


RESEARCH ARTICLE

A novel navigation system for an autonomous mobile robot in an uncertain environment

Meng-Yuan Chen¹, Yong-Jian Wu² and Hongmei He^{3,*} 

¹College of Electrical Engineering, Anhui Polytechnic University, Wuhu, China, ²Wuhu HIT Robot Technology Research Institute Co. Ltd., Wuhu, China and ³School of Computer Science and Informatics, De Montfort University, Leicester, LE1 9BH, UK

*Corresponding author. E-mail: h.he@dmu.ac.uk

Received: 20 March 2020; **Revised:** 10 February 2021; **Accepted:** 2 April 2021; **First published online:** 2 August 2021

Keywords: sliding window; adaptive threshold clustering; obstacle recognition; collision prediction; Morphin path planning

Abstract

In this paper, we developed a new navigation system, called ATCM, which detects obstacles in a sliding window with an adaptive threshold clustering algorithm, classifies the detected obstacles with a decision tree, heuristically predicts potential collision and finds optimal path with a simplified Morphin algorithm. This system has the merits of optimal free-collision path, small memory size and less computing complexity, compared with the state of the arts in robot navigation. The modular design of 6-steps navigation provides a holistic methodology to implement and verify the performance of a robot's navigation system. The experiments on simulation and a physical robot for the eight scenarios demonstrate that the robot can effectively and efficiently avoid potential collisions with any static or dynamic obstacles in its surrounding environment. Compared with the particle swarm optimisation, the dynamic window approach and the traditional Morphin algorithm for the autonomous navigation of a mobile robot in a static environment, ATCM achieved the shortest path with higher efficiency.

1. Introduction

Researchers have thoroughly investigated the mobile robot navigation problems and achieved a certain progress in this area. As the navigation of autonomous mobile robots with free collisions could increase the range of their applications, it has played an important role in various application domains, such as transportation, rescue services, detection, mining, space exploration and military. For example, autonomous vehicles are constantly improving with the autonomous navigation capability [21] and have obtained the qualification on roads in some countries (e.g., Google autonomous vehicle). In the fields of coal mines, intelligent mobile robots have been used to replace the manual work and tedious repetitive operations [32]. Robots are key to future space exploration [6], and the autonomous navigation of mobile robots allows them to have better operation capabilities. In the military field, mobile robots with autonomous navigation can better help reach military missions, such as military patrols on the desert border [20]. For the success of all applications above, effectively autonomous navigation of robots is the precondition and guarantee.

Robot's ability to navigate in an environment without a need for physical or electro-mechanical guidance devices has made it more promising and useful. During the navigation of a robot in a recognised or uncontrolled environment, usually it cannot take the direct route from its starting point to the destination. This means the robot must have the capability of perception, localisation, cognition and motion planning [2]. Hence, autonomous navigation requires a robot to be able to precept the surrounding environments through processing or fusing the data, collected from sensors, and highly performed robot perception enables a robot to make a right decision and thus to have a right response to any anomaly situation

in its surrounding environment. The major challenges of robot navigation lie in localisation, obstacle avoidance and motion planning [2, 23], of which, obstacles avoidance in dynamic environments is a crucial issue in the navigation of a robot and has been paid much attention by researchers [24, 40, 51]. Earlier research focused on static obstacle avoidance. The research of Weerakoon et al. [44] shows that the classic artificial potential field (APF) algorithm, which treats the robot's configuration as a point in a potential field that combines attraction to the goal and repulsion from obstacles to generate a robot's trajectory, has a major problem that the robot is easy to be trapped at a local minimum before reaching its goal.

One of the main objectives for robot navigation is to improve the navigation performance. Improving real-time performance of robot's navigation is a critical challenge. While a robot can avoid any static or dynamic obstacles on its paths, the following three performance indicators are often used to evaluate the navigation of a robot.

Positioning accuracy, including static single point positioning error and dynamic track error, where single point positioning error consists of X error, Y error and total error maximum, expectation, entropy and super entropy, and dynamic trajectory error includes robot direction error, robot position maximum error, error mean, error expectation, error entropy and error hyper-entropy. He et al. [16] used linguistic decision tree (LDT) for robot routing problem, and the fuzzy technology in the LDT enables the robot to effectively overcome the stochastic errors due to such factors as mechanical properties of robot's motor and the friction changing on the floor even in a static environment, thus reducing the accumulated error of robot's position.

Speed index, including the maximum speed or average speed of the mobile robot during the whole test process. Especially, the maximum speed depends on the computing complexity and mechanical properties of a robot. Under the mechanical properties of a robot, the computing complexity of robot navigation determines the maximum speed, representing the real-time performance.

Navigation efficiency, including the length of time T used by the mobile robot to complete a specific task, which is related to the speed V , the total length L of the entire process trajectory. Under a specific speed V , the length of the path that robot travels from a start point to the target represents the navigation efficiency.

The navigation of a robot should be robust in respects of the performance indicators above, reflecting on all steps of the whole process. However, most of the existing research addressed partial steps of a navigation process. There was little investigation of the whole navigation system. This is not helpful for evaluating the performance of the whole process of a robot's navigation.

To overcome the challenges in robot's navigation, we proposed a holistic autonomous navigation system of a robot, named as ATCM [9]. In this research, we further improve the new navigation system with the six stages from data collection, obstacle detection, obstacle classification, potential collision prediction and optimal path planning to the robot's behaviour updating. The merits of ATCM lie in that (1) we proposed a clustering algorithm with an adaptive threshold [55] for obstacle detection and a decision tree for obstacle classification, based on the natural features of different types of obstacles, represented by the data from sensor on the robot; (2) a heuristic algorithm is provided for potential collision prediction in terms of the dynamics of the robot and obstacles; (3) we further simplify the Morphing algorithm [35] for optimal path planning without losing the performance and (4) modular design of 6-steps navigation provides a holistic methodology to implement and verify the performance of a robot's navigation system (Fig. 1).

Hence, in this paper, we examine how all the steps work harmonically and time changes for eight scenarios, which are typical scenarios in real dynamic world. The navigation system is simulated and integrated in a physical robot. The experiments are conducted on the simulation system and a physical robot.

This paper is organised as follows: Section 2 reviews the state of the arts in robots' navigation; Section 3 provides the details of the proposed navigation system; Section 4 provides three groups of experiments, including simulation, validation on a physical robot and comparison with other three

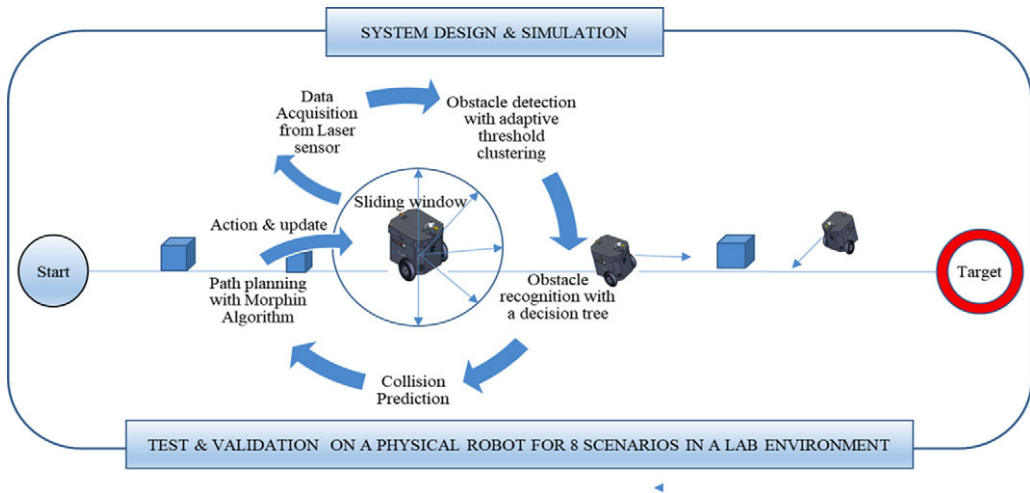


Fig. 1. The flow chart of the navigation system.

path planning algorithms under static environment; finally, conclusions and future work are given in Section 5.

2. Existing Work

Navigation of mobile robots is a classic issue in robotics. The robot's route learning problem is one of the robot's navigation problems. Conventionally, an NARMAX model, a non-linear system identification approach, was trained to represent the sensor-motor task [15]. To improve the robustness of the robot's navigation and overcome the drawback of NARMAX in losing performance due to the dynamics of a running robot, a LDT was developed for the robot routing problem [16] and achieved excellent performance. In fact, both of the two methods are guided by human, in which the data are retrieved when human drives the robot along a specific path in a fixed environment, and then the model (e.g., NARMAX or LDT) is trained. Hence, this work is suitable for a robot working on a specific task in a fixed environment.

However, for most cases, we expect robots can work in an uncertain environment with free collision. There has been much research in this area. Generally, it can be categorised to map-based navigation and map-less navigation. Whereas map-based navigation can be subdivided into metric map-based navigation and topological map-based navigation, map-less navigation can include reactive techniques based on qualitative characteristics extraction, appearance-based localisation, optical flow, features tracking, plane ground detection/tracking, etc [4]. A grid-based mapping technique is usually used in the map-based navigation system for an autonomous mobile robot [10].

Simultaneous localisation and mapping (SLAM) is a technique, used by robots and autonomous vehicles to build a map within an unknown environment or to update a map within a known environment, while keeping track of their current locations [3]. Various SLAM techniques have been developed. For example, Kovacs et al. [22] proposed a landmark-selection approach based on explicit and spatial information for solving mobile robot navigation problems, where the visual odometry-based motion estimation supports the template matching of landmarks. The important characteristic of SLAM techniques that could assist in autonomous navigation is the ability of a mobile robot to concurrently construct a map for an unknown environment and localise itself within the same environment. However, a SLAM system might fail in handling extremely dynamic or harsh environments. Storing the map during long-term operation is still an open problem. Even when the data are stored on the cloud, raw data points or

volumetric maps may cost much memory; similarly, storing feature descriptors for vision-based SLAM quickly becomes burdensome [7].

To implement a robot's autonomy, its perception to external environments is very important. Advanced sensors provide enabling techniques for robots' perception. Most frequently, used sensors include laser sensors [16, 49, 54], visual sensors [52], infrared sensors [41] and ultrasonic sensors [43]. Different sensor techniques may decide different navigation techniques.

Now, laser sensors are increasingly used, as they have the advantages of wide detection range, high-precision measures, high reliability, good stability, strong anti-interference and lightweight. Recently, Xin et al. [47] developed a dynamic obstacle detection model by fusing the Velodyne data from a 3D laser sensor and the motion state information from a 4-wire laser sensor to derive the position of a moving obstacle in a grid map based on the confidence distance theory. In the research [48], an approach to detecting the speed and direction of an obstacle was proposed, and the obstacle avoidance was implemented, regarding the least Euclidean distance from the robot to the edges of the obstacle. As only circular objects were addressed, this method is not robust for the diversity of obstacles.

Vision techniques have played significant roles in robots' navigation [4]. For example, in the early stage, the highly notable planetary vehicle, Mars Pathfinder [25], used a stereo camera to shoot images of the Mars surface, when the rover explores the environment, which is controlled by human through selecting the goal point in 3D representations of previously captured images of the terrain. Further, pattern recognition from images has been widely used for obstacle detection in autonomous robot navigation [8, 38]. Recently, Dirik et al. [11] provided a novel design for the kinematic control structure of the wheeled mobile robot (WMR), using their previous visual-servo path planning method based on interval type-2 fuzzy logic (IT2FIS) [12].

Artificial intelligence techniques are important drivers for implementing the autonomy of a robot. Especially, machine learning techniques have been widely applied for obstacle detection. For example, neural networks have been developed for a robot's path planning [30]. A support vector machine based on the spatial-time feature vector was developed to recognise dynamic obstacles, but without the further exploration of the path planning for obstacle avoidance [18]. Wang and Zhou [42] proposed a geomagnetic gradient bionic model with a parallel approach for robot navigation, which becomes a multi-objective convergence problem. Mohanta and Keshari [26] proposed a knowledge-based fuzzy control system for target search behaviour and path planning of mobile robots. This approach includes two stages: the first stage is to generate a shortest path between the starting position and the target position in a previously known messy environment, wherein the probability roadmap is used to construct a straight path by connecting the intermediate nodes; the second folding step helps to convert the sharp angle to a smooth curve throughout the path. Recently, Rubio et al. [33] proposed a fuzzy approach for path following of a nonholonomic mobile robot, based on the information of a frontal camera, including three stages: detect and isolate the desired path, estimate the orientation and use a fuzzy control system to steer the direction of the mobile robot. The research of Sanchez et al. [34] shows that generalised Type-2 fuzzy controllers outperform their Interval Type-2 and Type-1 fuzzy controller counterparts in the presence of external perturbations for controlling a mobile robot.

It is commonly believed that deep convolutional neural networks (CNNs) are good for vision-based pattern recognition problems. Hence, CNNs have been applied to improve the perception of robots. For example, Steccanella et al. [36] proposed a two-stage approach to detecting waterline and obstacles based on the images taken by the cameras mounted on a low-cost autonomous surface vehicle for environmental monitoring, through using a CNN-based image segmentation to infer the position of waterline, which is then used for obstacle detection, however, an embedded GPU board is needed for the segmentation; Wu et al. [45] proposed a two-stage path planning method based on a CNN. Firstly, the comprehensive features are extracted directly from original images of roads through a CNN; then, robots determine their moving direction in terms of the classification results from the CNN. This approach is good for a static environment, but not good for dynamic environments. Hence, it could be applied for robots' pre-training to get an initial path in a static environment. Due to the computing complexity, the first stage

is completed offline. Online neural network training to adapt the dynamic environment is still an open question.

Various meta-heuristic algorithms have been developed for robot path planning as well. Usually, these approaches transform the path planning problem to an optimisation problem to implement offline planning by producing a shortest path in a static environment with stationary obstacles, such as the ant colony algorithm [5, 31], genetic algorithms (GAs) [19], the simulated annealing-based approach [14] and particle swarm optimisation (PSO) of APF [1]. As they are time-consuming, offline planning is needed. Based on the path generated by offline planning, the robot travels through the stationary obstacles, then the robot will recalculate the path when a dynamic obstacle appears on the path. The issue of such two-stage approaches is that the remaining path might not be the best after recalculating the path to pass the detected dynamic obstacle, especially in dense obstacle environments.

Herojit Singh and Khelchandra [17] proposed a mobile robot navigation approach based on fuzzy GA. Information about the distance and angle of obstacles from the robot is obtained by the exploration of three directions in front of the robot (40° , 0° and 40°). When all three paths are blocked by obstacles, the fuzzy system is used to avoid obstacles; otherwise, the conflict-free path is selected from the three directions. The GA is used to find the optimal range of linguistic values of the variables of the membership function. It is shown that fuzzy GA with the three-path concept is computationally efficient, compared to other hybrid methods. Wan et al. [39] proposed an improved ant colony algorithm, combining the Morphin algorithm for mobile robot path planning. In the process of the global optimisation, the Morphin algorithm is applied for local path planning in each step. Recently, Orozco-Rosas et al. [29] proposed a hybrid path planning algorithm based on membrane pseudo-bacterial potential field, where a pseudo-bacterial GA is applied to find the best parameters in the APF method.

The Morphin algorithm was proposed based on the Ranger algorithm for obstacle avoidance and safeguarding of a lunar rover in space [35]. The Morphin path planning is an area-based approach. The basic idea is that the patches of terrain are analysed to determine the traversability of each patch. The Morphin algorithm has the merits of low computing complexity, and thus, it can implement the real-time performance for a robot's path planning [50]. Hence, the Morphin algorithm has been widely applied for path planning in an uncertain environment with dynamic obstacles [39, 57], and it, integrating in a global path planning algorithm, has achieved good performance for the path planning in indoor and complex environments [46, 53].

Various approaches for robot path planning have been developed by researchers [28] and obtained significant performance in solving certain aspects of the path planning problem. However, they have their own disadvantages [27, 37]. Also, many researchers only validated their algorithms in simulation. Therefore, a high-performance autonomous navigation system in effectiveness and efficiency is still demanded.

3. The Proposed System

A navigation system of a mobile robot is thoroughly investigated, regarding all the six steps shown in Fig. 1. Initially, the robot sets a sliding window, the centre of which is the robot; the environmental data are collected from the laser scanner on the robot; static and/or dynamic obstacles in the surrounding environment are detected based on the data within the window, using the adaptive-threshold clustering algorithm; a simple decision tree, learned from experienced parameters, is used to determine the types of obstacles (new, dynamic or static); the movement of dynamic obstacles is computed, in terms of their speed and direction and the relation between the robot and the detected obstacles, and the potential of conflict is predicted; the Morphin algorithm is applied to avoid obstacles if a potential collision in front of the robot is not avoidable; finally, the robot updates its state, moving towards the generated local sub-target, and correspondingly, the sliding window is updated; the process is repeated until the robot reaches the global target. Algorithm 1 provides the pseudo-code of the navigation system.

Algorithm 1 RobotNavigation($r_w, R, Target$)

```

Initialise( $r_w, R$ ); /* a sliding window with radius of  $r_w$  */
 $t=0$ ;
while ( $R$  has not reached the  $Target$ ) do
 $D=ReadData()$ ; /* from laser scanner */
 $O_{chain}= Clustering (D)$ ;
  for ( $O_k(t) \in O_{chain}(t)$ ) do
 $O_{type}=ObstacleRecognition(O_k(t), O_{chain}(t-1))$ ;
 $O_{info}=calculateObstacleInfo(O_k(t), O_{type})$ ;
 $Collision = CollisionPrediction(O_{info}, R)$ ;
    if ( $Collision \neq NULL$ ) then
      if ( $Collision$  is avoidable) then
 $V_R= V_R - \Delta V_R$ ; /* slow down the robot */
      else
        Mophin( $Collision, R$ );
      end if
    end if
  end for
  update( $R, r_w$ );
   $t = t + 1$ ;
end while

```

3.1. Data acquisition from the laser sensor

A grid map with a specified resolution is usually created in map-based navigation [56]. In this research, a grid-based environment model is established. Assume the robot is in the global coordinate system, sharing the origin of the global coordinate system. To collect the information of obstacles in the surrounding environment of the robot, a two-dimensional laser sensor, the product of the German company SICK, is installed on the robot. The value of readings from the laser scanner on the robot is proportion to the distance between the robot to an obstacle, which reflects the time interval between sending and return of the laser beams. In each step of data collection, the laser scanner scans the front semicircle of the robot, ranging in $[0^\circ, 180^\circ]$, with an angular interval of 0.5° .

Figure 2 illustrates the positions of a robot and an obstacle and their relationship in positions. The pair of (ρ_i, α_i) represents the location of an obstacle in the local polar coordinates, where the robot is the original point, ρ_i indicates the length of a laser beam, representing the distance between the obstacle and the robot, $\psi_i \in [0^\circ, 180^\circ]$, $i = 0 \dots 360$, indicating the index of the laser beam; α_i is the angle between a laser beam and the robot's direction and θ_R is the angle of the robot in the global polar coordinates. The coordinates of the obstacle's position in the global coordinate system can be calculated with Eq. (1). The global coordinates (x_o, y_o) of an object can be transferred to the coordinates (x'_o, y'_o) in the grid map with Eq. (2).

$$\begin{cases} x_0 = x_R + \rho_i \cos(\theta_R - \alpha_i), \\ y_0 = y_R + \rho_i \sin(\theta_R - \alpha_i). \end{cases} \quad (1)$$

$$\begin{cases} x'_o = \text{floor}\left(\frac{x_o}{r} + \frac{1}{2}\right), \\ y'_o = \text{floor}\left(\frac{y_o}{r} + \frac{1}{2}\right). \end{cases} \quad (2)$$

where r is the resolution of the grid map (see Fig. 3).

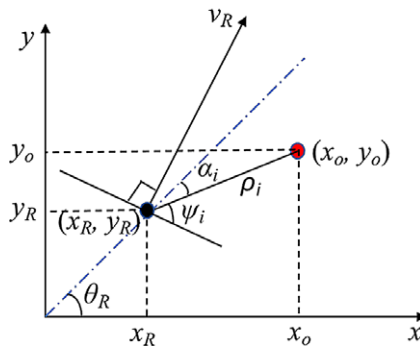


Fig. 2. The robot model.

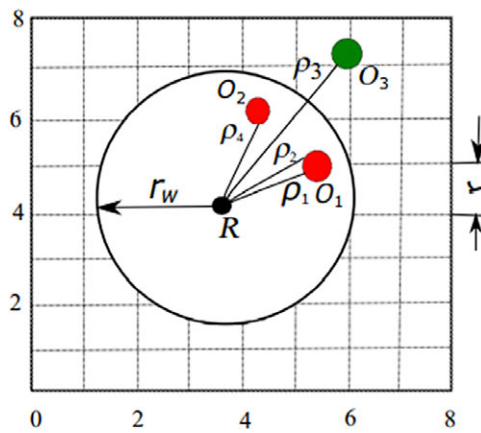


Fig. 3. The grid map model.

3.2. Data clustering for obstacle detection

After collecting the data from the laser scanner, the robot will check if there exist any obstacles within the sliding window. An adaptive threshold nearest-neighbour clustering method is developed to group data points. It is believed that the data out of the sliding window (i.e., $\rho > r_w$) do not provide any hazard to the robot. Hence, it will not be used for clustering, like O_3 in Fig. 3, as $\rho_3 > r_w$. The Euclidean distance is used to represent the distance between two data points. Two available consecutive data points (ρ_2 and ρ_4) are believed to belong to different obstacles, respectively, if the distance between them is larger than the distance between two neighbouring laser beams that have the identical length and the angular interval 0.5 (e.g., ρ_1 and ρ_2). Therefore, a threshold ϑ is defined, linearly related to the value of $[\rho(t)\sin(0.5)]$, which is the approximate of the distance between two neighbouring data points in one cluster. ϑ is proportional to the current laser beam $\rho(t)$. As the shape of an obstacle could be irregular, we use an adaptive rate to indicate the irregularity of obstacle shapes. Two closed obstacles could be viewed as one cluster. This could improve the robustness of the clustering algorithm. For the rectangle of obstacles, λ can be set to close to 1.

$$\vartheta = \lambda \rho(t) \sin(0.5^\circ) \tag{3}$$

To cluster the data points, one can calculate the distance between the current data point and the previous data point in turns of the data points from 0° to 180° . If the distance is larger than the threshold, then the current data point belongs to a new cluster, otherwise, it belongs to the cluster that the previous data point belongs to. A cluster represents an obstacle. Finally, the clustering algorithm will produce a

chain of obstacles O_{chain} (Eq. (4)) at time t .

$$O_{chain} = O_1(t), O_2(t), \dots, O_n(t) \tag{4}$$

The attributes of an obstacle, $Ok(t)$, can be represented as the vector of four items in Eq. (5).

$$Ok(t) = (Z_k(t), S_k(t), \xi_k(t), V_k(t)), \tag{5}$$

where $Z_k(t)$ is the centre of $O_k(t)$, $S_k(t)$ is the area of the grids that $O_k(t)$ occupies, $\Xi_k(t)$ is the coincidence of $O_k(t)$ to an obstacle at time $t-1$ and $V_k(t)$ is the speed of $O_k(t)$. If $O_k(t)$ is a dynamic obstacle, then $V_k(t) > 0$, otherwise, $V_k(t) = 0$. Assume $O_k(t)$ is represented by cluster C_k , which includes n_k laser beams, l_1, \dots, l_{n_k} , and $l_i = (\rho_i, \alpha_i)$. The centre $Z_k(t)$ of $O_k(t)$ can be calculated with Eq. (6), and the global coordinates of the centre can be calculated with Eq. (1).

$$\begin{cases} \bar{\alpha}_k = \frac{\sum_{i=1}^{n_k} \alpha_i}{n_k} \\ \bar{k} = \frac{\sum_{i=1}^{n_k} l_i}{n_k} \end{cases} \tag{6}$$

It is easy to obtain the pairs $(\min(\rho), \min(\alpha))$ and $(\max(\rho), \max(\alpha))$ in cluster C_k , which determine the edges of cluster C_k . The global coordinates of cluster edges $(x_{k,\min}, y_{k,\min})$ and $(x_{k,\max}, y_{k,\max})$ can be calculated with Eq. (1). Further, the grid coordinates of cluster edges $(x'_{k,\min}, y'_{k,\min})$ and $(x'_{k,\max}, y'_{k,\max})$ can be calculated with Eq. (2). Hence, the area S_k covers all grids within the coordinate ranges, expressed by Eq. (7).

$$\begin{cases} x' \in [x'_{k,\min}, x'_{k,\max}] \\ y' \in [y'_{k,\min}, y'_{k,\max}] \end{cases} \tag{7}$$

One can calculate the global and grid coordinates of all data points in C_k at time t with Eqs. (1) and (2), respectively. Assume obstacles $O(t)$ and $O(t-1)$ occupy areas $S(t)$ and $S(t-1)$, respectively. The grid coordinates of $S(t)$ and $S(t-1)$ are shown in Eq. (8).

$$\begin{cases} x(t) \in [x_{\min}(t), x_{\max}(t)] \\ y(t) \in [y_{\min}(t), y_{\max}(t)] \\ x(t-1) \in [x_{\min}(t-1), x_{\max}(t-1)] \\ y(t-1) \in [y_{\min}(t-1), y_{\max}(t-1)] \end{cases} \tag{8}$$

If $S(t)$ and $S(t-1)$ fully overlap, the (x_{\min}, y_{\min}) and (x_{\max}, y_{\max}) of the two areas should be the same. Hence, Eq. (9) is true.

$$\begin{cases} S(t) \cap S(t-1) = S(t) \\ S(t) \cup S(t-1) = S(t) \end{cases} \tag{9}$$

If $x_{\max}(t-1) < x_{\min}(t)$ or $x_{\min}(t-1) > x_{\max}(t)$ or $y_{\max}(t-1) < y_{\min}(t)$ or $y_{\min}(t-1) > y_{\max}(t)$, then the two areas do not overlap, otherwise, the two areas overlap partially or fully.

If $S(t)$ and $S(t-1)$ partially overlap, their x ranges and y ranges overlap. Hence, one can sort the boundary values of x at times t and $t-1$, and the boundary values of y at times t and $t-1$. Then the order of x boundary values $x_{b1} \leq x_{b2} \leq x_{b3} \leq x_{b4}$ and the order of y boundary values $y_{b1} \leq y_{b2} \leq y_{b3} \leq y_{b4}$ can be obtained. Hence, the overlapping area has the boundary of $[(x_{b2}, x_{b3}), (y_{b2}, y_{b3})]$, and the number of overlapping grids can be calculated with Eq. (10).

$$S(t) \cap S(t1) = (x_{b3} - x_{b2} + 1)(y_{b3} - y_{b2} + 1) \tag{10}$$

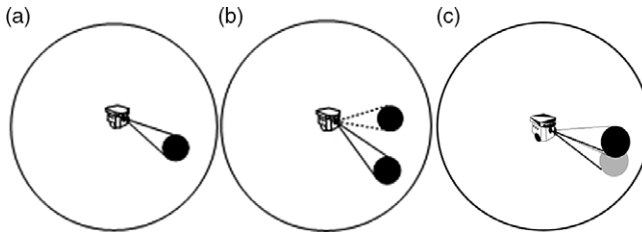


Fig. 4. The types of obstacles. (a) New, (b) static and (c) dynamic.

The total number of grids that the two obstacles occupy can be calculated with Eq. (11).

$$S(t) \cup S(t - 1) = S(t) + S(t - 1) - S(t) \cap S(t - 1). \tag{11}$$

Further, the coincidence of O_k can be expressed as Eq. (12).

$$\xi_k(t) = \frac{S_{(t)} \cap S_{(t-1)}}{S_{(t)}}. \tag{12}$$

The spatial correlation ($\zeta_{k1,k2}$) between two obstacles can be expressed as a function of two parameters: the distance (δ) between the centres of two clusters and the non-overlapping rate (η), and $\zeta_{k1,k2}$ can be calculated with Eq. (13), in which δ and η are expressed with Eq. (14), and y_δ and y_η are the efficiencies.

$$\zeta_{k1,k2} = \zeta(O_k(t), O_k(t - 1)) = \frac{y_\delta}{\delta + 1} + \frac{y_\eta}{\eta + 1} \tag{13}$$

$$\begin{cases} \delta = \|Z_k(t), Z_k(t - 1)\|, \\ \eta = 1 - \frac{S_{O_{k1}}(t) \cap S_{O_{k2}}(t - 1)}{S_{O_{k1}}(t) \cup S_{O_{k2}}(t - 1)}. \end{cases} \tag{14}$$

If $O_k(t)$ and $O_k(t-1)$ represent the same obstacle, the distance between their centres should be zero (i.e., $\delta = 0$). If two obstacles fully overlap, $\eta = 0$. If two obstacles are isolated, $\eta = 1$. Assume $y_\delta = 0.5$ and $y_\eta = 0.5$. Two fully overlapped obstacles have $\zeta = 1$. The maximal value in all spatial correlations between $O_k(t)$ and all $O_{k2}(t-1) \in O_{chain}(t-1)$ is denoted as $\zeta_{k(t),max}$, and expressed in Eq. (15). The maximal spatial correlations of $O_k(t)$ are one of the important parameters for distinguishing the type of the obstacle.

$$\zeta_{k(t),max} = \max_{k2=1..nk(t-1)} \zeta_{k(t),k2(t-1)} \tag{15}$$

3.3. Recognition of obstacle types

For each obstacle, $O_k(t) \in O_{chain}(t)$, one can easily calculate the centre $Z_k(t)$, the grid area $S_k(t)$ and the coincidence $\Xi_k(t)$ with Eqs. (6)–(12), as well as the spatial correlation with Eqs. (13) and (14). Further, one can identify the maximal spatial correlation $\zeta_{k(t),max}$ of obstacle $O_k(t)$ to the obstacles in $O_{chain}(t-1)$. Figure 4(a)–(c) shows three types of obstacles: new, static and dynamic. To recognise which type of obstacle is met by the robot in its surrounding environment, a simple decision tree is used, as shown in Fig. 5. In the decision tree, the first attribute node is the maximal spatial correlation $\zeta_{k,max}$ of the obstacle $O_k(t)$. Two thresholds $\vartheta_{\zeta 1}$ and $\vartheta_{\zeta 2}$ of the spatial correlation are set to recognise whether an obstacle is new or static, where $(0 < \vartheta_{\zeta 1} < \vartheta_{\zeta 2} < 1)$. If $\zeta_{k,max} < \vartheta_{\zeta 1}$, then it is a new obstacle (Fig. 4(a)); if $\zeta_{k,max} > \vartheta_{\zeta 2}$, then the obstacle is static (Fig. 4(c)); if $\zeta_{k,max} \in [\vartheta_{\zeta 1}, \vartheta_{\zeta 2}]$, then it might be a dynamic obstacle (Fig. 4(b)) or not. So, it will be further assessed in terms of the distance ($0 \leq \delta < r_w$) between two obstacles' centres.

A threshold ϑ_δ of the centre distance is set to further judge whether the obstacle is static or not. If $\delta < \vartheta_\delta$, it can be recognised as a static obstacle, otherwise, a threshold ϑ_Ξ of the obstacle coincidence

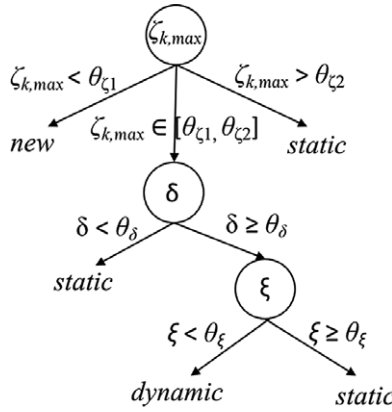


Fig. 5. Obstacle recognition with decision tree.

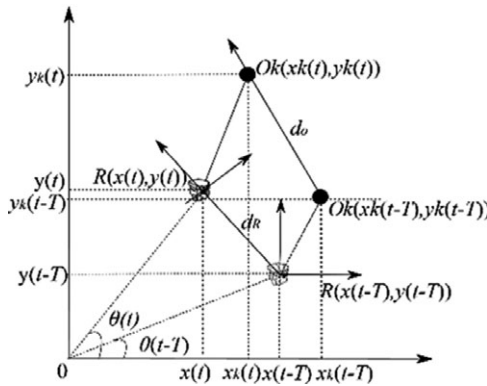


Fig. 6. Motion process of an obstacle.

$\Xi_{k(t)}$ is used to further judge whether the obstacle is static or dynamic. If $\Xi_{k(t)} < \vartheta_{\Xi}$, then it is dynamic, otherwise, static. Here, all threshold values are obtained through primary experiments.

3.4. Movement of a dynamic obstacle

If the obstacle is dynamic, the speed and angle of its movement will be further calculated. Figure 6 illustrates the motion process of the obstacle and the robot from $t-T$ to t in the global coordinates system, where T is the time period of laser scanning from 0° to 180° . A moving robot in the environment has the global coordinates $R(x(t), y(t))$ at time t and $R(x(t-T), y(t-T))$ at time $t-T$. One can easily calculate the global coordinates of the obstacle, $O_k(x_k(t), y_k(t))$ at time t and $O_k(x_k(t-T), y_k(t-T))$ at time $t-T$, in terms of the values of laser beams at time t and $t-T$, via Eq. (1). It is easy to calculate the speed v_o , the direction angle α_o and the distance d_o of the moving obstacle from $t-T$ to t via Eq. (16). In the same way, one can calculate the speed v_r , the direction angle α_r and the distance d_r of the moving robot from time $t-T$ to t .

$$\left\{ \begin{array}{l} v_o = \frac{d_o}{T} \\ \alpha_o = \arctan \left(\frac{x_k(t) - x_k(t-T)}{y_k(t) - y_k(t-T)} \right) \\ d_o = \sqrt{(x_k(t) - x_k(t-T))^2 + (y_k(t) - y_k(t-T))^2} \end{array} \right. \quad (16)$$

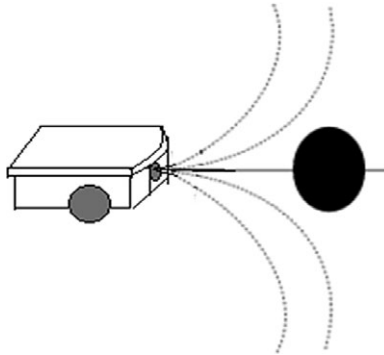


Fig. 7. Morphin paths.

The potential collisions ahead can be predicted in terms of the states of the robot and the obstacle. There could be eight scenarios where a robot is running, as shown in Fig. 8. Scenario (a) is the simplest that there is no obstacle, where the robot is running in the straight direction towards the target; in scenario (b), an obstacle is statically staying on the path where the robot is going, and the potential collision is just at the place where the obstacle is, if the robot does not change its path; in scenario (c), an obstacle at the probing area of the robot may cross the path of the robot before the robot arrives the crossing point, which may be a potential collision point (PCP) if robot speeds up; in scenario (d), an obstacle at the probing area of the robot may cross the path of the robot, but it has not arrived at the PCP when the robot arrives there; in scenario (e), the robot would collide with the obstacle, when they arrive at the crossing point between the robot's path and the obstacle's path at the same time; in the scenario (f), the obstacle is moving on the path of the robot but towards the robot; hence, the robot would collide with the obstacle, if it does not change its path; in scenario (g), the robot and the obstacle are moving on the same path, and the robot is behind the obstacle, but faster than the obstacle, then the robot would collide with the obstacle at a point on the path if the robot neither deduces its speed nor changes its path; scenario (h) shows both dynamic and static obstacles are on the path of the robot. This requires robot could avoid both obstacles on the path in real time, no matter how close the two obstacles are.

3.5. Obstacle collision avoidance

The classic Morphin algorithm is used to implement the local path planning based on limited environment information. If a robot identifies an obstacle and finds that it might collide with the obstacle, then it will set up a few alternative paths and select the best path in the alternative paths to replace the path where the robot and the obstacle will meet, thus avoiding the obstacle (Fig. 7). In the eight scenarios (Fig. 8), scenarios (b), (e)–(h) exist potential collisions. Hence, the robot needs to adjust their running parameters. For scenario (e), the robot and the dynamic obstacle are running on the crossing paths and could meet at the crossing point; hence, the robot only needs to reduce its speed, thus allowing the dynamic obstacle to pass the PCP before the robot reaches the point, or increase its speed to enable itself to pass the PCP before the obstacle arrives at the point; for scenario (g), the robot can reduce its speed, make it keep a certain distance to the obstacle in front of the robot. The Morphin algorithm assumes that the robot is facing to the obstacle. Hence, it is applicable for scenarios (b) and (f). We can connect the robot's current position and the centre of the obstacle to form a centreline and draw several arcs on the two sides of the centreline, respectively (Fig. 7). Strictly, the central line should be the link between the robot's position and the PCP (red point in Fig. 8(b) and (f)).

As the robot is running in a grid environment, each of alternative arcs could occupy several grids. In the robot navigation, the path length is an important performance indicator. Moreover, the robot should not bear away from original path very much; hence, the turning angle should be as small as

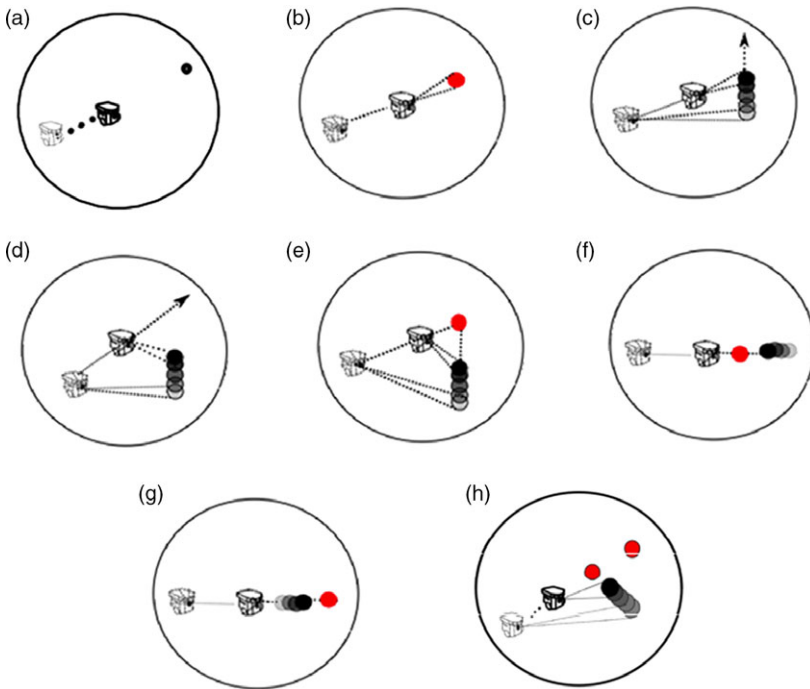


Fig. 8. Eight scenarios.

possible. Hence, the evaluation of each arc can be done in terms of the parameters of the arc in the grid environment, such as the arc length, the angle to the line from the robot to the target, as well as the distance between the path end and the sub-target. Wan et al. [39] formulated the evaluation function of each arc as Eq. (17).

$$y = \begin{cases} \infty, & \text{if the path crosses an obstacle (a PCP),} \\ \epsilon_1 L + \epsilon_2 G + \epsilon_3 \Delta L + \epsilon_4 W, & \text{others.} \end{cases} \quad (17)$$

where L is the length of each arc path, which is represented with the number of grids that the arc goes through from the start point (the place of robot) to the end point of the arc; G is the parameter at the inflexion point on each arc path, and it is to ensure the robot does not go far away; ΔL is the average distance from each grid where the arc goes through to the sub-target on the global path, and it is to ensure the alternative path has a small distance to the sub-target; for scenarios (a) and (e), the sub-target could be set at the point of next grid close to the obstacle or collision point; $W = \frac{1}{m+1}$, m is the number of grids where both the arc and the global path go through. The parameter W is to ensure the alternative path as close to the global path as possible. $\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4$ are the coefficients of the four items, respectively. If the arc crosses the obstacle, the value of y is ∞ , and the smallest y indicates the alternative path is the best in the local probing area of the robot.

For the simplicity of computing, the arcs can be drawn in straight lines (Fig. 9), and the length of all paths beside the central line is the same as the length of the central line from the robot to the PCP. Hence, the evaluation function could be simplified as Eq. (18) regardless of the arc length L .

$$y = \begin{cases} \infty, & \text{if the path crosses an obstacle (a PCP),} \\ \mathcal{K}_1 |G| + \mathcal{K}_2 \Delta L + \mathcal{K}_3 W, & \text{others.} \end{cases} \quad (18)$$

where G is the angle between an alternative path and the central line (e.g., G_1 , and G_2), $-\pi/2 < G < \pi/2$; ΔL is the distance between the target and the sub-target (R' in Fig. 9), divided by the grid edge length; W represents the number of times when the line between an alternative path end point and the sub-target

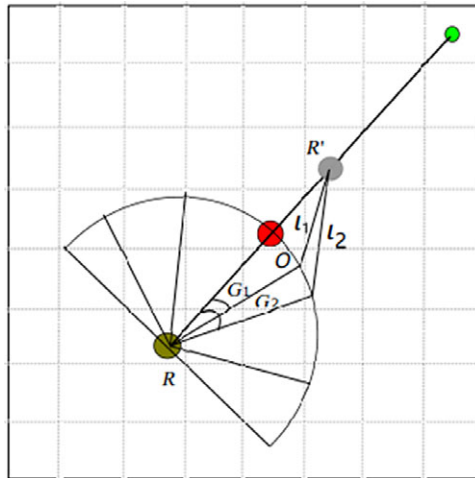


Fig. 9. Morphin paths on a grid map.

point intersects the global path. Obviously, if the current position of the robot is on the global path (the straight line from start to the target), W is at most $1/2$. If the robot is not on the global path, it is possible that a path does not intersect any grid on the global path, for which, $W = 1$. The larger the number of grids where the path intersects the global path, the smaller the value of W . $\mathfrak{R}_1, \dots, \mathfrak{R}_3$ are the weights of parameters in the evaluation function. For example, as shown in Fig. 9, $G_1 < G_2$, $\Delta L_1 < \Delta L_2$, the number of grids where both l_1 and the global path go through is 4, while the number of grids where both l_2 and the global path go through is 3. Hence, $W_1 = \frac{1}{1+4} = 0.2$, $W_2 = \frac{1}{1+3} = 0.25$, and $W_1 < W_2$. Obviously, l_1 is a better path than l_2 , as $y_{l_1} < y_{l_2}$.

There is no potential collision in scenarios (c) and (d). Hence, the robot will not change its moving parameters and continue. In scenario (e), the robot will stop until the obstacle passes the predicted collision point; in scenarios (b) and (f), a potential collision is identified, and the robot has to change its direction. Hence, the robot calls the Morphin algorithm to select the optimal alternative path instead of the original path to avoid the potential collision; in scenario (g), the robot could slow down to avoid the potential collision. However, if the PCP has been in the sliding window, to avoid the rear-end accident, the robot either stops or calls the Morphin algorithm to change the direction. Scenario (g) is different to scenario (f), as the robot and the obstacle in scenario (g) are moving in the same direction. In this research, for scenario (g), the robot will slow down or stop to avoid potential collision.

4. Experiments and Evaluation

Three groups of experiments are conducted: (1) the simulation experiments; (2) scenario-based validation and robustness experiments on a physical robot and (3) comparison of ATCM with other path planning algorithms, such as the PSO algorithm, the traditional Morphin algorithm and the dynamic window approach (DWA) [13], which is a velocity-based local planner that translates a Cartesian goal (x,y) into a velocity (v,w) command for a mobile robot. DWA includes two main goals, calculate a valid velocity search space that produces a safe trajectory, and select the optimal velocity to maximise the robot's clearance, maximise the velocity and obtain the heading closest to the goal.

4.1. Simulation experiments

The simulation is to verify the effectiveness of the proposed navigation system through some experiments. A specific start point and a specific target point are set up for robot's navigation tasks. All

parameters in the experiments are set through the primary experiments with trial-and-error method. A 20×20 grid environment with many static obstacles is set up. The edge of each grid is set to 500 mm. The radius r_w of the sliding window should be larger than the running distance of the robot in a running cycle, then robot can predict the collision earlier; thus, it has enough time to deal with any potential collision before reaching the PCP. Namely, it should meet the condition in Eq. (19), where T_{ATCM} is the total time of a six-step cycle and the actuator's response. T_{ATCM} depends on the running environment. If the environment is more complex, then T_{ATCM} is longer. The average T_{ATCM} is about 0.157s, according to our primary experiments. In the simulation, r_w is set to 8 grids, while the robot is running at the speed of 100 mm/s. Hence, r_w is set to 4 m, such that the system is robust for various environments.

$$r_w \gg T_{ATCM} \times V_R \quad (19)$$

As all obstacles added to the grid environment have a regular shape, the value of adaptive threshold for data points clustering is set to 1.2; for simplicity, the values of thresholds in the decision tree (Fig. 5) for obstacle recognition are set as: $\vartheta_{t1}=0.30$, $\vartheta_{t2}=0.7$, $\vartheta_{\delta}=0.4$ and $\vartheta_{\varepsilon}=0.5$, respectively. The parameters ($\mathfrak{R}_1 \sim \mathfrak{R}_3$) of Morphin algorithm are set to 1, 1.3 and 0.6, the same as the corresponding values of $\varepsilon_2 \sim \varepsilon_4$ in the research [39]. Three experiments are conducted: (1) static obstacles only, (2) some dynamic obstacles appear in the grid environment and (3) static and dynamic obstacles appear instantly.

4.1.1. Static obstacles in the environment.

Assume there is a density of static obstacles in the experiment environment (Fig. 10) and no dynamic obstacles appear during the navigation of the robot. The start point is the bottom-left corner of the grid map, and the target is the top-right corner of the grid map. A mobile robot is moving from the specific start point to the specific target point in the unknown environment. The global optimal path is the straight line from the start to the target. During the navigation, if the mobile robot finds a static obstacle on its global path in the grid environment, it will call the Morphin path planning to avoid the obstacle towards next sub-target. In each step of the navigation, robot uses its current position as the centre to update the sliding window and moves towards the next step. The local sub-targets are updated towards the global target step by step and constructs an actual trajectory from the start to the target (Fig. 10(a)–(d)). Finally, the robot completes the navigation. The experimental results demonstrate that the robot can effectively avoid the obstacles with multiple static obstacles on the global optimal path from start to the target.

4.1.2. Instantly appeared dynamic obstacles in the environment.

No matter what scenarios in which a robot is running, the robot will generate a local sub-target in each step. The computing complexity is increased for predicting PCP and path planning to avoid the dynamic obstacle. Figure 11 shows the simulation of the local path planning process by adding three types of dynamic obstacles to the experimental environment. The mobile robot starts from the bottom-left corner, retrieves the data from the laser scanner and identifies the dynamic obstacle Ob_1 in the current probing area. The robot calculates the motion parameters of Ob_1 and predicts that it will not reach the point where Ob_1 crosses the path. This is scenario (c) in Fig. 8. Hence, the robot will not collide with Ob_1 , and it will keep the original speed and the moving direction. The mobile robot continues and finds Ob_2 , a moving obstacle. The robot predicts that it will reach the point at the time when the obstacle crosses the path. Namely, the robot could collide with Ob_2 , as shown in scenario (e) of Fig. 8. Hence, it stops or slows down until Ob_2 passes the collision point (Fig. 11(c)). When a dynamic obstacle Ob_3 appears on the path that the robot is going through in the opposite direction towards the robot, as scenario (f) in Fig. 8, the collision is unavoidable if the robot does not change its direction. The robot predicts the PCP, calls the Morphin algorithm to get an optimal path, moves to the sub-target and follows the global path until reaching the final target, as shown in Fig. 11(d).

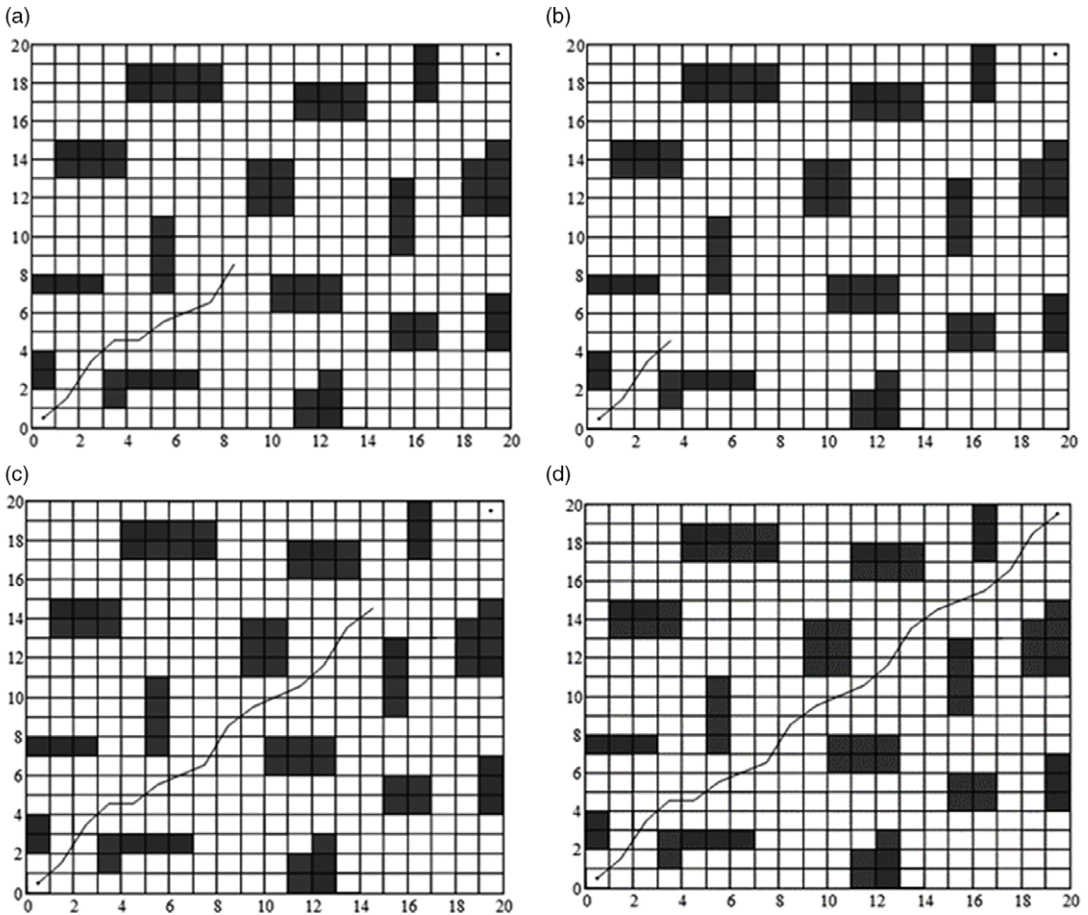


Fig. 10. Local path planning in a static environment. (a) Moving process 1: start point. (b) Moving process 2. (c) Moving process 3. (d) Moving process 4.

4.1.3. A mixed case.

Figure 12 illustrates the navigation process of instantly adding static and dynamic obstacles in the environment. In each step, the sliding window is updated in real time, and the trajectory of the robot forms the central trajectory of the sliding window. In Fig. 12, V_1, V_2, V_3 and V_4 denote the sliding windows, S_1, S_2, S_3 and S_4 are static obstacles in the environment and $D_1, D_2, D_{3-1}, D_{3-2}$ and D_4 are the dynamic obstacles appearing in the environment. The speeds and the angles of the four dynamic obstacles detected in the environment are shown in Table I.

As illustrated in Fig. 12, when the robot creates the sliding window V_1 , at point A, it finds the static obstacle S_1 and executes the Morphin path planning to avoid the collision with S_1 . The robot moves from A to B and finds the dynamic obstacle D_1 with the speed of 450 mm/s and the angle of 80.83° ; it predicts that there is no potential collision with S_1 and continues towards the target; a sliding window of V_2 is created immediately after the robot runs beyond the probing area of V_1 . When the robot moves from C to D, it finds the static obstacle S_2 and the dynamic obstacle D_2 with the speed of 760 mm/s and the angle of 62.47° and predicts that there will be no collision with both of S_2 and D_2 . At point D, the robot finds S_3 with potential collision, it executes the Morphin path planning again to avoid S_3 ; in the same way, when the robot is beyond the range of V_2 , a sliding window of V_3 is updated. At point E, the robot finds the dynamic obstacle D_{3-1} at a speed of 510 mm/s and the angle of 91.05° and predicts that it could collide with the obstacle at point O_1 . The robot stops until the obstacle D_{3-1} passes the

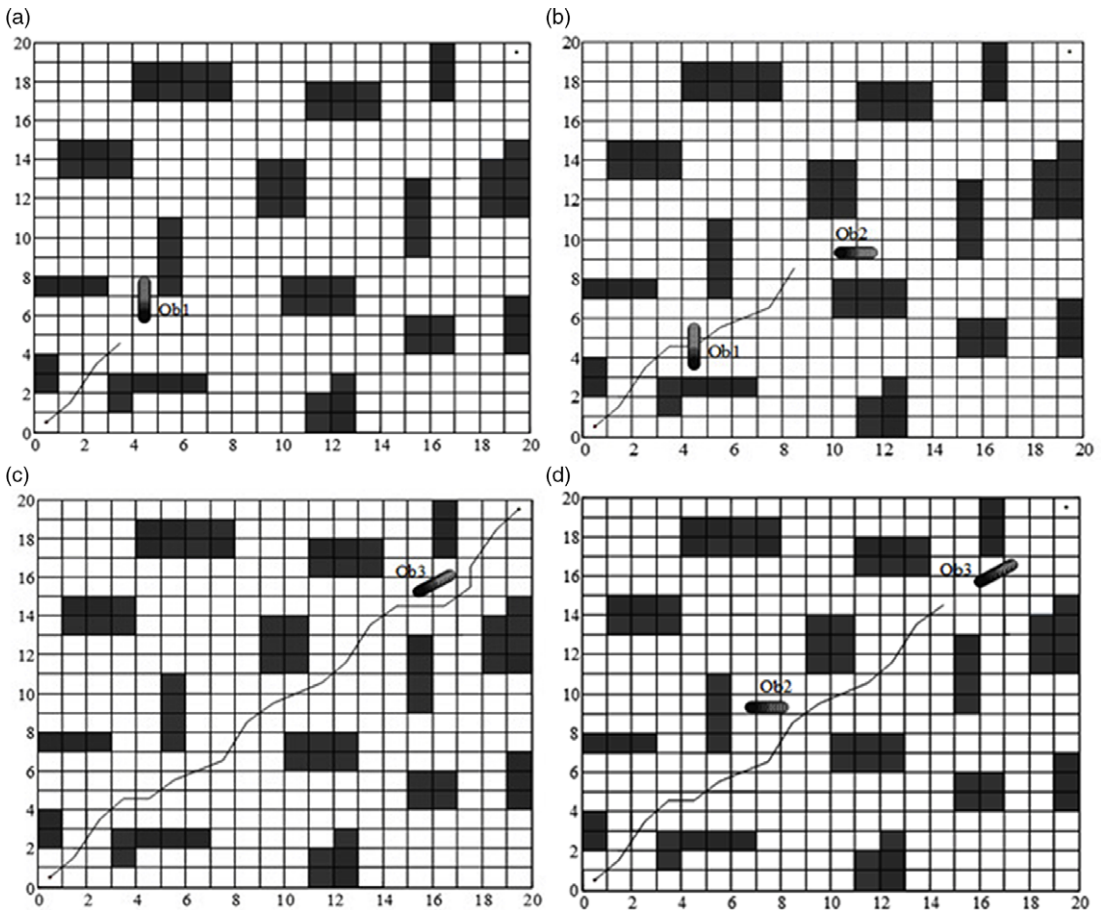


Fig. 11. Local path planning in an environment with dynamic obstacles added instantly. (a) Starting point: adding Ob1. (b) Avoiding Ob1, adding Ob2. (c) Avoiding Ob2, adding Ob3. (d) Avoiding Ob3, reaching the target.

PCP on the path to become D_{3-2} in the environment. In the sliding window of V_4 , at point F , the robot finds the dynamic obstacle D_4 in the opposite direction towards the robot, at the speed of 805 mm/s and the angle of 180.05° . The robot predicts that the obstacle could collide with it at point O_2 , if it does not change its path immediately. Hence, the robot executes the Morphin path planning immediately to select an optimal path, thus avoiding the collision with the obstacle D_4 .

4.2. Experiments on a physical robot

4.2.1. The robot in the experiment.

An ‘UP-Voyager II-A’ mobile robot (Fig. 13(a)), the product of Beijing Bochuang Xingsheng Robot Technology Co., Ltd., is used for the performance test experiment. The robot is equipped with a SICK two-dimensional laser scanner, a twelve-sonar ring, a Logitech camera and an odometer. But the data are collected from the laser sensor. The robot has a self-loading computer with i7-6700HQ, 4-Core processor, 16GB memory, GTX960M and 2G discrete graphics. Table II provides the specification of technical parameters of the robot. On the Ubuntu 14.04 of the PC, ROS indigo is running.

Table I. Movement speed and angle of obstacle.

	D_1	D_2	D_3	D_4
v	450 mm/s	760 mm/s	510 mm/s	805 mm/s
α	80.83°	62.47°	91.05°	180.09°

Table II. Specification of Voyager-II.

Item names	Description
SICK Laser scanner	Cover 0°–180°, 0.5° resolution, 361 points in the range [0°, 180°]
12 sonar sensors	Cover 360°, each sensor can detect obstacle within 30° cone
Camera	Logitech camera
Motor encoder	500 units per circle
Dell XPS15-9550-R4825	Control and communicate with the robot

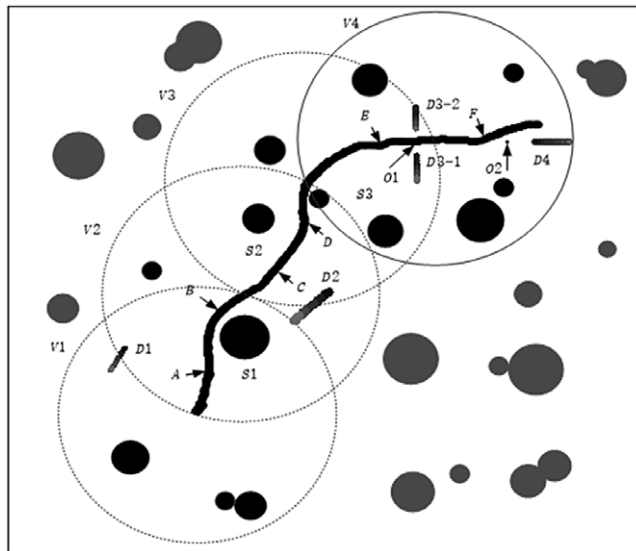


Fig. 12. Simulation results for the mix case.

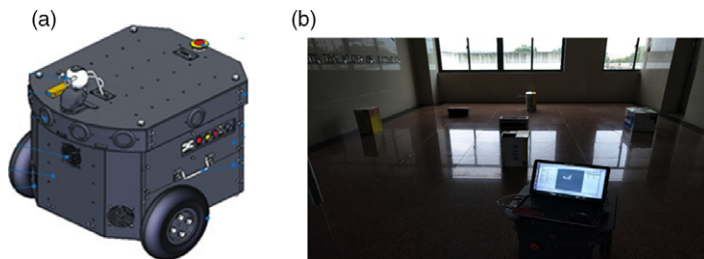


Fig. 13. The UP-Voyager IIA robot and the lab environment. (a) UP-Voyager IIA. (b) The lab environment.

Table III. Approaching times of the robot for different scenarios.

Scenarios	Approaching time(s)			Av. (s)
(a) No obstacle	61.77	62.00	61.98	61.92
(b) A static obstacle in front of the robot	65.48	64.37	63.89	64.58
(c) A faster dynamic obstacle	62.68	62.27	61.57	62.17
(d) A slower dynamic obstacle	62.05	62.24	61.94	62.07
(e) Predicted collision with a dynamic obstacle	64.53	65.02	64.97	64.87
(f) A dynamic obstacle in opposite direction towards the robot	76.12	77.20	76.45	76.59
(g) A slower dynamic obstacle in the same direction in front of the robot	75.61	76.01	75.92	75.85
(h) Mix obstacles	77.85	78.56	78.97	78.46

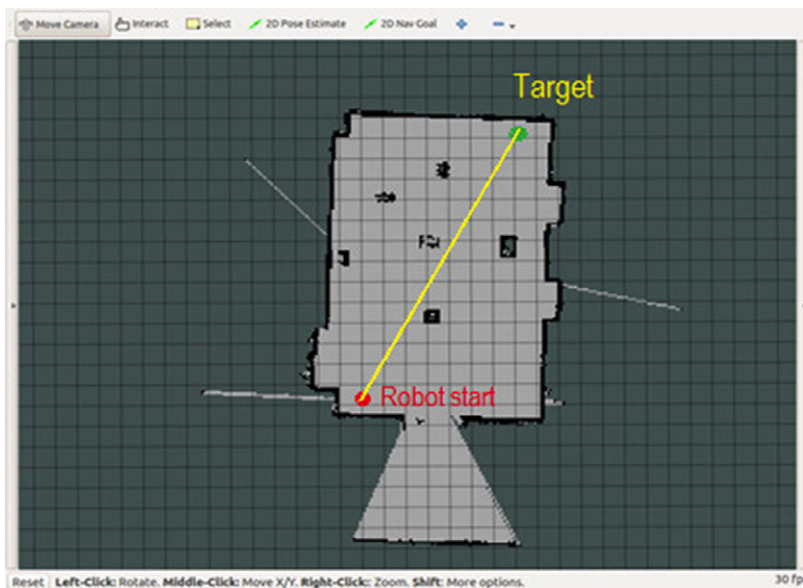
**Fig. 14.** The grid map of the robot lab environment.

Figure 13(b) depicts the environment for the experiments of robot's navigation. The developed navigation system is embedded into the robotic system. The robot is driven within the specified environment without any obstacle, and the environment information is collected. A high quality of grid map of the environment is produced and visualised in the window of the RVIZ system, as shown in Fig. 14.

4.2.2. Scenario-based experiments and assessment.

The experiments are conducted for all the eight scenarios in Figs. 7(a)–(h) in the robotics lab. A mobile robot will start from the same point, automatically search and avoid obstacles on its path, and finally reach the same target. The initial speed of the robot is $V_R = 100$ mm/s, and the straight-line distance between the start point and the target is 6 m. The start time is recorded when the mobile robot starts, and the end time is recorded when the robot reaches the target. To reduce the random errors, each experiment is repeated three times, and the average running time of the robot is calculated. The experimental results are shown in Table III.

Experimental results show: for scenario (a), the robot does not meet any obstacle, thus it directly reaches the target with smallest time; for scenario (b), the robot finds a static obstacle, successfully avoids the obstacle, and reaches the target; for scenario (c), the robot finds a dynamic obstacle, which has a faster speed than the robot, the robot predicts the obstacle will pass the path before it reaches the PCP, keeps the original speed and direction and safely reaches the target; for scenario (d), the robot finds a slower obstacle, the robot predicts that it will not conflict with the detected obstacle if the robot keeps the original speed and the direction; for scenarios (e), (f) and (g), the robot must adjust moving direction or speed; otherwise, it will conflict with the obstacle. For scenario (e), the robot either speeds up or slows down to avoid the obstacle; for scenario (f), the robot must change direction, so the robot applies the Morphin algorithm to avoid the obstacle; for scenario (g), the robot reduces its speed, so that it is slower than the obstacle in its front. The experimental results demonstrate that the robot can continuously deal with different obstacles in its front. From the results in Table III, it can be concluded that:

- (1) the results for scenario (a) show the robot spent the shortest time to reach the target, namely for all other scenarios, the robot needs to spend some time to detect and avoid obstacles select a proper path.
- (2) for scenarios (c) and (d), although the robot does not need to change its moving direction and speed to reach the target, it still needs to spend time to detect the obstacle and make the decision. For these two scenarios, the running times of the robot are similar, but they are shorter than that for scenarios (b) and (e), in which the robot needs to run the Morphin algorithm to select a suitable path.
- (3) for scenarios (b) and (e), the running time of the robot is for obstacle detection, obstacle classification, collision prediction and path planning. Obviously, the running time is longer than that in scenarios (c) and (d). The average time for static obstacle (scenario (b)) is slightly less than that for dynamic obstacle (scenario (e)).
- (4) for scenario (f), as the obstacle is running in the opposite direction towards the robot, the robot detects the obstacle multiple times, and even after the robot changes its direction, the obstacle is still in the range of the sliding window. Hence, the running time is longer than scenarios (a)–(e).
- (5) for scenario (g), the obstacle is running in the same direction as the robot, the robot is faster than the obstacle and follows after the obstacle; hence, the distance between the robot and the obstacle is getting closer and closer. In this scenario, the robot detects the obstacle many times, and once the robot identifies that it could collide with the obstacle, it will slow down and keep a certain distance to the obstacle. Hence, the running time of the obstacle is longer than scenarios (a)–(e), but slightly shorter than scenario (f); as in scenario (g), the robot does not call the Morphin function for path planning.
- (6) In scenario (h), the robot needs to deal with two obstacles: a static obstacle and a dynamic obstacle. Hence, the robot continues detecting obstacles on the path towards the target. When the robot finds out the dynamic obstacle, it reduces its speed to avoid the potential collision; when the robot finds out the static obstacle on the path, the robot needs to call the Morphin function for path planning. Therefore, the running time of the robot, reaching the target, is longest in all scenarios.

4.2.3 Robustness of the navigation system in an environment with multiple obstacles.

The robot, used for the validation of robustness, is the same as in Section 4.2.2, and the experiments are conducted in the same lab environment, but with multiple static or dynamic obstacles. Robot's speed $V_R = 100$ mm/s, $V_{Ob1} = 0$ mm/s, $V_{Ob2} = 75$ mm/s, $V_{Ob3} = 150$ mm/s, $V_{Ob4} = 100$ mm/s.

First, a grid map of the lab environment is created. Second, the start and the end points of robot navigation are set up. Under the static environment without dynamic obstacles, the robot can run along

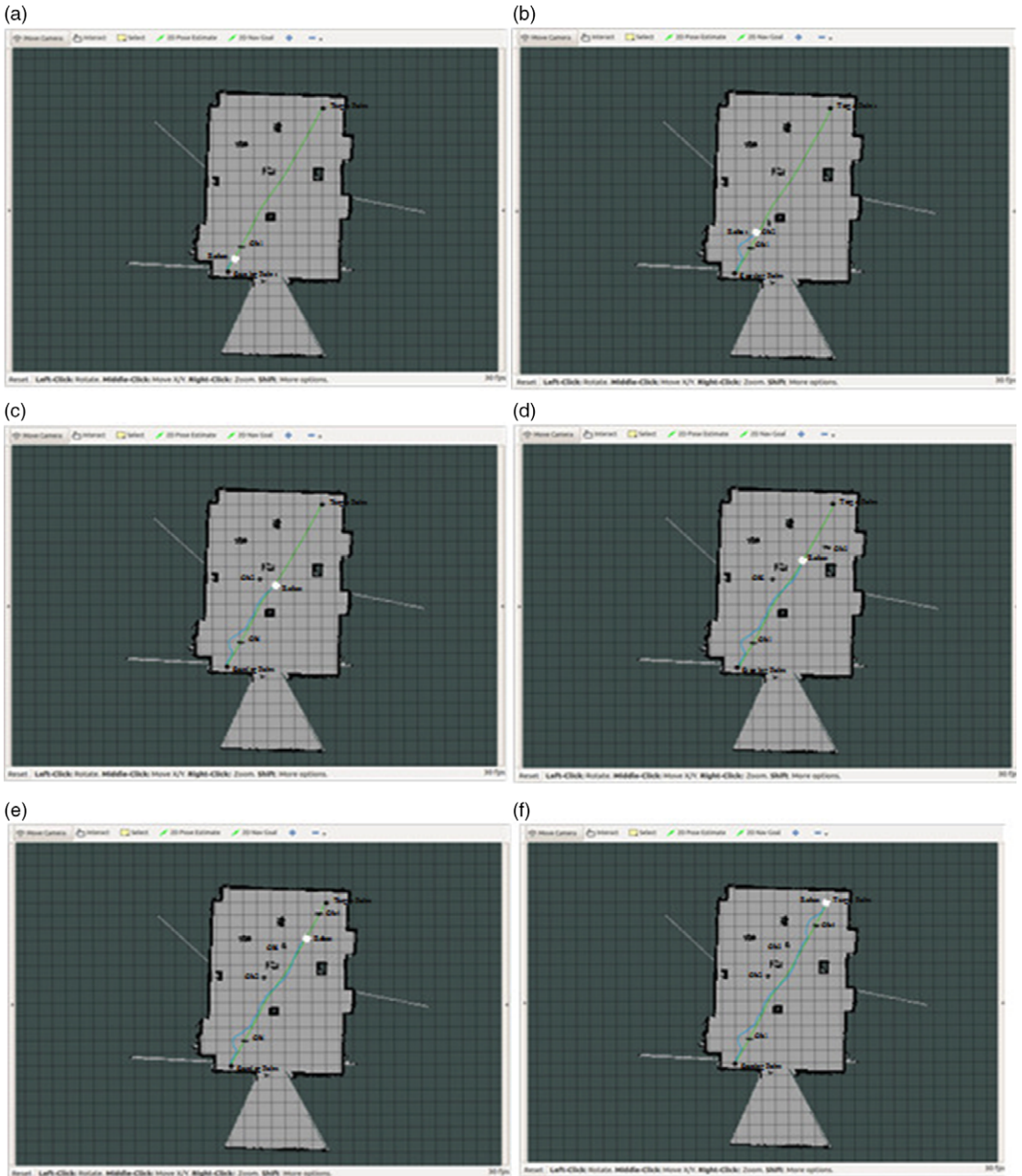


Fig. 15. Local path planning in the lab environment. (a) Starting point: static Ob1 detected. (b) Avoiding static Ob1, dynamic Ob2 detected. (c) Keep moving with original speed, avoiding Ob2. (d) Dynamic Ob3 detected. (e) Avoiding Ob3, dynamic Ob4 detected. (f) Avoiding Ob4, reaching the target.

with the optimal path, the straight line from the start point to the target, shown as the yellow line in the grid map in Fig. 14.

In the lab environment, adding some temporary static and dynamic obstacles, the mobile robot detects obstacles through the laser scanner, classifies obstacles, predicts the state of obstacles and avoids them on the path. Figure 15 depicts the process of robot navigation from the start to the target. The green line

Table IV. The main experimental parameters of the four algorithms.

PSO	Population	Inertia weight	Learning factor C_1 and C_2	Maximum iteration
	50	0.9	1.2	300
Traditional Morphin	Searching paths 15	Searching radius 2 m	Temporal resolution 0.1 s	Maximum iteration 300
DWA	Line speed resolution 0.01m/s	Angle speed resolution 1.0° /s	Temporal resolution 0.1 s	Maximum iterations 300
ATCM	Laser scanning angle interval 0.5°	Sliding window radius 4 m	Clustering adaptive threshold 1.2	Maximum iteration 300

is the optimal line, and the blue line is the real path that the robot executes. As shown in Fig. 15(a), the actual moving path of the mobile robot is along with the blue line, while the static and dynamic obstacles are added successively. The mobile robot moves along the global path from the start point. When finding a static obstacle Ob_1 appeared on the global path (scenario (b)), the robot calls the Morphin algorithm and changes its path. After safely avoiding the obstacle Ob_1 , the robot gets back to the global path; after moving for a distance, a dynamic obstacle Ob_2 is detected (Fig. 15(b)), and the potential collision is predicted to occur at the front of the robot (scenario (e)). The obstacle avoidance strategy is that the mobile robot stops in the place until the obstacle passes the path and leaves away (Fig. 15(c)). As shown in Fig. 15(d), when the dynamic obstacle Ob_3 is detected, the mobile robot predicts the potential collision and judges that the original speed can be maintained (scenario (d)). As shown in Fig. 15(e), when the obstacle Ob_4 is detected, the predicted collision is inevitable (scenario (f)); hence, the robot carried out the Morphin algorithm immediately and changes its path before the collision to avoid the obstacle, then goes back to the global path, and finally, it reaches the target (Fig. 15(f)).

Briefly, the experimental results have proved that the proposed navigation system is effective. As the computing is simple, the navigation can perform in real time. It can be seen that, for scenarios (b) and (f), the robot can effectively avoid the potential collision if it changes its path, using the Morphin path planning; for scenarios (e) and (g), the robot can avoid the potential collision by stopping or slowing down its speed; for scenario (g), if the robot needs to overtake the slowly moving obstacle in its front, then it needs to call the Morphin algorithm to change its path, which is similar to the scenario (f). But it should be noticed that the sub-target, at which the robot goes back to the global path for scenario (g), should be further than the sub-target on the global path for scenarios (b) and (f), as the obstacle in scenario (g) is running on the same direction as the robot.

4.3. Comparison experiments

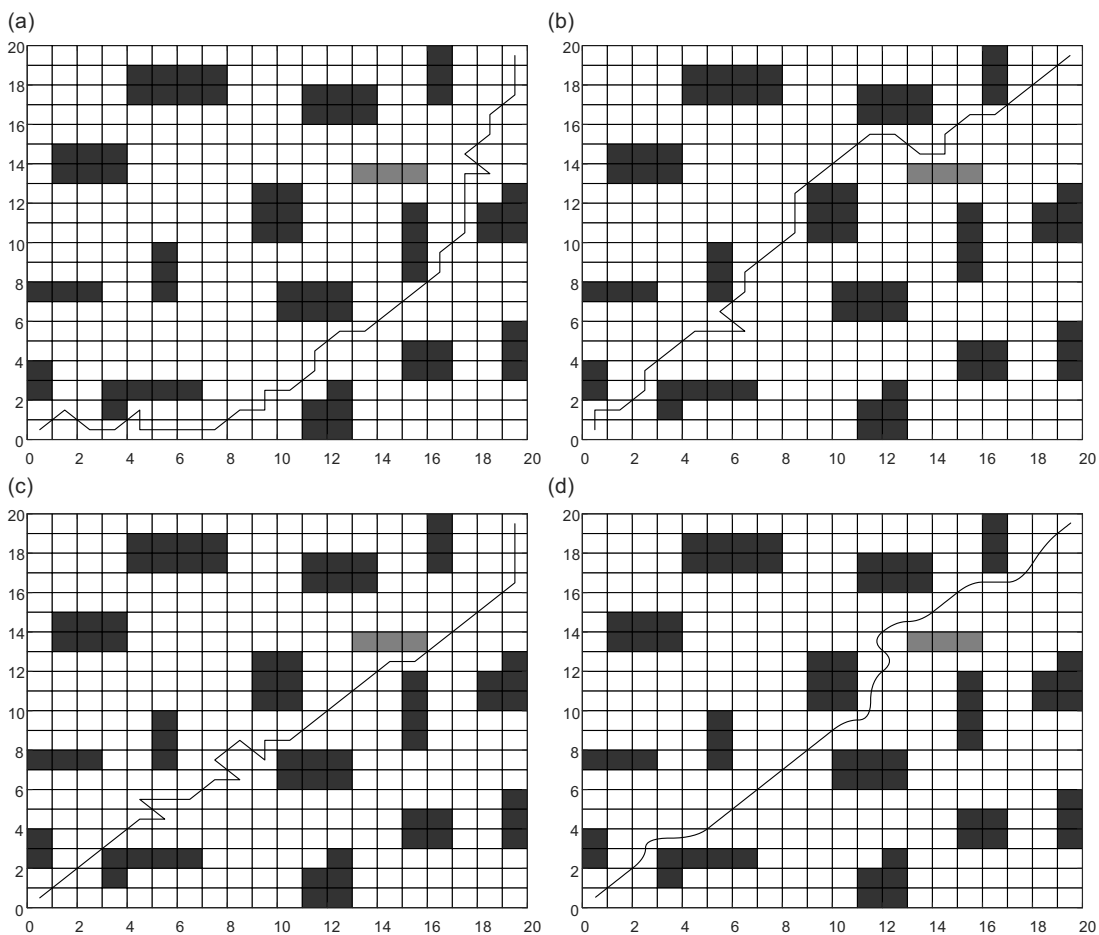
4.3.1 Performance of different algorithms under new static obstacle environments.

For easy comparison, the simulation experiments of the four path planning algorithms are conducted under the environment with static obstacles. The experimental parameters are set as shown in Table IV.

The black areas in the grid map, shown in Fig. 16, are inherent obstacles to the environment and the grey areas are new static obstacles. In order to reduce the interference caused by experimental chance on the experimental data, the PSO, the DWA, the traditional Morphine algorithm and ATCM were tested repeatedly in the same environment, and the optimal results of path planning for the four algorithms are shown in Fig. 16. Table IV displays a comparison of the performance metrics for each algorithm.

Table V. Comparison of performance metrics of different algorithms.

Algorithms	Optimal path length (m)	Running time (s)	Number of iterations
PSO	48.5	2.835	54
DWA	21.02	1.495	43
Traditional Morphine	22.265	1.432	19
ATCM	14.605	1.256	8

**Fig. 16.** Path planning results of the four algorithms with additional static obstacles. (a) PSO, (b) DWA, (c) traditional Morphine algorithm and (d) ATCM.

The simulation results and performance metrics show that all four algorithms are able to well complete the navigation from start to the specific target. However, while the other three algorithms have large turning angles in the process of obstacle avoidance, ATCM is not only able to avoid obstacles safely but also has a smoother trajectory. The optimal path is obtained through eight iterations of convergence, which is only 14.8% of the iterations in the particle swarm algorithm; the length of the optimal path obtained by ATCM is 14.605(m), which is 34.4% less than that obtained by the traditional Morphine algorithm. Moreover, ATCM takes the shortest time to complete the navigation, and the total global path time is 1.256 s. In short, ATCM shows advantages in respect of the three performance indicators as shown in Table V.

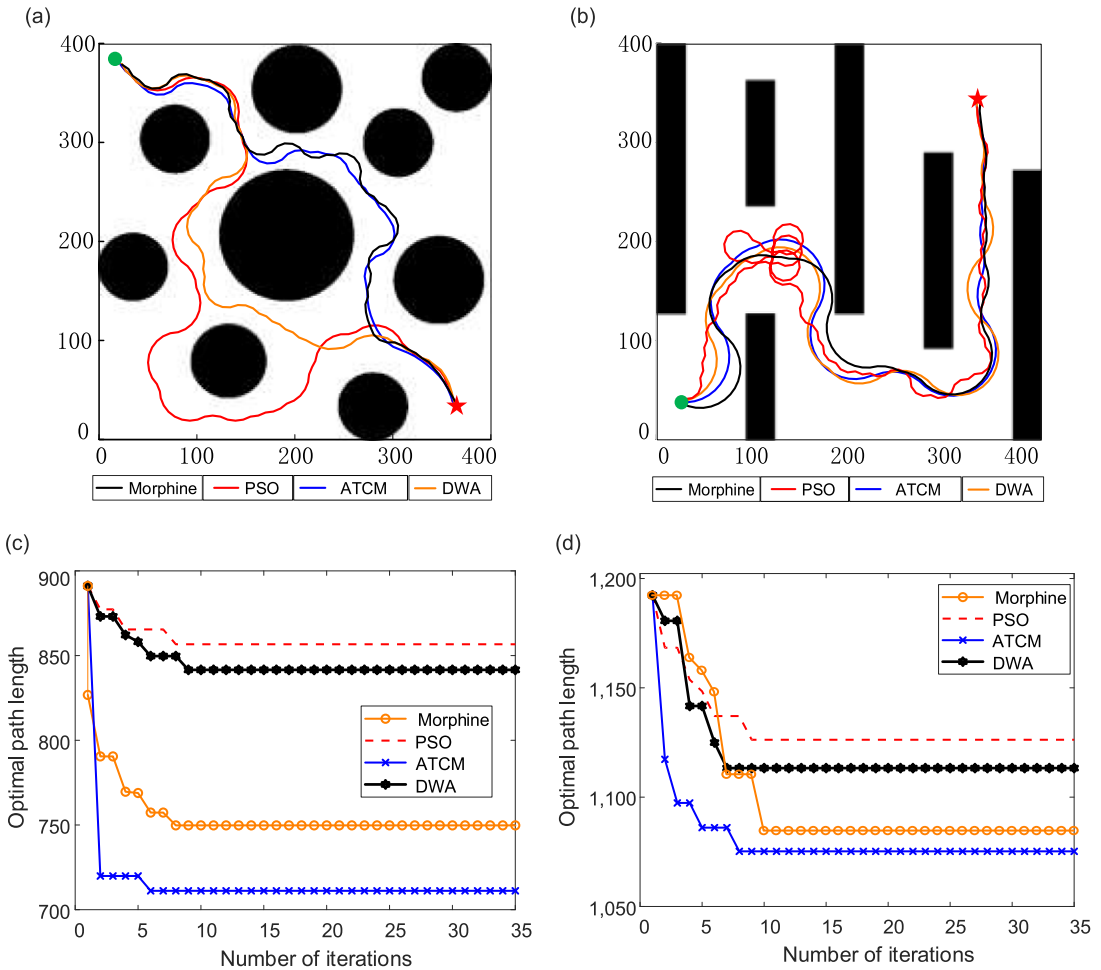


Fig. 17. Path planning results of the four algorithms in different geometric obstacle environments. (a) Circular obstacle environment. (b) Rectangular obstacle environment. (c) Evolutional path length on the number of iterations in a circular obstacle environment. (d) Evolutional path length on the number of iterations in a rectangular obstacle environment.

4.3.2. Performance of the four algorithms in different geometric obstacle environments.

Figure 17(a) and (b) depicts the path planning effects of the four algorithms under different geometric obstacle environments, with the green dot in the figure being the starting point and the red pentagram showing the end point, and the simulation experimental environment being a square area of 400×400 . The results demonstrate ATCM, based on the proposed ‘Data Clustering for Obstacle Detection’, can better identify and classify the types of obstacles in the environment, and plan an optimal path solution in the space unoccupied by obstacles. The blue curves in Fig. 17(a) and (b) are the optimal paths obtained by ATCM in different types of obstacle environments, and Fig. 17(c) and (d) illustrates the relationship between the iteration numbers and their optimal path lengths. Figure 17(c) shows that ATCM obtains the shortest path length at the smallest iteration number 6 in all algorithms; Fig. 17(d) shows that ATCM obtained the shortest path in the four algorithms, although DWA has a slightly smaller number of convergence iterations than ATCM.

5. Conclusions

We have developed a new navigation system, using an adaptive threshold clustering algorithm to detect the obstacles, a simple decision tree based on the dynamics of the detected obstacles to identify the obstacle types, a scenario-based heuristic algorithm to predict the potential collision and a simplified Morphin path planning algorithm to find the optimal path. The computing complexity of the proposed algorithm is improved, as it avoids the training process of machine learning model, thus implementing the full autonomy of a robot with a fast and effective navigation in an unknown environment. The eight scenarios are analysed and tested in simulation and on a physical robot. The experimental results demonstrate that the proposed navigation system enables a mobile robot to avoid any static and dynamic obstacles on the path effectively and efficiently, where the robot goes through. Comparing with other path planning techniques, ATCM obtained shortest length of navigation path. The modular design of the six-step navigation system for a mobile robot provides a framework of navigation system for further research and development.

To further improve the robustness of the navigation system, we can use fuzzy decision tree to classify the types of obstacles, which will be our future work. The study of scenario (g), where a robot is faster than a dynamic obstacle ahead of the robot in the same direction, could be applied for autonomous vehicles to avoid the potential rear-end accidents, which is a critical challenge in autonomous vehicles. Hence, further research on scenario (g), overtaking a dynamic obstacle to avoid the rear-end accident, the verification of the maximal speed of the robot in different cases in a real environment, as well as the improvement of the real-time performance of the navigation system, will be the future work.

Acknowledgement. This work was sponsored by National Natural Science Foundation of China (61903002).

References

- [1] A. A. Ahmed, T. Y. Abdalla and A. A. Abed, "Path planning of mobile robot by using modified optimized potential field method," *Int. J. Comput. Appl.* **113**(4), 6–10 (2015).
- [2] M. B. Alatise and G. P. Hancke, "A review on challenges of autonomous mobile robot and sensor fusion methods," *IEEE Access* (2020). doi: [10.1109/ACCESS.2020.2975643](https://doi.org/10.1109/ACCESS.2020.2975643).
- [3] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (slam): Part II," *IEEE Rob. Autom. Mag.* **13**(3), 108–117 (2006).
- [4] F. Bonin-Font, A. Ortiz and G. Oliver, "Visual navigation for mobile robots: A survey," *J. Intell. Rob. Syst.* **53**, Article Number: 263 (2008). doi: [10.1007/s10846-008-9235-4](https://doi.org/10.1007/s10846-008-9235-4).
- [5] M. Brand, M. Masuda, N. Wehner and X. Yu, "Ant Colony Optimization Algorithm for Robot Path Planning," *International Conference On Computer Design and Applications*, vol. **3**, Qinhuangdao, China (2010).
- [6] D. Britt, "Robots are key to future space exploration," *Robots Vs Astronauts*. Blog. Accessed 22/07/2019.
- [7] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid and J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Rob.* **32**(6), 1309–1332 (2016).
- [8] K.-H. Chen and W.-H. Tsai, "Vision-based obstacle detection and avoidance for autonomous land vehicle navigation in outdoor roads," *Autom. Const.* **10**(1), 1–25 (2000).
- [9] M. Chen, Y. J. Wu and H. He, "A Comprehensive Obstacle Avoidance System of Mobile Robots Using an Adaptive Threshold Clustering and the Morphin Algorithm," In: *Advances in Computational Intelligence Systems, UKCI2018* (Springer Nature America, Inc., 2018) pp. 312–324.
- [10] D. W. Cho and J. H. Lim, "A new certainty grid-based mapping and navigation system for an autonomous mobile robot," *Int. J. Adv. Manuf. Technol.* **10**, 139–148 (1995). doi: [10.1007/BF01179282](https://doi.org/10.1007/BF01179282).
- [11] M. Dirik, A. F. Kocamaz and O. Castillo, "Global path planning and path-following for wheeled mobile robot using a novel control structure based on a vision sensor" *Int. J. Fuzzy Syst* **22**, 1880–1891 (2020). doi: [10.1007/s40815-020-00888-9](https://doi.org/10.1007/s40815-020-00888-9).
- [12] M. Dirik, O. Castillo, A. F. Kocamaz and A. Fatih, "Visual-servoing based global path planning using interval type-2 fuzzy logic control," *Axioms* **8**(2), Article Number: 58 (2019). doi: [10.3390/axioms8020058](https://doi.org/10.3390/axioms8020058).
- [13] D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Rob. Autom. Mag.* **4**(1), 23–33 (1997). doi: [10.1109/100.580977](https://doi.org/10.1109/100.580977).
- [14] M. S. Ganeshmurthy and G. R. Suresh, "Path Planning Algorithm for Autonomous Mobile Robot in Dynamic Environment," *The 3rd International Conference on Signal Processing, Communication and Networking (ICSCN)*, Chennai, India (2015) pp. 1–6.
- [15] B. Gardiner, S. Coleman, T. M. McGinnity and H. He, "Robot control code generation by task demonstration in dynamic environment," *Rob. Auto. Syst.* **60**(12), 1508–1519 (2012).

- [16] H. He, T. M. McGinnity, S. Coleman and B. Gardiner, "Linguistic decision making for robot route learning," *IEEE Trans. Neural Networks Learn. Syst.* **25**(1), 203–215 (2014).
- [17] N. Herojit Singh and T. Khelchandra, "Mobile robot navigation using fuzzy-GA approaches along with three path concept," *Iranian J. Sci. Technol. Trans. Electr. Eng.* **43**, 277–294 (2019). doi: [10.1007/s40998-018-0112-2](https://doi.org/10.1007/s40998-018-0112-2).
- [18] R. Huang, H. Laing and J. e. a. Chen, "Lidar based dynamic obstacle detection, tracking and recognition method for driverless cars," *Robot* **38**(4), 437–443 (2016). doi: [10.13973/j.cnki.robot.2016.0437](https://doi.org/10.13973/j.cnki.robot.2016.0437).
- [19] M. Jiang, X. Fan, Z. Pei, J. Jiang, Y. Hu and Q. Wang, "Robot path planning method based on improved genetic algorithm," *Sens. Trans.* **166**(3), 255–260 (2014).
- [20] S. Joshi, Indian scientists are working on a robot to patrol the borders. Blog, 03 May 2019. Accessed on 22/07/2019.
- [21] J. Kocić, N. Jovičić and V. Drndarević, "Sensors and Sensor Fusion in Autonomous Vehicles," *26th Telecommunications Forum (TELFOR)*, Belgrade, Serbia (2018).
- [22] G. Kovacs, Y. Kunii, T. Maeda and H. Hashimoto, "Saliency and spatial information-based landmark selection for mobile robot navigation in natural environments" *Adv. Rob.* **33**(10), 520–535 (2019).
- [23] J. Liu, Q. Yan and Z. Tang, "Simulation research on obstacle avoidance planning for mobile robot based on laser radar," *Comput. Eng.* **41**(4), 306–310 (2015).
- [24] J. Luo, C. Liu and F. Liu, "Piloting-following formation and obstacle avoidance control of multiple mobile robots," *CAAI Trans. Intell. Syst.* **12**(2), 1–10 (2017).
- [25] L. Matthies, E. Gat, R. Harrison, B. Wilcox, R. Volpe and T. Litwin, "Mars Microrover Navigation: Performance Evaluation and Enhancement," *Proceedings of IEEE International Conference of Intelligent Robots and Systems (IROS)* (1995) pp. 433–440.
- [26] J. C. Mohanta and A. Anupam Keshari, "A knowledge based fuzzy-probabilistic roadmap method for mobile robot navigation," *Appl. Soft Comput. J.* **79**, 391–409 (2019). doi: [10.1016/j.asoc.2019.03.055](https://doi.org/10.1016/j.asoc.2019.03.055).
- [27] O. Motlagh, S. Tang, N. Ismail and A. R. Ramli, "An expert fuzzy cognitive map for reactive navigation of mobile robots," *Fuzzy Sets Syst.* **201**, 105–121(2012). doi: [10.1016/j.fss.2011.12.013](https://doi.org/10.1016/j.fss.2011.12.013).
- [28] F. K. A. Muslim, S. Tang, W. Khaksar, N. Zulkifli and S. A. Ahmad, "A review on motion planning and obstacle avoidance approaches in dynamic environments," In: *Advances in Robotics & Automation*, vol. **4** (2015). doi: [10.4172/2168-9695.1000134](https://doi.org/10.4172/2168-9695.1000134).
- [29] U. Orozco-Rosas, K. Picos and O. Montiel, "Hybrid path planning algorithm based on membrane pseudo-bacterial potential field for autonomous mobile robots," *IEEE Access* **7**, 156787–156803 (2019). doi: [10.1109/ACCESS.2019.2949835](https://doi.org/10.1109/ACCESS.2019.2949835).
- [30] H. Ouarda, "Neural path planning for mobile robots," *Int. J. Syst. Appl. Eng. Dev.* **5**(3), 367–376 (2011).
- [31] R. Rashid, N. Perumal, I. Elamvazuthi, M. K. Tageldeen, M. K. A. Ahamed Khan and S. Parasuraman, "Mobile Robot Path Planning Using Ant Colony Optimization," *The 2nd IEEE International Symposium on Robotics and Manufacturing Automation (ROMA)*, Ipoh, Malaysia (2016) pp. 1–6.
- [32] D. Ray, R. Das, B. Sebastian, B. Roy and S. Majumder, "Design and Analysis Towards Successful Development of a Tele-Operated Mobile Robot for Underground Coal Mines," In: *CAD/CAM, Robotics and Factories of the Future, Lecture Notes in Mechanical Engineering* (Springer, New Delhi, 2016).
- [33] Y. Rubio, K. Picos, U. Orozco-Rosas, C. Sepúlveda, E. Ballinas, O. Montiel, O. Castillo and R. Sepúlveda, "Path Following Fuzzy System for a Nonholonomic Mobile Robot Based on Frontal Camera Information," In: *Fuzzy Logic Augmentation of Neural and Optimization Algorithms: Theoretical Aspects and Real Applications, Studies in Computational Intelligence* (O. Castillo, P. Melin and J. Kacprzyk, eds), vol. 749 (Springer, Cham, 2018). doi: [10.1007/978-3-319-71008-2_18](https://doi.org/10.1007/978-3-319-71008-2_18).
- [34] M. A. Sanchez, O. Castillo and J. R. Castro, "Generalized Type-2 Fuzzy Systems for controlling a mobile robot and a performance comparison with Interval Type-2 and Type-1 Fuzzy Systems," *Expert Syst. Appl.* **42**(14), 5904–5914 (2015).
- [35] R. Simmons, L. Henriksen and L. Chrisman, "Obstacle Avoidance and Safeguarding for a Lunar Rover," *Proceedings of AIAA Forum on Advanced Development in Space Robotics* (1996).
- [36] L. Steccanella, D. D. Bloisi, A. Castellini and A. Farinelli, "Waterline and obstacle detection in images from low-cost autonomous boats for environmental monitoring," *Rob. Auto. Syst.* **124**, Article Number: 103346 (2020). doi: [10.1016/j.robot.2019.103346](https://doi.org/10.1016/j.robot.2019.103346).
- [37] D. Tang, J. Yang and X. Cai, "Grid Task Scheduling Strategy Based on Differential Evolution-Shuffled Frog Leaping Algorithm," *The IEEE International Conference on Computer Science and Service System (CSSS2012)*, Nanjing, China (2012) pp. 1702–1708.
- [38] C. N. Viet and I. Marshall, "Vision-based Obstacle Avoidance for a Small Low-Cos Robot," *International Conference on Informatics in Control, Automation, and Robotics (ICINCO)*, Angers, France (2007).
- [39] X. F. Wan, H. W. and B. Zheng, "Robot path planning method based on improved ant colony algorithm and Morphing algorithm," *Sci. Technol. Rev.* **33**(3), 84–89 (2015).
- [40] J. L. Wang, J. Zhou and H. Gao, "Obstacle avoidance method for mobile robots based on the identification of local environment shape features," *Inf. Control* **44**(1), 91–98 (2015).
- [41] M. Wang, Y. Fan, X. Wang and C. Dong, "Design of infrared FPA detector simulator," *Laser Infrared* **46**(12), 1481–1485 (2016).
- [42] Q. Wang and J. Zhou, "A geomagnetic gradient bionic navigation method with parallel proximity," *J. Northwest. Polytech. Univ.* **36**(4), 611–617 (2018).
- [43] Z. Wang, X. Cui and C. Hou, "Analysis and countermeasures to the problem of ultrasonic sensor receives the ultrasonic signal asymmetric," *Chin. J. Sens. Actuators* **28**(1), 81–85 (2015).
- [44] T. Weerakoon, K. Ishii and A. Nassiraei, "An artificial potential field based mobile robot navigation method to prevent from deadlock," *J. Artif. Intell. Soft Comput. Res.* **5**(3), 189–203 (2015).

- [45] P. Wu, Y. Cao, Y. He and D. Li, "Vision-Based Robot Path Planning with Deep Learning," In: *Computer Vision Systems. ICVS 2017, Lecture Notes in Computer Science* (M. Liu, H. Chen, Vincze M. eds), vol. 10528 (Springer, Cham, 2017). https://doi.org/10.1007/978-3-319-68345-4_9.
- [46] Y. Wu, Research on Hybrid Path Planning for Mobile Robots in Indoor Environment *Thesis* (Anhui Polytechnic University, Wuhu, China, 2018).
- [47] Y. Xin, H. Liang, T. Mei, R. Huang, M. Du, Z. Wang, J. Chen and P. Zhao, "Dynamic obstacle detection and representation approach for unmanned vehicles based on laser sensor," *Robot* **36**(6), 654–661 (2014). doi: [10.13973/j.cnki.robot.2014.0654](https://doi.org/10.13973/j.cnki.robot.2014.0654).
- [48] Y. Yang, F. Han and Z. e. a. Cao, "Laser sensor based dynamic fitting strategy for obstacle avoidance control and simulation," *J. Syst. Simul.* **25**(4), 118–122 (2013).
- [49] D. Zhang, W. Li, H. Wu and Y. Chen, "Mobile robot adaptive navigation in dynamic scenarios based on learning mechanism," *Inf. Control* **45**(5), 521–529 (2016). doi: [10.13976/j.cnki.xk.2016.0521](https://doi.org/10.13976/j.cnki.xk.2016.0521).
- [50] J. Zhang, H. Hu and Y. Wan, "Dynamic Path Planning Algorithm Based on an Optimization Model," In: *Signal and Information Processing, Networking and Computers. ICSINC 2018, Lecture Notes in Electrical Engineering* (S. Sun, M. Fu and L. Xu, eds), vol. 550 (Springer, Singapore, 2019) pp. 105–114.
- [51] Q. Zhang, P. Wang and Z. Chen, "Velocity space based concurrent obstacle avoidance and trajectory tracking for mobile robots," *Control Decis.* **32**(2), 358–362 (2017). doi: [10.13195/j.kzyjc.2015.1376](https://doi.org/10.13195/j.kzyjc.2015.1376).
- [52] Q. Zhang, X. Yang and T. Liu, "Design of a smart visual sensor based on fast template matching," *Chin. J. Sens. Actuators* **26**(8), 1039–1044 (2013).
- [53] Y. Zhang, F. Du and Y. Luo, "A local path planning algorithm based on improved Morphin search tree," *Electr. Opt. Control* **23**(7), 15–19 (2016).
- [54] Y. Zhang, J. Xu, L. Chen and Z. Liu, "Design of terrain recognition system based on laser distance sensor," *Laser Infrared* **46**(3), 265–270 (2016).
- [55] X. Zhong, X. Peng and J. Zhou, "Detection of Moving Obstacles for Mobile Robot Using Laser Sensor," *The 20th IEEE Chinese Control Conference (CCC)*, Yantai, China (2011).
- [56] J. Zhu, Y. Zhou, X. Wang, W. Han and L. Ma, "Grid map merging approach based on image registration," *Acta Automatica Sinica* **41**(2), 285–294 (2015).
- [57] C. Zhu-Ge, Z. Tang and Z. Shi, "Local path planning algorithm for UGV based on multilayer Morphin search tree," *Robot* **36**(4), 491–497 (2014). doi: [10.13973/j.cnki.robot.2014.0491](https://doi.org/10.13973/j.cnki.robot.2014.0491).