scientific reports

Check for updates

OPEN

Route planning of mobile robot based on improved RRT star and TEB algorithm

Xiong Yin, Wentao Dong, Xiaoming Wang, Yongxiang Yu & Daojin Yao ✉

This paper presents a fusion algorithm based on the enhanced RRT* TEB algorithm. The enhanced RRT* algorithm is utilized for generating an optimal global path. Firstly, proposing an adaptive sampling function and extending node bias to accelerate global path generation and mitigate local optimality. Secondly, eliminating path redundancy to minimize path length. Thirdly, imposing constraints on the turning angle of the path to enhance path smoothness. Conducting kinematic modeling of the mobile robot and optimizing the TEB algorithm to align the trajectory with the mobile robot's kinematics. The integration of these two algorithms culminates in the development of a fusion algorithm. Simulation and experimental results demonstrate that, in contrast to the traditional RRT* algorithm, the enhanced RRT* algorithm achieves a 5.8% reduction in path length and a 62.5% decrease in the number of turning points. Utilizing the fusion algorithm for path planning, the mobile robot generates a superior, seamlessly smooth global path, adept at circumventing obstacles. Furthermore, the local trajectory meticulously conforms to the kinematic constraints of the mobile robot.

Keywords RRT* algorithm, TEB algorithm, Path planning, AGV, Kinematic

With the rapid advancements in robot technology, the potential for robots to supplant human roles has emerged[1,2]. The mobile robot stands out prominently in both domestic and international arenas as one of the focal points in robot research[3]. Its applications span across hospitals, railway stations, warehouses, and shopping malls[4]. With robots capable of operating continuously, a foreseeable shift involves them taking over repetitive manual tasks from humans[5]. At the core of this shift lies the essence of path planning for mobile robots[6].

Currently, a multitude of algorithms cater to mobile robot path planning. These encompass the Voronoi algorithm[7], A* algorithm[8], ant colony algorithm[9], grey Wolf algorithm[10], genetic algorithm[11], RRT algorithm[12], particle swarm algorithm[13], ant colony algorithm[14], artificial potential field algorithm[15], neural network algorithm[16], deep learning algorithm[17], and more.

Path planning algorithms can be categorized into global path planning and local path planning based on their characteristics[18]. Among global planning algorithms, the RRT* algorithm, which employs sampling for global path planning, is widely adopted in mobile robot applications due to its accelerated path generation rate. Numerous researchers have contributed enhancements to the RRT* algorithm, including the bidirectional RRT* algorithm[19], Informed RRT* algorithm[20], and others.

In the realm of path planning, numerous algorithms have been proposed by scholars[21,22]. Tang et al. [23] introduces an enhanced A* algorithm within the domain of global path planning. The method consists of two main steps: firstly, removing irregular waypoints with functions P (x, y) and W (x, y); secondly, smoothing the path using B splines. This approach significantly reduces both the path length and the number of nodes. Meanwhile, Mashayekhi et al.[24] delve into a hybrid RRT algorithm. Their method begins by employing a bidirectional tree for searching, then optimizing the current node through informed sampling once the two trees are connected. This technique notably enhances the path quality. Moving on, Zhu et al.[25] adopt a reverse labeling Dijkstra algorithm for global path planning. Initially, they theoretically substantiate the algorithm's rationality and demonstrate its low complexity. Subsequently, they validate its effectiveness by applying it to a real road network. In the realm of local path planning, Zhang et al.[26] present an improved ant colony algorithm. The strategy includes using the artificial potential field to determine force direction in the initial ship phase, improving the attraction potential field function, and creating pseudo-random transfer rules. These changes significantly improve the algorithm's convergence. Similarly, Du et al.[27] propose a dynamic artificial potential field algorithm. The strategy begins by

School of Electrical and Automation Engineering, East China Jiaotong University, Nanchang 330000, China. ✉email: ydaojin@whu.edu.cn

adjusting the safe distance dynamically and then improving the potential field force's effectiveness. Ultimately, the algorithm is enhanced by introducing a steering force to change the drone's direction, ensuring safer and more stable paths. Lastly, Kobayashi et al.[28] introduce an algorithm comprising the DWA algorithm and VM. This composite approach furnishes the DWA algorithm with modified variable speeds and predicted obstacle data from VM, facilitating the generation of candidate paths.

While the mentioned algorithms have their merits, they mainly concentrate on either global or local path planning, frequently overlooking the robot's kinematic and dynamic constraints. This paper addresses this limitation by incorporating considerations for both global and local path planning, as well as accounting for the kinematics of the mobile robot. The study includes simulating the improved algorithm, comparing it with classical algorithms. Then implementing it on an experimental prototype to confirm its effectiveness.

In summary, propose a fusion algorithm for a mobile robot to navigate along an optimal path globally while obeying its kinematic constraints. This contribution differs from existing work in three key aspects:

(1) The paper introduces the mobile robot platform and its system, followed by kinematic modeling for the mobile robot;
(2) Enhancements to the RRT* algorithm include optimizing the sampling function. During node expansion, weights are increased to counteract expansion. Redundant path nodes are eliminated to shorten the path. Constraints are applied to turning angles for a smoother trajectory;
(3) Addressing the kinematic constraints of the mobile robot, the TEB algorithm is optimized, incorporating constraints such as obstacle limitations, speed restrictions and acceleration constraints;
(4) The effects of Angle threshold and path-point spacing on the performance of improved RRT* algorithm are discussed.

This paper is structured into six sections, with "Introduction of mobile robot" detailing the robot modeling, "Improved RRT* Algorithm" focusing on RRT* algorithm improvements, "Timed-elastic-band approach optimization" concentrating on TEB algorithm optimization, and "Experimental" encompassing simulations and experiments. Finally, "Conclusion" provides the conclusion of this study.

Introduction of mobile robot

Platform hardware introduction

The mobile robot comprises a depth camera, laser radar, main control unit and robot chassis. The main control unit is built around a Raspberry Pi, incorporating an STM32 chip. The robot chassis is constructed with two servo motors, two driving wheels, and an omnidirectional wheel, as depicted in Fig. 1.

Platform system introduction

The platform system integrates the ROS operating system into Raspberry Pi, with key components encompassing environment awareness, data processing and path navigation. ROS, widely recognized as the most extensively used open-source robot software platform, greatly improves the efficiency of robot development. The laser radar collects environmental data, while the Raspberry Pi and STM32 chip work together to process it and plan an optimal, obstacle-free path. The servo motor is controlled to start tire rotation, helping the mobile robot move along the planned path, as illustrated in Fig. 2.

Kinematic model of the mobile robot

This paper conducts kinematic modeling for the mobile robot[29]. Within the world coordinate system (x_w, y_w), the state vector $s = [x, y, \theta]$ denotes the current position and pose information of the mobile robot. This is depicted in Fig. 3.

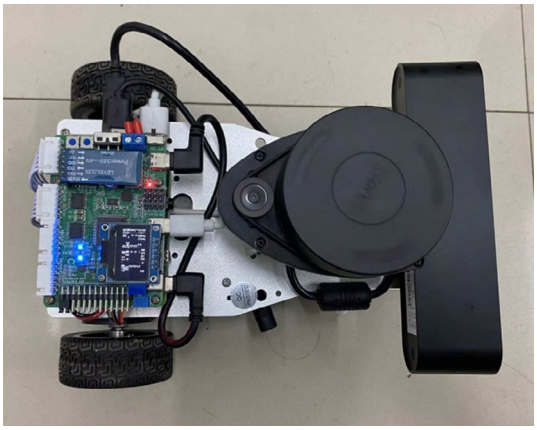

Figure 1. Mobile robot diagram.

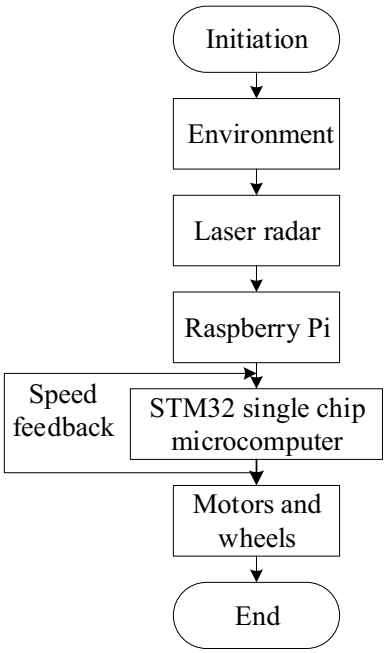

Figure 2. System flow chart.

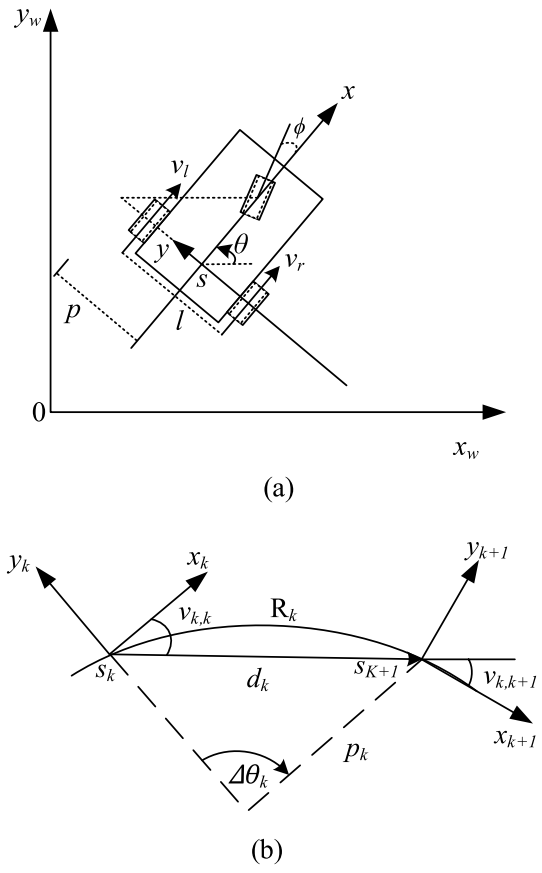

Figure 3. Mobile robot shape and posture diagram.

Given that the mobile robot operates on a two-wheeled differential system, each drive wheel is independently powered by a dedicated motor. Here, r represents the radius of the driving wheel, while $w_l(t)$ and $w_r(t)$ denote the angular speed of the left and right driving wheels, respectively. The linear speed of the left and right drive wheels is expressed as $v_l(t)$ and $v_r(t)$.

$$\begin{aligned} v_l(t) &= w_l(t)r \\ v_r(t) &= w_r(t)r \end{aligned} \tag{1}$$

where t represents the turning time period, p is half of the wheelbase of the driving wheel. Consequently, the local coordinate system for the mobile robot positions the y-axis along the horizontal direction of its two driving wheels, with the x-axis aligned with its moving direction. The speed of the mobile robot, denoted as $v(t)$. Speed of the mobile robot $v(t) = (v_l(t) + v_r(t))/2$. This leads to the conclusion that the motion of the mobile robot can be characterized by a nonlinear equation.

$$\dot{S}(t) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} v(t)\cos(\theta(t)) \\ v(t)\sin(\theta(t)) \\ w(t) \end{bmatrix} \tag{2}$$

where $s(t)$ represents the position of the mobile robot at time t, $u(t) = [v(t), w(t)]^{\mathrm{T}}$ denotes the control input vector.

$$u(t) = \begin{bmatrix} v(t) \\ w(t) \end{bmatrix} = \begin{bmatrix} \frac{v_l(t)+v_r(t)}{2} \\ \frac{v_l(t)-v_r(t)}{l} \end{bmatrix} \tag{3}$$

where l is two driving wheels wheelbase, v_l(t) is the left-wheel speed, v_r(t) is the right-wheel speed, $v_{k,k}$ is the angle between s_k pose at moment k, the direction $d_k = [x_{k+1}\text{-}x_k, y_{k+1}\text{-}y_k, 0]^{\mathrm{T}}$ of the mobile robot. $v_{k,k+1}$ is the angle between the d_k direction and s_{k+1} posture of the mobile robot at time $k+1$.When $v_{k,k} = v_{k,k+1}$, a common arc of constant curvature can be obtained.

$$v_{k,k} = v_{k,k+1} \tag{4}$$

$$h_k(s_{k+1}, s_k) = \left(\begin{bmatrix} \cos(\theta_k) \\ \sin(\theta_k) \\ 0 \end{bmatrix} + \begin{bmatrix} \cos(\theta_{k+1}) \\ \sin(\theta_{k+1}) \\ 0 \end{bmatrix} \right) \times d_k = 0 \tag{5}$$

According to Fig. 3b, p_k is the turning radius, arc length $R_k = p_k \Delta\theta_k$. Angle $\Delta\theta_k = \theta_{k+1} - \theta_k$ is the angle change between the s_k position and s_{k+1} position of the mobile robot. The turning radius p_k is.

$$p_k = \frac{\|d_k\|_2}{\left|2\sin(\frac{\triangle\theta_k}{2})\right|} \overset{\triangle\theta_k \ll 1}{\approx} \frac{\|d_k\|_2}{|\triangle\ \theta_k|} \tag{6}$$

where meet $p_k \geq p_{min}$, p_{min} is the minimum turning radius.

Improved RRT* algorithm

Traditional RRT* algorithm

The Rapidly Exploring Random Tree Star (RRT*) stands as a global path planning algorithm, representing an enhancement over the Rapidly Exploring Random Tree (RRT)[30]. The core method involves several steps: first, randomly sampling within the space. Once a sample point is obtained, the algorithm finds the nearby node. Second, a new node is created between this nearby node and the sampled one. Next, collision detection is done on the new node. If there's no collision, it's added to the path. The parent node is then adjusted for the new one. This continues until the algorithm reaches the target point.

Adaptive sampling function

In the conventional RRT* algorithm, the initial path is formed through random sampling across the entire obstacle space, resulting in a considerable number of sampling nodes. To enhance the target-oriented nature of sampling points, an adaptive sampling function is introduced. A proposed adaptive sampling function involves setting a target bias probability denoted as p. Generate sampling points based on the associated probability. This adaptive sampling function accelerates the generation of the initial path, reduces the quantity of sampling points, improves sampling efficiency, and prevents the initial path from succumbing to local optima. The adaptive sampling function is instrumental in achieving these enhancements.

$$X_{rand} = \begin{cases} x_{rand}(p = 0.8) \\ X_{Goal}(p = 0.2) \end{cases}, \quad p \in (0, 1) \tag{7}$$

where p represents probability, x_{rand} denotes the sampling point generated by random sampling, X_{Goal} signifies target point, X_{Goal} is sampling point.

Node bias expansion

Given the relatively lengthy trajectory of the mobile robot, it is essential to constrain the expansion direction of the new node, denoted as X_{new}. This constraint is crucial for expediting path planning[31]. In the traditional RRT*

algorithm, the new node X_{new} is traditionally extended towards a randomly generated point, X_{rand}. However, this random expansion approach often results in an unfocused expansion of nodes, leading to prolonged path planning durations. To address this inefficiency, this paper introduces a node expansion bias. This bias leverages target points to guide and constrain the expansion of new nodes. Specifically, different weights are assigned to the direction of the target point and the sampling point. The weight in the direction of the goal point, X_{Goal}, is denoted as g, while the weight in the X_{rand} direction is denoted as r. This configuration is illustrated in Fig. 4. The introduction of this node expansion bias aims to optimize the expansion process and enhance the efficiency of path planning.

Remove redundant points in the path

The path generated from the starting point to the destination point often contains numerous redundant points, contributing to the elongation of the path[32]. Reducing redundant points decreases the time needed for the mobile robot to complete the task, making it essential to trim them for a collision-free optimal path. This paper employs the greedy algorithm for path redundancy reduction. In Fig. 5, the blue line segment represents the branch of the random tree, the black line segment corresponds to the initial path, the red dotted line indicates the path after redundancy removal, and the gray circles symbolize obstacles. To remove redundancy path: $X_1 \rightarrow X_2 \rightarrow X_3 \rightarrow X_4 \rightarrow X_5 \rightarrow X_6 \rightarrow X_7 \rightarrow X_8 \rightarrow X_9$. Iterate through the path in turn. Until node X_4 is encountered, connecting nodes X_1 and X_4 will collide with obstacles. Connect X_1 and X_3, delete node X_2. Repeat the process until reaching node X_9. The path with redundancy removed can be obtained: $X_1 \rightarrow X_3 \rightarrow X_4 \rightarrow X_8 \rightarrow X_9$. As shown in the Fig. 5.

Path smoothing

Once redundancy is eliminated, the path may still face issues with excessively wide turning angles. The path turning angle is too large, which does not conform to the kinematics of the mobile robot. Therefore, it is necessary to constrain the turning angle of the path. Firstly, the coordinates of three consecutive waypoints are obtained. The angular deviation of two adjacent coordinates is calculated. The angle deviation threshold A_d is set. The calculated result is greater than the threshold. Then, the midpoint of the preceding. Following two points is used as the coordinates of the intermediate points, so as to achieve path smoothing. Secondly, the angle normalization function is set up. It combines angles from two nearby points in the map system into the range of $(-\pi, \pi)$ to make angle comparisons consistent. Obtain the coordinates of the current time and the previous time waypoint. Calculate the distance. If the distance obtained is greater than the threshold, a new waypoint is inserted between the two waypoints. The coordinate is the point coordinate of the two path points, so as to increase the path point density. Finally, the path smoothing function is established to obtain the coordinates of three adjacent waypoints. Then the angle value between two adjacent coordinates is calculated. The two angle standardization deviations are obtained by calculating and analyzing the angle standardization function. The calculation result is greater than the threshold. The middle coordinate points are substituted with the median value of the rear coordinate points. This process is repeated continuously to globally optimize the entire path, as shown in the Fig. 6.

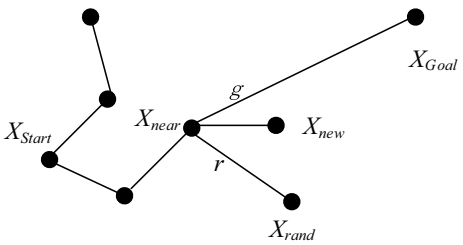

Figure 4. New node offset expansion diagram.

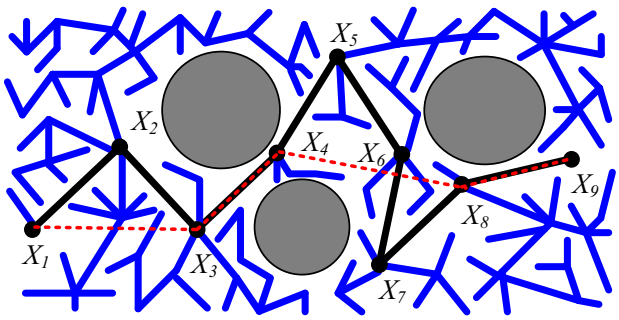

Figure 5. Path de-redundancy diagram.

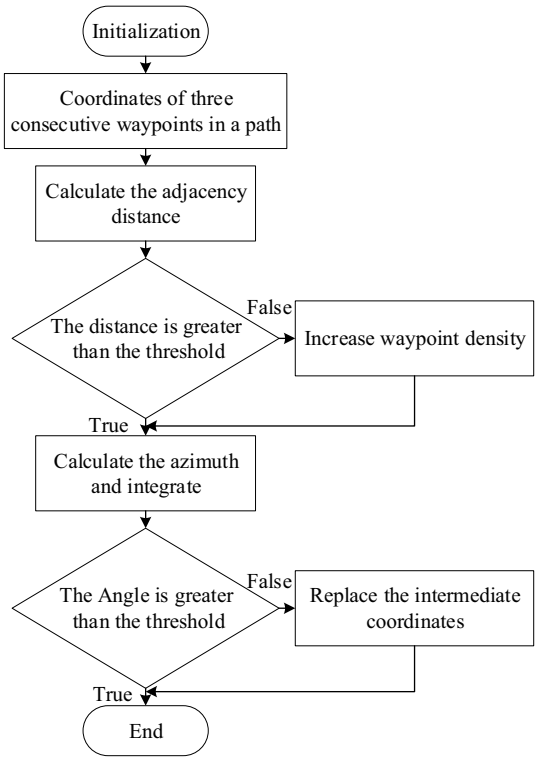

Figure 6. Path smoothing schematic diagram.

Timed-elastic-band approach optimization

Basic description of algorithm

The Timed Elastic Band (TEB) algorithm is a popular method for tackling local obstacle avoidance problems faced by mobile robots[33]. This algorithm is formally defined as a tuple comprising two sequences:

$$B := (Q, \tau) \tag{8}$$

where Q represents the sequence of robot poses, and τ signifies the sequence of time intervals associated with each pose. The basic idea is to dynamically adjust and optimize the algorithm with respect to configuration and time intervals. This is achieved through a real-time weighted multi-objective optimization process.

$$f(B) = \sum_{k} \gamma_k f_k(B) \tag{9}$$

$$B* = \arg\min_{B} f(B) \tag{10}$$

where B^* refers to the optimized Timed Elastic Band (TEB) algorithm, where f_k represents the weighted sum of components contributing to the objective function $f(B)$. The objective function consists of four integral elements: (1) A penalty function, strategically employed to guide the intermediate points toward the original path and circumvent obstacles. (2) Utilization of dynamics to impose restrictions on the robot's speed and acceleration. (3) Adherence to nonholonomic kinematic constraints. (4) Minimization of the square of the sum of the time difference sequence, aiming to identify the swiftest path. The TEB algorithm introduces a constraint on the robot's movement time, thereby enabling real-time control over the mobile robot's motion[34]. This additional constraint enhances the algorithm's ability to govern the movement of the mobile robot in dynamic environments[35].

Algorithm optimization

The application of the TEB algorithm involves optimizing both the dynamics and kinematics of a mobile robot[36]. The following is the optimization formula of TEB algorithm[37,38]. The main aim is to guide the robot smoothly from start to finish along the best path, considering dynamics and kinematics. The optimization problem is defined as a finite dimensional parameter vector $(s_k)_{k=1,2,\ldots,n-1}$.consisting of a discretized sequence of n robot poses… n minus one. The parameter set to be optimized is:

$$B := \{s_1, \Delta T_1, s_2, \Delta T_2, \ldots, s_{n-1}, \Delta T_{n-1}, s_n\} \tag{11}$$

where $(\Delta T_k)_{k=1,2,\ldots,n-1}$ indicates strictly positive time series. Simultaneously, the optimization problem undergoes a transformation into a non-linear program, which involves a series of equations and inequalities.

$$\min_{B} \sum_{k=1}^{n-1} \Delta T_k^2 \tag{12}$$

Subject to

$$s_1 = s_c,\ s_n = s_f,\ 0 \le \Delta T_k \le \Delta T_{\max}, \tag{13}$$

$$h_k(s_{k+1}, s_k) = 0,\ r_k(s_{k+1}, s_k) \ge 0, \tag{14}$$

$$o_k(s_k) \ge 0, \tag{15}$$

$$v_k(s_{k+1}, s_k, \Delta T_k) \ge 0,\ (k = 1, 2, \ldots, n-1) \tag{16}$$

$$\alpha_k(s_{k+2}, s_{k+1}, s_k, \Delta T_{k+1}, \Delta T_k) \ge 0, (k = 2, 3, \ldots, n-2) \tag{17}$$

$$\alpha_1(s_2, s_1, \Delta T_{k+1}) \ge 0, \alpha_n(s_n, s_{n-1}, \Delta T_{n-1}) \ge 0. \tag{18}$$

Among them, the initial pose s_1 and the end pose s_n are replaced by the current mobile robot state s_c and target state s_f. Equality constraint $h_k(\cdot)$satisfies Eq. (5). $r_k(\cdot)$imposes a minimum turning radius and $r_k(\cdot) = r_k - p_{min}$. o is the simply connected area of obstacle model in obstacle map. When there are many obstacles, o will increase the subscript. For example, $o_{l,\ l=1,\ldots,n}$. ψ is an obstacle model. $\delta(s_k, \psi)$ is the distance between the pose of the mobile robot and the obstacle model. The minimum distance of all obstacles is constrained by inequality δ_{min}:

$$o_k(s_k) = [\delta(s_k, \psi_1), \delta(s_k, \psi_2), \ldots, \delta(s_k, \psi_R)]^T - [\delta_{\min}, \delta_{\min}, \ldots, \delta_{\min}]^T \tag{19}$$

where v_k is the linear velocity of the robot at the pose s_k. v_k is defined as:

$$v_k = \frac{p_k \Delta\theta_k}{\Delta T_k} \gamma(s_k, s_{k+1}) \overset{\Delta\theta_k \ll 1}{\approx} \frac{\|d_k\|_2}{\Delta T_k} \gamma(s_k, s_{k+1}) \tag{20}$$

where $\gamma(\cdot) \in [-1,1]$is sign of mobile robot velocity. The result of mapping the direction vector $q_k = [\ cos\theta_k, sin\theta_k, 0]^{\mathrm{T}}$ to the distance vector d_k is:

$$\gamma(s_k, s_{k+1}) = sign(\langle q_k, d_k \rangle) \approx \frac{\kappa \langle q_k, d_k \rangle}{1 + |\kappa \langle q_k, d_k \rangle|} \tag{21}$$

where $<.,.>$ is the scalar computor. The angular velocity is $w_k = \Delta\theta_k / \Delta T_k$ and $|w_k| \le w_{max}$ the conditions are satisfied. $w_{max} = v_{max} p^{-1}{}_{min}$. Linear acceleration is described as:

$$a_k = \frac{2(v_{k+1} - v_k)}{\Delta T_k + \Delta T_{k+1}} \tag{22}$$

where linear velocity and angular velocity constraints are $v_k(s_{k+1}, s_k, \Delta T_k) = [v_{max} - |v_k|, w_{max} - |w_k|]^{\mathrm{T}}$, the inequality is $a_k(s_{k+2}, s_{k+1}, \Delta T_{k+1}, \Delta T_k) = a_{max} - |a_k|$. The precise nonlinear programming is transformed into an approximate nonlinear square optimization problem. The optimization of the solution formula is achieved through the use of the approximate least squares method. The solver's properties are harnessed to approximate the first derivative, ensuring an effective solution. The equality constraint, denoted as h, is represented by the quadratic penalty with a scalar weight σ:

$$\phi(h_k, \sigma_h) = \sigma_h h_k^T I h_k = \sigma_h \|h_k\|_2^2 \tag{23}$$

The inequality approximates the weighted unilateral quadratic penalty:

$$\chi(v_k, \sigma_v) = \sigma_v \|\min\{0, v_k\}\|_2^2 \tag{24}$$

Equation (24) can be approximated by the overall unconstrained optimization of the objective function $\widetilde{V}(B)$:

$$B^* = \arg \min_{B \setminus \{s_1, s_n\}} \widetilde{V}(B) \tag{25}$$

$$\widetilde{V}(B) = \sum_{k=1}^{n-1} [\Delta T_k^2 + \phi(h_k, \sigma_h) + \chi(\widetilde{r_k}, \sigma_r) + \chi(\upsilon_k, \sigma_\upsilon) + \chi(o_k, \sigma_o) + \chi(\alpha_k, \sigma_\alpha)] + \chi(\alpha_n, \sigma_\alpha) \tag{26}$$

where B* is the optimal solution vector. When ownership values all approach infinity, B* coincides with the actual minimum value of Eq. (25). In order to solve (Eq. 12), TEB algorithm adopts Levenberg Marquardt (LM) method. The graph optimization framework g2o implements an efficient sparse variant of LM.

Algorithm fusion

Combining the global and local path planning algorithms creates a hybrid algorithm. The global path planning involves multiple steps: firstly, employing an adaptive sampling function to generate random sampling points; secondly, generating a new node through node bias and adding it to the random tree; thirdly, removing redundant points in the initial path; fourthly, imposing constraints on the path turning angle and smoothing the path. The resultant global path information is then fed into the local path planning algorithm. The local path planning process consists of obtaining the path, applying kinematics constraints to the trajectory, verifying the trajectory, finally calculating the control inputs. This iterative process continues until the mobile robot reaches the target point, as depicted in Fig. 7.

Experimental

To evaluate the feasibility and effectiveness of the enhanced algorithm described in this paper, the Robot Operating System (ROS) serves as the operational environment. The simulation experiment takes place on a computer, and the prototype experiment is implemented on the mobile robot, enabling a thorough assessment.

The model is constructed using Gazebo, a sophisticated robot simulation software renowned for its high-fidelity physical simulation capabilities. Gazebo excels in accurately and efficiently replicating the intricate operations of robots within diverse indoor and outdoor settings. As a simulator, it swiftly validates the algorithm's effectiveness by creating a Gazebo model based on the experimental map. Following the construction of the simulation model, the next step involves employing the gmapping technique. This step is crucial for validating the improved RRT* algorithm's effectiveness. To thoroughly assess its performance, the A* algorithm, Voronoi algorithm, RRT algorithm, RRT connect algorithm, B spline smooth RRT connect algorithm, RRT* algorithm, B spline smooth RRT* algorithm, and the enhanced RRT* algorithm are individually executed within this mapped environment. The trajectory, depicted in Fig. 8, delineates specific elements: the gray area signifies safety, while the black cylinder represents obstacles, and the blue border denotes obstacle expansion. The mobile robot is denoted by the orange dot, and the red path illustrates the route planned by the global algorithm. Corresponding data is presented in Table 1 and Fig. 9. Notably, the path generated by the improved RRT* algorithm, as demonstrated in Fig. 8, exhibits a smoother trajectory compared to traditional algorithms. It significantly reduces the number of turning points, effectively enhancing the overall path quality. Further analysis, as evidenced in Table 1 and Fig. 9, reveals that compared to the traditional RRT* algorithm, the improved version reduces the path length by 5.8% and decreases the number of turning points by 62.5%. This reduction in path length enables the mobile robot to reach its destination more swiftly, while fewer waypoints conserve the robot's memory usage.

The resulting trajectory aligns with expectations. As illustrated in Fig. 10, The angular velocity and linear velocity of various RRT* algorithms applied to mobile robots are obtained and compared. In Fig. 10, the linear velocity consistently remains within the range of [−0.5 m/s, 1 m/s], The angular velocity also remains within the range of [-1 rad/s, 1 rad/s]. Of particular note is that only the improved algorithm maintains a constant speed of 1 m/s between [5 s, 20 s], indicating a smooth operation of the mobile robot during this time interval. Compared

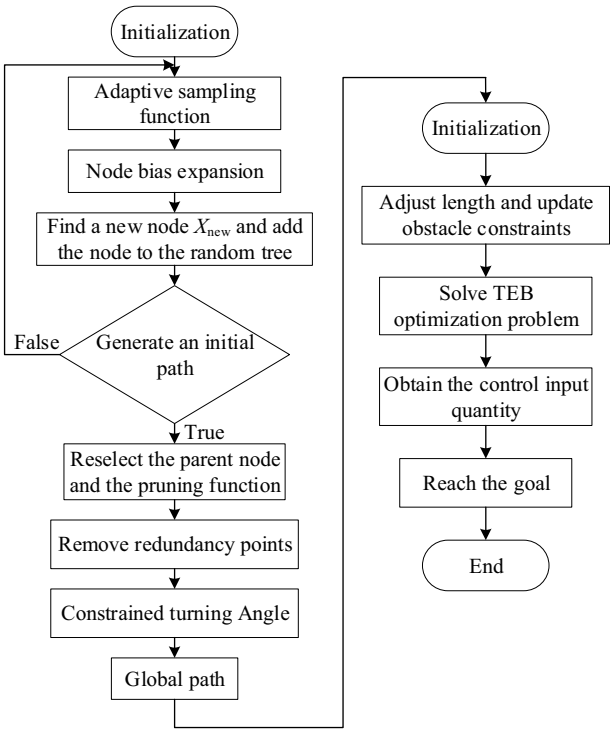

Figure 7. Fusion algorithm flow chart.

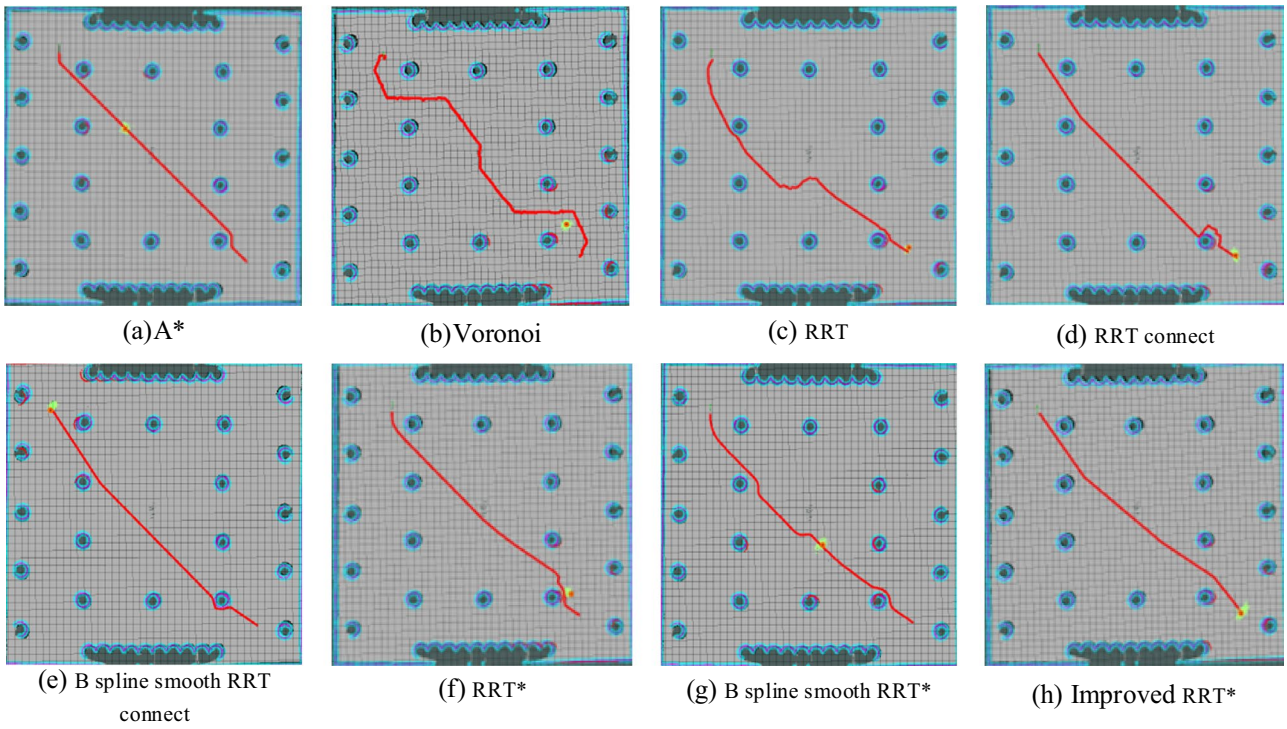

Figure 8. Algorithm comparison diagram.

Algorithm	Path length(m)	Number of nodes	Time to plan a path (s)
A*	20.21	291	384.93
Voronoi	27.20	461	97.09
RRT	22.53	173	70.99
RRT connect	23.11	159	61.26
B spline smooth RRT connect	21.43	963	158.07
RRT*	20.72	86	49.10
B spline smooth RRT*	22.97	749	76.1
Improved RRT*	19.52	771	52.10

Table 1. Comparison of algorithm performance indexes before and after improvement.

with other algorithms, the angular velocity fluctuation range of the improved algorithm is the smallest in the period of [5 s,18 s]. These results confirm that the improved algorithm is more consistent with the kinematic model of the mobile robot, and can follow the planned path more effectively and move more smoothly. As can be seen from Fig. 11, in the environment of Fig. 8, the optimization convergence speed of the three algorithms is relatively close, but the Improved RRT* algorithm has the best curve, which can effectively improve the efficiency of path planning and reduce the time of path planning.

As can be seen from Fig. 12, the Improved RRT* is put into the map of three different environments for simulation experiments, and the Improved RRT* successfully generates the optimal path without colliding with obstacles. The results show that the Improved RRT* is relatively robust and can successfully plan paths and avoid obstacles in different environments. In Fig. 13, the influence of Angle deviation threshold A_d on the number of path turns is analyzed, and it is found that setting the Angle at about π/20 degrees generates fewer path turns and smoother path. In Fig. 14, the relationship between the distance between the waypoints and the path length in the improved RRT*algorithm is analyzed. As shown in the figure, when the distance between the waypoints reaches 0.025 m, the impact on the distance is small, so 0.025 m is selected as the path point distance of the algorithm.

The hybrid algorithm is implemented in the mobile robot, and its effectiveness is verified through experimentation in a controlled environment, as illustrated in Fig. 15. Within the constructed environment, a hybrid algorithm is employed for path planning, and the outcome is depicted in Fig. 16. In Fig. 16b, point a designates the initial position of the mobile robot, point b represents the intermediate node during the robot's movement, and point c signifies the ultimate goal of the mobile robot. The purple path illustrates the global trajectory generated by the enhanced RRT* algorithm, while the red path depicts the local trajectory produced by the optimized

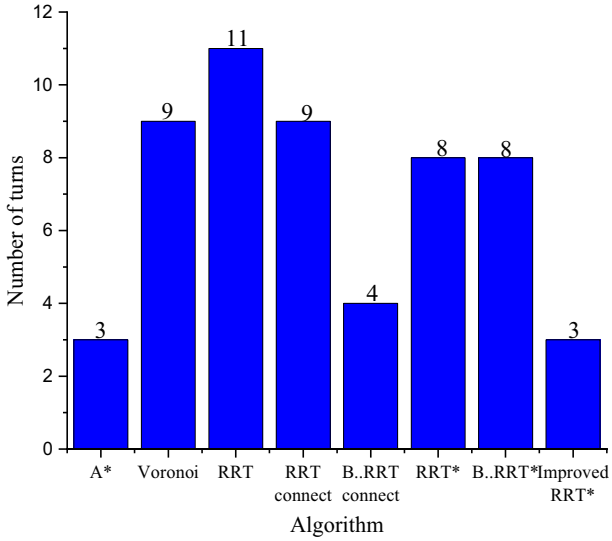

Figure 9. Number of turns diagram.

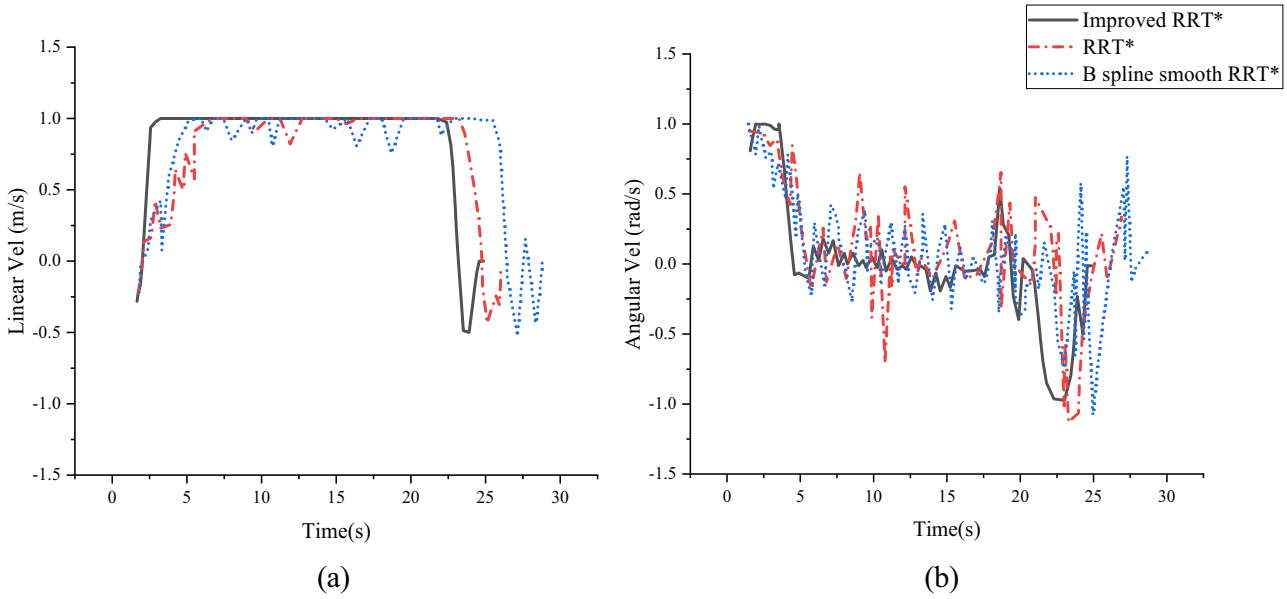

Figure 10. Simulation model velocity diagram.

TEB algorithm. Using this algorithm for path planning guarantees a smooth trajectory that matches the robot's kinematic traits. Crucially, it avoids collisions with obstacles, achieving the intended result.

Conclusion

Within this paper, introduce a fusion algorithm that synergizes the advancements of the improved RRT* algorithm and the TEB algorithm. The proposed approach not only expedites the generation of an optimal global path but also refines the trajectory, taking into careful consideration the dynamic constraints inherent to the mobile robot. The global path planning algorithm commences by employing an adaptive sampling function. Subsequently, Afterward, nodes expand with bias, redundant points are removed from the path, and turning angles are constrained. Perform kinematic modeling for the mobile robot. Incorporate dynamic constraints specific to the mobile robot into the local path planning algorithm, effectively integrating the algorithms. This holistic approach empowers the algorithm to strategize the mobile robot's path, leading to the attainment of a path of high quality.

In forthcoming research efforts, the emphasis transitions to utilizing a fusion algorithm for path planning within the framework of multiple robots. This necessitates addressing concerns such as priority assignment and potential conflicts in paths among the robots. The collaboration of multiple robots holds the promise of extending

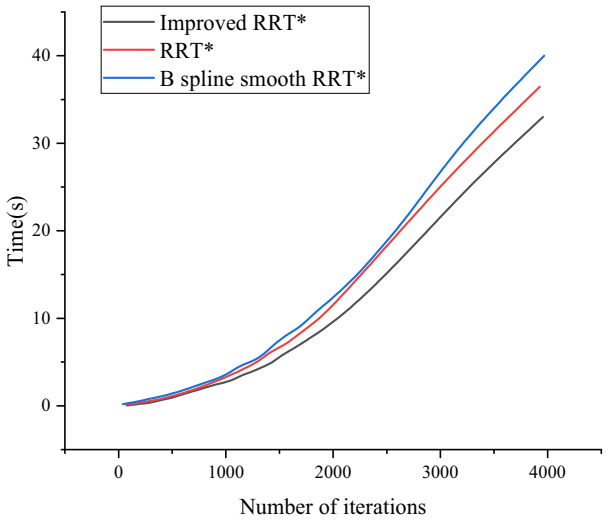

Figure 11. Time and number of iterations.

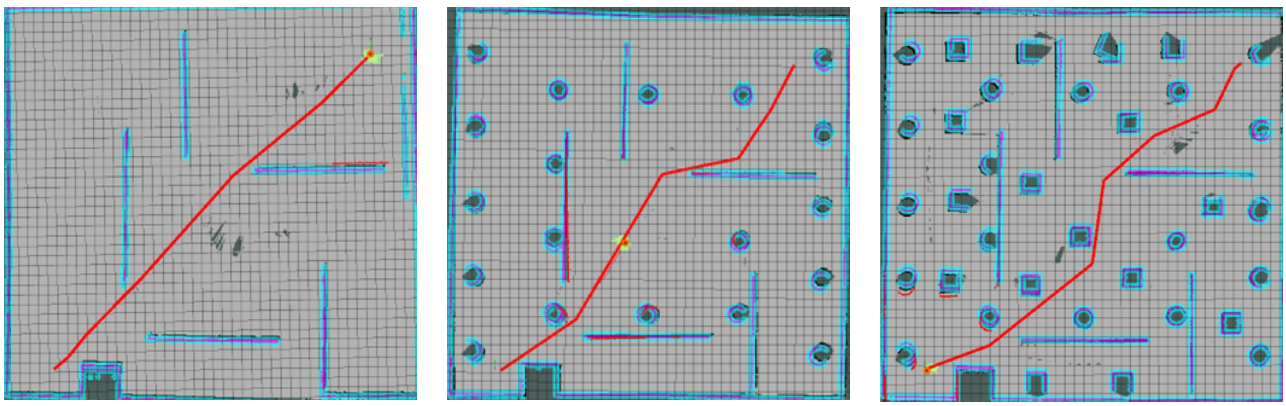

Figure 12. Algorithm simulation diagram under multiple maps.

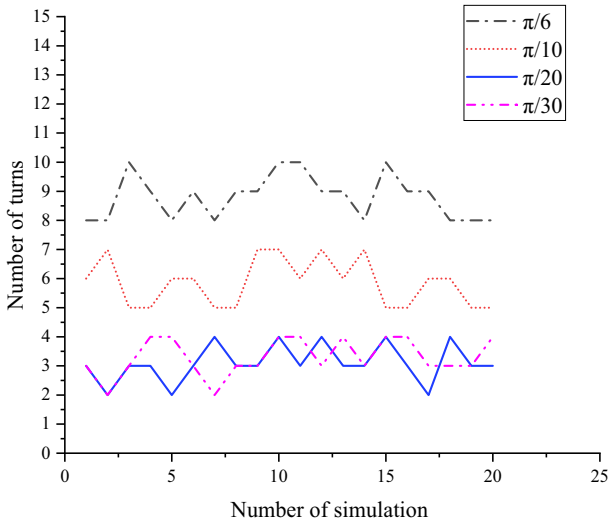

Figure 13. Influence of angle deviation threshold on improved RRT* algorithm.

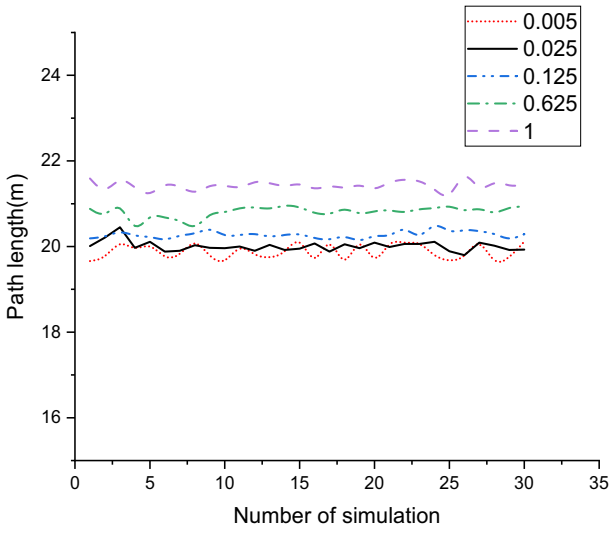

Figure 14. Influence of waypoint distance on improved RRT* algorithm.

Figure 15. Experiment environment diagram.

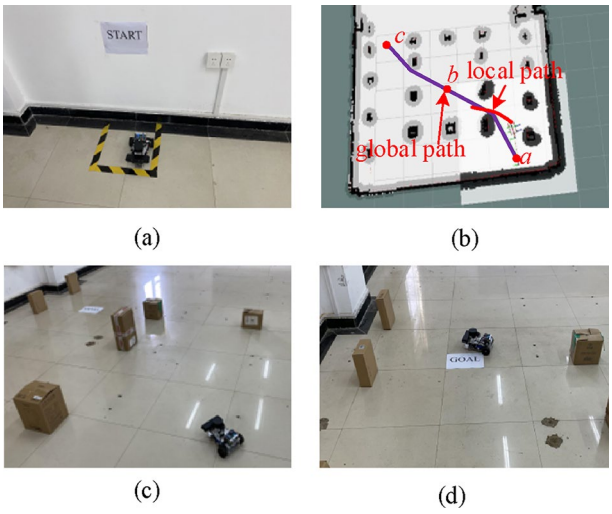

Figure 16. Mobile robot experiment diagram.

the operational radius and enhancing overall efficiency. Concurrently, there is an examination of autonomously assigning execution tasks and areas to mobile robots. This strategy is designed to enhance robot efficiency and markedly improve the quality of work. Furthermore, the research delves into investigating how mobile robots navigate and avoid dynamic obstacles.

Data availability

The datasets used and analysed during the current study available from the corresponding author on reasonable request.

Received: 10 July 2023; Accepted: 10 April 2024
Published online: 18 April 2024

References

1. Xiang, D., Lin, H., Ouyang, J. & Huang, D. Combined improved A* and greedy algorithm for path planning of multi-objective mobile robot. *Sci. Rep.* **12**(1), 13273 (2022).
2. Yin, X. *et al.* Dynamic Path Planning of AGV Based on Kinematical Constraint A* Algorithm and Following DWA Fusion Algorithms. *Sensors* **23**(8), 4102 (2023).
3. Zhong, X., Tian, J., Hu, H. & Peng, X. Hybrid path planning based on safe A* algorithm and adaptive window approach for mobile robot in large-scale dynamic environment. *J. Intell. Rob. Syst.* **99**, 65–77 (2020).
4. Chen, Z. *et al.* Patrol robot path planning in nuclear power plant using an interval multi-objective particle swarm optimization algorithm. *Appl. Soft Comput.* **116**, 108192 (2022).
5. Kashyap, A. K. & Parhi, D. R. Implementation of intelligent navigational techniques for inter-collision avoidance of multiple humanoid robots in complex environment. *Appl. Soft Comput.* **124**, 109001 (2022).
6. Gao, Y., Li, S., Wang, X., & Chen, Q. A patrol mobile robot for power transformer substations based on ROS. In *2018 Chinese Control and Decision Conference* (*CCDC*). 456–460 (IEEE, 2018).
7. Huang, S. K., Wang, W. J. & Sun, C. H. A new multirobot path planning with priority order based on the generalized Voronoi diagram. *IEEE Access* **10**, 56564–56577 (2022).
8. Li, F., Kim, Y. C., Lyu, Z. & Zhan, H. Research on path planning for robot based on improved design of non-standard environment map with ant colony algorithm. *IEEE Access* **11**, 99776–99791 (2023).
9. Zhang, W., Zhang, S., Wu, F. & Wang, Y. Path planning of UAV based on improved adaptive grey wolf optimization algorithm. *IEEE Access* **9**, 89400–89411 (2023).
10. Zheng, J., Ding, M., Sun, L. & Liu, H. Distributed stochastic algorithm based on enhanced genetic algorithm for path planning of multi-UAV cooperative area search. *IEEE Trans. Intell. Transp. Syst.* **24**(8), 8290–8303 (2023).
11. Cao, M., Zhou, X. & Ju, Y. Robot motion planning based on improved RRT algorithm and RBF neural network sliding. *IEEE Access* **11**, 121295–121305 (2023).
12. Daniali, S. M., Khosravi, A., Sarhadi, P. & Tavakkoli, F. An automatic parking algorithm design using multi-objective particle swarm optimization. *IEEE Access* **11**, 49611–49624 (2023).
13. Liu, E., Tan, R. & Chen, Y. Research on global ship path planning method based on improved ant colony algorithm. *J. East China Jiaotong University* **37**(6), 103–107 (2023).
14. Goricanec, J., Milas, A., Markovic, L. & Bogdan, S. Collision-free trajectory following with augmented artificial potential field using UAVs. *IEEE Access* **11**, 83492–83506 (2023).
15. Zhang, Z., Wang, S., Chen, J. & Han, Y. A bionic dynamic path planning algorithm of the micro UAV based on the fusion of deep neural network optimization/filtering and hawk-eye vision. *IEEE Trans. Syst. Man Cybern. Systems.* **53**(6), 3728–3740 (2023).
16. Yao, P., Wu, K. & Lou, Y. Path planning for multiple unmanned surface vehicles using Glasius bio-inspired neural network with Hungarian algorithm. *IEEE Syst. J.* **17**(3), 3906–3917 (2022).
17. Kumaar, A. N. & Kochuvila, S. Mobile service robot path planning using deep reinforcement learning. *IEEE Access* **11**, 100083–100096 (2023).
18. Zhang, R., Chai, R., Chai, S. & Tsourdos, A. Design and practical implementation of a high efficiency two-layer trajectory planning method for AGV. *IEEE Trans. Indus. Electron.* **71**(2), 1811–1822 (2023).
19. Dai, J., Zhang, Y. & Deng, H. Novel potential guided bidirectional RRT* with direct connection strategy for path planning of redundant robot manipulators in joint space. *IEEE Trans. Indus. Electron.* **71**(3), 2737–2747 (2023).
20. Mashayekhi, R., Idris, M. Y. I., Anisi, M. H., Ahmedy, I. & Ali, I. Informed RRT*-connect: An asymptotically optimal single-query path planning method. *IEEE Access* **8**, 19842–19852 (2020).
21. Liu, C., He, W., Cai, X., Xie, Z. *et al.* Laser slam-based autonomous navigation for fire patrol robots. In *2023 IEEE 2nd International Conference on Electrical Engineering, Big Data and Algorithms* (*EEBDA*). 742–746. (IEEE, 2023).
22. Wang, B., Ju, D., Li, Q. *et al.* SS-RRT*: A safe and smoothing path planner for mobile robot in static environment. In *2022 China Automation Congress* (*CAC*). 3677–3682. (IEEE, 2022).
23. Tang, G. *et al.* Geometric A-star algorithm: An improved A-star algorithm for AGV path planning in a port environment. *IEEE Access* **9**, 59196–59210 (2021).
24. Mashayekhi, R. *et al.* Hybrid RRT: A semi-dual-tree RRT-based motion planner. *IEEE Access* **8**, 18658–18668 (2020).
25. Zhu, D. D. & Sun, J. Q. A new algorithm based on Dijkstra for vehicle path planning considering intersection attribute. *IEEE Access* **9**, 19761–19775 (2021).
26. Du, Y., Zhang, X. & Nie, Z. A real-time collision avoidance strategy in dynamic airspace based on dynamic artificial potential field algorithm. *IEEE Access* **7**, 169469–169479 (2019).
27. Kobayashi, M. & Motoi, N. Local path planning: Dynamic window approach with virtual manipulators considering dynamic obstacles. *IEEE Access* **10**, 17018–17029 (2022).
28. Nakamura, T., Kobayashi, M. & Motoi, N. Path planning for mobile robot considering turnabouts on narrow road by deep Q-network. *IEEE Access* **11**, 19111–19121 (2023).
29. Lan, W. *et al.* Improved RRT algorithms to solve path planning of multi-glider in time-varying ocean currents. *IEEE Access* **9**, 158098–158115 (2021).
30. Wang, B. *et al.* CAF-RRT*: A 2D path planning algorithm based on circular arc fillet method. *IEEE Access* **10**, 127168–127181 (2022).
31. Chen, L. *et al.* A fast and efficient double-tree RRT $^* $-like sampling-based planner applying on mobile robotic systems. *IEEE/ASME Trans. Mechatron.* **23**(6), 2568–2578 (2018).
32. Rösmann, C., Feiten, W., Wösch, T. *et al.* Trajectory modification considering dynamic constraints of autonomous robots. In *ROBOTIK 2012; 7th German Conference on Robotics.* 1–6. (VDE, 2012).
33. Rösmann, C., Feiten, W., Wösch, T. *et al.* Efficient trajectory optimization using a sparse model. In *2013 European Conference on Mobile Robots.* 138–143 (IEEE, 2013).

34. Rösmann, C., Hoffmann, F., & Bertram, T. Kinodynamic trajectory optimization and control for car-like robots. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems* (*IROS*). 5681–5686. (IEEE, 2017).
35. Rösmann, C., Hoffmann, F., & Bertram, T. Planning of multiple robot trajectories in distinctive topologies. In *2015 European Conference on Mobile Robots* (*ECMR*). 1–6. (IEEE, 2015).
36. Shi, K., Wu, P., & Liu, M. Research on path planning method of forging handling robot based on combined strategy. In *2021 IEEE International Conference on Power Electronics, Computer Applications* (*ICPECA*). 292–295. (IEEE, 2021).
37. Sui, F. *et al.* ACO+ PSO+ A*: A bi-layer hybrid algorithm for multi-task path planning of an AUV. *Comput. Ind. Eng.* **175**, 108905 (2023).
38. Liu, E., Tan, R. & Chen, Y. Path planning of intelligent line patrol Robot based on improved human ant colony. *J. East China Jiaotong Univ.* **37**(06),103–107 (2019).

Acknowledgements
This work was supported in part by the National Natural Science Foundation of China under Grant: 52365003, 52165069, 52367015; Jiangxi Provincial Natural Science Foundation under Grants: 20232BAB214045, 20224BAB214051, 20224BAB204051, 20232BAB214064; Training Plan for Academic and Technical Leaders of Major Disciplines in Jiangxi Province under Grants:20232BCJ23027; Key R & D plan of Jiangxi Province under Grant: 20212BBE51010; Jiangxi Graduate Student Innovation Special Fund Project: YC2023-S468.

Author contributions
All authors contributed to the study conception and design. Xiong Yin wrote the manuscript and did the research. Technical support was provided by Wentao Dong, Xiaoming Wang, Yongxiang Yu. Valuable comments on manuscript revisions were put forward by Daojin Yao. All authors read and approved the final manuscript.

Competing interests
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to D.Y.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.