46

Advanced Robotics, Vol. 18, No. 10, pp. 1001–1024 (2004)
© VSP and Robotics Society of Japan 2004.
Also available online - www.vsppub.com

Full paper

Inevitable collision states — a step towards safer robots?

THIERRY FRAICHARD [1],* and HAJIME ASAMA [2]

[1] *National Research Institute in Computer Science and Control, INRIA Rhône-Alpes Research Unit, ZIRST, 655 avenue de l'Europe, Montbonnot, 38334 Saint Ismier Cedex, France*
[2] *Research into Artifacts, Center for Engineering, University of Tokyo, Tokyo, Japan*

Received 22 June 2004; accepted 25 July 2004

Abstract—An inevitable collision state for a robotic system can be defined as a state for which, no matter what the future trajectory followed by the system is, a collision with an obstacle eventually occurs. An inevitable collision state takes into account the dynamics of both the system and the obstacles, fixed or moving. The main contribution of this paper is to lay down and explore this novel concept (and the companion concept of inevitable collision obstacle). Formal definitions of the inevitable collision states and obstacles are given. Properties fundamental for their characterization are established. This concept is very general, and can be useful both for navigation and motion planning purposes (for its own safety, a robotic system should never find itself in an inevitable collision state). To illustrate the interest of this concept, it is applied to a problem of safe motion planning for a robotic system subject to sensing constraints in a partially known environment (i.e. that may contain unexpected obstacles). In safe motion planning, the issue is to compute motions for which it is guaranteed that, no matter what happens at execution time, the robotic system never finds itself in a situation where there is no way for it to avoid collision with an unexpected obstacle.

Keywords: Safety; navigation; motion planning; sensing constraints; collision avoidance.

1. INTRODUCTION

The configuration (a set of independent variables that uniquely determines the position and orientation of every point of the system [1]) space of a robotic system is the appropriate framework to address path planning problems where the focus is on the geometric aspects of motion planning (no collision between the system and the fixed obstacles of the workspace) [1, 2]. The state (a set of variables such that the knowledge of these variables at time t_0 together with the knowledge of the controls applied to the system for $t \geqslant t_0$ completely determines the behavior of the system for any time $t \geqslant t_0$ [3]) space, on the other hand, is more appropriate when it comes to address trajectory planning problems where the dynamics of the system

*To whom correspondence should be addressed. E-mail: thierry.fraichard@inrialpes.fr

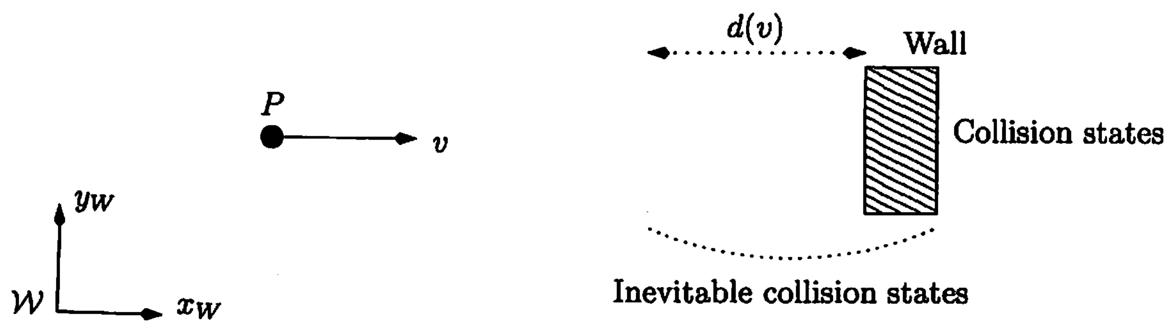


Figure 1. Collision states versus inevitable collision states.

is taken into account [4, 5]. Similarly, the time-state space is appropriate to address trajectory planning problems involving moving obstacles [6–8].

In the configuration space, the notion of forbidden or collision configurations, i.e. configurations yielding a collision, is well-known and so is the notion of configuration obstacles, i.e. the set of configurations yielding a collision between the system and a particular obstacle [1]. Transposing these notions in the state space, it is straightforward to define collision states and state obstacles (*idem* in the time-state space).

However, be it in state space or time-state space, it takes a simple example such as the one depicted in Fig. 1 to illustrate the interest of extending these notions so as to take into account the dynamics of the system by introducing the concept of inevitable collision states.

Considering Fig. 1, let P be a point mass that can only move to the right with a variable speed. A state of P is characterized by its position (x, y) and its speed v. If the workspace $\mathcal{W}$ features a wall, the states whose position corresponds to the wall are obviously collision states. On the other hand, assuming that it takes P a certain distance $d(v)$ to slow down and stop, the states corresponding to the wall and the states located at a distance less than $d(v)$ left of the wall are such that, when P is in such a state, no matter what it does in the future, a collision will occur. These states are inevitable collision states for P. Clearly, for P's own safety, when it is moving at speed v, it should never be in one of these inevitable collision states. The size of the inevitable collision states region, i.e. the grey region located to the left of the wall, depends on the distance $d(v)$ which in turns depends on the current speed of P. Assuming that $d(v)$ varies linearly with v, the complete set of inevitable collision states is a prism embedded in the state space $\mathcal{S}$ of P (Fig. 2).

In general, an inevitable collision state for a given robotic system can be defined as a state for which, no matter what the future trajectory followed by the system is, a collision eventually occurs with an obstacle of the environment. Similarly, it is possible to define an inevitable collision obstacle as the set of inevitable collision states yielding a collision with a particular obstacle.

Except for a brief mention of it in Ref. [9], this concept does not seem to have been considered before by the robotics community. A close idea can be found in [10]: it introduces braking prisms, i.e. subsets of the configuration space

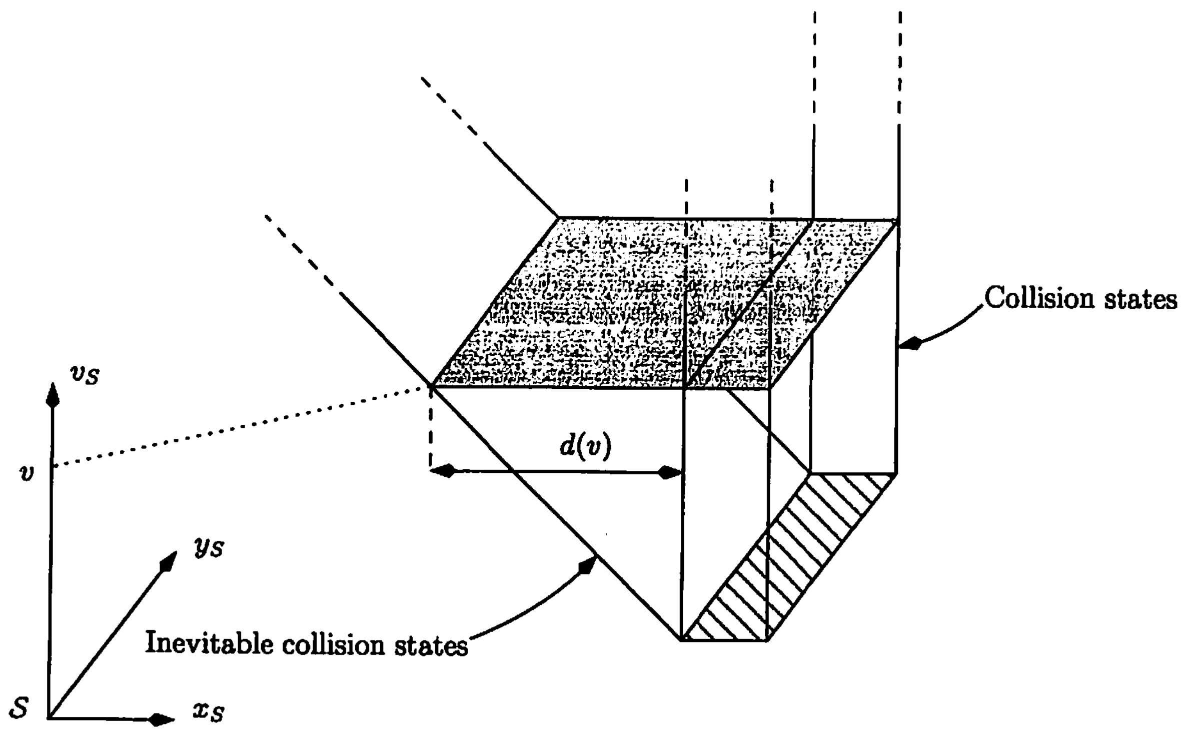


Figure 2. Full representation in the xyv state space $\mathcal{S}$ of P of the inevitable collision states corresponding to the situation depicted in Fig. 1.

associated with a given state of a robotic system and known to contain the braking trajectory corresponding to a given braking policy (picked up from a restricted set of braking policies). A configuration without any braking prism included in the free configuration space must be avoided because it would yield a collision no matter which braking policy is used. In a similar vein, Ref. [11] describes a trajectory planning scheme for a robotic system with a limited field of view. It characterizes robust states as states for which the robotic system can safely stop simply by braking even when placed in a environment with unknown moving obstacles (basically, the field of view is shrunken according to the moving obstacles' maximum possible speed). Also, inevitable collision states are to some extent related to the danger zone concept that can be found in the Air Traffic Control literature [12, 13]. Outside such danger zones, evasive maneuvers are provably safe.

All these approaches share the same principle: one or several evasive maneuvers are defined, and every state for which no such evasive maneuvers (usually braking maneuvers) are collision-free is labeled as being dangerous and is avoided. In a sense, the inevitable collision state concept encompasses all these approaches. It is general, it takes into account the dynamics of both the robotic system and the obstacles (fixed or moving, known or unknown), and it considers maneuvers other than braking maneuvers (as a matter of fact, it considers all possible maneuvers). We therefore believe that it is a concept worth exploring and that it can be very useful be it for motion planning or navigation purposes.

Consider navigation first (by navigation, we basically mean the problem of determining the elementary motion that the robotic system should perform during the next time-step). The primary concern of navigation is to ensure the safety of the robotic system. In a environment featuring moving obstacles, this safety concern is critical, and it is important to take into account both the dynamics of the robotic system and the future behavior of the moving obstacles. A number of research papers have addressed these issues recently [10, 14–19]. In this framework, the interest of the inevitable collision state concept is obvious. By design, inevitable collision states integrate the dynamics of both the robotic system and the obstacles, fixed or moving.

When it comes to motion planning, the inevitable collision state concept is also useful. Consider the problem of planning motions for a robotic system moving in a partially known environment. The system is subject to sensing constraints (a limited field of view) and it moves in an environment containing obstacles; some of them are known beforehand, while others are not (imagine a surveillance robot, it has a map of the building it must patrol, but it does not know *a priori* the position of the small furniture or if people are moving around). Based on the *a priori* information available, a nominal trajectory for the robotic system can be computed. However, what if, at execution time, the robotic system finds itself in a situation where an unknown obstacle is detected so late that avoiding it is impossible. The issue here is to compute safe motions, i.e. motions for which it is guaranteed that, no matter what happens at the execution time, the robotic system never finds itself in a situation where there is no way for it to avoid collision with an unexpected obstacle. This issue is related to the dependency that exists between motion planning and navigation, dependency which is usually ignored by motion planning systems (with the exception of Ref. [11]). We show by an example how this issue can be addressed using the inevitable collision state concept and how safe motions (in the sense given above) can be planned.

The main contribution of this paper is to lay down and explore the concept of inevitable collision states. To begin with, a formal definition of what inevitable collision states and inevitable collision obstacles are is given. Properties that are fundamental for their characterization are established (Section 3). To illustrate the use of these properties, a basic example is studied (Section 4). Finally, an example of an application of the inevitable collision state concept to safe motion planning is given (Section 5).

2. NOTATIONS AND PRELIMINARY DEFINITIONS

Before defining the inevitable collision states and obstacles, useful definitions and notations are introduced. Let $\mathcal{A}$ denote a robotic system. It is assumed that its dynamics can be described by a differential equation such as: $\dot{s} = f(s, u)$, where $s \in \mathcal{S}$ is the state of $\mathcal{A}$, $\dot{s}$ its time derivative and $u \in \mathcal{U}$ a control. $\mathcal{S}$ and $\mathcal{U}$, respectively, denote the state space and the control space of $\mathcal{A}$. Let $\phi \in \Phi$ denote a control input, i.e. a time-sequence of controls. ϕ represents a trajectory for $\mathcal{A}$.

Starting from an initial state s_0 (at time 0) and under the action of a control input ϕ, the state of $\mathcal{A}$ at time t is denoted by $\phi(s_0, t)$.

Given a control input ϕ and a state s_0 (at time 0), a state s is reachable from s_0 by ϕ iff $\exists t, \phi(s_0, t) = s$. Let $\mathcal{R}(s_0, \phi)$ denote the set of states reachable from s_0 by ϕ. Likewise, $\mathcal{R}(s_0)$ denotes the set of states s reachable from s_0, i.e. such that $\exists \phi, s \in \mathcal{R}(s_0, \phi)$:

$$\mathcal{R}(s_0, \phi) = \{s \in \mathcal{S} | \exists t, \phi(s_0, t) = s\},$$
$$\mathcal{R}(s_0) = \{s \in \mathcal{S} | \exists \phi, s \in \mathcal{R}(s_0, \phi)\}.$$

Introducing $\phi^{-1}(s_0, t)$ to denote the state s such that $\phi(s, t) = s_0$, it is possible to define $\mathcal{R}^{-1}(s_0)$ (respectively, $\mathcal{R}^{-1}(s_0, \phi)$), as the set of states from which it is possible to reach s_0 (respectively, to reach s_0 by ϕ):

$$\mathcal{R}^{-1}(s_0, \phi) = \{s \in \mathcal{S} | \exists t, \phi(s, t) = s_0 \Leftrightarrow \phi^{-1}(s_0, t) = s\},$$
$$\mathcal{R}^{-1}(s_0) = \{s \in \mathcal{S} | \exists \phi, s \in \mathcal{R}^{-1}(s_0, \phi)\}.$$

Let $\mathcal{W}$ denote the workspace of $\mathcal{A}$ ($\mathcal{W} = \mathbb{R}^2$ or $\mathbb{R}^3$); it contains a set of obstacles that are defined as closed subsets of $\mathcal{W}$. Let $\mathcal{WB}$ denote such an obstacle. When $\mathcal{WB}$ is moving, $\mathcal{WB}(t)$ represents the subset of $\mathcal{W}$ occupied by $\mathcal{WB}$ at time t. When $\mathcal{WB}$ is fixed, the time index is omitted: $\forall t, \mathcal{WB}(t) = \mathcal{WB}(0) = \mathcal{WB}$. In the configuration space, every obstacle has an image called a configuration obstacle which is the set of configurations yielding a collision between the robotic system and the obstacle considered [1]. Likewise, every obstacle has an image in the state space: the set of states yielding a collision between the robotic system and an obstacle $\mathcal{WB}(t)$ determines the state obstacle of $\mathcal{WB}(t)$ which is denoted $\mathcal{B}(t)$. $\mathcal{B}(t) = \{s \in \mathcal{S} | \mathcal{A}(s) \cap \mathcal{WB}(t) \neq \emptyset\}$, where $\mathcal{A}(s)$ denotes the closed subset of $\mathcal{W}$ occupied by $\mathcal{A}$ in state s. Once again, when $\mathcal{WB}$ is fixed, the time index is omitted. A state s is a collision state at time t iff $\exists \mathcal{B}, s \in \mathcal{B}(t)$. In this case, s is a collision state at time t with $\mathcal{B}$.

The rest of the article places itself in the state space framework. For the sake of simplicity, state obstacles are called obstacles only and the time index is indicated only when necessary.

3. INEVITABLE COLLISION STATES AND OBSTACLES

Based on the definitions and notations introduced in the previous section, the inevitable collision states and the inevitable collision obstacles are formally defined.

DEFINITION 1 (Inevitable Collision State). Given a control input ϕ, a state s is an inevitable collision state for ϕ iff $\exists t$ such that $\phi(s, t)$ is a collision state at time t. Now, a state s is an inevitable collision state iff $\forall \phi, \exists t$ such that $\phi(s, t)$ is a collision state at time t. Likewise, s is an inevitable collision state with $\mathcal{B}$ for ϕ iff $\exists t$ such that $\phi(s, t)$ is a collision state at time t with $\mathcal{B}$. Finally, s is an inevitable collision state with $\mathcal{B}$ iff $\forall \phi, \exists t$ such that $\phi(s, t)$ is a collision state at time t with $\mathcal{B}$.

DEFINITION 2 (Inevitable Collision Obstacle). Given an obstacle $\mathcal{B}$ and a control input ϕ, $ICO(\mathcal{B}, \phi)$, the inevitable collision obstacle of $\mathcal{B}$ for ϕ is defined as:

$$\begin{aligned} ICO(\mathcal{B}, \phi) &= \{s \in \mathcal{S} \mid s \text{ is an inevitable collision state with } \mathcal{B} \text{ for } \phi\} \\ &= \{s \in \mathcal{S} \mid \exists t, \phi(s, t) \text{ is a collision state at time } t \text{ with } \mathcal{B}\} \\ &= \{s \in \mathcal{S} \mid \exists t, \phi(s, t) \in \mathcal{B}(t)\}. \end{aligned}$$

Now, $ICO(\mathcal{B})$, the inevitable collision obstacle of $\mathcal{B}$, is defined as:

$$\begin{aligned} ICO(\mathcal{B}) &= \{s \in \mathcal{S} \mid s \text{ is an inevitable collision state with } \mathcal{B}\} \\ &= \{s \in \mathcal{S} \mid \forall \phi, \exists t, \phi(s, t) \text{ is a collision state at time } t \text{ with } \mathcal{B}\} \\ &= \{s \in \mathcal{S} \mid \forall \phi, \exists t, \phi(s, t) \in \mathcal{B}(t)\}. \end{aligned}$$

Based upon the two definitions above, the following property can be established. It shows that $ICO(\mathcal{B})$ can be derived from the $ICO(\mathcal{B}, \phi)$ for every possible control input ϕ.

PROPERTY 1 (Control Inputs Intersection).

$$ICO(\mathcal{B}) = \bigcap_{\Phi} ICO(\mathcal{B}, \phi).$$

Proof.

$$\begin{aligned} s \in ICO(\mathcal{B}) &\Leftrightarrow \forall \phi, \exists t, \phi(s, t) \text{ is a collision state at time } t \text{ with } \mathcal{B} \\ &\Leftrightarrow \forall \phi, s \in ICO(\mathcal{B}, \phi) \\ &\Leftrightarrow s \in \bigcap_{\Phi} ICO(\mathcal{B}, \phi). \end{aligned}$$ □

Assuming now that $\mathcal{B}$ is the union of a set of obstacles, $\mathcal{B} = \bigcup_i \mathcal{B}_i$, the following property can be established. It shows that $ICO(\mathcal{B}, \phi)$ can be derived from the $ICO(\mathcal{B}_i, \phi)$ for every subset $\mathcal{B}_i$.

PROPERTY 2 (Obstacles Union).

$$ICO\left(\bigcup_i \mathcal{B}_i, \phi\right) = \bigcup_i ICO(\mathcal{B}_i, \phi).$$

Proof.

$$\begin{aligned} s \in ICO\left(\bigcup_i \mathcal{B}_i, \phi\right) &\Leftrightarrow \exists t, \phi(s, t) \text{ is a collision state at time } t \text{ with } \bigcup_i \mathcal{B}_i \\ &\Leftrightarrow \exists \mathcal{B}_i, \exists t, \phi(s, t) \text{ is a collision state at time } t \text{ with } \mathcal{B}_i \\ &\Leftrightarrow \exists \mathcal{B}_i, s \in ICO(\mathcal{B}_i, \phi) \\ &\Leftrightarrow s \in \bigcup_i ICO(\mathcal{B}_i, \phi). \end{aligned}$$ □

Combining the two properties above, the following property is derived. It is the property that permits the formal characterization of the inevitable collision obstacles for a given robotic system.

PROPERTY 3 (ICO Characterization). *Let $\mathcal{B} = \bigcup_i \mathcal{B}_i$,*

$$ICO(\mathcal{B}) = \bigcap_{\Phi} \bigcup_{i} ICO(\mathcal{B}_i, \phi).$$

Proof.

$$ICO(\mathcal{B}) \overset{1}{=} \bigcap_{\Phi} ICO(\mathcal{B}, \phi) \overset{2}{=} \bigcap_{\Phi} \bigcup_{i} ICO(\mathcal{B}_i, \phi). \qquad \square$$

Consider Property 1 (and Property 3), it establishes that $ICO(\mathcal{B})$ can be derived from the $ICO(\mathcal{B}, \phi)$ for every possible control input ϕ. In general, there is an infinite number of control inputs which leaves little hope of being actually able to compute $ICO(\mathcal{B})$. Fortunately, it is possible to establish a property which is of a vital practical value since it shows how to compute a conservative approximation of $ICO(\mathcal{B})$ by using a subset only of the whole set of possible control inputs.

PROPERTY 4 (ICO Approximation). *Let $\mathcal{I}$ denote a subset of the set of possible control inputs Φ:*

$$ICO(\mathcal{B}) \subset \bigcap_{\mathcal{I}} ICO(\mathcal{B}, \phi).$$

Proof.

$$\begin{aligned} ICO(\mathcal{B}) &\overset{1}{=} \bigcap_{\mathcal{I} \cup \overline{\mathcal{I}}} ICO(\mathcal{B}, \phi) \\ &= \bigcap_{\mathcal{I}} ICO(\mathcal{B}, \phi) \cap \bigcap_{\overline{\mathcal{I}}} ICO(\mathcal{B}, \phi) \\ &\subseteq \bigcap_{\mathcal{I}} ICO(\mathcal{B}, \phi). \qquad \square \end{aligned}$$

The interest of these properties to characterize inevitable collision obstacles appears in the next sections.

4. BASIC CASE STUDY

The purpose of this section is to illustrate on a simple (and not necessary realistic!) example the notions introduced earlier. A more realistic example is dealt with later in Section 5.

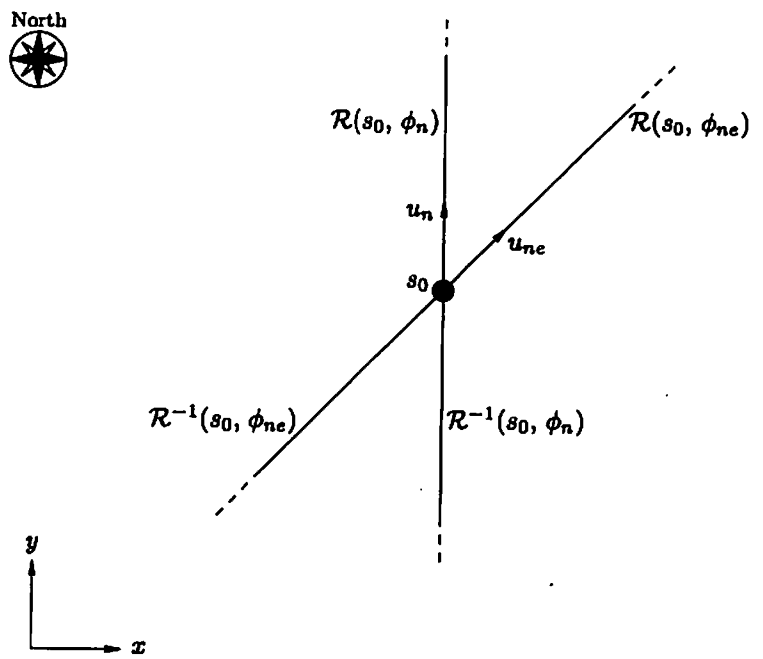

North
$\mathcal{R}(s_0, \phi_n)$
$\mathcal{R}(s_0, \phi_{ne})$
u_n
u_{ne}
s_0
$\mathcal{R}^{-1}(s_0, \phi_{ne})$
$\mathcal{R}^{-1}(s_0, \phi_n)$
y
x

Figure 3. Reachable states for the 'North, North–East' system.

4.1. *'North, North–East' system*

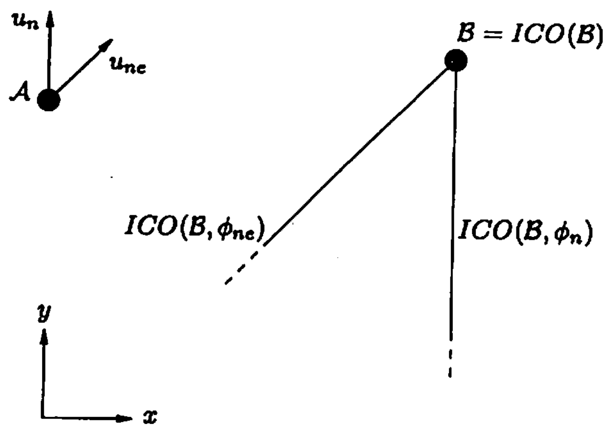
We consider the case of a planar point $\mathcal{A}$ that can move in two directions only (North and North–East) at constant unit speed (Fig. 3). A state of $\mathcal{A}$ is $s = (x, y) \in \mathbb{R}^2$ and a control u can take two values: either $u_n = \pi/2$ (North direction) or $u_{ne} = \pi/4$ (North–East direction). This simple system has only two possible constant control inputs: ϕ_n and ϕ_{ne}, they respectively correspond to motions in the North and North–East directions. Since both ϕ_n and ϕ_{ne} are constant, it means that once $\mathcal{A}$ has started to move in a given direction, it cannot change its motion direction anymore.

$\mathcal{R}(s_0)$, i.e. the set of states reachable from an initial state s_0, is easily defined in this case: it is the union of two half-lines starting at s_0 and extending respectively in the North and North–East directions: $\mathcal{R}(s_0) = \mathcal{R}(s_0, \phi_n) \cup \mathcal{R}(s_0, \phi_{ne})$. Likewise, $\mathcal{R}^{-1}(s_0)$, i.e. the set of states from which s_0 is reachable, is the union of two half-lines starting at s_0 and extending respectively in the South and South–West directions: $\mathcal{R}^{-1}(s_0) = \mathcal{R}^{-1}(s_0, \phi_n) \cup \mathcal{R}^{-1}(s_0, \phi_{ne})$ (Fig. 3).

The next sections show how to determine the inevitable collision obstacles corresponding to the 'North, North–East' system. We proceed step by step by considering fixed obstacles first and then moving obstacles. In each case, we address point obstacles first before moving to arbitrary obstacles.

4.2. *Fixed obstacle*

4.2.1. Point obstacle. Let $\mathcal{B}$ be a fixed point obstacle. According to Property 1, $ICO(\mathcal{B})$ is derived from the characterizations of $ICO(\mathcal{B}, \phi)$ for every possible control input ϕ. $ICO(\mathcal{B}, \phi)$ is trivially equal to $\mathcal{R}^{-1}(\mathcal{B}, \phi)$ and the following



u_n
u_{ne}
$\mathcal{A}$
$\mathcal{B} = ICO(\mathcal{B})$
$ICO(\mathcal{B}, \phi_{ne})$
$ICO(\mathcal{B}, \phi_n)$
y
x

Figure 4. Inevitable collision obstacle for a fixed point obstacle.

derivation is made (Fig. 4):

$$\begin{aligned}ICO(\mathcal{B}) &\overset{1}{=} ICO(\mathcal{B}, \phi_n) \cap ICO(\mathcal{B}, \phi_{ne})\\ &= \mathcal{R}^{-1}(\mathcal{B}, \phi_n) \cap \mathcal{R}^{-1}(\mathcal{B}, \phi_{ne})\\ &= \mathcal{B},\end{aligned}$$

which makes sense: unless $\mathcal{A}$ is already in collision with $\mathcal{B}$, $\mathcal{A}$ can always avoid collision with $\mathcal{B}$. The state corresponding to $\mathcal{B}$ is the only inevitable collision state.

4.2.2. Linear and arbitrary obstacle. Let us now assume that $\mathcal{B}$ is a fixed linear obstacle extending from point $\mathcal{B}_1$ to point $\mathcal{B}_2$. $\mathcal{B}$ is the union of a set of fixed point obstacles: $\mathcal{B} = \bigcup_i \mathcal{B}_i$. Now, $ICO(\mathcal{B})$ is derived using both Properties 1 and 2:

$$\begin{aligned}ICO(\mathcal{B}) &\overset{1}{=} ICO(\mathcal{B}, \phi_n) \cap ICO(\mathcal{B}, \phi_{ne})\\ &= ICO\Bigl(\bigcup_i \mathcal{B}_i, \phi_n\Bigr) \cap ICO\Bigl(\bigcup_i \mathcal{B}_i, \phi_{ne}\Bigr)\\ &\overset{2}{=} \bigcup_i ICO(\mathcal{B}_i, \phi_n) \cap \bigcup_i ICO(\mathcal{B}_i, \phi_{ne}).\end{aligned}$$

Consider Fig. 5, $\bigcup_i ICO(\mathcal{B}_i, \phi_n)$ is the region swept by $ICO(\mathcal{B}_i, \phi_n)$ for every point $\mathcal{B}_i$ between $\mathcal{B}_1$ and $\mathcal{B}_2$ (*idem* for $\bigcup_i ICO(\mathcal{B}_i, \phi_{nw})$). The intersection between these two regions yields a simple triangular region which is $ICO(\mathcal{B})$. Sure enough, when $\mathcal{A}$ is anywhere inside this region, no matter what it does, it eventually crashes against $\mathcal{B}$. Likewise, it is possible to characterize $ICO(\mathcal{B})$ for fixed obstacles with arbitrary shape (Fig. 6).

Note on Property 3. It is important to note that the obstacle union is nested within the control input intersection in Property 3. Accordingly, computing the inevitable collision obstacle of a set of distinct obstacles does not reduce to computing the union of the inevitable collision obstacles for each obstacle independently (Fig. 7). In other words, to determine whether a state is an inevitable collision state,

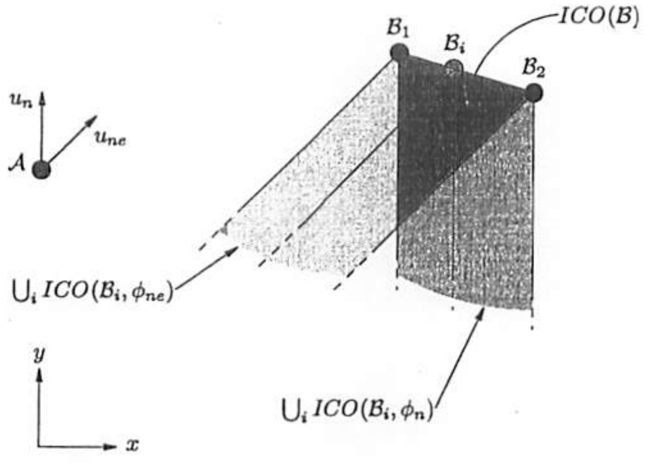


Figure 5. Inevitable collision obstacle for a fixed linear obstacle.

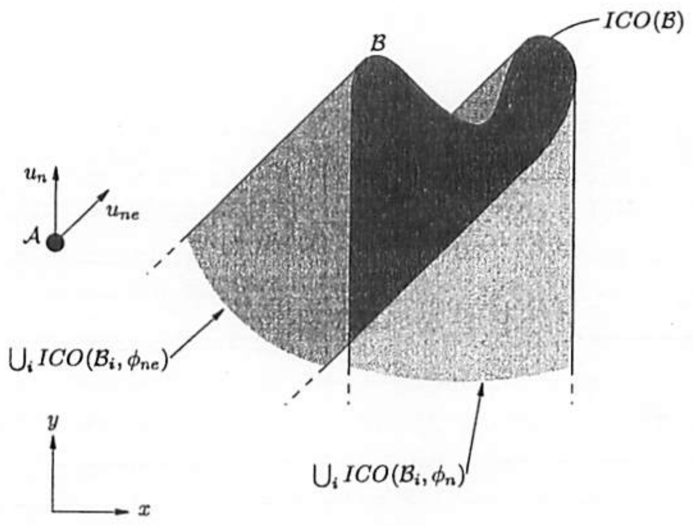

$ICO(\mathcal{B})$
$\mathcal{B}$
u_n
u_{ne}
$\mathcal{A}$
$\bigcup_i ICO(\mathcal{B}_i, \phi_{ne})$
y
x
$\bigcup_i ICO(\mathcal{B}_i, \phi_n)$

Figure 6. Inevitable collision obstacle for a fixed arbitrary obstacle.

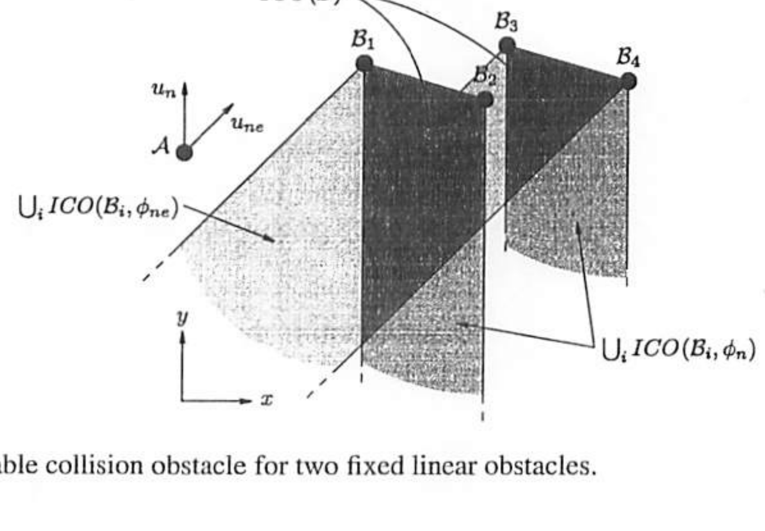


Figure 7. Inevitable collision obstacle for two fixed linear obstacles.

it is necessary to compute the inevitable collision obstacle of the union of the whole set of state obstacles $\mathcal{B}_i$ considered then as a single obstacle.

4.3. Moving obstacle

4.3.1. Point obstacle. Let $\mathcal{B}$ be a moving point obstacle. Recall that $\mathcal{B}(t)$ gives the position of $\mathcal{B}$ at time t. In order to characterize $ICO(\mathcal{B})$, we consider $\mathcal{B}$ as the union $\bigcup_t \mathcal{B}(t)$ and we proceed step by step as we did in the fixed obstacle case. Given a control input ϕ, let us characterize $ICO(\mathcal{B}, \phi)$ first: $ICO(\mathcal{B}, \phi) = \bigcup_t ICO(\mathcal{B}(t), \phi)$. Now, according to Definition 2, $ICO(\mathcal{B}(t), \phi)$ is the set of states s such that if $\mathcal{A}$ starts from s (at time 0) and is subject to the control input ϕ, it reaches $\mathcal{B}(t)$ (at time t). Such a state s belongs to $\mathcal{R}^{-1}(\mathcal{B}(t), \phi)$ and it is actually the unique solution of the equation $\phi(s, t) = \mathcal{B}(t) \Leftrightarrow s = \phi^{-1}(\mathcal{B}(t), t)$. In conclusion, $ICO(\mathcal{B}, \phi) = \bigcup_t \phi^{-1}(\mathcal{B}(t), t)$ and we have:

$$\begin{aligned}
ICO(\mathcal{B}) &\overset{1}{=} ICO(\mathcal{B}, \phi_{\mathrm{n}}) \cap ICO(\mathcal{B}, \phi_{\mathrm{ne}}) \\
&= ICO\left(\bigcup_t \mathcal{B}(t), \phi_{\mathrm{n}}\right) \cap ICO\left(\bigcup_t \mathcal{B}(t), \phi_{\mathrm{ne}}\right) \\
&\overset{2}{=} \bigcup_t ICO(\mathcal{B}(t), \phi_{\mathrm{n}}) \cap \bigcup_t ICO(\mathcal{B}(t), \phi_{\mathrm{ne}}) \\
&\doteq \bigcup_t \phi_{\mathrm{n}}^{-1}(\mathcal{B}(t), t) \cap \bigcup_t \phi_{\mathrm{ne}}^{-1}(\mathcal{B}(t), t).
\end{aligned}$$

Consider Fig. 8 where it is assumed that $\mathcal{B}$ has a linear motion at constant velocity. For both control inputs ϕ_{n} and ϕ_{ne}, $ICO(\mathcal{B}, \phi)$ is a linear curve starting from $\mathcal{B}(0)$ whose slope depends upon the relative velocities of $\mathcal{A}$ and $\mathcal{B}$. The application of Property 1 yields $ICO(\mathcal{B}) = \mathcal{B}(0)$.

4.3.2. Arbitrary obstacle. Let us now assume that $\mathcal{B}$ is a moving obstacle of arbitrary shape. $\mathcal{B}$ is the union of a set of moving point obstacles and we can write: $\mathcal{B} = \bigcup_i \bigcup_t \mathcal{B}_i(t)$. $ICO(\mathcal{B})$ is derived as before:

$$ICO(\mathcal{B}) \overset{1}{=} ICO(\mathcal{B}, \phi_{\mathrm{n}}) \cap ICO(\mathcal{B}, \phi_{\mathrm{ne}})$$

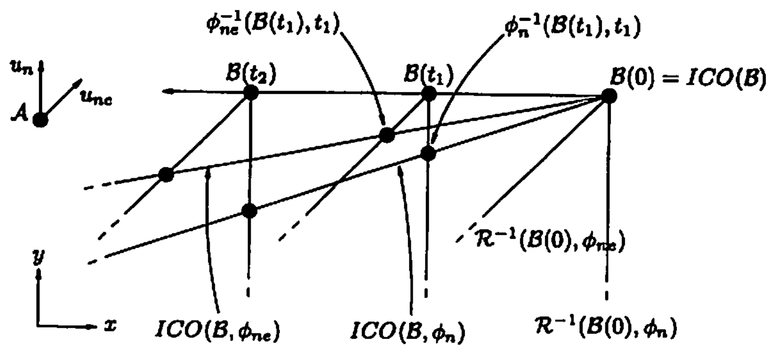

$\phi_{ne}^{-1}(\mathcal{B}(t_1), t_1)$
$\phi_{n}^{-1}(\mathcal{B}(t_1), t_1)$
u_n
u_{ne}
$\mathcal{A}$
$\mathcal{B}(t_2)$
$\mathcal{B}(t_1)$
$\mathcal{B}(0) = ICO(\mathcal{B})$
$\mathcal{R}^{-1}(\mathcal{B}(0), \phi_{ne})$
y
x
$ICO(\mathcal{B}, \phi_{ne})$
$ICO(\mathcal{B}, \phi_{n})$
$\mathcal{R}^{-1}(\mathcal{B}(0), \phi_{n})$

Figure 8. Inevitable collision obstacle for a moving point obstacle.

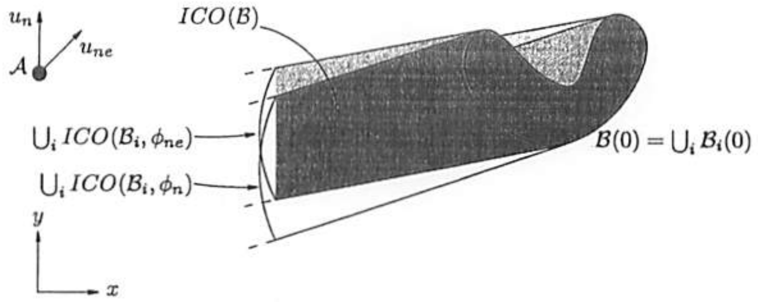

u_n
u_{ne}
$\mathcal{A}$
$ICO(\mathcal{B})$
$\bigcup_i ICO(\mathcal{B}_i, \phi_{ne})$
$\bigcup_i ICO(\mathcal{B}_i, \phi_n)$
$\mathcal{B}(0) = \bigcup_i \mathcal{B}_i(0)$
y
x

Figure 9. Inevitable collision obstacle for a moving arbitrary obstacle.

$$
\begin{aligned}
&= ICO\Big(\bigcup_i \bigcup_t \mathcal{B}_i(t), \phi_{\mathrm{n}}\Big) \cap ICO\Big(\bigcup_i \bigcup_t \mathcal{B}_i(t), \phi_{\mathrm{ne}}\Big) \\
&\overset{2}{=} \bigcup_i \bigcup_t ICO(\mathcal{B}_i(t), \phi_{\mathrm{n}}) \cap \bigcup_i \bigcup_t ICO(\mathcal{B}_i(t), \phi_{\mathrm{ne}}) \\
&= \bigcup_i \bigcup_t \phi_{\mathrm{n}}^{-1}(\mathcal{B}_i(t), t) \cap \bigcup_i \bigcup_t \phi_{\mathrm{ne}}^{-1}(\mathcal{B}_i(t), t).
\end{aligned}
$$

Figure 9 depicts the inevitable collision obstacle obtained for an arbitrary obstacle with a motion at constant velocity similar to that of the point obstacle in Section 4.3.1. Whenever $\mathcal{A}$ is inside the region $ICO(\mathcal{B})$ at time 0, no matter what it does in the future, it eventually collides with $\mathcal{B}$.

This simple example has illustrated how, thanks to the inevitable collision state concept, it is possible to characterize forbidden regions of the state-space: the inevitable collision obstacles. This characterization takes into account the dynamics of the robotic system and also the future behavior of the moving obstacles.

5. SAFE MOTION PLANNING APPLICATION

The purpose of this section is to demonstrate how the inevitable collision state concept can be used to address safe motion planning problems.

5.1. What is safe motion planning?

Consider the problem of planning motions for a vehicle $\mathcal{A}$ moving in a partially known environment that contains a set of fixed obstacles whose position is *a priori* known. It also contains unexpected obstacles, fixed or moving, whose position is not known beforehand. Finally, $\mathcal{A}$ is subject to sensing constraints, it has a limited field of view. In a given state s, $\mathcal{A}$ perceives only a subset $FoV(s)$ of its environment (Fig. 10, left). In this framework, what does planning a safe motion mean? Safe motions were defined earlier as motions for which it is guaranteed that, no matter what happens at the execution time, the vehicle never finds itself in a situation where there is no way for it to avoid collision with an unexpected obstacle. At the execution time, an unsafe situation occurs when an unexpected obstacle suddenly

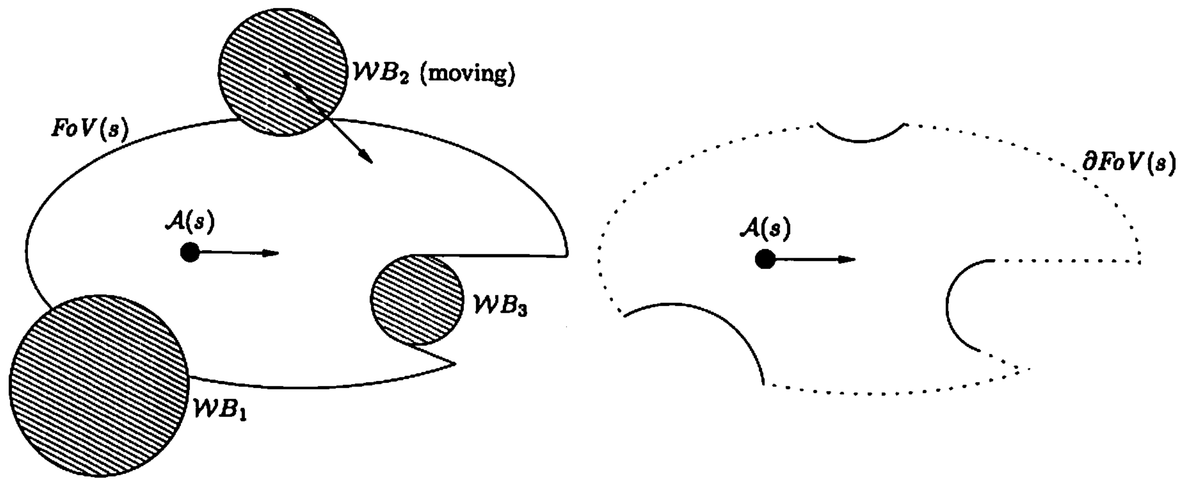


Figure 10. The field of view of $\mathcal{A}$ (left) and its boundary (right).

appears in the field of view of $\mathcal{A}$ and it is too late for $\mathcal{A}$ to brake or engage into an evasive maneuver. In other words, an unsafe situation occurs when an unexpected obstacle appears and $\mathcal{A}$ suddenly finds itself in an inevitable collision state.

At planning time, it is by definition impossible to characterize the inevitable collision states with respect to the unexpected obstacles. This characterization can be done with respect to the known obstacles only. However, it is possible to exploit the fact that unexpected obstacles appear on the boundary of the field of view only. When $\mathcal{A}$ is in state s, it is possible to compute the boundary of the field of view with respect to the *a priori* known obstacles. This boundary has two parts: the part corresponding to the known obstacles and the part corresponding to the limit of the field of view, e.g. the dotted curve in the right-hand side of Fig. 10. Let $\partial FoV(s)$ denote this part. What can be done then is to consider $\partial FoV(s)$ as a potential unexpected obstacle and to determine whether s is an inevitable collision state based on this assumption (it is precisely the approach taken in Ref. [11]).

This is the key to safe motion planning. A safe motion is a sequence of safe states where a safe state s is defined as a state which is not an inevitable collision state with respect to the known obstacles (whether visible or not) and with respect to $\partial FoV(s)$ treated as an unexpected obstacle, in other words:

DEFINITION 3 (Safe State). s is safe state iff $s \notin ICO(\bigcup \mathcal{B}_i \cup \mathcal{S}(\partial FoV(s)))$, where $\mathcal{S}(\partial FoV(s))$ represents the image of $\partial FoV(s)$ in the state space $\mathcal{S}$.

In the definition above, it is worth noting that *a priori* knowledge about the unexpected obstacles is required in order to compute $ICO(\mathcal{S}(\partial FoV(s)))$. Indeed, recall that the inevitable collision obstacle associated with a given obstacle is different if the obstacle is fixed or moving. Such preliminary knowledge determines the characteristic of $\mathcal{S}(\partial FoV(s))$ which in turn determines $ICO(\mathcal{S}(\partial FoV(s)))$. It may be assumed, for instance, that the unexpected obstacles are always fixed. It could also be assumed that the unexpected obstacles are moving. In such case, additional information is required regarding their potential moving direction, speed range, future behavior, etc. (it is the case in Ref. [11] where the moving obstacles

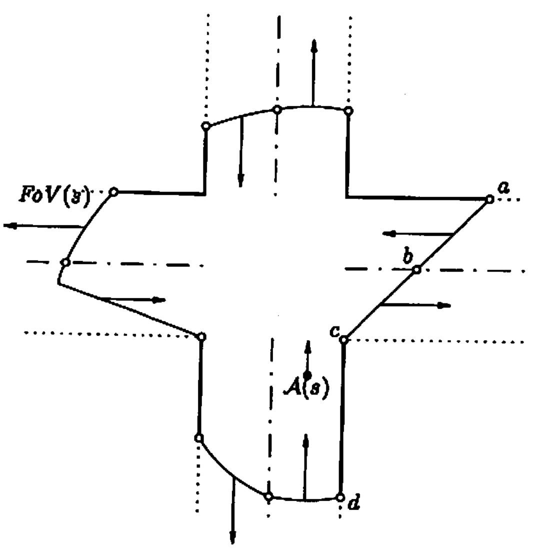


Figure 11. The field of view of $\mathcal{A}$ placed in a roadway-like environment.

are assumed to move freely in every direction up to a maximum speed). Once this information is available, it can be used to compute $ICO(\mathcal{S}(\partial FoV(s)))$.

Thanks to its generality, the inevitable collision state concept permits us to deal with situations as complex as the one depicted in Fig. 11: $\mathcal{A}$ moves in a roadway-like environment with known fixed obstacles (the limits of the roadway) and unexpected moving obstacles (the other vehicles). It is assumed that the moving obstacles obey the highway code and therefore follow the environment lanes at prescribed speeds (this is the *a priori* knowledge). In such a situation, $\partial FoV(s)$ is split in a number of parts with different characteristics depending on the location of the boundary part with respect to the environment lanes. In Fig. 11 for instance, the part *ab* (respectively, *bc*) corresponds to potential moving obstacles travelling to the left (respectively, to the right). The part *cd* corresponds to a fixed obstacle (the limits of the roadway).

The next sections present a worked-out example of safe motion planning for a car-like vehicle in a partially known environment with fixed obstacles only. The problem is defined in Section 5.2. Section 5.3 presents the computation of the inevitable collision obstacles and Section 5.4 details the safe motion planning algorithm.

5.2. Statement of the problem

Let us consider a robotic system $\mathcal{A}$ whose shape is a disk of radius $r_{\mathcal{A}}$. $\mathcal{A}$ moves like a car-like vehicle and its dynamics follows a bicycle model. A state of $\mathcal{A}$ is defined by the 4-tuple $s = (x, y, \theta, v)$ where (x, y) are the coordinates of the rear wheel, θ is the main orientation of $\mathcal{A}$ and v is the linear velocity of the front wheel (Fig. 12). A control of $\mathcal{A}$ is defined by the couple (u^{ξ}, u^{v}) where u^{ξ} is the steering angle and u^{v} the linear acceleration. The motion of $\mathcal{A}$ is governed by the following

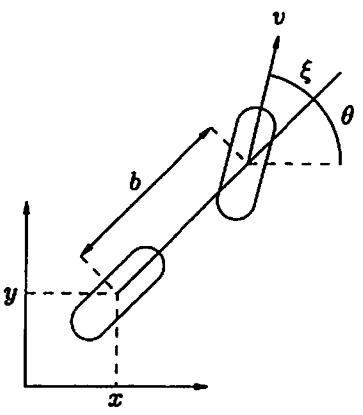


Figure 12. The car-like vehicle $\mathcal{A}$ (bicycle model).

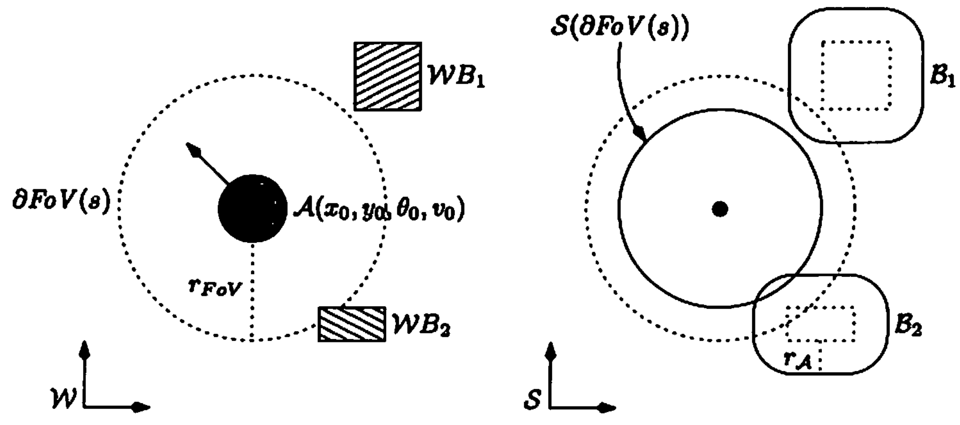


Figure 13. $\mathcal{A}$ in its workspace $\mathcal{W}$ (left), and the corresponding two-dimensional ($\theta = \theta_0$, $v = v_0$) slice of its state space $\mathcal{S}$ (right).

differential equations:

$$\begin{cases} \dot{x} = v\cos\theta\cos u^{\xi} \\ \dot{y} = v\sin\theta\cos u^{\xi} \\ \dot{\theta} = v\sin u^{\xi}/b \\ \dot{v} = u^{v}, \end{cases}$$

with $|u^{\xi}| \leqslant \xi_{\max}$ and $|u^{v}| \leqslant u^{v}_{\max}$. b is the wheelbase of $\mathcal{A}$.

$\mathcal{A}$ moves on a planar workspace $\mathcal{W}$ cluttered up with a set of fixed polygonal obstacles $\mathcal{W}B_i$. Part of the obstacles are *a priori* known while the others are not. $\mathcal{A}$ is equipped with an omnidirectional sensor with a limited range r_{FoV}.

The state space $\mathcal{S}$ of $\mathcal{A}$ is four-dimensional. It is not attempted to compute the inevitable collision obstacles in the full four-dimensional state space. Instead, the structure of $\mathcal{S}$ is exploited and the inevitable collision obstacles are computed in two-dimensional slices of $\mathcal{S}$ only. The slices considered are slices with constant θ and v. Such slices are interesting because it is straightforward to compute, for such a slice, the state obstacles $\mathcal{B}_i$ and $\mathcal{S}(\partial FoV(s))$, i.e. the image in $\mathcal{S}$ of the boundary of the sensor field of view. Since $\mathcal{A}$ is a disk of radius $r_{\mathcal{A}}$, $\mathcal{B}_i$ is obtained by isotropically growing $\mathcal{W}B_i$ of $r_{\mathcal{A}}$ [20]. Accordingly, $\mathcal{B}_i$ is a generalized polygon, its boundary is made up of straight segments and circular arcs of radius $r_{\mathcal{A}}$. Likewise,

$\mathcal{S}(\partial FoV(s))$ is obtained by shrinking $\partial FoV(s)$ of $r_{\mathcal{A}}$. It is, therefore, a disk of radius $r_{\mathrm{FoV}} - r_{\mathcal{A}}$ (Fig. 13).

5.3. *Inevitable collision obstacles*

A prerequisite to safe motion planning is to have a characterization of the inevitable collision states for $\mathcal{A}$ or, similarly, a characterization of the inevitable collision obstacles. The car-like vehicle $\mathcal{A}$ is unfortunately much more complicated a system than the 'North, North–East' one. Chiefly, the fact that the number of possible control inputs for $\mathcal{A}$ is infinite makes it difficult to use Property 1 directly in order to compute the inevitable collision obstacles.

Fortunately, it is possible to take advantage of Property 4 in order to compute a conservative approximation of the inevitable collision obstacles (conservative in the sense that the actual inevitable collision obstacle is included in the approximated one). To do so, only a finite subset $\mathcal{I}$ of the whole set of possible control inputs Φ is considered. The subset $\mathcal{I}$ selected contains the control inputs ϕ of arbitrary duration with constant steering angle u^ξ and constant linear acceleration u^v:

$$\mathcal{I} = \{\phi \in \Phi | \forall T \in \mathbb{R}^+, \forall t \in [0, T], \phi(t) = (u^\xi, u^v)\}.$$

As far as $\mathcal{A}$ is concerned, it corresponds to simple evasive maneuvers with fixed wheel orientation and changing velocity (constantly accelerating or decelerating). It includes the braking trajectories, i.e. trajectories where $\mathcal{A}$'s velocity becomes and remains null, but not only. It also includes trajectories where $\mathcal{A}$ brakes and move in reverse.

Let $ICO_{\mathcal{I}}(\mathcal{B})$ denote $\bigcap_{\mathcal{I}} ICO(\mathcal{B}, \phi)$. $ICO_{\mathcal{I}}(\mathcal{B})$ is the conservative approximation of $ICO(\mathcal{B})$. The characterization of $ICO_{\mathcal{I}}(\mathcal{B})$ is done in the next sections. Once again, we proceed in a step by step manner by considering different families of control inputs ϕ. First, $\mathcal{I}$ is split into two subsets $\mathcal{I}_S$ and $\mathcal{I}_T$ corresponding, respectively, to control inputs for which $\mathcal{A}$ is moving straight, i.e. $u^\xi = 0$, and control inputs for which $\mathcal{A}$ is turning, i.e. $u^\xi \neq 0$. Then, the set of control inputs $\mathcal{I}_T^\xi$ is introduced. It is the set of control inputs for which $\mathcal{A}$ is turning with the steering angle u^ξ.

Sections 5.3.1 and 5.3.2 detail how to compute $ICO_{\mathcal{I}_S}(\mathcal{B})$ and $ICO_{\mathcal{I}_T^\xi}(\mathcal{B})$, respectively. Then, Section 5.3.3 presents how to determine $ICO_{\mathcal{I}}(\mathcal{B})$.

5.3.1. *Computing $ICO_{\mathcal{I}_S}(\mathcal{B})$ ($\mathcal{A}$ is moving straight).*

As mentioned earlier in Section 5.2, it is not attempted to compute the inevitable collision obstacles in the full four-dimensional state space $\mathcal{S}$. Two-dimensional slices of $\mathcal{S}$ are considered instead: slices of constant orientation θ and velocity v. In such a (θ, v) slice, the obstacles are represented by generalized polygons.

Let us first consider the (θ, v) slice and a particular point $\mathcal{B}_i$ of an obstacle $\mathcal{B}$. The set of control inputs $\mathcal{I}_S$ is further split into two subsets $\mathcal{I}_S^+$ and $\mathcal{I}_S^-$, respectively, corresponding to control inputs for which $\mathcal{A}$ is accelerating, i.e. $u^v \geqslant 0$, and

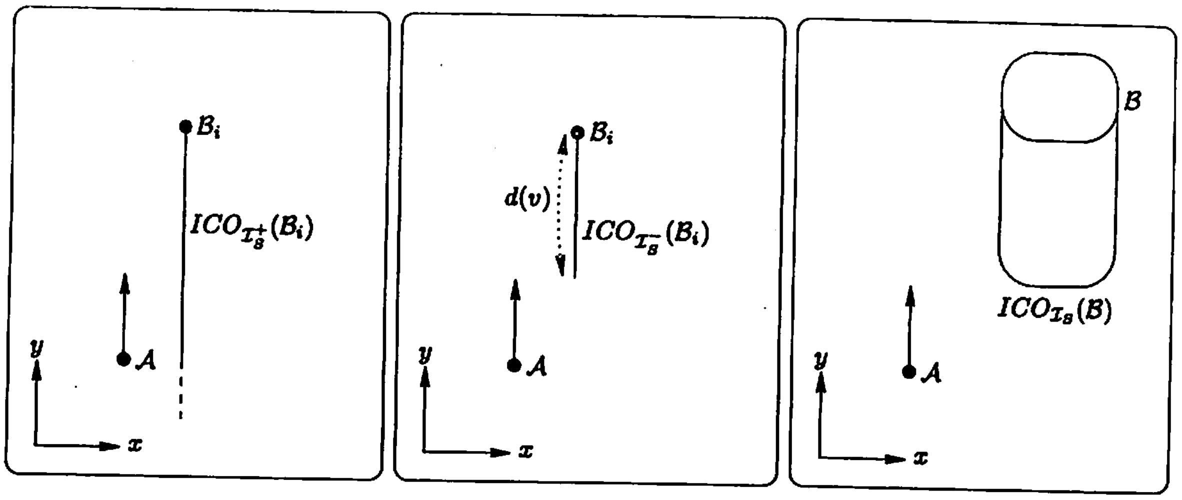

$\mathcal{B}_i$
$ICO_{\mathcal{I}_S^+}(\mathcal{B}_i)$
$\mathcal{A}$
y
x
$\mathcal{B}_i$
$d(v)$
$ICO_{\mathcal{I}_S^-}(\mathcal{B}_i)$
$\mathcal{A}$
y
x
$\mathcal{B}$
$ICO_{\mathcal{I}_S}(\mathcal{B})$
$\mathcal{A}$
y
x

Figure 14. $ICO_{\mathcal{I}_S}(\mathcal{B})$ when $\mathcal{A}$ is moving straight. Point obstacle case and $\mathcal{A}$ accelerating (left). Point obstacle case and $\mathcal{A}$ decelerating with a braking distance $d(v)$ (middle). Generalized polygonal obstacle case and $\mathcal{A}$ accelerating or decelerating (right).

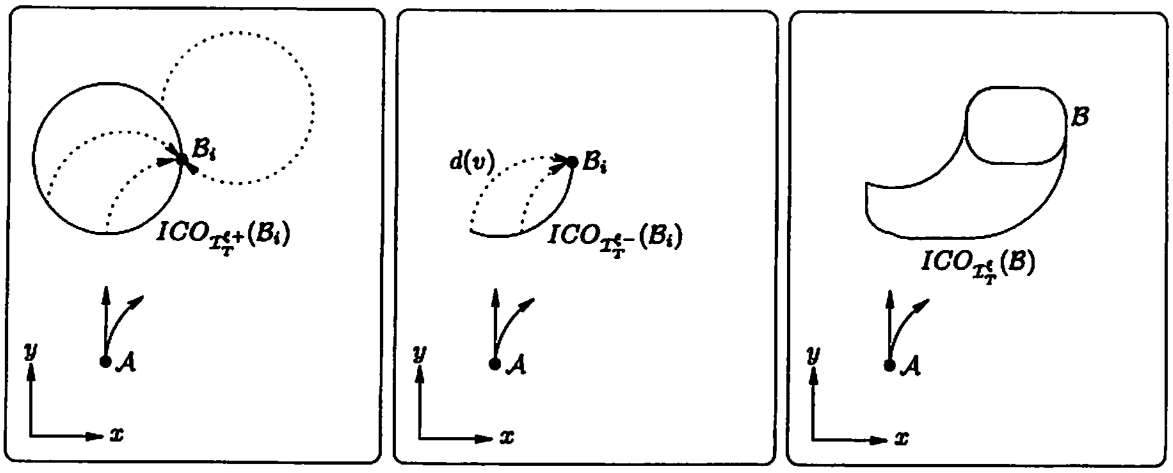
decelerating, i.e. $u^v < 0$. When $\mathcal{A}$ is moving straight and accelerating, it eventually crashes into $\mathcal{B}_i$ as soon as its orientation points towards $\mathcal{B}_i$. Accordingly, $ICO_{\mathcal{I}_S^+}(\mathcal{B}_i)$ is simply the half-line starting from $\mathcal{B}_i$ in the $-\theta$ direction (Fig. 14, left). Now, when $\mathcal{A}$ is moving straight and decelerating, it eventually crashes into $\mathcal{B}_i$ iff its orientation points towards $\mathcal{B}_i$ and its distance to $\mathcal{B}_i$ is less than $d(v)$, the minimum braking distance of $\mathcal{A}$: $d(v) = v^2/2u^v_{\max}$ with $u^v_{\max}$ the maximum linear deceleration. Accordingly, $ICO_{\mathcal{I}_S^-}(\mathcal{B}_i)$ is simply the segment of length $d(v)$ starting from $\mathcal{B}_i$ in the $-\theta$ direction (Fig. 14, middle). Finally, $ICO_{\mathcal{I}_S}(\mathcal{B}_i)$, which is the intersection between $ICO_{\mathcal{I}_S^+}(\mathcal{B}_i)$ and $ICO_{\mathcal{I}_S^-}(\mathcal{B}_i)$, reduces to $ICO_{\mathcal{I}_S^-}(\mathcal{B}_i)$, i.e. the segment of length $d(v)$ starting from $\mathcal{B}_i$ in the $-\theta$ direction.

Let us consider now the whole obstacle $\mathcal{B} = \bigcup_i \mathcal{B}_i$. Computing $ICO_{\mathcal{I}_S}(\mathcal{B})$ is straightforward. As per Property 2, it is, therefore, the union of $ICO_{\mathcal{I}_S}(\mathcal{B}_i)$ for every point $\mathcal{B}_i$ of $\mathcal{B}$. It is, therefore, the convolution between $\mathcal{B}$ and the segment of length $d(v)$ and direction $-\theta$. More precisely, it is the Minkowski Sum between $\mathcal{B}$ and the segment of length $d(v)$ starting from $(0, 0)$ in the $-\theta$ direction (Fig. 14, right). The Minkowski sum of two sets A and B in a vector space is equal to $\{a + b : a \in A, b \in B\}$ [21]. When $\mathcal{B}$ is a generalized polygon, $ICO_{\mathcal{I}_S}(\mathcal{B})$ is also a generalized polygon [20]. An efficient algorithm to compute the Minkowski Sum between generalized polygons can be found in Ref. [1].

5.3.2. *Computing $ICO_{\mathcal{I}_T^\xi}(\mathcal{B})$ ($\mathcal{A}$ is turning).*

Computing $ICO_{\mathcal{I}_T^\xi}(\mathcal{B})$ is achieved in a way similar to that of the straight motion case. $\mathcal{I}_T^\xi$ is split into two subsets $\mathcal{I}_T^{\xi+}$ and $\mathcal{I}_T^{\xi-}$, respectively, corresponding to control inputs for which $\mathcal{A}$ is accelerating, i.e. $u^v \geqslant 0$, and decelerating, i.e. $u^v < 0$.

Let us first consider the (θ, v) slice and a particular point $\mathcal{B}_i$ of an obstacle $\mathcal{B}$. When $\mathcal{A}$ is turning with the steering angle u^ξ and accelerating, it follows a circle of radius $b/\tan u^\xi$. $\mathcal{A}$ eventually crashes into $\mathcal{B}_i$ as soon as the circle it follows



$\mathcal{B}_i$
$ICO_{\mathcal{I}_T^\xi+}(\mathcal{B}_i)$
$d(v)$
$\mathcal{B}_i$
$ICO_{\mathcal{I}_T^\xi-}(\mathcal{B}_i)$
$\mathcal{B}$
$ICO_{\mathcal{I}_T^\xi}(\mathcal{B})$
$\mathcal{A}$
y
x

Figure 15. $ICO_{\mathcal{I}_T^\xi}(\mathcal{B})$ when $\mathcal{A}$ is turning. Point obstacle case and $\mathcal{A}$ accelerating (left). Point obstacle case and $\mathcal{A}$ decelerating with a braking distance $d(v)$ (middle). Generalized polygonal obstacle case and $\mathcal{A}$ accelerating or decelerating (right).

intersects $\mathcal{B}_i$. A straightforward geometric analysis shows that $ICO_{\mathcal{I}_T^\xi+}(\mathcal{B}_i)$ is the circle of radius $b/\tan u^\xi$ tangent to $\mathcal{B}_i$ with a tangent orientation at $\mathcal{B}_i$ of θ (Fig. 15, left). Now, when $\mathcal{A}$ is turning with the steering angle u^ξ and decelerating, it eventually crashes into $\mathcal{B}_i$ iff it is on a collision course and its distance to $\mathcal{B}_i$ is less than $d(v)$. Accordingly, $ICO_{\mathcal{I}_T^\xi-}(\mathcal{B}_i)$ is the circular arc of radius $b/\tan u^\xi$ and arc length $d(v)$ starting from $\mathcal{B}_i$ in the $-\theta$ direction (Fig. 15, middle). Finally, $ICO_{\mathcal{I}_T^\xi}(\mathcal{B}_i)$, which is the intersection between $ICO_{\mathcal{I}_T^\xi+}(\mathcal{B}_i)$ and $ICO_{\mathcal{I}_T^\xi-}(\mathcal{B}_i)$, reduces to $ICO_{\mathcal{I}_T^\xi-}(\mathcal{B}_i)$, i.e. the circular arc of radius $b/\tan u^\xi$ and arc length $d(v)$ starting from $\mathcal{B}_i$ in the $-\theta$ direction.

Let us consider now the whole obstacle $\mathcal{B} = \bigcup_i \mathcal{B}_i$. As in the straight motion case, $ICO_{\mathcal{I}_T^\xi}(\mathcal{B})$ is the Minkowski Sum between $\mathcal{B}$ and the circular arc of radius $b/\tan u^\xi$ and arc length $d(v)$ starting from $(0, 0)$ in the $-\theta$ direction (Fig. 15, right). Once again, when $\mathcal{B}$ is a generalized polygon, $ICO_{\mathcal{I}_T^\xi}(\mathcal{B})$ is also a generalized polygon.

5.3.3. Computing $ICO_\mathcal{I}(\mathcal{B})$. The two previous sections have characterized the inevitable collision obstacles for different subsets of $\mathcal{I}$, the whole set of control inputs considered. The final characterization of the inevitable collision obstacles is determined using:

$$ICO_\mathcal{I}(\mathcal{B}) = \bigcap_{\mathcal{X}\in\{\mathcal{I}_S,\mathcal{I}_T^\xi\}} ICO_\mathcal{X}(\mathcal{B}),$$

which amounts to computing the intersection between a set of generalized polygons. Such intersection computation can be carried out efficiently using software packages such as LEDA [22]. Figure 16 depicts the result obtained for a single linear obstacle and Fig. 17 for two distinct linear obstacles (see the note on Property 3 in Section 4.2.2).

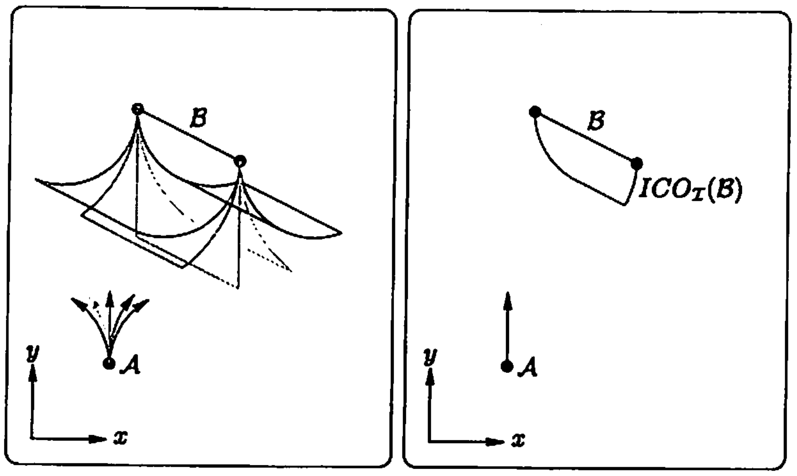


Figure 16. $ICO_{\mathcal{X}}(\mathcal{B})$ for a number of control input families $\mathcal{X}$ of $\mathcal{I}$ with different u^{ξ} values (left). $ICO_{\mathcal{I}}(\mathcal{B})$ (right).

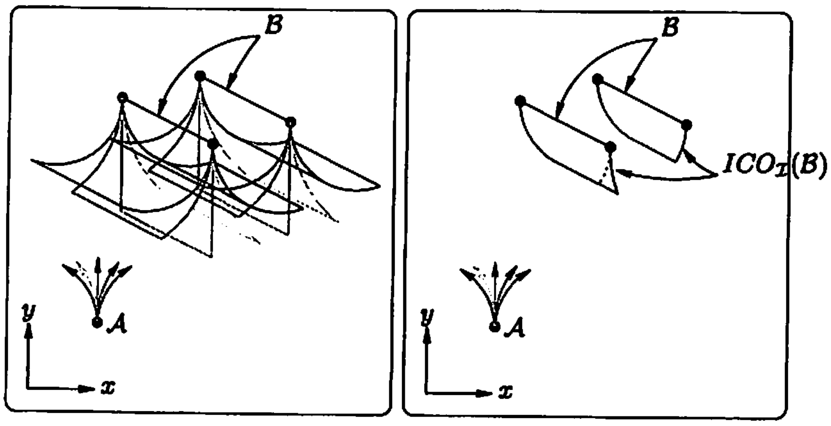


Figure 17. $ICO_{\mathcal{X}}(\mathcal{B})$ for a number of control input families $\mathcal{X}$ of $\mathcal{I}$ with different u^{ξ} values (left). $ICO_{\mathcal{I}}(\mathcal{B})$.

Note that what is actually represented on the right-hand side of Figs 16 and 17 is only a two-dimensional slice of $ICO_{\mathcal{I}}(\mathcal{B})$. Recall that $ICO_{\mathcal{I}}(\mathcal{B})$ is defined in the four-dimensional state-space of $\mathcal{A}$. The slice depicted is the $(\theta = \pi/2, v)$ slice. When $\mathcal{A}$ has an orientation $\pi/2$ and a velocity v, it inevitably crashes against $\mathcal{B}$ as soon as it is located in the region $ICO_{\mathcal{I}}(\mathcal{B})$ depicted. The slices for other values of θ and v are obtained similarly.

5.4. Safe motion planning

Thanks to the results presented above, it is now possible to determine whether a state is safe or not. As far as solving the motion planning problem at hand is concerned, it was decided to use a classical motion planning scheme based on the Rapidly-Exploring Random Tree algorithm [23]. Such an algorithm explores the state-space by incrementally expanding a tree rooted at the initial state. The tree is expanded through elementary motions in randomly selected directions. Such an algorithm is very efficient at exploring high-dimensional spaces.

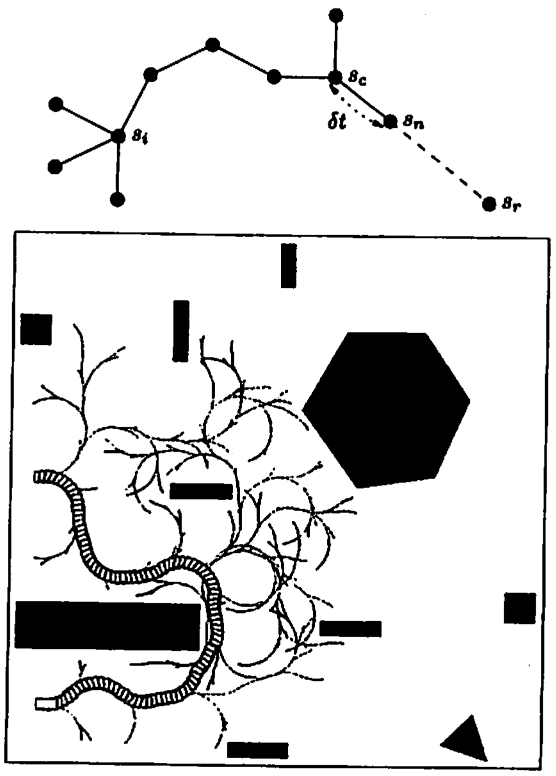


Figure 18. The Rapidly-Exploring Random Tree algorithm [23].

The top part of Fig. 18 sketches one step of the planning algorithm: a state s_r is picked up randomly and the closest node of the tree is found, say s_c. An elementary trajectory is then computed in the direction of s_r and it is checked for safety. If it is safe, the state at the end of this elementary trajectory, say s_n, becomes a new node of the tree. The process is repeated until the goal state is reached. The bottom part of Fig. 18 depicts the result of this kind of exploration.

Figure 19 presents some preliminary safe motion planning results obtained for the car-like vehicle $\mathcal{A}$. The field of view of $\mathcal{A}$ is a rectangular area (visible at a state along the result trajectories).

In the left part of Fig. 19, the trajectory obtained is collision-free only (the sensing constraints and the possible presence of unexpected fixed obstacles is not taken into account). In the right part of Fig. 19, the trajectory obtained is collision-free too but it is also safe, i.e. it is a sequence of safe states (in the sense of Definition 3). It does take into account the limits of the field of view and the possible presence of unexpected fixed obstacles.

Remember that the exploration scheme is random. It accounts for the strange twists and turns of the trajectories obtained. However, it can be noticed that the safe trajectory does not graze the obstacles (especially near the end of the two walls). This makes sense — suppose you have to pass the corner of a wall. The wall prevents you from seeing what is on the other side of the corner. So, if you believe that there may be unexpected obstacles on the other side, you have two strategies possible:

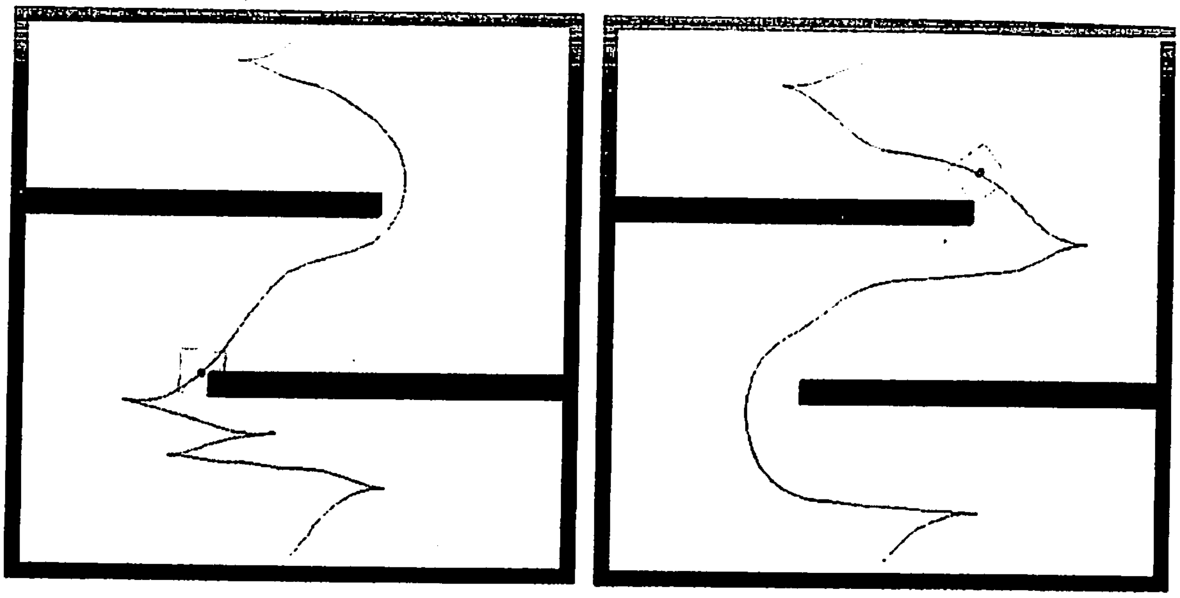


Figure 19. Safe motion planning results.

(i) Graze the corner while slowing down so that when you pass the corner, your speed is slow enough for you to stop before hitting a possible unexpected obstacle.

(ii) Stay away from the corner so as to have a better view of what is on the other side. In this case, you do not have to slow down.

In our experiments, the goal was to optimize the time of the trajectory. It naturally resulted in a solution trajectory following the second strategy and the trajectory obtained is safe. At execution time, no matter how many unexpected fixed obstacles are placed in the environment, it is guaranteed that, when such an unexpected obstacle is detected, $\mathcal{A}$ is not in an inevitable collision state, it can avoid the unexpected obstacle.

Future experiments will concern the safety with respect to unexpected moving obstacles. In this case, it is necessary to have some *a priori* knowledge about the moving obstacles, e.g. the maximum speed they can have, their expected motion direction, etc. This information is required to compute the inevitable collision obstacle corresponding to the moving obstacles (*cf.* Section 5).

6. DISCUSSION AND CONCLUSIONS

This paper has introduced the novel concept of inevitable collision states for a given robotic system, i.e. states for which, no matter what the future trajectory followed by the system is, a collision eventually occurs with an obstacle of the environment. An inevitable collision state takes into account the dynamics of both the robotic system and the obstacles, fixed and moving.

The main contribution of this paper was to lay down and explore this novel concept (along with a companion concept, that of inevitable collision obstacle).

A formal definition of what inevitable collision states and obstacles are was given. Properties that are fundamental for their characterisation were established. This concept is very general and we believe it can be useful both for navigation and motion planning purposes. An example of its application to safe motion planning was given. However, there are still a lot of issues to be addressed.

For a start, like its configuration space counterpart, the inevitable collision state concept faces the 'curse of dimensionality', i.e. the complexity of characterizing the inevitable collision states of high-dimensional robotic systems. The approximation property is but a partial answer to this complexity problem. This approximation property raises the question of the quality of the approximation obtained by considering a particular subset of the whole set of possible future trajectories for the robotic system at hand. It is true that if the approximation is too coarse, you might end up with most states being labeled as inevitable collision states. It may or may not be an issue depending on the problem at hand. For instance, in the safe motion planning problem presented in the article, should most of the states be inevitable collision states, it might indicate that the on-board sensing device has too small a range or that the evasive maneuvers selected are not appropriate.

Property 3 gives a general characterization of the inevitable collision states concept. However, it does not yield a general method to compute the inevitable collision states for arbitrary systems. Thus, for the two examples addressed in the paper, i.e. the 'North, North–East' system and the car-like system, an appropriate characterization of the inevitable collision states was performed. It was exact for the 'North, North–East' system and approximate for the car-like system. In the latter case, by considering two-dimensional slices of the four-dimensional state space of the car-like system, it proved possible to efficiently compute the inevitable collision states within such a two-dimensional slice. *Ad hoc* characterizations might allow to consider complex robotic systems. However, it could be interesting (at least from a theoretical point of view) to design such a generic inevitable collision states computing algorithm.

Acknowledgements

This work was partially supported by the Japan Society for the Promotion of Science and Lafmi, the French–Mexican Computer Science Laboratory. An earlier version of this work appeared in Ref. [24].

REFERENCES

1. J.-C. Latombe, *Robot Motion Planning*. Kluwer, Dordrecht (1991).
2. T. Lozano-Perez, Spatial planning, a configuration space approach, *IEEE Trans. Comput.* **32**, 108–120 (1983).
3. K. Ogata, *Modern Control Engineering*. Prentice-Hall, Englewood Cliffs, NJ (1990).
4. J. Canny, B. Donald, J. Reif and P. Xavier, On the complexity of kynodynamic planning, in: *Proc. Symp. on the Foundations of Computer Science*, White Plains, NY, pp. 306–316 (1988).

5. P. G. Xavier, Provably-good approximation algorithms for optimal kinodynamic robot motion plans, PhD Thesis, Cornell University (1992).
6. Th. Fraichard, Dynamic trajectory planning with dynamic constraints: a 'state-time space' approach, in: *Proc. IEEE–RSJ Int. Conf. on Intelligent Robots and Systems*, Yokohama, Vol. 2, pp. 1394–1400 (1993).
7. Th. Fraichard, Trajectory planning in a dynamic workspace: a 'state-time' approach, *Adv. Robotics* **13**, 75–94 (1999).
8. D. Hsu, R. Kindel, J.-C. Latombe and S. Rock, Randomized kinodynamic motion planning with moving obstacles, in: *Proc. Workshop on the Algorithmic Foundations of Robotics*, Hanover, NH, pp. 233–255 (2000).
9. S. LaValle and J. Kuffner, Randomized kinodynamic planning, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Detroit, MI, Vol. 1, pp. 473–479 (1999).
10. T. S. Wikman, M. S. Branicky and W. S. Newman, Reflexive collision avoidance: a generalized approach, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Atlanta, GA, Vol. 3, pp. 31–36 (1993).
11. R. Alami, T. Siméon and K. Madhava Krishna, On the influence of sensor capacities and environment dynamics onto collision-free motion plans, in: *Proc. IEEE–RSJ Int. Conf. on Intelligent Robots and Systems*, Lausanne, pp. 2395–2400 (2002).
12. A. R. Pritchett, Pilot performance at collision avoidance during closely spaced parallel approaches, *Air Traffic Control Q.* **7**, 47–75 (1999).
13. R. Teo and C. Tomlin, Computing danger zones for provably safe closely spaced parallel approaches, *J. Guid. Dyn. Control* **26**, 434–443 (2003).
14. R. Simmons, The curvature-velocity method for local obstacle avoidance, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Minneapolis, MN, pp. 3375–3382 (1996).
15. D. Fox, W. Burgard and S. Thrun, The dynamic window approach to collision avoidance, *IEEE Robotics Automat. Mag.* **4**, 23–33 (1997).
16. P. Fiorini and Z. Shiller, Motion planning in dynamic environments using velocity obstacles, *Int. J. Robotics Res.* **17**, 760–772 (1998).
17. N. Y. Ko and R. Simmons, The lane-curvature method for local obstacle avoidance, in: *Proc. IEEE–RSJ Int. Conf. on Intelligent Robots and Systems*, Victoria, BC, pp. 1615–1621 (1998).
18. O. Brock and O. Khatib, High-speed navigation using the global dynamic window approach, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Detroit, MI, pp. 341–346 (1999).
19. F. Large, S. Sekhavat, Z. Shiller and C. Laugier, Towards real-time global motion planning in a dynamic environment using the NLVO concept, in: *Proc. IEEE–RSJ Int. Conf. on Intelligent Robots and Systems*, Lausanne, pp. 607–612 (2002).
20. J.-P. Laumond, Obstacle growing in a non-polygonal world, *Information Process. Lett.* **25**, 41–50 (1987).
21. S. Skiena, in: *The Algorithm Design Manual*, pp. 395–396. Springer-Verlag, Berlin (1997).
22. Algorithmic Solutions Software GMBH, *LEDA: Library of Efficient Data types and Algorithms*. http://www.algorithmic-solutions.com/enleda.htm
23. S. Lavalle, Rapidly-exploring random trees: a new tool for path planning, *Research Report 98-11*, Department of Computer Science, Iowa State University (1998).
24. Th. Fraichard and H. Asama, Inevitable collision states. a step towards safer robots? in: *Proc. IEEE–RSJ Int. Conf. on Intelligent Robots and Systems*, Las Vegas, NV, pp. 388–393 (2003).

ABOUT THE AUTHORS

Since January 2003, **Thierry Fraichard** has been a Research Associate in the e-Motion team of Inria Rhône-Alpes and the Gravir laboratory (CNRS Mixed Research Unit 5527). From December 1994 to December 2002, he was a member of the Sharp project of Inria. He received his PhD in Computer Science from the Institut National Polytechnique de Grenoble in April 1992 for his dissertation on 'Motion planning for a nonholonomic mobile in a dynamic workspace'. From December 1993 to November 1994, he was a Postdoctoral Fellow in the Manipulation Laboratory of the Robotics Institute at Carnegie Mellon University in Pittsburgh, PA. In 1997, he took part in the organization of the IEEE–RSJ International Conference on Intelligent Robots and Systems (Secretary, Member of the Programme Committee and Local Arrangements Committee). From November 2000 to January 2001, then again in November 2001, he was a Tan Chin Tuan Fellow in the Intelligent Systems Laboratory of the Nanyang Technological University in Singapore. From September to December 2002, he was a JSPS Fellow in the Distributed Adaptive Research Unit of the Riken Institute in Tokyo. His research focuses on motion autonomy for vehicles with a special emphasis on motion planning for non-holonomic systems, motion planning in dynamic workspaces, motion planning in the presence of uncertainty and the design of control architectures for autonomous vehicles.

Hajime Asama received MS and DS degrees in Engineering from the University of Tokyo in 1984 and 1989, respectively. He worked at RIKEN (The Institute of Physical and Chemical Research, Japan) as a research associate, a research scientist, a senior research scientist and a senior scientist from 1986 to 2002, and became Professor of RACE (Research into Artifacts, Center for Engineering), the University of Tokyo in 2002. He received JSME Robotics and Mechatronics Division Robotics and Mechatronics Award in 1995, JSME Robotics and Mechatronics Division Robotics and Mechatronics Academic Achievement Award in 2000, JIDPO Good Design Award in New Frontier Design Category in 2002, among others. He was editor of the second and fifth volume of *Distributed Autonomous Robotics Systems*, which were published by Springer (Tokyo) in 1994, 1996 and 2002 respectively. He is a member of IEEE, RSJ, JSME, SICE and the New York Academy of Science, among others. His main interests are distributed autonomous robotic systems, cooperation of multiple autonomous mobile robots, emergent robotic systems, intelligent data carrier systems and service engineering.