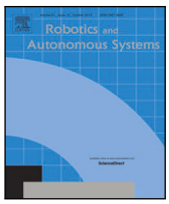
Robotics and Autonomous Systems 174 (2024) 104645

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot



# A survey on Velocity Obstacle paradigm ☆

Federico Vesentini *, Riccardo Muradore, Paolo Fiorini

*University of Verona, Strada Le Grazie 15, Verona, Italy*

ARTICLE INFO

*Keywords:*
Robot motion planning
Dynamic environments
Collision avoidance
Autonomous agents

ABSTRACT

Collision avoidance for multi-agent systems in dynamic and possibly crowded environment is a field of research whose popularity has increased during the last three decades. The approaches to solve the problem belong to two categories according to where the computation is done: centralized and decentralized (i.e. at the agent level). The principal decentralized collision avoidance methods currently available in literature are Artificial Potential Fields, Velocity Obstacle, Social Force, Dynamic Window and Model Predictive Controller. The aim of this paper is to offer a complete and comprehensive review of the most popular decentralized Velocity Obstacle based collision avoidance schemes. A taxonomy will describe the evolution of the field of research back to the original paper in the early 1990s.

## 1. Introduction

The problem of planning collision-free trajectories for autonomous robots is a crucial aspect of the design of multi-agent systems such as automated warehouses, surveillance systems based on unmanned ground or aerial vehicles, automatic transportation lines or, in general, for the safe navigation of autonomous robots.

Collisions between autonomous agents or with moving obstacles in the workspace could cause serious problems, e.g. blocking of production lines, workers injuries, chain collision with other robots, etc. In order to achieve the best level of production efficiency and safety for human workers, every autonomous entity should be able to accomplish programmed tasks by following collision-free trajectories with respect to other agents moving in the same environment and to unexpected fixed or moving obstacles.

In the current literature, there are two main types of methods that guarantee collision-free trajectories for multi-robot systems in which the agents operate in possibly crowded dynamic environments: *Centralized multi-robot navigation* and *Decentralized multi-robot navigation* methods.

In centralized motion planning, all the agents are able to communicate with a common processing unit, sharing information about their state (e.g. current position and velocity) at every instant and also other types of information such as actual goal positions. Several works covering different aspects about multi-robot systems have been proposed, like task allocation [1,2], formation control of robots [3–6], and transportation of objects [7–10]. Recently Shourkry et al. [11] proposed a method based on Satisfiability Modulo Convex optimization (SMC), a combination of satisfiability solving (SAT) and convex programming. The global communication among every agent and the central controller guarantees optimal and safe trajectories [12–15]. The major advantage of this kind of approaches is that collision-free trajectories are computed offline from the beginning to the end and re-planning is done in an optimal way. Unfortunately, they suffer from several drawbacks: (i) they become computationally very expensive when dealing with a large number of agents, thus they do not scale well in crowded scenarios, (ii) the global communication assumption can be unfeasible in real-world applications, since it would require an exceptionally complex as well as expensive communication system between the global controller and every agent within the dynamic environment, (iii) the global communication system or the central global controller may be affected by disturbances, and finally (iv) they assume the perfect knowledge of the environment, so they cannot be applied in presence of unexpected fixed or moving obstacles (e.g. human workers or other not expected objects that may interrupt the navigation of the agents).

Motion planning for autonomous robots based on decentralized methods has been extensively studied during the last decades. The most important and successful approaches to the problem are represented by *Potential Fields*, Khatib [16], *Social Force*, Helbing and Molnar [17], *Dynamic Window*, Fox et al. [18] and *Model Predictive Control*, (MPC) [19].

These approaches are in general relatively simple from a theoretical point of view and highly adaptable to dynamic scenarios and agents' model. Their major drawbacks are essentially the difficulty to handle heterogeneous agents and, since in some cases the behaviors of the

☆ This work was partly supported by the project MIUR "Dipartimenti di Eccellenza" 2018–2022, Italy.

* Corresponding author.
*E-mail addresses:* federico.vesentini@univr.it (F. Vesentini), riccardo.muradore@univr.it (R. Muradore), paolo.fiorini@univr.it (P. Fiorini).

https://doi.org/10.1016/j.robot.2024.104645
Received 8 October 2021; Received in revised form 28 December 2023; Accepted 25 January 2024
Available online 5 February 2024

agents are regulated by a cost function (e.g. a function used to prioritize stopping policies in place of avoidance maneuvers), they may require extra and possibly offline non-trivial calculations in order to tune the related parameters.

Another decentralized method for collision avoidance in multi-agent scenarios is represented by the Velocity Obstacle (VO) Paradigm, Fiorini and Shiller [20,21] and its enhanced formulations. VO is a geometric approach for the navigation of a single agent that has to avoid collisions with several moving obstacles, arising from the concept of "Configuration Space Obstacle" by Lozano-Perez [22] extended to dynamic environments.

The successive formulations have been designed with the purpose to extend the paradigm to dynamic multi-agent environments, Van Den Berg et al. [23], to implement optimal velocity selection for every agent via linear optimization, Van Den Berg et al. [24], to include a constraint of maximum acceleration for the agents, Van Den Berg et al. [25], to enhance the capabilities of the early paradigms in multi-agent scenario, Snape et al. [26] and to extend the paradigm to non-holonomic agents [27]. Generally speaking, VO based approaches assume that agents do not communicate, but only perceive each other.

Agents are characterized by five values: current position, current velocity, radius, initial and goal position. The first three can be sensed by every agent, and the collision avoiding maneuvers are computed using these measurements at every cycle of sensing-and-acting of the algorithm.

This paper aims at analyzing VO-based algorithms and is organized as follows. Section 2 briefly recalls the most popular decentralized methods for collision avoidance of autonomous agents, together with references for further readings. Section 3 describes in details the original Velocity Obstacle paradigm. Section 4 illustrates the taxonomy of the research field. Section 5 describes the most important contributions that determined the evolution of the VO paradigm over the years. Section 6 shows how the original paradigm has been adapted to solve real-time collision avoidance problem in real-world, laboratory or simulated scenarios. Section 7 explains the major drawbacks of Velocity Obstacle and successive formulations, and what has been done to remedy them. Finally in Section 8 we drawn a few conclusions and give directions for future research.

## 2. Decentralized methods

There exists several alternative decentralized approaches for motion planning of autonomous agents in dynamic environment, besides the Velocity Obstacle paradigm. In this section, we briefly describe the core of the most popular approaches, namely Potential Fields, Social Force, Dynamic Window and Model Predictive Controller, and provide some references for the reader.

We make this description for completeness and because the reader should be aware of the other principal decentralized motion planning methods, of the context in which they are adopted (e.g. robotics, human interactions in pedestrian dynamics), their strengths and limits in modeling real-world or simulated collision avoidance scenarios and the complexity of their mathematical formulation.

We believe that this preliminary presentation of alternative decentralized methods may help to understand the potential of Velocity Obstacles, as well as to have a general idea about the research field of decentralized motion planning for autonomous agents in dynamic environments.

### 2.1. Potential Fields and Social Force

In Potential Fields methods by Khatib [16] and Koditschek [28], robot $A$ is assumed to be a point on the Configuration Space $C$, under the action of an Artificial Potential Field (APF) $U$ that is a function of $q \in C$, i.e. position in the configuration space. The potential field $U$ is constructed in such a way that $A$ is globally attracted to the goal and at the same time is locally repulsed from obstacles. The motion planning technique for $A$ is based on the *steepest gradient descent*: if the robot $A$ is in a certain configuration $q$ at time $t$, it should move in the direction of the associated field of *artificial forces* $\vec{F}(q) = -\nabla U(q)$ getting to a new collision-free position. This procedure is re-iterated until the goal configuration, $q_{goal}$, is reached. The potential field is a map $U : C \to \mathbb{R}$ defined by $U(q) \triangleq U_{att}(q) + U_{rep}(q)$, i.e. the sum of an *attractive potential* $U_{att}(q)$ associated to $q_{goal}$ and a *repulsive potential* $U_{rep}(q)$ associated to the configuration space obstacle $C_o$. The attractive potential $U_{att}(q)$ is given by

$$U_{att}(q) \triangleq \begin{cases} \frac{1}{2}\xi\rho_g^2(q), & \text{if} \quad \rho_g(q) \le d \\ d\xi\rho_g(q), & \text{if} \quad \rho_g(q) > d \end{cases} \tag{1}$$

where $\xi$ is a scaling positive constant, $d$ is a scalar that represents a certain distance from the goal and $\rho_g(q) \triangleq \|q - q_{goal}\|_2$, with $\|\cdot\|_2$ being the Euclidean distance.

The first part of Eq. (1) is called *parabolic well potential* and it ensures stability and convergence to the goal of the motion planning technique, while the second part is called *conic well potential* and it ensures that $\vec{F}(q)$ remains bounded if $\rho_g(q)$ increases. The repulsive potential $U_{rep}(q)$ associated to $C_o$ is defined by

$$U_{rep}(q) \triangleq \begin{cases} \frac{1}{2}\eta\left(\frac{1}{\rho(q)} - \frac{1}{\rho_0}\right), & \text{if} \quad \rho(q) \le \rho_0 \\ 0, & \text{if} \quad \rho(q) > \rho_0 \end{cases} \tag{2}$$

where $\rho(q)$ is the minimum distance from $C_o$ to $q$, $\eta > 0$ is a scaling factor and $\rho_0 > 0$ is the distance beyond which the obstacle begins to repel the robot.

In this formulation, agent $A$ is represented by a point, thus it has only two degrees of freedom (i.e. two planar coordinates for its position), however in [29] the authors proposed a method based on APFs that can be applied to robots with several degrees of freedom.

The advantages of this method are the simple theoretical formulation and the speed of execution; the major inconvenience is represented by local minima: following the gradient descent, $A$ can get stuck in a point of local minima for the artificial force field $\vec{F}$ induced by the potential $U$, never reaching the goal. This happens for example in case of U-shaped obstacles. However, to overcome the problem, Barraquand et al. [30] proposed a graph-based method to escape local minima. Koren and Borestein [31] showed that potential fields methods may fail in finding smooth trajectories when obstacles are close to each other. Even though the original formulation of APF was designed to deal with fixed obstacles, many works Qixin et al. [32], Ge and Cui [33] and Poty et al. [34] extended the paradigm to dynamic environments in which obstacles and targets move.

The concept of *Social Force*, Helbing and Molnar [17], can be seen as an adaptation of Potential Fields approach applied to pedestrian dynamics. It is commonly believed that human behaviors are often unpredictable or, at least, irregular: robots have the sole objective of reaching their goal position while avoiding collisions with obstacles or other agents. Pedestrians want to do the same, but trying to satisfy other additional conditions due to their nature of human beings, e.g. respecting a certain social distance from other pedestrians that usually depends on the density of the crowd, not only avoid collisions with fixed obstacles, but also proceeding by staying at least at a certain distance from them, not getting too close to a region that is perceived as dangerous, etc. Lewin [35] suggested that behavioral changes of people are triggered by so called *social fields* or *social forces*, that are not real physical forces exerted on human body, but rather consequences of human perception of the environment. Humans quickly get used to situations they normally encounter for long periods of time, thus reactions are usually automatic and based on the personal experience of each pedestrian and this fact allows to model behavioral changes in pedestrians' velocities with equations of motion. Adding up we can

assert that pedestrians act as they are subjected to the action of external forces, induced by situations and the surrounding environment [36].

Let consider a pedestrian $\alpha$ that moves in a dynamic heterogeneous environment in presence of other pedestrians. The main effects that influence their motion are: (i) *desired direction*, (ii) *private sphere* and (iii) *attractive effects*. $\alpha$ has to reach a certain destination which position is given by $r_\alpha^0$ following the shortest possible path. The optimal path can be represented by a sequence of waypoints $r_\alpha^1, r_\alpha^2, \dots, r_\alpha^n := r_\alpha^0$. Between one waypoint and the next one, the *desired direction* is defined by

$$\vec{e}_\alpha(t) = \frac{r_\alpha^k - r_\alpha(t)}{\|r_\alpha^k - r_\alpha(t)\|_2}, \tag{3}$$

where $r_\alpha(t)$ denotes the current position and $r_\alpha^k$ is the actual waypoint to reach. Without any obstacles affecting the motion of $\alpha$, the person would proceed from a waypoint to the next one by moving with preferred velocity $\vec{v}_\alpha^p(t) = v_\alpha^p \vec{e}_\alpha(t)$. A voluntary deviation of the actual velocity $\vec{v}_\alpha(t)$ from the desired one, caused by the necessity to perform a collision avoidance maneuver or a deceleration, produces the tendency of the pedestrian to return to $\vec{v}_\alpha^p$ within a certain time $\tau$. This behavior can be described by the force

$$\vec{F}_\alpha^p\left(\vec{v}_\alpha, v_\alpha^p e_\alpha\right) \triangleq \frac{1}{\tau}(v_\alpha^p \vec{e}_\alpha - \vec{v}_\alpha). \tag{4}$$

Another essential effect that needs to be taken into account is the *private sphere* of each pedestrian [37]: if the environment allows it, $\alpha$ tends to keep a certain distance from a stranger $\beta$ who walks nearby him. This fact produces a repulsive effect that can be modeled as

$$f_{\alpha\beta}(\vec{r}_{\alpha\beta}) \triangleq -\nabla_{\vec{r}_{\alpha\beta}} V_{\alpha\beta}\left(b(\vec{r}_{\alpha\beta})\right), \tag{5}$$

where $\vec{r}_{\alpha\beta}$ is the relative distance between pedestrian and stranger, $V_{\alpha\beta}$ is the repulsive potential intended as a monotonically decreasing function of $b$, the semi-minor axis of an ellipse oriented according to the motion direction, given by

$$2b \triangleq \sqrt{(\|\vec{r}_{\alpha\beta}\|_2 + \|\vec{r}_{\alpha\beta} - s_\beta \vec{e}_\beta\|_2)^2 - s_\beta^2},$$

where $\vec{e}_\beta$ and $s_\beta = v_\beta \Delta t$ are the motion direction and the order of the step width of $\beta$, respectively.

A pedestrian wants also to keep a certain distance from borders of buildings, streets, walls and other similar obstacles [38]: so that their repulsive effects can be described by

$$\vec{F}_{\alpha B}(\vec{r}_{\alpha\beta}) \triangleq -\nabla_{\vec{r}_{\alpha\beta}} U_{\alpha B}(\|\vec{r}_{\alpha\beta}\|_2), \tag{6}$$

where $U_{\alpha B}(\|\vec{r}_{\alpha\beta}\|_2)$ is a repulsive monotonic decreasing potential and $\vec{r}_{\alpha B} = \vec{r}_\alpha - \vec{r}_B^{\,\alpha}$ is the relative distance between pedestrian and border.

A pedestrian is attracted by other people like friends or shop windows. The *attractive effects* are modeled as

$$\vec{f}_{\alpha i}(\|\vec{r}_{\alpha i}\|_2, t) \triangleq -\nabla_{\vec{r}_{\alpha i}} W_{\alpha i}(\|\vec{r}_{\alpha i}\|_2, t), \tag{7}$$

where $W_{\alpha i}$ are time-dependent attractive monotonically increasing potentials. $\vec{f}_{\alpha i}$ is assumed to decrease in norm with time $t$, because of pedestrians' loss of interest. Note that attractive and repulsive forces described by Eqs. (4)–(7) are really effective only in the direction of motion (3) and with a certain *effective angle of sight*, meaning that all the attractive and repulsive entities behind $\alpha$ have a significantly weaker magnitude. The *social force* is then given by

$$\frac{d\vec{w}_\alpha}{dt} = \vec{F}_\alpha(t) + \xi, \tag{8}$$

where $\vec{F}_\alpha(t)$ is the sum of all the attractive/repulsive forces in Eqs. (4)–(7), $\vec{w}_\alpha$ is the preferred velocity and $\xi$ is a random term that incorporates situations like ambiguities or behavioral alternatives for pedestrians. Furthermore, since the actual velocity of the pedestrian $\alpha$ is constrained to be smaller than a maximum acceptable speed we have a further equation given by

$$\frac{d\vec{r}_\alpha}{dt} = v_\alpha(t) = \vec{w}_\alpha(t)\, g\left(\frac{v_\alpha^{max}}{\|\vec{w}_\alpha\|_2}\right), \tag{9}$$

called *realized motion*, where

$$g\left(\frac{v_\alpha^{max}}{\|\vec{w}_\alpha\|_2}\right) \triangleq \begin{cases} 1, & \text{if } \|\vec{w}_\alpha\|_2 \le v_\alpha^{max} \\ v_\alpha^{max}/\|\vec{w}_\alpha\|_2, & \text{otherwise} \end{cases}.$$

Eqs. (8) and (9) are nonlinear coupled Langevin equations and constitute the pedestrian model. Applications of this model can be found in [39–41] and [42].

### 2.2. Dynamic Window

The Dynamic Window approach for online collision avoidance of autonomous robots has been proposed in [18]. It is derived from the dynamics of synchro-drive mobile robots, specifically designed to handle kinodynamic constraints of maximum velocity and acceleration typical of mobile robots and it has been successfully implemented on a B21 RHINO robot, manufactured by Real World Interface Inc. Synchro-drive robots are differential-drive robots, with three degrees of freedom $(x, y, \theta)$ expressing position and orientation. Letting $x(t_n)$, $y(t_n)$, $x(t_0)$, $y(t_0)$ and $\theta(t_n)$, $\theta(t_0)$ be the $(x, y)$-position and orientation of the robot at time $t_n$ and $t_0$, respectively, the equations of motion can be expressed as

$$x(t_n) = x(t_0) + \int_{t_0}^{t_n} v(t)\cos\theta(t)dt \tag{10}$$

$$y(t_n) = y(t_0) + \int_{t_0}^{t_n} v(t)\sin\theta(t)dt \tag{11}$$

with

$$v(t) = v(t_0) + \int_{t_0}^{t_n} \dot{v}(\hat{t})\, d\hat{t}$$

$$\theta(t) = \theta(t_0) + \int_{t_0}^{t}\left(\omega(t_0) + \int_{t_0}^{\hat{t}} \dot{\omega}(\tilde{t})\, d\tilde{t}\right) d\hat{t}$$

for $\hat{t}, \tilde{t} \in [t_0, t_n]$, where $v(t)$, $\omega(t)$ and $\dot{v}(t)$, $\dot{\omega}(t)$ are the current linear and angular velocities and accelerations. Since Eqs. (10)–(11) are too complex for a feasible implementation, by considering $v_i(t)$ and $\omega_i(t)$ constants on sufficiently small intervals $[t_i, t_{i+1}]$, for $i = 0, 1, \dots, n$, they can be discretized as

$$x(t_n) = x(t_0) + \sum_{i=0}^{n-1}\left(F_x^i(t_{i+1})\right) \tag{12}$$

$$y(t_n) = y(t_0) + \sum_{i=0}^{n-1}\left(F_y^i(t_{i+1})\right) \tag{13}$$

where

$$F_x^i(t) = \begin{cases} \frac{v_i}{\omega_i}\left[\mathrm{s}\theta(t_i) - \mathrm{s}(\theta(t_i) + \omega_i(t - t_i))\right], & \text{if } \omega_i \neq 0 \\ v_i\, \mathrm{c}\theta(t_i)\, t, & \text{if } \omega_i = 0 \end{cases}$$

$$F_y^i(t) = \begin{cases} \frac{v_i}{\omega_i}\left[\mathrm{c}(\theta(t_i) + \omega_i(t - t_i)) - \mathrm{c}\theta(t_i)\right], & \text{if } \omega_i \neq 0 \\ v_i\, \mathrm{s}\theta(t_i)\, t, & \text{if } \omega_i = 0 \end{cases}$$

The dynamic window approach is based on a search in the two-dimensional space of velocities $(v, \omega)$ controlling the robot. This search is restricted to velocities that are collision-free with respect to obstacles and satisfy the kinetic constraints of the robot itself.

The approach consists of two steps:

**Step 1**: Construction of the Search Space of the possible velocities, obtained in three passages

(i) *Circular Trajectories*: using the approximated equations of motion (12)–(13) it can be proved that all the feasible trajectories are combinations of straight lines and circular arcs. The curvature of each piece of trajectory is uniquely determined by the pair $(v, \omega)$ of translational and rotational velocities, resulting in a two-dimensional search space. In order to determine a trajectory on $[t_0, \dots, t_n]$ to the goal position not intersecting obstacles, the robot has to find $(v_i, \omega_i)$ for $i = 0, \dots, n$. The Dynamic Window approach computes only the velocities at the first

time interval and considers the other $n-1$ to be constant, i.e. zero acceleration from $t_1$ to $t_n$. This simplification is made to manage the complexity of the search space for velocity vectors $(v, \omega)$.

(ii) *Admissible Velocities*: a velocity $(v, \omega)$ is considered to be admissible if the robot can stop its motion before colliding with any obstacle in the relative section of trajectory. The set of admissible velocities is defined as

$$V_a \triangleq \left\{ v, \omega \ : \ v \leq \sqrt{2\,\mathrm{d}(v,\omega)\dot{v}_b} \wedge \omega \leq \sqrt{2\,\mathrm{d}(v,\omega)\dot{\omega}_b} \right\},$$

where $\dot{v}_b$ and $\dot{\omega}_b$ are the brake decelerations and $d(v, \omega)$ is the clearance function of $(v, \omega)$, representing the distance to the closest obstacle intersecting the current trajectory section. In case of no obstacles, the clearance is set to a very large constant value.

(iii) *Dynamic Window*: the overall search space for velocities is further reduced since the motors have bounded accelerations. The reduction is the Dynamic Window, the set of velocities given by

$$V_d \triangleq \left\{ v, \omega \ : \ v \in V(v_a) \wedge \omega \in W(\omega_a) \right\},$$

where $(v_a, \omega_a)$ are the current linear and angular velocities, $V(v_a) = [v_a - \dot{v}t, v_a + \dot{v}t]$ and $W(\omega_a) = [v_a - \dot{v}t, v_a + \dot{v}t]$ are intervals whose length depends on the maximum possible accelerations. The total search space $V_r$ is then

$$V_r = V_s \cap V_a \cap V_d, \tag{14}$$

where $V_s$ is the space of all possible velocities.

**Step 2**: Maximization over the space $V_r$ given by Eq. (14) of a gain function $G(v, \omega)$ defined by

$$G(v,\omega) \triangleq \sigma\left(\alpha\,\mathrm{head}(v,\omega) + \beta\,\mathrm{d}(v,\omega) + \gamma\,\mathrm{vel}(v,\omega)\right),$$

where the "head" term represents a measure of the progress of the robot towards the goal position, the "d" term is the distance to the closest obstacle (it represents the clearance), the "vel" term is the forward velocity of the robot and $\sigma$ is a function that smoothes the sum of these three components.

The dynamic window is based on the optimization of the gain function $G$ of the translational and angular velocities of the robot, over a three-dimensional space of velocities that satisfy constraints on both admissibility and time availability. A similar constrained optimization for local collision avoidance has been proposed by Simmons [43]. An extension of the dynamic window method, called *global dynamic window* (GDW), allows the execution of high velocity motion for non-holonomic robots in unknown and dynamic environments [44]. Ogren and Leonard [45] addressed the convergence properties of the method, in order to solve known deadlock situations.

The Dynamic Window has the ability to quickly adapt to changes in the environment and the robot's dynamics, making it suitable for real-time applications where robots need to navigate in dynamic and uncertain environments while avoiding collisions. However, the approach presents some drawbacks including limited scalability in case of high-dimensional state spaces, that increase the computational complexity of evaluating and searching through potential trajectories and its tendency to getting stuck in local minima, where the robot may be unable to find a collision-free path due to its limited exploration capabilities.

### 2.3. Model Predictive Control

Consider a number $N_v$ of agents, each one described by a discrete-time equation of the type

$$x^i_{k+1} = f^i(x^i_k, u^i_k), \quad k \geq 0, \quad i = 1, \dots, N_v \tag{15}$$

where $x^i_k \in \mathcal{X}^i \subseteq \mathbb{R}^{n^i}$ and $u^i_k \in \mathcal{U}^i \subseteq \mathbb{R}^{m^i}$ are the states and inputs of every system.

The $N_v$ Eqs. (15) define the *Team System*, denoted by $x_{k+1} = f(x_k, u_k)$ where $x_k = (x^1_k, \dots, x^{N_v}_k) \in \mathbb{R}^{\sum_{i=1}^{N_v} n^i}$ and $u_k = (u^1_k, \dots, u^{N_v}_k) \in \mathbb{R}^{\sum_{i=1}^{N_v} m^i}$. The equilibrium points of the $i$th system is $(x^i_e, u^i_e)$, whereas $(x_e, u_e)$ is the equilibrium point of the team system.

The team coupling is represented by a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{A}\}$ where $\mathcal{V} = \{1, \dots, N_v\}$ is the set of nodes associated to each agent and $\mathcal{A} = \mathcal{V} \times \mathcal{V}$ is the set of bidirectional arcs $(i, j)$ connecting the $i$th agent to its $j$th neighboring one. Let $\tilde{x}^i = \{x^j \in \mathbb{R}^{n^j} \ : \ (j,i) \in \mathcal{A}\}$ be the states of all the neighboring agents of the $i$th one and, analogously, $\tilde{u}^i = \{u^j \in \mathbb{R}^{m_j}\}$ be the inputs of these systems.

The interconnection constraints between $i$th agent and all its neighbors is described by the map $g^i(x^i, u^i, \tilde{x}^i, \tilde{u}^i) : \mathbb{R}^{n^i} \times \mathbb{R}^{m^i} \times \mathbb{R}^{\tilde{n}^i} \times \mathbb{R}^{\tilde{m}^i} \to \mathbb{R}^{n_{c,i}}$ where $\tilde{n}^i = \sum_{j|(j,i)\in\mathcal{A}} n^j$ and $\tilde{m}^i = \sum_j m^j$.

*Model Predictive Control* (MPC) consists in solving a series of optimal control problems over discrete time intervals of the form $[k, \dots, k+N]$, $k = 0, 1, 2, \dots$, given by

$$J^*(\tilde{x}_k) \triangleq \min_{U_k} \sum_{h=0}^{N-1} \ell\left(\tilde{x}_{k,h}, \tilde{u}_{k,h}\right) + \ell_N\left(\tilde{x}_{k,N}\right) \tag{16}$$

$$\text{subj.to} \begin{cases} x^i_{k+1,h} = f^i(x^i_{k,h}, u^i_{k,h}) \\ g^{ij}(x^i_{k,h}, u^i_{k,h}, x^j_{k,h}, u^j_{k,h}) \leq 0 \\ \tilde{x}_{k,0} = \tilde{x}_k \end{cases} \tag{17}$$

where $N$ is the *prediction horizon*, $h = 0, \dots, N-1$, $U_k = \{u_{k,0}, \dots, u_{k,N-1}\}$, $\tilde{x}_{k,N} \in \mathcal{X}_f \subset \mathbb{R}^{\sum_{i=1}^{N_v} n^i}$ and $x^i_{k,h} \in \mathcal{X}^i$ is the state of $i$th agent predicted at time $h+k$, $\ell$ is the convex cost function associated to the Team System, given by

$$\ell\left(\tilde{x}_{k,h}, \tilde{u}_{k,h}\right) = \sum_{i=1}^{N_v} \ell^i\left(x^i, u^i, \tilde{x}^i, \tilde{u}^i\right),$$

with $\ell^i$ convex cost function of every single agent, $\ell_N$ is the cost on the terminal state. If problem (16)–(17) is feasible, the solution is a sequence of optimal control inputs $\{u^*_{k,0}, u^*_{k,1}, u^*_{k,2}, \dots\}$ that lead the team to the equilibrium point.

Discussions about the sufficient conditions on the terminal cost function $\ell_N$ and the terminal region $\mathcal{X}_f$ that ensures stability of the closed-loop control system can be found in [10,46].

Keviczky et al. [19] used MPC technique for formation control and collision avoidance of UAVs forced to fly at certain fixed altitude. Their decentralized control scheme is explainable in four steps

1. the $i$th node, (i.e $i$th agent), measures its state $x^i_k$ and the state of its neighbors $\tilde{x}^i_k$ at time $k$,
2. each node $i$ solves a problem $P_i$ in the form (16)–(17),
3. once the optimal solution is found, each node $i$ implements the first sample of $\bar{U}^{i*}_k$, $u^i_k = u^{i*}_{k,0}$,
4. each node repeats steps from 2 to 4 at the next time-step, based on the new state information $x^i_{k+1}$.

The UAVs are modeled as two-dimensional points of mass which dynamics is described by a linear state-space discrete model which state $x_k \in \mathbb{R}^4$ contains position and velocity coordinates, and input vectors $u_k \in \mathbb{R}^2$ are the accelerations along the $x$ and $y$ axes, respectively. Park et al. [47] proposed an obstacle avoidance scheme for UGVs where safe trajectories are generated by adopting a non-linear MPC approach, in order to better approximate the dynamics of each vehicle. In a similar work Jiang et al. [48] designed an MPC scheme based on convex quadratic programming (CQP) that provides safe trajectories for autonomous vehicles which dynamics is described by a linear time-varying equation, obtained by performing a linear approximation of a non-linear time-varying model.

MPC is a powerful motion planning approach, but in case of non-linear dynamic models, it is necessary to adopt linearization techniques to guarantee sufficiently fast control clock. Moreover, it requires high-performance hardware for its deployment, in real-world applications.

**Table 1**
VO symbols list.

| Symbol | Meaning |
|---|---|
| $A$ | Autonomous agent $A$ |
| $\hat{A}$ | $A$ represented in its configuration space |
| $B$ | Moving obstacle or agent $B$ |
| $\hat{B}$ | $B$ represented in the config. space of $A$ |
| $\mathcal{C}$ | Configuration space or configuration manifold |
| $CA_A$ | Collision avoiding velocities of $A$ |
| $CC_{A_i}$ | Absolute collision cone of $A$ |
| $CC_{ABi}$ | Relative collision cone of $A$ induced by $B_i$ |
| $FA(t_0)$ | Set of feasible accelerations at time $t_0$ |
| $MVO_A$ | Multiple velocity obstacle of agent $A$ |
| $p(t)$ | Motion in $\mathrm{T}\mathbb{R}^n \simeq \mathbb{R}^n \times \mathbb{R}^n$ |
| $r_A$ | Radius of agent $A$ |
| $RAV(t)$ | Reachable avoidance velocities at time $t$ |
| $RV(t)$ | Reachable velocities at time $t$ |
| $\mathrm{T}\mathbb{R}^n$ | Tangent bundle of $\mathbb{R}^n$ |
| $trj_{AB_i}$ | Trajectory of $A$ with respect to $B_i$ |
| $v_A$ | Current velocity of $A$ |
| $v_{AB}$ | Relative velocity of $A$ w.r.t $B$ |
| $v_A^{pref}$ | Preferred velocity of $A$ |
| $v_A^{new}$ | New collision-free velocity for $A$ |
| $VO_{AB_i}$ | Vel. obstacle of $A$ induced by $B_i$ |
| $x_A$ | Position of the $A$ |
| $x_{AB}$ | Relative position of the $A$ w.r.t $B$ |
| $x_{g_A}$ | Goal position for $A$ |
| $x_p$ | Position of point $p \in \mathbb{R}^n$ |
| $v_p$ | Velocity of point $p \in \mathbb{R}^n$ |
| $\oplus$ | Minkowski Sum |

## 3. The Velocity Obstacle paradigm

The Velocity Obstacle (VO) paradigm has been formalized in [21] by summing up several previous works Fiorini and Shiller [20,49,50,51].

Fiorini and Shiller [20] extended the Configuration Space Obstacle by Lozano-Perez [22] to time-varying environments. The Configuration Space or Configuration Manifold $\mathcal{C}$ is the space of all the configurations that a mechanical system, defined as a collection of points and rigid bodies, can assume. The Configuration Space Obstacle represents all the points of the Configuration Space occupied by an obstacle. Table 1 summarizes the symbols related to the Velocity Obstacle, to facilitate the reading of the following section.

### 3.1. Velocity Obstacle

Fiorini and Shiller [20] proposed the *Relative Velocity Paradigm* (RVP) to detect possible collisions between an autonomous agent and moving obstacles, and also to compute evasive maneuvers. The dynamic environment consists of an agent $A$ and moving obstacles $B_i$, $i = 1, \dots, m$, that are assumed to be holonomic and disc-shaped.

The agent is characterized by position $x_A$, radius $r_A$ and velocity $v_A$ and it is able to detect the position $x_{B_i}$, radius $r_{B_i}$ and velocity $v_{B_i}$ of every obstacle $B_i$. The RVP allows to represent the dynamic environment into the Velocity Space that, for entities that move in a two or three dimensional space, is represented by the tangent bundle $\mathrm{T}\mathbb{R}^n$ of $\mathbb{R}^n$, $n = 2, 3$ that in this case can be represented by $\mathbb{R}^n$ itself.

The motion of a point $p$ on the Velocity Space is given by

$$p(t) = \left(x_p(t), v_p(t)\right) \in \mathrm{T}\mathbb{R}^n \simeq \mathbb{R}^n \times \mathbb{R}^n,$$

where $x_p$ and $v_p$ are position and velocity of $p$, respectively. Let

$$v_{AB_i} = v_A - v_{B_i} \quad i = 1, \dots, m,$$

be the relative velocity of $A$ with respect to $B_i$. The corresponding relative trajectory in the velocity space is defined as

$$trj_{AB_i} = \{(x, \dot{x}) : x(t_0) = x_{AB_i}, \dot{x}(t_0) = v_{AB_i}\}.$$

It is now possible to state the core of Relative Velocity Paradigm: *A collision between $A$ and $B_i$ occurs if and only if the relative velocity $v_{AB_i}$ does not change and $trj_{AB_i} \cap B_i \neq \emptyset$*. The set of all relative velocities for which the statement is satisfied is defined as

$$CC_{AB_i} \triangleq \{v_{AB_i} \mid trj_{AB_i} \cap B_i \neq \emptyset\}$$

and it is called *Relative Collision Cone* of $A$ induced by $B_i$. Geometrically, it can be constructed on the two-dimensional plane by using the concept of Configuration Space Obstacle, by reducing the agent $A$ to a single point $\hat{A}$ and enlarging the obstacle $B_i$ by the radius of $A$, obtaining $\hat{B}_i$. The Relative Collision Cone is delimited by two straight lines tangent to $\hat{B}$ and whose intersection point is $\hat{A}$. The Velocity Obstacle $VO_{AB_i}$ (Fig. 1) is the set of all the collision velocities that the agent $A$ must avoid

$$VO_{AB_i} \triangleq \{v_A \mid v_{AB_i} \in CC_{AB_i}\}.$$

Geometrically it is obtained by translating the relative collision cone $CC_{AB_i}$ by $v_{B_i}$ via Minkowski sun, i.e.

$$VO_{AB_i} = CC_{AB_i} \oplus v_{B_i}.$$

Agent $A$ does not collide with $B_i$ as long as its velocity remains outside $VO_{AB_i}$. To avoid collisions with multiple obstacles $B_1, \dots, B_m$, agent $A$ must select a velocity outside the *Multiple Velocity Obstacle* (MVO)

$$MVO_A \triangleq \bigcup_{i=1}^{m} VO_{AB_i}.$$

This approach works also in the three-dimensional space, where agent $A$ and obstacles $B_i$ are modeled as spheres and the relative and absolute collision cones are sets of velocities in $\mathbb{R}^3$.

### 3.2. Reachable avoidance velocities

Fiorini and Shiller [50,51] proposed the name Velocity Obstacle in place of Relative Velocity and described how to construct the set of Reachable Avoidance Velocities (RAV) for a robot characterized by a dynamics.

Suppose that the motion of the robot $A$ is described by $\ddot{x} = f(x, \dot{x}, u)$ where $x$ is the position, $\dot{x}$ is the velocity and $u \in \mathcal{U}$, a set of controls. The set of *Feasible Accelerations* (FA) at time $t_0$, $FA(t_0)$, is defined by

$$FA(t_0) = \{\ddot{x} \mid \ddot{x} = f(x, \dot{x}, u), u \in \mathcal{U}\}.$$

An agent $A$ with dynamics $\ddot{x} = f(x, \dot{x}, u)$ at time $t_0$ changes its velocity at time $t_0 + \Delta t$ by selecting it among all those contained into the set of all *Reachable Velocities*

$$RV(t_0 + \Delta t) = \left\{v \mid v = v_A(t_0) \oplus \Delta t \cdot FA(t_0)\right\}.$$

To avoid the obstacle $B_i$, $i = 1, \dots, m$, agent $A$ has to select its new velocity from the *Reachable Avoidance Velocities* (RAV) set

$$RAV(t_0 + \Delta t) = RV(t_0 + \Delta t) \setminus VO(t_0).$$

## 4. Taxonomy

The major contributions that extended the original Velocity Obstacle paradigm over the last 25 years are shown in Fig. 2. Starting from the original *Velocity Obstacle* paradigm (VO, Fiorini and Shiller [21]), around 2008–2009 there is the first cluster of important extensions given by *Reciprocal Velocity Obstacle* (RVO, Van Den Berg et al. [23]), *Finite-Time Velocity Obstacle* (FVO, Guy et al. [52]) and *Optimal Reciprocal Collision Avoidance* (ORCA, Van Den Berg et al. [24]).

The first work Van Den Berg et al. [23] introduced the reciprocal collision avoidance, i.e. the autonomous robots share responsibility in performing collision avoidance maneuvers with respect each other, improving the algorithm performance in multi-agent systems. The second work Guy et al. [52] focused on avoiding sub-optimal collision-free maneuvers between agents by introducing the concept of truncated

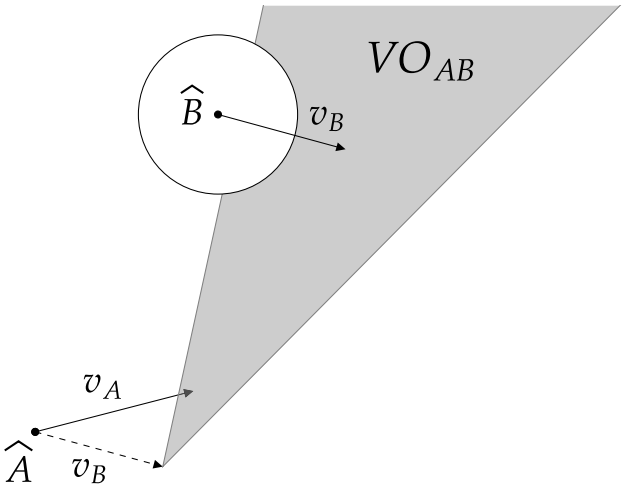



**Fig. 1.** The Velocity Obstacle of agent $A$ with respect to obstacle $B$.

collision cones, that allow to impose a policy among agents and moving obstacles. The last work Van Den Berg et al. [24] of the group incorporates reciprocal collision avoidance and truncated cones with linear constraints optimization for collision-free velocity optimal selection.

The *Generalized Velocity Obstacle* (GVO, Wilkie et al. [53]) extended the original paradigm to car-like robots in the same year, to overcome the holonomic robots assumption. Shiller et al. [54] proposed the *Non-Linear Velocity Obstacle* (NLVO), a direct extension of the original paradigm based on warped collision cones design specifically to deal with obstacles that follow non-linear known trajectories.

The *Hybrid Reciprocal Velocity Obstacle* (HRVO, Snape et al. [26]) extends the RVO paradigm in order to solve the problem of reciprocal dances, i.e. an undesired phenomenon that induces two agents to perform oscillatory trajectories while trying to avoid colliding each other. The method is based on enlarged collision cones that allow a robot to identify which is the most natural direction to take while avoiding collision with another robot. The HRVO paradigm solved the reciprocal dances, but the obtained trajectories may be non-smooth in crowded dynamic environments.

The ORCA paradigm [24] on this regard, is able to produce smooth and non-oscillating trajectories also in crowded environments, but it has the problem of deadlocks, i.e. an event for which the linear optimization that characterized the method leads an autonomous agent to select a collision-free velocity vector equal to zero, forcing it to stop and never reaching the target.

Section 7.3 accounts this problem and explains how Battisti and Muradore [55] analyzed this issue and proposed a solution based on *dummy* goals.

*Acceleration Velocity Obstacle* (AVO, Van Den Berg et al. [25]) includes constraints on maximum acceleration for mobile robots in the ORCA paradigm. This allows to cope with instantaneous changes of velocity that are non realistic in real world applications.

The year 2013 marks the arrival of two other important contributions: *Optimal Reciprocal Collision Avoidance for Multiple Non-holonomic Robots* (NH-ORCA, Alonso-Mora et al. [27]) and *Goal Velocity Obstacles for Spatial Navigation of Multiple Autonomous Robots or Virtual Agents* (Goal VO, Snape and Manocha [56]). The first directly extends the ORCA paradigm to the case of differential-drive robots, the most common type of autonomous robots operating in automatic warehouses and other industrial facilities; the second one exploits the collision cones not only to model velocities that lead to collision, but also those that lead to a desired and possibly moving goal regions. This is justified by the fact that in several applications many robots may have to share the same goal position, such as batteries recharge spots.

In 2015, Bareiss and Van Den Berg [57] generalized the problem of collision avoidance in order to propose a unified framework where vehicles with different non-linear dynamics can be considered at the same time, since until then only homogeneous multi-agent systems where considered.

Finally, Kim and Oh [58] proposed the last direct refinement to the original Velocity Obstacle paradigm that allows the robots to prioritize stopping in place of avoidance maneuvers. The motivation behind this contribution is that when moving obstacles are very fast, the most natural action is to stop the robot and wait until the obstacle has gone away, instead of performing a hazardous collision free maneuver at all costs.

## 5. Extensions and improvements

Collision avoidance of autonomous robots using the original Velocity Obstacle has some limitations and drawbacks. VO algorithms may produce non-smooth trajectories for an agent that deals with a large number of moving obstacles and sub-optimal trajectories in multi-agent scenarios. Moreover, it assumes the perfect sensing of position, radius and velocity of every obstacle and other agents at any distance. This is not realistic in real-world applications, since on-board sensors are characterized by a maximum range and are affected by noise, causing uncertainty about the sensed pose of the other robots. The problem of noisy sensing is addressed in Section 7.2.

Throughout this section we will see how the original VO paradigm has been extended in order to provide smooth and optimal collision-free trajectories. Although Section 3.2 describes how to construct a set of feasible collision-free velocities in presence of robots with kinodynamic constraints, in most extended versions of VO paradigm the robots/agents are assumed to be holonomic and this assumption is not realistic in real-world scenarios. For this reason, we will also discuss contributions regarding the adaptation of some of these extensions to the case of wheeled non-holonomic robots. It is worth highlighting that such extensions have been formulated as a generalized collision avoidance problem in [57].

We will also discuss some extensions that do not radically modify the VO paradigm in its geometrical aspects, but rather integrate it with interesting features such as the possibility of having moving regions as targets for the autonomous agents and an optimal collision-free velocity selection policy.

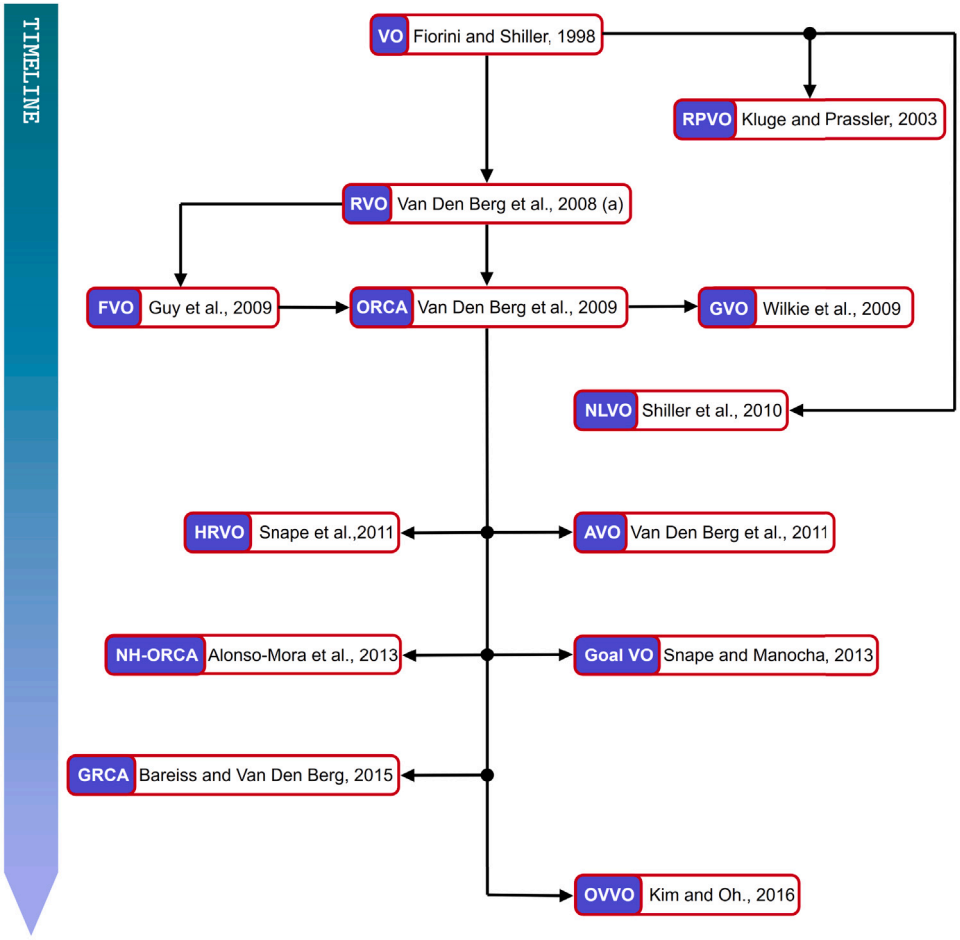



**Fig. 2.** Taxonomy showing the evolution of the paradigm.

### 5.1. Reciprocal Velocity Obstacle

Van Den Berg et al. [23] provided one of the first adaptation to multi-agent frameworks. In this formulation all the agents are meant to share the responsibility to avoid collision. Agents are decision making entities with the same collision avoidance policy.

Consider an agent $A$ that has to avoid collisions with another agent $B$ traveling in the same workspace. At the current time $t$, $A$ moves with current velocity $v_A$ and computes the velocity obstacle $VO_{AB}$ with respect to $B$. Also $B$, moving with velocity $v_B$ computes a velocity obstacle cone $VO_{BA}$. If the agents are on a collision course, it means that at time $t$ we have that $v_A \in VO_{AB}$ and $v_B \in VO_{BA}$. In order to perform a correct maneuver, at time $t+\Delta t$, $A$ must select a new velocity vector $v_A^{new} = v_A + w$ where $w \in \mathbb{R}^2$ is vector such that $v_A + w \notin VO_{AB}$ and $B$ does the same selecting $v_B^{new} = v_B + w \notin VO_{BA}$.

Such velocity selection produces sub-optimal collision avoidance maneuvers, since the two agents would go more far away than necessary. To restore optimality, we must impose that $A$ selects $v_A^{new} = v_A + w/2$ provided that $B$ does the same by choosing $v_B^{new} = v_B + w/2$. This means sharing the responsibility among agents in computing collision avoidance maneuvers.

Formally, this means that $A$ must select a new velocity vector whose tip is outside of the *Reciprocal Velocity Obstacle* of $A$ with respect to agent $B$, given by

$$RVO_{AB} = \left\{ v_A^{new} \mid 2v_A^{new} - v_A \in VO_{AB} \right\}.$$

From a geometric point of view $RVO_{AB}$ is a cone equal to $VO_{AB}$, but translated by $(v_A + v_B)/2$ instead of $v_B$ as shown Fig. 3. Let us consider a multi-agent scenario where a certain number $n \in \mathbb{N}$ of agents $A_1, \dots, A_n$ move in the same work space. Suppose that every agent $A_i$ has a preferred velocity $v_{A_i}^{pref}$, i.e. the velocity vector leading to its goal position $x_{g_{Ai}}$ in absence of obstacles. At every cycle of the algorithm, $A_i$ selects a new velocity vector such that

$$v_{A_i}^{new} = \underset{v \notin RVO_{A_i}}{\arg\min} \| v - v_{A_i}^{pref} \|_2,$$

where $RVO_{A_i}$ is given by

$$RVO_{A_i} = \bigcup_{j \neq i}^{n} RVO_{A_i A_j}.$$

The Relative Velocity Obstacle paradigm represents the most straightforward extension of the original Velocity Obstacle to multi-agent dynamic environments, providing on-line safe navigation trajectories for every decision making entity. However, it has the issue of *reciprocal dances*, i.e. an oscillating behavior occurring when two agents $A$ and $B$ navigate one towards the other. Reciprocal dances are due to the fact that once $A$ and $B$ have updated their current velocities with the new ones and moved to the new positions, $x_A^{new}$ and $x_B^{new}$, the old velocities might return to be collision free in the successive cycle of the algorithm, making agents $A$ and $B$ adopting them once again and so on (see Fig. 4). This fact could bring to a deadlock.

In Section 5.6, we will see how this problem has been solved by Snape et al. [26].

### 5.2. Optimal Reciprocal Collision Avoidance

Van Den Berg et al. [24] presented an Optimal Reciprocal Collision Avoidance (ORCA) based on the original Velocity Obstacle (Section 3)

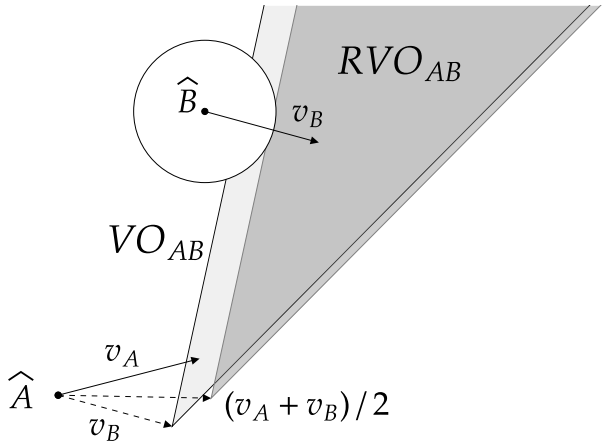



**Fig. 3.** Velocity Obstacle $VO_{AB}$ (light gray cone) compared to the Relative Velocity Obstacle $RVO_{AB}$ (dark gray cone). Note that the apex of $RVO_{AB}$ on the tip of $(v_A+v_B)/2$ vector.

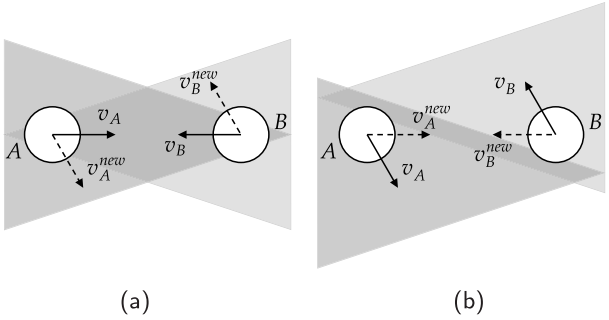



**Fig. 4.** Reciprocal dance between $A$ and $B$. (a) Each agent chooses the closest velocity to the current one outside the VO induced by the other, then (b) at the next sense-and-acting step each robot has attained its new velocity leaving the other one outside of the current velocity obstacle, that will be selected again in the following step.

and on the Reciprocal Velocity Obstacle (Section 5.1). The selection of collision avoidance velocity for every agent $v_A^{new}$ is carried out by solving a linear programming problem in which the constraints are the straight lines defining the half-planes of feasible velocities.

The algorithm can be described as follows. Suppose an agent $A$, moving with $v_A$, has to avoid collisions with $B$ moving with velocity $v_B$. The original VO paradigm allows to construct the cone $VO^{\tau}_{AB}$ of all velocities for $A$ that can cause a collision with $B$ within a certain time horizon $\tau > 0$,

$$VO^{\tau}_{AB} \triangleq \left\{v \mid \exists t \in [0,\tau],\, tv \in \mathcal{D}(x_B - x_A, r_A + r_B)\right\}$$

where $x_i$ and $r_i$, $i \in \{A, B\}$, denote the positions and radii of the agents, respectively. $\mathcal{D}$ is a disc centered in $x_B - x_A$ of radius $r_A + r_B$ as depicted in Fig. 5

If $A$ is on a collision course with $B$, it means that $v_{AB} \in VO^{\tau}_{AB}$. Let $w$ be the vector from $v_{AB}$ to the closest point of the boundary $\partial VO^{\tau}_{AB}$ of the Velocity Obstacle

$$w = (\underset{v \in \partial VO^{\tau}_{AB}}{\arg\min} \|v - v_{AB}\|_2) - v_{AB},$$

and $n$ be the outward normal vector at point $v_{AB} + w \in \partial VO^{\tau}_{AB}$. Then, the set of all collision free velocities for $A$ is defined by the half-plane pointing in the direction $n$ originating at the point $v_A + w/2$,

$$ORCA^{\tau}_{AB} \triangleq \{v | (v - (v_A + w/2)) \cdot n \geq 0\}.$$

The set $ORCA^{\tau}_{BA}$ for $B$ is defined in the same manner, see Fig. 6. If $A$ has to avoid collisions with a certain number of other agents $B_1, \ldots, B_m$,

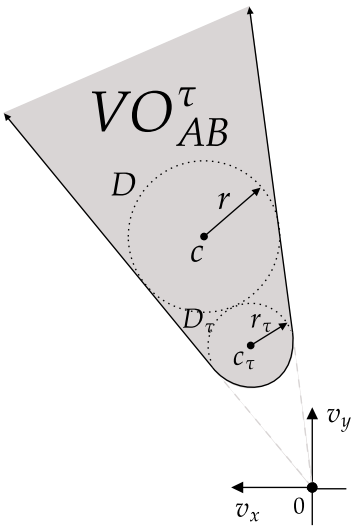



**Fig. 5.** The truncated cone $VO^{\tau}_{AB}$. $D$ is a disc of radius $r = r_A + r_B$ centered in $c = x_B - x_A$ tangent to the cone edges and $D_\tau$ is the cut-off circle whose radius $r_\tau$ and center $c_\tau$ are scaled by $\tau > 0$.

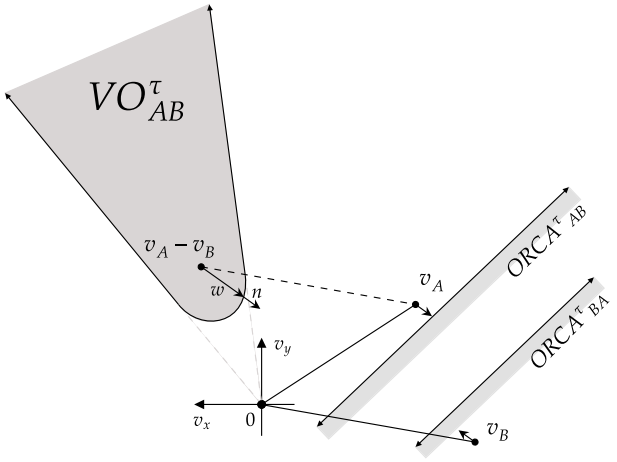



**Fig. 6.** The $ORCA^{\tau}_{AB}$ half-plane for $A$ induced by $B$ and its counterpart, $ORCA^{\tau}_{BA}$, for $B$ induced by $A$.

then the set of all the permitted velocities for agent $A$ is given by

$$ORCA^{\tau}_{A} \triangleq \mathcal{D}(0, v_A^{max}) \cap \bigcap_{i=1}^{m} ORCA^{\tau}_{AB_i}, \tag{18}$$

where $\mathcal{D}(0, v_A^{max})$ is a disc centered in the origin with radius given by the maximum speed that an agent can reach. The new collision-free velocity for agent $A$, is given by

$$v_A^{new} = \underset{v \in ORCA^{\tau}_{A}}{\arg\min} \|v - v_A^{pref}\|_2.$$

$ORCA^{\tau}_{A}$ is a convex region and the straight lines that defines the half-planes are the constraints in a linear programming problem, whose solution is $v_A^{new}$ at every cycle of the algorithm. The idea of adopting truncated cones in VO and RVO paradigms that allow to take into account the proximity of collision has been proposed in [52] under the name *Finite-Time Velocity Obstacle* (FVO). The algorithm is able to provide optimal and non-oscillating safe trajectories; however, two agents $A$ and $B$ may get stuck in a deadlock situation, e.g., while navigating along a straight line, one towards the other, having each one the purpose to reach a goal position located beyond the other on

a straight line. We will see in Section 6.4 a possible solution to the problem, Battisti and Muradore [55].

### 5.3. Generalized velocity obstacle

Wilkie et al. [53] presented a generalization of the original VO paradigm to non-holonomic car-like robots driven by some controls $u$ belonging to a control set $\mathcal{U}$.

Consider a car-like robot $A$ that has to avoid collisions with a moving obstacle $B$. The approach identifies all the controls $u$ that cause a collision between the robot and $B$ in the future. Let $u^{pref}$ be the preferred controls that would lead the robot to its goal in absence of obstacles. The set of controls $\mathcal{U}_c \subset \mathcal{U}$ that will cause a collision at time $t > 0$ with a moving obstacle $B$ is defined as

$$\mathcal{U}_c \triangleq \left\{u \,|\, \exists\, t > 0 \; s.t. \; \|x_A(t,u) - x_B(t)\|_2 < r_A + r_B\right\},$$

where $x_A(t,u)$ is the position of agent $A$ at time $t$ given the control $u$ and $x_B(t)$ is the position of obstacle $B$ at time $t$. The first step is to find the minimum $t_{min}$, solving $\frac{d}{dt}\|x_A(t,u) - x_B(t)\|_2 = 0$ in order to check if the actual control for robot $A$ is collision free or not. If a collision is foreseen, then a new control has to be selected according to

$$u = \underset{u' \notin VO_{AB}}{\arg\min} \|u^{pref} - u'\|_2,$$

as the closest command to $u^{pref}$ that does not belong to $VO_{AB}$.

### 5.4. Non linear velocity obstacle

Shiller et al. [54] introduced the Non-Linear Velocity Obstacle (NLVO) as a modification of the original VO paradigm for robot $A$ that travels in a dynamic environment populated by moving obstacles $B_i$ that follow a general non-linear *known* trajectory $c(t)$, for $t \in [0, +\infty)$.

The paradigm is stated considering $A$ as a single point, but it can be extended also to vehicles characterized by more complex kinematics or dynamics. Given the current time $t_0 \geq 0$, the infinite-time horizon non-linear velocity obstacle $NLVO^{\infty}_{t_0}$ is given by

$$NLVO^{\infty}_{t_0}(t) \triangleq \bigcup_t NLVO(t), \tag{19}$$

where $NLVO(t)$ is the set of all velocities $v_A$ of $A$ that would cause a collision with $B(t)$, where $B(t)$ denotes the region of the plane that at time $t$ is occupied by $B$. If $p \in B(t)$, a velocity that would cause a collision with $p$ at $t > t_0$ can be expressed in closed form by

$$v = \frac{c(t) + p_r}{t - t_0} = H_{A,k}(c(t) + p_r), \quad k = \frac{1}{t - t_0}, \tag{20}$$

where $p_r$ is the vector connecting $c(t)$ to $p$ and $H_{A,k}$ is the homothetic transformation, centered in $A$ and scaled by $k$, of the point $c(t) + p_r$.

$NLVO(t)$ is then given by applying (20) to every point of $B(t)$

$$NLVO(t) = H_{A,k}(B(t)), \quad k = \frac{1}{t - t_0}.$$

Contrary to what happens in the original VO, where $VO_A(t)$ is a proper cone, the $NLVO_A(t)$ appears to be a warped cone originated from $A$.

The NLVO paradigm can be improved by considering only collisions that may occur within a certain finite time-horizon $\tau$: it will not construct (19), but an analogous truncated warped cone $NLVO^{\tau}_{t_0}(t)$. The optimal value for $\tau$ is found by solving a minimization problem of the form

$$\min \int_{t_0}^{\tau} 1 \, dt$$

subjected to

1. initial conditions $x(t_0)$ and $\dot{x}(t_0)$,
2. terminal condition $x(t_h) \notin NLVO^{\infty}_{\tau}$,
3. vehicle dynamics $\ddot{x} = f(x, \dot{x}, u)$, $u \in \mathcal{U}$.

A robot $A$ characterized by some dynamics selects a new velocity $v_A(t_0 + \Delta t)$ belonging to $NLVO^{\tau}_{t_0}(t)^C \cap ACV(t)$, where

$$ACV(t) \triangleq \{v \mid v = v(t) + u\Delta t, \, u \in \mathcal{U}\}$$

is the set of all *attainable Cartesian velocities* and

$NLVO^{\tau}_{t_0}(t)^C$ is the complementary set of $NLVO^{\tau}_{t_0}(t)$. The shape of $ACV$ depends on the dynamics itself. The major disadvantage of this approach is that the planner is able to guarantee reachability of the goal position in optimal time only in presence of just one moving obstacle.

### 5.5. Acceleration velocity obstacle

Van Den Berg et al. [25] improve the ORCA paradigm by considering mobile robots subjected to acceleration constraints. The introduction of this type of constraints is motivated by the necessity to model robot and other autonomous vehicle moving at high speed (e.g. aerial vehicles like UAVs).

Consider a robot $A$ that has to avoid collisions with another robot $B$ within time $\tau$. At every sensing-and-acting cycle, the idea is to substitute the collision velocity cone $VO^{\tau}_{AB}$ with a new one called *Acceleration-Velocity Obstacle for A induced by B within time* $\tau$, $AVO^{\delta,\tau}_{AB}$, where $\delta$ is a control parameter whose dimension is time. The acceleration $a_A(t)$ at time $t$ is proportional to the difference between the new velocity $v^{new}_A$ and the velocity $v_A(t)$ at time $t$, i.e.

$$a_A(t) = \frac{v^{new}_A - v_A(t)}{\delta}.$$

The solution of the previous differential equation is given by

$$v_A = v^{new}_A - e^{-t/\delta}\left(v^{new}_A - v_{A,0}\right),$$

where $v_{A,0}$ is the velocity of $A$ at $t = 0$. Integrating this solution gives the position of the agent at time $t$

$$x_A = x_{A,0} + v^{new}_A t + \delta\left(e^{-t/\delta} - 1\right)\left(v^{new}_A - v_{A,0}\right) \tag{21}$$

where $x_{A,0}$ is the position of $A$ at time $t = 0$. Let $x^{\tau,\delta}_e \triangleq \left(e^{-t/\delta} - 1\right)$, for simplicity, and let $x_{AB}$, $v_{AB}$ and $a_{AB}$ be the relative position, velocity and acceleration of $A$ with respect to $B$. A collision occurs at time $t$ if $\|x_{AB}\|_2 < r_{AB} = r_A + r_B$. Exploiting (21), such inequality can be re-written as

$$\|x_A + v^{new}_A t + \delta x^{\tau,\delta}_e \left(v^{new}_A - v_A\right)\|_2 < r_{AB}.$$

It is possible to re-arrange this equation into the following one

$$\|v^{new}_{AB} - \frac{\delta x^{\tau,\delta}_e v_{AB} - x_{AB}}{t + \delta x^{\tau,\delta}_e}\|_2 < \frac{r_{AB}}{t + \delta x^{\tau,\delta}_e},$$

which defines the disc of all relative velocities $v^{new}_{AB}$ that cause a collision between $A$ and $B$ at time $t$. The $AVO^{\delta,\tau}_{AB}$ can be written as

$$AVO^{\delta,\tau}_{AB} \triangleq \bigcup_{0<t\leq\tau} \mathcal{D}\left(\frac{\delta x^{\tau,\delta}_e v_{AB} - x_{AB}}{t + \delta x^{\tau,\delta}_e}, \frac{r_{AB}}{t + \delta x^{\tau,\delta}_e}\right),$$

where $\mathcal{D}(\cdot,\cdot)$ is a disc whose center and radius are the first and second argument, respectively. In order to avoid collisions with $B$, robot $A$ must select a new velocity vector outside of $AVO^{\delta,\tau}_{AB} \oplus v_A$, where $\oplus$ denotes the Minkowski sum. The approach is designed for holonomic disc-shaped autonomous robots, but it can be extended also to robots that have to satisfy kinematic constraints. However since robot $A$ is in any case subjected to acceleration constraint, the set of collision-free velocities for $A$, $CA_A$, is constructed as follows

$$CA_A \triangleq \mathcal{D}(v_A, \delta a^{max}_A) \setminus \bigcup_B AVO^{\delta,\tau}_{AB} \oplus v_A.$$

Given a preferred velocity vector $v^{pref}_A$ for robot $A$, the algorithm will select a new velocity such that

$$v^{new}_A = \underset{v \in CA_A}{\arg\min} \|v - v^{pref}_A\|_2.$$

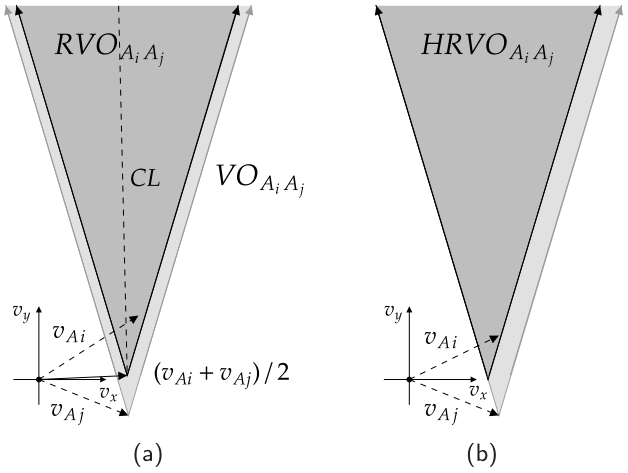



**Fig. 7.** $HRVO_{A_iA_j}$ construction. The agent $A_i$ (a) computes $RVO_{A_iB_j}$ induced by $A_j$ and verifies that the tip of $v_A$ is on the right of the center-line $CL$, then (b) calculates $HRVO_{AB}$ by intersecting the right edge of $RVO_{A_iB_j}$ with the left edge of $VO_{A_iB_j}$.

### 5.6. Hybrid reciprocal velocity obstacle

Snape et al. [26] formalized the so called Hybrid Reciprocal Velocity Obstacle (HRVO) with the aim to solve the reciprocal dances problem of RVO paradigm seen in Section 5.1, by forcing a collision avoidance maneuver policy among all the agents enlarging on purpose a side of every absolute collision cone.

Consider a certain number $m \in \mathbb{N}$ of autonomous agents $A_1, \dots, A_m$ moving in the same workspace. The first three steps to construct the $HRVO$ of $A_i$ induced by $A_j$ are: (i) compute $VO_{A_iA_j}$ as in Section 3, (ii) compute the corresponding $RVO_{A_iA_j}$ as in Section 5.1 and (iii) draw the center line CL splitting $RVO_{A_iA_j}$ into two identical halves. Suppose now that the tip of the current velocity vector of agent $A_i$, $v_{A_i}$, is on the right of CL, Fig. 7(a). It means that $A_i$ will perform a collision-free maneuver to the right side of the other agent $A_j$. $HRVO_{A_iA_j}$ is defined as the cone which apex is the intersection point between the left edge of $VO_{A_iA_j}$ and the prolongated right edge of $RVO_{A_iA_j}$.

The exact same construction can be done for symmetry, when the tip of $v_A$ is assumed to be on the left of $CL$. Fig. 7 summarizes the $HRVO$ cone construction from agent $A_i$ perspective.

Finally, supposing that in the environment there are also $m \in \mathbb{N}$ static obstacles $O_1, \dots, O_m$; at every cycle of the algorithm, $A_i$ selects a new velocity vector such that

$$v_{A_i}^{new} = \underset{v_{A_i} \notin HRVO_{A_i}}{\arg\min} \|v_{A_i} - v_{A_i}^{pref}\|_2,$$

where $HRVO_{A_i}$ is given by

$$HRVO_{A_i} \triangleq \bigcup_{j \neq i}^{n} HRVO_{A_iA_j} \cup \bigcup_{k=1}^{m} HRVO_{A_iO_k}.$$

Computing $HRVO_{A_i}$ instead of $RVO_{A_i}$ ensures that there are no reciprocal dances between agents.

The major drawback is the increased computational burden: if the RVO paradigm implies that every agent $A_i$ has to compute one collision cone for every $A_j$, $i \neq j$, then HRVO implies to compute an additional cone for every other agent.

### 5.7. Non-holonomic optimal reciprocal collision avoidance

In [27], the authors extended the ORCA paradigm to non-holonomic cart-like robots. A cart-like vehicle has three degrees of freedom i.e. two

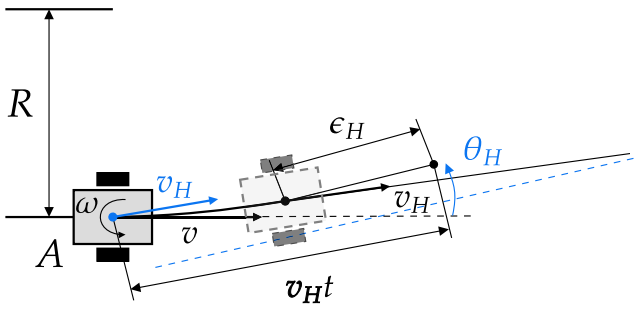



**Fig. 8.** Geometric interpretation of the tracking error $\epsilon_H$. $\boldsymbol{v_H}$ and $\theta_H$ (blue) defines the holonomic trajectory to be tracked by the robot $A$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

variables $(x, y)$ for the position and one $\theta$ for the orientation; its kinematics is

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \omega, \tag{22}$$

where $v$ is the linear control velocity and $\omega$ is the angular control velocity. The idea behind the approach is that a robot $A$ with kinematics described by (22) is able to track the motion of an "ideal" holonomic robot, namely a holonomic trajectory, within a certain bounded tracking error $\epsilon_H$. Referring to Fig. 8, if the current velocity of robot $A$ is fixed to $\boldsymbol{v_H} = (v_H \cos\theta_H, v_H \sin\theta_H)$, where $v_H$ is the speed of such ideal holonomic robot, then the maximum tracking error for the aforementioned holonomic trajectory is

$$\epsilon_H^2 = (v_H t - R\sin\theta_H)^2 + (R(1-\cos\theta_H))^2,$$

$$= v_H^2 t^2 - \frac{2v_H t \sin\theta_H}{\omega} v + \frac{2(1-\cos\theta_H)}{\omega^2} v^2,$$

where $R$ is the curvature radius of the trajectory realized by $A$ and $t = k\Delta t$, with $k$ the iteration index and $\Delta t$ the time-step.

The set $D(0, v_A^{max})$ for agent $A$ defined in Section 5.2 is substituted by another set denoted by $S_{FHV}$, that stands for the set of feasible holonomic velocities under the kinematic constraint. Let $S_{NHC}$ be the set of non-holonomic controls, the feasible holonomic velocities are defined as

$$S_{FHV} = \{\boldsymbol{v_H} : \exists (v(s), \omega(s)) \in S_{NHC}, \\ \|x + s \cdot \boldsymbol{v_H} - \hat{x}^k(s)\|_2 \leq \epsilon, \forall \tau \geq 0\},$$

where $x$ is the robot current position, $\hat{x}^k(s)$ is the expected robot position at time $k\Delta t + s$ if the controls $(v(s), \omega(s))$ are applied at time $k\Delta t$. In other words, this is the set of all allowed holonomic velocities $\boldsymbol{v_H}$ for which there exist control inputs that guarantee a tracking error below a certain fixed threshold $\epsilon_H$ as shown in Fig. 8.

The optimal linear velocity input $v^*$ that allows the cart to track $\boldsymbol{v_H} = (v_H \cos\theta_H, v_H \sin\theta_H)$ within a certain maximum tracking error $\epsilon_H$ is given by

$$v^* = \frac{v_H t \sin\theta_H \omega}{2(1-\cos\theta_H)} = \frac{\theta_H \sin(\theta_H)}{2(1-\cos\theta_H)}.$$

If the optimal linear velocity is not feasible, the optimal controls are

$$\omega = \theta_H / T \leq \omega_{max} \quad and \quad v = v^* \leq v_{max},$$
$$\omega = \theta_H / T \leq \omega_{max} \quad and \quad v = v_{max},$$
$$\omega = \omega_{max} \quad and \quad v = 0,$$

where $T$ is a fixed amount of time given to the robot to achieve the correct orientation, $v_{max}$ and $\omega_{max}$ are the maximum linear and angular velocities, respectively.

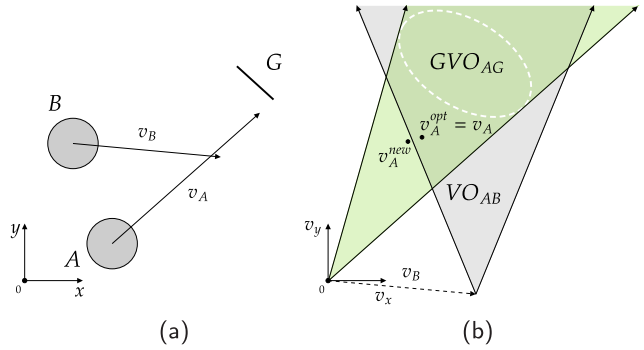



**Fig. 9.** Geometric representation of the Goal Velocity Obstacle. In (a) agents $A$, $B$ and goal $G$ (black segment). In (b) the Goal Velocity Obstacle $GVO_{AG}$ of $A$ induced by $G$ (green cone). The white dashed region is $G$ represented in the velocity space. $v_A^{new}$ is selected in order to be simultaneously outside $VO_{AB}$ and inside $GVO_{AG}$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

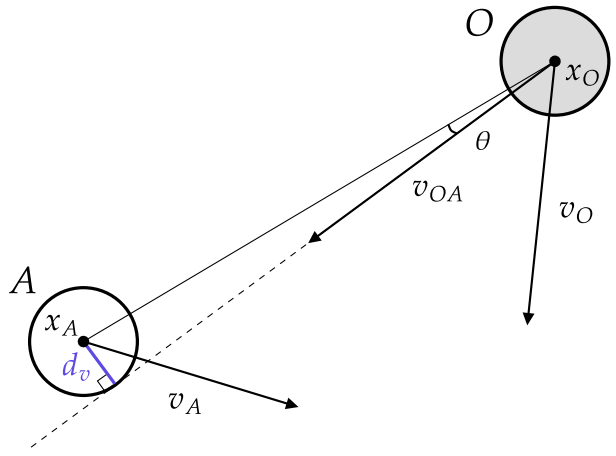



**Fig. 10.** Geometrical interpretation of the clearance $d_v$ (blue segment). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

### 5.8. Goal velocity obstacle

This formulation by Snape and Manocha [56] extended the original VO paradigm to multi-agent frameworks in which the agents have to reach a goal region $G$ rather than a target point. The reason behind this idea lies in the fact that in some applications all the agents or subgroups may have to share the same goal. The problem is solved by adopting the VO/RVO concepts to identify all the collision-free velocities that lead to a prescribed goal region by introducing the *Goal Velocity Obstacle* (GVO). Consider an agent $A$, moving with current velocity $v_A$, that has to enter in the goal region $G$, then the Goal Velocity Obstacle for $A$ induced by $G$ is

$$GVO_{AG} \triangleq \{v \mid \exists\, s \in [0,t],\ s(v - v_A) \in G \oplus -A\},$$

as shown in Fig. 9.

$GVO_{AG}$ is the cone of all velocities that allow the agent $A$ to enter in its goal region $G \subset \mathbb{R}^2$. At every cycle of sensing-and-acting, the agent $A$ must select a new velocity such that

$$v_A^{new} = \underset{v \in GVO_{A|G} \setminus VO_A^{\tau}}{\arg\min} \|v - v_A^{opt}\|_2,$$

where $v_A^{opt}$ is the *optimal velocity*, that is defined as follows. If the current velocity $v_A$ is pointing to $G$ then the agent must choose $v_A^{opt} = v_A$, otherwise it means that $v_A \notin GVO_{AG}$ and thus the agent must choose

$$v_A^{opt} = \underset{v \in GVO_{A|G}}{\arg\min} \|v - v_A\|_2.$$

When the number of autonomous agents in the dynamic environment is very large, it may occur that $GVO_{AG} \subset VO_A^{\tau}$, that means there are no collision free velocities leading to the goal region. This problem can be solved by relaxing the constraints on the velocity space: by prioritizing the pure collision avoidance velocities forgetting the goal region for a while or not computing the collision cones with respect to the most distant agents. The approach takes into account moving regions, multiple regions and with/without time windows.

### 5.9. Optimal velocity selection for velocity obstacle

Kim and Oh [58] proposed an optimal velocity selection method for VO, named *Optimal Velocity selection for Velocity Obstacle* (OVVO). The motivation is that in very crowded scenarios, the original VO method is not able to select a safe velocity for each robot due to the lack of feasible candidates among which select the new safe velocity. The authors proposed an optimization method based on minimizing a cost function for the desired velocity, allowing the robot to prioritize avoidance maneuvers over stopping policy and viceversa.

Consider an autonomous robot $A$ and a dynamic obstacle $O$, moving with current velocities $v_A$ and $v_O$, respectively. The cost function to be minimized is the sum of two components. The first component is

$$f_1 = k_{vd} \|v_A^{pref} - v_A\|_2,$$

where $k_{vd}$ is a positive constant and $v_A^{pref}$ is the robot preferred velocity. The second component is a function of the *clearance* (see Fig. 10) and the *pass-time*. The clearance is the distance between a relative velocity and a relative distance

$$d_v = \frac{v_{OA} \times (x_A - x_O)}{\|v_{OA}\|_2},$$

where $v_{OA} = v_O - v_A$ is the relative velocity.

A small $d_v$ means that the obstacle is dangerous to the robot, thus $v_{OA}$ is not safe. The function depending on the clearance is defined as follows

$$f_{dv} = k_1 d_v^{-c_1}$$

where $k_1$ and $c_1$ are positive constants.

The pass-time is the time needed by the robot to pass an obstacle,

$$t_p = \frac{(x_O - x_A)}{(v_O - v_A)}.$$

If $t_p$ is negative, then the robot has already passed the obstacle, if $t_p$ is large, then the obstacle is not dangerous for the robot. The function depending on the pass-time is given by

$$f_{tp} = k_2 t_p^{-c_2},$$

where $k_2$ and $c_2$ are positive constants.

The second component $f_2$ of the cost function is the product between $f_{dv}$ and $f_{tp}$

$$f_2 = k_{tp} d_c^{-c_1} t_p^{-c_2}.$$

The overall cost is given by

$$J(v) \triangleq f_1 + f_2 = k_{vd} \|v_A^{pref} - v_A\|_2 + k_{tp} d_c^{-c_1} t_p^{-c_2}.$$

The ratio between the constants $k_{vd}$ and $k_{tp}$ is of primary importance since higher $k_{tp}$ means that the robot will give more importance to collision-free maneuvers while higher $k_{vd}$ makes the robot keeping its actual velocity $v_A$ close to the desired one $v_A^{pref}$, even in presence of nearby obstacles. In the experimental results, authors focused on comparing the OVVO results with those coming from FVO, Guy et al. [52].

**Table 2**
Summary table.

| Paradigm | Environment | Agents model | Parameters | Description |
|---|---|---|---|---|
| VO | Agent-Obs | Holonomic | Deterministic | The agent computes an absolute collision cone for every moving obstacle and selects a new velocity out of it, as close as possible to the preferred one. |
| RVO | Multi-agent | Holonomic | Deterministic | Extension of VO paradigm to multi-agent systems: all the moving objects are decision-making entities able to compute Velocity Obstacles. |
| ORCA | Multi-agent | Holonomic | Deterministic | Extension of RVO paradigm: to guarantee optimality, smooth and safe trajectories, the new velocity is selected by solving a constrained linear optimization problem. |
| GVO | Agent-Obs | Car-like | Deterministic | Generalization of the VO paradigm to car-like robots in which the absolute collision cones are represented on the space of controls that move the agent. |
| NLVO | Agent-Obs | Non-holonomic | Deterministic | Extension of the VO paradigm: the agent computes a warped cone to avoid collisions with obstacles following a known non-linear trajectory. |
| AVO | Multi-agent | Holonomic car-like | Deterministic | Extension of ORCA paradigm: every agent selects its new velocity satisfying a maximum acceleration constraint. |
| HRVO | Multi-agent | Holonomic Cart-like | Probabilistic, Kalman filtered | Extension of RVO paradigm, designed to solve the *reciprocal dances* problem. Position and velocity of agents are modeled as Gaussian random variables in order to model sensor noise. |
| NH-ORCA | Multi-agent | Cart-like | Deterministic | Extension of ORCA paradigm to cart-like robots. Differential driven agents track holonomic velocities within a certain bounded error. |
| Goal VO | Multi-agent | Holonomic | Deterministic | Extension of VO paradigm: every agent computes an additional cone, containing all velocities that allows it to reach a possibly moving goal region. |
| OVVO | Agent-Obs | Holonomic | Deterministic | VO paradigm integrated with optimal collision-free velocity selection, prioritizing clearance over pass-time and viceversa. |

Table 2 briefly summarizes the Velocity Obstacle extensions analyzed so far. *Paradigm* denotes the name of the algorithm. *Environment* explains what type of scenarios the algorithm is able to handle. *Agent model* stands for the type of agents the algorithms are designed for. *Parameters* refer to agent's position, velocity, radius and orientation: deterministic means that all these measurements do not contain uncertainty and are perfectly known by each agent, probabilistic means that some parameters are modeled as random variables. *Description* provides a quick overview of the method.

## 6. Applications of VO to real and simulated scenarios

The Velocity Obstacle Paradigm together with its latter formulations have been successfully employed to solve collision avoidance problems in real or simulated scenarios. In this section, we describe how the approaches introduced in Section 5 can be implemented to model realistic scenarios. Note that the majority of contributions concern collision avoidance for unmanned ground vehicles (UGVs), very few for unmanned aerial vehicles (UAVs).

### 6.1. MAid robotic wheelchair

In [59,60], a robotic wheelchair named MAid, *Mobility Aid for Elderly and Disabled People*, was developed to assist people in moving in densely crowded dynamic environments.

Authors focused their attention on implementing two navigation modes, *Narrow Area Navigation* (NAN) and *Wide Area Navigation* (WAN). NAN is a semiautonomous navigation mode in which the user commands MAid to execute local maneuvers in a narrow, cluttered spaces. Moving in narrow spaces requires precision and frequent adjustments of the maneuvers, thus the semi-autonomous mode allows the users to improve their control skills while using MAid.

WAN is a completely autonomous mode that addresses MAid's navigation in wide, rapidly changing, crowded areas such as airport concourses, shopping malls, or convention centers. The user has to provide a goal position, while planning and trajectory execution to the goal are completely taken care by MAid navigation control via Velocity Obstacle. MAid wheelchair has been tested in the central station of Ulm during rush-hour and in the exhibition halls of the Hannover Messe '98, one of the world's biggest industrial fairs. In these dynamic and rapidly changing environment, it collected a total amount of 36 h of safe navigation among people.

### 6.2. Collision avoidance for iRobot create agents

Snape et al. [61] tested the performance of HRVO by implementing the paradigm on four differential driven *iRobot Create* programmable robots, iRobot manufacturer [62], in two laboratory scenarios

1. *Four corners*: each robot starts from a corner with the goal to reach the opposite corner of the workspace, and
2. *Moving obstacle*: one of the four robots plays the role of a moving obstacle and the other three have to avoid collisions.

Since iRobot Create has very limited sensing capabilities, that do not allow to properly localize themselves via odometry and thus to localize every robot with sufficient accuracy, the authors use a ceiling-mounted Point Grey Grasshopper digital video camera with a FireWire 800 connection to obtain images at a resolution of $1024 \times 768$ px at a refresh rate of 15 Hz. Every robot is equipped with a marker on top and uses the ARToolKit augmented reality system by Kato and Billinghurst [63] to determine position and orientation, with an absolute error smaller than 10 mm. Data from sensors may be affected by disturbances, thus the velocity of the robots is inferred from the position and orientation measurements using an extended Kalman filter, Welch and Bishop [64]. The computations that HRVO requires are performed on a single computer, but to ensure that the approach is also applicable when each robot uses its own on-board sensors and computing units, i.e. to maintain a decentralized approach to the collision avoidance problem, only the localization processing is centrally computed. The computations for each robot related to the navigation are performed in separate processes, without any exchange of information, then each one sends the computed wheel speeds to its associated robot over a Bluetooth virtual serial connection with a latency of 50 ms.

In [65], the authors repeated the experiments made in laboratory using the iRobot Create robots replacing the HRVO paradigm with the ORCA paradigm adapted to deal with differential-drive robots. They proved that the collision free trajectories provided by the algorithm are continuous.

### 6.3. Simulation of an automated warehouse

Piccinelli et al. [66] adapted the VO paradigm to model real-time collision avoidance for autonomous agents moving in an automated warehouse. Safe trajectories with respect to known fixed obstacles

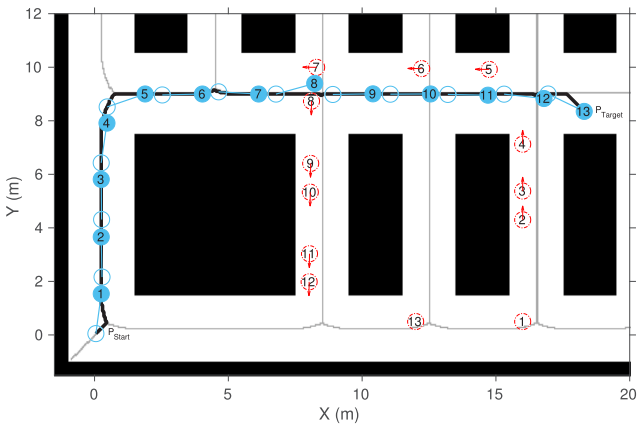



**Fig. 11.** Time-lapse of a simulated scenario in which two cooperating agents (filled and empty blue circles) has to reach a goal position while avoiding collision with a third agent (empty red circle) moving along Voronoi trajectories. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)
*Courtesy:* Piccinelli et al. [66].

and moving unexpected obstacles are obtained by combining ORCA paradigm with a global planner based on Voronoi Diagrams, Aurenhammer [67] and Foskey et al. [68], extended to polygons, see Fig. 11.

The authors assumed two robots under a rigid body constraint forcing them to remain at a certain distance $d$, in order to cooperate in carrying long and heavy payloads. The cooperation of the agents is obtained by selecting the new velocities in such a way that

$$v_A^{new} \in \overline{ORCA}_A^{\tau} \cap RB$$
$$v_B^{new} \in \overline{ORCA}_B^{\tau} \cap RB$$

where $RB$ is the set of all velocities that do not allow the agents to brake the rigid body constraint

$$RB = \{(v_A^{new}, v_B^{new}) \ : \ \|x_A^{new} - x_B^{new}\|_2 = d\},$$

and $\overline{ORCA}_{A,B}^{\tau}$ half-spaces are equivalent to those defined in Eq. (18), but constructed in such a way that $A$ ignores the presence of $B$ and viceversa, i.e. $A$ does not compute $VO_{AB}^{\tau}$ and $B$ does not compute $VO_{BA}^{\tau}$.

### 6.4. Collision avoidance for unmanned aerial vehicles

Battisti and Muradore [55] simulated the autonomous landing of a Unmanned Aerial Vehicle (UAV) onto a Unmanned Ground Vehicle (UGV) avoiding collisions with flying moving obstacles (Fig. 12). They also proposed to use the ORCA paradigm to compute a virtual feedback force for haptic device in a bilateral teleoperation architecture for remotely controlling the UAV. The UAV has to satisfy constraints on maximum acceleration $a^{max}$ and steering angle, given by

$$\theta = \theta^{min} + (\theta^{max} - \theta^{min}) \left(1 - \frac{\|v\|_2}{v^{max}}\right)$$

where $\theta^{min}$ is the maximum possible steering angle at the minimum velocity and $\theta^{max}$ is the maximum possible steering angle at the maximum velocity $v^{max}$. Three conditions must be satisfied before starting the landing procedure of the UAV onto the UGV: there must be no imminent collisions detected, UAV must be at least at a certain hovering height $h_s$ over the UGV, and the relative velocity of the UAV with respect to the UGV must be below a threshold.

The virtual force feedback perceived by the operator, that remotely controls the UAV, must be directly proportional to the difference between the (collision free) velocity provided by the ORCA algorithm and the current velocity imposed by the user that teleoperates the UAV via a haptic device, i.e.

$$\mathbf{f} \propto v^{ORCA} - v^{TELEOP}.$$

If the operator deliberately ignores or misunderstands the feedback force the *Safe Flight Mode* activates, correcting the magnitude of $v^{TELEOP}$ according to the distance between the UAV and the detected obstacle, in order to guarantee safety.

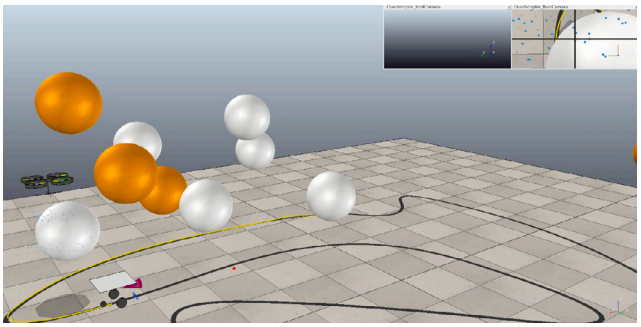


**Fig. 12.** A snapshot of an autonomous landing operation simulated in V-REP.
*Courtesy:* Battisti and Muradore [55].

## 7. VO-based methods drawbacks

The Velocity Obstacle paradigm and its improvements have some drawbacks: (1) computational burden in scenarios with a large number of autonomous agents/obstacles, (2) the assumption of perfect knowledge of the surroundings (i.e. other agents' position, radius and velocity) that is not realistic, and (3) deadlocks.

In real-world applications, autonomous robots do not perfectly perceive other agents as well as the immediately surrounding environments, i.e. sensors have a certain range of action and data quality is affected by noise and disturbances. This kind of issue makes the assumption of perfect knowledge of the surrounding world unrealistic. Section 7.2 shows how to overcome this issue. Deadlocks are unpleasant situations in which one or more agents are not able to select non-trivial safe velocities preventing them from reaching their goal position, occurring when they assume particular configurations. Section 7.3 shows a possible solutions to such problem. The section concludes with a comparative study of the principal extensions of the Velocity Obstacle.

### 7.1. Computational burden reduction

He and Van Den Berg [69] introduced an intermediate level between local and global path planning, called *meso planning*, in order to cluster groups of moving obstacles agents that navigate close to each other and with similar speeds. Consider an agent $A$ that has to avoid collisions with $m \in \mathbb{N}$ other autonomous agents or moving obstacles $A_1, \dots, A_m$. The meso planning algorithm groups the agents with similar positions and velocity vectors via the following equivalence relation

$$A_i \sim A_j \triangleq \left\{ \|x_{A_i} - x_{A_j}\|_2 < \epsilon_A^p \wedge \|v_{A_i} - v_{A_j}\|_2 < \epsilon_A^v \right\},$$

where $i \neq j$ and $\epsilon_A^p$ and $\epsilon_A^v$ are thresholds, that can be different from agent to agent. $A_1, \dots, A_m$ are clustered into $G_1, \dots, G_n$ with $n \leq m$ and $A$ considers them as single entities calculating $VO_{AG_i}$ for $i = 1, \dots, n$ in order to avoid collisions with entire groups of agents. The procedure avoids the calculation of unnecessary collision cones, reducing the computational burden.

Van Den Berg et al. [70] analyzed the collision avoidance of crowds of autonomous agents adopting RVO method in realistic scenarios. Every agent computes the collision cones only with respect to the $k \in \mathbb{N}$ nearest agents, in order to reduce the computational burden.

In [71] authors solve the problem of long-range collision avoidance: for each agent $i$ in the dynamic environment, the objective is to find an updated velocity $v_i$ as close as possible to the preferred one $v_i^{pref}$ that avoids congestion in front of the agent at a range of distances that spans from "far" to "near", while also avoiding collisions with neighboring agents. Authors opted for decomposing the long-range collision avoidance problem into a sequence of Local Collision Avoidance (LCA) queries, meaning that the algorithm can be executed by adopting already existing LCA algorithms. The main idea of the lookahead approach is to address the LCA problems not only in the current observable state of the crowd, but also in the future state, over possibly large future time intervals. Since the future motions of other agents are unknown, the possible future locations are modeled as probability distribution centered in the current agent position $x_i(t)$, with a certain spread $\sigma(t)$ that becomes larger as far as one looks into the future.

### 7.2. Noisy sensing

Kluge and Prassler [72] investigated the problem of how uncertainty affects the perception of the dynamic environment around an autonomous robot. They proposed the Recursive Probabilistic Velocity Obstacle (RPVO), an extension of the original VO paradigm in which the velocity of moving obstacles $B_i$ is a probability density function $V_i : \mathbb{R}^2 \to \mathbb{R}_{\geq 0}$ to model on-board sensor noise. An agent $A$ avoids collisions with an obstacle $B_i$ by selecting a new velocity $v_A^{new}$ outside the *Probabilistic Velocity Obstacle* $PVO_{AB_i}(v_A)$, where $PVO_{AB_i} : \mathbb{R}^2 \to [0, 1]$ is a probability measure, giving the probability of collision for the agent moving with current velocity $v_A$.

In [26] sensor data are simulated by considering position, velocity and radius of agents to be modeled by bivariate Gaussian and univariate Gaussian random variables, respectively. Let $X_A \sim \mathcal{N}(\hat{x}_A, \Sigma_A)$ be the Gaussian distribution for the current position of agent $A$, with mean $\hat{x}_A$ and covariance matrix $\Sigma_A$ and $X_B \sim \mathcal{N}(\hat{x}_B, \Sigma_B)$ be the Gaussian distribution for the current position of agent $B$ with mean $\hat{x}_B$ and covariance matrix $\Sigma_B$. The collision cones for agent $A$ are build on the relative position given by the following distribution

$$X_{B-A} \sim \mathcal{N}(\hat{x}_B - \hat{x}_A, \Sigma_{BA}),$$

which is a bivariate Gaussian distribution with mean $\hat{x}_B - \hat{x}_A$ and covariance matrix $\Sigma_{BA}$. The same holds true for the radii, i.e. $R_{A+B} \sim \mathcal{N}(\hat{r}_A + \hat{r}_B, \sigma_{r_{AB}})$ is the Gaussian distribution of the measured $r_A + r_B$, and for the velocity of $B$, i.e. $v_B \sim \mathcal{N}(\hat{v}_B, \Sigma_{v_B})$. Accurate estimates of the three parameters are then obtained via Kalman filtering and used to construct the absolute collision cones for agents.

Gopalakrishnan et al. [73] recently proposed a similar work, incorporating uncertainty about position and velocity of holonomic robots in the RVO method. Furthermore authors assumed that robot actuation is not perfect, meaning that if robot $A$ needs to assume the collision-free velocity $v_A^{new}$ in order to avoid collisions with other robots or obstacle, the executed velocity $\tilde{v}_A^{new}$ is obtained by any sample of a Gaussian random variable $\tilde{v}_A^{new} \sim \mathcal{N}(v_A^{new}, \Sigma_A)$.

Inspired by the ORCA Paradigm (Section 5.2), Fan et al. [74] proposed a method for multi-robot collision avoidance via Deep Reinforcement Learning (DRL). Agents do not access directly position, radius and velocity of other agents but perform collision avoidance maneuvers based on a learned policy obtained via a deep neural network (DNN). Similarly to Velocity Obstacles, all the $n \in \mathbb{N}$ agents are assumed to be homogeneous disc-shaped decision making entities, that do not perfectly perceive the dynamic environment. The policy for collision avoidance is trained via neural network over observations taken on a large number of robots in rich, complex environments using a policy-gradient-based reinforcement-learning algorithm.

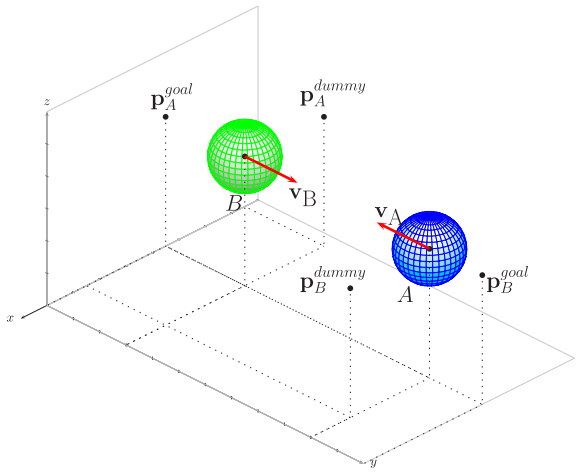



**Fig. 13.** Dummy goal $p_A^{dummy}$ for agent $A$ (blue) and $p_B^{dummy}$ for agent $B$ (green) are placed to avoid a deadlock. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)
*Courtesy:* Battisti and Muradore [55].

### 7.3. Deadlocks

ORCA paradigm is very popular among researchers who deal with decentralized collision avoidance of autonomous agents, thanks to the very clear theoretical formulation, high adaptability, and availability of libraries written in various programming languages [75].

Unfortunately when certain conditions occur, two or more agents can get stuck in a deadlock situation that prevents them from reaching their goal positions.

Battisti and Muradore [55] analyzed the situation occurring when two agents $A$ and $B$ moves on the same straight line one towards each other with the objective to reach a goal position that lies behind the other agent as shown in Fig. 13. In the original ORCA, $A$ will select a new collision avoidance velocity $v_A^{new}$ whose magnitude decreases to zero within every cycle of sensing-and-acting, and symmetrically $B$ does the same thing, leading the two agents to stop one in front the other and eventually also getting back. This happens because the ORCA half-planes $ORCA_A^{\tau}$ and $ORCA_B^{\tau}$ are parallel at every cycle of the algorithm. The proposed solution is based on placing a temporary *dummy goal* for agent $A$, namely $p_A^{dummy}$, on the right of $B$ at a certain safety distance. The same policy is applied to $B$ at the same time, allowing to avoid the deadlock, as shown in Fig. 14. The agent $A$ initially travels towards $p_A^{dummy}$ until it is reached and afterwards it discards the dummy goal, going back to consider its original goal position in order to conclude the task.

### 7.4. Comparative study of VO approaches

Douthwaite et al. [76] produced a comparative analysis between the most popular extensions or enhanced Velocity Obstacle approaches. In particular, the authors concentrated on the original VO formulation, the RVO, HRVO and finally the ORCA algorithm. The objective is to discuss their performance in crowded situations and with different assumptions about the knowledge of the environment. The algorithms have been tested over a range of scenarios with several levels of difficulty and obstacle numbers, assuming both perfect knowledge of every other agent state and by simulating sensors. Advantages and disadvantages of every approaches have been demonstrated and discussed via Monte Carlo simulations. The agents are assumed to move on a 2D Cartesian plane. The position vector of agent $i$ at time $t_{k+1}$ is defined by $x_{k+1}^i = x_k^i + \Delta t\, v_k^i$ meaning that agents are assumed to be holonomic. A constraint of maximum achievable speed is assumed $|v^i| \leq v^{max}$.

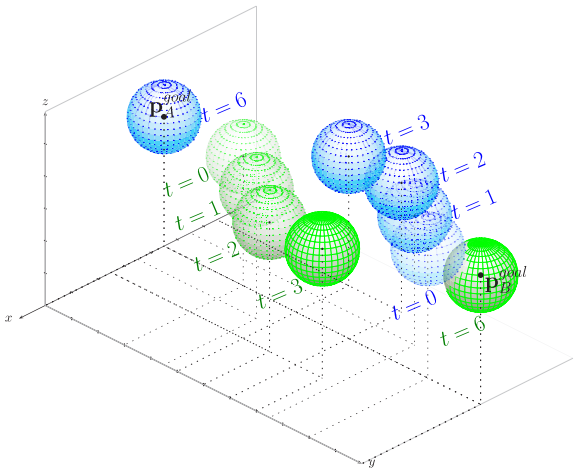



**Fig. 14.** Time lapse of collision avoidance maneuver between the two agents $A$ (blue) and $B$ (green) once the situation of deadlock is solved. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)
*Courtesy:* Battisti and Muradore [55].

The simulated agents are equipped with virtual onboard cameras and range finder to simulate individual observations of the environment. Robot's sensors defined the spherical position of every other agent, in the robot frame of reference, via several parameters such as elevation $\theta \in [-\pi, \pi]$, azimuth angle $\lambda \in [-\pi, \pi]$, range $d \in [0, d_{max}]$ and width $\alpha \in [-\pi, \pi]$, where $d_{max}$ describes the maximum visual range of an agent. The velocity of every other agent $i$ is captured via successive position observations, i.e. $v_k^i = (x_k^i - x_{k-1}^i)/\Delta t$, where $\Delta t$ is the sample-time.

Agents' paths are based on way-points in every simulation. Let $x_{wp}^h$ be the position of the $h$th waypoint for the agent, the preferred velocity is given by

$$v_k^{i,pref} = \frac{x_{wp}^h - x_k^i}{\|x_{wp}^h - x_k^i\|_2} \|v^{pref}\|_2,$$

where $x_k^i$ is the agent $i$ position and $\|v^{pref}\|_2$ is the preferred speed. The simulation scenario consists in an increasing number of agents arranged in a circle, each one aiming to reach a specific waypoint placed in antipodal position with respect to the initial position without colliding with the other agents. Collisions are detected whenever the following condition is violated

$$\|x_k^i - x_k^j\|_2 < (r_i + r_j) - K_{tol}, \quad i \neq j,$$

where $K_{tol}$ is a fixed tolerance to distinguish between collisions and near-misses, $r_i$ and $r_j$ are the agents' radii. Higher level strategies are adopted in order to overcome known deadlock situations. The Monte Carlo analysis has been performed over 1000 simulations for every scenario, where the initial position of every agent is randomly perturbed of 0.5 m. The collision avoidance schemes are tested in condition of perfected sensing of the surroundings and by simulating onboard sensors, with uncertainty derived from the presence of measurement noise. Authors compared the mean number of collisions, the mean minimum separation distance between agents and the mean computation time among all the approaches.

The results showed that the worst performing collision avoidance schemes are VO and RVO, showing a high-number of collisions, non-smooth trajectories and reciprocal dancing of the agents together with bad performance scalability on the number of agents. HRVO is the worst from a computational time perspective, but it solves the problem of reciprocal dances, achieves good values for minimum separation and handles sensors uncertainty. ORCA provides the smoothest trajectories and the best total computation time, but is not as good as HRVO in dealing with uncertainty.

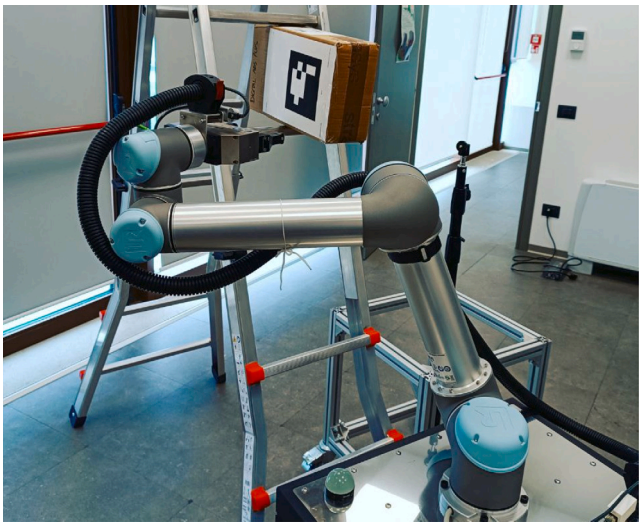


**Fig. 15.** The UR5 manipulator by Universal Robotics used to test the Velocity Obstacle approach in [80].

### 7.5. General suggestions

Beyond assessing the performance of individual paradigms detailed in Section 5, their distinctive characteristics enable us to discern which methods are more or less suitable based on the specific situation and environment. In simulated scenarios, every paradigm that assumes holonomic agents can be suitable, especially ORCA that is specifically designed to deal with large numbers of autonomous agents (~100), Goal VO that allows to handle multiple moving goal regions and OVVO when we are in presence of particularly fast or slow agents in dynamic environment, see e.g., [77]. However, in real-world scenarios involving ground vehicles, the assumption of holonomic autonomous robots is often impractical or not realistic. Therefore, recommended algorithms include GVO, AVO and ORCA for car-like or bike-like robots (e.g., autonomous drive cars, Berg 700 UGV), or alternatively, HRVO and NH-ORCA for cart-like robots (e.g., Pioneer P3-DX, Clearpath Jackal), see e.g., [78]. Autonomous aerial robots, primarily drones (e.g., quad-rotors) with three-dimensional movement capabilities, can reasonably be approximated as holonomic agents. In this case, the recommended approach is ORCA, as it overcomes many drawbacks of VO and RVO methodologies, and essentially ad-hoc adaptations of the original paradigm as in e.g., [79].

## 8. Conclusions and future developments

We presented a survey on the VO paradigm originated by Fiorini and Shiller [21]. We described the original VO formulation, preceded by a brief digression about other popular decentralized collision-avoidance approaches. We then constructed the taxonomy comprehending the VO extensions and improvements along 20+ years of evolution. The aim of the article is to provide a complete overview of the VO methods for addressing the decentralized collision avoidance problem of multi-agent systems.

The VO has been studied, developed, and generalized only for the collision avoidance of both ground and aerial mobile robots. The large number of enhanced versions and the versatility of the method have driven us to explore other fields of application, even radically different from mobile robotics, and to integrate the paradigm with other trajectory planning algorithms. We managed to adapt the (Finite-Time) VO for generating collision-free trajectories in the case of a two degree-of-freedom planar serial manipulator [81] and to a three degree-of-freedom anthropomorphic manipulator [80] that have to avoid dynamic obstacles in their workspace. In particular, in the latter

work we have tested the collision avoidance algorithm by implementing it directly on a UR5 6 degrees-of-freedom robotic arm by Universal Robots (see Fig. 15). Furthermore, in [82], authors have introduced VO-FABRIK, an algorithm that combines the VO paradigm to the Forward and Backward Reaching Inverse Kinematics algorithm, for the planning of safe trajectories for hyper-redundant manipulators. In [83], we merged the MPC and VO techniques into a novel approach for the collision avoidance among mobile robots: VO paradigm is exploited to define constraints on the configuration space of the controlled system. Other authors Bonanni et al. [84] combined Monte Carlo Tree Search with VO to discard colliding velocities from the action space of mobile robots in dynamic environments.

In all its applications, the VO considers obstacles as mobile entities of circular or spherical shape. This approximation, while convenient from a computational perspective, may not be very realistic or too conservative when dynamic obstacles to be avoided are e.g., elongated rigid bodies. For this reason, we believe that the main problem to solve in the near future is to extend the VO to the case of dynamic non-spherical obstacles.

## CRediT authorship contribution statement

**Federico Vesentini:** Writing – original draft, Writing – review & editing. **Riccardo Muradore:** Supervision, Writing – review & editing. **Paolo Fiorini:** Supervision.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

No data was used for the research described in the article.

## References


[1] J. Chen, D. Sun, Resource constrained multirobot task allocation based on leader-follower coalition methodology, Int. J. Robot. Res. 30 (12) (2011) 1423–1434.

[2] P. Stone, M. Veloso, Task decomposition, dynamic role assignment and low-bandwith communication for real-time strategic teamwork, Artificial Intelligence 110 (2) (1999) 241–273.

[3] T. Balch, R.C. Arkin, Behaviour-based formation control for multirobot teams, IEEE Trans. Robot. Autom. 14 (6) (1998) 1501–1514.

[4] J. Chen, D. Sun, J. Yang, H. Chen, Leader-follower formation control of multiple non-holonomic mobile robots incorporating a receding-horizon scheme, Int. J. Robot. Res. 29 (6) (2010) 727–747.

[5] C. De La Cruz, R. Carelli, Dynamic model based formation control and obstacle avoidance of multi-robot systems, Robotica 26 (3) (2008) 345–356.

[6] D. Sun, C. Wang, W. Shang, G. Feng, A synchronization approach to trajectory tracking of multiple mobile robots while maintaining time-varying formation, IEEE Trans. Robot. 25 (5) (2009) 1074–1086.

[7] J. Alonso-Mora, S. Baker, Rus D., Multi-robot formation control and object transport in dynamic environments via constrained optimization, Int. J. Robot. Res. 36 (9) (2017) 1000–1021.

[8] M. Turpin, N. Michael, V. Kumar, Concurrent assignment and planning of trajectories for multiple robots, Int. J. Robot. Res. 33 (1) (2014) 98–112.

[9] S. Hu, D. Sun, Automatic transportation of biological cells with a robot-tweezer manipulation system, Int. J. Robot. Res. 30 (14) (2011) 1681–1694.

[10] N. Michael, J. Fink, V. Kumar, Cooperative manipulators and transformation with aerial robots, Auton. Robots 30 (1) (2011) 73–86.

[11] Y. Shourkry, P. Nuzzo, A.L. Sangiovanni-Vincentelli, S.A. Seshia, G.J. Pappas, P. Tabuada, SMC: Satisfiability modulo convex optimization, in: Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control, 2017, pp. 19–28.

[12] R. Luna, K.E. Bekris, Efficient and complete centralized multi-robot path planning, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2011, pp. 3268–3275.

[13] G. Sharon, R. Stern, A. Felner, N.R. Sturtevant, Conflict based search for optimal multi-agent path finding, Artificial Intelligence 219 (2015) 40–66, (Elsevier).

[14] J. Yu, S.M. LaValle, Optimal multirobot path planning on graphs: Complete algorithms and effective heuristics, IEEE Trans. Robot. 32 (5) (2016) 1163–1177.

[15] S. Tang, J. Thomas, V. Kumar, Hold or take optimal plan (HOOP): A quadratic programming approach to multi-robot trajectory generation, Int. J. Robot. Res. 37 (9) (2018) 1062–1084.

[16] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, Int. J. Robot. Res. 5 (1) (1986) 396–404.

[17] D. Helbing, P. Molnar, Social force model for pedestrian dynamics, Phys. Rev. E 51 (5) (1995) 4282–4428.

[18] D. Fox, W. Burgand, S. Thrun, The dynamic window approach to collision avoidance, IEEE Robot. Autom. Mag. 4 (1) (1997) 23–33.

[19] T. Keviczky, F. Borrelli, G.J. Balas, A study on decentralized receding horizon control for decoupled systems, in: IEEE American Control Conference, Vol. 6, 2004, pp. 4921–4926.

[20] P. Fiorini, Z. Shiller, Motion planning in dynamic environments using the relative velocity paradigm, in: Proceedings IEEE International Conference on Robotics and Automation, 1993, pp. 560–565.

[21] P. Fiorini, Z. Shiller, Motion planning in dynamic environments using velocity obstacles, Int. J. Robot. Res. 17 (7) (1998) 760–772.

[22] T. Lozano-Perez, Spatial planning: A configuration space approach, in: Autonomous Robot Vehicles, Springer, 1990.

[23] J. Van Den Berg, M. Lin, D. Manocha, Reciprocal velocity obstacles for real-time multi-agent navigation, in: IEEE the International Conference on Robotics and Automation, 2008, pp. 1928–1935.

[24] J. Van Den Berg, S.J. Guy, M. Lin, D. Manocha, Reciprocal n-body collision avoidance, in: The International Symposium on Robotics Research, 2009, pp. 3–19.

[25] J. Van Den Berg, J. Snape, S.J. Guy, D. Manocha, Reciprocal collision avoidance with acceleration-velocity obstacles, in: IEEE International Conference on Robotics and Automation, 2011, pp. 3475–3482.

[26] J. Snape, J. Van Den Berg, S.J. Guy, D. Manocha, The hybrid reciprocal velocity obstacle, IEEE Trans. Robot. 27 (4) (2011) 696–706.

[27] J. Alonso-Mora, A. Breitenmoser, M. Rufli, P. Beardsley, R. Siegwart, Optimal reciprocal collision avoidance for multiple non-holonomic robots, in: Springer Tracts in Advanced Robotics, vol. 83, 2013, pp. 203–216.

[28] D.E. Koditschek, Robot planning and control via potential functions, Robot. Rev. 1 (1989) 349–367.

[29] J. Barraquand, J.C. Latombe, Robot motion planning: A distributed representation approach, Int. J. Robot. Res. 10 (1991) 628–649.

[30] J. Barraquand, B. Langlois, J.C. Latombe, Numerical potential field techniques for robot path planning, IEEE Trans. Syst. Man Cybern. 22 (2) (1992) 224–241.

[31] Y. Koren, J. Borestein, Potential fields methods and their inherent limitations for mobile robot navigation, in: IEEE International Conference of Robotics and Automation, Vol. 2, 1991, pp. 1398–1404.

[32] C. Qixin, H. Yanwen, Z. Jingliang, An evolutionary artificial potential field algorithm for dynamic path planning of mobile robot, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006, pp. 3331–3336.

[33] S.S. Ge, Y.J. Cui, Dynamic motion planning for mobile robots using potential field method, Auton. Robots 13 (3) (2002) 207–222.

[34] A. Poty, P. Melchior, A. Oustaloup, Dynamic path planning by fractional potential, in: Second IEEE International Conference on Computational Cybernetics, 2004, pp. 365–371.

[35] K. Lewin, Field Theory in Social Science: Selected Theoretical Papers, Harper, New York, 1951.

[36] D. Helbing, Boltzmann-like and Boltzmann-Fokker–Planck equations as a foundation of behavioral models, Physica A 196 (1993) 546–573.

[37] A.E. Scheflen, N. Ashcraft, Human Territories: How We Behave in Space-Time, Prentice-Hall, Englewood Cliffs, NJ, 1976.

[38] TRB, Highway Capacity Manual, Special Report No. 209, Transportation Research Board (TRB) editor National Research Council, Washington DC, 1985, Chap. 13.

[39] R.D. Freimuth, L. Lam, in: L. Lam, V. Naroditsky (Eds.), Modeling Complex Phenomena, Springer, New York, 1992.

[40] L. Lam, Nonlinear Physics for Beginners: Fractals, Chaos, Solitons, Pattern Formation, Cellular Automata and Complex Systems, World Scientific, 1998.

[41] F. Schweitzer, L. Schimansky-Geier, Clustering of active walkers in a two-component system, Physica A 206 (3–4) (1994) 359–379.

[42] D. Helbing, P. Molnar, F. Schweitzer, Computer simulation of pedestrian dynamics and trail simulation, in: Evolution of Natural Structures, Sonderforschungsbereich 230, Stuttgart, 1994, pp. 229–234.

[43] R. Simmons, The curvature-velocity method for local obstacle avoidance, in: IEEE International Conference on Robotics and Automation, Vol. 4, 1996, pp. 3375–3382.

[44] O. Brock, O. Khatib, High-speed navigation using the global dynamic window approach, in: IEEE International Conference on Robotics and Automation, Vol. 1, 1999, pp. 341–346.

[45] P. Ogren, N.E. Leonard, A convergent dynamic window approach to obstacle avoidance, IEEE Trans. Robot. 21 (2) (2005) 188–195.

[46] D.Q. Mayne, Control of constrained dynamic systems, Eur. J. Control 7 (2–3) (2017) 87–99.

[47] J.M. Park, D.W. Kim, Y.S. Yoon, J.H. Kim, K.S. Yi, Obstacle avoidance of autonomous vehicles based on model predictive control, Proc. Inst. Mech. Eng. D 223 (2009) 1499–1516.
[48] H. Jiang, Z. Wang, Q. Chen, J. Zhu, Obstacle avoidance of autonomous vehicles with CQP-based model predictive control, in: IEEE International Conference on Systems, Man and Cybernetics, SMC, 2016, pp. 1668–1673.
[49] P. Fiorini, Z. Shiller, Robot Motion Planning among Moving Obstacles (Ph.D. thesis), University of California, 1995.
[50] P. Fiorini, Z. Shiller, Time optimal trajectory planning in dynamic environments, in: IEEE International Conference on Robotics and Automation, Vol. 2, 1996, pp. 1553–1558.
[51] P. Fiorini, Z. Shiller, Robot motion planning in dynamic environments, in: Robotics Research, Springer, 1996, pp. 237–248.
[52] S.J. Guy, J. Chhugani, C. Kim, N. Satish, M. Ling, D. Manocha, P. Dubey, ClearPath: Highly parallel collision avoidance for multi-agent simulation, in: Eurographics/ ACM SIGGRAPH Symposium on Computer Animation, 2009, pp. 177–187.
[53] D. Wilkie, J. Van Den Berg, D. Manocha, Generalized velocity obstacle, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009, pp. 5573–5578.
[54] Z. Shiller, O. Gal, T. Fraichard, The nonlinear velocity obstacle revisited: The optimal time horizon, in: IEEE International Conference on Robotics and Automation, Workshop on Guaranteeing Safe Navigation in Dynamic Environments, 2010.
[55] T. Battisti, R. Muradore, A velocity obstacles approach for autonomous landing and teleoperated robots, Auton. Robots 44 (2) (2020) 217–232.
[56] J. Snape, D. Manocha, Goal Velocity Obstacles for Spatial Navigation of Multiple Autonomous Robots or Virtual Agents, International Foundation for Autonomous Agents and Multi-agent Systems, 2013.
[57] D. Bareiss, J. Van Den Berg, Generalized reciprocal collision avoidance, Int. J. Robot. Res. 34 (12) (2015) 1501–1514.
[58] M. Kim, J. Oh, Study on optimal velocity selection using velocity obstacle (OVVO) in dynamic and crowded environment, Auton. Robots J. 40 (8) (2016) 1459–1470.
[59] E. Prassler, J. Scholz, P. Fiorini, Navigating a robotic wheelchair in a railway station during rush hour, Int. J. Robot. Res. 18 (7) (1999) 711–727.
[60] E. Prassler, J. Scholz, P. Fiorini, A robotic wheelchair for crowded public environments, IEEE Robot. Autom. Mag. 8 (1) (2001) 38–45.
[61] J. Snape, J. Van Der Berg, S.J. Guy, D. Manocha, Independent navigation of multiple mobile robots with hybrid reciprocal velocity obstacles, IEEE/RSJ Int. J. Intell. Robots Syst. (2009) 5917–5922.
[62] iRobot manufacturer, iRobot create, 2007, https://en.wikipedia.org/wiki/IRobot_Create.
[63] H. Kato, M. Billinghurst, Marker tracking and HMD calibrationfor a video-based augmented reality conferencing system, in: The Second IEEE/ACM International Workshop on Augmented Reality, 1999, pp. 85–94.
[64] G. Welch, G. Bishop, An Introduction to the Kalman Filter, Department of Computer Science University of North Carolina at Chapel HillChapel Hill, NC 27599-3175, 1995, pp. 127–132.
[65] J. Snape, J. Van Den Berg, S.J. Guy, D. Manocha, Smooth and collision-free navigation for multiple robots under differential-drive constraints, in: The IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010, pp. 4584–4589.
[66] N. Piccinelli, F. Vesentini, R. Muradore, Planning with real-time collision avoidance for cooperating agents under rigid body constraints, in: Design, Automation & Test in Europe Conference & Exhibition, DATE, Florence, Italy, 2019, pp. 1261–1264.
[67] F. Aurenhammer, Voronoi diagrams — A survey of a fundamental geometric data structure, ACM Comput. Surv. 23 (3) (1991) 345–405.
[68] M. Foskey, M. Garber, M. Lin, D. Manocha, A voronoi-based hybrid motion planner, in: Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vol. 1, 2001, pp. 55–60.
[69] L. He, J. Van Den Berg, Meso-scale planning for multi-agent navigation, in: IEEE International Conference on Robotics and Automation, 2013, pp. 2839–2844.
[70] J. Van Den Berg, S. Patil, J. Sewall, D. Manocha, M. Lin, Interactive navigation of multiple agents in crowded environments, in: I3D Symphosium on Interactive 3D Graphics and Games, 2008, pp. 139–147.
[71] A. Golas, R. Narain, S. Curtis, M.C. Lin, Hybrid long-range collision avoidance for crowd simulation, IEEE Trans. Vis. Comput. Graphics 20 (7) (2014) 1022–1034.
[72] B. Kluge, E. Prassler, Recursive probabilistic velocity obstacle for reflective navigation, in: Field and Service Robotics, Springer, 2003, pp. 71–79.
[73] B. Gopalakrishnan, A.K. Singh, M. Kaushik, K.M. Krishna, D. Manocha, PRVO: Probabilistic reciprocal velocity obstacle for multi robot navigation under uncertainty, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, 2017, pp. 1089–1096.
[74] T. Fan, P. Long, W. Liu, J. Pan, Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios, Int. J. Robot. Res. 39 (7) (2020) 856–892.
[75] J. Van Den Berg, S.J. Guy, J. Snape, M.C. Lin, D. Manocha, http://gamma.cs.unc.edu/RVO2/. Department of Computer Science, University of North Carolina at Chapel Hill, 2016.
[76] J.A. Douthwaite, S. Zhao, S.L. Mihaylova, A comparative study of velocity obstacle approaches for multi-agent systems, in: UKACC 12th International Conference on Control, 2018, pp. 289–294.
[77] K. Guo, D. Wang, T. Fan, J. Pan, VR-ORCA: Variable responsibility optimal reciprocal collision avoidance, IEEE Robot. Autom. Lett. 6 (3) (2021) 4520–4527.
[78] J. Alonso-Mora, A. Breitenmoser, P. Beardsley, R. Siegwart, Reciprocal collision avoidance for multiple car-like robots, in: IEEE International Conference on Robotics and Automation, RiverCentre, Saint Paul, Minnesota, USA, 2012, pp. 360–366.
[79] I. Khare, J. Poonganam, B. Gopalakrishnan, K.M. Krishna, Probabilistic inverse velocity obstacle for free flying quadrotors, in: European Control Confer- ence, ECC, Rotterdam, Netherlands, 2021, pp. 1711–1718.
[80] F. Vesentini, N. Piccinelli, R. Muradore, Velocity obstacle-based trajectory planner for anthropomorphic arms, Eur. J. Control (2023) 100901, Elsevier.
[81] F. Vesentini, R. Muradore, Velocity obstacle-based trajectory planner for two-link planar manipulator, in: European Control Conference, ECC, Rotterdam, Netherlands, 2021, pp. 687–692.
[82] C. Morasso, D. Meli, Y. Divet, S. Sessa, A. Farinelli, Planning and inverse kinematics of hyper-redundant manipulators with VO-FABRIK, in: Publication in Springer Proceedings in Advanced Robotics, Special Issue for the European Robotics Forum, ERF, 2024, pp. 1–7.
[83] N. Piccinelli, F. Vesentini, R. Muradore, MPC based motion planning for mobile robots using velocity obstacle paradigm, in: European Control Conference (ECC) 2023, Bucharest, Romania, 2023, pp. 1–6.
[84] L. Bonanni, D. Meli, A. Castellini, A. Farinelli, Monte Carlo planning for mobile robots in large action spaces with velocity obstacles, in: Publication on the Proceeding of the 10th Italian Workshop on Artificial Intelligence and Robotics, AIRO, 2023, pp. 1–7.

**Federico Vesentini** is currently a Post-Doc Researcher at the Department of Engineering for Innovation Medicine, at the University of Verona. He received his Bachelor's Degree in Applied Mathematics in 2013, followed by a Master's Degree in Mathematics in 2017, and ultimately, the Ph.D. in Computer Science in 2022, all from the University of Verona in Italy. His research interests center around hybrid systems, stochastic filtering, machine learning and collision avoidance for autonomous mobile robots and robotic manipulators.

**Riccardo Muradore** is Associate Professor at the Department of Engineering for Innovation Medicine of the University of Verona. He received the Laurea degree in Information Engineering in 1999 and the Ph.D. degree in Electronic and Information Engineering in 2003 both from the University of Padova (Italy). He held a post-doctoral fellowship at the Department of Chemical Engineering, University of Padova, from 2003 to 2005, working on statistical control and monitoring. Then he spent three years at the European Southern Observatory in Munich (Germany) as control engineer working on adaptive optics systems. In 2008 he joined the ALTAIR Robotics laboratory, University of Verona (Italy). He was the coordinator of the Horizon 2020 project SARAS (https://saras-project.eu/) and he is the local coordinator for UNIVR of the Horizon Europe GEYEDANCE project (https://geyedance.eu/) and the coordinator of the Horizon Europe ROBIOPSY project (https://robiopsy-project.eu/). His research interests include control and system theory, teleoperation, robotics, surgical robotics, predictive maintenance, networked control systems and adaptive optics.

**Paolo Fiorini** received the Laurea in Electronic Engineering from the University of Padova, (Italy), the MSEE from the University of California at Irvine (USA), and the Ph.D. in ME from UCLA (USA). From 1977 to 1985 he worked for companies in Italy and in the USA. From 1985 to 2000, he was with NASA Jet Propulsion Laboratory, California Institute of Technology. In 2001 returned to the School of Science and Engineering of the University of Verona (Italy) as Professor of Computer Science. In 2001 he founded the ALTAIR robotics laboratory. His research was funded by EU programs FP6, FP7, H2020 and ERC Advanced Grant. He is an IEEE Fellow (2009). In 2022 he founded Needleye Robotics to commercialize his research in medical robotics. In 2023 he retired from academia and is now the CEO of Needleye Robotics.