INNER AND OUTER CAPTURE BASIN APPROXIMATION WITH SUPPORT VECTOR MACHINES

Laetitia Chapel
Lab-STICC, Université Européenne de Bretagne, Université de Bretagne Sud, 56017 Vannes Cedex, France

Guillaume Deffuant
Laboratoire d'Ingnierie pour les Systèmes Complexes, Cemagref, 63172 Aubière Cedex, France

Keywords: Viability theory, Capture basin, Optimal control, Support Vector Machines.

Abstract: We propose a new approach to solve target hitting problems, that iteratively approximates capture basins at successive times, using a machine learning algorithm trained on points of a grid with boolean labels. We consider two variants of the approximation (from inside and from outside), and we state the conditions on the machine learning procedure that guarantee the convergence of the approximations towards the actual capture basin when the resolution of the grid decreases to 0. Moreover, we define a control procedure which uses the set of capture basin approximations to drive a point into the target. When using the inner approximation, the procedure guarantees to hit the target, and when the resolution of the grid tends to 0, the controller tends to the optimal one (minimizing time to hit the target). We use Support Vector Machines as a particular learning method, because they provide parsimonious approximations, from which one can derive fast and efficient controllers. We illustrate the method on two simple examples, Zermelo and car on the hill problems.

1 INTRODUCTION

We focus on the problem of defining the control function for driving a dynamical system to reach a given target compact set $\mathcal{C} \subset \mathcal{K}$ in minimum time, without going out from $\mathcal{K}$, where $\mathcal{K}$ is a given set (called constraint set). This problem, often called the reachability problem, can be addressed by optimal control methods, solving Hamilton-Jacobi-Bellman (HJB) or Isaacs (HJI) equations. Several numerical techniques are available; for example, (Mitchell et al., 2005) propose an algorithm that computes an approximation for the backward reachable set of a system using a time dependent HJI partial differential equation, (Lygeros, 2004) builds the value function of the problem which can be then used to choose the best action at each time step.

Reachability problem can also be addressed in the viability theory framework (Aubin, 1991). To apply viability theory to target hitting problems, one must add an auxiliary dimension to the system, representing the time elapsed before reaching the target. The approach computes an approximation of the envelopes of all t-capture basins, the sets of points for which there exists a control function that allows the system to reach the target in a time less than t. (Frankowska, 1989) shows that the boundary of this set is the value function in the dynamical programming perspective. Hence, solving this extended viability problem also provides the minimal time for a state x to reach the target $\mathcal{C}$ while always staying in $\mathcal{K}$ (minimal time function $\vartheta_{\mathcal{C}}^{\mathcal{K}}(x)$ (Cardaliaguet et al., 1998)). This function can then be used to define controllers that drive the system into the target.

Several numerical algorithms (Saint-Pierre, 1994; Cardaliaguet et al., 1998) provide an overapproximation of capture basins. (Bayen et al., 2002) implement an algorithm proposed by (Saint-Pierre, 2001) that computes a discrete underapproximation of continuous minimal time functions (and thus an overapproximation of capture basins), without adding an additional dimension. (Lhommeau et al., 2007) present an algorithm, based on interval analysis, that provides an inner and outer approximation of the capture basin. In general, capture basin and minimal time function approximation algorithms face the curse of dimensionality, which limits their use to problems of low dimension (in the state and control space).

This paper proposes a new method to solve target hitting problems, inspired by our work on via-

Chapel L. and Deffuant G..
INNER AND OUTER CAPTURE BASIN APPROXIMATION WITH SUPPORT VECTOR MACHINES.
DOI: 10.5220/0003536000340040
In *Proceedings of the 8th International Conference on Informatics in Control, Automation and Robotics* (ICINCO-2011), pages 34-40
ISBN: 978-989-8425-74-4

bility kernel approximation (Deffuant et al., 2007). The principle is to approximate iteratively the capture basins at successive times t. To compute time t-capture basin approximation, we use a discrete grid of points covering set $\mathcal{K}$, and label $+1$ the points for which there exists a control leading the point into the $t-\delta t$-capture basin approximation, and -1 otherwise. Then, we use a machine learning method to compute a continuous boundary between +1 and -1 points of the grid. We state the conditions the learning method should fulfil (they are similar to the one established to approximate viability kernels (Deffuant et al., 2007)) in order to prove the convergence toward the actual capture basins.

We consider two variants of the algorithm: one provides an approximation that converges from outside, and the other from inside. Although no convergence rate is provided, the comparison of the two approximations gives an assessment of the approximation error for a given problem. Moreover, we define a controller that guarantees to reach the target when derived from the inner approximation.

We consider Support Vector Machines (SVMs (Vapnik, 1995; Scholkopf and Smola, 2002)) as a relevant machine learning technique in this context. Indeed, SVMs provide parsimonious approximations of capture basins, that allow the definition of compact controllers. Moreover, they make possible to use optimisation techniques to find the controls, hence problems with control spaces in more dimensions become tractable. We can also more easily compute the control on several time steps, which improves the quality of the solution for a given resolution of the grid.

We illustrate our approach with some experiments on two simple examples. Finally, we draw some perspectives.

2 PROBLEM DEFINITION

We consider a controlled dynamical system in discrete time (Euler approximation), described by the evolution of its state variable $x \in \mathcal{K} \subset \mathbb{R}^n$. We would like to define the set of controls to apply to the system starting from point x in order to reach the target $\mathcal{C} \subset \mathcal{K}$ in minimal time:

$$\begin{cases} x(t+dt) = x(t) + \varphi(x(t),u(t)).dt, & \text{if } x(t) \notin \mathcal{C} \\ x(t+dt) = x(t), & \text{if } x(t) \in \mathcal{C} \\ u(t) \in \mathcal{U}, & \end{cases} \quad (1)$$

where φ is a continuous and derivable function of x and u. The control u must be chosen at each time step in the set $\mathcal{U}$ of admissible controls.

The capture basin of the system is the set of states for which there exists at least one series of controls such that the system reaches the target in finite time, without leaving $\mathcal{K}$. Let $G(x,(u_1,..,u_n))$ be the point reached when applying successively during n time steps the controls $(u_1,..,u_n)$, starting from point x. Let the *minimal time function* (or hitting time function) be the function that associates to a state $x \in \mathcal{K}$ the minimum time to reach $\mathcal{C}$:

$$\begin{aligned} \vartheta_{\mathcal{C}}^{\mathcal{K}}(x) = \inf\{&n|\exists(u_1,..,u_n) \in \mathcal{U}^n \\ &\text{such that } G(x,(u_1,..,u_n)) \in \mathcal{C} \\ &\text{and for } 1 \leq j \leq n, G(x,(u_1,..,u_j)) \in \mathcal{K}\}. \end{aligned} \quad (2)$$

This is the value function obtained when solving HJB equations in a dynamic programming approach. It takes values in $\mathbb{N}^+ \cup +\infty$, specifically $\vartheta_{\mathcal{C}}^{\mathcal{K}}(x) = 0$ if $x \in \mathcal{C}$ and $\vartheta_{\mathcal{C}}^{\mathcal{K}}(x) = +\infty$ if no trajectory included in $\mathcal{K}$ can reach $\mathcal{C}$. The *capture basin* of $\mathcal{C}$ viable in $\mathcal{K}$ is then defined as:

$$Capt(\mathcal{K},\mathcal{C}) = \left\{x \in \mathcal{K} | \vartheta_{\mathcal{C}}^{\mathcal{K}}(x) < +\infty\right\}, \quad (3)$$

and we can also define the capture basin in finite time n:

$$Capt(\mathcal{K},\mathcal{C},n) = \left\{x \in \mathcal{K} | \vartheta_{\mathcal{C}}^{\mathcal{K}}(x) \leq n\right\}. \quad (4)$$

To solve a target hitting problem in the viability perspective, one must consider the following extended dynamical system $(x(t),y(t))$ when $x(t) \notin \mathcal{C}$:

$$\begin{cases} x(t+dt) = x(t) + \varphi(x(t),u(t)).dt \\ y(t+dt) = y(t) - dt. \end{cases} \quad (5)$$

and $(x(t+dt) = x(t), y(t+dt) = y(t))$ when $x(t) \in \mathcal{C}$. (Cardaliaguet et al., 1998) prove that approximating minimal time function comes down to a viability kernel approximation problem of this extended dynamical problem. In a viability problem, one must find the rule of controls for keeping a system indefinitely within a constraint set. (Bayen et al., 2002; Saint-Pierre, 2001) give examples of such an application of viability approach to solve a target hitting problem.

(Deffuant et al., 2007) proposed an algorithm, based on (Saint-Pierre, 1994), that uses a machine learning procedure to approximate viability kernels. The main advantage of this algorithm is that it provides continuous approximations that enable to find the controls with standard optimization techniques, and then to tackle problems with control in large dimensional space. The aim of this paper is to adapt (Deffuant et al., 2007) to compute directly an approximation of the capture basin limits, without adding the auxiliary dimension, and then to use these approximations to define the sequence of controls.

3 MACHINE LEARNING APPROXIMATION OF VALUE FUNCTION CONTOURS AND OPTIMAL CONTROL

For simplicity, we denote $Capt(\mathcal{K},\mathcal{C},n.dt) = Capt^n$. In all the following, continuous sets are denoted by rounded letters and discrete sets in capital letters.
We consider function G:

$$G(x,u) = \begin{cases} x+\varphi(x,u).dt & \text{if } x \notin \mathcal{C} \\ x & \text{if } x \in \mathcal{C} \end{cases} \quad (6)$$

We suppose that G is μ-lipschitz with respect to x:

$$\forall (x,y) \in \mathcal{K}^2, \forall u \in U |G(x,u) - G(y,u)| < \mu|x-y|. \quad (7)$$

We define a grid K_h as a discrete subset of $\mathcal{K}$, such that:

$$\forall x \in \mathcal{K}, \exists x_h \in K_h \text{ such that } |x-x_h| \leq h. \quad (8)$$

Moreover, we define an algebraic distance $d_a(x,\partial\mathcal{E})$of a point x to the boundary $\partial\mathcal{E}$ of a continuous closed set $\mathcal{E}$, as the distance to the boundary when x is located inside $\mathcal{E}$, and this distance negated when x is located outside $\mathcal{E}$:

$$\text{if } x \in \mathcal{E}, d_a(x,\partial\mathcal{E}) = d(x,\partial\mathcal{E}), \quad (9)$$
$$\text{if } x \notin \mathcal{E}, d_a(x,\partial\mathcal{E}) = -d(x,\partial\mathcal{E}). \quad (10)$$

3.1 Capture Basin Approximation Algorithms

In this section, we describe two variants of an algorithm that provides an approximation $\mathcal{C}_h^n$ of the capture basin at time $n.dt$, one variant approximates the capture basins from outside and the other one from inside. At each step n of the algorithm, we first build a discrete approximation $C_h^n \subset K_h$ of the capture basin $Capt^n$, and then we use a learning procedure L (for instance Support Vector Machines, as shown below) to generalise this discrete set into a continuous one $\mathcal{C}_h^n$:

$$\mathcal{C}_h^n = L(C_h^n) \quad (11)$$

To simplifying the writing, we first define:

$$\overline{C_h^n} = \{x_h \in K_h \text{ s.t. } x_h \notin C_h^n\}, \quad (12)$$
$$\overline{\mathcal{C}_h^n} = \{x \in \mathcal{K} \text{ s.t. } x \notin \mathcal{C}_h^n\}. \quad (13)$$

The two variants differ on the conditions for defining the discrete set C_h^{n+1} from C_h^n, and on the conditions the learning procedure L must fulfil. For both variant, we construct an increasing sequence of approximations at time $n.dt$, by adding the points of the grid for which there exists at least one control that drives the system not too far away (in an algebraic sense - negative distance in the outer case and positive distance in the inner one) from the boundary of the previous approximation. They also both require conditions on the learning procedure, in order to guarantee the convergence toward the actual capture basin when the step of the grid h decreases to 0. In the inner approximation case, the condition is stricter on set $\mathcal{C}_h^n$ and more relaxed on set $\overline{\mathcal{C}_h^n}$, while the outer case requires converse conditions. We now describe in more details both variants and conditions.

3.1.1 Outer Approximation

Given initial sets $C_h^0 = \mathcal{C} \cap K_h$, $\mathcal{C}_h^0 = \mathcal{C}$, and supposing that we define continuous approximations $\mathcal{C}_h^n$ from C_h^n (eq. (11)), we define discrete sets C_h^n as follows:

$$C_h^{n+1} = C_h^n \bigcup \{x_h \in \overline{C_h^n} \text{ s.t. } \exists u \in U, \\ d_a(G(x_h,u),\partial\mathcal{C}_h^n) \geq -\mu h\}. \quad (14)$$

Proposition. If the learning procedure L respects the following conditions:

$$\forall x \in \overline{\mathcal{C}_h^n}, \exists x_h \in \overline{C_h^n} \text{ s.t. } |x-x_h| \leq h \quad (15)$$
$$\exists \lambda \geq 1 \text{s.t.} \forall h, \forall x \in \mathcal{C}_h^n, \exists x_h \in C_h^n \text{ s.t. } |x-x_h| \leq \lambda h \quad (16)$$

then the convergence of the approximation from outside is guaranteed:

$$\forall n, Capt^n \subset \mathcal{C}_h^n, \quad (17)$$
$$\mathcal{C}_h^n \rightarrow Capt^n \text{ when } h \rightarrow 0. \quad (18)$$

Proof. The proof is given on appendix 5.

3.1.2 Inner Approximation

Given initial sets $C_h^0 = \mathcal{C} \cap K_h$, $\mathcal{C}_h^0 = \mathcal{C}$, and supposing that we define continuous approximations $\mathcal{C}_h^n$ from C_h^n, we define discrete sets C_h^n as follows:

$$C_h^{n+1} = C_h^n \bigcup \{x_h \in \overline{C_h^n} \text{ s.t. } \exists u \in U, \\ d_a(G(x_h,u),\partial\mathcal{C}_h^n) \geq \mu h\}. \quad (19)$$

Proposition. If the learning procedure L respects the following conditions:

$$\forall x \in \mathcal{C}_h^n, \exists x_h \in C_h^n \text{ s.t. } |x-x_h| \leq h \quad (20)$$
$$\exists \lambda \geq 1 \text{s.t.} \forall h, \forall x \in \overline{\mathcal{C}_h^n}, \exists x_h \in \overline{C_h^n} \text{ s.t. } |x-x_h| \leq \lambda h \quad (21)$$

and that the dynamics are such that:

$$\forall x \in \mathcal{K} \text{ with } d_a\left(x,\partial Capt^{n+1}\right) > 0, \\ \exists u \in \mathcal{U} \mid d_a\left(G(x,u),\partial Capt^n\right) > 0. \quad (22)$$

then the convergence of the approximation from inside is guaranteed:

$$\forall n, \mathcal{C}_h^n \subset Capt^n, \quad (23)$$
$$\mathcal{C}_h^n \rightarrow Capt^n \text{ when } h \rightarrow 0. \quad (24)$$

Proof. Convergence proof of the algorithm from inside requires an additional condition on the dynamics (eq. (22)): a point x of the interior of capture basin at time $(n+1).dt$, should be such that there exists $y \in G(x)$ belonging to the interior of capture basin at time $n.dt$ (and not on $\partial Capt^n$). The proof of convergence is on appendix 5.

3.2 Optimal Control

The aim of the optimal controller is to choose a control function that reaches the target in minimal time, without breaking the viability constraints. The idea is to choose the controls which drive the system to cross $\mathcal{C}_h^n$ boundaries in a descending order.

Proposition. Consider x_0, such that $x_0 \in \mathcal{C}_h^{n+1}$ and $x_0 \notin \mathcal{C}_h^n$. The procedure which, for $i = 0$ to n computes $u^*(x_i)$ defined such that $x_{i+1} = G(x_i, u^*(x_i)) \in \mathcal{C}_h^{n-i}$, converges to the control policy minimizing the hitting time, when h and dt tend to 0.

Proof. By construction, if $x_i \in \mathcal{C}_h^{n+1-i}$ and $x_i \notin \mathcal{C}_h^{n-i}$, there exists a control value $u^*(x_i)$ such that $x_{i+1} = G(x_i, u^*(x_i)) \in \mathcal{C}_h^{n-i}$ (see Appendix 5, part I.). Therefore, the procedure leads to the target in $n+1$ time steps, i.e. in $(n+1).dt$ time. Moreover, by definition, the optimum time for reaching the target from a point x located on the boundary of capture basin $Capt^n$ is $n.dt$. Hence, the optimum time for reaching the target from point x such that $x \in Capt^{n+1}$ and $x \notin Capt^n$, with the dynamics defined by φ, is between $n.dt$ and $(n+1).dt$. Then, the fact that $\mathcal{C}_h^n$ converges to $Capt^n$ when h tends to 0, ensures that the number of time steps needed by the procedure applied to this point x will tend to $(n+1).dt$. When dt tends to 0, the difference with the optimum time to reach the target, which is smaller than dt, tends to 0.

4 EXPERIMENTS

4.1 SVM as a Learning Procedure

We use Support Vector Machines (Vapnik, 1995; Scholkopf and Smola, 2002) as the learning procedure L to define capture basin approximations $\mathcal{C}_h^n = L(C_h^n)$. At each iteration, we construct a learning set: let S_h^n be a set of couples (x_h, y_h), where $x_h \in K_h$ and $y_h = +1$ if $x_h \in C_h^n$ and -1 otherwise. Running a SVM on learning sets S_h^n provides a separating function f_h^n between points of different labels and hence, allows us to define a continuous approximation $\mathcal{C}_h^n$ of C_h^n as follows:

$$\mathcal{C}_h^n = \{x \in K \text{ such that } f_h^n(x) \geq 0\}. \qquad (25)$$

Points x of the boundary $\partial\mathcal{C}_h^n$ are those such that $f_h^n(x) = 0$. The fulfilment of the conditions guaranteeing convergence is discussed in (Deffuant et al., 2007) and the same arguments hold in both variants of the algorithm.
In the following examples, we use libSVM (Chang and Lin, 2001) to train the SVMs. As we did in (Deffuant et al., 2007), we use the SVM function as a proxy for the distance to the boundary, in order to simplify the computations.

4.2 Zermelo Problem

The state $(x(t), y(t))$ of the system represents the position of a boat in a river. There are two controls: the thrust u and the direction θ of the boat. The system in discrete time defined by a time interval dt can be written as follows:

$$\begin{cases} x(t+dt) = x(t) + (1 - 0.1y(t)^2 + u\cos\theta).dt \\ y(t+dt) = y(t) + (u\sin\theta).dt, \end{cases} \qquad (26)$$

where $u \in [0; 0.44]$ and $\theta \in [0; 2\pi]$. The boat must remain in a given interval $\mathcal{K} = [-6; 2] \times [-2; 2]$, and reach a round island $\mathcal{C} = B(0; 0.44)$. We suppose that the boat must reach the island before time $T = 7$.
For this simple system, it is possible to derive analytically the capture basin, hence we can compare the approximations given by the two algorithms with the actual capture basin. Figure 1 compares some results obtained with the outer and inner approximation. In any cases, the quality of the approximation can be assessed by comparing both approximations: by construction, the contour of the actual capture basin is surrounded by inner and outer approximations. A example of a optimal trajectory defined with the optimal controller is also presented: with the inner approximation, the trajectory will enable the boat to reach the target, while it is not guaranteed in the outer case.

4.3 Car on the Hill

We consider the well-known car on the hill problem. The state is two-dimensional (position and velocity) and the system can be controlled with a continuous one-dimensional action (thrust). For a description of the dynamics and the state space constraints, one can refers to (Moore and Atkeson, 1995). The aim of the car on the hill system is to keep the car inside a given set of constraints, and to reach a target (the top of the hill) as fast as possible. Figure 2 shows the approximation of the contours of the value function using outer and inner variants of the algorithm, with an example of optimal trajectories.

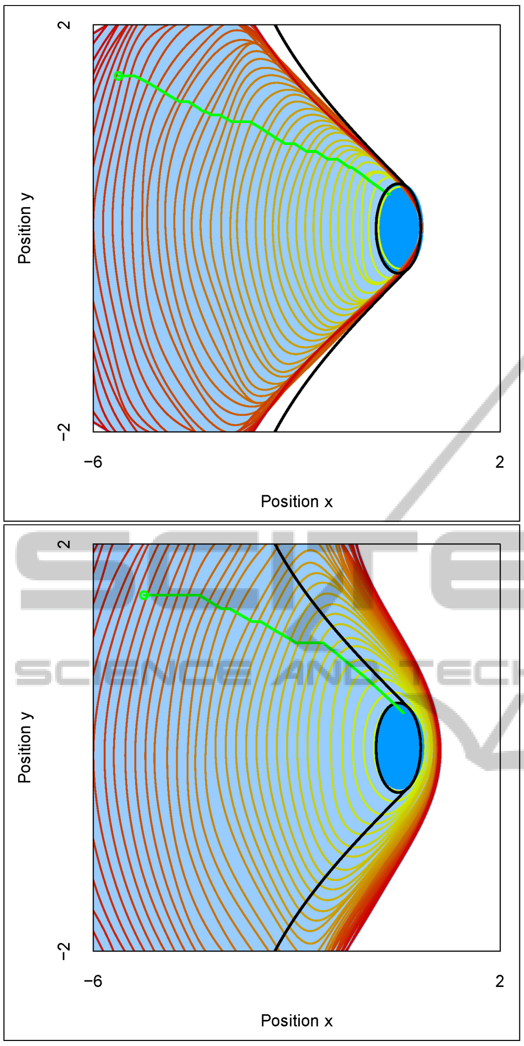

Figure 1: Approximation from inside (top) and from outside (bottom) for Zermelo problem. The horizontal axis represents the position x and the vertical one position y. $\mathcal{K}$ is the rectangle. The capture basin is represented in blue. The black thick line limits the boundary of the actual capture basin. The level lines represent approximation of the contours of the capture basins for successive time steps. The grid contains 41 points by dimension. The optimisation is made on 4 time steps, with $dt = 0.05$. Each figure presents an example of trajectory (in green) using the SVM optimal controller.

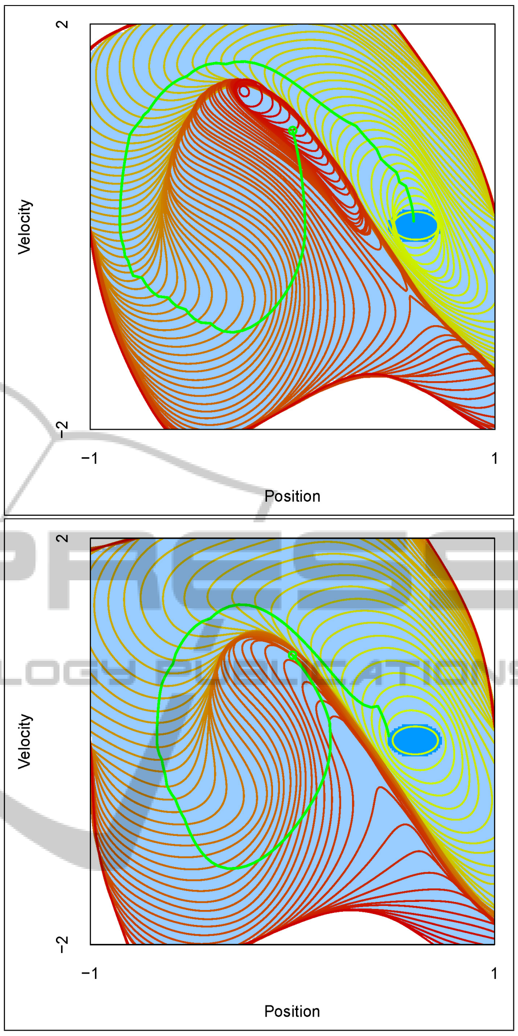

Figure 2: Inner (top) and outer (bottom) approximation for the car on the hill problem. The grid contains 51 points by dimension. The optimisation is made on 2 time steps. An example of an optimal trajectory is depicted in green.

5 DISCUSSION

We proposed an algorithm that approximates capture basins and minimal time functions, using a classification procedure, in two variants (inner and outer). The inner approximation can be used to define a optimal controller that guarantees to find a series of controls that allows the system to reach the target. SVMs appear as a particularly relevant classification procedure for this approach, because they provide parsimonious representations of the capture basins and enable to use optimization techniques to compute the controls. This latter point is particularly important to deal with high dimensional control space. The parcimonious property may allow to define compact and fast controller, even for high dimensional state space. However, although we generally manage to find parameters in which the result respect the conditions of convergence, this is not guaranteed. Therefore, considering other learning algorithms that would be even more appropriate seems a relevant research direction. A second direction of research is to investigate other problems that could be solved by this approach. For instance in the definition of resilience proposed by

(Martin, 2004), there is a problem of target hitting in which a cost function is associated with the states. We think that our approach could easily be adapted to this problem.

REFERENCES

Aubin, J.-P. (1991). *Viability theory*. Birkhäuser.

Bayen, A., E., C., and & Tomlin, C. (2002). Guaranteed overapproximations of unsafe sets for continuous and hybrid systems: Solving the hamilton-jacobi equation using viability techniques. *Lecture Notes In Computer Science*, 2289:90–104.

Cardaliaguet, P., Quincampoix, M., and Saint-Pierre, P. (1998). *Set-Valued Numerical Analysis for Optimal control and Differential Games*. Annals of the International Society of Dynamic Games.

Chang, C.-C. and Lin, C.-J. (2001). Libsvm: a library for support vector machines.

Deffuant, G., Chapel, L., and Martin, S. (2007). Approximating viability kernels with support vector machines. *IEEE transactions on automatic control*, 52(5):933–937.

Frankowska, H. (1989). Optimal trajectories associated with a solution of the contingent hamilton-jacobi equation. *Applied Mathematics and Optimization*, 19(1):291–311.

Lhommeau, M., Jaulin, L., and Hardouin, L. (2007). Inner and outer approximation of capture basin using interval analysis. In *4th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2007)*.

Lygeros, J. (2004). On reachability and minimum cost optimal control. *Automatica*, 40:917–927.

Martin, S. (2004). The cost of restoration as a way of defining resilience: a viability approach applied to a model of lake eutrophication. *Ecology and Society*, 9(2).

Mitchell, I., Bayen, A., and Tomlin, C. (2005). A time-dependent Hamilton-Jacobi formulation for reachable sets for continuous dynamic games. *IEEE Transactions on Automatic Control*, 50(7):947–957.

Moore, A. and Atkeson, C. (1995). The parti-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces. *Machine Learning*, 21:199–233.

Saint-Pierre, P. (1994). Approximation of the viability kernel. *Applied Mathematics & Optimization*, 29(2):187–209.

Saint-Pierre, P. (2001). Approche ensembliste des systèmes dynamiques, regards qualitatifs et quantitatifs. *Matapli, Société de Mathématiques appliquées et Industrielles*, page 66.

Scholkopf, B. and Smola, A. (2002). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA.

Vapnik, V. (1995). *The nature of statistical learning theory*. Springer Verlag.

APPENDIX

Proof of Proposition 3.1.1

Part I. First, let us prove by induction that $\forall h > 0, Capt^n \subset \mathcal{C}_h^n$.
By definition, $Capt^0 = \mathcal{C} = \mathcal{C}_h^0$. Suppose that at step n, $Capt^n \subset \mathcal{C}_h^n$. Consider $x \in Capt^{n+1}$. Let us recall that $G(x,u) = x + \varphi(x,u).dt$ when $x \notin \mathcal{C}$.
Defining $B_h(x,d)$ the set of points of K_h such that $|x - x_h| \leq d$, we can easily show that condition (16) can be rewritten as:

$$B_h(x,h) \subset C_h^n \Rightarrow x \in \mathcal{C}_h^n. \qquad (27)$$

By definition, we know that there exists $u \in \mathcal{U}$ such that $G(x,u) \in Capt^n$, which implies that for all $x_h \in B_h(x,h)$, $d\left(G(x_h,u), Capt^n\right) \leq \mu h$, because G is μ-Lipschitz.
Moreover, for all $x_h \in B_h(x,h)$, $d(G(x_h,u), \mathcal{C}_h^n) \leq \mu h$, because, by hypothesis, $Capt^n \subset \mathcal{C}_h^n$. Thus $x_h \in C_h^{n+1}$.
Therefore, $x \in \mathcal{C}_h^{n+1}$ (because of condition (27)).
We can thus conclude $Capt^{n+1} \subset \mathcal{C}_h^{n+1}$.

Part II. Now, we prove by induction that for any n, $\mathcal{C}_h^n \rightarrow Capt^n$ when $h \rightarrow 0$.
Suppose now that for a given value n, $\mathcal{C}_h^n \rightarrow Capt^n$ when $h \rightarrow 0$.
Because $Capt^n \subset \mathcal{C}_h^n$, we have:
$\forall x \in \mathcal{K} \mid x \notin Capt^n, \exists h > 0 \mid x \notin \mathcal{C}_h^n$.
Now, consider $x \in \mathcal{K}$ such that $x \notin Capt^{n+1}$.
This implies that for all $u \in \mathcal{U}$ such that $d\left(G(x,u), Capt^n\right) > 0$. One can choose $h' > 0$ and h such that for all $u \in \mathcal{U}$, $d\left(G(x,u), Capt^n\right) > h' + \mu\lambda h$.
Condition (15) can be rewritten as:

$$B_h(x,\lambda h) \subset \overline{C_h^n} \Rightarrow x \in \overline{\mathcal{C}_h^n}. \qquad (28)$$

In this case, for all $x_h \in B_h(x,\lambda h)$, all $u \in \mathcal{U}$, $d\left(G(x_h,u), Capt^n\right) > h'$, because G is μ-Lipschitz.
Since $\mathcal{C}_h^n \rightarrow Capt^n$ when $h \rightarrow 0$, there exists h, such that, for all $x_h \in B_h(x,\lambda h)$, and all $u \in \mathcal{U}$, $G(x_h,u) \in \overline{\mathcal{C}_h^n}$, hence $x_h \in \overline{\mathcal{C}_h^n}$. Hence, there exists h such that $x \notin \mathcal{C}_h^n$ (because of condition 27).
Therefore $\mathcal{C}_h^{n+1} \rightarrow Capt^{n+1}$ when $h \rightarrow 0$.
Conclusion. $Capt^n \subset \mathcal{C}_h^n$ and $\mathcal{C}_h^n \rightarrow Capt^n$ then $\mathcal{C}_h^n$ is an outer approximation of the capture basin at time $n.dt$, which tends to the actual capture basin when the resolution of the grid h tends to 0.

Proof of Proposition 3.1.2

Part I. We begin to show by induction that $\mathcal{C}_h^n \subset Capt^n$.
Suppose that $\mathcal{C}_h^n \subset Capt^n$ and consider $x \in \mathcal{C}_h^{n+1}$.
Because of condition (20):

$$\exists x_h \in C_h^{n+1} \text{ such that } |x - x_h| < h.$$

By definition of C_h^{n+1}:

$$\exists u \in U \text{ such that } d_a\left(G(x_h,u), \mathcal{C}_h^n\right) > \mu h.$$

By hypothesis of induction $\mathcal{C}_h^n \subset Capt^n$, hence : $d_a\left(G(x_h,u), Capt^n\right) > \mu h$. By hypothesis on G, $|G(x_h,u) - G(x,u)| < \mu|x_h - x|$, hence $d_a\left(G(x,u), Capt^n\right) > 0$. Therefore $x \in Capt^{n+1}$. Thus $\mathcal{C}_h^{n+1} \subset Capt^{n+1}$.

Part II. We prove by induction that, when $h \to 0$, $\mathcal{C}_h^n \to Capt^n$.
Suppose that $\mathcal{C}_h^n \to Capt^n$ when $h \to 0$.
Because $\mathcal{C}_h^n \subset Capt^n$, we have:

$$\forall x \in Capt^n \mid d_a(x, \partial Capt^n) > 0, \exists h > 0 \mid x \in \mathcal{C}_h^n.$$

We use the rewriting of condition (21):

$$B_h(x, \lambda h) \cap C_h^n \Rightarrow x \in \mathcal{C}_h^n. \tag{29}$$

Consider $x \in Capt^{n+1}$ such that $d_a(x, \partial Capt^{n+1}) > 0$. One can choose h such that $d_a(x, \partial Capt^{n+1}) > (\mu + \lambda)h$. With such a choice, for each $x_h \in B_h(x, \lambda h)$, $d_a(x_h, \partial Capt^{n+1}) > \mu h$, hence, there exists $u \in \mathcal{U}$ such that $d_a\left(G(x_h,u), Capt^n)\right) > 0$ (because G is μ-Lipschitz).
By induction hypothesis, there exists h such that $d_a\left(G(x_h,u), \mathcal{C}_h^n\right) > \mu h$, hence $x_h \in C_h^{n+1}$. Taking the smallest value of h this is true for all $x_h \in B_h(x, \lambda h)$.
Therefore $x \in \mathcal{C}_h^{n+1}$ (because of condition (29)).
Therefore $\mathcal{C}_h^{n+1} \to Capt^{n+1}$ when $h \to 0$.
Conclusion. $\mathcal{C}_h^n \subset Capt^n$ and $\mathcal{C}_h^n \to Capt^n$ then $\mathcal{C}_h^n$ is an inner approximation of the capture basin in finite time $n.dt$, which tends to the actual capture basin when the resolution of the grid tends to 0.