

Dynamic Trajectory Planning, Path-Velocity Decomposition and Adjacent Paths*

Th. Fraichard and C. Laugier

LIFIA-INRIA Rhône-Alpes

46, av. Félix Viallet, 38031 Grenoble Cedex 1, France

[tf, laugier]@lifia.imag.fr

Abstract — This paper addresses **Dynamic Trajectory Planning**, which is defined as Motion Planning for a robot $\mathcal{A}$ moving in a **dynamic workspace** $\mathcal{W}$, i.e. with moving obstacles. Besides $\mathcal{A}$ is subject both to **kinematic constraints** and **dynamic constraints**. We consider the case of a car-like robot $\mathcal{A}$ with bounded velocity and acceleration, moving in a dynamic workspace $\mathcal{W} = \mathbb{R}^2$. Our approach is an extension to the Path-Velocity Decomposition [Kant and Zucker, 1986]. We introduce the concept of **adjacent paths** and we use it within a novel planning schema which operates in two complementary stages: (a) **Paths Planning** and (b) **Trajectory Planning**. In Paths Planning, a set of adjacent paths, one of which leading $\mathcal{A}$ to its goal, are computed. These paths are collision-free with the stationary obstacles and respect $\mathcal{A}$'s kinematic constraints. In Trajectory Planning, knowing that $\mathcal{A}$ is able to shift from one path to an adjacent one freely, we determine the motion of $\mathcal{A}$ along and between these paths so as to avoid the moving obstacles while respecting $\mathcal{A}$'s dynamic constraints.

1 Introduction

1.1 The Problem

Ever since Nilsson's work in 1969 [Nilsson, 1969], Motion Planning has been extensively studied (the reader is referred to [Latombe, 1990] for a recent survey of this topic). Previous works can be classified according to the type of motions which are planned; it is possible to differentiate between *Path Planning* which is characterized by the search of a continuous sequence of configurations¹ and *Trajectory Planning* which is concerned with the time history of such a sequence.

Path Planning is restricted to the geometric aspects of Motion Planning. The only constraints which can be dealt with are time-independent constraints such as stationary obstacles and *kinematic constraints*, i.e. constraints involving the configuration parameters of the robot and their derivatives. Depending on whether it is *holonomic* or not,² a kinematic constraint either re-

duces the set of allowed configurations or restricts the geometric shape of feasible paths (see [Barraquand and Latombe, 1990] for more details). On the other hand, Trajectory Planning permits to take into account time-dependent constraints such as moving obstacles and *dynamic constraints*, i.e. constraints on the forces or the accelerations applied to the robot.

When planning the motion of an actual robot, it is important to take into account the various constraints which restrict its motion capabilities. It is also important to deal with moving obstacles since an actual workspace will often be dynamic, i.e. with moving obstacles. These points have been addressed in the past, but, as far as we know, never simultaneously³ and it is our goal to try to do so. Thus, in this paper, we will study **Dynamic Trajectory Planning** which is defined as Trajectory Planning for a robot moving in a dynamic workspace. Besides the kinematic and dynamic constraints of the robot will be taken into account.

1.2 The Approach

We consider the case of a car-like robot $\mathcal{A}$ with bounded velocity and acceleration, moving in a dynamic planar workspace $\mathcal{W}$. Note that, because it is a wheeled robot, $\mathcal{A}$ is subject to non-holonomic constraints (see §3.1).

There are no complexity results for the two-dimensional instance of Dynamic Trajectory Planning that we consider. However results about similar problems suggest that it is intractable (cf. the NP-hard result established in [Canny, 1988] and the P-Space algorithm presented in [Canny *et al.*, 1990]). On the other hand, the one-dimensional case seems less intricate (cf. the polynomial algorithm presented in [Ó'Dúnlaing, 1987]). These results motivated us to use Path-Velocity Decomposition [Kant and Zucker, 1986] which addresses Motion Planning in two complementary stages: (a) planning a geometric path which avoids the stationary obstacles and (b) planning the velocity along this path so as to avoid the moving obstacles. This approach is of a practical interest because it decomposes the original problem into two more simple problems. However it presents a serious

i.e. if the derivative of the configuration parameters can be eliminated.

³Except in [Fujimura and Samet, 1989] and [Ó'Dúnlaing, 1987] but with simplifying assumptions.

*This work was supported by the French Ministry of Research and Space and the European EUREKA projet PROMETHEUS Pro-Art.

¹The *configuration* of a robot is a set of independent parameters of minimal cardinality which uniquely defines the position and orientation of every point of the robot.

²A kinematic constraint is *holonomic* if it is integrable,

drawback: it cannot find a solution if a moving obstacle stops right on the computed path. The usual answer to this problem is to consider a set of candidate paths [Pan and Luo, 1990]. Our own answer is the novel concept of **adjacent paths**. Adjacent paths will be formally defined in §5. Meanwhile, an informal illustration of what adjacent paths are can be found in the roadway: roads are usually divided into several adjacent lanes and every driver knows what passing from one lane to an adjacent one means.

Thanks to this concept, we have designed a novel motion planning schema which operates in two complementary stages. The first stage, **Paths Planning**, takes into account all the time-independent features of the problem at hand (stationary obstacles and $\mathcal{A}$'s kinematic constraints) whereas the second stage, **Trajectory Planning**, deals with the time-dependent ones (moving obstacles and $\mathcal{A}$'s dynamic constraints). In Paths Planning, a set of adjacent paths, one of which leading $\mathcal{A}$ to its goal, are computed. These paths are collision-free with the stationary obstacles and respect $\mathcal{A}$'s kinematic constraints. In Trajectory Planning, knowing that $\mathcal{A}$ is able to shift from one path to an adjacent one freely, we determine the motion of $\mathcal{A}$ along and between these paths so as to avoid any collision with the moving obstacles while respecting $\mathcal{A}$'s dynamic constraints.

1.3 Content of the Paper

§2 reviews the works related to Dynamic Trajectory Planning. Then §3 formally states the problem at hand. Afterwards §4 describes an algorithm which solves this problem by using Path-Velocity Decomposition while §5 presents the concept of adjacent paths and shows how it can be used to extend the previous algorithm. Finally experimental results are presented in §6.

2 Related Works

To our knowledge, [Fujimura and Samet, 1989] and [Ó'Dúnlaing, 1987] are the only references addressing Dynamic Trajectory Planning but with far too simplifying assumptions. On the other hand, there are numerous works which are of a particular interest to our problem. The works addressing either moving obstacles, non-holonomic kinematic constraints, or dynamic constraints are reviewed in the next three sections.

2.1 Moving Obstacles

The general approach which deals with moving obstacles is the 'configuration-time space' approach: it consists in adding the time dimension to the robot's configuration space. Accordingly the different approaches developed in order to solve the path planning problem in the configuration space can be used (cf. [Latombe, 1990]). Among the various existing works, we can distinguish those based upon extension of the visibility graph [Erdmann and Lozano-Perez, 1986] and those based upon cell decomposition [Shih *et al.*, 1990] ([Fujimura and Samet, 1989] falls into this latter category).

2.2 Non-Holonomic Constraints

As for car-like robots, Laumond [1986] was the first to give the condition of existence of a feasible path. He shows that if some path exists then it is always possible to transform it into a feasible path. In the wake of this result, several authors [Jacobs *et al.*, 1991; Jacobs *et al.*, 1990; Jiang *et al.*, 1992] have proposed path planners which operate in two steps: (a) a path disregarding the non-holonomic constraints is computed and (b) this path is turned into a feasible path. Other approaches are presented in [Barraquand and Latombe, 1989] and [Tournassoud and Jehl, 1988]. Furthermore several authors [Jacobs and Canny, 1989; Laumond, 1987; Wilfong, 1988] have studied the problem of planning a path without manœuvres, i.e. without inversion of the direction of motion; this problem appears to be more difficult. The crucial fact about all these methods⁴ is that they generate paths made up of straight segments connected with tangential circular arcs of minimum radius (such paths are the shortest ones for car-like robots [Dubins, 1957; Reeds and Shepp, 1990]).

2.3 Dynamic Constraints

There are results for exact time-optimal trajectory planning for Cartesian robots subject to bounds on their velocity and acceleration [Canny *et al.*, 1990; Ó'Dúnlaing, 1987]. Besides optimal-control theory provides exact results in the case of robots with full dynamics moving along a given path [Bobrow *et al.*, 1985; Shin and McKay, 1985]. However the difficulty of the general problem and the need for practical algorithms led some authors to develop approximate methods. Their basic principle is to define a grid which is searched in order to find a near-optimal solution. Such grids are defined either in the workspace [Shiller and Dubowsky, 1988], the configuration space [Sahar and Hollerbach, 1985], or the state space of the robot [Jacobs *et al.*, 1989; Xavier, 1992].

3 Statement of the Problem

3.1 The Robot $\mathcal{A}$

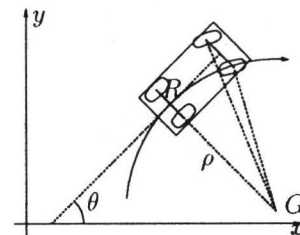


Figure 1: a car-like robot

Let $\mathcal{A}$ be a car-like robot, it has two rear wheels and two directional front wheels. It is modelled as a polygon moving on $\mathbb{R}^2$. A configuration of $\mathcal{A}$ is completely defined by the 3-tuple $(x, y, \theta) \in \mathbb{R}^2 \times [0, 2\pi[$ where (x, y) are the coordinates of the rear axle midpoint R , and θ the orientation of $\mathcal{A}$ (Fig.1).

⁴Except [Jacobs *et al.*, 1991].

3.1.1 Kinematic Constraints

A body moving on the plane has only one centre of rotation. Let G be $\mathcal{A}$'s center of rotation. Assuming pure rolling condition, a wheel can only move in a direction which is normal to its axle. Therefore, when $\mathcal{A}$ is moving, the axles of its wheels intersect at G . The orientation of the rear wheels being fixed, G is located on the rear wheels axle (possibly at an infinite distance) and R moves in a direction which is normal to this axle. In other words, the following constraint holds :

$$\tan \theta = \dot{y}/\dot{x} \quad (1)$$

Besides, due to the fact that the front wheels orientation is mechanically limited, the distance ρ between R and G , i.e. the curvature radius at point R , is lower bounded, namely:

$$\rho \geq \rho_{\min} \quad (2)$$

Relations (1) and (2) are non-holonomic. As we will see further down, they restrict the geometric shape of feasible paths for $\mathcal{A}$.

3.1.2 Dynamic Constraints

Let v be the velocity of $\mathcal{A}$ measured along its main axis and let $a_{\tan}$ and a_{rad} be respectively the tangential and radial components of the acceleration applied to $\mathcal{A}$. The following dynamic constraints should be satisfied:

$$0 \leq v \leq v_{\max} \quad (3)$$

$$-a_{\max} \leq a_{\tan} \leq a_{\max} \quad (4)$$

$$-g_{\max} \leq a_{\text{rad}} \leq g_{\max} \quad (5)$$

The meaning of these constraints is quite straightforward. Inequality (3) is a simple speed limit and (4) is an upper bound on the rate of change of speed. The purpose of (5) is to ensure that the radial acceleration does not exceed the counteracting centrifugal acceleration which is supplied by friction between the wheels and the ground. Although simple, these dynamics constraints are rich enough to demonstrate the interest of our approach. More accurate dynamics constraints could easily be dealt with (cf. [Fraichard, 1993]).

3.2 The Workspace $\mathcal{W}$

$\mathcal{W} = \mathbb{R}^2$, it is cluttered up with stationary obstacles $\mathcal{B}_i^S$, $i = 1 \dots s$, and with moving obstacles $\mathcal{B}_j^M$, $j = 1 \dots m$. Both types of obstacles are modelled as convex polygonal regions of $\mathcal{W}$.

3.3 The Solution

In this framework, a trajectory between an initial configuration q_s and a final configuration q_g is defined by a mapping Γ taking a time $t \in [0, t_f]$ to a configuration $\Gamma(t) = (x(t), y(t), \theta(t))$ and such that $\Gamma(0) = q_s$ and $\Gamma(t_f) = q_g$. The time for the trajectory Γ is t_f . Γ must be **collision-free**, i.e. it must respect:

$$\forall t \in [0, t_f], \forall i \in \{1 \dots s\}, \forall j \in \{1 \dots m\},$$

$$\mathcal{A}(t) \cap \mathcal{B}_i^S = \emptyset \text{ and } \mathcal{A}(t) \cap \mathcal{B}_j^M(t) = \emptyset$$

where $X(t)$ designates the region of $\mathcal{W}$ occupied by the object X at time t . Besides Γ must be **feasible**, i.e. it must respect the constraints (1)-(5) presented earlier.

Of course we are interested in finding a **time-optimal** trajectory, i.e. a solution Γ such that t_f should be minimal.

4 Path-Velocity Decomposition

This section presents an algorithm which solves the problem at hand by using Path-Velocity Decomposition. It operates in two complementary stages: the first stage — **Path Planning** — computes a geometric path leading $\mathcal{A}$ to its goal. This path is collision-free with the stationary obstacles and respect $\mathcal{A}$'s kinematic constraints. The second stage — **Velocity Planning** — determines the motion of $\mathcal{A}$ along this path so as to avoid the moving obstacles while respecting $\mathcal{A}$'s dynamic constraints.

4.1 Path Planning

A path for $\mathcal{A}$ is a continuous sequence of configurations, i.e. a curve in the $xy\theta$ -space between $q_s = (x_s, y_s, \theta_s)$ and $q_g = (x_g, y_g, \theta_g)$. It must be collision-free with the stationary obstacles while respecting the non-holonomic constraints (1) and (2). Equation (1) implies that the xy -curve followed by R completely defines a path. Accordingly, a path for $\mathcal{A}$ can be defined by a mapping $\Upsilon : [0, v] \rightarrow \mathbb{R}^2$ such that $\Upsilon(0) = (x_s, y_s)$ and $\Upsilon(v) = (x_g, y_g)$. v is the curvilinear length of Υ . As a consequence of (1), Υ must be piecewise of class C^1 . Besides (2) implies that the curvature of Υ (wherever it is defined) must be upper-bounded by $1/\rho_{\min}$ and that the cusp points of Υ should correspond to inversion of $\mathcal{A}$'s direction of motion. Since (3) implies that $\mathcal{A}$ is not allowed to back-up, Υ must be of class C^1 (no cusp points).

To sum up, Path Planning has to compute a xy -curve between (x_s, y_s) and (x_g, y_g) which is of class C^1 and whose curvature (wherever it is defined) is upper-bounded by $1/\rho_{\min}$, and which avoids the stationary obstacles.

There already are a few methods which have been developed in order to solve the problem at hand (cf. §2.2). However we designed our own path planner. It was presented in [Fraichard, 1991] so we will not detail it here. Suffice it to say that, as the aforementioned methods, it generates a path made up of straight segments connected with tangential circular arcs whose radii are greater than $\rho_{\min}$.

4.2 Velocity Planning

In order to determine $\mathcal{A}$'s motion along a path Υ , we start by parameterizing it with a single variable p representing the distance traveled along Υ . Accordingly a configuration of $\mathcal{A}$ becomes a 1-tuple (p) . Afterwards the dynamic constraints (3), (4) and (5) are transformed into constraints on the velocity v and the acceleration a along Υ , namely:

$$-a_{\max} \leq a \leq a_{\max} \quad (6)$$

$$0 \leq v \leq \min(v_{\max}, \sqrt{\rho \cdot g_{\max}}) \quad (7)$$

where ρ is the radius of curvature of Υ at the current position of $\mathcal{A}$. Constraint (7) is expressed by a veloc-

ity limit curve in the state space, i.e. the $p \times v$ plane. On the other hand, the constraints imposed by the moving obstacles can be represented by forbidden regions of the $p \times t$ space, where t represents the time dimension [Kant and Zucker, 1986]. In order to deal simultaneously with these two types of constraints, we use the concept of **state-time space** [Fraichard, 1993], i.e. the $p \times v \times t$ space. This concept permits to model simultaneously the constraints imposed by the moving obstacles and the constraints on the velocity. Let ST be the state-time space of $\mathcal{A}$, a curve of ST represents a motion along Υ . If such a curve respects (6), remains under the velocity limit curve (7) and avoids the forbidden regions due to the moving obstacles, then it is a solution to the Velocity Planning problem. The algorithm we designed in order to find such a curve operates in the following way: it chooses a time-step τ and assumes that the acceleration applied to $\mathcal{A}$ during such a time-step is either minimum, null or maximum. Accordingly a state-time st has three neighbours and all the state-times that $\mathcal{A}$ can reach from a given state-time lie on a regular grid embedded in ST (Fig.2). This grid is then searched in order to find a solution. Accordingly Velocity Planning is reduced to graph search (cf.[Fraichard, 1993]).

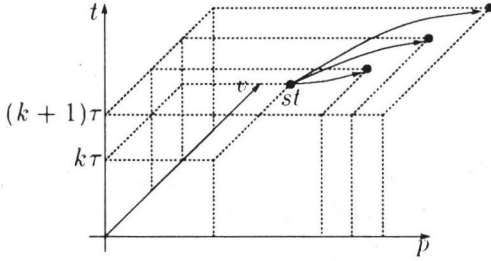


Figure 2: the grid embedded in ST

5 Adjacent Paths

This section presents the concept of adjacent paths and shows how it can be used to extend the algorithm described in the previous section.

5.1 Definition

A path for $\mathcal{A}$ is a xy-curve Υ of class C^1 whose curvature (wherever it is defined) is upper-bounded by $1/\rho_{\min}$. Assuming that Υ does not intersect itself, a point P located at a distance from Υ smaller than $\rho_{\min}$ has a unique normal projection P' on Υ . Let $d_{\Upsilon}(P)$ be the signed distance⁵ between P and its projection on Υ . The path **adjacent to Υ on its left at a distance δL** is defined as: $\Upsilon_l = \{P \in \mathcal{W} \mid d_{\Upsilon}(P) = \delta L\}$ (Fig.3). Similarly, the path **adjacent to Υ on its right at a distance δL** is defined as: $\Upsilon_r = \{P \in \mathcal{W} \mid d_{\Upsilon}(P) = -\delta L\}$.

Let Υ_N be the nominal path leading $\mathcal{A}$ to its goal, it is possible to recursively compute a set of paths adjacent to Υ_N on its left and on its right. Let $\mathcal{P} = \{\Upsilon_k, k =$

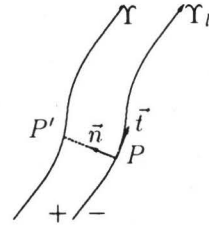


Figure 3: Υ_l , a path adjacent to Υ on its left

$1, \dots, n\}$ be the whole set of adjacent paths, Υ_N included. Let us denote by $\mathcal{R}(\Upsilon)$ the set of points whose distance to Υ is smaller than $\delta L/2$. The distance δL between any two adjacent paths is chosen so as to ensure that, $\forall \Upsilon \in \mathcal{P}$:

1. The region swept by $\mathcal{A}$ when it follows a path Υ is included in $\mathcal{R}(\Upsilon)$.
2. The region swept by $\mathcal{A}$ on its left (resp. right) side when it leaves a path Υ by making a right (resp. left) turn of radius $\rho_{\min}$, is included in $\mathcal{R}(\Upsilon)$ (Fig.4-a).
3. The region swept by $\mathcal{A}$ on its left (resp. right) side when it reaches a path Υ by making a right (resp. left) turn of radius $\rho_{\min}$ is included, in $\mathcal{R}(\Upsilon)$ (Fig.4-b).

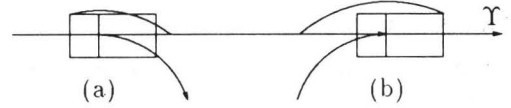


Figure 4: regions swept by $\mathcal{A}$ on its left when leaving/reaching a path Υ by making a right turn

As we will see further down, these properties permits to simplify collision checking when $\mathcal{A}$ moves along a path (§5.2) and passes from one path to an adjacent one (§5.3).

5.2 Checking Adjacent Paths

The nominal path is collision-free with the stationary obstacles and respects the kinematic constraints (1) and (2). Unfortunately these two properties do not necessarily hold for an adjacent path. It may happen that such a path is no longer collision-free or does no longer respect the curvature constraint (2). Let $\Upsilon \in \mathcal{P}$, it is a mapping $[0, v] \rightarrow \mathcal{W}$. Collision and curvature checking lead us to compute a set of forbidden intervals in the range $[0, v]$ corresponding to parts of Υ which violate either of these constraints. Let us denote by $\rho(p)$ the curvature radius at position p along Υ . The set of forbidden intervals for Υ is formally defined as:

$$\mathcal{F}(\Upsilon) = \{[a, b] \subset [0, v] \mid (\forall s \in [a, b], \rho(p) < \rho_{\min}) \text{ or } (\exists \mathcal{B}_i^S \mid \text{Proj}(\mathcal{B}_i^S, \Upsilon) = [a, b])\}$$

where $\text{Proj}(\mathcal{B}_i^S, \Upsilon)$ is the projection of the stationary obstacle $\mathcal{B}_i^S$ on the path Υ , i.e. the part of Υ which entails

⁵If P is on the left (resp. right) side of Υ , then $d_{\Upsilon}(P)$ is positive (resp. negative).

a collision between $\mathcal{A}$ and $\mathcal{B}_i^S$, namely:

$$\text{Proj}(\mathcal{B}_i^S, \Upsilon) = \{[a, b] \subset [0, v] \mid \forall p \in [a + l_f, b - l_r], \\ \exists P \in \mathcal{B}_i^S \cap \mathcal{R}(\Upsilon) \mid P' = \Upsilon(p)\}$$

where P' is the normal projection of P on Υ and where l_f (resp. l_r) denotes the distance between $\mathcal{A}$'s rear axle and its front-most (resp. rear-most) point. Figure 5 depicts an example of forbidden intervals for a path Υ . The interval $[a_1, b_1]$ corresponds to $\text{Proj}(\mathcal{B}_i^S, \Upsilon)$ while $[a_2, b_2]$ corresponds to a part of Υ which does not respect the curvature constraint (2).

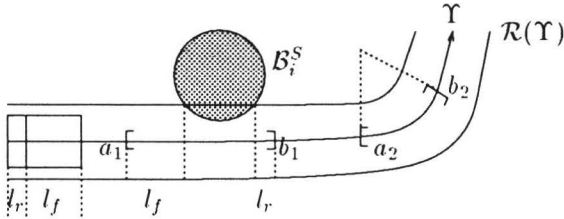


Figure 5: the 'forbidden' intervals of Υ

Note that $\text{Proj}(\mathcal{B}_i^S, \Upsilon)$ is defined by using $\mathcal{R}(\Upsilon)$ instead of the exact region swept out by $\mathcal{A}$ when it moves along Υ . This choice is sound because the region swept out by $\mathcal{A}$ is included in $\mathcal{R}(\Upsilon)$ (property 1 of §5.1).

5.3 Path-Changing

Let us study now how $\mathcal{A}$ can move from one path to an adjacent one, i.e. make a path-changing. This is depicted in Fig.6-a. At time t_1 , $\mathcal{A}$ shifts smoothly from its current path Υ_k to an adjacent path Υ_{k+1} . $\mathcal{A}$ follows a nominal trajectory Ω and reaches Υ_{k+1} at a certain time t_2 . This type of motion is two-dimensional. However it is possible to take advantage of the properties of the adjacent paths in order to reduce it to a one-dimensional motion.

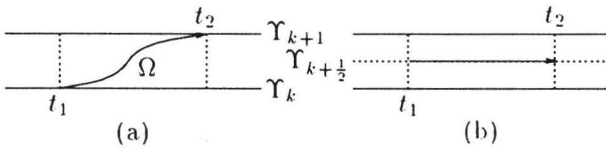


Figure 6: path-changing

Properties 2 and 3 of §5.1 ensure that, when $\mathcal{A}$ moves along Ω , it always remains included in the domain $\mathcal{R}(\Upsilon_k) \cup \mathcal{R}(\Upsilon_{k+1})$. Consequently, evaluating whether Ω is collision-free can be done very simply by checking out potential collision in both paths Υ_k and Υ_{k+1} during the time interval $[t_1, t_2]$. Note that, in this case, the obstacles on a path Υ include both the forbidden intervals $\mathcal{F}(\Upsilon)$ and the projections of the moving obstacles $\text{Proj}(\mathcal{B}_j^M(t), \Upsilon)$, such projections are time-dependent.

Accordingly it is possible to model path-changing as a simultaneous motion along Υ_k and Υ_{k+1} during $[t_1, t_2]$, or equivalently, as a three-stage process: (a) at time t_1 ,

$\mathcal{A}$ instantaneously shifts from Υ_k to a fictitious intermediate path $\Upsilon_{k+1/2}$, (b) $\mathcal{A}$ moves along $\Upsilon_{k+1/2}$ during $[t_1, t_2]$ (the obstacles of both Υ_k and Υ_{k+1} are assumed to be projected on $\Upsilon_{k+1/2}$ and (c) at time t_2 , $\mathcal{A}$ instantaneously shifts from $\Upsilon_{k+1/2}$ to Υ_{k+1} (Fig.6-b). Accordingly this modelling reduces path-changing to a one-dimensional motion along a fictitious path.

5.4 Algorithm Extension

The algorithm presented in §4 can easily be extended in order to integrate path-changings: a state-time of $\mathcal{A}$ simply becomes a 4-tuple (L, p, v, t) where L is the index of the current path of $\mathcal{A}$. Given a method which determines the path-changing trajectory Ω^6 , it is possible to determine the state-time reached by $\mathcal{A}$ at the end of a path-changing. A given state-time has now five neighbours: three on the same path and two on each adjacent paths. All the state-times reachable from a given state-time still lie on a regular grid embedded in ST and it is still possible to search this grid in order to find a solution (Fig.7).

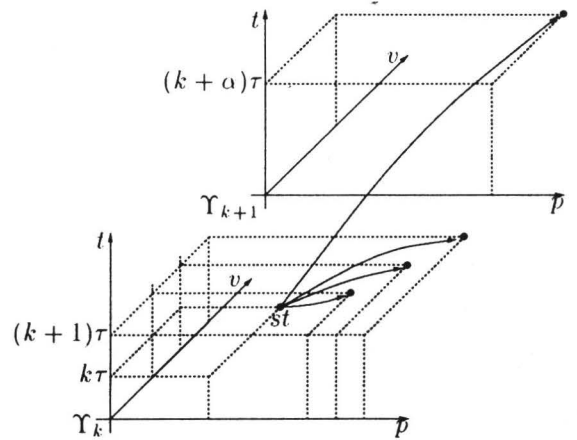


Figure 7: the grid embedded in the extended ST

6 Experimental Results

The algorithm presented above has been implemented in C on a Sun SPARC I. In these experiments, the moving obstacles are generated at random without caring whether they collide with each other. An example with two adjacent paths is depicted in Fig.8. A path is associated with two windows: a trace window showing the part of the grid which has been explored and a result window displaying the final trajectory. Any such window represents the 'time x position' space of the path. The thick black segments represent the trails left by the moving obstacles and the little dots are points of the underlying grid. Note that the vertical spacing of the dots corresponds to the time-step τ . In the example, $\mathcal{A}$ starts on the first path (lane 0) at position 0 (upper-left corner) with a null velocity. It is to reach the end of the first

⁶Such a method is described in [Fraichard and Laugier, 1993] for a car-like robot.

path (right border) with a null velocity. $\mathcal{A}$ can overtake by using the second path (lane 1).

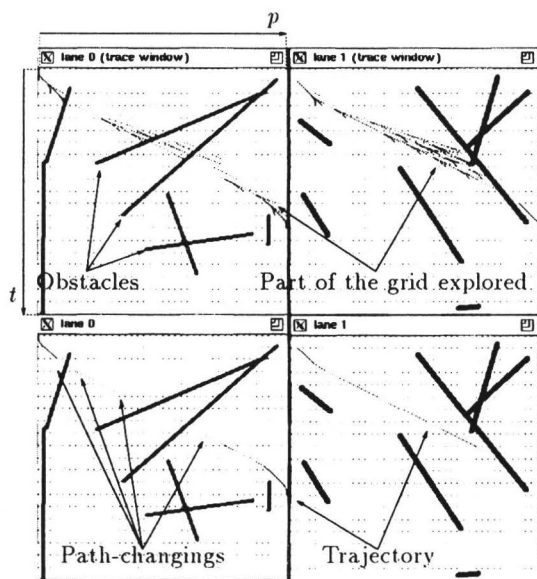


Figure 8: an example of trajectory planning with two paths: the solution has four path-changings

References

- [Barraquand and Latombe, 1989] J. Barraquand and J.-C. Latombe. – On non-holonomic mobile robots and optimal maneuvering. – *Revue d'intelligence Artificielle*, 3(2):77–103, 1989.
- [Barraquand and Latombe, 1990] J. Barraquand and J.-C. Latombe. – Controllability of mobile robots with kinematic constraints. – Research Report STAN-CS-90-1317, Robotics Lab., Computer Science Dept., Stanford Univ. CA (USA), June 1990.
- [Bobrow *et al.*, 1985] J.E. Bobrow, S. Dubowsky, and J.S. Gibson. – Time-optimal control of robotic manipulators along specified paths. – *Int. Journal of Robotics Research*, 4(3):3–17, Fall 1985.
- [Canny *et al.*, 1990] J. Canny, A. Rege, and J. Reif. – An exact algorithm for kinodynamic planning in the plane. – In *Proc. of the ACM Symp. on Computational Geometry*, pages 271–280, Berkeley, CA (USA), 1990.
- [Canny, 1988] J. Canny. – *The complexity of robot motion planning*. – MIT Press, Cambridge, MA (USA), 1988.
- [Dubins, 1957] L.E. Dubins. – On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. – *American Journal of Mathematics*, 79:497–516, 1957.
- [Erdmann and Lozano-Perez, 1986] M. Erdmann and T. Lozano-Perez. – On multiple moving objects. – A.I. Memo 883, MIT AI Lab., Boston, MA (USA), May 1986.
- [Fraichard and Laugier, 1993] Th. Fraichard and C. Laugier. – Kinodynamic planning in a structured and time-varying workspace. – In C. Laugier, editor, *Lecture Notes in Computer Science*. Springer-Verlag, To appear 1993. –
- [Fraichard, 1991] Th. Fraichard. – Smooth trajectory planning for a car in a structured world. – In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 318–323, Sacramento, CA (USA), Apr. 1991.
- [Fraichard, 1993] Th. Fraichard. – Dynamic trajectory planning with dynamic constraints: a 'state-time space' approach. – In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Yokohama (J), July 1993.
- [Fujimura and Samet, 1989] K. Fujimura and H. Samet. – A hierarchical strategy for path planning among moving obstacles. – *IEEE Trans. Robotics and Automation*, 5(1):61–69, Feb. 1989.
- [Jacobs and Canny, 1989] P. Jacobs and J. Canny. – Planning smooth paths for mobile robots. – In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 2–7, Scottsdale, AZ (USA), May 1989.
- [Jacobs *et al.*, 1989] P. Jacobs, G. Heinzinger, J. Canny, and B. Paden. – Planning guaranteed near-time-optimal trajectories for a manipulator in a cluttered workspace. – Research Report ESRC 89-20/RAMP 89-15, Engineering Systems Research Center, Univ. of California., Berkeley, CA (USA), Oct. 1989.
- [Jacobs *et al.*, 1990] P. Jacobs, J.-P. Laumond, and M. Taix. – A complete iterative motion planner for a car-like robot. – In *Journées de géométrie algorithmique*, Nice (F), June 1990. Inst. Nat. de la Recherche en Informatique et en Automatique.
- [Jacobs *et al.*, 1991] P. Jacobs, J.-P. Laumond, and A. Rege. – Non-holonomic motion planning for hilare-like mobile robots. – In M. Vidyasagar and M. Trivedi, editors, *Intelligent Robotics*, pages 338–347. McGraw-Hill, Bangalore (In), Jan. 1991. – *Proc. of the Int. Symp. on Intelligent Robotics*.
- [Jiang *et al.*, 1992] K. Jiang, L.D. Seneviratne, and S.W.E. Earles. – A motion strategy for a mobile robot with holonomic and non-holonomic constraints. – In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 461–468, Raleigh, NC (USA), July 1992.
- [Kant and Zucker, 1986] K. Kant and S. Zucker. – Toward efficient trajectory planning: the path-velocity decomposition. – *Int. Journal of Robotics Research*, 5(3):72–89, Fall 1986.
- [Latombe, 1990] J.-C. Latombe. – *Robot motion planning*. – Kluwer Academic Press, 1990.
- [Laumond, 1986] J.-P. Laumond. – Feasible trajectories for mobile robots with kinematic and environment constraints. – *Intelligent Autonomous Systems*, pages 346–354, Dec. 1986.
- [Laumond, 1987] J.-P. Laumond. – Finding collision-free smooth trajectories for a non-holonomic mobile robot. – In *Proc. of the Int. Joint Conf. on Artificial Intelligence*, pages 1120–1123, Milan (I), Aug. 1987.
- [Nilsson, 1969] N.J. Nilsson. – A mobile automaton: an application of artificial intelligence techniques. – In *Proc. of the Int. Joint Conf. on Artificial Intelligence*, pages 509–520, Washington, DC (USA), 1969.
- [Ó'Dúnlaing, 1987] C. Ó'Dúnlaing. – Motion planning with inertial constraints. – *Algorithmica*, 2:431–475, 1987.
- [Pan and Luo, 1990] T.-J. Pan and R.C. Luo. – Motion planning for mobile robots in a dynamic environment with moving obstacles. – In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 578–583, Cincinnati, OH (USA), May 1990.
- [Reeds and Shepp, 1990] J.A. Reeds and L.A. Shepp. – Optimal paths for a car that goes both forwards and backwards. – *Pacific Journal of Mathematics*, 145(2):367–393, 1990.
- [Sahar and Hollerbach, 1985] G. Sahar and J. H. Hollerbach. – Planning of minimum-time trajectories for robot arms. – In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 751–758, St Louis, MI (USA), March 1985.
- [Shih *et al.*, 1990] C.L. Shih, T.T. Lee, and W.A. Gruver. – Motion planning with time-varying polyhedral obstacles based on graph search and mathematical programming. – In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 331–337, Cincinnati, OH (USA), May 1990.
- [Shiller and Dubowsky, 1988] Z. Shiller and S. Dubowsky. – Global time optimal motions of robotic manipulators in the presence of obstacles. – In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 370–375, Philadelphia, PA (USA), Apr. 1988.
- [Shin and McKay, 1985] K.G. Shin and N.D. McKay. – Minimum-time control of robotic manipulators with geometric path constraints. – *IEEE Trans. Autom. Contr.*, 30:531–541, June 1985.
- [Tournassoud and Jehl, 1988] P. Tournassoud and O. Jehl. – Motion planning for a mobile robot with a kinematic constraint. – In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, Philadelphia, PA (USA), Apr. 1988.
- [Wilfong, 1988] G. Wilfong. – Motion planning for an autonomous vehicle. – In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 529–533, Philadelphia, PA (USA), Apr. 1988.
- [Xavier, 1992] P.G. Xavier. – *Provably-good approximation algorithms for optimal kinodynamic robot motion plans*. – PhD thesis, Cornell Univ., Ithaca, NY (USA), april 1992.