The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Avoiding Moving Obstacles: the Forbidden Velocity Map

Bruno Damas and José Santos-Victor

Abstract— **Robotic obstacle avoidance in cluttered and dense environments is an important issue in robotic navigation. Over the past few years a number of techniques has been proposed to deal with safe navigation among obstacles in unknown scenarios. Unfortunately many of these methods do not consider obstacle velocities, which can rise some serious questions concerning their safety [1]. This paper will deal with a novel approach to moving obstacle avoidance in holonomic robots. It proposes the Forbidden Velocity Map, a generalization of the Dynamic Window concept [2] that considers obstacle and robot shape, velocity and dynamics, resulting in a safe, reactive real-time navigation algorithm that is able to deal with navigation in unpredictable and cluttered scenarios.**

I. Introduction and Related Work

Obstacle avoidance for autonomous robotic navigation has traditionally been handled by two major techniques: the deliberative approach, usually consisting of a motion planner taking world static information as its input ([3] provides an extensive review on planning methods), and the reactive approach, where the instantaneous information accessible to the robot (the positions and eventually the robot and obstacles velocities) is used to calculate the robot actuation at each time step. Among the reactive techniques we have, for instance, the potential fields approach [4], [5], the vector field histogram [6], the curvature method [7] or the dynamic window approach [2], to name just a few.

One of the main advantages of reactive methods is their low demanding of computing resources, making them very suitable to real-time robot navigation. These methods can be seen as local navigation algorithms, using only sensor information concerning the vicinity of the robot. Since they use only local information these methods are prone to being trapped in some obstacle configurations. Global planning algorithms, on the other hand, process information relative to the all environment in order to find a sequence of actuation commands that guarantees a convergence to the goal. Typically such methods are very time consuming; moreover the assumption of a known static environment sometimes is not a very reasonable one, since most scenarios are dynamic and partially or completely unknown. As a consequence the trajectory generated by the planning algorithm must be recalculated from time to time to take into account environment changes. To overcome such limitations some methods that incorporate both local and global techniques have been developed: Thrun et al. use a path planner to calculate intermediate points on the path to the goal [8], while dynamic window approach has been extended by Brock and Khatib to incorporate a global planner that assures convergence to the goal [9].

Bruno Damas and José Santos-Victor are with the Instituto de Sistemas e Robótica, Instituto Superior Técnico, Lisboa, Portugal. Bruno Damas is also with Escola Superior de Tecnologia de Setúbal, Portugal. {bdamas,jasv}@isr.ist.utl.pt

This work was (partially) supported by the FCT Programa Operacional Sociedade de Informação (POSI) in the frame of QCA III, and by the EU Proj. (FP6-EU-IST-045062) URUS.

Surprisingly, many of the obstacle avoidance literature of the past two decades does not explicitly take into account the obstacle velocities. Simply discarding obstacle velocities can have severe consequences for the robot safety [10], [1]; considering the obstacles velocities, on the other hand, can improve the smoothness of the robot trajectories, as will be shown. In [11] a planning algorithm is proposed that considers obstacle velocities. Each obstacle yields a cone shaped forbidden velocity: if the robot velocity enters such cone a collision will occur in some latter time. A planned trajectory is obtained taking into account all obstacle cones and considering admissible velocities those whose expected collision occurs after a fixed time horizon. [12] presents an obstacle avoidance algorithm for non-holonomic robots that takes the obstacles velocities into consideration, by obtaining a time to collision surface on the velocity space of the robot. Their method has some similarities to the one presented in this paper, but in our method a forbidden velocity region is calculated for which a collision will occur if the robot maintains the same direction of motion, as opposed to the time to collision surface. Our method also has the ability to effortless incorporate information on obstacle position and velocity uncertainty, as it will be shown in the next sections.

The obstacle avoidance algorithm proposed in this paper is related to the Dynamic Window Approach in the sense that a set of robot forbidden velocities is obtained at each time step. This method, however, does include information on obstacle velocity to generate motor actuations. It also uses obstacle and robot shape and dynamic considerations to obtain the Forbidden Velocity Map for obstacle avoidance navigation. Other contribution is the fact that no discretization of the velocity space is performed to obtain the set of admissible velocities: instead, a piecewise linearization of the Forbidden Velocity Map boundary is performed that allows a faster search for admissible velocities. Also, uncertainty on obstacle position and velocity is considered on the proposed algorithm, that can be conceptually described by the following steps:

- For each obstacle a polar curve is obtained that indicates the distance to collision for each robot direction of motion. The obstacle is (temporarily) assumed to be static. Such curve only depends on shape considerations on the robot and on the obstacle.
- Each distance to collision curve is then transformed into

- a forbidden velocity zone. Such zone is the set of current robot velocities that lead to collision if the robot is supposed to decelerate using its maximum break power (and maintaining its direction of movement) in the following time step. This transformation to a forbidden velocity zone depends only on the dynamic of the robot.
- Each forbidden velocity zone is then translated according the corresponding obstacle velocity. After that a piecewise linearization of each forbidden zone is performed and, in the end, all these individual zones are gathered to obtain a global forbidden velocity map. This map consists of an union of (possible non convex) polygons describing the set of velocities for which a collision may occur.
- At each time step the robot desired velocity is checked about its admissibility: if it leads to a possible collision another velocity is picked in the boundary of the forbidden velocity map according to some criteria.

II. The Forbidden Velocity Map for Moving Obstacles

Although the obstacle avoidance algorithm presented in this paper can potentially deal with robots with arbitrary geometry and locomotion type, we will henceforth, for the sake of simplicity, consider a robot of omni-directional type: these kind of vehicles have a kinematic model that allows for a decoupled control for x, y and θ, the Cartesian position and heading of the robot that altogether constitute the robot configuration variables. This independent control of the three configuration space variables is possible due to the full rank nature of the Jacobian relating motor commands and the time derivatives of the configuration space variables. Forbidden velocity map for non-holonomic robots can be obtained following the approach presented in [12].

A. General assumptions

A discrete time control scheme is assumed, with fixed sample time T. At each sample instant, the robot has access to information regarding obstacle shape, position and velocity, along with its own position and velocity. This information will be used to calculate the motor controls to be applied to the robot at the next sampling instant, in order to reach a desired target while avoiding the environment moving obstacles.

Since control actions lag correspondent sensor estimates by exactly one sampling period T, the obstacle avoidance algorithm presented in this paper must take into account that when a motor command is issued the robot has already traveled a vT distance along current robot direction of movement, where v is the respective robot speed measured at sensing time.

B. Distance to collision

Consider a circular robot of radius R and a circular static obstacle O_i with radius r_i, as represented in Figure 1, where α_i represents the obstacle direction in the robot referential and h_i is the distance between robot and obstacle. Making

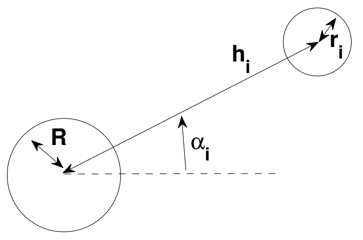


Fig. 1. The static obstacle problem formulation

the robot a singular point by expanding the obstacle radius, $R_i = R + r_i$, the distance to collision can be seen as a simple line-circle interception problem, and after some geometric considerations the following equation for the distance to collision can be obtained, written as a function of the robot direction of movement θ:

$$d_i(\theta) = h_i \left(\cos(\theta - \alpha_i) - \sqrt{\frac{R_i^2}{h_i^2} + \cos^2(\theta - \alpha_i) - 1} \right) , \quad (1)$$

with

$$\alpha_i - \Delta\alpha_i \leq \theta \leq \alpha_i + \Delta\alpha_i , \quad (2)$$

where $\Delta\alpha_i = \arccos \sqrt{1 - R_i^2/h_i^2}$.

Eq. (1) depends only on robot and obstacle geometry, and similar equations can be derived for other types of geometric objects such as, for instance, straight lines modeling walls, using a similar reasoning. In [13] another method is proposed that takes into consideration the robot shape.

C. Collision free velocities

If the robot is to stop using its maximum breaking power, a_{max}, the time t_{stop} until immobilization is given by

$$v - a_{max} t_{stop} = 0 \Leftrightarrow t_{stop} = \frac{v}{a_{max}} , \quad (3)$$

where v is the robot speed at the time the braking order is issued. If the robot is moving toward a static obstacle, the distance traveled by the robot must be, at most, the distance d until a collision occurs with such an obstacle:

$$vT + vt_{stop} - \frac{1}{2} a_{max} t_{stop}^2 \quad = \quad d .$$

Plunging equation (3) and solving for v we get

$$v_{max}(\theta) = \sqrt{2a_{max} d(\theta) + a_{max}^2 T^2} - a_{max} T , \quad (4)$$

where a dependence on direction θ as been assumed. This way, if the collision distance with an obstacle located in direction θ is given by $d(\theta)$ and if the robot is moving toward such an obstacle in a straight line, the maximum speed allowed for the robot that avoids collision, using maximum break power, is given by Eq. (4). Notice that this latter equation describes the robot dynamics and does not take into account any geometric considerations.

Equations (4) and (1) define a **forbidden velocity map** for each static obstacle — a velocity set for which a collision can occur — in terms of robot velocity polar coordinates:

$$v_i(\theta) > \sqrt{2a_{max}d_i(\theta) + a_{max}^2 T^2} - a_{max}T \ , \qquad (5)$$

where $d_i(\theta)$ is given by equation (1).

If multiple static obstacles exist, the velocity forbidden map corresponds to the union of the individual forbidden maps for each obstacle O_i. Figure 2(a) represents the map that corresponds to the three obstacles depicted in Figure 2(b).

D. The forbidden map for moving obstacles

If an obstacle O_i is moving equation (5) still holds if the robot speed v_i is replaced by v_i^*, the *relative speed* between the robot and obstacle O_i. Such a relative speed can be obtained if the difference between robot and obstacle velocities is considered,

$$\begin{bmatrix} v_{xi}^* \\ v_{yi}^* \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \end{bmatrix} - \begin{bmatrix} v_{xi} \\ v_{yi} \end{bmatrix} , \qquad (6)$$

where (v_{xi}^*, v_{yi}^*), (v_x, v_y) and (v_{xi}, v_{yi}) are respectively the relative velocity, the robot velocity and the obstacle velocity, all expressed in Cartesian coordinates. It is easy to see that taking into account the obstacle velocity is equivalent to performing a linear translation on the velocity forbidden map for that obstacle by (v_{xi}, v_{yi}), as is depicted in Figure 2(c). A simple algebraic expression for the robot forbidden velocity map in the presence of multiple moving obstacles, however, is no longer possible to obtain: equation (5) presents the forbidden velocity zone in polar coordinates, while the relative velocity is stated as a difference between Cartesian coordinates.

E. Forbidden map linearization

Consider an obstacle O_i whose state is characterized by α_i, h_i, R_i, v_{xi} and v_{yi}. The forbidden map boundary is, according to (4), the curve defined by

$$F(v^*, d) = v^* - \sqrt{2a_{max}d + a_{max}^2 T^2} - a_{max}T = 0 \ , \qquad (7)$$

where d depends on θ and obstacles according to expression (1). This dependence on θ can be written, for future convenience, as

$$\begin{bmatrix} v^* \\ d \end{bmatrix} = G_i(v^*, \theta) = $$
$$= \begin{bmatrix} v^* \\ h_i\cos(\theta-\alpha_i) - h_i\sqrt{\frac{R_i^2}{h_i^2} + \cos^2(\theta-\alpha_i) - 1} \end{bmatrix}, \qquad (8)$$

again with θ defined over the interval given by Eq. (2).

The robot absolute velocity (v_x, v_y) can be obtained from the relative velocity to obstacle O_i according to (6), and thus a relation between robot absolute velocity and polar coordinates of relative velocity can easily be obtained:

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = H_i(v^*, \theta) = \begin{bmatrix} v^*\cos\theta \\ v^*\sin\theta \end{bmatrix} + \begin{bmatrix} v_{xi} \\ v_{yi} \end{bmatrix} . \qquad (9)$$

The forbidden map boundary B_i for obstacle O_i can thus be expressed, using the previous relations, *as a function of the robot absolute velocity* (v_x, v_y),

$$B_i(v_x, v_y) = F \circ G_i \circ H^{-1}(v_x, v_y) = 0 \ . \qquad (10)$$

Performing a piecewise linearization for every obstacle boundary $B_i(v_x, v_y)$ not only transforms this boundary into a polygonal one, thus greatly lowering the computational burden of the algorithm, but also greatly simplifies the task of finding the union of forbidden zones for multiple obstacles. To do that, note that the gradient of $B_i(v_x, v_y)$ is readily obtained from (10):

$$\nabla B_i(v_x, v_y) = \nabla F \ \nabla G_i \ \nabla H^{-1}(v_x, v_y) \ , \qquad (11)$$

where we have

$$\nabla F(v^*, d) = \begin{bmatrix} 1 & -\left(2\frac{d}{a_{max}} + T^2\right)^{-\frac{1}{2}} \end{bmatrix} , \qquad (12)$$

$$\nabla G_i(v^*, \theta) =$$
$$\begin{bmatrix} 1 & 0 \\ 0 & h_i\sin(\theta-\alpha_i)\left(\frac{\cos(\theta-\alpha_i)}{\sqrt{\cos^2(\theta-\alpha_i)+\frac{R_i^2}{h_i^2}-1}} - 1\right) \end{bmatrix} \qquad (13)$$

and

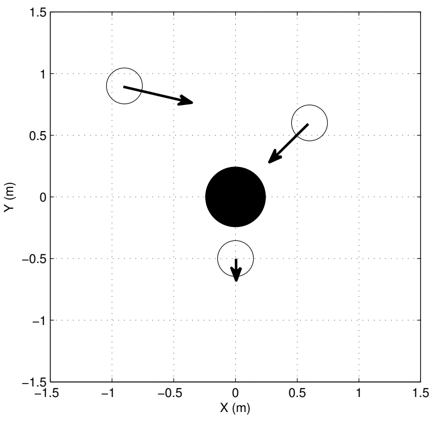
$$\nabla H^{-1}(v_x, v_y) =$$
$$\left.\begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta)/v^* & \cos(\theta)/v^* \end{bmatrix}\right|_{(V^*,\theta)=H^{-1}(v_x,v_y)} . \qquad (14)$$

The piecewise linearization of the forbidden map boundary can now be performed by considering a corresponding discretization of the admissible values of θ, as given in Eq. 2, for each existing obstacle. In this paper we consider a uniform sampling for the interval (2) into a set of N_θ different angles θ_{ik}, with $1 < k < N_\theta$, although any other kind of reasonable sampling can also be performed.

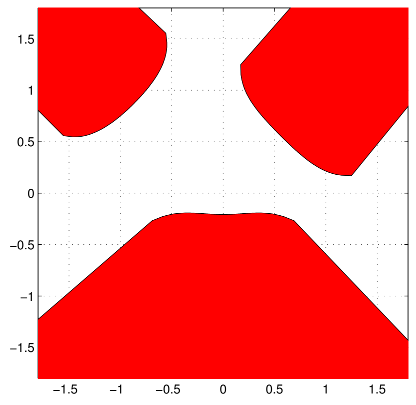
The linearized forbidden velocity boundary for each obstacle can now be represented by a set of N_θ intercepting straight lines, each of them described by the following parameters:

- Application point $B_{ik} = (v_{xk}, v_{yk})$, where (v_{xk}, v_{yk}) is obtained for each value θ_{ik} by successively obtaining corresponding values of d and $v*$ using equations (8), (7) and (9).
- Normal direction $\nabla B_{ik} = \nabla B_i(v_{xk}, v_{yk})$, obtained using Eq. (10). Note that this normal vector points to the interior of the forbidden velocity map.

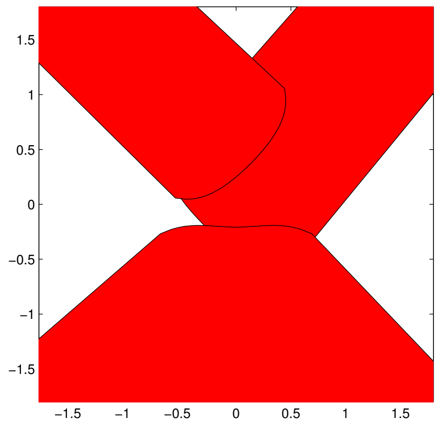
The final Forbidden Velocity Map is found by performing the union of the polygons describing each obstacle forbidden velocity zone.



(a)



(b)



(c)

Fig. 2. (a) Three distinct obstacles and corresponding velocities. (b) The velocity forbidden map for this obstacle configuration, assuming null velocities. (c) The velocity forbidden map for this obstacle configuration, assuming corresponding velocities.

III. Uncertainty

An uncertainty Δr in robot and obstacles position can readily be modeled as an increase in total radius R_i, *i.e.*, $R_i = R + r_i + \Delta r$. This uncertainty typically results from noisy sensors and measures and can easily be estimated *a priori*. One can have, of course, a different position uncertainty for each obstacle, but for most of the cases it just turns out to be sufficient to use a global uncertainty measure Δr.

On the other hand, robot and obstacle velocity uncertainty plays a fundamental role on the robot navigation behavior: a high uncertainty on obstacle velocities denote the possibility of an abrupt change of obstacle velocity on the next sampling time: therefore, the robot must have a more cautious behavior, which, on the other hand, implies the construction of an accordingly more conservative forbidden velocity map.

To take into account a speed uncertainty Δv we just have to enlarge each obstacle velocity forbidden map by Δv. This is particularly simple to achieve after the linearization of the forbidden map has been done: just move each point B_{ik} a distance Δv outwards the forbidden map, along the correspondent gradient ∇B_{ik}.

This uncertainty on obstacle velocity just comes to be an excellent free parameter that allows the obstacle avoidance algorithm to exhibit a wide range of dynamic behaviors: keeping such an uncertainty high makes the algorithm more conservative, since each obstacle velocity is expected to change by a higher quantity each time step; alternatively, obstacles with a low uncertainty on the respective velocity account typically for slow time varying trajectories or static obstacles, for which a more aggressive control is possible.

IV. Getting to the goal

The proposed obstacle avoidance method does not consider how the robot can reach a desired posture, thus allowing a easy integration with a large variety of navigation schemes. In fact, this algorithm can be interpreted as a kind of dynamic perturbation on the actuation provided by some navigation algorithm. Such a perturbation causes a deviation from the planned robot path in order to avoid an imminent collision.

In each time step the algorithm obtains a region on the robot velocity space where a collision will occur if the robot and obstacles velocities remain constant. So, given a desired velocity, provided by some kind of navigation algorithm, we must check if such a velocity is an admissible one, *i.e.*, one that does not belong to the forbidden velocity map. When a velocity is not considered safe the proposed algorithm rectifies the velocity so as to move it away from the forbidden zone. Such a procedure will produce a velocity belonging to the border of the forbidden velocity map. This makes the algorithm computationally very efficient, since finding the minimum distance point of a polygon to a given location is very fast and scales linearly with the number of edges of the polygon describing the obstacles.

Note that no claims on optimality or collision free trajectories are made in this paper. The obstacles motion and position is not known *a priori*, making quite infeasible the task of obtaining a collision free optimal sequence of motor commands regarding some time or displacement criterion. On the other hand, in dense and cluttered environments some obstacle configuration can be achieved where an obstacle moves in the direction of the robot and the robot has no place to go in order to avoid a collision. Such claims can be obtained only in highly controlled environments.

A. The dynamic window and saturation window

Besides the velocity restrictions due to the presence of obstacles, one must also take into account actuators saturation and acceleration constraints. Actuators saturation restrains the set of possible reachable velocities, while the acceleration constraints limit the achievable robot velocities in the next time step. Together, they respectively produce a saturation window and a dynamic window on the velocity space. Other works [9] use these windows to narrow the velocity space search. In this paper, however, these restrictions are needed in order to ensure the correctness of the algorithm: otherwise the desired velocity, in order to avoid an obstacle, could be transformed in a velocity physically impossible to attain. The dynamic and saturation windows appear as two additional restriction zones in the velocity forbidden map:

$$\left|(v_x, v_y)\right| > v_{max}$$

and

$$\left|(v_x, v_y) - (v_x, v_y)^{(t)}\right| > a_{max}T \ ,$$

where $(v_x, v_y)^{(t)}$ is the current robot velocity. The dynamic and saturation window can thus be incorporated in the algorithm by simply including these forbidden zones in the previously obtained Forbidden Velocity Map.

B. Additional physical constraints

There is a set of other physical constraints that can be dealt with by this algorithm in a very elegant way. A robot pushing a box on the floor, for instance, can be viewed as a balance in the robot velocities and accelerations that must be achieved in order to not let the box slip away. Similarly, the robotic soccer problem of moving fast while carrying the ball can be viewed in the same perspective [14]. It turns out that, if such problems can be put as a set of restrictions and dependences on the robot velocities, they can, in principle, be handled quite easily by the proposed method.

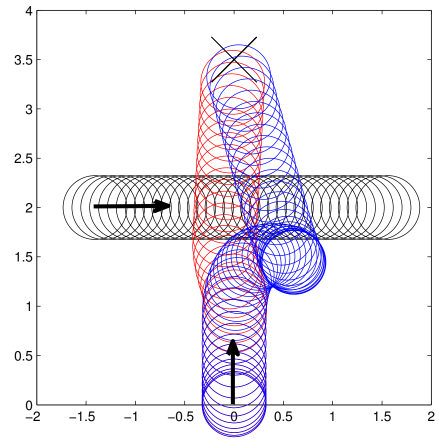


Fig. 3. Robot behavior when avoiding a crossing obstacle. Black: obstacle path; red: Forbidden Velocity Map; blue: Modified Potiential Fields [14].

V. Experimental Results

There is one particular test where most reactive obstacle avoidance methods based only on obstacle positions fail: an obstacle crossing perpendicularly the robot path, as illustrated in Fig. 3. Since these schemes consider obstacles at each time step to be static ones, when a obstacle starts to cross their path coming, say, from the left, they usually try to avoid collision turning to the right. However, since the obstacle is also moving in that direction, that results in a robot trajectory side by side to the obstacle, as if trying to circumvent an imaginary wall.

The Forbidden Velocity Map, on the other hand, by considering the obstacle velocity, does not exhibit such undesirable behavior, as can be seen in the previous figure. In this example the robot *turns* in the direction of the obstacle, *i.e.*, to the left, in order to maintain a velocity as close as possibly to the maximum speed in the direction of its target.

Figure 4 shows the role of the uncertainty on obstacle velocity. Such uncertainty can be a result of sensor noise, but mainly will arise as an estimate of the maximum change in obstacle trajectories at each time step. A high value for this free parameter makes the robot very conservative, since the sensed obstacle velocity can be very different from the true one. On the other hand, a very low value for this uncertainty implicitly assumes constant velocity for the obstacles, *i.e.*, static obstacles or constant speed linear trajectories. In this latter case the algorithm causes the robot to pass very close to the obstacles, as their trajectories are presumed to be fully predictable. In the figure two trajectories are shown for the same static obstacle, for different values of the uncertainty on the obstacle velocity. Blue trajectory corresponds to a higher value of this parameter, causing a broader trajectory around the obstacle.

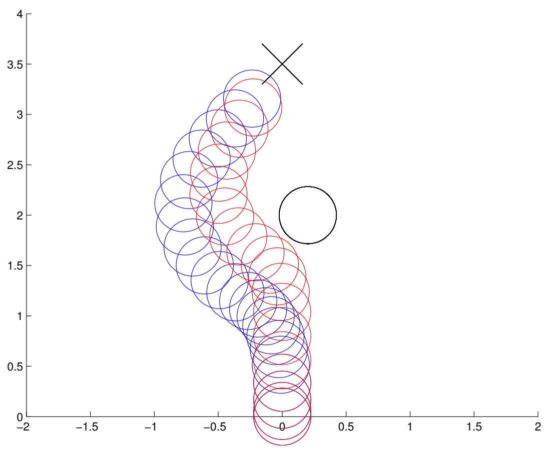


Fig. 4. Uncertainty on the obstacle velocity affects the overall robot behavior.

The video that was jointly submitted with this paper presents five different tries of an omni-directional robot to reach a desired target, marked with an X, using the Forbidden Velocity Map. The first three scenarios, however, are cluttered with many obstacles moving with random motion and that change their velocity from time to time. This is a very demanding environment (can correspond, for instance, to a room filled with people) for which the proposed algorithm, nevertheless, can react with quite success. The other two experiences depicted in the video show on one hand the ability of the proposed algorithm to deal with fast crossing obstacles and in the other hand how it can successfully be confronted with fast parallel obstacles, that can simulate, for instance, traffic on a highway.

VI. Conclusions

In this paper we present a computationally fast algorithm that allows a robot to avoid collision with multiple moving obstacles. Instead of performing an exhaustive search on the robot velocity space, the proposed method computes the Forbidden Velocity Map, a union of polygonal zones corresponding to the non admissible velocities.

The resolution of the linearization performed in section II-E is a free parameter that sets a compromise between the computational burden of the algorithm and the quality of the approximation to the true Forbidden Velocity Map. Note, however, that given the extremely low computational cost of the algorithm this is hardly a problem in modern computers: when integrated into a more broad robotic platform, the CPU use of the proposed obstacle avoidance algorithm is typically irrelevant.

The proposed method can be easily extended to other robot shapes and different dynamic models: while Equation (1) alone defines the geometric considerations for the obstacles and the robot, Equation (4) deals with the dynamic model for the robot. In this way we can deal with arbitrary obstacle shapes (circles and lines and composition of circles and lines, for instance).

Another advantage of the forbidden velocity approach to obstacle avoidance is the effortless integration of uncertainty on obstacle positions and velocities in the algorithm. In fact, the uncertainty on the variation in obstacle velocities every time step constitutes a great free parameter to control the global algorithm behavior, allowing it to exhibit a more conservative motion or a more unsafe navigation. No other parameters need to be tuned.

Finally, although the presented method does not guarantee a collision free navigation, it can identify the velocities that lead to collision if no alteration occurs on the obstacles and robot velocities. This provides a very convenient way to adapt a desired velocity in the presence of obstacles in order to prevent a collision. We intend to test the presented method in the real, demanding environment of the robot soccer.

References

[1] T. Fraichard. A Short Paper about Motion Safety. In *2007 IEEE International Conference on Robotics and Automation*, pages 1140–1145, 2007.

[2] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4(1):23–33, 1997.

[3] J.C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[4] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 5(1):90–98, 1986.

[5] M. Khatib and R. Chatila. An extended potential field approach for mobile robot sensor-based motions. In *Proc. of Intelligent Autonomous Systems*, pages 490–496, 1995.

[6] J. Borenstein and Y. Koren. The vector field histogram-fast obstacle avoidance for mobilerobots. *IEEE Transactions on Robotics and Automation*, 7(3):278–288, 1991.

[7] R. Simmons. The Curvature-Velocity Method for Local Obstacle Avoidance. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 3375–3382, 1996.

[8] S. Thrun, A. Bücken, W. Burgard, D. Fox, T. Fröhlinghaus, D. Hennig, T. Hofmann, M. Krell, and T. Schmidt. Map Learning and High-Speed Navigation in RHINO. *AI-based Mobile Robots: Case Studies of Successful Robot Systems*, 1998.

[9] O. Brock and O. Khatib. High-speed navigation using the global dynamic window approach. In *IEEE Int. Conf. on Robotics and Automation*, pages 341–346, 1999.

[10] T. Fraichard and H. Asama. Inevitable Collision States A Step Towards Safer Robots? *Advanced Robotics*, 18(10):1001–1024, 2004.

[11] P. Fiorini, Z. Shiller, and T.A. Ansari. Motion Planning in Dynamic Environments using Velocity Obstacles. *International Journal of Robotics Research*, 17(7):760–772, 1998.

[12] E. Owen and L. Montano. Motion planning in dynamic environments using the velocity space. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005.(IROS 2005)*, pages 2833–2838, 2005.

[13] J. Minguez, L. Montano, and J. Santos-Victor. Abstracting Vehicle Shape and Kinematic Constraints from Obstacle Avoidance Methods. *Autonomous Robots*, 20(1):43–59, 2006.

[14] B. Damas, P. Lima, and L. Custodio. A modified potential fields method for robot navigation applied to dribbling in robotic soccer. *Lecture Notes in Computer Science*, 2752:65–77, 2003.