A wall-following algorithm based on dynamic virtual walls for mobile robots navigation

Xin Wei, Erbao Dong*, *IEEE Member*, Chunshan Liu, Guangming Han, and Jie Yang

***Abstract*— The behavior of wall-following for mobile robots has extended applications, such as intelligent inspection, security patrol and venue service. In this paper, we propose a novel wall-following algorithm based on dynamic virtual walls for mobile robots. The information of environments is firstly obtained by a laser range finder, which is attached to the mobile robot. Then partial sensor data is selected to construct a virtual environment, which is a dynamic virtual wall in this paper. A PD controller is successively utilized to control mobile robots to move along the virtual wall smoothly at adaptively adjusted speed. Meanwhile, obstacle avoidance is also achieved in this one general framework with no switching scheme introduced. Simulations and experiments in various environments have been carried out to verify the effectiveness and evaluate the performance of the proposed algorithm.**

I. INTRODUCTION

Navigation is one of the most fundamental functions of intelligent mobile robots, which enable mobile robots to move autonomously. Lots of efforts have been devoted to solve this problem for its wide application background. Some other sub-problems have been also proposed and studied for decades, such as simultaneous localization and mapping (SLAM) [1], [2], path planning [3], [4], teach and repeat (T&R) [5], [6], wall-following behavior [7] and so on. Wall-following behavior refers to moving along contours of walls or the edge of obstacles and keeping a constant distance between mobile robots and the wall. It is a widely used behavior in mobile robots navigation problems. For instances, when the environment is unknown or partially unknown, wall-following is a simple and functional path planning method, the bug algorithm family is a classical example [8]. If the pose of a mobile robot has been already known, wall-following behavior can be adopted to explore environments and build a map of the environment [9]. Besides, when obstacles are too large for sensors to detect entire contours and thus can't plan a complete path, wall-following is an effective approach to escape the local minimum and perform obstacle avoidance [10]. Moreover, wall-following is an essential behavior in vast kinds of applications, such as sweeping robots, inspection robots, service robots and so on.

The wall-following control of mobile robots is a nonlinear problem, and walls are commonly unknown. Moreover, the sensor data may contain some noise, so it's not easy to design a wall-following algorithm. Many scholars have done a research on this problem, and generally use ultrasonic sensors

*Research supported by National Natural Science Found of China (No. 51275501 and 51105349).

Xin Wei, Erbao Dong*, Chunshan Liu, Guangming Han and Jie Yang are with the Department of Precision Machinery and Precision Instrumentation, University of Science and Technology of China, Hefei, Anhui Province, 230026, China(*corresponding author to provide phone: 086-551-63601482 ; e-mail: ebdong@ustc.edu.cn)

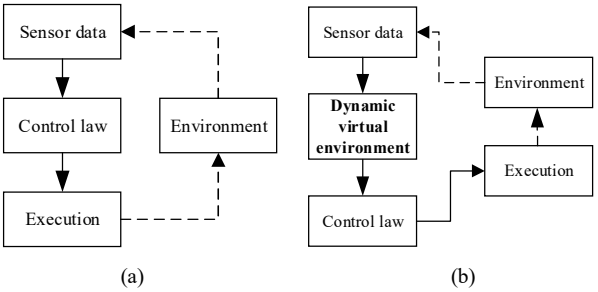


Fig. 1 (a) General flow of previous algorithms (b) Flow chart of our algorithm

and laser ranging finders to perceive the environment. Van [11] firstly presented the perception of wall-following in 1992, and preliminarily achieve the straight-wall-following behavior using sonar sensors. During the development of the last several decades, a large number of algorithm have been proposed andsome of them have been verified in their simulations and experiments. Generally, these algorithm can be divided into two categories. The first category is the reactive wall-following algorithm [12], [13], [14], and [15]. ANDO [12] proposed a finite-state machine based wall-following algorithm using a sonar ring consist of 12 sonar sensors. They clarified the indoor environment into several kinds and designed some if-then rules to control the robot to move along the wall, but no collision avoidance mechanism was introduced. Toibero [13] presented a distance information and odometry based algorithm, meanwhile a novel switching scheme is introduced to avoid possible obstacles. However, only the steering angle was controlled in [13] and the average speed of wall-following seemed a little slow. Charifa [14] proposed a contour-following algorithm guided by artificial potential fields, besides a coupling control of translation velocity and angular velocity was achieved. Nevertheless, a prior knowledge of environment was additionally needed. Matveev [15] presented a sliding mode based strategy and applied it to the problems of patrolling the border of domain and reaching a target through a dynamic environment. Another category is the population-based optimization methods, [7], [16], [17], [18] and [19] fall into this category. Among of this algorithms, a fuzzy controller is usually adopted because it is capable of making inference even under some uncertainty. Carelli [16] proposed an information-filter based wall-following algorithm and applied it to corridor navigation, besides the fuzzy control was firstly introduced to solve wall-following problems. However, this method needs additional odometry information to estimate the relative state between the robot and the wall. In addition, the process of manually designing fuzzy rules seems to be time-consuming. From then on, lots of efforts have been devoted to design fuzzy rules autonomously and generally begin to utilize the typer-2 fuzzy control to increase the robustness. Juang [17] proposed a reinforcement ant optimized a fuzzy controller and applied it to wall-following. Hsu [18] defined three objectives: maintain a constant distance from the wall, collision-free, move smoothly at a high speed, and proposed

a multi-objective, rule-coded, advanced continuous-ant-colony optimization (MO-RACACO) algorithm for fuzzy controller. Chen [19] presented a classification-based learning by particle swarm optimization for wall-following navigation. Both of them proposed an autonomous method to avoid the time-consuming process of designing fuzzy rules and the exhaustive collection of input-output training pairs, for example Q-value aided ant colony optimization in [17], advanced continuous-ant-colony optimization in [18] and particle swarm optimization in [19]. But neither of them have covered the translation velocity control of mobile robots, which is somewhat unreasonable for a mobile robot in real environments. Hsu [7] proposed an evolutionary wall-following control method and used the Type-2 fuzzy controller with species-DE-activated continuous ant colony optimization. Two interval type-2 fuzzy control were designed to respectively control the translation and angular speed of robots.

Though a large past literature on wall-following algorithms, there is an obvious distinction between our and the previous algorithms, Fig. 1 highlights this distinction in detail. This paper is organized as follows. Section II describes perception of the environment, in which part two sets of points are built. Section III describes the process of constructing the virtual environment as well as the control law in detail. Section IV develops various simulations and experiments to verify the effectiveness and evaluate the performance of the proposed algorithm. Finally, Section V draws the conclusion.

II. PERCEPTION OF THE ENVIRONMENT

For simplicity, we assume that the wall that is on the right hand side of the robot is going to be followed. In the process of navigation, only the obstacles in front of the mobile robots are concerned to be avoided, as the obstacles on the right side is regarded as a part of walls. In order to control the robot to navigate along a typical indoor wall safely and smoothly and deal with possible obstacles, there are some objectives to be achieved. First, the distance d_w between the robot and the wall should be maintained in a certain area. Second, the course angle of mobile robots should be as parallel to the wall as possible, otherwise the distance between the robot and the wall will be obviously changed. Meanwhile, the distance d_o between the robot and obstacles ahead should be measured for dealing with possible obstacles.

In order to achieve this behavior, two sets of points $\boldsymbol{P_w}$ and $\boldsymbol{P_o}$ are defined. It should be noticed that all of the perception information is obtained by a laser range finder, which is attached to the robot, and no odometry or encoder information is additionally needed. As shown in Fig. 2, the set of points $\boldsymbol{P_w}$ record the position in the local cartesian coordinate system XOY of the right ahead environment. Rather than directly utilizing these points to control the robot to navigate along the wall, which usually also combines with some control laws, in this paper, the points are firstly utilized to get a virtual representation of the environment, which can be regarded as a process of fitting. In order to reduce the computation cost, only those points that corresponding to these 8 orientations are selected, these orientations are empirically chosen as 0°, -4.5°, -6°, -9°, -18°, -45°, -60°, and -90° since they are effective to fit a wall that should be followed. The distances corresponding to these orientations should be firstly validated, only those measurement distance values that are smaller than the maximum ranging of the laser range finder are regarded as valid measurements. In this paper, the value $\boldsymbol{N}$ is given as the total valid measurements in 8 orientations. Thus the points set $\boldsymbol{P_w}$ can be represented as in (1).

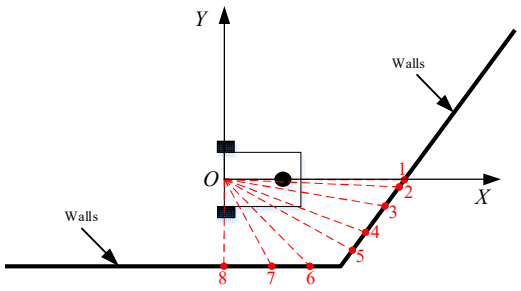


Fig. 2. Those selected 8 orientations in $\boldsymbol{P_w}$

As shown in Fig. 3, the points set $\boldsymbol{P_o}$ record the position of the wall in front of the mobile robot. As similar to the points set $\boldsymbol{P_w}$, three orientations in front of the robot are selected, and these orientations are -18°, 0°, and 18°. It must be noted that there are some differences between the two sets of points $\boldsymbol{P_w}$ and $\boldsymbol{P_o}$. For the points set $\boldsymbol{P_w}$, only the measured distance that is smaller than 5m is regard as validation. However, as for the points set $\boldsymbol{P_o}$, the measured distance value that is larger than 5m is modified to 5m, which can be regarded as the effective distance for the robot. For the convenience of computation, the points set $\boldsymbol{P_o}$ is computed in Polar coordinate and then can be expressed as in (2).

$$P_w = \{(x_i, y_i) \mid i = 1, 2, ..., N\} \tag{1}$$

$$P_o = \{(r_i, \theta_i) \mid i = 1, 2, 3\} \tag{2}$$

III. WALL-FOLLOWING ALGORITHM

In the previous section, two sets of points $\boldsymbol{P_w}$ and $\boldsymbol{P_o}$ have been obtained, which are utilized to construct a local virtual environment. Next the proposed wall-following algorithm will be described combined with these two sets of points.

When the mobile robot is performing the wall-following behavior, it will encounter various kinds of walls, such as straight wall, inner corner wall and outer corner wall. Generally, typical indoor environments can be seen as composing of these three kinds of walls. Therefore, the robot must be able to achieve adaptable wall-following in these three kinds of walls for a safe navigation. The wall-following algorithm presented in this paper is generally divided into two stages. First, those two sets of points are processed to construct a local virtual environment, which is, for simplicity, fitted as dynamic virtual wall in this paper. In this way the robot's state relative to the virtual environment is determined. Then a PD controller is designed and performed to determine the robot's next time step's motion based on the relative state between the robot and the virtual environment.

A. Construction of Local Virtual Environment

In order to improve the average velocity of mobile robots in the process of wall-following, meanwhile guarantee the performance of obstacle avoidance, the robot should be able to adaptively adjust the magnitude of velocity v. In this paper, the velocity v is adjusted based on the distance d_o between the robot and obstacles, which is determined by the points set P_o. In this paper, d_o is defined as the average magnitude

of those three values projected ahead. Thus, d_o can be expressed as in (3) using a simple geometric calculation.

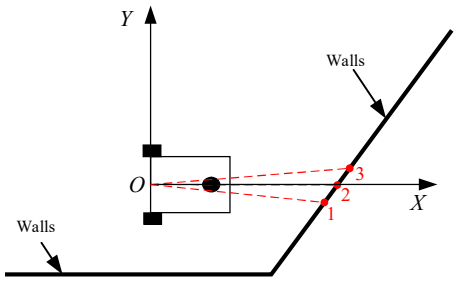


Fig. 3. Those selected 3 orientations in $\boldsymbol{P_o}$

$$d_o = \frac{\sum_{i=1}^{3} r_i \cos\theta_i}{3} \tag{3}$$

As discussed before, the distance between a robot and the wall should be kept in a certain area while the robot is performing wall-following behavior. Besides as shown in Fig. 4, as the differential wheel platform, which is utilized in our simulations and experiments, is thus constrained with nonholonomic constraints, the angle between the robot's course angle and direction of the wall should be also kept at about 0°. Therefore, this wall-following problem turn into a coupling control of d and θ. Now that two sets of points $\boldsymbol{P_w}$ and $\boldsymbol{P_o}$ have been obtained, they are utilized to control d and θ.

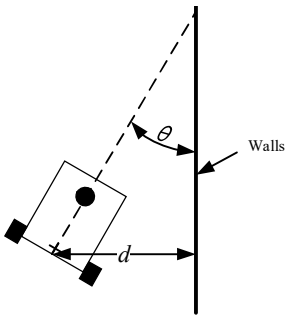


Fig. 4. Decoupling control of d and θ

In order to achieve the wall-following behavior, this paper firstly presents the conception of dynamic virtual wall, which can be regarded as a kind of simplification of virtual environment. Our key idea is that in each scan period, two sets of points $\boldsymbol{P_w}$ and $\boldsymbol{P_o}$ are firstly obtained and then utilized to fit a dynamic virtual wall. Then we can control the robot to follow the dynamic virtual straight wall, which is a much easier problem to be solved. Obviously this kind of thought suit all typical indoor environments. In this paper, as shown in Fig. 5, the points set $\boldsymbol{P_w}$ is utilized to fit a dynamic virtual wall by the least squares method, which can be expressed as in (4), (5), and (6). Note that $\bar{x}$ and $\bar{y}$ are respective the average value of x_i and y_i. Having got the expression of the dynamic virtual straight wall, d and θ can be redefined according to Fig. 4. d is defined as the liner distance between the axle center and the virtual straight wall. θ is defined as the included angle between the course angle of mobile robots and the direction of the virtual straight wall. Thus, the robot's state relative to the virtual environment can been parameterized with three parameters d_o, d and θ.

$$y = ax + b \tag{4}$$

$$a = \frac{\sum_{i=1}^{N} x_i y_i - \frac{1}{N}\sum_{i=1}^{N} x_i \sum_{i=1}^{N} y_i}{\sum_{i=1}^{N} x_i^2 - \frac{1}{N}\left(\sum_{i=1}^{N} x_i\right)^2} \tag{5}$$

$$b = \bar{y} - a\bar{x} \tag{7}$$

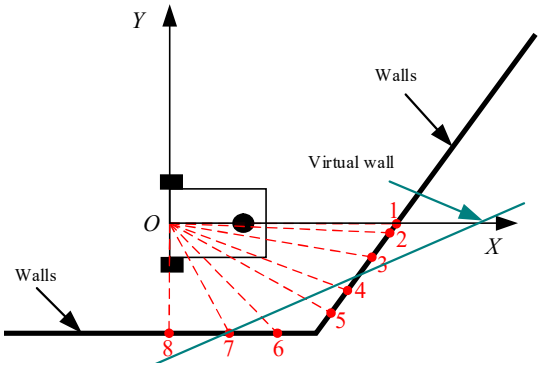


Fig.5 Dynamic virtual wall

B. Control Laws

In the last section, the robot's state relative to the dynamic virtual wall has been determined. Then the velocity (v, w) should be determined according to the relative state.

As discussed before, the robot should adaptively adjust its velocity according to the distance between the robot and obstacles ahead. When there is no obstacles ahead or the distance between the robot and the obstacle ahead is larger than a certain threshold, the robot should move forward at maximum speed. When the distance is smaller than the setting threshold and larger than the safety radius of robots, the robot should adaptively adjust its velocity in some linear or nonlinear method. When the distance is even smaller than the safety radius of robots, the robot should immediately stop to ensure the safety of the robot. Therefore, the linear velocity of mobile robots can be expressed as in (8). Notes that d_o is the distance between the robot and obstacles ahead, r_{safe} is the safety radius of robots, which is set as 0.3 m. And $d_{effecet}$ is the effective distance, which is set as the maximum ranging of the laser range finder. v_{max} is the maximum speed of the robot and it is set as 1 m/s. Obviously v grows linearly with d_o when the value of d_o is smaller than the setting threshold $d_{effecet}$.

$$v = \frac{d_o - r_{safe}}{d_{effect} - r_{safe}} v_{\max} \tag{8}$$

As the linear velocity v of the robot has been determined by (8), the angular velocity w should be determined next. A PD controller is utilized to control the distance between the robot and the wall, which should be kept in the certain threshold area. Thus the input of the PD controller corresponding to angle is still θ, the input of the PD controller is d_1, as shown in (9). It should be noticed that d_w is set as the distance threshold between the robot and the wall, which is going to be maintained.

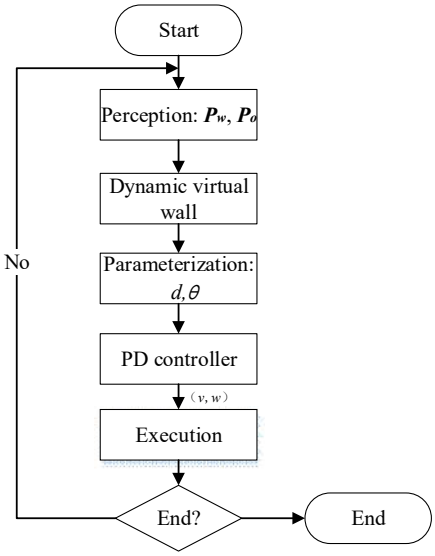


Fig. 6 Flow chart of the proposed wall-following algorithm

Hence the PD controller can be expressed as in (10), (11) and (12). Note that v_l and v_r are respectively the linear velocity of the left and right wheel of the robot, and w is the angular velocity of the robot. b is the length of the robot's axle, which is set as 0.3 m corresponding to the size of robot used in this paper. Besides θ' and d_1' are respectively the variation of d and θ between two successive control time steps.

$$d_1 = d - d_w \tag{9}$$

$$\begin{aligned} v_l &= v - \Delta v \\ v_r &= v + \Delta v \end{aligned} \tag{10}$$

$$\omega = \frac{v_r - v_l}{b} = \frac{2\Delta v}{b} \tag{11}$$

$$\Delta v = (K_{P\theta}\theta + K_{D\theta}\theta') + (K_{Pd}d_1 + K_{Dd}d_1') \tag{12}$$

Hence the velocity v and w of the robot have been determined, the robot then execute the velocity and the wall-following behavior can be achieved in typical indoor environments, a flow chart of the proposed wall-following algorithm is shown in Fig. 6. A more intuitive explain is shown in Fig. 7. Assuming that the wall in front of the robot is a corner, and the robot is initially localized in position 1, the dynamic virtual wall 1 is then obtained. Thus the PD controller control the robot to move along the wall to position 2 and a new dynamic wall 2 is obtained by fitting methods. Therefore the robot move to the position 3 and a wall-following behavior is achieved. Obviously this kind of mechanism suits all typical indoor environments.

IV. Simulations and Experiments

In order to verify the effectiveness, as well as performance of the proposed wall-following algorithm, simulations in various kinds of typical indoor environments have been carried out in the MATLAB environment. Then experiments in real world is carried out to validate the proposed wall-following algorithm using the Wall-Guard platform. As shown in Fig. 8, the Wall-Guard platform is a differential mobile platform and is equipped with a laser range finder, which has a 1 degree angle resolution and 0.2 cm ranging resolution at 5.5 Hz scanning frequency.

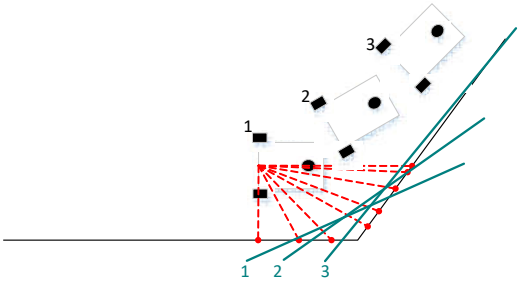


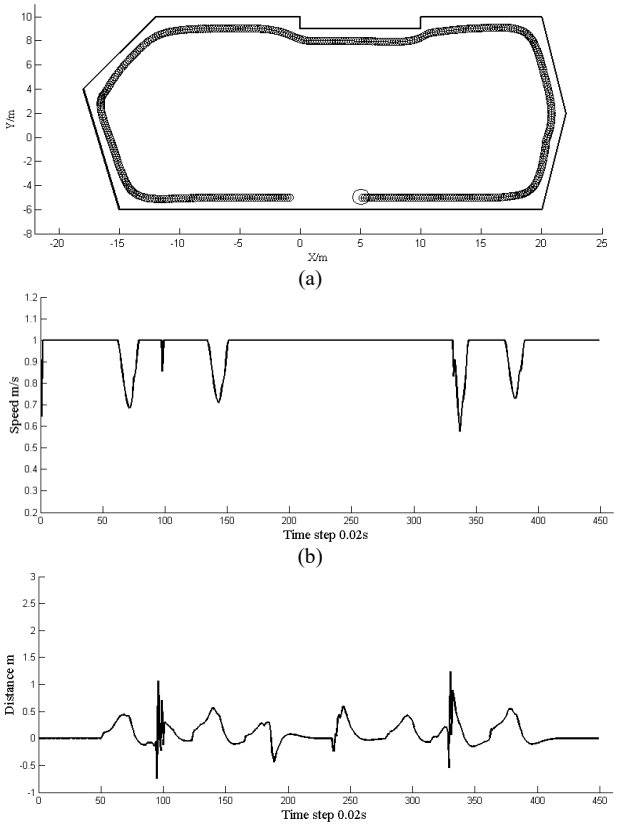
Fig. 7 Illustration of the proposed wall-following algorithm

Fig. 8 Wall-Guard Experimental platform

A. Simulations

In this paper, simulation results in two typical indoor environments are firstly presented. In the process of simulation, the maximum range of the laser range finder is set as 5 m, the maximum linear velocity of the robot is equal to 1 m/s, and the safety radius of the robot is 0.3m, besides the distance threshold between the robot and the wall is equal to 0.8 m. Those four parameters of the PD controller is also predetermined empirically, $K_{p\theta}$ equals to 0.3, $K_{D\theta}$ equals to 0.05, K_{pd} equals to 0.2, and K_{Dd} equals to 0.35. Fig. 9 and Fig. 10 respectively show simulation results in two typical indoor environments. Notes that trajectories shown in the two figure are given by an odometry.

The simulation result of scene 1 verifies the effectiveness of the algorithm proposed in this paper. The trajectory figure verifies the ability of wall-following smoothly, and the speed curve shows translation velocity varies with time. When the robot is following a straight wall, the robot generally moves at maximum speed and the distance error is generally kept in a small area. It should be noticed that the distance curve in this paper refers to the distance error between the robot and the dynamic virtual wall. Besides, the average linear velocity in scene 1 is 0.94 m/s and the average distance error is 0.13 m. Simulation result of scene 2 verifies the ability of obstacle avoidance. The trajectory in Fig .10 shows that the robot have an ability of obstacle avoidance in the process of wall-following. As a comparison with [13], no switching scheme for dealing with possible obstacles is introduced. The function

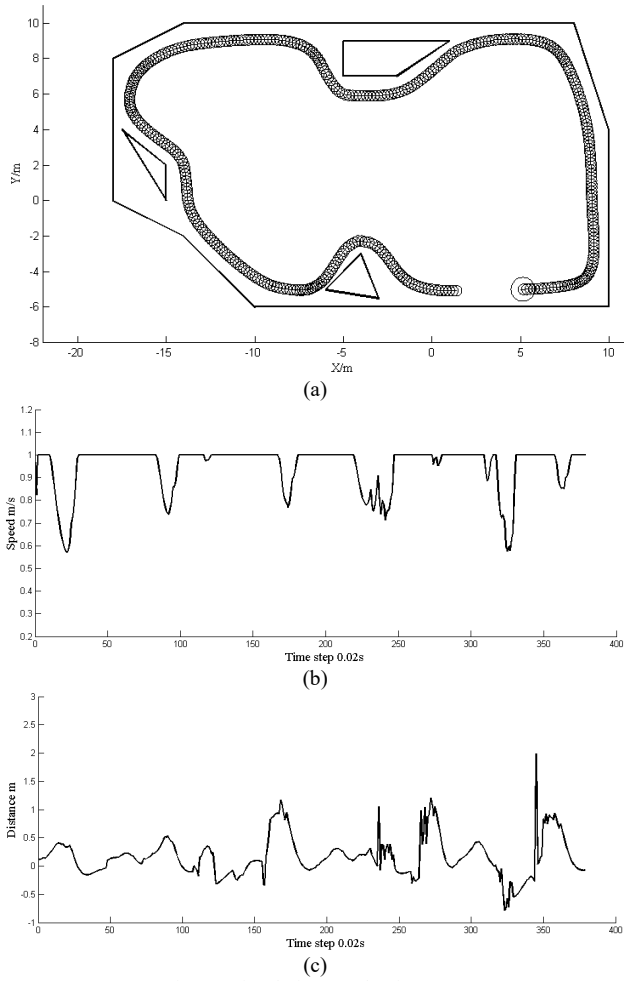


(a)

(b)

(c)

Fig. 9 Simulation result of scene 1



(a)

(b)

(c)

Fig. 10 Simulation result of scene 2

of wall-following and collision free is achieved in only one general framework proposed in this paper. The speed curve in Fig. 10 is similar to the speed curve in Fig. 9. The speed reaches maximum when the robot is following a straight wall and reduces when there is an obstacle or wall ahead. The distance error showed in Fig. 10 seems to be somewhat large. It doesn't mean that our algorithm can't perform wall-following smoothly. It occurs because of the existence of obstacles, which are unpredicted. In other words, we don't have any prior knowledge of the obstacles or walls and thus this situation that the distance error suddenly jump to a big value can't be avoided. What really matters is that the distance decreases as quickly as possible when this situation happens, as shown in the distance error curve in Fig. 10.

B. Experiments and Analysis

In order to evaluate the performance of our algorithm in real world, experiments in various typical indoor environment have been carried out using Wall-Guard mobile platform as shown in Fig. 8. For a mobile robot in real world, there is an obvious time-delay between the speed instruction and the actual execution by motors. In addition, this time-delay has a great influence for the robots' motion, especially for obstacles avoidance. Therefore, the maximum speed in our experiment is given as 0.6 m/s, which is the 60 percent of the maximum speed in simulations. In addition, time step in our experiment is set as 0.1 s.

Fig. 11 is partially selected images of the wall-following algorithm in a real-world experiment using the Wall-Guard platform. And Fig. 12 is respectively the measured value of the robot moving speed and the distance error between the robot and the virtual wall. These two figures verify the effectiveness of the proposed wall-following algorithm in a real world. As can be seen from those two figures (a) and (b) in Fig. 12, when the robot is following a straight wall, moving speed of the robot nearly reaches the maximum and the distance error is close to 0, which means maintaining a constant distance from the wall and moving smoothly at a high speed. When there is an obstacle ahead, the controller control the robot to decelerate and steer, thus the distance error increases at first and converge to 0 rapidly. The moving speed of the robot fluctuate between 0.1 m/s and 0.6 m/s, the distance error fluctuate between -0.1 m and 0.5 m. The average moving speed is 0.32 m/s and the average distance error is 0.18 m.

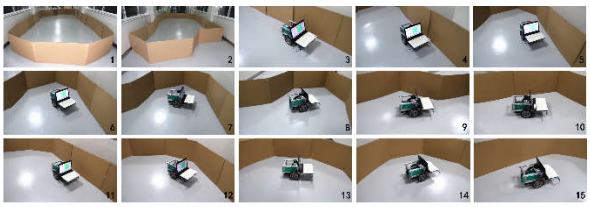
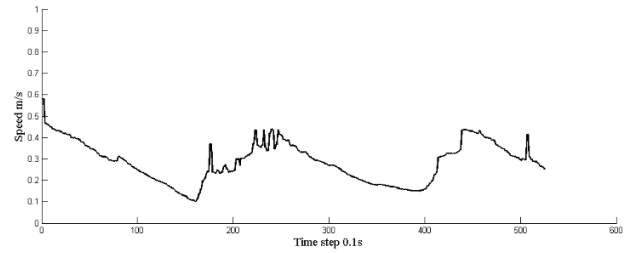
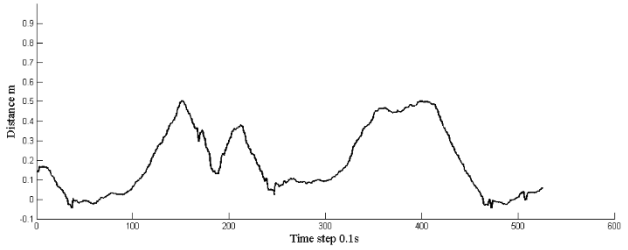


Fig. 11 Partial images of the wall-following algorithm in a real world experiment



(a)



(b)

Fig. 12 (a) Measured value of the robot moving speed at each time step (b) Measured value of distance error between the robot and the dynamic virtual wall

V. CONCLUSION

This paper proposes a novel wall-following algorithm based on dynamic virtual environments. In this approach, a new conception of virtual environments, which is temporarily the dynamic virtual wall in this paper, is presented and applied to the wall-following control of a mobile robot. In addition, a coupling control of moving speed and steering angle is achieved. Meanwhile obstacle avoidance is also processed in this one general framework. Simulations and experiment in various environments have been carried out to verify the effectiveness and performance of the algorithm. In the future, a more complex virtual environment model, rather than the linear fitting will be studied and applied to some other robot control and optimization problems.

REFERENCES

[1] Mazuran M, Burgard W, Tipaldi G D. Nonlinear factor recovery for long-term SLAM[J]. The International Journal of Robotics Research, 2016, 35(1-3): 50-72.

[2] Cadena C, Carlone L, Carrillo H, et al. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age[J]. IEEE Transactions on Robotics, 2016, 32(6): 1309-1332.

[3] Liu C, Dong E, Lu D, et al. Collision-free navigation for mobile robots by grouping obstacles[C]//Robotics and Biomimetics (ROBIO), 2015 IEEE International Conference on. IEEE, 2015: 1686-1691.

[4] Von Hundelshausen F, Himmelsbach M, Hecker F, et al. Driving with tentacles: Integral structures for sensing and motion[J]. Journal of Field Robotics, 2008, 25(9): 640-673.

[5] Cherubini A, Chaumette F. Visual navigation of a mobile robot with laser-based collision avoidance[J]. The International Journal of Robotics Research, 2013, 32(2): 189-205.

[6] Krüsi P, Bücheler B, Pomerleau F, et al. Lighting-invariant Adaptive Route Following Using Iterative Closest Point Matching[J]. Journal of Field Robotics, 2015, 32(4): 534-564.

[7] Hsu C H, Juang C F. Evolutionary robot wall-following control using type-2 fuzzy controller with species-DE-activated continuous ACO[J]. IEEE Transactions on Fuzzy Systems, 2013, 21(1): 100-112.

[8] Ng J, Bräunl T. Performance comparison of bug navigation algorithms[J]. Journal of Intelligent & Robotic Systems, 2007, 50(1): 73-84.

[9] Katsev M, Yershova A, Tovar B, et al. Mapping and pursuit-evasion strategies for a simple wall-following robot[J]. IEEE Transactions on Robotics, 2011, 27(1): 113-128.

[10] Yun X, Tan K C. A wall-following method for escaping local minima in potential field based motion planning[C]//Advanced Robotics, 1997. ICAR'97. Proceedings., 8th International Conference on. IEEE, 1997: 421-426.

[11] Van Turennout P, Honderd G, Van Schelven L J. Wall-following control of a mobile robot[C]//Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on. IEEE, 1992: 280-285.

[12] Ando Y, Yuta S. Following a wall by an autonomous mobile robot with a sonar-ring[C]//Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on. IEEE, 1995, 3: 2599-2606.

[13] Toibero J M, Roberti F, Carelli R. Stable contour-following control of wheeled mobile robots[J]. Robotica, 2009, 27(01): 1-12.

[14] Charifa S, Bikdash M. Adaptive boundary-following algorithm guided by artificial potential field for robot navigation[C]//Robotic Intelligence in Informationally Structured Space, 2009. RIISS'09. IEEE Workshop on. IEEE, 2009: 38-45.

[15] Matveev A S, Wang C, Savkin A V. Real-time navigation of mobile robots in problems of border patrolling and avoiding collisions with moving and deforming obstacles[J]. Robotics and Autonomous systems, 2012, 60(6): 769-788.

[16] Carelli R, Freire E O. Corridor navigation and wall-following stable control for sonar-based mobile robots[J]. Robotics and Autonomous Systems, 2003, 45(3): 235-247.

[17] Juang C F, Hsu C H. Reinforcement ant optimized fuzzy controller for mobile-robot wall-following control[J]. IEEE Transactions on Industrial Electronics, 2009, 56(10): 3931-3940.

[18] Hsu C H, Juang C F. Multi-objective continuous-ant-colony-optimized FC for robot wall-following control[J]. IEEE Computational Intelligence Magazine, 2013, 8(3): 28-40.

[19] Chen Y L, Cheng J, Lin C, et al. Classification-based learning by particle swarm optimization for wall-following robot navigation[J]. Neurocomputing, 2013, 113: 27-35.