Independent Navigation of Multiple Mobile Robots with Hybrid Reciprocal Velocity Obstacles

Jamie Snape, *Student Member, IEEE*, Jur van den Berg, Stephen J. Guy, and Dinesh Manocha

***Abstract*— We present an approach for smooth and collision-free navigation of multiple mobile robots amongst each other. Each robot senses its surroundings and acts *independently* without central coordination or communication with other robots. Our approach uses both the current position and the velocity of other robots to predict their future trajectory in order to avoid collisions. Moreover, our approach is *reciprocal* and avoids oscillations by explicitly taking into account that the other robots also sense their surroundings and change their trajectories accordingly. We build on prior work related to velocity obstacles and reciprocal velocity obstacles and introduce the concept of *hybrid reciprocal velocity obstacles* for collision avoidance that takes into account the kinematics of the robots and uncertainty in sensor data. We apply our approach to a set of *iRobot Create* robots using centralized sensing and show natural, direct, and collision-free navigation in several challenging scenarios.**

I. Introduction

As mobile robots become ubiquitous in everyday life, the need arises for smooth and collision-free navigation of these robots amongst each other. Each robot should sense its surroundings and act *independently* without communication with other robots or centralized coordination, just as humans do when they move amongst each other.

Many works in robotics have addressed the problem of collision-free navigation of a robot in dynamic environments with moving obstacles [1], [2], [3], [4], [5]. Most approaches predict where the moving obstacles might be in the future by extrapolating their current velocities, and let the robot avoid collisions accordingly. However, such an approach does not suffice when the robot encounters other robots, because treating the other robots as (passively) moving obstacles overlooks the *reciprocity* between robots, i.e. the fact that the other robots react to you in the same way as you react to them. Hence, the future trajectories of other robots cannot be estimated by extrapolating their current velocities, and doing so could inherently cause undesirable *oscillations* in the motion of the robots [6], [7].

In this paper, we introduce the concept of *hybrid reciprocal velocity obstacles* for collision avoidance amongst robots that specifically considers this reciprocity. Informally speaking, reciprocity lets a robot take half of the responsibility of avoiding collisions with another robot and assumes that the other robot takes care of the other half. Each robot executes an independent continuous cycle of sensing and acting, in which a robot chooses its new velocity based on observations of the positions and velocities of the other robots. In principle, our method allows each robot to observe its own surroundings, however, for ease of implementation, we use centralized sensing. The robots do not communicate with each other, but implicitly assume that the other robots use the same collision-avoidance strategy. Our overall approach can also deal with moving obstacles and we make no assumptions on their motion.

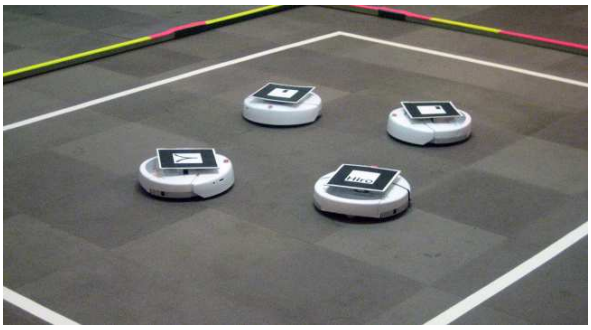


Fig. 1. Four *iRobot Create* mobile robots in our experimentation setting.

Our formulation is an extension of the concept of *reciprocal velocity obstacles* [6] that was introduced to address similar issues in multi-agent and crowd simulation. However, this formulation has some limitations, particularly that it frequently causes agents to end up in a "reciprocal dance" as they cannot find agreement on which side to pass each other. To overcome this limitation, our formulation combines the reciprocal velocity obstacle with the original velocity obstacle concept [1]. In addition, our approach takes into account both the *kinematics* of a robot and *sensor uncertainty*, which makes it specifically suitable for navigation of mobile robots.

We have implemented our approach and applied it to a set of *iRobot Create* robots moving in an indoor environment (see Fig. 1). The robots are controlled through a Bluetooth connection, and use localization data obtained from an overhead camera. Our experiments with several challenging scenarios show that our approach achieves natural, direct, and collision-free navigation even when sensor uncertainty and control signal latency are artificially increased.

The rest of this paper is organized as follows. We begin by summarizing related work in Section II. In Section III, we introduce our formulation of hybrid reciprocal velocity obstacles that we use for collision avoidance. In Section IV, we use this formulation for navigation of multiple mobile robots and take into account the kinematics of the robots as

This work was supported in part by ARO contracts DAAD19-02-1-0390 and W911NF-04-1-0088, NSF awards 0400134, 0429583, and 0404088, DARPA/RDECOM contract N61339-04-C-0043, and Intel.

The authors are with the Department of Computer Science, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599, USA. Email: {snape, berg, sjguy, dm}@cs.unc.edu. Website (with videos): http://gamma.cs.unc.edu/HRVO/.

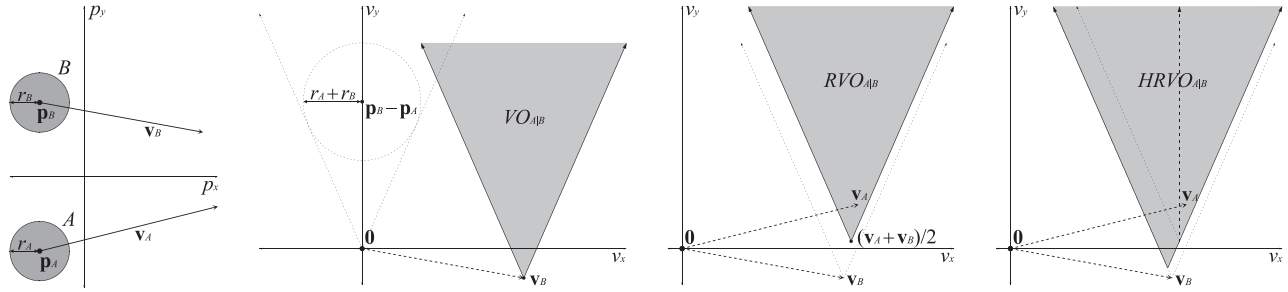


Fig. 2. (a) A configuration of two disc-shaped robots A and B in the plane with radii r_A and r_B, positions $\mathbf{p}_A$ and $\mathbf{p}_B$, and velocities $\mathbf{v}_A$ and $\mathbf{v}_B$, respectively. (b) The velocity obstacle $VO_{A|B}$ for A induced by B. (c) The reciprocal velocity obstacle $RVO_{A|B}$ for A induced by B. (d) The hybrid reciprocal velocity obstacle $HRVO_{A|B}$ for A induced by B. Note that $\mathbf{v}_A$ is right of the centerline (dashed) of $RVO_{A|B}$, so the apex of $HRVO_{A|B}$ is the intersection point of the right side of $RVO_{A|B}$ and the left side of $VO_{A|B}$.

well as sensor uncertainty. We discuss actual implementation and highlight the experimental results in Section V.

II. Previous Work

In this section, we give a brief overview of prior work on navigating in dynamic environments amongst moving obstacles, multi-robot planning, and existing variations of the velocity obstacle concept.

Previous work has addressed the issue of a single robot navigating amongst multiple (passive) moving obstacles [2], [3], [4], [5]. A particularly successful concept is velocity obstacles [1], [8], which has been used in practice for real-world robots, for example, navigating robotic wheelchairs through crowded public environments [9], automated cars [10], and mission planning for unmanned aerial vehicles (UAVs) [11].

Several variations of the velocity obstacle formulation have been proposed for multi-robot systems, generally by attempting to incorporate the reactive behavior of the other entities in the environment. Variations such as reciprocal velocity obstacles [6], [12], recursive probabilistic velocity obstacles [13], [14], and common velocity obstacles [15] use various means to handle reciprocity, but each have their own shortcomings. Specifically, the approach of [14] may fail to converge, while other concepts [6], [15] are limited to dealing with only two robots.

Other work has focused mainly on follow-the-leader behavior when navigating robots in real-world settings [16], [17]. Also, there is a large body of work on centrally coordinating the motions of multiple robots [18]. However, to the best of our knowledge, there is little work on navigation of multiple independent robots to arbitrary goals in real-world settings while taking into account the reactive behavior of other robots.

III. Collision Avoidance

In this section, we describe how robots avoid collisions with each other. We briefly review the concepts of *velocity obstacles* [1] and *reciprocal velocity obstacles* [6], and then introduce our formulation of *hybrid reciprocal velocity obstacles* that we use for multi-robot navigation.

A. Velocity Obstacles

The concept of velocity obstacles was introduced for navigating a robot amongst moving obstacles [1]. It is defined as follows. Let A be a robot and let B be an obstacle moving in the plane $\mathbb{R}^2$. For simplicity, we assume A and B are discs with radii r_A and r_B, respectively. Let $\mathbf{p}_A$ and $\mathbf{p}_B$ denote the current positions of the centers of A and B, respectively, and let $\mathbf{v}_B$ be the velocity of B (see Fig. 2(a)). Then, the velocity obstacle for A induced by B, denoted $VO_{A|B}$, is the set of all velocities of A that will result in a collision between A and B at some moment in time, assuming that B maintains its velocity $\mathbf{v}_B$. More formally, let $D(\mathbf{p}, r)$ denote an open disc of radius r centered at $\mathbf{p}$, then

$$VO_{A|B} = \{\mathbf{v} \mid \exists t > 0 :: t(\mathbf{v} - \mathbf{v}_B) \in D(\mathbf{p}_B - \mathbf{p}_A, r_A + r_B)\}.$$

It follows that if A chooses a velocity inside $VO_{A|B}$, then A and B will collide at some point in time. If the velocity chosen is outside $VO_{A|B}$, such a collision will never occur. Fig. 2(b) shows a geometric interpretation of velocity obstacles.

The principle of velocity obstacles has been successfully applied to navigate one robot in the presence of moving obstacles by having the robot select a velocity in each time step that is outside any of the velocity obstacles induced by the moving obstacles [1], [9], [13]. Unfortunately, the velocity obstacle concept cannot be applied to multi-robot navigation, since it fails to take into account that the other robots are not (passive) moving obstacles, but active decision-making entities that similarly adapt their velocities to their surroundings. If all robots were to use the velocity obstacle concept to choose a new velocity, this would inherently result in *oscillations*. This is because if any pair of robots select new velocities outside each other's velocity obstacles, then their old velocities will be valid with respect to the velocity obstacles based on the new velocities. Hence, the robots will oscillate between these two velocities, as shown in [6].

B. Reciprocal Velocity Obstacles

The reciprocal velocity obstacle formulation introduced in [6] addresses the problem of oscillations by incorporating the reaction of other robots. Instead of having to take 100% of the responsibility for avoiding collisions, as is the case

with velocity obstacles, the reciprocal approach lets a robot take only 50% of the responsibility of avoiding a collision, while assuming the other robot involved takes care of the other half. More precisely, when selecting the new velocity of robot A, the average is taken of its current velocity, $\mathbf{v}_A$, and a velocity outside the velocity obstacle $VO_{A|B}$ induced by the other robot B. The reciprocal velocity obstacle $RVO_{A|B}$ for robot A induced by B is defined accordingly as

$$RVO_{A|B} = \{\mathbf{v} \,|\, 2\mathbf{v} - \mathbf{v}_A \in VO_{A|B}\}.$$

Fig. 2(c) illustrates that the velocity obstacle is effectively translated such that its apex is at $(\mathbf{v}_A + \mathbf{v}_B)/2$.

The reciprocal velocity obstacle formulation gives the following theoretical guarantee: If each robot selects a velocity outside the reciprocal velocity obstacle induced by the other, *and* both robots choose to pass each other on the same side, i.e. either on the right side of each other, or on the left side, then the trajectories of both robots will be free of collisions and oscillations. If the free velocity is selected that is closest to the robot's current velocity, then the robots automatically pass each other on the same side. However, this only holds when there is only one other robot. In practice, each robot is also required to select the velocity closest to its *preferred* velocity (rather than its *current* velocity) to make progress towards its goal. Unfortunately, this means that robots may not necessarily choose the same side to pass, which may result in oscillations known as "reciprocal dances" [7]. Even though reciprocal dances are common in natural human motion, they are undesirable for multi-robot navigation as robots are unable to resolve them.

C. Hybrid Reciprocal Velocity Obstacles

To remedy this situation, we introduce the concept of the *hybrid reciprocal velocity obstacle*, as shown in Fig. 2(d). For two robots, A and B, if $\mathbf{v}_A$ is to the right of the centerline of $RVO_{A|B}$, which implies by symmetry that $\mathbf{v}_B$ is to the right of the centerline of $RVO_{B|A}$, we wish A to choose a velocity to the right of $RVO_{A|B}$. To encourage this, the reciprocal velocity obstacle is enlarged by replacing the edge on the side we do *not* wish the robots to pass (in this instance the left side) by the edge of the velocity obstacle $VO_{A|B}$. The apex of the resulting obstacle is then the point of intersection between the right side of $RVO_{A|B}$ and the left side of $VO_{A|B}$. If $\mathbf{v}_A$ is to the left of the centerline, we mirror the procedure, exchanging left and right. As a *hybrid* of a reciprocal velocity obstacle *and* a velocity obstacle, we call the result a *hybrid reciprocal velocity obstacle*, and denote it $HRVO_{A|B}$.

The hybrid formulation has the consequence that if robot A attempts to pass on the wrong side of robot B, perhaps because of the presence of other robots, then it has to give full priority to robot B, in accordance with the velocity obstacle concept. However, if it does choose the correct side, then it can assume the cooperation of robot B and retains equal priority, as for the reciprocal velocity obstacle concept. This distinction greatly reduces the amount of oscillations, while not overconstraining the motion of each robot.

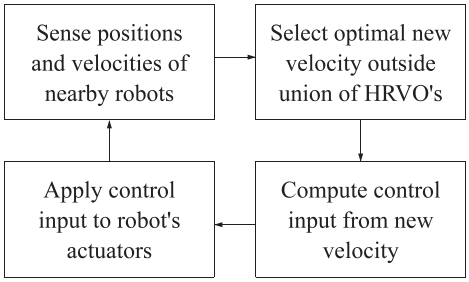


Fig. 3. Schematic overview of the continuous cycle of sensing and acting that each robot executes independently.

IV. Multi-Robot Navigation

In this section, we show how we apply the hybrid reciprocal velocity obstacle formulation to the navigation of multiple mobile robots. We begin by sketching the global approach, and then explain how to take into account the kinematics and sensor uncertainty of the robot.

A. Global Approach

Let there be a set of mobile robots sharing an environment. Each robot A has a current position $\mathbf{p}_A$, a current velocity $\mathbf{v}_A$, and a radius r_A. These parameters are part of the robot's *external* state, i.e. they can be sensed or measured by other robots. Furthermore, each robot has a preferred velocity $\mathbf{v}_A^{\text{pref}}$, which is the velocity directed towards its goal with a magnitude equal to its preferred speed, and is the velocity the robot would have selected had no other robots been in its way. We consider the preferred velocity part of the *internal* state of the robot, therefore the other robots cannot sense or measure it.

The overall approach is as follows. Each robot A independently performs a continuous cycle of sensing and acting (see Fig. 3 for a schematic overview). In the sensing phase, the robot acquires its own position and velocity, and those of the surrounding robots. Based on this information, the robot infers for each of its neighboring robots B the hybrid reciprocal velocity obstacle $HRVO_{A|B}$, and selects a *new* velocity $\mathbf{v}_A^{\text{new}}$ for itself that is closest to its preferred velocity $\mathbf{v}_A^{\text{pref}}$ amongst all velocities outside the union of the hybrid velocity obstacles induced by the neighboring robots:

$$\mathbf{v}_A^{\text{new}} = \underset{\mathbf{v} \notin \bigcup_{B \neq A} HRVO_{A|B}}{\arg\min} \|\mathbf{v} - \mathbf{v}_A^{\text{pref}}\|. \qquad (1)$$

We use the efficient geometric algorithm of [12] to find this velocity.

While the robot should assume this new velocity $\mathbf{v}_A^{\text{new}}$, this may not be directly possible due to its *kinematic* constraints. Therefore, the velocity $\mathbf{v}_A^{\text{new}}$ is transformed into a control input for the robot that will let the robot assume velocity $\mathbf{v}_A^{\text{new}}$ "as soon as possible." In Section IV-B, we detail how to do this for the case of a differential-drive robot.

After this, the robot starts a new cycle of sensing and acting, and continues doing so indefinitely. Note that the above process does not require the robots to communicate with each other. Robots only use information they can sense independently.

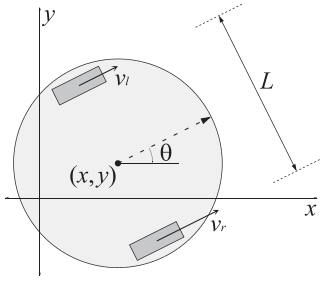


Fig. 4. The kinematic model of a differential-drive robot. Each wheel is attached to a separate motor and may assume a different speed. Note that the robot can spin in place and follow any continuous path.

B. Kinematics

We consider a simple differential-drive robot (see Fig. 4), whose kinematics are applicable to many contemporary mobile robotic systems. It has two main wheels, fixed in orientation, each attached to its own motor such they can be driven at different speeds.

The configuration of our differential-drive robot is given by its position (x, y) and its orientation θ. If the distance between the two wheels of the robot is L, and the left and right wheel speeds are v_l and v_r, respectively, then the configuration transition equations are

$$\dot{x} = \frac{v_l + v_r}{2} \cos\theta, \tag{2}$$

$$\dot{y} = \frac{v_l + v_r}{2} \sin\theta, \tag{3}$$

$$\dot{\theta} = \frac{v_r - v_l}{L}. \tag{4}$$

Furthermore, the wheel speeds are bounded to a given maximum $v_{\max}$, such that

$$-v_{\max} \leq v_l \leq v_{\max}, \quad -v_{\max} \leq v_r \leq v_{\max}. \tag{5}$$

The speeds of the wheels are the control input of the robot. When $v_l = v_r > 0$, the robot will move straight forwards; when $v_l > v_r > 0$, it will arc right; and when $v_l = -v_r \neq 0$, it will spin in place. In fact, the center of the robot is able to follow any continuous path within the environment [18].

For our approach, the task is to transform the velocity $\mathbf{v}_A^{\text{new}}$ as given by (1) to a pair of wheel speeds v_l and v_r, given the current orientation θ of the robot. Ignoring effects of inertia, we choose to set v_l and v_r such that the velocity $\mathbf{v}_A^{\text{new}}$ is obtained precisely after a preset amount of time τ to ensure smooth motion. More specifically, suppose that $\mathbf{v}_A^{\text{new}} = (v_x, v_y)$. Then, the target orientation is $\theta' = \arctan(v_y/v_x)$ and the target speed is $\|\mathbf{v}_A^{\text{new}}\|$. The difference between the target orientation and the current orientation is $\Delta\theta = \theta' - \theta$, such that $\Delta\theta \in [-\pi, \pi]$. To move from the current orientation θ to the target orientation θ' in τ time, it follows directly from (4) that

$$v_r - v_l = \frac{L\Delta\theta}{\tau}. \tag{6}$$

To obtain the target speed, it follows from (2) and (3) that

$$v_r + v_l = 2\|\mathbf{v}_A^{\text{new}}\|. \tag{7}$$

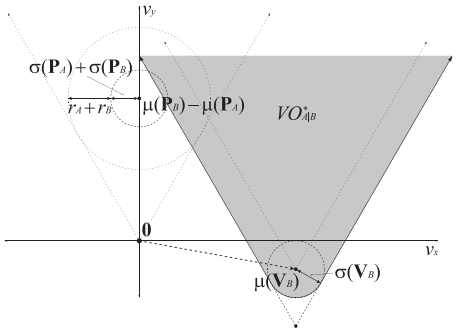


Fig. 5. The velocity obstacle $VO^*_{A|B}$ adjusted to account for uncertainty in positions $\mathbf{p}_A$ and $\mathbf{p}_B$ and uncertainty in velocity $\mathbf{v}_B$.

The desired values of v_l and v_r can now be solved from the system of equations formed by (6) and (7).

Unfortunately, the constraints of (5) may invalidate the computed values of v_l and v_r. In this case, we first attempt to move v_l and v_r into the interval $[-v_{\max}, v_{\max}]$ while keeping $v_r - v_l$ constant, such that the target orientation is obtained after τ time. If this also fails, in which case $|v_r - v_l| > 2v_{\max}$, then v_l and v_r are clamped to the extremes of the interval, such that the robot maximally rotates in place.

The choice of τ must be sufficiently small to allow the robot to quickly react to other robots in its path. However, if set too low, i.e. lower than the duration Δt of each sensing/acting cycle, the robot overshoots its target orientation, leading to oscillations. In addition, a low value of τ may result in less smooth paths, since the robot may frequently have to rotate in place to achieve its target orientation. Our experiments indicated that a value of $\tau = 3\Delta t$ gives good results.

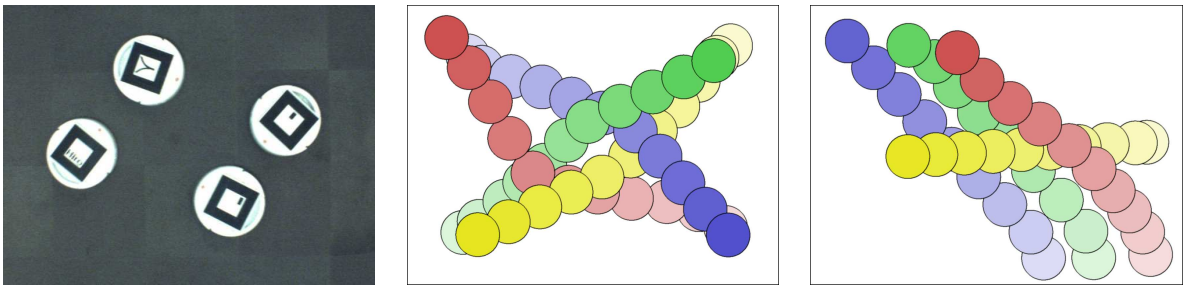
C. Sensor Uncertainty

To calculate the hybrid reciprocal velocity obstacles, each robot requires the current position and velocity of every robot. Because this data is obtained using sensors, it inevitably contains uncertainty. This may jeopardize the correct functioning of our approach.

To mitigate the uncertainty effects, we use a Kalman filter [19] to obtain accurate estimates of the positions and velocities of the robots. In addition, the Kalman filter provides an estimate of the variance, and hence the standard deviation, of the measured quantities. We *explicitly* model this in our hybrid reciprocal velocity obstacles as follows.

Let $\mathbf{P} = (P_x, P_y)$ denote a bivariate normal distribution of a measured position $\mathbf{p}$ with mean $\mu(\mathbf{P}) = (\mu(P_x), \mu(P_y))$. For simplicity, we assume that there is the same measurement uncertainty in the x-direction as in the y-direction, such that the standard deviations $\sigma(P_x)$ and $\sigma(P_y)$ are equal. We informally refer to this value as $\sigma(\mathbf{P}) = \sigma(P_x) = \sigma(P_y)$.

Now, we explicitly take into account one standard deviation in the construction of the hybrid reciprocal velocity obstacle to capture most uncertainty of the measurements. We denote the uncertainty-adjusted hybrid reciprocal velocity obstacle by $HRVO^*_{A|B}$, and define it as the union of the simple hybrid velocity obstacles over all positions and



(a) (b) (c)

Fig. 6. (a) A zoomed-in view from the overhead camera showing the *iRobot Create* robots with their fiducial markers. (b, c) The traces of the robots in the *four corners* scenario (b), and the *moving obstacle* scenario (c). The positions of the robots every 1.25 seconds are shown with a disc. Later positions are drawn on top of earlier positions, and in a darker shade. Each robot is denoted by a different color, with the yellow discs in (c) corresponding to the path of the moving obstacle.

velocities within one standard deviation around their mean:

$$HRVO^*_{A|B} = \bigcup_{\mathbf{p}_A \in D(\mu(\mathbf{P}_A),\sigma(\mathbf{P}_A))} \bigcup_{\mathbf{p}_B \in D(\mu(\mathbf{P}_B),\sigma(\mathbf{P}_B))} \bigcup_{\mathbf{v}_A \in D(\mu(\mathbf{V}_A),\sigma(\mathbf{V}_A))} \bigcup_{\mathbf{v}_B \in D(\mu(\mathbf{V}_B),\sigma(\mathbf{V}_B))} HRVO_{A|B},$$

where $D(\mathbf{p}, r)$ is a disc as previously defined.

$VO^*_{A|B}$ and $RVO^*_{A|B}$ are defined similarly. It is easy to show that $VO_{A|B}$ is contained within the velocity obstacle $VO^*_{A|B}$, with $\mathbf{p}_A = \mu(\mathbf{P}_A)$, $\mathbf{p}_B = \mu(\mathbf{P}_B)$, and $\mathbf{v}_B = \mu(\mathbf{V}_B)$, where $\sigma(\mathbf{P}_A)$ and $\sigma(\mathbf{P}_B)$ are added to the radii of A and B, respectively, and whose sides have been perpendicularly moved outwards over a distance of $\sigma(\mathbf{V}_B)$ (see Fig. 5). The actual result is a truncated cone, but for simplicity, we use the full cone in our implementation. The same procedure can be applied for constructing $RVO^*_{A|B}$. In this case, its sides need be moved outward over a distance of $(\sigma(\mathbf{V}_A) + \sigma(\mathbf{V}_B))/2$ to account for the uncertainty in the velocities of A and B. From $VO^*_{A|B}$ and $RVO^*_{A|B}$, the uncertainty-adjusted hybrid reciprocal velocity obstacle $HRVO^*_{A|B}$ is constructed as explained in Section III-C.

V. Implementation and Experimentation

In this section, we describe our implementation of the approach presented above and report experimental results from several scenarios involving mobile robots.

A. Implementation Details

We implemented our approach using a centralized sensing system, with simple, low-cost robots controlled remotely over Bluetooth radio from a single computer with access to the localization data of each robot.

Our chosen platform is the *iRobot Create* programmable robot, based on the popular *Roomba* vacuum-cleaning robot. This is a differential-drive robot with two powered wheels and a third passive caster wheel to maintain balance. Each wheel may be controlled independently with a maximum speed of 500 mm/s, and weighing less than 2.5 kg, it has a favorable power-to-weight ratio that allows us to assume near-instantaneous acceleration. The limited sensing power of the *iRobot Create* does not allow it to localize itself with any degree of accuracy.

To localize the robots, we use a ceiling-mounted *Point Grey Grasshopper* digital video camera with a *FireWire 800* connection to obtain images at a resolution of 1024x768 at a refresh rate of 15 Hz. We attach a fiducial marker to the top of each robot, as shown in Fig. 6(a), and use the *ARToolKit* augmented reality system [20] to determine the position and orientation of each robot, with an absolute error of less than 10 mm. The velocity of the robots is inferred from the position and orientation measurements using an extended Kalman filter.

All calculations are performed on a single computer. However, to ensure that our approach is also applicable when each robot uses its own on-board sensing and computing, only the acquisition of the localization data is performed centrally. The calculations for each robot related to our navigation approach are performed in separate processes that do not communicate with each other. Each process sends the computed wheel speeds to its associated robot over a Bluetooth virtual serial connection (which we found to have a typical latency of around 50 ms).

B. Experimental Results

Using our camera-based localization system and four *iRobot Create* robots, we tested our approach in the following scenarios:

- *Four corners:* A robot is placed at each corner of a rectangular environment, and its goal is to navigate to the opposite corner of the environment on the diagonal. The robots will meet and have to negotiate around each other in the middle.
- *Moving obstacle:* One robot takes the role of a moving obstacle, traveling at constant velocity across the environment. The other robots have to cross its path to navigate to their goals, while avoiding collisions.

In both scenarios, each robot has the same preferred speed of 250 mm/s.

The traces of the robots are shown in Fig. 6(b) for the *four corners* scenario (see also the video accompanying this paper). As can be seen from the figure (and video), the paths computed by the robots are smooth and direct, with no oscillations or collisions. Each of the robots makes just enough room for the other robots, so that they can precisely pass each other in the center without substantially slowing down. It is clear in the results that the robots make use of the information regarding both the positions and velocities of

the other robots to plan a path that exactly avoids collisions while maintaining as direct a path to the goal as possible. This leads to natural motions, and the reciprocity of the hybrid reciprocal velocity obstacle formulation ensures that the robots do not exhibit oscillations.

To test the robustness of our formulation, we artificially decreased the frequency with which sensor readings come in and commands are sent to the robots (from 15 Hz to 5 Hz) and increased the preferred speed of the robots (from 250 mm/s to 400 mm/s). Even with these changes, our approach continues to perform well and computes collision-free motion with no oscillations.

The *moving obstacle* scenario shows that our approach can naturally deal with the presence of an agent that will not necessarily adapt its motion to the presence of other robots. In this case, the robots use the original velocity obstacle formulation, with our adjustments for uncertainty, to avoid collisions with the moving obstacle along with the hybrid reciprocal velocity obstacle formulation to avoid collisions with each other. We do not consider how to identify the agent as a moving obstacle, simply that our method is capable of handling the distinction should it be made. Fig. 6(c) shows that two robots increase speed to cross in front of the moving obstacle, while the third slows and crosses behind. In this scenario, there were again neither collisions nor oscillations.

Extended videos of these and other scenarios are available at http://gamma.cs.unc.edu/HRVO/.

VI. Conclusion

In this paper, we have introduced the concept of *hybrid reciprocal velocity obstacles* for smooth, direct, and collision-free navigation of multiple robots sharing an environment. We have incorporated the kinematics and sensor uncertainty of the robot in our formulation, and implemented our approach on mobile robots. The key feature of our formulation is that it explicitly considers *reciprocity*, such that each robot can assume that other robots are cooperating to avoid collisions. Yet, each of the robots fully acts independently, and does not communicate with other robots.

In our current implementation, each of the robots receives their sensor readings from an overhead camera. An important next step is to equip each robot with purely localized sensing and computing, such as in [21], which combines odometry, orientation sensors, and relative positions to accurately estimate global positions. Our approach can be applied without adaptation if data is gathered locally, and the hybrid reciprocal velocity obstacles are defined just as well using only *relative* positions and velocities of robots, rather than absolute positions and velocities as used in this paper.

Our current implementation focuses on differential-drive robots, but our approach can easily be adapted for other kinematic systems, in particular car-like robots as they have similar kinematics.

As our approach extrapolates current velocities, it is in principle capable of handling robots moving at high speeds. However, in our current implementation, we have ignored inertia as its effects proved negligible compared to those due to latency. At higher speeds, this may no longer be the case, so dynamics may need to be explicitly addressed.

Future work also includes extending our approach to deal with complex environments that contain static obstacles, for instance walls in an office environment. In this case, our formulation will need to be augmented with global path planning to direct the robots towards their goals, potentially following the approach of [22].

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, 1998.

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

[3] D. Hsu, R. Kindel, J.-C. Latombe, and S. Rock, "Randomized kinodynamic motion planning with moving obstacles," *Int. J. Robot. Res.*, vol. 21, no. 3, pp. 233–255, 2002.

[4] A. Kushleyev and M. Likhachev, "Time-bounded lattice for efficient planning in dynamic environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2009, pp. 1662–1668.

[5] S. Petti and T. Fraichard, "Safe motion planning in dynamic environments," in *Proc. IEEE RSJ Int. Conf. Intell. Robot. Syst.*, 2005, pp. 2210–2215.

[6] J. van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2008, pp. 1928–1935.

[7] F. Feurtey, "Simulating the collision avoidance behavior of pedestrians," Master's thesis, Univ. Tokyo, 2000.

[8] F. Large, S. Sckhavat, Z. Shiller, and C. Laugier, "Using non-linear velocity obstacles to plan motions in a dynamic environment," in *Proc. IEEE Int. Conf. Control Autom. Robot. Vision*, 2002, pp. 734–739.

[9] E. Prassler, J. Scholz, and P. Fiorini, "A robotic wheelchair for crowded public environments," *IEEE Robot. Autom. Mag.*, vol. 8, no. 1, pp. 38–45, 2001.

[10] C. Reinholtz, "Victor Tango," DARPA Urban Chall. Tech. Pap., 2007.

[11] J. S. Dittrich, F. Adolf, A. Langer, and F. Thielecke, "Mission planning for small UAV systems in unknown environments," in *AHS Int. Spec. Mtg. Unmanned Rotorcraft Syst.*, 2007.

[12] S. J. Guy, J. Chhugani, C. Kim, N. Satish, M. Lin, D. Manocha, and P. Dubey, "ClearPath: Highly parallel collision avoidance for multi-agent simulation," in *Proc. ACM SIGGRAPH Eurographics Symp. Comput. Animat.*, 2009, pp. 177–187.

[13] C. Fulgenzi, A. Spalanzani, and C. Laugier, "Dynamic obstacle avoidance in uncertain environment combining PVOs and occupancy grid," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2007, pp. 1610–1616.

[14] B. Kluge and E. Prassler, "Reflective navigation: Individual behaviors and group behaviors," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2004, pp. 4172–4177.

[15] Y. Abe and M. Yoshiki, "Collision avoidance method for multiple autonomous mobile agents by implicit cooperation," in *Proc. IEEE RSJ Int. Conf. Intell. Robot. Syst.*, 2001, pp. 1207–1212.

[16] K. C. Ng and M. M. Trivedi, "A neuro-fuzzy controller for mobile robot navigation and multirobot convoying," *IEEE T. Syst. Man Cyb. B*, vol. 28, no. 6, pp. 829–840, 1998.

[17] S. Carpin and L. E. Parker, "Cooperative motion coordination amidst dynamic obstacles," in *Proc. Int. Symp. Distrib. Auton. Robot. Syst.*, 2002, pp. 145–154.

[18] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge Univ. Pr., 2006.

[19] G. Welch and G. Bishop, "An introduction to the Kalman filter," Univ. N. Carolina Chapel Hill, Tech. Rep. 95-041, 1995.

[20] H. Kato and M. Billinghurst, "Marker tracking and HMD calibration for a video-based augmented reality conferencing system," in *Proc. IEEE ACM Int. Workshop Augment. Real.*, 1999, pp. 85–94.

[21] S. I. Roumeliotis and I. M. Rekleitis, "Propagation of uncertainty in cooperative multirobot localization: Analysis and experimental results," *Auton. Robot.*, vol. 17, pp. 41–54, 2004.

[22] J. van den Berg, S. Patil, J. Sewall, D. Manocha, and M. Lin, "Interactive navigation of multiple agents in crowded environments," in *Proc. Symp. Interact. 3D Graph. Game.*, 2008, pp. 139–147.