Sains Malaysiana 53(12)(2024): 3409-3423
http://doi.org/10.17576/jsm-2024-5312-22

Multi-Robot Path Planning Based on the Improved Nutcracker Optimization Algorithm and the Dynamic Window Approach

(Perancangan Laluan Berbilang Robot Berdasarkan Algoritma Pengoptimuman *Nutcracker* yang Diperbaiki dan Pendekatan Tetingkap Dinamik)

JIANGRONG ZHAO[1], HONGWEI DING[1,*], YUANJING ZHU[2], ZHIJUN YANG[1,3,4], PENG HU[5] & ZONGSHAN WANG1

[1]*School of Information Science and Engineering, Yunnan University, Kunming, China*
[2]*College of science and engineering, Dianchi College of Yunnan University, Kunming, China*
[3]*Key Laboratory of Educational Informatization for Nationalities, Yunnan Normal University, Kunming, China*
[4]*Educalion instruments and Facilities Service Center, Educational Department of Yunnan Province, Kunming, China*
[5]*Youbei Technology Co., Ltd, Kunming, China*

Received: 6 August 2024/Accepted: 11 October 2024

ABSTRACT

Multi-robot path planning faces challenges such as conflict avoidance, collaboration, and dynamic environments. This paper proposes a multi-robot path planning algorithm that integrates the improved nutcracker optimization algorithm with the improved dynamic window approach. To address the nutcracker algorithm's sensitivity to initial conditions and slow convergence, a population initialization strategy is introduced for more diverse initial populations. Additionally, a simplified path node strategy is also designed to shorten paths and reduce steering times. By incorporating a dynamic inertia weight factor w, the balance between global exploration and local optimization is improved. To address the limitations of the dynamic window approach, which is unable to avoid dynamic obstacles instantly and is prone to falling into local optimal solutions, the target distance subfunction, the path evaluation subfunction and the deviation from danger zone subfunction are added to the evaluation function. Finally, the two algorithms were fused together and we conducted four experiments to validate the performance of the MANOA, IDWA, and MANOA-IDWA algorithms, as well as the application of MANOA-IDWA in multi-robot path planning. Results show that MANOA-IDWA significantly increases path planning success rates in dynamic environments, producing shorter and smoother paths, thus enhancing the safety and stability of multi-robot operations.

Keywords: Dynamic window approach; fusion algorithm; multi-robot path planning; nutcracker optimization algorithm

ABSTRAK

Perancangan laluan berbilang robot menghadapi cabaran seperti mengelakkan konflik, kolaborasi dan persekitaran dinamik. Kertas ini mencadangkan algoritma perancangan laluan berbilang robot yang mengintegrasikan algoritma pengoptimuman *nutcraker* yang dipertingkatkan dengan pendekatan tetingkap dinamik yang dipertingkatkan. Untuk menangani kepekaan algoritma *nutcracker* kepada keadaan awal dan penumpuan perlahan, strategi pemula populasi diperkenalkan untuk populasi awal yang lebih pelbagai. Selain itu, strategi nod laluan yang dipermudahkan juga direka bentuk untuk memendekkan laluan dan mengurangkan masa stereng. Dengan menggabungkan faktor berat inersia dinamik w, keseimbangan antara penerokaan global dan pengoptimuman tempatan dipertingkatkan. Untuk menangani batasan pendekatan tetingkap dinamik yang tidak dapat mengelakkan halangan dinamik serta-merta dan terdedah kepada penyelesaian optimum tempatan, subfungsi jarak sasaran, subfungsi penilaian laluan dan sisihan daripada subfungsi zon bahaya ditambahkan pada fungsi penilaian. Akhirnya, kedua-dua algoritma telah digabungkan bersama dan kami menjalankan empat uji kaji untuk mengesahkan prestasi algoritma MANOA, IDWA dan MANOA-IDWA serta aplikasi MANOA-IDWA dalam perancangan laluan berbilang robot. Keputusan menunjukkan bahawa MANOA-IDWA dengan ketara meningkatkan kadar kejayaan perancangan laluan dalam persekitaran dinamik, menghasilkan laluan yang lebih pendek dan lancar, sekali gus meningkatkan keselamatan dan kestabilan operasi berbilang robot.

Kata kunci: Algoritma gabungan; algoritma pengoptimuman *nutcracker*; pendekatan tetingkap dinamik; perancangan laluan berbilang robot

INTRODUCTION

The ongoing advancement of robotics technology has rendered it challenging for a single mobile robot to accomplish complex work tasks autonomously. Consequently, the concept of multi-robot systems has emerged. The process of multi-robot path planning entails the determination of the optimal route for a robot from its point of origin to its destination. Multi-robot systems require effective communication and coordination to share information, collaborate, and avoid conflicts (Lin et al. 2022; Madridano et al. 2021). Robots must also adapt in real-time to dynamic environments, making rapid path adjustment to avoid collisions a critical challenge. Therefore, high-performance path planning algorithms are essential for multi-robot systems. Robot path planning algorithms are usually divided into two parts: Global path planning and local path planning (Jian et al. 2021).

In recent years, scholars both domestically and internationally have used metaheuristic algorithms for global path planning. Metaheuristic algorithms generally possess strong global optimization capabilities. These algorithms effectively explore the solution space and improve the quality and convergence speed of solutions by simulating natural phenomena or social behaviors. Common metaheuristic algorithms include the firefly algorithm (Yang 2009; Yu et al. 2021) (FA), the cuckoo search algorithm (Kanoon, Al-Araji & Abdullah 2022)(CS), the sparrow search algorithm (Xue & Shen 2020)(SSA). However, some drawbacks still exist such as difficulty in parameter setting, high consumption of computational resources, and excessive memory usage. Therefore, several scholars have enhanced the conventional metaheuristic algorithms. Chen et al. (2019) introduced two innovative and effective strategies, namely Levy flight and chaotic local search, simultaneously to better balance its global and local search capabilities in complex environments in WOA. Liu et al. (2021) improved the SSA algorithm for optimizing UAV routes and avoiding obstructions and collisions within a given flight area by introducing a chaotic strategy, adaptive inertia weights, and a Cauchy-Gaussian mutation strategy. Wang et al. (2016) proposed the NAFA algorithm to address excessive gravitational attraction, which can cause oscillations and increase computational time complexity during the search process of FA. The nutcracker optimization algorithm (Ida Evangeline et al. 2024) (NOA) is a new metaheuristic algorithm and was proposed in 2023 by Abdel-Basset et al. (2023) based on two different behaviors of nutcracker. Twenty-three standard functions, test suites of CEC2014, CEC-2017, and CEC-2020 were employed in this work to evaluate the performance of NOA. The experimental results show that NOA can well balance global exploration and local exploitation, with faster convergence and better adaptation. Common local path planning methods include the potential field method (Ge & Cui 2002), the dynamic window method (Han et al. 2022)(DWA), and sampling-based methods (Karaman & Frazzoli 2011). Among these methods, DWA stands out due to its high real-time performance, adjustable parameters and consideration of robot dynamic constraints.

In dynamic environments, multiple robots encounter moving obstacles and changing terrain, complicating path planning. The improved metaheuristic algorithms can effectively plan collision-free paths in static conditions. However, increased obstacle coverage or dynamic obstacles significantly raises the likelihood of robots getting trapped in local optima and reduces path planning success rates. To facilitate effective conflict avoidance and coordination among multiple robots, greater algorithm complexity is required. After comprehensive consideration, this paper proposes a path planning method that combines the multi-strategy adaptive nutcracker optimization algorithm (MANOA), with the improved dynamic window approach (IDWA).

THE TRADITIONAL NUTCRACKER OPTIMIZATION ALGORITHM

The foraging and storage strategy as well as the cache search and retrieval strategy are proposed to mimic nutcrackers' behaviors.

Foraging and storage strategy

During this phase, NOA balances global exploration with local exploitation based on probability P_{a1}. The formulas for updating the location during this phase are as follows:

$$\vec{X}_i^{t+1} = \begin{cases} X_{i,j}^t & \tau_1 < \tau_2 \\ \begin{cases} X_{m,j}^t + \gamma \times (X_{A,j}^t - X_{B,j}^t) + \mu \times (r^2 \times U_j - L_j) & t \le \frac{T_{\max}}{2.0} \\ X_{C,j}^t + \mu \times (X_{A,j}^t - X_{B,j}^t) + \mu \times (r_1 < \delta) \times (r^2 \times U_j - L_j) & otherwise \end{cases} & otherwise \end{cases} \quad (1)$$

$$\mu = \begin{cases} \tau_3 & r_1 < r_2 \\ \tau_4 & r_2 < r_3 \\ \tau_5 & r_1 < r_3 \end{cases} \quad (2)$$

$$\vec{X}_i^{t+1} = \begin{cases} \vec{X}_i^t + \mu \times (\vec{X}_{best}^t - \vec{X}_i^t) \times |\lambda| + r_1 \times (\vec{X}_A^t - \vec{X}_B^t) & \tau_1 < \tau_2 \\ \vec{X}_{best}^t + \mu \times (\vec{X}_A^t - \vec{X}_B^t) & \tau_1 < \tau_3 \\ \vec{X}_{best}^t \times l & otherwise \end{cases} \quad (3)$$

$$\vec{X}_i^{t+1} = \begin{cases} Eq.(1) & \phi > P_{a_1} \\ Eq.(3) & otherwise \end{cases} \quad (4)$$

where $\vec{X}_i^{t+1}$ represents the i-th nutcracker's position at the $(t+1)$-th iteration; $X_{i,j}^t$ represents the i-th nutcracker's j-dimensional position at the t-th iteration; $X_{m,j}^t$ represents the mean of all nutcrackers' j-dimensional position at the t-th iteration; $\vec{X}_{best}^t$ represent the optimal nutcracker's position at the t-th iteration; A, B, and C represent three randomly selected nutcrackers without duplication; U_j and L_j are the upper bound as well as the lower bound of the j-dimensional position; γ , λ and τ_5 are random numbers generated by levy flight; τ_1, τ_2, τ_3, r, r_1, r_2, r_3, and ϕ are uniformly distributed over the interval [0,1]; τ_4 is a random number that follows a normal distribution; the value of δ is 0.05; P_{a1} and l are variables that decrease linearly with the number of iterations.

During the summer and autumn, nutcrackers collect and store pine nuts. Their location information is generated through one iteration of either Equations (1) or (3). Equation (1) corresponds to the foraging strategy, enabling global exploration to identify regions that may contain near-optimal solutions. Equation (3) corresponds to the storage strategy, which focuses solutions in the regions that contain near-optimal solutions to enhance coverage.

Cache-search and recovery strategy

During this phase, NOA balances global exploration with local exploitation based on probability P_{a2}.The formulas for updating the location during this phase are as follows:

$$RPs = \begin{bmatrix} \overrightarrow{RP}^t_{1,1} & \overrightarrow{RP}^t_{1,2} \\ \vdots & \vdots \\ \overrightarrow{RP}^t_{i,1} & \overrightarrow{RP}^t_{i,2} \\ \vdots & \vdots \\ \overrightarrow{RP}^t_{N,1} & \overrightarrow{RP}^t_{N,2} \\ \vdots & \vdots \end{bmatrix} \tag{5}$$

$$\overrightarrow{RP}^t_{i,k} = \overrightarrow{X}^t_i + \alpha \times \cos(\theta) \times (\overrightarrow{X}^t_A - \overrightarrow{X}^t_B), K = 1 \tag{6}$$

$$\overrightarrow{RP}^t_{i,k} = \overrightarrow{X}^t_i + \alpha \times \cos(\theta) \times ((\overrightarrow{U} - \overrightarrow{L}) \times \tau_3 + \overrightarrow{L}) \times \overrightarrow{U_2}, K = 2 \tag{7}$$

$$\overrightarrow{U_2} = \begin{cases} 1 & \overrightarrow{r_2} < P_{rp} \\ 0 & otherwise \end{cases} \tag{8}$$

$$\overrightarrow{RP}^t_{i,1} = \begin{cases} \overrightarrow{X}^t_i + \alpha \times \cos(\theta) \times (\overrightarrow{X}^t_A - \overrightarrow{X}^t_B) + \alpha \times RP & \theta = \dfrac{\pi}{2} \\ \overrightarrow{X}^t_i + \alpha \times \cos(\theta) \times (\overrightarrow{X}^t_A - \overrightarrow{X}^t_B) & otherwise \end{cases} \tag{9}$$

$$\overrightarrow{RP}^t_{i,2} = \begin{cases} \overrightarrow{X}^t_i + (\alpha \times \cos(\theta) \times ((\overrightarrow{U} - \overrightarrow{L}) \times \tau_3 + \overrightarrow{L}) + \alpha \times RP) \times \overrightarrow{U_2} & \theta = \dfrac{\pi}{2} \\ \overrightarrow{X}^t_i + \alpha \times \cos(\theta) \times ((\overrightarrow{U} - \overrightarrow{L}) \times \tau_3 + \overrightarrow{L}) \times \overrightarrow{U_2} & otherwise \end{cases} \tag{10}$$

$$\alpha = \begin{cases} (1 - \dfrac{t}{T_{\max}})^{2\frac{t}{T_{\max}}} & r_1 > r_2 \\ (\dfrac{t}{T_{\max}})^{\frac{2}{t}} & otherwise \end{cases} \tag{11}$$

$$\overrightarrow{X}^{t+1}_i = \begin{cases} \overrightarrow{X}^t_i & f(\overrightarrow{X}^t_i) < f(\overrightarrow{RP}^t_{i,1}) \\ \overrightarrow{RP}^t_{i,1} & otherwise \end{cases} \tag{12}$$

$$X^{t+1}_{i,j} = \begin{cases} X^t_{i,j} & \tau_3 < \tau_4 \\ X^t_{i,j} + r_1 \times (X^t_{best,j} - X^t_{i,j}) + r_2 \times (\overrightarrow{RP}^t_{i,1} - X^t_{C,j} & otherwise \end{cases} \tag{13}$$

$$\overrightarrow{X}^{t+1}_i = \begin{cases} \overrightarrow{X}^t_i & f(\overrightarrow{X}^t_i) < f(\overrightarrow{RP}^t_{i,2}) \\ \overrightarrow{RP}^t_{i,2} & otherwise \end{cases} \tag{14}$$

$$X^{t+1}_{i,j} = \begin{cases} X^t_{i,j} & \tau_5 < \tau_6 \\ X^t_{i,j} + r_1 \times (X^t_{best,j} - X^t_{i,j}) + r_2 \times (\overrightarrow{RP}^t_{i,2} - X^t_{C,j} & otherwise \end{cases} \tag{15}$$

$$\overrightarrow{X}^{t+1}_i = \begin{cases} Eq.(13) & \tau_7 < \tau_8 \\ Eq.(15) & otherwise \end{cases} \tag{16}$$

$$\overrightarrow{X}^{t+1}_i = \begin{cases} Eq.(12) & f(Eq.(12)) < f(Eq.(14)) \\ Eq.(14) & otherwise \end{cases} \tag{17}$$

$$\overrightarrow{X}^{t+1}_i = \begin{cases} Eq.(16) & \phi < P_{a_2} \\ Eq.(17) & otherwise \end{cases} \tag{18}$$

where $\overrightarrow{RP}^t_{i,1}$ and $\overrightarrow{RP}^t_{i,2}$ represent *RPs* of the i-th nutcracker in the current generation t; α is a variable that decreases linearly with the number of iterations; θ is a random radian between 0 and π; r_2 is a vector that includes values randomly generated between 0 and 1; P_{rp} takes the value of 0.2; RP is a random position; $T_{\max}$ represents the maximum number of iterations; τ_3 , τ_4 , τ_5 , τ_6 , τ_7 , τ_8 , r_1 , r_2 and ϕ are uniformly distributed over the interval [0,1]; P_{a2} takes the value of 0.4.

In winter and spring, nutcrackers search for and consume stored pine nuts. Equations (5) to (11) define the reference positions of nutcrackers. Equation (17) corresponds to the cache-search strategy, designed to quickly guide the algorithm toward the near-optimal solution by exploring regions near the reference positions (RPs). Equation (16) corresponds to the recovery strategy, allowing NOA to perform fine searches within the near-optimal region while also enabling global searches to avoid local minimum.

Implementation of NOA

$$\overrightarrow{X}^t_{i,j} = +(\overrightarrow{U}_j - \overrightarrow{L}_j) \times \overrightarrow{RM} + \overrightarrow{L}_j, I = 1,2,\ldots,N, j = 1,2,\ldots,D \tag{19}$$

$$\overrightarrow{X}^{t+1}_i = \begin{cases} \overrightarrow{X}^{t+1}_i & f(\overrightarrow{X}^{t+1}_i) < f(\overrightarrow{X}^t_i) \\ \overrightarrow{X}^t_i & otherwise \end{cases} \tag{20}$$

where $\overrightarrow{RM}$ is a random vector in the interval [0,1].

Equation (19) defines the initialization method for the nutcracker population. Equation (20) provides a method for identifying the precise location of the nutcracker following one iteration, based on its fitness value.

THE TRADITIONAL DYNAMIC WINDOW APPROACH

DWA aims to select the best speed and steering angle to avoid obstacles. Its evaluation function computes a score based on the robot's current state and the actions taken, assessing their effectiveness. The speed constraints and evaluation function in DWA are as follows:

$$\begin{cases} 0 \le v_t \le v_{\max} \\ v_t - a_{v\_\max}\Delta t \le v_{t+1} \le v_t + a_{v\_\max}\Delta t \\ -\omega_{\max} \le \omega_t \le \omega_{\max} \\ \omega_t - a_{\omega\_\max}\Delta t \le \omega_{t+1} \le \omega_t + a_{\omega\_\max}\Delta t \end{cases} \tag{21}$$

$$G(v,\omega) = \sigma(\alpha \times heading(v,\omega) + \beta \times dist(v,\omega) + \gamma \times vel(v,\omega)) \tag{22}$$

where $v_{\max}$ and $\omega_{\max}$ are the maximum value of the linear velocity and the angular velocity; $a_{v\_\max}$ and $a_{\omega\_\max}$ are the maximum linear acceleration and maximum angular acceleration.

IMPROVED NOA ALGORITHM

IMPROVED POPULATION INITIALIZATION STRATEGY

Starting from the beginning of the initial path, grids are continually inserted between adjacent grids to create a continuous path. If a grid and its eight surrounding grids are non-insertable, NOA will discard this path. However, some subsets of the population inadequately represent the solution space. To address this the paper introduces a new grid using the following strategy. The specific steps for inserting a grid are as follows:

Step 1 Generate a random number r uniformly distributed in [0,1], and let $c = 1$;

Step 2 Calculate the total number of rows and columns between the two grids. If $r \ge \frac{1}{2}$, proceed to step 3 or 5 based on the parity of $rows$; otherwise, proceed to step 4;

Step 3 Firstly, check if moving the grid with the larger serial number up by (c+1) units results in a position that is free of obstacles and within the map. If both conditions are met, insert the grid and exit the loop; otherwise, move the grid with the smaller serial number up by c units and re-evaluate the conditions. If both conditions are met, insert the grid and exit the loop; otherwise, proceed to step 10;

TABLE 1. The way to insert a new node

Random number *r*	Corresponding step	Is it true that the grid with the larger serial number also has a larger column?	rows	columns	A grid with a larger serial number / A grid with a smaller serial number
$r \ge \frac{1}{2}$	Step 3	—	Odd number	—	Move up（c+1）units / Move down c units
$r \ge \frac{1}{2}$	Step 5	—	Even number	—	Move up c units /Move down（c+1）units
$r < \frac{1}{2}$	Step 6	Yes	—	Odd number	Shift right（c+1）units / Shift left c units
$r < \frac{1}{2}$	Step 7	Yes	—	Even number	Shift right c units / Shift left（c+1）units
$r < \frac{1}{2}$	Step 8	No	—	Odd number	Shift left c units / Shift right（c+1）units
$r < \frac{1}{2}$	Step 9	No	—	Even number	Shift left（c+1）units / Shift right c units

Step 4 Firstly, check if the column of the grid with the larger serial number is larger than that of the smaller one. If this condition is met, proceed to step 6 or 7 based on the parity of *columns*; otherwise, proceed to step 8 or 9;

Steps 5 to 9 These steps correspond to the corresponding steps in Table 1. If the specific conditions are met, insert the grid; otherwise, proceed to step 10;

Step 10 Let $c = c + 1$ and proceed to step 2.

After inserting a new grid between the two existing grids, the algorithm assesses whether the first two and last two grids can generate a path. This enhanced strategy reduces the likelihood of discarding initial paths, prevents the algorithm from becoming trapped in local minima due to limited solution diversity, and results in shorter robot trajectories. Additionally, the algorithm's computational efficiency improves due to a more even distribution of initial solutions across the decision interval.

SIMPLIFIED PATH NODE STRATEGY

The path planned by NOA has too many grids, which increases the length of the trajectory and the number of steering operations, requiring more precise control. Therefore, this paper further processes the trajectory obtained after initialization.

It is necessary to clarify that nodes a, b and c represent three immediately adjacent nodes in the path, with node a is initially situated at the starting point. If b is at the endpoint, subsequent operations will not be executed. The main idea is to delete node b when the distances between the straight-line segments connecting a and c and the obstacles are within the safe range. The path is updated, keeping node a's position unchanged, and then assessing node b can be deleted; When the distances are not within the safety range, a takes the position of b, and the process continues to evaluate whether node b can be deleted.

ADD A DYNAMIC INERTIA WEIGHT FACTOR w

The inertia weight factor was originally proposed by Kennedy and Eberhart (1995) and is set to 1 for the position of the previous generation nutcracker during iterations. However, it must be adjusted based on specific circumstances to balance global exploration and local exploitation. This paper introduces a dynamic inertia weight factor w with chaotic perturbation, expressed as follows:

$$k(t) = |1 - 2 \times k(t-1)^2| \tag{23}$$

$$v(t) = (w_{\max} - w_{\min}) \times (1 - (\frac{t}{T})^{\frac{1}{4}} + |k(t)| \times (1 - (\frac{t}{T})^2 - (1 - \frac{t}{T})^{\frac{1}{4}})) + w_{\min} \tag{24}$$

$$w(t) = \begin{cases} \begin{cases} 1 + (1 - v(t)) & rand_1 \geq \frac{1}{2} \\ v(t) & othersize \end{cases} & v(t) \geq \frac{7 \times w_{\max}}{9} \\ v(t) & othersize \end{cases} \tag{25}$$

where $k(0) \neq \frac{1}{2}$; $w_{\max}$ and $w_{\min}$ represent the maximum and minimum values of the dynamic inertia weight factor w, with $w_{\max}$ is set to 0.9 and $w_{\min}$ is set to 0 in this work; $rand_1$ is uniformly distributed over the interval [0,1].

In early iterations, the position of the previous generation nutcracker is weighted more heavily to expand the NOA's search space. Occasionally, chaotic factors may reduce this weight, enabling local optimization. In later iterations, enabling local optimization, although chaotic factors may sometimes increase it to prevent NOA from falling into a local optimum.

EVALUATION METRICS

The evaluation function for assessing the performance of paths planned by MANOA incorporates three metrics, necessitating normalization due to differing bases for each indicator:

1. Path length: Reducing path length typically enhances efficiency, minimizing time and energy consumption;
2. Smoothness: This metric describes path coherence and is evaluated based on the curvature, the ratio of straight to curved segments, and the rate of change of the path segment;
3. Number of path nodes: An optimal number of nodes balances path accuracy and computational efficiency, assessed through factors such as the time required for path planning, the memory for storing path data, and the number of nodes.

IMPROVED DWA ALGORITHM

ADD A PATH EVALUATION SUBFUNCTION

In dynamic environments, globally planned paths may become invalid or unsafe. DWA is particularly suited for scenarios requiring rapid response and real-time adjustments (Yang et al. 2022), but its effectiveness is limited in complex environments due to window range constraints. By fusing MANOA and DWA together, the robot can adjust in real-time based on current environmental information.

To fuse the two algorithms, this paper introduces a path evaluation subfunction and the concepts of interim starting points and endpoints. DWA extracts critical nodes from the MANOA-planned path to create starting point and endpoint matrices. The robot follows the DWA-planned path from interim starting point a_1 to interim end point a_2. Reaching the point a_2 indicates that the robot has reached a new interim starting point, which is stored in the starting point matrix, and continues towards the new interim endpoint, which is stored in the endpoint matrix, until it reaches the final destination. The specific expression for the subfunction $dist\_line(v,\omega)$ is as follows:

$$d_0 = \frac{(\overrightarrow{goal_i} - \overrightarrow{start_i})}{\left|\overrightarrow{goal_i} - \overrightarrow{start_i}\right|} \tag{26}$$

$$d_1 = \vec{x}_{end} - \overrightarrow{start_i} \tag{27}$$

$$\vec{P} = \begin{cases} \overrightarrow{start_i} & \dfrac{d_0 \cdot d_1}{|d_0|} \le 0 \\ \overrightarrow{goal_i} & \dfrac{d_0 \cdot d_1}{|d_0|} \ge \left|\overrightarrow{goal_i} - \overrightarrow{start_i}\right| \\ \overrightarrow{start_i} + \dfrac{d_0 \cdot d_1}{|d_0|} \times d_0 & otherwise \end{cases} \tag{28}$$

$$dist\_line_0(v,\omega) = \begin{cases} r_{obstacle} \times 5 & \overrightarrow{start_i} = \vec{x}_{end} \\ \left|\vec{x}_{end} - \vec{P}\right| & otherwise \end{cases} \tag{29}$$

$$dist\_line(v,\omega) = \frac{\left|\overrightarrow{goal_i} - \overrightarrow{start_i}\right|}{3} - dist\_line_0(v,\omega) \tag{30}$$

where $\overrightarrow{start_i}$ and $\overrightarrow{goal_i}$ represent the current interim starting point and end point of the robot, corresponding to the i-th and $(i+1)$-th grid positions of the trajectory planned by NOA; x_{end} represents the position of the end point of the predicted trajectory; $r_{obstacle}$ represents the safety distance of the robot, which is set to 0.2 in this work.To ensure that the robot's next position aligns as closely as possible with the NOA-planned path, the distance $dist\_line_0$ between the predicted trajectory's endpoint at the forward prediction time Δt and the line connecting the start and end points should be small, while $dist\_line(v,\omega)$ should be large. So, the expression for $dist\_line(v,\omega)$ should be as shown in Equation (30). Specifically, when the end point coincides with the starting point (i.e., $dist\_line_0$ =0), this position's fitness is higher. Setting $dist\_line_0$ $=r_{obstacle} \times 5$ helps avoid the robot getting stuck at the starting point.

ADD A TARGET DISTANCE SUBFUNCTION

To prevent the increasing distance between the robot and the destination, and to minimize excessive path length due to obstacle avoidance, this paper adds a target distance subfunction $dist_{\text{goal}}(v,\omega)$ to the evaluation function. The specific expression of the subfunction is as follows:

$$dist_{\text{goal}}(v,\omega) = \left|\vec{x}_0 - \overrightarrow{goal_i}\right| \times \frac{3}{2} - \left|\vec{x}_{end} - \overrightarrow{goal_i}\right| \tag{31}$$

where $\vec{x}_0$ represents the current position of the robot.

The smaller the distance from the end of the predicted trajectory after the forward prediction time Δt, the larger $dist_{\text{goal}}(v,\omega)$ will become, which in turn affects the magnitude of the fitness of the candidate velocity vector (v,ω). The target distance subfunction $dist_{\text{goal}}(v,\omega)$ can continue to guide the robot to move towards the stage endpoint based on guiding the robot to bypass the obstacle, thus, improving the robustness and adaptability of the robot.

ADD A DEVIATION FROM THE DANGER ZONE SUBFUNCTION

The limited search range of DWA's linear and angular velocities makes it challenging for robots to effectively avoid dynamic obstacles. To enhance real-time responsiveness to environmental changes, this paper adds the deviation from the danger zone subfunction $devi\_danger(v,w)$ to the evaluation function. The specific expression of the subfunction is as follows:

$$brak\_dist_0 = \max\left(\frac{v_0^{\,2}}{2 \times a_{\max_v}}, r_{obstacle} \times 5\right) \tag{32}$$

$$dist_{danger\_zone\_j}(v,\omega) = \begin{cases} \left|\vec{x}_0 - \overrightarrow{obs_{0_j}}\right| & \left|\vec{x}_0 - \overrightarrow{obs_{0_j}}\right| > brak\_dist_0 \\ dist_{danger_1\_j}(v,\omega) & otherwise \end{cases} \tag{33}$$

$$angl_{danger\_zone\_j}(v,w) = \begin{cases} \dfrac{\pi}{2} & \left|\vec{x}_0 - \overrightarrow{obs_{0_j}}\right| > brak\_dist_0 \\ angl_{danger_1\_j}(v,w) & otherwise \end{cases} \tag{34}$$

$$\begin{aligned} devi\_danger(v,w) &= \beta_1 \times dist_{danger\_zone\_m}(v,\omega) \\ &+ \beta_2 \times angl_{danger\_zone\_m}(v,w) \end{aligned} \tag{35}$$

where v_0 represents the current velocity of the robot; $a_{\max_v}$ represents the maximum value of linear deceleration; obs_{0_j} represents the j-th obstacle's current position; $dist_{danger\_zone\_m}$ and $angl_{danger\_zone\_m}$ are the average values of the $dist_{danger\_zone\_j}(v,\omega)$ function and the $angl_{danger\_zone\_j}(v,w)$ function; both β_1 and β_2 are set to 0.5.

If the braking distance $brak\_dist_0$ is too small, the robot may incorrectly assume that numerous obstacles will not hinder its movement, resulting in DWA's inability to handle complex dynamic environments.

When $\left|\vec{x}_0 - \overrightarrow{obs_0}\right| > brak\_dist_0$, it indicates that the robot will not collide with this obstacle when it decelerates and moves. Therefore, it can be considered that the robot is not situated within the danger zone. Otherwise, it is not possible to determine whether the robot has entered the danger zone. The following will discuss the specific expression of this subfunction when $\left|\vec{x}_0 - \overrightarrow{obs}_0\right| \le brak\_dist_0$.

1. On the basis that $\left|\vec{x}_0 - \overrightarrow{obs}_0\right| \le brak\_dist_0$ and the position of the obstacle changes with time, the specific expressions for the $dist_{danger_1\_j}(v,\omega)$ function and the $angl_{danger_1\_j}(v,w)$ function are as follows:

$$\theta_{2_j} = \theta_{obstacle_j} - \pi \tag{36}$$

$$dist_{danger_1\_j}(v,\omega) = \begin{cases} dist_{danger_2\_j}(v,\omega) & \theta_{3_j} \in (\min(\theta_1, \theta_{2_j}), \max(\theta_1, \theta_{2_j})) \\ \left|\vec{x}_0 - \overrightarrow{obs}_{0_j}\right| & otherwise \end{cases} \tag{37}$$

$$angl_{danger_1\_j}(v,w) = \begin{cases} angl_{danger_2\_j}(v,w) & \theta_{3_j} \in (\min(\theta_1, \theta_{2_j}), \max(\theta_1, \theta_{2_j})) \\ \dfrac{\pi}{2} & otherwise \end{cases} \tag{38}$$

where $\theta_{obstacle_j}$ represents the j-th dynamic obstacle's heading angle; θ_1 represents the robot's heading angle moving along a candidate velocity vector; θ_{3_j} represents the angle between the j-th dynamic obstacle's current position and the horizontal direction; $\theta_{obstacle_j}, \theta_1, \theta_{2_j}, \theta_{3_j} \in [-\pi, \pi)$.

The central angle θ_{0_j} of the danger zone is defined by the moving direction of both the robot and the obstacle, the radius corresponds to the braking distance $brak\_dist_0$, with the vertex at the robot's current position. When the obstacle is within the danger zone, it indicates the robot may enter a danger zone. The purpose of this function is to reduce the risk of collision after a forward prediction time (Δt). We need to determine the positions and heading angles of the robot and the obstacle after a forward prediction time. The specific expressions for the $dist_{danger_2\_j}(v,\omega)$ function and the $angl_{danger_2\_j}(v,w)$ function are as follows:

$$brak\_dist_{end\_j} = \max\left(\frac{v_{end_j}^{\ 2}}{2 \times a_{\max_v}}, r_{obstacle} \times 5\right) \tag{39}$$

$$dist_{danger2\_j}(v,\omega) = \left|\vec{x}_{end} - \overrightarrow{obs}_{end_j}\right| \tag{40}$$

$$\theta_{6_j} = \theta_{obstacle_j} - \pi \tag{41}$$

$$\theta_{4_j} = \max(\theta_5, \theta_{6_j}) - \min(\theta_5, \theta_{6_j}) \tag{42}$$

$$angl_{danger_2\_j}(v,w) = \left|\frac{\theta_{4_j}}{2} + \min(\theta_5, \theta_{6_j}) - \theta_{7_j}\right| \tag{43}$$

where $\vec{x}_{end}$ represents the robot's position after Δt; $\overrightarrow{obs}_{end_j}$ represents the j-th dynamic obstacle's position after Δt; θ_{4_j} represents the central angle of the danger zone formed by the robot and the j-th dynamic obstacle after Δt; θ_5 represents the robot's heading angle after Δt; θ_{7_j} is the angle between the j-th dynamic obstacle's position and the horizontal direction after Δt; $\theta_{obstacle_j}, \theta_{4_j}, \theta_5, \theta_{6_j}, \theta_{7_j} \in [-\pi, \pi)$.

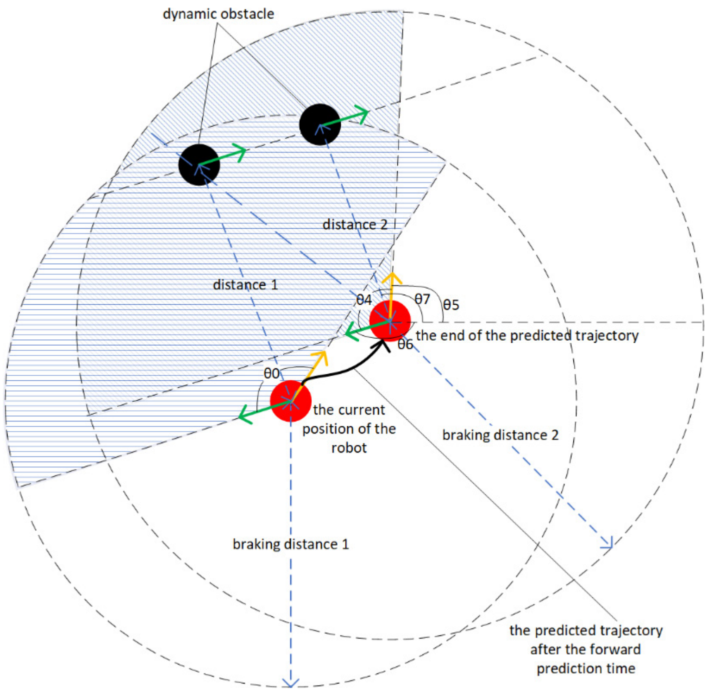

dynamic obstacle
distance 2
distance 1
θ4
θ7
θ5
θ6
the end of the predicted trajectory
θ0
the current position of the robot
braking distance 2
braking distance 1
the predicted trajectory after the forward prediction time

FIGURE 1. A robot that falls into the danger zone

If the obstacle is within the danger zone after the forward prediction time, it indicates that the robot is in the danger zone. The subfunction assigns a low fitness value to the candidate velocity vector that would lead the robot to that position, thereby preventing its selection in the iteration process.

2. In the map, there are also many static obstacles. So, on the basis of $|\vec{x}_0 - \vec{obs}_0| \le brak\_dist_0$, the specific expressions for the $dist_{danger_1\_j}(v,\omega)$ function and the $angl_{danger_1\_j}(v,w)$ function are as follows:

$$\theta_{2_j} = \theta_1 \tag{44}$$

$$dist_{danger_1\_j}(v,\omega) = |\vec{x}_0 - \overrightarrow{obs}_{0_j}| \tag{45}$$

$$angl_{danger_1\_j}(v,w) = \begin{cases} 0 & \theta_{3_j} = \theta_1 \\ \dfrac{\pi}{2} & otherwise \end{cases} \tag{46}$$

When the v in the candidate velocity vector points towards the obstacle, the robot is considered to have entered the danger zone. Introduction this subfunction into the evaluation function allows for real-time path adjustments when a high probability of collision is anticipated.

EVALUATION METRICS

The evaluation function of DWA is shown in Equation (22), while IDWA's evaluation function is presented in Equation (47). The original metrics are enhanced by adding the path evaluation, the target distance, and the deviation from the danger zone functions, significantly improving the robot's environmental perception and enhancing safety and efficiency in path planning. Due to the inconsistency of the base for each indicator, normalizing the results of each evaluation metric is essential to avoid errors.

$$\begin{aligned} G(v,\omega) = \sigma(&\alpha_1 \times heading(v,\omega) + \alpha_2 \times dist(v,\omega) + \alpha_3 \times vel(v,\omega) \\ &+ \alpha_4 \times dist\_line(v,\omega) + \alpha_5 \times dist_{goal}(v,\omega) + \alpha_6 \times devi\_danger(v,w)) \end{aligned} \tag{47}$$

where α_1, α_2, α_3, α_4, α_5 and α_6 are the weights of the individual functions.

FUSION ALGORITHM

Figure 2 shows the specific implementation steps of the fusion algorithm. Firstly, MANOA plans the complete path based on the static environmental information. Subsequently, IDWA formulates a dynamic route that enables the robot to evade obstacles and reach the destination point in a continually changing environment, utilizing both path data and environmental information.

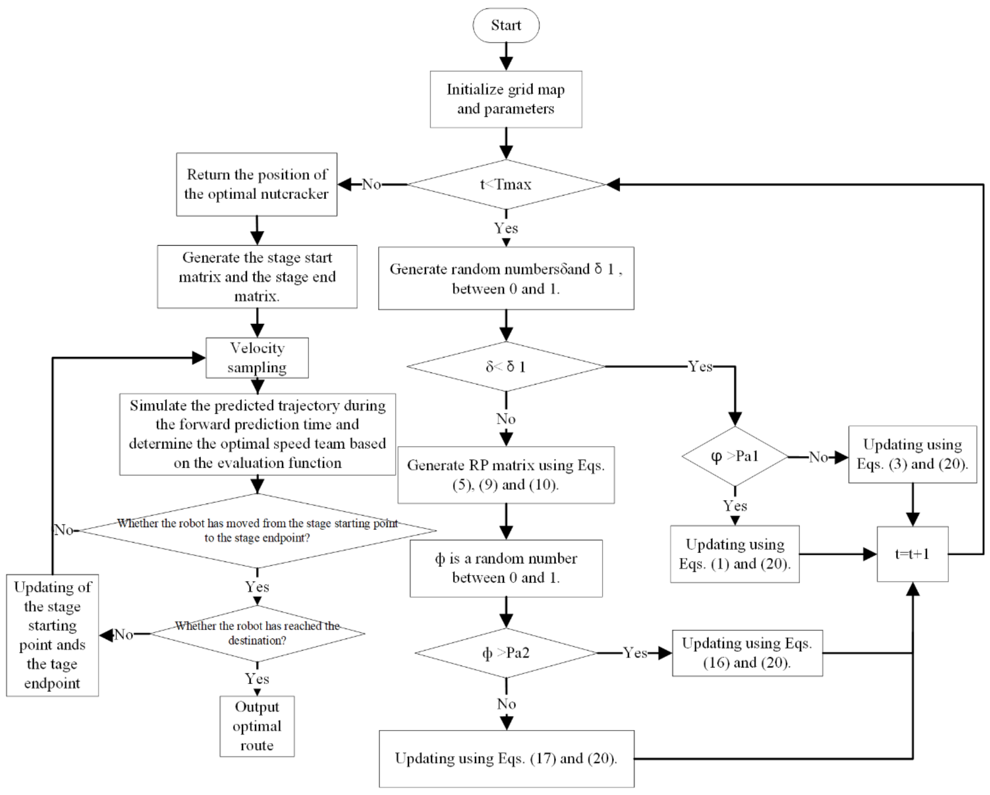

Start
Initialize grid map and parameters
t<Tmax
No
Yes
Return the position of the optimal nutcracker
Generate the stage start matrix and the stage end matrix.
Generate random numbersδand δ 1 , between 0 and 1.
Velocity sampling
δ< δ 1
Yes
No
Simulate the predicted trajectory during the forward prediction time and determine the optimal speed team based on the evaluation function
φ >Pa1
No
Updating using Eqs. (3) and (20).
Generate RP matrix using Eqs. (5), (9) and (10).
Yes
Whether the robot has moved from the stage starting point to the stage endpoint?
No
Updating using Eqs. (1) and (20).
t=t+1
ϕ is a random number between 0 and 1.
Yes
Updating of the stage starting point ands the tage endpoint
No
Whether the robot has reached the destination?
ϕ >Pa2
Yes
Updating using Eqs. (16) and (20).
Yes
No
Output optimal route
Updating using Eqs. (17) and (20).

FIGURE 2. MANOA-IDWA algorithm flowchart

EVALUATION METRICS

The evaluation function used to validate the performance of the paths planned by MANOA-IDWA considers four metrics: obstacle avoidance success rate, moving time, path length growth rate compared to MANOA and path smoothness.

MULTI-ROBOT PATH PLANNING

The objective of multi-robot path planning is to address the issue of path planning when multiple robots are engaged in collaborative tasks. Such tasks may entail collaboration, resource allocation, distributed search, and other operations, including goods sorting in logistics centers, unmanned aerial vehicle formation flight, and multi-robot exploration. In multi-robot systems, preventing collisions between robots is crucial (Gerkey & Mataric 2003; Huang, Cao & Zhu 2019; Matoui et al. 2020; Nunes, McIntire & Gini 2017). Real-time monitoring of robot paths allows the system to detect potential collisions and take appropriate measures to avoid them.

In multi-robot systems, robots should share position, velocity, and target information in real time to coordinate their actions effectively. Firstly, MANOA-IDWA is employed to generate a collision-free path for each robot. Robot priorities are determined based on path lengths, with shorter paths receiving higher priority (Andreychuk & Yakovlev 2018). Robots gather local information by sensing their environment and monitoring the states of neighboring robots. When a higher-priority robot approaches a lower-priority robot, it transmits its position, velocity, and task status. The lower-priority robot processes this information to whether to adopt a secondary departure strategy or a priority departure strategy, effectively coordinating their movements.

RESULTS AND DISCUSSIONS

Four sets of experiments were used to further evaluate the performance of the MANOA-IDWA. Each set of comparison experiments was conducted 20 times.

MANOA ALGORITHM PERFORMANCE VALIDATION EXPERIMENTS

A comparative analysis was conducted to evaluate the performance of MANOA, NOA, HHO, WOABAT, FHO and SO using a map with 40% obstacle coverage. As illustrated in Figure 3, the comparison algorithms all fall into a local optimum. As evidenced in Figure 3(b) and Table 2, the MANOA-planned path demonstrates superior performance in comparison to the comparison algorithms, exhibiting both reduced path length and a smaller number of path nodes. The fitness curves indicate that the MANOA-based path achieves faster convergence and superior fitness compared to the comparison algorithms. This illustrates MANOA's ability to identify the optimal path more quickly while maintaining robust global search capabilities in later iterations, providing a significant advantage over the inherent limitations of the less stable comparison algorithms.

IDWA ALGORITHM PERFORMANCE VALIDATION EXPERIMENTS

The performance of DWA and IDWA was evaluated by varying the number of randomly distributed dynamic obstacles within the map. As demonstrated in Figure 4 and Table 3, in environments with a high density of dynamic obstacles, DWA's probability of planning a complete path is significantly reduced, hindering task completion. In contrast, IDWA remains capable of planning a complete path in complex dynamic environments. The IDWA-based path length is shorter and the path is smoother when both robots reach the same destination. The high stability of IDWA ensures that the robot is able to operate safely and complete tasks effectively in a range of environments.

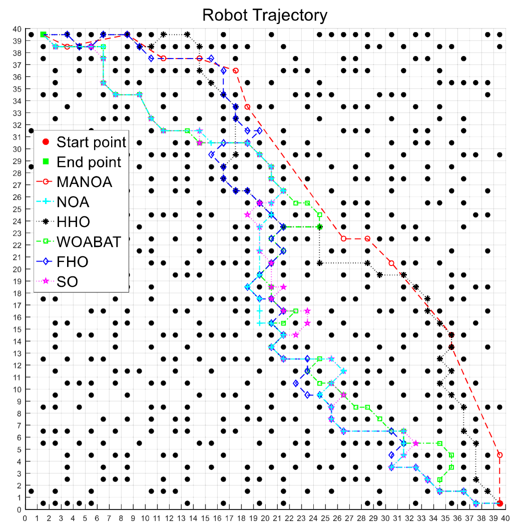
FUSION ALGORITHM PERFORMANCE VALIDATION EXPERIMENTS

The performance of NOA-DWA, HHO-DWA, WOABAT-DWA, FHO-DWA, and SO-DWA was evaluated by randomly introducing 20 or 40 dynamic obstacles into the map. The starting and ending points indicated in Figure 5(a) to 5(f) and 5(g) to 5(l) differ, with 5(g)-5(l) having a greater distance between them, which increases the likelihood of obstacle encounters and complicates task completion. As illustrated in Figure 5(a) to 5(f) and Table 4, with an increase in dynamic obstacles, the path length of MANOA-IDWA is shorter and smoother compared to the comparison algorithms. In more complex environments, the likelihood of the comparison algorithms successfully planning complete paths is significantly reduced. At this juncture, MANOA-IDWA is demonstrably more adaptive, robust, and practical.

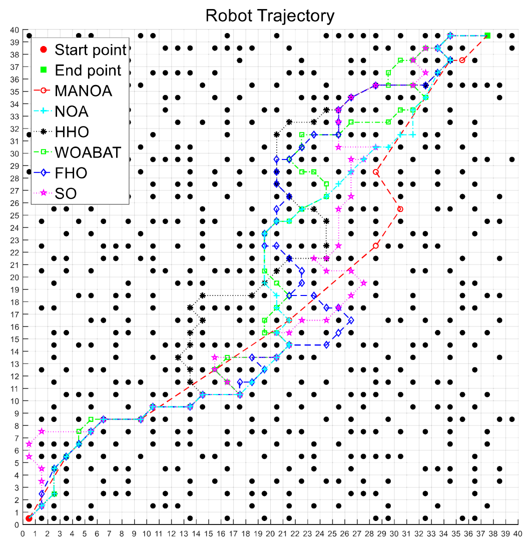
MULTI-ROBOT PATH PLANNING VALIDATION EXPERIMENTS

The performance of NOA-DWA and MANOA-IDWA was evaluated through a comparative analysis, with the addition of either 4 or 8 robots and 20 randomly distributed dynamic obstacles in each trial. The black dashed line in Figure 6 represents the path generated by MANOA.

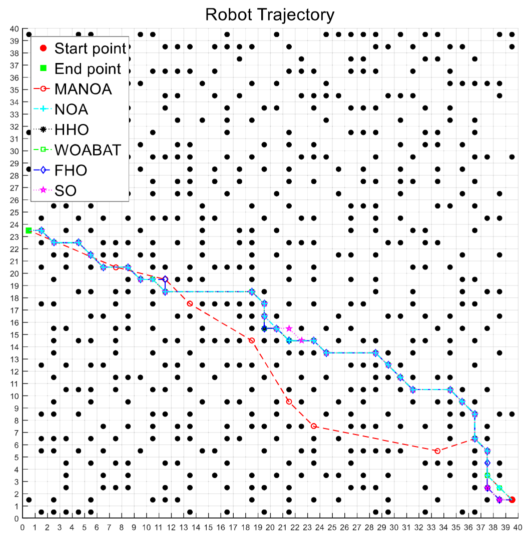
Figure 6 demonstrates that the MANOA-IDWA-based multi-robot systems is capable of performing the assigned tasks in an efficient manner. In 20 experiments, the environment in which the robots are located has been changing dynamically, and the success rate of obstacle avoidance of the MANOA-IDWA-based multiple robots is about 75%. Furthermore, it can be observed that the



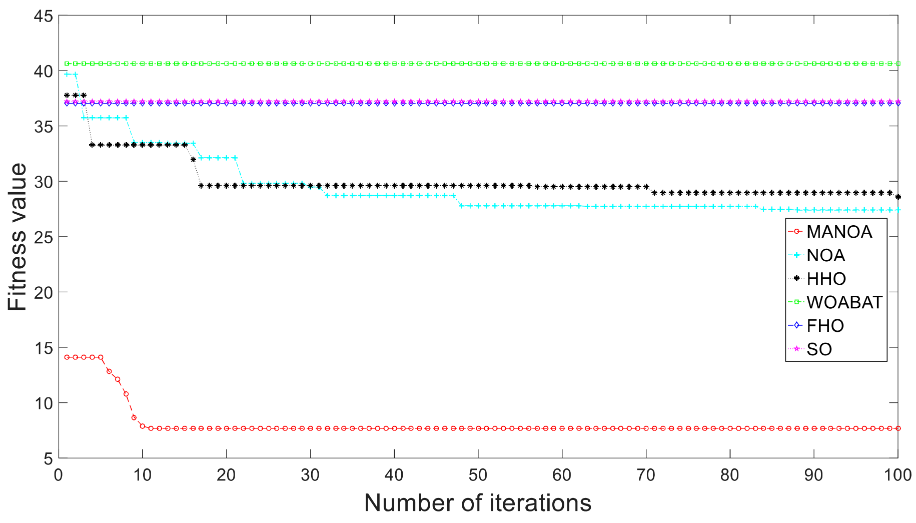
(a) Scene 1



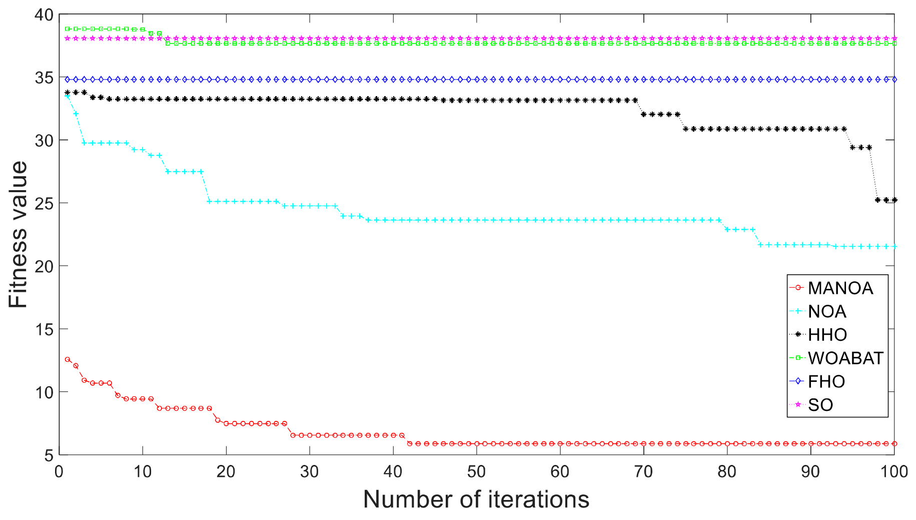
(b) Scene 2



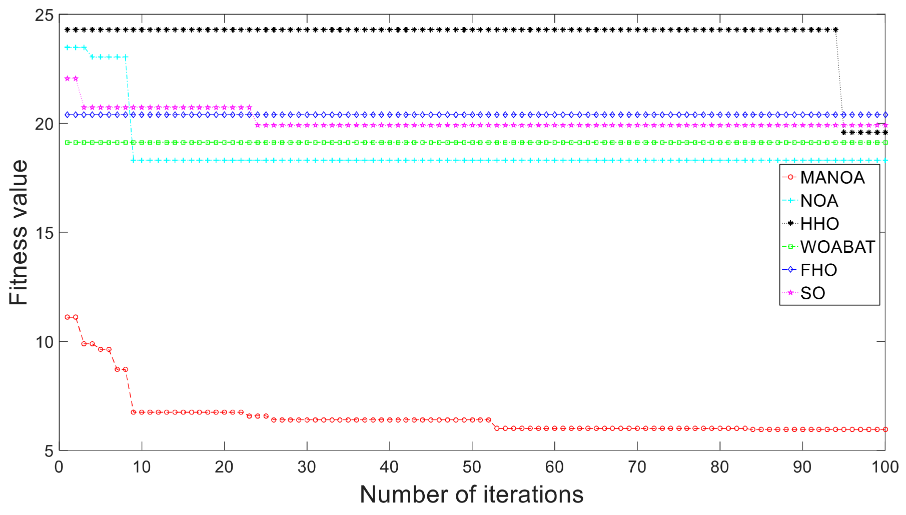
(c) Scene 3



(d) Fitness curves in scene 1



(e) Fitness curves in scene 2

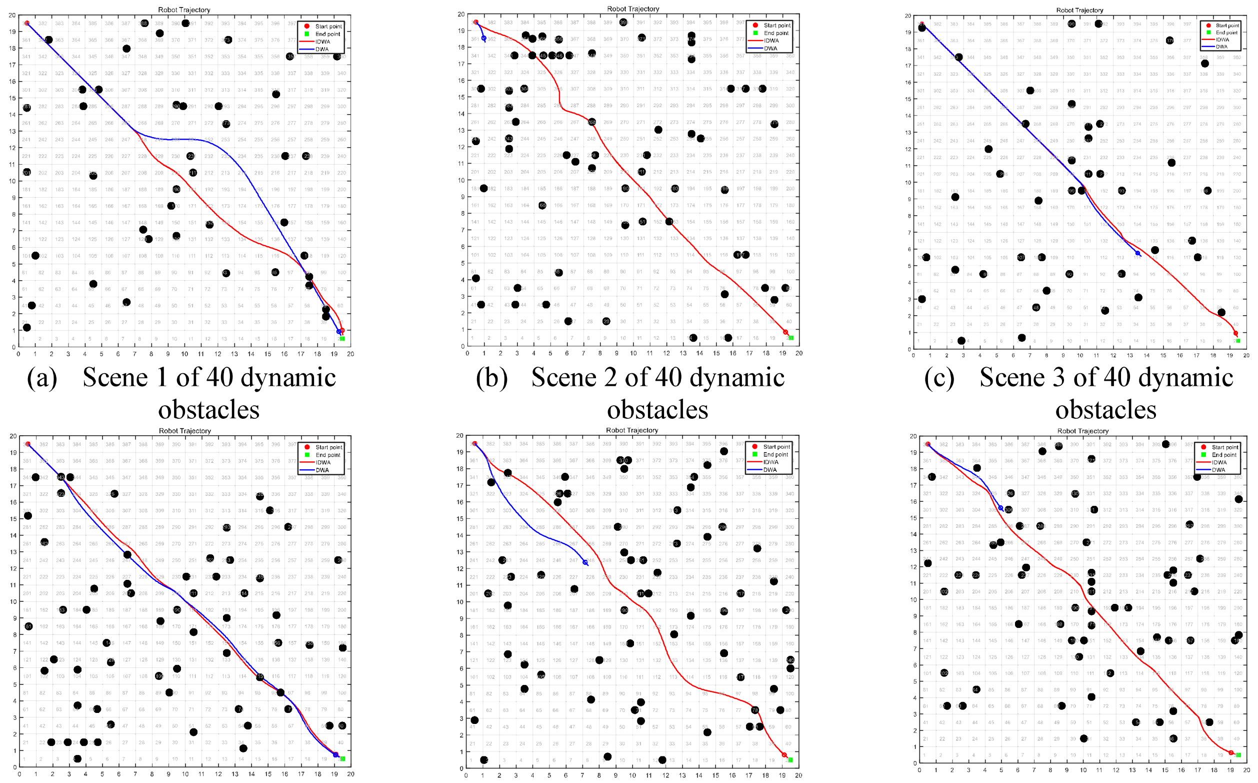


(f) Fitness curves in scene 3

FIGURE 3. Paths planned by six different algorithms, and their respective fitness curves

TABLE 2. Experimental results of path planning with six different algorithms

—	NOA	HHO	WOABAT	FHO	SO	MANOA
Average number of nodes	50	58	61	59	60	11
Average path length	72.248	84.834	91.969	89.076	90.669	63.075



(a) Scene 1 of 40 dynamic obstacles

(b) Scene 2 of 40 dynamic obstacles

(c) Scene 3 of 40 dynamic obstacles

(d) Scene 1 of 60 dynamic obstacles

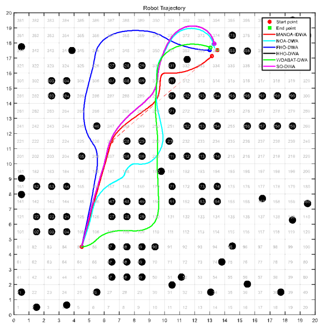
(e) Scene 2 of 60 dynamic obstacles

(f) Scene 3 of 60 dynamic obstacles

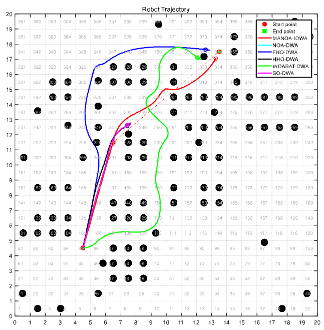
FIGURE 4. Paths planned by the two algorithms for dynamic obstacle numbers of 40 and 60, respectively

TABLE 3. Comparison of paths based on DWA and IDWA

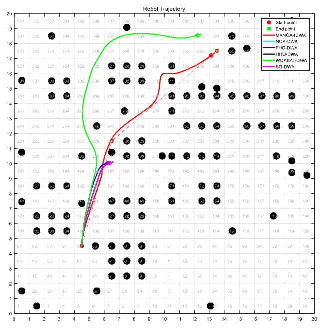
Number of dynamic obstacles	Algorithm	Obstacle avoidance success rate	Average length of feasible paths	Average moving time /s	Path smoothness
20	DWA	0.39	30.996	76.274	0.656
	IDWA	0.80	26.125	65.791	0.402
40	DWA	0.15	36.476	260.123	1.56
	IDWA	0.74	28.952	148.241	0.720



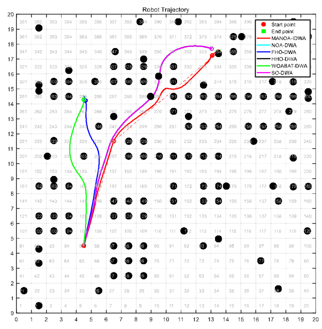
(a) Scene 1 of 20 dynamic obstacles



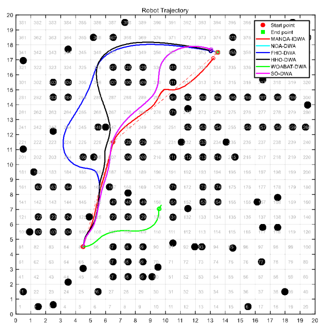
(b) Scene 2 of 20 dynamic obstacles



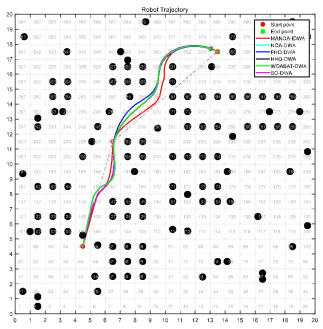
(c) Scene 3 of 20 dynamic obstacles



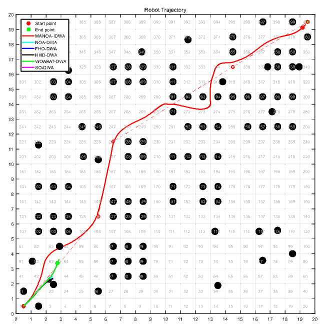
(d) Scene 1 of 40 dynamic obstacles



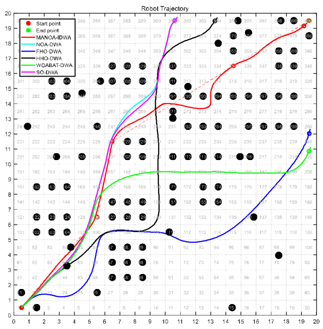
(e) Scene 2 of 40 dynamic obstacles



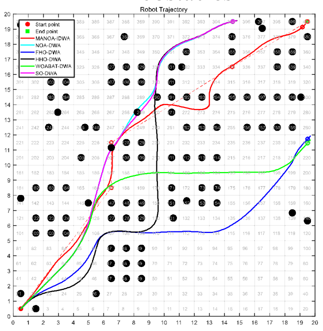
(f) Scene 3 of 40 dynamic obstacles



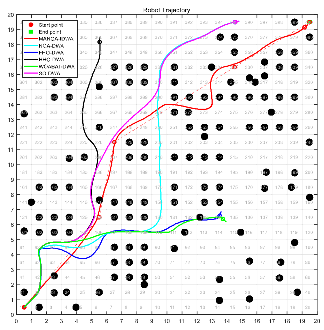
(g) Scene 1 of 20 dynamic obstacles



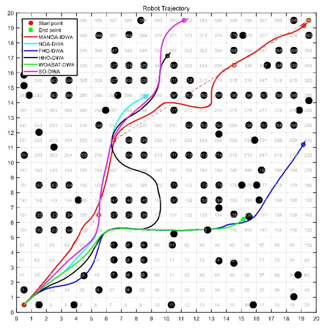
(h) Scene 2 of 20 dynamic obstacles



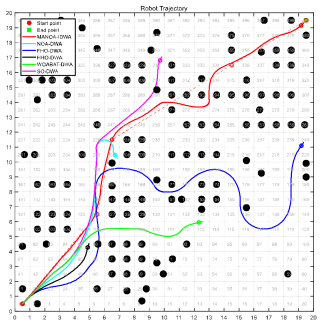
(i) Scene 3 of 20 dynamic obstacles



(j) Scene 1 of 40 dynamic obstacles



(k) Scene 2 of 40 dynamic obstacles

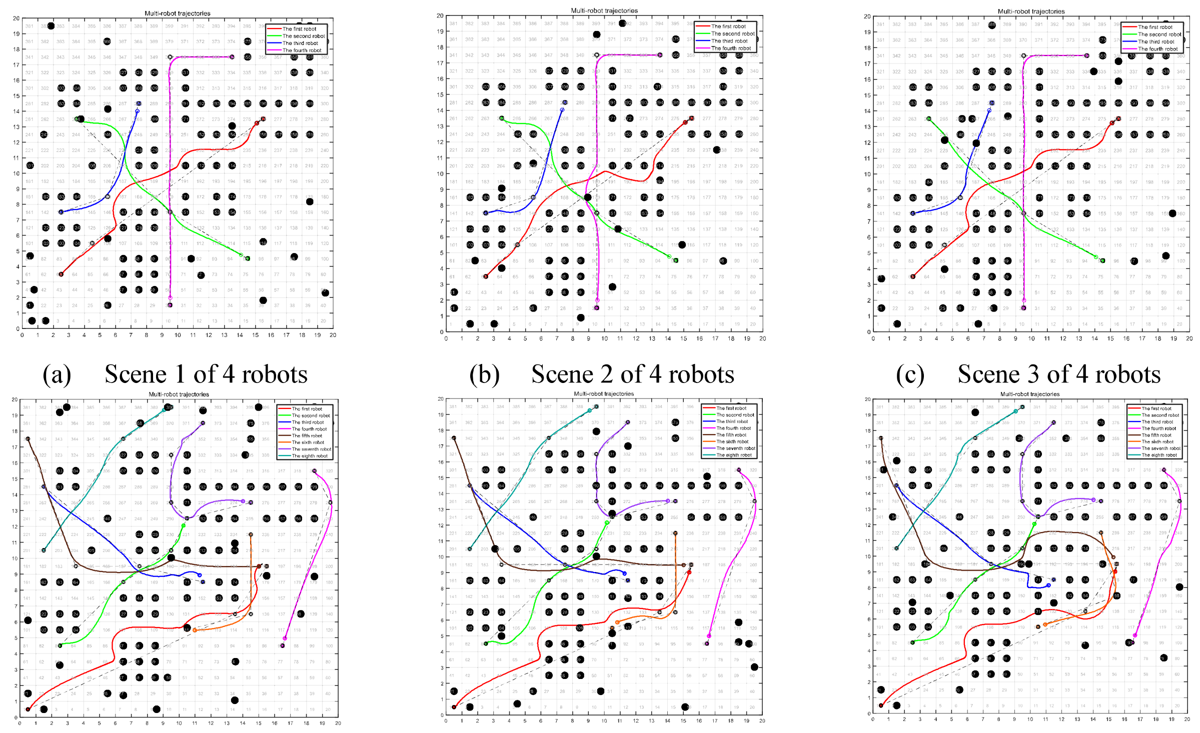


(l) Scene 3 of 40 dynamic obstacles

FIGURE 5. Paths planned by fusion algorithms

TABLE 4. Paths based on MANOA-IDWA compared to paths based on comparison algorithms

Algorithm	Obstacle avoidance success rate	Average moving time/s	Path length growth rate compared to MANOA	Path smoothness
MANOA-IDWA algorithm	0.75	96.150	0.091	1.701
Other 5 comparison algorithms	0.16	169.562	0.501	4.126



(a) Scene 1 of 4 robots (b) Scene 2 of 4 robots (c) Scene 3 of 4 robots

(d) Scene 1 of 8 robots (e) Scene 2 of 8 robots (f) Scene 3 of 8 robots

FIGURE 6. Paths planned by multiple robots

actual routes of the robots do not deviate significantly from the static paths planned by MANOA. This indicates that the paths based on MANOA-IDWA continue to exhibit exceptional fitness.

CONCLUSION

The domain of multi-robot path planning confronts a myriad of challenges. To address these challenges, this paper proposes a path planning method that combines the improved nutcracker optimization algorithm named multi-strategy adaptive nutcracker optimization algorithm (MANOA), with the improved dynamic window approach (IDWA). Global planning uses MANOA to generate static paths for robots. Through innovative strategies like population initialization and simplified path node, MANOA aims to diversify initial populations, reduce path complexity, thereby enhancing movement stability. By adding the dynamic inertia weight factor w, the optimization of performance in different stages has been achieved. Local planning uses IDWA to handle obstacle avoidance and trajectory adjustment problems. The evaluation ability of the evaluation function is enhanced by adding the other three subfunctions to improve planning efficiency, robustness, and adaptability.

ACKNOWLEDGEMENTS

We would like to thank the National Nature Science Foundation of China for the financial assistant through Grant/Award Number: 6146105.

REFERENCES

Abdel-Basset, M., Mohamed, R., Jameel, M. & Abouhawwash, M. 2023. Nutcracker optimizer: A novel nature-inspired metaheuristic algorithm for global optimization and engineering design problems. *Knowledge-Based Systems* 262: 110248.

Andreychuk, A. & Yakovlev, K. 2018. Two techniques that enhance the performance of multi-robot prioritized path planning. *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS' 18)*. doi:10.48550/arXiv.1805.01270

Chen, H.L., Xu, Y.T., Wang, M.J. & Zhao, X.H. 2019. A balanced whale optimization algorithm for constrained engineering design problems. *Applied Mathematical Modelling* 71: 45-59.

Ge, S.S. & Cui, Y.J. 2002. Dynamic motion planning for mobile robots using potential field method. *Autonomous Robots* 13(3): 207-222.

Gerkey, B. & Mataric, M. 2003. A formal framework for the study of task allocation in multi-robot systems. *International Journal of Robotic Research* 23(9): 939-954.

Han, S., Wang, L., Wang, Y.T. & He, H.C. 2022. A dynamically hybrid path planning for unmanned surface vehicles based on non-uniform Theta* and improved dynamic windows approach. *Ocean Engineering* 257: 111655.

Huang, X.Q., Cao, Q.X. & Zhu, X.X. 2019. Mixed path planning for multi-robots in structured hospital environment. *Journal of Engineering* 2019(14): 512-516.

Ida Evangeline, S., Darwin, S., Peter Anandkumar, P. & Sreenivasan, V.S. 2024. Investigating the performance of a surrogate-assisted nutcracker optimization algorithm on multi-objective optimization problems. *Expert Systems with Applications* 245: 123044.

Jian, Z.Q., Zhang, S.Y., Chen, S.T., Nan, Z.X. & Zheng, N.N. 2021. A global-local coupling two-stage path planning method for mobile robots. *IEEE Robotics and Automation Letters* 6(3): 5349-5356.

Kanoon, Z., Al-Araji, A. & Abdullah, M. 2022. Enhancement of cell decomposition path-planning algorithm for autonomous mobile robot based on an intelligent hybrid optimization method. *International Journal of Intelligent Engineering and Systems* 15(3): 161-175.

Karaman, S. & Frazzoli, E. 2011. Sampling-based algorithms for optimal motion planning. *International Journal of Robotic Research* 30(7): 846-894.

Kennedy, J. & Eberhart, R. 1995. Particle swarm optimization. *IEEE International Conference on Neural Networks Proceedings* doi: 10.1109/icnn.1995.488968

Lin, S.W., Liu, A., Wang, J.G. & Kong, X.Y. 2022. A review of path-planning approaches for multiple mobile robots. *Machines* 10(9): 773.

Liu, G.Y., Shu, C., Liang, Z.W., Peng, B.H. & Cheng, L.F. 2021. A modified sparrow search algorithm with application in 3D route planning for UAV. *Sensors* 21(4): 1224.

Madridano, A., Al-Kaff, A., Martín, D. & de la Escalera, A. 2021. Trajectory planning for multi-robot systems: Methods and applications. *Expert Systems with Applications* 173: 114660.

Matoui, F., Boussaid, B., Metoui, B. & Abdelkrim, M.N. 2020. Contribution to the path planning of a multi-robot system: Centralized architecture. *Intelligent Service Robotics* 13(1): 147-158.

Nunes, E., McIntire, M. & Gini, M. 2017. Decentralized multi-robot allocation of tasks with temporal and precedence constraints. *Advanced Robotics* 31(22): 1193-1207.

Wang, H., Wang, W., Zhou, X., Sun, H., Zhao, J., Yu, X. & Cui, Z. 2016. Firefly algorithm with neighborhood attraction. *Information Sciences* 382: 374-387.

Xue, J.K. & Shen, B. 2020. A novel swarm intelligence optimization approach: Sparrow search algorithm. *Systems Science & Control Engineering* 8(1): 22-34.

Yang, L.W., Fu, L.X., Li, P., Mao, J.L. & Guo, N. 2022. An effective dynamic path planning approach for mobile robots based on ant colony fusion dynamic windows. *Machines* 10(1): 50.

Yang, X.S. 2009. Firefly algorithms for multimodal optimization. In *Stochastic Algorithms: Foundations and Application.* SAGA 2009. Lecture notes in computer science, vol 5792*,* edited by Watanabe, O. & Zeugmann, T. Berlin, Heidelberg: Springer. doi:10.1007/978-3-642-04944-6_14

Yu, G., Wang, H., Zhou, H.Z., Zhao, S.S. & Wang, Y. 2021. An efficient firefly algorithm based on modified search strategy and neighborhood attraction. *International Journal of Intelligent Systems* 36(8): 4346-4363.

*Corresponding author; email: zth1320359@163.com