Fully-Discrete Schemes for the Value Function of Pursuit-Evasion Games with State Constraints

Emiliano Cristiani
Dipartimento di Matematica
SAPIENZA – Università di Roma
Rome, Italy
emiliano.cristiani@gmail.com

Maurizio Falcone
Dipartimento di Matematica
SAPIENZA – Università di Roma
Rome, Italy
falcone@mat.uniroma1.it

Abstract

We deal with the approximation of a generalized Pursuit-Evasion game with state constraints. Our approach is based on the Dynamic Programming principle and on the characterization of the lower value v of the game via the Isaacs equation. Our main result is the convergence of the fully-discrete scheme for Pursuit-Evasion games under continuity assumptions on v and some geometric assumptions on the dynamics and on the set of constraints $\overline{\Omega}$. We also analyze the Tag-Chase game in a bounded convex domain when the two players have the same velocity and we prove that in the constrained case the time of capture is finite. Some hints to improve the efficiency of the algorithm on serial and parallel machines will be also given. An extensive numerical section will show the accuracy of our method for the approximation of the value function v and of the corresponding optimal trajectories in a number of different configurations.

Key words. Differential games, pursuit-evasion games, state constraints, Isaacs equation, fully-discrete scheme, feedback controls, Tag-Chase game, parallel algorithms.

AMS Subject Classifications. Primary 65M12; Secondary 49N70, 49L25.

1 Introduction

In this paper, we present and analyze a numerical approximation scheme for 2-player Pursuit-Evasion games with state constraints. The scheme is based on the dynamic programming approach and the convergence results are obtained in the

P. Bernhard et al. (eds.), *Advances in Dynamic Games and Their Applications*,
Annals of the International Society of Dynamic Games 10, DOI: 10.1007/978-0-8176-4834-3_11,

framework of viscosity solutions (see the survey papers [3,7,17] for a general introduction to this topic without constraints).

Our main result shows that the solution of the fully-discrete problem converges to the time-discrete value function as the mesh size Δx goes to zero provided a technical “consistency" assumption on the triangulation is satisfied. Moreover, an *a priori* error bound on that approximation is proved in Theorem 3.1 and a very easy sufficient condition guaranteeing consistency is shown (Corollary 3.3). The proof of the main result is obtained by extending to games a technique presented in [18] for the minimum time problem without state constraints and adapting the approach for state-constrained control problems presented in [10].

Note that the convergence of the fully-discrete solution to the solution of the continuous problem in the free (*i.e.,* unconstrained) case is proved in [6], but this result cannot be directly extended to the constrained case. In [10], a convergence result is proved for constrained control problems, but it strictly relies on the fact that the time-discrete value function is continuous so we cannot apply the same ideas here.
In order to prove convergence of the approximate solution of the fully-discrete scheme to the value function, we have coupled our result with the convergence result obtained by Bardi et al. [8] (see also [21,20]) in the framework of Pursuit-Evasion games. This allows to conclude that, under suitable assumptions, the convergence of the fully-discrete solution converge to the solution of the continuous problem when the time and space steps, Δt and Δx, go to zero (although a precise estimate of the order of convergence is still missing).

It should be noted that very few results on *constrained* differential games are available although several interesting problems with state constraints have been studied in the classical books by Isaacs [19] and Breakwell [9]. The aim of those contributions is mainly to analyze the games under study in order to determine directly the optimal strategies for the players *avoiding* in this way the Isaacs equation. The main theoretical contributions to the characterization of the value function for state constrained problems are, by our knowledge, the papers by Alziary de Roquefort [1], Bardi *et al.* [8], and Cardaliaguet *et al.* [12]. From the numerical point of view, the list of contributions is even shorter. The first examples of computed optimal trajectories for Pursuit-Evasion games appeared in the work by Alziary de Roquefort [2]. In Bardi *et al.* [6], there are some interesting tests in $\Omega \subset \mathbb{R}^2$ with state constraints and discontinuous value function. In [4], the effect of the boundary conditions for the free problem in $\mathbb{R}^4$ is studied. In the paper Cardaliaguet *et al.* [11], a modified viability kernel algorithm (see [13] for more details on this approach) is presented and a convergence proof for that approximation scheme is given. Finally, let us also mention the paper of Pesch *et al.* [22] where the optimal trajectories are computed by means of neural networks (again avoiding the solution of the Isaacs equation).

Another contribution of this paper is the analysis of the constrained Tag-Chase game where the two players run one after the other in a bounded convex domain *with the same velocity*. In this case, the value function is discontinuous and most of the theoretical results we know for the Tag-Chase game do not hold. We prove that the time of capture is finite if the capture occurs whenever the distance between the Pursuer and the Evader is less than a positive parameter ε (the radius of capture). This shows that the presence of constraints can change dramatically the result of the game. In fact, in the unconstrained Tag-Chase game where both the players have the same velocity, the Evader always wins (and the time of capture is $+\infty$). The paper is organized as follows. In Sec. 2, we set up our problem, introduce the notations and present our approximation scheme. Section 3 is devoted to the convergence analysis, we prove first some properties of the discrete value functions v_h and v_h^k corresponding, respectively, to the solutions of the time-discrete and fully-discrete schemes. The final convergence result is obtained coupling the error estimate of Theorem 3.1 with the results in [8]. In Sec. 4, we deal with the Tag-Chase game in a convex domain showing that this problem has a finite capture time also when the two players have the same maximal velocity. Section 5 presents some hints for the construction of the algorithms and, in particular, it deals with two features which allow to reduce the computational cost for the solution of the Isaacs equation: a high-dimensional interpolation technique and the symmetry properties of the Tag-Chase game played in a square domain. Finally, Sec. 6 presents several tests for different geometric configurations of the state constraints (convex and non-convex) as well as for various choices of the relative velocities of the two players. We analyze the results in terms of the value function v but also in terms of the optimal trajectories that one can compute using v.

2 The fully-discrete approximation scheme

Let us start introducing the problem and our notations. The system describing the dynamics is

$$\begin{cases} \dot{y}(t) = f(y(t), a(t), b(t))\,, \;\; t > 0 \\ y(0) = x\,, \end{cases} \tag{1}$$

where $y(t) \in \mathbb{R}^{2n}$ is the state of the system, $a(\cdot) \in \mathcal{A}$ and $b(\cdot) \in \mathcal{B}$ are, respectively, the controls of the first and the second player, $\mathcal{A}$ and $\mathcal{B}$ being the sets of *admissible strategies* defined as

$$\mathcal{A} = \{a(\cdot) : [0, +\infty) \to A, \text{ measurable}\}\,,$$

$$\mathcal{B} = \{b(\cdot) : [0, +\infty) \to B, \text{ measurable}\}\,,$$

and A and B are given compact sets of $\mathbb{R}^m$. We will always assume that

$$\begin{cases} f : \mathbb{R}^{2n} \times A \times B \to \mathbb{R}^{2n} \text{ is continuous w.r. to all the variables and} \\ \text{there exists } L > 0 \text{ such that } |f(y_1, a, b) - f(y_2, a, b)| \le L|y_1 - y_2| \\ \text{ for all } y_1, y_2 \in \mathbb{R}^{2n}, \; a \in A, \; b \in B. \end{cases} \tag{2}$$

We will denote the solution of (1) by $y(t; x, a(\cdot), b(\cdot))$.
A target set $\mathcal{T} \subset \mathbb{R}^{2n}$ is given and it is assumed to be closed. The first player, called the *Pursuer* and denoted by P, wants to drive the system to $\mathcal{T}$. The second player, called the *Evader* and denoted by E, wants to drive the system away.

We deal with the natural extension of the minimum time problem, so we define the *payoff* of the game as the first (if any) time of arrival $T(x)$ on the target $\mathcal{T}$ for the trajectory solution of (1) starting at x. Note that, as usual, we set $T(x) = +\infty$ if the trajectory will not reach the target. The two players are opponents since the first player wants to minimize the payoff associated to the solution of the system whereas the second player wants to maximize it.

From now on, we restrict our analysis to Pursuit-Evasion games although some results are still valid in a more general context. We denote the coordinate of the space by $x = (x_P, x_E)$ where $x_P, x_E \in \mathbb{R}^n$. Each player can control only his own dynamics, *i.e.,* f has the form $f(x, a, b) = \big(f_P(x_P, a), f_E(x_E, b)\big)$. The state of the system is $y(t) = \big(y_P(t), y_E(t)\big)$ and a typical target has the form

$$\mathcal{T} = \{(x_P, x_E) \in \mathbb{R}^{2n} : |x_P - x_E| \leq \varepsilon\}, \quad \varepsilon \geq 0$$

so in the unconstrained case the target is unbounded. As we said in the Introduction, we want to construct a numerical approximation for Pursuit-Evasion games *with state constraints*. This means that player P has to steer the system to the target satisfying the constraint $y_P(t) \in \overline{\Omega}_1$ for every t whereas player E must satisfy $y_E(t) \in \overline{\Omega}_2$ for every t, where Ω_1, Ω_2 are given bounded sets. The whole problem is set in $\overline{\Omega} \subset \mathbb{R}^{2n}$ where $\Omega := \Omega_1 \times \Omega_2$. Note that one player cannot force the other to respect or ignore the state constraints just because every player can affect only his dynamics and he is completely responsible for his strategy/trajectory. In the constrained game it is natural to replace $\mathcal{T}$ with $\mathcal{T} \cap \overline{\Omega}$.

The analysis of the continuous model with state constraints via dynamic programming techniques which is the basis for our approximation can be found in [8]. Let us start giving the time-discrete and the corresponding fully-discrete version of the differential game with state constraints. We will consider a discrete version of the dynamics based on the Euler scheme, namely

$$\begin{cases} y_{n+1} = y_n + hf(y_n, a_n, b_n) \\ y_0 = x \,, \end{cases}$$

where $h = \Delta t$ is a positive time step and we denote by $y(n; x, \{a_n\}, \{b_n\})$ its solution at time nh corresponding to the initial condition $x = (x_P, x_E)$ and to the discrete strategies $\{a_n\}, \{b_n\}$. The state constraints obviously require that $y(n; x, \{a_n\}, \{b_n\}) \in \overline{\Omega}$ for all $n \in \mathbb{N}$.
We define the *constrained admissible strategies* for each player

$$\mathcal{A}_x := \{a(\cdot) \in \mathcal{A} : y_P(t; x, a(\cdot)) \in \overline{\Omega}_1 \,, \text{ for all } t \geq 0\}$$

$$\mathcal{B}_x := \{b(\cdot) \in \mathcal{B} : y_E(t; x, b(\cdot)) \in \overline{\Omega}_2 \,, \text{ for all } t \geq 0\}$$

and their time-discrete version

$$\mathcal{A}_x^h := \{\{a_n\} : a_n \in A \text{ and } y_P(n; x, \{a_n\}) \in \overline{\Omega}_1 \,, \text{ for all } n \in \mathbb{N}\}$$

$$\mathcal{B}_x^h := \{\{b_n\} : b_n \in B \text{ and } y_E(n; x, \{b_n\}) \in \overline{\Omega}_2 \,, \text{ for all } n \in \mathbb{N}\}.$$

Note that the constrained strategies now depend on x and on the state constraints. We will always assume that $\mathcal{A}_x \neq \emptyset$, $\mathcal{B}_x \neq \emptyset$, $\mathcal{A}_x^h \neq \emptyset$, and $\mathcal{B}_x^h \neq \emptyset$ for all $x \in \overline{\Omega}$ and h sufficiently small.

In the same way, we have to define the sets of admissible controls for every point $x \in \overline{\Omega}$. Let us start with the continuous problem. Following [20,21], we will select the subsets of admissible controls, denoted by $A(y)$ and $B(y)$, for every $y = (y_P, y_E) \in \overline{\Omega \setminus \mathcal{T}}$,

$$A(y) = \{a \in A : \exists\, r > 0 \text{ such that } y_P(t; y_P', a) \in \overline{\Omega}_1 \text{ for } t \in [0, r] \text{ and } y_P' \in B(y_P, r) \cap \overline{\Omega}_1\}, \quad (3)$$

$$B(y) = \{b \in B : \exists\, r > 0 \text{ such that } y_E(t; y_E', b) \in \overline{\Omega}_2 \text{ for } t \in [0, r], \text{ and } y_E' \in B(y_E, r) \cap \overline{\Omega}_2\}. \quad (4)$$

For the time-discrete dynamics we define an analogue of subsets of $A(y)$ and $B(y)$ as follows:

$$A_h(y) := \{a \in A : y_P + h f_P(y_P, a) \in \overline{\Omega}_1\}\,, \qquad y \in \overline{\Omega \setminus \mathcal{T}}, \quad (5)$$

$$B_h(y) := \{b \in B : y_E + h f_E(y_E, b) \in \overline{\Omega}_2\}\,, \qquad y \in \overline{\Omega \setminus \mathcal{T}}. \quad (6)$$

The meaning of the above definitions is very clear: in order to guarantee that his trajectory satisfies his own state constraints over a time interval h, player P (respectively, player E) has to choose his control in $A_h(y)$ (respectively, $B_h(y)$). These subsets describe at every point $y \in \overline{\Omega \setminus \mathcal{T}}$ the “allowed directions" for each player, naturally they depend also on h, the dynamics and the constraints. Note that $A_h(y) \equiv A$ (respectively, $B_h(y) \equiv B$) in the unconstrained case.
We will also assume that

$$\exists\, h_0 > 0 \text{ s.t. } A_h(x) \neq \emptyset \text{ and } B_h(x) \neq \emptyset \text{ for all } (h, x) \in (0, h_0] \times \overline{\Omega}\,. \quad (7)$$

Definition 2.1. A *discrete strategy* for the first player is a map $\alpha_x : \mathcal{B}_x^h \to \mathcal{A}_x^h$. It is *nonanticipating* if $\alpha_x \in \Gamma_x^h$, where

$$\Gamma_x^h := \{\alpha_x : \mathcal{B}_x^h \to \mathcal{A}_x^h : b_n = \tilde{b}_n \text{ for all } n \leq n' \text{ implies } \alpha_x[\{b_k\}]_n = \alpha_x[\{\tilde{b}_k\}]_n \text{ for all } n \leq n'\}. \quad (8)$$

Let us define the reachable set as the set of all points from which the system can reach the target

$$\mathcal{R}^h := \{x \in \mathbb{R}^n : \text{ for all } \{b_n\} \in \mathcal{B}^h_x \text{ there exists } \alpha_x \in \Gamma^h_x \text{ and } \bar{n} \in \mathbb{N} \\ \text{such that } y(\bar{n}; x, \alpha_x[\{b_n\}], \{b_n\}) \in \mathcal{T}\}. \quad (9)$$

Then we define

$$n_h(x, \{a_n\}, \{b_n\}) := \begin{cases} \min\{n \in \mathbb{N} : y(n; x, \{a_n\}, \{b_n\}) \in \mathcal{T}\} & x \in \mathcal{R}^h \\ +\infty & x \notin \mathcal{R}^h \ . \end{cases}$$

We will consider for our approximation the discrete lower value of the game, which is

$$T_h(x) := \inf_{\alpha_x \in \Gamma^h_x} \sup_{\{b_n\} \in \mathcal{B}^h_x} h n_h(x, \alpha_x[\{b_n\}], \{b_n\})$$

and its Kružkov transform

$$v_h(x) := 1 - e^{-T_h(x)}, \quad x \in \overline{\Omega}. \quad (10)$$

Note that a similar construction can be done for the upper value of the game. The Dynamic Programming Principle (DPP) for Pursuit-Evasion games with state constraints is proved in [8] which also gives a characterization of the lower and upper value of the game in terms of the Isaacs equation. From the discrete version of the DPP (see [8]), we can conclude that the time-discrete value function v_h is the unique bounded solution of

$$\begin{cases} v_h(x) = \max\limits_{b \in B_h(x)} \min\limits_{a \in A_h(x)} \{\beta v_h(x + hf(x, a, b))\} + 1 - \beta & x \in \overline{\Omega} \backslash \mathcal{T} \\ v_h(x) = 0 & x \in \mathcal{T} \cap \overline{\Omega} \end{cases} \quad (\text{HJI}_h\text{–}\Omega)$$

where $\beta := e^{-h}$ and the $maxmin$ is obviously computed on the sets of admissible controls for the constrained game. In order to achieve the fully-discrete equation we build a regular triangulation of $\overline{\Omega}$ denoting by X the set of its nodes $x_i, i = 1, \ldots, N$ and by $\mathcal{S}$ the set of simplices $S_j,\ j = 1, \ldots, L$. $V(S_j)$ will denote the set of the vertices of a simplex S_j and the space discretization step will be denoted by k where $k := \max_j\{diam(S_j)\}$. Let us define $D \equiv (\overline{\Omega} \backslash \mathcal{T}) \cap X$.

The fully-discrete approximation scheme for the constrained case is

$$\begin{cases} v^k_h(x_i) = \max\limits_{b \in B_h(x_i)} \min\limits_{a \in A_h(x_i)} \left\{\beta v^k_h(x_i + hf(x_i, a, b))\right\} + 1 - \beta & x_i \in D \\ v^k_h(x_i) = 0 & x_i \in \mathcal{T} \cap X \\ \\ v^k_h(x) = \sum_j \lambda_j(x) v^k_h(x_j),\ 0 \le \lambda_j(x) \le 1, \quad \sum_j \lambda_j(x) = 1 & x \in \overline{\Omega}\ . \end{cases} \quad (\text{HJI}^k_h\text{–}\Omega)$$

As in the unconstrained problem, the choice of linear interpolation is not an obligation and it was made here just to simplify the presentation. Let us denote by W^k

the set

$$W^k := \{w \in C(\overline{\Omega}) : \nabla w(x) = \text{constant for all } x \in S_j\,, j = 1, \dots, L\}\,.$$

The proof of the following theorem can be obtained by simple adaptations of the standard proof for the free fully-discrete scheme (see, e.g., [6]).

Theorem 2.2. Equation (HJI_h^k–Ω) has a unique solution $v_h^k \in W^k$ such that $v_h^k : \overline{\Omega} \to [0,1]$.

Sketch of the proof. The right-hand side of the first equation is (HJI_h^k–Ω) defines a map $F : \mathbb{R}^N \to \mathbb{R}^N$ where N is the cardinality of the set of nodes in the triangulation. The proof relies on the fact that F is a contraction map so there exists a unique fixed point V^* and $v_h^k(x_i) = V_i^*$, $i = 1, \dots, N$. □

3 Convergence of the fully-discrete numerical scheme

The convergence of the fully-discrete scheme will be based on two ingredients. The first is an *a priori* bound for $v_h^k - v_h$ which will be obtained studying the properties of v_h on a family of approximate “reachable sets". This bound is proved for a *general dynamics* f and does not depend on the regularity of v. Then, we couple this bound with the convergence result in [8] where they prove that v_h converges to v uniformly in $\overline{\Omega \setminus \mathcal{T}}$.
Let us define $\mathcal{R}_0 := \mathcal{T}$ and

$$\mathcal{R}_n := \left\{ x \in \overline{\Omega} \setminus \bigcup_{j=0}^{n-1} \mathcal{R}_j : \text{ for all } b \in B_h(x) \text{ there exists } \hat{a}_x(b) \in A_h(x) \right.$$
$$\left. \text{such that } x + hf(x, \hat{a}_x(b), b) \in \mathcal{R}_{n-1} \right\}, \quad n \geq 1. \quad (11)$$

See [18] for an analogous definition in the framework of the minimum time problem.

Remark 3.1. By definition, the shape of the sets $\{\mathcal{R}_n\}_{n\in\mathbb{N}}$ strictly depends on f, Ω, A and B. Moreover, the following properties hold true:

1. $\mathcal{R}_n \cap \mathcal{R}_m = \emptyset$ for all $n \neq m$;
2. If $\mathcal{R}_p = \emptyset$ for some $p \in \mathbb{N}$, then $\mathcal{R}_q = \emptyset$ for any $q \geq p$;
3. The sets $\{\mathcal{R}_n\}_{n\in\mathbb{N}}$ are the level sets of v_h and v_h has jump discontinuities on the boundary of each of them.

In the sequel will always assume that

$$\overline{\Omega} = \bigcup_{j=0}^{\infty} \mathcal{R}_j. \quad (12)$$

Note that (12) can be interpreted as a sort of small time controllability assumption and that it is not really restrictive since if there exists a point $x \in \overline{\Omega} \setminus \bigcup_j \mathcal{R}_j$ this means that player P cannot win the game from that point (*i.e.,* he cannot drive the system to the target) and then $v_h(x) = 1$.

We introduce two important assumptions on the triangulation. Let $S \in \mathcal{S}$ be a simplex, the first assumption is (see Fig. 1):

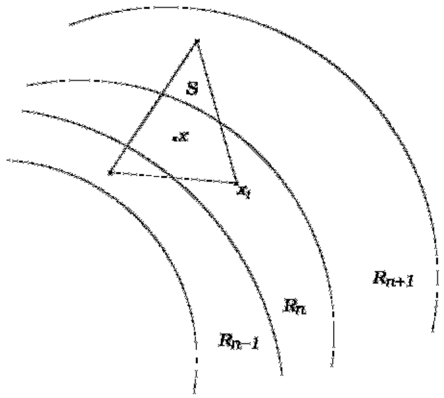

Figure 1: A simplex S crossing $\mathcal{R}_n$, $\mathcal{R}_{n-1}$, and $\mathcal{R}_{n+1}$.

$$x \in S \cap \mathcal{R}_n \Rightarrow V(S) \subset \mathcal{R}_{n-1} \cup \mathcal{R}_n \cup \mathcal{R}_{n+1}. \tag{13}$$

It means that the space discretization cannot be too coarse with respect to the time discretization. The second assumption is the "consistency" of the triangulation (see Fig. 1).

Definition 3.2. We say that a triangulation is "consistent" if $S \cap \mathcal{R}_n \neq \emptyset$ implies that there exists at least one vertex $x_i \in V(S)$ such that $x_i \in \mathcal{R}_n$.

The above assumption requires that every simplex of the triangulation cannot cross a level set $\mathcal{R}_n$ without having a vertex in it and, as we will see, is crucial for the convergence of the scheme. This condition will be always satisfied for k sufficiently small as we will see in Corollary 3.4. Let v_h and v_h^k denote, respectively, the solution of (HJI$_h$–Ω) and (HJI$_h^k$–Ω). We now state the main result of the paper.

Theorem 3.3. Let Ω be an open bounded set. Let (2), (12), (13) hold true and let the triangulation be "consistent". Then, for $n \geq 1$:

a) $v_h(x) \leq v_h(y)$, for any $x \in \bigcup_{j=0}^{n} \mathcal{R}_j$, for any $y \in \overline{\Omega} \setminus \bigcup_{j=0}^{n} \mathcal{R}_j$;

b) $v_h(x) = 1 - e^{-nh}$, for any $x \in \mathcal{R}_n$;
c) $v_h^k(x) = 1 - e^{-nh} + O(k) \sum_{j=0}^{n} e^{-jh}$ for any $x \in \mathcal{R}_n$;
d) There exists a constant $C > 0$ such that

$$|v_h(x) - v_h^k(x)| \leq \frac{Ck}{1 - e^{-h}}, \quad \text{for any } x \in \mathcal{R}_n.$$

Proof. $a)$ By induction. For $n = 0$ the statement is true since

$$0 = v_h(x) \leq v_h(y) \quad \text{for all } x \in \mathcal{T}, \quad \text{for all } y \in \overline{\Omega} \backslash \mathcal{T}.$$

Let the statement be true up to $n-1$. Suppose by contradiction that

$$\text{there exists } x \in \bigcup_{j=0}^{n} \mathcal{R}_j \text{ and } y \in \overline{\Omega} \backslash \bigcup_{j=0}^{n} \mathcal{R}_j \text{ such that } v_h(y) < v_h(x).$$

Therefore, there exists a (discrete) trajectory that starts from $\overline{\Omega} \backslash \bigcup_{j=0}^{n} \mathcal{R}_j$ and reaches the target in less then n time steps passing through $\mathcal{R}_n$. The contradiction follows by the definition of $\mathcal{R}_n$.

$b)$ By the definition of $\mathcal{R}_n$, for any $x \in \mathcal{R}_n$ we can find $n+1$ points $x^{(q)}$, $q = 0, \ldots, n$ such that $x^{(0)} = x$ and $x^{(q)} \in \mathcal{R}_{n-q}$. Introducing for simplicity the notations $a_q := a_{x^{(q)}}$ and $b_q := b_{x^{(q)}}$, we can write the sequence of the points $x^{(q)}$ more explicitly as

$$x^{(q+1)} = x^{(q)} + hf(x^{(q)}, \hat{a}_q(b_q^*), b_q^*),$$

where we use the * to indicate the optimal choice. As a consequence, the state of the system can reach the target in n steps and then $v_h(x) \leq 1 - e^{-nh}$. Suppose by contradiction that $v_h(x) < 1 - e^{-nh}$. As in $b)$, this means that the state has reached the target starting at x in less then n time steps but this is impossible since $x \in \mathcal{R}_n$.

$c)$ By construction we have $v_h^k(x_i) = 0$ for all $x_i \in \mathcal{R}_0 \cap X$. We now consider a generic point $x \in \mathcal{R}_0$ and let S be the simplex containing x. Since the triangulation is "consistent", S must have at least a vertex $x_{i_0} \in \mathcal{R}_0$ and then $v_h^k(x) = O(k)$ for all $x \in \mathcal{R}_0$ since $v_h^k \in W^k$. This implies, for all $x_i \in \mathcal{R}_1 \cap X$,

$$v_h^k(x_i) = \beta v_h^k(x_i + hf(x_i, a^*, b^*)) + 1 - \beta = \beta O(k) + 1 - \beta,$$

since $x_i + hf(x_i, a^*, b^*) \in \mathcal{R}_0$. We now consider a generic point $x \in \mathcal{R}_1$. By the same arguments there exists at least one vertex $x_{i_1} \in \mathcal{R}_1$ such that

$$v_h^k(x) = v_h^k(x_{i_1}) + O(k) = \beta O(k) + 1 - \beta + O(k) = 1 - \beta + (1 + \beta) O(k).$$

For any $x_i \in \mathcal{R}_2 \cap X$,

$$v_h^k(x_i) = \beta(1 - \beta + (1 + \beta) O(k)) + 1 - \beta = 1 - \beta^2 + (\beta + \beta^2) O(k),$$

and, for any $x \in \mathcal{R}_2$ it exists $x_{i_2} \in \mathcal{R}_2$ such that

$$v_h^k(x) = v_h^k(x_{i_2}) + O(k) = 1 - \beta^2 + (1 + \beta + \beta^2)O(k).$$

Continuing by recursion we obtain, for any $x \in \mathcal{R}_n$

$$v_h^k(x) = 1 - \beta^n + O(k)\sum_{j=0}^{n} \beta^j.$$

$d)$ By $b)$ and $c)$ there exists a positive constant C_1 such that

$$|v_h(x) - v_h^k(x)| = C_1 k \sum_{j=0}^{n} \beta^j \leq \frac{C_1 k}{1-\beta} = \frac{C_1 k}{1 - e^{-h}}.$$

□

Corollary 3.4. Let Ω be an open-bounded set. Let (2), (12) hold true. Moreover assume that

$$\min_{x,a,b} |f(x,a,b)| \geq f_0 > 0 \quad \text{and} \quad 0 < k \leq f_0 h. \tag{14}$$

Then, for $k \to 0^+$, v_h^k converges to v_h uniformly in $\overline{\Omega}$ for any $h > 0$ fixed.

Proof. First note that condition (14) is a sufficient condition for (13) and for the *consistency* of the triangulation. Therefore we can apply Theorem 3.3 and we easily conclude. □

Remark 3.5. The result in the above corollary does not rely on the fact that we use a split dynamic $f = (f_P, f_E)$ but we notice that it is not clear how to get the Hamilton-Jacobi-Isaacs equation associated to the problem in the case of a general dynamics with state constraints. In [21] there is an attempt in this direction but unfortunately the case considered there does not include Pursuit-Evasion games.

In order to obtain uniform convergence of v_h^k to the solution of the continuous problem when h and k tend to 0^+, we couple our result with those in [8] which are restricted to Pursuit-Evasion games. Let us denote by v the value function for the continuous problem as defined in [8]. In order to provide sufficient conditions for the continuity of v, we need to introduce further assumptions. Whenever we say that $\omega : [0, +\infty) \to [0, +\infty)$ is a *modulus* we mean that ω is nondecreasing, it is continuous at zero, and $\omega(0) = 0$. The first assumption is about the behavior of the value function v near the target $\mathcal{T}$.

$$\text{There is a modulus } \omega \text{ such that } v(x) \leq \omega(d(x, \mathcal{T})) \quad \text{for all } x \in \overline{\Omega} \tag{C1}$$

where $d(x, \mathcal{T}) = \inf_{z \in \mathcal{T}}\{|x - z|\}$.
The second is a small time controllability assumption for the Pursuer.

$$\begin{cases} \text{There is } \omega_P(\cdot, R) \text{ modulus for all } R > 0 \text{ such that for all} \\ w_1, w_2 \in \overline{\Omega}_1 \text{ there are } a(\cdot) \in \mathcal{A}_{w_1} \text{ and a time } t_{w_1,w_2} \text{ satisfying} \\ y_P(t_{w_1,w_2}; w_1, a(\cdot)) = w_2 \text{ and } 0 \le t_{w_1,w_2} \le \omega_P(|w_1 - w_2|, |w_2|). \end{cases} \tag{C2}$$

The third is a small time controllability assumption for the Evader.

$$\begin{cases} \text{There is } \omega_E(\cdot, R) \text{ modulus for all } R > 0 \text{ such that for all} \\ z_1, z_2 \in \overline{\Omega}_2 \text{ there are } b(\cdot) \in \mathcal{B}_{z_1} \text{ and a time } t_{z_1,z_2} \text{ satisfying} \\ y_E(t_{z_1,z_2}; z_1, b(\cdot)) = z_2 \text{ and } 0 \le t_{z_1,z_2} \le \omega_E(|z_1 - z_2|, |z_2|). \end{cases} \tag{C3}$$

The proof of the next theorem can be found in [8].

Theorem 3.6. Assume that (2), (C1), (C2), and (C3) hold. Then, the value function v is continuous in $\overline{\Omega}_1 \times \overline{\Omega}_2$.

Let us introduce the following regularity hypothesis on the boundary of $\mathcal{T}$.

$$\begin{cases} \text{For each } x \in \partial\mathcal{T} \text{ there are } r, \theta > 0 \text{ and } \Xi \in \mathbb{R}^{2n} \text{ such that} \\ \bigcup_{0<t<r} B(x' + t\Xi, t\theta) \subset \Omega\backslash\mathcal{T} \text{ for any } x' \in B(x, r) \cap \overline{\Omega\backslash\mathcal{T}}. \end{cases} \tag{15}$$

Note that a comparison principle for sub- and super-solutions for the same Hamiltonian is proved in [8] under additional regularity assumptions on $\partial(\Omega\backslash\mathcal{T})$. More precisely, the assumptions needed are the uniform interior cone conditions for Ω_1, Ω_2, and $\mathcal{T}$.
We have the following

Theorem 3.7. Let Ω be an open-bounded set. Let (2), (7), (C1), (C2), (C3), and (15) hold true. Finally, assume that

$$f_P(x_P, A(x)) \quad \text{and} \quad f_E(x_E, B(x)) \quad \text{are convex sets.} \tag{16}$$

Then, for $h \to 0^+$, v_h converges to v uniformly in $\overline{\Omega}$.

Proof. The assumption (16) guarantees that the value function v_h for the time-discrete problem defined in (10) coincides with that used in [8]. Moreover, assumptions of Theorem 3.6 are fulfilled so that $v \in C(\overline{\Omega})$. Then, the proof follows by Theorem 4.2 in [8]. □

Coupling the previous results we can prove our convergence result for the approximation of Pursuit-Evasion games.

Corollary 3.8. Let the assumptions of Corollary 3.4 and Theorem 3.7 hold true. Moreover, assume that $k = O(h^{1+\alpha})$, for $\alpha > 0$. Then v_h^k converges to v uniformly in $\overline{\Omega}$ for h tends to 0^+.

Proof. Since $1-e^{-h}=O(h)$ for h tending to 0^+ we have for any $x\in\Omega$:

$$|v_h^k(x)-v(x)|\leq|v_h^k(x)-v_h(x)|+|v_h(x)-v(x)|\leq O(h^\alpha)+\|v_h(x)-v(x)\|_\infty.$$

□

As we said in the Introduction, a convergence theorem has been proved in [11,12] for a different approximation scheme based on viability theory. The approach is different in several respects. The first is that the techniques used in the proof are based on the characterization of the epigraph of the value function of the game in terms of a *Discriminating Kernel* for a suitable problem. By this technique the authors can easily deal with semicontinuous Hamiltonians and construct a discrete Discriminating Kernel algorithm. This technique is based on an external approximation of the epigraph of the value function via a sequence of closed sets D_p, $p\in\mathbb{N}$ (see [13] p. 224). This construction is rather expensive for games and can be hard to pursue particularly in high-dimension even if one can try to localize the algorithm near the boundary of the Discriminating Kernel (*i.e.,* nearby the graph of the value function).

4 The Tag-Chase game with state constraints

The Tag-Chase game is a particular case of Pursuit-Evasion games. We consider two boys, P and E, which run one after the other in the same 2-dimensional domain, so that the game is set in $\overline{\Omega}=\overline{\Omega}_1^2\subset\mathbb{R}^4$ where Ω_1 is an open-bounded set of $\mathbb{R}^2$. We denote by (x_P,x_E) the coordinates of $\overline{\Omega}$ where $x_P,\ x_E\in\overline{\Omega}_1$. P (respectively, E) can run in every direction with velocity V_P (respectively, V_E) so that the dynamics of the game is

$$\begin{cases}\dot{x}_P=V_P\,a & a\in B_2(0,1)\\ \dot{x}_E=V_E\,b & b\in B_2(0,1)\,,\end{cases}$$

where $B_2(0,1)=\{z\in\mathbb{R}^2:|z|\leq 1\}$. The case $V_P>V_E$ is completely studied in [1,2]. The value function $T=-\ln(1-v)$, which represents the capture time, is continuous and bounded in its domain of definition. Moreover, the convergence result we obtained in Sec. 3 applies to this case.

On the other hand, the most interesting case is certainly $V_P=V_E$, *i.e.,* when the players have the same dynamics and no advantage is given to any of them. In this case it is easily seen that the value function T is discontinuous (at least on $\partial\mathcal{T}$) and then all theoretical results based on the continuity of the value function does not hold.

In this section we will give an answer to the following question: "if $V_P=V_E$, is the capture time finite?".

If the Tag-Chase game is played without constraints on the state and both players play optimally, it is immediately seen that the distance between P and E remains constant and then capture never happens (the optimal strategy for E is to move for ever as fast as he can along the line joining P and E in the opposite direction with respect to the position of P). On the contrary, if the state is constrained in a bounded domain, such a restriction seems to play a key role against the Evader, as the following proposition shows.

Proposition 4.1. Let Ω_1 be open and bounded. Moreover, let the target be

$$\mathcal{T} = \{(x_P, x_E) \in \overline{\Omega} : \ |x_P - x_E| \leq \varepsilon\}, \quad \varepsilon \geq 0. \tag{17}$$

Then,

1. If $V_P > V_E$, then the capture time $t_c = T(x_P, x_E)$=$-\ln(1 - v(x_P, x_E))$ is finite and bounded by
$$t_c \leq \frac{|x_P - x_E|}{V_P - V_E}.$$
2. If $V_P = V_E$, $\varepsilon \neq 0$ and Ω_1 is convex then the capture time t_c is finite.

Proof.

1. This first part of the proof can be found in [1]. We fix a strategy for P and leave E free to decide his optimal strategy. First, P reaches the starting point of E covering the distance $|x_P - x_E|$ and then he follows the E's trace. The conclusion follows by elementary computations.
2. The basic idea of the proof is the same of the previous case but we have to change the strategy for the Pursuer in order to have a finite upper bound. P runs after E always along the line joining P and E (P can do it by the convexity of Ω_1) while E chooses his own optimal trajectory as before. We can characterize the strategy of E by a function $\theta(t) : [0, +\infty) \to [0, 2\pi)$ which represents at every time the smallest angle between the velocity vector of E and the line joining P and E (see Fig. 2). Let us denote by $d_{PE}(t)$ the

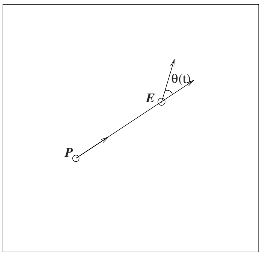

Figure 2: Trajectories of P and E in Proof of Proposition 4.1.

distance between P and E at time t. We claim that, for any fixed t,

$$\theta(t) \neq 0 \Rightarrow d'_{PE}(t) < 0 \tag{18}$$

where $' = \frac{d}{dt}$. Due to the state constraints, $\theta(t)$ cannot be equal to 0 for a time interval longer then $diag(\Omega_1)/V_P$ and after that must be different from 0 for a finite time interval because E must change his trajectory at least when he touches $\partial\Omega_1$. Therefore, if (18) holds then $d_{PE}(t) \to 0$ for $t \to \infty$ and then for any $\varepsilon > 0$ there exists a time $\bar{t}$ such that $d_{PE}(\bar{t}) \leq \varepsilon$ (the capture occurs). In order to prove (18), let us define the two vectors $E(t)$ and $P(t)$ which are, respectively, the position of P and E at time t and the vector $r(t) := E(t) - P(t)$. By definition, we have $d_{PE}(t) = |r(t)|$. Without loss of generality, suppose that at time t, $P(t)$ is in the origin and $E(t)$ lies on the x-axis and

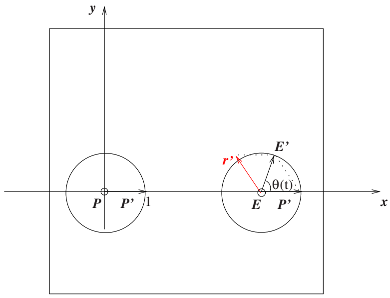

Figure 3: Vectors P, P', E, E', and r' as in Proposition 4.1.

that $V_P = V_E = 1$ (see Fig. 3).
Then

$$P'(t) = (1, 0) \quad \text{and} \quad \frac{r(t)}{|r(t)|} = (1, 0).$$

Moreover, by construction we have

$$E'(t) = (\cos\theta(t), \sin\theta(t)) \quad \text{and} \quad r'(t) = E'(t) - P'(t).$$

It follows that $r'(t) = (\cos\theta(t) - 1, \sin\theta(t))$ and

$$d'_{PE}(t) = \frac{r(t)}{|r(t)|} \cdot r'(t) = \cos\theta(t) - 1 \tag{19}$$

so that (18) holds. □

5 Some hints for the algorithm

In this section we give some hints for an efficient implementation of the algorithm for the solution of differential games. The main goal is to reduce the computational cost since this is a crucial step toward applications. The first hint deals with a fast way to compute the term $v_h^k(x_i + hf(x_i, a, b))$ in high-dimensional spaces.

In fact, the linear interpolation used in the definition of the fixed point iteration is appealing from the theoretical point of view but not very efficient since it would require the solution of a linear system of size $2n+1$ for every x_i, a, and b. The procedure we suggest solves this problem by a sequence of linear interpolations in 1-dimension. The second hint exploits the natural symmetry in the game problem whenever it is played in a square domain in order to reduce (by a factor 2 in two dimensions and 4 in four dimensions) the domain where the solution is actually computed. Both procedures have shown to be very efficient and have contributed to a dramatic reduction of the CPU time.

5.1 Interpolation in high-dimensions

It is important to note that the semi-Lagrangian scheme (HJI$_h^k$–Ω) requires that at every iteration, at every node and every a and b, the value $v_h^k(x_i + hf(x_i, a, b))$ is computed and this computation needs an interpolation of the values of v_h^k at the nodes. [14] extensively analyzed a fast and efficient interpolation method in high-dimension suitable to our purposes. We recall briefly this method giving a precise error estimate.

Consider a point $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ and the cell of the grid which contains it (see Fig. 4 for an example in 3D). Suppose that a function f is known in the 2^n vertexes of the cell and we want to compute the value $f(x)$ by linear interpolation. The basic idea is to project the point x onto lower and lower dimensional subspaces until dimension 1 is reached. More precisely, choose a dimension (in Fig. 4 we chose x_1) and project the point x in that dimension on both sides of the cell finding the points P_1^1 and P_2^1. Then, choose a direction different from the first one (we chose x_2) and project the points P_1^1 and P_2^1 on the sides of the cell finding the points P_1^2, P_2^2, P_3^2, and P_4^2. Iterate the projection procedure $2^{n+1} - 2$ times in the same way until all vertexes of the cell are reached. At this stage a tree structure containing all points P_i^j , $i = 1, \ldots, 2^n$, $j = 1, \ldots, n$ is computed from top to bottom. Now evaluate by *unidimensional* linear interpolations the values of f at the points P_i^j , $i = 1, \ldots, 2^n$, $\quad j = 1, \ldots, n$ in the reverse order with respect to that used to find them (from bottom to top). This procedure leads to an approximate value of $f(x)$ obtained by $2^n - 1$ unidimensional linear interpolations. It is interesting to give a precise error estimates of this first-order interpolation method.

Theorem 5.1. Let $\mathbb{R}^n \supset Q := [a_1, b_1] \times \ldots \times [a_n, b_n]$ and $x = (x_1, \ldots, x_n)$. Assume $f \in C^2(Q;\mathbb{R})$ and let $q(x)$, $x \in Q$ be the approximate value of $f(x)$ obtained by the n-dimensional linear interpolation described above.
Then, the error $E(x) := f(x) - q(x)$ is bounded by

$$|E(x)| \leq \sum_{i=1}^{n} \frac{\Delta_i^2}{8} M_i \,, \qquad \text{for all } x \in Q \,,$$

where $M_i = \max\limits_{x \in Q} |\frac{\partial^2 f(x)}{\partial x_i^2}|$ and $\Delta_i = b_i - a_i$.

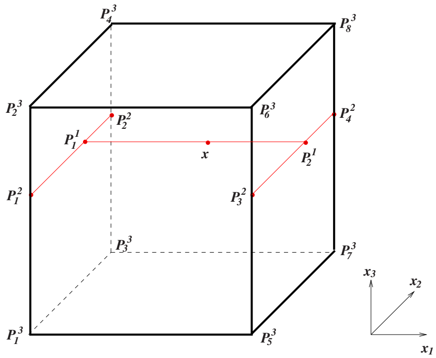

Figure 4: Example in 3D.

Proof. The proof is easily obtained by induction using the basic theory of linear interpolation. The interested reader can find the complete proof in [15]. □

5.2 Reducing the size of the problem

If the dimension of the space is greater than 4, the algorithm has a high computational cost. As already noted in [2,22], due to the state constraints it is not possible in general to use reduced coordinates $\tilde{x} = x_P - x_E$ unless the problem has a special structure. In fact, using reduced coordinates we loose every information about the real positions of the two players, so that we cannot detect when they touch the boundary of the domain (and then change the dynamics consequently). Note that if we consider the 2-player Tag-Chase game constrained in a circle (see for example [9]), the problem can be described by three coordinates instead of four since the problem is invariant with respect to the rotation of the domain. Obviously, this is not true if the game's field is a square as in the case of the numerical tests we carry on in this paper.

Although it is not always possible to describe the game in a reduced space due to the state constraints, we can simplify the computation taking into account the symmetries of the problem, if any. We explain our technique first in the 1-dimensional case and then in the 2-dimensional case. From now on we denote by n the number of grid nodes in each dimension.

Unidimensional Tag-Chase game
Assume that each player can move along a line in the interval $[-x_0, x_0]^2$, then the game is set in the square $[-x_0, x_0]^2$.

In Fig. 5 we show the level sets of the solution $T = -\log(1 - v)$ in the case $V_P = 2$, $V_E = 1$, $x_0 = 2$ and an optimal trajectory starting from $(-1.5, 0)$. It is

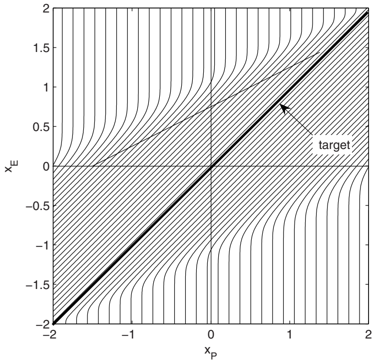

Figure 5: Level sets of the solution $T = -\log(1-v)$ for $V_P = 2$, $V_E = 1$.

easy to see that

$$v(x_P, x_E) = v(-x_P, -x_E) \quad \text{for all } x_P, x_E \in [-x_0, x_0]$$

so that we can recover the entire solution either from the triangular sector $S_{NW} = \{(x_P, x_E) : x_P \le x_E\}$ or from the rectangular sector $S_W = \{(x_P, x_E) : x_P \le 0\}$. This corresponds to the fact that it is sufficient to compute the solution for all the initial positions of P and E in which P is on the left of E or P is in the left side of the domain $[-x_0, x_0]$ (see Fig. 6). There is an important difference

Figure 6: Two initial positions which correspond to the same value for v.

between the two approaches. In fact, the target $\mathcal{T} = \{(x_P, x_E) : x_P = x_E\}$ is entirely contained in S_{NW} but not in S_W. Moreover, since the target divides the domain $\overline{\Omega} = [-x_0, x_0]^2$ in two parts and no characteristics can pass from one part to the other, all the optimal trajectories starting from S_{NW} remains in S_{NW}. This is clearly not true for S_W. As a consequence, if we compute the solution only in S_W this will be not correct because not all the usable part of the target is visible from the domain.

Unfortunately, the domain S_{NW} has not a correspondence in the two-dimensional Tag-Chase game. In fact, the target $\mathcal{T}$ does not divide the entire space $\overline{\Omega} = ([-x_0, x_0] \times [-x_0, x_0])^2$ into two parts since the co-dimension of the target is strictly greater than 1. On the contrary, we will see that the domain S_W has a natural generalization in the two-dimensional case.

For this reason it is preferable to localize the computation only in S_W. In order to do this we adopt the following idea. First of all we choose n even. Then we compute the approximation of v at the node corresponding to the indices (i, j), for $i = 1, \ldots, n/2, j = 1, \ldots, n$ via the numerical scheme (HJI$_h^k$–Ω) (note that now i is the index corresponding to the position of the player P so is a column index whereas j is a row index). After every iteration we copy the line $(i = n/2,\ j = 1 : n)$ in $(i = n/2 + 1,\ j = n : 1)$ as a sort of "periodic boundary condition" for S_W. In this way the information coming from the south-western part of the target can substitute the missing information needed by the north-western part of the domain.

When the algorithm reached the convergence we can easily recover the solution on all over the domain Ω.

Two-dimensional Tag-Chase game

As we did in the unidimensional case, we want to use the symmetries of the problem to avoid useless computation.

We assume that each player can move in a square so that the game is set in a four-dimensional hypercube. The positions of P and E will be denoted, respectively, by (x_P, y_P) and (x_E, y_E). In this case we have more than one symmetry. In fact, it easy to check that the following three inequalities hold (see Fig. 7):

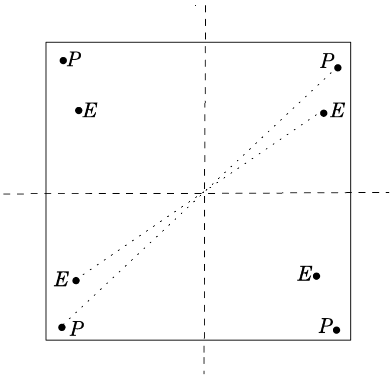

Figure 7: Four initial positions which correspond to the same value for v.

$$v(x_P, y_P, x_E, y_E) = v(-x_P, -y_P, -x_E, -y_E) \tag{20}$$

$$v(x_P, y_P, x_E, y_E) = v(-x_P, y_P, -x_E, y_E) \tag{21}$$

$$v(x_P, y_P, x_E, y_E) = v(x_P, -y_P, x_E, -y_E). \tag{22}$$

We note that once we take into account the symmetry (20) we took into account automatically the symmetry in (21).

The following nested `for`'s take into account only the symmetry (22) and they allow to compute correctly the whole 4D matrix containing the grid nodes.

```
for i=1:n
  for j=1:n/2
    for k=1:n
      for l=1:n
        {v(i,j,k,l)=SLscheme(...);
        v(i,n-j+1,k,n-l+1)=v(i,j,k,l);}
```

Now we are ready to make use of symmetry (20) by means of the technique introduce for the unidimensional Tag-Chase game. We compute just half matrix corresponding to the indexes $i = 1,\ldots,n/2$ and after every iteration we copy the submatrix $(i = n/2,\ j = 1 : n,\ k = 1 : n,\ l = 1 : n)$ in $(i = n/2+1,\ j = n : 1,\ k = n : 1,\ l = n : 1)$ as a boundary condition.

At the end of computation we easily recover the solution in the whole domain.

Remark 5.2. We ran a Fast Sweeping [23] version of the one-dimensional Tag-Chase game. We noticed that no improvements about the number of iterations is achieved. This is probably due to the presence of state constraints so that the information first propagates from the target and then it comes back after hitting the boundary. A Fast Marching scheme for the unconstrained game in reduced coordinates has been presented in [16].

6 Numerical experiments

In this section we present some numerical experiments for two-dimensional constrained Tag-Chase game. We consider the case $V_P > V_E$ as well as $V_P = V_E$ and $V_P < V_E$. To our knowledge, these two last cases appear for the first time in a numerical test. The code is written in C++ and its parallel version has been obtained by means of OpenMP directives. The algorithm ran on a PC equipped with a processor Intel Pentium dual core 2×2.80 GHz, 1 GB RAM and on an IBM system p5 575 equipped with 8 processors Power5 at 1.9 GHz and 32 GB RAM located at CASPUR[1].

Notations and choice of parameters

We denote by n the number of nodes in each dimension. We denote by n_c the number of admissible directions/controls for each player, *i.e.,* we discretize the unit ball $B(0,1)$ with n_c points. We restrict the discretization to the boundary $\partial B(0,1)$ and in some cases we add the central point (in this case we denote the number of directions by n_c^-+1 where $n_c^- = n_c - 1$).

[1]Consorzio interuniversitario per le Applicazioni di Supercalcolo per Università e Ricerca, `www.caspur.it`.

We always use a uniform structured grid with four-dimensional cells of volume Δx^4 and we choose the (fictitious) time step h such that $\|hf(x,a,b)\| \leq \Delta x$ for all x, a, b (so that the interpolation is made in the neighboring cells of the considered point).

We introduce the following stopping criterion for the fixed point iteration $V^{p+1} = F(V^p)$ (where $V_i = v_h^k(x_i)$)

$$\|V^{(p+1)} - V^{(p)}\|_\infty \leq \varepsilon\,, \qquad \varepsilon > 0.$$

We remark that the quality of the approximate solution depends on h, k and also (strictly) on the ratio h/k (see [4])

The real game is played in a square $[-2,2]^2$ so the problem is set in $\overline{\Omega} = [-2,2]^4$. The numerical target is $\mathcal{T} = \{(i,j,k,l) \in \{1,\ldots,n\}^4 : |i-k| \leq 1$ and $|j-l| \leq 1\}$.

Once we computed the approximate solution, we recover the optimal trajectories. At this stage we have to choose the time step Δt in order to discretize the dynamical system by Euler scheme. It should be noted that this parameter can, in general, be different from the (fictitious) time step h chosen for the computation of the value function (our choice is $\Delta t = \Delta x/2$) and this is true also for the number of controls n_c. Moreover, computing optimal trajectory requires the evaluation of the argminmax which is done again choosing a value for h, and this value can be in principle different from that used in the first computation.

We plot some flags (circles for the Pursuer, squares for the Evader) on the approximate optimal trajectories every s time steps where s varies from 5 to 20 depending on the test. This allows to follow the position of one player with respect to the other during the game.

We denote by $v(x_P, y_P, x_E, y_E)$ the approximate value function and by $T(x_P, y_P, x_E, y_E) = -\ln(1 - v(x_P, y_P, x_E, y_E))$ the time of capture.

6.1 Case $V_P > V_E$

The case $V_P > V_E$ is the classical one and it was already studied by Alziary de Roquefort [2]. In this case, the value function v is continuous and all theoretical results we presented in this paper hold true. In the following we name “CPU time” the sum of the times taken by the CPUs and by “wallclock time" the elapsed time.

Test 1

We choose $\varepsilon = 10^{-3}$, $V_P = 2$, $V_E = 1$, $n = 50$, $n_c = 48 + 1$. Convergence was reached in 85 iterations. The CPU time (IBM - 8 procs) was 17h 36m 16s, the wallclock time was 2h 47m 37s. Figure 8 shows the value function $T(0, 0, x_E, y_E)$ and its level sets (we fix the Pursuer’s position at the origin). It is immediately seen that if the distance between P and E is greater than $V_P - V_E = 1$ then the state constraints have a great influence on the solution. Moreover, it is clear that

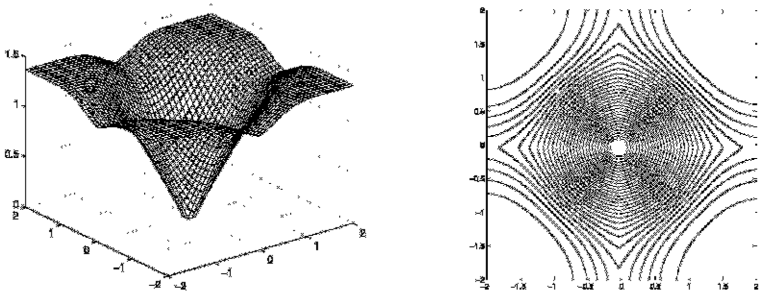

Figure 8: Test 1. Value function $T(0, 0, x_E, y_E)$ (left) and its level sets (right).

the presence of state constraints gives an advantage to the Pursuer.
Figure 9 shows four optimal trajectories corresponding to the starting points:

$$\begin{cases} P = (-1, 0) \\ E = (0, 0) \end{cases} \begin{cases} P = (-2, -2) \\ E = (1, 0.7) \end{cases} \begin{cases} P = (-1.8, -1.8) \\ E = (0.5, -1.6) \end{cases} \begin{cases} P = (-1.8, -1.8) \\ E = (0.5, -1.8). \end{cases}$$

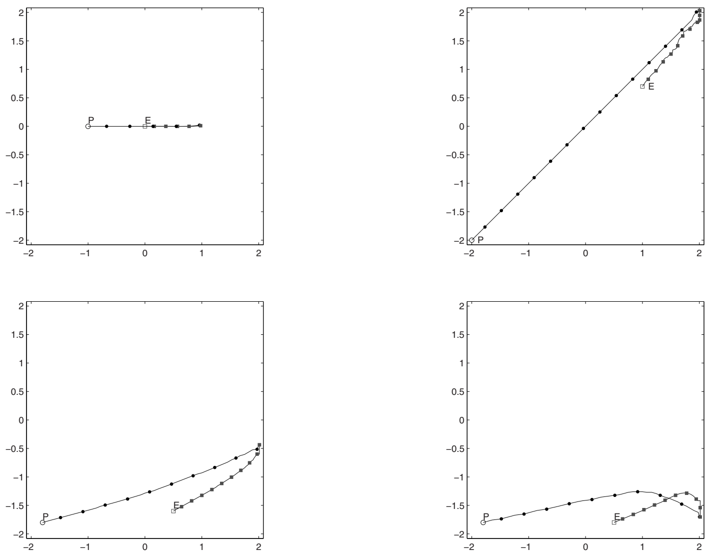

Figure 9: Optimal trajectories for Test 1.

Test 2
The second test is just to compare the CPU time corresponding to the two architectures mentioned above. It is interesting to test the new dual core processors in

order to understand how much they can be useful in parallel scientific computing. They are indeed conceived mainly to deal with distributed computing or simply multitasking. The performances of the parallel code are measured in terms of two well-known parameters, the *speed-up* and the *efficiency*. Let T_{ser} and T_{par} be the times corresponding, respectively, to the execution of the serial and parallel algorithms over n_p processors. We define

$$\textit{speed-up} := \frac{T_{ser}}{T_{par}} \qquad \text{and} \qquad \textit{efficiency} := \frac{\textit{speed-up}}{n_p}.$$

Note that an ideal parallel algorithm would have *speed-up* $= n_p$ and *efficiency* $= 1$. Table 1 shows the wallclock time, the *speed-up* and the *efficiency* for the following

Table 1: CPU time for Test 2

architecture	wallclock time	speed–up	efficiency
IBM serial	26m 47s	-	-
IBM 2 procs	14m 19s	1.87	0.93
IBM 4 procs	8m 09s	3.29	0.82
IBM 8 procs	4m 09s	6.45	0.81
PC dual core, serial	1h 08m 44s	-	-
PC dual core, parallel	34m 51s	1.97	0.99

choice of parameters: $\varepsilon = 10^{-5}$, $V_P = 2$, $V_E = 1$, $n = 26$, $n_c = 36 + 1$.

Test 3

In this test the domain has a square hole in the center. The side of the square is 1.06. We choose $\varepsilon = 10^{-4}$, $V_P = 2$, $V_E = 1$, $n = 50$, $n_c = 48 + 1$. Convergence was reached in 109 iterations. The CPU time (IBM - 8 procs) was 1d 00h 34m 18s, the wallclock time was 3h 54m 30s. Figure 10 shows the value function

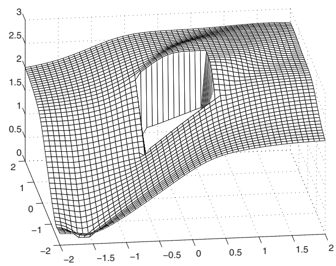

Figure 10: Test 3. Value function $T(-1.5, -1.5, x_E, y_E)$.

$T(-1.5, -1.5, x_E, y_E)$.
Figure 11 shows two optimal trajectories corresponding to the starting points:

$$\begin{cases} P = (-1.9, -1.9) \\ E = (1.9, 1.9) \end{cases} \qquad \begin{cases} P = (-1.9, 0) \\ E = (1, 0). \end{cases}$$

It is interesting to note that in both cases the Evader waits until the Pursuer decides

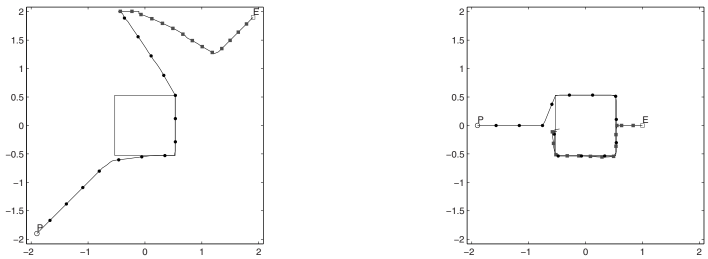

Figure 11: Optimal trajectories for Test 3.

if he wants to skirt around the obstacle clockwise or counterclockwise. After that, the Evader goes in the opposite direction. If both players touch the obstacle, they run around it until the capture occurs.

Test 4
In this test the domain has a circular hole in the center. The radius r of the circle is $7\Delta x$. We choose $\varepsilon = 10^{-4}$, $V_P = 2$, $V_E = 1$, $n = 50$, $n_c = 48 + 1$. Convergence was reached in 108 iterations. The CPU time (IBM - 8 procs) was 1d 17h 27m 43s, the wallclock time was 6h 39m 00s. Note that handling with a circular obstacle inside the domain of computation is not easy as in the previous test where the boundary of the obstacle matches with the lattice. We adopt the following procedure. First of all, we define the radius r of the circle as a multiple of the space step Δx. Then, at every node $(P{=}(i,j), E{=}(k,l))$, we compute the distance d_{PO} (resp., d_{EO}) between P (resp., E) and the center of the domain. Let us focus on P, E being treated in the same way. If $r \le d_{PO} < r + \Delta x$, then we say that P in on the "numerical boundary" of the circle. The exterior normal vector $\eta(i,j)$ to the (numerical) boundary of the circle is simply given by the coordinates of the node (i,j), so that we can easily compute the scalar product $\eta \cdot a$ where a is the desired direction of P. If the scalar product is negative, we label the direction a as *not admissible*.

Figure 12 shows two optimal trajectories corresponding to the starting points:

$$\begin{cases} P = (-1.9, -1.9) \\ E = (1.9, 1.9) \end{cases} \qquad \begin{cases} P = (-0.6, 0) \\ E = (1, 0.4). \end{cases}$$

The behavior of the optimal trajectories is similar to the previous Test.

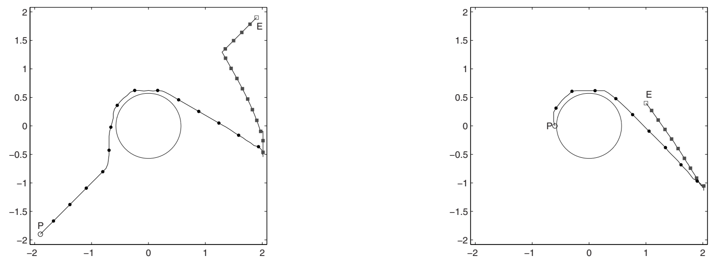

Figure 12: Optimal trajectories for Test 4.

6.2 Case $V_P = V_E$

When $V_P = V_E$ the value function v is discontinuous on $\partial\mathcal{T}$. In this case no convergence results are known, nevertheless the numerical scheme seems to work very well. We remember that results in Sec. 4 guarantee that $v < 1$ (the capture always occurs). This is confirmed by the following test.

Test 5
We choose $\varepsilon = 10^{-3}$, $V_P = 1$, $V_E = 1$, $n = 50$, $n_c = 36$. Convergence was reached in 66 iterations. Figure 13 shows the value function $T(0, 0, x_E, y_E)$ and its level sets. Figure 14 shows the value function $T(1.15, 1.15, x_E, y_E)$ and its

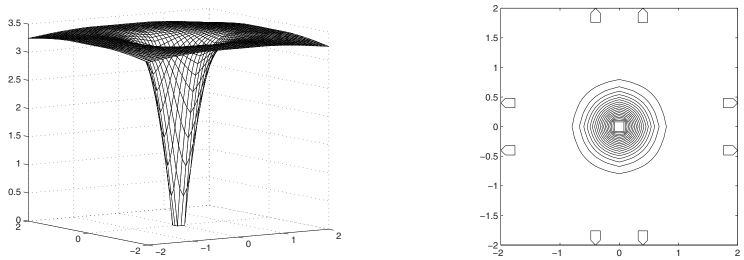

Figure 13: Test 5. Value function $T(0, 0, x_E, y_E)$ (left) and its level sets (right).

level sets. Figure 15 shows four optimal trajectories corresponding to the starting points:

$$\begin{cases} P = (0, 1) \\ E = (0, 0) \end{cases} \quad \begin{cases} P = (1, 1.5) \\ E = (-0.5, 0) \end{cases} \quad \begin{cases} P = (1.3, 1.8) \\ E = (0, 0) \end{cases} \quad \begin{cases} P = (-1.9, -1.9) \\ E = (-1.7, -1.9). \end{cases}$$

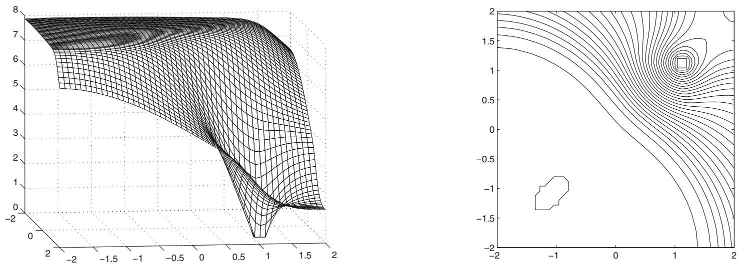

Figure 14: Test 5. Value function $T(1.15, 1.15, x_E, y_E)$ (left) and its level sets (right).

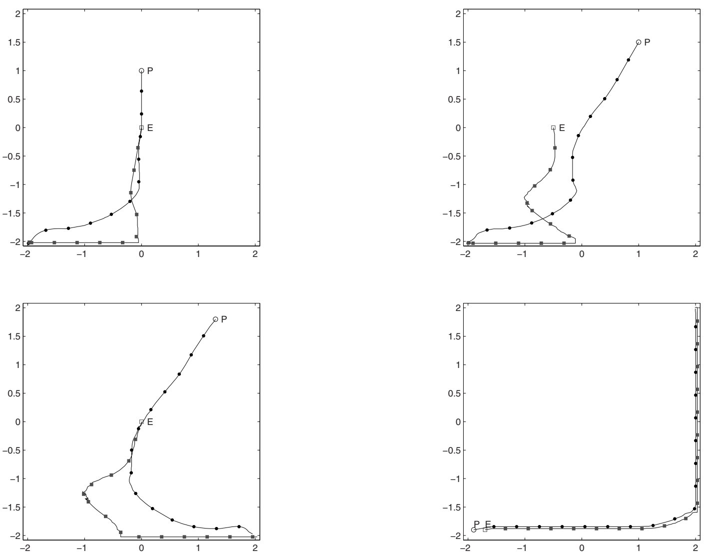

Figure 15: Optimal trajectories for Test 5.

Test 6

In this test the domain has a circular hole in the center. The radius of the circle is $7\Delta x$. Since the domain is no more convex, we have no guarantee that the time of capture is finite. Numerical results show that the value function v is equal to 1 in a large part of the domain.

It is well known that it is not possible to recover the optimal trajectories whenever $v = 1$ $(T = \infty)$ since from that regions capture never happens. Indeed, if $V_P \leq V_E$ the approximate solution shows a strange behavior. Even if $v < 1$, in some cases the computed optimal trajectories tend to stable trajectories such that P never reaches E. Although this is due to some numerical error, these trajectories are extremely realistic so they give to us a guess about the optimal strategies of the players in the case E wins. In this Test (and others below) we show this behavior.

We choose $\varepsilon = 10^{-4}$, $V_P = 1$, $V_E = 1$, $n = 50$, $n_c = 48 + 1$. Convergence was reached in 94 iterations. The CPU time (IBM - 8 procs) was 1d 12h 05m 22s. Figure 16 shows one optimal trajectory corresponding to the starting point

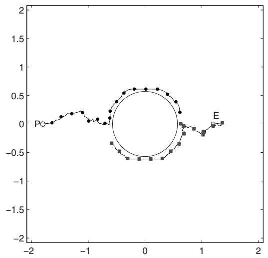

Figure 16: Optimal trajectories for Test 6.

$$\begin{cases} P = (-1.8, 0) \\ E = (1.2, 0). \end{cases}$$

In this example, there is no capture within 150 time steps. The asymptotic behavior of the trajectory is stable since once the two players reached the internal circle, they run around it forever. It should be noted that, at the beginning of the game, E leaves the time to go by in order to touch the boundary of the circle exactly when P touches it.

This strange behavior urges us to invent some method to compute rigorously the trajectories corresponding to the E's win, in order to confirm our guess. Maybe we can do it considering the time-dependent problem (so that we work in $\mathbb{R}^5$ as Alziary de Roquefort does [2]). This allows one to choose a time-dependent velocity $V_E(t)$ such that it is very fast for $0 \le t < \bar{t}$ (capture impossible) and very slow for $t > \bar{t}$ (capture unavoidable). For such a velocity we have $v < 1$ so we can compute optimal trajectories but, for $0 \le t < \bar{t}$, E will attempt to maintain a trajectory such that capture does not occur.

6.3 Case $V_P < V_E$

If $V_P < V_E$ the value function v is discontinuous on $\partial\mathcal{T}$. Moreover, we have no guarantee that the time of capture is finite. Numerical results show that the value function v is equal to 1 in a large part of the domain.

Test 7
We choose $\varepsilon = 10^{-3}$, $V_P = 1$, $V_E = 1.25$, $n = 50$, $n_c = 48 + 1$. Convergence was reached in 53 iterations. The CPU time (IBM - 8 procs) was 12h 43m 02s, the wallclock time was 2h 18h 06s.
Figure 17 shows the value function $T(-1, -1, x_E, y_E)$ and its level sets.

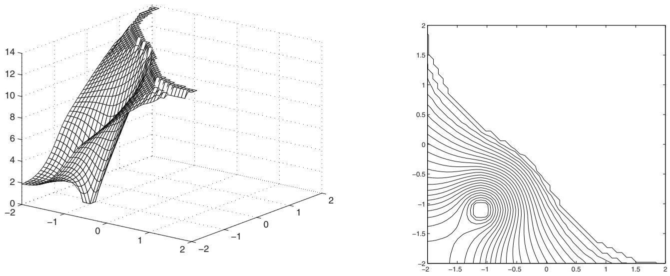

Figure 17: Test 7. Value function $T(-1, -1, x_E, y_E)$ (left) and its level sets (right).

Figure 18 shows two optimal trajectories corresponding to the starting points

$$\begin{cases} P = (-1, -1) \\ E = (-1, 1) \end{cases} \qquad \begin{cases} P = (-1, -1) \\ E = (-0.5, -0.5). \end{cases}$$

Note that the Pursuer approaches the corner in which capture occurs along the

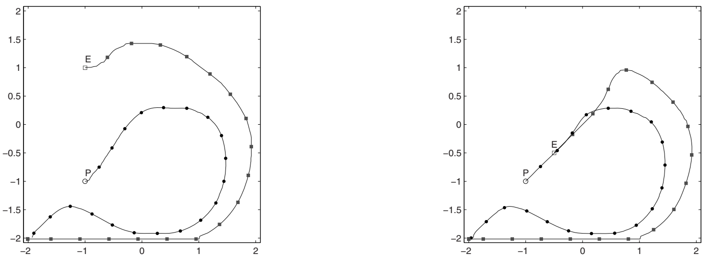

Figure 18: Optimal trajectories for Test 7.

diagonal of the square in order to block off the Evader's escape.

Test 8
We choose $\varepsilon = 10^{-4}$, $V_P = 1$, $V_E = 1.5$, $n = 50$, $n_c = 36$. Convergence was

reached in 65 iterations. The CPU time (IBM - 8 procs) was 15h 48m 46s, the wallclock time was 2h 30m 19s.
Figure 19 shows two optimal trajectories corresponding to the starting points:

$$\begin{cases} P = (0.5, 0.5) \\ E = (1.5, 1.5) \end{cases} \qquad \begin{cases} P = (0, -0.8) \\ E = (-0.3, -1.3). \end{cases}$$

In the example on the left, E makes believe he wants to be caught in the upper-left

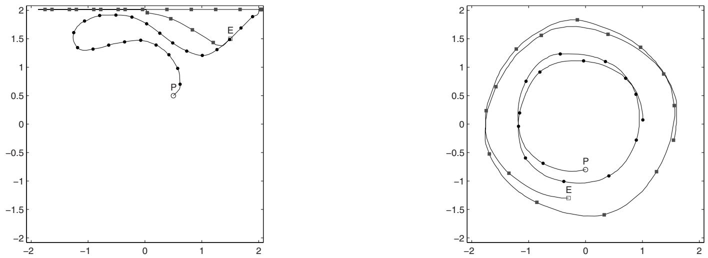

Figure 19: Optimal trajectories for Test 8.

corner but after a while he turns on the right toward the upper-right corner. In the example on the right, there is no capture within 2,000 time steps (see Test 6) and the asymptotic behavior of the trajectories is quite stable. Moreover, we note that the ratio between the two radii of the circles are about 1.5 as the ratio between the velocities of the two players (so that they complete a rotation in the same time).

Acknowledgements

The authors wish to thank D. Graziani, S. Cacace, and E. Cotognini for helpful discussions on the Tag-Chase game.

This work has been partially supported by funds from the MIUR Project 2005 “Modellistica Numerica per il Calcolo Scientifico ed Applicazioni Avanzate." We gratefully acknowledge the technical support given by CASPUR Consortium to the development of this research.

References

[1] Alziary de Roquefort B., *Jeux différentiels et approximation numérique de fonctions valeur,* 1^{re} *partie: étude théorique*, RAIRO Modél. Math. Anal. Numér., **25** (1991), 517–533.

[2] Alziary de Roquefort B., *Jeux différentiels et approximation numérique de fonctions valeur, 2^e partie: étude numérique*, RAIRO Modél. Math. Anal. Numér. **25** (1991), 535–560.

[3] Bardi M., *Some applications of viscosity solutions to optimal control and differential games*, in I. Capuzzo Dolcetta, P.-L. Lions (eds.), "Viscosity solutions and applications (Montecatini Terme, 1995)", Lecture Notes in Mathematics **1660**, 44–97, Springer, Berlin, 1997.

[4] Bardi M., Bottacin S., Falcone M., *Convergence of Discrete Schemes for Discontinuous Value Functions of Pursuit-Evasion Games*, in G. J. Olsder (ed.), "New trends in Dynamic Games and Application", Annals of the International Society of Dynamic Games **3**, 273–304, Birkhäuser, Boston, 1995.

[5] Bardi M., Capuzzo Dolcetta I., *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations*, Birkhäuser, Boston, 1997.

[6] Bardi M., Falcone M., Soravia P., *Fully discrete schemes for the value function of pursuit-evasion games*, in T. Başar and A. Haurie (eds.), "Advances in Dynamic Games and Application", Annals of the International Society of Dynamic Games **1**, 89–105, Birkhäuser, Boston, 1994.

[7] Bardi M., Falcone M., Soravia P., *Numerical methods for pursuit-evasion games via viscosity solutions*, in M. Bardi, T. Parthasarathy e T.E.S. Raghavan (eds.), "Stochastic and differential games: theory and numerical methods", Annals of the International Society of Dynamic Games **4**, 289–303, Birkhäuser, Boston, 1999.

[8] Bardi M., Koike S., Soravia P., *Pursuit-evasion games with state constraints: dynamic programming and discrete-time approximations*, Discrete and Continuous Dynamical Systems **6** (2000), 361–380.

[9] Breakwell J. V., *Time-optimal pursuit inside a circle*, Differential Games and Applications (Sophia-Antipolis, 1988), Lecture Notes in Control and Information Sciences **119**, 72–85, Springer, Berlin, 1989.

[10] Camilli F., Falcone M., *Approximation of optimal control problems with state constraints: estimates and applications*, in B. S. Mordukhovic, H. J. Sussman (eds.), "Nonsmooth analysis and geometric methods in deterministic optimal control", IMA Volumes in Applied Mathematics **78**, 23–57, Springer Verlag, Berlin, 1996.

[11] Cardaliaguet P., Quincampoix M., Saint-Pierre P. *Pursuit differential games with state constraints*, SIAM Journal of Control and Optimization **39** (2001), 1615–1632.

[12] Cardaliaguet P., Quincampoix M., Saint-Pierre P. *Differential games with state-constraints*, in the Proceedings of the Conference of the International

Society of Dynamics Games, St. Petersburg 2002.

[13] Cardaliaguet P., Quincampoix M., Saint-Pierre P., *Set valued numerical analysis for optimal control and differential games*, in M. Bardi, T. Parthasarathy and T. E. S. Raghavan (eds.), ”Stochastic and differential games: theory and numerical methods”, Annals of the International Society of Dynamic Games **4**, 177–247, Birkhäuser, Boston, 1999.

[14] Carlini E., Falcone M., Ferretti R., *An efficient algorithm for Hamilton-Jacobi equations in high-dimension*, Computer and Visual Sciences **7** (2004), 15–29.

[15] Cristiani E., *Fast Marching and semi-Lagrangian methods for Hamilton-Jacobi equations with applications*, Ph.D. Thesis, Dipartimento di Metodi e Modelli Matematici per le Scienze Applicate, Università ”La Sapienza”, Rome, Italy, 2006.

[16] Cristiani E., Falcone M., *A Fast Marching Method for Pursuit-Evasion Games*, published electronically in “Communications to SIMAI Congress", SIMAI 2006 (Baia Samuele, Ragusa, Italy, May 22-26, 2006), 1 (2006). http://cab.unime.it/journals/index.php/congress/article/view/136 /136

[17] Falcone M., *Numerical methods for differential games based on partial differential equations*, International Game Theory Review **8** (2006), 231–272.

[18] Falcone M., *The minimum time problem and its applications to front propagation*, in A. Visintin e G. Buttazzo (eds.), ”Motion by mean curvature and related topics”, De Gruyter Verlag, Berlin, 1994.

[19] Isaacs R., *Differential Games*, John Wiley and Sons, New York, 1965.

[20] Ishii H., Koike S., *A new formulation of state constraint problems for first-order PDEs*, SIAM Journal of Control and Optimization **34** (1996), 554–571.

[21] Koike S., *On the state constraint problem for differential games*, Indiana University Mathematics Journal **44** (1995), 467–487.

[22] Pesch H. J., Gabler I., Miesbach S., Breitner M. H., *Synthesis of optimal strategies for differential games by neural networks*, in G. J. Olsder (ed.), ”New trends in Dynamic Games and Application”, Annals of the International Society of Dynamic Games **3**, 111–141, Birkhäuser, Boston, 1995.

[23] Zhao H., *A fast sweeping method for eikonal equations*, Mathematics of Computation **74** (2005), 603–627.