An Improved Dynamic Window Approach for Intelligent Pedestrian Avoidance of Mobile Robot

Bijun Tang[1,2], Kaoru Hirota[1,2], Junkui Wang[1,2], Yaping Dai[1,2], Zhiyang Jia[1,2,*]

[1]School of Automation, Beijing Institute of Technology, Beijing 100081, China
E-mail: 3120180926@bit.edu.cn
[2]State Key Laboratory of Intelligent Control and Decision of Complex System, Beijing 100081, China
E-mail: 3120180926@bit.edu.cn

Abstract. To ensure the safety of pedestrian, an improved dynamic window approach (IDWA) based on the pedestrian's motion information is applied to the mobile robot to navigate in the complex environment with dynamic pedestrian. The robot navigates by the IDWA not only preventing collisions with pedestrian and static obstacle but also not affecting the movement of the pedestrian. Through the Robot Operating System (ROS) with Gazebo simulator, the experiments are carried out to compare the classic DWA with the IDWA. It is shown that the IDWA keeps the distance between the robot and the pedestrian above 0.4 meters, and the robot bypasses the pedestrian along the safe path generated by the IDWA behind the pedestrian. When the pedestrian is close to the static obstacle, the IDWA can still generate a smooth and safe path between the pedestrian and the static obstacle. Applying the IDWA to the mobile robot can solve the problem of automatic robot navigation in public places.

Keywords: Mobile Robot, Robot Navigation, Dynamic Window Approach, ROS Platform

1. INTRODUCTION

The widely used local path planning algorithms are the potential fields (PFs) and dynamic window approach (DWA). The basic idea of the PFs is to regard the movement of the robot in the planning space as a kind of force movement in the virtual force field. The PFs has the advantages of small amount of calculation and easy to understand. However, there may be some trap areas due to the local minima. And there may be oscillations in the paths generated by the PFs before the obstacles and in a narrow passage. In addition, if obstacle is near the target, the robot may not be able to reach the target. Due to the problems mentioned above, different kinds of improvements have been made [1]-[4]. As for the problem of avoiding moving obstacles, an extension of the PFs is proposed, where the relative velocity of the obstacle affects the vector summation in order to avoid dynamic obstacle [5]. The DWA is an autonomous obstacle avoidance algorithm to directly search the optimal velocity of the robot in the velocity space. The algorithm fully considers the robot's physical constraints, environmental constraints and other factors, which is more suitable for the robot's path planning problem. However, since the algorithm does not consider the speed information of the moving obstacle, the robot tends to stop in front of the obstacle or even hits the obstacle, which shows poor performance in safety and intelligence. Molinos et al. propose two adaptations of the DWA by taking time into account and evaluating two level windows [6]. Yu et al. propose a dynamic window with virtual goal method for local obstacle avoidance in dynamic environment [7]. In order to solve the problem of avoiding moving obstacles, various methods based on other algorithms have been proposed. Benzerrouk et al. use the relative velocity of the obstacle to choose the direction of avoidance [8]. In addition, there are many navigation algorithms tend to avoid pedestrian by predicting the movements or actions of pedestrians through vision, laser and other methods [9]-[13].

In view of the existing problems of the DWA mentioned above, the cost function is expanded in the classic DWA to improve the performance of pedestrian avoidance. The IDWA mainly consists of four steps. The first step is to determine whether the robot is in the pedestrian area. If the robot enters the pedestrian area, the second step is carried out to generate a semicircular safe path behind the pedestrian for the robot to go through the pedestrian safely based on the position and speed of the pedestrian. The third step is to modify the path generated in the second step according to the position and the shape of the static obstacle. The fourth step is to add a speed smoothing term C_v and two pedestrian avoidance terms C_{pped} and C_{gped} to the cost function according to the current velocity of the robot and the path generated in third step.

The speed smoothing term C_v of the IDWA can make the robot's speed at the next moment not change much from the current speed, which ensures the smoothness of the movement of the robot. The pedestrian avoidance terms C_{pped} and C_{gped} not only can keep a safe distance between the robot and pedestrian but also lead the robot to navigate along the path that does not cross any path of a moving pedestrian, which makes the movement of the robot not affect the movement of pedestrian. And the safe path generated by the IDWA can change its shape according to the position of static obstacles, which makes the robot prevent collision with pedestrian and static obstacles. The IDWA can make the robot navigate in the complex environment with dynamic pedestrian in a safe and socially acceptable way.

Experiments are carried out with the ROS platform and Gazebo simulator under Linux system, using C++ as the programming language. The leg detector function package is used to detect pedestrian's position and speed. The Rviz is used to visualize the planned path. Firstly, three comparative experiments in different scenarios are carried out to test the performance of the IDWA working in a simple environment with only a pedestrian: 1) Robot and pedestrian meet vertically. 2) Robot and pedestrian meet diagonally forward. 3) Robot and pedestrian meet head-on. These three scenarios almost include all situations where pedestrian and robot meet. The distance between the robot and the pedestrian and the direction chosen by the robot to avoid pedestrian are used to compare the safety of the trajectories planned by the DWA and the IDWA. Secondly, to test the performance of the IDWA working in a complex environment containing both static obstacle and dynamic pedestrian, the robot navigation between the pedestrian and the static obstacle is simulated.

The DWA and the process of expanding the cost function of the DWA are demonstrated in 2. In 3, the results of experiments on ROS are presented to compare the performances of the classic DWA and the IDWA.

2. IMPROVED DYNAMIC WINDOW APPROACH

2.1. Dynamic Window Approach

The classic DWA includes three parts: velocity sampling, trajectory prediction and trajectory evaluation. The algorithm turns the obstacle avoidance problem into a constrained optimization problem in velocity space, which contains two types of constraints: 1) Robot's velocity and acceleration limits. 2) Environment and obstacle constraints. According to these two types of constraints, a constrained two-dimensional velocity space is obtained (see **Fig. 1**). The abscissa is the angular velocity, and the ordinate is the linear velocity. The dark gray area represents the velocities at which the robot will hit the obstacles. The light gray represents the safe velocities. The white area represents the safe velocities that satisfy the velocity and acceleration limits, namely dynamic window.

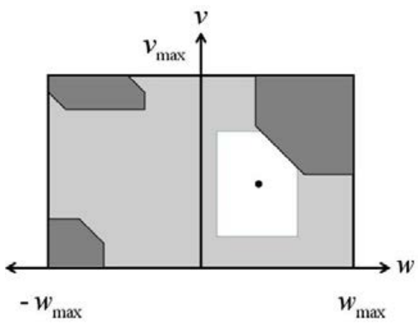


Fig. 1 Velocity space of DWA.

According to the robot's velocity limit, the velocity combination $V_s(v,\omega)$ (v represents linear velocity, ω represents angular velocity) satisfies

$$V_s(v,\omega)=\{(v,\omega)\,|\,0\le v\le v_{\max},-\omega_{\max}\le\omega\le\omega_{\max}\}\,. \quad (1)$$

Due to the limitation of robot dynamics, both the linear acceleration $\dot{v}$ and angular acceleration $\dot{\omega}$ have upper and lower limits. The achievable velocity combination $V_d(v,\omega)$ at the next moment satisfies

$$V_d(v,\omega)=\left\{(v,\omega)\left|\begin{matrix}v_{\text{cur}}-\dot{v}_{\max}\text{d}t\le v\le v_{\text{cur}}+\dot{v}_{\max}\text{d}t\\ \omega_{\text{cur}}-\dot{\omega}_{\max}\text{d}t\le\omega\le\omega_{\text{cur}}+\dot{\omega}_{\max}\text{d}t\end{matrix}\right.\right\},\quad (2)$$

where v_{cur} is the current linear velocity, ω_{cur} is the current angular velocity and $\text{d}t$ is the time interval.

In order to be able to stop before hitting an obstacle under the maximum deceleration, the velocity has a range as follows:

$$V_a(v,\omega)=\{(v,\omega)\,|\,v\le\sqrt{2\text{dis}(v,\omega)\dot{v}},\omega\le\sqrt{2\text{dis}(v,\omega)\dot{\omega}}\}\,,\quad (3)$$

where $\text{dis}(v,\omega)$ represents the distance from the robot to the nearest obstacle.

In the process of predicting the trajectory of the robot, it is assumed that the robot only can move forward or rotate. Since the prediction period is very short, the trajectories can be approximated as straight lines. So, the predicted trajectory in a period of time is expressed as

$$x_{t+1}=x_t+v\cos\theta_t\text{d}t\,,\quad (4)$$

$$y_{t+1}=y_t+v\sin\theta_t\text{d}t\,,\quad (5)$$

$$\theta_{t+1}=\theta_t+\omega\text{d}t\,.\quad (6)$$

There are two versions of the objective function to evaluate the trajectories. The classic version of the objective function is defined as

$$C(v,\omega)=\alpha_0\text{heading}(v,\omega)+\beta_0\,\text{dis}(v,\omega)+\gamma_0\text{v}(v,\omega)\,,\quad (7)$$

where $\text{heading}(v,\omega)$ represents the angle between the robot heading and the target point. This term prizes the trajectories that head the robot towards the goal. $\text{dis}(v,\omega)$ represents the minimum distance from the predicted trajectory to the obstacle. This term increases the security of the trajectories. $\text{v}(v,\omega)$ represents the linear velocity in the trajectory and this term prices moving at high speed. All three terms of the objective function are normalized to $[0,1]$. α_0, β_0, γ_0 are three weight parameters. The velocity combination (v,ω) that maximizes the value of $C(v,\omega)$ is the optimal velocity.

When DWA is used as a local path planning algorithm, there is another version of the objective function:

$$C(v,\omega)=\alpha_1 C_{\text{obs}}+\beta_1 C_{\text{gdis}}+\gamma_1 C_{\text{pdis}}, \qquad (8)$$

where C_{obs} is the sum of grid cell costs through which the trajectory passes. C_{gdis} and C_{pdis} are the estimated shortest distances from the end point of the trajectory to the goal and global path respectively. The cost function can make the robot prefer trajectories that stay far away from the obstacles, go toward the local goal and stay near the global path. This cost function meets our need, so this cost function is used.

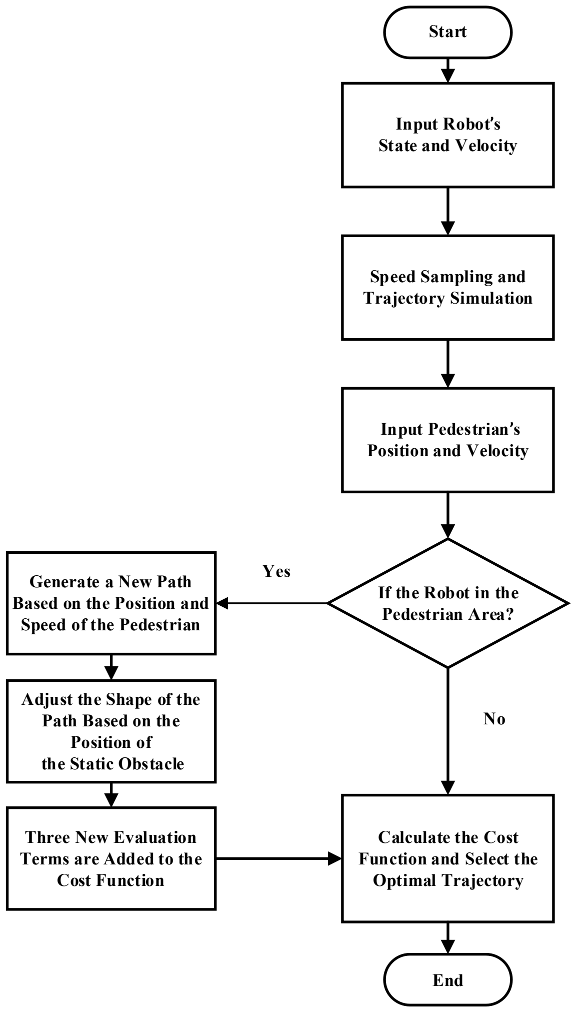

Start
Input Robot's State and Velocity
Speed Sampling and Trajectory Simulation
Input Pedestrian's Position and Velocity
If the Robot in the Pedestrian Area?
Yes
No
Generate a New Path Based on the Position and Speed of the Pedestrian
Adjust the Shape of the Path Based on the Position of the Static Obstacle
Three New Evaluation Terms are Added to the Cost Function
Calculate the Cost Function and Select the Optimal Trajectory
End

Fig. 2 Flow chart of the IDWA.

2.2. Extended Cost Function Based on Pedestrian's Motion Information

The classic DWA considers the physical constraints of the robot and can plan a smooth trajectory, but it has poor performance in avoiding moving pedestrian. This is because the movement of the pedestrian is not considered. So, the motion information of the pedestrian is used to generate a safe path. And three new evaluation terms are added if the robot has entered the pedestrian area. The flow chart of the IDWA in one cycle is shown in **Fig. 2**. The four steps of the IDWA are introduced as follows:

The first step is to determine whether the robot has entered the pedestrian area. The pedestrian area is a circle whose center is the position of the pedestrian and radius is the safe distance R (see **Fig. 3**). If the robot enters the pedestrian area, the second step is carried out. Otherwise, the classic DWA is still used for local path planning.

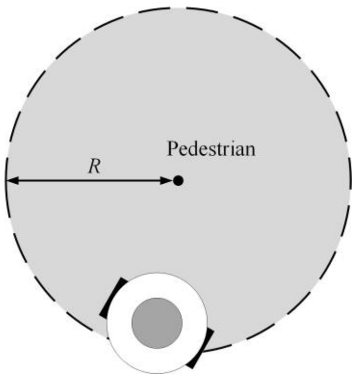


Fig. 3 Pedestrian area.

The second step is to generate a path for the robot to bypass pedestrian based on the position and speed of the pedestrian. The generated path needs to satisfy two principles: 1) The generated path needs to maintain a safe distance from pedestrian. 2) The generated path cannot cross the path of pedestrian. To meet these two conditions, the semicircle behind the pedestrian is chosen as the path for the robot to safely bypass the pedestrian (see **Fig. 4**). The points in the path are expressed as

$$\begin{cases} x_i = x_{\text{ped}} + R\cos(\alpha_{\text{ped}}+\frac{\pi}{2}+i\frac{\pi}{2}), i=0,\ldots,n \\ y_i = y_{\text{ped}} + R\sin(\alpha_{\text{ped}}+\frac{\pi}{2}+i\frac{\pi}{2}), i=0,\ldots,n \end{cases}, \qquad (9)$$

where (x_i, y_i) is the position of the ith point on the path. $(x_{\text{ped}}, y_{\text{ped}})$ is the position of the pedestrian and α_{ped} is the angle of the pedestrian's movement.

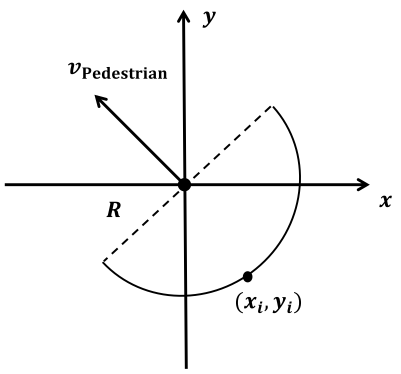


Fig. 4 The path to bypass pedestrian.

The third step is to adjust the shape of the path generated in the second step according to the position of the static obstacles. If some points in the generated path collide with static obstacles, the radius of these points will be reduced to $3/4$ of the original radius. If the points still collide with static obstacles when the radius is reduced to less than half of the original radius, the path will be cancelled. Then a gradient descent smoother is used to make the generated path smoother and safer. The gradient descent progresses stepwise with a step size proportional to the negative gradient of the function. A fixed number of iterations was chosen to optimize the path. The objective function is expressed as

$$P = P_{\text{obs}} + P_{\text{smo}}, \tag{10}$$

$$P_{\text{obs}} = \begin{cases} \omega_{\text{obs}} \sum_{i=1}^{N} (|x_i - o_i| - d_{\text{thr}})^2, & |x_i - o_i| \le d_{\text{thr}} \\ 0 & ,|x_i - o_i| > d_{\text{thr}} \end{cases}, \tag{11}$$

$$P_{\text{smo}} = \omega_{\text{smo}} \sum_{i=1}^{N} (\Delta x_{i+1} - \Delta x_i)^2, \tag{12}$$

where P_{obs} penalizes collisions with obstacles and P_{smo} evaluates the smoothness of the path. o_i is the location of the nearest obstacle to the point x_i, d_{thr} is the threshold for the maximum distance that obstacle can affect the path. ω_{obs} and ω_{smo} are the weights. The obtained path is shown in **Fig. 5**. The dotted line in the figure is the path that has not been optimized by the objective function, and the solid line is the final optimized path.

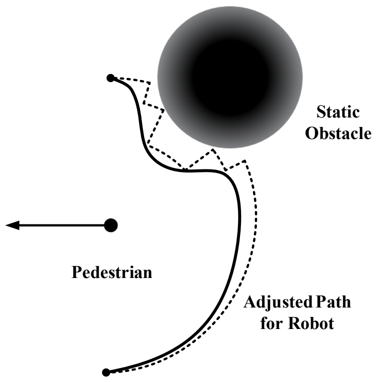


Fig. 5 Adjusted path based on static obstacles.

The fourth step is to expand the cost function of the classic DWA. Two new evaluation terms are added to make the robot drive along the path generated in the third step. The extended cost function is expressed as

$$C(v,\omega) = \alpha_2 C_{\text{pped}} + \beta_2 C_{\text{gped}}, \tag{13}$$

where C_{pped} and C_{gped} represent the minimum distance from the end point of the predicted trajectory to the generated path and the local goal on the generated path respectively. These terms price the trajectories that guide the robot to move along the generated path.

At the same time, in order to ensure the smoothness of the robot's movement, a speed smoothing term C_v is also added to the cost function. This term makes the robot's speed at the next moment not change much from the current speed of the robot. The speed smoothing term and the extended cost function are expressed as

$$C_v = \| v - v_0 \|, \tag{14}$$

$$C(v,\omega) = \alpha_2 C_{\text{pped}} + \beta_2 C_{\text{gped}} + \gamma_2 C_v. \tag{15}$$

The new cost function can not only ensure the smoothness of the forward speed, but also can guide the robot to avoid the pedestrian along the safe path generated behind the pedestrian without collision.

3. EXPERIMENTS ON PATH PLANNING WITH IMPROVED DYNAMIC WINDOW APPROACH

The experiments are done by using the ROS platform and the Gazebo simulator. The Gazebo simulator is used to simulate the robot with differential drive controller, the static environment and the moving pedestrian (see **Fig. 6**). The paths generated by the DWA and the IDWA are visualized in Rviz. The pedestrian is set to walk in a straight line. The leg detector package is used to detect the position and the velocity of the pedestrian, which takes laser scan data as input and uses a machine-learning-trained classifier to find leg-like patterns. The algorithm is implemented in the move-base navigation package, in which the A* algorithm is used as global path planning algorithm.

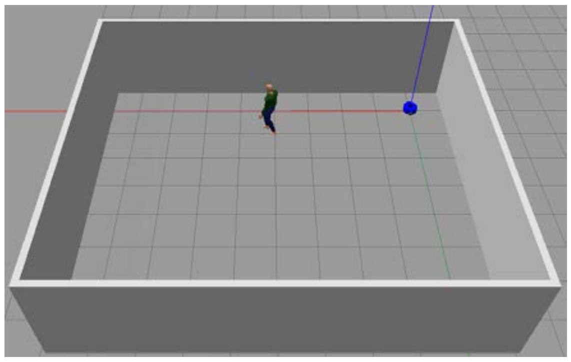
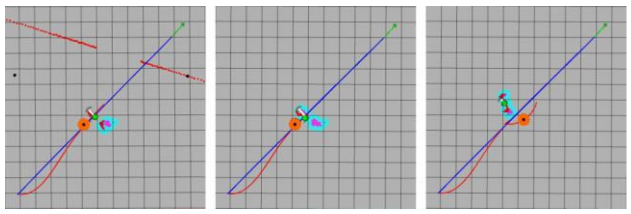


Fig. 6 Simulation experiment environment.

Firstly, to test the performance of the IDWA working in the environment where there is only a pedestrian and no static obstacle, three experiments in different scenarios are conducted: 1) Robot and pedestrian meet vertically (see **Fig. 7**). 2) Robot and pedestrian meet diagonally forward (see **Fig. 8**). 3) Robot and pedestrian meet head-on (see **Fig. 9**). In **Figs. 7–9**, the blue curves are the global paths planned by the A* algorithm. The blue areas around pedestrians are inflation areas. The red lines in front of the robots are the optimal trajectories obtained by the used local path planning algorithms and the red lines behind the robots are the paths that the robots have traveled. The **Table. 1** shows the summary of the data collected during the experiments in the three scenarios. When the robot bypasses the pedestrian safely, the table lists the distances between the robot and the pedestrian. In the experiments, the pedestrian's speed is about 0.3m/s, and the maximum speed of the robot is 0.5m/s. The experiments in three scenarios are introduced as follows:


(a) Path planning based on the DWA

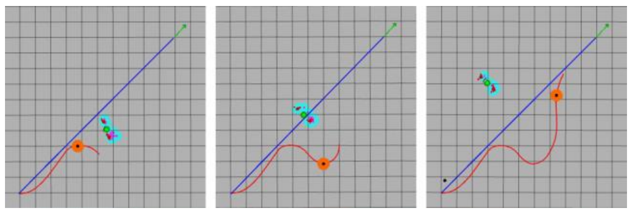
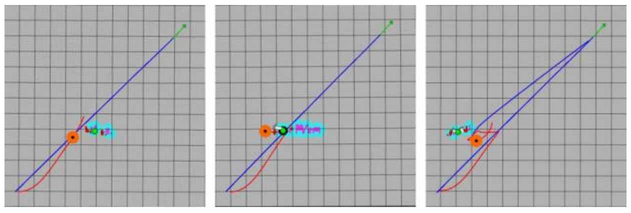

(b) Path planning based on the IDWA

Fig. 7 Robot and pedestrian meet vertically.

First scenario: As shown in **Fig. 7**(a), when the robot and pedestrian meet vertically, the classic DWA starts to avoid pedestrian when the robot is close to the pedestrian. And the robot tends to avoid pedestrian in the wrong direction, which causes the robot and pedestrian to collide. Then the robot waits for pedestrian to pass before continuing planning. This shows that DWA only generates the predicted trajectories within a short time and does not consider the motion of the pedestrian. On the contrary, the IDWA can plan a trajectory in advance without collision. The IDWA makes the robot drive along the path generated behind the pedestrian (see **Fig. 7**(b)). The trajectory planned by the IDWA is safer and more efficient than the trajectory planned by the classic DWA in this scenario.


(a) Path planning based on the DWA

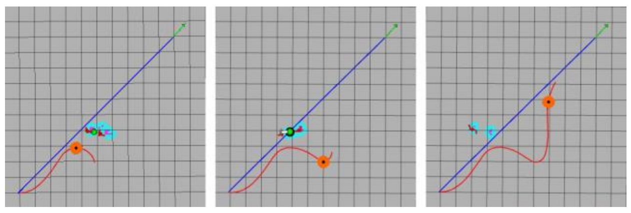
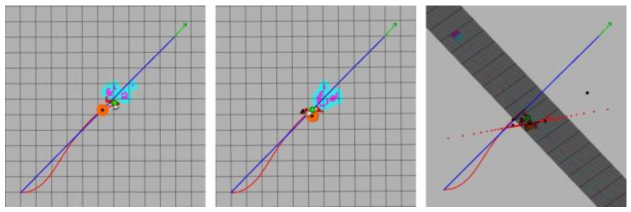

(b) Path planning based on the IDWA

Fig. 8 Robot and pedestrian meet diagonally forward.

Second scenario: As shown in **Fig. 8**(a), when the pedestrian moves from the right front of the robot toward the robot, the classic DWA makes the robot try to avoid the pedestrian. However, since the direction of pedestrian's movement is not considered, the trajectory planned by DWA crosses the path of pedestrian. This causes that the robot and the pedestrian collide and the robot re-plans a global path to continue moving. On the contrary, the IDWA allows the robot to drive along the path on the side that does not affect pedestrian's movement. And the robot quickly returns to the global path after bypassing the pedestrian (see **Fig. 8**(b)). The trajectory planned by the IDWA is safer than the trajectory planned by the classic DWA in this scenario.


(a) Path planning based on the DWA

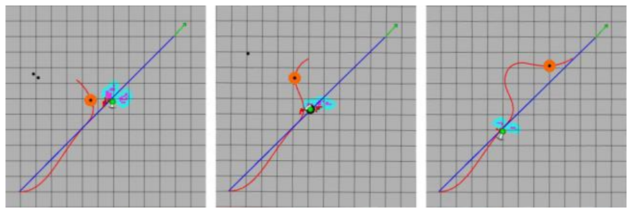

(b) Path planning based on the IDWA

Fig. 9 Robot and pedestrian meet head-on.

Third scenario: As shown in **Fig. 9**(a), when the pedestrian moves towards the front of the robot, the classic DWA does not make the robot avoid the pedestrian in time. Therefore, pedestrian continues to collide with the robot as pedestrian keeps moving forward, and eventually the pedestrian knocks the robot over. On the contrary, when the robot and the pedestrian meet head-on, the IDWA pulls the robot to the side of the pedestrian, avoiding the collision between the robot and the pedestrian (see **Fig. 9**(b)). The trajectory planned by the IDWA is still safer than the trajectory planned by the classic DWA in this scenario.

Table. 1 Results of pedestrian avoidance in **Figs. 7-9**.

	Meet vertically (distance/m)	**Meet diagonally forward (distance/m)**	**Meet head-on (distance/m)**
IDWA	Safe (0.726)	Safe (0.637)	Safe (0.405)
Classic DWA	Collision (robot waits before pedestrian passing)	Collision (robot re-plans a global path)	Collision (pedestrian knocks the robot over)

As shown in the **Table. 1**, the IDWA can help the robot navigate through pedestrian safely in all three different scenarios. The distances between the center of the robot and the pedestrian are above 0.4 meters, while the radius of the robot is 0.2 meters. So, the robot and pedestrian can maintain a safe distance of more than twice the radius of the robot. When classic DWA is used as the local path planning algorithm, the robot and pedestrian collide in three scenarios. The trajectory planned by the IDWA is safer and more efficient than the trajectory planned by the classic DWA.

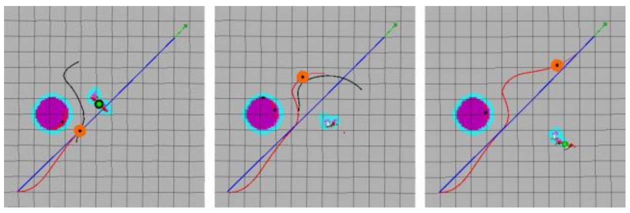


Fig. 10 Navigation in a complex environment.

Secondly, to test the performance of the IDWA working in a complex environment containing pedestrian and static obstacle, the experiment simulates the scenario where there is an obstacle nearby when the robot bypasses pedestrian. In **Fig. 10**, the black lines are the paths generated by the IDWA and the purple circular area is a static obstacle. As shown in **Fig. 10**, when the robot and pedestrian meet vertically and there is a static obstacle near the pedestrian, the path generated by the IDWA is adjusted according to the position and shape of the static obstacle. A safe and smooth path between pedestrian and obstacle is generated by the IDWA. The path not only guides the robot to bypass pedestrian from the side that does not affect pedestrian's movement but also avoids collision with static obstacle. The generated path by the IDWA keeps the robot away from pedestrian and static obstacle at the same time.

4. CONCLUSION

The results of the experiments show that although the DWA fully considers the robot's physical constraints, environmental constraints and other factors, the robot cannot make a timely turn and tends to avoid pedestrian in the wrong direction when the pedestrian approaches the robot. So, the robot tends to stop in front of the pedestrian or even hits the pedestrian, which shows poor performance in safety and intelligence. On the contrary, the IDWA not only allows the robot to avoid collisions, but also keeps the robot and pedestrian at a safe distance of more than twice the radius of the robot in the first three scenarios of the experiments. The IDWA allows the robot to move along the path generated behind the moving pedestrian. And the robot can quickly return to the global path after bypassing the pedestrian. In addition, when the pedestrian is close to static obstacle, the shape of the generated semicircular path is adjusted according to the position and shape of the obstacle. The robot navigates safely and smoothly along the path generated by the IDWA between the pedestrian and the static obstacle.

Compared to the experimental results of the IDWA and classic DWA, the IDWA effectively reduces the risk of collision and keeps a safe distance between the robot and pedestrian while the classic DWA makes robot and pedestrian collide in the first three scenarios. When a walking pedestrian is detected by the mobile robot, the IDWA not only ensures the smoothness of robot's movement by taking the previous velocities of the robot into account but also makes the planned trajectories do not affect the original movement of pedestrian according to the motion information of the pedestrian. When the pedestrian and the static obstacle are very close, IDWA is able to find a safe and smooth path between the pedestrian and the static obstacle. So, the robot can safely bypass pedestrian without collision with static obstacle. And the generated path by the IDWA keeps a safe distance from both pedestrian and static obstacle.

The IDWA is confirmed that it can flexibly bypass a moving pedestrian without collision with pedestrian and static obstacle in the simulation environment. The IDWA will be applied to the service robot in the real environment to improve the performance of automatic navigation among pedestrians in public places.

Acknowledgements

This work is supported by the State Recruitment Program of Global Experts under Grant WQ20141100198.

REFERENCES:

[1] J. Zhao, B. Yan, and R. Ye, "The Flight Navigation Planning Based on Potential Field Ant Colony Algorithm," International Conference on Advanced Control, 2018.

[2] T. Weerakoon, K. Ishii, and A. Nassiraei, "An Artificial Potential Field Based Mobile Robot Navigation Method To Prevent From Deadlock," Journal of Artificial Intelligence and Soft Computing Research, vol. 5, no. 3, pp. 189-203, 2015.

[3] Z. Yu, J. Yan, and J. Zhao, "Mobile robot path planning based on improved artificial potential field method," Journal of Harbin Institute of Technology, 2011.

[4] N. Zhang, Y. Zhang, and C. Ma, "Path planning of six-DOF serial robots based on improved artificial potential field method," 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), Macau, pp. 617-621, 2017.

[5] F. A. Yaghmaie, A. Mobarhani and H. D. Taghirad, "A new method for mobile robot navigation in dynamic environment: Escaping algorithm," 2013 First RSI/ISM International Conference on Robotics and Mechatronics (ICRoM), Tehran, pp. 212-217, 2013.

[6] E. J. Molinos, A. Llamazares, and M. Ocaña, "Dynamic window based approaches for avoiding obstacles in moving," Robotics and Autonomous Systems, vol. 118, pp. 112-130, 2019.

[7] X. Yu, Y. Zhu, L. Lu, et al, "Dynamic Window with Virtual Goal (DW-VG): A New Reactive Obstacle Avoidance Approach Based on Motion Prediction," Robotica, vol. 37, no. 8, pp. 1-19, 2019.

[8] A. Benzerrouk, L. Adouane, and P. Martinet, "Dynamic obstacle avoidance strategies using limit cycle for the navigation of multi-robot system," in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2012.

[9] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch. "Human-aware robot navigation: A survey," Robotics and Autonomous Systems, vol. 61, no. 12, pp. 1726-1743, 2013.

[10] B. Okal and K. O. Arras, "Learning socially normative robot navigation behaviors with Bayesian inverse reinforcement learning," 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, pp. 2889-2895, 2016.

[11] G. Ferrer, A. Garrell and A. Sanfeliu, "Robot companion: A social-force based approach with human awareness-navigation in crowded environments," 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Tokyo, pp. 1688-1694, 2013.

[12] M. Kuderer, H. Kretzschmar, C. Sprunk, and W. Burgard, "Feature-based prediction of trajectories for socially compliant navigation," In Robotics: science and systems, 2012.

[13] T. Randhavane, A. Bera, E. Kubin, A. Wang, K. Gray and D. Manocha, "Pedestrian Dominance Modeling for Socially-Aware Robot Navigation," 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, pp. 5621-5628, 2019.