Safe Motion Planning in Dynamic Environments

Stéphane Petti
Inria Rocquencourt & Ecole des Mines de Paris
Email: stephane.petti@inria.fr

Thierry Fraichard
Inria Rhône-Alpes & Gravir Lab., Grenoble
Email: thierry.fraichard@inria.fr

***Abstract*— This paper addresses the problem of motion planning (MP) in dynamic environments. It is first argued that dynamic environments impose a real-time constraint upon MP: it has a limited time only to compute a motion, the time available being a function of the *dynamicity* of the environment. Now, given the intrinsic complexity of MP, computing a complete motion to the goal within the time available is impossible to achieve in most real situations. Partial Motion Planning (PMP) is the answer to this problem proposed in this paper. PMP is a motion planning scheme with an anytime flavor: when the time available is over, PMP returns the best partial motion to the goal computed so far. Like reactive navigation scheme, PMP faces a safety issue: what guarantee is there that the system will never end up in a critical situation yielding an inevitable collision? The answer proposed in this paper to this safety issue relies upon the concept of Inevitable Collision States (ICS). ICS takes into account the dynamics of both the system and the moving obstacles. By computing ICS-free partial motion, the system safety can be guaranteed. Application of PMP to the case of a car-like system in a dynamic environment is presented.**

***Index Terms*— Safety - Motion Planning - Dynamic Environments.**

I. Introduction

A. Overview of the Problem

The problem of autonomous navigation has attracted a lot of interest for various robotic systems during last decades. Two main paradigms have arisen to tackle this problem, the deliberative approaches resulting in global motion planning schemes, i.e. the determination of a complete plan based on a priori known information and the reactive approaches taking decision from real-time data by means of exteroceptive sensors, while moving. Though first problems involved mostly simple systems evolving among a stationary environment, it became clear that for real applications, dynamic systems as well as a dynamic environment had to be considered. Within a dynamic environment however, the system has the obligation to make a decision, within a bounded time, otherwise it might be in danger by the sole fact of being passive. This limited available time for the system to make a decision, i.e. plan a motion, depends on the nature and dynamicity of the environment and is a *hard real time constraint*. Unfortunately, global motion planning schemes based on techniques from computational geometry that were used for the first problems [1], did not give much hope to fulfil such a timing constraint due to their inherent NPHard complexity [2]. Hence, reactive methods have been preferred and several schemes have been presented [3], [4], [5]. Nevertheless, reactive approaches exhibit strong limitations. In order to take fast decisions, these methods explore locally the velocity space of the system from which one admissible control is selected at a time. As a consequence these methods exhibit a lack of lookahead, conducting the robot to be trapped in local minima during its trip, and a weak goal directedness keeping the robot from reaching the objective. Despite recent modifications on a few schemes in order to improve these techniques, once systems with kinematic or dynamic constraints are considered, specific schemes are proposed [3], [6], [7] but cannot handle such complex systems in a general form. Fortunately, in the mid 90's, probabilistic planners appeared [8] and brought a new powerful tool for rapid exploration of high dimensional state-time space, framework presented in [9] used for complex systems and dynamic environment description. Some recent work could demonstrate fast planning within a dynamic environment using probabilistic techniques [10] and [11].

However, this work does not take into consideration the fact that a complete trajectory to the goal might not be found before the available allotted time has elapsed. In fact, as [12], we believe that for complex systems or environment, a complete trajectory to the goal cannot be found over a limited time, in general. In such a case, it becomes of the utmost importance to consider the behaviour of the system at the end of the trajectory. What if a car ends its trajectory in front of a wall at high speed? It becomes clear that strong guarantees should be given to this trajectory in order to handle the *safety* issues raised by such a *partial planning*.

A few papers only have addressed the issue of safety for partial or incomplete trajectory by providing, in another context braking policies [13] or more recently by calculating evasive plan [14] or by insuring the system to be collision free during an empirical period of time τ [12]. We believe that these methods propose partial answers to the problem of safety. In Fig. 1 we consider a selected milestone of a point mass robot with non zero velocity moving to the right (a state of P is therefore characterised by its position (x, y) and its speed v). Depending upon its speed there is a region of states (in grey) for which P, even though it is not in collision and is τ-safe, will not have the time to brake and avoid collision. These states for which no matter what the future trajectory of the system is, a collision with the obstacle occurs, have been

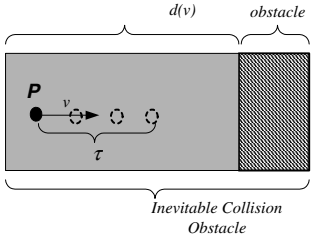


Fig. 1. τ-safe State vs. Inevitable Collision State

characterized by [15] as *Inevitable Collision States* (ICS). The framework of ICS provides a suitable framework to establish the relation of the collision states and the dynamic constraints in order to have strong safety guarantees.

B. Contribution of the Paper

In this paper, we first detail the mechanism of a Partial Motion Planner (PMP) designed to handle the partial plans that would result from planning under hard real time constraint within a dynamic environment. Then, we address the safety issue related to such a scheme by calculating trajectories of *safe states* using the framework of ICS. We provide a sufficient condition for hard safety guarantee for the PMP, allowing discrete calculation of the ICS, thus reducing the computational complexity of the scheme while insuring the safety of the whole trajectory.

The paper is organised as follows: §2 briefly reviews the work related to trajectory planning within a dynamic environment and detail the existing safety guarantees that have been provided in recent approaches. After introducing in §3 useful notations and definitions, we detail in §4 the mechanism of the PMP algorithm for which we provide and demonstrate in §5 a sufficient hard safety condition. We present finally in §6 our preliminary results for a car-like robot and draw our conclusion in §7.

II. Previous Works

Recently, a few reactive approaches have been modified in order to increase their lookahead and goal directedness. Basically, these modified methods rely on a map, built at execution-time, like an occupancy grid, or known a priori. These modified reactive methods use techniques stemming from global motion planning techniques, in order to explore the velocity space of the system and select a discrete sequence of velocities, introducing in a sense the concept of *partial planning*. The trajectory to be followed, *ie* this discrete sequence of velocities, is determined by mean of a navigation function [16] or an incremental algorithm that builds a tree within the velocity space [17], [18], [19]. However, complex systems or environments remain difficult to handle in a general form.

Global motion planning schemes have been modified as well in order to gain some reactivity toward changes within the environment, resulting thus in safer motion. Beside early work based on dynamic programming, like the dynamic A* algorithm (D*) [20], [21], the approach of motion planning has been mainly reconsidered with the new tools of probabilistic techniques based on graph or tree construction, showing impressive results [10], [11]. Only latest work on this issue, recognise the possibility of complete trajectory planning failure [12], [14] and attempts to provide safety guarantee. However the guarantees apply over specific, empirical time period without obvious physical relation with the considered dynamic system. In [14] the authors complement the work presented in [10] and define safe planning as the capacity of the planner to calculate an *escape plan* in case it has failed finding a complete trajectory to the goal during the allotted time. But no details with respect to the valid time length of the escape plan is given. In the work of [12] the authors do not attempt to plan at each step a complete trajectory, the computational constraints being to high, hence considering the eventuality of partial planning. They present the concept of τ*-safety* as a guarantee of no collision during τ seconds for each milestone of the tree. Since the planner builds a tree between primary milestones at the equilibrium state, such a criteria might suffice in the presented scheme, nevertheless it is too weak for a general kinodynamic framework.

III. Notations and Definitions

Let $\mathcal{A}$ denote a robotic system placed in a workspace $\mathcal{W}$. The motion model of $\mathcal{A}$ is described by a differential equation of the form $\dot{s} = f(s, u)$ where $s \in \mathcal{S}$ is the state of the system, $\dot{s}$ its time derivative and $u \in \mathcal{U}$ a control. $\mathcal{S}$ is the state space and $\mathcal{U}$ the control space of $\mathcal{A}$. Let $\phi \in \Phi$: $[t_0, t_f] \longmapsto \mathcal{U}$ denote a control input, *ie*. a time-sequence of controls. Starting from an initial state s_0, at time t_0, and under the action of a control input ϕ, the state of the system $\mathcal{A}$ at time t is denoted by $s(t) = \phi(s_0, t)$. An initial state and a control input define a trajectory for $\mathcal{A}$, *ie* a time sequence of states. The environment is cluttered with a set of obstacles. An obstacle $\mathcal{B}$ is a closed subset of $\mathcal{W}$. The definition of a moving obstacle is time-dependent and denoted by $\mathcal{B}(t)$. Let $\mathcal{B}(t_0, \infty)$ denotes the obstacle $\mathcal{B}$ from time t_0 to ∞, it models the future behaviour of $\mathcal{B}$. A state s is a *collision state* at time t if and only if $\exists \mathcal{B}$ such that $\mathcal{A}(s) \cap \mathcal{B}(t) \neq \emptyset$. As per [15], let us recall the formal definition of an Inevitable Collision State.

Definition 1 (Inevitable Collision State)
A state s is an Inevitable Collision State (ICS) if and only if $\forall \phi$, $\exists t_c$ (for a given ϕ) such that $\phi(s, t_c)$ is a collision state.

A safe trajectory is defined as:

Definition 2 (Safe Trajectory)
A trajectory, defined by the initial state s_0 at time t_0 and the

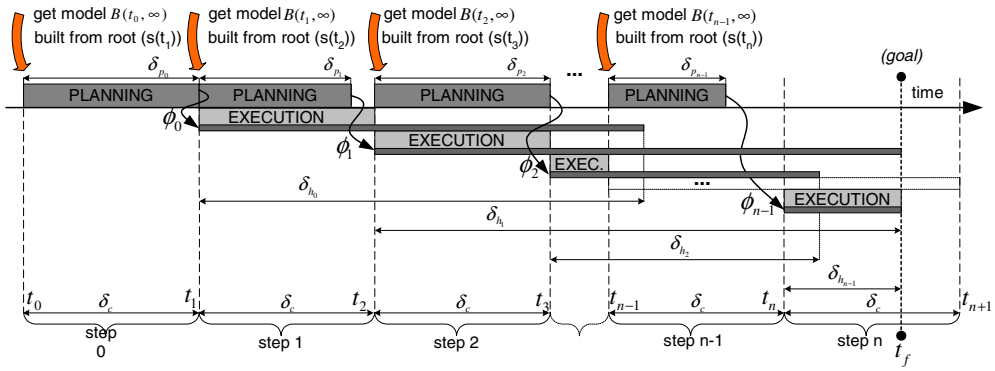


Fig. 2. Partial Motion Planning iterative cycle

control input ϕ over $[t_0, t_f]$, is safe if and only if $\forall t \in [t_0, t_f]$, $s(t) = \phi(s_0, t)$ is not an ICS.

Using these notations and definitions, the next section details the Partial Motion Planning (PMP) algorithm.

IV. Partial Motion Planner

In general, a robotic system cannot safely stand still in a dynamic environment (it might be collided by a moving obstacle). It has to plan a motion within a bounded time and then execute it in order to remain safe. The time δ_c available to calculate a new motion is function of the nature and dynamicity of the environment. Today, the only motion planning techniques able to deal with reasonably complex robotic systems are randomized. Unfortunately, these techniques do not have any running-time upper-bound: there is no guarantee whatsoever that a complete motion can be computed within the time δ_c available. Like motion planning, partial motion planning requires a model of the environment, the first step is aimed at getting this model. This paper does not address however the problem of model construction. This model can be given a priori or built using sensor observations. In space applications for instance, most of the moving obstacles follow Kepler's law, which provides a means to compute a full a priori model of the future [12]. In many other applications however, the moving obstacles have their own free will and their future behaviour is only partially predictable (if at all). In this kind of situations, the model of the future must be predicted using prediction techniques such as the ones presented in [22] or [23].

Note that when predictions are used, it is likely that the model of the future that is obtained will have a limited *validity duration* δ_v. This is an additional argument for partial motion planning: what is the point of a complete exploration of the future if the model of the future used is likely to be grossly inaccurate? It is better to iterate a partial motion planning process taking as input a regularly updated predicted model of the future. The periodic iterative PMP scheme proposed in this paper therefore accounts for both the planning time constraints and the validity duration of the model of the environment.

Thus, the PMP algorithm iterates over a cycle of duration $\delta_c \leq \delta_v$ as depicted in Fig. 2. It is assumed that the initial state of $\mathcal{A}$ is ICS-free. Let us focus on the planning iteration starting at time t_i:

1) An updated model of the future is acquired, *ie* $\mathcal{B}(t_i, \infty)$ for each moving obstacles.
2) The state-time space of $\mathcal{A}$ is searched using an randomised tree rooted at the state $s(t_{i+1})$ with $t_{i+1} = t_i + \delta_c$.
3) At time t_{i+1}, the current iteration is over, the best safe partial trajectory ϕ_i in the tree is selected according to a given criterion and is fed to the robot that will execute it from now on. ϕ_i is defined over $[t_{i+1}, t_{i+1} + \delta_{h_i}]$ with δ_{h_i} the trajectory duration.

The algorithm operates until the last state of the planned trajectory reaches a neighbourhood of the goal state.

V. Safety Issues

Like every method that computes partial motion only, PMP has to face a safety issue: since PMP has no control over the duration of the partial trajectory that is computed, what guarantee do we have that $\mathcal{A}$ will never end up in a critical situations yielding an inevitable collision? The answer to that problem lies in the very fact that the partial trajectory that is computed is ICS-free. Meaning that, even in the worst case scenario where the duration δ_{h_i} of the partial trajectory is shorter than the cycle time δ_c, $\mathcal{A}$ can always execute one of the existing safe trajectory. The overall safety is guaranteed as long as the initial state is ICS-free (which is something that has been assumed).

Now, determining whether a given state of $\mathcal{A}$ is an ICS or not is a complex task since it requires to consider all possible future trajectories for $\mathcal{A}$. However, it i possible to take advantage of the approximation property demonstrated in [15] in order to compute a conservative approximation of

the set of ICS. This is done by considering only a subset of the full set of possible future trajectories (see §VI.B).

Besides, in order to further reduce the complexity of the PMP algorithm, we present a property that simplifies the safety checking of a trajectory. To begin with, we demonstrate a property that states that for a given control input, all the states between an ICS and the corresponding collision state, are ICS.

PROPERTY 1

Let s be an ICS at time t_0. For a given control input ϕ, let t_c denote the time at which a collision occurs. Then $\forall t \in [t_0, t_c], s(t) = \phi(s, t)$ is also an ICS.

Proof: suppose that a state $s_i = \phi(s, t_i)$, with $t_i \in [0, t_c]$, is not an ICS. By definition, $\exists \phi^j$ that yields no collision when applied to s_i. Let ϕ^i denote the part of ϕ defined over $[t_0, t_i]$. Clearly, the combination of ϕ^i and ϕ^j also yields no collision when applied to $s \rightsquigarrow$ contradiction. $\diamond$

We can now provide a sufficient safety condition for a partial trajectory that states that provided a trajectory is collision free, if the last state of the trajectory is not an inevitable collision state then none of the states of the trajectory are inevitable collision states.

PROPERTY 2 (PMP SUFFICIENT SAFETY CONDITION)

Given a trajectory defined over $[t_0, t_f]$, if
(H1) *the trajectory is collision-free and*
(H2) *$s(t_f)$ is not an ICS*
then $\forall t \in [t_0, t_f]$, $s(t)$ is not an ICS.

Proof: Suppose that $\exists t_i \in [t_0, t_f]$ such that $s_i = s(t_i)$ is an ICS. Then, by definition, $\forall \phi, \exists t_c$ such that $\phi(s_i, t_c)$ is a collision state. If $t_0 \leq t_c \leq t_f$ then collision occurs before $t_f \rightsquigarrow$ contradiction with H1. Now, if $t_c > t_f$ then by previous property P1, we must have $s(t_f)$ is an ICS $\rightsquigarrow$ contradiction with H2. $\diamond$

VI. CASE STUDY

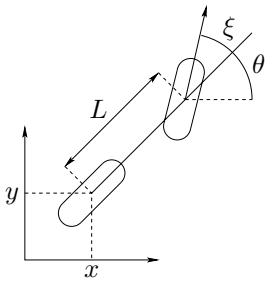
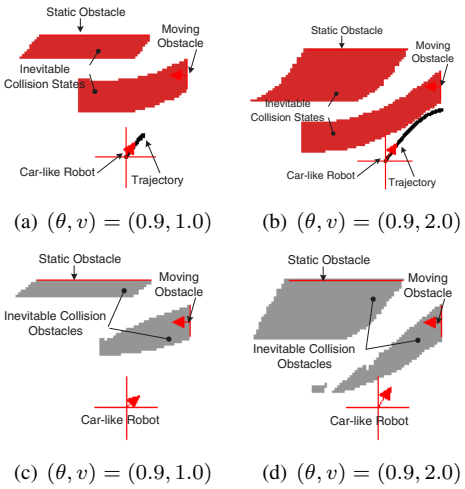


Fig. 3. The car-like vehicle $\mathcal{A}$ (bicycle model).

In this section we present the application of PMP to the case of a car-like vehicle $\mathcal{A}$ moving on a *planar* surface $\mathcal{W}$ and within a fully observable environment cluttered with stationary and dynamic obstacles.



(a) $(\theta, v) = (0.9, 1.0)$ (b) $(\theta, v) = (0.9, 2.0)$

(c) $(\theta, v) = (0.9, 1.0)$ (d) $(\theta, v) = (0.9, 2.0)$

Fig. 4. (θ, v)-slices of the state space of $\mathcal{A}$. Shaded regions are ICS respectively defined for the braking trajectory of control $(\alpha_{min}, \dot{\xi}_{min})$ (top), and all braking trajectories of control selected from $[(\alpha_{min}, \xi_{max}), (\alpha_{min}, 0), (\alpha_{min}, \dot{\xi}_{min})]$ (bottom).

A. Vehicle Model

$\mathcal{A}$ moves like a car-like vehicle and its dynamics follows the bicycle model. A state of $\mathcal{A}$ is defined by the 5-tuple $s = (x, y, \theta, v, \xi)$ where (x, y) are the coordinates of the rear wheel, θ is the main orientation of $\mathcal{A}$, v is the linear velocity of the rear wheel, and ξ is the orientation of the front wheels (Fig. 3). A control of $\mathcal{A}$ is defined by the couple (α, γ) where α is the rear wheel linear acceleration. and γ the steering velocity. The motion of $\mathcal{A}$ is governed by the following differential equations:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v}_r \\ \dot{\xi} \end{bmatrix} = \begin{bmatrix} v_r \cos\theta \\ v_r \sin\theta \\ \frac{\tan\xi v_r}{L} \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \alpha + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \gamma \qquad (1)$$

with $\alpha \in [\alpha_{min}, \alpha_{max}]$ (acceleration bounds), $\gamma \in [\gamma_{min}, \gamma_{max}]$ (steering velocity bounds), and $|\xi| \leq \xi_{max}$ (steering angle bounds). L is the wheelbase of $\mathcal{A}$.

B. ICS Calculation

In general, computing the ICS for a given system is an intricate problem since it would require to consider the set of all the possible future trajectories. To compute in practice the ICS for a system such as $\mathcal{A}$, it is taken advantage of the approximation property established in [15] showing that a conservative approximation of the ICS can be obtained by considering only a finite subset $\mathcal{I}$ of the whole set of possible future trajectories. For practical reasons, the duration of the trajectories of $\mathcal{I}$ has to be limited to a given *time horizon* that determines the overall level of safety of $\mathcal{A}$. In partially predictable environment, this horizon can be

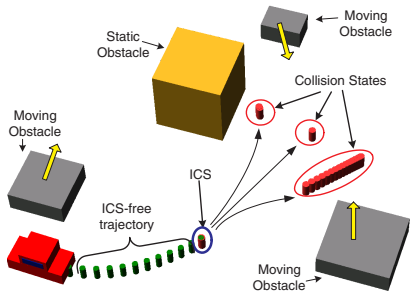


Fig. 5. The state labelled ICS is an ICS since the three braking trajectories issued from it yield collisions.

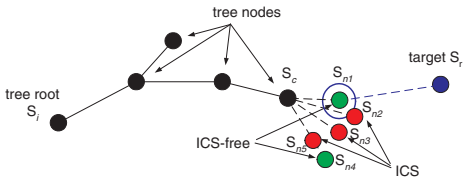


Fig. 6. Tree construction principle.

set up to δ_v, thus providing the maximum possible safety guarantee. In our case, the subset $\mathcal{I}$ considered includes the braking trajectories with a constant control selected from $[(\alpha_{min}, \dot{\xi}_{max}), (\alpha_{min}, 0), (\alpha_{min}, \dot{\xi}_{min})]$, and applied over the time necessary for $\mathcal{A}$ to stop.

Figure 4 depicts the ICS obtained when different set of braking trajectories are considered. Each subfigure represents a (θ, v)-slice of the full 5D state space of $\mathcal{A}$. In the top subfigures, only the braking trajectory of control $(\alpha_{min}, \dot{\xi}_{min})$ is considered.

For instance, the shaded regions in subfigure (a) correspond to states with $(\theta, v) = (0.9, 1.0)$ for which the braking trajectory of control $(\alpha_{min}, \dot{\xi}_{min})$ yields a collision with one of the obstacles, fixed or moving. In the bottom subfigures, the three braking trajectories are considered.

In PMP, checking whether a state is an ICS or not is carried out by testing if all the braking trajectories yield a collision with one of the moving obstacles. In case all the trajectories appear to be in collision in the future, this state is an ICS and is not selected. Fig. 5 illustrates how a state of the partial trajectory is checked to be an ICS or not. The collision states represent the collision that will occur in the future from this state for all trajectories of $\mathcal{I}$. In this case, since all trajectories collide in the future, this state is an ICS.

C. Tree Construction

The exploration method used is the well known Rapidly-Exploring Random Tree method (RRT) [24]. RRT incrementally builds a tree in the state space of $\mathcal{A}$. The basic principle of RRT is depicted in Fig. 6. A state s_r is randomly selected first. Then, the closest node in the tree, say s_c, is determined. Constant controls selected from $\mathcal{U}$=$\{$ $(\alpha,\dot{\xi})$ $|$ α $\in(\alpha_{min}, 0, \alpha_{max})$ and $\dot{\xi}$ $\in(\dot{\xi}_{max}, 0, \dot{\xi}_{min})$ $\}$ are then applied to s_c for a duration ϵ, they yield a set of candidate trajectories ending in given states s_{ni}. These candidate trajectories are pruned out: only are kept the trajectories that are collision-free and whose final state is ICS-free (as per property 2, such trajectories are ICS-free). Finally, the trajectory whose final state is closer to s_r is selected and added up to the tree. This process is repeated until the end of the time available where the best partial trajectory extracted from the tree is returned.

(a) Cycab platform from INRIA. (b) Cameras on a parking lot.

Fig. 8. Parking lot configuration for practical experiments

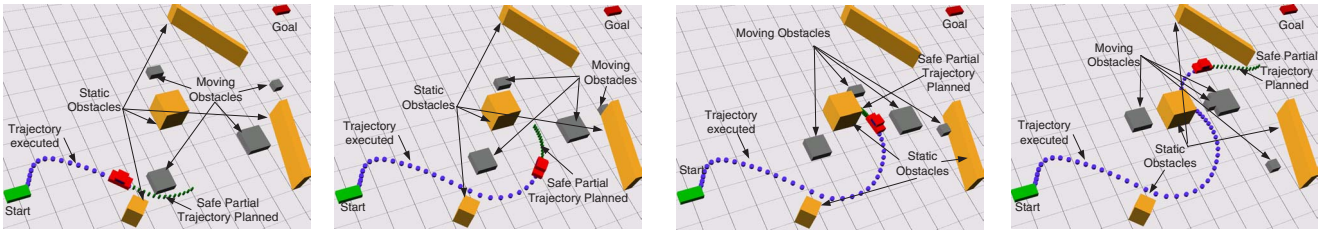
D. Preliminary Results

In this implementation, the duration of a cycle δ_c is 1s, and the integration step of the differential equation is 0.5s. In Fig. 7 we can see an example of a navigation from a still starting state to a still goal state. The environment is cluttered by moving and static obstacles. In 7(a) one can observe in front of the car how the safe partial trajectory is calculated and planned within the time-state space in order to avoid the obstacle moving upward. The states behind the car, define the trajectory, built from partial trajectories from the previous PMP cycles and (ideally) executed by the robot. In 7(b) we can observe that the car was obliged to slow down at the intersection of several obstacles, since no other safe trajectories could be found, before to re-accelerate. In 7(d) the system has planned a partial trajectory that avoids the last static obstacle. Videos of this simulation can be found at `http://emotion.inrialpes.fr/film-gallery.php`.

This work is currently being integrated on a real platform (fig. 8(a)) moving within a parking lot. External cameras above the parking (fig. 8(b)) have been installed in order to fully observe the environment and thus localize our system and detect static (parked cars) as well as moving obstacles (pedestrians and cars) whereas the trajectories of the moving obstacles will be predicted using the work of [23].

VII. Discussion and Conclusion

In this paper we tackled the problem of motion planning within a dynamic environment and proposed a Partial Motion Planning scheme (PMP) which handles the *real-time constraint* inherent to such environment. The PMP algorithm consists in iteratively exploring the state-time space during



(a) Avoidance of the first moving obstacle (b) Braking in front of unavoidable moving obstacles (c) Braking in front of moving obstacles (d) Avoidance of the last static obstacle

Fig. 7. Results of a 2D safe iterated Partial Motion Planning ($\delta_c = 1s$, $v_{max} = 2.0m/s$, $\xi_{max} = \pi/3rad$, $\dot{\xi}_{max} = 0.2rad/s$, $\alpha_{max} = 0.1m/s^2$)

a fixed limited time, by building a tree using probabilistic techniques. During a cycle, a complete trajectory calculation to the goal can not be guaranteed in general, which raises the issue of the *safety* of our system. We use the formalism of the *Inevitable collision States* (ICS) as the theoretical answer to this safety problem. A trajectory of ICS-free states guarantees that our system always have a possibility to escape critical situations. ICS computation remains however a very complicated task in its general form and we have to rely on conservative approximation for practical ICS calculation. In this paper, we also demonstrate a property that simplifies the safety checking of a trajectory. Finally, we present an implementation of the PMP algorithm for a car-like robot and provide simulation results of the safe trajectory found by PMP for our system.

Future work includes the coupling of the PMP algorithm with a closed loop control and its integration on an experimental vehicle. Our goal is to perform experimentations within a real environment, for which a model of the future obstacles' behaviour will be determined thanks to a prediction technique.

ACKNOWLEDGMENT

This work was partially supported by Aisin AW Corp. and the French CNRS Parknav and Predit Mobivip projects.

REFERENCES

[1] J.-C. Latombe, *Robot motion planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[2] J. H. Reif and M. Sharir, "Motion planning in the presence of moving obstacles," in *Symp. on the Foundations of Computer Science*, Portland, OR (US), Oct. 1985, pp. 144–154.

[3] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance, Tech. Rep. IAI-TR-95-13, 1 1995.

[4] M. Khatib, "Sensor-based motion control for mobile robots," Ph.D. dissertation, LAAS-CNRS December, 1996, 1996.

[5] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *International Journal of Robotics Research*, vol. 17, no. 7, pp. 760–772, July 1998.

[6] R. Simmons, "The curvature velocity method for local obstacle avoidance," in *International Conference on Robotics and Automation*, Minneapolis (USA), april 1996, pp. 3375–3382.

[7] J. Minguez, L. Montano, and J. Santos-Victor, "Reactive navigation for non-holonomic robots using the ego kinematic space," in *Int. Conf. on Robotics and Automation*, Washington (US), May 2002.

[8] L. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, pp. 566–580, 1996.

[9] T. Fraichard and C. Laugier, "Kinodynamic planning in a structured and time-varying 2D workspace." in *Proceedings IEEE International Conference on Robotics and Automation*, Nice, (FR), May 1992.

[10] D. Hsu, R. Kindel, J.-C. Latombe, and S. Rock, "Kinodynamic motion planning amidst moving obstacles," in *Int. Conf. on Robotics and Automation*, San Francisco (US), April 2000.

[11] J. Bruce and M. Veloso, "Real-time randomized path planning for robot navigation," in *Int. Conf. on Intelligent Robots and Systems*, Lausanne, Switzerland, October 2002.

[12] E. Feron, E. Frazzoli, and M. Dahleh, "Real-time motion planning for agile autonomous vehicles," in *AIAA Conference on Guidance, Navigation and Control*, Denver (US), August 2000.

[13] T. S. Wikman, M. S. Branicky, and W. S. Newman, "Reflexive collision avoidance: a generalized approach," in *Int. Conf. on Robotics and Automation*, vol. 3, Atlanta (US), May 1993, pp. 31–36.

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

[15] T. Fraichard and H. Asama, "Inevitable collision states - a step towards safer robots?" *Advanced Robotics*, vol. 18, no. 10, pp. 1001–1024, 2004.

[16] O. Brock and O. Khatib, "High-speed navigation using the global dynamic window approach," in *Int. Conf. on Robotics and Automation*, Detroit (US), May 1999.

[17] L. Ulrich and J. Borenstein, "VFH*: Local obstacle avoidance with look-ahead verification," in *Int. Conf. on Robotics and Automation*, San Fransisco (US), April 2000, pp. 2505–2511.

[18] F. Large, S. Sekhavat, Z. Shiller, and C. Laugier, "Towards real-time global motion planning in a dynamic environment using the NLVO concept," in *Int. Conf. on Intelligent Robots and Systems*, Lausanne, Switzerland, October 2002.

[19] C. Stachniss and W. Burgard, "An integrated approach to goal-directed obstacle avoidance under dynamic constraints for dynamic environments," in *Int. Conf. on Intelligent Robots and Systems*, Lausanne, Switzerland, October 2002.

[20] A. Stentz, "The focussed D* algorithm for real-time replanning," in *Int. Joint Conf. on Artificial Intelligence*, Montreal, Quebec, 1995, pp. 1652–1659.

[21] S. Koenig and M. Likhachev, "Improved fast replanning for robot navigation in unknown terrain," in *Int. Conf. on Robotics and Automation*, 2002.

[22] M. Bennewitz, W. Burgard, and S. Thrun, "Learning motion patterns of persons for mobile service robots," in *Int. Conf. on Robotics and Automation*, Washington DC (US), 2002, pp. 3601–3606.

[23] D. Vasquez and T. Fraichard, "Motion prediction for moving objects: a statistical approach," in *Int. Conf. on Robotics and Automation*, New Orleans, LA, April 2004.

[24] S. LaValle and J. Kuffner, "Randomized kinodynamic planning," in *Int. Conf. on Robotics and Automation*, Detroit (US), May 1999, pp. 473–479.