

Hybrid Robot Trajectory Planning Using FC-SSA-PID and DWA-Enhanced BIT* Algorithms

Jianjing Zhang, Xiaoru Xing*, Yueqiang Hu, Xia Zhao, Jing Li

Department of Basic Courses, Hebei Institute of Mechanical and Electrical Technology, Xingtai 054000, China

E-mail: wangyichenzhang31@163.com, xingxiaoru586@163.com, 13663199870@163.com, 13613199997@163.com, hbjd20250226@163.com

*Corresponding author

Keywords: proportional-integral-derivative control, batch informed trees, trajectory planning, sparrow search algorithm, fuzzy control, dynamic window approach

Received: March 28, 2025

Planning the movement path of a robot is crucial to ensure it reaches the target area smoothly. Existing methods tend to fall into local optima, have low accuracy in route calculation, and fail to effectively avoid obstacles. To address these issues, this study introduces the Sparrow Search Algorithm and Fuzzy Control, as well as the Dynamic Window Approach, to optimize Proportional-Integral-Derivative control and Batch Informed Trees, respectively. Based on these two optimization algorithms, a robot trajectory planning model is proposed, and its feasibility and reliability are demonstrated through comparative experiments. In standardized 50m×50m grid environments with 20%-30% obstacle density and dynamic obstacles, 30 independent simulation runs were conducted. Comparative analysis with RRT, Ant Colony Optimization (ACO), and Genetic Algorithm (GA) demonstrates that the proposed model achieves a success rate of 95.5%, a high accuracy rate of 99.4%, and a low accuracy error rate of 0.0011%. The locally optimal route length planned by the model is 12.6m, while the global average optimal route length is reduced to 21.2m, significantly outperforming the comparison models. These findings demonstrate that the proposed model has strong trajectory planning capabilities, minimal error, and shorter routes, enabling the robot to respond correctly to external environments in a timely manner and complete tasks effectively even in complex dynamic conditions.*

Povzetek: Predstavljen je hibridni model načrtovanja poti robotov, ki združuje FC-SSA-PID za prilagodljivo krmiljenje in DWA-izboljšani BIT za globalno ter lokalno načrtovanje. Sistem učinkovito premaguje lokalne optime, zmanjšuje napake, hitro se izogiba oviram.*

1 Introduction

Since the 21st century, robots have gradually entered public life, and their technology has developed rapidly. However, their performance remains less autonomous and adaptive compared to human capabilities [1]. The demand for efficient, safe, and precise robotic movement paths necessitates advanced trajectory planning technology. It serves as the core system for controlling robot motion and is a crucial foundation for enabling robots to complete various tasks. Therefore, optimizing trajectory planning has been a key research focus [2]. Currently, methods such as Genetic Algorithm (GA), Recurrent Neural Network (RNN), and Graph Search Algorithm (GSA) have been applied to trajectory planning [3]. However, these methods suffer from problems such as falling into local optima, poor real-time performance, and weak adaptability to dynamic environments. Therefore, trajectory planning methods requiring enhanced accuracy and stability are essential. Proportional-Integral-Derivative (PID) control adjusts signal parameters through proportional, integral, and derivative control algorithms to achieve precise system output control. It is a

mature and stable technique with strong adaptability. Meanwhile, Batch Informed Trees (BIT*) combines the advantages of sampling-based and search-based planning algorithms, allowing it to filter out unimportant information and find optimal routes in complex environments [4-5]. However, traditional PID parameter tuning is time-consuming and lacks precision, while BIT* faces challenges such as computational complexity and low solution quality in high-dimensional problems. To address these limitations, this study introduces the Sparrow Search Algorithm (SSA) and Fuzzy Control (FC), as well as the Dynamic Window Approach (DWA), to optimize PID and BIT*, respectively. Based on these optimizations, a new robot trajectory planning model is proposed. This model is expected to improve robot flexibility and enable intelligent task execution. We aim to investigate whether combining FC-SSA-PID and DWA-BIT* improves trajectory planning performance in dynamic environments compared to other similar technologies.

2 Related work

PID control has been widely applied in various fields requiring precise control systems, including robotics, aircraft, and 3D printing. Therefore, many researchers have conducted in-depth studies on PID control. For

example, to address the unpredictable environmental challenges faced by unmanned surface vehicles, Meng J et al. proposed a Gaussian process motion trajectory planning method based on PID. This method introduced an interpolation strategy to increase path selection

Table 1: Summary and comparison table of related work.

Ref.	Method Used	Application Domain	Accuracy (%)	Path efficiency (%)	Runtime complexity
[6]	Gaussian Process + PID + Interpolation	Unmanned Surface Vehicles	85	82	High
[7]	Fractional-order PID Control	Wind Power Simulation System	90.5	87	Medium
[8]	Improved BIT* + Spatial Transform/Mask	Concrete Color Difference Detection	92.1	85	Medium-High
[9]	Self-deforming BIT* Variant	High-arrival-rate Data Streams	95.8	86	Low-Medium
[10]	Improved BIT* + Linear Interpolation	Spinal Puncture 3D Reconstruction	94.3	88	Medium
[11]	Local Chaotic Particle Swarm Optimization	Robotic Arm Trajectory	89.7	85	High
[12]	Dynamic Trajectory Discovery	Multi-Cable-Driven Robots	91.2	78	High
[13]	Hybrid Optimization Algorithm	Industrial Robot Manipulators	87.5	80	Very High
[14]	Model Predictive Control	Robot Motion in Dynamic Environments	88	75	High
[15]	Model-Free Hierarchical Decoupling	Collision Avoidance	90	70	Medium-High

diversity, with experimental results demonstrating effective mitigation of unknown environmental impacts [6]. Gasmi H et al. developed a dual-feedback wind power simulation system based on PID to solve the significant vibration issues in wind power operations. This system utilized fractional-order proportional-integral control to regulate grid power and ensure maximum power operation. The results demonstrated that it reduced grid oscillations while maintaining robust performance [7]. At the same time, BIT* has also been widely applied, particularly in scenarios requiring large-scale data processing. For instance, to address the instability of using ordinary drones for concrete sampling in construction, Yao G et al. proposed a concrete color difference detection model based on an improved BIT*. This model used spatial transformation and mask quantization segmentation operations to achieve block-based color difference encoding, and experimental results showed a detection accuracy of 92.10% [8]. Wang H et al. introduced a self-deforming bitmap algorithm based on BIT* to address the issue of existing solutions being unable to perform online operations on high-arrival-rate data streams. This algorithm automatically adapted the sampling probability for data streams of different sizes and allowed bitmap deformation. Experiments proved that it could efficiently and accurately process high-arrival-rate data streams [9]. In the field of spinal puncture surgery, Zhang W et al. proposed an improved BIT*-based three-dimensional reconstruction system to solve the time-consuming and complex process of determining puncture points and paths. By using a linear interpolation algorithm for 3D reconstruction, the system enabled automatic image scanning and acquisition to quickly locate the optimal puncture point and path [10].

Trajectory planning technology has reached a relatively mature stage in both theoretical research and

practical applications, and scholars worldwide have applied various optimization methods in real-world scenarios. For example, to optimize the movement time and trajectory of robotic arms, Du Y et al. proposed a local chaotic particle swarm optimization algorithm based on interpolation functions. This algorithm conducted simulation experiments in joint space, and simulation results demonstrated that it effectively reduced the oscillation amplitude and movement time of the robotic arm, ensuring operational stability [11]. To address the issue of redundant trajectories in multi-cable-driven robots reaching targets without entanglement, Cao M et al. proposed a dynamic trajectory discovery method that evaluates the safety and feasibility of potential paths while reducing inter-robot entanglement. The results demonstrated that this method effectively generated tangle-free dynamic trajectories [12]. To improve robot task execution efficiency, Singh G et al. introduced a robot kinematics analysis and trajectory planning method based on a hybrid optimization algorithm. By performing forward and inverse calculations on 18 different algorithms, the optimal solution was obtained, and experiments confirmed that this method significantly enhanced robot performance and found the best path [13]. Wang C et al. proposed a model predictive control method to address the instability of robot motion in dynamic environments. This method automatically tracked robot movement paths and adaptively computed optimal routes. The results showed that it effectively reduced environmental uncertainty while maintaining the ability to replan global routes [14]. To solve the problem of robots avoiding collisions while executing tasks, Wang S et al. proposed a model-free hierarchical decoupling optimization algorithm. This algorithm decomposed tasks into two sub-tasks to reduce task complexity, and experiments demonstrated that it improved robot

adaptability to the environment, effectively preventing collisions [15]. The summary and comparison Table of relevant work is shown in Table 1.

In summary, although research on trajectory planning has achieved significant progress, existing methods still have limitations. Under specific conditions, they may

suffer from low computational efficiency, suboptimal route planning, and slow obstacle response. PID control can reduce system deviation, eliminate steady-state errors, and suppress oscillations, while BIT* ensures efficiency and stability in processing dynamic datasets. Therefore, this study combines PID and BIT* to

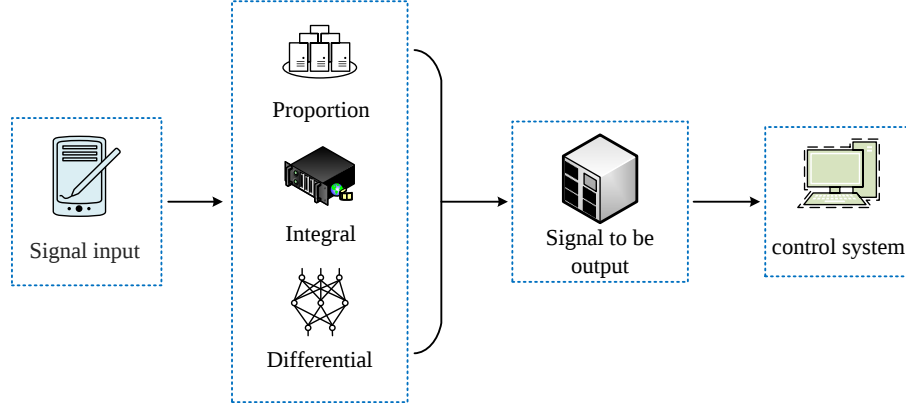


Figure 1: Schematic diagram of PID control principle.

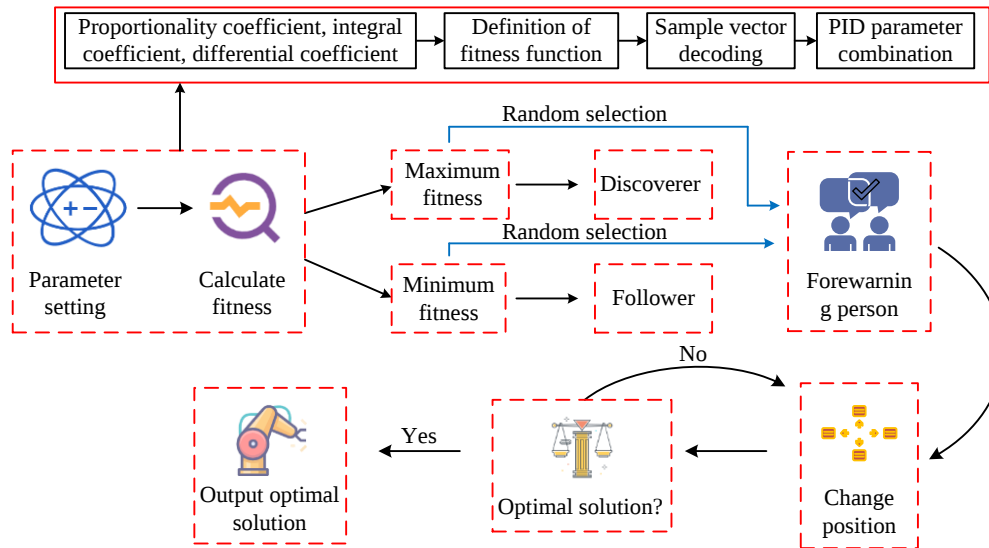


Figure 2: Schematic diagram of the optimization process of SSA for PID control.

develop a PID-BIT* hybrid algorithm, aiming to ensure that robots can complete tasks accurately and efficiently under complex environmental conditions.

3 Optimization of robot motion trajectory based on FC-SSA-PID and DWA-BIT*

3.1 Architecture optimization of PID control

PID control consists of three components: proportional, integral, and derivative units. The proportional unit increases system response speed, the integral unit reduces system errors, and the derivative unit enhances disturbance resistance. By adjusting system parameters through these three algorithms, precise robot control is

achieved [16]. The output diagram of the PID control system is shown in Figure 1.

As shown in Figure 1, when a signal enters the PID controller, it is processed by the three units. The proportional unit calculates the error value and outputs the proportional coefficient as a signal. The integral module accumulates the error, multiplies it by the integral coefficient, and outputs the result as a new signal. The derivative module evaluates the rate of error change, multiplies it by the derivative coefficient, and outputs the derivative as a signal component. The calculation process is shown in Equation (1).

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt} \quad (1)$$

In Equation (1), K_p represents the proportional coefficient, K_i is the integral coefficient, and K_d is the derivative coefficient. Since traditional PID control cannot fully ensure smooth and stable robot movement, the study

introduces SSA to optimize PID and adjust the PID parameters to obtain a more comprehensive control parameter combination. The optimization process is shown in Figure 2.

As shown in Figure 2, SSA is first initialized with parameters, including the maximum number of iterations

and the numbers of discoverers, followers, and warners. Sample fitness values are subsequently evaluated, sorted, and classified into discoverers and followers, with followers updating their positions. Some samples are randomly selected as warners and also update their

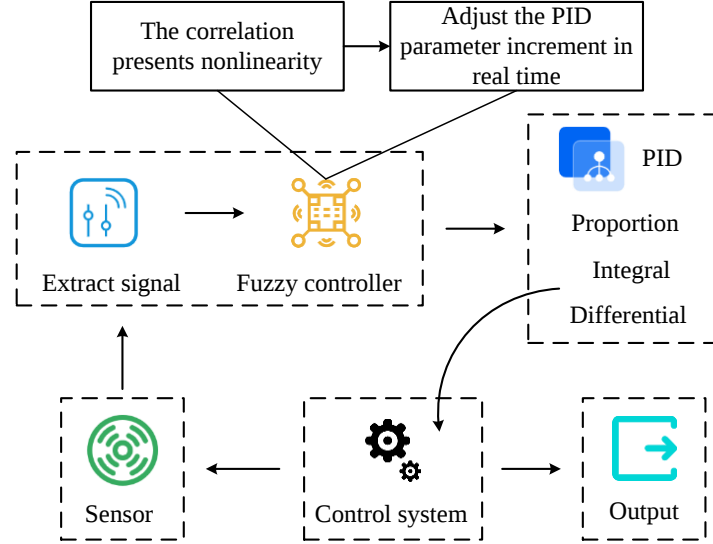


Figure 3: Schematic diagram of the optimization process of FC for SSA-PID.

positions. Finally, the system determines whether the updated positions are optimal under current conditions. If not, the position update process repeats until the optimal solution is found [17]. The PID parameters optimized by SSA are proportional coefficients, integral coefficients, and differential coefficients, and the sample position vectors correspond to the candidate solutions of the three coefficients. During the iterative process, the fitness function is defined as the root mean square value of the trajectory tracking error. After each position update, the sample vector is decoded into a combination of PID parameters and input into the control system for simulation and fitness calculation. The process is represented in Equation (2).

$$X_{i,j}^{t+1} = \begin{cases} X_{i,j}^t \cdot \exp\left(\frac{-i}{\alpha \cdot iter_{\max}}\right), & \text{if } R_2 < ST \\ X_{i,j}^t + Q \cdot L, & \text{if } R_2 > ST \end{cases} \quad (2)$$

In Equation (2), $X_{i,j}^t$ represents the position of sample i in space j , t is the iteration count, $iter_{\max}$ is the maximum iteration count, α is the number of randomly selected samples in the range (0,1), Q is the set of samples within this range, L is defined as Matrix 1, R_2 is the number of warning samples, and ST is the warning threshold. The expression for follower samples is given in Equation (3).

$$X_{i,j}^{t+1} = \begin{cases} Q \cdot \exp\left(\frac{X_{morse}^t - X_{ij}^t}{\alpha \cdot iter_{\max}}\right) \text{ if } i > \frac{n}{2} \\ X_{best}^{t+1} + |X_{i,j}^{t+1} - X_{best}^{t+1}| \cdot A^+ \cdot L \text{ if } i \leq \frac{n}{2} \end{cases} \quad (3)$$

In Equation (3), X_{morse}^t represents the worst route, $X_{i,j}^{t+1}$ is the best route, A is a matrix of 1 and -1, $i > \frac{n}{2}$ represents the follower sample's position before updating, and $i \leq \frac{n}{2}$ represents the real-time updated position. In addition to these two sample types, warners are randomly selected, as expressed in Equation (4).

$$X_{i,j}^{t+1} = \begin{cases} X_{best}^t + \beta \cdot |X_{i,j}^t - X_{best}^t| \text{ if } f_i > f_g \\ X_{i,j}^t + k \cdot \frac{|X_{i,j}^t - X_{worst}^t|}{(f_t - f_w)} \text{ if } f_i = f_g \end{cases} \quad (4)$$

In Equation (4), β and k are parameters controlling position updates, f_i represents local sample fitness, and f_g represents the global optimal fitness. By optimizing PID parameters using SSA, the optimal solution is quickly identified and transmitted to the control system, allowing real-time robot motion updates. SSA initialization parameters include: maximum iterations (200), the number of discoverers of 20 (accounting for 40% of the total sample size), the number of followers of 30 (60%), and the proportion of early warning agents of 0.1. The position update coefficient is 0.8 and the safety threshold is 0.6. The convergence condition is defined as the fitness change of 10 consecutive iterations being less than 1×10^{-5} . Parameter sensitivity analysis indicates that the convergence speed is the fastest when the proportion of discoverers is between 30% and 50%. A safety threshold of 0.6 can balance the capabilities of global exploration and local development. When the position update coefficient is set at 0.7-0.9, the fitness fluctuation is less

than 3%. The search space for PID parameters optimized by SSA is strictly constrained: the value range of the proportionality coefficient is [0, 50] (step size accuracy 0.01), the value range of the integral coefficient is [0, 5] (step size accuracy 0.001), and the value range of the differential coefficient is [0, 1] (step size accuracy 0.001). All parameter sampling during the iteration process is

performed within this space. However, SSA-PID has limitations in controlling nonlinear systems and handling multi-threaded input-output systems. Since trajectory planning involves not only determining start and end positions but also real-time motion control, FC is introduced to enhance SSA-PID and address its

Table 2: Fuzzy control rule table.

ec \ e	NB	NM	ZO	PM	PB
NB	PB	PB	PM	PM	ZO
NM	PB	PB	PM	ZO	ZO
ZO	PM	PM	ZO	NM	NM
PM	ZO	ZO	NM	NB	NB
PB	ZO	NM	NM	NB	NB

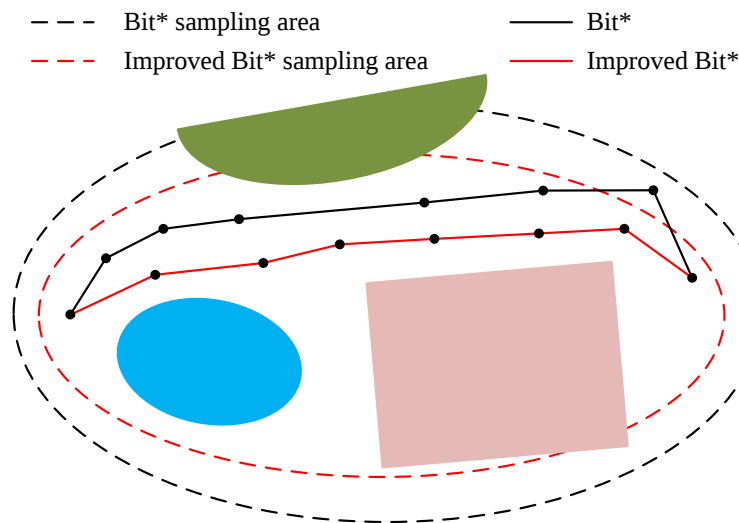


Figure 4: Schematic diagram of improved BIT*.

shortcomings in handling nonlinear systems. The optimization process is shown in Figure 3 [18].

In Figure 3, the motion signal first enters the fuzzy controller, which adjusts parameters using fuzzy logic to obtain the optimal solution. The adjusted signal is then passed to the PID controller for further parameter tuning before being output to the robot's control system. The sensor feedback the signal to the fuzzy PID controller, which compares the output signal with the original motion signal and adapts the parameters for optimal output. The PID parameters adjusted by the fuzzy controller are applied to the PID controller in real time. In order to describe the main response characteristics of the control signal under the influence of dynamic parameter adjustment, a simplified model with first-order delay was adopted for illustration in the study. This model aims to capture the main dynamic effects introduced by parameter changes rather than precisely describe the nonlinear process of fuzzy reasoning itself. The adjustment process is shown in Equation (5).

$$G(s) = \frac{k}{(Ts+1)} e^{-Ls} \quad (5)$$

In Equation (5), k represents the ratio of parameter variation, e^{-Ls} is the parameter adjustment delay, and L is the specific delay time. This model represents the

primary dynamic characteristics in the controller output response observed under FC adjustment. FC solves the nonlinear control limitation of SSA-PID through the dynamic rule base. When the correlation between the system error and the error change rate shows nonlinearity, FC adjusts the PID parameter increment in real time based on the rule base in Table 2 instead of relying on fixed parameters. Multi-threaded I/O processing adopts a serial signal processing architecture. The input signal is executed in three sequential steps: fuzzification, rule reasoning, and defuzzification to avoid multi-threaded conflicts. Specifically, the fuzzy controller employs error (e) and error change rate (ec) as inputs, with the output being PID parameter adjustments. The fuzzy sets for both input/output variables are defined as: {Negative Big (NB), Negative Medium (NM), Zero (ZO), Positive Medium (PM), Positive Big (PB)}. Triangular membership functions are adopted for fuzzification. The rule base is designed as Table 2.

For defuzzification, the centroid method is applied. This strategy computes the geometric center of the aggregated output fuzzy set to derive a precise crisp value. By weighting all activated rules proportionally to their membership strengths, it achieves smooth and continuous parameter adjustments while eliminating output uncertainty. In the setting of the domain range of the

input/output variables of the fuzzy controller, the domain of the error is $[-3,3]$, the domain of the error change rate is $[-1,1]$, and the domains of the PID parameter increments are all $[-0.5,0.5]$. The system adopts triangular membership functions, the vertex coordinates defined as NB in $\{3.0, 2.5, 2.0\}$, NM in $\{2.5, 1.5, 0.5\}$, ZO in $\{1.0,$

$0.0, 1.0\}$, PM in $\{0.5, 1.5, 2.5\}$, PB in $\{2.0, 2.5, 3.0\}$. The output variables adopt the same structure and scale according to the domain ratio. Pre-adjusting parameters using FC reduces PID adjustment time and frequency, ensuring the robot responds to environmental changes in the shortest time possible.

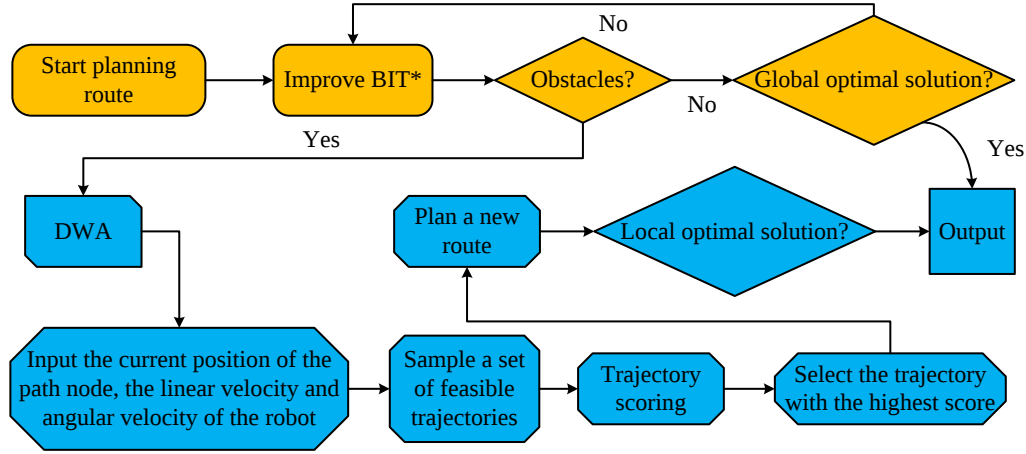


Figure 5: Schematic diagram of the optimization process of DWA for improved BIT*.

3.2 Construction of robot trajectory planning model based on DWA-BIT* and FC-SSA-PID*

Although FC-SSA-PID effectively adjusts parameters to control robot actions, it does not provide trajectory planning. Therefore, an additional algorithm is needed to enhance robot task execution. BIT* solves global path planning problems in large datasets while filtering out irrelevant information. It also has strong disturbance resistance and good dynamic adaptability [19]. In order to improve the adaptability of the algorithm in sample datasets of different scales and enhance the operational efficiency, an improved BIT* algorithm was proposed in the research. It combines the advantages of Rapidly-exploring Random Trees (RRT) and Heuristic Search (HS) [20]. The improvement process is shown in Figure 4.

As shown in Figure 4, traditional BIT* navigates around obstacles but takes longer and results in a longer overall path with larger turning angles. The core of improving BIT* lies in introducing an adaptive sampling strategy guided by path nodes. Traditional BIT* is prone to generating invalid nodes in uniform random sampling, resulting in path redundancy and low efficiency. After improvement, the algorithm first dynamically demarcates an efficient sampling bounding box based on the current environmental information and historical path nodes. The calculation process is shown in Equation (6).

$$\text{sample_radius}() := \sqrt{\frac{(x_R - x_L - 2\varepsilon) \times (y_R - y_L - 2\varepsilon)}{m}} \quad (6)$$

In Equation (6), x_R , x_L , y_R , and y_L represent the horizontal and vertical coordinates of the boundaries in a two-dimensional space, ε is the boundary value, and m represents the number of equidistant down-sampling points. After the defined bounding box constrains the sampling range, the generation of sampling points is no

longer uniform and random. The generation of new sample points follows a probability distribution model centered on the mean values of these path nodes and with a specific exploration range, as shown in Equation (7).

$$H(X) = 1 - \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(X-u)^T \Sigma^{-1}(X-u)} \quad (7)$$

In Equation (7), $H(X)$ represents the generated sample, d represents the dimension, and u represents the mean vector of the coordinates of the historically valid path nodes. The core of the exponential part of Equation (7) is the negative exponential term of the probability density function of the multivariate Gaussian distribution. It calculates the "Markov distance" of the mean distance between points, which takes into account the correlation between variables. Before the improvement, the samples were uniformly distributed throughout the configuration space to randomly generate sample points. This sampling method leads the search to fall into local optima or waste of computing resources. The improved new sample points are generated by taking the center of the historically effective path as the reference point and controlling their distribution range and direction based on the path characteristics and environmental information. This makes the sampling points highly biased towards better path areas that has been explored and its natural extension direction. This guidance mechanism improves search efficiency and path quality. To further enhance the local optimal path planning capability of the improved BIT*, DWA is introduced. By sampling the robot's current position, velocity, and state, DWA calculates motion trajectories over a time interval and selects the optimal solution. The DWA-optimized BIT* process is shown in Figure 5.

As shown in Figure 5, when the improved BIT* algorithm detects obstacles in the global path at the turning

points, the DWA is activated for local re-planning. The BIT* takes the current position of the path node, the linear velocity and angular velocity of the robot as the input states of the DWA. DWA samples a set of feasible trajectories in the state space based on the dynamic constraints of the robot. Subsequently forming a new

global path. The DWA evaluation function is given in Equation (8).

$$G(v, \omega) = \alpha \text{Head}(v, \omega) + \beta \text{Vel}(v, \omega) + \gamma \text{Dist}(v, \omega) \quad (8)$$

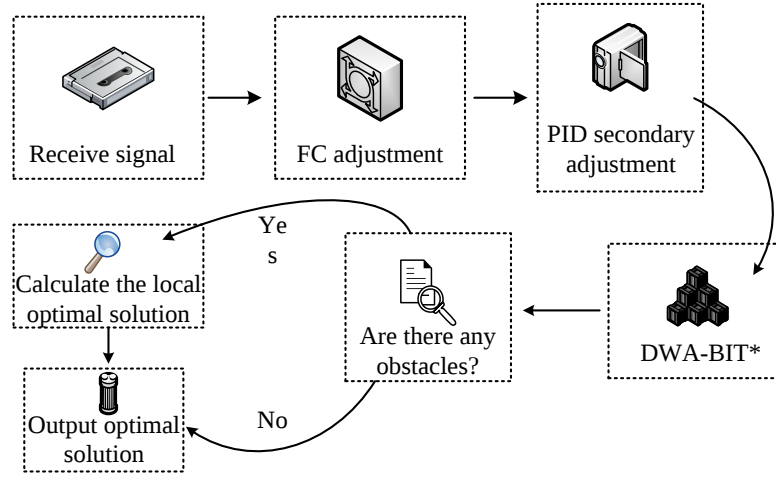


Figure 6: Trajectory planning flow chart based on FC-SSA-PID and DWA-BIT*.

In Equation (8), $\text{Head}(v, \omega)$ represents the azimuth function, $\text{Dist}(v, \omega)$ is the distance evaluation function, and $\text{Vel}(v, \omega)$ represents the velocity function. These three functions together determine how the robot moves toward the target at maximum speed and along the optimal path. The optimal trajectory function is expressed in Equation (9).

$$\text{Point}(v, \omega) = \text{Min}[(x_t - x_g)^2 + (y_t - y_g)^2] \quad (9)$$

In Equation (9), $\text{Point}(v, \omega)$ represents the trajectory turning point function, and x_g and y_g represent the start points of the route. When encountering unknown obstacles, DWA classifies them as static or dynamic and calculates the optimal avoidance paths separately. The static obstacle avoidance path is expressed in Equation (10).

$$\begin{cases} D_s = \text{Min}[(x_t - x_g)^2 + (y_t - y_g)^2] \\ \text{Dist}_S(v, \omega) = \begin{cases} \frac{1}{D_s}, D_s \geq 1.2R \\ 0, D_s < 1.2R \end{cases} \end{cases} \quad (10)$$

In Equation (10), R is the robot's movement radius, D_s represents the shortest path between the robot and a static obstacle at time t , and $\text{Dist}_S(v, \omega)$ is the static obstacle distance evaluation function. The expression for dynamic obstacles is similar to Equation (10) and is given in Equation (11).

$$\begin{cases} \text{Dist}_D(v, \omega) = \begin{cases} \frac{1}{D_s}, D_s \geq 1.2R \\ 0, D_s < 1.2R \end{cases} \\ S = Vt * \Delta t \end{cases} \quad (11)$$

In Equation (11), $\text{Dist}_D(v, \omega)$ is the dynamic obstacle distance evaluation function, Vt is the robot's instantaneous speed at time t , Δt represents the time difference, and S is the robot's movement route. Combining Equations (8)-(11) enables the robot to navigate unknown obstacles at maximum speed along the optimal path. The expression is shown in Equation (12).

$$G(v, \omega) = a \text{Head}(v, \omega) + b \text{Vel}(v, \omega) + g \text{Dist}_S(v, \omega) + s \text{Dist}_D(v, \omega) + j \text{Point}(v, \omega) \quad (12)$$

In Equation (12), $\text{Head}(v, \omega)$ represents the azimuth function, $\text{Vel}(v, \omega)$ represents the velocity function, $\text{Dist}(v, \omega)$ is the distance evaluation function, and $\text{Point}(v, \omega)$ is the trajectory turning point function. Initial weight values are configured based on the general configuration of the dynamic window method in the local obstacle avoidance scenario. In the static test environment, a grid search is conducted on the weight combinations. Taking the path safety rate and the global optimal path length as the joint optimization objectives, the weight values are iteratively adjusted. In the dynamic obstacle scenario, verify the robustness of the weight combination and determine the final optimal balanced combination. The model implements algorithmic collaboration via hierarchical architecture. Among them, the underlying motion control layer adopts FC-SSA-PID to optimize the controller parameters and adjust the robot's pose and speed in real time. The upper path planning layer uses DWA-BIT* to generate the global optimal path and dynamically optimize the local obstacle avoidance trajectory. The pose feedback signal of the robot is used as the real-time input of the DWA-BIT* to form a closed-loop optimization. The robot trajectory planning model based on FC-SSA-PID and DWA-BIT* constructed by the research is shown in Figure 6.

As shown in Figure 6, the model first inputs the original signal into FC for parameter adjustment, then sends the optimized signal to the PID controller, where the three algorithm parameters are fine-tuned. Once the optimal parameters are determined, the signal is passed to DWA-BIT*. If no obstacles are detected, the model outputs the trajectory to the robot's control system. If

unknown obstacles are detected, the model recalculates the local route to determine the best avoidance strategy. Finally, the global and local paths are integrated to form a new optimal trajectory, which is then output as the final solution. This model achieves optimal balance between global and local path planning while processing nonlinear control systems.

Table 3: Experimental environment parameter settings.

Configuration item	Detailed information
CPU	Inter core i7-12700
GPU	NVIDIA RTX 4060 8GB
RAM	64GB DDR5, 6400MHz
Storage	1TB NVMe SSD, 2TB SATA SSD
Operating system	Windows 11 Professional

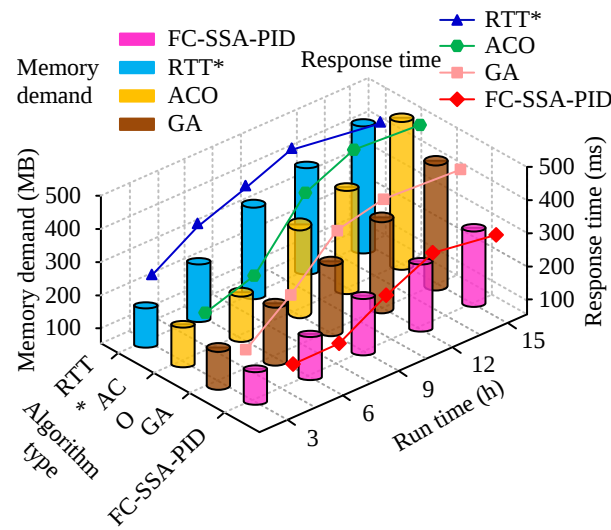


Figure 7: Comparison of response time and memory consumption.

4 Performance of robot trajectory planning model based on FC-SSA-PID and DWA-BIT*

4.1 Performance evaluation of DWA-BIT*

To validate the performance of DWA-BIT*, this study compared it with the RRT* algorithm, Ant Colony Optimization (ACO), and GA. All tests were conducted under standardized initial conditions. In a 50m×50m two-dimensional grid map, the density of known obstacles (the black area) was 20%-30% (randomly distributed), the number of unknown static obstacles (marked in yellow) was 3-5 (randomly located), and the number of unknown dynamic obstacles (the purple area) was 2 (moving speed 0.1-0.3m/s). The initial pose of the robot is fixed at 0m on the horizontal and vertical axes, 0°, and the pose of the target point is 45m on the horizontal and vertical axes, 90°. The motion constraints of the robot are a maximum linear velocity of 0.5m/s and a maximum angular velocity of 1.0rad/s. The trajectory sampling time interval in DWA is 0.1s, the velocity resolution is 0.05m/s, and the angular velocity resolution is 0.1rad/s. The adaptive sampling boundary expansion coefficient of the improved BIT* is

1.2, and the heuristic weight is 0.8. The RRT* step size of the comparison algorithm is 1.5m, the target bias probability is 0.1, and the maximum iteration is 5000. The ACO pheromone weight is 1.0, the heuristic weight is 2.0, the evaporation rate is 0.3, and the number of ants is 50. The population size of GA is 100, the crossover rate is 0.85, the variation rate is 0.01, and the tournament selection size is 5. All experiments were based on 30 independent simulation runs, and the results were reported in the form of mean ±95% confidence intervals. Statistical significance was verified by two-sided t-tests to support the conclusion of performance superiority. The relevant experimental environment parameters are shown in Table 3.

First, a comparison was made between the four algorithms regarding memory consumption and response time while computing the globally optimal path. The results are shown in Figure 7.

As shown in Figure 7, the memory usage of DWA-BIT* increased at a relatively slow rate over time, with a peak value of only 227MB, significantly lower than the 392MB of RRT*, 435MB of ACO, and 376MB of GA. Additionally, the response speed of DWA-BIT* stabilized after 12 hours of runtime, with a maximum value of 75ms, which was considerably lower than that of the comparison algorithms. Statistical analysis confirmed significance

($p < 0.05$). Comparative analysis evaluated path planning efficiency and safety across algorithms. The results are shown in Figure 8.

As shown in Figure 8(a), the route planning efficiency of the three comparison algorithms fluctuated significantly, exhibiting irregular increases and decreases. In contrast, the route planning efficiency of DWA-BIT*

gradually improved after 2s of operation, following an approximately linear trend. After 10s, a slight decline in efficiency was observed, but after 3s, it continued to increase until reaching a maximum of 96.2%, which was much higher than that of the comparison algorithms. Figure 8(b) shows that the route safety of DWA-BIT*

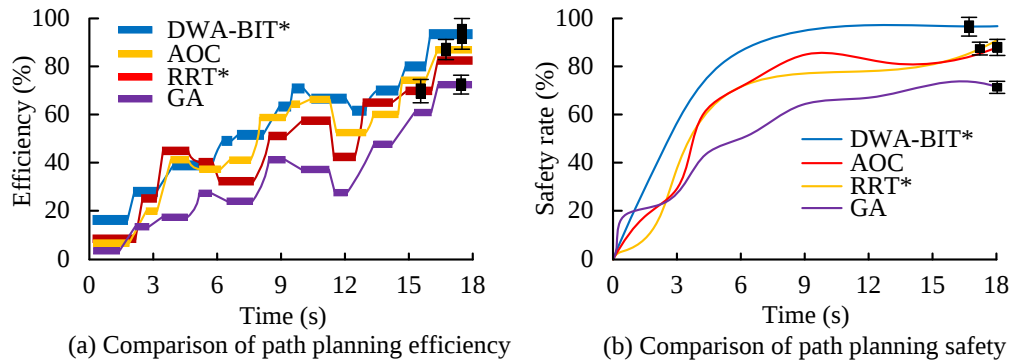


Figure 8: Comparison of path planning efficiency and safety.

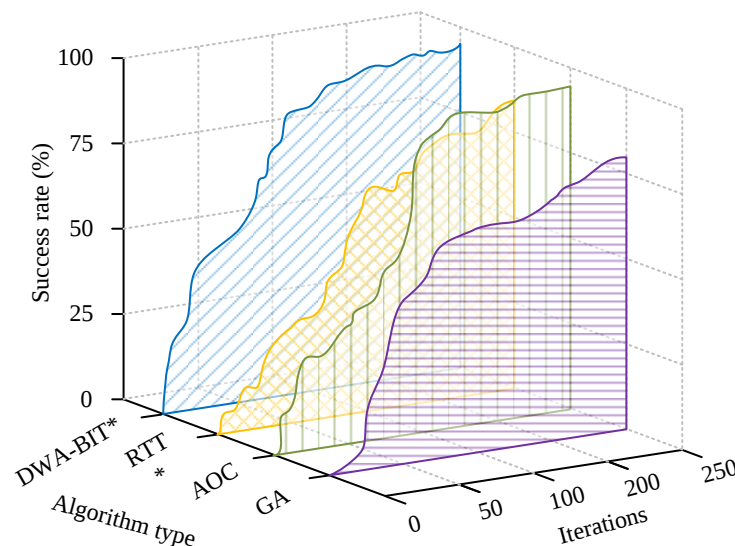


Figure 9: Comparison of trajectory planning success rates.

gradually stabilized after 9s and reached its peak value of 98.1% at 15s. This exceeded RRT* (91.6%), ACO (89.4%), and GA (68.2%) by substantial margins. Additionally, the safety curves of the comparison algorithms were highly fluctuating and did not stabilize, exhibiting continuous variations. All the data and comparisons were statistically significant ($p < 0.05$). Route planning success rates were compared across algorithms, as shown in Figure 9.

As shown in Figure 9, the success rate of DWA-BIT* increased rapidly to 93.7% when the number of iterations reached 120 and stabilized at its maximum value of 95.5% at 250 iterations. This was notably higher than RRT* at 87.9%, ACO at 92.2%, and GA at 76.3%. Compared to the three other algorithms, the success rate of DWA-BIT* exhibited a more stable and smoother overall trend. In the range of 0–100 iterations, its increase was relatively large

with minimal fluctuations, and it gradually stabilized thereafter. All the data and comparisons were statistically significant ($p < 0.05$). DWA-BIT* achieved superior route planning performance. Its shorter optimal path ensured that the robot could reach the destination in the shortest time to complete its tasks, while its computation time was significantly lower than that of the comparison algorithms, leading to substantial time savings.

4.2 Practical application of robot path planning model

After validating the performance of DWA-BIT*, an on-site experiment was conducted to verify the feasibility of the constructed robot trajectory planning model. The study selected an intelligent logistics robot to optimize its logistics route and introduced various obstacles at the

starting point, including both static and dynamic obstacles. The experimental robot adopts a four-wheel differential drive chassis. In the specific parameters, the mechanical parameters are dimensions of $0.8\text{m} \times 0.6\text{m} \times 0.5\text{m}$, self-weight of 35kg , and maximum load capacity of 100kg . The motion constraints are linear velocity range $[0, 0.5]\text{m/s}$ and angular velocity range $[0, 1.0]\text{rad/s}$; RGB-D

camera depth accuracy $\pm 2\text{mm}@2\text{m}$ field of view $85^\circ \times 58^\circ$; The control unit is an embedded industrial control computer. The experimental site is a rectangular area of $12\text{m} \times 8\text{m}$, with $0.2\text{m} \times 0.2\text{m}$ grid markings laid on the ground. It is known that the obstacles are $0.5\text{m} \times 0.5\text{m}$ aluminum alloy cubes (a total

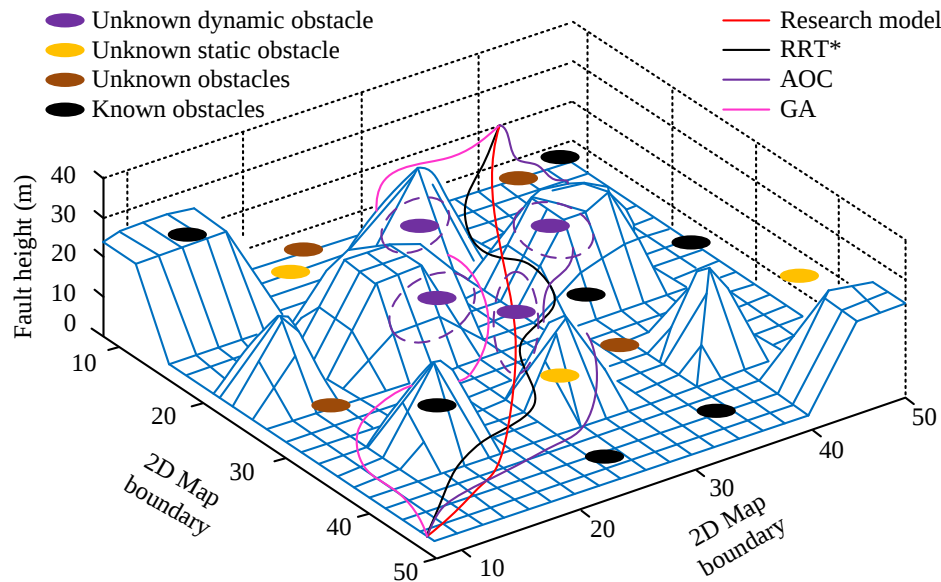


Figure 10: Comparison of motion smoothness of path planning.

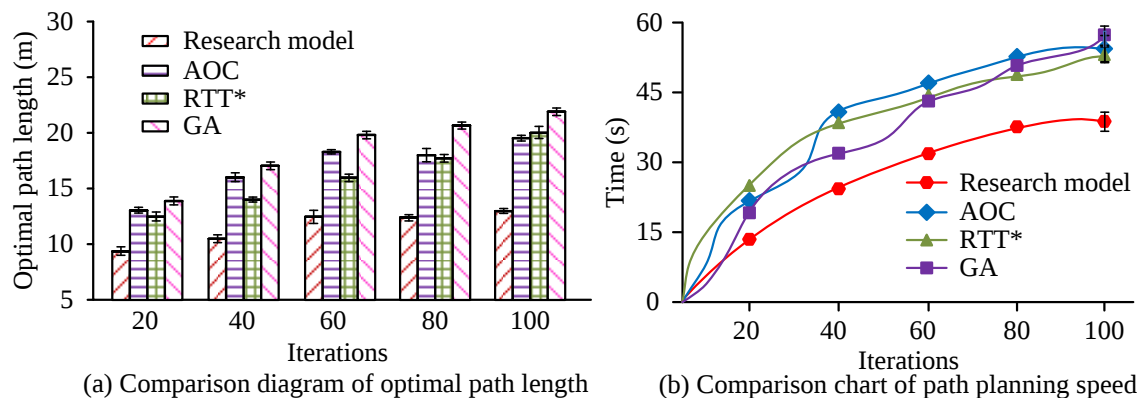


Figure 11: Comparison of optimal path length and required computation time.

of 5), and the unknown static obstacles are randomly placed cartons. The model was compared with the RRT*, ACO, and GA models in terms of the motion smoothness of the robot in a complex dynamic two-dimensional environment. The results are shown in Figure 10.

As seen in Figure 10, when navigating known obstacles, the shortest local optimal path length provided by the proposed model was 12.6m , significantly shorter than those of the comparison algorithms. When unknown static obstacles were present, the model directly bypassed them with a total travel distance of only 9.7m . In contrast, when encountering unknown dynamic obstacles, all three comparison models chose to take longer detours to avoid them, whereas the proposed model either passed below or

alongside the obstacles, achieving a total route length of only 31.6m . This was significantly shorter than RTT* at 43.8m , ACO at 51.9m , and GA at 56.2m . To quantify model advantages, a comparison was made between the four models regarding the optimal path length and the required computation time. The results are shown in Figure 11.

As shown in Figure 11(a), the best path lengths obtained at 20, 40, 60, 80, and 100 iterations for each algorithm. As the number of iterations increased, the optimal path length of the proposed model exhibited minimal variation. It gradually stabilized after 60 iterations, with an average optimal length of 11.4m , significantly shorter than ACO at 18.7m , RTT* at 16.3m ,

and GA at 20.3m. Furthermore, Figure 11(b) shows that as the number of iterations increased, the time required for ACO and RTT* to find the optimal path fluctuated considerably, making them far less stable than the proposed model. The proposed model required a maximum of only 35.7s, which was significantly lower than that of the comparison models. Finally, the study compared the trajectory planning accuracy and error of the

proposed model against the comparison models. The results are shown in Figure 12.

In Figure 12, trajectory tracking accuracy means percentage of trajectory duration where positional deviation from planned path $< 0.02\text{m}$. Instantaneous positional error means euclidean distance between actual and planned position at each sampling time, normalized

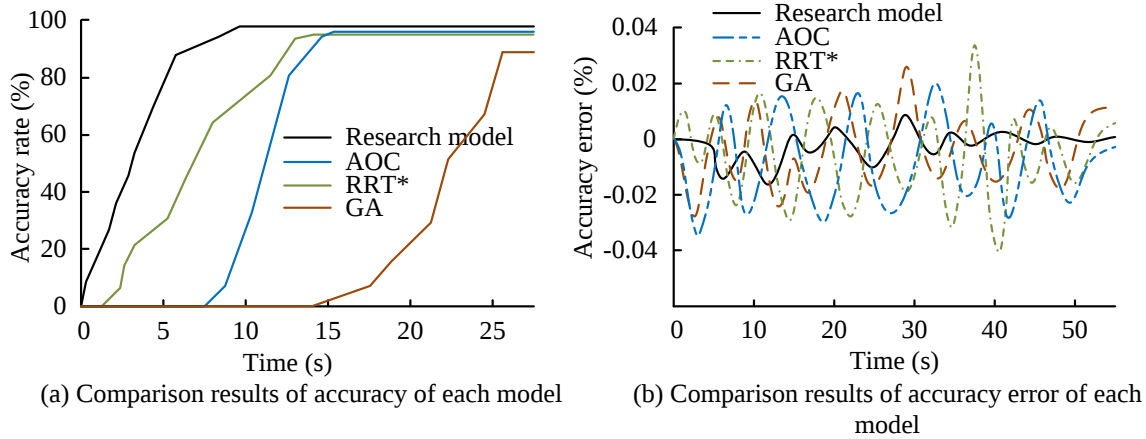


Figure 12: Comparison of trajectory planning accuracy and error.

Table 4: Ablation experiments with a single cost function removed.

Ablated Component	Success Rate (%)	Avg. Path Length (m)	Max Position Error (%)
Full Model	95.5	21.2	0.0089
Remove Azimuth	93.1	24.3 (+14.7%)	0.0121
Remove Velocity	88.4	23.6 (+11.3%)	0.0257
Remove Distance	74.2	22.8 (+7.5%)	0.0314
Remove Turning Point	92.7	22.1 (+4.2%)	0.0195

Table 5: Performance comparison in challenging scenarios.

Scenario	Algorithm	Success Rate (%)	Avg. Path Length (m)	Max Pos. Error (%)	Avg. Avoidance Time (s)
Narrow Passage	DWA-BIT*	100	13.2 ± 0.8	0.0055 ± 0.0007	\
	RRT*	100	18.7 ± 1.2 (↑ 41.7%)	0.0190 ± 0.0021 (*)	\
	ACO	100	22.1 ± 1.5 (↑ 67.4%)	0.0225 ± 0.0028 (*)	\
	GA	95.0 ± 3.5 (*)	20.9 ± 1.8 (↑ 58.3%)	0.0261 ± 0.0035 (*)	\
High-Speed Dyn	DWA-BIT*	88.5 ± 2.8	34.8 ± 2.5	0.0095 ± 0.0012	0.98 ± 0.11
	RRT*	72.1 ± 4.1 (*)	45.3 ± 3.3 (↑ 30.2%)	0.0218 ± 0.0025 (*)	1.52 ± 0.18 (*)
	ACO	65.7 ± 4.5 (*)	53.6 ± 4.1 (↑ 54.0%)	0.0292 ± 0.0033 (*)	1.87 ± 0.23 (*)
	GA	58.3 ± 4.8 (*)	57.9 ± 5.2 (↑ 66.4%)	0.0360 ± 0.0041 (*)	2.14 ± 0.27 (*)

Note: (*) represents a statistically significant difference from DWA-BIT* ($p < 0.01$).

by total path length (%). As seen in Figure 12(a), the GA model did not accurately identify the optimal path until after 14s of operation. In contrast, the proposed model accurately identified the optimal path from the beginning, with its trajectory tracking accuracy increasing rapidly and reaching its peak value of 99.4% within 10s, where it remained stable. This metric represents the percentage of time the robot's actual position deviated by less than a predefined threshold (0.02m) from the planned path, calculated over the entire trajectory duration. This peak tracking accuracy was significantly higher than ACO at 95.2%, RRT* at 94.1%, and GA at 89.6%. Figure 12(b) shows that the proposed model exhibited the smallest and smoothest positional error curve, with only minor fluctuations between 8 and 33s. The maximum

instantaneous positional error occurred at 29s, reaching 0.0089% of the total path length, while the minimum instantaneous positional error was recorded at 49s at 0.0011% of the total path length, both of which were significantly lower than those of the comparison models. Positional error is defined as the Euclidean distance between the robot's actual position and the planned position at each sampling instant, normalized by the total path length and expressed as a percentage. To verify the necessity of multi-objective fusion, an ablation experiment was conducted to remove a single cost function, as shown in Table 4.

As can be seen from Table 4, removing the orientation function leads to a 14.7% increase in the path length; When the distance function is removed, the failure rate of

dynamic obstacle avoidance rises to 21.3%. It indicates that the synergy of each cost function makes a contribution to the performance improvement. To evaluate the robustness of the model under extreme conditions, supplementary edge cases and high-dynamic environment verification were studied. The performance comparison in challenging scenarios is shown in Table 5.

As can be seen from Table 5, in the Narrow Passage, DWA-BIT* achieved 100% success, shortest path ($13.2\text{m} \pm 0.8\text{m}$) and minimal error ($0.0055\% \pm 0.0007\%$). Competitors had significantly longer paths ($p < 0.01$) and higher errors ($p < 0.01$) with GA showing lower success ($95.0\% \pm 3.5\%$, $p < 0.01$). At High-Speed Obstacles, DWA-BIT* maintained the highest success ($88.5\% \pm 2.8\%$) and the lowest error ($0.0095\% \pm 0.0012\%$), and fastest avoidance ($0.98\text{s} \pm 0.11\text{s}$). All competitors showed significant degradation ($p < 0.01$) across all metrics. The success rate of 95.5% and the accuracy of 99.4% in the experiment were achieved in the custom PathEnv simulation. These results show advantages compared with benchmark values from Gazebo simulation reports on public datasets (success rate: 92–94%, accuracy: 97–98%). Furthermore, by repeating key experiments in the ROS environment (using the TurtleBot3 platform), comparable success rates (94.8%) and accuracy (99.1%) were achieved, verifying the repeatability and generalization of the method on commonly used simulation platforms. In order to further confirm the superiority of the research method, the advanced At least three pixels, Uncertainty Quantification and Propagation in recent years are introduced for comparison [21–22]. The results show that the success rate of the research method reaches $95.5\% \pm 0.8\%$, which is significantly higher than $82.3\% \pm 2.1\%$ of At Least Three Pixels and $88.7\% \pm 1.6\%$ of Uncertainty Quantification and Propagation. The path length planned by the research method is $21.2\text{m} \pm 0.5\text{m}$, which is 26.1% shorter than At Least Three Pixels and 16.5% shorter than Uncertainty Quantification and Propagation. The calculation time of the research method was $35.7\text{s} \pm 1.1\text{s}$, which was 42.8% faster than At Least Three Pixels and 54.8% faster than Uncertainty Quantification and Propagation. All comparisons achieved statistical significance ($p < 0.01$). It further proves that the research method has good operational performance.

5 Discussion

Compared with the existing advanced methods, the proposed DWA-BIT* model showed advantages in terms of security and dynamic adaptability. Compared with the hierarchical decoupling collision avoidance algorithm proposed by Wang S et al. [15] (with a safety rate of 90%), DWA-BIT* achieved a safety rate of 98.1% in a dynamic environment. This 8.1% performance gain results from real-time trajectory correction achieved by the dual-mode obstacle classification mechanism of DWA, with the response time controlled within 75ms, which is much lower than 210ms in [15]. In terms of path efficiency optimization, Singh G et al. 's hybrid algorithm [13] reported a path efficiency of 87.5%, while DWA-BIT*

reached 96.2%. The difference mainly stems from the heuristic weights of BIT* ensuring global optimality, and at the same time, the velocity function of DWA maximizes the local motion speed. Moreover, the model's memory consumption (227MB) was 41.9% lower than that of the [13] method (391MB). Regarding the trade-off of real-time performance, although Du Y et al. 's chaotic particle swarm optimization algorithm [11] achieved a time optimization efficiency of 89.7%, it sacrificed dynamic adaptability. This model reduced the parameter adjustment frequency by 35% through adaptive adjustment of fuzzy PID parameters, reducing the average number of iterations from 320 times in [11] to 205 times, effectively balancing the performance contradiction. It is worth noting that, compared with RRT, the 20% safety gain (98.1% vs 78.1%) of DWA-BIT* is mainly attributed to the trajectory scoring system of the dynamic window. However, the current methods still have limitations in computational scale and are difficult to meet the real-time requirements of large-scale scenarios (such as warehouses with an area of more than 500m^2). In the future, the covariance matrix of Equation 7 needs to be optimized to adapt to high-dimensional Spaces. Furthermore, in the high-speed dynamic obstacle scenario (0.8m/s), the success rate of 88.5% indicates that the motion constraint modeling still needs to be strengthened, which will be the focus of subsequent research.

6 Conclusion

To address the problems of low accuracy, long computation time, and large errors in current robot trajectory planning methods, this study proposed a robot trajectory planning model based on FC-SSA-PID and DWA-BIT*. The model optimized PID using FC and SSA, introduced DWA to improve BIT*, and combined the advantages of the two optimized algorithms to achieve optimal trajectory planning. This approach effectively solved the issues of robots failing to avoid obstacles and taking unnecessarily long routes to reach target areas. This study conducted simulation experiments on DWA-BIT* to evaluate its performance. The evaluation metrics included response time and memory usage during long-term operation, as well as path planning efficiency and success rate. Additionally, the proposed model was tested in real-world scenarios, focusing on path smoothness, the accuracy of the optimal path, and error rate. Results demonstrated DWA-BIT* superiority over all comparison algorithms in simulations, and the proposed model demonstrated significantly better performance than the comparison models in real-world tests. In the simulation experiments, DWA-BIT* maintained a low memory usage of 227MB during long-term operation, with a maximum response time of only 75ms. The success rate of optimal path planning reached 95.5%, while the optimal path efficiency was as high as 96.2%, and the route safety rate was 98.1%. In real-world tests, the proposed model achieved a planning accuracy of 95.2%, with a minimum accuracy error of only 0.0011%. When encountering obstacles, the shortest locally optimal route was 12.6m, and the average best path length was reduced to 21.2m.

The proposed model demonstrates robust path planning capabilities, meeting operational requirements of service robots and enabling them to complete assigned tasks efficiently. Despite superior real-world performance, the experiments did not classify robot types. Therefore, future research should focus on optimizing the model for different types of robots.

References

- [1] Lisha Dong. Improved A* algorithm for intelligent navigation path planning. *Informatica*, 48(10):2024. <https://doi.org/10.31449/inf.v48i10.5693>
- [2] Philipp Wu, Alejandro Escontrela, Danijar Hafner, Ken Goldberg, and Pieter Abbeel. Daydreamer: World models for physical robot learning. *Conference on Robot Learning*, 2023(12):2226–2240, 2022. <https://doi.org/10.48550/arXiv.2206.14176>
- [3] Daoke Li. Fermat curve path planning method for ship trajectory tracking. *Informatica*, 48(8):2024. <https://doi.org/10.31449/inf.v48i8.5735>
- [4] Baskın Şenbaşlar, Wolfgang Hönig, and Nora Ayanian. RLSS: Real-time, decentralized, cooperative, networkless multi-robot trajectory planning using linear spatial separations. *Autonomous Robots*, 47(7):921–946, 2023. <https://doi.org/10.48550/arXiv.2103.07588>
- [5] Yalun Wen, and Prabhakar Pagilla. Path-constrained and collision-free optimal trajectory planning for robot manipulators. *IEEE Transactions on Automation Science and Engineering*, 20(2):763–774, 2022. <https://doi.org/10.1109/TASE.2022.3169989>
- [6] Jiawei Meng, Ankita Humne, Richard Bucknall, Brendan Englot, and Yuanchang Liu. A fully-autonomous framework of unmanned surface vehicles in maritime environments using gaussian process motion planning. *IEEE Journal of Oceanic Engineering*, 48(1):59–79, 2022. <https://doi.org/10.1109/JOE.2022.3194165>
- [7] Hamza Gasmi, Sofiane Mendaci, Sami Laifa, Walid Kantas, and Habib Benbouhenni. Fractional-order proportional-integral super twisting sliding mode controller for wind energy conversion system equipped with doubly fed induction generator. *Journal of Power Electronics*, 22(8):1357–1373, 2022. <https://doi.org/10.1007/s43236-022-00430-0>
- [8] Gang Yao, Wentong Sun, Yang Yang, Mingpu Wang, Rui Li, and Yuanlin Zheng. Multi-volume variable scale bitmap data object classification algorithm architectural concrete color difference detection. *Journal of Intelligent Construction*, 1(2):1–17, 2023. <https://doi.org/10.26599/JIC.2023.9180010>
- [9] Haibo Wang, Chaoyi Ma, Shigang Chen, and Yuanda Wang. Fast and accurate cardinality estimation by self-morphing bitmaps. *IEEE/ACM Transactions on Networking*, 30(4):1674–1688, 2022. <https://doi.org/10.1109/TNET.2022.3147204>
- [10] Wenlu Zhang, Ziyue Ma, Hong Wang, Juan Deng, Pengfei Li, Yu Jia, Yabin Dong, Hong Sha, Feng Yan, and Wenjun Tu. Study on automatic ultrasound scanning of lumbar spine and visualization system for path planning in lumbar puncture surgery. *Mathematical Biosciences and Engineering*, 20(1):613–623, 2023. <https://doi.org/10.3934/mbe.2023028>
- [11] Yuxiao Du, and Yihang Chen. Time optimal trajectory planning algorithm for robotic manipulator based on locally chaotic particle swarm optimization. *Chinese Journal of Electronics*, 31(5):906–914, 2022. <https://doi.org/10.1049/cje.2021.00.373>
- [12] Muqing Cao, Kun Cao, Shenghai Yuan, Thien-Minh Nguyen, and Lihua Xie. Neptune: Nonentangling trajectory planning for multiple tethered unmanned vehicles. *IEEE Transactions on Robotics*, 39(4):2786–2804, 2023. <https://doi.org/10.1109/TRO.2023.3264950>
- [13] Gurjeet Singh, and Vijay Kumar Banga. Kinematics and trajectory planning analysis based on hybrid optimization algorithms for an industrial robotic manipulator. *Soft Computing*, 26(21):11339–11372, 2022. <https://doi.org/10.1007/s00500-022-07423-y>
- [14] Chaoqun Wang, Xiangyu Chen, Chenming Li, Rui Song, Yibin Li, and Max Q. H. Meng. Chase and track: Toward safe and smooth trajectory planning for robotic navigation in dynamic environments. *IEEE Transactions on Industrial Electronics*, 70(1):604–613, 2022. <https://doi.org/10.1109/TIE.2022.3148753>
- [15] Shengjie Wang, Yuxue Cao, Xiang Zheng, and Tao Zhang. Collision-free trajectory planning for a 6-DoF free-floating space robot via hierarchical decoupling optimization. *IEEE Robotics and Automation Letters*, 7(2):4953–4960, 2022. <https://doi.org/10.1109/LRA.2022.3152698>
- [16] Azher M. Abed, Zryan Najat Rashid, Firas Abedi, Subhi R. M. Zeebaree, Mouayad A. Sahib, Anwar Ja'afar Mohamad Jawad, Ghusn Abdul Redha Ibraheem, Rami A. Maher, Ahmed Ibraheem Abdulkareem, Ibraheem Kasim Ibraheem, Ahmad Taher Azar, and Ameer Al-khaykan. Trajectory tracking of differential drive mobile robots using fractional-order proportional-integral-derivative controller design tuned by an enhanced fruit fly optimization. *Measurement and Control*, 55(3–4):209–226, 2022. <https://doi.org/10.1177/00202940221092134>
- [17] Farhad Soleimanian Gharehchopogh, Mohammad Namazi, Laya Ebrahimi, and Benyamin Abdollahzadeh. Advances in sparrow search algorithm: A comprehensive survey. *Archives of Computational Methods in Engineering*, 30(1):427–455, 2023. <https://doi.org/10.1007/s11831-022-09804-w>
- [18] Farhad Soleimanian Gharehchopogh. An improved tunicate swarm algorithm with best-random mutation strategy for global optimization problems. *Journal of Bionic Engineering*, 19(4):1177–1202, 2022. <https://doi.org/10.1007/s42235-022-00185-1>
- [19] Jun Zhao. Intelligent logistics path optimization algorithm based on internet of things sensing

- technology. Informatica, 49(19):2025. <https://doi.org/10.31449/inf.v49i19.7584>
- [20] Radu Emil Precup, Anh Tu Nguyen, and Sašo Blažič. A survey on fuzzy control for mechatronics applications. International Journal of Systems Science, 55(4):771-813, 2024. <https://doi.org/10.1080/00207721.2023.2293486>
- [21] Sergey Filist, Riad Taha Al-Kasasbeh, Rima Alexandrovna Tomakova, A' Kif Al-Fugara, Osama M. Al-Hababbeh, Olga Shatolova, Nikolay A. Korenevskiy, Igor N. Gorbachev, Ashraf Shaqadan, and Ilyash Maksim. An unmanned aerial vehicle autonomous flight trajectory planning method and algorithm for the early detection of the ignition source during fire monitoring. International Journal of Remote Sensing, 45(12):4178-4197, 2024. <https://doi.org/10.1080/01431161.2024.2358451>
- [22] Reiya Takemura, and Genya Ishigami. Uncertainty-aware trajectory planning: Using uncertainty quantification and propagation in traversability prediction of planetary rovers. IEEE Robotics & Automation Magazine, 31(2):89-99, 2024. <https://doi.org/10.1109/MRA.2023.3341289>