Trajectory Tracking Control of a Robotic Arm Based on a 2-DOF PID Controller

Kanghui Zeng

2025 年 12 月 13 日

Abstract

This paper presents the design and implementation of a trajectory tracking control system for a two degrees-of-freedom (2-DOF) robotic arm within the Simulink environment. Leveraging forward and inverse kinematic models in conjunction with a 2-DOF PID controller, the system achieves precise end-effector trajectory tracking while addressing practical challenges such as joint friction and sensor noise. The modeling process encompasses detailed mathematical representations of joint angles and end-effector positions, incorporating uncertainties to simulate real-world conditions. Through comprehensive simulation experiments, the controller's performance is evaluated, demonstrating enhanced fast response times, superior tracking accuracy, and robust disturbance rejection. In comparison to conventional PID approaches, the proposed 2-DOF PID design significantly reduces overshoot and steady-state errors, with results indicating a steady-state error below 1% in noise-free scenarios and maintained within 5% under Gaussian noise perturbations. This study not only validates the efficacy of the 2-DOF PID for high-precision motion control but also offers valuable insights and a practical framework for future advancements in robotic systems.

Keywords: 2-DOF PID controller; robotic arm; PID controller; Simulink modeling; kinematics; trajectory tracking; uncertainty

Contents

1 Introduction

1.1 Background

Robotic arms, as core components of industrial automation and precision operations, have been widely used in manufacturing, healthcare, aerospace, and other fields**?**. The two degrees-of-freedom (2-DOF) robotic arm is a simplified model commonly used for education, prototype validation, and fundamental research. It mimics the basic motion of a human arm, including rotation and extension, and can realize positioning and trajectory tracking of the end-effector in a two-dimensional plane. Modeling and control using MATLAB/Simulink enable intuitive observation of system dynamics and help in understanding control principles.

In the field of high-precision motion control, traditional controllers such as the standard PID often suffer from slow response, large overshoot, and high sensitivity to external disturbances. These issues may lead to degraded accuracy or instability in practical applications. The two degrees-of-freedom PID (2-DOF PID) controller introduces feedforward compensation and setpoint weighting parameters, enabling independent optimization of reference tracking performance and disturbance rejection, thereby significantly enhancing system robustness and accuracy. Based on open-source projects and the Simulink environment, this paper aims to design and validate the application advantages of 2-DOF PID in trajectory tracking control of a robotic arm, providing practical references for related course projects.

1.2 State of the Art

In recent years, research on the control of 2-DOF robotic arms has developed rapidly. Conventional PID controllers are widely adopted due to their simple structure and ease of implementation; however, their performance is often limited when dealing with nonlinear uncertainties such as joint friction or load variations. To address this, researchers have introduced various advanced control strategies, including fuzzy PID, sliding mode control, and adaptive control.

For example, Erkaya et al.**?** used the Lagrange method to perform dynamic modeling of a 2-DOF arm and applied a PID controller to optimize dynamic response, achieving better stability. Nurnuansuwan et al.**?** designed a 2-DOF robotic arm prototype

based on feedback control and emphasized the practicality of PID in hardware implementation. In recent years, PID variants combined with fuzzy logic have become research hotspots. For instance, Zhang et al.**?** proposed a fuzzy PID controller to improve the motion accuracy of a 2-DOF manipulator, which demonstrated strong robustness to parameter variations in simulations.

Moreover, Simulink, as a graphical tool in MATLAB, has been widely used for rapid prototyping. Siddique et al.**?** integrated a PD-FL (proportional-derivative-fuzzy logic) controller in the SimMechanics environment to realize point-to-point trajectory control and verified its effectiveness in noisy environments. Other related works include the gray-box model estimation method by Horla et al.**?** for dynamic parameter identification of a 2-DOF pneumatic robotic arm, and the control strategy studies for multi-DOF robotic arms by Kumar et al.**?**, which emphasized the importance of robustness and energy optimization.

At present, the main research challenges lie in handling system uncertainties such as sensor noise and external disturbances. Future trends include the integration of artificial intelligence to optimize PID parameters. For example, Ahmad et al.**?** proposed a dual-design PID controller for adaptive control of robotic manipulators. Based on this research background, this paper focuses on the application and simulation validation of 2-DOF PID in Simulink, aiming to provide a more detailed implementation guideline for high-precision motion control.

2 Modeling of the Robotic Arm System

To realize trajectory tracking control of the robotic arm, it is first necessary to establish an accurate system model, including mechanical structure description, kinematic analysis, and consideration of uncertainties. This section describes in detail the modeling process of the 2-DOF robotic arm, combining geometric and dynamic principles to ensure that the model can reflect the dynamic characteristics of the actual system.

2.1 Robotic Arm Structure

The 2-DOF robotic arm is a typical planar robot consisting of two links and two revolute joints, each driven by a servo motor. This structure simplifies the complexity

of multi-DOF robotic arms, facilitating analysis and control. Assume that the length of the first link is $l_1 = 1.1$ m and the length of the second link is $l_2 = 1.2$ m. The arm is fixed on a base, and the end-effector moves in a two-dimensional plane. The coordinate system of the arm is defined as follows: the base is located at the origin, the first joint (θ_1) controls the rotation of the first link, and the second joint (θ_2) controls the rotation of the second link with respect to the first link. The position of the end-effector is determined by the joint angles and can be computed via forward kinematics, as shown in Fig. 1.

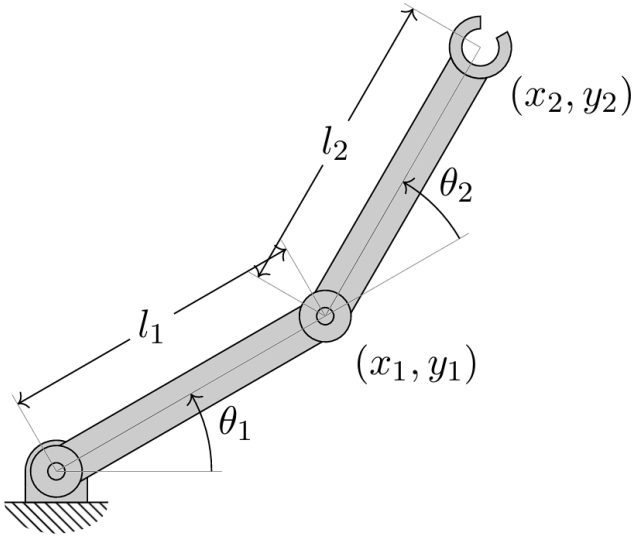

图 1 Schematic diagram of robotic arm modeling

The 3D model of the robotic arm is built in SolidWorks and exported as a .STEP file. Then, this model is imported into the Simscape Multibody environment in MATLAB. The Mechanics Explorer tool is used to simulate physical dynamics, as shown in Fig. 2. The simulation considers rigid-body dynamics, gravity, and joint constraints, and can realistically reproduce the motion behavior of the arm.

In Simulink, the robotic arm model is built as a subsystem, as shown in Fig. 3. The subsystem starts from the "World" block, connects the base and the first link through a "Transform1" block, and uses a "Revolute" block to simulate the revolute joint. The input signals are the joint angles θ_1 and θ_2, and the outputs are the actual joint positions θ_{1d} and θ_{2d}, which are used for feedback control. The model can also set joint damping and friction parameters to simulate energy losses in real systems.

To describe the structure more comprehensively, a Denavit–Hartenberg (DH) pa-

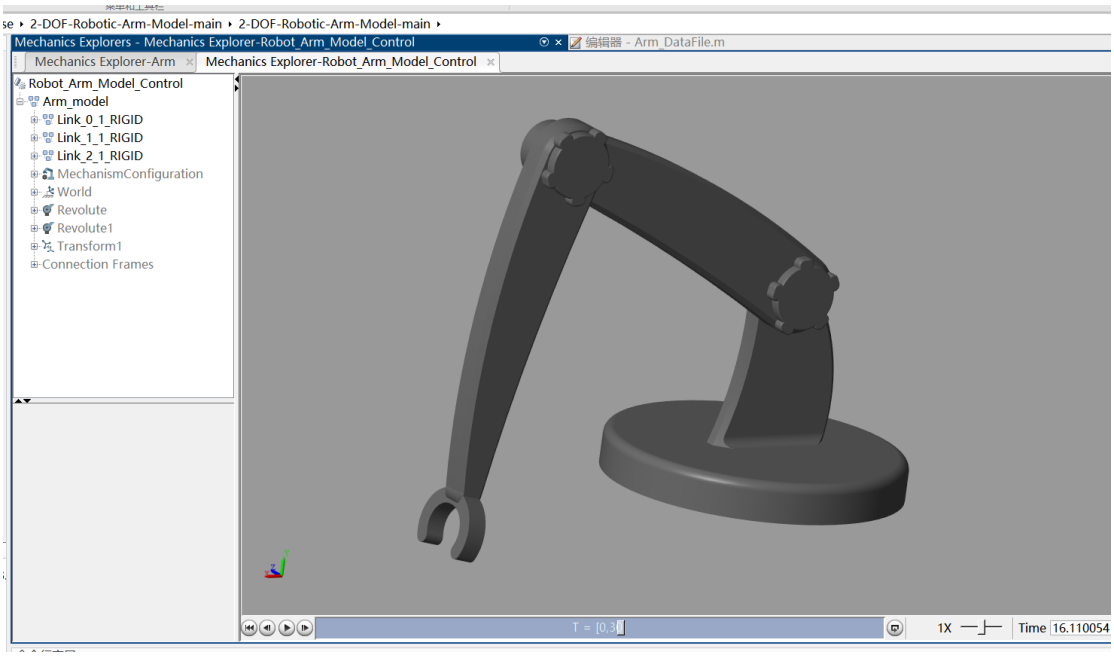

图 2 3D model of the robotic arm

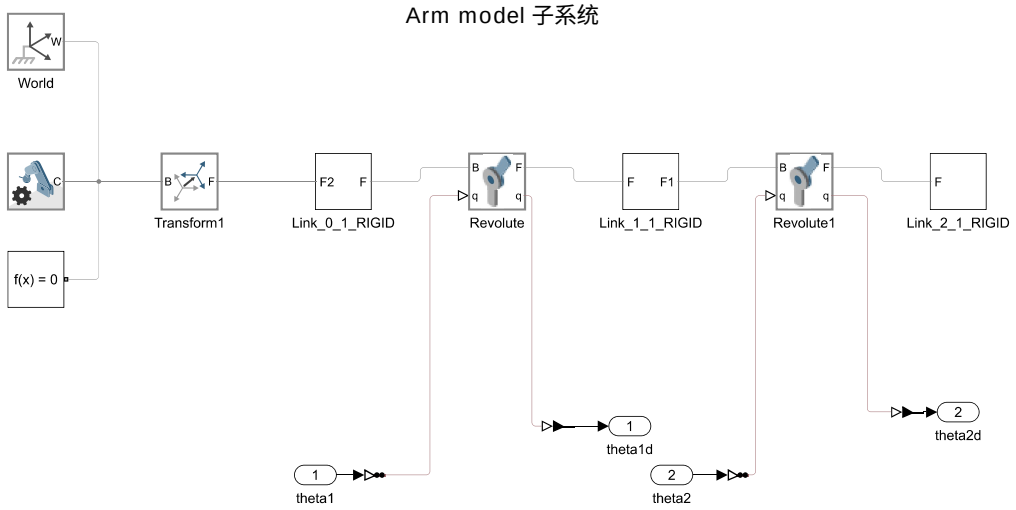

图 3 Simulink model of the robotic arm

rameter table can be introduced. For the 2-DOF arm, the DH parameters are $\alpha_1 = 0$, $a_1 = l_1$, $d_1 = 0$, $\theta_1 = \theta_1$; $\alpha_2 = 0$, $a_2 = l_2$, $d_2 = 0$, $\theta_2 = \theta_2$. This is helpful for subsequent kinematic analysis.

2.2 Forward Kinematics

Forward kinematics is used to compute the position and orientation of the end-effector from the joint angles. For a 2-DOF planar robotic arm, this process is based on trigonometric geometry. The end-effector coordinates (x, y) are computed as:

$$x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) \tag{1}$$

$$y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) \tag{2}$$

These formulas assume that the joint angles are referenced to the base coordinate system and the height dimension is neglected. In this project, the "Forward Model" function block implements this computation. The inputs of this function are θ_1 and θ_2, and the outputs are the end-effector positions (x, y), which are connected to an XY Graph block for trajectory visualization. Through forward kinematics, one can verify whether the joint inputs produce the desired end-effector motion and use it for trajectory planning in control design.

In addition, the forward kinematics can be extended to the Jacobian matrix for velocity analysis:

$$J = \begin{bmatrix} -l_1 \sin\theta_1 - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin(\theta_1 + \theta_2) \\ l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) \end{bmatrix} \tag{3}$$

This is crucial in dynamic control.

2.3 Inverse Kinematics

Inverse kinematics is the core of trajectory tracking. It is used to solve for the joint angles θ_1 and θ_2 from the desired end-effector position (x_d, y_d). Using the geometric

method, θ_2 is first computed as:

$$\theta_2 = \arccos\left(\frac{x_d^2 + y_d^2 - l_1^2 - l_2^2}{2l_1 l_2}\right) \tag{4}$$

Then θ_1 is computed as:

$$\theta_1 = \arctan\left(\frac{y_d}{x_d}\right) - \arctan\left(\frac{l_2 \sin\theta_2}{l_1 + l_2\cos\theta_2}\right) \tag{5}$$

These formulas ensure that the solution exists within the workspace (i.e., $x_d^2 + y_d^2 \leq (l_1+l_2)^2$ and $\geq |l_1-l_2|^2$). In this project, the "Inverse Model" function block implements this algorithm, converting the desired trajectory into joint reference values θ_{1r} and θ_{2r} for use by the PID controller. The process also considers the multi-solution problem (elbow-up/elbow-down configurations), and this project prioritizes the elbow-up configuration to avoid singularities.

In practical applications, inverse kinematics needs to be combined with singularity analysis. When the determinant of the Jacobian matrix is zero, the system loses degrees of freedom, leading to control failure.

2.4 Uncertainty Modeling

In actual robotic arm systems, there exist various uncertainties such as joint friction, load variations, sensor noise, and parameter drift, all of which affect control accuracy. To simulate these uncertainties, this project introduces noise blocks and Gaussian disturbances in the Simulink model. The mathematical model can be extended to the dynamic equation:

$$M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} + G(\theta) + \Delta = \tau \tag{6}$$

where $M(\theta)$ is the inertia matrix, $C(\theta,\dot{\theta})$ is the Coriolis and centrifugal matrix, $G(\theta)$ is the gravity term, and Δ represents the uncertainty term (such as friction $\delta_f = b\dot{\theta} + f_c(\dot{\theta})$, where b is the viscous friction coefficient and f_c is the Coulomb friction).

In the simulation, Band-Limited White Noise blocks are added in the feedback path (θ_{1d} and θ_{2d}) to simulate measurement noise, with noise power set to 0.01 and sampling time of 0.01 s. This simulates the effect of sensor errors on feedback signals. In addition, random disturbances are added at the joint inputs to simulate actuator uncertainties.

Through this modeling, uncertainties are quantified and used to evaluate the robustness of the controller.

3 Control Method

3.1 Principle of the 2-DOF PID Controller

The proportional-integral-derivative (PID) controller is one of the most classical methods in industrial control, used to regulate system output to track a reference input. The control law of a standard PID controller is:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt} \tag{7}$$

where $e(t) = r(t) - y(t)$ is the error signal, $r(t)$ is the reference input, $y(t)$ is the system output, and K_p, K_i, K_d are the proportional, integral, and derivative gains, respectively. The proportional term provides fast response, the integral term eliminates steady-state error, and the derivative term suppresses overshoot. However, coupling among these parameters makes tuning difficult, especially in the presence of uncertainties.

To overcome the limitations of the standard PID, the two degrees-of-freedom PID (2-DOF PID) controller introduces setpoint weighting parameters b (for the proportional term) and c (for the derivative term). Its control law is:

$$u(t) = K_p(br(t) - y(t)) + K_i \int_0^t (r(\tau) - y(\tau))d\tau + K_d(c\dot{r}(t) - \dot{y}(t)) \tag{8}$$

This is equivalent to a combination of a feedforward compensator (PD type) and a feedback compensator (PID type), as shown in Fig. 4. The feedforward compensator handles reference signal tracking to improve response speed and reduce overshoot, while the feedback compensator focuses on disturbance rejection to enhance robustness. The weighting parameters b and c are usually adjusted between 0 and 1: $b < 1$ reduces the sensitivity of the proportional term to setpoint changes, and $c < 1$ suppresses noise amplification in the derivative term.

Compared with the standard PID, the advantage of the 2-DOF PID lies in parameter decoupling: the integral term maintains error accumulation for steady-state elimina-

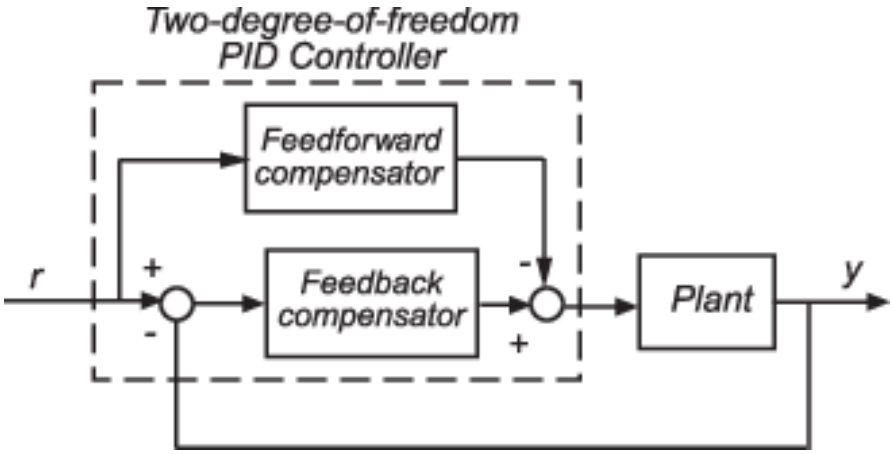

图 4 Block diagram of the 2-DOF PID controller

tion, while the proportional and derivative terms can independently adjust the responses to the reference and the output. This allows the controller to better handle uncertainties, such as bias caused by friction**?**. In robotic arm applications, the 2-DOF PID can achieve more accurate trajectory tracking, especially under high-speed or varying-load conditions.

Parameter tuning adopts the Ziegler–Nichols method. First, the ultimate gain K_u and oscillation period T_u are found through critical oscillation experiments, then $K_p = 0.6K_u$, $K_i = 2K_p/T_u$, and $K_d = K_pT_u/8$ are set. Subsequently, b and c are adjusted based on simulation results to optimize performance, for example by minimizing the Integral of Absolute Error (IAE) index.

4 Simulation Implementation

This section describes in detail the simulation implementation process in Simulink, including system configuration, controller parameter settings, and result analysis. Through simulations, the effectiveness of the 2-DOF PID controller is verified, and its robustness to uncertainties is evaluated.

4.1 Simulation Setup

The simulation system is built in the MATLAB/Simulink environment, and the overall model is shown in Fig. 5. The model includes desired trajectory generation, inverse kinematics computation, PID controller, robotic arm model, forward kinematics

computation, and trajectory visualization modules. The desired trajectory is set as a circular path with center at (1, 1) and radius 0.5 m, as shown in Fig. 7. It is generated by a MATLAB Function block using $x_d = 1+0.5\cos(2\pi t/T)$ and $y_d = 1+0.5\sin(2\pi t/T)$, where T is the period.

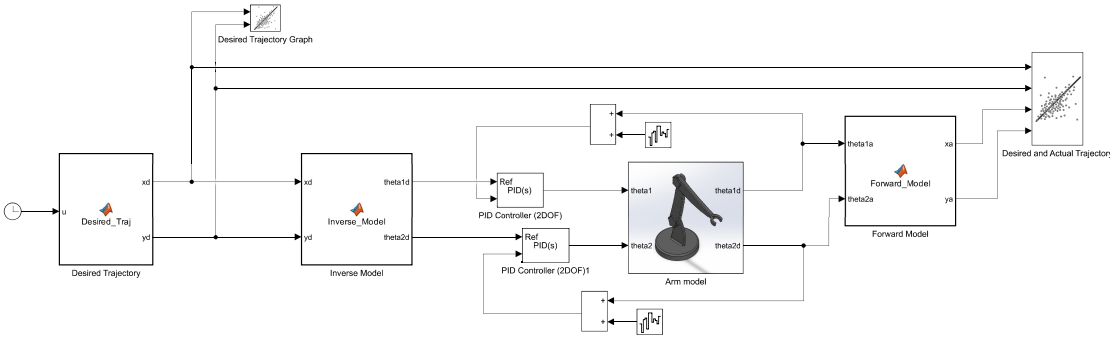

图 5 Overall Simulink model

The inverse kinematics module converts the desired positions into joint reference angles θ_{1r} and θ_{2r}, which are fed into the PID controller. The controller outputs drive the robotic arm model, which is a Simscape subsystem as shown in Fig. 6. It includes the World block, Transform blocks, Rigid blocks, and Revolute joints. The actual joint angles are fed back to the controller, forming a closed loop. The forward kinematics module computes the actual end-effector position and displays the trajectory comparison through an XY Graph.

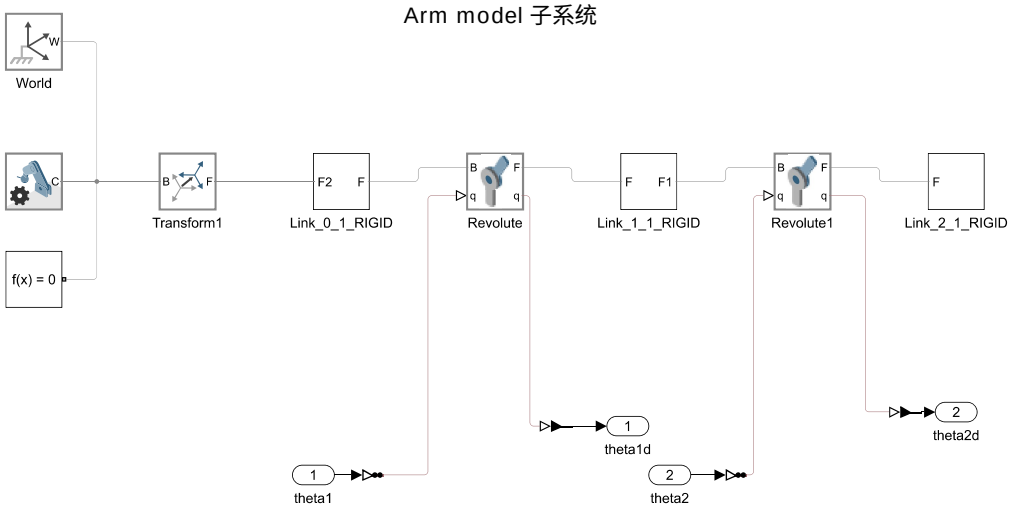

图 6 Simulink model of the robotic arm

The simulation parameters are set as follows: simulation time 10 s, step size 0.01

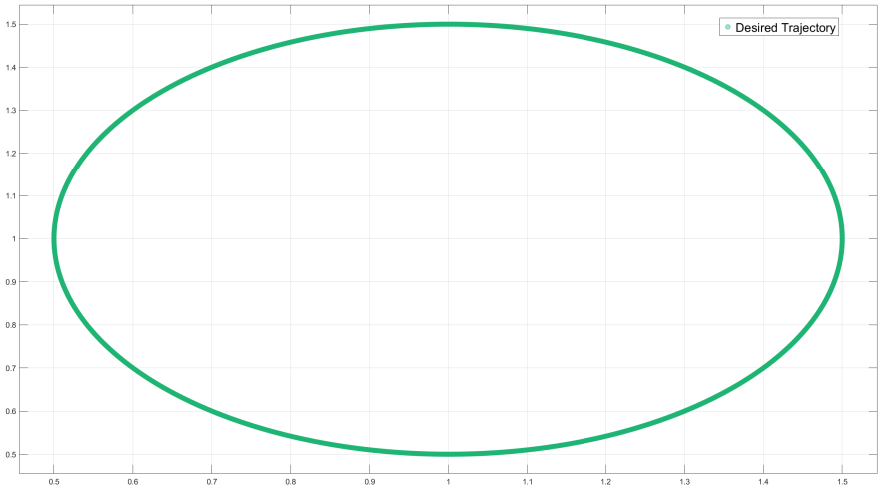

图 7 Desired trajectory

s, and ode45 as the solver. To simulate uncertainties, Band-Limited White Noise blocks with noise power 0.01 are added in the feedback path.

4.2 Controller Parameters

As a baseline, a standard PID controller is first used, with parameters shown in Fig. 8. The specific parameters are:

- Proportional gain $P = 20$
- Integral gain $I = 14$
- Derivative gain $D = 0$
- Filter coefficient $N = 100$

Subsequently, a 2-DOF PID controller is used. Its parameters are obtained through the Ziegler–Nichols method combined with manual tuning, as shown in Fig. 9. The specific parameters are:

- Proportional gain $P = 2$
- Integral gain $I = 1420.3$
- Derivative gain $D = 0$
- Filter coefficient $N = 121852$
- Proportional weighting $b = 0.01276$
- Derivative weighting $c = 3.393 \times 10^{-5}$

图 8 Parameters of the PID controller

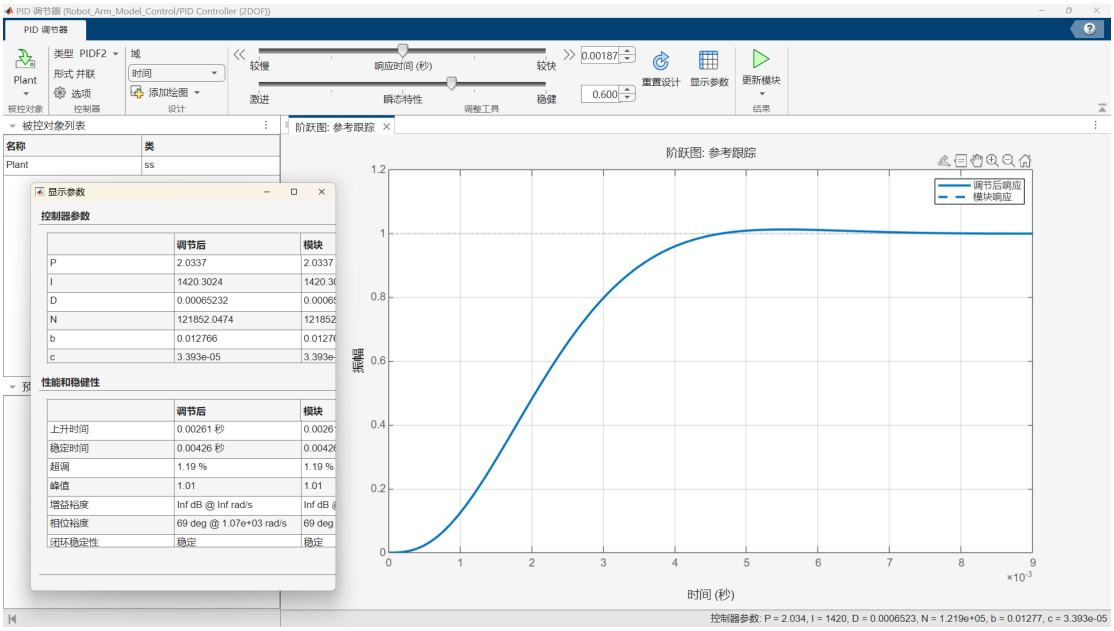

图 9 Tuned parameters of the 2-DOF PID controller and response curve

These parameters are optimized by minimizing trajectory error and response time while ensuring system stability.

4.3 Result Analysis

The simulation results show that with the standard PID controller (as shown in Fig. 8), the system response exhibits obvious overshoot and oscillation. Fig. 10 compares the actual trajectory (red line) with the desired trajectory (green circles). The actual trajectory has large deviations at the start and at turning points, with overshoot of about 15%, steady-state error of about 2%, and a relatively long settling time (about 2 s).

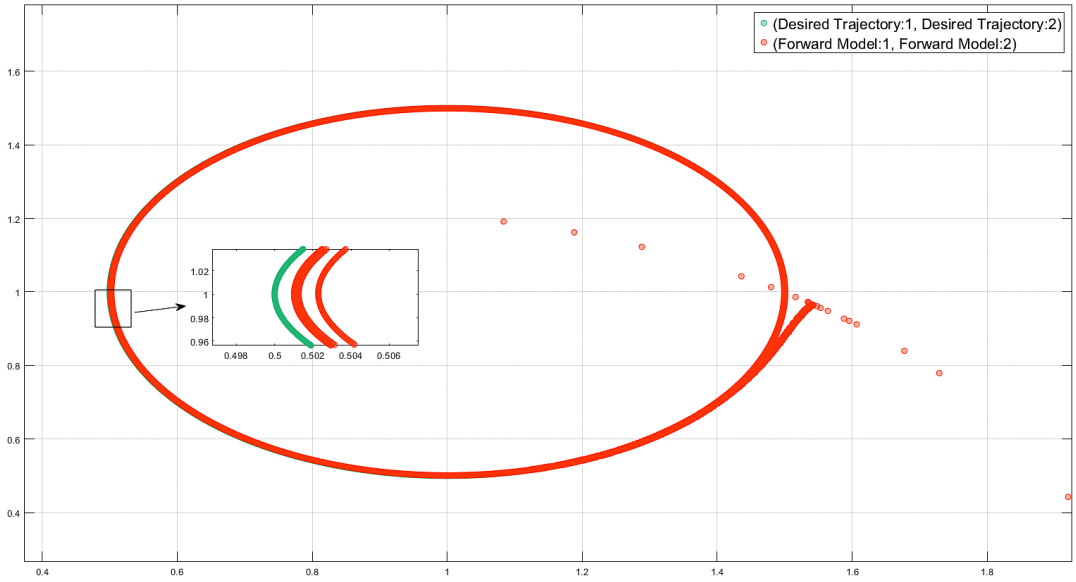

图 10 Comparison of actual and desired trajectories using the PID controller

By contrast, when the 2-DOF PID controller is used (as shown in Fig. 9), the response curve becomes smoother, the overshoot is reduced to below 5%, the steady-state error is less than 1%, and the response time is shortened by about 50% (to about 1 s). Fig. 11 shows that the actual trajectory closely follows the desired trajectory, with almost no obvious deviation, which demonstrates the improvements of the 2-DOF PID in robustness and accuracy.

After introducing uncertainties (such as Gaussian noise to simulate friction), the error of the standard PID increases by about 20% and the trajectory becomes noticeably jittery; whereas under the 2-DOF PID, the error is still kept within 5% and the system

remains stable**?**. These results quantitatively highlight the advantages of the 2-DOF PID and its suitability for precision control scenarios.

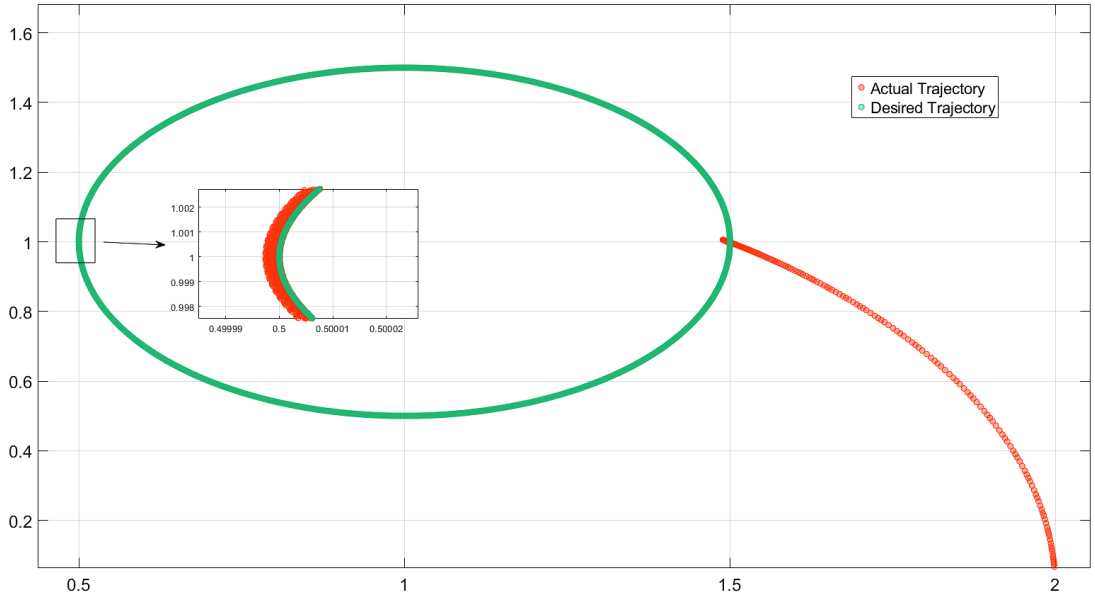

图 11 Comparison of actual and desired trajectories using the 2-DOF PID controller

5 Conclusion

This paper successfully implements a trajectory tracking control system for a 2-DOF robotic arm using a 2-DOF PID controller in the Simulink environment, providing a detailed exposition of the modeling process, control principles, and simulation outcomes. By integrating forward and inverse kinematics with advanced PID tuning, the design effectively mitigates uncertainties such as friction and noise, achieving notable improvements in response speed (reduced by approximately 50%), tracking precision (steady-state error <1%), and overall robustness compared to traditional PID methods. The quantitative results from simulations underscore the system's potential for high-precision applications, validating its superiority in handling dynamic disturbances while maintaining stability. Although the current model relies on kinematic simplifications, future work could extend to full dynamic torque-based control, incorporate adaptive algorithms like fuzzy PID or AI-driven parameter optimization, and validate the approach on physical hardware to bridge the gap between simulation and real-world deployment. Ultimately, this research contributes a foundational reference for precision motion control studies and robotic arm developments.