

RESEARCH ARTICLE

# A new gap-based obstacle avoidance approach: follow the obstacle circle method

Hosein Houshyari and Volkan Sezer\* 

Autonomous Mobility Group, Control and Automation Engineering Department, Istanbul Technical University, Istanbul, Turkey

\*Corresponding author. E-mail: [sezerv@itu.edu.tr](mailto:sezerv@itu.edu.tr)

**Received:** 13 February 2021; **Revised:** 7 July 2021; **Accepted:** 12 October 2021; **First published online:** 18 November 2021

**Keywords:** obstacle avoidance, autonomous robots, path planning, motion planning, local planning

## Abstract

One of the most challenging tasks for autonomous robots is avoiding unexpected obstacles during their path following operation. Follow the gap method (FGM) is one of the most popular obstacle avoidance algorithms that recursively guides the robot to the goal state by considering the angle to the goal point and the distance to the closest obstacles. It selects the largest gap around the robot, where the gap angle is calculated by the vector to the midpoint of the largest gap. In this paper, a novel obstacle avoidance procedure is developed and applied to a real fully autonomous wheelchair. This proposed algorithm improves the FGM's travel safety and brings a new solution to the obstacle avoidance task. In the proposed algorithm, the largest gap is selected based on gap width. Moreover, the avoidance angle (similar to the gap center angle of FGM) is calculated considering the locus of the equidistant points from obstacles that create obstacle circles. Monte Carlo simulations are used to test the proposed algorithm, and according to the results, the new procedure guides the robot to safer trajectories compared with classical FGM. The real experimental test results are in parallel to the simulations and show the real-time performance of the proposed approach.

## 1. Introduction

Motion planning refers to the robots' ability to travel from the initial point toward the target without collisions. These algorithms are developed to find the admissible path starting from the initial state toward the final state, depending on whether the map is given or not, and the type of obstacles, whether they are stable and static or dynamic. Given a goal state and a map of the environment and assuming that the obstacles are static and not changing during the travel time, the global planning algorithms can calculate the admissible trajectory on the maps' collision-free spaces; the trajectory would be admissible because the map used in these categories of path planning algorithms is static and not updated dynamically [1, 2]. Many different approaches and studies have been made for global path planning purposes, like A\*, [3] probabilistic road maps, [4] Dijkstra, [5] rapidly exploring random trees (RRTs), [6] RRT\*, [7] cell decomposition methods, [8] Voronoi diagrams, [9] and visibility graphs. [10] However, these algorithms' efficiency and accuracy can be precarious when the information about the environment and obstacles contains uncertainty. While the robot performs the path planning task, the working environment can be unknown, contains dynamic obstacles, or even the robot's motion can be uncertain. In these situations, the local planning algorithms are used to prevent the robot from colliding with the obstacles while the robot is trying to track the admissible trajectory from the initial state to the final state. It is noteworthy to consider that local path planning algorithms use less information about the environment, which causes them to have lower computational complexity in comparison with the global path planning algorithms. Like the global planners, local path planning algorithms are being studied and improved in recent years. One of the firstly developed obstacle avoidance methods is the Bug 1 algorithm. [11] Using this algorithm, the robot starts to circumnavigate the possible obstacles while moving toward the target,

and it remembers how close it gets to the final state during this maneuver, and then it returns to the closest point and continues to the target. The Bug 2 algorithm [12] is an improvement of Bug 1. In Bug 2, the robot starts moving toward the goal, and when it faces an obstacle, the robot encounters it until there is no obstacle between the robot and the target and then continues moving toward the target. Artificial potential field (APF) [13] is another common obstacle avoidance method that uses the potential field for declaring the goal state and obstacles inside the environment. In this method, the repulsive APF is provided by obstacles, and the goal point generates an attractive APF. While the robot is traveling using this approach, the attractive potential field from the goal point creates a pulling force, and the repulsive force generates a pushing force, which causes the robot to avoid the obstacles. Although the APF is simple, it has a local minimum problem, which causes the robot to stuck in the environment. [14] This issue happens when the sum of the forces coming from the goal point and obstacles become zero. Improvement for this method can be found in refs. [15, 16]. Another method similar to the APF is obstacle restriction method [17] that uses the repulsive angle sets and subgoals instead. The virtual force field (VFF) method [18] is another popular method used in obstacle avoidance task. Firstly, the obstacles are represented by a two-dimensional cartesian histogram grid, wherein the possibility of comprising an obstacle at that point is shown by each cell in the histogram grid. After this step, the APF is applied to the histogram grid. Similar to the APF, the VFF algorithm has the local minimum problem too. The vector field histogram (VFH) approach [19] is similar to the VFF, where a two-dimensional cartesian histogram grid represents the obstacles at the robot's momentary location. However, the VFH reduces it to a one-dimensional obstacle density polar histogram. This one-dimensional polar histogram is then used for comparing the sectors around the robot, and the algorithm selects the sector with the lowest obstacle density value. Finally, the algorithm sets the robot's heading angle in compliance with the chosen sector's direction. VFH\* [20] is an improved version of standard VFH. The curvature velocity method (CVM) [21] uses the robots' dynamic model to calculate possible curvature paths by discarding those that collide with the obstacles. Another commonly used obstacle avoidance method is the dynamic window approach (DWA). [22] This method, similar to the CVM, uses the robot's dynamic properties; however, the mechanism is different. It selects the optimum admissible velocity set (V,W) by discarding the sets that cause the robot to collide and then maximizing a cost function. An improvement study for DWA can be found in ref. [23]. Timed elastic band (TEB) method [24] is another popular method used in obstacle avoidance task. This method optimizes the robot's trajectory locally, regarding the trajectory execution time. It is noteworthy to mention that TEB is an extended version of elastic bands. An improved study for this method can be found in ref. [25]. In addition to the mentioned algorithms, there are some metaheuristic-based methods used in mobile robot path planning. The example of these algorithms can be found in refs. [26, 27]. One of these methods uses membrane evolutionary APF. [27] This method mixes membrane computing with the metaheuristic genetic algorithm and the APF method to obtain the parameters to find an accessible and safe path.

Apart from these, another obstacle avoidance approach, "follow the gap method (FGM)," is one of the most popular obstacle avoidance algorithms that recursively guides the robot to the goal state by considering the angle to the goal point and the distance to the closest obstacles. [28] According to the comparison study in ref. [29], the original FGM is shown to be the most efficient one comparing to several famous obstacle avoidance approaches. Many other research and studies have been made to improve this method in recent years. Improved follow the gap method (FGM-I) [30] brings a new solution to the classic FGM's zigzag and the trajectory length problem. Another study that aims to improve the FGM is called FGM-DWA. [31] This method combines the DWA and FGM by modifying the DWA's mechanism of selecting the optimum velocity pair by changing the heading function inside the DWA's objective function and generating a new heading score using the FGM's safe heading angle for each set. In 2019, FGM was used for overtaking maneuver. [32] As another improvement for FGM, FGM-I2 [33] brings a new solution to obstacle avoidance task by mixing the FGM-I and DWA. The latest study aiming to improve the FGM is represented in follow the dynamic gap method, [34] wherein the algorithm develops a new gap selection strategy for FGM, resulting in safe trajectories by considering the gap borders' velocity vectors and the future gap change.

In this paper, a novel procedure for improving the classical FGM is developed and applied to a fully autonomous wheelchair. This proposed algorithm, entitled as “follow the obstacle circle method (FOCM),” brings a new solution to the obstacle avoidance task that improves FGM’s travel safety. In the FGM, the gaps are selected based on their size in angle, and the final heading angle of the robot is calculated considering the angle to the gaps’ center point. While FGM works well in most cases, we propose a safer strategy for both gap selection and final angle calculation parts. The main contributions of this newly proposed approach are:

- We propose “obstacle circle” concept to maximize the distance to obstacle in all conditions.
- We select the gaps based on gaps’ width instead of their angular values to get safer paths.
- We show the statistically significance of the results for the proposed improvements, which has not been done for FGM and its variants previously.

In the subsequent parts of this paper, we address, in detail, that these two contributions make the proposed approach safer without a significant extension of the travel length.

The remainder of the paper is organized in the following way: Section 2 presents the FGM. Section 3 introduces the new approach. Simulation results are demonstrated in Section 4, Section 5 introduces the experimental platform, Section 6 discusses the experimental results, and finally, the conclusion is presented in Section 7.

## 2. Follow the gap method

FGM is a safety-based geometric obstacle avoidance algorithm that recurrently leads the robot to the goal point while keeping the robot from colliding with the obstacles. [28] It selects the gaps with maximum angle size, using the sensory information. FGM is a three-stage algorithm. The first stage is calculating the gap arrays. In this stage, the algorithm uses the current sensory information like the LIDAR sensor to generate a gap array. This array contains information about the size of the existing gaps around the robot in angle form. FGM selects the largest gap at the end of this stage, which is called the gap selection process. FGM calculates the angle to the gap’s center point in the second stage, using specific geometric relations. Finally, in the third stage, FGM calculates the final heading angle ( $\varphi_{final}$ ), using Eq. (1). The weighted function mentioned in Eq. (1) consists of the angle to the largest gap’s center point ( $\varphi_{gap-c}$ ), the angle to the goal point ( $\varphi_{goal}$ ), the distance to the closest obstacle ( $d_{min}$ ), and a safety factor named alpha ( $\alpha$ ). The alpha parameter’s higher values cause the robot to keep its distance from the obstacles and follow the safe gap’s center. In contrast, alpha’s lower values cause the robot to follow the goal point and get too close to obstacles in some cases. These changes in alpha cause the robot to travel on different trajectories, as shown in Fig. 1. The alpha is a user-defined parameter as it is defined in the original paper. Furthermore, it can be noticed in Fig. 1 that the selection of alpha is entirely free and practical, so to continue, we will select  $\alpha = 40$  in this paper.

$$\varphi_{final} = \frac{\frac{\alpha}{d_{min}} \varphi_{gap-c} + \varphi_{goal}}{\frac{\alpha}{d_{min}} + 1} \quad (1)$$

The visualization of the available gaps around the robot, the angle to the midpoint of the largest gap, the angle to the final goal, and the FGM’s final heading angle are shown in a robot-obstacle configuration represented in Fig. 2.

Furthermore, although FGM was selected as the most effective and safest obstacle avoidance method in a study by Zohaib et al., [29] it faces a wide range of inherent drawbacks, including the zigzag and the trajectory length problem which were improved by FGM-I. [30] Another con of this method is that FGM puts the robot at risk and causes some safety problems using the simple gap selection and final heading angle calculation procedures simultaneously. The next part of this study will cover this drawback in detail and provide a solution by proposing FOCM.

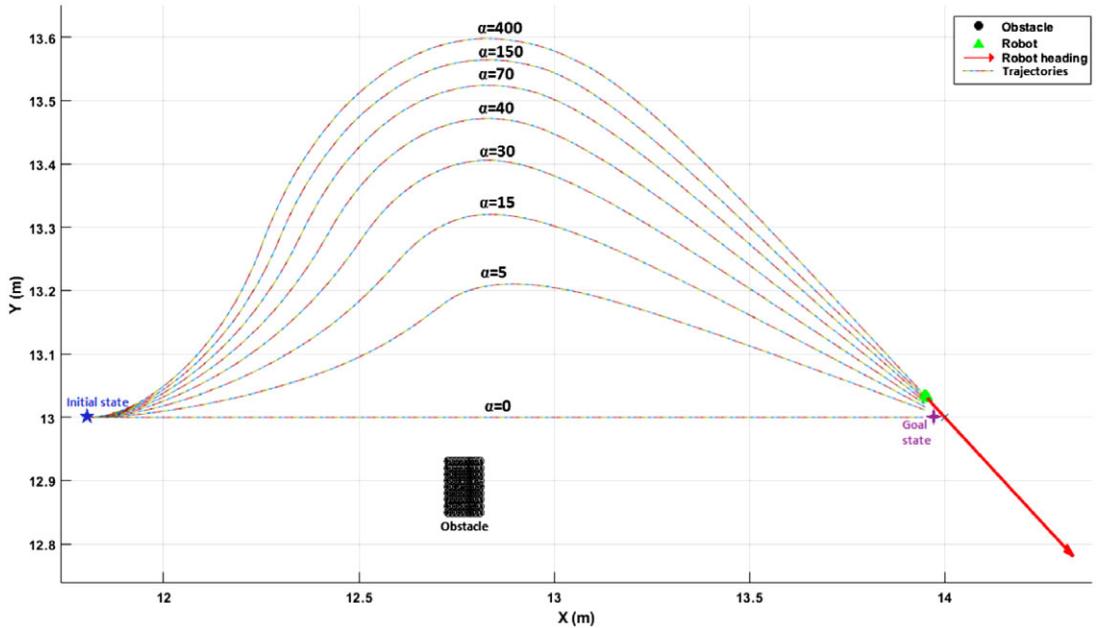


Figure 1. Visualization of robot's different paths based on different values of  $\alpha$ .

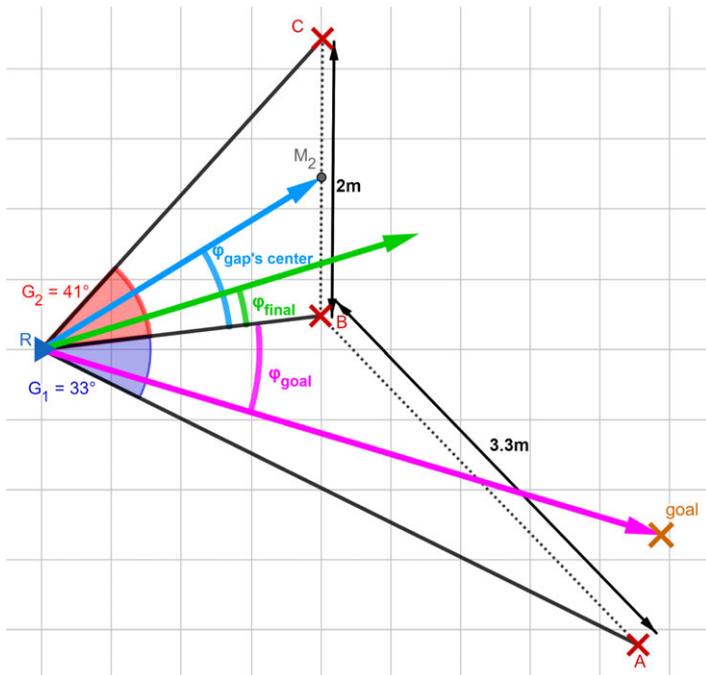
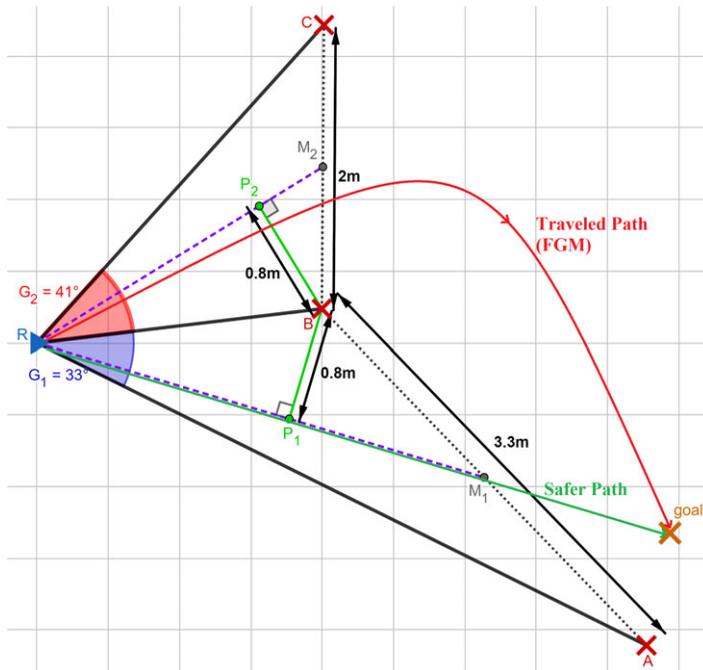


Figure 2. Robot-obstacle configuration, obstacles (A, B, and C), gaps ( $G_1$ ,  $G_2$ ), the midpoint of the widest angular gap ( $M_2$ ), goal point (X), angle to the goal point ( $\varphi_{goal}$ ), final heading angle ( $\varphi_{final}$ ), and angle to the largest gap's center point ( $\varphi_{gap-c}$ ).



**Figure 3.** Robot-obstacle configuration, where selecting the largest gap by FGM causes the unsafe path in red color. A, B, and C are the obstacles.  $M_1, M_2$  are the center of the  $G_1$  and  $G_2$  gaps, respectively, and the goal point is represented by (X).

### 3. Follow the obstacle circle method

FGM is developed to keep the robot safe during travel. This method’s primary part is calculating the final heading angle based on the angle to the gap’s center. In the gap selection stage, the method selects the desired gap by comparing the gaps based on their size in angle, and it does not consider the gaps’ width (in the distance). Although comparing the gaps using the angles is practical, in some cases, it makes the robot travel on unsafe trajectories. An example of this configuration can be seen in Fig. 3, where the FGM puts the robot in a perilous path by selecting the largest gap ( $G_2$ ) among the other available gaps. As shown in Fig. 3, the robot gets too close to obstacle B compared to the safer path shown in green color. This inherent problem cannot be solved by just modifying the FGM’s gap selection procedure because regardless of which gap has been selected, FGM calculates the angle to the gaps’ center point, and the robot has to navigate to that center point straightforwardly for crossing the gap. In other words, FGM uses the heading lines, which start from the robot’s current position and end on the gaps’ center points, so it is possible for the robot to get close to the obstacles during the travel, and in some configuration puts the robot in the risk by getting it too close to the obstacles as shown in Fig. 4. As it is seen in Fig. 4, the shortest distances between the first path, which is created by selecting the largest gap, and the obstacles A and B are  $BM_1 = AM_1 = 2$  m. However, the shortest distances between the second path, which is caused by selecting the gap with the largest width, and the obstacles are  $(BP_2 = CP_3 = 1.5$  m), so selecting the gap with the largest width does not always lead the robot to safer trajectories when using the FGM. In addition to what has been stated, the safest and the optimum available path for the robot is represented in green color in Fig. 4. It is noteworthy to consider that FGM does not provide this kind of path. The solution to this problem, regardless of the gap selection procedure, requires changing the gap selection procedure and the whole mechanism of the FGM’s final heading calculation. This solution will result in a safe green path represented in Fig. 4. The newly proposed algorithm uses a new concept named “obstacle circles” to compute the final heading angle. Moreover, using this approach allows the robot to travel on safer trajectories using the same sensory information.



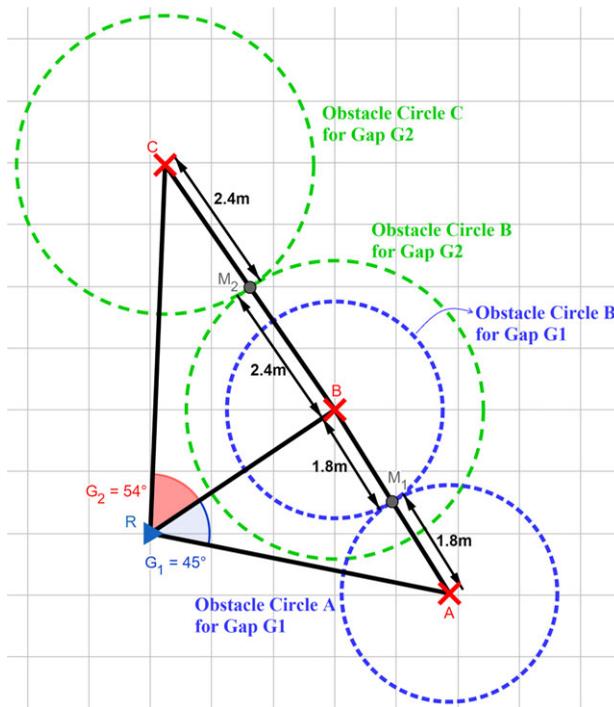
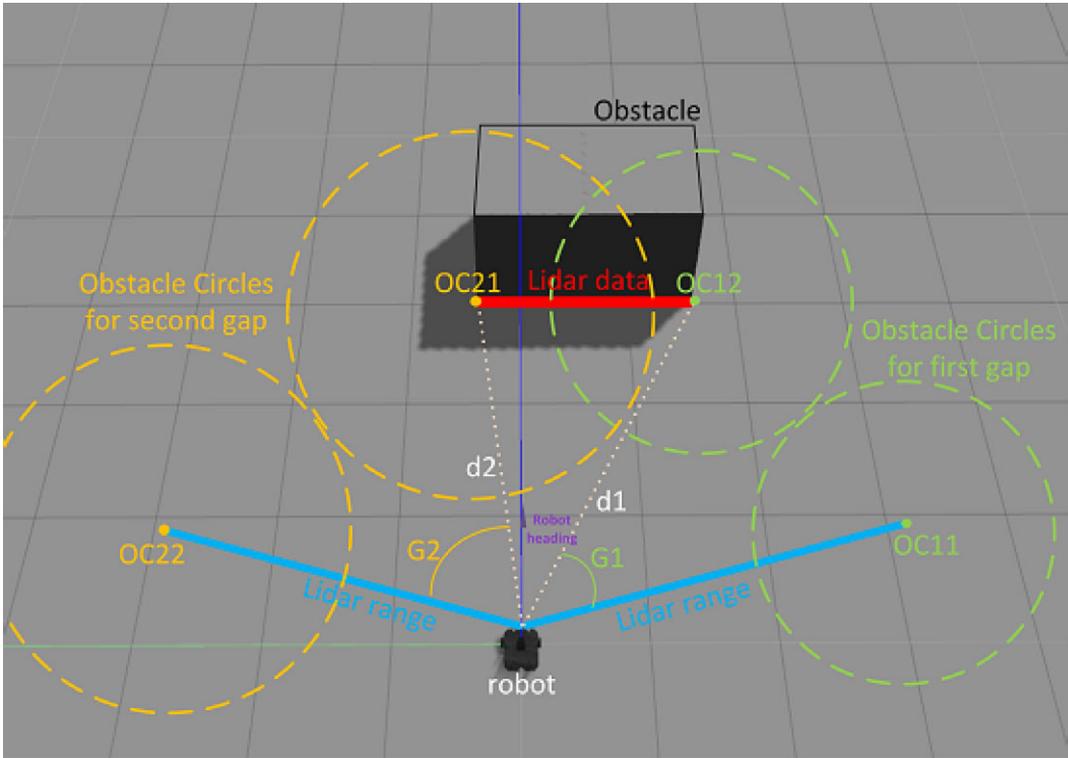


Figure 6. Obstacle circles for the obstacles A, B, and C.

through the gap baseline because the robot will have the maximum distance from obstacles there. Considering the fact that the robot will have this distance in the best situation will lead us to use the mentioned distance as a margin between the robot and obstacles, where this margin is a locus of points equidistant from obstacles in 2D that creates “obstacle circles” with the radius of the minimum distance from the obstacles to the center of each gap baseline. We will call this radius as gap radius ( $r_{gap}$ ). Considering the robot-obstacle configuration scenario shown in Fig. 6, the obstacle circles are notable in each gap. It is noteworthy to mention that the center of these obstacle circles are gap borders (the edges of obstacles); these centers can be calculated by obtaining the distance and the angle information from the robot to the obstacles’ edges (gap borders) using the sensory data. The calculation contains scanning the LIDAR data continuously and detecting the obstacle edges then converting the polar coordinates of obstacle borders to the Cartesian coordinate systems. Figure 7 represents an actual scene of the robot-obstacle configuration, and visualization of robot, the obstacle (box in black), LIDAR data in red, robot’s LIDAR range (representing LIDAR field of view [FOV]), first gap ( $G_1$  in green), second gap ( $G_2$  in gold), center of obstacle circles (OC11 and OC12 for the first gap; OC21 and OC22 for the second gap), obstacle circles for each gap, and distance from robot to the center of obstacle circles ( $d_1$  and  $d_2$ ).

Following the obstacle circle with the largest radius will guarantee the robot’s safety during the travel because it would have the most considerable maximum distance to obstacles among the other gaps, not only on the gap baseline but also everywhere on the robot-obstacles configuration. Figure 8 demonstrates the possible path that tracks the obstacle circle without getting inside the circle (Path 1) and the path created by FGM (Path 2). It is evident that by considering the mentioned obstacle circles instead of the FGM’s gap center angle, the new algorithm drives the robot to safer trajectories by keeping the robot out of the dangerous zone. The shortest path that a robot can travel with the constraint of ( $d_{min} \geq r_{gap}$ ), is Path 1, which is represented in blue color in Fig. 6, where  $d_{min}$  is the shortest distance from the robot to the obstacles in the selected gap. The calculation of Path 1 depends on the robot’s position regarding the obstacle circles, whether the robot is inside/on the obstacle circle or outside the obstacle circle. The



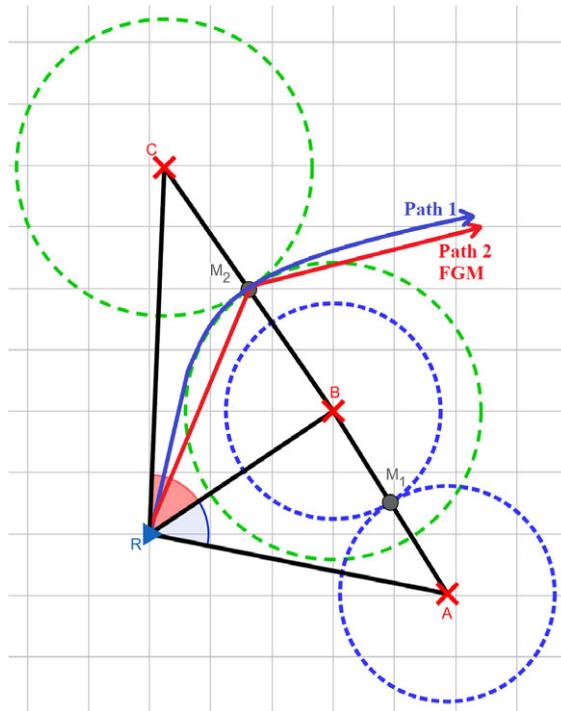
**Figure 7.** Visualization of the robot-obstacle configuration, where it shows: robot, obstacle (box in black), LIDAR data in red, robot’s LIDAR range in blue (representing the LIDAR FOV), first gap (G1 in green), second gap (G2 in gold), center of obstacle circles (OC11 and OC12 for the first gap; OC21 and OC22 for the second gap), obstacle circles for each gap, and distance from robot to the center of obstacle circles ( $d1$  and  $d2$ ).

proposed algorithm compares the minimum distance between the robot and obstacles and the gap radius ( $r_{gap}$ ) for each gap. Consequently, there are two possible cases, as follows:

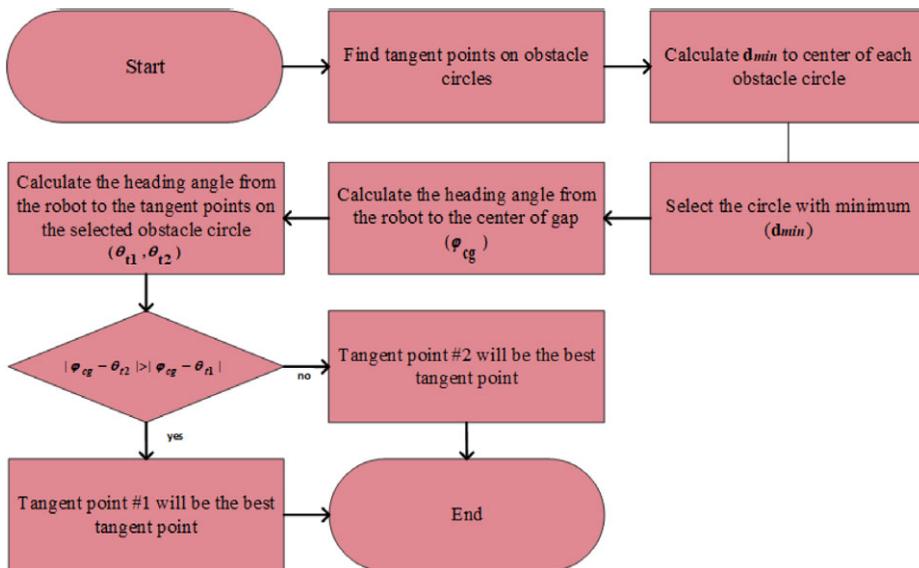
- 1- If  $d_{min}$  is more extensive than  $r_{gap}$  as shown in Eq. (2), the robot is outside the obstacle circles:

$$d_{min} > r_{gap} \tag{2}$$

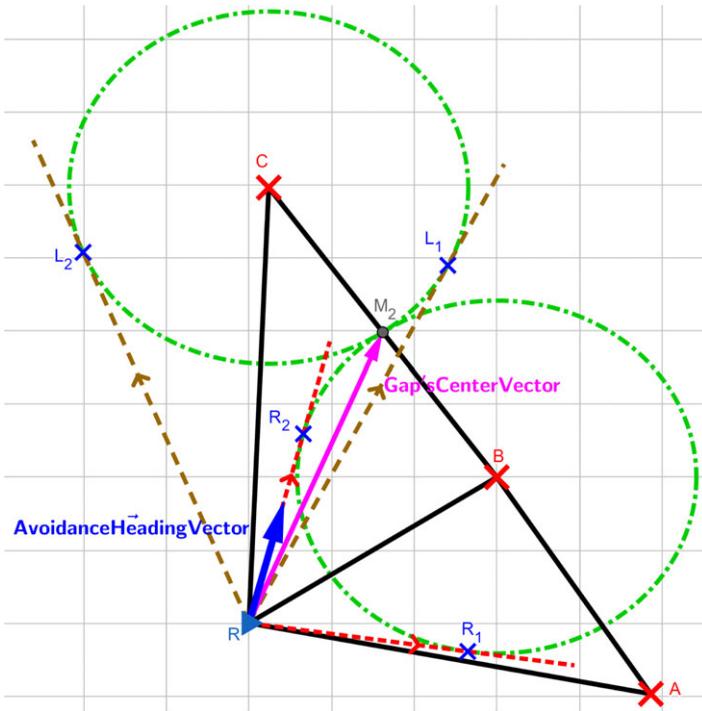
In this case, the robot will try to get itself as close as possible to the gap center with the constraint of ( $d_{min} > r_{gap}$ ). To make this maneuver possible, the algorithm calculates a heading angle named as “avoidance heading angle” ( $\varphi_{Avoid}$ ) to the best tangent point on the closest obstacle circles. It considers the distance to the center of each obstacle circle (finding the closest circle – discarding two tangent points out of four) and then compares the angle between the robot, the gap’s center point, and two tangent points. Then it selects the minimum one to find the best tangent. Figure 9 demonstrates the flowchart used for selecting the best tangent point. Figure 10 represents an example of this procedure for a robot-obstacle configuration. As it is seen in this figure, the algorithm first selects the obstacle circle  $B$  as the closest obstacle circle (selecting the tangent points  $R1$  and  $R2$ ), then it compares the angle between the gap’s center point, the robot, and the tangent points and as a result, selects the  $R2$  as the best tangent point ( $M_2\hat{R}R_2 < M_2\hat{R}R_1$ ). Also, it is notable to mention that each tangent point can be calculated by knowing the obstacle circle’s equation (center and radius) and a point outside the circle, which in our case, it is the robot’s position. Figure 11 represents a path traveled by the robot under this circumstance for the robot-obstacle configuration mentioned in Fig. 10.



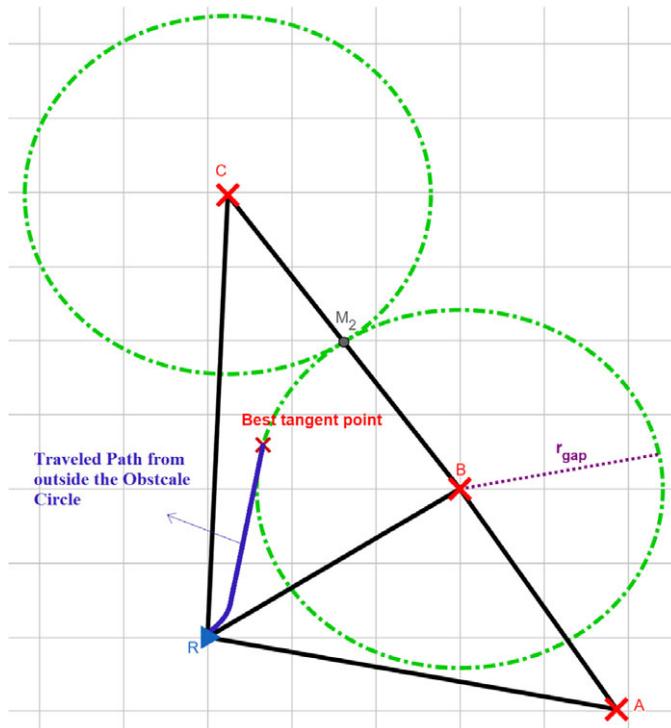
**Figure 8.** Demonstrates the possible path that tracks the obstacle circle without getting inside the circle (Path 1) and the path created by FGM.



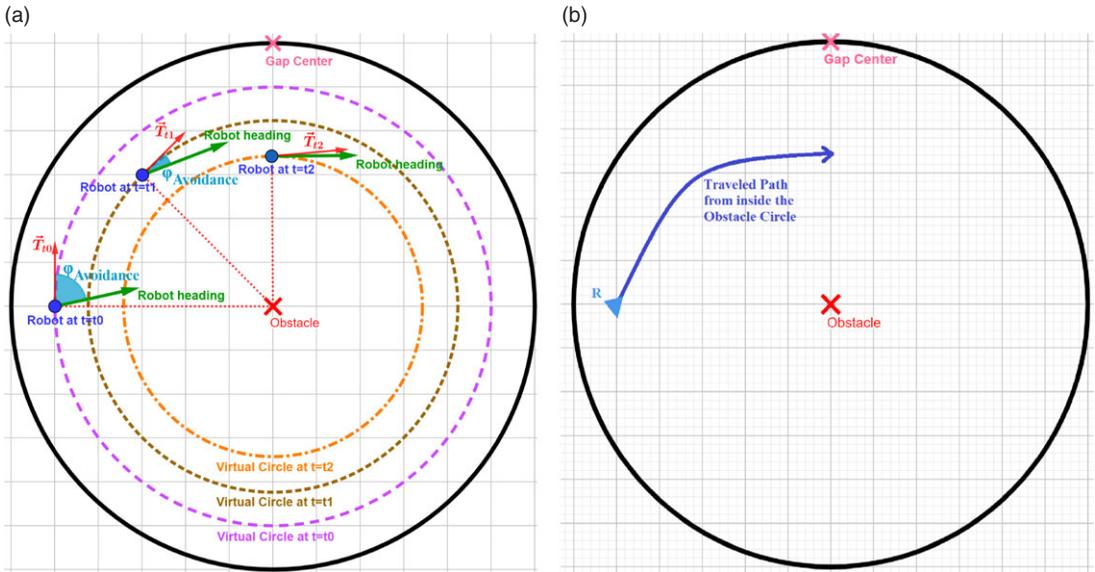
**Figure 9.** The flowchart of selecting the best tangent point.  $\varphi_{cg}$  is the heading angle from the robot to the center of the gap,  $\Theta_{t1}$  and  $\Theta_{t2}$  are the heading angles from the robot to tangent points #1 and #2.



**Figure 10.** Robot-obstacles configuration, where the  $R_2$  tangent point has been selected as the best tangent point.  $A$ ,  $B$ , and  $C$  are the obstacles.  $R_1$ ,  $R_2$ ,  $L_1$ ,  $L_2$  are the tangent points.  $M_2$  is the center of the largest gap in width.



**Figure 11.** Visualization of the traveled path of the robot from outside the circle where  $d_{\min} > r_{\text{gap}}$ .



**Figure 12.** (a): Visualization of how the robot tracks a circular arc inside the obstacle circle recursively updating its heading angle. (b): Visualization of the traveled path by the robot while tracking a circular arc, recursively updating its heading angle in the identical configuration as shown in Fig. 10(a).

2- If  $d_{min}$  is smaller than or equal to  $r_{gap}$  as shown in Eq. (3), the robot is inside or on the obstacle circles:

$$d_{min} \leq r_{gap} \tag{3}$$

In this case, the robot, whether on the obstacle circle or inside the circle, can have any heading angle, and its distance to the center of the obstacle circle is at least  $d_{min}$ . In this situation, for considering all possible cases simultaneously, the algorithm creates a virtual circle with the center of the closest obstacle coordinate and the radius of  $d_{min}$  with the same logic used for creating the obstacle circles, which was using the minimum distance from the robot to the obstacles and using this distance as the margin between them. The robot needs to update its heading angle recursively to track this curve, keeping/making itself tangent to the curve. So the algorithm calculates the difference between the robot's heading angle and the angle of the vector  $\vec{T}$ , which is perpendicular to the radius, as shown in Fig. 12(a). This difference angle would be the target heading angle ( $\varphi_{Avoid}$ ) for the robot. Figure 12(b) demonstrates the robot's traveled path while tracking circular arcs inside the obstacle circle and recursively updating its heading angle in the identical configuration as shown in Fig. 12(a).

### 3.3. Calculation of the final heading angle for each gap

After calculating the avoidance heading angle ( $\varphi_{Avoid}$ ) from the previous steps, the algorithm is now able to calculate the final heading angle. For this step, the algorithm uses the FGM's final heading angle equation as mentioned in Eq. (1). The only difference is, instead of using the angle to the gaps' center point, it uses avoidance heading angle ( $\varphi_{Avoid}$ ).

To have a clear vision about the difference between the FGM and the newly proposed algorithm FOCM, the simulation results of two different robot-obstacle configurations containing multiple obstacles are visualized in Fig. 11. As shown in Fig. 13(a), the traveled paths for both FGM and FOCM algorithms from the initial state until the moment where the robot detects the obstacle are identical. After detecting the obstacle, the FOCM forms a proper obstacle circle and passes the obstacle by tracking the circle and keeping its margin ( $r_{gap}$ ) from the obstacle. However, FGM calculates the heading angle

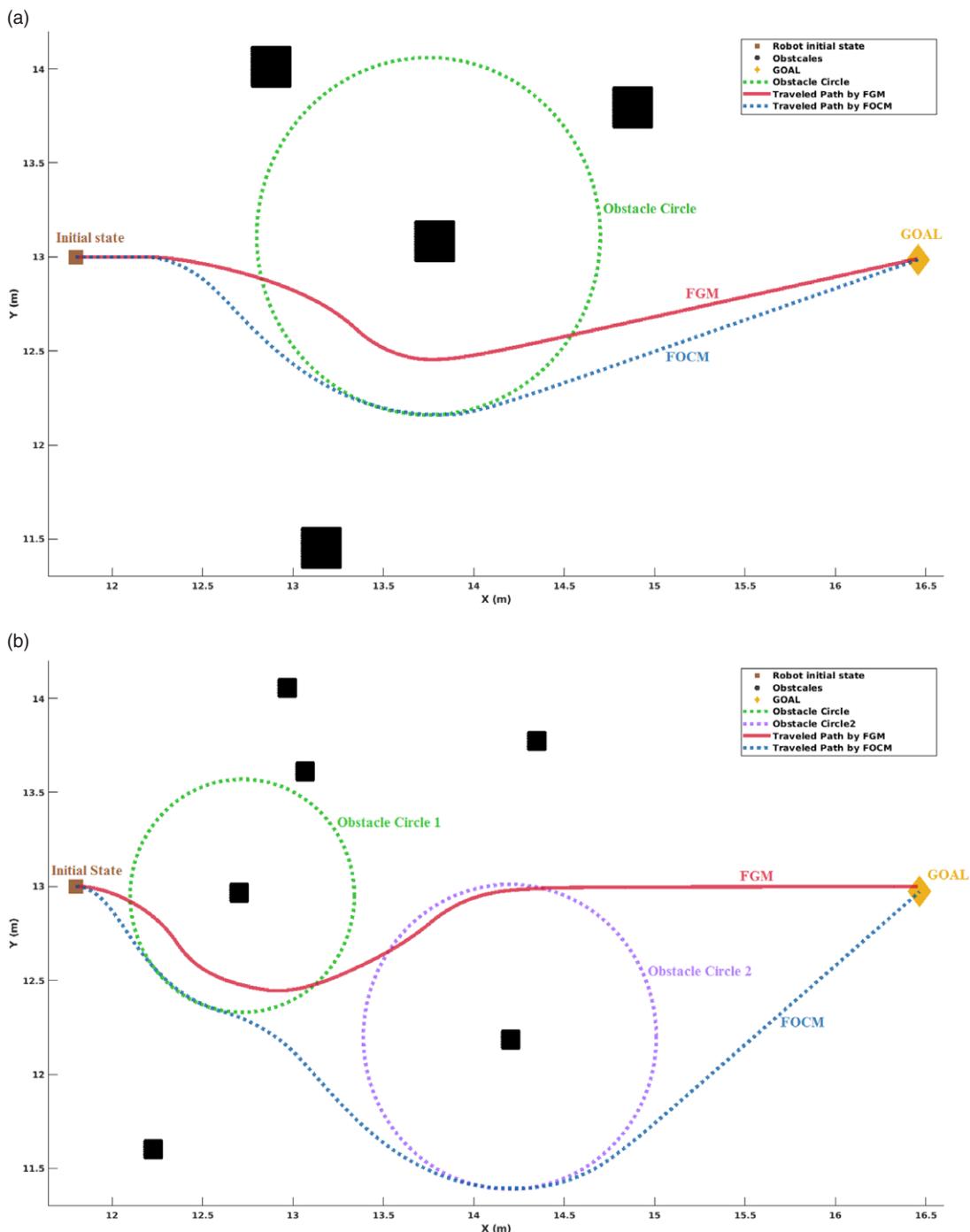


Figure 13. Comparison of the FGM and FOCM.

to the center of the largest gap instead. As a result, FGM enters the dangerous zone (inside the obstacle circle). Although both algorithms reach the target state, the path caused by FOCM is safer than the path created using the FGM. Figure 13(b) demonstrates a complex robot-obstacle configuration where the newly proposed algorithm was tested multiple times during the travel. As shown in Fig. 13(b), the robot

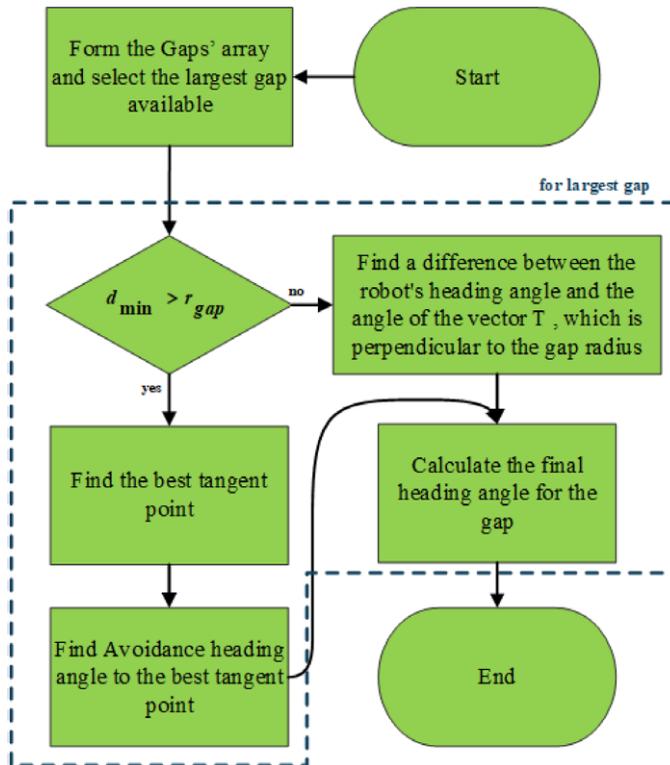


Figure 14. The flowchart of the methodology.

using the FGM entered into the obstacle circles, and consequently, it leads the robot to an unsafe path compared with the FOCM.

### 3.4. Algorithm's summary

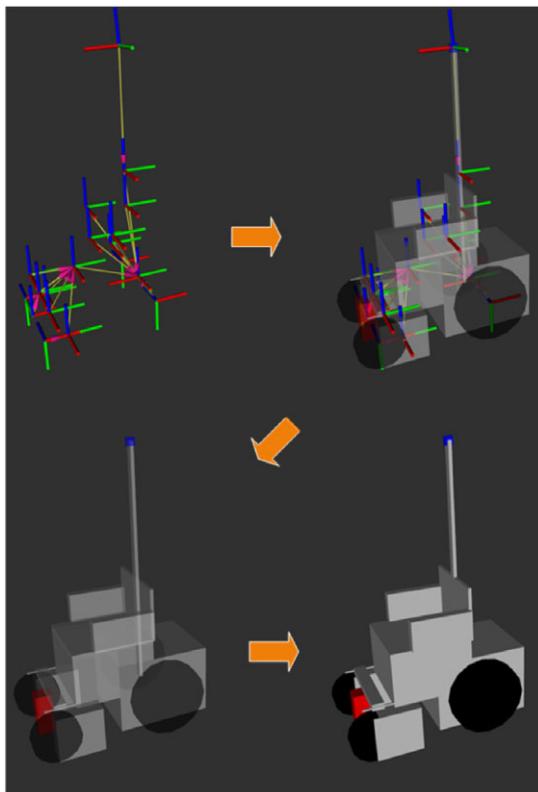
FOCM is a practical safety-based obstacle avoidance algorithm. Gap selection and avoidance strategies are improved comparing to FGM. A flowchart is represented in Fig. 14, which demonstrates the summary of the proposed methodology.

## 4. Simulation results

To have a fair comparison between FOCM and the classical FGM, Monte Carlo simulations are performed. A CAD model of differential drive wheelchair, which is identical to the real autonomous wheelchair used in experimental tests, is prepared as shown in Fig. 15 and is used in simulations. Matlab is used for developing a simulation environment with an area of 7 m × 14 m. The starting and the final points' coordinates are chosen as [11.8-13] and [16.5-13], respectively. A LIDAR sensor with a total of 180 degrees FOV is used in the simulations.

To obtain a fair benchmarking, both methods used the identical proportional and integral (PI) heading angle controller during the simulations, and the linear velocity of both methods is selected as constant 0.15 m/s. This value is selected small similar to the output of a fuzzy-logic-based longitudinal velocity planner, mentioned in ref. [35]. The mentioned heading angle controller is introduced in Eq. (4):

$$w = k_p(\varphi_{final} - \theta_{robot}) + k_i \left( \int (\varphi_{final} - \theta_{robot}) dt \right) \tag{4}$$



**Figure 15.** *Differential drive wheelchair’s CAD model.*

Where  $\theta_{robot}$  is the robot’s momentary heading angle, and  $\varphi_{final}$  is the algorithm’s final guiding angle.  $K_p$  and  $K_i$  are the PI gains, respectively. These values are tuned empirically by trial and error tests in our previous wheelchair conversion study. [36] For a fair comparison of FOCM and FGM, same controller coefficient values are used.  $K_p$  and  $K_i$  are selected as 0.3 and 0.5, respectively. Increasing the values of  $K_p$  and  $K_i$  decreases the rising time and increases the overshoot, from a classical PI controller design perspective. Although performing an impartial comparison between the two algorithms is challenging, 600 Monte Carlo simulations are performed to compare both methods. In these simulations, the obstacles are spread uniformly where the positions are set randomly. A computer with the 10th generation of i7 4.6 GHz Intel CPU and 16 GB of RAM, operating the Ubuntu 16.04 OS, was used to perform the simulations. To have a clear vision of the proposed algorithm’s efficiency, multiple performance measures such as obstacle avoidance safety metric and the travel’s average length are used. The obstacle avoidance safety metric used in this study is given in Eq. (5) as it is used in ref. [37]. The p’th norm of any  $f(t)$  function can be calculated using Eq. (6):

$$f(t) = \begin{cases} \frac{1}{d_{min}} - \frac{1}{d_0}, & \text{for } d_{min} < d_0 \\ 0, & \text{for } d_{min} \geq d_0 \end{cases} \tag{5}$$

$$\|f\|_p = (|f(t)|^p dt)^{1/p} \tag{6}$$

where  $d_{min}$  is the shortest distance between the robot and the existing obstacles, and the scalar  $d_0$  stands for the distance to an obstacle that is not dangerous for the robot. This is a user-defined analysis parameter and considering the indoor environment condition,  $d_0$  is selected as 2 m. This is a parameter for analysis of the resulting paths in terms of the danger. While increasing  $d_0$  considers unnecessary distant

**Table I.** Monte Carlo simulations result ( $\alpha = 40$ )

|      | Average safety metric | Average traveled distance (m) |
|------|-----------------------|-------------------------------|
| FGM  | 2.58                  | 4.91                          |
| FOCM | 2.27                  | 5.07                          |

measurements, decreasing  $d_0$  ignores critically close obstacles. This value can be higher for outdoor conditions. The mentioned collision avoidance safety metric is inversely proportional to the closest distance between the obstacles and the robot. In other words, it is a penalizing function that penalizes the robot when it gets closer to obstacles than the scalar  $d_0$ , a smaller average value means a safer travel. Same collision avoidance metric was previously used in refs. [28, 30, 34, 37]. Furthermore, in this study, the infinity norm ( $p = \infty$ ) of the obstacle avoidance safety metric is used for comparison.

Table I demonstrates the average of mentioned performance measures for the total 600 Monte Carlo simulations of FGM and FOCM ( $\alpha = 40, d_0 = 2$  m).

According to Table I, the newly proposed FOCM is 12.79% safer than the FGM, while the average travel distance values are almost equal. The FOCM average traveled distance is 1.71%, longer than FGM.

In order to test the statistical significance of the algorithm in performing the safety of the obstacle avoidance task, we apply Z test since the sample size (number of simulations:  $n = 600$ ) is large enough. This allows us to assume that the sampling distribution is approximately normal with the mean of FGM’s safety metric  $\mu_{FGM}$ , and its standard deviation of  $s = 2.732$ . We use the variable Z as the test statistic under the hypothesis represented in Eq. (7):

$$\begin{cases} H_0, \mu_{FOCM} \geq \mu_{FGM} & \text{(Null Hypothesis)} \\ H_1, \mu_{FOCM} < \mu_{FGM} & \text{(Alternative Hypothesis)} \end{cases} \tag{7}$$

It is noteworthy to mention that the Z value is calculated as  $-2.4995$ , using the general Eq. (8), where  $\bar{x}$  is the mean score of FOCM’s safety metric ( $\mu_{FOCM}$ ) and  $\mu$  is the mean score of the FGM’s safety metric ( $\mu_{FGM}$ ).

$$Z = \frac{\bar{x} - \mu}{\frac{s}{\sqrt{n}}} \tag{8}$$

The confidence level is chosen as  $c = 0.95$ , so the significance level is 0.05 which is a common selection in most of the tests of statistically significance. The critical Z value, which represents the boundary between rejection and acceptance of the null hypothesis  $H_0$  for the defined significance level, is  $-1.96$ . Furthermore,  $p$ -value is a number which describes how likely it is that our data would have occurred by random chance under the assumption of null hypothesis is true. In our case, for the calculated Z value  $-2.7794$ ,  $p$ -value is 0.0028. Since the calculated Z value falls outside the critical value which means that the  $p$ -value is lower than the defined significance level ( $p < \text{significance level}$ ), the  $H_0$  is successfully rejected. This means that the mean of the safety metric of FOCM is significantly lower than FGM, which shows the safety advantage of FOCM over FGM statistically. Figure 16 represents the normal distribution, Z value, and critical value.

### 5. Experimental platform

After getting the satisfying results from the simulations, FOCM is implemented on the real autonomous platform to test its real-time performance. The real system used in this study is a fully autonomous electrical wheelchair testbed, which is designed and developed by the Autonomous Mobility Group of Istanbul Technical University [36]. It is constituted with multiple sensors like three LIDARs, an RGB-D camera, an IMU sensor, and wheel encoders. The computational hardware mounted on this wheelchair includes Nvidia Jetson-TX2 and AAEON Intel Upboard for high-level operations. An embedded ST

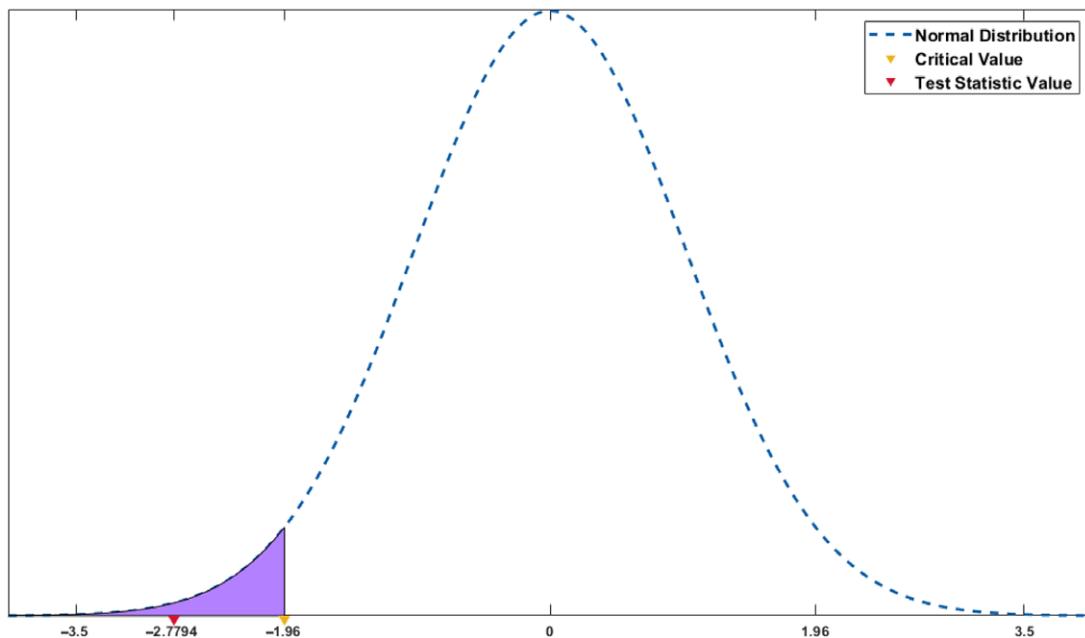
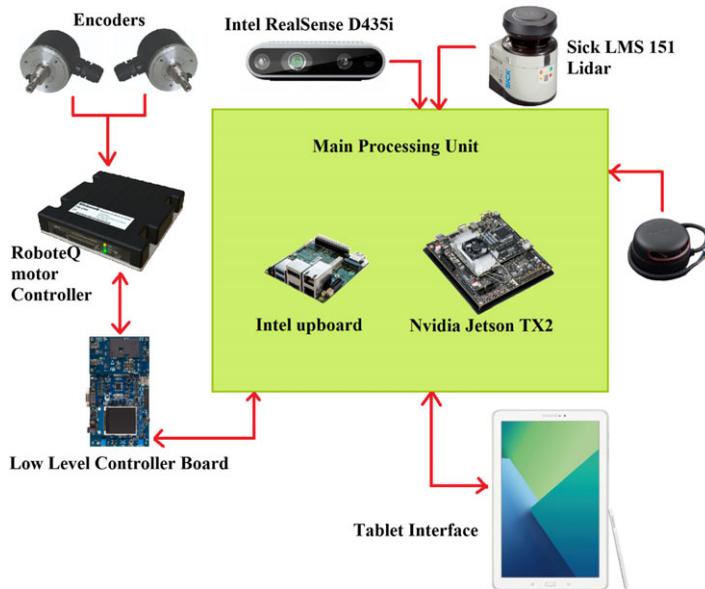


Figure 16. Visualization of the normal distribution, test statistic value, and critical value of left-tail test.



Figure 17. Visualization of fully autonomous wheelchair and its components as an experimental platform.



**Figure 18.** Visualization of the communication architecture used in the experimental platform.

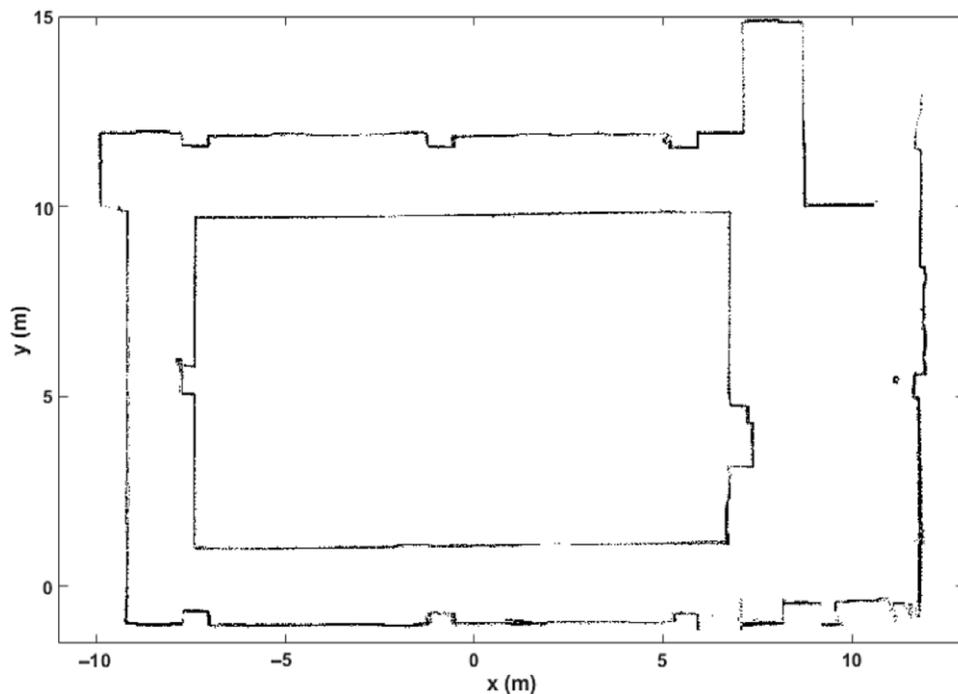
B-F446E96B01A board is used for low-level control applications. Figure 17 illustrates the wheelchair with its sensors and computational hardware. The autonomous wheelchair uses the robot operating system (ROS) as the central platform for communicating and has the ability to be controlled semi/fully autonomously and manually by an operator. Figure 18 visualizes the communication architecture used for developing this wheelchair. More detailed information about the experimental platform can be found in ref. [36].

## 6. Experimental results

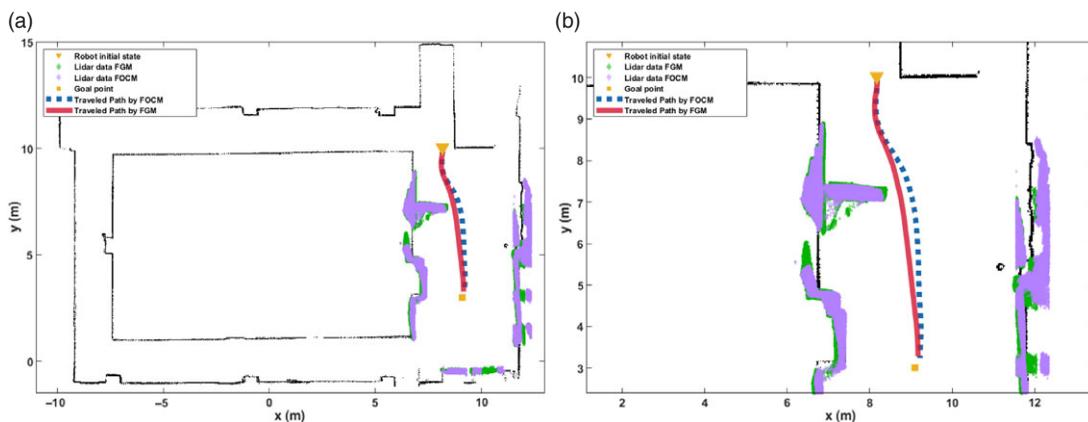
Both FOCM and FGM are coded in Python as independent ROS nodes. The tunable parameter alpha is chosen as 40 ( $\alpha = 40$ ) for both methods to have a fair comparison. The wheelchair uses identical PI heading angle controller we used during the simulations. Since the tests are mostly done in challenging environments, the linear velocity of both methods is selected similar to the output of a fuzzy-logic-based longitudinal velocity planner in such complex scenarios. [35] The SICK LMS 151 LIDAR is used for perception with a limited FOV of  $150^\circ$  and a limited maximum range of 3 m. The adaptive Monte Carlo localization method [38] is used for localization during the experimental tests. Figure 19 represents the real occupancy grid map of the Mechatronics Educational and Research Center (MEAM) of Istanbul Technical University, which is chosen as the test field. Four complex scenarios are chosen to perform the tests. The results of the first scenario of experimental tests are represented in Fig. 20. Figure 20(a) shows the initial point of the autonomous wheelchair, the paths created by both FOCM and FGM algorithms, the final point, and the obstacle measurements from the LIDAR sensor. As it can be seen in Fig. 20(b), which is the zoomed version of Fig. 17(a), the wheelchair's initial coordinate is selected as [8.2,10], and the goal point's coordinates is [9.1,3]. The path created by FGM is shown in red color, and the traveled path caused by FOCM is represented in blue color. The purple and the green points are the obstacle measurements coming from the LIDAR. The obstacle measurement clusters are not identical because of the noise and the uncertainty that the LIDAR contains during the measurement. Figure 21 shows the sequential recorded images of the wheelchair's travel during the tests. Figure 21(a) depicts the test using the FGM, whereas Fig. 21(b) illustrates the travel of the wheelchair using the newly purposed FOCM. In order to show the safety effect of the proposed approach, Fig. 22. shows the merged image of both

**Table II.** Experimental results of first scenario ( $\alpha = 40$ )

|      | Average safety metric | Average traveled distance (m) | Standard deviation of safety metric | Standard deviation of traveled distance |
|------|-----------------------|-------------------------------|-------------------------------------|---|
| FGM  | 0.5875                | 7.52                          | 0.0279                              | 0.1438                                  |
| FOCM | 0.5370                | 7.64                          | 0.0272                              | 0.1402                                  |



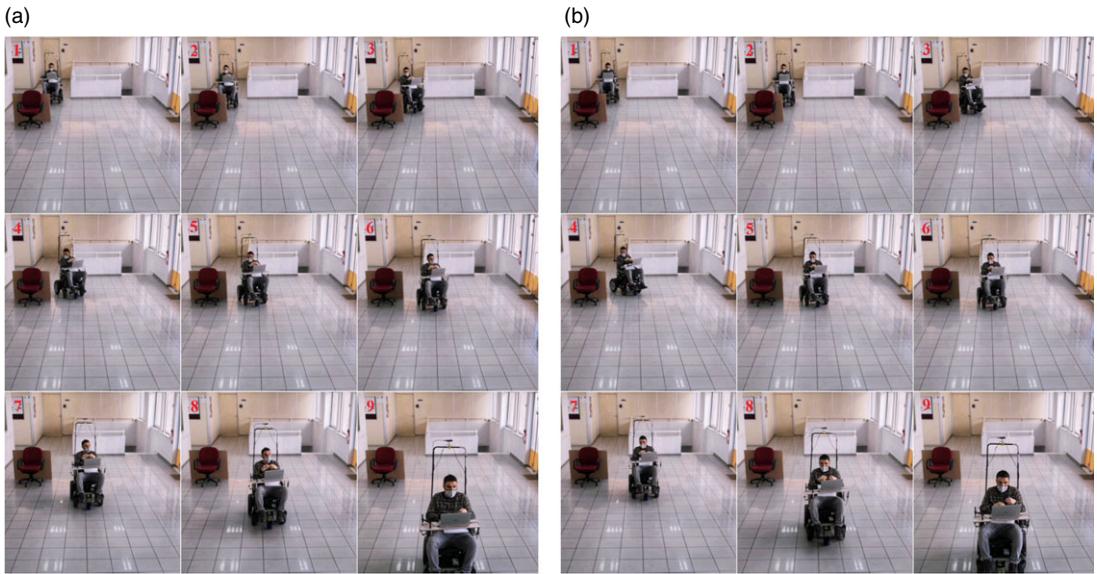
**Figure 19.** The real occupancy grid map of the test field.



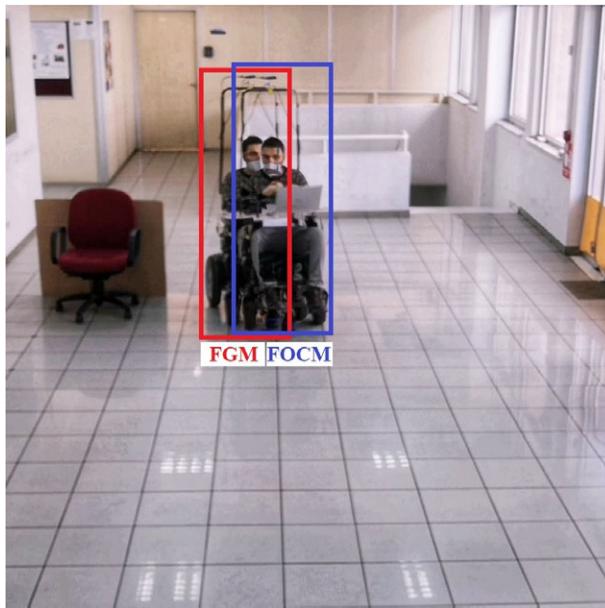
**Figure 20.** The results of the first scenario of experimental test.

algorithms at the critical moment while overtaking the obstacle. Moreover, the real video of these tests can be watched online by clicking here, or from Supplementary Material part of the submission.

Table II demonstrates the average and the standard deviation of the safety metric for the five tests of FGM and FOCM in the first scenario ( $\alpha = 40$ ,  $d_0 = 2$  m).



**Figure 21.** The recorded images of the first scenario of the experimental tests (a): The experimental test using the FGM; (b): The experimental test using the FOCM.



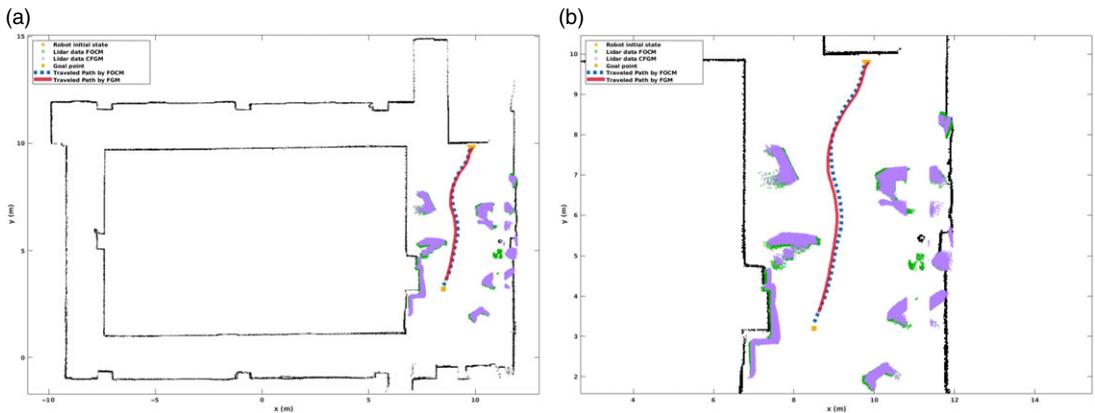
**Figure 22.** The merged image of both algorithms' critical moment.

According to the real experimental tests, as shown in Table II, the newly proposed FOCM is 8.6% safer than FGM, while the average travel distance values are almost equal. The FOCM average traveled distance is 1.57% longer than FGM.

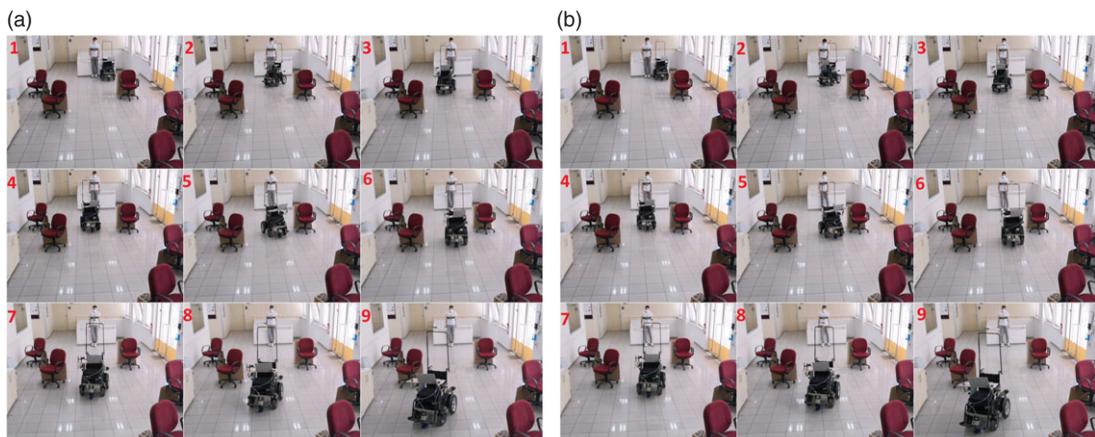
Figure 23 represents the second scenario of the experimental tests. As it can be seen in Fig. 23(b), which is the zoomed version of Fig. 23(a), the wheelchair's initial coordinate is selected as [9.8,10], and the goal point's coordinates is [8.5,3.5]. The path created by FGM is shown in red color, and the traveled path caused by FOCM is represented in blue color. The purple and the green points are the

**Table III.** Experimental results of second scenario ( $\alpha = 40$ )

|      | Average safety metric | Average traveled distance (m) | Standard deviation of safety metric | Standard deviation of traveled distance |
|------|-----------------------|-------------------------------|-------------------------------------|---|
| FGM  | 0.6413                | 7.93                          | 0.0026                              | 0.136                                   |
| FOCM | 0.5921                | 8.19                          | 0.0031                              | 0.131                                   |



**Figure 23.** The results of the second scenario of experimental test.



**Figure 24.** The recorded images of the second scenario of the experimental tests (a): The experimental test using the FGM; (b): The experimental test using the FOCM.

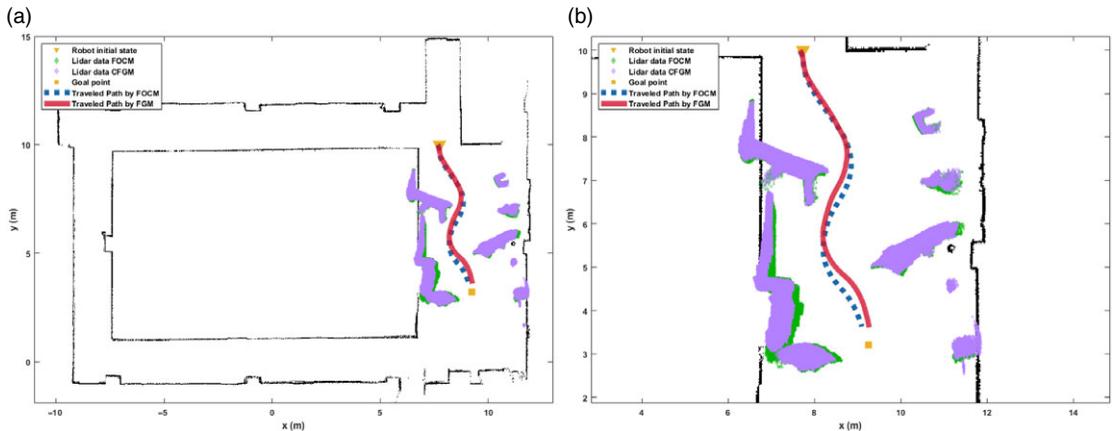
obstacle measurements coming from the LIDAR. Figure 24 illustrates the sequential recorded images of the wheelchair’s travel during the tests. Figure 24(a) depicts the test using the FGM, whereas Fig. 24(b) shows the travel of the wheelchair using the FOCM. Moreover, Table III demonstrates the average and the standard deviation of the safety metric for the five tests of FGM and FOCM in the second scenario.

According to the experimental tests, as shown in Table III, FOCM average traveled distance is 3.1% longer than FGM, and FOCM is 7.7% safer than FGM, while the average travel distance values are almost equal.

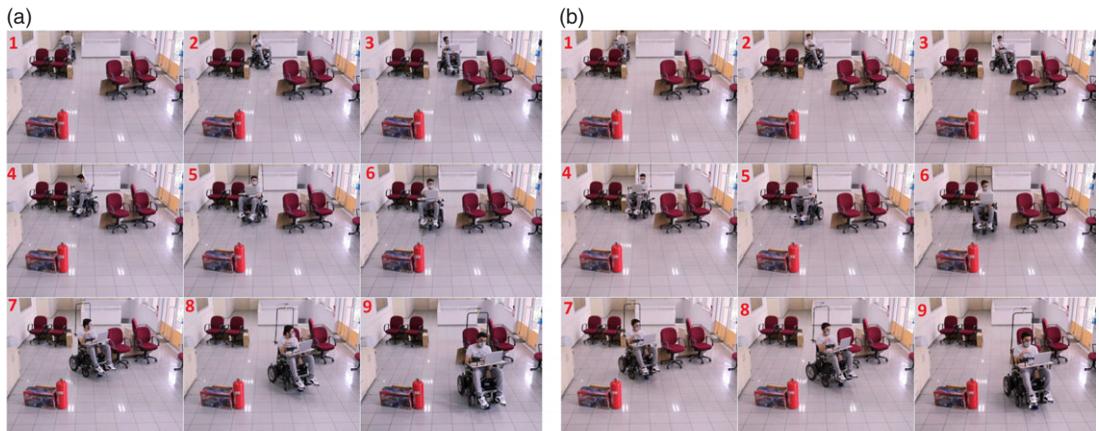
Figure 25 visualizes the third and the most complex scenario available in the MEAM. As it can be seen in Fig. 25(b), which is the zoomed version of Fig. 25(a), the wheelchair’s initial coordinate is selected as [8.2,10], and the goal point’s coordinates is [9.2,3.2]. Similar to the previous tests, the path created

**Table IV.** Experimental results of third scenario ( $\alpha = 40$ )

|      | Average safety metric | Average traveled distance (m) | Standard deviation of safety metric | Standard deviation of traveled distance |
|------|-----------------------|-------------------------------|-------------------------------------|---|
| FGM  | 0.9345                | 8.49                          | 0.0037                              | 0.156                                   |
| FOCM | 0.7961                | 8.46                          | 0.0044                              | 0.148                                   |



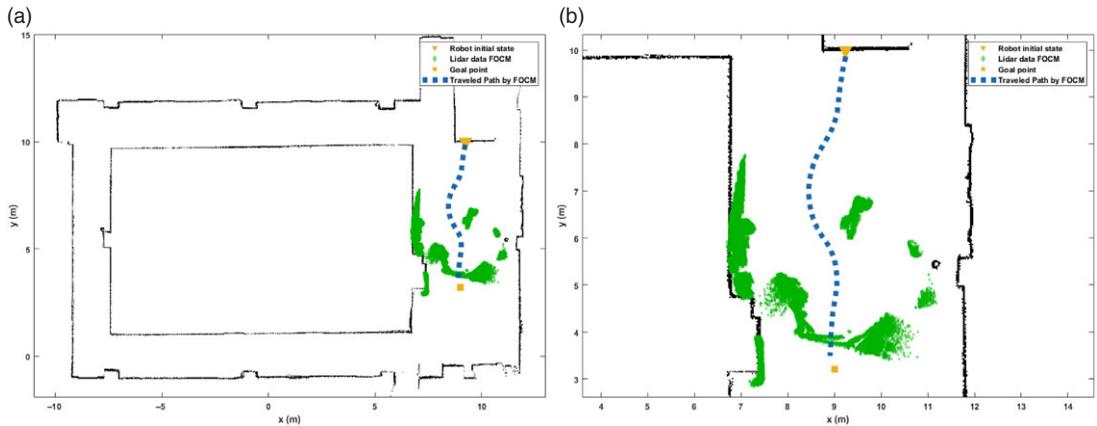
**Figure 25.** The results of the third scenario of experimental test.



**Figure 26.** The recorded images of third scenario of the experimental tests (a): The experimental test using the FGM; (b): The experimental test using the FOCM.

by FGM is shown in red color, and the traveled path caused by FOCM is represented in blue color. The purple and the green points are the obstacle measurements coming from the LIDAR. Figure 26 depicts the sequential recorded images of the wheelchair’s travel during the tests. Figure 26(a) shows the test using the FGM, whereas Fig. 26(b) shows the travel of the wheelchair using the FOCM. Furthermore, Table IV demonstrates the average and the standard deviation of the safety metric for the five tests of FGM and FOCM in the third scenario.

According to Table IV, the mean of traveled distance caused by FOCM is averagely 0.35% longer than FGM; however, FOCM leads the robot to safer trajectories, and it is 14.81% safer than FGM, while the average travel distance values are almost equal.



**Figure 27.** The results of the fourth scenario with the presence of the random dynamic obstacles, for showing the capability of FOCM to handle the scenarios with the presence of dynamic obstacles.



**Figure 28.** The recorded images of the fourth scenario of the experimental tests using FOCM with the presence of random dynamic obstacles.

Finally, comparing the FGM and FOCM with the presence of dynamic obstacles is challenging. It is not realistic to provide the same scenarios (same position, velocity, and heading configuration of dynamic obstacles for same time segments) for both algorithms. On the other hand, in order to show the capability of FOCM to handle the scenarios with the presence of dynamic obstacles, we provided the fourth test scenario represented in Fig. 27. As it can be seen in Fig. 27(b), which is the zoomed version of Fig. 27(a), the wheelchair's initial coordinate is selected as  $[9.2, 10]$ , and the goal point's coordinates are  $[9.0, 3.2]$ . The traveled path caused by FOCM is represented in blue color. The green points are the obstacle measurements coming from the LIDAR. Although the wheelchair's path and the obstacle measurements seem to intersect, they are not collisions since the robot and the obstacles are at the same place in different times. This can be seen in Fig. 28, which illustrates the sequential recorded images of the wheelchair's travel during the tests using the FOCM.

## 7. Conclusion

In this paper, a novel procedure is developed and applied to a fully autonomous wheelchair platform. The proposed methodology improves FGM's obstacle avoidance safety and brings a new solution to the collision avoidance task. In the original FGM, it is possible to select the widest angular gap which is not the safest one in terms of the width size. Moreover, heading towards the gap center for avoidance can result with paths which are passing near the obstacle. This kind of problems has been illustrated in the previous sections. In order to eliminate these problems, we propose a new mechanism for obstacle avoidance task by introducing the "obstacle circles" concept, instead of calculating the gap center vector. Additionally, we select the gaps based on their width size instead of their angular value for safer paths.

Six hundred Monte Carlo simulations are used to assess the proposed procedure's efficiency, where the results declare that the newly proposed FOCM drives the robot to safer paths comparing with the FGM, while the average traveled distance is almost the same. The results obtained from the newly proposed algorithm show a statistically significant difference in safety metric in comparison with classical FGM. Furthermore, multiple complex scenarios are used to perform the real-world experimental tests. These tests show the real-time performance of the proposed method which is in parallel to the simulation results.

Future studies may include improvements in behavior under dynamic obstacle scenarios and robustness under measurement uncertainty. Furthermore, comparing various obstacle avoidance algorithms and testing their potentials in multiple configurations can be another future work.

**Acknowledgements.** This work was supported by the Turkish Scientific and Technological Research Council (TUBITAK) under project no 118E809.

**Supplementary material.** To view supplementary material for this article, please visit <https://doi.org/10.1017/S0263574721001624>.

## References

- [1] A. Chakravarthy and D. Ghose, "Obstacle avoidance in a dynamic environment a collision cone approach," *IEEE Trans. Syst. Man Cybern. Part A Syst. Humans* 28(5), 562–574 (1998).
- [2] K. Fujimura, *Motion Planning in Dynamic Environments* (Springer-Verlag, New York, 2012).
- [3] P. E. Hart, N. J. Nilsson and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.* 4(2), 100–107 (1968).
- [4] L. E. Kavraki, P. Svestka, J.-C. Latombe and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Rob. Autom.* 12(4), 566–580 (1996).
- [5] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik* 1(1), 269–271 (1959).
- [6] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning", *Technical report*, Computer Science Department, Iowa State University (1998).
- [7] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Rob. Res.* 30(7), 846–894 (2011).
- [8] R. Siegwart, "*Nourbakhsh, Introduction to Autonomous Mobile Robots*," *hc alk. paper*, vol. 1 (2004) pp. 13–36.
- [9] F. Aurenhammer, "Voronoi diagrams—a survey of a fundamental geometric data structure," *ACM Comput. Surv. (CSUR)*, 23(3), 345–405 (1991).
- [10] T. Lozano-Perez and M. A. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Commun. ACM* 22(10), 560–570 (1979).
- [11] H. M. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, S. Thrun and R. C. Arkin, *Principles of Robot Motion Theory, Algorithms, and Implementation* (MIT Press, 2005).
- [12] V. Lumelsky and A. Stepanov, "Dynamic path planning for a mobile automaton with limited information on the environment," *IEEE Trans. Autom. Control* 31(11), 1058–1063 (1986).
- [13] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," In: *Autonomous Robot Vehicles* (Springer, 1986) pp. 396–404.
- [14] Y. Koren, J. Borenstein, "Potential Field Methods and their Inherent Limitations for Mobile Robot Navigation," *ICRA*, vol. 2 (1991) pp. 1398–1404.
- [15] Q. Yao, Z. Zheng, L. Qi, H. Yuan, X. Guo, M. Zhao, Z. Liu and T. Yang, "Path planning method with improved artificial potential field—a reinforcement learning perspective," *IEEE Access* 8, 135 513–135 523 (2020). <https://ieeexplore.ieee.org/document/9146273>

- [16] M. C. Lee and M. G. Park, "Artificial Potential Field Based Path Planning for Mobile Robots Using a Virtual Obstacle Concept," *Proceedings 2003 IEEEASME International Conference on Advanced Intelligent Mechatronics (AIM 2003)*, vol. 2 (IEEE, 2003) pp. 735–740.
- [17] J. Minguez, "The Obstacle-Restriction Method for Robot Obstacle Avoidance in Difficult Environments," *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IEEE, 2005)* pp. 2284–2290.
- [18] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," *IEEE Trans. Syst. Man Cybern.* 19(5), 1179–1187 (1989).
- [19] J. Borenstein, Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," *IEEE Trans. Rob. Autom.* 7(3), 278–288 (1991).
- [20] W. Chen, N. Wang, X. Liu and C. Yang, "VFH Based Local Path Planning for Mobile Robot," *2019 2nd China Symposium on Cognitive Computing and Hybrid Intelligence (CCHI)* (IEEE, 2019) pp. 18–23.
- [21] R. Simmons, "The Curvature-Velocity Method for Local Obstacle Avoidance," *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 4 (IEEE, 1996) pp. 3375–3382.
- [22] D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Rob. Autom. Mag.* 4(1), 23–33 (1997).
- [23] X. Zhong, J. Tian, H. Hu and X. Peng, "Hybrid path planning based on safe a algorithm and adaptive window approach for mobile robot in large-scale dynamic environment," *J. Intell. Rob. Syst.* 99(1), 65–77, (2020).
- [24] M. Keller, F. Hoffmann, C. Hass, T. Bertram and A. Seewald, "Planning of optimal collision avoidance trajectories with timed elastic bands," *IFAC Proc. Vol.* 47(3), 9822–9827 (2014).
- [25] F. Ulbrich, D. Goehring, T. Langner, Z. Boroujeni and R. Rojas, "Stable Timed Elastic Bands with Loose Ends," *2017 IEEE Intelligent Vehicles Symposium (IV)* (IEEE, 2017) pp. 186–192.
- [26] 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, 156 787–156 803 (2019). <https://ieeexplore.ieee.org/abstract/document/8884165>
- [27] U. Orozco-Rosas, O. Montiel and R. Sepúlveda, "Mobile robot path planning using membrane evolutionary artificial potential field," *Appl. Soft Comput.* 77, 236–251 (2019). [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1568494619300420>
- [28] V. Sezer and M. Gokasan, "A novel obstacle avoidance algorithm "follow the gap method," *Rob. Auton. Syst.* 60(9), 1123–1134 (2012).
- [29] M. Zohaib, M. Pasha, R. Riaz, N. Javaid, M. Ilahi and R. Khan, "Control strategies for mobile robot with obstacle avoidance," arXiv preprint arXiv:1306.1144, 2013.
- [30] M. Demir and V. Sezer, "Improved Follow the Gap Method for Obstacle Avoidance," *2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)* (IEEE, 2017) pp. 1435–1440.
- [31] A. Ozdemir and V. Sezer, "A Hybrid Obstacle Avoidance Method Follow the Gap with Dynamic Window Approach," *2017 First IEEE International Conference on Robotic Computing (IRC)* (IEEE, 2017) pp. 257–262.
- [32] M. D. Demir and V. Sezer, "Autonomous Overtaking Maneuver Design Based on Follow the Gap Method," *Proceedings of the 16th International Conference on Informatics in Control, Automation and Robotics - Volume 1 ICINCO, INSTICC (SciTePress, 2019)* pp. 295–302.
- [33] E. cCakmak, S. Tekin, A. Ozdemir and S. Boosyan, "Gap Based Novel Approach for Safe and Fast Obstacle Avoidance for Autonomous Platforms," *2020 IEEE 29th International Symposium on Industrial Electronics (ISIE)* (IEEE, 2020) pp. 1392–1397.
- [34] H. Houshyari and V. Sezer, "A new gap selection strategy for follow the gap method," *Int. J. Mech. Eng. Rob. Res.* 10(5), 248–254 (2021).
- [35] V. Sezer, Z. Ercan, H. Heceoglu, S. Bogosyan and M. Gokasan, "A New Fuzzy Speed Planning Method for Safe Navigation," *2012 IEEE International Conference on Vehicular Electronics and Safety (ICVES 2012)* (2012) pp. 381–386.
- [36] V. Sezer, R. S. Zengin, H. Houshyari and M. C. Yilmaz, "Conversion of a Conventional Wheelchair into an Autonomous Personal Transportation Testbed," In: *Service Robotics* (IntechOpen, 2020).
- [37] S. Nair and M. Kobilarov, "Collision Avoidance Norms in Trajectory Planning," *Proceedings of the 2011 American Control Conference* (2011) pp. 4667–4672.
- [38] F. Dellaert, D. Fox, W. Burgard and S. Thrun, "Monte Carlo Localization for Mobile Robots," *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, vol. 2 (1999) pp. 1322–1328.