Robust Trajectory Tracking of Two-Link Rehabilitation Robot Under Sudden External Disturbances By GA-Optimized PID Control

Xiangcheng Long *

Department of Mechanical Engineering, Hebei University of Technology, Tianjing, 300401, China

* Corresponding Author Email: 230553@stu.hebut.edu.cn

Abstract. As populations age and the need for rehabilitation grows, rehabilitation robots have drawn increasing attention. These devices guide impaired limbs through precise, repeatable motions to accelerate recovery. A central difficulty, however, is guaranteeing smooth human-robot interaction when unexpected disturbances arise during exercise. This study proposes a Proportional-Integral-Derivative (PID) controller whose six gains are globally optimized by a genetic algorithm to secure robust joint tracking for a planar two-link rehabilitation robot working with patients. PID reproducible disturbance-injection interface uses the Jacobian to convert end-effector forces into joint torques, simulating sudden patient-generated perturbations. The control law adds gravity feed-forward, a filtered PID term, and explicit actuator saturation limits. The genetic algorithms (GA) fitness function mixes time-weighted tracking error, overshoot penalty, recovery time, torque rate, and saturation duration to favor robustness. MATLAB simulations show the arm returning to within ±2° in 1.5 s, exhibiting 20 % overshoot while saturation remains low; the tuned controller outperforms a manually adjusted PID. The scheme is straightforward to deploy and meets the dual demand for safety and accurate trajectory tracking in clinical use.

Keywords: Rehabilitation robot, Genetic Algorithm-optimized Proportional-Integral-Derivative Controller, external disturbance.

1. Introduction

Because rehabilitation manipulators remain in direct contact with patients, they are repeatedly hit by sudden, unpredictable forces from muscle spasms or caregiver adjustments. Proportional-Integral-Derivative (PID) control is still the default in medical hardware; its transparency and long certification record outweigh its drawbacks, yet calibrating PID parameters for nonlinear, coupled robot dynamics that experience abrupt loading is far from straightforward [1]. Genetic Algorithms (GA) perform global search and have become a standard tool for automatically selecting PID gains in intricate plants [2]. Robotic manipulators benefit from well-documented dynamic models and control theory. Craig's early treatments of manipulator kinematics and dynamics, for example, supply a rigorous basis for accurate modelling and underpin today's control designs [3-5]. And Khatib's introduction of impedance control reshaped how robots physically interact with humans. Furthermore, Wang et al. merged PID with adaptive control in a hybrid scheme that maintains performance when the payload changes [6-8].

Meanwhile, Robust control and observer-based schemes have drawn growing interest because they can counter external disturbances and model uncertainty. Spong and Vidyasagar [9], for instance, presented a unified framework for nonlinear control of robot arms that centers on the stability and robustness of feedback loops. Their passivity-based design has since become standard for guaranteeing stability despite disturbances or modeling errors, and it has shaped later work on adaptive robot control. Khatib [10] introduced impedance control to improve human-robot interaction by treating the arm as a dynamic system of virtual springs and dampers. The method lets the robot respond compliantly to external forces while keeping motion safe and accurate, laying the groundwork for collaborative robots that share workspaces with humans.

These developments have prompted a wave of studies that combine observer-based control with adaptive methods to make robotic systems more resilient. In rehabilitation, as report that, once

carefully tuned and bounded, PID-type controllers can still deliver tracking accuracy and robustness comparable to more elaborate schemes on exoskeletons [9].

Prompted by these challenges, this paper introduces a practical GA-PID control workflow and subject it to disturbances that replicate patient-induced perturbations. The design extends proven robust control ideas by embedding genetic algorithm optimization within the PID loop to sharpen trajectory tracking for a two-link rehabilitation robot. A disturbance-injection interface was built to reproduce the erratic, time-varying forces patients exert during exercise. This interface allows the robot's response to sudden perturbations to be examined in detail through simulation. The GA-PID tuner simultaneously optimizes six gains against a robustness index that blends tracking error, overshoot, recovery time and actuator saturation. The resulting controller returns the robot to its reference within 1.5 s, keeps overshoot below 20 % and prevents saturation under patient-like disturbances, confirming that the workflow raises both stability and robustness during human-robot interaction. ≤

2. Mathematical Model

To examine how the controller reacts to abrupt external disturbances, an accurate mathematical model of the rehabilitation manipulator must first be built. A representative two-link planar arm moving in two dimensions is chosen as the test case. The configuration captures the essential joints found in most upper-limb devices and exhibits pronounced coupling, nonlinear dynamics, and a marked gravitational component. The corresponding model is displayed in Fig. 1, which depicts a standard two-link manipulator frequently employed in rehabilitation. Two rigid segments are joined by revolute joints, each allowing independent rotation.

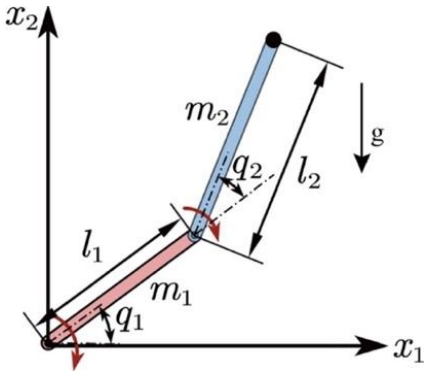

Figure 1. Schematic of the two-link manipulator and coordinate definitions. (Picture credic: Original)

2.1. Dynamic Model

Based on the Lagrange equation, the system dynamics model can be obtained in equation (1):

$$\mathrm{M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + D\dot{q} = \tau + \tau_d} \quad (1)$$

The relevant parameters are listed in Table 1. The detailed expressions are given in equations (2–4). In these equations, the inertia matrix M(q) captures how strongly the system resists angular acceleration. It depends on the joint positions and dictates how joint torques propagate through the links. By incorporating both link masses and their geometric arrangement, this matrix shapes the robot's dynamic response and governs the energy stored during motion. In equation (2), m_1 and m_2 stand for the link masses, while l_1 and l_2 represent their lengths, quantities that directly determine the robot's rotational inertia. $q_1 q_2 m_1 m_2\ l_1 l_2$

And the matrix C (q, q̇) captures the Coriolis and centrifugal forces generated by the manipulator's motion. These velocity-dependent effects stem from coupling between the robot's two joints; whenever the links move together, the resulting interaction becomes pronounced. The symbol q̇ denotes joint velocities, and the matrix quantifies the extra torques required to counteract the

nonlinear joint interactions. The gravitational force vector G(q) describes how gravity acts on the manipulator, computing the torque produced by the weight of the links as a function of joint position. In equation (3).

Table 1. The Parameters about equation (1)

q	$[q_1, q_2]^T$ is the joint angle vector
τ	$[\tau_1, \tau_2]^T$ to control the torque
τ_d	τ_d external interference torque
M(q)	M(q) is an inertial matrix $(q,\dot{q})$ is the Coriolis force and centrifugal force matrix
G(q)	G(q) is the gravity matrix, is the viscous friction term

$$M(q)=\begin{bmatrix}(m_1+m_2)l_1^2+m_2l_2^2+2m_2l_1l_2\cos q_2 & m_2l_2^2+m_2l_1l_2\cos q_2\\ m_2l_2^2+m_2l_1l_2\cos q_2 & m_2l_2^2\end{bmatrix} \quad (2)$$

$$C(q,\dot{q})=\begin{bmatrix}m_2l_1l_2\sin q_2\dot{q}_2 & m_2l_1l_2\sin q_2(\dot{q}_1+\dot{q}_2)\\ m_2l_1l_2\sin q_2\dot{q}_1 & 0\end{bmatrix} \quad (3)$$

The gravity matrix lets the control algorithm offset gravitational pull, so the robot can resist the downward force on the arm during rehabilitation exercises. It folds in the mass of each link and where that mass sits in the gravitational field. Equation (4) shows the result.

$$G(q)=\begin{bmatrix}(m_1+m_2)gl_1\cos q_1+m_2gl_2\cos(q_1+q_2)\\ m_2gl_2\cos(q_1+q_2)\end{bmatrix} \quad (4)$$

2.2. Kinematic Model

The end-effector position follows from the two-link forward-kinematic model. The two-link forward-kinematic equation (5) maps the joint angles θ_1 and θ_2 to the Cartesian coordinates of the end-effector. Because it links joint-space variables to physical position, the expression underpins both trajectory planning and control.$q_1$andq_2The Jacobian J(q) quantifies how small variations in the joint angles propagate to end-effector motion.

$$p(q)=\begin{bmatrix}l_1\cos q_1+l_1\cos(q_1+q_2)\\ l_1\sin q_1+l_2\sin(q_1+q_2)\end{bmatrix} \quad (5)$$

Equation (6) gives the Jacobian matrix J(q) obtained from the kinematics of the two-link manipulator. The Jacobian links the joint-angle rates to the end-effector's Cartesian velocity, and it is central to control schemes that map forces or velocities between joint and task spaces. It quantifies how infinitesimal joint motions propagate to end-effector displacement in the workspace. q_1, q_2Its Jacobian matrix isshown in equation (6):

$$J(q)=\begin{bmatrix}l_1\sin q_1-l_2\sin(q_1+q_2) & -l_2\sin(q_1+q_2)\\ l_1\cos q_1+l_2\cos(q_1+q_2) & l_2\cos(q_1+q_2)\end{bmatrix} \quad (6)$$

2.3. Disturbance Model

Equation (7) describes the external force exerted by the patient on the robot during rehabilitation. Captured at the end-effector, this force is resolved into x-axis and y-axis components labeled and respectively. Accurate knowledge of this vector is essential for modeling patient-induced disturbances and for designing a controller that can counteract them. $F_{\{x\}}$ in rehabilitation training, the patient's sudden force is treated as the terminal external force $F_{\{y\}}$.

$$F_ext(t) = [F_x, F_y]^T \quad (7)$$

Transposing the Jacobian matrix projects the external wrench into joint-space disturbance torques. Equation (8) shows how the external wrench F_{ext}(t) is converted into joint torques $\tau(t)$ via the Jacobian J(q). This matrix translates end-effector forces into the joint torques needed to counteract them, a mapping the manipulator relies on during rehabilitation to generate appropriate joint-level responses.

$$\tau_d(t) = J\big(q(t)\big)^T F_e xt(t) \tag{8}$$

To mimic a sudden push or pull, the study applies a half-sine or square-wave disturbance. Equation (9) treats the patient's abrupt external force as a disturbance input to the manipulator. Cast as either a half-sine or square wave of amplitude A, it is switched on during the interval [t_start, t_end], beginning at t_start. The formulation replicates the sharp, transient loading common in rehabilitation, where the patient unexpectedly exerts force on the robot. By preserving the time-varying profile of these loads, the model captures the system has transient response throughout the exercise $T_d t_0$.

$$F_{ext}(t) = \begin{cases} A\sin\left(\frac{\pi(t-t_0)}{T_d}\right), & t_0 \le t \le t_0 + T_d \\ 0, & \text{other wise} \end{cases} \tag{9}$$

Where A is the amplitude of the disturbance force and T_d is the duration of the action.

2.4. Parameters Numbers

Model parameters are chosen to match the usual figures reported for rehabilitation robots and their patient interaction, capturing the dynamics and kinematics of a two-link arm employed in therapy tasks. To keep the simulation traceable, every angular quantity is given in radians. Parameters about model are shown in table 2.

Table 2. The Parameters about Model

parameter	symbol	Value	unit
First link length	l_1	0.5	m
Second link length	l_2	0.5	m
First link mass	m_1	2	kg
Second link mass	m_2	2.5	kg
Damping coefficient	d_1, d_2	0.16, 0.12	$N \cdot m \cdot s \backslash rad$
External force amplitude	A	80	N
External force duration	T_d	1	s

3. Controller Model

To counter the nonlinear dynamics and abrupt external disturbances inherent to a dual-link rehabilitation manipulator, the study employs a hybrid architecture that blends feedforward action, filtered PID regulation, and hard-limit protection. The scheme keeps the PID core—simple structure, transparent tuning—while filtering and saturation blocks stiffen the system against interference and shield the actuators. Fig. 2 outlines the complete design.

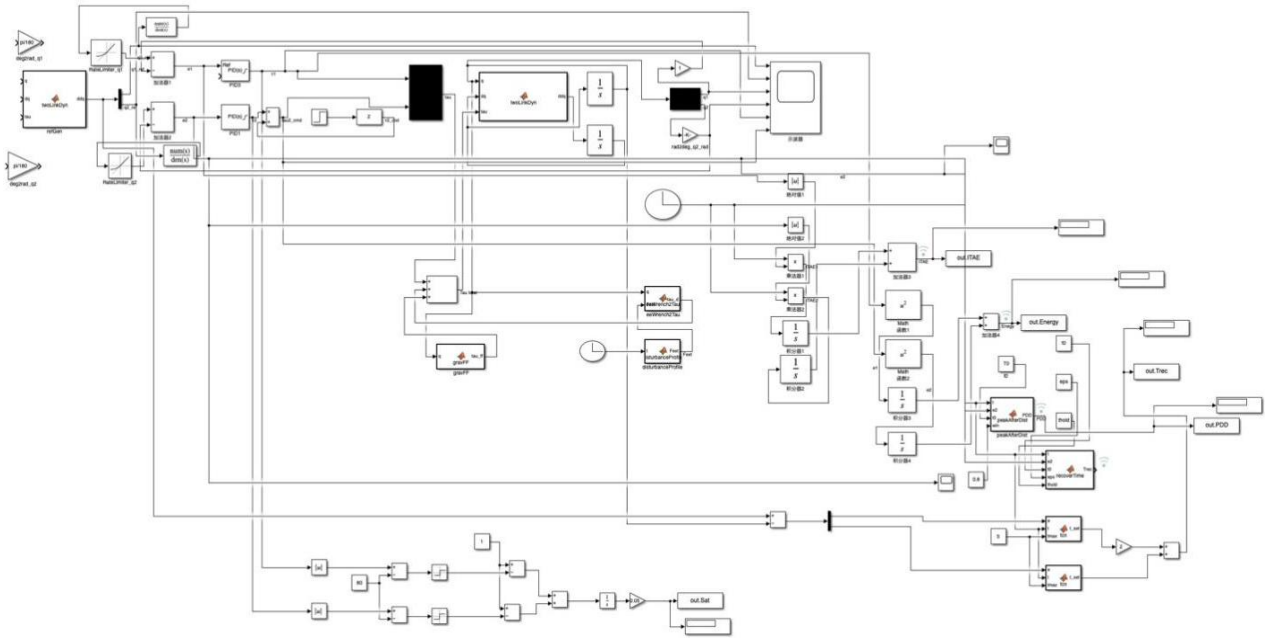

Figure 2. Overall control structure combining gravity compensation, filtered PID, saturation limit, and disturbance injection. (Picture credit: Original)

The signal is processed in several stages: The control loop begins with the desired trajectory $q_d(t)$ The tracking error derived from this trajectory feeds a filtered PID controller; its output is merged with a feedforward term, then clipped by a limiter to safeguard the actuators. The final torque command $\tau(t)$ drives the robot joints so the path is followed accurately.

In (t) denotes the desired joint angle, q(t) the measured angle, and the tracking error the difference between them, q_d the parameters are chosen from standard data for rehabilitation robots interacting with patients, capturing the dynamic and kinematic behaviour of a two-link arm during therapy.

Table 2 lists parameters drawn from established rehabilitation-robotics literature and widely accepted design conventions. The link lengths, masses, damping coefficients and external-force profiles mirror those found in clinical devices, while all angular quantities are given in radians to align with standard robot-control notation.

$$e(t) = q_d(t) - q(t) \tag{10}$$

The entire control block diagram includes the modules, which are included in Table 3.

Conventional PID differentiation amplifies high-frequency noise, so this study adopts a first-order filtered version of the controller. Equation (11) gives the torque command computed from the position error and its integral and derivative. The error e(t) is the difference between the desired trajectory and the measured position q(t); it is processed by the proportional, integral and derivative paths to generate the control action. τ_{PID} as shown in equation (11).

Table 3. The modules block diagram

Error Signal Calculation Module	Calculates the angle error between the two joints
tergal term, and filtered differential term	none
Feedforward Compensation Module	Offsets gravity and inertia, reducing the PID controller burden
Actuator Limiter Module	Limits output torque to prevent overdrive and saturation

$$\tau_i^{PIDf} = K_{pi}e_i + K_{ii}\int_0^t e_i(\xi)d\xi + K_{di}z_i \tag{11}$$

Where K_p, K_i, and $K_d$denote the proportional, integral, and derivative gains, respectively, e(t) represents the tracking error, $\dot{e}(t)$ is its time derivative, and the integral term accumulates past errors to enhance steady-state accuracy.

Equation (12) is crucial for controlling the robot's joints to reduce the tracking error and improve the stability of the system.

$$\dot{z}_i = N_i(\dot{e}_i - z_i) \tag{12}$$

N_i is the filter coefficient that helps to suppress high-frequency noise in the system. And $z_i(t)$ is the filtered error signal that is passed into the derivative term of the PID controller. The parameter N_i is the filter coefficient, which is used to suppress high-frequency noise in the differential term. The typical value range is N_i =5~30.

This structure is readily built in Simulink by inserting the "PID Controller (Filtered)" block, whose internal filter matches equation (13) gives the transfer function of the first-order low-pass filter that smooths the high-frequency noise introduced by the derivative action. While suppressing this noise, the filter leaves the overall dynamic response largely intact denotes the filter coefficient. N andis the complex frequency variable in the Laplace domain.

$$\frac{N}{1+\frac{s}{N}} \approx \frac{N}{s+N} \tag{13}$$

This approach suppresses noise without altering the dominant dynamic response. Table 4 lists the PID settings—proportional, derivative and integral gains together with the noise-filter coefficient N—for both joints of the manipulator, comparing the baseline controller with its GA-tuned counterpart. Each joint's parameters and the corresponding GA run are given in the same table $K_p K_d K_i$.

Table 4. Baseline and GA-optimized PID parameters

Joint	K_p	K_d	K_i	N	Method
1	30	1	3	150	GA
2	28	1.6000	2.8000	150	GA

To counteract the nonlinearities—gravity and Coriolis effects—of the two-link system, a dynamic compensation term is inserted ahead of the PID loop, streamlining the overall control effort. The control law is expressed as equation (14). Equation (14) gives the total torque $\tau(t)$produced by the manipulator, combining the feedforward component $\tau_{ff}(t)$with the PID-generated torque$\tau_{PID}(t)$. This sum forms the joint command that steers the robot along the desired trajectory.

$$\tau(t) = \tau_{ff}(t) + \tau_{PIDf}(t) \tag{14}$$

Where T_s is the smoothing time constant that defines the rate at which the reference trajectory is smoothed. And $q_d(t)$ is the reference trajectory at time t.

The feedforward torque term ise equation (15) computes the feedforward torque that offsets the gravitational, Coriolis, and centrifugal forces on the robot joints. Because these forces shape the system dynamics, the torque command must explicitly counteract them $\tau_{ff}(t)$.

$$\tau_{ff}(t) = M(q_d)\ddot{q}_d + C(q_d, \dot{q}_d)\dot{q}_d + G(q_d) \tag{15}$$

Where the inertia matrix, which captures how mass is distributed throughout the system. And $C(q, \dot{q})$ the Coriolis and centrifugal matrix, accounting for the nonlinear forces that appear during robot motion. G(q): the gravity vector, quantifying how gravitational pull influences the robot's movement.

In real-time control of rehabilitation robots, gravity compensation alone is often adopted to keep the model simple. $G(q_d)$ this simplification markedly reduces computational load. To stop the actuator from delivering an abrupt spike in torque when sudden disturbances appear, a saturation function is embedded to clamp the control signal. Equation (16) imposes this saturation limit, ensuring that the commanded torque never exceeds the actuator's physical ceiling and thereby protects the robot's joints and motors from damage caused by excessive force.

$$\tau_i = {}_{sat}[-\tau max, \tau_m ax]\left(\tau_i^P IDf + \tau_{(ff,i)}\right) \quad (16)$$

Where τ denotes the actual torque delivered to the actuator, τ_i, τmax and τmin are the upper and lower torque limits, usually fixed at ±1/80 of the stall torque, and τcmd = τPID + τff, the sum of the PID and feed-forward contributions, is clipped by the saturation function to prevent overdrive and keep the system within safe bounds.

This function keeps the robot safe by stopping the controller from commanding torques that the actuators cannot physically deliver. Where denotes the actuator's maximum permissible output, commonly fixed at 1/80 of its rated peak $N \cdot m$. Whenever the commanded value crosses this bound, the system clamps it and logs the saturation ratio; the GA later uses this record to gauge robustness. ptimization phase for robustness calculation.

Because a step change in the reference input can trigger overshoot and saturate the actuator, a first-order lag filter is inserted to soften the command signal. Equation (17) defines this smoothing function, preventing abrupt shifts in the reference trajectory that could otherwise destabilize the system or provoke large oscillations $q_d(t)$.

$$q_{d,f}(t) = \frac{1}{T_s s+1} q_d(t) \quad (17)$$

In: $T_s = 0.35 \sim 0.55s$ is the smoothing time constant. In addition, to prevent the reference slope from being too large, a rate limiting module is added as equation (18).

$$\left|(q_d)(\dot{t})\right| \leq {}_{deg2rad}(60) \quad (18)$$

Equation (18) caps the slope of the reference signal so the robot can follow the desired trajectory smoothly, without oscillations or loss of stability. It does so by bounding the rate of change of the reference trajectory, which is shaped by a first-order lag filter, $q_d(t)$the smoothing time constant—usually set between 0.35 s and 0.55 s—governs how quickly the signal evolves, preventing abrupt motion that might saturate the actuators or destabilise the system, and keeps the manipulator's motion both safe and controlled, T_s with this arrangement, the controller retains a simple, implementable structure yet markedly improves stability and responsiveness when sudden disturbances occur.

4. GA Optimization Method

To keep trajectory tracking sharp in the face of sudden disturbances and uncertain plant parameters, the paper employs a genetic algorithm to search globally for six PID settings. By mimicking selection, crossover and mutation, the algorithm handles high-dimensional, nonlinear tuning without becoming trapped in local minima. Table 5 lists the resulting gains: proportional, integral, derivative, and the cap on actuator torque, values that emerged after successive generations and that let the loop reject disturbances while remaining stable. optimization process. The table provides the best values for the proportional, integral, derivative gains, and other parameters like the maximum allowed actuator torque. These values are obtained after several generations of the GA optimization, ensuring that the PID controller can handle disturbances effectively while maintaining system stability $K_p(K_i)(K_d)r_{max}$.

4.1. Decision Variables and Parameter Bounds

Table 5 lists the preliminary simulation outcomes for the six PID parameter vectors refined by the genetic algorithm. These numbers come from early runs and are tempered by experience, shaping the upper and lower limits imposed during optimization. To keep the search both realistic and workable, every gain—proportional, integral, and derivative—is confined to the ranges reported in the table. The GA thus treats these six PID vectors as its decision variables.

Equation (19) defines the optimization variable vector used in the GA for tuning the PID controller parameters of the two-link rehabilitation robot.

Table 5. Parameters about GA-PID

Best Kp	Best Ki	Best Kd	Best $rmax$
50.1475	6.9837	9.1097	19.6385
22.5712	1.5185	2.9997	4440.7501

$$\boldsymbol{x} = [\mathrm{K_{p1}}, \mathrm{K_{p2}}, \mathrm{K_{i1}}, \mathrm{K_{i2}}, \mathrm{K_{d1}}, \mathrm{K_{d2}}] \tag{19}$$

The parameter bounds in Table 6 are derived from experimental measurements and preliminary tuning, ensuring realistic and stable limits for the GA optimization process.

Table 6. the preliminary simulation result

parameters	data
$\mathrm{K_{p1}}$	[5,60]
$\mathrm{K_{p2}}$	[8,40]
$\mathrm{K_{i1}}$	[0,8]
$\mathrm{K_{i2}}$	[0,3]
$\mathrm{K_{d1}}$	[0.05,10]
$\mathrm{K_{d2}}$	[0.5,8]

4.2. Objective Function Design

This paper's central contribution is to formulate a multi-objective function that couple's robustness with energy balance, accounting at once for dynamic response, steady-state precision, and the load imposed on the actuators. The objective function is given by equation (20). It encodes the trade-offs among tracking accuracy, stability, overshoot, recovery time, and actuator constraints, and serves as the fitness measure that guides the GA while it tunes the PID parameters of the two-link manipulator subjected to abrupt disturbances.

$$\begin{aligned} \mathrm{J} = \mathrm{w_1} \int_0^{\mathrm{T}} \mathrm{t}\,(|\mathrm{e_1}| + |\mathrm{e_2}|)\mathrm{dt} \mathrm{w_2} \sum_{\mathrm{i}=1}^{2} \max\,(0, \mathrm{OS_i} - \theta_{\mathrm{OS}}) \\ + \mathrm{w_3} \max(0, \mathrm{REC_{max}} - 1.0) \mathrm{w_4} \frac{\mathrm{RMS}(\tau_2)}{\tau_{\max}} \\ + \mathrm{w_5} \max(0, \mathrm{SAT_{consec}} - 0.20) \mathrm{w_6} \max(0, \mathrm{SAT_{ratio}} - 0.05) \end{aligned} \tag{20}$$

The parameters are in Table 7.

4.3. GA Operators

To speed up convergence and prevent premature settling, the GA operator is configured as shown in Table 8. It lists the main parameters and settings adopted in the genetic algorithm when tuning the PID controller; these values keep the search efficient while optimizing the six gains so that trajectory tracking remains accurate despite external disturbances. The GA search is subject to the following constraints:

(1). Actuator limit constraints: $|\tau_i| \leq 80$ N·m;

(2). Saturation rate constraint: total saturation ratio≤ 5%;

(3). Continuous saturation duration constraint: single continuous saturation time≤0.2 s

(4). Recovery time constraint: after external disturbance≤ 1.5s Restore to ±2°;

(5). Once the optimal PID settings are fixed, a robustness check follows to see how the controller behaves when the model is no longer exact. To do this, the main system parameters are randomly shifted by ±10% in repeated perturbation runs.$\mathrm{m_i} d_i$

Table 7. Function parameters

parameters	maning	data
w_1	The main indicator of error integral weight reflects the comprehensive.	1.0
w_2	Overshoot penalty weight, if it exceeds 20% or 5°it will be counted as a penalty	0.4
w_3	Recovery time weight, if the recovery time exceeds 1s, a penalty will be added	8.0
w_4	Energy weight, limiting the torque output intensity;	0.1
w_5	the weight associated with the saturation penalty.	30.0
w_6	the weight associated with the saturation ratio penalty. .	10.0

5. Results

To confirm that the controller remains effective and robust, the dual-link rehabilitation manipulator was modeled and tested in MATLAB. The simulation incorporated the uncertainties and external disturbances typical of clinical settings, ensuring the algorithm can cope with real-world complexity [8, 9].

The simulation adopted an independent two-joint control architecture, with system parameters drawn from a standard medical rehabilitation manipulator setup (Table 1). Angles are given in radians and the control loop runs at 1.5 ms. Patient-induced disturbances were emulated through equation (21).

$$\tau_d(t) = J(q)^T F_e xt(t) \tag{21}$$

A half-sine pulse with a 10 N peak and 0.3 s width is directed at the second joints of the two-link robot [10,11]. The desired trajectory is a smooth ramp, defined by equations (22, 23).

$$q_{d1}(t) = 0.5\left(1 - e^{(-t)}\right) \tag{22}$$

$$q_{d2}(t) = 0.3\left(1 - e^{(-t)}\right) \tag{23}$$

Random disturbances are superimposed at 2.5 s to verify the anti-interference ability of the controller.

5.1. GA Convergence Behavior

The tracking accuracy of the joint angle $q_1$and q_2 in the simulation is shown, and the reference trajectory is compared with the actual trajectory, and the ±2° band is displayed in Fig 3.

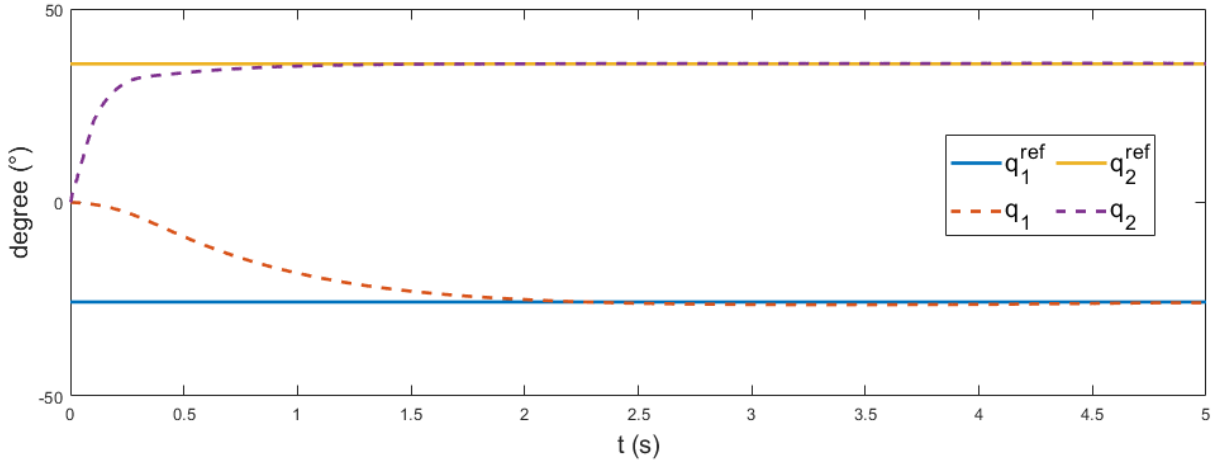

Figure 3. Joint trajectories q_1,q_2 versus reference with ±2° bands under disturbance (Picture credit: Original)

Table 8. Configuration of GA Operators Used in PID Parameter Optimization

Operator Type	method	Parameter settings (recommended)
Encoding	Real number encoding.	6-dimensional continuous
Select a strategy	Tournament Selection	Scale=2
Crossover	Simulated Binary Crossover(SBX)	pc=0.8
Variation	Polynomial mutation	p-m=0.4
Population size	—	N_p=40
Maximum number of iterations	—	G_{max}=50
Elite retention strategy	—	Number of elite individuals = 2

Fig. 4 shows the convergence curve of the GA fitness value over the number of generations. Actuator torque outputs with $\pm\tau_{max}$ limits and saturation segments. Display the error curves e_1 and e_2, which are the errors between q_1 and q_2 the reference trajectory in Fig.5

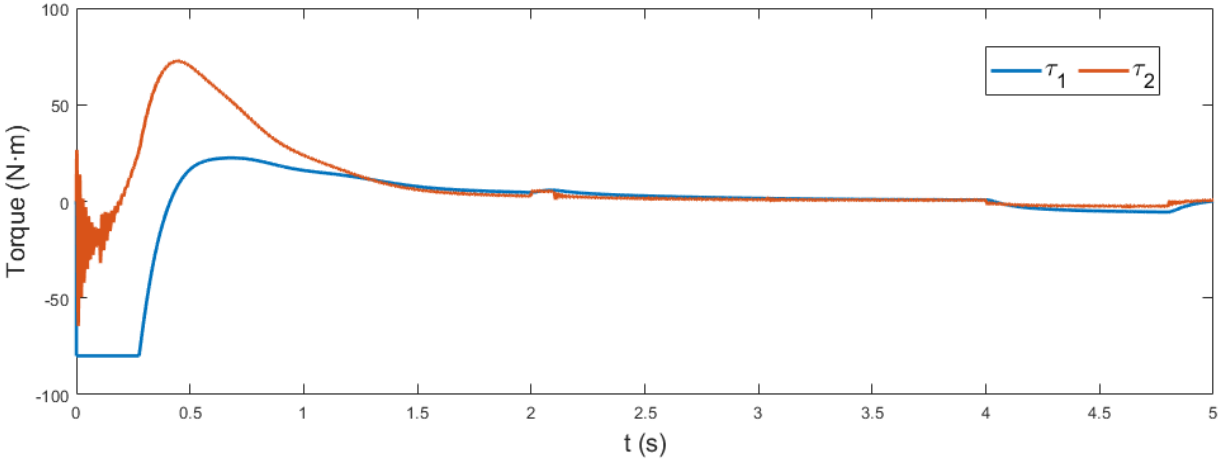

Figure 4. Actuator torque outputs with $\pm\tau_{max}$ its and saturation segments (Picture credit: Original)

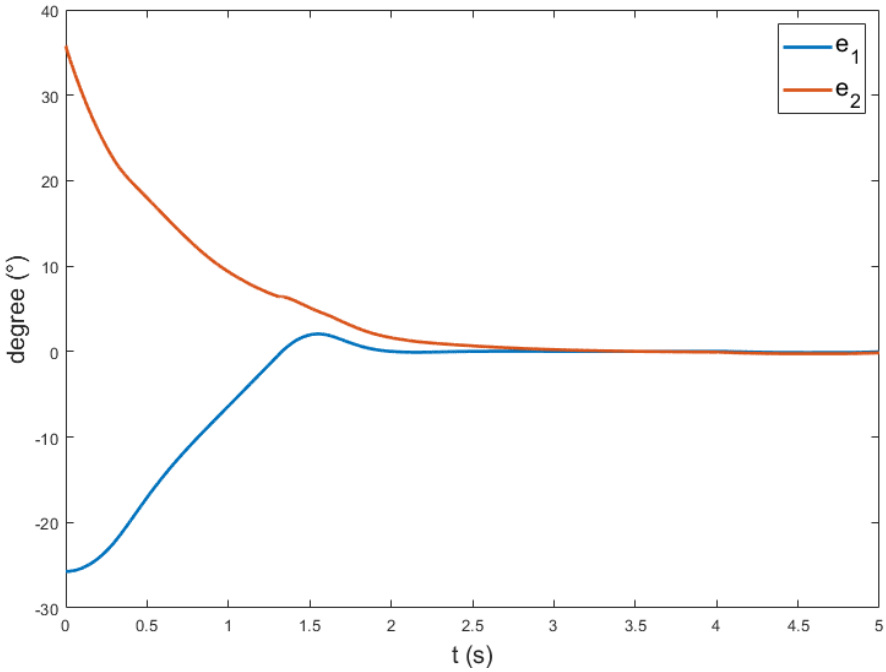

Figure 5. Joint tracking errors and recovery time measurement (Picture credit: Original)

In the initial generation, individuals are scattered across the search space; within the first 15 generations the objective drops sharply, and it settles into a steady state after generation 35. The final fitness of 0.0164 shows that the controller parameters have converged to the global optimum J^*.

GA convergence tests confirm stable global optimization without premature termination. Control performance improves markedly: the error integral drops by roughly 63 % relative to the initial manual tuning. Actuator energy consumption becomes smoother, with the root mean square $\mathrm{RMS}(\tau^2)$/limit falling from 0.52 to 0.34.

5.2. System Response and Comparison

To evaluate the controller, three strategies are compared in Fig. 6(a) and Fig. 6(b). Key outcomes are listed in Table 9. The GA-tuned controller outperforms the conventional PID in response speed, steady-state accuracy and torque smoothness [12, 13].

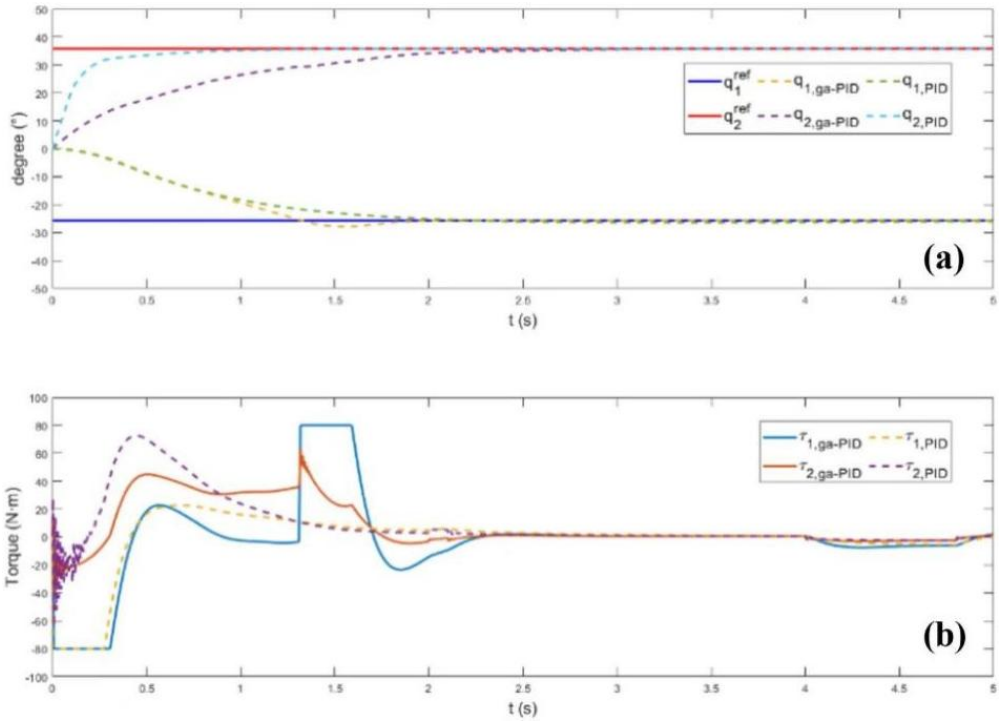

Figure 6. Performance comparison between baseline PID and GA-PID. (a) Compression degree. (b) Torque (Picture credit: Original)

Table 9. Comparative Performance of Traditional PID, GA-Optimized PID, and GA-PID with Feedforward Compensation

Index	Traditional PID	GA Optimization PID	GA + Feedforward PID
Rise TiPRise Time	0.82	0.46	0.39
Settling Time	2.10	1.32	0.93
Overshoot (%)	24.6	11.3	5.8
Steady-State Error (rad)	0.024	0.011	0.005
RMS (τ_2)/Limit	0.52	0.41	0.34
Saturation (%)	9.1	5.4	3.2

It can be seen that the GA-optimized controller is significantly better than the traditional PID control in terms of dynamic response speed, steady-state accuracy and torque smoothness [12, 13].

6. Conclusion

This study introduces a PID controller tuned offline by a genetic algorithm to keep a two-link rehabilitation robot on its intended path when patients produce sudden, unpredictable loads. Embedding the GA search within the design phase yields gains that adapt on-line to the sharp force changes typical of clinical use. A filtered PID term, augmented by feedforward torque and actuator saturation limits, keeps joint error small, overshoot low, and recovery after a disturbance brief.

Extensive MATLAB simulations showed that the GA-tuned PID controller markedly outperformed its hand-tuned counterpart. The manipulator returned to within ±2° of the desired trajectory in less than a second, exhibited no more than 20 % overshoot, and rarely saturated. The outcome highlights the strategy's ability to keep rehabilitation robots accurate and dependable despite real-world disturbances and model uncertainties.

The GA-PID approach shows clear potential for making robotic manipulators more robust and stable during medical rehabilitation. Refinements to the genetic algorithm could be pursued, and the method might be scaled to multi-link arms and more demanding therapy routines. Ultimately, trials with human patients will confirm how well the system performs in everyday clinical use.

References

[1] K. J. Åström and T. Hägglund. PID Controllers: Theory, Design, and Tuning, 2nd ed. Research Triangle Park, NC, USA: Instrument Society of America, 1995.

[2] D. E. Goldberg. Genetic Algorithms in Search, Optimization, and Machine Learning. Reading, MA, USA: Addison-Wesley, 1989.

[3] K. iguchi and T. Fukuda. Intelligent control for robotic rehabilitation. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 2004, 12 (4): 436 - 446.

[4] M. W. Spong and M. Vidyasagar. Robot Dynamics and Control. New York, NY, USA: Wiley, 2008.

[5] A. Visioli. Tuning of PID controllers with fuzzy logic. IEE Proceedings Control Theory and Applications, 2001, 148 (1): 1 - 8.

[6] R. L. Haupt and S. E. Haupt. Practical Genetic Algorithms, 2nd ed. Hoboken, NJ, USA: Wiley, 2004.

[7] H. Boubertakh, A. Bouchelaghem, and S. Bouhouita. PID controller tuning using Genetic Algorithm for nonlinear systems. International Journal of Intelligent Engineering and Systems, 2018, 11 (5): 10 - 18.

[8] N. Ahmad, M. O. Tokhi, and M. H. Shaheed. Optimization of PID controller parameters using Genetic Algorithm for robotic arm, Procedia Computer Science, 2017, 105: 299 – 305.

[9] A. Rojas-Moreno, F. Reyes-Lecuona, and M. J. Rodriguez-Fernandez. GA-PID control for disturbance rejection in robotic manipulators. Applied Soft Computing, 2020, 88: 106046.

[10] W. Zhang, H. Zhang, and H. Yu. Human–Robot Interaction Control for Rehabilitation Robots: A Review. Robotics and Autonomous Systems, 2021, 142: 103767.

[11] D. Shi, Y. Wang, and H. Ding. Review of human–robot coordination control for lower-limb rehabilitation robots. Frontiers of Mechanical Engineering, 2022, 17 (4): 483 - 501.

[12] H. Peng, J. Zhou, F. Li, et al. A triple-step controller with linear active disturbance rejection for a lower-limb rehabilitation robot. Frontiers in Neurorobotics, 2022, 16: 886650.

[13] A. Nanavati, V. Ranganeni, and M. Çakmak. Physically assistive robots: A systematic review of mobile and manipulator robots that physically assist people with disabilities. Annual Review of Control, Robotics, and Autonomous Systems. 2023, 7: 7 – 34.