International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:13, No:4, 2019

An Improved Dynamic Window Approach with Environment Awareness for Local Obstacle Avoidance of Mobile Robots

Baoshan Wei, Shuai Han, Xing Zhang

***Abstract*—**Local obstacle avoidance is critical for mobile robot navigation. It is a challenging task to ensure path optimality and safety in cluttered environments. We proposed an Environment Aware Dynamic Window Approach in this paper to cope with the issue. The method integrates environment characterization into Dynamic Window Approach (DWA). Two strategies are proposed in order to achieve the integration. The local goal strategy guides the robot to move through openings before approaching the final goal, which solves the local minima problem in DWA. The adaptive control strategy endows the robot to adjust its state according to the environment, which addresses path safety compared with DWA. Besides, the evaluation shows that the path generated from the proposed algorithm is safer and smoother compared with state-of-the-art algorithms.

***Keywords*—**Adaptive control, dynamic window approach, environment aware, local obstacle avoidance, mobile robots.

I. Introduction

LOCAL obstacle avoidance is an important issue for autonomous robot research. Given a start location and a goal location, the robot is asked to move from the start point to the goal and avoid collision with obstacles. The environment is partially known or dynamic [1]. A typical case is that only sensor information is used for the local obstacle avoidance algorithm. Although considerable efforts have been devoted to this field, it remains challenging for the robot to escape from local minima and avoid collision in cluttered environments.

The local obstacle avoidance algorithm can be divided into two categories, that is, environment-based and search-based. The environment-based method generates motion commands based on the goal and the obstacle information. Early remarkable techniques include Artificial Potential Field (APF) [2], Virtual Field Histogram (VFH) [3] and its successor VFH+ [4]. APF is quite simple, but local minima as well as oscillation often happens owing to its nature. VFH and VFH+ address this problem by introducing the polar histogram. The polar histogram represents the distribution of virtual forces of surrounding obstacles. Then, a guide direction can be deduced from the polar histogram. However, oscillation still exists due to the discontinuity of the guide direction. Reference [5] proposes an improved Artificial Potential Method (I-APF). The author claims that by adding the turning-around strategy, the robot is able to escape from local minima. Improved Follow the Gap (I-FGM) method is proposed in [6]. The geometry information of obstacle gaps is utilized to determine the best motion direction. The consideration of kinematic constraints of the robot is insufficient in nearly all environment-based method. They pay more attention on constraints of the environment.

On the contrary, the search-based method generates feasible paths based on kinematic constraints of the robot and modify it by collision check. Early studies include Curvature Velocity Method (CVM) [7] and Dynamic Window Approach (DWA) [8]. Nevertheless, they suffer the trouble of local minima and high risk of collision in cluttered environments. Later researchers try to compute with local methods recursively to expand all possible robot paths after several steps. The motion command that can lead the robot to the best path is chosen. Virtual Field Histogram with Look-Ahead Verification (VFH*) [9] and Dynamic Window Approach with Look-Ahead Verification (DWA*) [10] are two typical algorithms. The cost is that more time is spent on path finding. References [11] and [12] aim to find feasible and collision free paths based on random sampling in the state space. However, deciding an accurate distance measurement is a non-trivial work. References [13] and [14] propose methods based on the state lattice. Even though the state lattice has a powerful ability to generate a feasible path under differential constraints, it requires a high fidelity dynamic model of the system and the sampled path is sub-optimal.

The proposed method in this paper combines environment-based method (VFH+) and search-based method (DWA) to deal with the problem of local minima and path safety. Local goal and adaptive control are proposed to integrate the environment information into the algorithm. The local goal is within the sensory area of the robot and moving with the robot. It helps the robot to jump out of local minima and moving toward the final goal. The adaptive control leads to adaptive speed and adaptive attention on obstacle avoidance according to the environment. It accounts for the safety in cluttered areas. Apart from that, the proposed algorithm inherits the kinematic constraint in DWA, thus, the path generated is smooth and easily for the robot to execute.

The paper is organized as follows. Next section briefly describes key metrics to characterize the environment. Section III detailed explains the proposed algorithm. Performance evaluation is carried in Section IV. Finally, conclusions are given in Section V.

Baoshan Wei is with the Beijing University of Posts and Telecommunications, China (e-mail: wbsbupt@bupt.edu.cn).

II. Environment Characterization

We borrow the polar histogram from VFH+ and furthermore derive another two quantitative descriptors of the environment, i.e., opening spaciousness and "corridor" length.

A. Polar Histogram

The polar histogram proposed in VFH+ [4] is a modification of that in VFH [3]. It successfully represents the distribution of obstacles by using the magnitude, i.e., the virtual force. In the meantime, the geometry of the robot is integrated into the distribution.

An example of the robot and obstacles is shown in Fig. 1. In this paper, the world frame is chosen as the fixed frame and counter-clockwise is adopted as the positive direction. Besides, $0°$ refers to the positive direction of x-axis. The robot position coordinate is expressed as $x^{(r)} = (x_r, y_r, \theta_r)$. An obstacle with coordinate (i, j) in the grid map is denoted as $obs_{i,j}$. Each obstacle is enlarged by the robot radius (r_{robot}), which is the distance from the robot center to its furthest perimeter point. For safety insurance, the obstacle is further enlarged by r_{safe}. The radius of the solid circle and the radius of the dashed circle around the obstacle are r_{robot} and r ($r = r_{robot} + r_{safe}$) respectively. The distance to the robot is $d_{i,j}$ and the relative direction to the robot is $\beta_{i,j}$. The surrounding direction of the robot is divided into sectors with resolution of θ_{reso}. The default θ_{reso} is configured as $5°$, thus, 72 sectors are generated. The sensory area of the robot is limited within the large dashed circle of whom the radius is r_{act}. Fig. 2 shows the normalized polar histogram in Fig. 1.

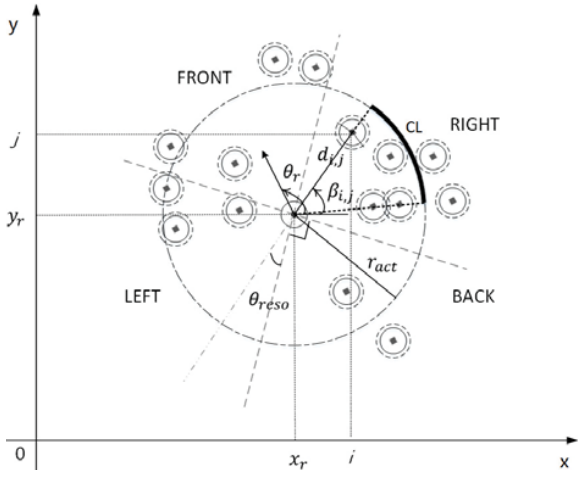


Fig. 1 Robot and obstacles representation in world frame

B. Opening Spaciousness

Although the polar histogram provides detailed obstacle distribution information around the robot, it is not suitable for compressing the overall information of the surrounding environment.

As shown in Fig. 1, the local active region (the large dashed circle) is divided into four sections in the robot frame:

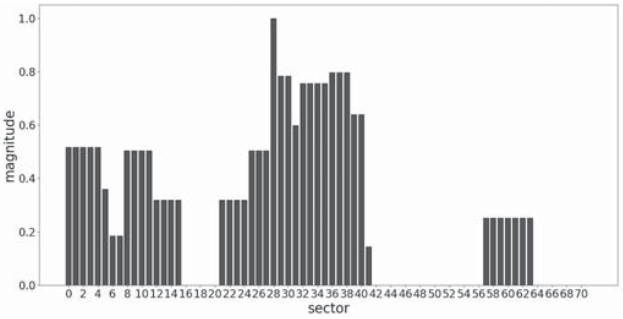


Fig. 2 Polar histogram corresponding to robot position in Fig. 1

left section ($[135°, 225°]$), front section ($[45°, 135°]$), right section ($[-45°, 45°]$) and back section ($[225°, 315°]$). For convenience, they are numbered as section 1, 2, 3 and 4, respectively. Since the robot is restricted to move froward, the back section is ignored.

Three quantitative metrics are built to characterize the obstacle property in the i^{th} section ($1 \leq i \leq 3$).

1) Available distance for the robot to move: The median value of clearances from obstacles of the robot, denoted as Q_i, is adopted to represent the feature. Compared with the average value and the extreme value, the median value is less sensitive to outliers. The local active region is considered as spacious if this metric is large.
2) Dispersion of the obstacle distribution: The interquartile range of the surrounding obstacle directions, denoted as D_i, is used. It reflects whether the surrounding obstacles are gathered in specific direction or dispersed in many directions. The local active region is considered as spacious if this metric is small.
3) Obstacle density: The amount of space occupied by obstacles, denoted as ρ_i, is used to characterize the density. The local active region is considered as spacious if this metric is small.

After that, they are combined into one metric, namely, the opening spaciousness (P_i) in section i, which is defined as:

$$P_i = \frac{\sigma \cdot Q_i - (1-\sigma) \cdot \log(D_i)}{\rho_i}, \tag{1}$$

where σ is a balanced weight factor of available distance and dispersion of the obstacle distribution. The introduction of $\log$ function aims to scale D_i to be comparable with the other two metrics.

Then, the spaciousness metrics from three sections are merged into one metric, i.e., the spaciousness of the local active region, which is denoted as P:

$$P = \sum_{i=1}^{3} \delta_i \cdot P_i, \tag{2}$$

where δ_i is the linear weight factor of the i^{th} section.

Owing to the fact that P is updated in real time, drastic value change may happen. In order to prevent this problem, a simple filter is recommended, which is defined as:

$$\hat{P}^{(n)} = \epsilon \cdot P^{(n)} + (1-\epsilon) \cdot \hat{P}^{(n-1)}, \tag{3}$$

where $\hat{P}^{(n)}$ and $P^{(n)}$ are smoothed spaciousness and raw spaciousness of the local active region respectively at the n^{th} control loop. A smooth factor, denoted as ϵ, is introduced to serve as a filter weight factor. The larger it is, the more weight is given to current observed spaciousness.

C. "Corridor" Length

It is beneficial for the robot to recognize corridors in indoor environments. A metric (CL) is conducted to satisfy this demand, which is defined as:

$$CL = r_{act} \cdot \max(\beta^l_{max} - \beta^l_{min}, \beta^r_{max} - \beta^r_{min}). \quad (4)$$

As shown in Fig. 1, CL can be thought as the corridor length when a number of close obstacles are aligned. β^l_{min}, β^r_{min}, β^l_{max} and β^r_{max} are the minimum direction of obstacles on the left side, the minimum direction of obstacles on the right side, the maximum direction of obstacles on the left side and the maximum direction of obstacles on the right side respectively.

III. EA-DWA: Environment Aware Dynamic Window Approach

The proposed Environment Aware Dynamic Window Approach (EA-DWA) will be explained in this section. It is within the Model Predictive Control framework. In order to add the "Environment Aware" property to DWA, two strategies, i.e., the local goal and the adaptive control are proposed. The local goal is created to replace the global goal in DWA. It can better guide the robot through complex environments. The adaptive control accounts for adaptive speed and adaptive attention on obstacle avoidance. When the surrounding area is cluttered, the robot will actively slow down and pay more attention on obstacle avoidance, otherwise, it will speed up and focus more on the local goal. The algorithm detail is shown in the pseudo.

The robot is differential driven, whose motion model is defined as:

$$\begin{aligned} \dot{x}(t) &= (v_{left}(t) + v_{right}(t))cos\theta(t)/2 \\ \dot{y}(t) &= (v_{left}(t) + v_{right}(t))sin\theta(t)/2 \\ \dot{\theta}(t) &= (v_{right}(t) - v_{left}(t))/L, \end{aligned} \quad (5)$$

where $v_{left}(t)$, $v_{right}(t)$ and $\theta(t)$ are the translational velocity of the left wheel, the translational velocity of the right wheel and the heading direction of the robot respectively. The translational velocity of the robot is $v = (v_{left}(t) + v_{right}(t))/2$. L is the distance between the left wheel center and the right wheel center.

A. Local Goal Position

The local goal acts as a guide to lead the robot. Two steps are needed to decide its position: firstly, choosing the best heading, secondly, determining the proper position according to the selected heading.

Algorithm 1 EA-DWA: Environment Aware Dynamic Window Approach

Input: grid map map, initial robot position $start(x_0, y_0, \theta_0)$, goal position $goal(x_t, y_t)$ and distance tolerance to the goal d_{tol}
Output: $path$=$[(x_0, y_0), (x_1, y_1), ..., (x_n, y_n)]$ from $start$ to $goal$

```
x := start; path := []
while d(x, goal) < d_tol do
    add x to path
    localmap := calcLocalMap(map, x);
    if localGoalChangeFlag then
        θ_g := calcGoalDirection(x, localmap, goal)
        localgoal := calcGoalPosition(x, θ_g)
    end if
    (v,w) := adaptiveDWA(x, localgoal, localmap)
    x := motion(x, v, w)
end while
Return path
```

1) Local Heading: We adopt the direction selection procedure in VFH+ and make three major improvements:

- Threshold of the polar histogram: Only one threshold is used to classify openings and barriers.
- Candidate sector generation: More candidates are generated.
- Direction cost function: Opening width is added to modify the cost function in VFH+.

After generating a normalized continuous polar histogram, a threshold (τ_h) need to de defined to generate a binary polar histogram. The sector whose magnitude value is below this threshold is considered to be included in the opening. What is different from VFH+ is that only one threshold is applied and no masked polar histogram is generated. The reason is that DWA can model kinematic constraints of the robot better than the masked polar histogram.

The candidate generation in VFH+ is too rough and may leave out better candidate choice, so it is reasonable to increase candidate numbers. We use $width$ to represent the opening width, i.e., the sector numbers in that opening. Three types of openings are defined: very narrow openings ($width < 2$), narrow openings ($2 \leq width < 6$) and spacious openings ($width \geq 6$). The opening width is computed as:

$$width = k_r - k_l, \quad (6)$$

where k_r, k_l are the rightmost and leftmost sector of the opening respectively. The candidate sector is generated as follows. For very narrow openings, there is no candidate sector. For narrow openings, only middle sector ($k_{center} = (k_l + k_r)/2$) is chosen and for spacious openings, candidate sectors are chosen from $k_l + 1$ to $k_r - 1$, with increment of 1.

The candidate spaciousness, denoted as $\mathrm{width}(c)$, is not considered in the cost function of VFH+. Here, the candidate spaciousness means the width of the opening where the candidate is located. A modified cost with candidate spaciousness is defined as:

$$\hat{\mathrm{g}}(c) = \mathrm{g}(c) + \lambda_4 \cdot \mathrm{width}(c), \quad (7)$$

where

$$\mathrm{width}(c) = k_r^{(c)} - k_l^{(c)}, \quad (8)$$

$$g(c) = \lambda_1 \cdot \Delta(c, k_t) + \lambda_2 \cdot \Delta(c, \frac{\theta_r}{\theta_{reso}}) + \lambda_3 \cdot \Delta(c, k_{d,n-1}), \quad (9)$$

$$\Delta(c_1, c_2) = \min\{|c_1 - c_2|, |c_1 - c_2 - s|, |c_1 - c_2 + s|\}, \quad (10)$$

$$s = 2\pi/\theta_{reso}. \quad (11)$$

c is a candidate sector. $g(c)$ is the direction cost function in VFH+. $k_l^{(c)}$, $k_r^{(c)}$ are the leftmost and rightmost sector of the opening in which the candidate c is located respectively. k_t is the sector in which the goal direction is located. θ_r is the current heading direction of the robot. θ_{reso} is the predefined resolution of angles. $k_{d,n-1}$ is the selected candidate sector in previous control loop. Linear weighting factors of the candidate direction function are denoted as λ_i. The constraints of them are defined as:

$$\lambda_1 > 0.5, \lambda_i > 0, i = 2, 3$$
$$\sum_i \lambda_i = 1. \quad (12)$$

2) Local Position: DWA employs the goal heading alignment to deal with the relationship between the robot and the goal. However, the robot gets trapped frequently due to this cost function. In DWA, the robot always tries to align with the goal heading all the time and it severely restricts the sample range in velocity space. Even no such velocity may avoid colliding with obstacles when the robot is surrounded by obstacles. On the contrary, if the heading cost is replaced by distance cost, the problem no longer exists. So only local heading is not enough, a local goal with coordinate (x_g, y_g) is necessary.

As shown in Fig. 3, after choosing the best local goal heading, we need to determine how far (d_g) it is away from the robot current position. The rule is as follows:

$$d_g = \cos(v_{max} - v_r) \cdot r_{act}. \quad (13)$$

d_g is no larger than r_{act}. This constraint accounts for the meaning of "local", which means the local goal is always inside the sensory range of the robot. Besides, it is reactive to velocity change. v_{max} is the maximum translational velocity of the robot. When current speed (v_r) is high, it tends to be more conservative, that is, more close to the robot. When the speed is slow, it tends to be aggressive instead. The motivation is that high speed brings about high risk of colliding with obstacles.

B. Local Goal Update Frequency

It is of no need and even harmful for the local goal to change after each control loop (Δt). First, during Δt, the robot cannot move far and the surrounding environment change is not notable. Second, high update frequency may worsen the discontinuity of the selected local goal direction. The key point is how to determine local goal update frequency. Three metrics are defined to solve this problem, namely, distance traveled, angle rotated (in place or not) and distance to the nearest obstacle.

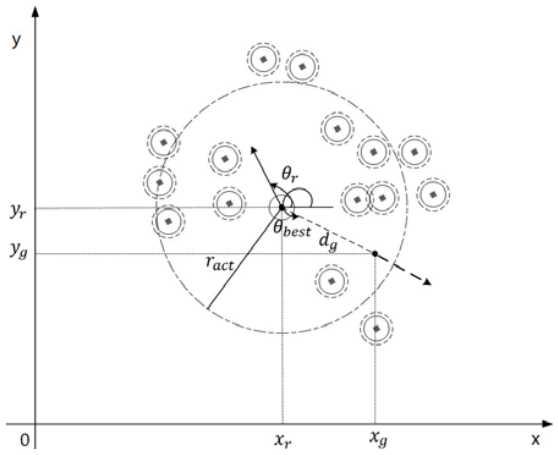


Fig. 3 Local goal position

1) Distance Traveled: If a fixed distance ($\bar{d}_r$) is defined as a threshold of distance traveled, the local goal is not updated until the robot moves so far. In such way, no environment information is utilized. It is hard to manually set this parameter since no one is sure that what distance threshold is best, but some metrics of the environment can be extracted to help the decision. In section II, "corridor length" (CL) is mentioned, which can be added to modify the fixed distance threshold to become a dynamic one ($\hat{d}_r$). The dynamic threshold of distance traveled is calculated by:

$$\hat{d}_r = \min(\bar{d}_r, CL). \quad (14)$$

2) Angle Rotated: When the robot tries to rotate itself, it may happen that some obstacles block the way to the goal, then it is exact the time to update the local goal. However, the threshold of angle rotated is no easy to manually set as well. The barrier width can be utilized though. Barrier means continuous sectors which are of value 1 in binary polar histogram, the way of calculating the i^{th} barrier width ($width_{obs}^{(i)}$) is the same as opening width which is defined in (6). The threshold of angle rotated is expressed as:

$$\hat{\theta}_r = \min_{i \in B}(\theta_{reso} \cdot width_{obs}^{(i)}), \quad (15)$$

where B is the barrier set.

3) Distance to Nearest Obstacle: When the robot travels in cluttered area, it can be very close to obstacles. To ensure safety, a distance threshold must be set, thus the robot is able to correct the local goal more frequently without approaching obstacles too close. Currently, the threshold is defined as:

$$\hat{d}_{obs} = \eta \cdot r, \quad (16)$$

where η is a factor which adjust the sensitivity to obstacles. The value ranging from 1.5 to 3.0 is recommended.

C. Adaptive DWA

1) Adaptive Speed: The speed is vital for obstacle avoidance. The robot must be able to adjust the speed according to the outer dynamic changes, including distance to the final goal and the surrounding environment. The cost

function defined in DWA cannot meet this demand. Besides, the speed cost in DWA keeps speeding up the robot all through the navigation process, which causes high risk of colliding with obstacles and less choice in velocity space.

In the new proposed method, the configured maximum speed (v_{max}) of the possible velocity space V_p becomes dynamic ($\hat{v}_{max}$), and it is calculated by:

$$\hat{v}_{max} = v_{max} \cdot \tanh(d_t/kv_t) \cdot \tanh(\hat{P}/kv_o), \quad (17)$$

where d_t is the distance to the final goal. $\hat{P}$ is local active region spaciousness defined from (1)-(3). kv_t and kv_o are smooth factors of goal awareness and obstacle awareness respectively.

In this way, the possible velocity space V_p becomes adaptive to the goal and the environment. V_p accounts for the velocity limit of the robot. When the robot sample velocities from the space, the candidate velocity becomes adaptive as well.

2) Adaptive Cost: This section presents two modifications: modified cost function and adaptive cost weight.

As mentioned above, the goal heading cost function draws back the DWA performance, thus, a better goal distance cost function is designed to take over it. The goal here means local goal, denoted as (x_g, y_g). Suppose the robot is in position $x^{(r)} = (x_r, y_r, \theta_r)$, and there are n candidate velocities (v_i, w_i), $i \in \{1, 2, ...n\}$. v and w are translational velocity and rotational velocity of the robot respectively. For each candidate, apply the motion model defined in (5) for $T/\Delta t$ times (T is the simulation time), then new robot positions $(\hat{x}^{(1)}, \hat{x}^{(2)}, ...\hat{x}^{(n)})$ can be acquired. The goal distance cost is defined as:

$$\text{goal}(v_i, w_i) = \text{d}(\hat{x}^{(i)}, (x_g, y_g)). \quad (18)$$

Compared with the goal heading cost, the goal distance cost can start up robot without the help of speed cost. The reason is that higher v with direction aligned with the local goal will have a lower cost, thus is preferred among all candidate velocities.

When it comes to the clearance cost, the distance to possible collision obstacles is replaced by euclidean distance to all obstacles in the local active region. In this way, the robot is able to keep away from obstacles rather than just prevent hitting obstacles. The clearance cost is designed as a piece-wise function and defined as:

$$\text{clearance}(v, w) = \begin{cases} 1/r_{act} & \text{if } d_{obs} > r_{act} \\ d_{inf} & \text{if } d_{obs} < r \\ 1/d_{obs} & \text{otherwise} \end{cases}, \quad (19)$$

where

$$d_{obs} = \min_{obs \in OBS} \text{d}(v, w, obs). \quad (20)$$

d_{obs} means the euclidean distance to the nearest obstacle in local active region. d_{inf} is the configured value, which punishes the near collision heavily.

In different scenarios, the robot need to take different reactions. For example, if no obstacle blocks the robot from the goal, then it should be more goal oriented, otherwise, it should pay more attention to obstacle avoidance. The solution is adaptive cost weight with respect to the local active region spaciousness and it is presented as:

$$\hat{\text{f}}(v, w) = \hat{\mu}_1 \cdot \text{goal}(v, w) + \hat{\mu}_2 \cdot \text{clearance}(v, w) + \hat{\mu}_3 \cdot \text{speed}(v), \quad (21)$$

where

$$\text{speed}(v) = 1 - v/v_{max}, \quad (22)$$

$$\begin{aligned} \hat{\mu}_2 &= \hat{\mu}_2^{(0)} \cdot (1 - \hat{\mu}_3 - \tanh(\hat{P}/kp_o)) \\ \hat{\mu}_1 &= 1 - \hat{\mu}_3 - \hat{\mu}_2. \end{aligned} \quad (23)$$

$\hat{\mu}_1$, $\hat{\mu}_2$ and $\hat{\mu}_3$ are goal cost weight, clearance cost weight and speed cost weight respectively. Among them, only $\hat{\mu}_3$ is manually set. The initial clearance cost weight is denoted as $\hat{\mu}_2^{(0)}$. Usually, $\hat{\mu}_3$ is no more than 0.2 and $\hat{\mu}_2^{(0)}$ is larger than 0.5. kp_o is the smooth factor of the adaptive clearance cost weight. The weight factor of the goal cost and the clearance cost become dynamic compared with DWA.

IV. Evaluation

In this section, experimental results are obtained in the simulated environment. Simulation configuration, improvement compared with DWA and comparison with state-of-the-art algorithms are detailed explained.

The simulated environment is built with Gazebo, which serves as a powerful platform in robot research community. Fig. 4 shows the simulation environment in our experiment.

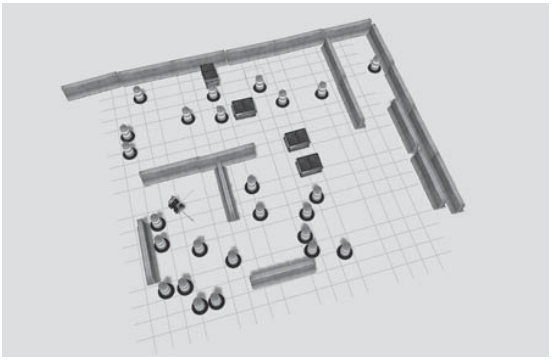


Fig. 4 Gazebo simulation environment

Clearpath Husky robot was used for the algorithm evaluation. It is a commercial robot platform with differential drive. The maximum translational velocity (v_{max}) and maximum rotational velocity (w_{max}) are $1m/s$ and $40°/s$ respectively. The maximum accelerations are $0.2m/s^2$ and $40°/s^2$ for translation and rotation respectively. The robot radius (r_{robot}) is set to $0.5m$ and the safe distance (r_{safe}) is set to $0.2m$. The sensory area (r_{act}) is set to $4.0m$, which is the working range of the low-cost LIDAR.

We compare the proposed EA-DWA with two state-of-the-art algorithms, i.e., I-APF [5] and I-FGM [6]. They also make use of obstacle environment information to escape from local minima. I-APF adds on a virtual force to enable turning-around behavior of the robot, while I-FGM considers the geometry information of obstacles. The algorithm parameters are configured as follows. In EA-DWA,

the control loop (Δt) is set to $0.1s$ and the prediction time (T) is set to $3.0s$. In I-APF, the attraction force weight and the repulsive force weight are set to 35.0 and 30.0 respectively. In I-FGM, the sight range of the robot is set to $180°$.

Figs. 5-7 visualize paths in three scenarios: T-shape scenario, U-shape scenario and cluttered scenario. Obstacles are colored in black in the occupancy grid map. The cross dot stands for the robot. The circle point stands for the start position and the star point stands for the goal position. The dashed circle is the active region around the robot. Both the robot heading and the goal direction are represented as arrows.

Figs. 5 and 6 show the simulation in T-shape and U-shape scenarios respectively, which are typical scenarios of local minima. In both scenarios, the robot equipped with DWA is trapped at the start position and cannot move at all, thus, a modified DWA is used for the comparison. In the modified version, the heading cost is replaced with the distance cost defined in (18), except for that the local goal (x_g, y_g) is replaced with the final goal (x_t, y_t). This configuration is denoted as "global goal". As for the EA-DWA, we eliminate adaptive control in the two scenarios in order to evaluate the effect of the local goal, which is denoted as "local goal". The other two are paths generated from I-APF and I-FGM. In T-shape scenario (Fig. 5), although the modified DWA (global goal) is able to reach the final goal, the path is quite close to obstacles. On the contrast, the local goal tries to guide the robot through the center of openings, therefore the path is safer. In U-shape scenario (Fig. 6), with global goal, the robot is trapped. With the local goal, the robot firstly moves toward the opening in the front left and then moves toward the final goal. That is exactly what the robot should do to escape from the local minima. The I-APF method and the I-FGM method can escape from local minima, however, some drawbacks are obvious. Both of them tend to be too much close to obstacles. The reason is that none of them consider collision checking. In order to prevent collision, only manual adjustment of attractive force weight and repulsive force weight is available for I-APF. I-FGM tries to manually adjust gap direction weight and goal direction weight instead. Apart from that, the path generated from I-FGM is unnecessarily long. The reason is that to make the path safer, it focuses more on passing large gap rather than going toward the goal.

The experiment carried in Fig. 7 aims to evaluate the effect of the adaptive control of EA-DWA. Fig. 7 (b) eliminates the adaptive control from EA-DWA, which is denoted as "non-adaptive control". Fig. 7 (a) reserves the adaptive control, which is denoted as "adaptive control". Figs. 7 (c) and (d) are paths generated from I-APF and I-FGM respectively. Compared with others, the path with adaptive control is smoother and safer. Furthermore, unlike I-APF and I-FGM, the path is feasible for the controller to execute due to the consideration of velocity constraints of the robot in EA-DWA.

Figs. 8 and 9 share the same scenario with that in Fig. 7. The translational velocity comparison between adaptive control and non-adaptive control is shown in Fig. 8. Without the adaptive control, the robot blindly accelerates until reaching the speed limit or collision happens, as shown in Fig. 8 (b). The three steep falls in the graph indicate three collisions. However, with

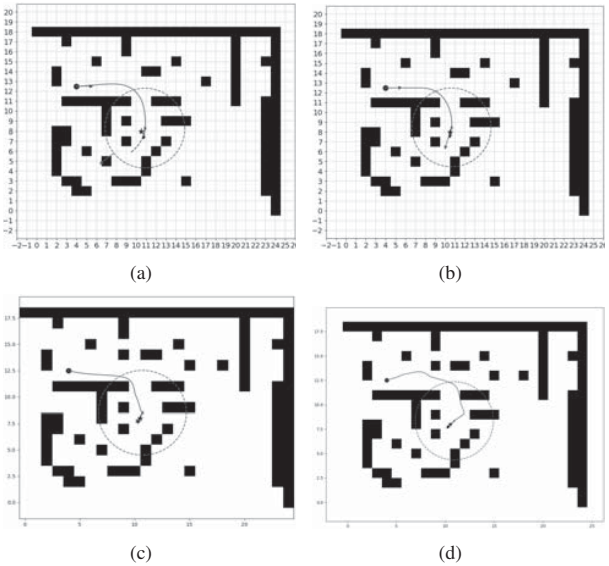


Fig. 5 Path comparison in T-shape scenario: (a) local goal, (b) global goal, (c) I-APF, (d) I-FGM

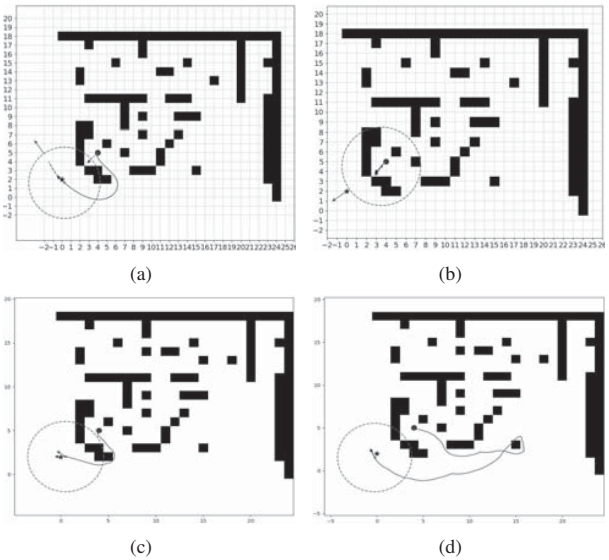
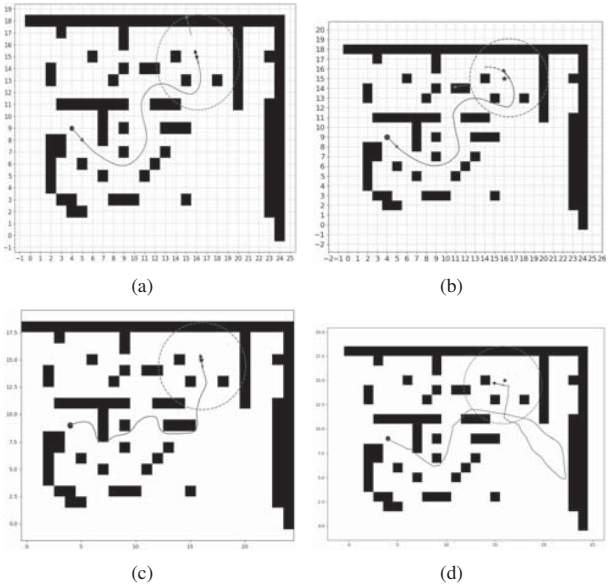


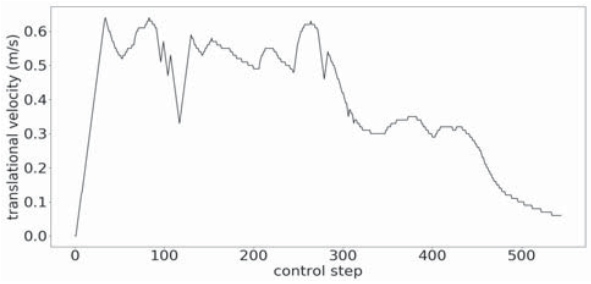
Fig. 6 Path comparison in U-shape scenario: (a) local goal, (b) global goal, (c) I-APF, (d) I-FGM

the assistance of adaptive control, the robot is aware of the goal and the environment property. When the robot approaches the goal, the overall trend of the speed is decreasing. The robot may also accelerate if the environment spaciousness increases. As a result, the adaptive speed reduces the risk of collision and guarantees high efficiency of speed utilization. Besides, the translational velocity command is smoother, which is vital for energy efficiency. Fig. 9 shows the obstacle cost weight change along the path in Fig. 7 (a). There are two valleys (one is near the 300^{th} step and the other is near the 400^{th} step) corresponding to the two hills in Fig. 8 (a). It indicates that more aggressive speed selection and less attention on

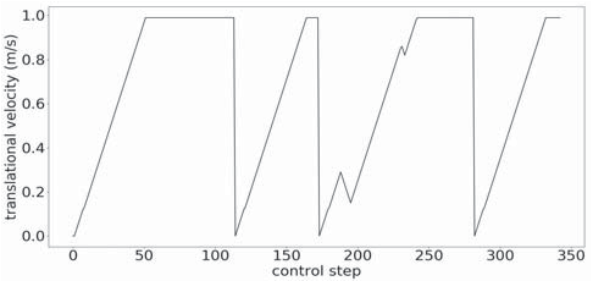


(a) (b) (c) (d)

Fig. 7 Path comparison in cluttered scenario: (a) adaptive control, (b) non-adaptive control, (c) I-APF, (d) I-FGM



(a)



(b)

Fig. 8 Translational velocity comparison between adaptive control and non-adaptive control in cluttered scenario. (a) adaptive control, (b) non-adaptive control

obstacle avoidance occur with the increase of environment spaciousness. Compared with I-APF and I-FGM, the proposed EA-DWA can automatically change the attention on obstacle avoidance based on the surrounding environment. Therefore, the path is more optimal.

In order to evaluate the performance quantitatively, four metrics are defined, that is, path safety, path length, path smoothness and control steps needed to reach the goal. The

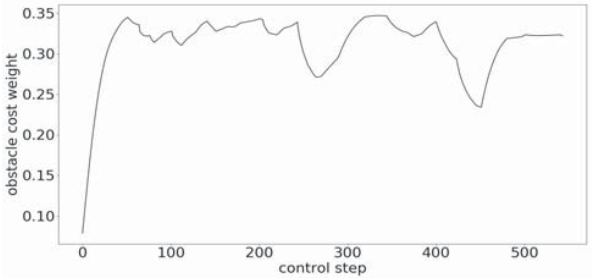


Fig. 9 Obstacle cost weight change with adaptive control

path safety is the minimum distance to obstacle border. The control step is the execution times of motion controller before reaching the goal. The Bending Energy [15] is utilized to characterize the path smoothness ($smooth$), which is defined as:

$$\begin{aligned} smooth &= \frac{1}{n}\sum_{i=1}^{n} k^2(x_i, y_i) \\ k(x_i, y_i) &= \frac{f''(x_i)}{(1+(f'(x_i))^2)^{\frac{3}{2}}} \\ y_i &= f(x_i), \end{aligned} \tag{24}$$

where (x_i, y_i) is the coordinate of the i^{th} point in the path. Smaller $smooth$ indicates smoother path.

Table I shows the quantitative comparison of EA-DWA, I-APF and I-FGM. It can be seen that in all three scenarios, the path generated from EA-DWA is safer and smoother. Compared with I-APF, the path is longer and more control steps are needed, which is the cost of more precise control. Due to the ability of adjusting attention on obstacles adaptively, EA-DWA can find optimal path with shorter length compared with I-FGM.

TABLE I
PERFORMANCE COMPARISON OF EA-DWA, I-APF AND I-FGM

	Metrics	EA-DWA	I-APF	I-FGM
T-shape	Safety	0.72	0.30	0.07
	Path length	10.26	9.40	12.00
	Path smoothness	0.18	2.89	7.64
	Control steps	348	94	120
U-shape	Safety	0.32	0.13	-0.01
	Path length	10.51	9.50	33.90
	Path smoothness	0.51	1.26	2.14
	Control steps	274	95	339
Cluttered	Safety	0.54	0.01	-0.10
	Path length	21.33	20.40	40.30
	Path smoothness	0.19	0.67	3.62
	Control steps	543	204	403

V. CONCLUSION

The proposed algorithm integrates environment awareness into the Model Predictive Control framework. The quantitative descriptor of the environment is the source of environment awareness while the two key techniques, i.e., the local goal and the adaptive control, are tools for environment awareness. It makes great improvement on local minima escaping and path safety compared with DWA. Besides, experiments show that the algorithm is competitive in path safety and

path smoothness compared with state-of-the-art methods. We believe that such strategy has dramatic application prospect in cluttered environments for mobile robots.

REFERENCES

[1] Hoy, Michael, Alexey S. Matveev, and Andrey V. Savkin, "Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey," *Robotica*, vol. 33, pp. 463-497, 2015.
[2] Borenstein, Johann, and Yoram Koren, "Real-time obstacle avoidance for fast mobile robots," *IEEE Transactions on systems, Man, and Cybernetics*, vol. 19, pp. 1179-1187, 1989.
[3] Borenstein, Johann, and Yoram Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," *IEEE transactions on robotics and automation*, vol. 7, pp. 278-288, 1991.
[4] Ulrich, Iwan, and Johann Borenstein, "VFH+: Reliable obstacle avoidance for fast mobile robots," in *Proc. ICRA Conf.*, 1998, pp. 1572-1577.
[5] Weerakoon, Tharindu, Kazuo Ishii, and Amir Ali Forough Nassiraei, "An artificial potential field based mobile robot navigation method to prevent from deadlock," *Journal of Artificial Intelligence and Soft Computing Research*, vol. 5, pp. 189-203, 2015.
[6] Demir, Mustafa, and Volkan Sezer, "Improved follow the gap method for obstacle avoidance," in *Advanced Intelligent Mechatronics (AIM)*, 2017, pp. 1435-1440.
[7] Simmons, Reid, "The curvature-velocity method for local obstacle avoidance," in *Proc. ICRA Conf.*, 1996, pp. 3375-3382.
[8] Fox, Dieter, Wolfram Burgard, and Sebastian Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, pp. 23-33, 1997.
[9] Ulrich, Iwan, and Johann Borenstein, "VFH*: Local obstacle avoidance with look-ahead verification," in *Proc. ICRA Conf.*, 2000, pp. 2505-2511.
[10] Chou, Chih-Chung, Feng-Li Lian, and Chieh-Chih Wang, "Characterizing indoor environment for robot navigation using velocity space approach with region analysis and look-ahead verification," *IEEE Transactions on Instrumentation and Measurement*, vol. 60, pp. 442-451, 2011.
[11] Blanco, Jose Luis, Mauro Bellone, and Antonio Gimenez-Fernandez, "TP-Space RRT: kinematic path planning of non-holonomic any-shape vehicles," *International Journal of Advanced Robotic Systems*, vol. 12, pp. 55-63, 2015.
[12] Devaurs D, Simon T and Corts J, "Optimal path planning in complex cost spaces with sampling-based algorithms," *IEEE Transactions on Automation Science and Engineering*, vol. 13, pp. 415-424, 2016.
[13] Samaniego, Ricardo, Joaquin Lopez, and Fernando Vazquez, "Path planning for non-circular, non-holonomic robots in highly cluttered environments," *Sensors*, vol. 17, pp. 1876-1894, 2017.
[14] Napoli, Michael E., Harel Biggie, and Thomas M. Howard, "Learning models for predictive adaptation in state lattices," *Field and Service Robotics*, vol. 5, pp. 285-300, 2018.
[15] Van Vliet, Lucas J., and Piet W. Verbeek, "Curvature and bending energy in digitized 2D and 3D images," in *Proceedings of the Scandinavian Conference on Image Analysis*, 1993, pp. 1403-1410.