2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Dynamic Obstacle Avoidance in uncertain environment combining PVOs and Occupancy Grid

Chiara Fulgenzi, Anne Spalanzani, and Christian Laugier
Laboratoire d'Informatique de Grenoble, INRIA Rhône-Alpes, France
Email: firstname.lastname@inrialpes.fr

Abstract— **Most of present work for autonomous navigation in dynamic environment doesn't take into account the dynamics of the obstacles or the limits of the perception system. To face these problems we applied the Probabilistic Velocity Obstacle (PVO) approach [1] to a dynamic occupancy grid. The paper presents a method to estimate the probability of collision where uncertainty in position, shape and velocity of the obstacles, occlusions and limited sensor range contribute directly to the computation. A simple navigation algorithm is then presented in order to apply the method to collision avoidance and goal driven control. Simulation results show that the robot is able to adapt its behaviour to the level of available knowledge and navigate safely among obstacles with a constant linear velocity. Extensions to non-linear, non-constant velocities are proposed.**

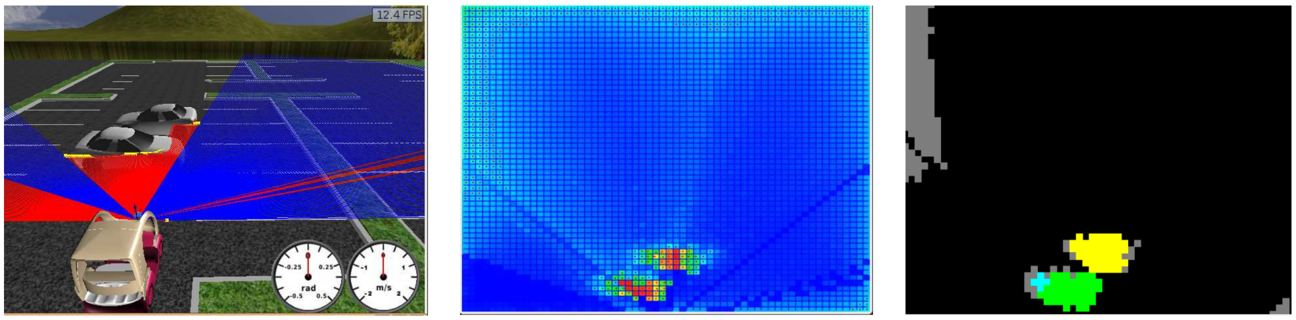
I. Introduction

Mobile robots navigation in dynamic environments represents still a challenge for real world applications. The robot should be able to gain its goal position navigating safely among moving people or vehicles, facing the implicit uncertainty of the surrounding world and the limits of its perception system.
The problem of autonomous navigation has been deeply studied in literature and several techniques have been developed. The global approaches (path planning algorithms) compute a complete path from the robot actual position to the goal [2]. In the case of moving obstacles, a common technique is to add the time dimension to the state space and reduce the problem to a static one [3]. Also if global methods can provide optimal solutions, their major drawback is that they assume a complete and deterministic knowledge of the environment: in practical applications they are usually combined with local methods in order to avoid unexpected obstacles [4], [5]. These last ones, also called reactive methods, generate just the next input control: they use only the nearest portion of the environment and update the world model according to the current sensor observation. Most of the developed techniques, as the Dynamic window approach [4], [6], the curvature velocity [7] and the lane curvature method [8] don't take into account the dynamic information of the environment, considering all the obstacles as static ones. On the other side, the Velocity Obstacles approach [9], [10], the Inevitable Collision States concept [11] and [12] use a deterministic knowledge about the velocity of the obstacles to compute collision-free controls. All the cited methods however, rely on a complete knowledge of the static and dynamic environment and a deterministic representation of the world.
In this paper we propose a reactive obstacle avoidance based on a probabilistic framework such to make the connection between the perception and the navigation system of the robot. In [1], the Probabilistic Velocity Obstacle approach (PVO) has been proposed as an extension of the VOs to the case of uncertain estimation of velocity and of the radius of circular obstacles. We combined this method with the dynamic occupancy grid provided by a general sensor system. The hypotheses on the robot and obstacle shape are removed. The sensors provide a probabilistic estimation of the occupied and free space around the robot and of the velocity with which the objects are moving; the observations update a 4D probabilistic occupancy grid (space and velocity) [13]; the probability of collision in time is estimated for each reachable velocity of the robot. A simple navigation algorithm is also proposed in order to apply the best control with respect to safety issues and convergence to the goal. Simulation results show how the developed algorithm takes directly into account limited range and occlusions, uncertain estimation of velocity and position of the obstacles, allowing the robot to navigate safely toward the goal and to modify its behaviour according to the quality of its perception.
The paper is structured as follows: in Section II the Bayesian Occupancy Filter (BOF) and the Velocity Obstacle framework are recalled and discussed; in Section III the developed solution is described in detail. In Section IV simulation results are shown and discussed. Section V closes the document with remarks and purposes for future activities.

Fulgenzi Chiara is supported by a grant from the European Community under the Marie-Curie project VISITOR MEST-CT-2004-008270

II. Related works

The method here developed combines two existing frameworks: the Bayesian Occupancy Filter [14] and the Linear Velocity Obstacles [9]. The Bayesian Occupancy Filter (BOF) is a dynamic occupancy grid where an estimation of velocity is stored as well as the probability of occupation. Sensor observations are processed from the BOF and the resulting grid is given as input to the obstacle avoidance algorithm. The following paragraphs recall respectively the BOF algorithm and the Linear Velocity Obstacles approach. Paragraph II-C discusses the advantages of the combination of the two methods.



(a) (b) (c)

Fig. 1. Simulated detection of two cars crossing each others. (a) Simulated environment : the robot equipped with a laser range finder detects a car moving from left to right and a second car moving from right to left. (b) Dynamic occupancy grid: red is high, blue is low probability of occupation. The space behind the cars has low probability of occupation. (c) Clustering: different colours characterise objects and occluded or free space.

A. *The Bayesian Occupancy filter (BOF)*

Probabilistic occupancy grids are well known structures used for environmental representation. The space is divided in a finite number of cells, each representing a position in the 2D plane (X, Y), $X = [\frac{x}{q}], Y = [\frac{y}{q}]$, where q is the discretization step. The estimation of the state of the system $x(t)$ at time t is the list of the states of all the cells of the grid: Occ, when the cell is occupied or Emp if the correspondent space is free. Given a probabilistic sensor model $P(z(t)|x(t))$ where $z(t)$ is the current observation, the grid is updated following the Bayes rule. Under the hypothesis that each cell of the grid is statistically independent from its neighbourhood, each cell state estimation is updated independently [15].
If some moving obstacles is present, the precedent structure is not sufficient to describe the state of the environment and it is necessary to introduce a description of velocity and a dynamical model. To perform an estimation, the state of the grid is firstly modified according the dynamical model (prediction step) and then compared with the acquired observation (updating step). These ideas are at the basis of the Bayesian Occupancy Filter [13]. Each cell maintains not only an estimation of its occupation probability, but also a discretized representation of the probabilistic distribution function (pdf) over velocities. A minimum and maximum velocity value is considered for eventual objects in the space, so that the pdf is given by a finite histogram over velocity values v_n with $n = 1...N$. The discrete approximation is performed according to the spatial and time discretization: given τ the time step, only integer velocities in terms of $\frac{q}{\tau}$ are taken under consideration, in order to perform fast and rigorous prediction and updating steps. Here we present a brief scheme of the algorithm:

1) At the beginning of the estimation, the occupancy grid is initialised with the prior knowledge of the environment: if no knowledge is available all the cells are initialised with a 0.5 probability of occupation and a uniform distribution over velocities;
2) A prediction step is performed according to the state of the environment and a constant velocity dynamical model. For each cell $c = [i, j]$ and for each value of velocity $v_n = [di, dj]$, an antecedent cell is considered : $c_a(n) = [i - di, j - dj]$. Under the hypothesis that each cell is independent, the predicted occupation of each cell is computed as follows:
$$\hat{P}_c(Occ) = \sum_n P_{c_a(n)}(Occ) \cdot P_{c_a(n)}(v_n) \qquad (1)$$
The predicted probability distribution function of velocity of a cell c is obtained by a normalisation over all velocity probability values $P(v_n)$ of each $c_a(n)$;
3) Sensor data are acquired and an observed occupation grid is built according to the probabilistic observation model;
4) The grid is updated following the Bayes rule:
$$P_c(Occ|z(t)) \propto P_c(z(t)|Occ) \cdot \hat{P}(Occ) \qquad (2)$$
5) The grid is searched for clusters: first the 4-connection recursive algorithm is applied, than each cluster is checked for coherent velocity profiles. In case of two or more groups of cells with coherent velocity, the cluster is divided again. Each cell is given a cluster index and a velocity profile for each cluster is calculated according to the estimation of each cell;
6) Back to step 2.

For further details the interested reader may refere to the original papers [14]. Fig. 1(a) shows a simulated environment: the cycab is equipped with a laser range finder and perceives two cars: the nearest moving from left to right and another just behind, moving from right to left. Fig. 1(b) shows the dynamic occupancy grid computed: red stays for high probability of occupation, while blue is for low probability. Fig. 1(c) shows the clusters found on the grid, correspondent to the two cars.

B. *Linear Velocity Obstacle*

Here we describe the classical approach to VO in terms that could help the understanding of the cell-to-cell approach; the original algorithm has been introduced by Fiorini and Shiller in [9].

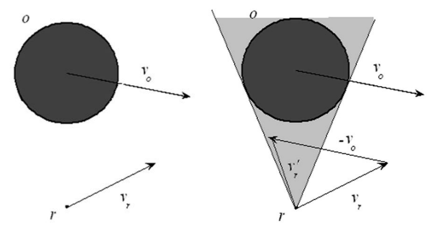


Fig. 2. Collision Cone for a punctual robot and a circular obstacle with linear velocity v_o; v_r is in collision.

Lets consider a punctual robot r in $[x_r, y_r]$ free to move in the 2D plane, and an obstacle o of arbitrary shape, with centre in $[x_o, y_o]$ and constant linear velocity v_o . With this definition of the robot, the configuration space (i.e. the space where each point corresponds to a configuration of the robot and obstacles correspond to configurations in collision) is equivalent to the Euclidean space. The velocity space is defined as the configuration space where linear velocities are described by vectors attached to the centre of objects. The idea is to work directly in this space and determine the set of all linear velocities that lead the robot to a collision in future time (Velocity Obstacle). Let's then define the Collision Cone CC_{ro} of the robot r relative to the obstacle o as the set of all relative velocities $v_r' = (v_r - v_o)$ that leads the robot in collision with o in future time:

$$CC_{ro} = \left\{ v_r' | \exists t > 0, (x_r + v_r' \overrightarrow{i} \times t, y_r + v_r' \overrightarrow{j} \times t) \in o \right\} \tag{3}$$

where $\overrightarrow{i}$ and $\overrightarrow{j}$ are the unity vectors of x and y directions, respectively.

The CC_{ro} is the positive angle with vertex in (x_r, y_r) and rays the right and left tangent to object o . To know if a velocity v_r is in collision with the obstacle o it is sufficient to consider the relative vector $v_r' = (v_r - v_o)$ and to verify if it points in the CC_{ro} , i.e. check if the extension of the vector in the positive direction intercepts the obstacle.

The velocity obstacle VO_{ro} is obtained translating CC_{ro} by the obstacle velocity v_o : all and only the velocities v_r pointing outside the cone are collision free. If more than one obstacle is present in the environment, it is sufficient to consider the union of each velocity obstacle [9] :

$$VO_r = \bigcup_{k=1 \ldots K} VO_{ro_k} \tag{4}$$

If the robot is circular with centre in (x_r, y_r) , the corresponding configuration space is given by a punctual robot in (x_r, y_r) and all the obstacles enlarged by the radius of the robot. Under the hypothesis of circular robot and obstacles, uncertainty in radius and in velocity can be taken into account [1].

C. Discussion on the chosen methods

To perform a safe navigation in a unknown or partially known dynamic environment, the mobile robot has to rely on a feasible representation of the world constantly updated according to the sensor observations and that allows to predict the future state of the system with some level of confidence. Many approaches use a list of objects and corresponding tracks and velocities which are considered a priori known or are learned in an off-line phase. The major drawback of these methods is that they are suitable only in industrial controlled environments, where a deterministic and complete knowledge on other agents is available. A second class of methods rely on an on-line estimation of the position and velocity of each object. These methods lie in general on a multi-target tracking algorithm and a data association technique which can encounter problems in cluttered environments and do not face the uncertainty due to the unobserved space. The advantage of considering a dynamic occupancy grid is that the robot maintains a full probabilistic information about the present occupation of the space and an estimation of the velocity of each occupied cell in the spatial grid. The absence of high level models makes the robot able to cope with unexpected situations and previously unknown obstacles. Furthermore, observations coming from different sensors can be directly integrated into the grid, so that the method is easy adaptable to different mobile bases.
The cell-to-cell approach to the linear velocity obstacles allows to reduce the hypothesis of the method, taking into consideration robot and obstacles of whatever shape and whatever discretized approximation of uncertainty in position and velocity of the obstacles. In contrast with the worst case approaches [16], the non observed space contributes directly to the computation of the probability of collision, leading to a full probabilistic framework.

III. THE DISCRETE PROBABILISTIC VELOCITY OBSTACLES

In this section we explain in detail the developed algorithm. Paragraph III-A describes the generalisation of velocity obstacles to the cell-to-cell approach. Paragraph III-B explains how the probability of collision in time is computed. Paragraph III-C, finally, details how the control input is chosen for the obstacle avoidance.

A. Cell-to-cell approach

The VO approach explained in the previous section is a geometric method that determines if a linear velocity leads the robot to a collision in the future. In order to generalise the method for a probabilistic approach and to the input provided by an occupancy grid, we developed a cell-to-cell approach. In this paragraph we make reference to a deterministic representation: the occupation of cells considered is $P(Occ) = \{0, 1\}$ and the velocity is a priori known: $P_c(v_o) = 1, P_c(v_n) = 0 \; \forall n \neq o$. The grid is relative to the robot.
Lets consider the robot and the obstacles as clusters of occupied cells. The velocities we study for the robot are integer linear velocities $v_n = [i \cdot \frac{q}{\tau}, j \cdot \frac{q}{\tau}]$ where $i, j \in \mathbf{N}$. The search space is reduced to the velocities reachable within the next time step: dynamic and kinematic constraints as a maximum acceleration value and a maximum and

minimum velocity in each direction are specified. Following the framework detailed in the previous section, a velocity $(\Delta i_r, \Delta j_r)$ leads the robot to a collision if it belongs to at least one of the $VO_{p_r c_o}$ between one of the points of the robot p_r and an occupied cell of the grid c_o. Given $[\Delta i_o, \Delta j_o]$ the velocity of an obstacle in the space, and $[\Delta_i, \Delta_j]$ an admissible velocity of the robot, each relative velocity $(\Delta' i, \Delta' j) = (\Delta i - \Delta i_o, \Delta j - \Delta j_o)$ is considered. Reasoning in the velocity space, this velocity corresponds to the vector attached to $p_r = [x_r, y_r]$ pointing $[x_r + \Delta' i, y_r + \Delta' j]$. As shown in Fig. 3(a), this velocity belongs to the VO relative to p_r iff there is at least an occupied cell with velocity $(\Delta i_o, \Delta j_o)$ in the positive direction of the extension of the velocity vector.

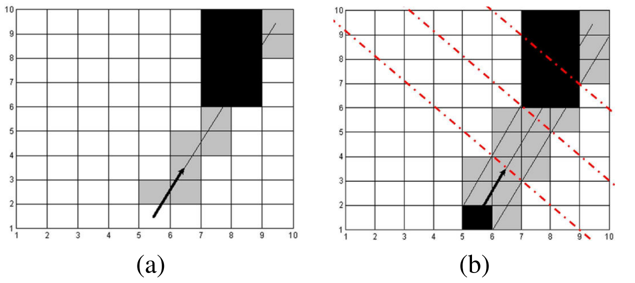


Fig. 3. Grey cells are searched for obstacles; black cells are occupied. A point of the robot and a cell are respectively considered in (a) and (b) image. The arrow is the considered relative velocity while the red dotted lines delimit searching areas for different times to collision.

To consider all the points in the robot-cell centred in $(x_r, y_r\ )$ we have to check all the cells which fall (also partially) between the two parallel lines tangent to the cells centred respectively in (x_r, y_r) and $(x_r + \Delta' i, y_r + \Delta' j\ )$ in the positive direction. This region is the search obstacle region of velocity v'_r relative to cell r and we denote it $SO(v'_r, r)$. To compute the probability of collision of the relative velocity at a given time instant t in the future, we consider the set $SO_t(v'_r, r)$ that should be traversed by the robot in the interval $[t-1, t]$. The considered velocity is in collision if for at least one of the cell of the robot it is found one occupied cell with velocity $(\Delta i_o, \Delta j_o)$. It is possible to introduce some simplification. For each velocity v_r :

- It is sufficient to consider just the 4-connected cells on the contour of the robot, where collisions occur first.
- We can consider just the last contour cells in the velocity direction.

For what concerns the number of velocities to study, the search space is reduced to the velocities that can be reached within the next time step. This dynamic window is centred around the actual velocity of the robot and is limited by the maximum acceleration that the motors can exert and eventually the maximum allowed velocity in function of the direction.

B. Compute the probability of collision

As detailed in Section II-A, the environment is represented by a dynamic occupancy grid. Each cell stores a probabilistic estimation of its state:

- a value of probability of occupation $P(Occ)$;
- a probabilistic distribution function on a histogram of possible velocities $P(v_n)$, $n = 1...N$;
- an index $k = -1, 0, \ldots, K+1$, where K is the estimated number of obstacles in the space (clusters of the grid); cells considered as free are given -1 index, cells occluded or not reached by the sensor range are given index 0; cells indicating the robot are given index $K+1$.

Given a robot velocity v_r and an obstacle velocity v_n, the probability of collision of a cell r with a cell o in the $SO(v_{r'}, r)$ is:

$$P_{coll}(v_r, v_n, r) = P_o(Occ) \cdot P_o(v_n) \tag{5}$$

as $P_r(Occ) = 1$. Considering the whole robot dimension, the maximum probability of collision in the interval $[t-1, t]$ is kept for each object k :

$$P_{coll}(v_r, k, v_n) \quad = \quad \max_{o \in O} P_o(Occ) \cdot P_o(v_n) \cdot \delta(k_o) \tag{6}$$

where o is each cell in $SO_t(v_{r'}, r)$ and $\delta(k_o) = 1$ if $k_o = k$, 0 otherwise.
To compute the probability of collision $P_{coll}(v_r)$ of the absolute velocity at time instant t, all the possible velocities of obstacles have to be considered. This value is computed as follows: for collisions with the same obstacle k the probability is given by the sum of the probability of each velocity, as $P(v_i|v_j) = 0$ for each $i, j = 1...N$ and $i \neq j$:

$$\begin{aligned} P_{coll(k)}(v_r) &= P_{coll(k)}(v_r, v_1) \vee \ldots \vee P_{coll(k)}(v_r, v_N) \\ &= \sum_{n=1\ldots N} P_{coll(k)}(v_r, v_n) \end{aligned} \tag{7}$$

If the collisions considered are due to different obstacles, the total probability is given by:

$$P_{coll}(v_r) = 1 - \sum_{k=1\ldots K} (1 - P_{coll(k)}(v_r)) \tag{8}$$

Both equations 7 and 8 are presented and used in the PVO approach [1].

A cumulative probability of collision from time 0 to the time step t under investigation is recursively computed. Applying a velocity v_r from present to t leads to a collision if there is a collision in the interval $[0, t-1]$ or if there is a collision at time instant t:

$$P_{0\ldots t}(v_r) = P_{0\ldots t-1}(v_r) + (1 - P_{0..t-1}(v_r)) \times P_t(v_r) \tag{9}$$

with the hypothesis that $P_{coll,0} = 0$.
Fig. 4 shows how the computation of the probability of collision in time reflects the uncertain information about the environment. We simulated the input that could be provided by a distance sensor as a laser range finder or a radar. A maximum range of $20 \cdot q$ is considered. Since we are working with a probabilistic representation, each cell has in general a positive probability of occupation: the free space scanned by the sensor is characterised by a probability of occupation that is nearly 0 while not sensed environment has $P(Occ) = 0.5$.

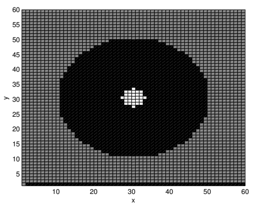
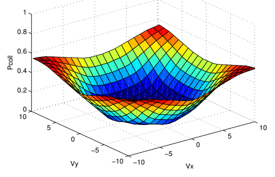
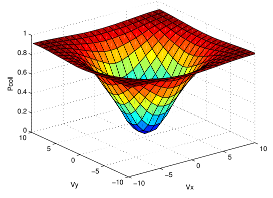
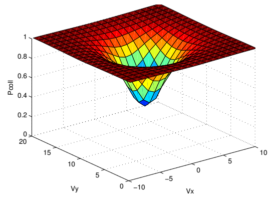

(a)

(b)

(c)

(d)

Fig. 4. (a) Simulated occupancy grid: the robot in the centre perceives free space all around, with limited range. (b) The probability of collision for each velocity of the robot considering T=2, (c) T=4, and (d) T=5.

For what concerns velocities, on each cell we will have a pdf more and more extended as the prediction is less reliable: in this example, free cells present uniform distribution over velocities in $v_x = [-3, 3]$, $v_y = [-3, 3]$. Cells that fall out of the grid are given $P_o(Occ) = 0.5$ and a uniform pdf over velocities. Fig. 4(a) represents an occupancy grid where the robot, in the centre, observes the free space around. The collision probability has been studied for velocities of the robot in the interval $v_x = [-10, 10]$, $vy = [-10, 10]$. Fig. 4(b), (c) and (d) are plots of the computed values respectively for $T = 2, T = 4$ and $T = 5$ time steps. The probability of collision is bigger for bigger velocities in each direction and it grows with the time of application: also if the robot stands still ($v_x = v_y = 0$) the probability of collision grows with time as the hypothesis that some unseen obstacle could go toward the robot is considered. Also in the case of no obstacles in the space, the robot will not move too fast if its perception is limited to a short range or some portion of the space is occluded.

C. Choice of the control input

In the dynamic and probabilistic case, the navigation of the mobile robot has to attend two major issues: minimise the risk of collision and reach the goal position. The method described in the previous section gives us a tool to compute the probability of collision in time for each admissible linear velocity of the robot, but it is not enough to perform safe navigation. We consider the robot safe if it can stop before running into a collision. This means a velocity v can be applied for an interval ϵ if the robot will not run into collision up to:

$$T_{safe}(v) = \epsilon + T_{brake}(v) \tag{10}$$

where $T_{brake}(v)$ is the minimum time to stop applying the maximum negative linear acceleration. To have an estimation of the time to collision a threshold of probability of collision is a priori chosen. This threshold defines the maximum risk we want to keep while navigating and we call it P_{safe}. We call T_{pred}, the interval of time for which the hypothesis of constant motion models is reliable. For a given velocity v, the probability of collision is recursively computed for each time step t; when $P_{coll,0..t}(v) > P_{safe}$ then the time of collision is estimated as the minimum between t and T_{pred}:

$$T_{coll}(v) = min(T_{pred}, t \,|\, P_{coll,0..t}(v) > P_{safe},) \tag{11}$$

If $T_{coll}(v) <= T_{safe}(v)$, the velocity is considered dangerous and discarded, otherwise it'll be considered safe enough to be applied.

For each time step, the admissible velocities of the robot are computed taking into account its kinematic and dynamic constraints. For each velocity, the next robot position and heading are computed, so to calculate a utility value and lead the robot toward the goal:

$$U(v) = 1/(dist(Robot, Goal, v) \tag{12}$$

In the simple case, the function $dist(Robot, Goal, v)$ is just the Euclidean distance between the future robot position and the goal location; in presence of local minima however, or if some different optimisation parameter is considered, a different distance function could be defined in a previous phase of motion planning. The velocity with the maximum utility is considered first. $T_{safe}(v)$ and $T_{coll}(v)$ are computed. If the velocity is found to be safe enough it is chosen as the next control for the robot, otherwise it is discarded. The algorithm is iterated until a safe velocity is found. The chosen control is then applied and the algorithm is iterated. If none of the admissible velocities is safe enough, the robot performs an emergency braking manoeuvre, i.e. reduces at minimum the module of its velocity.

IV. Simulation Results

We implemented the algorithm in a Matlab application and tested it in various simulated environments. The following paragraphs show and discuss the obstacle avoidance strategy in two scenarios and at the variation of the perception capabilities of the robot.

A. Occlusion

This experiment shows how the occlusion influences the robot strategy. Fig. 5(a) shows the complete simulated environment: the robot is the circle at the bottom and has to go up toward the goal. The initial velocity of the robot is 0. A circular obstacle is moving in the y direction with velocity $v_y = 3$. The robot can't steer and its admissible velocities are $v_x = 0$, $0 \leq v_y \leq 5$; a maximum acceleration of $2 \cdot q$ per time step in the y direction is considered. The velocities that can be represented in the dynamic occupancy grid are integer values in the interval $v_x = [-4, 4], v_y = [-4, 4]$; a maximum time of prediction is fixed a $T = 5$ and the probability threshold is fixed at 0.1. Two different sensor input are

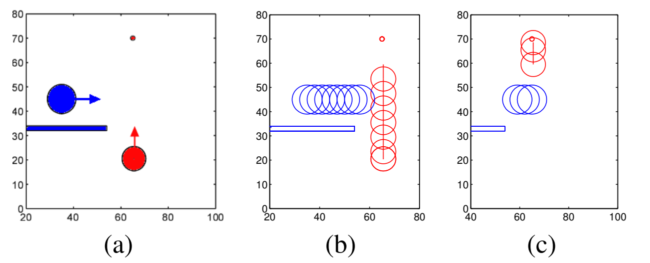


Fig. 5. The robot has to move up to the goal, while an obstacle comes from the left. (a) A perfect knowledge of the world is simulated; (b) the robot accelerates and passes before the obstacle, reaching the goal (c).

simulated: in the first case (Fig. 5) the input grid represents the whole environment: the velocity of the obstacle is known with certainty and there are not occluded zones. The robot perceives the moving obstacle and knows its velocity, so it can safely accelerate at maximum speed and reaches the goal passing before the obstacle (Fig. 5(b), (c)). In the second

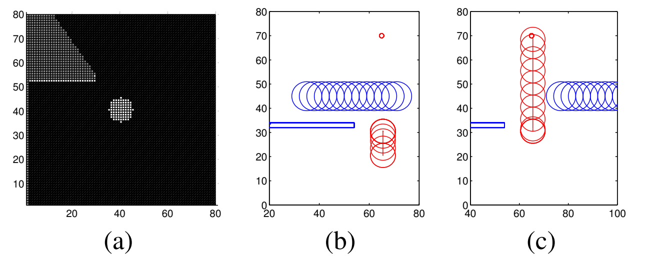


Fig. 6. (a) Occupancy grid in the case of occlusion: the robot can't observe the moving obstacle. (b) The robot maintains a low speed approaching the static obstacle and brakes when it sees the moving one, letting it pass before reaching the goal (c).

experiment (Fig. 6) a distance sensor is simulated: the static obstacle hides the moving one. The robot maintains a lower speed as the probability of collision given by the occluded space forbids higher speeds. The robot arrives later at the crossing and brakes to let the obstacle pass (Fig. 6(b)). Then the robot passes also and reaches the goal (Fig. 6(c)).

B. Crossroad

In this example the robot faces a crossing. Fig. 7 shows the complete simulated environment: the robot is the circle at the bottom and has a positive velocity in the y direction. Static obstacles delimit the environment and two other obstacles, respectively at the right and left of the image move toward the centre of the crossing. The robot goal location is on the upper part of the crossing. The admissible velocities for the robot are $-5 \leq v_x \leq 5$, $0 \leq v_y \leq 5$ and an admissible acceleration of $2 \cdot q$ per time step in both x and y direction is considered. As in the previous paragraph, the velocities that can be represented in the dynamic occupancy grid are integer values in the interval $v_x = [-4, 4]$, $v_y = [-4, 4]$; a maximum time of prediction is fixed a $T = 5$ and the probability threshold is fixed at 0.1. In the first experiment (Fig. 8(a)) the input grid represents the whole environment: the velocity of the obstacles is known with certainty and there

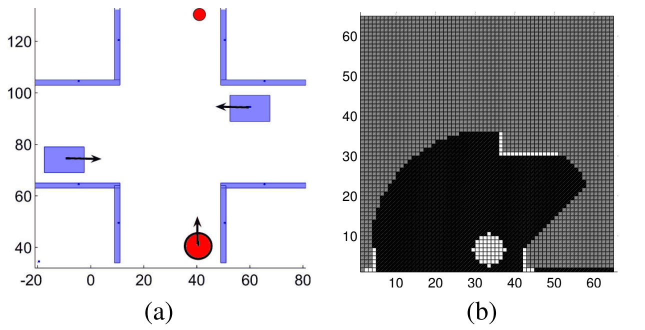


Fig. 7. (a) The robot is the circle at the bottom. It faces a crossroad with two obstacles moving in opposite direction. The goal location is the point at the top of the image. (b) Input occupancy grid in case of limited range: entering the crossing, the robot doesn't see the moving obstacle on the left.

are not occluded zones. In this case the robot performs the obstacle avoidance passing near the obstacles and reaching the goal with a short trajectory: it deviates from the straight line just to avoid the obstacle coming from the right. In the second experiment (Fig. 8(b)) a Gaussian uncertainty on obstacles velocity is simulated. The medium value is the real velocity value, while the standard deviation is $\sigma = 1.5 \cdot q/\tau$. Since the beginning the robot tries to keep its trajectory further away from obstacles; the path results longer as the robot performs wider curves to avoid collisions. In the third experiment (Fig. 8(c)) a distance sensor input is simulated. The visible distance is limited by a short range ($30 \cdot q$) and the occluded zones hide obstacles and their shape (Fig. 7(b)). The obstacles velocities are known with the same Gaussian uncertainty of the second experiment. The robot goes slower; as it approaches the crossing it still doesn't see the obstacle on the left and enters the crossing; when it perceives the obstacles it is forced to escape from it and reaches the goal only after waiting the right obstacle to leave the crossing. It is however able to reach the goal and, more important, it reacted appropriately to the unexpected obstacle. Tests with a lower range cause the robot to perform an emergency manoeuvre (brake and stop) before facing the crossing.

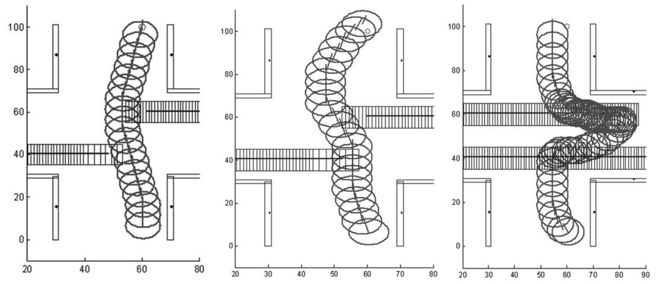


Fig. 8. The robot trajectory with (a) precise estimation of obstacles velocities; (b) Gaussian uncertainty on obstacles velocities; (c) Gaussian uncertainty and limited visibility range.

V. Conclusions and Future Work

In this paper we proposed a method to compute the probability of collision in time for linear velocities of the robot and a reactive algorithm to perform obstacle avoidance in dynamic uncertain environment. The novelty of the method consists in the explicit consideration of uncertainty in the perception system, rising both from the errors and noise in the model of the environment and from occlusions, sensor range, noise and failures. The input to the algorithm is an occupancy grid, it is highly reactive to the environmental changes and is well suited to be applied in various sensor settings. The developed algorithm computes a probability to collision in time working directly in the velocity space. The dynamic and kinematic constraints of the robot are so taken into account and the study is reduced to the current reachable velocities. The case of holonome robot and linear constant motion of the obstacles has been analysed in this paper: future work will deal with the generalisation of the method following the Non Linear Velocity Obstacle approach [10], [17]. For what concerns the navigation algorithm, the simulation results show that the robot is able to navigate among static and moving obstacles facing unexpected situations and moving toward the goal. The robot adapts its behaviour to the quality of information received and modifies its trajectory according to the incomplete and uncertain perception of the environment. However, due to the reactive nature of the algorithm and to the limited knowledge of the environment, there is no guarantee that the robot achieves the goal or that it doesn't put itself in emergency conditions that could have been avoided. In order to achieve a better performance, we plan then to integrate the information of the probability and time to collision in a motion planning algorithm able to face more complex scenarios, combining a priori knowledge and on-line perception, and to test the method on a real mobile base (Cycab [18]).

REFERENCES

[1] B. Kluge and E. Prassler, "Reflective navigation: individual behaviours and group behaviours," in *IEEE International Conference on Robotics and Automation, ICRA*, 2004.

[2] J. C. Latombe, *Robot Motion Planning*. Dordrecht, The Netherlands: Kluwer, 1991, vol. SECS 0124.

[3] T. Fraichard and A. Scheuer, "Car-like robots and moving obstacles," in *IEEE International Conference on Robotics and Automation, ICRA*, 1994.

[4] M. Seder, K. Macek, and I. Petrovic, "An integrated approach to real-time mobile robot control in partially known indoor environments," in *In Proceeding of the 31st Annual Conference of the IEEE Industrial Electronics Society*, 2005.

[5] C. Stachniss and W. Burgard, "An integrated approach to goal-directed obstacle avoidance under dynamic constraints for dynamic environments," in *IEEE International Conference on Intelligent Robots and Systems, IROS*, 2002.

[6] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, Mar. 1997.

[7] R. Simmons, "The curvature-velocity method for local obstacle avoidance," in *IEEE International Conference on Robotics and Automation, ICRA*, 1996.

[8] Y. K. Nak and R. Simmons, "The lane-curvature method for local obstacle avoidance," in *IEEE International Conference on Robotics and Automation, ICRA*, 1998.

[9] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *Int. Journal of Robotics Research*, no. 17, pp. 711–727, 1998.

[10] F. Large, "Navigation autonome d'un robot mobile en environnement dynamique et incertain," Ph.D. dissertation, Université de Savoie, 2003.

[11] T. Fraichard and H. Asama, "Inevitable collision states. a step towards safer robots?" in *IEEE International Conference on Intelligent Robots and Systems, IROS*, 2003.

[12] E. Owen and L. Montano, "A robocentric motion planner for dynamic environments using the velocity space," in *IEEE International Conference on Intelligent Robots and Systems, IROS*, 2006.

[13] C. Coué, C. Pradalier, C. Laugier, T. Fraichard, and P. Bessière, "Bayesian occupancy filtering for multitarget tracking: an automotive application," *Int. Journal of Robotics Research*, vol. 25, no. 1, pp. 19–30, January 2006. [Online]. Available: http://emotion.inrialpes.fr/bibemotion/2006/CPLFB06

[14] C. Coué, "Modèle bayésien pour l'analyse multimodale d'environnements dynamiques et encombrés: application à l'assistance à la consuite automobile en milieu urbain." Ph.D. dissertation, Inst. Nat. Polytechnique de Grenoble, 2003.

[15] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, pp. 46–57, June 1989.

[16] R. Benenson, S. Petti, M. Parent, and T. Fraichard, "Integrating perception and planning for autonomous navigation of urban vehicles," in *IEEE/RSJ International Conference on Intelligent Robots and Systems,IROS*, 2006.

[17] Z. Shiller and S. Large, F.and Seckavat, "Motion planning in dynamic environments: obstacles moving along arbitrary trajectories," in *IEEE International Conference on Robotics and Automation, ICRA*, 2001.

[18] C. Pradalier, J. Hermosillo, C. Koike, C. Braillon, P. Bessière, and C. Laugier, "The cycab: a car-like robot navigating autonomously and safely among pedestrians," *Robotics and Autonomous Systems*, vol. 50, no. 1, pp. 51–68, 2005.