electronics

MDPI

Article

A Local Path Planning Algorithm for Robots Based on Improved DWA

Xue Gong [1,2], Yefei Gao [1,2,*], Fangbin Wang [1,2,*], Darong Zhu [1,2], Weisong Zhao [1,2], Feng Wang [3] and Yanli Liu [1]

[1] School of Mechanical and Electrical Engineering, Anhui Jianzhu University, Hefei 230601, China; zhaoweisong@ahjzu.edu.cn (W.Z.)
[2] Key Laboratory of Construction Machinery Fault Diagnosis and Early Warning Technology, Anhui Jianzhu University, Hefei 230601, China
[3] Polarized Light Imaging Detection Technology Anhui Provincial Key Laboratory, Hefei 230031, China; wfissky@sina.com
* Correspondence: gaoyf@std.ahjzu.edu.cn (Y.G.); wangfb@ahjzu.edu.cn (F.W.)

Abstract: In order to solve the problem whereby the original DWA algorithm cannot balance safety and velocity due to fixed parameters in complex environments with many obstacles, an improved dynamic window approach (DWA) of local obstacle avoidance for robots is proposed. Firstly, to assure the path selection stationarity and enhance the navigation ability of inspection robot, the velocity cost function of the original DWA was improved and the distance cost function of the target point was added. Then, the distances among the inspection robot, observed obstacles, and target points were input into a fuzzy control module, and the fuzzy weights of the velocity and distance cost functions were obtained, by which the motion of the inspection robot can continuously self-adjust and adapt to the unknown environment. Finally, several simulations and experiments were conducted. The results show that the improved DWA algorithm can effectively improve the obstacle avoidance ability of inspection robots in complex environments. The path can be more reasonably selected and the safety of inspection robots can be enhanced, while the safe distance, path length, and the number of samples can also be optimized by the improved DWA compared to the original DWA.

Keywords: local path planning; DWA; fuzzy control; obstacle avoidance

Citation: Gong, X.; Gao, Y.; Wang, F.; Zhu, D.; Zhao, W.; Wang, F.; Liu, Y. A Local Path Planning Algorithm for Robots Based on Improved DWA. *Electronics* **2024**, *13*, 2965. https://doi.org/10.3390/electronics13152965

Academic Editor: Andrea Bonci

Received: 18 June 2024
Revised: 21 July 2024
Accepted: 24 July 2024
Published: 27 July 2024

1. Introduction

Path planning is one of the key technologies for autonomous navigation of robots [1,2]. When facing a complex working environment, it is difficult to obtain complete spatial information, and robots are required to have stronger path planning and obstacle avoidance capabilities [3,4]. Therefore, it is of significance and practical value to study robot path planning algorithm [5,6].

Path planning algorithms are usually classified into global and local path planning based on the environmental information acquired. Frequently used global path planning algorithms include A*, fast random search tree (RRT), and various intelligent algorithms such as GA [7–9]. These algorithms usually require all known spatial map information and are difficult to apply in real-time complex environments. Local path planning is commonly used in unknown environments, where robots need real-time information about environmental obstacles based on sensors such as LiDAR and cameras [10,11] and feasible paths can be effectively planned. Commonly used local planning algorithms include artificial potential field (APF) [12] and Time Elastic Band (TEB) [13]. APF takes the target point as a gravitational source, obstacles as a repulsive source, and the resultant force to calculate the robot's motion direction and posture. The real-time performance of APF is very good for local obstacle avoidance, but prone to falling into local optima [14]. TEB treats the given initial path as a deformable elastic band, can change the path locally based on the position and shape of obstacles, and the global path can be replanned. The dynamic

window approach (DWA) [15] algorithm samples linear and angular velocity in the velocity space with dynamic constraints, then generates candidate paths according to the kinematic model of the robot and evaluates the candidate paths by a cost function. The optimal path with the maximum cost function value is selected from the candidate paths as the path of the robot in the next motion stage. However, there exists some application problems in complex and dense obstacle environments for DWA:

(1) Robots may approach obstacles during traveling. When encountering pedestrians or moving objects, the safety and humanization of the robots may be reduced and collisions occur.
(2) The weights of the DWA cost function are usually fixed, and cannot adapt to the complex obstacle environment, resulting in poor obstacle avoidance performance.

To address the above issues, Ballesteros et al. [16] proposed an improved DWA, which evaluates the mobile robot by considering the free-space relationship between its dimensions and the obstacles. However, the method is prone to fall into a local optimum. Therefore, Wei B, et al. [17] proposed a new DWA based on environmental perception, which can guide the robot to travel to an unobstructed area before arriving at the target point by setting local target points and solve the problem of local minimum values. However, the setting of local target points in this algorithm is not the optimal distance from the final target point, so there are still unreasonable factors in path selection. Chang L, et al. [18] proposed a novel algorithm adaptively adjusting DWA parameters online by combining Q-learning into training aiming to address the above problem. On other hand, DWA requires model training before use and the actual planning efficiency for unexpected situations is reduced. Wang YX, et al. [19] designed adaptive rules to dynamically adjust the weights in the DWA objective function. However, this method only considers the influence of velocity weights in the cost function, while not the contribution from other cost functions.

However, due to only simulating and evaluating the trajectory of the next step, there is a lack of foresight and a tendency to fall into local optima, and each time, the path selected through the cost function is the optimal path for the next step, rather than the global optimal path. Once the weight of the cost function in the DWA algorithm is determined, it cannot be dynamically adjusted and cannot adapt to various complex environments. Therefore, an improved DWA is proposed in this paper. The cost function is optimized and fuzzy control theory is introduced to dynamically adjust the weights of the cost function to adapt to a complex obstacle environment so that a robot can quickly and smoothly avoid obstacles without collision.

2. Theoretical Basis

DWA is one of the practical local path planning methods that was proposed by Fox et al. based on the correspondence between robot position and velocity [20]. DWA transforms the position control into velocity control for a robot, taking obstacle avoidance problems as optimization ones with velocity space constraints. The velocity space of the robot according to the current state of the robot and the robot motion model is calculated, and it is the dynamic window mentioned in the name of the algorithm by which the candidate paths of the mobile robot are calculated in a certain period of time and evaluated through the cost function. As a result, the optimal path is selected according to the cost function for path planning. The DWA algorithm generally includes three parts: kinematic model establishment, velocity space collection, and candidate path evaluation.

(1) Kinematic model establishment

In this process, the motion model of the robot is required to be established for DWA [21,22]. Usually, the motion model of the robot is built as shown in Figure 1.

In Figure 1, X and Y denote the axes of the world coordinate, and Xrobot and Yrobot denote the axes of the current motion coordinate of mobile robot. The origin of the world coordinate system is set as the initial position of the robot. The origin of the current motion coordinate system is set as the center point of the robot. θ is the angle between the current

motion direction and the horizontal direction of the robot. Assuming that the position coordinate of the robot at moment t is $(x(t), y(t))$, the position relation equation of the robot at moment $t + \Delta t$ can be written as follows:

$$\begin{bmatrix} x(t+\Delta t) \\ y(t+\Delta t) \\ \theta(t+\Delta t) \end{bmatrix} = \begin{bmatrix} x(t) + v(t) * cos(\theta(t)) * \Delta t \\ y(t) + v(t) * cos(\theta(t)) * \Delta t \\ \theta(t) + \omega(t) * \Delta t \end{bmatrix} \tag{1}$$

where $v(t)$ is the linear velocity of the robot at moment t, $\omega(t)$ is the angular velocity of the robot, and Δt is the sampling time step.

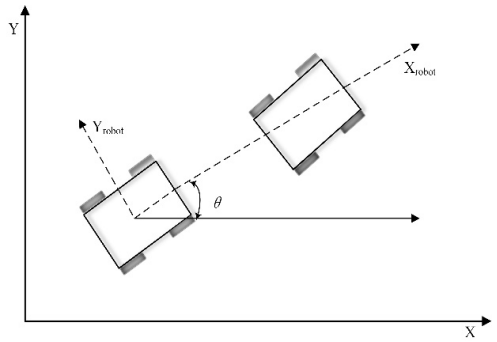


Figure 1. Simplified motion model of DWA robot.

(2) Velocity space collection

From Equation (1), there are many velocity combinations (v, ω) in the velocity space of the robot, but only some of them are consistent with the actual operation of the robot. Therefore, this paper adopts three kinds of constraints to constrain the velocity used for sampling in a reasonable range, which are self-velocity limitation, dynamic performance constraints, and safety constraints.

The self-velocity limit of mobile robot, which is determined from the specification of the robot, must be satisfied by the velocity combination $V_s(v, \omega)$:

$$V_s = \{(v, \omega) | v \in [v_{\max}, v_{\min}], \omega \in [\omega_{\max}, \omega_{\min}]\} \tag{2}$$

where $v_{\max}$ and $v_{\min}$ are the maximum and minimum linear velocities limitation of the robot and $\omega_{\max}$ and $\omega_{\min}$ are the maximum and minimum angular velocity limitation, respectively.

Due to motor torque performance, the instantaneous linear and angular acceleration velocities of the robot must be limited in an achievable range within Δt time. Given the current linear and angular velocities $v(t)$ and $\omega(t)$, the velocity limitation by the motor performance at the next $(t + \Delta t)$ moment can be represented as:

$$V_d = \left\{(v, \omega) \middle| v \in \left[v_t - \dot{v}_t \Delta t, v_t + \dot{v}_t \Delta t\right], \omega \in \left[\omega_t - \dot{\omega}_t \Delta t, \omega_t + \dot{\omega}_t \Delta t\right]\right\} \tag{3}$$

where $v(t)$ and $\omega(t)$ are the current linear velocity and angular velocity of the robot and $\dot{v}_t$ and $\dot{\omega}_t$ are the current maximum linear and angular acceleration of the robot, respectively.

In order to avoid collisions, the robot must be able to stop moving before hitting an obstacle, where the set of linear and angular velocities can be satisfied and defined as follows:

$$V_a = \left\{(v, \omega) \middle| v \leq \left(2 \cdot dist(v, \omega) \cdot \dot{v}_t\right)^{1/2}, \omega \leq \left(2 \cdot dist(v, \omega) \cdot \dot{\omega}_t\right)^{1/2}\right\} \tag{4}$$

where $dist(v, \omega)$ is the distance between the endpoint of the predicted path and the nearest obstacle. The collision condition calculates whether the current sampling velocity can be reduced to 0 before hitting the obstacle according to the distance between the robot and the obstacle after the path is simulated. If the robot can stop, the velocity is allowed.

For a mobile robot, velocity space V_t can be represented as the intersection of three constraints, which is the dynamic window in the name of the algorithm:

$$V_t = V_s \cap V_d \cap V_a \tag{5}$$

(3) Candidate path evaluation

The robot continuously samples its velocity in velocity space during operation. According to the motion model and velocity space, several candidate paths are generated in the current motion coordinate and evaluated by the cost function. The cost function $G(v, \omega)$ can be constructed as:

$$G(v, \omega) = \alpha head(v, \omega) + \beta dist(v, \omega) + \gamma vel(v, \omega) \tag{6}$$

where $head(v, \omega)$ denotes the azimuth between the robot head and the target at the end position of the predicted path. It is the cost of the robot orientation to the target. $dist(v, \omega)$ denotes the distance between the endpoint of the predicted path and the nearest obstacle, $vel(v, \omega)$ denotes the robot's traveling velocity, and α, β and γ are weight coefficients. The optimal path is chosen from candidate paths by maximizing the cost function.

3. Improved DWA Algorithm

In this section, an adaptive DWA based on fuzzy control module is proposed to address the problems caused by unreasonable path selection, poor obstacle avoidance ability and low adaptability caused by fixed cost function weights in dense obstacle environments. In the improved DWA, $vel(v, \omega)$ of the original DWA was optimized by adding a target point distance into the cost function to ensure the velocity of the chosen path is stable, which can enhance the robot's navigation ability to move towards a target, and the fuzzy control method was used to adaptively adjust the weights of the cost function to make the robot adapt the unknown complex environments.

3.1. Cost Function Optimization

3.1.1. Optimize the Velocity Cost Subfunction $vel(v, \omega)$

In the original DWA, $vel(v, \omega)$ considers only the linear velocity within the predicted path. The higher the linear velocity, the larger the value of $vel(v, \omega)$, which helps the robot reach the target point more quickly. However, in order to weaken the drastic velocity variation due to obstacle avoidance in complex environments, affecting the smoothness as well as the moving stability of the robot, $vel(v, \omega)$ should be optimized and was rewritten in this paper as follows:

$$vel(v, \omega)' = v_i - l * \left(\frac{|v_i - v_{i-1}|}{v_{\max}} \right) \tag{7}$$

where l is the penalty coefficient, v_i and v_{i-1} denote the linear velocity of the predicted path at the current and previous moment, respectively, and $v_{\max}$ is the maximum linear velocity of the robot.

In Equation (7), a correction function is adopted and a penalty is applied to the rapidly changing velocity to reduce the $vel(v, \omega)'$ value, by which a predicted path with gentle changes in velocity can be selected to maintain the stability of the robot's motion.

3.1.2. Introduce Target Deviation Evaluation

A new target deviation cost subfunction $goal(v, \omega)$ is added. It is used to evaluate the heading difference and distance between the robot and the target point, where it is expected that the robot can travel towards the target point and reduce the length of the predicted paths.

The $goal(v, \omega)$ function was defined as follows:

$$goal(v, \omega) = k * (1 - d_g / d) \tag{8}$$

where k is the control coefficient, d_g is the distance of the current robot from the target point, and d is the distance from the starting point to the target point. The closer the robot is to the target points, the higher the value of the $goal(v, \omega)$.

As shown in Figure 2, two predicted paths are compared with the same velocity. The distances of path 1 and path 2 from the nearest obstacle are d_{o1} and d_{o2}, and the azimuth angles with the target point are θ_1 and θ_2, respectively. If the original DWA algorithm is used to evaluate the performance of these two paths, since $d_{o1} < d_{o2}$ and $\theta_1 < \theta_2$, it indicates that path 1 has better directivity and path 2 has better security. It is difficult to determine which path is superior. If introducing $goal(v, \omega)$ and $d_{g1} < d_{g2}$, path 1 will obtain the higher value of the cost function and can bring the robot closer to the target point.

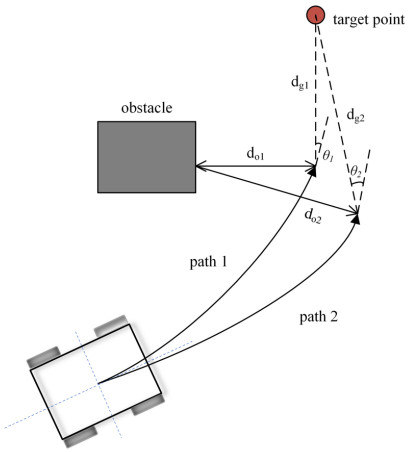


Figure 2. Path comparison analysis.

Both $head(v, \omega)$ and $goal(v, \omega)$ functions facilitate the navigation of the robot towards the target point, and the azimuth and distance between the endpoint of the path and the target point were considered carefully. $goal(v, \omega)$, a complement to $head(v, \omega)$, mainly focuses on the navigation in the latter stages of motion, and $goal(v, \omega)$ gradually plays a dominant role when the robot is closer to the obstacles. In contrast, in the pre-motion phase, navigation mainly relies on $head(v, \omega)$. The specific method can be realized by adaptively adjusting the weight θ through fuzzy control, as in Section 3.2.

Based on the above analysis, the improved prediction path cost function proposed in this paper was ultimately determined as follows:

$$G(v, \omega)' = \alpha head(v, \omega) + \beta dist(v, \omega) + \gamma vel(v, \omega)' + \theta goal(v, \omega) \tag{9}$$

3.2. Fuzzy Control Adaptive DWA Algorithm

To solve the problem that the cost function weights of original DWA algorithm are fixed and cannot be dynamically adjusted, which leaves the robot unable find the target position or choose a longer path to bypass obstacles, a fuzzy control method was introduced to adjust the cost function weights adaptively so that the driving path of the robot was optimized when avoiding obstacles.

The distance d_o between the robot and the obstacle and the distance d_g between the robot and the target point are taken as fuzzy control input variables with ranges of [0, 2] m and [0, 15] m, respectively, defined as close (C), medium (M), and far (F) in fuzzy languages. β, γ and θ are taken as the fuzzy control outputs, with a range of [0, 2], where β and θ are defined as small (S), medium (M), and large (L), and γ is defined as extremely small (XS),

small (S), medium (M), large (L), and extremely large (XL) in fuzzy languages. The input and output affiliation functions are shown in Figure 3.

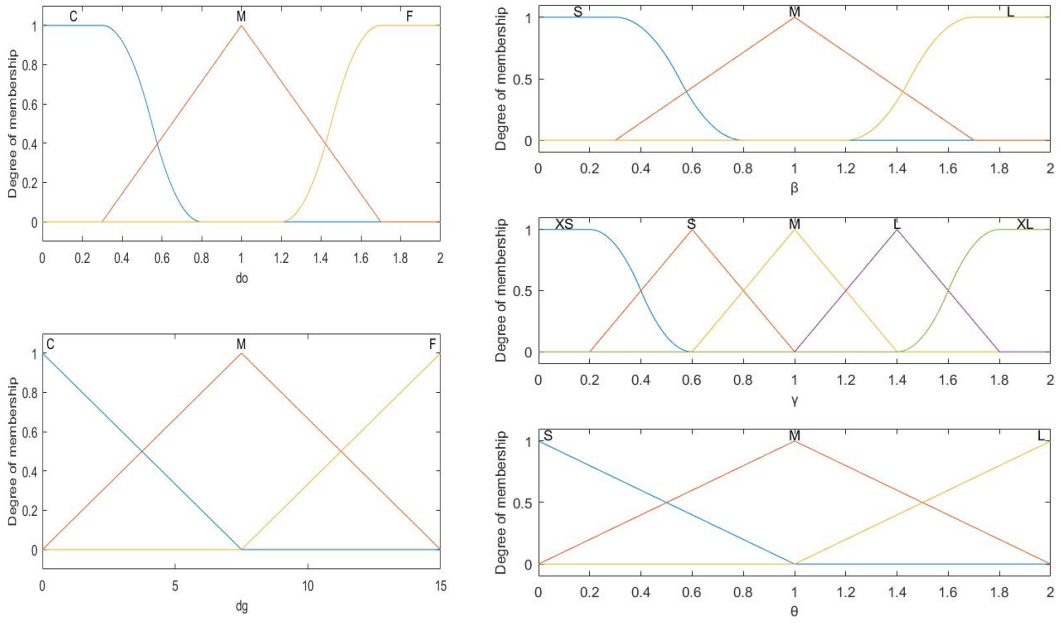


Figure 3. Input–output membership function diagram. (**a**) Input affiliation functions; (**b**) output affiliation functions.

According to the physical significance of the cost function of the DWA algorithm and considering the actual obstacle avoidance situation of the robot, the following fuzzy rules are formulated.

(1) When the distance d_o between the robot and the nearest obstacle is long and so is the distance d_g between the robot and the target, the robot has higher priority to drive towards the target at high speed and does not avoid the obstacle urgently, so the weight of the velocity cost subfunction γ increases as the distance increases. The distance between the robot and the obstacle and the direction of the target are non-major factors—their weights β and θ are smaller than γ.

(2) When the distance d_o between the robot and the nearest obstacle is short, obstacle avoidance must be prioritized from the safety point of view. While avoiding obstacles, it is necessary to slow the robot appropriately to avoid collision accidents. In this case, the weight of the velocity cost subfunction γ decreases as the distance d_o decreases, and β and θ are larger than γ.

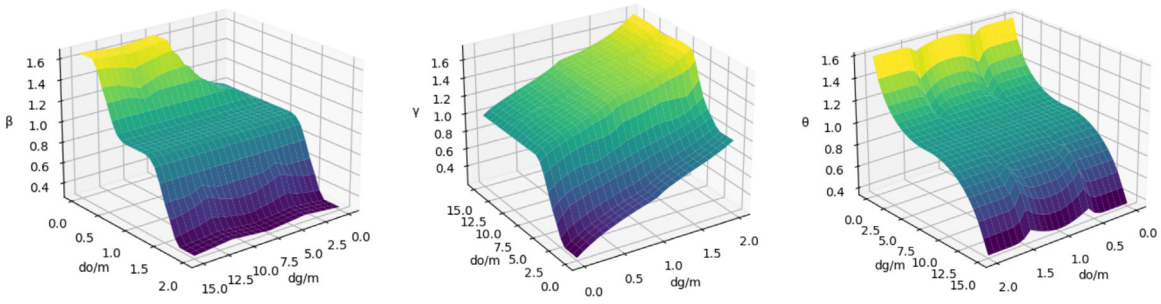
(3) When the distance d_g between the robot and the target point is short, the robot should adjust the driving direction to the target position and the moving speed should be reduced at the same time, so θ is increased and γ is decreased as the distance d_g decreases.

Above all, the fuzzy control idea is that in any case, the safety of the robot must be ensured. Therefore, it is necessary to prioritize avoiding obstacles and stable driving of the robot. The fuzzy rules are shown in Table 1.

The three-dimensional diagram of the output is shown in Figure 4. It can be seen that β reaches the maximum value when the robot is very close to the obstacle and far from the target, and the minimum value when it is far from the obstacle. On the other hand, γ reaches the maximum value when it is far from the obstacle and the target, and the minimum value when the robot is close to the obstacle and the target. Moreover, θ is negatively correlated with the size of d_g and becomes larger when the robot is closer to the target point.

Table 1. Fuzzy rules.

Input Variables		Output Variables		
d_o	d_g	β	γ	θ
C	C	M	XS	L
C	M	L	S	M
C	F	L	M	S
M	C	M	M	L
M	M	M	L	M
M	F	M	XL	S
F	C	S	M	L
F	M	S	L	M
F	F	S	XL	S



(**a**) (**b**) (**c**)

Figure 4. Output result three-dimensional diagram. (**a**) β; (**b**) γ; (**c**) θ.

4. Simulation Experiments and Result Analysis

To validate the feasibility of introducing fuzzy control module into DWA, several comparative simulations were conducted between the improved and original DWA in complex environments. Environmental maps with different obstacle distributions were established in Python language. In the maps, the black grids represent obstacle areas, while the white areas can be passed through by robots. "★" is the starting point at (1, 1) and "×" is the target point. The parameters of the robot and algorithm are shown in Tables 2 and 3, and the simulation parameters are set as shown in Table 4.

Table 2. Robot parameters.

Maximum Linear Velocity/(m/s)	Minimum Linear Velocity/(m/s)	Maximum Angular Velocity/(°/s)	Minimum Angular Velocity/(°/s)	Linear Acceleration/(m/s^2)	Angular Acceleration/(°/s^2)
1.0	0.0	45	−45	0.2	45

Table 3. Algorithm parameters.

Velocity Resolution/(m/s)	Angular Velocity Resolution/(°/s)	Sampling Interval/s	Path Prediction Time/s
0.01	0.5	0.1	3.0

Table 4. Experimental parameters.

Map Size (m)	Starting Point Position (m)	Initial Velocity (m/s)	Initial Angular Velocity (°/s)	Initial Heading Angle (°)	Direction Angle Weight α
12 × 12	(1, 1)	0 m/s	0	18	0.5

4.1. Dense Obstacle Environment

Map 1 shows the simulation environment used to evaluate the obstacle avoidance ability and the path selection reasonableness of the improved DWA in dense obstacles compared to the original DWA with different fixed weights. The target point position is set to (11, 10). The experimental results are shown in Figure 5.

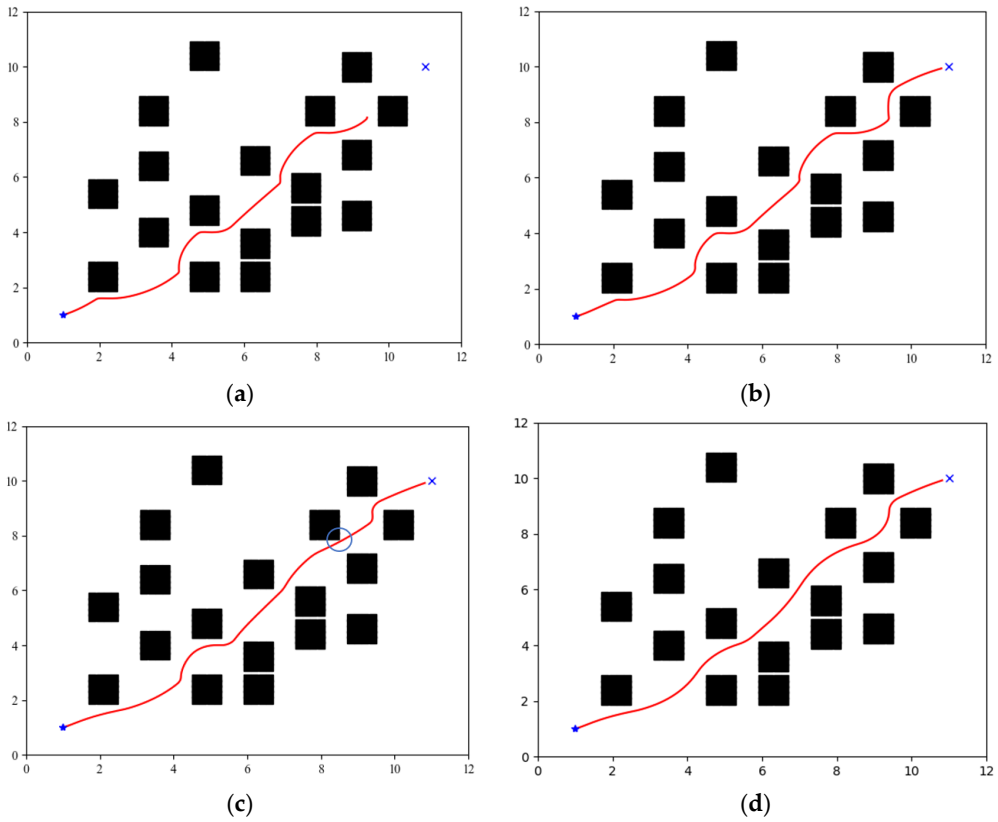


Figure 5. Map 1 simulation results. (**a**) $\beta = 1.7$, $\gamma = 0.4$; (**b**) $\beta = 1.2$, $\gamma = 1.3$; (**c**) $\beta = 0.8$, $\gamma = 0.3$; (**d**) dynamic weight.

Different weights have a great impact on the results of path planning for the original DWA algorithm. It can be seen from Figure 5a that when the cost function weights are not set accurately enough, for example, $\beta = 1.7, \gamma = 0.4$, the robot will stop in front of the obstacle and cannot reach the target position. When the fixed weight β is 1.2 and γ is 1.3, although the path can be successfully planned to reach the target point, the frequent path turning results in the robot not being smooth enough. When the fixed weight β is 0.8 and γ is 0.3, the path length is reduced, but the minimum safe distance among dense obstacles is too small, so the security is poor, as shown in the position circled in Figure 5c. The improved DWA algorithm can make the path length shorter and can successfully guide the robot through the dense obstacle area. Additionally, the velocity and safe distance (distance to the nearest obstacle) of the improved DWA are compared with the fixed DWA (β = 0.8, γ = 0.3) during the operation of the robot, as shown in Figure 6.

From Figure 6, it can be seen that the improved DWA has longer safe distance as well as higher overall velocity, while the velocity varies greatly and is not conducive to smooth operation of the robot by the original DWA. The experimental data are organized as shown in Table 5.

As can be seen from Table 5, the improved DWA reduced the number of sampling steps and path length by 35.42% and 1.28%, respectively, and enlarged the minimum safe distance by 62.08% compared to the fixed weights (β = 0.8, γ = 0.3) by DWA.

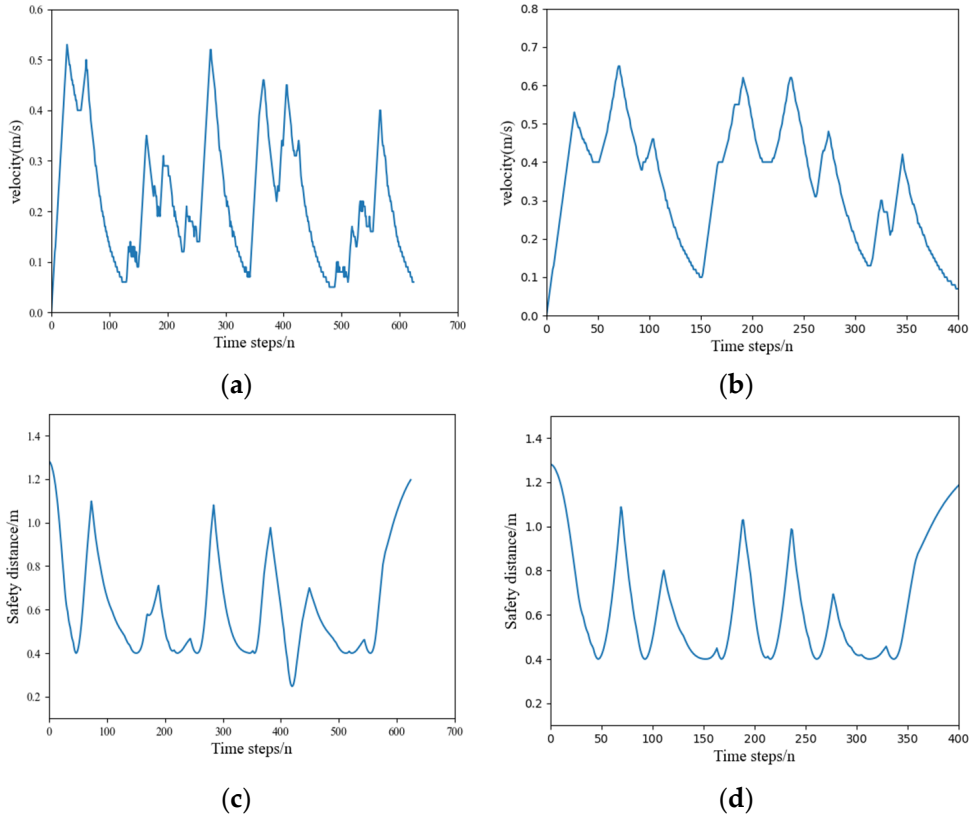


Figure 6. Comparison of velocity and safe distance on Map 1. (**a**) Velocity of the robot by DWA; (**b**) velocity of the robot by improved DWA; (**c**) safe distance of the robot by DWA; (**d**) safe distance of the robot by improved DWA.

Table 5. Map 1 experimental data.

	Minimum Safe Distance/m	Time Steps/n	Path Length/m
$\beta = 0.8, \gamma = 0.3$	0.24767	624	14.01
$\beta = 1.2, \gamma = 1.3$	0.40027	947	14.57
$\beta = 1.7, \gamma = 0.4$	/	/	/
dynamic weight	0.40143	403	13.83

4.2. Complex Obstacle Environment

To evaluate the performance of the improved DWA in complex environments, Map 2 is set with more obstacles of different shapes and sizes. The target point position is set to (11, 11), and the experimental results are shown in Figure 7.

As shown in Figure 7, when the fixed weights are set to $\beta = 1.3$, $\gamma = 0.5$ or $\beta = 1.2$, $\gamma = 1.3$, the original DWA fails to plan the path due to the inability to bypass the U-shaped obstacle near the target point. When the fixed weights are set to $\beta = 0.5$, $\gamma = 0.3$, the planned path is close to obstacles and has bad security. Additionally, the large turning points also affect the smooth motion of the robot. However, the improved DWA does not require setting weight values. It can adaptively adjust the weight values based on complex environments and its own motion state, and plan a smooth and safe path in Map 2.

On the other hand, the velocity and safe distance (distance to the nearest obstacle) of the improved DWA and original DWA with fixed weights ($\beta = 0.5$, $\gamma = 0.3$) during operation are compared, as shown in Figure 8.

From Figure 8, it can be seen that the improved DWA has a longer safe distance with less overall speed change and moves more smoothly, while the number of traveling steps are less. The experimental data of Map 2 are organized as shown in Table 6.

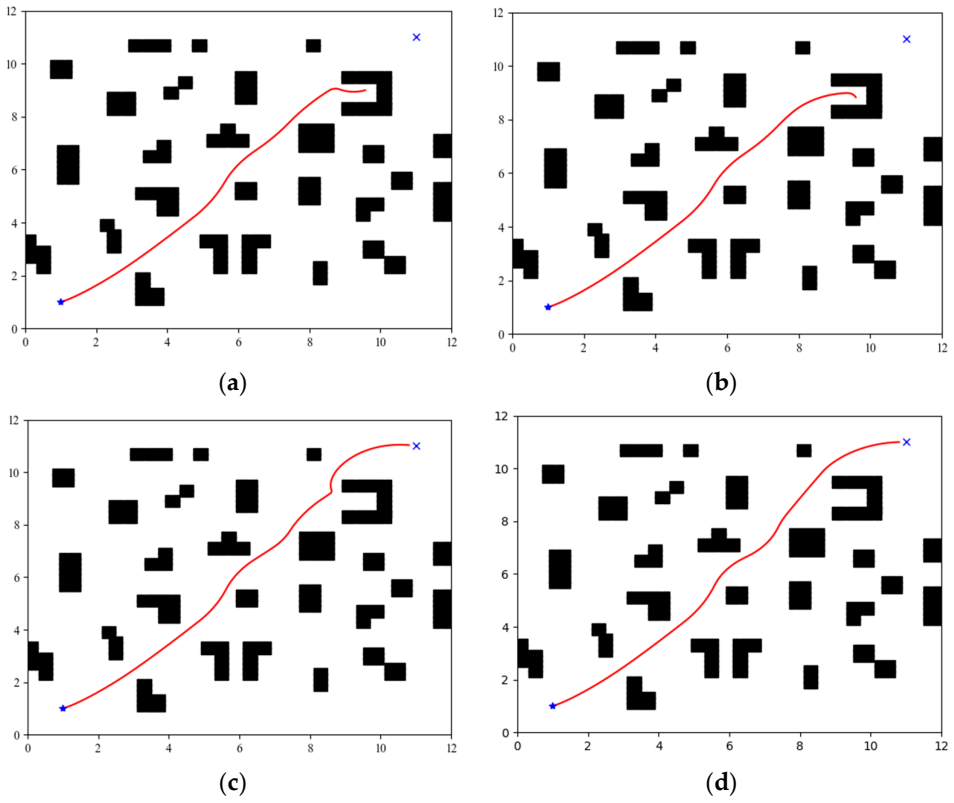


Figure 7. Map 2 simulation results. (**a**) $\beta = 1.3$, $\gamma = 0.5$; (**b**) $\beta = 1.2$, $\gamma = 1.3$; (**c**) $\beta = 0.5$, $\gamma = 0.3$; (**d**) dynamic weight.

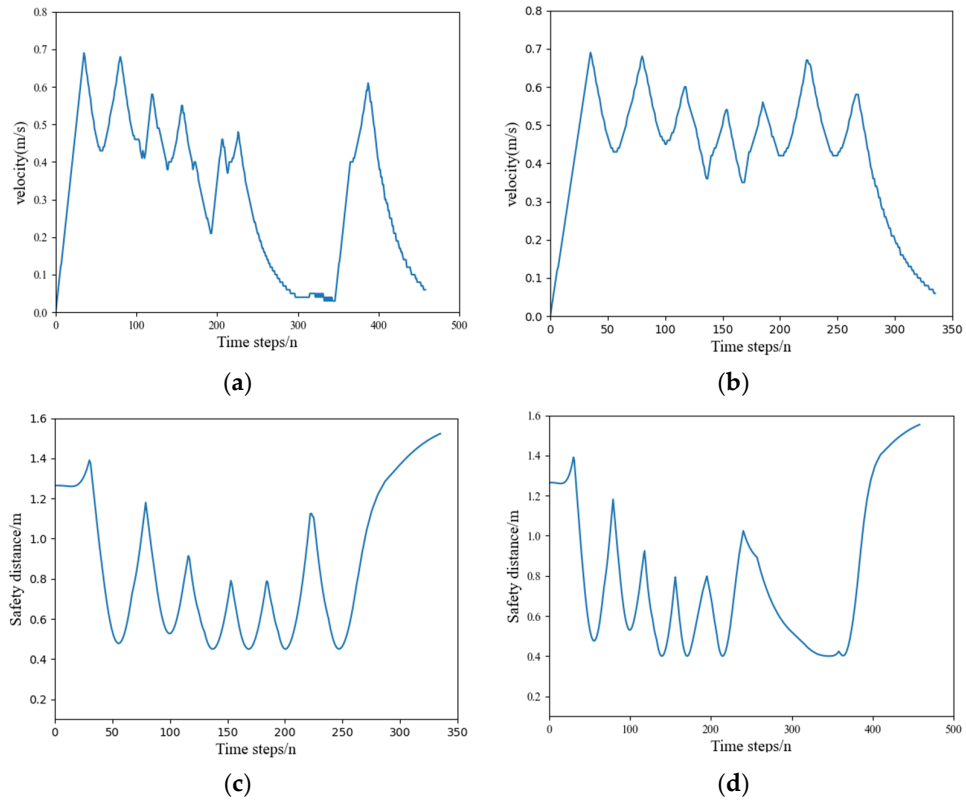


Figure 8. Comparison of velocity and safe distance on Map 2. (**a**) Velocity of the robot by DWA; (**b**) velocity of the robot by improved DWA; (**c**) safe distance of the robot by DWA; (**d**) safe distance of the robot by improved DWA.

Table 6. Map 2 experimental data.

	Minimum Safe Distance/m	Time Steps/n	Path Length/m
$\beta = 1.3$, $\gamma = 0.5$	/	/	/
$\beta = 1.2$, $\gamma = 1.3$	/	/	/
$\beta = 0.5$, $\gamma = 0.3$	0.40	458	14.62
dynamic weight	0.45	335	14.44

From Table 6, the improved DWA reduces sampling steps and path length by 26.86% and 1.23%, respectively, compared to the original DWA with fixed weights ($\beta = 0.5$, $\gamma = 0.3$), and improves safe distance by 12.5% in Map 2.

4.3. Random Obstacle Environment

To evaluate the ability of the improved algorithm to deal with unknown obstacles, random obstacle avoidance simulations were conducted.

Figure 9 shows the results of the improved DWA in a random obstacle environment. The locations of the starting and target points are the same as Figure 7. The red dashed line in Figure 9a represents the path planned by the improved DWA in the original Map 2 environment without adding random obstacles. To verify the obstacle avoidance ability of the improved DWA in the random obstacle environment, three randomly shaped obstacles were added to the planned path, which are shown as blue squares in Figure 9. Figure 9b–d show the obstacle avoidance process of the improved DWA. It can be seen that the algorithm proposed in this paper has strong obstacle avoidance ability and a relatively long safe distance from random obstacles. The safe distances from the obstacles are shown in Figure 10. It can be seen that the safe distance of the path planned by the algorithm proposed in this paper remains larger than 0.3 m, which represents good security.

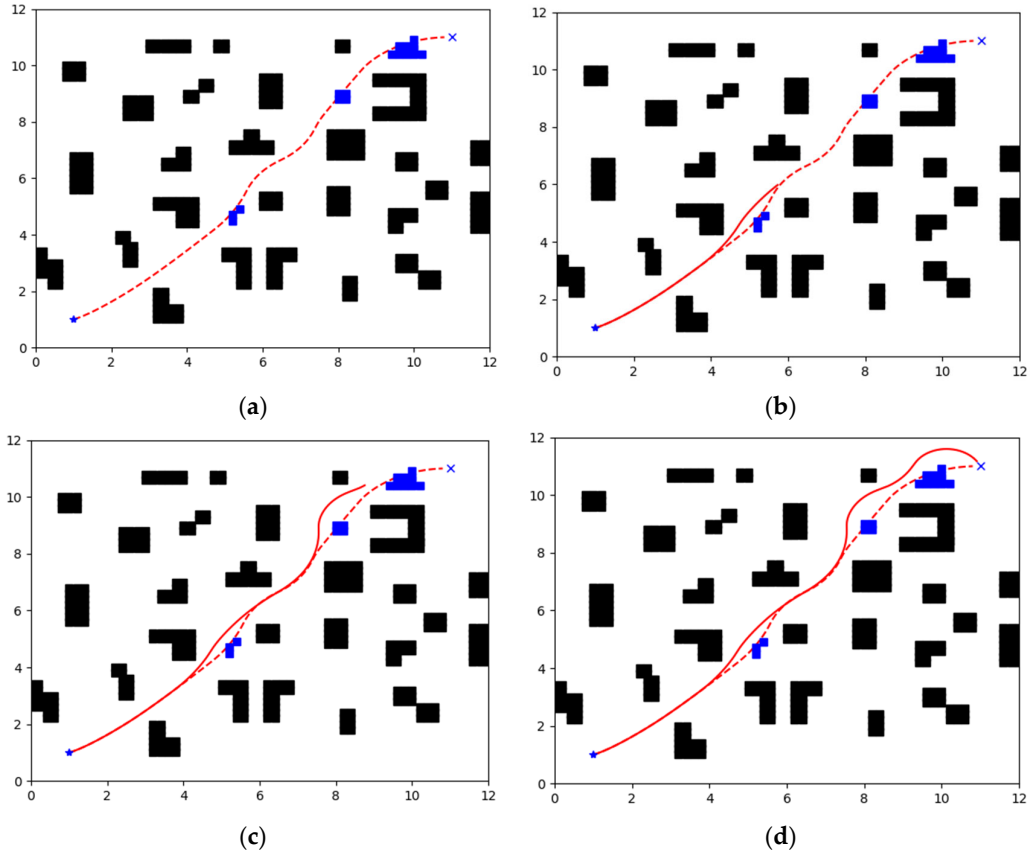


Figure 9. Obstacle avoidance results. (**a**) Add random obstacles; (**b**) avoid the first random obstacle; (**c**) avoid the second random obstacle; (**d**) avoid the third random obstacle.

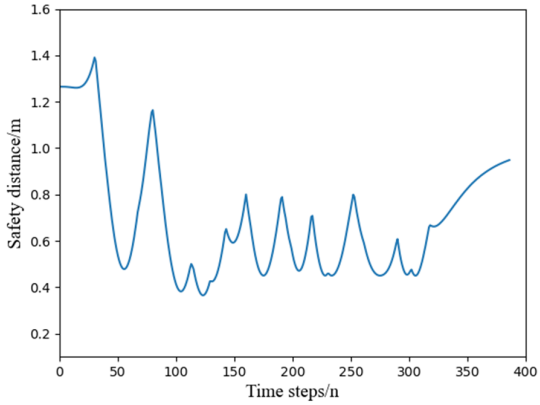


Figure 10. Safe distance diagram.

Based on the above simulation experiments, it can be concluded that the improved DWA proposed in this paper can effectively enhance the rationality of path selection and safe distance in variable obstacle environments, as well as the smoothness and mobility efficiency.

5. Conclusions

A dynamic weight adjustment DWA has been proposed to address the issues of irrational path selection, poor security, and large velocity fluctuations in complex and variable obstacle environments encountered by the original DWA. The proposed algorithm is improved as follows:

- The velocity cost subfunction of the original DWA is improved so that the path with gentle speed change among the candidate paths is more likely to be selected and the robot runs more smoothly.
- The target deviation cost subfunction is added to evaluate the candidate paths, and the robot's ability to navigate to the target is enhanced, so the path length is shorter.
- The fuzzy logic algorithm is used to adaptively and dynamically adjust the weight value of the cost function according to the environmental information, so the robot can drive to the target point at a higher speed in the area far from the obstacle and can pass smoothly and safely in the dense obstacle area.

The simulation results show that the improved DWA proposed in this paper can adapt to a complex obstacle environment. Compared with the original DWA, the velocity, safe distance, and path length are significantly improved.

Author Contributions: Conceptualization, F.W. (Fangbin Wang) and X.G.; methodology, X.G. and Y.G.; software, Y.G.; validation, X.G. and W.Z.; investigation, X.G. and Y.G.; data curation, X.G.; writing—original draft preparation, X.G. and Y.G.; writing—review and editing, F.W. (Feng Wang), Y.L. and D.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Anhui Natural Science Foundation (2008085UD09); Anhui University Collaborative Innovation Project (GXXT-2021-010); Anhui Construction Plan Project (2022-YF016, 2022-YF065, 2023-YF050); Anhui Province Higher Education Science Research Project (2022AH040044); Anhui Province University Outstanding Youth Research Project (2022AH020025); Anhui Province University Outstanding Young Talents Support Program (gxyq2017025).

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Acknowledgments: We would like to acknowledge the support from the Anhui Natural Science Foundation, Anhui Provincial Department of Education, and Anhui Provincial Department of Housing and Urban Rural Development.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Chang, L.; Shan, L.; Li, J.; Dai, Y. Sliding mode control of T-shaped pedestrian channel. *J. Shanghai Jiaotong Univ. Sci.* **2020**, *25*, 478–485. [CrossRef]
2. Durrant-Whyte, H. Where am I? A tutorial on mobile vehicle localization. *Ind. Robot. Int. J.* **1994**, *21*, 11–16. [CrossRef]
3. Cong, Y.F. The Path Planning Method Research Based on Rolling Optimization Theory. Master's Thesis, College of Communication Engineering, Jilin University, Changchun, China, 2007.
4. Peta, K.; Wlodarczyk, J.; Maniak, M. Analysis of trajectory and motion parameters of an industrial robot cooperating with a numerically controlled machine tools. *J. Manuf. Process.* **2023**, *101*, 1332–1342. [CrossRef]
5. Ding, S.; Chen, M.M.; Wang, M.; Gu, Z.M.; Ren, F. ROV global path planning method based on RRT* algorithm. *Ship Sci. Technol.* **2019**, *41*, 66–73.
6. Li, X.X.; Ma, X.L.; Wang, X.P. A Survey of Path Planning Algorithms for Mobile Robots. *Comput. Meas. Control* **2022**, *30*, 9–19.
7. Lin, Y.Z.; Feng, C.H.; Wang, Z.X.; Wang, Z. UAV design and control with A* algorithm. *Int. Core J. Eng.* **2021**, *7*, 212–224.
8. Qureshi, A.H.; Ayaz, Y. Potential functions based sampling heuristic for optimal path planning. *Auton. Robot.* **2016**, *40*, 1079–1093. [CrossRef]
9. Wang, X.Y.; Yang, L.; Zhang, Y.; Meng, S. Robot path planning based on improved ant colony algorithm with potential field heuristic. *Control Decis.* **2018**, *33*, 1775–1781.
10. Li, X.; Zhong, X.Y.; Peng, X.F.; Gong, Z.H.; Zhong, X.G. Fast ICP-SLAM method based on multi-resolution search and multi-density point cloud matching. *Robot* **2020**, *42*, 583–594.
11. Zhong, X.G.; Zhong, X.Y.; Peng, X.F. Robots visual servo control with features constraint employing Kalman-neural-network filtering scheme. *Neurocomputing* **2015**, *151*, 268–277. [CrossRef]
12. Khatib, O. Real-time obstacle avoidance for manipulators and mobile robots. *Int. J. Robot. Res.* **1986**, *5*, 90–98. [CrossRef]
13. Wen, Y.; Huang, J.S.; Jiang, T.; Su, X.J. Safe and smooth improved time elastic band path planning algorithm. *Control Decis.* **2022**, *37*, 2008–2016.
14. Li, Y.; Wei, W.; Gao, Y.; Wang, D.; Fan, Z. PQ-RRT*: An improved path planning algorithm for mobile robots. *Expert Syst. Appl.* **2020**, *152*, 113425. [CrossRef]
15. Zeng, D.; Chen, H.; Yu, Y.; Hu, Y.; Deng, Z.; Zhang, P.; Xie, D. Microrobot path planning based on the multi-module DWA method in crossing dense obstacle scenario. *Micromachines* **2023**, *14*, 1181. [CrossRef]
16. Ballesteros, J.; Urdiales, C.; Velasco, A.B.M.; Ramos-Jimenez, G. A biomimetical dynamic window approach to navigation for collaborative control. *IEEE Trans. Hum.-Mach. Syst.* **2017**, *47*, 1123–1133. [CrossRef]
17. Wei, B.; Han, S.; Zhang, X. An improved dynamic window approach with environment awareness for local obstacle avoidance of mobile robots. *Int. J. Mech. Mechatron. Eng.* **2019**, *13*, 303–310.
18. Chang, L.; Shan, L.; Jiang, C.; Dai, Y. Reinforcement based mobile robot path planning with improved dynamic window approach in unknown environment. *Auton. Robot.* **2021**, *45*, 51–76. [CrossRef]
19. Wang, Y.X.; Tian, Y.Y.; Li, X.; Li, L.H. Self-adaptive dynamic window approach in dense obstacles. *Control Decis.* **2019**, *34*, 927–936.
20. Fox, D.; Burgard, W.; Thrun, S. The dynamic window approach to collision avoidance. *IEEE Robot. Autom. Mag.* **1997**, *4*, 23–33. [CrossRef]
21. Lao, C.L.; Li, P.; Feng, Y. Path planning of greenhouse robot based on fusion of improved a algorithm and dynamic window approach. *Trans. Chin. Soc. Agric. Mach.* **2021**, *52*, 14–22.
22. Tian, Y.Y.; Li, L.H. Dynamic Window Approach Based on Judgment of Speed Direction. *Agric. Equip. Veh. Eng.* **2018**, *56*, 39–42.