Reactive Navigation in Crowds for Non-Holonomic Robots With Convex Bounding Shape

David J. Gonon, Diego Paez-Granados, and Aude Billard

Abstract—**This letter describes a novel method for non-holonomic robots of convex shape to avoid imminent collisions with moving obstacles. The method's purpose is to assist navigation in crowds by correcting steering from the robot's path planner or driver. We evaluate its performance using a custom simulator which replicates real crowd movements and corresponding metrics which quantify agents' efficiency and the robot's impact on the crowd and count collisions. We implement and evaluate the method on the standing wheelchair Qolo. In our experiments, it drives in autonomous mode using on-board sensing (LiDAR, RGB-D camera and a system to track pedestrians) and avoids collisions with up to five pedestrians and passes through a door.**

***Index Terms*—Collision avoidance, reactive and sensor-based planning, human-aware motion planning, velocity obstacle.**

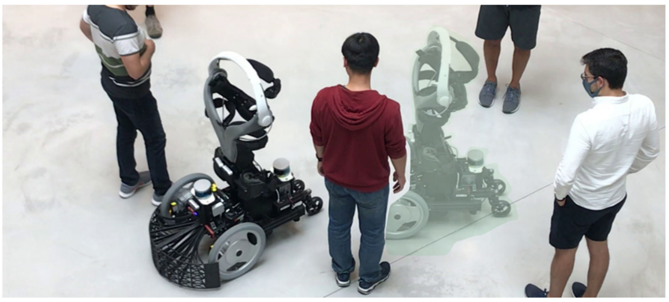
Fig. 1. The robot Qolo is passing between pedestrians using the proposed reactive controller (from right to left).

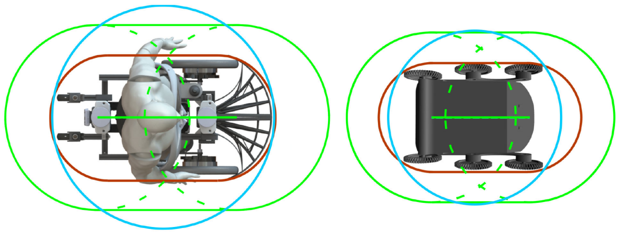
Fig. 2. Qolo (left) and Starship's delivery robot (right) have footprints which capsules can bound well tightly (red) or more conservatively (green) and still yield a smaller width than the tightest bounding circle (blue).

I. Introduction

THIS work considers robots that need to navigate within crowds to reach their goal (as in Fig. 1), such as e.g. electric wheelchairs and delivery robots (see also Fig. 2). This is challenging because individual pedestrians' decisions are uncertain and also, pedestrians expect cooperative behaviour, as they anticipate and leave space for each other's future motion. Thus, robots need to coordinate with pedestrians but also to adapt quickly to surprising behaviour and avoid imminent collisions that can endanger humans. When evaluating such a robot's controller, one needs to take into account its impact on pedestrians to quantify the robot's social performance.

This letter is about avoiding imminent collisions between pedestrians and a mobile robot that is non-holonomic (e.g. having wheels preventing sideways motion) and non-circular (e.g. being of elongated shape). We propose the method Reactive Driving Support (RDS) for such robots to correct nominal commands (from the driver or high-level planner) as far as necessary for avoiding previously unanticipated yet imminent collisions. RDS employs Velocity Obstacles (VO) [1], [2], whose basic concept we describe in detail in Section II-A. RDS constructs VO between each obstacle and the robot's closest subpart and constrains its velocity accordingly to avoid collisions locally. This constitutes a novel way to extend VO to non-holonomic robots of non-circular shape. This letter formulates RDS for a capsule, which is a generic shape that fits many delivery robots and robotic wheelchairs (see Fig. 2). RDS does not require pre-processing which merges overlapping obstacles (in contrast to e.g. [3], [4]) and is computationally lightweight itself even for very many obstacles. Its implementation is publicly available at https://github.com/epfl-lasa/rds/.

Manuscript received October 15, 2020; accepted February 20, 2021. Date of publication March 24, 2021; date of current version April 13, 2021. This letter was recommended for publication by Associate Editor L. Tapia and Editor N. Amato upon evaluation of the reviewers' comments. This work was supported by the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No. 779942 ("Crowdbot"). *(Corresponding author: David J. Gonon.)*

The authors are with the LASA, School of Engineering, Swiss Federal School of Technology in Lausanne - EPFL, Lausanne 1015 Vaud, Switzerland (e-mail: david.gonon@epfl.ch; dfpg@ieee.org; aude.billard@epfl.ch).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2021.3068660, provided by the authors.

Digital Object Identifier 10.1109/LRA.2021.3068660

A. Related Work

There haven been diverse research efforts about robotic navigation in crowds recently. They have mostly modeled robots as circles that can move omni-directionally, thereby idealizing the shape and kinematics to ease investigating the specific properties and challenges which crowds create. Particularly, they have explored navigating cooperatively and according to social norms [5]–[8], predicting the surrounding crowd's future motion [9], planning the robot's motion beyond the interactions with its immediate neighbours [10], and conservative collision avoidance under incomplete knowledge about obstacles' position and behaviour [11].

Several approaches exist for circular [12] or even non-convex [13] robots to avoid static obstacles by optimizing over discrete candidate trajectories under dynamic and kinematic constraints. Vector Field Histogram Plus [14] is another optimization-based method, which identifies narrow passages for the robot, but it also assumes static environments. The method in [11] guarantees that the robot is at rest when a collision happens for any behaviour of obstacles, which can be prohibitive in crowds. Another method [15] aims at increasing the feasible command space by conditioning on surrounding agents' most probable maneuvers, but it also assumes a circular robot.

Methods using VO are suitable for short-term planning in dynamic environments such as crowds. The classical VO [1] and its derivatives, e.g. Optimal Reciprocal Collision Avoidance (ORCA) [2], assume circular holonomic agents. Some frameworks enable their application to non-holonomic robots, including [16] and a simple approach that shifts the center (which we discuss in Section II-B). However, they artificially increase the robot's radius and thus reduce the capability to plan through narrow passages. An extension of the VO concept to non-linear motion control models [17] can directly treat non-holonomic circular vehicles. However, this relies on forward-integration to yield trajectory candidates from a discretized solution space, which is computationally expensive. Other methods [18], [19] treat non-circular robots with holonomic kinematics, where they separate the step for computing rotations from the step applying VO to find translations. Such an approach is not applicable to robots with non-holonomic constraints, where translation and rotation are coupled. [20] introduce a velocity-continuous formulation of VO which is applicable to non-holonomic but only circular robots.

Crowd simulations have been common as tools to evaluate and study methods for navigation in crowds [6], [8], [11], [21]. Many such evaluations assign a fixed goal to each agent and evaluate the ability to find an efficient trajectory to the goal by suitable metrics. Very common metrics include agents' path length and time for traveling from their starting to their goal positions for quantifying efficiency, the rate of success (reaching the goal without collision), and the number of collisions or minimum separation of agents for quantifying safety (e.g. in [5], [6], [8], [21]–[23]). The evaluation in [24] has the most similar perspective to ours, as it measures the robot's deviations from the high-level planner's (time-independent) reference path by the squared deviation integral.

B. Contributions

Our method RDS extends VO to non-holonomic non-circular robots for reactive control in crowds. This letter presents RDS and its evaluation tailored to this context, where we employ a custom simulator and four novel metrics for quantitative evaluation. The simulator combines original time-dependent trajectories of video-tracked pedestrians (from [25]) with local collision avoidance such that agents incorporate high-level planning, social coordination, and local collision avoidance. By using empirical reference trajectories, we aim at making agents' arrangements and motion patterns representative of the original crowd. Corresponding to these time-varying goals, we introduce the robot's and the crowd's average deviation from their reference trajectories as metrics to quantify how efficient a reactive controller's corrections are at avoiding collisions while continuing tracking of the reference motions. Thus, our first two metrics reflect the task's focus on complementing high-level motion plans. Further, we introduce two metrics that directly measure how a robot's presence and its controller impact other agents' velocities. While it is common to consider velocities, measuring one specific agent's impact on others is far less common. Both of these metrics relate to the compatibility between a robot's and pedestrians' different ways of navigation.

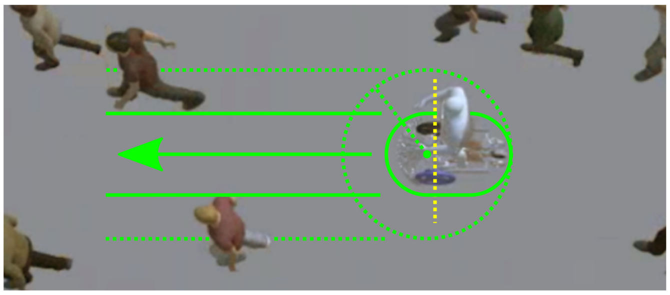
Fig. 3. The capsule approximation (solid green) fits through gaps which are not accessible with an abstracting circle (dotted green) centered ahead of the wheel axle (yellow).

Complementing this evaluation by experiments with the standing wheelchair Qolo [26] (in Fig. 1, 2, 3), on which we have implemented our method, we demonstrate the method to be effective at avoiding collisions and feasible in practice. The comparison with another method shows that our approach is more advantageous in crowds due to its ability to lead through narrow gaps between obstacles. The next section (Section II) provides the concepts underlying this letter's techniques. The method's description (Section III), its evaluation in simulation (Section IV) and with the robot Qolo (Section V) follow. Finally, Section VI concludes this work.

II. Background

This section reviews important technical prerequisites.

A. Velocity Obstacles for Circular Holonomic Robots

The original VO [1] describes for each circular obstacle the corresponding cone-shaped set of constant velocities for the circular robot that will eventually lead to a collision (as Fig. 4 shows). ORCA [2] uses VO for multi-agent navigation as follows (see also Fig. 4). It disregards collisions after the time horizon τ and thus spherically truncates the VO cone. Further, ORCA conservatively approximates the relative VO by the halfplane $\bar{H}_{rel}$ whose boundary touches the relative VO boundary at its point closest to the previous relative velocity $\mathbf{v}_{rel}^{-}$. Then, it obtains the reciprocal absolute VO for the first agent A by shifting the

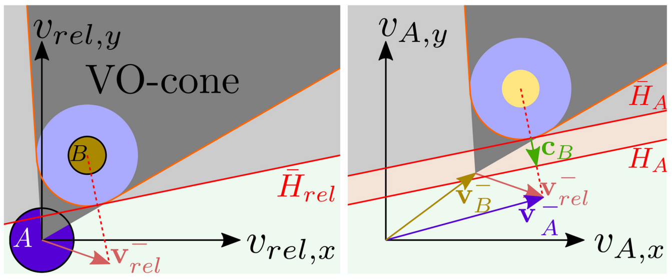

Fig. 4. In relative velocity space (left), ORCA truncates and linearizes the relative VO (dark gray) between the robot A (blue) and the obstacle B (gold) around the previous relative velocity $\mathbf{v}^-_{rel}$. Shifting the resulting halfplane $\bar{H}_{rel}$ by the obstacle's previous velocity $\mathbf{v}^-_B$ yields the halfplane $\bar{H}_A$ of avoiding velocities in the robot's velocity space (right), if B keeps its velocity. Shifting by B's reciprocal contribution $\mathbf{c}_B$ yields the halfplane H_A of reciprocally avoiding velocities.

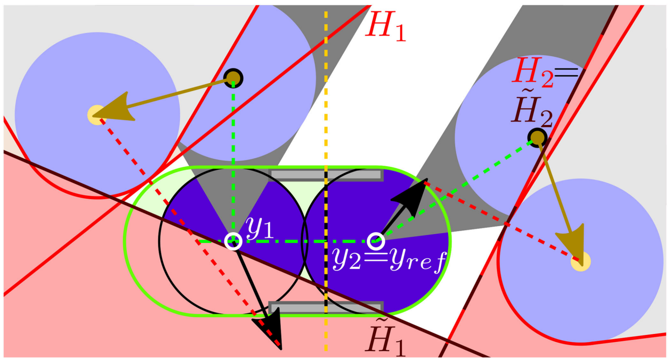

Fig. 5. The method constructs the velocity obstacles (light grey) for obstacles (gold) and the robot's closest incircle (dark blue), linearizes them as H_i (red) and re-maps them as $\tilde{H}_i$ (dark red) to the reference point's velocity space.

relative VO by the second agent B's previous velocity plus its reciprocal contribution $\mathbf{c}_B \in \mathbb{R}^2$, which accounts for the second agent's contribution to collision avoidance, and vice-versa for the second agent. ORCA thus generates a linear constraint for an agent's velocity due to each other agent, where each constraint is characterized by its normal $\mathbf{n}$ and offset b. Finally, it solves a quadratic program to find the velocity closest to the preferred one while adhering to the constraints.

B. Abstraction From the Robot's Shape and Kinematics

We define for comparison with our method later on the baseline method as using ORCA [2] (assuming no reciprocity, i.e. $\mathbf{c}_B = \mathbf{0}$) for the given non-holonomic non-circular robot by the following abstracting technique (from [27]). We consider a generic robot's kinematic model which is conceptually equivalent to an axle with two wheels that rotate independently around it and roll on a plane without slip. The model views the robot as a rigid body which moves in the plane and is subject to the non-holonomic constraint which requires its instantaneous center of rotation to be on the infinite line that contains the wheel axle.

For abstracting from such non-holonomic kinematics and the robot's possibly non-circular shape, [27] defines the control point as a point on the robot's body that does not lie on the wheel axle. The control point's cartesian velocity has degree of freedom two and thus it serves to receive velocity commands that address a holonomic robot. Further, a circle whose center is the control point and which contains the robot serves to mask its true shape such that one can apply conservatively to this virtual circle a method that avoids collisions for a circular robot. We note that the control point needs to be ahead of the wheel axle in the robot's preferred direction of travel (if it exists) to yield proper signs of angular velocities when avoiding obstacles, i.e. rotating clockwise/counter-clockwise when passing on the right/left, respectively. Consequentially, the virtual circle for this abstraction is particularly conservative when the robot is longer in the rearward than in the forward direction from the wheel axle (as Fig. 3 shows for Qolo).

III. Method

The method avoids collisions by constructing constraints for the velocity of a particular robot-fixed point, which we refer to as the reference point, according to Fig. 5. Each obstacle is taken into account by determining the robot's incircle which is closest to it and constructing the VO that the obstacle induces for the incircle. A linear constraint is derived for the incircle center's velocity and transformed into the equivalent constraint for the reference point's velocity via the robot's kinematic relation between different points' velocities.

A. Definitions

We assume the robot to have non-holonomic kinematics according to Section II-B. The robot's fixed right-handed coordinate system is defined such that the y-axis separates the wheels symmetrically and points forward and the x-axis coincides with the wheel axle's line. Any cartesian vector components in the method's description refer to this coordinate system. The method's description here assumes the robot's shape as a capsule which is symmetric in the y-axis and corresponds to sweeping a circle of radius r with its center on $x = 0$ from the rear end $y_{rear} < 0$ to the front end $y_{front} > 0$. An incircle is any circle with its center on the line between the endpoints and radius r.

The command vector $\mathbf{u} = [v, \omega]^T$ defines the robot's linear and angular velocity command v and ω in such a way that positive values yield forward translations and counter-clockwise rotations around the origin, respectively. Given any point (x, y), its cartesian velocity $\mathbf{v}$ corresponding to $\mathbf{u}$ can be expressed via the Jacobian $\mathbf{J}(x, y)$ as $\mathbf{v} = \mathbf{J}(x, y)\mathbf{u}$. One can show that

$$\mathbf{J}(x,y) = \begin{bmatrix} 0 & -y \\ 1 & x \end{bmatrix}, \quad \mathbf{J}^{-1}(x,y) = \begin{bmatrix} x/y & 1 \\ -1/y & 0 \end{bmatrix}, \quad (1)$$

where $\mathbf{J}^{-1}$ exists for any point (x, y) with $y \neq 0$. Finally, (x_{ref}, y_{ref}) defines the reference point with $y_{ref} \neq 0$. Alternatively to $\mathbf{u}$, its velocity $\mathbf{v}_{ref}$ can describe via the inverse of its Jacobian $\mathbf{J}_{ref}$ the robot's motion, as $\mathbf{u} = \mathbf{J}^{-1}_{ref}\mathbf{v}_{ref}$.

B. Velocity Constraints for Local Incircles

Assuming circular obstacles (with known radii, positions and velocities), the method constructs for each obstacle O_i the VO which it induces for the respective closest incircle, located at $(0, y_i)$. The VO's construction, truncation by the time horizon τ, and linearization around the incircle center's and the obstacle's previous relative velocity are identical to the approach in [2] (as in Section II-A) apart from reciprocity, i.e. the method assumes the obstacle to maintain its velocity (corresponding to $\mathbf{c}_B = \mathbf{0}$ in Fig. 4). Thus, each obstacle O_i creates for the velocity $\mathbf{v}_i$ of the point $(0, y_i)$ the linear constraint

$$\mathbf{n}_i^T \mathbf{v}_i \leq b_i, \tag{2}$$

which describes the feasible halfplane $H_i(\mathbf{n}_i, b_i)$ with the outwards unit normal $\mathbf{n}_i$ and the offset from the origin b_i.

C. Optimization Problem

The method aims at optimizing the command vector $\mathbf{u}$ for the robot to execute such that it minimally deviates from the nominal command $\bar{\mathbf{u}} = [\bar{v}, \bar{\omega}]^T$ e.g. by the driver. The optimization problem is formulated in the reference point's velocity space, wherein the nominal command is mapped as $\bar{\mathbf{v}}_{ref} = \mathbf{J}_{ref}\bar{\mathbf{u}}$. Further, the method maps in this space the constraints for the incircles' velocities due to N obstacles by expressing the local velocity in each constraint (2) as $\mathbf{v}_i = \mathbf{J}(0, y_i)\mathbf{J}_{ref}^{-1}\mathbf{v}_{ref}$ using (1). We incorporate these constraints and the objective in the quadratic program

$$\mathbf{v}_{ref}^* = \arg\min_{\mathbf{v}_{ref}} |\mathbf{v}_{ref} - \bar{\mathbf{v}}_{ref}|^2 \tag{3}$$

$$\text{s.t. } \mathbf{n}_i^T \mathbf{J}(0, y_i)\mathbf{J}_{ref}^{-1}\mathbf{v}_{ref} \leq b_i \quad \forall i \in 1, \ldots, N \tag{4}$$

$$\mathbf{n}_{v,j}^T \mathbf{v}_{ref} \leq b_{v,j} \quad \forall j \in 1, \ldots, 4 \tag{5}$$

$$\mathbf{n}_{a,k}^T \mathbf{v}_{ref} \leq b_{a,k} \quad \forall k \in 1, \ldots, 4 \tag{6}$$

where the additional constraints in (5), (6) represent the robot's velocity and acceleration limits, respectively. We assume that they result respectively from four fixed constraints for $\mathbf{u}$ and from the four box constraints around the previous command $[v^-, \omega^-]^T$ which encode $|v - v^-| \leq \hat{a}\Delta t$ and $|\omega - \omega^-| \leq \hat{\alpha}\Delta t$, with $\hat{a}$ and $\hat{\alpha}$ denoting the maximum absolute linear and angular acceleration, respectively, and the control cycle time Δt. These constraints' normals are multiplied by $\mathbf{J}_{ref}^{-T}$ and their offsets are adopted to yield $\mathbf{n}_{v,j}, b_{v,j}, \mathbf{n}_{a,k}, b_{a,k}$ in (5), (6). However, one can employ any given number and arrangement of constraints instead.

D. Solution and Command Computation

The method employs an incremental algorithm very similar to [28], [29] to solve the quadratic program (3)–(6) or determine that its constraints are infeasible. Importantly, the maximum number of obstacles bounds a priori the number of iterations which the algorithm requires. If the constraints are feasible, the solution defines the velocity command according to $\mathbf{u}^* = \mathbf{J}_{ref}^{-1}\mathbf{v}_{ref}^*$. Otherwise, the pre-defined maximum linear and angular braking decelerations $\hat{a} > 0$ and $\hat{\alpha} > 0$ define the command according to $v^* = h(v^-, \hat{a})$ and $\omega^* = h(\omega^-, \hat{\alpha})$, where $h(u, m) \coloneqq u - \text{sign}(u)\min(|u|, \Delta t\, m)$.

E. Discussion and Generalizations

Two simplifying approximations underlie the method. First, it reduces the robot's shape to the closest incircle for each obstacle. Second, it approximates the incircles' motions over the planning horizon as straight with individual constant velocities such that they only initially verify a rigid body's velocity distribution. Consequentially, the method requires a high control frequency to prevent collisions, i.e. it must update the robot's velocity command not just as the time horizon elapses but early enough such that both approximations remain valid. However, this is also necessary as obstacles may change their velocities faster than the horizon.

The choice (x_{ref}, y_{ref}) defines the relative costs for deviating from $\bar{\mathbf{u}}$ along different axes in the command space. When viewing $\mathbf{u}$ as the optimization variable in (3) by plugging in $\mathbf{v}_{ref} = \mathbf{J}_{ref}\mathbf{u}$, the objective becomes the quadratic form defined by $\mathbf{J}_{ref}^T\mathbf{J}_{ref}$, whose principal axes and eigenvalues can be tuned via (x_{ref}, y_{ref}).

The expression (4) maps each halfplane $H_i(\mathbf{n}_i, b_i)$ (constraining the local velocity $\mathbf{v}_i$) to the corresponding constraint for the reference point's velocity $\mathbf{v}_{ref}$. With $\mathbf{M}(x, y) :=\mathbf{J}_{ref}^{-T}\mathbf{J}(x, y)^T$, the latter constraint can be geometrically interpreted, if $\mathbf{M}(0, y_i)\mathbf{n}_i \neq \mathbf{0}$, as the transformed feasible halfplane $\tilde{H}_i(\tilde{\mathbf{n}}_i, \tilde{b}_i)$ with the unit normal $\tilde{\mathbf{n}}_i = \mathbf{M}(0, y_i)\mathbf{n}_i / ||\mathbf{M}(0, y_i)\mathbf{n}_i||$ and the offset $\tilde{b} = b / ||\mathbf{M}(0, y_i)\mathbf{n}_i||$ (see also Fig. 5). In the case $\mathbf{M}(0, y_i)\mathbf{n}_i = \mathbf{0}$, the constraint reads $0 \leq b_i$, which occurs if and only if $y_i = \mathbf{n}_{i,y} = 0$ (e.g. when an obstacle approaches the static robot along the line $y = 0$). There, the local VO constrains $\mathbf{v}_{i,x}$ independently (of $\mathbf{v}_{i,y}$) while kinematically any command must yield $\mathbf{v}_{i,x} = 0$. Thus, $\mathbf{v}_{ref}$ does not enter the constraint, which makes the optimization infeasible and triggers braking if $b_i < 0$. This effect limits the ability to escape close and fast obstacles approaching along $y \approx 0$.

The method's formulation here assumes a capsule-shaped robot. However, any convex shape which a convex polygon and a sweeping circle generate together can replace the capsule. This only requires a routine to compute for a given obstacle the robot's corresponding subpart as the closest instance of the sweeping circle.

IV. Experiments in Simulation

The simulations in this section compare our method, RDS, to the baseline method (from Section II-B). The comparison in Section IV-C additionally includes the method "Blank", which is defined as outputting directly its input, such that the robot executes the nominal command, i.e. setting $\mathbf{u}^* = \bar{\mathbf{u}}$. With the method "Blank", only the nominal command and other agents contribute to cooperative navigation, allowing to estimate a compared reactive controller's additional contribution.

We define the control point for all methods and the reference point for RDS to coincide (for simplicity), where the control point (whose velocity the baseline method acts on) additionally

Algorithm 1: In Each Simulation Update, Pedestrians Compute Their Velocities Via ORCA, and the Robot Reacts Via RDS.

```
for $i = 1, ..., N_p$ do
    $\mathbf{v}^p_{i,t} = ORCA(\bar{\mathbf{v}}^p_{i,t}, \mathbf{x}^p_{i,t}, \{\mathbf{x}_{j,t}, \mathbf{v}_{j,t-\Delta t}\}^{N_p+N_c}_{j=1\neq i})$
end for
$\mathbf{u}^r_t = RDS(\bar{\mathbf{u}}^r_t, \mathbf{u}^r_{t-\Delta t}, \mathbf{x}^r_t, \varphi^r_t, \{\mathbf{x}^p_{j,t}, \mathbf{v}^p_{j,t}\}^{N_p}_{j=1})$
$\mathbf{x}^p_{i,t+\Delta t} = \mathbf{x}^p_{i,t} + \Delta t\, \mathbf{v}^p_{i,t}$
$(\mathbf{x}^r_{t+\Delta t}, \varphi^r_{t+\Delta t}) =$
    $(\mathbf{x}^r_t + \Delta t\, \mathbf{v}^r(\mathbf{u}^r_t, \varphi^r_t), \varphi^r_t + \Delta t\, \omega^r(\mathbf{u}^r_t))$
```

represents the robot's position for tracking the reference trajectory given by the experiment. We choose the robot's parameters as $x_{ref} = 0$, $y_{rear} = -0.5$ m, $y_{front} = y_{ref} = 0.18$ m, $r = 0.45$m (corresponding to Qolo's conservative capsule in Fig. 2), and $\tau = 1.5$ s, $\hat{a} = 2$ m/s ^{2}, $\hat{\alpha} = 3$ rad/s^2.

The following Section IV-A introduces the simulation framework and its specific performance metrics. A qualitative comparison follows in Section IV-B. A test series for quantitative comparison follows in Section IV-C, whose source code is available in the provided repository.

A. Simulation Framework and Performance Metrics

This section describes our framework for simulating how the robot navigates in environments with pedestrians. We represent them by circular agents (of radius 0.3m) which track individual reference trajectories while avoiding collisions with each other and the robot by applying ORCA (assuming reciprocity, and $\tau = 1.5$ s). The robot also tracks a reference trajectory and in turn uses the method RDS (or baseline) to avoid collisions with the pedestrians, reacting to their current position and updated velocities (or not, with the method "Blank"). The simulation's update scheme (for the case with RDS) is given in Algorithm 1, with $\Delta t = 0.05$ s being its time step and also the cycle time for RDS and other agents' controllers. The following paragraphs introduce the notation and explain the simulation framework.

For each pedestrian i, let $\mathbf{x}^p_{i,t}$, $\mathbf{v}^p_{i,t} \in \mathbb{R}^2$ denote respectively the global position and velocity at time t. Let $\mathbf{x}^r_t \in \mathbb{R}^2$ and φ^r_t denote respectively the robot's position (i.e. where its control point is) and orientation at time t. Accordingly, let $\mathbf{v}^r_t \in \mathbb{R}^2$ and ω^r_t denote respectively the robot's global cartesian velocity (of its control point) and angular velocity at time t. Let $\mathbf{u}^r_t$ denote the robot's command vector $[v^*, \omega^*]^T$ at time t, whose components are respectively the forward and angular velocity which result from the method for collision avoidance. They prescribe the robot's global velocities, which are thus functions $\mathbf{v}^r_t = \mathbf{v}^r(\mathbf{u}^r_t, \varphi^r_t)$ and $\omega^r_t = \omega^r(\mathbf{u}^r_t)$.

Pedestrians perceive the robot as a collection of N_c virtual circular agents, which are attached to the robot, covering its actual capsule. When simulating with the baseline method for collision avoidance, the collection includes the enlarged bounding circle, otherwise it contains only several tightly fitting circles. The virtual agents adopt the position and velocity of their respective point of attachment on the robot. Let $\mathbf{x}_{j,t}, \mathbf{v}_{j,t} \in \mathbb{R}^2$ denote the position and velocity for a generic circular agent (i.e. a pedestrian or a virtual agent).

The robot's reference trajectory $\bar{\mathbf{x}}^r_t : \mathbb{R} \to \mathbb{R}^2$ defines the robot's reference position at each time t and prescribes the nominal velocity $\bar{\mathbf{v}}^r_t$ for the robot's control point according to

$$\bar{\mathbf{v}}^r_t = \frac{\mathrm{d}\bar{\mathbf{x}}^r_t}{\mathrm{d}t} + k\,(\bar{\mathbf{x}}^r_t - \mathbf{x}^r_t)\,. \qquad (7)$$

Therein, the reference trajectory's derivative forms a feedforward term and the tracking error is added as a feedback term (with the gain $k > 0$). For pedestrians, the reference trajectories $\bar{\mathbf{x}}^p_{i,t} : \mathbb{R} \to \mathbb{R}^2$ prescribe the corresponding nominal velocities $\bar{\mathbf{v}}^p_{i,t}$ in analogy to (7). Both terms in (7) together achieve a vanishing tracking error over time such that agents converge to their (moving) reference position even after perturbations. A set $\{\bar{\mathbf{x}}^r_t, \bar{\mathbf{x}}^p_{1,t}, ..., \bar{\mathbf{x}}^p_{N_p,t}\}$ which contains the robot's and N_p pedestrians' reference trajectories over a time window $[t_1, t_2]$ defines a particular *simulation configuration*.

Our metrics rely on the following definitions. The static area of evaluation A defines where relevant interactions are expected (Section IV-C). Let the indicator function $\mathbb{I}_{\{B\}}$ equal 1 if B is true or 0 otherwise. Further, let $\langle\cdot\rangle = \int_{t_1}^{t_2}(\cdot)\mathrm{d}t/(t_2 - t_1)$ and $\langle\langle\cdot\rangle\rangle = \sum_{i=1}^{N_p}\langle\cdot\rangle/N_p$ denote respectively averaging over the sample's time window and averaging over both the time window and the crowd. We use the following metrics.

- The robot's mean tracking error $E_r = \langle|\bar{\mathbf{x}}^r_t - \mathbf{x}^r_t|\rangle$.
- The pedestrians' mean tracking error $E_p = \langle\langle|\bar{\mathbf{x}}^p_{i,t} - \mathbf{x}^p_{i,t}|w_{i,t}\rangle\rangle$, with $w_{i,t} \propto \mathbb{I}_{\{\bar{\mathbf{x}}^p_{i,t}\in A\}}$ and $\langle\langle w_{i,t}\rangle\rangle = 1$.
- The crowd's velocity reduction due to the robot $V_c = V_{c,0}/V_{c,r}$, where $V_{c,r} = \langle\langle|\mathbf{v}^p_{i,t}|\hat{w}_i\rangle\rangle$ denotes the crowd's weighted average velocity for the case with the robot, and $V_{c,0}$ denotes the analogous quantity for the case without a robot. Herein, the pedestrians' weights $\hat{w}_i \propto \langle\mathbb{I}_{\{\mathbf{x}^p_{i,t}\in A\}}\rangle$ are proportional to their actual time in A and normalized as $\langle\langle\hat{w}_i\rangle\rangle = 1$.
- The neighbours-to-crowd velocity ratio $V_n = V_{n,r}/V_{c,r}$, where $V_{n,r} = \langle\langle|\mathbf{v}^p_{i,t}|\breve{w}_i\rangle\rangle$ and $\breve{w}_i \propto \langle\mathbb{I}_{\{|\mathbf{x}^p_{i,t}-\mathbf{x}^r_t|<D\}}\rangle$ are weights proportional to pedestrians' time D-close to the robot ($D = 3$ m) and normalized as $\langle\langle\breve{w}_i\rangle\rangle = 1$.
- C_r counting collisions with the robot's capsule.

The metric V_c compares the crowd's speed when the robot is not present (leaving its place to a regular agent instead) to the crowd's speed when the robot is present. Thus, evaluating V_c for a given simulation configuration and method (e.g. RDS) requires to execute one simulation without and one with the robot. The metric V_n compares the speed of the robot's neighbours to the entire crowd's speed. If the robot tends to slow down pedestrians, we expect that $V_c > 1$ and $V_n < 1$.

B. Crossing With Variable Head Start

The following experiment lets the robot and a pedestrian cross while both contribute to collision avoidance according to our simulation framework (Section IV-A). Their reference

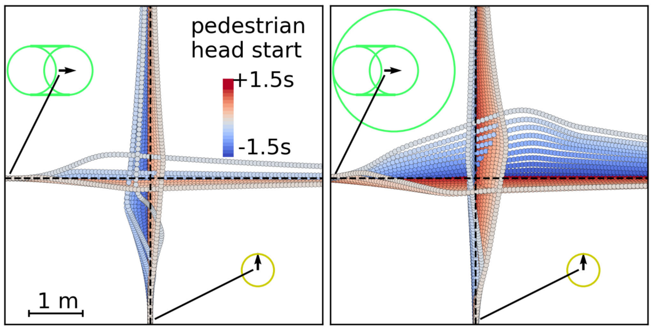

Fig. 6. The robot (moving to the right) and a pedestrian (moving upwards) cross with variable relative head starts, resulting in the corresponding (color-coded) trajectories and the particular crossing order. The robot uses either RDS (left) or the baseline method (right) for collision avoidance.

trajectories move at the same speed 1.3 m/s and their paths cross orthogonally, however, the pedestrian starts from a variable distance to the crossing point. The pedestrian's head start t_{hs}^p denotes the time difference between the moment when the pedestrian's reference trajectory reaches the crossing point and the moment when the robot's reference trajectory reaches it. Fig. 6 shows both agent's trajectories that result respectively for different values of the pedestrian's head start in the range $t_{hs}^p \pm 1.5$ s. The result shows for both methods how the crossing order changes around $t_{hs}^p = 0$. Over the test series, the pedestrian's mean tracking error is similar with both methods to control the robot ($E_p^{rds} = 0.10 \pm 0.12$ m, $E_p^{b.l.} = 0.09 \pm 0.06$ m), whereas the robot's tracking error is clearly lower with RDS ($E_r^{rds} = 0.20 \pm 0.05$ m, $E_r^{b.l.} = 0.35 \pm 0.14$ m).

C. Navigating in a Sparse Crowd

For quantitative comparison, this experiment series evaluates RDS, the baseline method, and the trivial method "Blank" (that adopts nominal commands) in simulations that are driven by original crowd movements (from a pedestrian intersection on a campus) which are available in the "Crowds-by-Example" dataset [25] as timed waypoint sequences. We use these original trajectories to generate 430 different sample simulation configurations by replacing a different original pedestrian by the robot and simulating the remaining original pedestrians via regular agents. For each agent (including the robot), we define its reference trajectory as the two cubic splines fitting the respective original pedestrian's 2D-waypoints over time. The area of evaluation A is chosen as a tight bounding box of all the waypoints. The time window $[t_1, t_2]$ for a given sample configuration's simulations matches the time window of the waypoints for the robot's original pedestrian. Other agents' trajectories outside their original waypoints' time windows are linear extrapolations (which are mostly outside A). For each sample configuration, we evaluate the metrics from Section IV-A for each method (simulating once without a robot). Fig. 7 shows for an exemplary sample configuration the robot's and the surrounding crowd's motion during sequential time windows for RDS and the baseline method, respectively. These motion snippets exemplify how the robot can often follow closely its nominal motion with RDS,

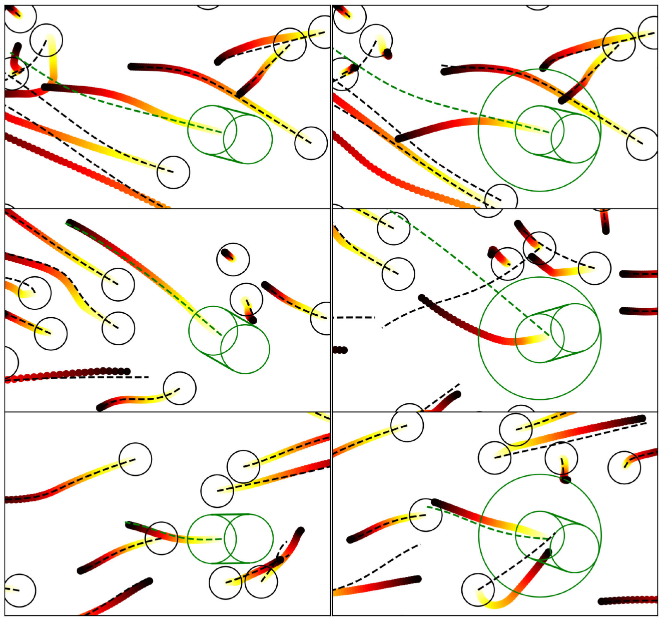

Fig. 7. The robot uses RDS (left) or the baseline method (right) to traverse the dynamic crowd in this empirically based simulation example. For three sequential time windows, the initial state of the robot (green capsule) and crowd (black circles) and their future motion (yellow to red) and future nominal motion (dashed lines) are shown.

TABLE I
THE METRICS' MEAN AND STANDARD DEVIATION (OR FOR C_r, THE SUM) IS SHOWN OVER THE SPARSE CROWD SIMULATIONS FOR THE THREE METHODS. BETWEEN THE BASELINE METHOD AND RDS, SUPERIOR MEAN VALUES ARE MARKED IN BOLD AND SIGNIFICANT DIFFERENCES BY ASTERISKS (EXCEPT FOR C_r)

Method	E_r [m]	E_p [m]	V_c [-]	V_n [-]	ΣC_r [-]
RDS	**0.8***±0.9	**0.20***±0.09	0.999*±0.007	1.07±0.24	0
Baseline	2.2*±2.0	0.21*±0.09	**0.994***±0.010	**1.08**±0.25	0
"Blank"	0.0±0.0	0.20±0.09	0.996±0.009	1.07±0.23	6

whereas the baseline method leads it on detours around dense groups.

Table I reports the metrics' sample averages and standard deviations over the 430 sample configurations for the three methods (or for C_r, the sum over all configurations). For all the metrics except C_r we compare their distributions for RDS against the baseline using a two-sample t-test with a significance level $\alpha = 0.05$. We find $p < \alpha$, i.e. significant differences in the mean values, for all the metrics except V_n.

In comparison to the baseline method, we attribute RDS' significantly lower tracking error for the robot and the pedestrians (i.e. $E_r^{rds} < E_r^{b.l}$, $E_p^{rds} < E_p^{b.l}$) to the tighter shape representation. It allows the robot to maneuver through narrow gaps between pedestrians and requires less deviations from them. On the other hand, the circular shape representation with the baseline method encourages agents to maneuver around the robot with increased velocity (as typical for ORCA), whereas the multi-circle shape representation they perceive for the robot with RDS often traps them between two such circles, thus $V_n^{b.l.} > V_n^{rds}$. While the assumption that other agents perceive the enlarged bounding circle for the robot with the baseline method is not realistic, we observe that it is actually favourable for the baseline method's performance, since otherwise the robot frequently experiences virtual collisions (with its bounding circle), and while trying to resolve them, it is prone to colliding truly, as it does not represent the robot's true capsule shape.

With RDS or the baseline method, collisions do not occur for the robot throughout the simulations. This is due to contributions from both the robot's reactive controller and the pedestrians' cooperative controller. The robot's contribution is still necessary sometimes to avoid collisions, as the method "Blank" (which does not contribute) leads to a few collisions ($\Sigma C_r > 0$).

Comparing E_p, V_c, and V_n between RDS and the method "Blank", we find that RDS does not facilitate other agents to follow their references and generally reduces the crowd's velocity in our simulation framework. However, also with the method "Blank" the robot already receives high-level guidance via the reference trajectory (which avoids other agents original positions) and therefore, this result mainly shows that the smaller holonomic pedestrians can resolve efficiently the slight collisions due to the robotic agent's larger shape even when the robot does not contribute.

In summary, these results show that RDS successfully corrects the robot's motion to account for its capsule shape (whose potential collisions the reference motion does not avoid) while at the same time achieving a low tracking error for the robot and the crowd.

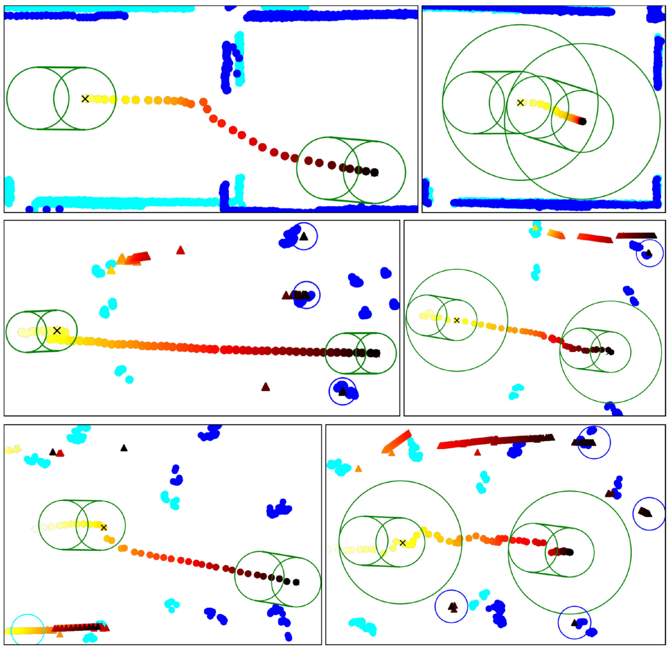

Fig. 8. The robot Qolo uses RDS (left) or the baseline method (right) to pass through a door (top), overtake three pedestrians (middle) or a surrounding crowd (bottom). Its trajectory and the tracker's estimates of persons are shown (blobs and triangles, respectively, encoding time in yellow-red). Also, the robot's footprint (green capsule), tracked persons' footprints (circles) and LiDAR scanpoints (blobs) are shown at the beginning (in cyan) and at the end (in blue).

V. Experiments With the Robot Qolo

The robot Qolo [26] is an electrically powered standing wheelchair (in Fig. 1, 2, 3). In this section's experiments, Qolo is driven by RDS which receives a constant forward-pointing command (i.e. with vanishing nominal angular velocity) simulating a driver's primitive input. RDS uses the same parameters' values given already for the experiments in simulation (Section IV), except for $\hat{a} = 1.5\ \text{m/s}^2$, $\hat{\alpha} = 1.5\ \text{rad/s}^2$. The experiments' videos are available in the letter's supplementary material.

A. Implementation

The robot's sensors include a front and a rear LiDAR and a front RGB-D camera. They inform the modules for SLAM, person detection tracking and collision avoidance.

1) SLAM: The robot estimates its own trajectory by matching scans from the rear LiDAR using the ROS package hector_slam [30], which allows the tracker to transform the sensors' spatial data into a static fixed reference frame and to estimate obstacles' absolute velocities.

2) Person Detection Tracking: The module tracks persons' positions from both LiDARs and the camera's RGB images (using the pipeline in [31]).

3) Collision Avoidance: The module implements RDS or the baseline method (as in Section IV). It treats every scanpoint from both LiDARs as a separate circular obstacle with a small radius and zero velocity. Further, every person track is perceived as a circular obstacle with 0.3 m radius and with the track's estimated velocity.

B. Test in a Static Environment

The test (in Fig. 8, top) compares how RDS and the baseline method can assist passing through a door. As the nominal command would drive the robot forward into the door frame, correction is necessary to avoid the collision and ideally lead through the door. The trajectories for the baseline method and RDS in Fig. 8 right and left, respectively, show that among both methods, only RDS can lead the robot through the door due to the tighter capsule shape representation.

C. Tests With Pedestrians

The following two tests evaluate the robot's ability to overtake pedestrians that move in the same direction but are distributed ahead of and around the robot.

1) Row of Pedestrians: The experiment (in Fig. 8, middle row) involves three pedestrians walking next to each other, forming a line with a larger gap between two of them such that the robot could pass in between while respecting a comfortable distance.

With the baseline method (Fig. 8, right), the robot approaches the moving pedestrians and then alternates between different angles while attempting to pass through the gap, which is due to the fact that the perceived orientation of the gap oscillates as the foot patterns and relative advancement of the pedestrians vary slightly over time. Using RDS (Fig. 8, left), the robot adjusts its orientation early towards the gap in order to avoid colliding with the middle pedestrian, and then it moves straight forward and passes through the gap.

2) Unidirectional Crowd: In the experiment (in Fig. 8, bottom), there are five pedestrians surrounding the robot. With the baseline method, the robot's motion is heavily constrained and it moves always towards small free areas created randomly by small irregularities in the crowd motion. With RDS, the robot adjusts its orientation early to avoid colliding with one pedestrian ahead, and subsequently it converges to a collision-free course and overtakes the surrounding crowd.

VI. Conclusion

We have developed a method to apply the Velocity Obstacle (VO) to non-holonomic capsule-shaped robots and highlighted its effectiveness at avoiding collisions with static obstacles and interacting pedestrians, both in simulation and physical experiments with the robot Qolo. The comparison with another method using VO has demonstrated our method's advantage due to allowing maneuvers through narrow gaps. Our simulations of agents tracking real crowds' motions show that the method avoids collisions efficiently such that the robot and pedestrians remain close to their references. We have described and evaluated four novel metrics to support this analysis.

Acknowledgment

We thank Dan Jia and Sabarinath Mahadevan for providing help and supporting us with the person tracker. We thank Fabien Grzeskowiak and Julien Pettré for proposing the metrics that quantify the robot's impact on the crowd's speed.

References

[1] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *Int. J. Robot. Res.*, vol. 17, no. 7, pp. 760–772, Jul. 1998.

[2] J. van den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," *Robotics Research, (Springer Tracts in Advanced Robotics*, C. Pradalier, R. Siegwart, and G. Hirzinger, Eds., vol. 70. Berlin, Heidelberg, Germany: Springer, 2011, pp. 3–19.

[3] L. Huber, A. Billard, and J.-J. Slotine, "Avoidance of convex and concave obstacles with convergence ensured through contraction," *IEEE Robot. Automat. Lett.*, vol. 4, no. 2, pp. 1462–1469, Jan. 2019.

[4] A. V. Savkin and C. Wang, "A simple biologically inspired algorithm for collision-free navigation of a unicycle-like robot in dynamic environments with moving obstacles," *Robotica*, vol. 31, no. 6, pp. 993–1001, May 2013.

[5] Y. F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Vancouver, BC, Canada, Sep. 2017, pp. 1343–1350.

[6] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Automat.*, Montreal, QC, Canada, May 2019, pp. 6015–6022.

[7] M. Kuderer, H. Kretzschmar, C. Sprunk, and W. Burgard, "Feature-based prediction of trajectories for socially compliant navigation," in *Proc. Robot.: Sci. Syst.*, Sydney, Australia, Jul. 2012.

[8] D. Vasquez, B. Okal, and K. O. Arras, "Inverse reinforcement learning algorithms and features for robot navigation in crowds: An experimental comparison," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Chicago, IL, USA, Sep. 2014, pp. 1341–1346.

[9] A. Vemula, K. Muelling, and J. Oh, "Social attention: Modeling attention in human crowds," in *Proc. IEEE Int. Conf. Robot. Automat.*, Brisbane, QLD, Australia, May 2018, pp. 4601–4607.

[10] Y. Luo, P. Cai, A. Bera, D. Hsu, W. S. Lee, and D. Manocha, "Porca: Modeling and planning for autonomous driving among many pedestrians," *IEEE Robot. Automat. Lett.*, vol. 3, no. 4, pp. 3418–3425, Jul. 2018.

[11] S. Bouraine, T. Fraichard, and H. Salhi, "Provably safe navigation for mobile robots with limited field-of-views in dynamic environments," *Auton. Robots*, vol. 32, no. 3, pp. 267–283, Nov. 2012.

[12] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robot. Automat. Mag.*, vol. 4, no. 1, pp. 23–33, Mar. 1997.

[13] C. Schlegel, "Fast local obstacle avoidance under kinematic and dynamic constraints for a mobile robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, vol. 1, Victoria, BC, Canada, Oct. 1998, pp. 594–599.

[14] I. Ulrich and J. Borenstein, "VFH: Reliable obstacle avoidance for fast mobile robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, vol. 2, Leuven, Belgium, May 1998, pp. 1572–1577.

[15] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Taipei, Taiwan, Oct. 2010, pp. 797–803.

[16] J. Alonso-Mora, A. Breitenmoser, M. Rufli, P. Beardsley, and R. Siegwart, "Optimal reciprocal collision avoidance for multiple non-holonomic robots," *Distributed Autonomous Robotic Systems*, (Springer Tracts in Advanced Robotics (STAR)), A. Martinoli, F. Mondada, N. Correll, G. Mermoud, M. Egerstedt, M. A. Hsieh, L. E. Parker, and K. Støy, Eds., vol. 83. Berlin, Heidelberg, Germany: Springer, 2013, pp. 203–216.

[17] D. Wilkie, J. van denBerg, and D. Manocha, "Generalized velocity obstacles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, St Louis, MO, USA, Dec. 2009, pp. 5573–5578.

[18] A. Best, S. Narang, and D. Manocha, "Real-time reciprocal collision avoidance with elliptical agents," in *Proc. IEEE Int. Conf. Robot. Automat.*, Stockholm, Sweden, May 2016, pp. 298–305.

[19] Y. Ma, D. Manocha, and W. Wang, "Efficient reciprocal collision avoidance between heterogeneous agents using CTMAT," in *Proc. 17th Int. Conf. Auton. Agents MultiAgent Syst.* Stockholm, Sweden: Int. Foundation for Auton. Agents Multiagent Syst., Jul. 2018, pp. 1044–1052.

[20] J. Van Den Berg, J. Snape, S. J. Guy, and D. Manocha, "Reciprocal collision avoidance with acceleration-velocity obstacles," in *Proc. IEEE Int. Conf. Robot. Automat.*, Shanghai, China, May 2011, pp. 3475–3482.

[21] T. Fraichard and V. Levesy, "From crowd simulation to robot navigation in crowds," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 729–735, Jan. 2020.

[22] H.-T. (Lewis) Chiang, B. HomChaudhuri, L. Smith, and L. Tapia, "Safety, challenges, and performance of motion planners in dynamic environments," *Robotics Research, (Springer Proceedings in Advanced Robotics (SPAR))*, N. M. Amato, G. Hager, S. Thomas, and M. Torres-Torriti, Eds., vol. 10. Cham: Springer, 2020, pp. 793–808.

[23] P. Trautman, J. Ma, R. M. Murray, and A. Krause, "Robot navigation in dense human crowds: Statistical models and experimental studies of human-robot cooperation," *Int. J. Robot. Res.*, vol. 34, no. 3, pp. 335–356, Feb. 2015.

[24] P. Bevilacqua, M. Frego, D. Fontanelli, and L. Palopoli, "Reactive planning for assistive robots," *IEEE Robot. Automat. Lett.*, vol. 3, no. 2, pp. 1276–1283, Jan. 2018.

[25] A. Lerner, Y. Chrysanthou, and D. Lischinski, "Crowds-by-Example data set," Accessed: Apr. 7, 2021. [Online]. Available: https://github.com/epfl-lasa/rds/tree/crowdbot/crowds_by_example_dataset, 2007

[26] D. Paez Granados, H. Kadone, and K. Suzuki, "Unpowered lower-body exoskeleton with torso lifting mechanism for supporting sit-to-stand transitions," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, Madrid, Spain, Oct. 2018, pp. 2755–2761.

[27] J. Snape, J. Van Den Berg, S. J. Guy, and D. Manocha, "Smooth and collision-free navigation for multiple robots under differential-drive constraints," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Taipei, Taiwan, Oct. 2010, pp. 4584–4589.

[28] R. Seidel, "Small-dimensional linear programming and convex hulls made easy," *Discrete Comput. Geometry*, vol. 6, no. 3, pp. 423–434, Dec. 1991.

[29] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry: Algorithms and Applications*. Berlin, Heidelberg: Springer, 2008, ch. 4, pp. 71–82.

[30] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf, "A flexible and scalable slam system with full 3d motion estimation," in *Proc. IEEE Int. Symp. Saf., Secur. Rescue Robot.*, Kyoto, Japan, Nov. 2011, pp. 155–160.

[31] S. Breuers, L. Beyer, U. Rafi, and B. Leibe, "Detection- Tracking for Efficient Person Analysis: The DetTA Pipeline," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Madrid, Spain, Oct. 2018, pp. 48–53.