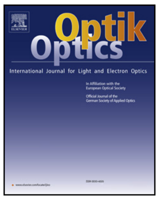
Optik - International Journal for Light and Electron Optics 219 (2020) 165096

Contents lists available at ScienceDirect

Optik

journal homepage: www.elsevier.com/locate/ijleo



Path planning for indoor Mobile robot based on deep learning

Lin Zhang, Yingjie Zhang*, Yangfan Li

School of Mechanical Engineering, Xi'an Jiaotong University, Xi'an, People's Republic of China

ARTICLE INFO

Keywords:
Path planning
Deep learning
Rapidly-exploring random tree
Ray tracing algorithm
Waiting rule
Obstacle avoidance

ABSTRACT

This paper aims to give an optimal path planning of a mobile robot in a known indoor environment. An algorithm based on deep learning, ray tracing algorithm, waiting rule, and Rapidly-exploring Random Tree is proposed to solve the problem of obstacle avoidance and path planning. Firstly, GoogLeNet is used to classify obstacles. Here, it helps to distinguish between static and dynamic obstacles. Secondly, for the static obstacle avoidance, the ray tracing algorithm is proposed to avoid the obstacles which are identified by GoogLeNet. And for the dynamic obstacle avoidance, this paper proposed a waiting rule for dynamic obstacle avoidance. Thirdly, the RRT method plans a path from the start point to the goal point. The novelty of this paper is that the type of obstacles is distinguished by deep learning, and the two different kinds of obstacles used ray tracing algorithm and waiting rule to avoid obstacles collision, respectively. In experimental results, rapidly-exploring random tree is compared with genetic algorithm, and particle swarm optimization method in static environment, and it is compared with artificial potential field approach in dynamic environment. Experiments are carried out in the two-dimensional environment and successfully applied to the path planning of mobile robots in multi-obstacles environment, and they prove the feasibility of the algorithm.

1. Introduction

A lot of research is going on around the globe to find a suitable intelligent control to be used for navigation of a mobile robot without human interaction. Mobile robots can perform tasks in places where humans are unreachable or dangerously unknown, and have been successfully used in many fields. Path planning is one of the most important key technologies in the field of mobile robot research [1].

Path planning means that a robot automatically searches a collision free and optimal path from the initial point to the target point in the environment. At the same time, optimization is reflected in distance, time and energy consumption, and the most commonly used criterion is the optimization of distance. The path planning environment is either static or dynamic. In static environment, the whole solution must be found before starting execution. However, for the dynamic or partially observable environments replanning are required frequently and more planning update time is needed.

The path planning problem of mobile robot can be described as following. Given a start point, a goal point, and a set of obstacles distributed in a workspace, and find a safe and efficient path for the mobile robot. Thus the robot can go from the start point to the goal point without colliding with any obstacles along the path, the obstacles include static obstacles and dynamic obstacles. Finally, the mobile robot can complete the task of navigation and obstacle avoidance.

In the past few decades, many researchers have invented a lot of traditional methods for path planning, such as ant colony

* Corresponding author.
E-mail address: yjzhang@mail.xjtu.edu.cn (Y. Zhang).

https://doi.org/10.1016/j.ijleo.2020.165096
Received 2 April 2020; Accepted 7 June 2020
Available online 11 June 2020

algorithm [2], Artificial Potential Field (APF) method [3], genetic algorithm (GA) [4], particle swarm optimization (PSO) [5], and so on. The genetic algorithm is a global optimization algorithm, which does not fall into the fast falling trap of the local optimal solution, but the speed of genetic algorithm is slow and it has a dependency on the selection of the initial population. The ant colony algorithm has a large amount of computation, a slow convergence rate, and is easy to fall into local optimum. Particle swarm optimization has a fairly fast approximation speed, which can effectively optimize the parameters of the system. However, it is prone to premature convergence and poor local optimization.

There are always some weaknesses in the path planning of robots only by traditional methods. Therefore, in order to solve these problems, some scholars present several ways to promote the strengths and avoid the weaknesses of the traditional methods. Especially, a lot of research work has been done on neural network-based robot path planning at home and abroad, and many neural network models for path planning have been proposed. Neural network algorithm refers to the human brain's image thinking, and it is a nonlinear fitting process, which can process and store information in parallel [6]. Chen [7] combines neural networks and genetic algorithms for mobile robot path planning, which is difficult to guarantee real-time requirements. In a different sight of view, a collision-free path based on two neural networks: principal component analysis (PCA) and a multilayer perceptron (MLP) were constructed in [8]. Furthermore, a neural network algorithm that enables robot to move safely in an unknown environment with static or dynamic obstacles was designed in [9]. The relevant parameters of the model have a great influence on the performance of the planner, and the setting of these parameters has no effective method, and can only be tested repeatedly. Since the goal of mobile robot is to move towards an unstructured, unknown natural environment, robot must have good environmental adaptability and autonomy, and be able to react in real time in a dynamic environment.

A real-time collision-free path planning solution for 3D navigation in complex dynamic environments is proposed by Sanchez-lopez [10], and high-level geometric primitives are employed to compactly represent the environment. In [11], seven performance indicators for evaluating the results of path planning are defined.

The drawback of neural network is that as the number of neural network layers deepens, there are three major problems: non-convex optimization problems, gradient disappearance problems, and over-fitting problems. Deep learning adopts a similar hierarchical structure of neural networks, and contains multiple hidden layers. The features do not need to be manually selected, but are selected by the network itself.

Deep learning network structure is the best simulation of the human brain cortex. Convolutional neural network (CNN) is a kind of feed forward neural network with convolutional computation and deep structure. It is one of the representative algorithms of deep learning [12]. CNN has been successfully used in image classification [13]. The focus of this paper is on applying deep neural network in robot path planning.

The Rapidly-exploring Random Tree (RRT) is a randomized algorithm to explore large state space in a relatively short time, and it can be developed for robot motion planning in high dimensional configuration spaces.

The difficulty of robot path planning lies in how to intelligently plan the path planning process and complete obstacle avoidance in this process. The method of this paper identifies the types of obstacles on the basis of deep learning, and then uses the idea of ray tracing to avoid obstacles. Finally, the path planning is completed by RRT method to automate the whole process.

The contributions of this paper are 1) the obstacle identification is realized by deep learning, and the obstacles in room is divided into two kinds (static obstacle and dynamic obstacle); 2) based on the obstacle identification step, the avoidance of static obstacle is using the thought of ray tracing, and the avoidance of dynamic obstacle is using the proposed waiting rule; 3) after obstacles avoidance, the path planning of mobile robot is using the RRT algorithm. In a word, deep learning tools help robot to classify indoor obstacles, ray tracing and waiting rules to avoid static obstacles and dynamic obstacles, respectively. The first two steps make the accuracy of the final path planning algorithm improved. In the experimental results, RRT is compared with GA and PSO method in static environment, and it is compared with APF in dynamic environment.

The remainder of this paper is organized as follows. Section 2 provides a model description of this paper. The parts of obstacles are described in Section 3, including obstacle identification and obstacle avoidance. Section 4 describes the algorithm for path planning in this paper. Simulation results and discussions will be included in Section 5, and this section contains the experiments in static environment and dynamic environment. Section 6 will summarize our conclusions and gives the directions for our future research in this area.

2. Model description

The mobile robot path planning problem is typically formulated as follows: given a mobile robot and a description of a known environment, we need to plan an optimal path between two specified locations, a start and goal point.

As shown in Fig. 1, the proposed algorithm is composed of three parts: obstacle identification, obstacle avoidance and path planning. The combination of path planning and deep learning makes the intelligence of the planning process, and the accuracy of path planning results is improved.

The priority of obstacle identification is higher than obstacle avoidance, and the priority of obstacle avoidance is higher than path planning. Then, refine the entire method model. The specific procedure can be described as follows.

Step1: Initializing the start point and the goal point, and the approximate path between them in a known environment is given.

Step2: When the robot starts moving, and the obstacles are identified by deep learning during the movement, including the shape and size of the obstacles, and the obstacles are finally divided into static obstacles or dynamic obstacles.

Step3: Based on the step 2, the obstacle avoidance is also divided into static obstacle avoidance and dynamic obstacle avoidance. Here, the thought of ray tracing algorithm is used to static obstacle avoidance, and the waiting rule is proposed to perform dynamic

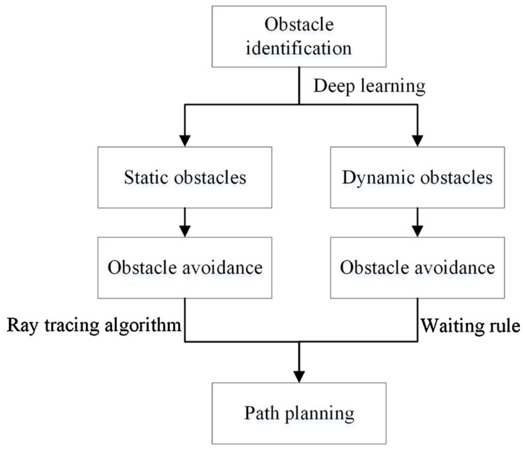


Fig. 1. The flowchart of the proposed method.

obstacle avoidance.

Step 4: Based on the above steps, then, RRT method is applied in the process of path planning of mobile robot.

Step 5: Repeat the second step until the robot reaches the goal point.

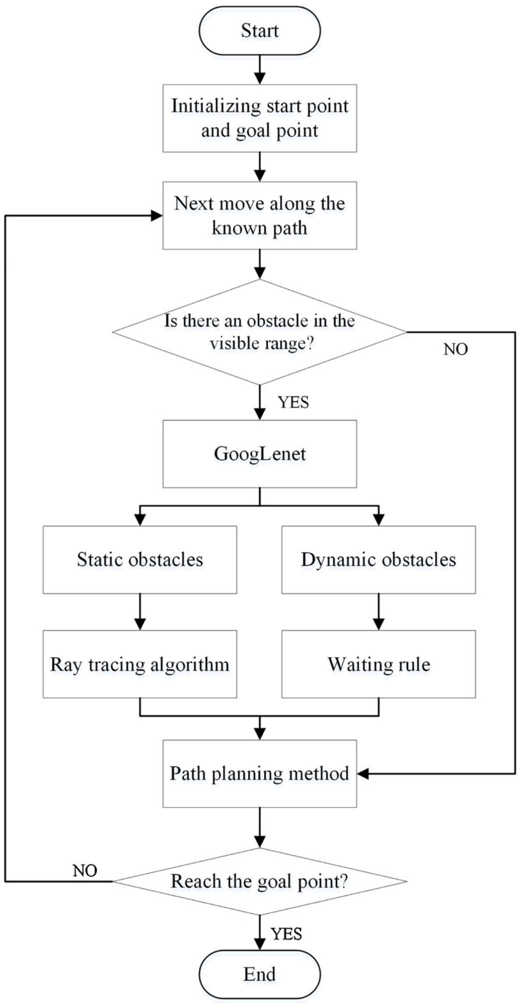


Fig. 2. The flow diagram of the proposed model.

To visualize these steps, the flow diagram of the model is given in Fig. 2.

As shown in Fig. 1, the flowchart of the proposed method is simple. Fig. 2 and the steps above are explanations of Fig. 1, and the corresponding pseudo code is described as following:

```
Input: I-captured image.
Initialize: Start point S, goal point G
Output: trajectory from start point to goal point.
Begin
Move
If Obstacle
Then
Obstacle identification by GoogLeNet
If static obstacle
Ray tracing algorithm// obstacle avoidance
Else dynamic obstacle
Waiting rule// obstacle avoidance
Then
Path planning// RRT algorithm
Else
If reach the goal point
End
Else
Return move
```

3. Obstacle identification and obstacle avoidance

3.1. Obstacle identification based on GoogLeNet

In this paper, with the aid of the auxiliary classifiers employed by GoogLeNet, the obstacles are classified as static obstacles and dynamic obstacles. GoogLeNet is a new deep learning structure proposed by [14]. Before this, AlexNet, VGG and other structures all achieved better training effects by increasing the depth (layer number) of the network, but the increase in the number of layers will bring a lot of negative effects, such as overfit, gradient disappearance, gradient explosion and so on. The idea of inception raises the training result from another angle: it can use computing resources more efficiently, and can extract more features under the same amount of calculation, thus improving the training results.

GoogLeNet and VGG are the top two in the 2014 ImageNet Challenge, and their common feature is deeper levels. VGG inherited some framework structures of LeNet and AlexNet, and GoogLeNet made a bolder network structure attempt. Although it was only 22 layers deep, it was much smaller than AlexNet and VGG. The number of parameters of the three of them is shown in Table 1. Therefore, GoogLeNet is a better choice when memory or computing resources are limited. From the model results, the performance of GoogLeNet is more superior.

The depth of GoogLeNet reaches to 22 and the number of convolutional layers reaches to 60, so that avoiding gradient disappearance. Specifically, GoogLeNet ensures that the errors always vanish with back propagation by adding two losses at different depths. In addition, a variety of cores 1×1, 3×3, 5×5, and direct max pooling are added in the width. However, if you simply apply these to the feature map, the feature map thickness of the concat will be very large, so the inception in GoogLeNet to avoid this phenomenon has the following structure, before 3×3 and before 5×5, the convolution sum added after max pooling, and it has the effect of reducing the thickness of the feature map.

The dynamic obstacles at home are animals and humans, and the pet types are mainly cats and dogs. The data set used here is CIFAR10, and each type contains 1000 different pictures. The static obstacles are furniture, such as sofa, chair, dining desks, and so on. The input images are shown in Fig. 3, and the corresponding identification results after dimensionality reduction are shown in Fig. 4. Here, the images in Fig. 3 is just some examples in CIFAR10 dataset, and it does not stand for there is only these kinds of obstacles indoor. Images in real-world is diverse, and the dataset can contain all of it. The dataset in CIFAR10 can be classified into static obstacles and dynamic obstacles based on GoogLeNet.

The results of obstacle identification based on GoogLeNet are summarized in Table 2. And the recognition accuracy of different obstacles are over 75 % and the execution time is within 3 s. Through this method, the obstacles are quickly and accurately identified, and it is easy to divide obstacles into static obstacles and dynamic obstacles.

Firstly, GoogLeNet has the best performance compared to VGG and AlexNet. Secondly, GoogLeNet's image recognition runs at a high speed and accuracy. Therefore, GoogLeNet is the most reasonable choice here.

The number of parameters of GoogLeNet, AlexNet, and VGG.

	GoogLeNet	AlexNet	VGG
number	500millions	12×500millions	$3 \times 12 \times 500$millions

Fig. 3. Input images. (a) cat. (b) sofa. (c) dog. (d) dining table.

high speed and accuracy. Therefore, GoogLeNet is the most reasonable choice here.

3.2. Obstacle avoidance

Obstacle avoidance is easily achieved with simple IF-THEN rules [15]. At the same time, in order for the mobile robot to perform the path tracking task well in the dynamic environment, the obstacle avoidance behavior of the mobile robot must be prioritized higher than the priority of the path tracking. Obstacle avoidance involves two situations, one is the obstacle avoidance of static obstacles, and the other is the obstacle avoidance of dynamic avoidance.

3.2.1. Obstacle avoidance of static obstacles

The ray tracing algorithm has the advantage of large computation, time consuming, and interference in the resulting image. The core operation of the ray tracing algorithm is to determine which object the tracked light encounters, that is, the intersection operation. Efficient intersection algorithm will greatly improve the speed of ray tracing. Set the position of the robot to the viewpoint, and assuming that the equation for the light emitted by the viewpoint is

$$P(t) = Dt + R_0 \tag{1}$$

$$R_0 = (r_1, r_2, r_3) \tag{2}$$

$$D = (d_1, d_2, d_3) \tag{3}$$

$$\|D\| = 1 \tag{4}$$

$$R = (x, y, z) \tag{5}$$

where R_0 is the location of robot, and D represents the direction of rays, and P is any point on the ray. Here, (x, y, z) are the three-dimensional coordinates.

As shown in Fig. 5, taking the camera of the robot as the viewpoint, and the ray is emitted by the viewpoint as a fix angle and intersects with the cylindrical obstacle. Similarly, the cylinder can also be replaced by a polygon and other irregularities. The above mentioned is a three-dimensional situation, then discuss the two-dimensional case. And the two-dimensional obstacles discussed in this paper are round face, polygonal and Bezier surface.

Taking the robot as the viewpoint, the emitted rays intersect with the circular surface, the polygonal surface, and the Bezier surface. If there is an intersection point within the ray range of a certain angle, it can only move in other areas except the red area in

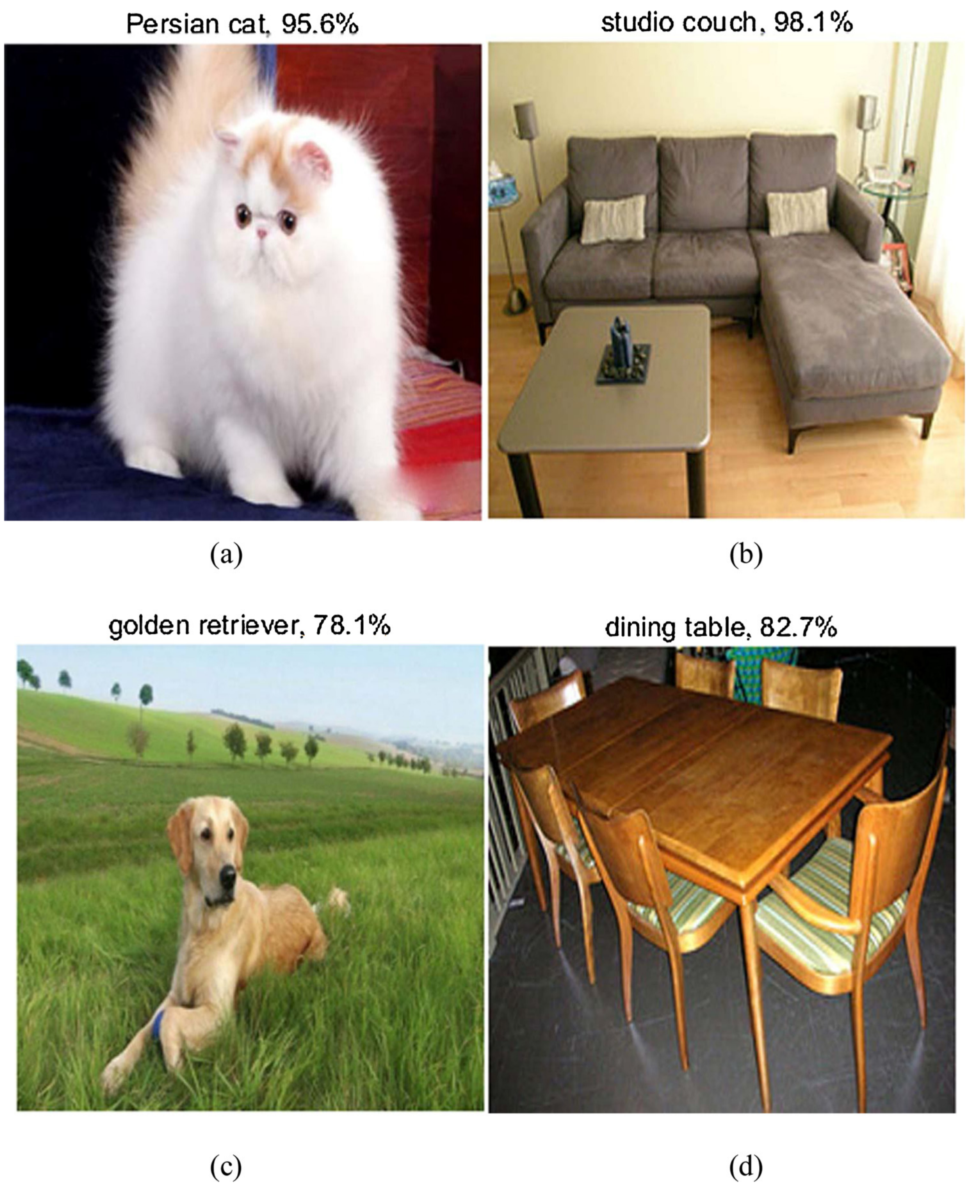


Fig. 4. The dimensionality reduction identification results of Fig. 3.

Table 2
The identification results based on GoogLeNet.

Images	Input size	Dimensionality reduction size	Recognition accuracy	Time
a	500 × 350	224 × 224	95.6%	2.16s
b	240 × 180	224 × 224	98.1%	2.38s
c	975 × 551	224 × 224	78.1%	2.32s
d	240 × 166	224 × 224	82.7%	1.82s

the beam range. If there is no intersection point, it can move freely within the beam range. And take the round surface as an example here, as shown in Fig. 6.

Projecting at a certain angle from the viewpoint, in the projection range, the part that intersects the target obstacle (yellow part of Fig. 6) is the range of motion of the robot, and the projection is continued during the motion, and the part of the projection motion intersecting the target obstacle is removed. The obstacle avoidance process is completed until the projection from the viewpoint does not intersect the obstacle. Finally, the motion of the robot is from point *A* to point *B*, and then from point *B* to point *C*, and so on, until the ray has no intersection with obstacle.

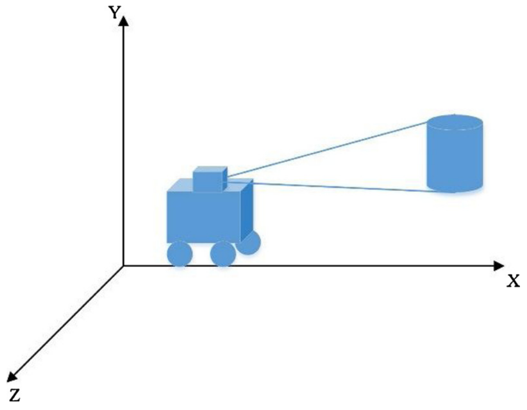


Fig. 5. The case of cylindrical obstacle.

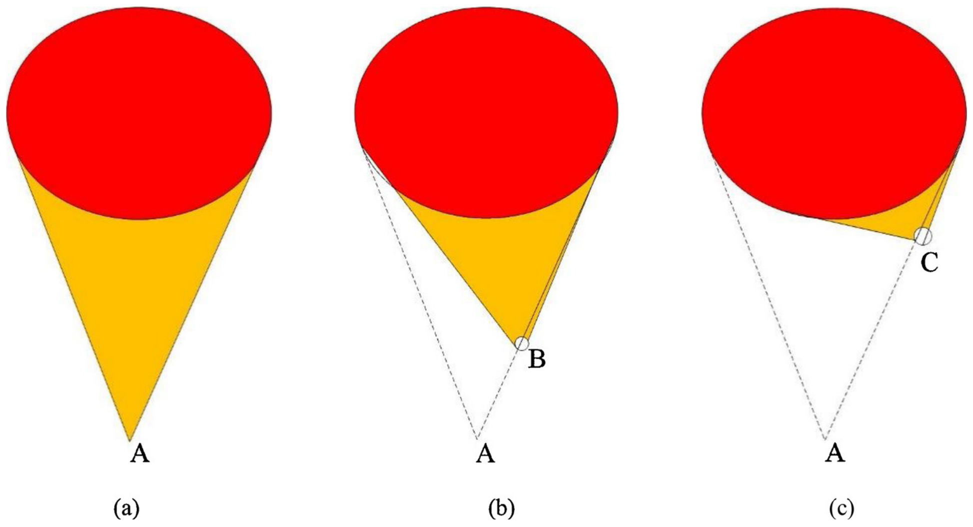


Fig. 6. The example of round surface.

In a word, this article uses the idea of ray tracing and improves it. When the ray intersects with the obstacle, the robot can only move within the ray range outside the obstacle area. When the ray does not intersect with the obstacle, the obstacle avoidance is completed and the next path planning can be performed.

(1) The intersection of light and round face

Assume that the center of the circle is O(x_0, y_0, z_0), and the unit normal vector of the plane is $N(x_1, y_1, z_1)$, and r is the radius of the circle. Then the round face equation can be expressed as

$$(P - O)\bullet N = 0 \tag{6}$$

The point where the ray intersects with the circle and the point in the circle should satisfy Eq. (7). Because the point P is in the circle, so the distance between P and O is less than r.

$$(P - O)(P - O) \le r^2 \tag{7}$$

Combining the Eq. (1) and Eq. (6), the intersection of light and the circle can be found, then

$$(Dt + R_0 - O)\bullet N = 0 \tag{8}$$

That is,

$$(D\bullet N)t + (R_0 - O)\bullet N = 0 \tag{9}$$

And the intersection is $t_0 = \frac{-(R_0 - O)N}{D \bullet N}$, when D• N$\neq$ 0.

If D• N= 0, then the ray is parallel to the circular surface without intersection. And if D• N$\neq$ 0, the intersection point is $P = Dt_0 + R_0$. Then combining with Eq. (7), if the equation is satisfied, the intersection point is the intersection of the ray and the circular surface. If it is not satisfied, it means that the ray has no intersection with the circular surface.

(2) The intersection of light and polygon

If the plane equation of the polygon is given, the intersection process is similar to the intersection method of the ray and the circular surface. If the plane equation is not given, the plane equation of the polygon can be obtained by any vertices, and assuming the polygon equation is shown as Eq. (10).

$$ax + by + cz + d = 0 \tag{10}$$

The unit normal vector of the polygon is N= (a, b, c) or N= $(-a, -b, -c)$, and the sign is determined by the right hand rule. Then the polygon equation is rewritten as follows.

$$N\bullet P + d = 0 \tag{11}$$

Combining the Eq. (11) and Eq. (1), then

$$N\bullet(Dt + R_0) + d = 0 \tag{12}$$

$$t_0 = \frac{-(N\bullet R_0 + d)}{N\bullet D},\ N\bullet D \neq 0 \tag{13}$$

Similarly, when $N\bullet D = 0$, there is no intersections, and when $N\bullet D \neq 0$, the ray has intersections with the plane of polygon.
(3) The intersection of light and Bezier surface
The equation of the Bezier surface is shown as Eq. (14), and the expression of Bezier surface is also described as Eq. (15).

$$S(t) = \sum_{i=0}^{n} P_i B_{i,n}(t) \tag{14}$$

$$S = Q(u, w) \tag{15}$$

where $B_{i,n}$ is a Bernstein polynomial and t∈ [0,1], P_0, P_1, $\cdots$,P_n are a set of $n + 1$ given control points. And $Q(u, w)$ is an expression of parametric equations.
Combining the Eq. (14) or Eq. (15) with Eq. (1), then the intersections of them is described as Eq. (16) and Eq. (17).

$$\sum_{i=0}^{n} P_i B_{i,n}(t) - (Dt + R_0) = 0 \tag{16}$$

$$Q(u, w) - (Dt + R_0) = 0 \tag{17}$$

Here, $\sum_{i=0}^{n} P_i B_{i,n}(t)$ is the expression of Bezier surface, and $(Dt + R_0)$ represents the expression of light, And the solution of Eq. (17) is the intersection point. While, because of it is a ternary higher order equation, the calculation can be simplified by the following. The ray and the parametric surface are all transformed by coordinate transformation method so that the ray coincides with the axis of the coordinate system Z-axis, and the parametric equation is written into the component quantity.

$$x = f(u_1, w_1) \tag{18}$$

$$y = g(u_1, w_1) \tag{19}$$

$$z = h(u_1, w_1) \tag{20}$$

At the intersection point, x− y= 0. Here, x means the expression of Eq. (18), and y means the expression of Eq. (19). So

$$f(u_1, w_1) = 0 \tag{21}$$

$$g(u_1, w_1) = 0 \tag{22}$$

According to the Eq.s (21) and (22), the parametric value u and w can be solved. Then, with Eq.s (20),(21), and (22), we can obtain the coordinate of intersection point. If the simultaneous equation has no real solution, it means that the ray does not intersect with the surface, and then the inverse transformation method is used to find the coordinates of the intersection point in the original coordinate system.

3.2.2. Obstacle avoidance of dynamic obstacles

When the robot encounters a dynamic obstacle, it is necessary to first determine the direction of the obstacle movement, and then intelligently adjust the next movement according to the current motion of the robot to complete the obstacle avoidance.

Similarly, let (x_r, y_r) be the position of the mobile robot, and θ_r is the orientation of the mobile robot. Assuming that both the robot and the dynamic obstacles are moving at a constant speed, v_r and v_o are the motion speed of the robot and obstacles, respectively. The position relationship between the robot and the obstacle is shown in Fig. 7. If the robot intersects the obstacle after the same time, a collision will occur. Then, according to the conditions that make up the triangle, the three sides of the triangle will satisfy Eq. (23). The three inequalities of Eq. (23) are based on the principle that the sum of two sides in a triangle is greater than the third side.

$$\begin{cases} v_r \bullet t + v_o \bullet t > d \\ v_r \bullet t + d > v_o \bullet t \\ v_o \bullet t + d > v_r \bullet t \end{cases} \tag{23}$$

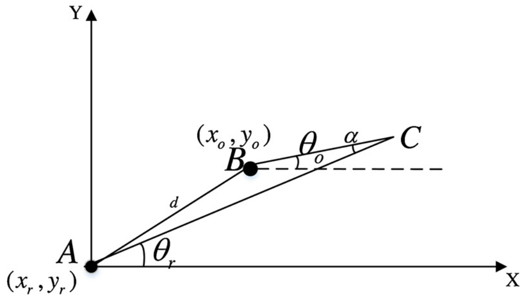


Fig. 7. The position relationship between the robot and obstacle.

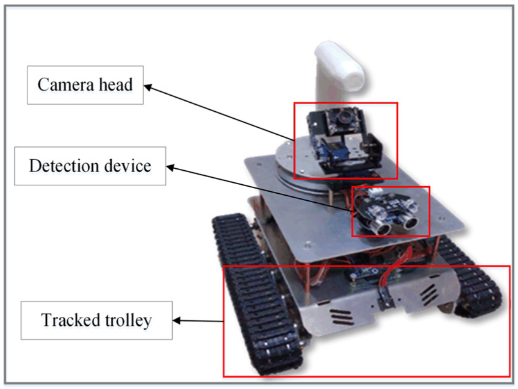


Fig. 8. The physical model of tracked trolley mobile robot.

This paper proposed a waiting rule, that is, when the mobile robot and dynamic obstacles are moving at different directions, the mobile robot can stop moving before the collision point. As shown in Fig. 7, the point C represent the intersection point of the moving direction of robot and dynamic obstacle. According to this distance calculation, the movement speed of the robot is re-adjusted to ensure that the mobile robot has decelerated to a standstill before the intersection point, and continues to move at the original speed after time t.

Assuming that the motion speed of robot is variable, if an obstacle is detected within the detection range of the robot, the speed of the robot will decrease, and at the same time, it is predicted whether the rapid movement at the current speed can collide with a dynamic obstacle. If it is predicted that the robot can collide, the robot will continue to decelerate to make the above judgment. If it cannot collide, the robot proceeds at the current speed.

For the proposed waiting rule, it is necessary to know where the robot should wait and calculate the position where the robot starts to decelerate. The angle between the direction of the robot's movement and the direction of the dynamic obstacle's movement includes three cases: zero, acute and obtuse. The case of acute angle is described in Fig. 7. Zero degree means that the two parallels will never collide, and the obtuse angle indicates there is no intersection or collision between the two movement directions.

$$\alpha = \theta_r - \theta_0 \tag{24}$$

According to cosine theorem, we have

$$\cos\alpha = \frac{(v_0 \bullet t)^2 + (v_r \bullet t)^2 - d^2}{2v_0 v_r t^2} \tag{25}$$

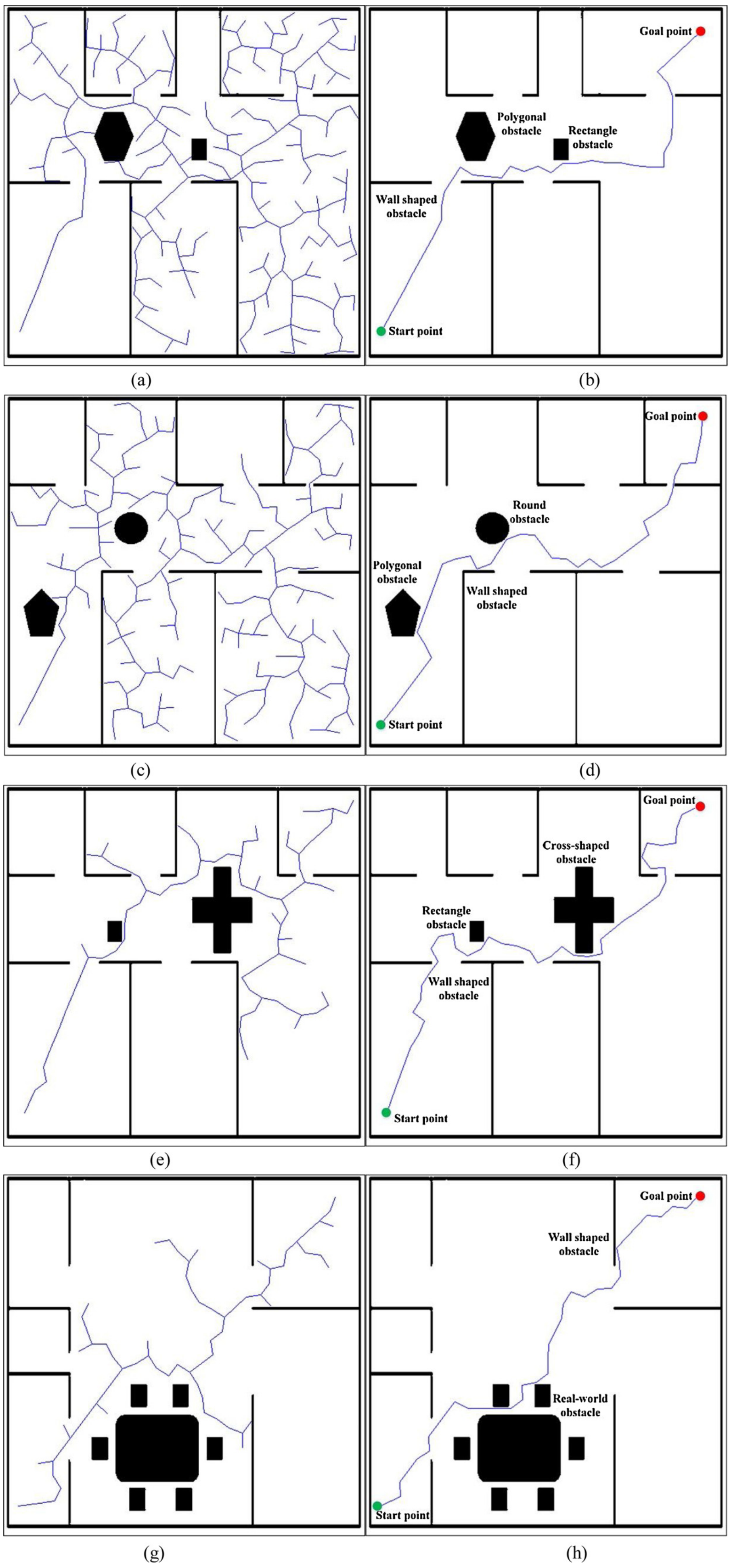
Therefore, the time t_0 required for the robot to reach the collision point C satisfies the Eq. (25). In the actual case, in order to avoid collision, the distance of the robot actually moves must be subtract from the length L_r of the robot itself. And the distance of robot can be expressed as follows.

$$S_r = v_r \bullet t_0 - L_r \tag{26}$$

And the acceleration of the robot decelerating to 0 can be solved by Eq. (27).

$$v_r - a \bullet \frac{S_r}{v_r} = 0 \tag{27}$$

Finally, for the case of dynamic obstacles, by judging the relationship between the direction of motion of the robot and the direction of motion of the dynamic obstacle, then, the mobile robot can perform the waiting rule to avoid dynamic obstacles. Subsequently, the sensor carried by the robot determines whether the dynamic obstacle has crossed the position of the robot. If it does



(caption on next page)

Fig. 9. The path planning results based on RRT.

Table 3
Experimental results based on RRT.

Number	Start point	Goal point	Processing time	Path length
b	(464,37)	(23,453)	3.74s	782pixel
d	(467,35)	(21,458)	2.55s	823pixel
f	(156,33)	(30,454)	3.74s	885pixel
h	(459,32)	(21,459)	3.40s	736pixel

Table 4
Coordinates of the start point and goal point in different environments.

Environment	Start point	Goal point	Grid size
1	(10,2)	(10,39)	20×40
2	(2,39)	(19,5)	20×40
3	(5,85)	(99,190)	100×200

not pass, the robot will continue to wait. And if it passes, the robot will continue to move at the previous speed.

4. Algorithm for path planning

The path planning algorithm based on RRT can avoid the modeling of space by collision detection of sampling points in state space, and can effectively solve the path planning problem of high-dimensional space and complex constraints. The method is characterized in that it can quickly and efficiently search for high-dimensional space, and the search results are directed to blank area through random sampling points in the state space, thereby finding a planning path from the starting point to the goal point, which is suitable for solving multi-degree-of-freedom robots in complex path planning in the environment and in a dynamic environment.

RRT is an efficient planning method in multidimensional space. It uses an initial point as the root node to generate a random expansion tree by randomly sampling the leaf nodes. When the leaf nodes in the random tree contain the goal points or enter the target area, a path from the initial point to the goal point can be found in the random tree.

The RRT algorithm grows and maintains a tree where each node of the tree is a point in the workspace. The area explored by the algorithm is the area occupied by the tree. Initially the algorithm starts with a tree which has source as the only node. At each iteration the tree is expanded by selecting a random state and expanding the tree towards that state. Expansion is done by extending the closest node in the tree towards the selected random state by a small step. The algorithm runs till some expansion takes the tree near enough to the goal. The size of the step is an algorithm parameter. Small values result in slow expansion, but finer paths or paths which can take fine turns. The tree expansion may be made biased towards the direction of the goal by selecting goal as the random state with some probability.

One weakness of RRT is that it is difficult to find a path in an environment with narrow channels. Because the narrow passage area is small, the probability of being hit is low. Sometimes RRT quickly finds a way out, sometimes it is trapped inside an obstacle.

5. Results and discussion

5.1. Experimental setup

The proposed method has been tested in a series of experiments to exhibit its effectiveness. The resolution of original maps are 500×500 in Section 5.2 and Section 5.4, and all experiments were coded and run on MATLAB R2018b (64-bit win64). All results were implemented on the environment of Intel i7, 4.2 GHz 4.2 GHz with 8GB RAM.

In order to validate the efficiency of the proposed path planning method, the methods were applied on tracked car mobile robot. The physical model of tracked trolley mobile robot is shown in Fig. 8, and it includes camera head, detection device and tracked trolley. There is an MG513 DC-coded geared motor on each side of the tracked trolley, and the wheelbase between the two drive wheels is 25 cm.

5.2. Simulation results and analysis based on RRT algorithm

As shown in Fig. 9, the path in blue is the optimal path planning results of the mobile robot from start point to goal point. It can be seen that for different types of obstacles, the obstacle avoidance function is realized. The Figs. 9(g) and 9(h) are considering the actual situation of dining table. However, the path obtained by this method is not compared with other methods to verify the performance. And the experimental results based on RRT is shown in Table 3. So, the comparison between RRT and APF is described as following.

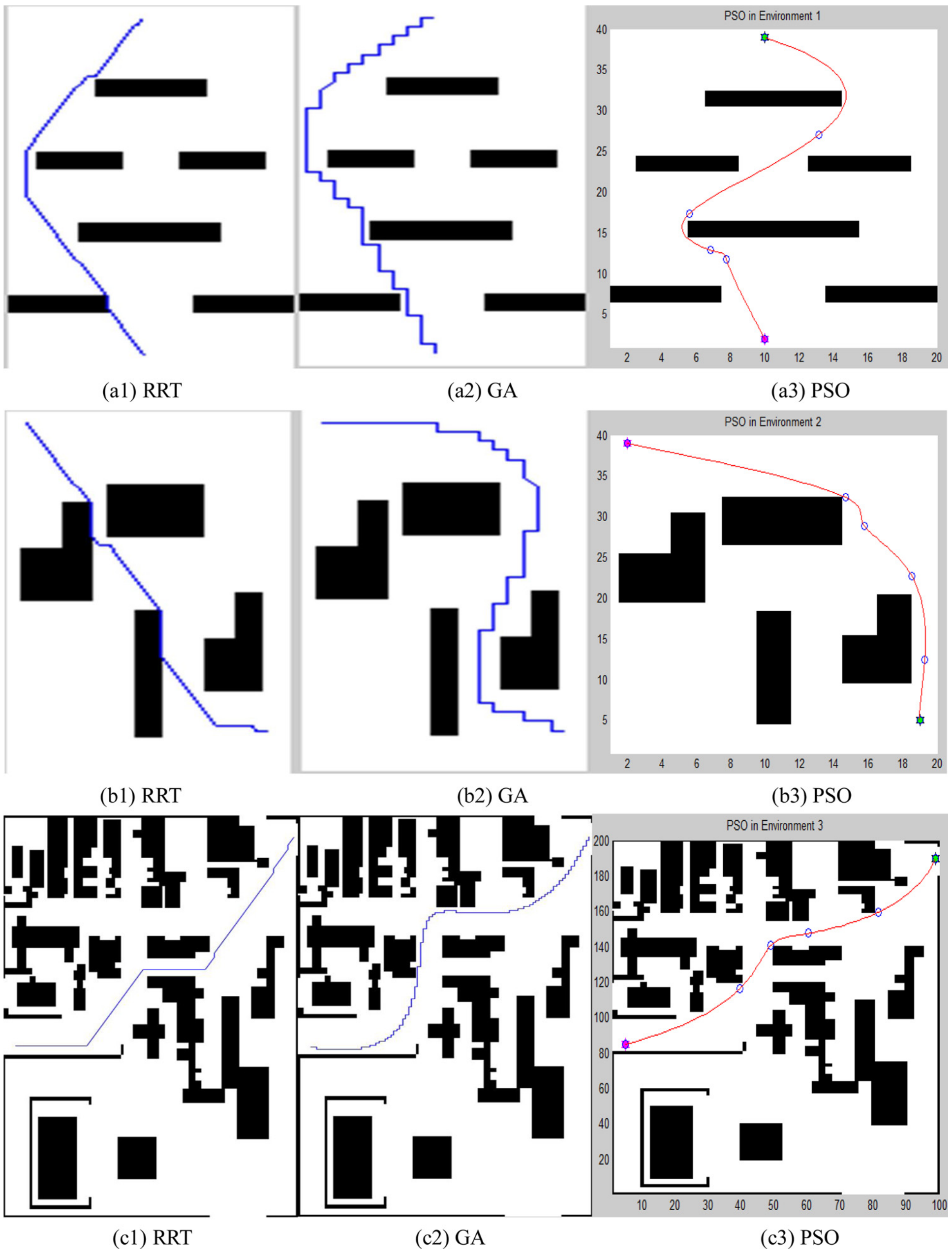


Fig. 10. path planning results based on RRT, GA, and PSO methods in different environments.

Table 5
Comparisons of performance index with three methods.

Environment	Methods	Running time	Path length
1	RRT	2.59s	41.68
	PSO	21.17s	43.08
	GA	326.30s	54.00
2	RRT	1.62s	41.22
	PSO	26.80s	42.69
	GA	338.5s	58.00
3	RRT	79.89s	142.82
	PSO	251.94s	147.54
	GA	309.11s	203.00

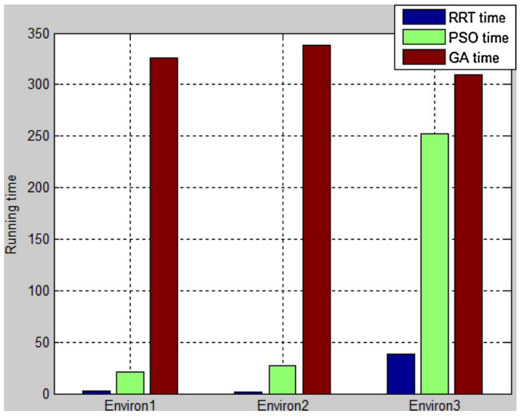


Fig. 11. Comparisons of running time based on RRT, PSO, and GA methods.

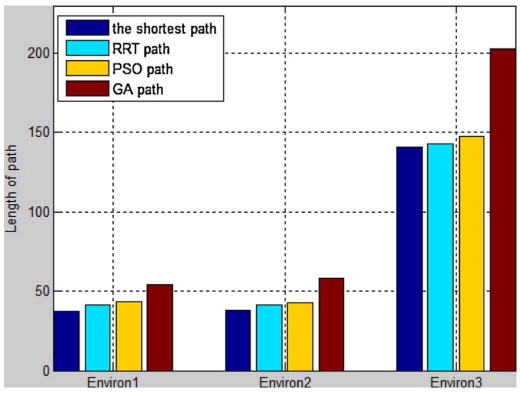


Fig. 12. Comparisons of path length based on RRT, PSO, and GA methods.

Table 6
Data of path optimal degree in different environments.

Environment	RRT	PSO	GA
1	99.87	99.84	99.54
2	99.92	99.88	99.47
3	99.99	99.95	99.56

From Fig. 9, we can see that the robot can arrive at the goal point from start point without colliding. And we can get a qualitative analysis of the path planning results based on Table 3, while we should use comparative experiments for quantitative analysis. Section 5.3 and Section 5.4 show the comparative experiments in static environments and dynamic environments, respectively.

5.3. Comparisons in static environments

In this section, we compared RRT with genetic algorithm (GA) and particle swarm optimization (PSO). The coordinated of start point and the goal point, and the Grid size of the three environments are contained in Table 4.

As shown in Fig. 10, subfigures (a1), (b1), and (c1) are the path planning results of RRT in three different environments, subfigures (a2), (b2), and (c2) are the path planning results of GA in three different environments, and the three column of subfigures (a3), (b3), and (c3) are the path planning results of RRT in three different environments.

From Fig. 10, we can only see the results of path planning intuitively, while we cannot get which method is optimal. The specific quantitative analysis is based on Table 5. From Table 5, we compared RRT with GA, and PSO methods in the performance index of running time and path length. To vividly compare these data of running time and path length, we describe them as a bar graph. Fig. 11 shows the comparisons of running time based on RRT, PSO, and GA methods, and Fig. 12 displays the comparisons of path length based on RRT, PSO, and GA methods, and the shortest path length.

From Fig. 11, we can clearly see that the running time of GA is much greater than PSO, and RRT. And the running time of RRT is the smallest. From Fig. 12, we can see that the path length of GA is longer than PSO, and the path length of PSO is larger than that of

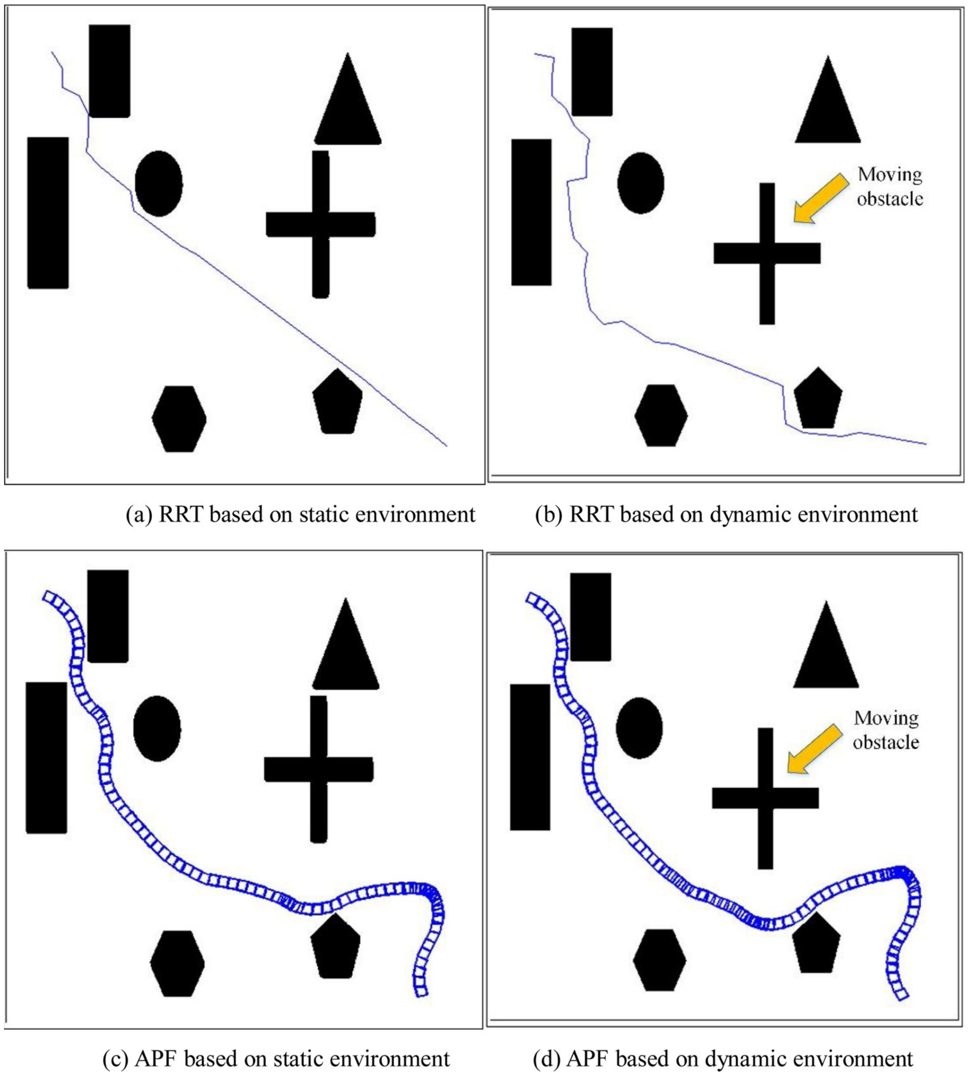

Moving
obstacle
(a) RRT based on static environment
(b) RRT based on dynamic environment
Moving
obstacle
(c) APF based on static environment
(d) APF based on dynamic environment

Fig. 13. The path planning result comparison between RRT and APF.

RRT.

In the absence of obstacles, the straight line distance between the two points is the shortest. Therefore, we define L_{SG} as the path length from start point to goal point, and it can be calculated by

$$L_{SG} = \sqrt{(x_S - x_G)^2 + (y_S - y_G)^2} \tag{28}$$

where (x_S, y_S) are the coordinates of the starting point S, and (x_G, y_G) are the coordinates of goal point G.

The shortest path length of the three environments are calculated as 37, 38.01, and 140.93, respectively. We add the shortest paths in the path length comparison, and use the shortest distance as a comparison criterion. Then, we define P_c as an indicator to evaluate the optimal path of the path planning method, and it is calculated by

$$P_c = 100 - \frac{L - L_{SG}}{L_{SG}} \tag{29}$$

where L represents the path length based on RRT, GA, and PSO, respectively.

Based on Eq. (29), the path optimal degree can be measured in Table 6. As described in Table 6, it can be clearly seen that the path optimal degree of RRT is higher than PSO, and PSO is higher than GA.

From Fig. 11 and Fig. 12, the running time and path length based on GA is much higher than other two. Compared RRT with PSO, the performance of RRT is better than PSO. Therefore, RRT is effective and efficient in path planning of mobile robot.

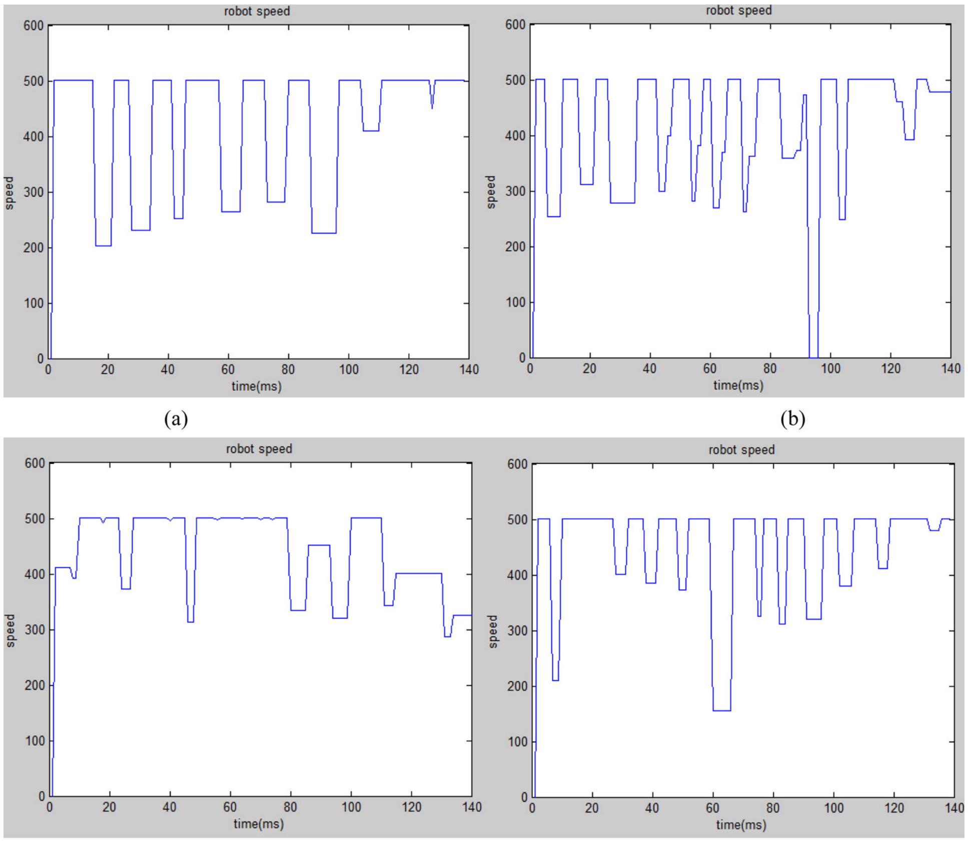


Fig. 14. Velocity of mobile robot in path planning.

Table 7
Comparison of the RRT and APF method.

NO.	Method	Start point	Goal point	Processing time	Path length
a	RRT	(50,50)	(460,460)	1.60s	638.5pixel
b	RRT	(50,50)	(460,460)	3.05s	704.5pixel
c	APF	(50,50)	(460,460)	1.78s	797.4pixel
d	APF	(50,50)	(460,460)	3.64s	910.8pixel

Table 8
DI values based on Table 7.

	DI_t	DI_l
Static environment	89.88 %	80.07 %
Dynamic environment	83.79 %	77.35 %

5.4. Comparisons in dynamic environments

In order to validate the efficiency of the proposed algorithm, we compare RRT method with APF method in dynamic environment. The path planning result comparison between RRT and APF is shown in Fig. 13. Here, Fig. 13(a) and Fig. 13(c) are the initial environment before obstacle moving. And Fig. 13(b) and Fig. 13(d) are the dynamic obstacle environment, where one of the obstacle moves. Concretely speaking, the cross-shaped obstacle moves in the direction indicated by the yellow arrow. And the speed of mobile robot is shown in Fig. 14.

It can be seen from Fig. 14 that the frequency of the speed change of the robot when dynamic obstacle is present is larger than that of a static obstacle. In addition, the speed is kept for a certain time because of waiting rules.

Table 7 gives the experimental results for the initial and modified environments, respectively. For the same input image, set the same start point and goal point. It is obviously shown on the results that the processing time and path length based on RRT is shorter than the corresponding result based on APF, whether in an initial or dynamic obstacle environment. In order to get a quantitative analysis of the path planning results based on two methods, we define DI to measure the degree of accuracy improvement.

$$DI_t = 1 - \frac{|t_{RRT} - t_{APF}|}{t_{APF}} \bullet 100\% \tag{30}$$

$$DI_l = 1 - \frac{|l_{RRT} - l_{APF}|}{l_{APF}} \bullet 100\% \tag{31}$$

According to the above two Equations, the degree of accuracy improvement is computed, as shown in Table 8. This table records the DI value of static environment (compared Fig. 9(a) with 9(c)) and dynamic environment (compared Fig. 9(b) with 9(d)) based on RRT and APF methods, respectively.

From Table 8, we can see that the processing time improved more than 80 %, while the path length improved more than 77 %. The path planning accuracy and running time in a dynamic obstacles environment will be further improved in the future.

In a word, RRT method is better than APF in computation time and path length. It can be seen from the above experimental results that the proposed path planning method has better path planning performance while avoiding obstacles.

6. Conclusion

In this paper, a novel intelligent path planning approach with obstacle avoidance of mobile robot in a dynamic environment is presented. The intelligence of this method lies in the use of deep learning to identify the type of obstacles and distinguish between static obstacles and dynamic obstacles. In two-dimensional space, the improved ray tracing is used to avoid static obstacles. And the dynamic obstacle avoidance is completed by the proposed waiting rule. Finally, based on the two steps, the path planning is implemented by RRT method. A software tool is developed using MATLAB R2018b to obtain the simulation results. RRT is compared with GA and PSO in static environment, and it is compared with APF in dynamic environment. In addition, the comparison performance includes running time and path length of path planning results. The experimental results show that the proposed algorithm is capable and flexible for path planning in a known indoor environment. However, there are also some limitations, such as it is not suited for the environment of outdoors, and the robot used here can only move forward, in the future, it can extend to mini-cars which can move forward and backward. This approach has the potential to make paths better and path planning more efficient.

The future research directions will focus on multi-robots adaptive navigation and robot positioning based on simultaneous localization and mapping.

Declaration of Competing Interest

We confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influence its outcome. We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the order of authors listed in the manuscript has been approved by all of us.We confirm that we have provided a current, correct email address which is accessible by the Corresponding Author and which has been configured to accept email from (email: zhanglinxjtu@126.com).

Acknowledgments

This research is supported by Natural Foundation Research Project of Shaanxi Province (Grant No.2017ZDJC-21) and National Natural Science Foundation of China (Grant No.51875445).

References

[1] R. Wei, Research and Implementation of Path Planning for Mobile Robot Based on Reinforcement Learning Guangzhou, School of Computer Science and Engineering, South China University of Technology, 2015.
[2] X.M. You, S. Liu, J.Q. Lv, Ant colony algorithm based on dynamic search strategy and its application on path planning of robot, Proc. IEEE Conf. Decis. Control 32 (2017) 552–556.
[3] L.X. Huang, Y.C. Geng, Robot path planning based on dynamic potential field method, Computer Measurement and Control. 25 (2017) 164–166.
[4] H.Y. Lee, H. Shin, J. Chae, Path planning for mobile agents using a genetic algorithm with a direction guided factor, Electronics 7 (10) (2018) 212.
[5] Z. Zhou, J. Wang, Z. Zhu, Tangent navigated robot path planning strategy using particle swarm optimized artificial potential field Optik, International Journal for Light and Electron Optics 158 (2018) 639–651.
[6] J.W. Li, L.H. Xu, Research on path planning based on simulated annealing algorithm and neural network algorithm, Automation & Instrumentation (2017).
[7] H.H. Chen, X. Du, W.K. Gu, Path planning method based on neural network and genetic algorithm, International Conference on Intelligent Mechatronics and Automation (2004) 667–671.
[8] D. Janglova, Neural networks in mobile robot motion, International Journal of Advanced Robotics Systems 1 (2004) 15–22.
[9] M.K. Singh, D.R. Parhi, Path optimisation of a mobile robot using an artificial neural network controller, Int. J. Syst. Sci. 42 (2011) 107–120.
[10] J.L. Sanchez-Lopez, M. Wang, M.A. Olivares-Mendez, A real-time 3D path planning solution for collision-free navigation of multirotor aerial robots in dynamic environments, J. Intell. Robot. Syst. 8 (2018) 1–21.
[11] E.G. Tsardoulias, A. Iliakopoulou, A. Kargakos, A review of global path planning methods for occupancy grid maps regardless of obstacle density, J. Intell. Robot. Syst. 84 (2016) 1–4.

[12] J.X. Gu, Z.H. Wang, Recent advances in convolutional neural networks, Pattern Recognit. 77 (2018) 354–377.
[13] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Commun. ACM 60 (2017) 84–90.
[14] C. Szegedy, W. Liu, Y. Jia, Going deeper with convolutions, IEEE Conference on Computer Vision and Pattern Recognition (2015) 1–9.
[15] H.A. Hagras, A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots IEEE Trans, Fuzzy Systems 12 (2004) 524–539.