2017 IEEE International Conference on Robotics and Automation (ICRA)
Singapore, May 29 - June 3, 2017

The Admissible Gap (AG) Method for Reactive Collision Avoidance

Muhannad Mujahed and Bärbel Mertsching

Abstract— **This paper presents a new concept, the *Admissible Gap*, for collision avoidance. An admissible gap AG is defined as the gap that a robot may safely pass through, while obeying the shape and motion constraints. By employing this concept, a new obstacle avoidance approach was developed, improving the navigation performance in unknown cluttered environments. Unlike most state-of-the-art methods, our approach explicitly considers the robot shape and kinematic constraints rather than adapting a method originally designed for a holonomic point-like robot. Experimental results demonstrated the power of the proposed AG approach. Moreover, a comparison with state-of-the-art methods showed that the AG approach generates more efficient, safer, and smoother trajectories.**

I. Introduction

One of the most important aspects of mobile robots is the capability to perform tasks in real-world environments which are often unknown and changes with time. In such environments, it is necessary to incorporate the sensory perceptions within the motion planning and the control loop. This is accomplished by reactive collision avoidance techniques.

Most of collision avoidance methods present limited capabilities in dense environments. Nevertheless, It has been shown that using a kind of high-level description of the sensory data is a successful methodology to deal with these environments [1]. Some methods follow this strategy by computing the moving direction based on the location of the free openings surrounding the robot, called *gap-based* methods [2] [3] [4] [5] [6]. However, these techniques ignore the vehicle's shape and kinematic constraints, which in turn may result in collisions or computing infeasible motions.

In order to cope with this drawback, some methods turn the holonomic solution into an admissible motion control [7] [8], relying on approximations [9]. A better solution is proposed in [9] where the workspace is mapped into another space, named Arc Reachable Manifold ARM. By applying any holonomic method in ARM, the shape and motion constraints are indirectly considered. Despite its generality, this approach has some limitations such as the assumption that configurations are only reachable by elemental circular paths. By this means, only gaps that are admissible from the current location may appear in ARM. Additionally, transforming the coordinates of ARM to comply with the kinematic constraints may result in detecting incorrect gaps.

In this paper, a new concept, the *Admissible Gap*, for real-time obstacle avoidance is proposed. A gap is admissible if it is possible to find a motion command, once executed, a robot safely traverses through it, while respecting the shape and motion constraints. With this concept, navigation in unknown cluttered environments has been successfully achieved. Unlike other gap-based techniques, the proposed approach directly considers the robot shape and kinematic constraints rather than adapting a method originally designed for a holonomic point-like robot. The basic idea is to find out the set of gaps visible from the current robot's location and select the one closest to the goal. An *admissible gap* is then constructed in such away that traversing it leads to the selected gap and a compromise between efficiency and safety is provided. Experiments in cluttered environments showed that the AG approach generates more efficient, safer, and smoother trajectories compared to state-of-the-art methods.

This paper presents the related work and some preliminary definitions in sections II and III. Section IV introduces the admissible gap concept, while section V describes how this concept is used to drive a mobile robot. In sections VI and VII, we discuss and evaluate the experimental results. Finally, section VIII highlights our conclusions and future work.

II. Related Work

Most of the obstacle avoidance methods are based on the Artificial Potential Field APF concept [10] [11]. APFs assume that the goal and obstacles exert attractive and repulsive forces and compute the steering angle based on the resultant force. Although APFs are fast and computationally efficient, they are prone to oscillations and local minima. By using a family of vector fields, a smoother behavior is achieved [12] [13]. The local minima problem was addressed in [14] by utilizing a navigation function. Unfortunately, these solutions are either restricted or based on strict assumptions [15].

The Bug algorithms [16] [17] are based on an intuitive and easy idea: the robot is simply driven towards the goal and collision avoidance is performed by circumnavigating the boundary of any encountered obstacle. With Bug algorithms, global convergence is guaranteed. However, they assume a point robot and highly depend on the sensor accuracy.

Other common methods [18] [19] [20] consider the vehicle constraints and choose a steering command rather than a moving direction. Perhaps, the Dynamic Window Approach DWA is the most well-known one [18]. Although these methods enable robots to move at higher speeds with smooth trajectory, they are limited by the parameters tuning. Moreover, they may fail to drive a robot through narrow spaces.

Some techniques are based on the concept of Velocity Obstacles VO [21] [22] [23] or Inevitable Collision States ICS [24] [25]. They avoid obstacles by characterizing the set of velocities that may lead to collision at a future time, and

M. Mujahed and B. Mertsching are with the Dept. of Electrical Engineering and Information Technology, GET-Lab, University of Paderborn, Pohlweg 47-49, D-33098 Paderborn, Germany {mujahed, mertsching}@get.uni-paderborn.de

select a velocity outside of this set. Although these methods consider the dynamics of obstacles, they assume a perfect knowledge of the scene, which is difficult to achieve [26]. Furthermore, VO-based methods require a careful setting of the time horizon, which is hard in dense environments [27].

One intriguing concept which has shown a lot of potential in mobile robotics is gaps: free areas between obstacles through which the vehicle may pass. The Nearness-Diagram ND [1] [28] was perhaps the first method based on gaps. It uses a set of criteria to identify the navigational situation and its corresponding action. Some variants [2] [29] improve the smoothness by considering the location of all obstacles. It has been proven in [6] that ND-based methods are likely to cause oscillatory robot motion and irrational deflections towards free spaces. These drawbacks have been addressed in [30] [31] by performing a tangential-based navigation. In [6], the robot's behavior in narrow spaces was further improved. Although gap-based methods enable robots to navigate through narrow spaces in cluttered environments, they are designed for holonomic point-like robots, and hence rely on approximations. In order to achieve safe navigation, it is critical to respect the exact shape and motion constraints. Addressing this drawback is the motivation of our work.

III. Preliminary Definitions

The robot and goal locations are denoted by $\mathbf{p}_r$ and $\mathbf{p}_g$. The sensor data is given as a list of points, $S = \{\mathbf{p}_1, ..., \mathbf{p}_n\}$ (scan points). The shape of the robot is approximated by a polygon with m edges denoted as $\mathcal{P}_e, e = 1, ..., m$.

An angle θ is projected to the range $[-\pi, \pi[$ as follows:

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

Let $\mathcal{F}$ be a frame centered at c and its orientation, relative to the robot's frame, is given by $\theta_{\mathcal{F}}$. The position of a point $\mathbf{p}_i$, relative to frame $\mathcal{F}$, is given as follows:

$${}^{\mathcal{F}}\mathbf{p}_i = R^{-1}(\mathbf{p}_i - c), \quad R = \begin{bmatrix} \cos(\theta_{\mathcal{F}}) & -\sin(\theta_{\mathcal{F}}) \\ \sin(\theta_{\mathcal{F}}) & \cos(\theta_{\mathcal{F}}) \end{bmatrix} \tag{2}$$

To limit x between a and b, the following function is used:

$$\text{sat}_{[a,b]}(x) = \begin{cases} a, & \text{if } x \leq a \\ x, & \text{if } a < x < b \\ b, & \text{if } x \geq b \end{cases} \tag{3}$$

There are several ways in which gaps can be extracted. Here, we follow the work in [29] due to its computational efficiency. Let G denotes the list of detected gaps. Each gap $\mathrm{g} \in G$ is characterized by two obstacle points, referred to as sides. We call the side with the smaller index in S a right side $(\mathbf{p}_{\mathrm{r}}(\mathrm{g}))$, while the other is a left side $(\mathbf{p}_{\mathrm{l}}(\mathrm{g}))$.

IV. Admissible Gap

This section presents the *Admissible gap* concept, providing a foundation to develop our approach. First, the robot's kinematic constraints is reviewed in section IV-A. Section IV-B presents our strategy of traversing gaps. In section IV-C, this strategy is used to identify the admissibility status of a specific gap. To enhance the readability, the notation defining the gap is removed, e.g. $\mathbf{p}_{\mathrm{r}}(\mathrm{g})$ is abbreviated $\mathbf{p}_{\mathrm{r}}$.

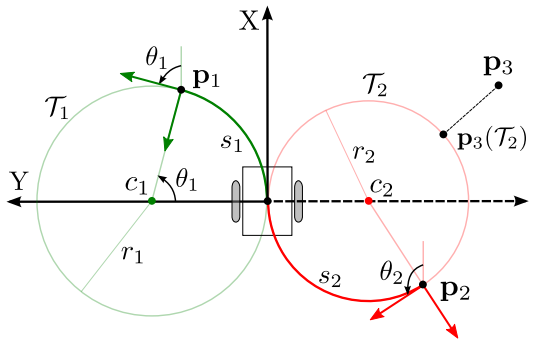


Fig. 1: Kinematic constraints. To reach $\mathbf{p}_1$ and $\mathbf{p}_2$ the robot moves along circles $\mathcal{T}_1$ and $\mathcal{T}_2$, whose radii and centers are $r1$, c_1, $r2$, and c_2, respectively. The robot orientation at points $\mathbf{p}_1$ and $\mathbf{p}_2$ is θ_1 and θ_2. The distance traveled is s_1 and s_2. The point on $\mathcal{T}_2$ closest to $\mathbf{p}_3$ is $\mathbf{p}_3(\mathcal{T}_2)$.

A. Kinematic Constraints

We consider here a differential-drive robot moving on a flat surface. The motion of this robot is constrained by [9]:

$$-\dot{x}\sin\theta + \dot{y}\cos\theta = 0 \tag{4}$$

where (x, y, θ) is the robot's pose in the world coordinates.

The kinematic model of such a robot is expressed by:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} w \tag{5}$$

where v and w are the translational and rotation velocities.

It has been shown that the trajectory is approximated by a sequence of circular arcs whose centers lie on the robot's y-axis [9]. Let $\mathbf{p}_i$ be any point, the robot moves along circle $\mathcal{T}_i$ to reach $\mathbf{p}_i$. The radius of $\mathcal{T}_i$ is r_i and given by (Fig. 1):

$$r_i = \frac{x_i^2 + y_i^2}{2y_i}, \; r_i \in]-\infty, \infty[\tag{6}$$

where (x_i, y_i) are the Cartesian coordinates of $\mathbf{p}_i$.

Circle $\mathcal{T}_i$ is uniquely defined by a *tangent direction*:

$$\chi_i = \begin{cases} \arctan\left(\frac{1}{r_i}\right), & \text{if } x_i \geq 0 \\ \text{sgn}\,(y_i) \cdot \pi - \arctan\left(\frac{1}{r_i}\right), & \text{otherwise} \end{cases} \tag{7}$$

The sign of x_i differentiates forward from backward motion.

The center of $\mathcal{T}_i$ is $c_i = (0, r_i)$. When the robot reaches $\mathbf{p}_i$, its orientation θ_i is tangent to $\mathcal{T}_i$ and defined by (Fig. 1):

$$\theta_i = \begin{cases} \arccos\left(\frac{r_i - y_i}{r_i}\right), & \text{if } \text{sgn}(x_i) = \text{sgn}(y_i) \\ -\arccos\left(\frac{r_i - y_i}{r_i}\right), & \text{otherwise} \end{cases} \tag{8}$$

The distance to $\mathbf{p}_i$ along $\mathcal{T}_i$ (arc length) is given by:

$$s_i = \begin{cases} |x_i|, & \text{if } y_i = 0 \\ |\theta_i \cdot r_i|, & \text{otherwise} \end{cases} \tag{9}$$

The point on $\mathcal{T}_i$ closest to a given point $\mathbf{p}_j$ is given by:

$$\mathbf{p}_j(\mathcal{T}_i) = \langle 0, r_i \rangle + \vec{\mathbf{u}}_a \cdot |r_i| \tag{10}$$

where $\vec{\mathbf{u}}_a = \vec{\mathbf{a}}/|\vec{\mathbf{a}}|$ is the unit direction vector of $\vec{\mathbf{a}} = \langle x_j, y_j - r_i \rangle$ and (x_j, y_j) the Cartesian coordinates of $\mathbf{p}_j$.

B. Traversing Gaps

The robot is driven through $\mathrm{g} \in G$ such that a compromise between efficiency and safety is achieved. A subgoal $\mathbf{p}_\mathrm{s}$ is assigned to g so that the robot circumnavigates one of its sides $\mathbf{p}_\mathrm{c}$ while keeping a safe distance $\mathrm{d_s}$ to it. Setting $\mathrm{d_s}$ is based on the width of g: for a narrow g, $\mathrm{d_s}$ is set to half of its width, but for a wide g, $\mathrm{d_s}$ is set to $R + \mathrm{d_{safe}}$, where R is the robot's radius and $\mathrm{d_{safe}}$ a suitable clearance to obstacles:

$$\mathrm{d_s} = \min\left(R + \mathrm{d_{safe}}, \frac{1}{2}\mathrm{w(g)}\right), \quad \mathrm{w(g)} = \|\mathbf{p_l} - \mathbf{p_r}\| \tag{11}$$

Let $\mathbf{p}_\mathrm{cg}$ be the side closer to the goal and $\mathbf{p}_\mathrm{m}$ the midpoint between both sides ($\mathbf{p}_\mathrm{r}$ and $\mathbf{p}_\mathrm{l}$). Since one of our objectives is to reduce the path length, $\mathbf{p}_\mathrm{c}$ is set to $\mathbf{p}_\mathrm{cg}$. However, if this causes the robot to get close to $\mathbf{p}_\mathrm{l}$ or $\mathbf{p}_\mathrm{r}$, the side closest to the robot along $\mathcal{T}_m$ (path leading to $\mathbf{p}_\mathrm{m}$), $\mathbf{p}_\mathrm{cr}$, is selected:

$$\mathbf{p}_\mathrm{c} = \begin{cases} \mathbf{p}_\mathrm{cg}, & \text{if } \|\mathbf{p}_\mathrm{l}(\mathcal{T}_\mathrm{m}) - \mathbf{p}_\mathrm{l}\| > \mathrm{d_s} \wedge \|\mathbf{p}_\mathrm{r}(\mathcal{T}_\mathrm{m}) - \mathbf{p}_\mathrm{r}\| > \mathrm{d_s} \\ \mathbf{p}_\mathrm{cr}, & \text{otherwise} \end{cases} \tag{12}$$

where $\mathbf{p}_\mathrm{l}(\mathcal{T}_\mathrm{m})$ and $\mathbf{p}_\mathrm{r}(\mathcal{T}_\mathrm{m})$ are the points on $\mathcal{T}_\mathrm{m}$ closest to $\mathbf{p}_\mathrm{l}$ and $\mathbf{p}_\mathrm{r}$, respectively (see Eq. (10)), and $\mathbf{p}_\mathrm{cr}$ is given by:

$$\mathbf{p}_\mathrm{cr} = \begin{cases} \mathbf{p}_\mathrm{l}, & \text{if } s_\mathrm{l}(\mathcal{T}_\mathrm{m}) \leq s_\mathrm{r}(\mathcal{T}_\mathrm{m}) \\ \mathbf{p}_\mathrm{r}, & \text{otherwise} \end{cases} \tag{13}$$

where $s_\mathrm{l}(\mathcal{T}_\mathrm{m})$ and $s_\mathrm{r}(\mathcal{T}_\mathrm{m})$ are the lengths of the arcs along $\mathcal{T}_\mathrm{m}$ connecting the robot origin to $\mathbf{p}_\mathrm{l}(\mathcal{T}_\mathrm{m})$ and $\mathbf{p}_\mathrm{r}(\mathcal{T}_\mathrm{m})$.

Let circle $\mathcal{S}$ be centered at $\mathbf{p}_\mathrm{c}$ with radius $\mathrm{d_s}$ (see Fig. 2), $\mathbf{p}_\mathrm{s}$ is placed at the tangent point $\mathbf{p}_t$ between $\mathcal{S}$ and the path leading to $\mathbf{p}_t$, $\mathcal{T}_t$. Circles $\mathcal{T}_t$ and $\mathcal{S}$ are mutually tangent if:

$$x_\mathrm{c}^2 + (y_\mathrm{c} - r_t)^2 = (|r_t| \pm \mathrm{d_s})^2 \tag{14}$$

where $(x_\mathrm{c}, y_\mathrm{c})$: coordinates of $\mathbf{p}_\mathrm{c}$ and r_t: radius of $\mathcal{T}_t$.

Solving for r_t, we get two circles tangent to $\mathcal{S}$ with radii:

$$r_t = \frac{x_\mathrm{c}^2 + y_\mathrm{c}^2 - \mathrm{d_s}^2}{2(y_\mathrm{c} \pm \mathrm{d_s})} \tag{15}$$

Given r_t, $\mathbf{p}_t$ can be expressed as follows:

$$\mathbf{p}_t = \begin{cases} \langle x_\mathrm{c}, 0\rangle, & \text{if } r_t = \infty \\ \langle 0, r_t\rangle + \vec{\mathbf{u}}_v \cdot |r_t|, & \text{otherwise} \end{cases} \tag{16}$$

where $\vec{\mathbf{u}}_v = \vec{\mathbf{v}}/|\vec{\mathbf{v}}|$, $\vec{\mathbf{v}} = \langle x_\mathrm{c}, y_\mathrm{c} - r_t\rangle$.

From Eqs. (15) and (16), it is apparent that we get two tangent points. We set $\mathbf{p}_\mathrm{s}$ to the tangent point $\mathbf{p}_\mathrm{t}$ located in the direction of the gap (see Fig. 2), i.e. $\mathbf{p}_\mathrm{t}$ that satisfies:

$$\mathrm{proj}\,(\chi_\mathrm{t} - \chi_\mathrm{c}) \cdot \Upsilon < 0 \tag{17}$$

where χ_t and χ_c are the tangent directions (Eq. (7)) associated with $\mathcal{T}_t$ and $\mathcal{T}_\mathrm{c}$ (path leading to $\mathbf{p}_\mathrm{c}$). Υ is given by:

$$\Upsilon = \begin{cases} +1, & \text{if } \mathbf{p}_\mathrm{c} \text{ is a left side} \\ -1, & \text{if } \mathbf{p}_\mathrm{c} \text{ is a right side} \end{cases} \tag{18}$$

C. Admissibility Status

In the context of obstacle avoidance, only paths that are the result of executing a single motion control are of interest. Assume that the execution of $u = (v, w)$ results in a path that passes through $\mathbf{p}_\mathrm{s}$, described in section IV by circle $\mathcal{T}_\mathrm{s}$. Let

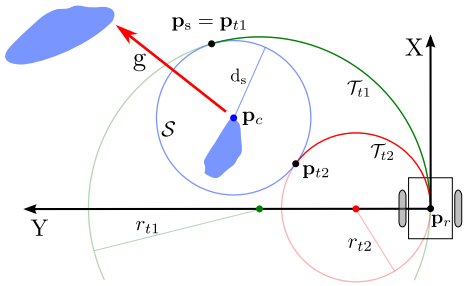


Fig. 2: Locating $\mathbf{p}_s$. First, we identify $\mathbf{p}_{t1}$ and $\mathbf{p}_{t2}$ that are located at the tangent between S and two circles centered at the robot's y-axis ($\mathcal{T}_{t1}$, $\mathcal{T}_{t2}$). Then, $\mathbf{p}_s$ is set $\mathbf{p}_{t1}$ as it is the one leading to the gap g.

$\tau\,[\mathbf{p}_r \to \mathbf{p}_\mathrm{s}]$ denote the arc followed along $\mathcal{T}_\mathrm{s}$ to reach $\mathbf{p}_\mathrm{s}$, a gap g is admissible from $\mathbf{p}_r$ ($\mathrm{AG}\,[\mathbf{p}_r \to \mathrm{g}]$) iff the region occupied by the robot while moving along $\tau\,[\mathbf{p}_r \to \mathbf{p}_\mathrm{s}]$, denoted as $\mathcal{T}\,[\mathbf{p}_r \to \mathbf{p}_\mathrm{s}]$, does not intersect obstacles:

$$\mathrm{AG}\,[\mathbf{p}_r \to \mathrm{g}] \iff \mathcal{T}\,[\mathbf{p}_r \to \mathbf{p}_\mathrm{s}] \cap \bigcup_{i=1}^{n} \mathbf{p}_i = \phi$$

Next, we present a procedure to identify whether a given gap is admissible or not. The idea is to extract the set of obstacles in collision with $\mathcal{T}\,[\mathbf{p}_r \to \mathbf{p}_\mathrm{s}]$, denoted $\mathcal{O}\,[\mathbf{p}_r \to \mathbf{p}_\mathrm{s}]$.

For each $\mathbf{p}_i \in \mathrm{S}$, we check if the circle centered at $c_s = (0, r_\mathrm{s})$ and goes through $\mathbf{p}_i$, denoted $\mathcal{C}(c_s, \mathbf{p}_i)$, intersects the robot's boundary. If not, $\mathbf{p}_i$ is discarded (e.g, $\mathbf{p}_3$ in Fig. 3 is discarded since $\mathcal{C}(c_s, \mathbf{p}_3)$ is collision-free). If an intersection occurs, say at point $\mathbf{p}_e$ (e.g. see circle $\mathcal{C}(c_s, \mathbf{p}_1)$ in Fig. 3), we identify the location of $\mathbf{p}_e$ when the robot reaches $\mathbf{p}_\mathrm{s}$:

$$\mathbf{p}_e^* = \begin{cases} \mathbf{p}_e + \mathbf{p}_\mathrm{s}, & \text{if } y_\mathrm{s} = 0 \\ R\,\mathbf{p}_e + t, & \text{otherwise} \end{cases} \tag{19}$$

where R and t are defined as follows:

$$R = \begin{bmatrix} \frac{x_\mathrm{s}^2 - y_\mathrm{s}^2}{x_\mathrm{s}^2 + y_\mathrm{s}^2} & -\frac{2x_\mathrm{s}y_\mathrm{s}}{x_\mathrm{s}^2 + y_\mathrm{s}^2} \\ \frac{2x_\mathrm{s}y_\mathrm{s}}{x_\mathrm{s}^2 + y_\mathrm{s}^2} & \frac{x_\mathrm{s}^2 - y_\mathrm{s}^2}{x_\mathrm{s}^2 + y_\mathrm{s}^2} \end{bmatrix}, t = \begin{bmatrix} x_\mathrm{s} \\ y_\mathrm{s} \end{bmatrix} \tag{20}$$

where $(x_\mathrm{s}, y_\mathrm{s})$ are the Cartesian coordinates of $\mathbf{p}_\mathrm{s}$.

It should be clear that $\mathbf{p}_i$ causes collision if it falls on the arc connecting $\mathbf{p}_e$ to $\mathbf{p}_e^*$ (In Fig. 3, $\mathbf{p}_1$ causes collision, but $\mathbf{p}_2$ doesn't). To check this condition, a frame $\mathcal{F}$ centered at c_s is defined whose orientation is given by:

$$\theta_\mathcal{F} = \mathrm{atan2}\,(y_e - r_\mathrm{s}, x_e) \tag{21}$$

where (x_e, y_e) are the Cartesian coordinates of $\mathbf{p}_e$.

Obstacle $\mathbf{p}_i$ is added to the output list $\mathcal{O}\,[\mathbf{p}_r \to \mathbf{p}_\mathrm{s}]$ if:

$$(\Delta \cdot {}^\mathcal{F}\theta_i) \bmod 2\pi \leq (\Delta \cdot {}^\mathcal{F}\theta_e^*) \bmod 2\pi$$

where ${}^\mathcal{F}\theta_i$ and ${}^\mathcal{F}\theta_e^*$ represent the angles towards $\mathbf{p}_i$ and $\mathbf{p}_e^*$, but relative to frame $\mathcal{F}$ (see Eq. (2)), and Δ is defined as:

$$\Delta = \begin{cases} +1, & \text{if } \mathrm{sgn}(x_\mathrm{s}) = \mathrm{sgn}(y_\mathrm{s}) \\ -1, & \text{otherwise} \end{cases} \tag{22}$$

V. Collision Avoidance Method

This section describes how to navigate a mobile robot using the AG concept. At each sensor update, it is checked

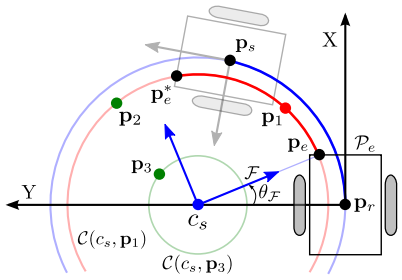


Fig. 3: Collision check along $\tau\,[\mathbf{p}_r \rightarrow \mathbf{p}_s]$ given 3 obstacle points $\mathbf{p}_1$ - $\mathbf{p}_3$.

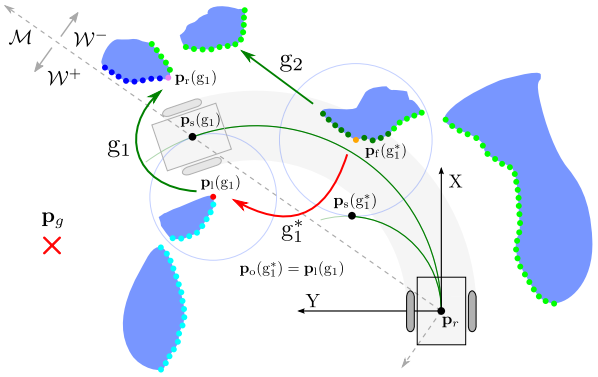


Fig. 4: Gap navigability check. Gap g_1 is navigable since a virtual gap g_1^* could be found, such that it is admissible and traversing it leads to g_1.

whether the goal is admissible or not (i.e. $\mathcal{O}\,[\mathbf{p}_r \rightarrow \mathbf{p}_g] = \phi$). If not, the robot is driven towards a gap. We select the closest to the goal and check its navigability, see section V-A. If it is non-navigable, we select another gap and the process repeats. Section V-B shows how to compute the motion controls that drive a robot towards the goal or the selected gap.

A. Gap Navigability Check

A gap g is *navigable* if the robot can safely navigate through it. Therefore, an admissible gap is navigable. A non-admissible gap is considered navigable if it is possible to create a virtual gap g^* in such away that $\mathrm{AG}\,[\mathbf{p}_r \rightarrow \mathrm{g}^*]$ holds and passing through g^* leads to g. Next, we present an algorithm to determine g^*, if possible. The algorithm initializes g^* to g and iteratively performs two steps:

Step 1: List S is divided into two sublists: S_{in} which includes the points located within g^* (i.e. $\{\mathbf{p}_{\mathrm{r}}(\mathrm{g}^*)$, $\mathbf{p}_{\mathrm{r}+1}(\mathrm{g}^*)$, ..., $\mathbf{p}_{\mathrm{l}-1}(\mathrm{g}^*)$, $\mathbf{p}_{\mathrm{l}}(\mathrm{g}^*)\}$) and S_{ex} which contains the other points. Non obstacle points are removed from both sublists resulting in O_{in} and O_{ex}. O_{ex} is then reduced to $\mathrm{O}'_{\mathrm{ex}}$ by eliminating every point making an angular difference $> \pi$ with $\mathbf{p}_{\mathrm{l}}(\mathrm{g}^*)$ and $\mathbf{p}_{\mathrm{r}}(\mathrm{g}^*)$, traveling in the direction of the gap:

$$\mathrm{O}'_{\mathrm{ex}} = \mathrm{O}^+_{\mathrm{ex}} \cup \mathrm{O}^-_{\mathrm{ex}} \tag{23}$$

with $\mathrm{O}^+_{\mathrm{ex}}$ and $\mathrm{O}^+_{\mathrm{ex}}$ given by:

$$\mathrm{O}^+_{\mathrm{ex}} = \{\mathbf{p}_i \in \mathrm{O}_{\mathrm{ex}} \mid \mathrm{proj}\,(\theta_i - \theta_{\mathrm{r}}(\mathrm{g}^*)) > 0\} \tag{24}$$

$$\mathrm{O}^-_{\mathrm{ex}} = \{\mathbf{p}_i \in \mathrm{O}_{\mathrm{ex}} \mid \mathrm{proj}\,(\theta_i - \theta_{\mathrm{l}}(\mathrm{g}^*)) < 0\} \tag{25}$$

where θ_i, $\theta_{\mathrm{r}}(\mathrm{g}^*)$ and $\theta_{\mathrm{l}}(\mathrm{g}^*)$ are the angles towards $\mathbf{p}_i$, $\mathbf{p}_{\mathrm{r}}(\mathrm{g}^*)$ and $\mathbf{p}_{\mathrm{l}}(\mathrm{g}^*)$. Notice that the points removed from O_{ex} are located behind the robot while moving towards $\mathbf{p}_{\mathrm{s}}(\mathrm{g}^*)$, and thus excluded from the collision check down.

Fig. 4 shows two gaps (g_1, g_2), where g_1 is the closest to the goal. O_{in} includes both sides of g_1 ($\mathbf{p}_{\mathrm{r}}(\mathrm{g}_1)$, $\mathbf{p}_{\mathrm{l}}(\mathrm{g}_1)$) in addition to the points visualized by dark blue. All other points belong to O_{ex}, where non of them makes an angular difference $> \pi$ with both sides, and hence $\mathrm{O}'_{\mathrm{ex}} = \mathrm{O}_{\mathrm{ex}}$.

Obstacles falling in $\mathrm{O}'_{\mathrm{ex}}$ are then checked for collision with $\mathcal{T}\,[\mathbf{p}_r \rightarrow \mathbf{p}_{\mathrm{s}}(\mathrm{g}^*)]$ as described in section IV-C, resulting in $\mathcal{O}'_{\mathrm{ex}}\,[\mathbf{p}_r \rightarrow \mathbf{p}_{\mathrm{s}}(\mathrm{g}^*)]$. In Fig. 4, $\mathcal{O}'_{\mathrm{ex}}\,[\mathbf{p}_r \rightarrow \mathbf{p}_{\mathrm{s}}(\mathrm{g}_1)]$ includes the orange and dark green points. If $\mathcal{O}'_{\mathrm{ex}}\,[\mathbf{p}_r \rightarrow \mathbf{p}_{\mathrm{s}}(\mathrm{g}^*)] \neq \phi$, it is apparent that g^* is not admissible. But, a new g^* that leads to g can be constructed, and thus we proceed to step 2. However, if $\mathcal{O}'_{\mathrm{ex}}\,[\mathbf{p}_r \rightarrow \mathbf{p}_{\mathrm{s}}(\mathrm{g}^*)] = \phi$, the obstacles in O_{in} are tested for collision, resulting in $\mathcal{O}_{\mathrm{in}}\,[\mathbf{p}_r \rightarrow \mathbf{p}_{\mathrm{s}}(\mathrm{g}^*)]$. If $\mathcal{O}_{\mathrm{in}}\,[\mathbf{p}_r \rightarrow \mathbf{p}_{\mathrm{s}}(\mathrm{g}^*)] = \phi$, the current value of g^* is returned as it is admissible. Otherwise, NULL is returned.

Step 2: A new gap g^{**} is created. Its first side $\mathbf{p}_{\mathrm{f}}(\mathrm{g}^{**})$ is set to the point falling in $\mathcal{O}'_{\mathrm{ex}}\,[\mathbf{p}_r \rightarrow \mathbf{p}_{\mathrm{s}}(\mathrm{g}^*)]$ and closest to the path towards $\mathbf{p}_{\mathrm{s}}(\mathrm{g}^*)$, $\mathcal{T}_{\mathrm{s}}(\mathrm{g}^*)$. In Fig. 4, $\mathbf{p}_{\mathrm{f}}(\mathrm{g}_1^*)$ is set to the orange point as it is the closest to $\mathcal{T}_{\mathrm{s}}(\mathrm{g}_1)$:

$$\mathbf{p}_{\mathrm{f}}(\mathrm{g}^{**}) = \underset{\mathbf{p}_i \in \mathcal{O}'_{\mathrm{ex}}[\mathbf{p}_r \rightarrow \mathbf{p}_{\mathrm{s}}(\mathrm{g}^*)]}{\mathrm{argmin}} \|\mathbf{p}_i - \mathbf{p}_i(\mathcal{T}_{\mathrm{s}}(\mathrm{g}^*))\| \tag{26}$$

Let $\mathcal{M}$ be a frame centered at $(0, 0)$ and oriented towards the center of g^*. Frame $\mathcal{M}$ divides the workspace into two regions; one to the left ($\mathcal{W}^+$) and the other to the right ($\mathcal{W}^-$) of the x-axis. Let $\mathbf{P}^+ = \{\mathbf{p}_{\mathrm{l}}(\mathrm{g}^*)$, $\mathbf{p}_{\mathrm{l}+1}(\mathrm{g}^*)$, ...$\}$ and $\mathbf{P}^- = \{\mathbf{p}_{\mathrm{r}}(\mathrm{g}^*)$, $\mathbf{p}_{\mathrm{r}-1}(\mathrm{g}^*)$, ...$\}$ be the lists of points located to the left and right of g^*, respectively. The other side of g^{**}, $\mathbf{p}_{\mathrm{o}}(\mathrm{g}^{**})$, is searched for in $\mathbf{P}^+$ if $\mathbf{p}_{\mathrm{f}}(\mathrm{g}^{**}) \in \mathcal{W}^-$ or in $\mathbf{P}^-$ if $\mathbf{p}_{\mathrm{f}}(\mathrm{g}^{**}) \in \mathcal{W}^+$. Search is performed in the same order of the list, where the point closest to $\mathbf{p}_{\mathrm{f}}(\mathrm{g}^{**})$ is selected. To ensure that traversing g^{**} leads to g^*, the angular distance between both sides must be $< \pi$. In Fig. 4, only the points visualized by red ($\mathbf{p}_{\mathrm{l}}(\mathrm{g}_1)$) and aqua are valid. Among these points, $\mathbf{p}_{\mathrm{o}}(\mathrm{g}_1^*)$ is set to $\mathbf{p}_{\mathrm{l}}(\mathrm{g}_1)$ as it is the closest to $\mathbf{p}_{\mathrm{f}}(\mathrm{g}_1^*)$.

Once g^{**} is constructed, g^* is set to g^{**} and we repeat step 1. Gap g is considered navigable only if the algorithm returns a valid value of g^*. In Fig. 4, the algorithm terminates in the second iteration, since $\tau\,[\mathbf{p}_r \rightarrow \mathbf{p}_{\mathrm{s}}(\mathrm{g}_1^*)]$ is collision-free.

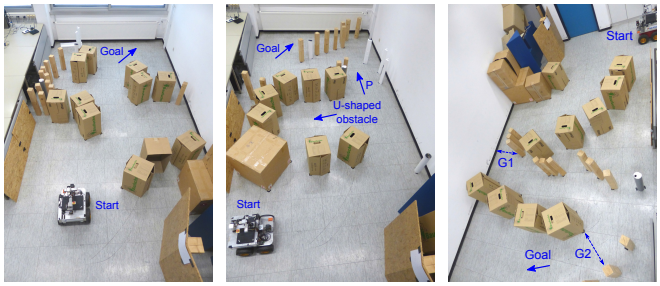
B. Setting Motion commands

We have seen how to identify the instantaneous goal, $\hat{\mathbf{p}}_g$, which could be the goal $\mathbf{p}_g$ or a subgoal $\mathbf{p}_{\mathrm{s}}(\mathrm{g}^*)$ corresponding to an admissible gap. For driving the robot towards $\hat{\mathbf{p}}_g$, we follow the solution proposed in [9]. The idea is to compute the control inputs such that the turning radius is maintained, i.e. satisfy $v = w\hat{r}_g$, where $\hat{r}_g$ is the radius of the path leading to $\hat{\mathbf{p}}_g$ (Eq. (6)). Let $\hat{x}_g$ be the x coordinate of $\hat{\mathbf{p}}_g$ and $\hat{\zeta}_g = \arctan(\hat{r}_g^{-1})$, the velocities are defined by:

$$v = \mathrm{sgn}(\hat{x}_g) \cdot S_{\mathrm{limit}} \cdot \cos(\hat{\zeta}_g) \tag{27}$$

$$w = \mathrm{sgn}(\hat{x}_g) \cdot S_{\mathrm{limit}} \cdot \sin(\hat{\zeta}_g) \tag{28}$$

The idea behind S_{limit} is to control the speed based the distance to nearby obstacles. While the work in [9] proposes



(a) (b) (c)

Fig. 5: Environment setup for experiments (a) 1, (b) 2, and (c) 3.

a linear equation, we use the following nonlinear function:

$$S_{\text{limit}} = \left(\sqrt{1 - \text{sat}_{[0,1]} \left(\frac{D_{vs} - r_{min}}{D_{vs}} \right)} \right) . S_{\text{max}} \quad (29)$$

where r_{min} is the distance to the closest obstacle, S_{max} the distance from the velocity origin to the rectangle of maximum velocities, and D_{vs} a parameter that identifies a zone around the robot in which the speed is limited.

VI. Experimental Results

We tested the AG method using our robot; a skid-steering Pioneer 3-AT equipped with an on-board computer and has a rectangular shape ($0.52\,m \times 0.48\,m$). To get a full field of view, two laser scanners were employed; a Hokuyo UTM-30LX (range = $30\,m$) and a Hokuyo URG-04LX (range = $5.6\,m$). Several experiments were performed to verify the power of the AG method. Next, we report the results of three experiments. Videos of these experiments are available at:
`http://getwww.uni-paderborn.de/research/videos/admiss_gap`.

The AG approach is compared to two methods; ARM-ND+ [9] and the robot navigation stack DWA-A* [32]. The methods have been implemented using ROS. To maintain safety, (v, w) were limited to $(0.5\,m/s, 1.0\,rad/s)$.

Experiment 1: The robot had to pass through randomly distributed boxes, see Fig. 5a. Oscillations in motion can be seen along the path followed by ARM-ND+, e.g. see points 1 - 3 in Fig. 6a. Using DWA-A*, the robot navigated near obstacles (e.g. A and B in Fig. 6b). The AG method smoothly and safely drove the robot towards the goal, see Fig. 6c.

Experiment 2: The main difficulty here was the existence of a U-shaped structure and a narrow curved passage (P) created by different obstacle sizes (Fig. 5b). The motion was oscillatory by running ARM-ND+, see points 1 - 4 in Fig. 6d. Furthermore, the robot got close to obstacles (e.g. A and D). By applying DWA-A*, the robot moved relatively fast, but close to obstacles B, C, and E (Fig. 6e). Furthermore, while entering passage P, the robot stopped and rotated on spot as a result of triggering a recover behavior. The AG method performed pretty well as shown in Fig. 6f.

Experiment 3: Here, a challenging environment was created (Fig. 5c). The robot had to move through extremely narrow passages, e.g. the width of gap G2 was less than $2R$. Also, the arena consisted of a mixture of different size obstacles. Only the AG method was able to drive the robot towards the goal (Fig. 6i), even with very smooth motion. By applying ARM-ND+, it was not possible to detect gap G1 and the algorithm terminated. Hence, the robot decelerated but hit wall A before coming to a full stop, Fig. 6g. Using DWA-A*, the motion was smooth until a deadlock occurred before gap G1, where the robot kept oscillating (Fig. 6h).

TABLE I: Performance evaluation of the proposed AG approach. The shown results present the averages over 10 runs for each of the metrics.

Exp.	Method	T_{tot}	C_{chg}	J_{acc}	R_{obs}
1	ARM-ND+	35.4	118.62	2.28	252.28
	DWA-A*	**19.0**	0.53	4.87	169.34
	AG	22.2	**0.39**	**0.13**	**149.14**
2	ARM-ND+	68.8	181.15	6.82	543.02
	DWA-A*	41.3	161.22	11.05	375.81
	AG	**35.41**	**1.00**	**0.35**	**215.89**
3	ARM-ND+	fail	fail	fail	fail
	DWA-A*	fail	fail	fail	fail
	AG	**51.0**	**1.07**	**0.37**	**657.48**

VII. Evaluation and Discussion

The following metrics are used to assess the behavior [6]:
1) Total time (T_{tot}). Time needed to complete a mission.
2) Curvature Change (C_{chg}). Used to detect oscillations:

$$\text{C}_{\text{chg}} = \frac{1}{\text{T}_{\text{tot}}} \int_0^{\text{T}_{\text{tot}}} |k'(t)|\, dt, \quad k(t) = \left| \frac{w(t)}{v(t)} \right| \quad (30)$$

3) Accumulated jerk (J_{acc}). Measures smoothness:

$$\text{J}_{\text{acc}} = \frac{1}{\text{T}_{\text{tot}}} \int_0^{\text{T}_{\text{tot}}} [\ddot{v}(t)]^2\, dt \quad (31)$$

4) Risk (R_{obs}). Reflects the proximity to obstacles.

$$\text{R}_{\text{obs}} = \int_0^{\text{T}_{\text{tot}}} \frac{1}{r_{\text{min}}(t)}\, dt \quad (32)$$

where $r_{\text{min}}(t)$ is the distance to the closest obstacle at time t. Notice that for each of the metrics describe above, the lower the value, the better the performance.

Based on these metrics, a performance evaluation was performed. Table I shows the results obtained. It can be seen that the AG achieves the best results. The performance of ARM-ND+ was poor since, by constructing ARM, only admissible gaps can be seen. Also, ND+ has the tendency to generate oscillatory motion [6]. Although DWA-A* integrates both global and local planners, the AG method behaved better, since it computes the avoidance maneuver based on the location of obstacles between the robot and the closest gap, providing lookahead, while maintaining d_{s} to obstacles.

We noticed that the AG is robust against changes of the environment. It managed all test scenarios. The other methods, especially ARM-ND+, were sensitive to the environmental changes. Another important aspect of the AG is its simplicity; unlike ARM-ND+, there is no need to create an abstraction layer. Moreover, Unlike DWA, it can drive a robot in narrow

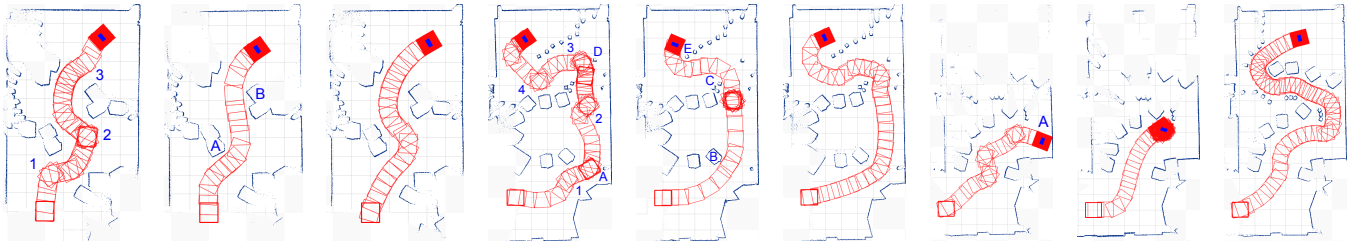


Fig. 6: Experiments. (a-c) Experiment 1. Trajectory followed by (a) ARM-ND+, (b) DWA-A*, and (c) AG. (d-f) Experiment 2. Trajectory followed by (d) ARM-ND+, (e) DWA-A*, and (f) AG. (g-i) Experiment 3. Trajectory followed by (g) ARM-ND+, (h) DWA-A*, and (i) AG.

spaces without being integrated with a global planner. A higher level planner is only needed to avoid cyclic loops.

VIII. Conclusions and Future Works

This paper presents the admissible gap concept, which addresses the question of whether it is possible to find a gap that a robot may safely pass through, while obeying the shape and motion constraints. With this concept, it has been possible to achieve navigation in unknown cluttered environments. Unlike other gap-based techniques, the proposed AG approach directly considers the robot shape and kinematic constraints rather than adapting a method originally designed for a holonomic point-like robot. The key idea is the construction of an admissible gap, once traversed, the robot progresses towards the goal and a compromise between efficiency and safety is achieved. Experiments showed that the AG approach generates efficient, safe, and smooth trajectories.

The AG assumes that velocities are executed, such that the robot moves along arcs. There is no guarantee that it always happens. Hence, it is of our interest to relax this assumption.

References

[1] 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.

[2] J. W. Durham and F. Bullo, "Smooth nearness-diagram navigation," in *IROS*, (Nice, France), pp. 690–695, September 2008.

[3] 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.

[4] M. Mujahed, D. Fischer, and B. Mertsching, "Smooth reactive collision avoidance in difficult environments," in *IEEE Conference on Robotics and Biomimetics (ROBIO)*, (China), pp. 1471 – 1476, 2015.

[5] M. Mujahed and B. Mertsching, "A new gap-based collision avoidance method for mobile robots," in *SSRR*, (Lausanne), pp. 220 – 226, 2016.

[6] M. Mujahed, D. Fischer, and B. Mertsching, "Tangential gap flow (tgf) navigation: A new reactive obstacle avoidance approach for highly cluttered environments," *Rob. Aut. Sys.*, vol. 84, pp. 15–30, 2016.

[7] A. Bemporad, A. D. Luca, G. Oriolo, and A. De, "Local incremental planning for a car-like robot navigating among obstacles," in *ICRA*, (Minneapolis, Minnesota), pp. 1205–1211, 1996.

[8] J. Minguez and L. Montano, "Robot navigation in very complex, dense, and cluttered indoor/outdoor evironments," in *15th IFAC World Congress*, (Barcelona, Spain), pp. 1–6, 2002.

[9] J. Minguez and L. Montano, "Extending collision avoidance methods to consider the vehicle shape, kinematics, and dynamics of a mobile robot," *IEEE Trans. Robot.*, vol. 25, no. 2, pp. 367–381, 2009.

[10] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Robot. Res.*, vol. 5, pp. 90–98, April 1986.

[11] B. Kovács, G. Szayer, F. Tajti, M. Burdelis, and P. Korondi, "A novel potential field method for path planning of mobile robots by adapting animal motion attributes," *Rob. Aut. Sys.*, vol. 82, pp. 24–34, 2016.

[12] D. Panagou, "Motion planning and collision avoidance using navigation vector fields," in *ICRA*, (Hong Kong), pp. 2513–2518, May 2014.

[13] R. Hegde and D. Panagou, "Multi-agent motion planning and coordination in polygonal environments using vector fields and model predictive control," in *ECC*, (Aalborg, Denmark), June 2016.

[14] A. A. Masoud, "A harmonic potential approach for simultaneous planning and control of a generic uav platform," *Journal of Intelligent and Robotic Systems*, vol. 65, no. 1-4, pp. 153–173, 2012.

[15] H. Chiang, N. Malone, K. Lesser, M. Oishi, and L. Tapia, "Path-guided artificial potential fields with stochastic reachable sets for motion planning in highly dynamic environments," in *ICRA*, (Seattle, USA), pp. 2347 – 2354, May 2015.

[16] V. J. Lumelsky and A. A. Stepanov, "Dynamic path.planning for a mobile automaton with limited information on the environment," *IEEE Trans. Aut. Control*, vol. 31, no. 5, pp. 1058–1069, 1986.

[17] A. S. Matveev, M. C. Hoy, and A. V. Savkin, "A method for reactive navigation of nonholonomic under-actuated robots in maze-like environments," *Automatica*, vol. 49, pp. 1268–1274, May 2013.

[18] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Rob. Aut. Mag.*, vol. 4, pp. 23–33, 1997.

[19] 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.

[20] D. A. de Lima and A. C. Victorino, "A hybrid controller for vision-based navigation of autonomous vehicles in urban environments," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 8, pp. 2310–2323, 2016.

[21] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *Int. J. Robot. Res.*, vol. 17, no. 7, 1998.

[22] D. Bareiss and J. Berg, "Generalized reciprocal collision avoidance," *Int. J. Robot. Res.*, vol. 34, pp. 1501 – 1514, April 2015.

[23] B. H. Lee, J. D. Jeon, and J. H. Oh, "Velocity obstacle based local collision avoidance for a holonomic elliptic robot," *Autonomous Robots*, pp. 1–17, June 2016. doi:10.1007/s10514-016-9580-2.

[24] T. Fraichard and H. Asama, "Inevitable collision states - a step towards safer robots?," *Adv. Robot.*, vol. 18, no. 10, pp. 1001–1024, 2004.

[25] A. Lawitzky, A. Nicklas, D. Wollherr, and M. Buss, "Determining states of inevitable collision using reachability analysis," in *IROS*, (Chicago, USA), pp. 4142 – 4147, September 2014.

[26] J. Jin, Y. Kim, S. Wee, and N. Gans, "Decentralized cooperative mean approach to collision avoidance for nonholonomic mobile robots," in *ICRA*, (Seattle, USA), pp. 35 – 41, May 2015.

[27] Z. Shiller and S. Sharma, "High speed on-line motion planning in cluttered environments," in *IROS*, (Portugal), pp. 596–601, 2012.

[28] J. Minguez, J. Osuna, and L. Montano, "A "divide and conquer" strategy based on situations to achieve reactive collision avoidance in troublesome scenarios," in *ICRA*, pp. 3855–3862, 2004.

[29] M. Mujahed, D. Fischer, B. Mertsching, and H. Jaddu, "Closest gap based (CG) reactive obstacle avoidance navigation for highly cluttered environments," in *IROS*, (Taipei, Taiwan), pp. 1805 – 1812, 2010.

[30] M. Mujahed, H. Jaddu, D. Fischer, and B. Mertsching, "Tangential closest gap based (TCG) reactive obstacle avoidance navigation for cluttered environments," in *SSRR*, (Linköping), pp. 1 – 6, 2013.

[31] M. Mujahed, D. Fischer, and B. Mertsching, "Robust collision avoidance for autonomous mobile robots in unknown environments," in *International World RoboCup Symposium*, vol. 9776, (Leipzig), 2016.

[32] E. Marder-Eppstein, E. Berger, T. Foote, B. P. Gerkey, and K. Konolige, "The office marathon: Robust navigation in an indoor office environment," in *ICRA*, (Alaska, USA), pp. 300–307, May 2010.