RIS : A Framework for Motion Planning among Highly Dynamic Obstacles

Pierre de Beaucorps, Anne Verroust-Blondet, Renaud Poncelet, Fawzi Nashashibi

RIS : A Framework for Motion Planning among Highly Dynamic Obstacles

Pierre de Beaucorps*, Anne Verroust-Blondet*, Renaud Poncelet* and Fawzi Nashashibi*

***Abstract*— We present here a framework to integrate into a motion planning method the interaction zones of a moving robot with its future surroundings, the *reachable interaction sets*. It can handle highly dynamic scenarios when combined with path planning methods optimized for quasi-static environments. As a demonstrator, it is integrated here with an artificial potential field reactive method and with a Bézier curve path planning. Experimental evaluations show that this approach significantly improves dynamic path planning methods, especially when the speeds of the obstacles are higher than the one of the robot. The presented approach is used together with a global planning approach in order to handle complex static environments in presence of fast-moving obstacles. When the ego vehicle is not holonomic the presented approach is able to take dynamic constraints into account, which improve the prediction accuracy.**

I. Introduction

Motion planning in a dynamic environment is of great importance in many robotics applications. In the context of an autonomous mobile robot, it requires computing a collision-free path from a start to a goal among moving and static obstacles. Numerous approaches have been proposed to solve this problem. We will focus here on a small subset of methods and let the reader consult a comprehensive study of the problem and a survey on the main techniques in [1]. Additional references can be found in [2] and [3].

Early approaches used the configuration space, which describes the set of possible transformations that could be applied to the robot moving among obstacles and extended it to a space-time configuration space in [4], [5], considering the motion of the dynamic obstacles. In [4] the configuration space-time is represented approximately by a list of configuration-space slices and the robot is supposed to move at a speed that is greater than any of the obstacle's speeds. Some works focus on computing a trajectory that validates a safety guarantee, regardless of the future movement of the moving obstacles. This guarantee comes with some drawbacks, which is to be expected since [6] shows that motion planning in a dynamic environment is NP-hard, even when the obstacles are moving with constant linear velocities without rotation. In [7] the motion of the moving obstacles is represented by disks that grow over time. They compute a time-minimal path among these growing disks, but it is required that the robot has a higher maximum speed than any of the moving obstacles. In [8] the exact reachable set of the robot is computed analytically by solving a Hamilton-Jacobi partial differential equation. The safety guarantee comes from the worst-case scenario where the other vehicle is expected to pursue the robot, trying to achieve a collision. This worst-case scenario quickly appears to be intractable in more complex scenarios when the density of obstacles increases.

Other methods proposed to reduce the search space of the possible velocities of the robot, such as the dynamic window approach [9], the collision cone [10] and the velocity obstacles [11], [12], [13], among others. Note that in the experiments involving velocity obstacles, the speeds of the obstacles do not exceed the speed of the robot.

Potential field methods [14] have been extended to dynamic obstacles in [15], [16] and [17]. In [15] the velocity of the obstacle is included in the definition of the repulsive potential function. In [16] the potential functions take into account both the relative position and the relative velocity of the robot with respect to the target and obstacles. In [17] the potential functions are not modified but a new formulation of the robot's planned velocity is introduced: it is determined by relative velocities and relative positions among robot, obstacles, and target. The combination of a fuzzy logic engine (FL) with an electrostatic potential field (EPF) path planner is proposed in [18]. For highly dynamic environments, another approach integrating stochastic reachable sets (SR sets) with artificial potential fields has been proposed in [19] for stochastic obstacles moving either on line segments or on arcs and extended in [20] for obstacles that can switch between line- and arc-following dynamics. In [15], [16] and [17], at each time t the artificial potential field based methods consider only the instantaneous positions of the obstacles and not the future ones to compute the potential fields even if the obstacles are moving, whereas in [19] a prediction of the motion of the obstacles, the future distribution of the stochastic moving obstacles, is described by SR sets that are used to generate potential fields.

We think that when the speeds of the robot and the obstacles are nearly the same, integrating only the speed and not the predicted positions of the obstacles in the artificial potential field-based methods can lead to satisfying solutions. But this cannot stand anymore when obstacles speeds are very high compared to the one of the robot. Thus we propose to integrate into a motion planning process a short-time prediction of the possible collisions of the robot with its surrounding obstacles in order to guide the robot more safely.

To the knowledge of the authors, the reachable sets (see [21]) are mostly used in the framework of safety guarantee, as in [8] and [22]. We intend here to use them in another framework: to find a proper path in complex scenarios,

* INRIA Paris, 2 rue Simone Iff 75012 Paris, France pierre.de-beaucorps@inria.fr

where no guarantee can be provided. This is achieved by considering the potential interactions between the robot and the predicted positions of obstacles, with no prior assumption on the future path of the robot but an assumption on its speed norm. Put another way, we introduce the *reachable interaction sets* (RIS) which are used in a motion planning framework and tested in an extensive comparative benchmark.

This paper is structured as follows: the next section presents the reachable interaction sets for the ego vehicle and moving obstacles and their integration into several motion planning processes. Experimental evaluations carried out on different static environments and several variations of trajectories, shapes, and densities of moving obstacles are presented in section III. Finally, a conclusion is given in section IV.

II. The Approach

Let us assume that at any time there exists a short time interval t_H during which the position and the velocity of each obstacle is known and its motion can be reasonably predicted with a constant yaw rate prediction. If we fix a speed value for the robot and if we consider it has a constant yaw rate during this short time interval, we are able to predict if a collision between the robot and the obstacles may occur and to compute the regions where the robot may interact with its neighboring obstacles. This is what we call the *the reachable interaction sets* (RIS). Then, instead of considering the actual positions of the obstacles, the robot should avoid the regions composing the RIS while moving during this short time interval.

A. Reachable Interaction Sets

1) Holonomic ego robot:

For clarity purpose, the *reachable interaction sets* will be introduced by considering the interaction between a holonomic ego robot A reduced to a point having a constant speed $v_A = \|\vec{v_A}\|$ and a dynamic obstacle B having a convex polygonal shape P_B and moving with a known trajectory given by its local frame $\{(x_B(t), y_B(t), \theta_B(t)), t \in [0, t_H]\}$, where t_H is the time horizon. The hypothesis that robot A can be reduced to a point is actually a dual version of a more general situation where both objects have a convex polygonal shape. In this case, we have to compute the Minkowski sum over the shapes of the two polygonal objects.

Let us assume, without loss of generality, that the initial position of A is at the origin and that B has position $p_B = (x_B, y_B)$ at $t = 0$. As A moves with a constant speed, the position of $A(t)$, the center of A at t, is located within a circle $C(O, tv_A)$ of radius $t \cdot v_A$ centered at the origin O. As B moves with a known trajectory during $[0, t_H]$, the obstacle $B(t)$ corresponds to the polygon P_B translated and rotated according to its trajectory. Collisions between A and B can occur when $A(t)$ and $B(t)$ overlap. Then $RIS_{AB}(0, t_H)$ is defined by:

$$RIS_{AB}(0, t_H) = \{p \in A(t) \cap B(t) \text{ for } t \in [0, t_H]\} \quad (1)$$

To interpret geometrically equation (1), let us add the time dimension and consider the 3D space-time (X, Y, t). Points (x,y,t) of the available future positions of A belong to a right circular cone having the origin $(0, 0, 0)$ as apex and $C(O, t_H \cdot v_A)$ as base in the plane (X, Y, t_H). Points (x,y,t) of the predicted future positions of B belong to a volume $\mathcal{B}$, extrusion of a polygon along the 3D curve given by the trajectory of B, volume swept between the polygon $P_B(0)$ in the plane $(X, Y, 0)$ and the polygon $P_B(t_H)$ in the plane (X, Y, t_H) (see Fig. 1). Then $RIS_{AB}(0, t_H)$ corresponds to the projection of the intersection of the cone and $\mathcal{B}$ on the 2D plane $(X, Y, 0)$. It may be empty or composed of one or multiple disconnected regions (cf. Fig. 1).

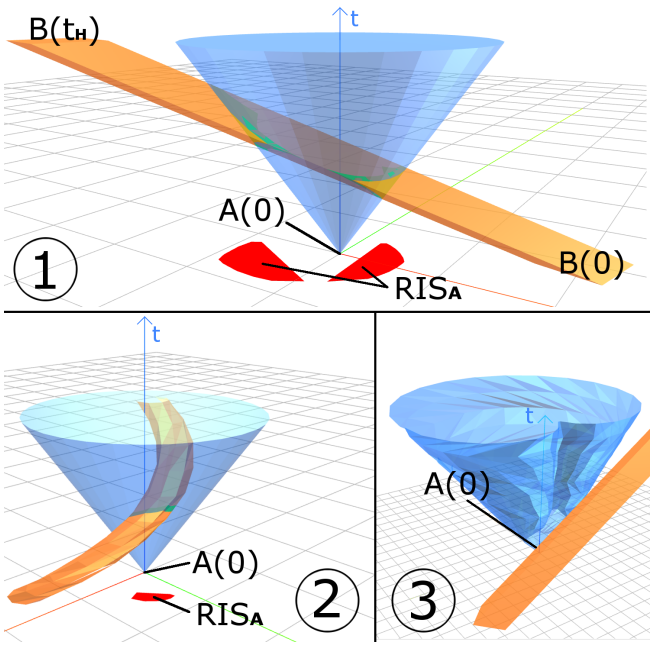


Fig. 1: Three examples of $RIS_{AB}(0, t_H)$ regions when A is holonomic (1 and 2) or nonholonomic (3). On the plane $t = 0$, $RIS_{AB}(0, t_H)$ is drawn in red, A and B being respectively the robot and the obstacle. $RIS_{AB}(0, t_H)$ can be empty (3), composed of multiple disconnected regions (1) or reduced to only one region (2).

Thus, if $\mathcal{O}$ is the set of obstacles surrounding A during $[0, t_H]$, we have: $RIS_A(0, t_H) = \cup_{Bin\mathcal{O}} RIS_{AB}(0, t_H)$. One can note that the time-limited reachable set of the robot that takes the obstacle region into account, denoted $R(x_0, \mathcal{U}_{\text{free}}, t_H)$ in [21], is equal to the complementary to $RIS_A(0, t_H)$ in $R(x_0, \mathcal{U}, t_H)$.

2) Nonholonomic dynamic constraints:

Let us consider another motion model for A and suppose that A is moving with dynamic constraints on acceleration and yaw rate as in the Dubins' vehicle model (cf. [21]):

$$\begin{cases} \dot{x} = \omega \cdot cos\,(\theta) \\ \dot{y}(t) = \omega \cdot sin\,(\theta) \\ \dot{\theta} = u \end{cases} \quad (2)$$

$$\omega = max\,(v_A + a_{max} \cdot t, v_A max) \quad (3)$$

Where u is the supposed yaw rate and ω is the longitudinal velocity of the ego robot. The conic surface representing the future positions of A in in 3D space-time (X, Y, t) is replaced by the surface $\mathcal{S}_\mathcal{A}$ build using equation 2 to compute the positions of (x, y, t) with $t \in [0, t_H]$ and u being any value over $[-u_{max}, u_{max}]$. In order to obtain a better prediction, we assume that the longitudinal velocity ω starts at current velocity v_A at $t = 0$ and then uses the maximum acceleration to reach the maximum available speed $v_A max$ at $t = t_H$ (cf. equation 3). If the surface $\mathcal{S}_\mathcal{A}$ contains self-intersections, we delete the internal part of $\mathcal{S}_\mathcal{A}$ before computing the intersection with the volume $\mathcal{B}$ corresponding to obstacle B. The obtained surface $\mathcal{S}_\mathcal{A}$ is shown in Fig. 1 (3). In practise, $\mathcal{S}_\mathcal{A}$ is computed numerically once for all for a range of different v_A values and cached for faster retrieval. As in the holonomic case, $RIS_{AB}(0, t_H)$ corresponds to the projection of the intersection of $\mathcal{S}_\mathcal{A}$ and $\mathcal{B}$ on the 2D plane $(X, Y, 0)$. A 3D interactive view of the generation of different RIS_{AB} is available at `http://deboc.fr/ris/`.

B. Motion planning using reachable interaction sets for collision avoidance

In order to simplify notations, in the rest of the paper the time parameter t will be implicit so that $RIS_{AB}(t, t_H)$, $RIS_A(t, t_H)$, $A(t)$, $\vec{v}_A(t)$ and $B(t)$ will be noted respectively RIS_{AB}, RIS_A, A, $\vec{v}_A$ and B; robot A will have a speed value $v_A \leq v_A max$ and $Dest$ will denote the destination point.

Our approach combines two planning algorithms: a global path planning that only considers static obstacles and a local motion planning method that adapts the trajectory computed by the global path planning to the dynamic obstacles nearby.

1) Global planning for static obstacles avoidance:
The global planning is performed only once per static environment. The path is built using the sampling-based method RRT* [23] for which only the static obstacles are considered. In fact, we can use any other method computing a collision-free path P_G to the destination $Dest$. The local motion planning algorithm uses the computed path P_G as a guide in order to avoid the static obstacles while computing its motion. At each time step, an intermediate target T belonging to P_G is selected and is used as a local destination instead of $Dest$ by the local motion planning method. The choice of T is made by computing the distance $dist_{AP}$ between A and P_G and $proj_A$, the point belonging to P_G and closest to A. The intermediate target T is the first point of P_G following $proj_A$ and positioned at a distance $d_T = max\,(maxDist - dist_{AP}, 0)$ from $proj_A$, where $maxDist$ is the distance that A can cover in a fixed number k of time-steps (in this work, $k = 40$).

2) Local planning for dynamic obstacles avoidance:
RIS computation is integrated into different local motion planning processes: a reactive method using artificial potential field (ris-apf), a planning method using Bézier splines (ris-bezier) and a hybrid method, mixing the two previous ones (ris-hybrid). These approaches follow the same set of steps: at each time step, RIS_A is updated with respect to the obstacles B of $\mathcal{O}$ and A as explained in the previous section, then one of the motion planning approaches is performed and finally, the desired velocity is provided to a controller which updates the position and velocity of A.
Let us now explain these three methods, assuming that RIS_A has been updated.

ris-apf: a static artificial potential field is used as a reactive control approach. Here, regions of RIS_A replace the obstacles B of $\mathcal{O}$. The definition of the repulsive and attractive potential forces (see equations 4, 5 and 6) are the same as the ones used in a standard artificial potential field method assuming a quasi-static environment as in [1].

$$\vec{F}_{rep}(B) = -K_{rep}\left(\frac{1}{d(A, RIS_{AB})^2} - \frac{1}{d_0^2}\right)\vec{u}_{A-RIS} \quad (4)$$

$$\vec{F}_{att} = K_{att}\frac{\vec{AT}}{\|\vec{AT}\|} \quad (5)$$

Where $\vec{u}_{A-RIS} = \frac{\vec{AP}}{\|\vec{AP}\|}$, P being the closest point on the boundary of RIS_A to A. The values of d_0, the influence range of the obstacles, K_{att} and K_{rep} are fixed empirically (Here $d_0 = 500$, $K_{att} = 10$ and $K_{rep} = 20000$).

$$\vec{F}_{total} = \vec{F}_{att} + \sum_{B in \mathcal{O}} \vec{F}_{rep}(B) \quad (6)$$

The new velocity of A is computed by this modified artificial potential field method. The speed value is bounded by $v_A max$. Sometimes A can intersect a region of RIS_{AB} without colliding any obstacle. Such a situation can happen when the prediction of obstacles movements was not accurate enough, due to some controller error, or when the straight path constant speed assumption for robot A required to compute RIS_A is not met (which occurs when the curvature of the planned path is too high). Then the corresponding repulsive potential force is used with the opposite direction in order to force robot A to exit the conflict zone before the collision.

ris-bezier: a path using a spline composed of one or two smoothly-joined cubic Bézier curves [24] is computed. Our goal here is to achieve reduced path curvature and G^1 continuity for the path joining A to T. In fact, this may not be satisfied for all obstacles configurations. Although naive, the ris-bezier approach is a proof of concept emphasizing the suitability of RIS in another planning framework with some constraints on path curvature.

Fig. 2.1 shows the different input parameters used at each time step: positions of A and T, $\vec{v_A}$, the current velocity of A and $\vec{u_T}$, the desired orientation from which the robot should approach the target. The first step of the path planning consists in building a Bézier curve $\mathcal{C}_0$ from A to T. It is defined by the control points (A, Ca, Ct, T) (see Fig. 2.2) where Ca is such that $\vec{ACa} = d \cdot \frac{\vec{v}_A}{\|\vec{v}_A\|}$ and Ct satisfies $\vec{TCt} = d \cdot \vec{u_T}$ with $d = min(d0, \frac{d(A,T)}{2})$, ($d0$ being empirically fixed to 40).

If $\mathcal{C}_0$ does not intersect RIS_A, it is used as a target path. Otherwise, $\mathcal{C}_0$ is replaced by a spline $\mathcal{C}_1$ composed of two

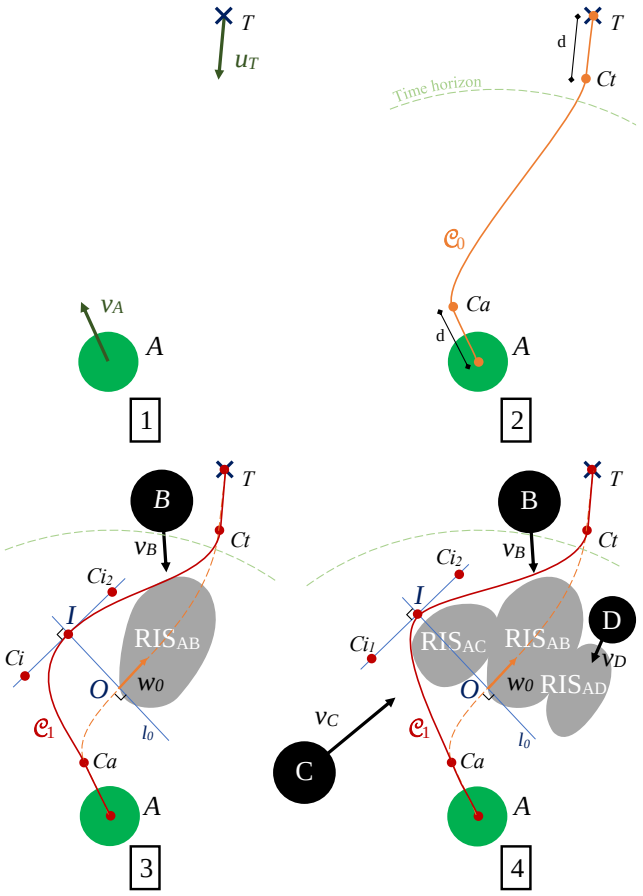


Fig. 2: The ris-bezier approach. *Time horizon* dotted curve represents the limit of the region reachable by A during the time horizon period. If not empty, RIS_A belongs to this region but target T can be outside. (1): the input parameters A, T, $\vec{v_A}$ and $\vec{u_T}$. (2): the initial curve $\mathcal{C}_0$, which is used as a path when RIS_A is empty or does not intersect it. (3) and (4): a new curve $\mathcal{C}_1$ is built when $\mathcal{C}_0$ intersects RIS_A.

Bézier curves defined by (A, Ca, Ci_1, I) and (I, Ci_2, Ct, T). $\mathcal{C}_1$ should be G^1 continuous at the joining point I and should not intersect RIS_A. Points I, Ci_1 and Ci_2 are computed as follows (cf. Fig. 2.3 and 2.4): let O be the first point of $\mathcal{C}_0$ intersecting RIS_A and $\vec{w_0}$ the tangent to $\mathcal{C}_0$ at O. I belongs to the line l_0 passing through O and perpendicular to $\vec{w_0}$, Ci_1 and Ci_2 belong to the line directed by $\vec{w_0}$ and passing through I. They are positioned at a distance d on both sides of I. Point I is the closest point to O on line l_0 such that $\mathcal{C}_1$ circumvent RIS_A. It is found by iterating over l_0.

If the distance between O and I is too large, the path curvature would be too high to be really efficient (in this study an empirical threshold is used, but a more precise comfort value could be computed, e.g. from a desired maximum lateral acceleration). The decision in this case is to force A to stop and wait for a better situation. When robot A is already inside RIS_A, $\mathcal{C}_0$ is used by default, since the decision to stop would be even more dangerous. These two special cases do not really define a proper path planning, thus the path will not be retained for the ris-hybrid approach described below. Once the final path is identified ($\mathcal{C}_0$ or $\mathcal{C}_1$), it is provided to a simple controller that will update the velocity of A to follow the given path.

ris-hybrid: this approach computes at each step the path planned by ris-bezier and uses it when it is defined. But in the case where no path can be found, the ris-apf approach is used to compute a target velocity instead of stopping the robot.

III. Experiments

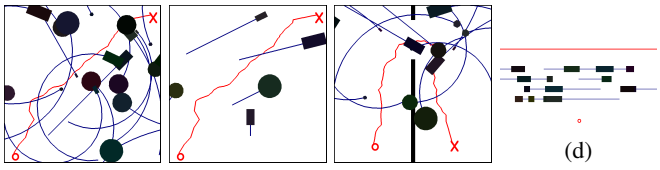
We used the following methods in our experiments: the three methods presented in the previous section (**ris-apf**, **ris-bezier** and **ris-hybrid**); a classical artificial potential field approach (**static-apf**) [14]; the artificial potential field approach of [16] for dynamic environments (**dynamic-apf**); and velocity obstacles approach (**vo**), using the VO algorithm adapted from the RVO2 C++ code [25] built from the ORCA algorithm [12]. We removed the reciprocal aspect from the ORCA algorithm so that the ego robot using VO approach does not expect other obstacles to perform any avoidance maneuver. The computation of *RIS* regions has been implemented numerically with a 3D mesh approximation of the cone and $\mathcal{S}_A$ surfaces.

A. Benchmark Environment

In order to compare these approaches, simulated experiments have been performed in three different environments:

- *'free'*: a 800x800 units area with moving obstacles with circular or straight trajectories and without static obstacles. A is starting at (50, 750) and must reach the destination (750, 50) (cf. Fig. 3 (a)).
- *'free-straight'*: a 800x800 units area with moving obstacles with straight trajectories and without static obstacles. A is starting at (50, 750) and must reach the destination (750, 50) (cf. Fig. 3 (b)).
- *'door'*: a 800x800 units area with two walls surrounding a door that must be passed through and with moving obstacles with circular or straight trajectories. A is starting at (200, 700) and must reach the destination (600, 700) (cf. Fig. 3 (c)).
- *'cross'*: a 1600x800 units area without static obstacles and with two flows of moving obstacles going straight, arranged as a 2+2 lanes road that must be crossed. A is starting at (800, 750) and must reach the upper line (x, 50) (cf. Fig. 3 (d)).

The ego robot A is a circular object of radius 30 units that can move with a maximum speed v_Amax of 4 units per step. Experiments presented in section III-B are conducted for different obstacles parameters that vary the number of moving obstacles and their speed range: the obstacles' speeds are randomly selected within $[0, v_Amax]$ unit/step for the *slower* speed range, and within within $[v_Amax, 2 \cdot v_Amax]$ unit/step for the *faster* one. Obstacles may overlap and do not react to the robot or to the other obstacles. When an obstacle hits the map boundaries, it bounces on the map frame (the sign of one of its velocity axis is modified),



(a) (b) (c) (d)

Fig. 3: Snapshots of the four test environments : *free*, *free-straight*, *door* and *cross*, with, respectively, 20, 5, 15 and 15 moving obstacles (their future trajectories are drawn in blue). The red "O" and "X" correspond respectively to the start and the destination positions, and the global static paths are drawn in red. As *cross* environment is slightly different, there is no global static path but only a global destination that is represented by a red line, since reaching any horizontal position at the top is interpreted as a success.

except in *cross* environment where it is moved to the opposite side of the map. Thus the density of obstacles remains the same during the whole process and the motions of the obstacles are entirely determined by their initial state. For each environment and each obstacles parameter, 100 initial states are randomly generated as follows:

- The shapes are randomly chosen, either circular ($10 \leq$ radius ≤ 60) or rectangular ($10 \leq$ height ≤ 60, width=height/2).
- The initial position is randomly distributed over the environment except around the starting position of A. In *cross*, obstacles must be positioned on a lane.
- The speed is randomly sampled into *slower* or *faster* speed ranges. In *cross*, speeds are sampled over range $[0, 2 \cdot v_A max]$ unit/step.
- in *free* and *door* environments, obstacles are given a constant yaw rate randomly sampled inside $[-\frac{\pi}{80}, \frac{\pi}{80}]$ rad/step.

The same experiments were also performed for a nonholonomic robot A in the same environments and the same sets of parameters. In addition to the maximum speed limit $v_A max$ of 4 units per step, A has a maximum acceleration of $1 unit/step^2$ and a maximum yaw rate of $\frac{\pi}{12} rad/step$. We retained only the *RIS*-based approaches using Bézier splines since they inherently take into account kinematic constraints. We compared *ris-hybrid* and *ris-bezier* using the *RIS* construction for the nonholonomic case as in section II-A.2 (called *ris-hybrid (dubins)* and *ris-bezier (dubins)*) with the two methods using *RIS* built for a holonomic robot as in section II-A.1 (called *ris-hybrid (cone)* and *ris-bezier (cone)*) and where the nonholonomic constraints are directly applied to the control and not integrated into any planning process.

B. Results Analysis

Let us compare the different approaches by examining their success rates (i.e. the number of setups where A reached T divided by 100) in the three environments.

Fig. 4 shows the curves corresponding to the success rates of the different methods for a holonomic vehicle with varying obstacles densities and *slower* speeds. The *faster* speeds case is shown in Fig. 5. Subfigures (a), (b) and (c) distinguish the *free*, *free-straight*, and *door* environments. For all environment, the presence of *faster* obstacles clearly decreases the performances of all approaches compared to the *slower* case, but it is interesting to notice that the relative gap of successes is lower for *RIS*-based approaches, as they handle better high-speed obstacles. In all cases, *ris-hybrid* is the best performing approach and, except *vo* in *door* with *slower* obstacles, another *RIS*-based approach is the second one. The *ris-bezier* approach is far from being optimized because of its naive rule to stop when no simple path as described in II-B.2 can be found. The *ris-hybrid* is the most successful approach, benefiting both from the efficient path of Bézier planning and the fast reactivity of a potential field method in conflict cases.

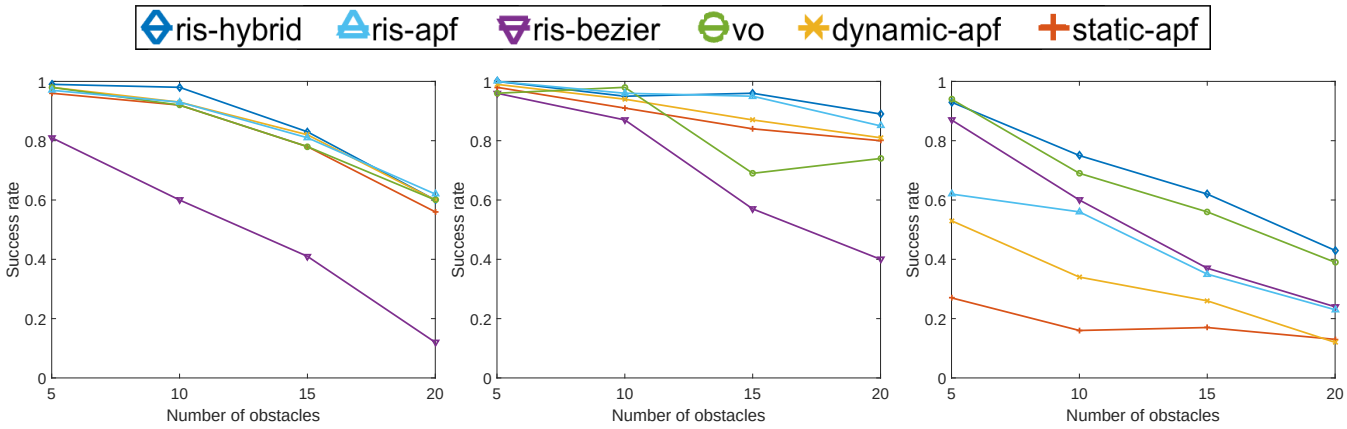
It is worth mentioning that *ris-bezier* is able to find a much shorter path than those found by any other approach, as can be observed in Fig. 6 where the path length displayed is normalized by the initial global path length.

Pictured in Fig. 7, *cross* is an environment where the *ris-bezier* approach is amongst the two bests, achieving more successes than *ris-apf*. This time, the decision to stop when no satisfying path is found is quite adapted to the encountered situation.

The success rates of a nonholonomic robot in *door* environment are shown in Fig. 8. We compare the holonomic hypothesis ('cone') versus the nonholonomic integration of the actual dynamic constraints ('dubins'). About the *RIS* computation itself, a discrepancy can be found between the 'dubins' and the 'cone' versions: the former performs better than the latter for the two approaches. That is probably the consequence of a better anticipation of the robot's dynamic properties. As any point of the 3D surface $\mathcal{S}_A$ is closer to $A(0)$ than any point of the 3D cone, fewer *RIS* are found in a given situation, which prevents the robot to try to avoid an obstacle that would actually not be reachable anyway.

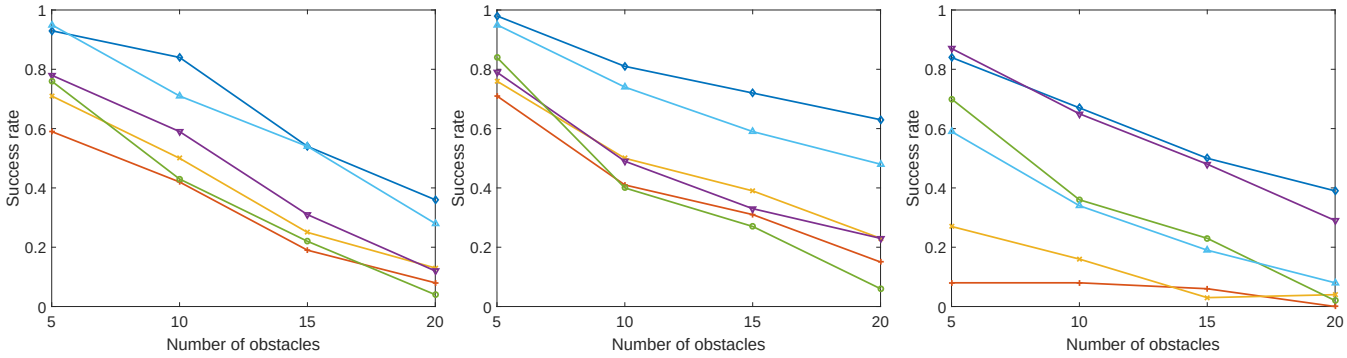
Two scenarios are presented in Fig. 9 and 10 for *free-straight* environment and disc-shaped obstacles. They all have been performed for a holonomic vehicle model. In order to display the motions of the robot and the obstacles, ten positions of the robot and the obstacles regularly extracted from the last fifty steps are shown using discs of smaller radii. One can notice that, in these scenarios, the obstacles are much faster than the robot.

Fig. 9 corresponds to a "success" scenario for *ris-apf*, *dynamic-apf* and *vo*. The latter one is generally the slowest approach over the whole scenario: *vo* only uses full speed when the robot is close to an obstacle. Approaches *ris-apf* and *dynamic-apf* behave more or less the same until (b), when *ris-apf* decides to slow down and let pass an obstacle coming fast from the left, while *dynamic-apf* get "pushed" to the right. Here we see the interest of integrating RIS_A instead of the velocities of the obstacles in an artificial potential field method: *dynamic-apf* influenced by the obstacle, follows its direction, then is forced to go backward in (c) and finally appears behind *ris-apf* in (d).



(a) In *free* environment (b) In *free-straight* environment (c) In *door* environment

Fig. 4: Holonomic robot in presence of *slower* obstacles



(a) In *free* environment (b) In *free-straight* environment (c) In *door* environment

Fig. 5: Holonomic robot in presence of *faster* obstacles

In Fig. 10, the three *RIS*-based approaches are compared and the benefit of *ris-hybrid* is demonstrated. In this scenario, only *ris-hybrid* is successful. In (b), *ris-bezier* is in danger because no suitable path has been found to circumvent the bigger obstacle coming fast. Curve $\mathcal{C}_0$ is used by default but leads to a collision just after (b). A few steps later, at (c), *ris-apf* and *ris-hybrid* have diverged since the first one reacts to a fast incoming obstacle (which can be seen on (d) going down right) by heading down while *ris-hybrid* succeeded to plan a path which circumvents the same obstacle while going toward the target. In step (d), *ris-apf* can be seen stuck in a dead end between multiple obstacles while the planning of *ris-hybrid* got him out of trouble.

Some scenarios are also recorded and presented in a video provided at `https://youtu.be/Ccu4lb56PjI`.

IV. Conclusion

We have proposed in this paper a representation of the future environment of a mobile robot, the *reachable interaction sets* that takes into account both the predicted trajectories of obstacles and some hypothesis on the moving abilities of the robot. This results in a generic, flexible approach that is suitable for trajectory planning. Whatever the nature of the prediction for obstacles or the vehicle model of the robot, the *reachable interaction sets* can be used to concatenate the future outcomes into one map on which a static trajectory planning can be performed.

The *reachable interaction sets* has been described and integrated into a motion planning framework in order to perform obstacle avoidance in highly dynamic scenes and in presence of static obstacles. Evaluations have been performed in simulated environments with different shapes, densities, and trajectories for moving obstacles and two different vehicle models for the robot. The *RIS* approach is especially relevant in scenarios involving obstacles that are faster than the controlled robot. Even if the assumption on the future motion of the robot is an approximation, the use of the *RIS* in an iterative process improves the motion planning compared to state of the art approaches (such as potential field based methods [14], [16] or velocity obstacles [12], [25]).

Work in progress includes the extension of our *reachable interaction sets* to more realistic environments. It consists especially by considering stochastic motion predictions for the obstacles, and by improving the planning method on top of *RIS* to fully integrate realistic driving constraints.

Acknowledgment

Part of this work was supported by Valeo Driving Assistance Research within the framework of a Valeo research project led by Benazouz Bradai.

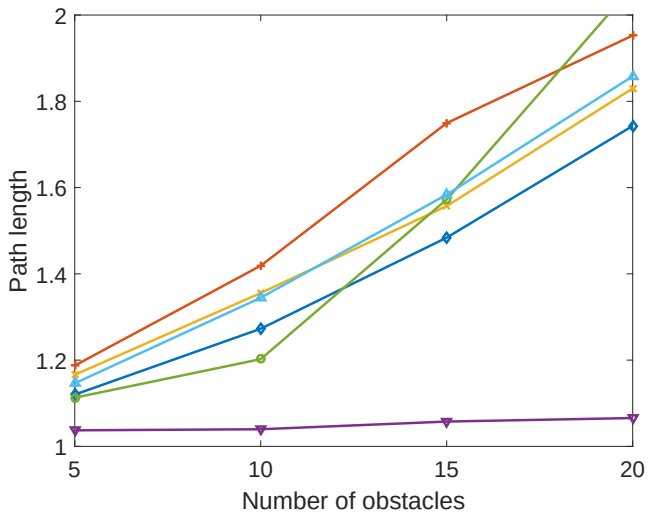


Fig. 6: Normalized path length for holonomic robot in *free* environment

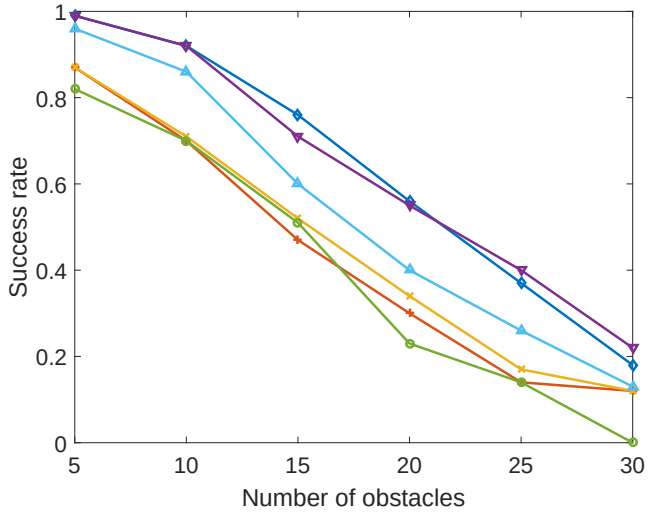


Fig. 7: Holonomic robot in *cross* environment. The colors of the different approaches are those used in Fig. 4

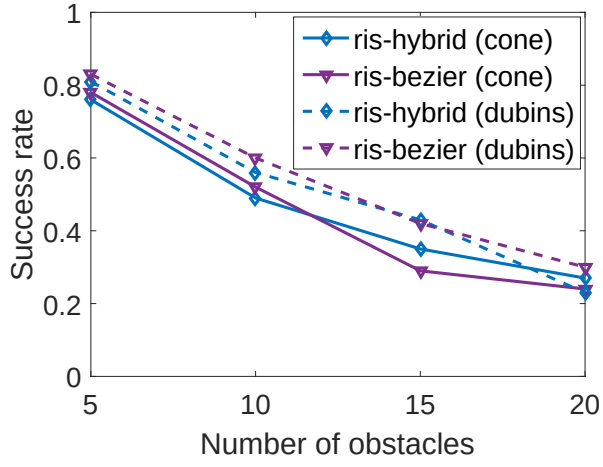


Fig. 8: Nonholonomic robot in *door* environment

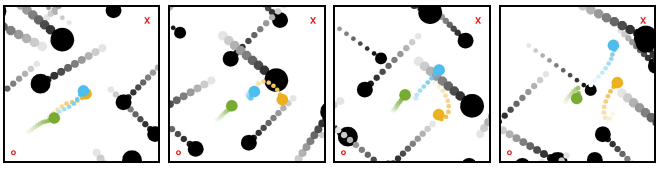


Fig. 9: Comparison of *ris-apf* (cyan), *dynamic-apf* (yellow) and *vo* (green) with 10 *faster* obstacles (black).

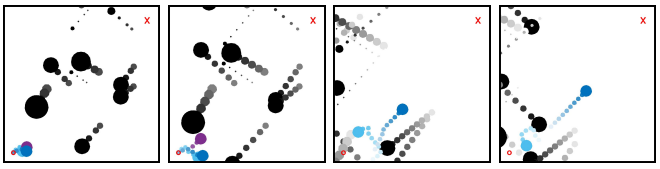


Fig. 10: Comparison of *ris-apf* (cyan), *ris-bezier* (purple) and *ris-hybrid* (blue) with 10 *faster* obstacles (black).

References

[1] J. Minguez, F. Lamiraux, and J.-P. Laumond, "Motion planning and obstacle avoidance," in *Springer Handbook of Robotics*, 2008, pp. 827–852.

[2] M. Hoy, A. S. Matveev, and A. V. Savkin, "Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey," *Robotica*, vol. 33, no. 3, p. 463–497, 2015.

[3] T. T. Mac, C. Copot, D. T. Tran, and R. De Keyser, "Heuristic approaches in robot path planning: A survey," *Robotics and Autonomous Systems*, vol. 86, pp. 13 – 28, 2016.

[4] M. Erdmann and T. Lozano-Perez, "On multiple moving objects," *Algorithmica*, vol. 2, pp. 477–521, January 1987.

[5] K. Fujimura and H. Samet, "Motion planning in a dynamic domain," in *ICRA 1990*, May 1990, pp. 324–330 vol.1.

[6] J. Canny and J. Reif, "New lower bound techniques for robot motion planning problems," in *28th IEEE Syrup. on Foundations of Computer Science*, 1987, pp. 49–60.

[7] J. van den Berg and M. H. Overmars, "Planning time-minimal safe paths amidst unpredictably moving obstacles," *I. J. Robotics Res.*, vol. 27, no. 11-12, pp. 1274–1294, 2008.

[8] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin, "A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games," *IEEE Transactions on Automatic Control*, vol. 50, no. 7, pp. 947–957, 2005.

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

[10] A. Chakravarthy and D. Ghose, "Obstacle avoidance in a dynamic environment: a collision cone approach," *IEEE Trans. Systems, Man, and Cy rnetics, Part A*, vol. 28, no. 5, pp. 562–574, Sep 1998.

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

[12] J. van den Berg, M. C. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *ICRA 2008*, 2008, pp. 1928–1935.

[13] D. Wilkie, J. van den Berg, and D. Manocha, "Generalized velocity obstacles," in *IROS 2009*, Piscataway, NJ, USA, 2009, pp. 5573–5578.

[14] J.-C. Latombe, *Robot Motion Planning*. Kluwer, 1991.

[15] N. Y. Ko and B. H. Lee, "Avoidability measure in moving obstacle avoidance problem and its use for robot motion planning," in *IROS 1996*, vol. 3, Nov 1996, pp. 1296–1303 vol.3.

[16] S. Ge and Y. Cui, "Dynamic motion planning for mobile robots using potential field method," *Autonomous Robots*, vol. 13, no. 3, pp. 207–222, Nov 2002.

[17] L. Huang, "Velocity planning for a mobile robot to track a moving target — a potential field approach," *Robotics and Autonomous Systems*, vol. 57, no. 1, pp. 55 – 63, 2009.

[18] N. C. Tsourveloudis, K. P. Valavanis, and T. Hebert, "Autonomous vehicle navigation utilizing electrostatic potential fields and fuzzy logic," *IEEE Trans. Robotics and Automation*, vol. 17, no. 4, pp. 490–497, Aug 2001.

[19] H. T. Chiang, N. Malone, K. Lesser, M. Oishi, and L. Tapia, "Path-guided artificial potential fields with stochastic reachable sets for motion planning in highly dynamic environments," in *ICRA 2015*, May 2015, pp. 2347–2354.

[20] N. Malone, H.-T. Chiang, K. Lesser, M. Oishi, and L. Tapia, "Hybrid dynamic moving obstacle avoidance using a stochastic reachable set-based potential field," *IEEE Trans. Robotics*, no. 99, pp. 1–15, 2017.

[21] S. M. La Valle, *Planning Algorithms*. Cambridge University Press, 2006.

[22] Y. Zhou and J. S. Baras, "Reachable set approach to collision avoidance for uavs," in *54th IEEE Conference on Decision and Control (CDC)*, Dec 2015, pp. 5947–5952.

[23] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *I. J. Robotics Res.*, vol. 30, no. 7, pp. 846–894, 2011.

[24] M. E. Mortenson, *Geometric Modeling*. John Wiley & Sons, Inc., 1997.

[25] J. van den Berg, S. J. Guy, J. Snape, M. C. Lin, and D. Manocha, "RVO2 library: Reciprocal collision avoidance for real-time multi-agent simulation," http://gamma.cs.unc.edu/RVO2/, 2009.