

Robust Collision Avoidance for Autonomous Mobile Robots in Unknown Environments

Muhammad Mujahed, Dirk Fischer, and Bärbel Mertsching

GET Lab, University of Paderborn,
Pohlweg 47-49, 33098 Paderborn, Germany
{mujahed,fischer,mertsching}@get.uni-paderborn.de
<http://getwww.uni-paderborn.de/>

Abstract. This paper presents a new collision avoidance method for mobile robots operating in unknown cluttered environments. The proposed method computes the steering angle based on the location of all obstacles surrounding the robot, not just the closest one. Hence, our technique is capable of generating smooth robot trajectories, particularly for unstructured environments. Moreover, the stability of the robot's motion is improved by providing a smoother bridge between avoiding obstacles and approaching the goal. Oscillations occurring in narrow corridors are reduced by considering the distribution of obstacles to both sides of the direction of motion. Simulation and experimental results are presented to demonstrate the performance of the proposed approach.

1 Introduction

Nowadays, applications such as exploration, search and rescue, or mining require autonomous mobile robots to perform difficult or dangerous tasks without human intervention. In such applications, the environment is usually unknown and changes over time. Moreover, unpredictable obstacles may block the robot's trajectory during mission operations. In order to guarantee a safe navigation in these environments, it is necessary to incorporate the sensory perceptions within the motion planning and the control loop. By this means, robots can detect environmental changes and re-plan dynamically to reach a goal safely. This is achieved by reactive obstacle avoidance navigation techniques.

The earlier obstacle avoidance methods for mobile robots as well as for manipulators were based on the concept of Artificial Potential Field APF (e.g. [1] [2]). Within this concept, obstacles exert a repulsive force while the target asserts an attractive force onto the robot. The vector sum of these forces determines the steering angle. APF methods are considered fast and computationally efficient. However, failure to find an oscillation-free motion and getting stuck in local minima are well-known problems in these methods. Recent works try to overcome such drawbacks, such as [3], by using a modified Newton method, and [4], by employing a family of 2D smooth vector fields.

Other common techniques add constraints, coming from physical limitations and sensory data, to the velocity space, and choose the speed that maximizes an

objective function and satisfies all constraints. The Dynamic Window [5] [6] and Curvature Velocity [7] [8] are popular methods that follow this strategy. Despite the fact that these methods drive the robot faster with smoother behavior, they are unsuitable for cluttered environments as they fail to drive robots in between closely spaced obstacles. Moreover, they still can be trapped in local minima.

Several methods are based on the concept of Velocity Obstacles VO [9] [10] or Inevitable Collision States ICS [11] [12]. These methods perform collision avoidance by identifying the set of robot's velocities/states that may cause collision at a future time, and choose velocities outside of this set. A VO variant, RVO [13] [14], deals with the problem of cooperative obstacle avoidance. Although these methods explicitly consider the velocity of obstacles, they assume known or predictable future of the scene which is hard to achieve in real applications [15]. Additionally, it is difficult to determine the suitable time horizon for VO-based techniques, particularly in cluttered environments [16].

The Nearness-Diagram ND approach [17] and its improved version ND+ [18] use a "divide and conquer" strategy to identify the current navigational situation and then generate the corresponding motion law. These methods can successfully drive the robot in dense and cluttered environments and avoid local trap situations. The Smooth Nearness-Diagram SND method [19] was then developed enhancing the smoothness of the paths generated by ND+. Unfortunately, SND can be trapped in narrow corridors if one of its sides has a large number of obstacle points compared to the other. The Closest Gap CG method [20] addresses this problem by providing a stricter deviation against the nearest obstacles and by considering the ratio of threats on the both sides of the robot's heading.

The above mentioned ND variants avoid collisions with nearby obstacles using an idea inspired from the APF concept, which is likely to cause an oscillatory robot motion in narrow passages. This may also lead to longer trajectories and more execution time. The Tangential Closest Gap TCG [21] was then developed solving this drawback by integrating two concepts: the Closest Gap for extracting openings surrounding the robot and the Tangential Escape [22] for reactive collision avoidance navigation. A similar approach [23] solves this problem by taking into account the angular width of the chosen gap, as seen by the robot, and the closest obstacle in collision with the path towards this gap. This approach has been enhanced in [24] by considering all colliding obstacle points.

In this paper, we introduce the TCG+ navigation approach which is an enhancement of the TCG method. Compared to the TCG, our new approach takes into account the location of all close obstacle points in determining the steering angle, rather than the closest one. As a consequence, a smoother robot motion is achieved. Furthermore, the TCG+ provides a smoother bridge between avoiding obstacles (circumnavigating an obstacle) and approaching the goal (satisfying the leaving condition), resulting in a more stable behavior. Considering the distribution of obstacles to both sides of the direction of motion reduces oscillations occurring in narrow passages. The ability to generate smooth and stable robot trajectories reduces the possibility of wheel skidding over ramps, and thus is extremely beneficial in performing RoboCup rescue missions. The TCG+ is de-

played as the reactive layer for our autonomous robot GETbot in the RoboCup Rescue Robot League competitions.

This paper presents the TCG+ reactive collision avoidance method design in section 2. In sections 3 and 4, we show and discuss the simulation and experimental results. Subsequently, in section 5 the performance of the proposed TCG+ approach is evaluated. Finally, section 6 highlights our conclusions.

2 Reactive Collision Avoidance Technique

This section presents the TCG+ collision avoidance method for mobile robots operating in unknown cluttered environments which serves as a reactive planner in a hybrid navigation system. The TCG+ approach works as follows: first, the sensory data is analyzed to determine the structure of obstacles surrounding the robot. Based on this analysis, an instantaneous waypoint which makes progress towards the goal is located in a collision-free area as described in section 2.2. The location of the waypoint is then adjusted to avoid nearby obstacles as explained in section 2.3.

2.1 Definitions and Notations

The robot and goal locations are denoted by \mathbf{p}_r and \mathbf{p}_g , respectively. The robot is wrapped into a circle whose radius is denoted by R . A scan point is denoted by \mathbf{p}_i , $i = 1, \dots, n$. The polar coordinates of \mathbf{p}_i are denoted by (r_i, θ_i) .

For any two angles (θ_1, θ_2) , the minimum angular distance between them is $\text{dist}(\theta_1, \theta_2) = \min(\text{dist}_c(\theta_1, \theta_2), \text{dist}_{cc}(\theta_1, \theta_2))$, where $\text{dist}_c(\theta_1, \theta_2) = (\theta_1 - \theta_2) \bmod 2\pi$ and $\text{dist}_{cc}(\theta_1, \theta_2) = (\theta_2 - \theta_1) \bmod 2\pi$.

In order to normalize an angle θ to the range $[-\pi, \pi]$, a projection function is defined as:

$$\text{proj}(\theta) = ((\theta + \pi) \bmod 2\pi) - \pi \quad (1)$$

Assume that $a < b$, a saturation function is defined as:

$$\text{sat}_{[a,b]}(x) = \begin{cases} a, & \text{if } x \leq a \\ x, & \text{if } a < x < b \\ b, & \text{if } x \geq b \end{cases} \quad (2)$$

2.2 Locating a Waypoint

If the direct path towards the goal is blocked¹, it is required to drive the robot towards an intermediate goal, referred to as a *waypoint*, rather than towards the goal itself. Locating the waypoint \mathbf{p}_{cg} is based on studying the distribution of obstacles surrounding the robot: First, the sensory data is searched for gaps which indicate potential free areas where the robot fits through. Several methodologies

¹ The direct path towards the goal is blocked if the line segment connecting the robot to it intersects an obstacle in the configuration space.

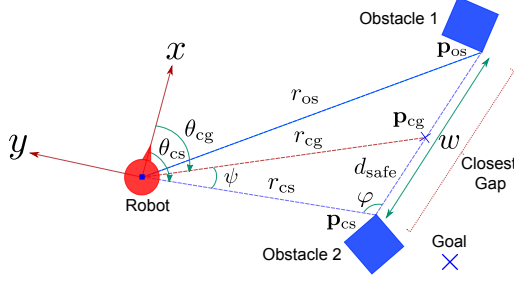


Fig. 1: Locating the closest gap waypoint.

can be found in the literature for extracting gaps (e.g. [17], [20]). Here, we follow the CG method [20] due to its simplicity and computational efficiency. The *navigable* gap closest to the goal, called *closest gap*, is then selected to navigate through. At each time instance, the gap is characterized by two points (sides), one of them is closer to the goal than the other. We refer to it by \mathbf{p}_{cs} and the other by \mathbf{p}_{os} (see Fig. 1). In general, \mathbf{p}_{cg} is located on the line segment connecting \mathbf{p}_{cs} to \mathbf{p}_{os} in such a way that a safe distance d_{safe} is preserved between \mathbf{p}_{cg} and \mathbf{p}_{cs} as the robot moves towards the closest gap. The value of d_{safe} is set to half of the width of the gap. However, when the gap is wide enough, it is better to limit d_{safe} so that the resultant trajectory is shorter:

$$d_{safe} = \begin{cases} R + D_s, & \text{if } \|\mathbf{p}_{cs} - \mathbf{p}_{os}\| > 2(R + D_s) \\ \frac{1}{2} \|\mathbf{p}_{cs} - \mathbf{p}_{os}\|, & \text{otherwise} \end{cases} \quad (3)$$

where D_s is a suitable clearance to be maintained between the robot and \mathbf{p}_{cs} .

Let θ_{cs} and θ_{os} be the angles between the x -axis of the robot and \mathbf{p}_{cs} and \mathbf{p}_{os} , respectively, and r_{cs} the distance between \mathbf{p}_r and \mathbf{p}_{cs} . The polar coordinates of \mathbf{p}_{cg} are defined as follows:

$$\theta_{cg} = \begin{cases} \text{proj}(\theta_{cs} - \psi), & \text{if } \theta_{cs} > \theta_{os} \\ \text{proj}(\theta_{cs} + \psi), & \text{otherwise} \end{cases} \quad (4)$$

$$r_{cg} = \sqrt{d_{safe}^2 + r_{cs}^2 - 2d_{safe}r_{cs}\cos(\varphi)} \quad (5)$$

with φ and ψ are defined by:

$$\varphi = \arccos\left(\frac{w^2 + r_{cs}^2 - r_{os}^2}{2wr_{cs}}\right) \quad (6)$$

$$\psi = \arccos\left(\frac{r_{cg}^2 + r_{cs}^2 - d_{safe}^2}{2r_{cg}r_{cs}}\right) \quad (7)$$

where r_{os} is the distance to \mathbf{p}_{os} .

2.3 Obstacle Avoidance Method

While driving the robot towards \mathbf{p}_{cg} , the TCG+ will adjust the trajectory by rotating \mathbf{p}_{cg} by an angle Γ temporarily in order to avoid the risk of collision with

nearby obstacles. In [21], the rotation angle is computed so that the robot takes the direction of the tangent to the closest obstacle point. In case of unstructured environments, the position of this point may change rapidly resulting in an abrupt change in the robot's trajectory. Sharp turns may also occur due to the transition from following an obstacle boundary to resuming the progress towards the goal and vice versa. The proposed approach solves these drawbacks by using all nearby obstacle points in computing the rotation angle. It also balances the transition between avoiding obstacles and approaching the goal.

The key idea of the TCG+ is the computation of the rotation angle caused by each of N obstacle points considered as a hazard while driving the robot towards \mathbf{p}_{cg} . An obstacle point \mathbf{p}_i is a hazard if the distance to it measured from the robot boundary is less than D_s and the angular distance between \mathbf{p}_{cg} and \mathbf{p}_i , relative to the laser scanner coordinate system is less than $\frac{\pi}{2}$, i.e.:

$$\|\mathbf{p}_r - \mathbf{p}_i\| \leq R + D_s \wedge \text{dist}(\theta_{cg}, \theta_i) \leq \frac{\pi}{2} \quad (8)$$

The TCG+ method considers dividing the workspace into two subspaces; one is to the right of the robot's direction of motion (to the right of $\overrightarrow{\mathbf{p}_r \mathbf{p}_{cg}}$) while the other is to the left. A hazard \mathbf{p}_i located in any of the two subspaces causes a rotation angle (γ_i) to \mathbf{p}_{cg} based on its position, the position of \mathbf{p}_{cg} , and the clearance to obstacles located on the other subspace. Assume that the subspace including \mathbf{p}_i is denoted by S_i and the other is \hat{S}_i . We consider the clearance to the obstacle point closest to the robot's boundary $\hat{\mathbf{p}}_c$, among those falling in \hat{S}_i , since it poses the highest risk.

$$\hat{\mathbf{p}}_c = \underset{\mathbf{p}}{\text{argmin}} \|\mathbf{p} - \mathbf{p}_r\|, \quad \mathbf{p} \in \hat{S}_i \quad (9)$$

The value of γ_i is determined so that $0.5 \|\mathbf{p}_i - \hat{\mathbf{p}}_c\|$ is maintained to whichever of \mathbf{p}_i and $\hat{\mathbf{p}}_c$ is closer to the robot boundary. By this means, the clearance to obstacles is maximized as the robot moves towards \mathbf{p}_{cg} . Moreover, oscillations occurring in narrow openings due to changing the location of the closest obstacle are reduced. If \hat{S}_i doesn't contain any hazard, γ_i is set in such a way that the robot points parallel to the tangent of \mathbf{p}_i :

$$\gamma_i = \begin{cases} \text{sat}_{[0, \chi]}(\text{dist}(\zeta, \theta_{cg})) \beta - \chi \beta, & \text{if } |\zeta - \theta_{cg}| \leq \pi \\ \chi \beta - \text{sat}_{[0, \chi]}(\text{dist}(\zeta, \theta_{cg})) \beta, & \text{otherwise} \end{cases} \quad (10)$$

with ζ , β and χ are given by:

$$\zeta = \begin{cases} \theta_i, & \text{if no hazard falls in } \hat{S}_i \\ \theta_{\min}, & \text{otherwise} \end{cases} \quad (11)$$

$$\beta = \begin{cases} 1, & \text{if } \zeta \geq \theta_{cg} \\ -1, & \text{otherwise} \end{cases} \quad (12)$$

$$\chi = \begin{cases} \frac{\pi}{2}, & \text{if no hazard falls in } \hat{S}_i \\ \arcsin\left(\frac{\frac{1}{2}\|\mathbf{p}_i - \hat{\mathbf{p}}_c\|}{r_{\min}}\right), & \text{otherwise} \end{cases} \quad (13)$$

where θ_{\min} and r_{\min} denote the angle and distance to whichever of \mathbf{p}_i and $\hat{\mathbf{p}}_c$ is closer to the robot's boundary, respectively, and θ_i the angle towards \mathbf{p}_i . The value of β determines the correct direction of escaping towards the closest gap. Having a closer look at Eq. (10), it is apparent that the value of γ_i becomes 0 if the angular distance between ζ and θ_{cg} gets greater than χ (a leaving condition). Notice that if no hazard falls in \hat{S}_i and \mathbf{p}_i is the obstacle point closest to the robot, the effect of (10) is analogous to the *virtual rotation angle* from [21].

Hazards falling within the right subspace cause *positive rotation angles*, while hazards falling in the left subspace cause *negative rotation angles*. The average positive and negative rotation angles are computed separately. This is to assign the same weight (relative importance) to hazards located on both sides of the direction of motion, even if one side has a large number of hazards compared to the other. In this regard, we avoid the problem of movement close to obstacles (or even hitting obstacles in narrow corridors) falling on the side with fewer hazards (see [20] for more details). Let N_{pos} and N_{neg} represent the number of hazards causing positive and negative rotation angles, respectively. The average positive and negative rotation angles are then defined as follows:

$$\Gamma_{\text{pos}} = \frac{\sum_{i=1}^{N_{\text{pos}}} \gamma_i}{N_{\text{pos}}}, \quad \gamma_i > 0 \quad (14)$$

$$\Gamma_{\text{neg}} = \frac{\sum_{i=1}^{N_{\text{neg}}} \gamma_i}{N_{\text{neg}}}, \quad \gamma_i < 0 \quad (15)$$

Finally, the rotation angle Γ is defined as the average of Γ_{pos} and Γ_{neg} :

$$\Gamma = \frac{\Gamma_{\text{pos}} + \Gamma_{\text{neg}}}{2} \quad (16)$$

The location of the waypoint after rotating it by Γ is denoted by $\tilde{\mathbf{p}}_g$ and computed as follows:

$$\tilde{\mathbf{p}}_g = \begin{bmatrix} \cos(\Gamma) & \sin(\Gamma) \\ -\sin(\Gamma) & \cos(\Gamma) \end{bmatrix} \mathbf{p}_{cg} \quad (17)$$

The TCG+ method considers limiting the robot's speed based on the distance to nearby obstacles [20] [21]:

$$v_{\text{cut}} = \sqrt{1 - \text{sat}_{[0,1]} \left(\frac{D_{vs} - r_c}{D_{vs}} \right)} \cdot v_{\text{max}} \quad (18)$$

where D_{vs} is a parameter that determines how much the speed is limited, v_{max} the maximum linear speed of the robot, and r_c the distance towards the obstacle point closest to the robot boundary.

For driving the robot towards $\tilde{\mathbf{p}}_g$, we use the same motion commands proposed in the TCG method [21]:

$$v = k_{\text{brake}} v_{\text{cut}} \cos(\tilde{\theta}_g) \quad (19)$$

$$w = \text{sat}_{[-w_{\text{max}}, w_{\text{max}}]} \left(k_{\text{max}} \tilde{\theta}_g + \frac{v \sin(\tilde{\theta}_g)}{\tilde{r}_g} \right) \quad (20)$$

where \tilde{r}_g and $\tilde{\theta}_g$ are the distance and angle towards $\tilde{\mathbf{p}}_g$, respectively, and w_{\max} the maximum angular speed of the robot. The value of k_{brake} in (19) is set to $\tanh(\tilde{r}_g)$, if $\tilde{\mathbf{p}}_g = \mathbf{p}_g$ (to ensure smooth braking while reaching the goal), and to 1 otherwise. The parameter k_{\max} in (20) is used to limit the angular speed of the robot. Having in mind that the maximum angular velocity in (20) corresponds to $\frac{\partial w}{\partial \theta_g} = 0$, the value of k_{\max} can be defined as follows:

$$k_{\max} = \frac{w_{\max} - \frac{1}{2}k_{\text{beak}}v_{\text{cut}}}{\frac{\pi}{4}} \quad (21)$$

3 Simulations

In this section, the differences in the behavior of the TCG and TCG+ methods are demonstrated using the stage robot simulator. They were implemented using the Robot Operating System (ROS) [25]. The simulated robot has a rectangular shape (0.52 m, 0.48 m) and works in a differential-driven mode. The maximum velocities were set to (0.5 m/s, 1.0 rad/s) while the safe distances (D_s, D_{vs}) were set to (0.7 m, 0.9 m). The sensing system adopted is a laser scanner which delivers 1100 measurements over 360° and covering a range of 10 m.

For these simulations, two environments with many narrow and curved passages were created, mimicking unstructured environments, as shown in Fig. 2.

The route chosen by the TCG method in the first scenario is shown in Fig. 2a. At the starting point, the leaving condition is fulfilled since the angular distance between the closest obstacle (on the side labeled A here) and \mathbf{p}_{cg} (falling within gap G1 created by A and B obstacles) exceeds $\pi/2$. In such a case, the robot navigates directly towards \mathbf{p}_{cg} . While navigating through G1, the distance to side B obstacles gets less than the distance to side A obstacles and \mathbf{p}_{cg} gets inside gap G2 created by B and C obstacles. At this moment, the angular distance between the closest obstacle and \mathbf{p}_{cg} is less than $\pi/2$. Therefore, the robot moves parallel to the tangent of side B obstacles until having fulfilled the leaving condition. This behavior continues until having passed all gaps. The frequent transition between following an obstacle boundary and resuming the progress towards \mathbf{p}_{cg} , and the quick variation in the location of the closest obstacle cause rapid changes in the steering angle. This can be interpreted from the large spikes visible in the velocity profile shown in Fig. 2e. The behavior of the TCG method in the second scenario is roughly similar to that of the first one as depicted in Figs. 2c and 2g.

By applying the proposed TCG+ approach, the robot managed to navigate both obstacle courses much smoother than the TCG method as can be seen from Figs. 2b and 2d. For example, see the trajectories of both methods at the points labeled 1 - 5 in Figs. 2a - 2d. We have confirmed our visualization by plotting the recorded motion commands versus time in Figs. 2f and 2h.

4 Experimental Results

The simulation results have been confirmed using our rescue mobile robot GET-bot, a skid-steering Pioneer 3-AT equipped with two laser scanners and an on-

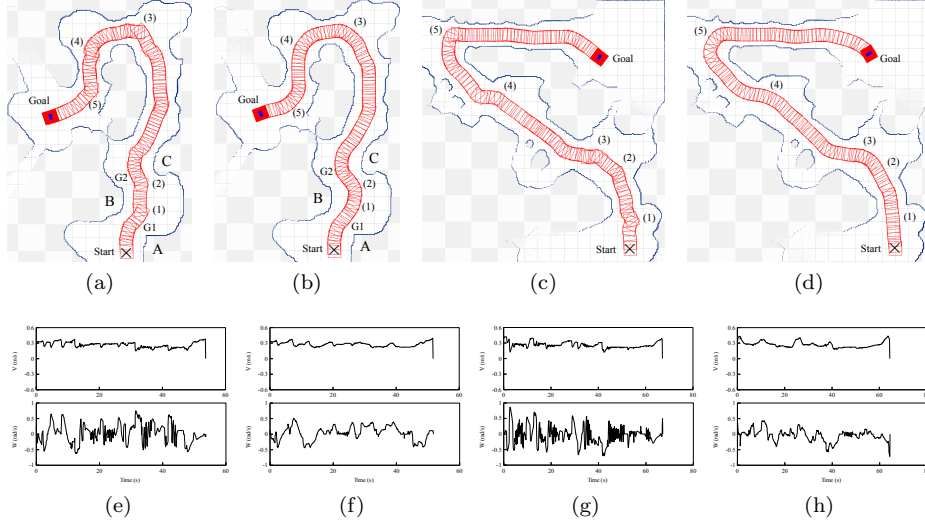


Fig. 2: Simulations. (a, b) Path followed by (a) TCG and (b) TCG+ methods in scenario 1. (c, d) Path followed by (c) TCG and (d) TCG+ in scenario 2. (e, f) Translational and rotational velocities versus time for (e) TCG and (f) TCG+ in scenario 1. (g, h) Translational and rotational velocities versus time for (g) TCG and (h) TCG+ in scenario 2.

board computer. One laser scanner is located at the front of the robot while the other is located at the back. The front laser scanner is a Hokuyo UTM-30LX having an angular resolution of 0.25° and a field of view of 270° with a maximum range of 30 m . The rear scanner is a Hokuyo URG-04LX having an angular resolution of 0.35° and a field of view of 240° covering 5.6 m . We merged the range data acquired from both laser scanners to get a 360° field of view. The robotic platform is rectangular (0.52×0.48 meters) with non-holonomic constraints. The maximum robot speeds (v_{\max} , w_{\max}) are 0.7 m/s and 2.4 rad/s , respectively. The safe distances (D_s , D_{vs}) were set to $(0.7\text{ m}, 0.9)$.

Next, we outline two experiments carried out using GETbot where the goal was the only information provided to the robot in advance². For the first experiment, we created an environment similar to the RoboCup Rescue Arena as shown in Fig. 3a. Notice that a stable and smooth navigation in such arena reduces the possibility of wheel skidding over ramps, and therefore of great importance in rescue missions. The maximum robot velocities (v_{\max} , w_{\max}) were limited to $(0.5\text{ m/s}, 1.0\text{ rad/s})$, since moving fast over ramps is risky. The route chosen for the second experiment contained places where the room available to maneuver is reduced as shown in Fig. 3d. Therefore, the maximum velocities were limited to $(0.4\text{ m/s}, 0.8\text{ rad/s})$. Both experiments have been carried out using the implementation of the TCG and TCG+ methods.

² Videos of both experiments are available at: <http://getwww.uni-paderborn.de/research/videos/tcgplus>

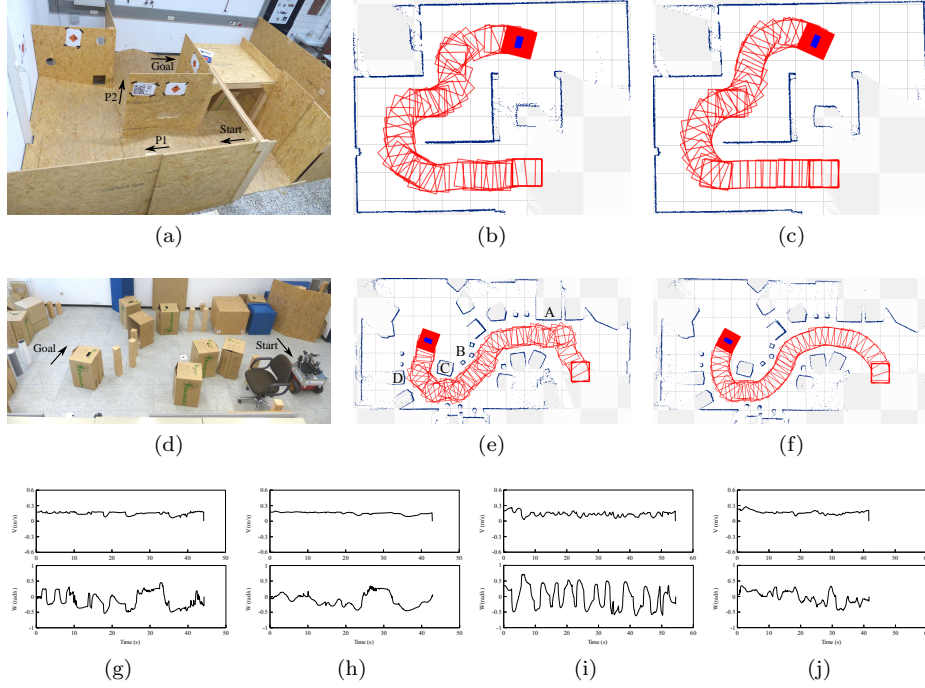


Fig. 3: Experiments. (a) Experimental setup for scenario 1. (b, c) Path followed by (b) TCG and (c) TCG+ in scenario 1. (d) Experimental setup for scenario 2. (e, f) Path followed by (e) TCG and (f) TCG+ in scenario 2. (g, h) Velocities versus time for (g) TCG and (h) TCG+ in scenario 1. (i, j) Velocities versus time for (i) TCG and (j) TCG+ in scenario 2.

The trajectories followed by the TCG method in experiments 1 and 2 are shown in Figs. 3b and 3e, respectively. The robot was able to traverse the courses of both experiments and reach the goal. However, sharp changes in the direction of motion occurred. For example, see the trajectory while traveling through the starting area (passage P1) and while leaving the passage labeled P2 in Fig 3b. The behavior was much worse in the second experiment as the motion was oscillatory almost during the complete mission as shown in Fig. 3e. Moreover, the robot moved close to the obstacles labeled A - D in the figure. This is due to the fact that the environment is composed of very narrow passages with different obstacle shapes and sizes rather than straight walls (maze-like environment) as in the first experiment. These frequent changes in the direction of motion are a result of performing obstacle avoidance based only on the closest obstacle point. In tight passages, the location of this point quickly varies as the robot moves. Moreover, the robot was switching between moving in the direction of the tangent to the closest obstacle boundary and moving directly towards \mathbf{p}_{cg} . The latter occurred whenever the leaving condition was satisfied. We recorded the linear and angular velocities over the course of the experiments. Figs. 3g and

3i show these velocities plotted versus time where the large spikes, particularly in the angular velocity profile, indicate the oscillatory motion. By taking all nearby obstacles into account and by considering the distribution of obstacles to both sides of the direction of motion, TCG+ avoided this limitation while still emphasizing the closest threat on each side (see Eq. (10)). Figs. 3c, 3f, 3h, and 3j demonstrate the increased stability and smoothness of the trajectories generated by the proposed TCG+ method.

5 Evaluation

The performance of the proposed TCG+ approach is evaluated and compared to that of the TCG method based on the following metrics:

- Execution Time (T_{tot}). The total amount of time the robot needs to reach the goal. A low execution time is desirable for better performance.
- Accumulated Jerk (J_{acc}). The jerk (third time derivative of position) reflects the abrupt changes in the forces exerted by the robot actuators. Hence, smoothness can be quantified as a function of jerk. Given the translational velocity of the robot at each time step, we define the accumulated jerk metric as follows:

$$J_{acc} = \frac{1}{T_{tot}} \int_0^{T_{tot}} [\ddot{v}(t)]^2 dt \quad (22)$$

Having less accumulated jerk is desirable as it indicates less oscillatory, more stable, and smoother behavior.

- Average Bending Energy (B_{avg}). The bending energy is a measure of the energy requirement of the robot motion, and thus can be used to assess the smoothness of the trajectory. Given the curvature k at any point along a trajectory, the average bending energy is given by [26]:

$$B_{avg} = \frac{1}{n} \int_{x_0}^{x_n} k^2(x) dx \quad (23)$$

where

$$k(x) = \frac{f''(x)}{(1 + (f'(x))^2)^{\frac{3}{2}}} dx \quad (24)$$

A low B_{avg} is preferred as it is an indication of an increased smoothness.

Based on the aforementioned metrics, a performance evaluation of the TCG+ approach was performed. Table 1 shows the results obtained for the simulations and experiments presented in sections 3 and 4. A significant improvement in the performance of the TCG+ over the TCG can be observed in all tests conducted. We believe that this enhancement is due to the fact that the TCG+ calculates Γ based on the location of all nearby obstacles and balances the transition between avoiding obstacles and approaching the goal. It also considers the distribution of obstacles to both sides of the direction of motion.

Table 1: Performance evaluation of the TCG and TCG+ methods.

Sim.	Method	T_{tot}	J_{acc}	B_{avg}	Exp.	Method	T_{tot}	J_{acc}	B_{avg}
1	TCG	54	0.543	0.176	1	TCG	44	0.155	0.935
	TCG+	51	0.073	0.071		TCG+	42	0.018	0.101
2	TCG	67	0.517	0.229	2	TCG	54	0.221	4.192
	TCG+	64	0.071	0.049		TCG+	41	0.036	0.237

6 Conclusions

This paper presents the TCG+ navigation approach for reactive obstacle avoidance. The TCG+ adapts the earlier developed Tangential Closest Gap method by making use of all close obstacle points in determining the motion direction and by providing a smoother bridge between obstacle avoidance and goal approach. The trajectories generated by the TCG+ are smoother and more stable when compared to those generated by the TCG. Additionally, taking into account the distribution of obstacles to both robot sides reduces oscillations in narrow corridors occurring in the TCG method. Simulation and experimental results demonstrated the effectiveness of the proposed approach.

References

1. O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *International Journal of Robotics Research*, vol. 5, pp. 90–98, April 1986.
2. H. Rezaee and F. Abdollahi, "Adaptive artificial potential field approach for obstacle avoidance of unmanned aircrafts," in *2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, (Kachsiung), pp. 1–6, July 2012.
3. J. Ren, K. A. McIsaac, and R. V. Patel, "Modified newtons method applied to potential field based navigation for nonholonomic robots in dynamic environments," *Robotica*, vol. 26, pp. 117–127, 2008.
4. D. Panagou, "Motion planning and collision avoidance using navigation vector fields," in *2014 IEEE International Conference on Robotics and Automation*, (Hong Kong, China), pp. 2513–2518, May 2014.
5. D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Rob. Aut. Magazine*, vol. 4, no. 1, pp. 23–33, 1997.
6. M. Seder and I. Petrovic, "Dynamic window based approach to mobile robot motion control in the presence of moving obstacles," in *IEEE International Conference on Robotics and Automation (ICRA)*, (Roma, Italy), pp. 1986–1991, April 2007.
7. R. Simmons, "The curvature-velocity method for local obstacle avoidance," in *IEEE International Conference on Robotics and Automation (ICRA)*, (Minnosota, USA), pp. 3375–3382, April 1996.
8. C. Shi, Y. Wang, and J. Yang, "A local obstacle avoidance method for mobile robots in partially known environment," *Robotics and Autonomous Systems*, vol. 58, pp. 425–434, may 2010.
9. P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The Int. J. Rob. Res.*, vol. 17, pp. 760–772, July 1998.

10. A. Wu and J. P. How, "Guaranteed infinite horizon avoidance of unpredictable, dynamically constrained obstacles," *Aut. Rob.*, vol. 32, no. 3, pp. 227–242, 2012.
11. T. Fraichard and H. Asama, "Inevitable collision states - a step towards safer robots?," *Advanced Robotics*, vol. 18, no. 10, pp. 1001–1024, 2004.
12. A. Lawitzky, A. Nicklas, D. Wollherr, and M. Buss, "Determining states of inevitable collision using reachability analysis," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, (USA), pp. 4142 – 4147, September 2014.
13. J. Berg, M. C. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *IEEE International Conference on Robotics and Automation (ICRA)*, (Pasadena, CA), pp. 1928 – 1935, May 2008.
14. D. Bareiss and J. Berg, "Generalized reciprocal collision avoidance," *The International Journal of Robotics Research*, vol. 34, pp. 1501 – 1514, April 2015.
15. J. Jin, Y. Kim, S. Wee, and N. Gans, "Decentralized cooperative mean approach to collision avoidance for nonholonomic mobile robots," in *IEEE International Conference on Robotics and Automation*, (USA), pp. 35 – 41, May 2015.
16. Z. Shiller and S. Sharma, "High speed on-line motion planning in cluttered environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Vilamoura, Portugal), pp. 596–601, October 2012.
17. J. Minguez and L. Montano, "Nearness diagram (ND) navigation: Collision avoidance in troublesome scenarios," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 1, pp. 45–59, 2004.
18. J. Minguez, J. Osuna, and L. Montano, "A "divide and conquer" strategy based on situations to achieve reactive collision avoidance in troublesome scenarios," in *IEEE International Conference on Robotics and Automation*, pp. 3855–3862, 2004.
19. J. W. Durham and F. Bullo, "Smooth nearness-diagram navigation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Nice, France), pp. 690–695, September 2008.
20. M. Mujahed, D. Fischer, B. Mertsching, and H. Jaddu, "Closest gap based (CG) reactive obstacle avoidance navigation for highly cluttered environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Taipei, Taiwan), pp. 1805 – 1812, October 2010.
21. M. Mujahed, H. Jaddu, D. Fischer, and B. Mertsching, "Tangential closest gap based (TCG) reactive obstacle avoidance navigation for cluttered environments," in *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, (Linköping, Sweden), pp. 1 – 6, October 2013.
22. A. Ferreira, F. G. Pereira, R. F. Vassallo, M. Sarcinelli-Filho, and T. F. Bastos-Filho, "An approach to avoid obstacles in mobile robot navigation: The tangential escape," *SBA. Sociedade Brasileira de Automatica*, vol. 19, pp. 395–405, 2008.
23. M. Mujahed, D. Fischer, and B. Mertsching, "Safe gap based (SG) reactive navigation for mobile robots," in *European Conference on Mobile Robots (ECMR)*, (Barcelona, Spain), pp. 325 – 330, June 2013.
24. M. Mujahed, D. Fischer, and B. Mertsching, "Smooth reactive collision avoidance in difficult environments," in *IEEE Conference on Robotics and Biomimetics (RO-BIO)*, (Zhuhai, China), pp. 1471 – 1476, December 2015.
25. M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.
26. N. Munoz, J. Valencia, and N. Londono, "Evaluation of navigation of an autonomous mobile robot," in *Proc. of Int. Workshop on Performance Metrics for Intelligent Systems Workshop (PerMIS)*, pp. 15–21, 2007.