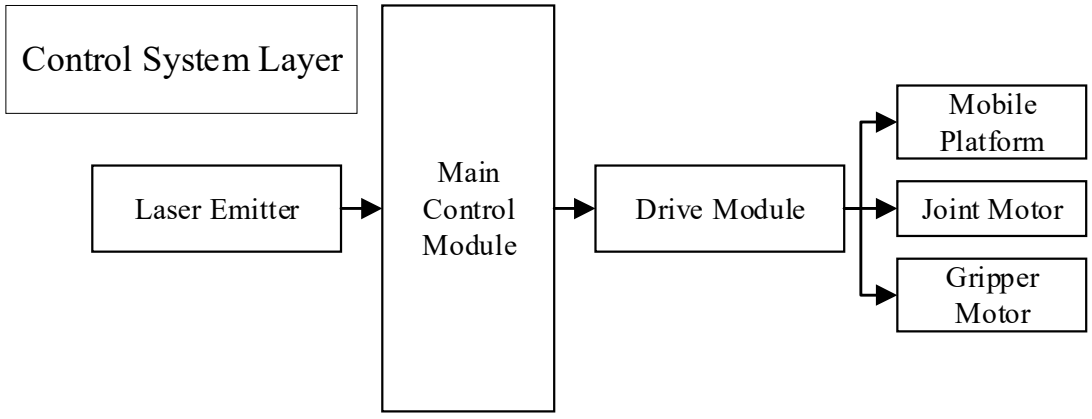


**Fig. 2** Control System Diagram of a 6R Robot

## 3. Establishment of Forward and Inverse Kinematic Models for a 6R Robot

### 3.1 Forward Kinematic Model of a 6R Robot

As shown in Fig. 3, the joint coordinate frames of the manipulator are established using the modified Denavit–Hartenberg (D–H) parameter method. During the construction of the manipulator coordinate system, the base coordinate frame {0} shares a common origin with the first and second joint coordinate frames {1} and {2}. The end-effector coordinate frame {6} is located at the distal end of the manipulator, while the fourth and fifth joint coordinate frames {4} and {5} are arranged to have coincident origins. Table 1 lists in detail the modified D–H parameter values of the manipulator.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202602_12(2).0005

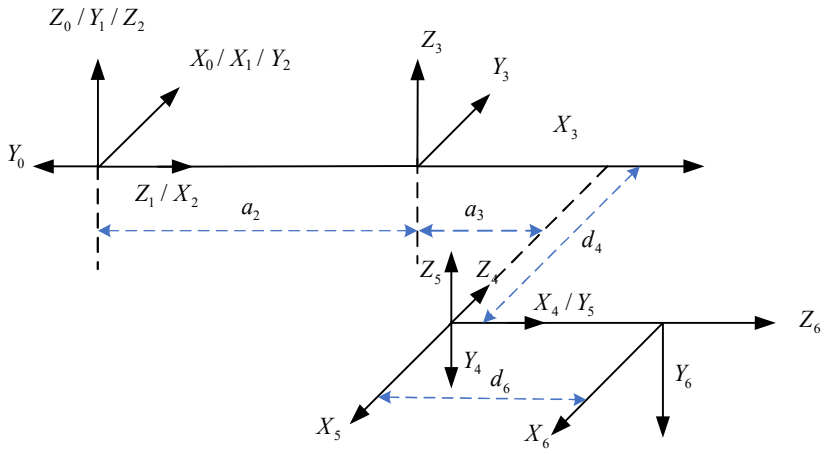


**Fig. 3** Link Coordinate Frames of a 6R Manipulator

**Table 1.** Modified D–H Parameter Table of the Manipulator

| n | $a_{i-1}(mm)$ | $\alpha_{i-1}(°)$ | $d_i(mm)$ | $\theta_i(°)$ | Joint Range |
|---|---|---|---|---|---|
| 1 | 90 | 0 | 0 | $\theta_1$ | [-180，180] |
| 2 | -90 | 0 | 0 | $\theta_2$ | [-85，85] |
| 3 | 0 | $a_2 = 472$ | 0 | $\theta_3$ | [-90，90] |
| 4 | -90 | $a_3 = 55$ | $d_4 = 470$ | $\theta_4$ | [-90，90] |
| 5 | 90 | 0 | 0 | $\theta_5$ | [-90，90] |
| 6 | -90 | 0 | $d_6 = 261$ | $\theta_6$ | [-120，120] |

Based on the link parameters, the homogeneous transformation from coordinate frame {i} to coordinate frame {i−1} can be determined, and the corresponding transformation matrix can be expressed as:

$$
\begin{aligned}
{}^{i-1}_{i}T &= R_X(\alpha_{i-1})D_X(a_{i-1})R_Z(\theta_i)D_Z(d_i) \\
&= \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1}d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{1}
$$

where,$c\theta_i=\cos\theta_i$,$s\theta_i=\sin\theta_i$,$c\alpha_{i-1}=\cos\alpha_{i-1}$,$s\alpha_{i-1}=\sin\alpha_{i-1}$.In the forward kinematics analysis of the manipulator, the pose transformation matrix of the end-effector coordinate frame {6} with respect to the base coordinate frame {0} can be derived by sequentially multiplying the homogeneous transformation matrices of the six links. This transformation matrix accurately represents the spatial position and orientation of the end-effector in the base coordinate frame:

$$
{}^{0}_{6}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T\,{}^{4}_{5}T\,{}^{5}_{6}T = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}
$$

ISSN: 2414-1895 DOI: 10.6919/ICJE.202602_12(2).0005

Therefore, ${}_6^0\mathrm{T}$ is the forward kinematic pose matrix of interest. It indicates that, for a given known configuration defined by the six joint angles, the corresponding position and orientation of the manipulator end-effector can be obtained.

### 3.2 Jacobian-Based Iterative Solution of the Inverse Kinematics for a 6R Robot

The previous section addressed the solution of the forward kinematics of the manipulator. In contrast, the inverse kinematics problem is one of the core issues in the control of manipulator motion, as its objective is to determine the joint angles that enable the end-effector to reach a specified position and orientation. Compared with forward kinematics, inverse kinematics is generally more complex, since the corresponding equations may admit multiple solutions or no solution at all, and closed-form analytical solutions are often difficult to obtain.

To this end, a Jacobian-based iterative numerical method can be employed to address this problem. This approach is capable of effectively handling the highly nonlinear nature of inverse kinematics and is applicable to robots with arbitrary degrees of freedom.

The Jacobian[1,3] matrix J is a linear transformation that maps joint velocities to the end-effector velocity and is defined as:

$$\dot{\mathrm{x}} = \mathrm{J}(\mathrm{q}) \cdot \dot{\mathrm{q}} \tag{3}$$

Here, $\dot{x}$ denotes the end-effector velocity, including both linear and angular components; $\Delta q$ represents the joint angular velocity vector; and$J^{+}(q)$is the Jacobian matrix dependent on the current configuration $q$. For a 6R robot, the Jacobian matrix is a 6x6 matrix, where the i column describes the contribution of the i joint axis to the end-effector velocity. According to the end-effector pose matrix obtained from forward kinematics, we define:

$$\begin{cases} p_x = f_1(\theta_1,\theta_2,\theta_3,\theta_4,\theta_5,\theta_6) \\ p_y = f_2(\theta_1,\theta_2,\theta_3,\theta_4,\theta_5,\theta_6) \\ p_z = f_3(\theta_1,\theta_2,\theta_3,\theta_4,\theta_5,\theta_6) \\ n = f_4(\theta_1,\theta_2,\theta_3,\theta_4,\theta_5,\theta_6) \\ o = f_5(\theta_1,\theta_2,\theta_3,\theta_4,\theta_5,\theta_6) \\ a = f_6(\theta_1,\theta_2,\theta_3,\theta_4,\theta_5,\theta_6) \end{cases} \tag{4}$$

Accordingly, the Jacobian matrix can be expressed as:

$$J(q) = \begin{bmatrix} \frac{\partial f_1}{\partial \theta_1} & \frac{\partial f_1}{\partial \theta_2} & \cdots & \frac{\partial f_1}{\partial \theta_6} \\ \frac{\partial f_2}{\partial \theta_1} & \frac{\partial f_2}{\partial \theta_2} & \cdots & \frac{\partial f_2}{\partial \theta_6} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_6}{\partial \theta_1} & \frac{\partial f_6}{\partial \theta_2} & \cdots & \frac{\partial f_6}{\partial \theta_6} \end{bmatrix} \tag{5}$$

ISSN: 2414-1895 DOI: 10.6919/ICJE.202602_12(2).0005

During the iterative process of inverse kinematics, it is necessary to compute the variations in joint angles based on the desired changes of the manipulator end-effector. By rearranging Eq. (3), we obtain:

$$\Delta \mathrm{q} = \mathrm{J}^{+}(\mathrm{q}) \cdot \Delta \tag{6}$$

Here, $\Delta \mathrm{x}$ denotes the desired variation of the end-effector with respect to the target position, $\Delta q$ represents the variation in joint angles, and $\mathrm{J}^{+}(\mathrm{q})$ is the pseudoinverse of the Jacobian matrix. When the manipulator is not in a singular configuration, the Jacobian matrix is invertible, and the pseudoinverse reduces to the conventional inverse matrix. When the manipulator is in or near a singular configuration, the Jacobian matrix becomes non-invertible; in such cases, the damped least squares (DLS) method is commonly introduced to compute the pseudoinverse. This approach is an improvement over the traditional Jacobian iterative method and enhances the numerical stability of the algorithm in the vicinity of singularities.[8,9]

First, an initial joint configuration of the manipulator is selected, which is typically taken as the current state of the manipulator:

$$q_0 = [\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6] \tag{7}$$

As the initial value for iteration, the forward kinematics equation (2) is used to compute the end-effector pose matrix under the current configuration, including the position vector and the rotation matrix. Subsequently, the end-effector pose error vector is constructed. The position error is defined as the difference between the current end-effector position and the target position:

$$\Delta p = p_t - p \tag{8}$$

Here, $p_t$ denotes the target position vector. The orientation error is described by the relative rotation between the current orientation and the target orientation:

$$\Delta R = R \cdot R_t^T \tag{9}$$

This relative rotation is then converted into an equivalent axis–angle representation to obtain the corresponding angular velocity error vector $\Delta\omega$. Finally, the position error and the orientation error are combined to form a unified six-dimensional error vector:

$$\Delta x = [\Delta p; \Delta \omega] \tag{10}$$

On this basis, the Jacobian matrix $J(q_0)$ corresponding to the current joint configuration is computed. To avoid numerical difficulties associated with inverting the Jacobian matrix near singular configurations, a regularization is applied, and the resulting generalized inverse is defined as:

$$J^{+} = J^T \cdot (J \cdot J^T + \lambda^2 I)^{-1} \tag{11}$$

Here, $\lambda$ denotes the damping factor and $I$ is the identity matrix. The selection of the damping factor has a significant influence on the performance of the algorithm: when $\lambda$ is small, the algorithm exhibits faster convergence in regions far from singular configurations; as $\lambda$ increases, numerical

ISSN: 2414-1895 DOI: 10.6919/ICJE.202602_12(2).0005

oscillations near singular configurations can be effectively suppressed, thereby improving the stability of the iterative process.

Using the above generalized inverse matrix, the increment of the joint angles can be calculated as:

$$\Delta q = J^{+} \cdot \Delta x \tag{12}$$

Here, α is the step-size factor, which is used to limit the magnitude of the update in each iteration. A proper choice of the step-size factor can ensure algorithmic stability while maintaining an adequate convergence rate, thereby avoiding oscillatory or divergent behavior.

Subsequently, the joint variables are updated as follows:

$$q_{k+1} = q_k + \Delta q \tag{13}$$

In practical joint control of the manipulator, the updated joint angles must also be constrained to ensure compliance with the physical limitations imposed by the mechanical structure and the drive system.

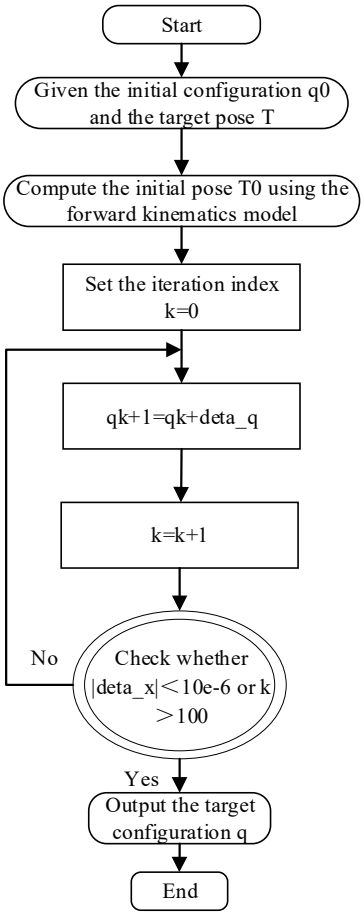


**Fig. 4** Flowchart of the Jacobian Iterative Algorithm

Finally, termination conditions are evaluated to determine whether the iterative process should be stopped. Common termination criteria include the norm of the error vector falling below a predefined threshold or the number of iterations reaching a specified maximum. When the termination condition is satisfied, the inverse kinematics problem is considered to be solved; otherwise, the next iteration is

ISSN: 2414-1895 DOI: 10.6919/ICJE.202602_12(2).0005

performed. Through the above iterative procedure, the joint configuration of the manipulator gradually converges to a solution that enables the end-effector to reach the target position and orientation. The overall algorithmic flowchart is shown in Fig. 4.

### 3.3 Trajectory Planning Verification via MATLAB Simulation Platform

To simulate the motion trajectory of the manipulator end-effector, a target point $P$ is defined in the Cartesian coordinate space, while the end-effector position corresponding to the initial configuration of the 6R manipulator is denoted as point $P_0$. These two points define a straight line segment $P_0P$ in space. Since the relationship between the end-effector position and the joint space is nonlinear, a Cartesian straight-line interpolation–based trajectory planning method is adopted to ensure that the manipulator moves along the desired linear path.

Specifically, the end-effector pose matrices $T_P$ and $T_{P0}$ are discretized along the linear direction such that each resulting intermediate pose matrix yields an end-effector motion in the Cartesian coordinate frame that approximates straight-line motion. In theory, increasing the number of subdivisions allows the trajectory to approach an ideal straight line; however, in practical applications, a finite number of subdivisions is sufficient to meet engineering requirements. Accordingly, the iii-th interpolated pose matrix between $T_P$ and $T_{P0}$ is given by:

$$T_{Pi} = \begin{bmatrix} R & i\Delta t(P-P_0) \\ 0 & 1 \end{bmatrix} \tag{14}$$

Here, $P$ and $P_0$ denote the spatial coordinate vectors of the manipulator end-effector corresponding to the target configuration and the initial configuration, respectively. *R* represents the rotation matrix of the end-effector with respect to the base coordinate frame. $\Delta t$ is the time interval between adjacent interpolated pose matrices; in this study, $\Delta t$=0.02s is adopted, and the total interpolation time for the overall trajectory planning is set to 3 s. Accordingly, a set of pose matrices corresponding to linear motion can be obtained as follows:[3,5]

$$Q = \{T_{p0}, T_{pi}, T_p\} \tag{15}$$

Accordingly, the straight-line trajectory planning in the Cartesian coordinate space is illustrated in the figure.

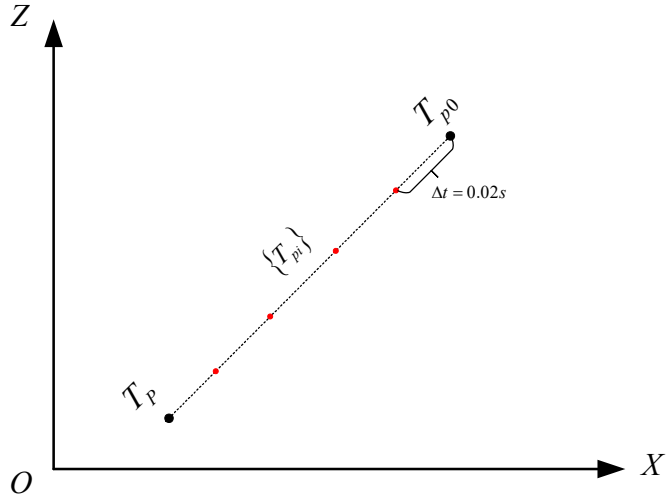


**Fig. 5** Straight-Line Trajectory Planning in the Cartesian Coordinate System

ISSN: 2414-1895 DOI: 10.6919/ICJE.202602_12(2).0005

When the manipulator takes $T_{P0}$ as the initial grasping pose matrix and moves along the direction of the vector $P_0P$, the velocities and accelerations at both $T_P$ and $T_{P0}$ are set to zero. For the intermediate points, the accelerations are set to zero, and the velocity direction at each point is defined to point toward the position of the subsequent time step. Since the manipulator end-effector performs only linear straight-line motion, no angular velocity of the end-effector is involved. Accordingly, the six-dimensional velocity vector of the manipulator end-effector is defined as:

$$V_{p0p} = \begin{bmatrix} v_x & 0 & v_z & 0 & 0 & 0 \end{bmatrix}^T \tag{16}$$

Here, $v_x$ and $v_z$ denote the components of the end-effector linear velocity along the X- and Z-axes, respectively. Accordingly, the joint-space angular velocity constraints at the remaining points can be expressed as:

$$\dot{q}_{pi} = J_{pi}^{-1} V_{p0p} \tag{17}$$

Here, $J_{pi}$ denotes the velocity Jacobian matrix of the manipulator at the configuration corresponding to point pi. By subsequently applying a quintic polynomial interpolation algorithm in joint space, smooth transitions of joint angular velocities can be achieved, thereby avoiding rigid impacts, discontinuities, and other undesirable dynamic effects during motion.

The MATLAB Robotics Toolbox can be used to generate the manipulator trajectory, as shown in Fig. 6. In the experiment, the manipulator end-effector moves along the predefined path AB, and the end-effector trajectory is a straight line. During the motion, the orientation of the end-effector remains constant, and only the position vector is linearly interpolated, thereby ensuring that the end-effector follows a strictly straight-line trajectory in Cartesian space.[6]

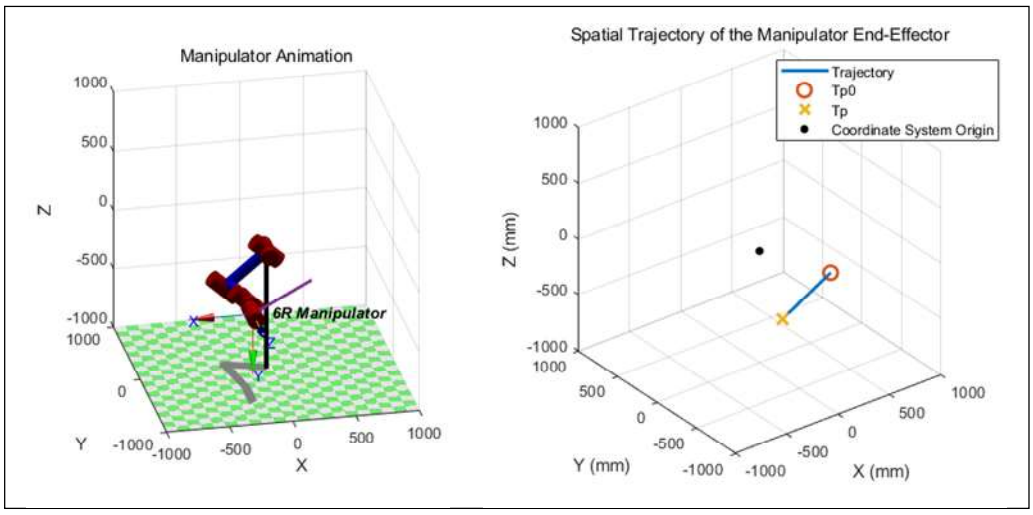


**Fig. 6** MATLAB Trajectory Simulation Results

## 4. Trajectory Planning Experiment of a 6R Manipulator

### 4.1 System Control Workflow and Test Platform Configuration

The testing procedure of the trajectory planning experimental platform is shown in Fig. 7, and the experimental setup of the platform is illustrated in Fig. 8. Specifically, the laser ranging module projects a laser point onto the graph paper to obtain the spatial coordinates of the laser spot. The microcontroller processes the acquired data and computes the corresponding manipulator

ISSN: 2414-1895 DOI: 10.6919/ICJE.202602_12(2).0005

configuration that reaches the laser point using the inverse kinematics formulation. Subsequently, a set of pose matrices from the initial position to the target position is generated using a Cartesian straight-line interpolation–based trajectory planning method.

The resulting pose matrix set is then sequentially converted into a set of joint configurations via the Jacobian-based iterative algorithm. Between successive joint configurations, quintic polynomial interpolation is applied to ensure the smoothness of joint velocities and joint accelerations, thereby preventing end-effector position jitter and discontinuities.[4]

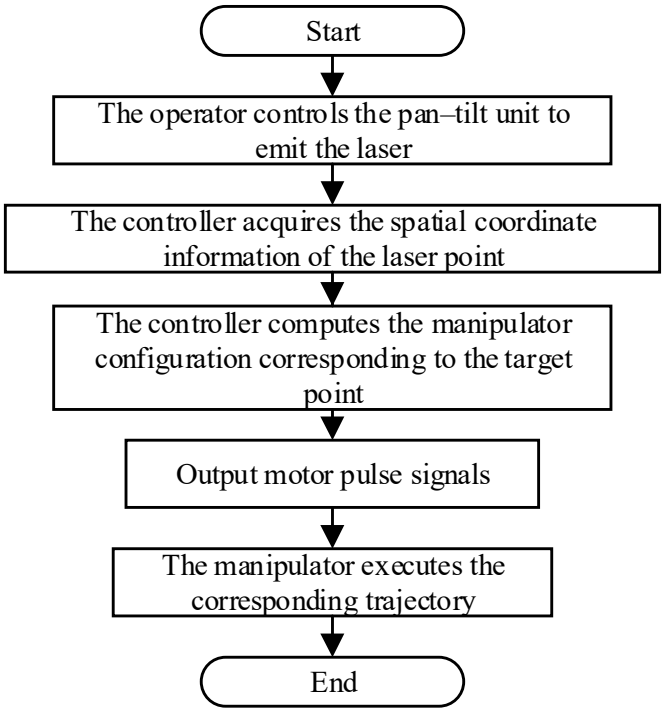


**Fig. 7** Schematic Diagram of the Trajectory Planning and Control System for a 6R Robot

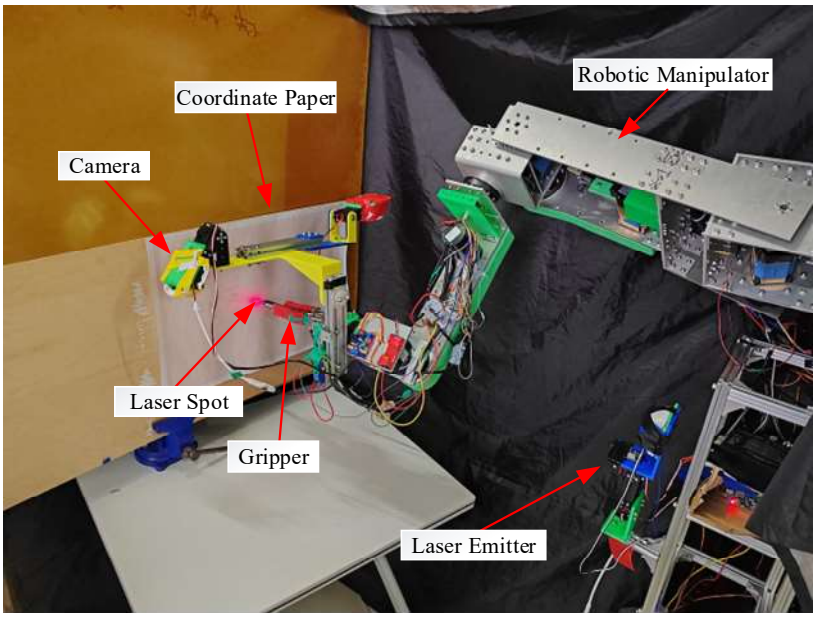


**Fig. 8** Construction of the Laser-Guided Trajectory Planning Experimental Platform

ISSN: 2414-1895 DOI: 10.6919/ICJE.202602_12(2).0005

As shown in Fig. 9, the experimental result of the final straight-line trajectory is presented.during the experiment, the manipulator successfully followed the prescribed straight-line trajectory from $T_{P0}$ to $T_P$, while the end-effector maintained a constant orientation throughout the motion. This result verifies the correctness of the Jacobian-based iterative method for solving the manipulator inverse kinematics and also demonstrates the reliability of the Cartesian straight-line interpolation–based trajectory planning approach.[11]

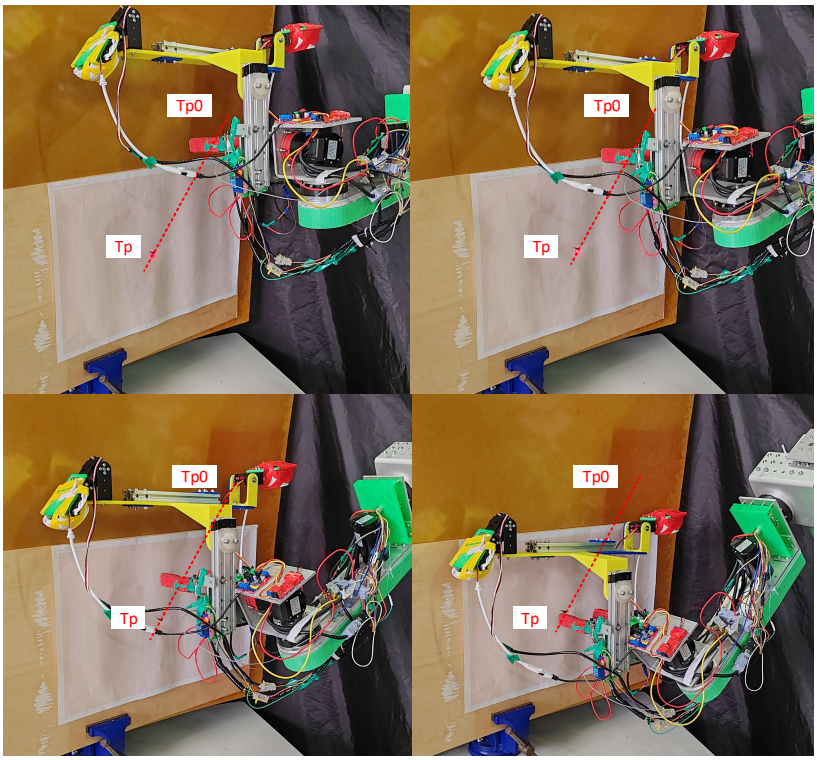


**Fig. 9** Experimental Setup for Laser-Guided Trajectory Planning

## 4.2 Analysis of Experimental Results

A coordinate system is established on graph paper to acquire the spatial coordinates of the laser point as well as the coordinates of the actual position reached by the manipulator end-effector, as shown in Fig. 10. Based on the Jacobian-based iterative method for solving inverse kinematics, the positioning errors between the manipulator end-effector and the laser point are analyzed under different numbers of iterations. The relationships between the end-effector position errors in the X, Y, and Z directions and the manipulator workspace are illustrated in Figs. 11, 12, and 13, respectively.

The experimental results indicate that, when an appropriate number of Jacobian iterations is applied, the end-effector position gradually converges to the laser point. The convergence errors in the X, Y, and Z directions relative to the laser-positioned point can all be controlled within 2 mm. Moreover, when the laser point corresponds to a larger manipulator motion range, a greater number of Jacobian iterations is required to achieve the same positioning accuracy.

The Jacobian-based iterative inverse kinematics method proposed in this study is capable of effectively achieving precise positioning of the manipulator end-effector under laser guidance. Experimental results demonstrate that, with appropriate selection of the initial configuration and iteration step size, the Jacobian iterative process exhibits favorable convergence properties. The end-effector position can stably converge to the laser point, with positioning errors in all three spatial directions controlled within the millimeter range.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202602_12(2).0005

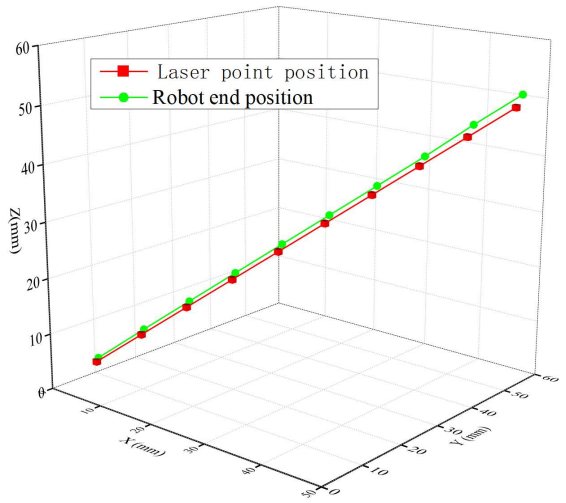


**Fig. 10** Position of laser point and robot end

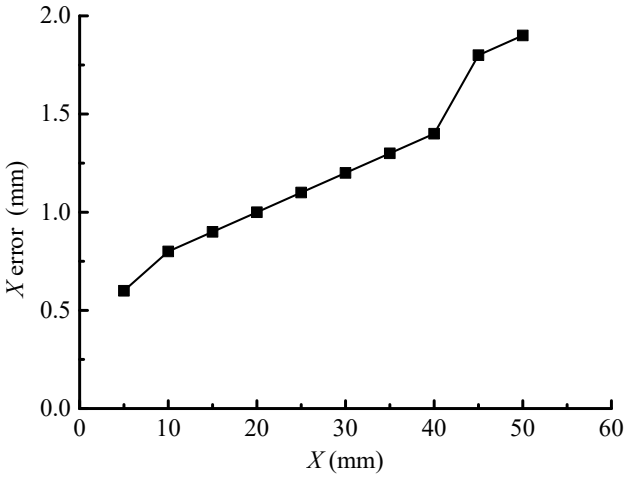


**Fig. 11** X position error of laser point and robot end

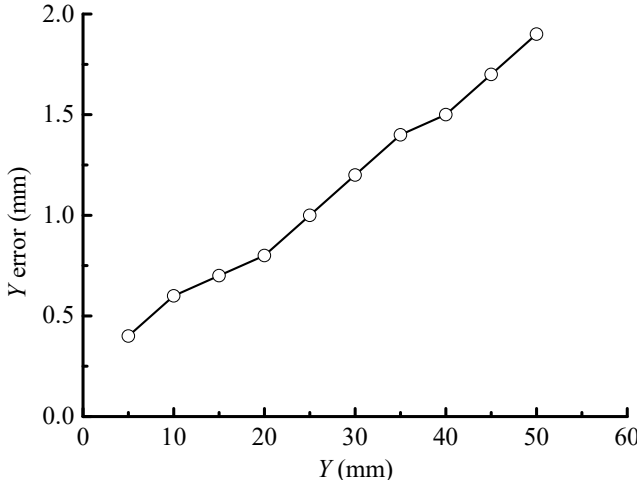


**Fig. 12** Y position error of laser point and robot end

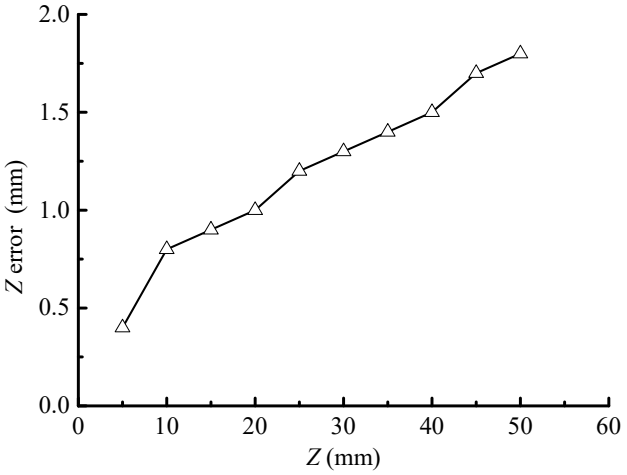


**Fig. 13** Z position error of laser point and robot end

Meanwhile, as the manipulator motion range associated with the target laser point increases, the nonlinear characteristics of the system become more pronounced, and a correspondingly larger number of Jacobian iterations is required to attain the same positioning accuracy. The above analysis verifies the effectiveness and engineering applicability of the adopted Jacobian-based iterative inverse kinematics method in laser-guided positioning control, and provides both theoretical and experimental support for subsequent fine-grained trajectory control and real-time positioning of robotic manipulators.[7,10]

## 5. Summary

This paper focuses on a 6R robot and proposes a Jacobian-based iterative algorithm for solving the robot inverse kinematics. On this basis, linear trajectory planning is implemented. A laser emitter projects a laser point onto graph paper to obtain the target point information, and the microcontroller acquires this information and computes the corresponding manipulator configuration for the target position. Under the conditions of a prescribed number of iterations and a sufficiently small error norm, the required joint angles of the manipulator configuration are obtained. Subsequently, quintic polynomial interpolation in joint space is applied to ensure that the end-effector follows the desired straight-line motion in Cartesian space.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202602_12(2).0005

After repeated experiments, the positioning accuracy of the laser-guided manipulator control system is evaluated. The experimental results indicate that, as the distance between the laser point and the initial configuration increases, the iterative positioning error correspondingly increases and exhibits an approximately linear distribution along the X, Y, and Z axes. Overall, the results demonstrate that the inverse kinematics errors obtained via the Jacobian iterative method generally satisfy the practical requirements of linear trajectory planning for the manipulator.

## Acknowledgements

Funded by the Nanchang Hangkong University Research Project (JY23048).

## References

[1] Siciliano, B., Sciavicco, L., Villani, L., and Oriolo, G., Robotics: Modelling, Planning and Control. Berlin, Germany: Springer, 2009.

[2] Buss, S. R., “Introduction to inverse kinematics with Jacobian transpose, pseudoinverse and damped least squares methods,” IEEE Journal of Robotics and Automation, 2004.

[3] Craig, J. J., Introduction to Robotics: Mechanics and Control, 3rd ed. Upper Saddle River, NJ, USA: Pearson, 2005.

[4] Spong, M. W., Hutchinson, S., and Vidyasagar, M., Robot Modeling and Control. New York, NY, USA: Wiley, 2006.

[5] Siciliano, B. and Khatib, O. (Eds.), Springer Handbook of Robotics. Berlin, Germany: Springer, 2008.

[6] Corke, P., Robotics, Vision and Control: Fundamental Algorithms in MATLAB. Berlin, Germany: Springer, 2011.

[7] Kucuk, S. and Bingul, Z., “Inverse kinematics solutions for industrial robot manipulators with offset wrists,” Applied Mathematical Modelling, vol. 38, no. 7–8, pp. 1983–1999, 2014.

[8] Hollerbach, J. M. and Suh, K. C., “Redundancy resolution of manipulators through torque optimization,” IEEE Journal of Robotics and Automation, vol. 3, no. 4, pp. 308–316, 1987.

[9] Safeea, M., Bearee, R., and Neto, P., “A modified damped least-squares scheme with controlled cyclic solution for inverse kinematics in redundant robots,” IEEE Transactions on Industrial Informatics, vol. 17, no. 12, pp. 8014–8023, 2021.

[10] Xu, J., Song, K., He, Y., Dong, Z., and Yan, Y., “Inverse kinematics for 6-DOF serial manipulators with offset or reduced wrists via a hierarchical iterative algorithm,” IEEE Access, vol. 6, pp. 52899–52910, 2018.

[11] Zhang, J., Li, Y., and Wang, H., “Laser-based positioning and control of robotic manipulators,” Robotics and Computer-Integrated Manufacturing, vol. 29, no. 6, pp. 511–521, 2013