Received: 24 April 2021 | Revised: 2 December 2021 | Accepted: 5 December 2021

DOI: 10.1002/rob.22055

RESEARCH NOTE

WILEY

Local path planning for autonomous mobile robots by integrating modified dynamic-window approach and improved follow the gap method

Tagor Hossain[1] | Habibullah Habibullah[1] | Rafiqul Islam[1] | Ricardo V. Padilla[2]

[1]Faculty of UniSA STEM, University of South Australia, Mawson Lakes, South Australia, Australia

[2]SESE, Southern Cross University, Lismore, New South Wales, Australia

Correspondence
Tagor Hossain, Faculty of UniSA STEM, University of South Australia, Mawson Lakes, SA 5095, Australia.
Email: md_tagor.hossain@mymail.unisa.edu.au

Funding information
University of South Australia

Abstract

Mobile robots need to automatically generate a safe, goal-oriented, and fast collision-free trajectory in real-time during the movement in an indoor/outdoor environment. A planned trajectory must be adaptable and drivable with environmental changes where various static and moving obstacles may be present. The ultimate goal of a robot is to reach the destination without hitting any obstacles, therefore, a reactive local path planning algorithm is needed. In this paper, a novel local algorithm is proposed by integrating dynamic window approach (DWA) and improved follow the gap method (IFGM) to generate a collision-free trajectory for a mobile robot which is capable to avoid any moving obstacles presenting in the surrounding environment. In this proposed method, first, a safety distance is maintained according to the relative position of obstacles and the robot. Moreover, find a feasible gap to direct the robot toward the desired goal. Besides, the heading angle is calculated to change the direction of the robot for avoiding collision with nearby obstacles. After that, calculate the appropriate velocity for the robot. Finally, a robust, safe, and goal-directed trajectory is generated which does not suffer from global convergence and local minima problems. The performance and effectiveness of this proposed algorithm are evaluated by experimental results.

KEYWORDS
autonomous mobile robot, collision avoidance, dynamic obstacles, IFGM-DWA algorithm, local path planning

1 | INTRODUCTION

Autonomous robots are an important part of today's world. Mobile robots can perform many tasks on a frequent basis by using the sensorial information from the surrounding environment and the goal coordinates. Path planning is an inevitable issue for autonomous mobile robots which makes collision-free trajectories for robots to travel from initial position to goal position without any human intervention in various environments where both static and dynamic obstacles can be present (Erke et al., 2020). Therefore, it draws growing attention from researchers over the last two decades (da Costa Barros & Nascimento, 2021; Lavalle, 2006). It depends on precisely sense static or moving obstacles and avoid them during the movement of robots. Real-time obstacle detection and avoidance system deal with different challenging situations in real environmental scenarios. There are two types of path planning strategies depending on the static and dynamic environments, one of them is a global path planning strategy and the other one is a local or reactive path planning strategy (da Costa Barros & Nascimento, 2021; Dolgov et al., 2009). The global path planning strategy uses prior information, where a map, a goal position, and stationary obstacles positions are known to the robot to generate a collision-free trajectory. This type of method is also called static path

planning because obstacles are assumed to be static and the given map is not updated dynamically by using the sensory information (Chakravarthy & Ghose, 1998). Many studies have been involved to improve the static path planning algorithms, among them, probabilistic roadmaps (PRMs; Kavraki et al., 1996), *A** algorithm (Duchoň et al., 2014), rapidly exploring random trees (RRTs; Lavalle, 1998), voronoi diagrams (Aurenhammer, 1991), cell decomposition methods (Siegwart et al., 2011), and visibility graphs (Lozano-Pérez & Wesley, 1979) are popular collision avoidance global path planning algorithms. These algorithms generate an accurate trajectory for the robots when the surrounding environments and obstacles are stationary. However, the global path planning methods take more execution time and suffer to create an accurate trajectory when obstacles are moving and scenarios of the surrounding environments are changing (Ozdemir & Sezer, 2017). Therefore, it is highly essential to improve a realtime reactive algorithm to avoid collision with dynamic obstacles with the changes of the environment scenarios that ensure the safe trajectory for mobile robots. The primary objective of this study is to generate an optimum collision-free short and smooth trajectory for autonomous robots in any dynamic environment where the starting position and the goal position is known. Many scholars have paid attention to developing real-time reactive/local path planning algorithms for avoiding unexpected stationery and moving obstacles. A literature survey is presented in a paper (Patle et al., 2019) where different classical and recent soft-computing algorithms such as fuzzy logic, nature-inspired optimization algorithm, and neural network are applied for obstacle avoidance and mobile robot navigation. In Ahmadzadeh and Ghanavati (2012), the particle swarm optimization (PSO) algorithm for multiple autonomous robots navigation is presented, whereas in Pandey and Parhi (2017) authors proposed a hybrid fuzzy controller by integrating a wind drive optimization method for mobile robots navigation and collision avoidance path planning in different stationary and dynamic environments. A PSO-tuned feedforward neural network (FNN) is designed and experimented in Pandey et al. (2020) for mobile robot navigation to minimize the path length between the start to the goal point in various environments. Multiple adaptive neuro-fuzzy inference system (MANFIS) architecture-based mobile robot navigation is proposed by Pandey et al. (2019), where different static and moving obstacles are present in the two-dimensional environments. There are other several techniques have been proposed to create an obstacle avoidance local path for mobile robots. Bug algorithms are the earliest version of obstacle avoidance reactive path planning methods (Lumelsky & Stepanov, 1987). In the bug algorithms, the robot follows the shortest path towards the goal until it finds any obstacles in its trajectory. However, sometimes very long and unsafe trajectory is generated by the bug algorithms (Choset et al., 2005). Another common obstacle avoidance local path planning algorithm is an artificial potential field (APF) algorithm (Khatib, 1985). In the APF method, the goal position of the robot is defined by an attractive potential field, whereas the obstacles are represented by artificial repulsive potential fields. The attractive potential field generates forces to push the robot toward the goal, while the repulsive potential generates forces of the robot to push away from the obstacles (Rostami et al., 2019). By this way, the APF method generates trajectory for mobile robots. However, the APF method has some drawbacks, among them, local minima is the most dangerous problem of this method (Min et al., 2015). Some methods try to find out the solution for the local minima problem, but none of them can generate a proper trajectory by solving local minima issue of the APF method (Azmi & Ito, 2020; Lazarowska, 2019). Vector field histogram (VFH) is another important method for generating a collision-free trajectory (Borenstein & Koren, 1991). In the VFH method, a histogram grid is used to show the environment where the robot travels. However, nonholonomic constraints are not considered in this method and sometimes it choose the wrong sector that can drive the robot in the wrong direction. There are some studies that have been done to improve the performance of the VFH algorithm to generate a collision-free trajectory (Babinec et al., 2018; Kazem et al., 2010; Ulrich & Borenstein, 2000). A novel follow the gap method (FGM) is proposed in Sezer and Gokasan (2012), which is another popular safety-focused obstacle avoidance path planning algorithm. This method calculates the largest gap's center angle between obstacles and determines an appropriate heading angle to drive the robot toward the destination without any collision. However, the FGM calculates only the desired heading angle of the robot, and the velocity of the robot is considered as a constant. Therefore, sometimes the robots fail to control motion in narrow spaces and this is the main issue of the FGM (Demir & Sezer, 2017). Moreover, this method sometimes generates a longer trajectory as the robot goes through the largest gap between obstacles (Ozdemir & Sezer, 2017). To solve this problems authors in Demir and Sezer (2017) proposed an improved FGM which is able to generate a shorter trajectory. Another method is proposed in Zohaib et al. (2014) by combining intelligent bug algorithm with the FGM to solve a dead-end scenario of the trajectory. Dynamic-window approach (DWA) is a widely used obstacle avoidance path planning algorithm that determines control commands for mobile robots directly from the velocity space by maximizing of robot's objective function (Fox et al., 1997). The objective function calculates optimum velocity pairs for mobile robots according to a minimum distance from obstacles, final heading angle, and speed values of robots. However, the DWA has local minima and the global convergence problem (Berti et al., 2008; Ozdemir & Sezer, 2017). To solve the local minima problem a global-DWA (Brock & Khatib, 1999) is proposed where the connectivity information of the free space is used to determine a motion command for mobile robots. A reduced-DWA is proposed by Arras et al. (2002) to generate a trajectory with less computational power. However, it is not able to select an appropriate velocity command when the robot heading is far away from the goal. To solve the global convergence problem of the DWA, a convergent DWA is proposed in Ogren and Leonard (2005) by integrating the DWA, model predictive control (MPC) method, and control lyapunov function (CLF) to create a trajectory for mobile robots. This method guarantees convergence but requires more processing time. A bionimetical DWA is proposed by Ballesteros et al. (2017), where human and robot commands are combined to navigation for collaborative control. An image-based DWA is presented in Kang et al. (2015) for dynamic obstacles avoidance by using the visual information

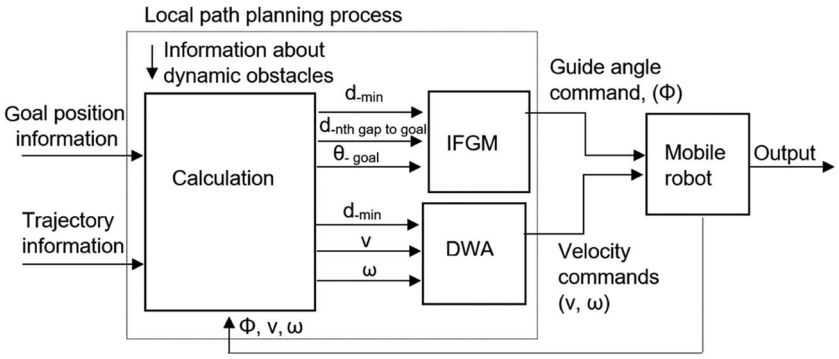

FIGURE 1 Architecture overview of the proposed algorithm

of the trajectory and the environment. Authors in Ji et al. (2021) proposed a method by integrating A^* algorithm and the DWA by considering complex environmental information for local path planning. There are some other methods, VFH (Borenstein & Koren, 1989), obstacle restriction method (ORM; Minguez, 2005), nearness diagram (ND; Minguez & Montano, 2000), and curvature-velocity method (CVM; Simmons, 1996) are used for reactive path planning.

The contribution of this paper is to develop a local path planning algorithm for generating a shorter, safe, and drivable collision-free path where different moving obstacles are present in dynamic environments. The proposed algorithm is capable to generate safe path by avoiding any nearby unexpected dynamic obstacles. This method calculates all the gaps among obstacles and finds the feasible gap from the IFGM, which makes the trajectory shorter and safer for the robot. Then it determines the final heading angle of the robot to move through the feasible gap towards the destination. Subsequently, the DWA continuously calculates the proper velocity while any unexpected moving obstacles are coming, then the robot reaches the admissible velocity. Therefore, the robot can stop before any collision with obstacles. Finally, the most important command is calculated for the robot from the objective function. In the objective function, the optimal trajectory is generated where the robot can move forward and even backwards depending on the situations. The path planning process of the proposed algorithm is shown in Figure 1.

The remainder of this paper is organized as follows: kinematic structure of a mobile robot is described in Section 2, whereas Section 3 broadly explains the proposed (IFGM-DWA) algorithm based collision-free trajectory generation. Section 4 describes the performance of the proposed method from different test scenarios and verify experimental results. Section 5 concludes the paper with possible future research directions.

2 | KINEMATIC MODEL OF A ROBOT

Kinematic structure of a wheeled mobile robot is shown in Figure 3. A car-like mobile robot should be considered while a robot is operated with great velocities or masses. However, for the path planning of a mobile robot with lower velocities or masses the kinematic structure usually suffices (Ghita & Kloetzer, 2012; Sezer & Gokasan, 2012; Siciliano, 2008; Tzafestas, 2014) of a car-like robot.

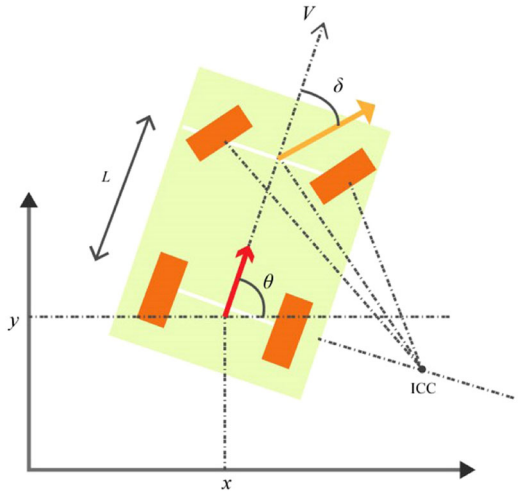

FIGURE 2 Mobile robot's kinematic structure (Ghita & Kloetzer, 2012) [Color figure can be viewed at wileyonlinelibrary.com]

2.1 | Ackerman kinematic model

Figure 2 represents the Ackerman kinematic model. The kinematic equations of the robot can illustrated in Equations (1), (2), and (3) as follows:

$$\dot{x} = V\cos(\theta), \tag{1}$$

$$\dot{y} = V\sin(\theta), \tag{2}$$

$$\dot{\theta} = \frac{V}{L}\tan(\delta), \tag{3}$$

where the cartesian coordinates (x, y) situated at the midpoint of the wheel axis and the velocity of the robot is V. ICC represents the instantaneous center of curveture. The robot's orientation angle and the steering angle are represented by θ and δ, respectively.

2.2 | Differential drive kinematic model

Considering the parameters from Figure 3, where R is the radius of the wheel, CP represents the center point, and $2L$ represents the

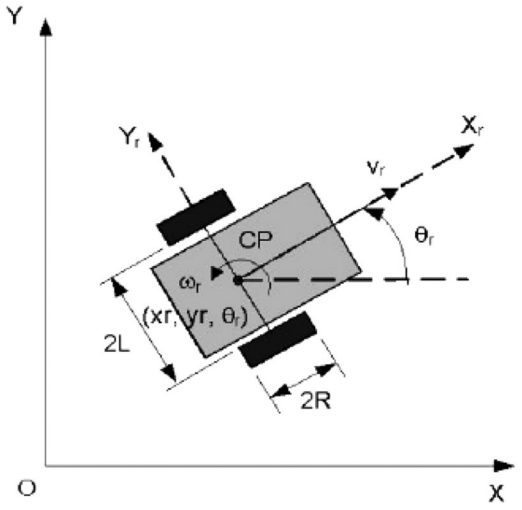

FIGURE 3 Mobile robot's kinematic structure (Filipescu et al., 2011)

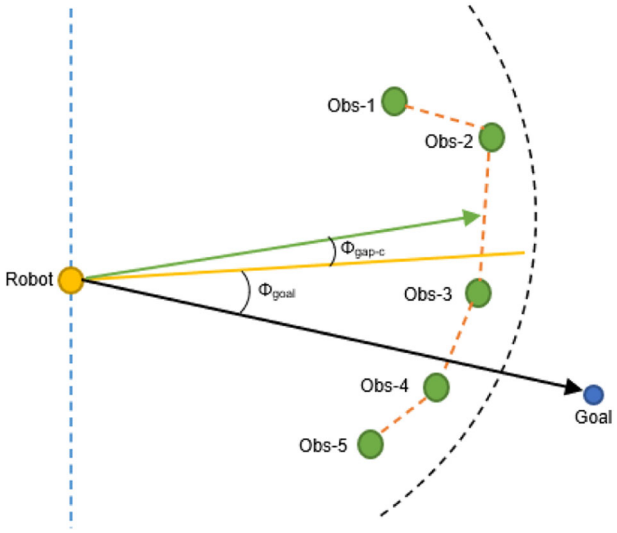

FIGURE 4 Largest gap calculation to generate a safe trajectory for a robot [Color figure can be viewed at wileyonlinelibrary.com]

distance between two wheel of the robot. The position of the wheeled robot can be considered as (x_r, y_r, θ_r) and the robot's kinematic equations can be written in Equations (4), (5), and (6):

$$\dot{x_r} = V_r \cos(\theta_r), \tag{4}$$

$$\dot{y_r} = V_r \sin(\theta_r), \tag{5}$$

$$\dot{\theta_r} = \omega_r, \tag{6}$$

where the cartesian coordinates (x_r, y_r, θ_r) shows the position on the x axis, y axis, and orientation angle of the mobile robot, respectively. The linear velocity of the robot is V_r, whereas the angular velocity is ω_r.

3 | IFGM-DWA COLLISION-FREE TRAJECTORY GENERATION

The traditional DWA considers motion dynamics of a robot to find an admissible velocity space under a specific time interval, after that this algorithm using its objective function to find the most appropriate linear and angular velocity pairs to control the motion of mobile robots. The traditional DWA successfully guide a robot to final position while static obstacles are present in the environment. However, this algorithm is suffering to avoid dynamic obstacles. Especially, the DWA is unable to avoid moving obstacles that are coming from the front because it has local minima and forward motion problems. On the other hand, the FGM is a geometric obstacle avoidance algorithm which calculates the gap array to determine the heading angle of a robot and does not consider the rotational and translation velocity. In this paper, the DWA combining with the improved FGM to avoid dynamic obstacles by considering motion dynamics and desired heading angle of a robot. This section describes the path planning algorithm for mobile robots. First, the FGM is presented. Then, the traditional DWA is described. Finally, the proposed algorithm based trajectory generation is presented.

3.1 | Traditional FGM

The traditional FGM is the most effective geometric obstacle avoidance technique which determines the heading angle of a robot by considering safety coefficient. The FGM calculates the final heading angle by using the largest gap center angle, a minimum distance to the closest obstacle, and the goal angle of the robot as shown in Figure 4. There are three stages in this method to find the final guide angle of the robot. First, the radius of an obstacle is enlarged with the radius of the robot to determine free spaces. Then, all the gaps ahead of the robot are calculated by using obstacles border angle values, and then the largest gap between obstacles are determined from border angle values (Sezer & Gokasan, 2012). A gap center angle is determined in the second stage from the robot's heading and the largest gap midpoint. After that, the goal angle is calculated from the goal coordinates and the heading of the robot. In the final stage, it finds a final guide angle of the robot by using the Equation (7):

$$\phi_{\text{final}} = \frac{\frac{\alpha}{d_{\min}}\phi_{\text{gap-c}} + \phi_{\text{goal}}}{\frac{\alpha}{d_{\min}} + 1}. \tag{7}$$

where ϕ_{final}, $\phi_{\text{gap-c}}$, and ϕ_{goal} are the final heading angle, the gap center angle, and the goal angle, respectively. d_{min} is the minimum distance between robot and obstacle and α is the weight coefficient for the gap. After calculating the final heading angle, the mobile robot can create a safe trajectory by moving through the largest gap.

3.2 | Traditional dynamic window approach (DWA)

The traditional DWA is searching the velocity space for commands to control the mobile robots and this algorithm considers the robot dynamics into account (Fox et al., 1997). The trajectories are shown in Figure 5.

There are three steps in this method to search the velocity space by maximizing the objective function. In the first step, the unreachable velocities are eliminated under the dynamic constraints of the robot which are coming from the robot's acceleration. Moreover, the DWA considers only reachable velocities that are safer regarding to the obstacles Fox et al. (1997). The second step removes all speed pairs from the remaining velocities of the robot, that are unable to stop the robot before colliding with obstacles. In the final step, an admissible velocity set is evaluated by maximizing the objective function. This method predicts the trajectory of the robot by evaluating each of the velocity pair candidates with respect to the linear velocity, the final heading angle, and the safe distance to obstacles is presented in Equation (8):

$$G(v, \omega) = \sigma(\alpha \cdot heading(v, \omega) + \beta \cdot dist(v, \omega) + \gamma \cdot vel(v, \omega)), \tag{8}$$

where (α, β, γ) are the weight constants and σ is the smoothing operator of the DWA. In the objective function, heading (v, ω) represents the angle between the robot heading and the goal coordinates, whereas the aim of the dist (v, ω) provides the safe navigation. It measures and calculates the closest distance between the robot and the closest obstacle on the path from sampled velocity pair (v, ω). Finally, the velocity (v, ω) calculates the highest value of the linear velocity from the objective function for the next motion command.

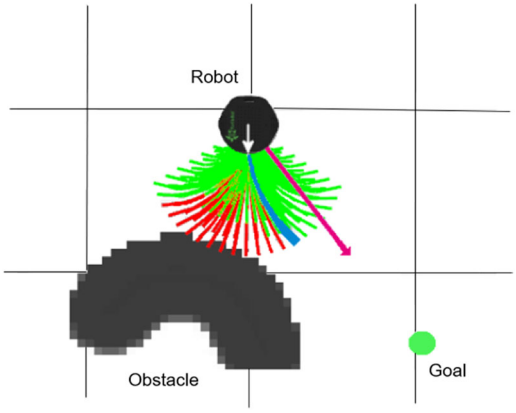

FIGURE 5 Admissible trajectories are shown by green color, whereas the red indicates the inadmissible trajectories. The magenda arrow and the blue show the heading angle and the best trajectory, respectively (Ozdemir & Sezer, 2017) [Color figure can be viewed at wileyonlinelibrary.com]

3.3 | Proposed IFGM-DWA method

The robot and dynamic obstacles are assumed to be circular objects to consider all physical boundaries and they have a fixed radius shown in Figure 6. Position and radius of the robot are presented by $(x_{rob}, y_{rob}, r_{rob})$, whereas nth obstacles central position and radius are given by $(x_{obn}, y_{obn}, r_{obn})$. The proposed algorithm is working cooperatively with global planner where the goal position is provided by the global planner. The robot does not have any prior information about the trajectory and the dynamic obstacles. The coordinates of moving obstacles are continuously changing, for this reason, their position can not be defined previously. Therefore, the robot needs to calculate the next step in every moment by considering all the obstacles boundaries and goal point, based on the robot's current location. The proposed IFGM-DWA algorithm has four stages, and these stages are described in the following four sections. In the first section, distance calculation between robot and dynamic obstacles is explained. Guide angle selection for the robot by using improved FGM is described in the second section. Calculation of the proper velocity is touched in the third section. Finally, an objective function of the proposed algorithm is explained in details.

3.3.1 | Distance calculation between robot and dynamic obstacles

The distance measurement between the robot and obstacles is very important for determination the heading angle and the velocity of the robot during the movement. A circular robot is considered as a point robot in the cartesian space for calculating the distance between robot and obstacle boundary and it is denoted by d and shown in Figure 6. By considering robot and obstacles geometry the Pythagorean theorem is applied to calculated (Sezer & Gokasan, 2012) as illustrated in Equation (9):

$$d = \sqrt{(x_{obn} - x_{rob})^2 + (y_{obn} - y_{rob})^2}. \tag{9}$$

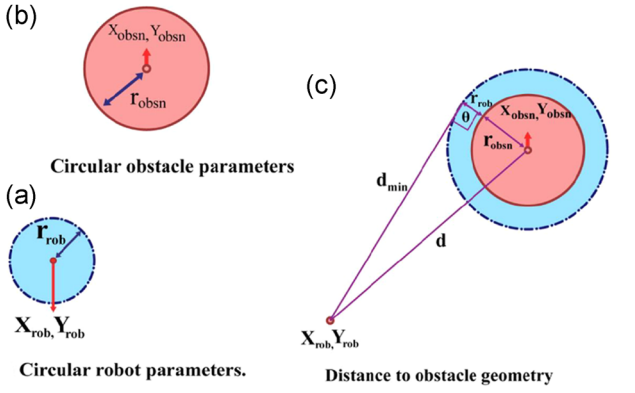

FIGURE 6 Parameters and geometry of the robot and the obstacle to calculate the distance between them. (a) Circular robot parameters, (b) circular obstacle parameters, (c) distance to obstacle parameters [Color figure can be viewed at wileyonlinelibrary.com]

The distance between the closest touching point of the circular robot and a circular obstacle is defined by d_{min} and illustrated in Equation (10):

$$d_{min} = \sqrt{(x_{obn} - x_{rob})^2 + (y_{obn} - y_{rob})^2 - (r_{obn} + r_{rob})^2}. \quad (10)$$

The collision will happen if

$$d_{min} < (r_{rob} + r_{obn}), \quad (11)$$

the minimum distance (d_{min}) is less than the summation of the robot and obstacle radius. After subtracting the robot radius and obstacle radius, a safe distance d_{safe} is calculated between the robot and obstacles. Equation (12) illustrates the safe distance:

$$d_{safe} = \frac{\left(v_{max}^2(t)\right)}{2\dot{v}_{max}(t)}, \quad (12)$$

where $v_{max}(t)$ is the maximum velocity and $\dot{v}_{max}(t)$ is the maximum acceleration of the robot.

3.3.2 | Find the guide angle with the improved FGM

A guide angle of a robot is a composition of the gap angle and the goal angle and it directs the mobile robot to the destination. The FGM is constructing gap arrays to calculate final heading angle (ϕ_{final}) for the robot and ϕ_{final} is derived in Equation (7). In the FGM, a robot selects the largest gap center during the movement, thereafter the robot suffers path length problem and it can be unstable if the size of the gaps is similar (Demir & Sezer, 2017). A feasible gap is chosen according to a utility function in the improved FGM and it eliminates the FGM drawbacks. The utility function for each gap is derived in Equation (13) which selects a new feasible gap for mobile robots. There are two variables are acting in the utility function; the size of gaps $d_{n\text{-}gap}$ and the angle δ_n between goal point and gap center as shown in Figure 7:

$$U_f = m_1 d_{n\text{-}gap} + m_2(\pi - \delta_n), \quad (13)$$

where $d_{n\text{-}gap}$ and δ_n are the size of the gaps and the angle between goal point, respectively. nth is the number of gap center. m_1 and m_2 are the weight coefficient for the size of the gaps and angle, respectively.

The gap angle δ_n between the goal point and the nth number of gap center is illustrated in Equation (14):

$$\delta_n = |\alpha_n - \beta|. \quad (14)$$

A feasible gap is calculated for the robot by considering size of gaps as well as its heading angle to the goal coordinates but the traditional FGM determines the largest gap only depending on the size of the gaps. This means that the improved FGM creates a shorter trajectory by directing the robot through the closest feasible gap toward the goal position. Therefore, it reduces trajectory length and guarantees a collision-free trajectory.

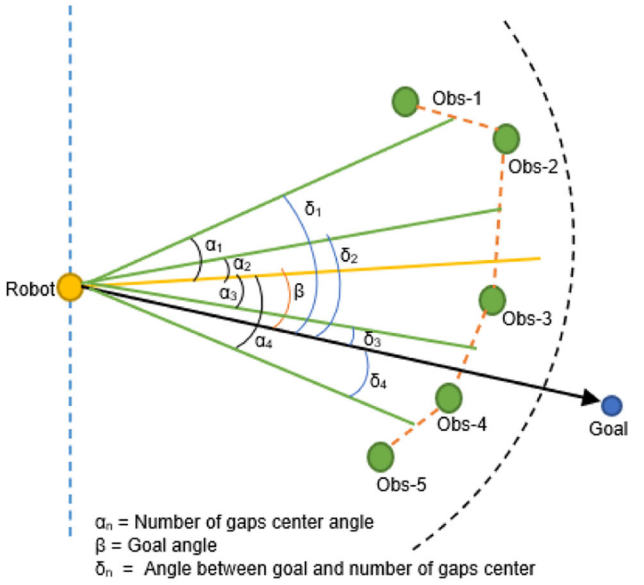

FIGURE 7 Find the feasible gap array for the robot [Color figure can be viewed at wileyonlinelibrary.com]

3.3.3 | Calculating the proper velocity of the robot

As described before, the DWA uses translational and rotational accelerations to determine the reachable velocities from the present velocities that can be achieved within a short time interval (t) (Berti et al., 2008). Let the current velocity is (v_c, ω_c), then the reachable velocity (V_r) is defined in Equation (15):

$$V_r = \{(v, \omega) | v \in [v_c - \dot{v}\,t, v_c + \dot{v}\,t] \wedge \omega \in [\omega_c - \dot{\omega}\,t, \omega_c + \dot{\omega}\,t].\} \quad (15)$$

The teachable velocity V_r considers only the safe velocities by eliminating excessive velocities come from the acceleration of the robot. It calculates an admission velocity V_a (admission velocity is able to stop the robot before the collision), when any obstacles present near to the robot. Depending on the minimum distance between robot and obstacles V_a is generated with the velocity pairs $d_{min}(v, \omega)$ and decelerations ($\dot{v}_b$, $\dot{\omega}_b$) of the mobile robot. The admissible velocity V_a is shown in Equation (16):

$$V_a = \{(v, \omega) | v \le (\sqrt{2d_{min}(v, \omega)\dot{v}_b}) \wedge \omega \le (\sqrt{2d_{min}(v, \omega)\dot{\omega}_b})\}, \quad (16)$$

V_a ensures that the DWA precisely control velocities of the robot during the movement.

3.3.4 | Objective function calculation for the proposed algorithm

As the proposed method is integrating the DWA and the IFGM, where both of the algorithms calculate the accessible velocity and the appropriate heading angle, respectively, to guide the robot toward the goal point. The path planning system is responsible for selecting the control commands to improve the performance of the robot. In this method, the robot even can move backwards when needed and the following Equations (17), (18), and (19) are developed according to Reeds and Shepp (1990). Therefore, the robot can turn back in the critical moment for example two or more moving obstacles coming from the front and there are no free spaces to move forward. However, in this proposed method the robot gives the priority to the forward motion to avoid obstacles by applying the proper heading angle and speed of the robot. Let $x(t)$, $y(t)$, and $\theta(t)$ present the robot's coordinate and the heading direction (robot's orientation) at time t in the global coordinate system. Therefore, the robot kinematic configuration can be expressed as (x, y, θ). The $x(t_i)$ and the $x(t_{i+1})$ describe the x-coordinates at time t_i and t_{i+1} of the mobile robot. The linear and the angular velocities of the robot are denoted by $v(t)$ and $\omega(t)$, respectively. The linear velocity $v(t)$ depends on the initial linear velocity (v_i) and the linear acceleration $\dot{v}(t)$ of the robot. Likewise, the angular velocity $\omega(t)$ can be defined by initial angular velocity $\omega(t_i)$ and the angular acceleration $\dot{\omega}(t)$, whereas the heading angle $\theta(t)$ depends on the initial orientation $\theta(t_i)$ of the robot. The operating system is shown in Figure 8.

When the robot is operating in the free spaces, the primary factor is the velocity command that means the robot gives the priority to the maximum linear velocity to go straight forward to the goal position and Equations (17), (18), and (19) are illustrated the straight forward movement of the robot toward the destination:

$$x(t_{i+1}) = x(t_i) + v(t)\cos\theta(t)dt, \tag{17}$$

$$y(t_{i+1}) = y(t_i) + v(t)\sin\theta(t)dt, \tag{18}$$

$$\theta(t_{i+1}) = \theta. \tag{19}$$

The robot changes the proper heading angle and operate on a moderate velocity, when it finds any obstacles in its trajectory.

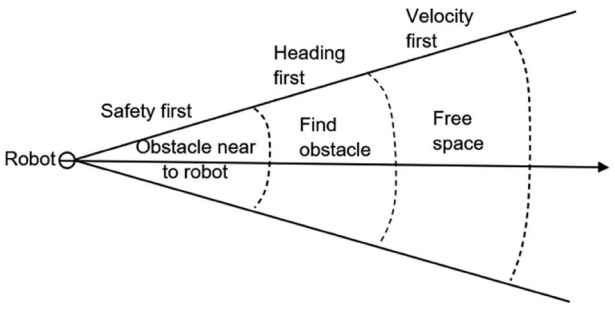

FIGURE 8 Different operating stages depending on dynamic obstacles distance

The heading angle and the velocity both of them are the priority factor of the robot to overcome obstacles. In this situation, the robot increases the heading angle and reduces the velocity as much as needed until it avoids obstacles. Equations (20), (21), and (22) illustrate the next movement from the current position according to the changes of the heading angle and the velocities of the robot, where $\phi = \int \omega(t)dt$:

$$x(t_{i+1}) = x(t_i) + \frac{v(t)}{\omega(t)}\sin(\phi + \theta) - \sin(\theta), \tag{20}$$

$$y(t_{i+1}) = y(t_i) + \frac{v(t)}{\omega(t)}\cos(\phi + \theta) - \cos(\theta), \tag{21}$$

$$\theta(t_{i+1}) = \theta + \phi. \tag{22}$$

The robot gives the priority to the safety factor when any dynamic obstacles is near to it. A safe distance must be maintained from nearby obstacles by increasing the maximum heading angle and reducing the maximum velocity. In the safety first stage the next movement actions are taken by the robot are shown in Equations (23), (24), and (25):

$$x(t_{i+1}) = x(t_i), \tag{23}$$

$$y(t_{i+1}) = y(t_i), \tag{24}$$

$$\theta(t_{i+1}) = \theta + \phi. \tag{25}$$

Distance calculation between the robot and the goal position is another important issue while the robot has to continuously calculate the distance and update its position until reaching the goal. Equations (26), (27), and (28) illustrate the distance calculation between robot and goal point, where the previous goal distance, the new goal distance, and the forward distance are denoted by d_{previous}, d_{new}, and d_{forward}, respectively:

$$d_{\text{previous}} = \sqrt{(x_{\text{rob}} - x_{\text{goal}})^2 + (y_{\text{rob}} - y_{\text{goal}})^2}, \tag{26}$$

$$d_{\text{new}} = \sqrt{(x_{\text{robnew}} - x_{\text{goal}})^2 + (y_{\text{robnew}} - y_{\text{goal}})^2}, \tag{27}$$

(x_{robnew} and y_{robnew} are the robot's predicted x and y coordinates). After calculating the robot's previous and new distance from the goal, a forward distance can be expressed as follows:

$$d_{\text{forward}} = d_{\text{previous}} - d_{\text{new}}. \tag{28}$$

Therefore, the robot can understand how much close it has moved to the final position. The improved FGM ensures the safety of the robot by directing it through the feasible gap and calculating a safe heading angle, whereas the DWA ensures proper velocity of the robot so that it can operate at an optimal speed except it closes to the final position. The robot operates at maximum velocity when far away from the goal position. However, the velocity of the robot must be reduced when it closes to the goal position. Otherwise, the robot will pass the goal

coordinates (Berti et al., 2008). The optimal velocity of the robot is determined from the objective function and it is defined by v_{opt} as shown in Equation (29):

$$v_{opt} = \begin{cases} \frac{v}{v_{max}}, & \text{when robot far away from goal,} \\ 1 - \frac{v}{v_{max}}, & \text{when robot close to the goal point.} \end{cases} \quad (29)$$

By this way, the proposed algorithm reduces the robot velocity when goes to the goal area which is less than the given limit distance. In this paper, the limit distance is considered 0.35 m. Therefore, the proposed algorithm prevents the global convergence problem and never passes the goal. The main steps of the proposed method are presented in Algorithm 1.

Algorithm 1: A feasible trajectory generation

```
initialization;
create random dynamic obstacles;
while location_robot - location_goal, Eq. (28) do
    robot moves until reach the destination;
    if find obstacles then
        calculate distance between the robot and obstacles, Eq.(10);
        find the feasible gap, Eq. (14);
        determine heading angle, Eq. (7,13);
        measure reachable velocity, Eq. (15);
        calculate admissible velocity, Eq. (16);
    else
        go straight in free spaces, Eq.(17, 18, and 19);
    if robot near to goal then
        reduce velocity, Eq. (29) ;
    else
        maintain maximum velocity, Eq. (29);
```

4 | SIMULATION RESULTS AND DISCUSSION

To verify the performance and the effectiveness of the proposed algorithm, it was implemented in the Python Pygame platform to create collision avoidance local trajectory for the mobile robot. The experimental environment is Intel (R) Core (TM) i7-9700T CPU @2.00 GHz, 1.99 GHz, and RAM 16.0 GB. The parameters of the proposed method are listed in Table 1. Every obstacle avoidance path planning algorithm has tuning different parameters to improve the performance of a mobile robot. Therefore, it is a very challenging issue to find a fair benchmarking for various local path planning methods. In this paper, the proposed method is compared with the original DWA and the classical APF algorithm, as they are most popular local path planning methods in robotics. The following subsection describe test scenarios and then verify the performance of the proposed method.

TABLE 1 Simulation parameters of the proposed method

Parameter	Value
Sampling time	dt = 0.1 s
Robot radius	r_{rob} = 0.15 m
Maximum linear velocity	v_{max} (t) = 0.25 m/s
Maximum leaner acceleration	$\dot{v}_{max}$ (t) = 0.17 m/s^2
Maximum angular velocity	ω_{max} (t) = 1.0 rad/s
Maximum angular acceleration	$\dot{\omega}_{max}$ (t) = 0.6 rad/s^2
Number of dynamic obstacles	N = 10
Obstacle radius	r_{obs} = 0.1 m
Obstacles velocity	v_{obs} = 0.05 m/s
weight coefficient for gap	m_1 = 0.4
weight coefficient for angle	m_1 = 0.6
Step length	0.014 m
Goal limit distance	0.35 m
Steps ahead to plan	10

4.1 | Test scenarios

Three methods (DWA, APF, and proposed algorithm) are tested in a series of simulations in different scenarios and the experimental results are compared in terms of the collision rate, the average time to goal, the average distance travel to the goal, and the average velocity of the robot. Ten circular dynamic obstacles with random coordinate values and movements are created in the experiments which are shown in Figure 9. Some test scenarios are created in this paper to verify the driving performance of a mobile robot. There are some common conditions that robots need to satisfy for local path planning system:

- keeping a minimum distance from the dynamic obstacles,
- increasing the velocity when there is no obstacle in the trajectory,
- reducing linear velocity and increasing angular velocity while avoiding obstacles,
- directing the robot towards the goal point, and
- stops when reaching the destination.

The aforementioned complicated conditions are few of the many situations for local path planning that must be satisfied by the robot.

4.1.1 | Test scenario I

As can be seen from Figure 10, The original DWA fails to reach the goal as a dynamic obstacle comes from the right front side Figure 10a. The classical DWA considers only the forward motion of the robot and when a robot finds any dynamic obstacle in its area, the robot starts to reduce its speed until the admission velocity becomes

zero. However, the dynamic obstacle is moving toward the robot and thereafter, the robot colliding with the obstacle. On the other hand, the APF and the proposed algorithm have easily overcome this problem and reach the desired destination Figures 10b,c. In the APF method, the goal position of the robot is defined by an attractive potential field, whereas the obstacles are represented by artificial repulsive potential fields. The attractive potential field generates forces to push the robot toward the goal, while the repulsive potential generates forces of the robot to push away from the obstacles. By this way, the APF method generates trajectory for mobile robots, whereas in the proposed method, when any dynamic obstacles are present infront of the robot it searches a feasible gap to avoid obstacle. If there is any feasible gap present in the trajectory the robot moves through the gap by selecting appropriate heading angle and parallelly reduces its velocity as much as needed to avoid obstacles. In this proposed method, the robot has applied forward motion to avoid any obstacles if there is at least one gap existing in the trajectory, otherwise, it has applied backward motion when moving obstacles coming toward the robot and there is no free space to move forward. Therefore, this algorithm is capable to create a safe and collision-free trajectory by avoiding moving obstacles.

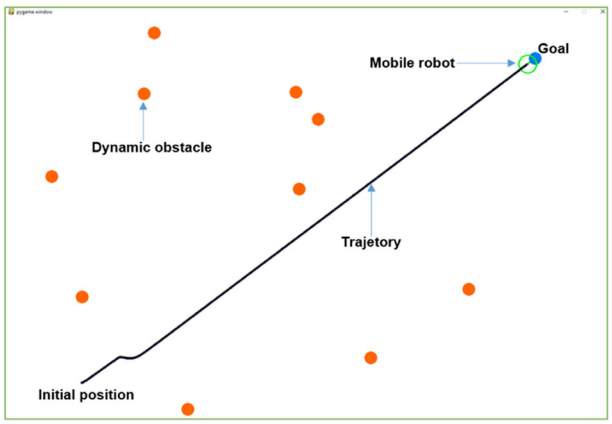

FIGURE 9 Trajectory is shown by the black color, whereas the red indicates 10 dynamic obstacles. The green and the lightblue represent the mobile robot and the goal position, respectively [Color figure can be viewed at wileyonlinelibrary.com]

4.1.2 | Test scenario II

When obstacles present at the beginning of the trajectory as shown in Figure 11 the robot gets stuck and can not move because the DWA is suffering from the local minimum problem as shown in Figure 11a. Therefore, the robot collides with the obstacle. On the other hand, the APF avoid the collision with the obstacles but suffers greatly while any obstacles are present near to the goal. However, in

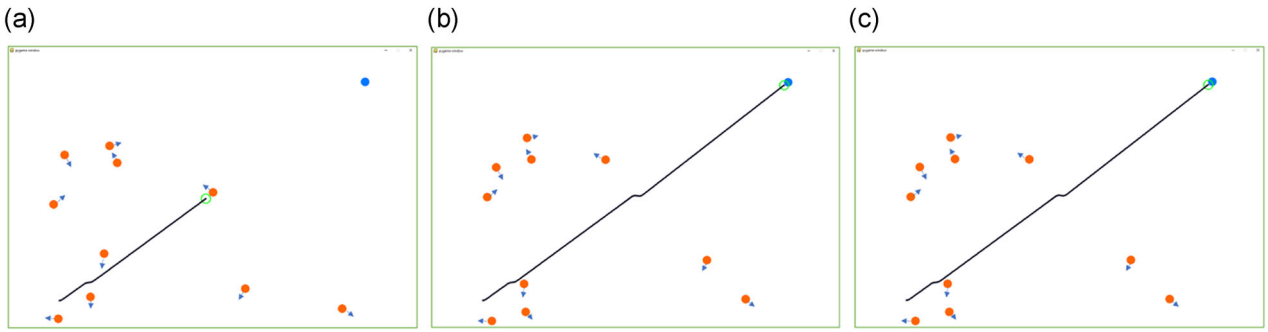

FIGURE 10 Scenario I: An obstacle is coming toward the robot. The traditional DWA (a) fails to reach the goal, whereas the APF (b), and the proposed algorithm (c) generate safe trajectories without any collision [Color figure can be viewed at wileyonlinelibrary.com]

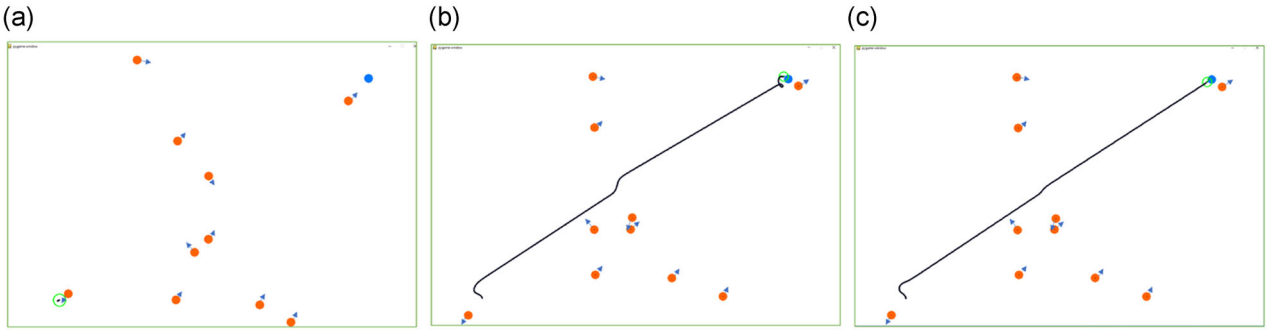

FIGURE 11 Scenario 2: Local minima problem. The DWA (a) collides with obstacle, while the APF (b) successfully reaches the goal but faces a noticeable local minima problem while obstacles are near to the goal position. However, the proposed algorithm (c) does not suffer with this problem and generate smooth trajectory [Color figure can be viewed at wileyonlinelibrary.com]

FIGURE 12 Scenario 3: Successful trajectories generated by all three methods: (a) represents the trajectory which is generated by the DWA, whereas (b) and (c) present the APF and the proposed method trajectories, respectively [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 2 The experimental results comparison among the DWA, the APF, and the proposed method

	Average distance travel (m)	Average time taken (s)	Average velocity (m/s)	Collision rates
Original DWA	8.59	45.52	0.19	33.33
Classical APF	9.08	77.47	0.12	0.00
Proposed method	9.05	64.31	0.14	0.00

TABLE 3 First four simulations are considered and compared, where all the methods (the DWA, the APF, and the proposed algorithm) successfully generate trajectories for the robot by avoiding collision

	Average distance travel (m)	Average time taken (s)	Average velocity (m/s)	Collision rates
Original DWA	8.60	46.28	0.19	0.00
Classical APF	9.02	50.78	0.185	0.00
Proposed method	8.98	44.05	0.20	0.00

the proposed method, the robot reaches the destination by avoiding these kind of obstacles because this algorithm does not suffer by the local minima problem Figure 11c.

4.1.3 | Test scenario III

The traditional DWA, the classical APF, and the proposed method generate almost similar trajectories when dynamic obstacles are coming from different sides as can be seen in Figure 12. The traditional DWA has successfully overcome any dynamic obstacles if they come from any sides. This algorithm allows moving obstacles to go first Figure 12a by reducing robot's speed and when moving obstacles pass its safety area, then it starts to move. On the other hand, the proposed algorithm increases the heading angle and reduces less velocity to overcome the obstacles that is shown in Figure 12c. For this reason, sometimes the proposed algorithm generates longer trajectory compared to the traditional DWA but it takes less time as the velocity is higher than the DWA. By contrast, the proposed algorithm always takes less time compared to the classical APF method. Tables 2 and 3 represent a summary of average travel time, average distance travel, collision rates, and average velocity of the traditional DWA algorithm, the APF method, and the proposed method: where Table 2 considers all simulation results and Table 3 represents only the first four successful experimental results.

4.1.4 | Test scenario IV

The robot starts to follow the trajectory and when there are two dynamic obstacles present in its path the robot must reduce its velocity and increase the heading angle to make enough space to avoid those obstacles shown in Figure 13. The APF and the proposed algorithm successfully avoid these two obstacles by reducing the maximum velocity and increasing the heading angle Figure 13c. However, it creates a longer trajectory and takes 171.10 s for the proposed method, while the APF takes maximum time 194.3s to reach the destination. By contrast, the traditional DWA collides with the obstacles as can be seen in Figure 13a.

4.1.5 | Test scenario V: The velocity of the dynamic obstacles is 0.10 m/s

The test has been further extended to include the effect of increasing the obstacle velocity to 0.10 m/s (from 0.05 m/s). Figure 14 shows

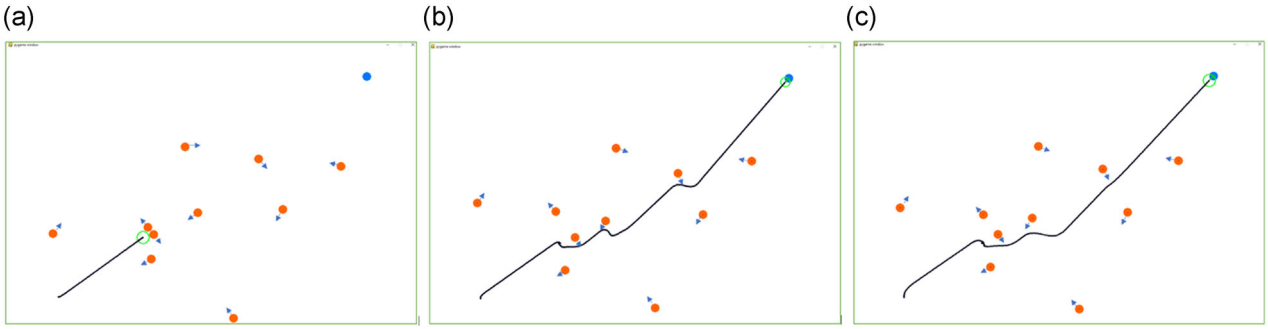

FIGURE 13 Scenario 4: The speed of the dynamic obstacles are 0.05 m/s and two dynamic obstacles present in the trajectory. (a), (b), and (c) present the DWA, the APF, and the proposed method trajectories, respectively [Color figure can be viewed at wileyonlinelibrary.com]

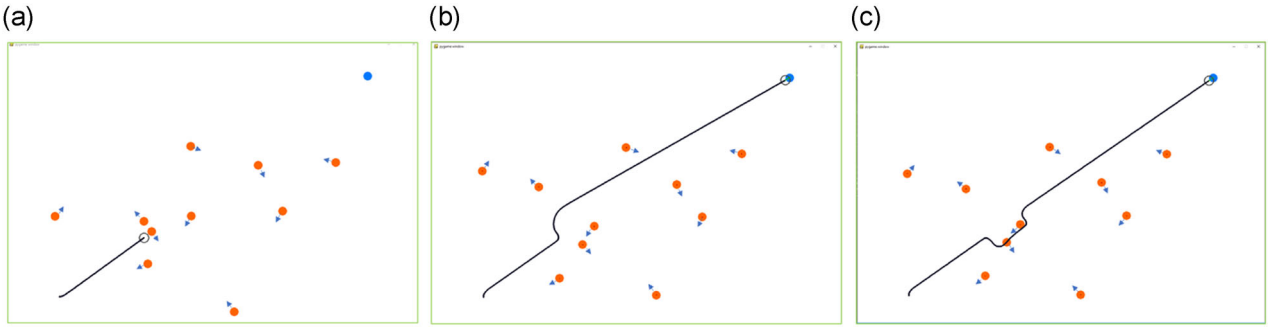

FIGURE 14 Scenario 5: The speed of the dynamic obstacles are 0.10 m/s and two obstacles present infront of the robot. (a), (b), and (c) present the DWA, the APF, and the proposed method trajectories, respectively [Color figure can be viewed at wileyonlinelibrary.com]

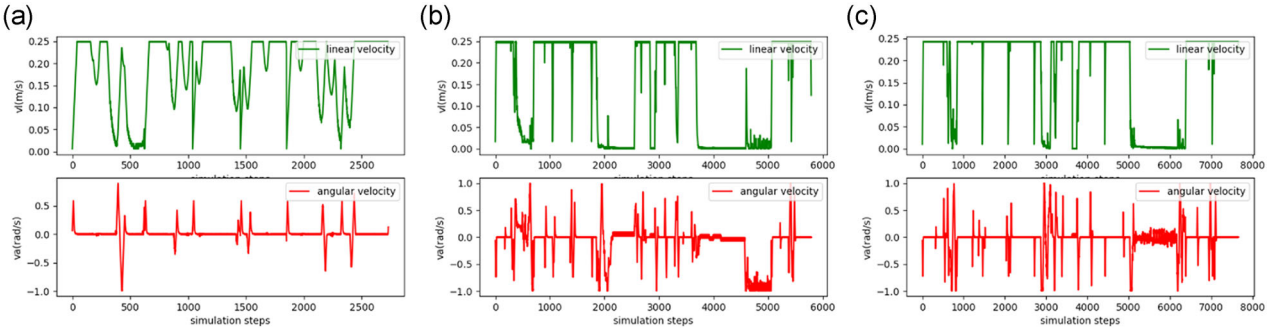

FIGURE 15 (a) and (b) represents the linear and angular velocity of the DWA and the APF method, respectively, where (c) shows the proposed method's linear and angular velocity throughout the experiments [Color figure can be viewed at wileyonlinelibrary.com]

the trajectories where the speed of the moving obstacles is 0.10 m/s, whereas, Figure 13 shows the paths when the dynamic obstacles velocities are 0.05 m/s.

In Figures 13a and 14a, it can be noticed that there is not much difference in the trajectories with higher and lower velocities. From Figures 13b,c and 14b,c it can be seen that the effect of increased obstacle velocity is the generation of shorter and straighter trajectories. The moving obstacles with higher velocity in Figure 14 pass the robot's safe area faster compared to the lower velocity of the obstacles in Figure 13 and the robot makes quicker decision to avoid collisions. Consequently, the robot takes less time and travels less distance to reach the set destination. Therefore, depending on the velocity of the obstacles the local path planning methods generate different trajectories.

4.2 | Drivability verification of the proposed method

Figure 15 shows the variations of the linear and angular velocities of the DWA, the APF, and the proposed method throughout the simulations. Drivability verification of two methods are compared on the basis of average velocity in Tables 2 and 3.

As can be seen from Table 2, the proposed method average velocity is about one quarter lower than the original DWA. From Figures 10, 11, and 13 the reason can be easily explained why the proposed approach velocity is lower. These experiments show that the traditional DWA failed to reach the goal, while the proposed method reached the destination but the robot was maintaining the lower velocities to avoid dynamic

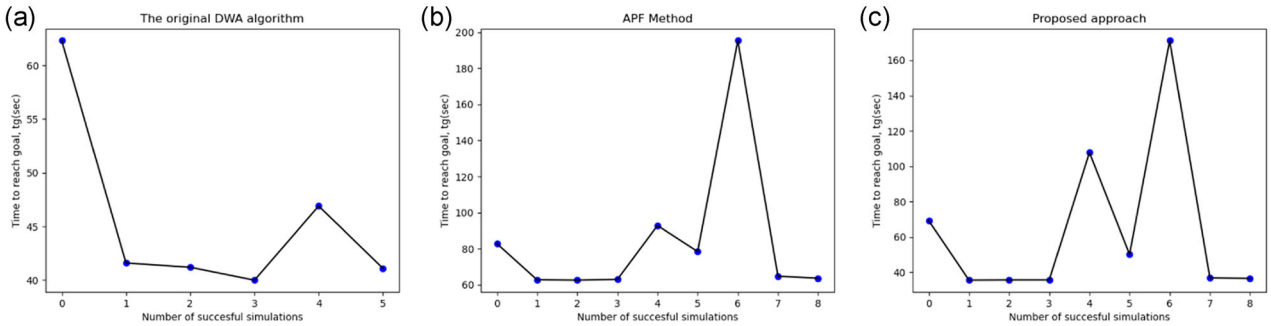

FIGURE 16 Comparison the successful number of simulations and time to reach the goal. The DWA (a) shows that it reaches destination six times among nine times simulations, whereas the APF (b) and the Proposed method (c) successfully achieve the goal nine times [Color figure can be viewed at wileyonlinelibrary.com]

obstacles. However, from Table 3 (where two methods have successfully reached the destination) it can be noticed that the proposed approach average velocity (0.20 m/s) is higher than the average velocity (0.19 m/s) of the DWA. Therefore, the proposed local path planning algorithm is properly maintaining the translational and rotational velocity of the robot during the movement. On the other hand, the average velocity of the proposed method is higher compared to the average velocity (0.12 m/s) of the APF method as the APF has a local minima problem, therefore, it reduces its maximum speed while obstacles are present near to the goal.

4.3 | Safety verification of the proposed algorithm

The most significant contribution of this paper is the improvement of the successful trajectory generation without any collision. Safety rate of the proposed algorithm is verified by comparing the collision rate between the proposed method, the APF, and the DWA. The speed of the dynamic obstacles are set to be 0.05 (m/s) and nine simulations have experimented for safety verification. Comparison of the collision rates is shown in Table 2. From Table 2, it can be seen that the APF and the proposed algorithm generates 0.0% collision rate that means these methods 100% successful to generate collision-free trajectories without any collision. By contrast, the original DWA collision rate is 33.33% means among nine times simulation the robot has failed three times to reach the destination. Therefore, the proposed method and the APF method generate much safer trajectory compared to the existing DWA.

4.4 | Distance and time verification

It can be noticed in Table 2, the proposed method generates about 5% longer trajectories than the traditional DWA. Moreover, this algorithm takes more average time than the DWA to reach the goal. The successful number of simulations and arrival times are shown in Figure 16. The traditional DWA takes average 45.52 s, while the average time to the goal of the proposed algorithm is 64.31 s. This means the proposed approach takes around 29% more average time compared to the DWA and travels longer distance. As we mentioned before, in some cases the DWA failed to avoid the dynamic obstacles, where the proposed method took more time and traveled longer distance to successfully generate safe trajectories by avoiding moving obstacles. However, the average travel distance of the APF and the Proposed method are almost similar but the APF takes more time than the proposed method to reach the goal.

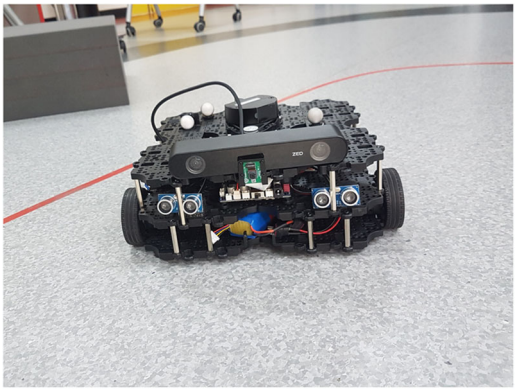

FIGURE 17 TurtleBot two-wheeled differential drive robot [Color figure can be viewed at wileyonlinelibrary.com]

According to Table 3, the Proposed method shows better average arrival time (44.05 s) than the average arrival time of the DWA (46.28 s) and the APF (50.78 s). However, in the proposed algorithm the robot travels longer average distance compared to the original DWA method and shorter average distance compared to the APF. The proposed method does not always guarantee to create the shortest trajectory, but it ensures safe and fast trajectory for the mobile robot.

5 | EXPERIMENTAL RESULTS AND DISCUSSION

The experimental results of the TurtleBot robot in different dynamic environments are presented in this section. The path planning of the TurtleBot robot has been performed through robot operating system (ROS) platform. It is a two-wheeled differential drive robot, as shown in Figure 17. There are two separate DC motors in this robot which are controlling the orientation and motion of the TurtleBot. It also has a caster wheel, which is used to support the chassis of the robot. The dimensions

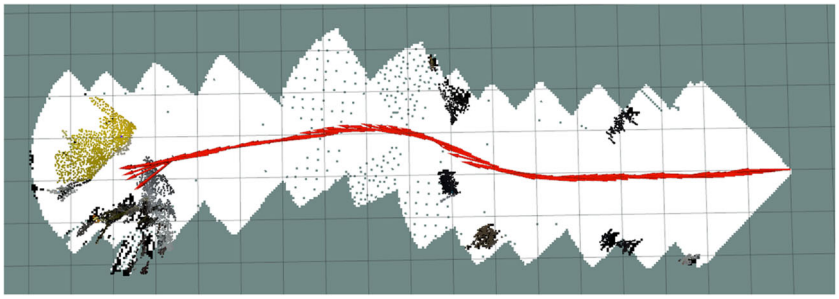

FIGURE 18 Experiment one result: 3D trajectory generation of the TurtleBot [Color figure can be viewed at wileyonlinelibrary.com]

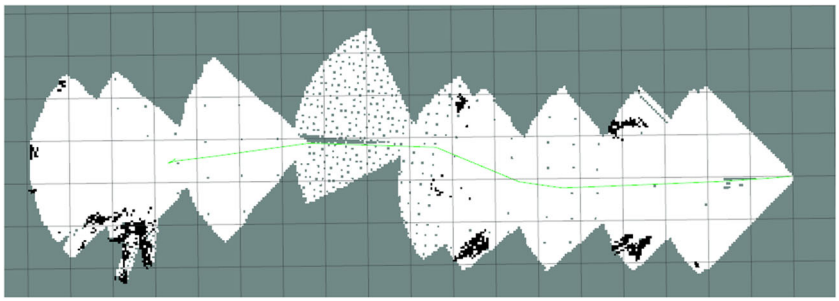

FIGURE 19 Two-dimensional navigation result for the experiment one of the TurtleBot between five moving obstacles in the environment [Color figure can be viewed at wileyonlinelibrary.com]

of the TurtleBot are 42.5 cm (length), 42.5 (width), and 16.8 cm (height), respectively. It is equipped with a ZED stereo camera and eight ultrasonic sensors. The ZED Stereo camera is a 3D sensor which is used for long-range depth perception and motion tracking of the robot. Ultrasonic sensors are used to measure the obstacles distance from 2.5 cm to 4.3 m approximately. The proposed method has been designed in Python, and this method controls the steering angle and the speed of the TurtleBot through ROS function. The ROS establish a connection between Python software and TurtleBot robot. The real-time sensor data, traveling time, traveling distance, and the positions (x-axis and y-axis) of the TurtleBot cab be extracted from the ROS function. The real-time path planning and motion control performance of the proposed method is shown in Figures 18, 19, and 20. Figures 18, and 19, present the 3D and 2D view of the experiment result of the TurtleBot by applying proposed algorithm, where five dynamic obstacles are present in the environment. Similarly, Figure 20 shows the 3D trajectory of the TurtleBot in experiment two to reach the target, where seven moving obstacles are present in the dynamic environment.

It can be observed from Figures 18, 19, and 20 that the proposed method-based TurtleBot has generated a smooth, short, and collision free path to reach the goal. In these experiments, the beginning position and the goal position of the TurtleBot has been predefined, while the position of dynamic obstacles are unknown to the robot. The distance of the obstacles received from sensors of the TurtleBot which are the inputs. The outputs are the steering angle and the speed of the differential drive robot. A threshold limit distance is set 60 cm between the TurtleBot and the obstacles to avoid collision in different environments. When the TurtleBot detects any obstacles within this threshold distance range, the proposed method will control the speed and steering angle of the robot to avoid collision. If there is no obstacles present within the threshold distance range of the robot, then it moves ahead for reaching the destination point. In this proposed method the robot is not bound to track a given route. The robot looks at its industrial environment and can redefine the shortest and best trajectory to reach the destination while moving obstacles are appearing in its trajectory. Successful experimental results show that the proposed method based robot is able to generate a robust, safe, and goal-directed trajectory in the dynamic environment by avoiding static and moving obstacles.

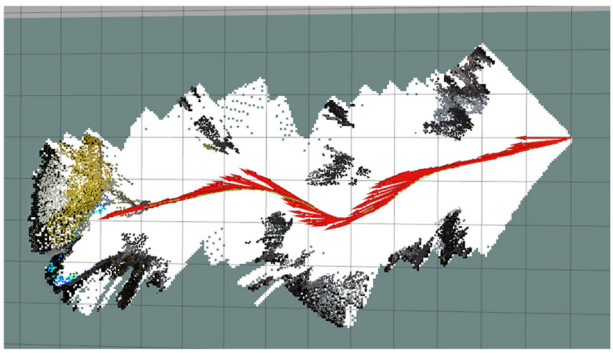

FIGURE 20 Experiment two: 3D trajectory is generated by applying the proposed method on the TurtleBot [Color figure can be viewed at wileyonlinelibrary.com]

6 | POTENTIAL REAL-TIME APPLICATION OF THE PROPOSED METHOD

The proposed method is very promising for industrial environments and allows the robot to work collaboratively with moving workers and people. It is suitable for autonomous picking, packing, and pelletizing, warehouse applications where the robot needs to

subassemblies products from one place to another place. It can calculate the most appropriate route for material transport by navigating safely around dynamic obstacles. The proposed path planning algorithm is also applicable for service robots such as restaurant food delivery robots for delivering food to customers from the counter with precision and efficiency. It can stop, take a turn, or even reverse direction of the mobile robot for choosing another trajectory when moving obstacles such as customers are present in its path. The robot can also safely avoid moving obstacles and keep on traveling by using the information from the environment. The autonomous robot can collide with customers that can cause harm if the path planning algorithm is not properly working. The proposed algorithm is capable to create a path around dynamic obstacles and reach the desired destination to serve food to customers without any collision.

The proposed method also could be a potential solution for family farming robots, for example, autonomous lawn mowers, ground robots for crop surveillance in tree-crop applications, autonomous weeding robot, where different moving obstacles including children, pets, and other animals can be present in the dynamic environment. This proposed method can facilitate a mobile robot to stop, back up, turn, and moves away from anything that it may unexpectedly encounter within the surrounding environment.

7 | CONCLUSION

In the dynamic environment, it is very challenging for mobile robots to automatically generate safe, robust, and goal-oriented collision-free trajectory in real-time. In this paper, a real-time reactive local path planning algorithm is proposed to generate a collision-free trajectory with respect to the different threat levels of the dynamic obstacles. There are different test scenarios are created where moving obstacles are coming from different coordinates and the proposed algorithm can prevent the collision of the robot from the nearby unexpected dynamic obstacles. It contributes to the following innovation points:

- The robot always maintains a safe distance from any closest moving obstacles and search the gap to avoid a collision. When the robot finds at least one gap, it can easily avoid the obstacle by moving through this gap which ensures the safe trajectory.
- The heading angle and the admissible velocity are parallelly working to improve the robustness and the stability of the mobile robot. They are applied depending on the relative position of the robot and moving obstacles. The heading angle is increasing to change the direction of the mobile robot. The velocity of the robot changes according to the threat levels of the dynamic obstacles. The threat level differs according to position and speed between the robot and obstacles. Higher threat level means moving obstacles are close to the robot, in this situation, the robot reduces the maximum speed and increases heading angle to avoid obstacles.
- The proposed algorithm does not suffer from the local minima problem, while the other local path planning methods seriously suffer from this problem.
- The proposed method takes less time to reach the goal, as it moves with the higher velocity compared to the existing DWA, but sometimes generate longer trajectory than the traditional DWA.

From the experiments, it can be shown that the proposed algorithm is capable to create safe, fast, and more goal-oriented collision-avoidance trajectory in different test scenarios with the moving obstacles. The experimental results show that the proposed algorithm outperforms the traditional DWA on the basis of the safety and the drivability to generate a collision-free trajectory. The proposed local path planning algorithm is applicable to the ground mobile robots with a lower velocity.

The future work will further develop the collision-free local path planning algorithm with the higher velocity of the robot. The noise and uncertainties of the robot will be considered precisely to create a more extensive, fast, and safe path planning algorithm.

ACKNOWLEDGMENTS

The authors would like to acknowledge the support from the Southern Cross University (SCU) and the University of South Australia (UniSA) for the stipend and Tuition Fee Offset scholarships. Moreover, the author would like to thank the SCU future lab and the UniSA robotics lab for the experimental facilities.

ORCID

Tagor Hossain http://orcid.org/0000-0002-8718-0627

REFERENCES

Ahmadzadeh, S., & Ghanavati, M. (2012). Navigation of mobile robot using the pso particle swarm optimization. *Journal of Academic and Applied Studies*, *2*.

Arras, K. O., Persson, J., Tomatis, N., & Siegwart, R. (2002). Real-time obstacle avoidance for polygonal robots with a reduced dynamic window. In *Proceedings 2002 IEEE International Conference on Robotics and Automation* (Cat. No.02CH37292) (Vol. 3, pp. 3050–3055).

Aurenhammer, F. (1991). Voronoi diagrams-a survey of a fundamental geometric data structure. *ACM Computing Surveys*, *23*, 345–405.

Azmi, M. Z., & Ito, T. (2020). Artificial potential field with discrete map transformation for feasible indoor path planning. *Applied Sciences*, *10*. https://www.mdpi.com/2076-3417/10/24/8987

Babinec, A., Duchoň, F., Dekan, M., Mikulová, Z., & Jurišica, L. (2018). Vector field histogram* with look-ahead tree extension dependent on time variable environment. *Transactions of the Institute of Measurement and Control*, *40*, 1250–1264.

Ballesteros, J., Urdiales, C., Velasco, A. B. M., & Ramos-Jiménez, G. (2017). A biomimetical dynamic window approach to navigation for collaborative control. *IEEE Transactions on Human-Machine Systems*, *47*, 1123–1133.

Berti, H., Sappa, A., & Agamennoni, O. (2008). Improved dynamic window approach by using lyapunov stability criteria. *Latin American Applied Research Pesquisa Aplicada Latino Americana*, *38*.

Borenstein, J., & Koren, Y. (1991). The vector field histogram-fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*, *7*, 278–288.

Borenstein, J., & Koren, Y. (1989). Real-time obstacle avoidance for fast mobile robots. *IEEE Transactions on Systems, Man, and Cybernetics*, *19*, 1179–1187.

Brock, O., & Khatib, O. (1999). High-speed navigation using the global dynamic window approach. In *Proceedings 1999 IEEE International Conference On Robotics and Automation* (cat. no.99ch36288c) (Vol. 1, pp. 341–346).

Chakravarthy, A., & Ghose, D. (1998). Obstacle avoidance in a dynamic environment: A collision cone approach. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, *28*, 562–574.

Choset, H., Lynch, K. M., Hutchinson, S., Kantor, G. A., Burgard, W., Kavraki, L. E., & Thrun, S. (2005). Principles of robot motion: Theory, algorithms, and implementations-trajectory planning, *Principles of robot motion: Theory, algorithms, and implementations* (pp. 373–400). MIT Press.

Demir, M., & Sezer, V. (2017). Improved follow the gap method for obstacle avoidance. In *2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)* (pp. 1435–1440).

Dolgov, D., Thrun, S., Montemerlo, M., & Diebel, J. (2009). Path planning for autonomous driving in unknown environments. In O. Khatib, V. Kumar, & G. J. Pappas (Eds.), *Experimental robotics* (pp. 55–64). Springer.

Duchoň, F., Babinec, A., Kajan, M., Beňo, P., Florek, M., Fico, T., & Jurišica, L. (2014). Path planning with modified a star algorithm for a mobile robot. *Procedia Engineering*, *96*, 59–69.

Erke, S., Bin, D., Yiming, N., Qi, Z., Liang, X., & Dawei, Z. (2020). An improved a-star based path planning algorithm for autonomous land vehicles. *International Journal of Advanced Robotic Systems*, *17*, 1729881420962263.

Filipescu, A., Mınzu, V., Dumitrascu, B., Filipescu, A., & Minca, E. (2011). Trajectory-tracking and discrete-time sliding-mode control of wheeled mobile robots. In *2011 IEEE International Conference on Information and Automation, ICIA 2011*.

Fox, D., Burgard, W., & Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics Automation Magazine*, *4*, 23–33.

Ghita, N., & Kloetzer, M. (2012). Trajectory planning for a car-like robot by environment abstraction. *Robotics and Autonomous Systems*, *60*, 609–619.

Ji, X., Feng, S., Han, Q., Yin, H., & Yu, S. (2021). Improvement and fusion of a* algorithm and dynamic window approach considering complex environmental information. *Arabian Journal for Science and Engineering*, *46*.

Kang, Y., de Lima, D. A., & Victorino, A. C. (2015). Dynamic obstacles avoidance based on image-based dynamic window approach for human-vehicle interaction. In *2015 IEEE Intelligent Vehicles Symposium (IV)* (pp. 77–82).

Kavraki, L. E., Svestka, P., Latombe, J., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, *12*, 566–580.

Kazem, B., Hamad, A., & Mozael, M. (2010). Modified vector field histogram with a neural network learning model for mobile robot path planning and obstacle avoidance. *International Journal of Advanced Computer Technology*, *2*, 166–173.

Khatib, O. (1985). Real-time obstacle avoidance for manipulators and mobile robots. In *Proceedings of 1985 IEEE International Conference on Robotics and Automation* (Vol. 2, pp. 500–505).

Lavalle, S. M. (1998). *Rapidly-exploring random trees: A new tool for path planning*. Technical report.

Lavalle, S. M. (2006). *Planning algorithms*. Cambridge University Press.

Lazarowska, A. (2019). Discrete artificial potential field approach to mobile robot path planning. *IFAC-PapersOnLine*, *52*, 277–282. 10th IFAC Symposium on Intelligent Autonomous Vehicles IAV 2019. https://www.sciencedirect.com/science/article/pii/S2405896319304161

Lozano-Pérez, T., & Wesley, M. A. (1979). An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, *22*, 560–570.

Lumelsky, V., & Stepanov, A. (1987). Path-planning strategies for a point mobile automation moving amdist unknown obstacles of arbitrary shape. *Algorithmica*, *2*, 403–430.

Min, H., Lin, Y., Wang, S., Wu, F., & Shen, X. (2015). Path planning of mobile robot by mixing experience with modified artificial potential field method. *Advances in Mechanical Engineering*, *7*, 1687814015619276.

Minguez, J. (2005). The obstacle-restriction method for robot obstacle avoidance in difficult environments. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 2284–2290).

Minguez, J., & Montano, L. (2000). Nearness diagram navigation (nd): A new real time collision avoidance approach. In *Proceedings of 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)* (Cat. No.00CH37113) (Vol. 3, pp. 2094–2100).

Ogren, P., & Leonard, N. E. (2005). A convergent dynamic window approach to obstacle avoidance. *IEEE Transactions on Robotics*, *21*, 188–195.

Ozdemir, A., & Sezer, V. (2017). A hybrid obstacle avoidance method: Follow the gap with dynamic window approach. In *2017 First IEEE International Conference on Robotic Computing (IRC)* (pp. 257–262).

Pandey, A., Panwar, V. S., Hasan, M. E., & Parhi, D. R. (2020). V-REP-based navigation of automated wheeled robot between obstacles using PSO-tuned feedforward neural network. *Journal of Computational Design and Engineering*, *7*, 427–434.

Pandey, A., & Parhi, D. R. (2017). Optimum path planning of mobile robot in unknown static and dynamic environments using fuzzy-wind driven optimization algorithm. *Defence Technology*, *13*, 47–58. https://www.sciencedirect.com/science/article/pii/S2214914716300824

Pandey, D. A., Kashyap, A. K., Parhi, D., & Patle, B. (2019). Autonomous mobile robot navigation between static and dynamic obstacles using multiple anfis architecture. *World Journal of Engineering*, *16*.

Patle, B., Babu LG, Pandey, A., Parhi, D., Jagadeesh, A. (2019). A review: On path planning strategies for navigation of mobile robot. *Defence Technology*, *15*, 582–606. https://www.sciencedirect.com/science/article/pii/S2214914718305130

Reeds, J., & Shepp, L. (1990). Optimal paths for a car that goes both forwards and backwards. *Pacific Journal of Mathematics*, *145*, 367–393.

Rostami, S. M. H., Sangaiah, A. K., Wang, J., & Liu, X. (2019). Obstacle avoidance of mobile robots using modified artificial potential field algorithm. *EURASIP Journal on Wireless Communications and Networking*, *2019*, 1–19.

Sezer, V., & Gokasan, M. (2012). A novel obstacle avoidance algorithm: "Follow the gap method'. *Robotics and Autonomous Systems*, *60*, 1123–1134.

Siciliano, B., & Khatib, O. (2008). *Springer handbook of robotics*. Springer-Verlag.

Siegwart, R., Nourbakhsh, I. R., & Scaramuzza, D. (2011). *Planning and navigation* (pp. 369–423). MIT Press.

Simmons, R. (1996). The curvature-velocity method for local obstacle avoidance. In *Proceedings of IEEE International Conference on Robotics and Automation* (Vol. 4, pp. 3375–3382).

da Costa Barros, Í. R., & Nascimento, T. P. (2021). Robotic mobile fulfillment systems: A survey on recent developments and research opportunities. *Robotics and Autonomous Systems*, *137*, 103729. https://www.sciencedirect.com/science/article/pii/S0921889021000142

Tzafestas, S. G. (2014). 2—Mobile robot kinematics. In S. G. Tzafestas (Ed.), *Introduction to mobile robot control* (pp. 31–67). Elsevier.

Ulrich, I., & Borenstein, J. (2000). local obstacle avoidance with look-ahead verification. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings* (Cat. No.00CH37065) (Vol. 3, pp. 2505–2511).

Zohaib, M., Pasha, S. M., Javaid, N., & Iqbal, J. (2014). Iba: Intelligent bug algorithm—A novel strategy to navigate mobile robots autonomously. In *Communication technologies, information security and sustainable development* (pp. 291–299). Springer International Publishing.

How to cite this article: Hossain, T., Habibullah, H., Islam, R., & Padilla, R. V. (2021). Local path planning for autonomous mobile robots by integrating modified dynamic-window approach and improved follow the gap method. *Journal of Field Robotics*, 1–16. https://doi.org/10.1002/rob.22055