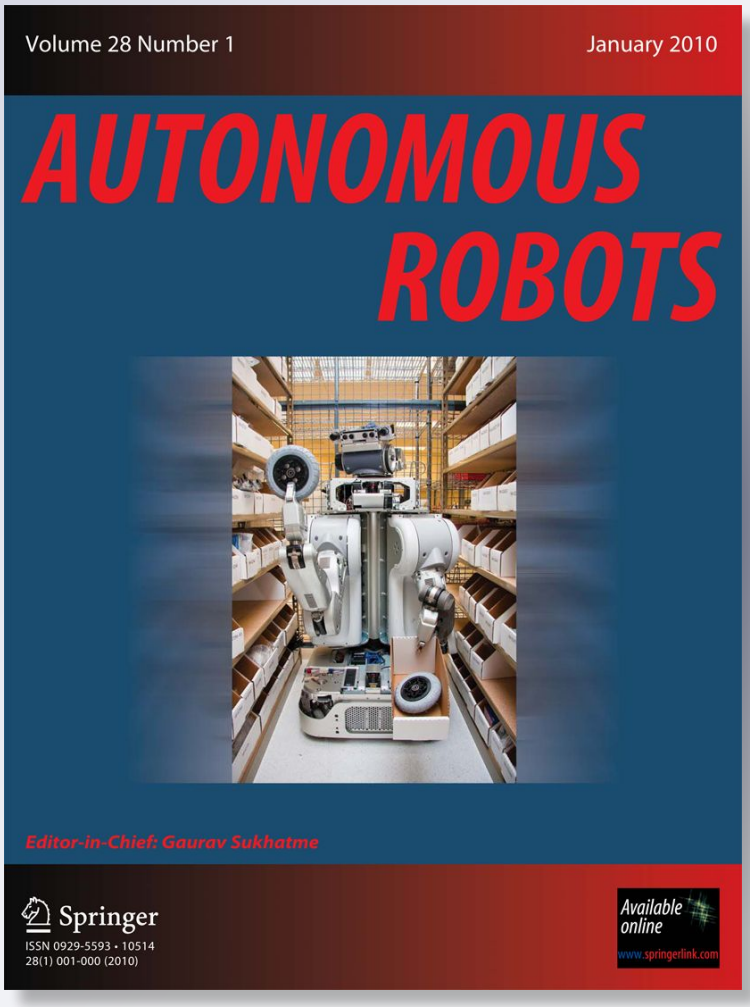
Provably safe navigation for mobile robots with limited field-of-views in dynamic environments

Sara Bouraine, Thierry Fraichard & Hassen Salhi

Autonomous Robots

ISSN 0929-5593

Auton Robot
DOI 10.1007/s10514-011-9258-8



Springer

Auton Robot
DOI 10.1007/s10514-011-9258-8

Provably safe navigation for mobile robots with limited field-of-views in dynamic environments

Sara Bouraine · Thierry Fraichard · Hassen Salhi

Received: 3 February 2011 / Accepted: 18 October 2011

Abstract This paper addresses the problem of navigating in a provably safe manner a mobile robot with a limited field-of-view placed in a unknown dynamic environment. In such a situation, absolute motion safety (in the sense that no collision will ever take place whatever happens in the environment) is impossible to guarantee in general. It is therefore settled for a weaker level of motion safety dubbed *passive motion safety*: it guarantees that, if a collision takes place, the robot will be at rest.

The primary contribution of this paper is the concept of *Braking Inevitable Collision States* (ICS), i.e. a version of the ICS corresponding to passive motion safety. Braking ICS are defined as states such that, whatever the future braking trajectory followed by the robot, a collision occurs before it is at rest. Passive motion safety is obtained by avoiding Braking ICS at all times.

It is shown that Braking ICS verify properties that allow the design of an efficient *Braking ICS-Checking algorithm*, i.e. an algorithm that determines whether a given state is a Braking ICS or not.

To validate the Braking ICS concept and demonstrate its usefulness, the Braking ICS-Checking algorithm is integrated in a reactive navigation scheme called PASSAVOID. It is formally established that PASSAVOID is *provably passively safe* in the sense that it is guaranteed that the robot will always stay away from Braking ICS no matter what happens in the environment.

Keywords Mobile robots · Dynamic environments · Autonomous navigation · Motion safety · Collision avoidance · Inevitable collision states

1 Introduction

Robotics technology has matured and Autonomous Ground Vehicles (AGVs) are becoming a reality: consider the successes of the DARPA Challenges[1] or the VisLab Intercontinental Autonomous Challenge.[2] They demonstrate robotics systems traveling significant distances at high speed in complex and realistic environments. However such systems remains prone to accidents (see Fletcher et al. 2008). While moving (especially at high speed), AGVs (and other robotic systems as well) can be potentially dangerous should a collision occur; this is a critical issue if such systems are to transport or share space with human beings.

Roboticists have long been aware of the motion safety issue; there is a rich literature on collision avoidance and collision-free navigation. Nonetheless, motion safety has for a long time remained a taken-for-granted and ill-defined notion (see Fraichard 2007). Demonstrating that a robot avoids

Electronic supplementary material The online version of this article (doi:10.1007/s10514-011-9258-8) contains supplementary material, which is available to authorized users.

S. Bouraine
CDTA, Algiers, Algeria
e-mail: s_bouraine@yahoo.fr

T. Fraichard (✉)
INRIA, CNRS-LIG and Grenoble University, Grenoble, France
e-mail: thierry.fraichard@inria.fr

H. Salhi
Blida University, Blida, Algeria
e-mail: hassensalhi@yahoo.fr

[1] www.darpa.mil/grandchallenge.

[2] www.IntercontinentalChallenge.eu.

Published online: 19 November 2011

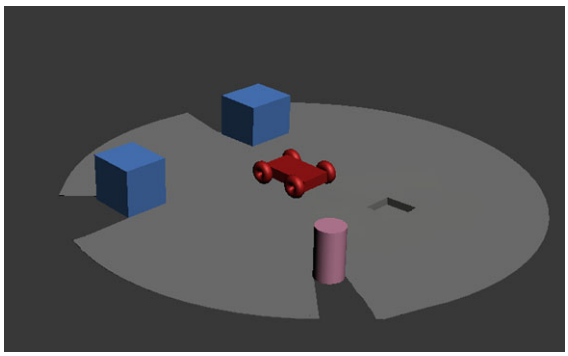


Fig. 1 Robot with a limited field-of-view in an unknown environment with fixed and moving objects (the *dark gray region* is unobserved and may contain anything)

collision on a limited set of experiments is not enough. If autonomous robots are ever to be deployed among human beings on a large scale, there is a need to design collision avoidance and navigation schemes for which motion safety can be characterized and even guaranteed. The literature review of Sect. 2 shows that the Robotics community is displaying a growing interest in designing such provably safe collision avoidance and navigation schemes. It also shows that motion safety in the real world remains an open problem as soon as the term real world implies that:

1. The environment features both fixed and moving objects whose future behavior is unknown.
2. The robot has only a partial knowledge of its surroundings because of its sensory limitations.

The purpose of this paper is precisely to address such problems, i.e. that of navigating autonomous robots with sensors having a limited field-of-view in unknown environments featuring moving objects whose future behavior is unknown (see Fig. 1). Furthermore, the primary motivation is to study whether, in such situations, it is possible to obtain strict motion safety guarantees, strict in the sense that they can be established formally.

Given that motion safety has to do with staying away from states where a collision occurs (now or eventually), the first position taken in this work is to address the motion safety issue within the formal *Inevitable Collision State* (ICS) framework developed in Fraichard and Asama (2004). An ICS is a state for which, no matter what the future trajectory of the robot is, a collision eventually occurs. ICS are defined with an *absolute motion safety* perspective (absolute in the sense that no collision will ever take place whatever happens in the environment).

In theory, absolute motion safety requires a complete knowledge of the future, up to infinity in some singular situations (see the motion safety criteria laid down in Fraichard 2007 and the discussion on motion safety of Macek et al. 2009). It can be argued then that absolute motion safety is impossible to guarantee in general unless questionable assumptions concerning the robot and its environment are made, *e.g.* requiring that the velocity of the robot is a multiple of the maximum velocity of the objects (Lumelsky and Tiwari 1994), or that the moving objects should appear beyond a distance which is a function of their number, sizes and velocities (Kohout et al. 1996).

In situations such as Fig. 1, absolute motion safety is impossible to guarantee (primarily because the lack of knowledge about the future renders ICS ineffective). To cope with that issue, the second position taken in this work is: *better guarantee less than guarantee nothing*. To that end, it is settled for a weaker level of motion safety that guarantees that, if a collision takes place, the robot will be at rest. As per Macek et al. (2009), this motion safety level is dubbed *passive motion safety*.

The primary contribution of this paper is the concept of *Braking ICS*, i.e. a version of the ICS corresponding to passive motion safety. Braking ICS are defined as states such that, whatever the future braking trajectory followed by the robot, a collision occurs before it is at rest. Passive motion safety is obtained by avoiding Braking ICS at all times.

It is shown that Braking ICS verify properties that allow the design of an efficient *Braking ICS-Checking algorithm*, i.e. an algorithm that determines whether a given state is a Braking ICS or not. This algorithm is derived from the original ICS-checking algorithm presented in Martinez-Gomez and Fraichard (2008).

To validate the Braking ICS concept and demonstrate its usefulness, the Braking ICS-Checking algorithm is integrated in a navigation scheme henceforth called PASSAVOID. It is a reactive scheme for a mobile robot with a limited field-of-view placed in an unknown dynamic environment. It operates with a given time step and its purpose is to compute the control that will be applied to the robot at the next time step. It is formally established that PASSAVOID is *provably passively safe* in the sense that it is guaranteed that the robot will always stay away from Braking ICS no matter what happens in the environment.

In itself, the central idea behind passive motion safety, i.e. using braking trajectories, is not new, it has been used before in different contexts (see Sect. 2). However, to the best of the authors' knowledge, it is the first time it is given a formal treatment in as general a context as possible whether it concerns the robot's dynamics, its field-of-view, or the knowledge (or lack thereof) about the future behavior of the moving objects. As limited as it may appear, passive motion safety is interesting for two reasons: (1) it allows to provide at least one form of motion safety guarantee in challenging scenarios such as Fig. 1, and (2) if every moving object in the environment enforces it then no collision will take place at all.

The paper is organized as follows: a review of the relevant literature is done in Sect. 2. Section 3 discusses mo-

tion safety issues and defines passive motion safety. The adaptation of the ICS concept to Braking ICS is done in Sect. 4. The Braking ICS-checking algorithm and the navigation scheme are respectively detailed in Sects. 5 and 6. Finally, experimental results obtained in simulation are presented in Sect. 7.

2 Related works

As mentioned above, the Robotics literature is teeming with works concerned with collision avoidance but most of them do not offer an explicit formulation of the safety guarantees they provide or the conditions under which they must operate (see Fraichard 2007).

The earliest relevant works addressed the so-called "Asteroid Avoidance Problem" (wherein objects traveling at a constant linear velocity must be avoided): in the 3D case, Reif and Sharir (1985) shows that collision avoidance is always possible if the robot's velocity is greater than the asteroids' velocities and if the robot is not initially in the "shadow" of an asteroid. In the 2D case, Kohout et al. (1996) shows that collision avoidance is always possible iff the asteroids appear beyond a "threat horizon", i.e. a distance which is a function of the number, size and velocity of the asteroids. Likewise, Lumelsky and Tiwari (1994) shows that, for a robot operating in a planar environment with arbitrarily moving objects, collision-free motion is guaranteed iff the maximum velocity of the robot is a multiple of the maximum velocity of the objects. Such results are very interesting. Unfortunately, they rely on assumptions that rarely occur in the real world.

A related family of research works are those seeking to coordinate the motion of a set of robots. Different distributed coordination schemes have been proposed for which collision avoidance is guaranteed (*e.g.* Pallottino et al. 2007; Van den Berg et al. 2008; Lalish and Morgansen 2008; Bekris et al. 2009). However, this guarantee is lost if the environment contains uncontrolled moving objects.

General motion safety issues have been studied thanks to the *Inevitable Collision States*[3] (ICS) concept developed in Fraichard and Asama (2004). An ICS is a state for which, no matter what the future trajectory of the robot is, a collision eventually occurs. ICS provides insight into the complexity of guaranteeing motion safety since it shows that it requires to *reason about the future* evolution of the environment and to do so with an *appropriate lookahead*[4] that can possibly be infinite. Such conditions being next to impossible to obtain in the real world plus the fact that ICS characterization is very complex has led a number of authors to consider relaxations of ICS such as:

- *ICS approximation* (*e.g.* Chan et al. 2007; Kalisiak and van de Panne 2007): such approximations being not conservative, the motion safety guarantee is lost.
- τ*-Safety* (*e.g.* Frazzoli et al. 2002; Vatcha and Xiao 2008): the robot is guaranteed to remain in states where it is safe for a given duration (hopefully sufficient to compute an updated safe trajectory...).
- *Evasive trajectories* (*e.g.* Hsu et al. 2002; Bekris and Kavraki 2010; Seder and Petrovic 2007; Macek et al. 2009): they guarantee that the robot can only be in states where it is possible to execute an evasive trajectory, *e.g.* a braking manoeuvre for a car or a circling manoeuvre for a plane.

Recently, authors have proposed probabilistic versions of the ICS concept, (*e.g.* Bautin et al. 2010; Althoff et al. 2010), so as to better capture the uncertainty that prevails in real world situations, in particular the uncertainty concerning the future behavior of the moving objects. These approaches are interesting but they offer no strict motion safety guarantees since probabilistic models are used.

There are a few research works taking into account sensory limitations. For instance, the occlusion problem, i.e. the existence of regions that are hidden by other objects, is addressed in a coarse manner in Sadou et al. (2004) and in a more principled manner in Chung et al. (2009). The occlusion and the limited field-of-view problems are addressed in Fraichard and Asama (2004) and Madhava Krishna et al. (2006). Fraichard and Asama (2004) addresses the case of a mobile robot moving in a static environment; its approach is general and ICS-based. While Madhava Krishna et al. (2006) considers dynamic environments, it does so primarily with a path-velocity decomposition perspective (Kant and Zucker 1986).

The contribution of this paper is an extension of Macek et al. (2009) that deals with limited field-of-views, occlusions and unknown future behavior of the objects. The approach proposed is based upon a relaxation of ICS that falls into the "evasive trajectories" family.

3 Safety issues

3.1 Outline of the problem

As mentioned in Sect. 1, this paper addresses the problem of navigating in a provably safe manner a mobile robot with sensors having a limited field-of-view in an unknown environment featuring fixed and moving objects with upper-bounded velocity and unknown future behavior. Let $\mathscr{A}$ denote the mobile robot at hand. It operates in a 2D workspace

[3] *Aka* Obstacle Shadow (Reif and Sharir 1985) or Region of Inevitable Collision (LaValle and Kuffner 1999).

[4] *I.e.* how far into the future the reasoning is done.

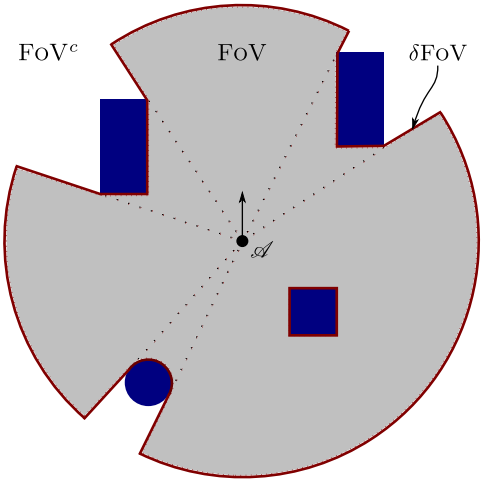


Fig. 2 Field-of-view for the scenario of Fig. 1. FoV is the gray area; ∂FoV and FoV^c have two connected components

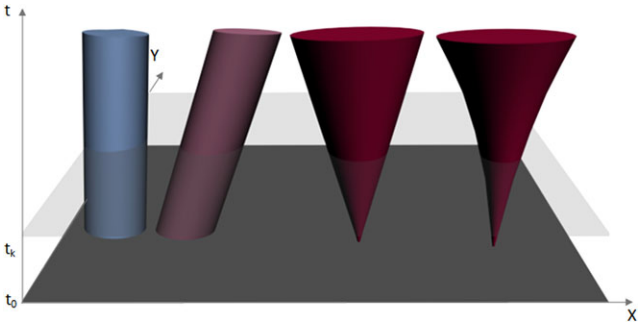


Fig. 3 Models of the future (from left to right): fixed disk (1); moving disk with constant velocity (2); conservative models for a moving point with unknown future motion and upper-bounded velocity (3), and upper-bounded acceleration and velocity (4)

$\mathcal{W}$; a state of $\mathcal{A}$ is denoted by s with $s \in \mathcal{S}$, the state space of $\mathcal{A}$. Assuming that $\mathcal{A}$ is equipped with range sensors such as laser telemeters or range cameras, it can only perceive a subset of $\mathcal{W}$; this subset is $\mathcal{A}$'s *field-of-view*; its shape is arbitrary; it depends on the current surroundings of $\mathcal{A}$ and the maximum range of its sensors. It is henceforth denoted FoV. Accordingly, $\mathcal{W}$ is partitioned in three subsets: (1) FoV, (2) FoV^c, the part which is unseen (FoV$^c = \mathcal{W} \setminus cl$(FoV)) and (3) ∂FoV, the boundary between the two. Both FoV and FoV^c are open sets. It seems reasonable to assume that $\mathcal{A}$ is "looking around itself"; in other words that $\mathcal{A}(s) \subset$ FoV where $\mathcal{A}(s)$ denotes the region of $\mathcal{W}$ occupied by $\mathcal{A}$ when it is in s. To account for the existence of 3D range sensors, *e.g.* Velodyne LIDAR or PrimeSensor range camera, FoV can contain "holes" representing objects entirely perceived by the sensory system of $\mathcal{A}$. Accordingly, FoV, ∂FoV and FoV^c are not necessarily singly connected (see Fig. 2). FoV represents the region of $\mathcal{W}$ which is free of objects at the sensing time.

This generic field-of-view model can further be enriched if the sensors of $\mathcal{A}$ can differentiate the fixed *vs* the moving objects. In that case, ∂FoV can be partitioned into three parts respectively corresponding to fixed objects, moving objects and so-called "unseen" objects, i.e. the sensing limits and the occluding lines:

$$\partial\text{FoV} = \partial\text{FoV}^f \cup \partial\text{FoV}^m \cup \partial\text{FoV}^u \quad (1)$$

When the sensors of $\mathcal{A}$ cannot differentiate between fixed and moving objects, ∂FoV $= \partial$FoV^u.

3.2 Modeling the future

The ICS concept brings to light two things: the first one is that there is more to motion safety than the simple fact that $\mathcal{A}$'s trajectory be collision-free; it must be ICS-free, i.e. $\mathcal{A}$ must always be in a state for which an evasive trajectory is available. The second one is that motion safety is always defined *wrt* the model of the future that is used. When dealing with objects whose future behavior is unknown, what model of the future should be used? The answer is to be *conservative*: one must consider all possible future motions for the object at hand. Consider the case of a point object with upper-bounded velocity whose future behavior is unknown. Given the initial position of the object, the region of the workspace that is possibly not collision-free is modeled by a disc that grows over time with a growth rate corresponding to the maximal velocity of the object (Van den Berg and Overmars 2008). In space×time, it is represented as an inverted cone (see Fig. 3). Such a cone is the *reachable set* (LaValle 2006, Chap. 14) of a point object whose dynamics is characterized by infinite acceleration and upper-bounded velocity capabilities. In general, reachable sets can be used to represent all possible future motions for object with arbitrary dynamics, *e.g.* an object with upper-bounded velocity *and* acceleration (see Fig. 3, right).

Now, in a situation such as the one depicted in Fig. 2, how does one take into account the unseen parts of $\mathcal{W}$ that belongs to ∂FoV^u or FoV^c? Walking in the footsteps of Fraichard and Asama (2004) or Madhava Krishna et al. (2006), the answer is once again to be conservative and to treat every point of ∂FoV^u or FoV^c as a potential moving object with unknown future behavior. In conclusion, the space×time model of the future for $\mathcal{A}$ can be defined as follows for the different components of $\mathcal{A}$'s field-of-view (see Fig. 4):

- ∂FoV$^u \cup$ FoV^C (the unseen objects): every point in this set is modeled as a disc that grows as time passes (i.e. a cone in space×time).
- ∂FoV^f (the fixed objects): every point in this set remains constant over time (i.e. a vertical line in space× time).
- ∂FoV^m (the moving objects): if the information about their future behavior is available and reliable, every point in this set is modeled accordingly (i.e. a curve in space×

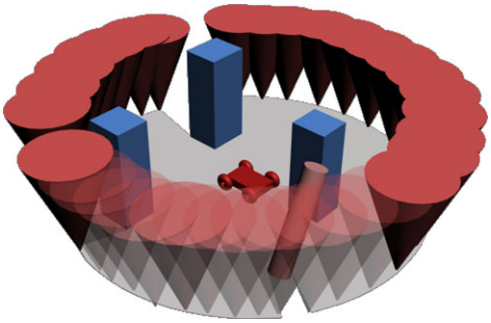


Fig. 4 Conservative model of the future (partially represented for visualization purposes) for the scenario of Fig. 1

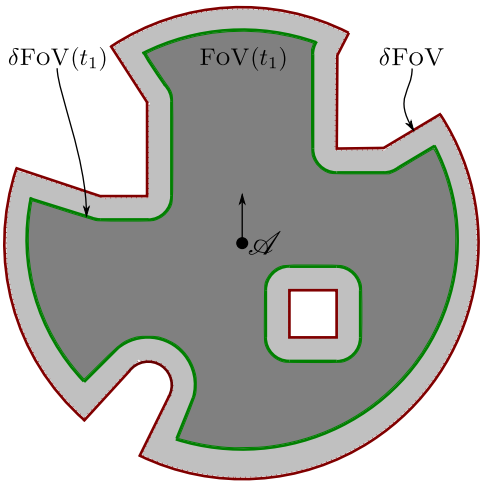


Fig. 5 Conservative model of the future: how FoV shrinks as time passes (for a time t_1 greater than the sensing time)

time), otherwise it is treated as an unseen object and modeled as a growing disc.

This of course is the case when the sensors of $\mathscr{A}$ can differentiate between fixed and moving objects. If it is not the case then every point in ∂FoV is modeled as a disc that grows as time passes (i.e. a cone in space$\times$time).

Within such a model of the future, it is worth noting that the region of $\mathscr{W}$ which is free of objects at the sensing time, i.e. FoV, gradually shrinks as time passes and eventually vanishes (see Fig. 5). Henceforth, FoV(t) denotes the region of $\mathscr{W}$ which is free of objects at time t in the conservative model of the future. Likewise, ∂FoV(t) denotes its boundary.

From now on, the worst possible scenario is considered: it is assumed that $\mathscr{A}$ cannot distinguish the fixed from the moving objects[5] and that it has no information whatsoever about their future behavior. In that case, the minimal knowledge required about the environment is an upper-bound on the objects' velocity (otherwise, it is impossible to derive a conservative and useful model of the future).

Now, if additional information is available about an object, *e.g.* whether it is fixed or moving, about its dynamics or its future behavior, it can easily be integrated into the model of the future one way (see Fig. 3). The important thing is to derive a conservative model so that the motion safety properties that will be obtained can actually be guaranteed no matter what happens in the environment.

3.3 Absolute vs. passive motion safety

The ICS concept laid down in Fraichard and Asama (2004) guarantees *absolute motion safety* in the sense that, for a state not to be an ICS, there must exist a collision-free trajectory of infinite duration. Now, an object with unknown future behavior is a challenge. If it is modeled conservatively as above then, at some point in the future, the whole workspace is entirely covered by the growing disc representing it. At that moment, the whole state space of the robot is forbidden and it becomes impossible to find a collision-free trajectory of infinite duration. This is a situation where the ICS concept becomes ineffective. In the authors' opinion, the only answer to this challenge is to settle for a weaker level of motion safety; the rationale being: *better guarantee less than guarantee nothing*. The choice here is to guarantee that, if a collision takes place, the robot will be at rest. This motion safety level, dubbed *passive motion safety* in Macek et al. (2009), seems a reasonable choice given the harsh constraints imposed by a limited field-of-view. It yields the following definition:

Definition 1 Given a model of the future workspace evolution, a *passively safe* or *p-safe* state for $\mathscr{A}$ is a state s such that there exists one braking trajectory starting at s which is collision-free until $\mathscr{A}$ has stopped.

4 From ICS to braking ICS

Using braking trajectories in order to evaluate the safety of a given state has been done before, *e.g.* Petti and Fraichard (2005), Bekris and Kavraki (2010), Seder and Petrovic (2007), Ferguson et al. (2008), Kuwata et al. (2009). The focus in this paper is to do it in the formal framework of the ICS concept. The concept of *Braking ICS* (ICS^b) is first derived from the original ICS concept. It is then used to design ICS^b-CHECK, i.e. the corresponding variant of the ICS checking algorithm proposed in Martinez-Gomez and Fraichard (2008), and to use it in the passively safe navigation scheme PASSAVOID. ICS^b-CHECK and PASSAVOID are respectively detailed in Sects. 5 and 6.

[5] Accordingly, every object that is observed is treated as a moving object.

4.1 Notations

The dynamics of the robot $\mathscr{A}$ is generally described by differential equations of the form:

$$\dot{s} = f(s,u) \quad \text{subject to} \quad g(s,\dot{s}) \leq 0 \tag{2}$$

where $s \in \mathscr{S}$ is the state of $\mathscr{A}$, $\dot{s}$ its time derivative and $u \in \mathscr{U}$ a control. $\mathscr{S}$ and $\mathscr{U}$ respectively denote the state space and the control space of $\mathscr{A}$. Let $\mathscr{A}(s)$ denote the closed subset of the workspace $\mathscr{W}$ occupied by $\mathscr{A}$ when it is in s.

Let $\tilde{u} : [0, t_f] \longrightarrow \mathscr{U}$ denote a *control trajectory*, i.e. a time-sequence of controls, t_f is the duration of $\tilde{u}$. The set of all possible control trajectories is denoted $\tilde{\mathscr{U}}$. Starting from an initial state s_0 at time 0, a *state trajectory* $\tilde{s}$, i.e. a time-sequence of states, is derived from a control trajectory $\tilde{u}$ by integrating (2); $\tilde{s}(s_0, \tilde{u}, t)$ denotes the state reached at time t.

A control trajectory $\tilde{u}_b \in \tilde{\mathscr{U}}$ such that $\tilde{s}_b(s_0, \tilde{u}_b, t_b)$ is a state where $\mathscr{A}$ comes to a halt (and remains so) is a *braking trajectory* for s_0 and t_b is its *braking time*. The set of all possible braking trajectories for s_0 is denoted $\tilde{\mathscr{U}}_b^{s_0}$.

In a situation such as the one depicted in Fig. 2, the open subset FoV is the free part of the workspace while ∂FoV^f, ∂FoV^m, ∂FoV^u and FoV^c represent objects (seen and unseen). Let $\mathscr{B}_i$ denote the space×time model of the future evolution of the corresponding object (according to the modeling rules defined in Sect. 3.2). At time 0, i.e. the sensing time, $\mathscr{B}_i(0)$ corresponds to a subset of ∂FoV^f, ∂FoV^m, ∂FoV^u or FoV^c. $\mathscr{B}_i(t)$ denotes the subset of $\mathscr{W}$ occupied by $\mathscr{B}_i$ at a particular time t in the conservative model of the future. It is assumed that each $\mathscr{B}_i(t)$ is a closed subset of $\mathscr{W}$ and that the total number of objects is n. Likewise $\mathscr{B}_i([t_1, t_2])$ denote the space×time region occupied by the object during the interval $[t_1, t_2]$. To ease notations, it is assumed that $\mathscr{B}_i \equiv \mathscr{B}_i([0, \infty))$.

4.2 Braking ICS definition

A Braking ICS (ICS^b) is informally defined as a state for which no matter what the future braking trajectory followed by $\mathscr{A}$ is, a collision occurs before $\mathscr{A}$ is at rest. Hence the following formal definition:

Definition 2 (Braking ICS) s is a ICS^b iff $\forall \tilde{u}_b \in \tilde{\mathscr{U}}_b^s, \exists t \in [0, t_b[, \tilde{s}(s, \tilde{u}_b, t)$ is a collision state at time t.

It is worth noting that when $\mathscr{A}$ is in a state s where $\mathscr{A}$ is at rest, $\tilde{\mathscr{U}}_b^s$ reduces to $\tilde{u}_b^\circ$ that denotes the braking trajectory where a null control is applied to $\mathscr{A}$. Accordingly, s is always p-safe (even if $\mathscr{A}(s)$ is in collision).

It is then possible to define the set of ICS^b yielding a collision with a particular object $\mathscr{B}_i$:

$$\text{ICS}^b(\mathscr{B}_i) = \{s \in \mathscr{S} | \forall \tilde{u}_b \in \tilde{\mathscr{U}}_b^s, \exists t \in [0, t_b[, \mathscr{A}(\tilde{s}(s, \tilde{u}_b, t)) \cap \mathscr{B}_i(t) \neq \emptyset\} \tag{3}$$

Likewise, the ICS^b set yielding a collision with $\mathscr{B}_i$ for a given trajectory $\tilde{u}_b$ is defined as:

$$\text{ICS}^b(\mathscr{B}_i, \tilde{u}_b) = \{s \in \mathscr{S} | \exists t \in [0, t_b[, \mathscr{A}(\tilde{s}(s, \tilde{u}_b, t)) \cap \mathscr{B}_i(t) \neq \emptyset\} \tag{4}$$

4.3 Braking ICS properties

The first two ICS^b properties that can be shown are the equivalent of two key ICS properties established in Fraichard and Asama (2004) and seminal in the design of an ICS checking algorithm. Let $\mathscr{B} = \bigcup_1^n \mathscr{B}_i$. The first property shows that $\text{ICS}^b(\mathscr{B})$ can be derived from $\text{ICS}^b(\mathscr{B}_i, \tilde{u}_b)$ for every object $\mathscr{B}_i$ and every possible braking trajectory $\tilde{u}_b$.

Property 1 (ICS^b characterization)

$$\text{ICS}^b(\mathscr{B}) = \bigcap_{\tilde{u}_b \in \tilde{\mathscr{U}}_b} \bigcup_{i=1}^{n} \text{ICS}^b(\mathscr{B}_i, \tilde{u}_b)$$

Proof The proof of Property 1 is done in two stages: it is first established that:

$$s \in \text{ICS}^b(\mathscr{B}) \quad \Leftrightarrow \quad s \in \bigcap_{\tilde{u}_b \in \tilde{\mathscr{U}}_b^s} \text{ICS}^b(\mathscr{B}, \tilde{u}_b)$$

and then that:

$$s \in \text{ICS}^b\left(\bigcup_{i=1}^{n} \mathscr{B}_i, \tilde{u}_b\right) \quad \Leftrightarrow \quad s \in \bigcup_{i=1}^{n} \text{ICS}^b(\mathscr{B}_i, \tilde{u}_b)$$

These two properties are demonstrated in a straightforward manner as in Fraichard and Asama (2004). Combining the two properties above yields Property 1. □

The next property permits to compute a conservative approximation of $\text{ICS}^b(\mathscr{B})$ by using a subset only of the whole set of possible braking trajectories.

Property 2 (ICS^b approximation)

$$\text{ICS}^b(\mathscr{B}) \subseteq \text{ICS}^b(\mathscr{B}, \mathscr{E})$$

with $\mathscr{E} \subseteq \tilde{\mathscr{U}}_b$, a subset of the whole set of possible braking trajectories.

Proof

$$\begin{aligned} \text{ICS}^b(\mathscr{B}) &= \bigcap_{\tilde{u}_b \in \mathscr{E}} \text{ICS}^b(\mathscr{B}, \tilde{u}_b) \cap \bigcap_{\tilde{u}_b \in \tilde{\mathscr{U}}_b \setminus \mathscr{E}} \text{ICS}^b(\mathscr{B}, \tilde{u}_b) \\ &\subseteq \bigcap_{\tilde{u}_b \in \mathscr{E}} \text{ICS}^b(\mathscr{B}, \tilde{u}_b) \end{aligned}$$

□

One distinctive feature of the ICS concept is that trajectories of infinite duration are checked for collision, i.e. it

has an infinite lookahead (it is this infinite lookahead that guarantees safety). While the ICS^b concept also considers trajectories $\tilde{u}_b$ of infinite duration, collision checking is limited to the time interval $[0, t_b[$ where t_b is the braking time of $\tilde{u}_b$. For an arbitrary subset $\mathscr{E}$ of the whole set of possible braking trajectories, a finite lookahead T_h exists:

$$T_h = \max_{\tilde{u}_b \in \mathscr{E}} \{t_b\} \tag{5}$$

T_h is a valid lookahead in the sense that, in order to compute ICS$^b(\mathscr{B}, \mathscr{E})$, it suffices to consider the model of the future up to time T_h. This is established by the following property:

Property 3 (ICS^b lookahead)

$$\text{ICS}^b(\mathscr{B}, \mathscr{E}) = \text{ICS}^b(\mathscr{B}([0, T_h[), \mathscr{E})$$

Proof Property 3 stems from the very definition of a Braking ICS which, for a given braking manoeuvre $\tilde{u}_b$, is only concerned with collisions taking place before $t_b < T_h$. □

Finally, recall from Sect. 3.1 that, for the case of a robot with a limited field-of-view, $\mathscr{B}$ comprises ∂FoV and FoV^c, i.e. the unseen part of $\mathscr{W}$. From a motion safety perspective, the next property is very important since it establishes that FoV^c can be ignored in the computation of ICS$^b(\mathscr{B})$. In other words, considering ∂FoV suffices to guarantee motion safety.

Property 4 (Field-of-view boundary)

$$\text{ICS}^b(\mathscr{B}) = \text{ICS}^b(\partial\text{FoV} \cup \text{FoV}^c) = \text{ICS}^b(\partial\text{FoV})$$

Proof The equality between ICS$^b(\mathscr{B})$ and ICS$^b(\partial$FoV$)$ is done is two stages. Let s denote a collision-free state whose corresponding position is located inside FoV and such that $s \in \text{ICS}^b(\partial\text{FoV})$. As per Definition 2, it stems that:

$$\forall \mathscr{B}_i,\ \forall \mathscr{B}_j, \quad \text{ICS}^b(\mathscr{B}_i) \subseteq \text{ICS}^b(\mathscr{B}_i \cup \mathscr{B}_j)$$

Accordingly:

$$s \in \text{ICS}^b(\partial\text{FoV}) \quad \Rightarrow \quad s \in \text{ICS}^b(\partial\text{FoV} \cup \text{FoV}^c).$$

It is assumed now that $s \in \text{ICS}^b(\text{FoV}^c)$, it means that $\forall \tilde{u}_b \in \tilde{\mathscr{U}}_b^s, \exists t \in [0, t_b[$ such that $\tilde{s}(s, \tilde{u}_b, t)$ is in collision with a point of FoV$^c(t)$. Since s is located inside FoV, it takes a simple topological argument to realize that $\exists t' < t$ such that $\tilde{s}(s, \tilde{u}_b, t')$ is in collision with a point of ∂FoV(t'). Accordingly $s \in \text{ICS}^b(\partial\text{FoV})$ and the following holds:

$$s \in \text{ICS}^b(\partial\text{FoV} \cup \text{FoV}^c) \quad \Rightarrow \quad s \in \text{ICS}^b(\partial\text{FoV}).$$

□

In other words, it suffices to consider ∂FoV in order to compute ICS$^b(\mathscr{B})$.

5 Braking ICS checking

PassAvoid primarily relies upon ICS^b-Check, an algorithm that checks whether a given state is a Braking ICS or not. ICS^b-Check is the passively safe version of the ICS checking algorithm (called ICS-Check) presented in Martinez-Gomez and Fraichard (2008). The passively safe version of ICS-Check can be designed because Properties 1 and 2 are verified for Braking ICS. The steps involved in checking whether a given state s_c is a ICS^b are given in Algorithm 1. Besides the state to be checked, the algorithm takes as input the model of the environment and the conservative space$\times$ model of the future (see Sect. 4.1). Steps 2, 3 and 4 are the direct translation of Property 1.

Algorithm 1: General ICS^b checking algorithm.

Input: s_c, the state to be checked; $\mathscr{B}_i, i = 1 \ldots n$.
Output: Boolean value.

1 Select $\mathscr{E} \subset \tilde{\mathscr{U}}_b^{s_c}$, a set of braking trajectories for s_c;
2 Compute ICS$^b(\mathscr{B}_i, \tilde{u}_b)$ for every $\mathscr{B}_i$ and every $\tilde{u}_b \in \mathscr{E}$;
3 Compute ICS$^b(\mathscr{B}, \tilde{u}_b) = \bigcup_{i=1}^n$ ICS$^b(\mathscr{B}_i, \tilde{u}_b)$ for every $\tilde{u}_b \in \mathscr{E}$;
4 Compute ICS$^b(\mathscr{B}, \mathscr{E}) = \bigcap_{\tilde{u}_b \in \mathscr{E}}$ ICS$^b(\mathscr{B}, \tilde{u}_b)$;
5 **if** $s_c \in$ ICS$^b(\mathscr{B}, \mathscr{E})$ **then**
6 | **return** True;
7 **else**
8 | **return** False;
9 **end**

As in Martinez-Gomez and Fraichard (2008), when $\mathscr{A}$ is planar, it becomes possible to design ICS^b-Check, i.e. an efficient version of Algorithm 1. In that case, a state s of $\mathscr{A}$ can be rewritten $s = (x, y, \hat{\mathbf{z}})$ with (x, y) the workspace coordinates of $\mathscr{A}$'s reference point, and $\hat{\mathbf{z}}$ the rest of the state parameters. The primary design principle behind ICS^b-Check is to compute the ICS^b set corresponding to a 2D slice of the state space $\mathscr{S}$ of $\mathscr{A}$ (instead of attempting to perform computation in the fully-dimensional state space), and then to check if s_c belongs to this set. Assuming the state to be checked is $s_c = (x_c, y_c, \hat{\mathbf{z}}_\mathbf{c})$, the 2D slice considered is the $\hat{\mathbf{z}}_\mathbf{c}$-slice and it is possible to define the ICS^b set of the $\hat{\mathbf{z}}_\mathbf{c}$-slice considered that yields a collision with $\mathscr{B}_i$ at a particular time $t \in [0, t_b[$ for the braking trajectory $\tilde{u}_b$:

$$\begin{aligned}&\text{ICS}^b{}_{\hat{\mathbf{z}}_\mathbf{c}}(\mathscr{B}_i, \tilde{u}_b, t)\\ &= \{s \in \hat{\mathbf{z}}_\mathbf{c}\text{-slice} | \mathscr{A}(\tilde{s}(s, \tilde{u}_b, t)) \cap \mathscr{B}_i(t) \neq \emptyset\}\end{aligned} \tag{6}$$

Likewise:

$$\text{ICS}^b{}_{\hat{\mathbf{z}}_\mathbf{c}}(\mathscr{B}_i, \tilde{u}_b) = \bigcup_{t \in [0, t_b[} \text{ICS}^b{}_{\hat{\mathbf{z}}_\mathbf{c}}(\mathscr{B}_i, \tilde{u}_b, t) \tag{7}$$

Applying this 2D reasoning principle, ICS^b-CHECK is similar to the general ICS^b Checking Algorithm detailed in Algorithm 1 except that, at all steps of the algorithm, ICS$^b{}_{\hat{\mathbf{z}}_\mathbf{c}}$ is computed instead of ICS^b (see Algorithm 2). It is by keeping all computations in 2D (notwithstanding the actual dimensionality of $\mathscr{S}$) that it is possible to efficiently compute the ICS^b set corresponding to a given $\hat{\mathbf{z}}_\mathbf{c}$-slice.

Algorithm 2: ICS^b-CHECK.

Input: s_c, the state to be checked; $\mathscr{B}_i, i = 1 \ldots n$.
Output: Boolean value.

```
Select 𝒞 ⊂ 𝒰̃_b^{s_c}, a set of braking trajectories for s_c;
forall the ũ_b ∈ 𝒞 do
    forall the ℬ_i do
        Compute ICS^b_ẑc(ℬ_i, ũ_b);
    endfall
    Compute ICS^b_ẑc(ℬ, ũ_b) =
    ⋃_{i=1}^n ICS^b_ẑc(ℬ_i, ũ_b);
endfall
Compute ICS^b_ẑc(ℬ, 𝒞) = ⋂_{ũ_b∈𝒞} ICS^b_ẑc(ℬ, ũ_b);
if s_c ∈ ICS^b(ℬ, 𝒞) then
    return TRUE;
else
    return FALSE;
end
```

For the sake of brevity and because of the similarity between ICS^b-CHECK and ICS-CHECK, the inner workings of ICS^b-CHECK are not detailed here. The reader is referred to Martinez-Gomez and Fraichard (2008) instead. Suffice to say that ICS^b-CHECK provides an efficient way to check whether a given state is a ICS^b or not.

6 Passively safe navigation

In order to demonstrate passive motion safety and to validate the Braking ICS concept, a navigation scheme (henceforth called PASSAVOID) has been developed for a mobile robot $\mathscr{A}$ with a limited field-of-view placed in a unknown dynamic environment. PASSAVOID's primary task is to keep $\mathscr{A}$ in p-safe states, or equivalently, to drive $\mathscr{A}$ away from Braking ICS. PASSAVOID *guarantees passive motion safety* no matter what happens in the environment. In other words, if a collision takes place, it is guaranteed that $\mathscr{A}$ will be at rest when it occurs. PASSAVOID relies upon ICS^b-CHECK to operate.

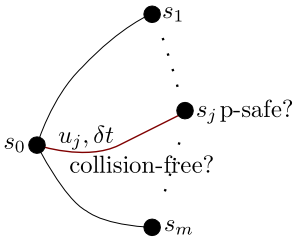


Fig. 6 PASSAVOID's operating principle (see text)

6.1 PASSAVOID's principle

PASSAVOID is a reactive navigation scheme that operates with a given time step δt. At each time step, its purpose is to compute the constant control u that will be applied to $\mathscr{A}$ during the next time step; u must be *admissible*, i.e. the corresponding state trajectory must be p-safe (in other words, it must be ICS^b-free).

PASSAVOID operates like most standard reactive collision avoidance schemes, *e.g.* (Fox et al. 1997; Fiorini and Shiller 1998). In all cases, their operating principle is to first characterize forbidden regions in a given control space and then select an admissible control, i.e. one which is not forbidden. Accordingly collision avoidance also depends on the ability of the collision avoidance scheme at hand to find such an admissible control. In the absence of a formal characterization of the forbidden regions, all schemes resort to some form of sampling of the control space with the inherent risk of missing the admissible regions. PASSAVOID also resorts to sampling in order to find an admissible control. However, in contrast with standard collision avoidance schemes, PASSAVOID is designed in such a way that it is guaranteed that an admissible control always exists and that it will be part of the sampling set.

The operating principle of PASSAVOID is illustrated in Fig. 6. Let s_0 denote the current state of $\mathscr{A}$ and U a sampled set of controls: $U = \{u_1 \ldots u_m\}$. A given control $u_j \in U$ is applied to $\mathscr{A}$ for a duration δt. It takes $\mathscr{A}$ from the state s_0 to the state $s_j = \tilde{s}(s_0, u_j, \delta t)$. If the state trajectory between s_0 and s_j is p-safe then u_j is admissible. Using the Sufficient Safety Condition established in Petti and Fraichard (2005), the admissibility of u_j can equivalently be verified by checking that (1) the state trajectory between s_0 and s_j is collision-free (with respect to the conservative model of the future $\mathscr{B}_i, i = 1 \ldots n$), and that (2) s_j is p-safe, i.e. it is not a Braking ICS. This procedure is applied for every control in U; it yields a set of admissible controls denoted U^* from which PASSAVOID can pick the control to apply during the next time step. This selection can be made arbitrarily if one is only concerned with the survival of $\mathscr{A}$ or it can be made so as to ensure convergence towards a given goal (using for instance a global navigation function, a potential field, or even a partial motion planning scheme).

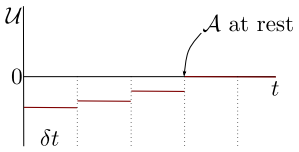


Fig. 7 Example of a δ-braking trajectory (1D case)

6.2 Passive motion safety guarantee

A scheme such as PASSAVOID works well as long as an admissible control can be found in U. But if, at the end of the day, U^* is empty, it means that every control in U takes $\mathscr{A}$ to a Braking ICS. In other words, passive motion safety will not be achieved and a collision will take place while $\mathscr{A}$ is still moving. To address this issue, it is necessary to guarantee that $U = \{u_1 \dots u_m\}$ is never empty and always contains at least one admissible control. It is possible to achieve this by carefully designing PASSAVOID. To that end, a number of definitions and properties are required. They are introduced now. The concepts of δ-braking trajectory and δ-passive safety are defined first.

Definition 3 (δ-Braking trajectory) A braking trajectory $\tilde{u}^* \in \tilde{\mathscr{U}}_b^{s_0}$ of duration t^* is a *δ-braking trajectory* if it is constant over intervals of fixed duration δt.

A δ-braking trajectory is just a special type of braking trajectory (see Fig. 7). It yields a corresponding type of passive motion safety:

Definition 4 (δ-Passive safety) A state s_0 is *δ-passively safe* or *δ-p-safe* if it exists one δ-braking trajectory $\tilde{u}^*$ starting at s_0 which is collision-free until $\mathscr{A}$ has stopped.

Then two useful properties are established:

Property 5 (P-Safe states) *If the state s_0 is p-safe and the braking trajectory $\tilde{u}_b \in \tilde{\mathscr{U}}_b^{s_0}$ starting at s_0 is collision-free until $\mathscr{A}$ has stopped then every state $\tilde{s}(s_0, \tilde{u}_b, t), 0 < t \leq t_b$ is also p-safe.*

Proof Suppose that $\exists t_i \in]0, t_b]$ such that $\tilde{s}(s_0, \tilde{u}_b, t_i)$ is not p-safe then, by definition, $\forall \tilde{u}_j \in \tilde{\mathscr{U}}_b^{s_i}$, $\tilde{u}_j$ yields a collision before $\mathscr{A}$ stops. This also applies to the braking trajectory corresponding to the restriction of $\tilde{u}_b$ to the time interval $[t_i, t_b]$ which yields a contradiction. □ □

Note that Property 5 also applies to δ-p-safe states and δ-passive safety.

Property 6 (δ-Passive safety guarantee) *If the state s_0 is δ-p-safe then there exists at least one admissible control u^* that can be used to drive $\mathscr{A}$ to a state which is also δ-p-safe.*

Proof Since s_0 is δ-p-safe, there exists at least a one δ-braking trajectory $\tilde{u}^*$ starting at s_0 which is collision-free until $\mathscr{A}$ has stopped. As per Property 5, the state $\tilde{s}(s_0, \tilde{u}^*, \delta t)$ is δ-p-safe. Let u^* denote the value of $\tilde{u}^*$ over the time interval $[0, \delta t[$, u^* is an admissible control. □

Algorithm 3: PASSAVOID.

Input: s_0, the current δ-p-safe state of $\mathscr{A}$;
$\mathscr{B}_i, i = 1 \dots n$; δt, the time step.
Output: u

1 Sample $\mathscr{U} \rightsquigarrow U = \{u_1 \dots u_m\}$; `// Select the control space sampling set U`
2 $U^* = K(s_0)$; `// Initialize admissible control set`
3 **forall the** $u_j \in U$; `// Compute admissible controls`
4 **do**
5 $\quad s(\delta t) = \tilde{s}(s_0, u_j, \delta t)$;
6 $\quad$ **if** $\tilde{s}(s_0, u_j, [0, \delta t[)$ *is collision-free and* $s(\delta t)$ *is* δ*-p-safe* **then**
7 $\quad\quad U^* = U^* \cup \{u_j\}$; `// u_j admissible`
8 $\quad$ **end**
9 **endfall**
`// Select and return one admissible control`
10 Select $u \in U^*$;
11 **return** u;

Property 6 is fundamental for the design of a version of PASSAVOID whose passive motion safety can be guaranteed. PASSAVOID simply has to drive $\mathscr{A}$ from one δ-p-safe state to the next. Now, assuming that s_0 is δ-p-safe, Property 6 guarantees the existence of at least one admissible control u^* which, if applied to $\mathscr{A}$ for the duration δt, will take it to another δ-p-safe state. In general, a δ-p-safe state s has more than one admissible control. Let $K(s)$ denote this set of admissible controls, it is dubbed the *kernel* of $K(s)$. Now, in order to guarantee its passive motion safety, PASSAVOID must include $K(s_0)$ in its control space sampling set. This is precisely what PASSAVOID does (see Algorithm 3, line #2).

PASSAVOID features two important steps: computing the kernel $K(s_0)$ (line #2) and checking whether the state $s(\delta t)$ is δ-p-safe (line #6). It turns out that these two procedures are related and can be done by an adaptation of ICS^b-CHECK which is detailed now: given that, by definition, a state which is not p-safe is a Braking ICS, ICS^b-CHECK is used to check whether a given state is

Algorithm 4: ICS^b-CHECK + kernel computation.

Input: s_c, the state to be checked; $\mathscr{B}_i, i = 1 \ldots n$.
Output: Boolean value; $K(s_c)$, the kernel of s_c.

1 Select $\mathscr{E} \subset \tilde{\mathscr{U}}_b^{s_c}$, a set of δ-braking trajectories for s_c;
2 $K(s_c) = \emptyset$;
3 **forall the** $\tilde{u}^* \in \mathscr{E}$ **do**
4 **forall the** $\mathscr{B}_i$ **do**
5 Compute $\mathrm{ICS}^b{}_{\hat{\mathbf{z}}_\mathbf{c}}(\mathscr{B}_i, \tilde{u}^*)$;
6 **endfall**
7 Compute $\mathrm{ICS}^b{}_{\hat{\mathbf{z}}_\mathbf{c}}(\mathscr{B}, \tilde{u}^*) = \bigcup_{i=1}^n \mathrm{ICS}^b{}_{\hat{\mathbf{z}}_\mathbf{c}}(\mathscr{B}_i, \tilde{u}^*)$;
8 **if** $s_c \notin \mathrm{ICS}^b{}_{\hat{\mathbf{z}}_\mathbf{c}}(\mathscr{B}, \tilde{u}^*)$ **then**
9 $K(s_c) = K(s_c) \cup u^*$; `// s_c is δ-p-safe for ũ*`
10 **end**
11 **endfall**
12 Compute $\mathrm{ICS}^b{}_{\hat{\mathbf{z}}_\mathbf{c}}(\mathscr{B}, \mathscr{E}) = \bigcap_{\tilde{u}_b \in \mathscr{E}} \mathrm{ICS}^b{}_{\hat{\mathbf{z}}_\mathbf{c}}(\mathscr{B}, \tilde{u}_b)$;
13 **if** $s_c \in \mathrm{ICS}^b(\mathscr{B}, \mathscr{E})$ **then**
14 **return** {TRUE, $K(s_c)$}; `// K(s_c) = ∅ in this case`
15 **else**
16 **return** {FALSE, $K(s_c)$};
17 **end**

p-safe or not. To that end, ICS^b-CHECK relies upon a selected set of braking trajectories. In a similar manner, ICS^b-CHECK can be used to check whether a given state is δ-p-safe or not by considering δ-braking trajectories instead (line #1 of Algorithm 1).

Besides, when ICS^b-CHECK computes $\mathrm{ICS}^b(\mathscr{B}, \tilde{u}_b)$ for a given braking trajectory $\tilde{u}_b$ (line # 3 of Algorithm 1), it is straightforward to determine if $\tilde{u}_b$ is collision-free when starting from s_c: it is the case if $s_c \notin \mathrm{ICS}^b(\mathscr{B}, \tilde{u}_b)$. In that case, $\tilde{u}_b$ is a candidate for $K(s_c)$, the kernel of s_c. Algorithm 4 is a version of ICS^b modified so as (1) to check if its input state s_c is δ-p-safe or not (line #1), and (2) to compute the kernel of s_c (line #9). It is this version of ICS^b-CHECK which is used inside PASSAVOID.

Provided that the initial state of the system $\mathscr{A}$ is δ-p-safe, Property 6 allows PASSAVOID to have at its disposal at each time step an admissible control that can be used to drive $\mathscr{A}$ from one δ-p-safe state to the next (forever if need be). Concerning the assumption on the initial state being δ-p-safe, it is satisfied when $\mathscr{A}$ is at rest (see Sect. 4.2), and the null control is admissible. In other words, starting with $\mathscr{A}$ at rest, PASSAVOID has an admissible control readily available that can be used right away if the situation demands it (this is true even if δt is very small).

At the end of the day, PASSAVOID is *provably passively safe* in the sense that it is guaranteed that the $\mathscr{A}$ will always stay away from Braking ICS no matter what happens in the environment.

6.3 Passively safe multi-robot navigation

In the introduction, it was stated that, if every moving object in a given environment was passively safe, i.e. stayed away from Braking ICS, then no collision should take place at all. It turns out that this property is straightforward to demonstrate.

Let $\mathscr{A}_1$ and $\mathscr{A}_2$ denote two robots that are driven by a provably passively safe navigation scheme such as PASSAVOID. As per Properties 5 and 6, both $\mathscr{A}_1$ and $\mathscr{A}_2$ are in a δ-p-safe state at all times. In other words, the following holds:

$$\forall t, \qquad s_1(t) \notin \mathrm{ICS}_1^b \quad \text{and} \quad s_2(t) \notin \mathrm{ICS}_2^b \tag{8}$$

where $s_i(t)$ and ICS_i^b respectively denote the state at time t and the corresponding Braking ICS set for robot $\mathscr{A}_i$, $i = 1, 2$.

Assuming that a collision can take place between $\mathscr{A}_1$ and $\mathscr{A}_2$ with one of them having a non zero velocity yields a contradiction. It cannot happen.

7 Simulation results

To validate the Braking ICS concept and demonstrate its usefulness, ICS^b-CHECK and PASSAVOID have both been implemented and tested in simulation on scenarios similar to that of Fig. 1.

7.1 Model of the robot

The model of $\mathscr{A}$ is that of a standard car-like vehicle with two fixed rear wheels and two orientable front wheels. A state of $\mathscr{A}$ is a 5-tuple $s = (x, y, \theta, v, \xi)$ with (x, y) the coordinates of the rear axle midpoint, θ the orientation of $\mathscr{A}$, v the linear velocity of system, and ξ the orientation of the front wheels (steering angle). A control of $\mathscr{A}$ is a couple $u = (u_\alpha, u_\xi)$ with u_α the linear acceleration of the rear wheels and u_ξ the steering angle velocity. Let L denote the wheelbase of $\mathscr{A}$. The motion of $\mathscr{A}$ is governed by the following differential equations:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v} \\ \dot{\xi} \end{bmatrix} = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ v\tan\xi/L \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} u_\alpha + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u_\xi \tag{9}$$

with $|v| \le v_{\max}$, $|\xi| \le \xi_{\max}$, $|u_\alpha| \le u_{\alpha_{\max}}$ and $|u_\xi| \le u_{\xi_{\max}}$.

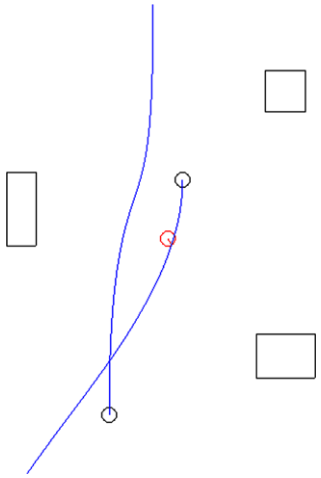


Fig. 8 Test scenario: three fixed rectangles and two moving discs (with their future trajectories). The robot $\mathscr{A}$ is the disc at the center

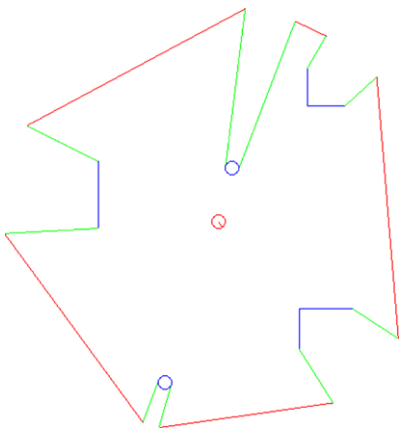


Fig. 9 Field-of-view FoV and its boundary ∂FoV for the scenario of Fig. 8

7.2 Workspace and field-of-view

A typical test scenario is depicted in Fig. 8. The planar workspace $\mathscr{W}$ contains polygonal objects and disks that can be fixed or moving with a given maximum velocity. Assuming that $\mathscr{A}$ is equipped with an omnidirectional laser range finder mounted at the center of $\mathscr{A}$, the field-of-view of $\mathscr{A}$ is depicted in Fig. 9. The circular arc corresponding to the maximum range of the range finder have been replaced by straight segments; this conservative simplification could easily be lifted.

7.3 ICS^b-Check at work

To briefly illustrate how ICS^b-Check works, the scenario in Fig. 8 is used. In contrast with the worst case assumption made in Sect. 3.2, it is assumed here that the sensors can differentiate between the fixed and the moving objects and that the future motion of the observed moving objects

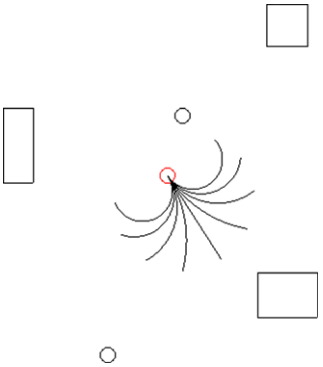


Fig. 10 $\mathscr{E}$, the set of braking trajectories considered by ICS^b-Check

is available.[6] It means that ∂FoV can actually be partitioned into three parts ∂FoV^f, ∂FoV^m and ∂FoV^u respectively corresponding to fixed objects, moving objects and unseen objects, i.e. the sensing limits and the occluding lines (see Sect. 3.1). The model of the future used for ∂FoV^f and ∂FoV^m is set according to the principles laid down in Sect. 3.2.

ICS^b-Check is called to determine whether the current state of $\mathscr{A}$ is a Braking ICS or not. This state is $s_c = (0, 0, -1, 20, 0)$. As per Algorithm 2, ICS^b-Check computes the ICS^b set for the corresponding $\hat{\mathbf{z}}_\mathbf{c}$-slice with $\hat{\mathbf{z}}_\mathbf{c} = (-1, 20, 0)$.

A set $\mathscr{E}$ of braking trajectories must be selected. They can be chosen arbitrarily since they always yield a conservative approximation of the ICS^b set (as per Property 2). In this case, $\mathscr{E}$ comprised nine braking trajectories defined by a constant minimum linear deceleration $u_\alpha = -u_{\alpha_{\max}}$ and a constant steering angle velocity $|u_\xi| \leq u_{\xi_{\max}}$. These braking trajectories are depicted in Fig. 10.

For each braking trajectory $\tilde{u}_b \in \mathscr{E}$, the set $\text{ICS}^b{}_{\hat{\mathbf{z}}_\mathbf{c}}(\mathscr{B}_i, \tilde{u}_b)$ is computed. Exploiting graphics rendering techniques, $\text{ICS}^b{}_{\hat{\mathbf{z}}_\mathbf{c}}(\mathscr{B}_i, \tilde{u}_b)$ yields a region of a given color on the OpenGL buffer representing the $\hat{\mathbf{z}}_\mathbf{c}$-slice. Figure 11 depicts the regions corresponding to four different braking trajectories. For a given braking trajectory $\tilde{u}_b$, a state corresponding to a pixel included in the colored region of the $\hat{\mathbf{z}}_\mathbf{c}$-slice is a Braking ICS for $\tilde{u}_b$. All the steps of ICS^b-Check that involves computing unions and intersections of arbitrary shapes are performed very efficiently on this OpenGL buffer by taking advantage of the Red-Green-Blue color coding and the bitwise logical operators available; for additional details, the reader is referred to Martinez-Gomez and Fraichard (2008). The final output of this process is illustrated in Figs. 12 and 13 where it appears that $s_c = (0, 0, -1, 20, 0)$ is not a ICS^b: the color of the $(0, 0)$ pixel in the $\hat{\mathbf{z}}_\mathbf{c}$-slice is not black.

[6] Through a priori knowledge or communication for instance.

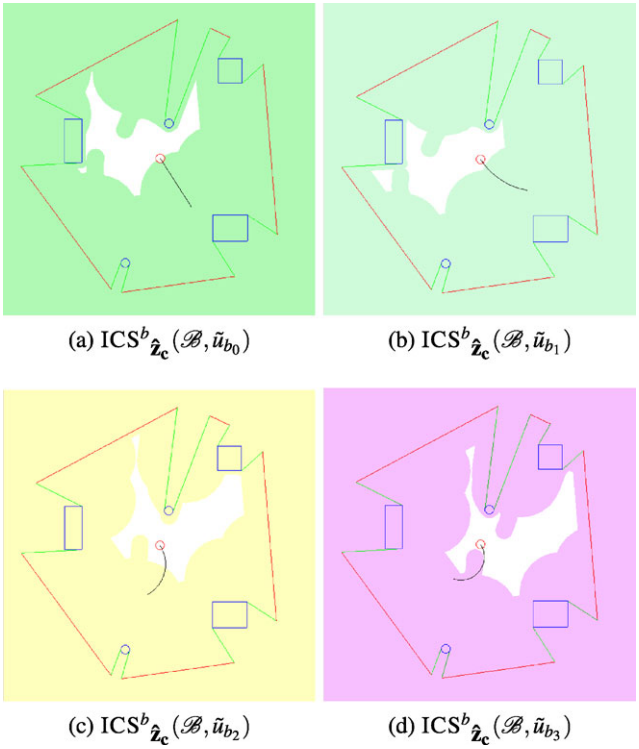


Fig. 11 $\mathrm{ICS}^b_{\hat{\mathbf{z}}_\mathbf{c}}(\mathscr{B}, \tilde{u}_b)$ for different braking trajectories

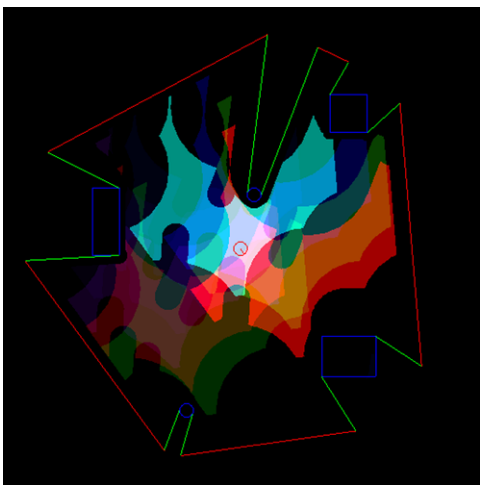


Fig. 12 Two dimensional $\hat{\mathbf{z}}_\mathbf{c}$-slice of the 5D state space of $\mathscr{A}$: a region of a given color indicates that it is a ICS^b for the corresponding braking trajectory. *Black regions* are ICS^b

7.4 PASSAVOID at work

To illustrate how PASSAVOID works, two scenarios have been selected. The first one is called the *1D Compactor scenario*, it is simple but it helps to understand the kind of behavior that PASSAVOID will yield when $\mathscr{A}$ is confronted to a clearly identified dangerous situation. The second one is called the *Blind Crowd scenario*; its primary purpose is to illustrate the performances of PASSAVOID in complex situations. These two scenarios are presented in the next sections. The results obtained are also illustrated in a short film provided as a multimedia attachment to this paper[7].

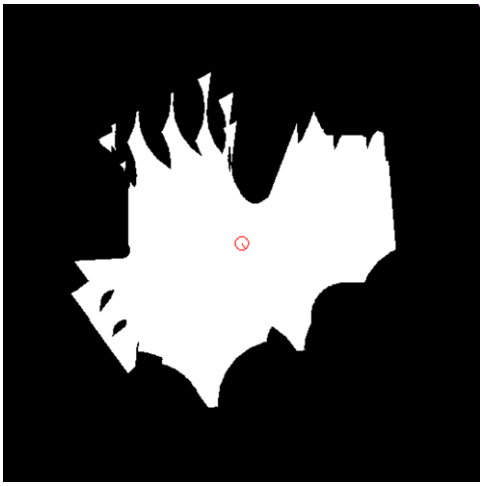


Fig. 13 Black and White version of Fig. 12: *white regions* correspond to p-safe states

In both cases, PASSAVOID had no information regarding the future trajectories of the moving objects. PASSAVOID did not attempt to drive $\mathscr{A}$ to a given goal. Its primary purpose was to keep $\mathscr{A}$ in p-safe states. Its secondary purpose was to keep $\mathscr{A}$ moving. In other words, the admissible control selection (line #10 of Algorithm 3) was biased towards controls yielding a non-zero linear velocity. This choice was made so as to avoid the straightforward answer to the passive motion safety problem which is simply to brake down and stop forever (by doing so, $\mathscr{A}$ reaches and stays in a p-safe state).

7.4.1 1D compactor scenario

The 1D Compactor scenario features one fixed object $\mathscr{B}_f$ and one moving object $\mathscr{B}_m$. The moving object is moving towards the fixed object (see Fig. 14). $\mathscr{B}_f$ and $\mathscr{B}_m$ are like the two jaws of a compactor (hence the name of the scenario). $\mathscr{A}$ is placed between $\mathscr{B}_f$ and $\mathscr{B}_m$ and it is further assumed that $\mathscr{A}$ can only move along the vertical line connecting $\mathscr{B}_f$ and $\mathscr{B}_m$. At the beginning, $\mathscr{A}$ is moving upward with a positive linear velocity. In such a situation, the initial state s_0 of $\mathscr{A}$ is clearly an ICS (no matter what $\mathscr{A}$ does it will end up being crushed by $\mathscr{B}_m$). It is however possible to select $\mathscr{A}$'s initial position and linear velocity such that s_0 is p-safe.

The parameters for this scenario were set as follows: $v_{\max} = 20\ \mathrm{m\,s^{-1}}$ (maximum velocity of $\mathscr{A}$ and $\mathscr{B}_m$), $u_{\alpha_{\max}} = 7\ \mathrm{m\,s^{-2}}$. The radius of $\mathscr{A}$ and $\mathscr{B}_m$ was 2.5 m and

[7] Downloadable from http://emotion.inrialpes.fr/fraichard/films/11-auro-passavoid.wmv.

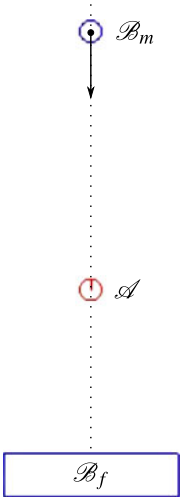


Fig. 14 1D compactor scenario

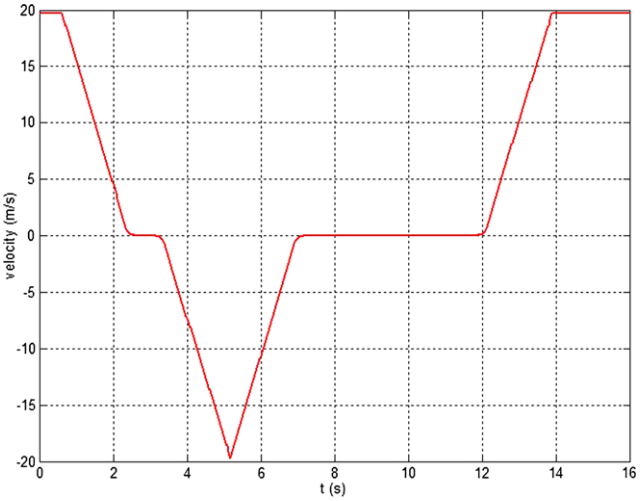


Fig. 15 Velocity profile of $\mathscr{A}$ for the 1D compactor scenario

the sensor range, i.e. the maximum radius of the field-of-view, was 80 m. The control space sampling set U was obtained through a regular discretization of the control set $[-u_{\alpha_{\max}}, u_{\alpha_{\max}}]$. The set of braking trajectories $\mathscr{E}$ used by ICS^b-CHECK comprised one δ-braking trajectory defined by a constant minimum linear deceleration $u_\alpha = -u_{\alpha_{\max}}$.

In this scenario, when driven by PASSAVOID, $\mathscr{A}$ exhibits the following behavior in order to always remain in p-safe states:

1. The increasing approach of $\mathscr{B}_m$ forces $\mathscr{A}$ to gradually decrease its velocity until it stops.
2. $\mathscr{A}$ backs up in order to avoid collision with $\mathscr{B}_m$ (recall that PASSAVOID is biased towards keeping $\mathscr{A}$ in motion).
3. While backing up, $\mathscr{A}$ gets closer to $\mathscr{B}_f$. At some point, it forces $\mathscr{A}$ to reduce its velocity.
4. $\mathscr{A}$ is now at rest next to $\mathscr{B}_f$, it will soon be hit by $\mathscr{B}_m$.
5. $\mathscr{A}$ is in collision with $\mathscr{B}_m$ ($t = 7$ s).
6. When the collision with $\mathscr{B}_m$ is over,[8] $\mathscr{A}$ resumes its upward motion.

The evolution of $\mathscr{A}$'s velocity in this scenario is depicted in Fig. 15. As simple as it may appear, this scenario shows how PASSAVOID seeks to avoid collision with $\mathscr{B}_m$ in a natural way (by braking down and shifting in reverse). However, when $\mathscr{A}$ is trapped, PASSAVOID guarantees that the robot will be at rest when the collision occurs.

7.4.2 Blind crowd scenario

The blind crowd scenario is more challenging. It features 22 moving objects moving arbitrarily in a 2D workspace. The objects are blind in the sense that their motion is unaffected by the other objects (Fig. 16).

[8] Assuming that $\mathscr{B}_m$ sort of passes through $\mathscr{A}$.

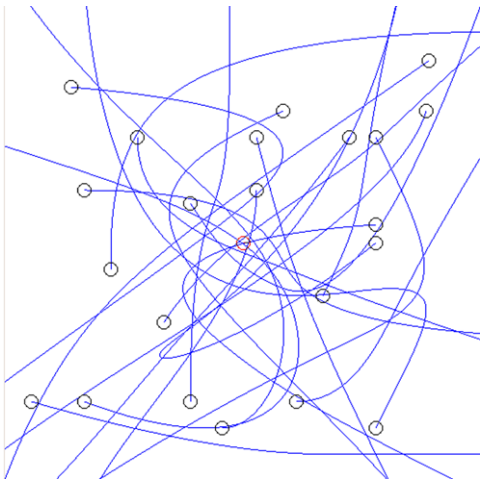


Fig. 16 Blind crowd scenario: it features 22 moving objects with their future trajectories. The robot $\mathscr{A}$ is at the center facing left

The parameters for this scenario were set as follows: $v_{\max} = 15\ \mathrm{m\,s^{-1}}$ (maximum velocity of $\mathscr{A}$ and of the moving objects), $\xi_{\max} = \pi/3$ rad, $u_{\alpha_{\max}} = 7\ \mathrm{m\,s^{-2}}$, $u_{\xi_{\max}} = 1.54\ \mathrm{rad\,s^{-1}}$. The radius of the disk objects was 2.5 m and the sensor range, i.e. the maximum radius of the field-of-view, was 80 m. The control space sampling set U was obtained through a regular discretization of the 2D control set $[-u_{\alpha_{\max}}, u_{\alpha_{\max}}] \times [u_{\xi_{\max}}, u_{\xi_{\max}}]$, and the set of braking trajectories $\mathscr{E}$ used by ICS^b-CHECK comprised 9 δ-braking trajectories defined by a constant minimum linear deceleration $u_\alpha = -u_{\alpha_{\max}}$ and a constant steering angle velocity $|u_\xi| \leq u_{\xi_{\max}}$.

Figure 17 presents four snapshots taken at different time instants of one run of PASSAVOID in this scenario. Each snapshot feature $\mathscr{A}$, the moving objects and the corresponding field-of-view. The set of ICS^b are also overlaid on the figure (black region).In the sequence, $\mathscr{A}$ is generally mov-

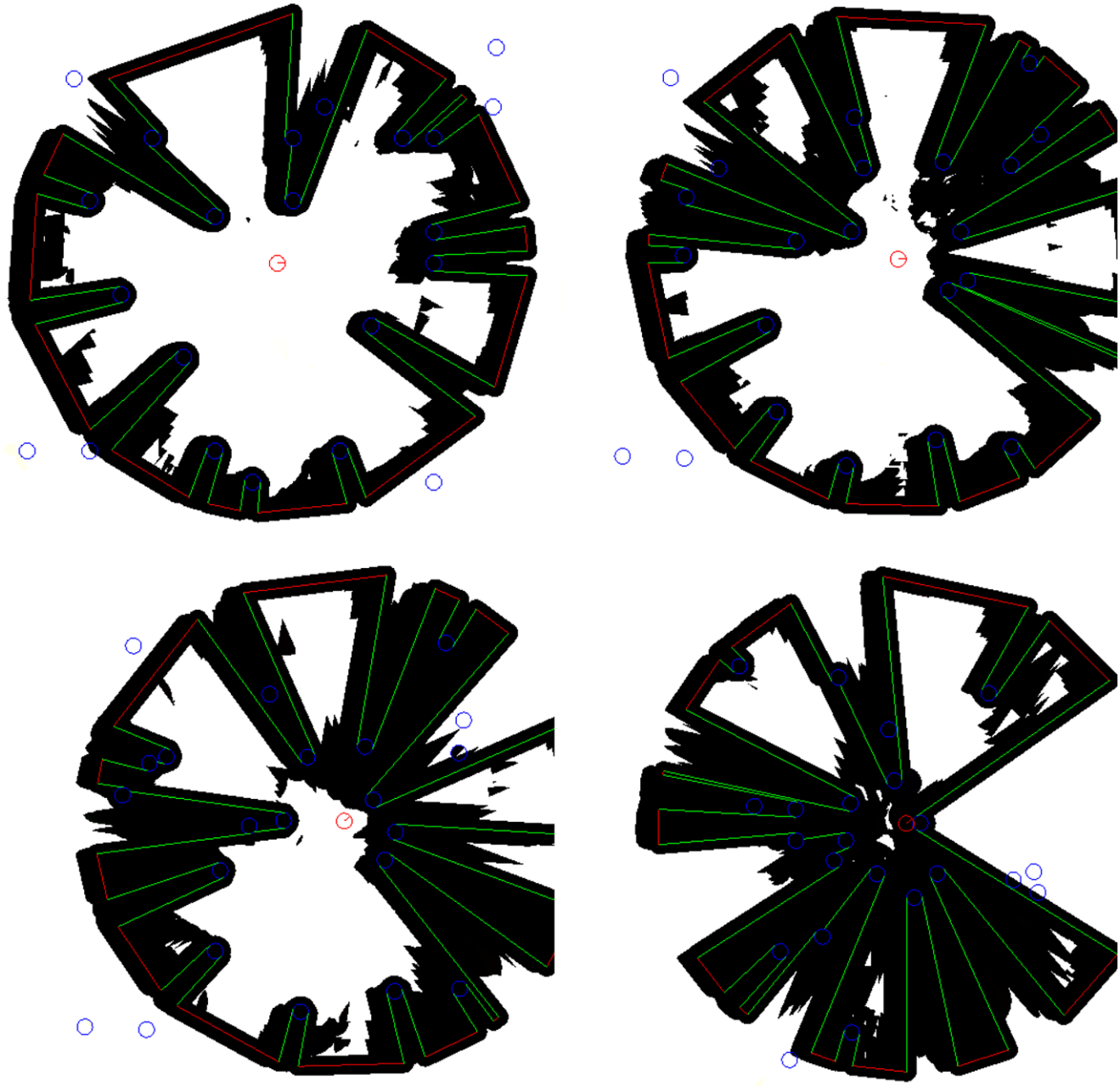


Fig. 17 Four snapshots of PASSAVOID at work in the blind crowd scenario (the *black region* represents the ICS^b). $\mathscr{A}$ is at rest in the *bottom-left snapshot* (a collision is imminent) and in collision in the *bottom-right one*

ing to the right. In the course of several runs, these experiments have demonstrated the capability of PASSAVOID to enforce passive motion safety: whenever a collision took place, $\mathscr{A}$ was at rest.

7.5 Complexity and performance

The computational time complexity of PASSAVOID grows linearly with n_s, the size of the control space sampling set U (*forall* loop of Algorithm 3), and the complexity of one iteration depends primarily on the complexity of ICS^b-CHECK (δ-p-safety test in line #6 of Algorithm 3).

Central to ICS^b-CHECK is the computation of $\mathrm{ICS}^b{}_{\hat{\mathbf{z}}_\mathbf{c}}(\mathscr{B}_i, \tilde{u}^*)$ (line #5 of Algorithm 4). Assuming a temporally discrete model of the future (with a fixed time step Δt), and thanks to the use of standard graphics rendering technique and GPU-based programming, this step can be done in time linear with $n_t = T_h/\Delta t$, the number of the time steps used to represent the model of the future. As illustrated in Sect. 7.3, the union and the intersection in lines #7 and #12 of Algo-

Table 1 Average running time of ICS^b-CHECK *wrt* n_o, the number of objects ($n_b = 9, n_t = 71$)

n_o	4	10	17	22
Running time (ms)	49	101	123	138

rithm 4 are readily obtained as by-products of the computation of $\mathrm{ICS}^b_{\hat{\mathbf{z}}_\mathbf{c}}(\mathscr{B}_i, \tilde{u}^*)$ for every braking trajectory.

Concerning the overall complexity of ICS^b-CHECK, it grows linearly with n_b, the size of the set of braking trajectories and n_o, the number of objects (**forall** loops of Algorithm 4). The final time complexity of PASSAVOID is $O(n_s n_b n_o n_t)$.

The current implementation of both ICS^b-CHECK and PASSAVOID has been done in C++ on an average laptop computer.[9] As expected, the running times observed for ICS^b-CHECK depends on the values of n_b, n_o and n_t. Table 1 gives the average running times of ICS^b-CHECK *wrt* n_0, the number of objects. These running times are encouraging and could further be improved thanks to code optimization[10] or the use of a more powerful desktop.

8 Conclusion and future work

This paper has addressed the problem of navigating in a provably safe manner a mobile robot with a limited field-of-view placed in a unknown dynamic environment. Since absolute motion safety is impossible to guarantee in such a situation, the position taken in this paper was to settle for a weaker level of motion safety dubbed *passive motion safety*: it guarantees that, if a collision takes place, the robot will be at rest. It seemed a reasonable choice given the harsh constraints imposed by a limited field-of-view and the lack of information about the environment and its future evolution. As limited as it may appear, passive motion safety is interesting for two reasons: (1) it allows to provide at least one form of motion safety guarantee in the challenging scenarios considered, and (2) if every moving object in the environment enforces it then no collision will take place at all.

The primary contribution of this paper has been the concept of *Braking ICS*, i.e. a version of the ICS corresponding to passive motion safety. Passive motion safety can be obtained by avoiding Braking ICS at all times. It has been shown that Braking ICS verified properties that have allowed the design of an efficient *Braking ICS-Checking algorithm*. The Braking ICS-Checking algorithm has then been integrated in a reactive navigation scheme called PASSAVOID whose passive motion safety has been formally established. PASSAVOID can drive a planar robot with arbitrary dynamics and a limited field-of-view in a unknown dynamic environment and it is guaranteed that the robot always stay away from Braking ICS no matter what happens in the environment. This work could be extended in the following directions:

- In certain situations, PASSAVOID may drive the robot to a collision state although such a collision could have been avoided. This is due to PASSAVOID's lack of foresight[11] and the fact that it is constrained to drive the robot from a ICS^b-free state to another ICS^b-free state. Besides, PASSAVOID is not concerned with driving the robot to a given goal. These issues could be addressed by turning PASSAVOID into a Partial Motion Planner, i.e. an interruptible motion planning scheme that strives to compute a trajectory towards a given goal and valid for the next k time steps, k being determined by the constraints imposed by the current situation (Petti and Fraichard 2005). Such an extension would yield a navigation scheme better able to avoid collisions and to reach a given goal while retaining the passive motion safety guarantee.
- In some applications, passive motion safety can be too limited; it could be interesting to explore more sophisticated levels of motion safety such as the *passive friendly motion safety* mentioned in Macek et al. (2009): it guarantees that, if a collision takes place, the robot will be at rest and the colliding object could have had the time to stop or avoid the collision (if it wanted to). Such a motion safety level assume that the moving objects have cognitive abilities and are not hostile (which happens to be true in many situations).

In general, it could be interesting to explore other forms of motion safety depending on the particulars of the navigation problem at hand.

References

Althoff, D., Althoff, M., Wollherr, D., & Buss, M. (2010). Probabilistic collision state checker for crowded environments. In *IEEE int. conf. on robotics and automation*, Anchorage, USA. doi:10.1109/ROBOT.2010.5509369.

Bautin, A., Martinez-Gomez, L., & Fraichard, T. (2010). Inevitable collision states, a probabilistic perspective. In *IEEE int. conf. robotics and automation*, Anchorage, USA. doi:10.1109/ROBOT.2010.5509233.

Bekris, K., & Kavraki, L. (2010). Greedy but safe replanning under kinodynamic constraints. In *IEEE int. conf. robotics and automation*, Rome, Italy. doi:10.1109/ROBOT.2007.363069.

[9] Intel Core i7 1.6 GHz CPU, 4 GB RAM, ATI Mobility Radeon HD 4500 GPU.

[10] A CUDA implementation is underway.

[11] PASSAVOID is purely reactive, its sole purpose is to compute the control to apply for the next time step.

Bekris, K., Tsianos, K., & Kavraki, L. (2009). Safe and distributed kinodynamic replanning for vehicular networks. *Mobile Networks and Applications*, 14(3). doi:10.1007/s11036-009-0152-y.

Chan, N., Zucker, M., & Kuffner, J. (2007). Towards safe motion planning for dynamic systems using regions of inevitable collision. In *Collision-free motion planning for dynamic systems workshop*, Rome, Italy.

Chung, W., Kim, S., Choi, M., Choi, J., Kim, H., Moon, C., & Song, J. (2009). Safe navigation of a mobile robot considering visibility of environment. *IEEE Transactions of Industrial Electronics, 56*(10). doi:10.1109/TIE.2009.2025293.

Ferguson, D., Howard, T., & Likhachev, M. (2008). Motion planning in urban environments. *Journal of Field Robotics, 25*(11–12). doi:10.1002/rob.20265.

Fiorini, P., & Shiller, Z. (1998). Motion planning in dynamic environments using velocity obstacles. *International Journal of Robotics Research, 17*(7). doi:10.1177/027836499801700706.

Fletcher, L., Teller, S., Olson, E., Moore, D., Kuwata, Y., How, J., Leonard, J., Miller, I., Campbell, M., Huttenlocher, D., Nathan, A., & Kline, F. R. (2008). The MIT—Cornell collision and why it happened. *International Journal of Field Robotics, 25*(10).

Fox, D., Burgard, W., & Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine 4*(1).

Fraichard, T. (2007). A short paper about motion safety. In *IEEE Int. conf. robotics and automation*, Roma, Italy.

Fraichard, T., & Asama, H. (2004). Inevitable collision states: a step towards safer robots? *Advanced Robotics, 18*(10).

Frazzoli, E., Feron, E., & Dahleh, M. (2002). Real-time motion planning for agile autonomous vehicle. *Journal of Guidance, Control, and Dynamics, 25*(1).

Hsu, D., Kindel, R., Latombe, J. C., & Rock, S. (2002). Randomized kinodynamic motion planning with moving obstacles. *International Journal of Robotics Research, 21*(3).

Kalisiak, M., & van de Panne, M. (2007). Faster motion planning using learned local viability models. In *IEEE int. conf. robotics and automation*, Roma, Italy.

Kant, K., & Zucker, S. (1986) Toward efficient trajectory planning: the path-velocity decomposition. *International Journal of Robotics Research, 5*(3).

Kohout, R., Hendler, J., & Musliner, D. (1996). Guaranteeing safety in spatially situated agents. In *AAAI nat. conf. artificial intelligence*, Portland.

Kuwata, Y., Karaman, S., Teo, J., Frazzoli, E., How, J., & Fiore, G. (2009). Real-time motion planning with applications to autonomous urban driving. *IEEE Transations on Control Systems Technology, 17*(5). doi:10.1109/TCST.2008.2012116.

Lalish, E., & Morgansen, K. (2008). Decentralized reactive collision avoidance for multivehicle systems. In *IEEE conf. decision and control*, Cancun.

LaValle, S. (2006). *Planning algorithms*. Cambridge: Cambridge University Press.

LaValle, S., & Kuffner, J. (1999). Randomized kinodynamic planning. In *IEEE int. conf. robotics and automation*, Detroit, MI, USA.

Lumelsky, V., & Tiwari, S. (1994). Velocity bounds for motion planning in the presence of moving planar obstacles. In *IEEE-RSJ int. conf. intelligent robots and systems*, München, Germany.

Macek, K., Vasquez-Govea, D., Fraichard, T., & Siegwart, R. (2009). Towards safe vehicle navigation in dynamic urban scenarios. *Automatika, 50*(3–4).

Madhava Krishna, K., Alami, R., & Simeon, T. (2006). Safe proactive plans and their execution. *Robotics and Autonomous Systems, 54*(3).

Martinez-Gomez, L., & Fraichard, T. (2008). An efficient and generic 2D inevitable collision state-checker. In *IEEE-RSJ int. conf. intelligent robots and systems*, Nice, France.

Pallottino, L., Scordio, V., Bicchi, A., & Frazzoli, E. (2007) Decentralized cooperative policy for conflict resolution in multivehicle systems. *IEEE Transations on Robotics, 23*(6). doi:10.1109/TRO.2007.909810.

Petti, S., & Fraichard, T. (2005). Safe motion planning in dynamic environments. In *Proc. of the IEEE-RSJ int. conf. on intelligent robots and systems*, Edmonton, Canada. doi:10.1109/IROS.2005.1545549.

Reif, J., & Sharir, M. (1985). Motion planning in the presence of moving obstacles. In *IEEE int. symp. foundations of computer science*, Portland, USA.

Sadou, M., Polotski, V., & Cohen, P. (2004). Occlusion in obstacle detection for safe navigation. In *IEEE intelligent vehicles symp.*, Parma, Italy. doi:10.1109/IVS.2004.1336472.

Seder, M., & Petrovic, I. (2007). Dynamic window based approach to mobile robot motion control in the presence of moving obstacles. In *IEEE int. conf. robotics and automation*, Roma, Italy.

Van den Berg, J., & Overmars, M. (2008). Planning time-minimal safe paths amidst unpredictably moving obstacles. *International Journal of Robotics Research, 27*(11–12). doi:10.1177/0278364908097581.

Van den Berg, J., Lin, M., & Manocha, D. (2008). Reciprocal velocity obstacles for real-time multi-agent navigation. In *IEEE int. conf. robotics and automation*, Pasadena, US. doi:10.1109/ROBOT.2008.4543489.

Vatcha, R., & Xiao, L. (2008). Perceived CT-space for motion planning in unknown and unpredictable environments. In *Workshop on algorithmic foundations of robotics*, Guanajuato, Mexico.

Sara Bouraine received the Electronics Engineer degree from the University of Blida (AG) in 2004. She received her Magistere degree (Degree of Advanced Studies) in Electronics option Image processing from the University of Blida (AG) in 2007. Currently, she is a researcher at the DRP (Division Robotique et Productique) of CDTA (Centre de Developpement des Technologies Avancees) and her Doctorate es-science degree is in progress at the University of Blida. Her research interests are mobile robots, motion planning, localization, safety, dynamic map building (urban environment), detection and tracking of moving object.

Thierry Fraichard Since October 1993, Thierry Fraichard is an INRIA Chargé de Recherche (Research Scientist) in the INRIA Grenoble Rhône-Alpes Research Center.

Thierry Fraichard received his Ph.D in Computer Science from the Institut National Polytechnique de Grenoble (INPG) in April 1992 for his dissertation on "Motion Planning for a Nonholonomic Mobile in a Dynamic Workspace". In March 2006, INPG awarded him the Habilitation à Diriger des Recherches (Accreditation to Supervise Research) for his work entitled "Contributions to Motion Planning". Thierry Fraichard's research focuses on motion autonomy for vehicles and mobile robots with a special empha-

sis on motion safety, motion planning (for nonholonomic systems, in dynamic workspaces, and in the presence of uncertainty), prediction of the future motion of moving objects, navigation amidst human beings, and the design of control architectures for autonomous vehicles.

Hassen Salhi was born in 1953 in Boufarik (AG). He obtained his B.S., M.S. and Ph.D. in Electrical Engineering from the University of Illinois at Urbana-Champaign in 1977, 1979 and 1983 respectively. He is currently an Associate Professor in Electrical Engineering at the University Saad Dahlab of Blida (AG), where he is the Head of the Research Laboratory SET in Control Engineering. His interests includes Control and Modeling of Biological Systems, Fractional Order Systems and Robotics.