Inner and outer approximation of capture basin using interval analysis

M. Lhommeau [1] L. Jaulin[2] L. Hardouin[1]

[1]Laboratoire d'Ingénierie des Systèmes Automatisés
ISTIA - Université d'Angers
62, av. Notre Dame du Lac, 49000 Angers, France

[2]E[3]I[2]
ENSIETA
2 rue Françoise Verny, 29806 Brest, France

International Conference on Informatics in Control, Automation and Robotics, 2007

Motivation

- Let us consider the following dynamical system :

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \quad (1)$$

- where
  - state vector $\boldsymbol{x}(t)$ is not allowed to exit a given compact set $\mathbb{U} \subset \mathbb{R}^n$;
  - control vector $\boldsymbol{u}(t)$ should belong to a given compact set $\mathbb{U} \subset \mathbb{R}^m$;
  - $\boldsymbol{f} \in \mathcal{C}^1\left(\mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m, \mathbb{R}^n\right)$.

Flow function

- $\phi^t(\boldsymbol{x}_0, \boldsymbol{u})$ is the solution of (1) for the initial vector $\boldsymbol{x}_0$ and for the input function $\boldsymbol{u}$.
- The path from t_1 to t_2 is defined by

$$\phi^{[t_1,t_2]}(\boldsymbol{x}_0, \boldsymbol{u}) \overset{\text{def}}{=} \left\{\boldsymbol{x} \in \mathbb{R}^n : \exists t \in [t_1, t_2], \boldsymbol{x} = \phi^t(\boldsymbol{x}_0, \boldsymbol{u})\right\}.$$

Motivation

- Let us consider the following dynamical system :

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \quad (1)$$

- where
  - state vector $\boldsymbol{x}(t)$ is not allowed to exit a given compact set $\mathbb{U} \subset \mathbb{R}^n$;
  - control vector $\boldsymbol{u}(t)$ should belong to a given compact set $\mathbb{U} \subset \mathbb{R}^m$;
  - $\boldsymbol{f} \in \mathcal{C}^1(\mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m, \mathbb{R}^n)$.

Flow function

- $\phi^t(\boldsymbol{x}_0, \boldsymbol{u})$ is the solution of (1) for the initial vector $\boldsymbol{x}_0$ and for the input function $\boldsymbol{u}$.
- The path from t_1 to t_2 is defined by

$$\phi^{[t_1,t_2]}(\boldsymbol{x}_0, \boldsymbol{u}) \overset{\text{def}}{=} \{\boldsymbol{x} \in \mathbb{R}^n \mid \exists t \in [t_1, t_2], \boldsymbol{x} = \phi^t(\boldsymbol{x}_0, \boldsymbol{u})\}.$$

Motivation

- Let us consider the following dynamical system :

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \quad (1)$$

- where
  - state vector $\boldsymbol{x}(t)$ is not allowed to exit a given compact set $\mathbb{U} \subset \mathbb{R}^n$;
  - control vector $\boldsymbol{u}(t)$ should belong to a given compact set $\mathbb{U} \subset \mathbb{R}^m$;
  - $\boldsymbol{f} \in \mathcal{C}^1(\mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m, \mathbb{R}^n)$.

Flow function

- $\phi^t(\boldsymbol{x}_0, \boldsymbol{u})$ is the solution of (1) for the initial vector $\boldsymbol{x}_0$ and for the input function $\boldsymbol{u}$.
- The path from t_1 to t_2 is defined by

$$\phi^{[t_1,t_2]}(\boldsymbol{x}_0, \boldsymbol{u}) \overset{\text{def}}{=} \left\{ \boldsymbol{x} \in \mathbb{R}^n \mid \exists t \in [t_1, t_2], \boldsymbol{x} = \phi^t(\boldsymbol{x}_0, \boldsymbol{u}) \right\}.$$

Motivation

- Let us consider the following dynamical system :

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \qquad (1)$$

- where
  - state vector $\boldsymbol{x}(t)$ is not allowed to exit a given compact set $\mathbb{U} \subset \mathbb{R}^n$;
  - control vector $\boldsymbol{u}(t)$ should belong to a given compact set $\mathbb{U} \subset \mathbb{R}^m$;
  - $\boldsymbol{f} \in \mathcal{C}^1 (\mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m, \mathbb{R}^n)$.

Flow function

- $\phi^t(\boldsymbol{x}_0, \boldsymbol{u})$ is the solution of (1) for the initial vector $\boldsymbol{x}_0$ and for the input function $\boldsymbol{u}$.
- The path from t_1 to t_2 is defined by

$$\phi^{[t_1,t_2]}(\boldsymbol{x}_0, \boldsymbol{u}) \overset{\text{def}}{=} \{\boldsymbol{x} \in \mathbb{R}^n : \exists t \in [t_1, t_2], \boldsymbol{x} = \phi^t(\boldsymbol{x}_0, \boldsymbol{u})\}.$$

Motivation

- Let us consider the following dynamical system :

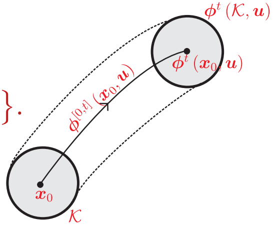
$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \quad (1)$$

- where
  - state vector $\boldsymbol{x}(t)$ is not allowed to exit a given compact set $\mathbb{U} \subset \mathbb{R}^n$;
  - control vector $\boldsymbol{u}(t)$ should belong to a given compact set $\mathbb{U} \subset \mathbb{R}^m$;
  - $\boldsymbol{f} \in \mathcal{C}^1 (\mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m, \mathbb{R}^n)$.

Flow function

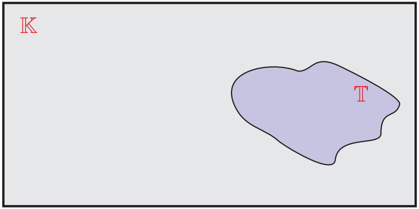
- $\boldsymbol{\phi}^t(\boldsymbol{x}_0, \boldsymbol{u})$ is the solution of (1) for the initial vector $\boldsymbol{x}_0$ and for the input function $\boldsymbol{u}$.
- The path from t_1 to t_2 is defined by

$$\boldsymbol{\phi}^{[t_1,t_2]}(\boldsymbol{x}_0, \boldsymbol{u}) \overset{\text{def}}{=} \{\boldsymbol{x} \in \mathbb{R}^n, \exists t \in [t_1, t_2], \boldsymbol{x} = \phi^t(\boldsymbol{x}_0, \boldsymbol{u})\}.$$



Capture Basin

The capture basin of a target $\mathbb{T}$ in $\mathbb{K} \subset \mathbb{R}^n$ is the set of points $\mathbb{C}$ of $\mathbb{K}$ such that there exist a trajectory starting at $\mathbb{C}$ reaching $\mathbb{T}$ in finite time.



Objective

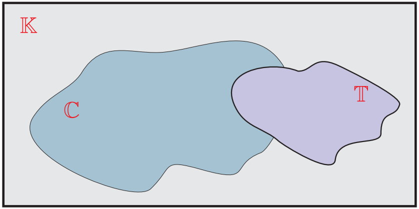
The aim of the paper is to provide an algorithm able to compute an inner and an outer approximation of set

$$\mathbb{C} \overset{\text{def}}{=} \{x_0 \in \mathbb{K}, \exists t \geq 0, \exists u, \phi^t(x_0, u) \in \mathbb{T} \text{ and } \phi^{[0,t]}(x_0, u) \subset \mathbb{K}\}.$$

i.e., to find two subsets $\mathbb{C}^-$ and $\mathbb{C}^+$ such that $\mathbb{C}^- \subset \mathbb{C} \subset \mathbb{C}^+$.

Capture Basin

The capture basin of a target $\mathbb{T}$ in $\mathbb{K} \subset \mathbb{R}^n$ is the set of points $\mathbb{C}$ of $\mathbb{K}$ such that there exist a trajectory starting at $\mathbb{C}$ reaching $\mathbb{T}$ in finite time.



Objective

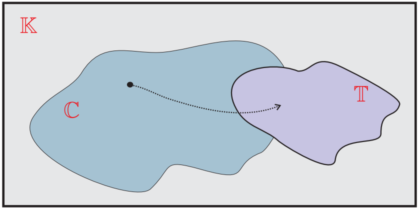
The aim of the paper is to provide an algorithm able to compute an inner and an outer approximation of set

$$\mathbb{C} \overset{\text{def}}{=} \{x_0 \in \mathbb{K}, \exists t \geq 0, \exists u, \phi^t(x_0, u) \in \mathbb{T} \text{ and } \phi^{[0,t]}(x_0, u) \subset \mathbb{K}\}.$$

i.e., to find two subsets $\mathbb{C}^-$ and $\mathbb{C}^+$ such that $\mathbb{C}^- \subset \mathbb{C} \subset \mathbb{C}^+$.

Capture Basin

The capture basin of a target $\mathbb{T}$ in $\mathbb{K} \subset \mathbb{R}^n$ is the set of points $\mathbb{C}$ of $\mathbb{K}$ such that there exist a trajectory starting at $\mathbb{C}$ reaching $\mathbb{T}$ in finite time.



Objective

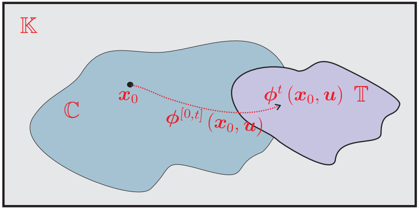
The aim of the paper is to provide an algorithm able to compute an inner and an outer approximation of set

$$\mathbb{C} \overset{\text{def}}{=} \{\boldsymbol{x}_0 \in \mathbb{K}, \exists t \geq 0, \exists \boldsymbol{u},\ \boldsymbol{\phi}^t(\mathbf{x}_0, \boldsymbol{u}) \in \mathbb{T} \text{ and } \boldsymbol{\phi}^{[0,t]}(\boldsymbol{x}_0, \boldsymbol{u}) \subset \mathbb{K}\},$$

i.e., to find two subsets $\mathbb{C}^-$ and $\mathbb{C}^+$ such that $\mathbb{C}^- \subset \mathbb{C} \subset \mathbb{C}^+$.

Capture Basin

The capture basin of a target $\mathbb{T}$ in $\mathbb{K} \subset \mathbb{R}^n$ is the set of points $\mathbb{C}$ of $\mathbb{K}$ such that there exist a trajectory starting at $\mathbb{C}$ reaching $\mathbb{T}$ in finite time.



Objective

The aim of the paper is to provide an algorithm able to compute an inner and an outer approximation of set

$$\mathbb{C} \stackrel{\text{def}}{=} \{\boldsymbol{x}_0 \in \mathbb{K}, \exists t \geq 0, \exists \boldsymbol{u},\ \boldsymbol{\phi}^t(\mathbf{x}_0, \boldsymbol{u}) \in \mathbb{T} \text{ and } \boldsymbol{\phi}^{[0,t]}(\boldsymbol{x}_0, \boldsymbol{u}) \subset \mathbb{K}\},$$

i.e., to find two subsets $\mathbb{C}^-$ and $\mathbb{C}^+$ such that $\mathbb{C}^- \subset \mathbb{C} \subset \mathbb{C}^+$.

Definitions

- Interval

$$[x] = \{x \in \mathbb{R} \mid \underline{x} \leq x \leq \overline{x}\}$$

- Width

$$w([x]) = \overline{x} - \underline{x}$$

- Midpoint

$$\text{mid}([x]) = \frac{\underline{x} + \overline{x}}{2}$$

- Intervals have a dual nature :
  - *sets* $\Rightarrow$ set-theoretic operations apply
  - *pairs of real-numbers* $\Rightarrow$ an arithmetic can be built

Definitions

- Interval

$$[x] = \{x \in \mathbb{R} \mid \underline{x} \leq x \leq \overline{x}\}$$

- Width

$$w([x]) = \overline{x} - \underline{x}$$

- Midpoint

$$\text{mid}([x]) = \frac{\underline{x} + \overline{x}}{2}$$

- Intervals have a dual nature :
  - *sets* $\Rightarrow$ set-theoretic operations apply
  - *pairs of real-numbers* $\Rightarrow$ an arithmetic can be built

Definitions

- Interval

$$[x] = \{x \in \mathbb{R} \mid \underline{x} \leq x \leq \overline{x}\}$$

- Width

$$w([x]) = \overline{x} - \underline{x}$$

- Midpoint

$$\mathrm{mid}([x]) = \frac{\underline{x} + \overline{x}}{2}$$

- Intervals have a dual nature :
  - *sets* $\Rightarrow$ set-theoretic operations apply
  - *pairs of real-numbers* $\Rightarrow$ an arithmetic can be built

Definitions

- Interval

$$[x] = \{x \in \mathbb{R} \mid \underline{x} \leq x \leq \overline{x}\}$$

- Width

$$w([x]) = \overline{x} - \underline{x}$$

- Midpoint

$$\text{mid}([x]) = \frac{\underline{x} + \overline{x}}{2}$$

- Intervals have a dual nature :
  - *sets* $\Rightarrow$ set-theoretic operations apply
  - *pairs of real-numbers* $\Rightarrow$ an arithmetic can be built

Operations on intervals

$$\begin{aligned}
[x]+[y] &= \left[\underline{x}+\underline{y}, \overline{x}+\overline{y}\right] \\
[x]-[y] &= \left[\underline{x}-\overline{y}, \overline{x}+\underline{y}\right] \\
[x]\times[y] &= \left[\min\left(\underline{x}, \underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{xy}\right), \max\left(\underline{x}, \underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{xy}\right)\right]
\end{aligned}$$

- If $0 \notin [y]$ then

$$[x]\,/\,[y] = [x]\times\left[1/\overline{y}, 1/\underline{y}\right]$$

$\Rightarrow$ Specific formulas available for division by interval containing zero.

Operations on intervals

$$
\begin{aligned}
[x]+[y] &= \left[\underline{x}+\underline{y}, \overline{x}+\overline{y}\right] \\
[x]-[y] &= \left[\underline{x}-\overline{y}, \overline{x}+\underline{y}\right] \\
[x]\times[y] &= \left[\min\left(\underline{x},\underline{y},\underline{x}\overline{y},\overline{x}\underline{y},\overline{xy}\right), \max\left(\underline{x},\underline{y},\underline{x}\overline{y},\overline{x}\underline{y},\overline{xy}\right)\right]
\end{aligned}
$$

- If $0 \notin [y]$ then

$$[x]\,/\,[y] = [x]\times\left[1/\overline{y}, 1/\underline{y}\right]$$

$\Rightarrow$ Specific formulas available for division by interval containing zero.

- Interval counterpart $[f]^*$ of f from $\mathbb{R}$ to $\mathbb{R}$ satisfies

$$[f]^*([x]) = [\{f(x) \mid x \in [x]\}].$$

- For any continuous function $[f]^*([x])$ is the image set $f([x])$.
- Elementary interval functions are expressed in terms of bounds
- For instance

$$[\exp]([x]) = [\exp(\underline{x}), \exp(\overline{x})]$$

- Specific algorithms for
  - trigonometric functions
  - hyperbolic functions

- Interval counterpart $[f]^*$ of f from $\mathbb{R}$ to $\mathbb{R}$ satisfies

$$[f]^*([x]) = [\{f(x) \mid x \in [x]\}].$$

- For any continuous function $[f]^*([x])$ is the image set $f([x])$.
- Elementary interval functions are expressed in terms of bounds
- For instance

$$[\exp]([x]) = [\exp(\underline{x}), \exp(\overline{x})]$$

- Specific algorithms for
  - trigonometric functions
  - hyperbolic functions

- Interval counterpart $[f]^*$ of f from $\mathbb{R}$ to $\mathbb{R}$ satisfies

$$[f]^*([x]) = [\{f(x) \mid x \in [x]\}].$$

- For any continuous function $[f]^*([x])$ is the image set $f([x])$.
- Elementary interval functions are expressed in terms of bounds
- For instance

$$[\exp]([x]) = [\exp(\underline{x}), \exp(\overline{x})]$$

- Specific algorithms for
  - trigonometric functions
  - hyperbolic functions

- Interval counterpart $[f]^*$ of f from $\mathbb{R}$ to $\mathbb{R}$ satisfies

$$[f]^*([x]) = [\{f(x) \mid x \in [x]\}].$$

- For any continuous function $[f]^*([x])$ is the image set $f([x])$.
- Elementary interval functions are expressed in terms of bounds
- For instance

$$[\exp]([x]) = [\exp(\underline{x}), \exp(\overline{x})]$$

- Specific algorithms for
  - trigonometric functions
  - hyperbolic functions

- Interval counterpart $[f]^*$ of f from $\mathbb{R}$ to $\mathbb{R}$ satisfies

$$[f]^*([x]) = [\{f(x) \mid x \in [x]\}].$$

- For any continuous function $[f]^*([x])$ is the image set $f([x])$.
- Elementary interval functions are expressed in terms of bounds
- For instance

$$[\exp]([x]) = [\exp(\underline{x}), \exp(\overline{x})]$$

- Specific algorithms for
  - trigonometric functions
  - hyperbolic functions

Interval vectors

Definition

- **Interval vector** (or box) is a Cartesian product of intervals

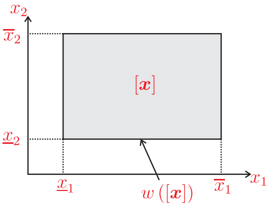
$$[\boldsymbol{x}] \;=\; [x_1] \times [x_2] \times \ldots \times [x_n] = ([x_1], [x_2], \ldots, [x_n])^{\mathsf{T}}$$

- Width

$$w([\boldsymbol{x}]) \;=\; \max_{1 \le i \le n} w([x_i])$$

Example

- A box $[\boldsymbol{x}] = [x_1] \times [x_2]$ of $\mathbb{IR}^2$



- The set of all boxes of $\mathbb{R}^n$ is denoted by $\mathbb{IR}^n$.

Interval vectors

Definition

- Interval vector (or box) is a Cartesian product of intervals

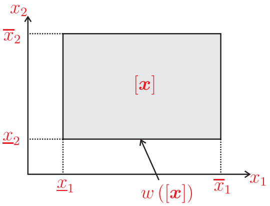
$$[\boldsymbol{x}] = [x_1] \times [x_2] \times \ldots \times [x_n] = ([x_1], [x_2], \ldots, [x_n])^{\mathsf{T}}$$

- Width

$$w([\boldsymbol{x}]) = \max_{1 \leq i \leq n} w([x_i])$$

Example

- A box $[\boldsymbol{x}] = [x_1] \times [x_2]$ of $\mathbb{IR}^2$



- The set of all boxes of $\mathbb{R}^n$ is denoted by $\mathbb{IR}^n$.

Interval vectors

Definition

- Interval vector (or box) is a Cartesian product of intervals

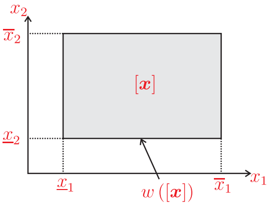
$$[\boldsymbol{x}] = [x_1] \times [x_2] \times \ldots \times [x_n] = ([x_1], [x_2], \ldots, [x_n])^{\mathsf{T}}$$

- Width

$$w([\boldsymbol{x}]) = \max_{1 \leq i \leq n} w([x_i])$$

Example

- A box $[\boldsymbol{x}] = [x_1] \times [x_2]$ of $\mathbb{IR}^2$



- The set of all boxes of $\mathbb{R}^n$ is denoted by $\mathbb{IR}^n$.

Interval vectors

Definition

- Interval vector (or box) is a Cartesian product of intervals

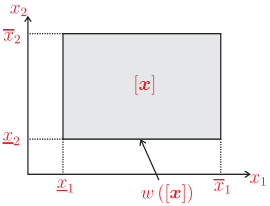
$$[\boldsymbol{x}] = [x_1] \times [x_2] \times \ldots \times [x_n] = ([x_1], [x_2], \ldots, [x_n])^{\mathsf{T}}$$

- Width

$$w([\boldsymbol{x}]) = \max_{1 \leq i \leq n} w([x_i])$$

Example

- A box $[\boldsymbol{x}] = [x_1] \times [x_2]$ of $\mathbb{IR}^2$



- The set of all boxes of $\mathbb{R}^n$ is denoted by $\mathbb{IR}^n$.

- Classical operations on vectors trivially extend to *interval vectors*

$$\begin{aligned}\alpha\,[\boldsymbol{x}] &= (\alpha\,[x_1]) \times \ldots \times (\alpha\,[x_n]) \\ [\boldsymbol{x}]^{\mathsf{T}}\,.\,[\boldsymbol{y}] &= [x_1]\,.\,[y_1] + \ldots + [x_n]\,.\,[y_n] \\ [\boldsymbol{x}] + [\boldsymbol{y}] &= ([x_1] + [y_1]) \times \ldots \times ([x_n] + [y_n])\end{aligned}$$

  and *interval matrices*

Inclusion functions

Definition

- $[\boldsymbol{f}]$ is an inclusion function for $\boldsymbol{f}$ if

$$\forall [\boldsymbol{x}] \in \mathbb{IR}^n \quad , \quad \boldsymbol{f}([\boldsymbol{x}]) \subset [\boldsymbol{f}]([\boldsymbol{x}]).$$

- f may be defined by an algorithm or even by a differential equation
  - Infinitely many inclusion functions for the same function

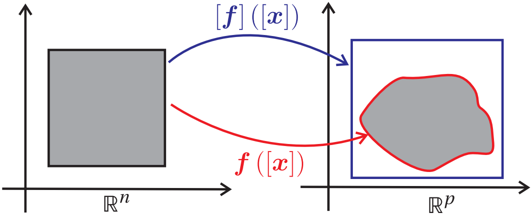
Inclusion functions

Definition

- $[\boldsymbol{f}]$ is an inclusion function for $\boldsymbol{f}$ if

$$\forall [\boldsymbol{x}] \in \mathbb{IR}^n \quad , \quad \boldsymbol{f}([\boldsymbol{x}]) \subset [\boldsymbol{f}]([\boldsymbol{x}]).$$

- $\boldsymbol{f}$ may be defined by an algorithm or even by a differential equation
  - Infinitely many inclusion functions for the same function



Subpavings

Intervals and boxes are not general enough
to describe all sets $\mathbb{S}$ of interest

$\Downarrow$

Motivates the introduction of subpavings

Subpaving of $[x]$ = union of nonoverlapping subboxes of $[x]$

Subpavings

Intervals and boxes are not general enough
to describe all sets $\mathbb{S}$ of interest

$\Downarrow$

Motivates the introduction of subpavings
Subpaving of $[\boldsymbol{x}]$ = union of nonoverlapping subboxes of $[\boldsymbol{x}]$

Inner and outer approximations

- If subpaving $\underline{\mathbb{S}}$ and $\overline{\mathbb{S}}$ are such that

$$\underline{\mathbb{S}} \subset \mathbb{S} \subset \overline{\mathbb{S}}$$

then $\mathbb{S}$ is bracketed between inner and outer approximations.

- The distance between $\underline{\mathbb{S}}$ and $\overline{\mathbb{S}}$ gives an indication of the quality of the approximation of $\mathbb{S}$
- Computation on subpavings
  - allows to approximate computation on compact sets
  - basic ingredient of the basin capture algorithm to be presented

Inner and outer approximations

- If subpaving $\underline{\mathbb{S}}$ and $\overline{\mathbb{S}}$ are such that

$$\underline{\mathbb{S}} \subset \mathbb{S} \subset \overline{\mathbb{S}}$$

then $\mathbb{S}$ is bracketed between inner and outer approximations.

- The distance between $\underline{\mathbb{S}}$ and $\overline{\mathbb{S}}$ gives an indication of the quality of the approximation of $\mathbb{S}$
- Computation on subpavings
  - allows to approximate computation on compact sets
  - basic ingredient of the basin capture algorithm to be presented

Inner and outer approximations

- If subpaving $\underline{\mathbb{S}}$ and $\overline{\mathbb{S}}$ are such that

$$\underline{\mathbb{S}} \subset \mathbb{S} \subset \overline{\mathbb{S}}$$

then $\mathbb{S}$ is bracketed between inner and outer approximations.

- The distance between $\underline{\mathbb{S}}$ and $\overline{\mathbb{S}}$ gives an indication of the quality of the approximation of $\mathbb{S}$
- Computation on subpavings
  - allows to approximate computation on compact sets
  - basic ingredient of the basin capture algorithm to be presented

Message

Interval Analysis allows guaranteed results to be obtained

$$\Downarrow$$

Considerable advantage over usual numerical methods

Guaranteed numerical integration

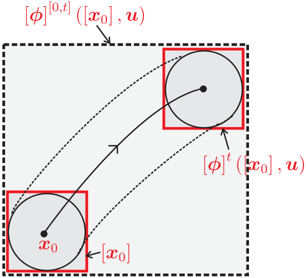
Interval flow

The *inclusion function of the flow* is a function

$$[\boldsymbol{\phi}]^{[t]} : \begin{cases} \mathbb{IR} \times \mathbb{IR}^n \times \mathbb{IR}^m & \to & \mathbb{IR}^n \\ ([\boldsymbol{x}], [\boldsymbol{u}]) & \to & [\boldsymbol{\phi}]^{[t]}([\boldsymbol{x}], [\boldsymbol{u}]) \end{cases}$$

such that $\forall t \in [t], \forall \boldsymbol{x} \in [\boldsymbol{x}], \boldsymbol{u} \in \mathbb{U},\ \boldsymbol{\phi}^t(\boldsymbol{x}, \boldsymbol{u}) \in [\boldsymbol{\phi}]^{[t]}([\boldsymbol{x}], [\boldsymbol{u}])$

- Guaranteed numerical integration is based on Picard Theorem and Taylor expansion
- Software tools are available to compute Guaranteed numerical integration
  - e.g., AWA, COSY or VNODE



Guaranteed numerical integration

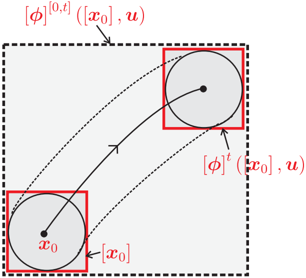
Interval flow

The *inclusion function of the flow* is a function

$$[\phi]^{[t]} : \begin{cases} \mathbb{IR} \times \mathbb{IR}^n \times \mathbb{IR}^m & \rightarrow & \mathbb{IR}^n \\ ([\boldsymbol{x}], [\boldsymbol{u}]) & \rightarrow & [\phi]^{[t]}([\boldsymbol{x}], [\boldsymbol{u}]) \end{cases}$$

such that $\forall t \in [t], \forall \boldsymbol{x} \in [\boldsymbol{x}], \boldsymbol{u} \in \mathbb{U},\ \phi^t(\boldsymbol{x}, \boldsymbol{u}) \in [\phi]^{[t]}([\boldsymbol{x}], [\boldsymbol{u}])$

- Guaranteed numerical integration is based on Picard Theorem and Taylor expansion
- Software tools are available to compute Guaranteed numerical integration
  - *e.g.*, AWA, COSY or VNODE



Algorithm

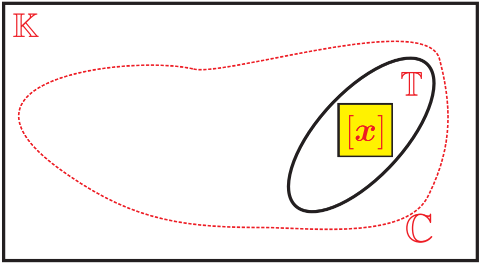
The capture basin algorithm is based on the following properties

(i) $[\boldsymbol{x}] \subset \mathbb{T} \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(ii) $[\boldsymbol{x}] \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$

(iii) $([\phi]^t([\boldsymbol{x}], \mathbf{u}) \subset \mathbb{C}^- \wedge [\phi]^{[0,t]}([\boldsymbol{x}], \mathbf{u}) \subset \mathbb{K}) \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(iv) $[\phi]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{C}^+ = \emptyset \wedge [\phi]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$



Algorithm

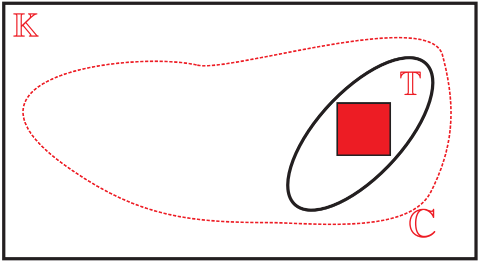
The capture basin algorithm is based on the following properties

(i) $[\boldsymbol{x}] \subset \mathbb{T} \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(ii) $[\boldsymbol{x}] \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$

(iii) $\left([\phi]^t([\boldsymbol{x}], \mathbf{u}) \subset \mathbb{C}^- \wedge [\phi]^{[0,t]}([\boldsymbol{x}], \mathbf{u}) \subset \mathbb{K}\right) \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(iv) $[\phi]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{C}^+ = \emptyset \wedge [\phi]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$



Algorithm

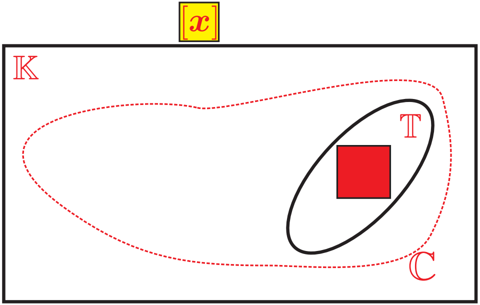
The capture basin algorithm is based on the following properties

(i) $[\boldsymbol{x}] \subset \mathbb{T} \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(ii) $[\boldsymbol{x}] \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$

(iii) $\left([\phi]^t\left([\boldsymbol{x}], \mathbf{u}\right) \subset \mathbb{C}^- \wedge [\phi]^{[0,t]}\left([\boldsymbol{x}], \mathbf{u}\right) \subset \mathbb{K}\right) \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(iv) $[\phi]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{C}^+ = \emptyset \wedge [\phi]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$



Algorithm

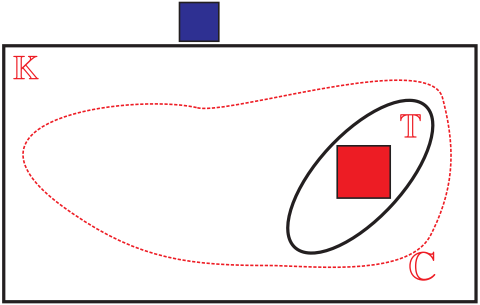
The capture basin algorithm is based on the following properties

(i) $[\boldsymbol{x}] \subset \mathbb{T} \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(ii) $[\boldsymbol{x}] \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$

(iii) $([\boldsymbol{\phi}]^t ([\boldsymbol{x}], \mathbf{u}) \subset \mathbb{C}^- \wedge [\boldsymbol{\phi}]^{[0,t]} ([\boldsymbol{x}], \mathbf{u}) \subset \mathbb{K}) \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(iv) $[\boldsymbol{\phi}]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{C}^+ = \emptyset \wedge [\boldsymbol{\phi}]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$



Algorithm

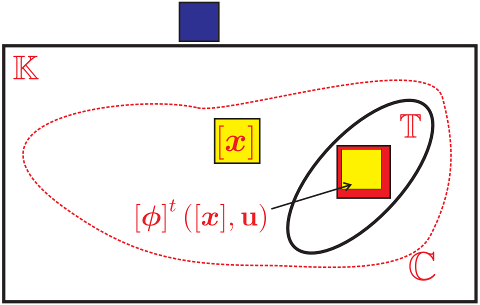
The capture basin algorithm is based on the following properties

(i) $[\boldsymbol{x}] \subset \mathbb{T} \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(ii) $[\boldsymbol{x}] \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$

(iii) $\left([\boldsymbol{\phi}]^t\left([\boldsymbol{x}], \mathbf{u}\right) \subset \mathbb{C}^- \wedge [\phi]^{[0,t]}\left([\boldsymbol{x}], \mathbf{u}\right) \subset \mathbb{K}\right) \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(iv) $[\phi]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{C}^+ = \emptyset \wedge [\phi]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$



Algorithm

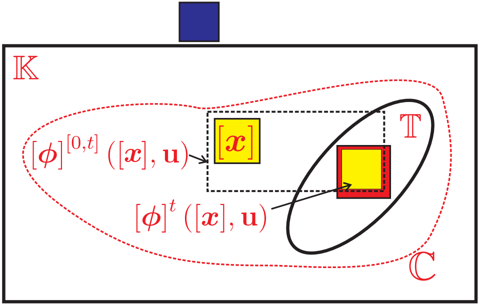
The capture basin algorithm is based on the following properties

(i) $[\boldsymbol{x}] \subset \mathbb{T} \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(ii) $[\boldsymbol{x}] \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$

(iii) $\left([\boldsymbol{\phi}]^{t}([\boldsymbol{x}], \mathbf{u}) \subset \mathbb{C}^{-} \wedge [\boldsymbol{\phi}]^{[0,t]}([\boldsymbol{x}], \mathbf{u}) \subset \mathbb{K}\right) \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(iv) $[\boldsymbol{\phi}]^{t}([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{C}^{+} = \emptyset \wedge [\boldsymbol{\phi}]^{t}([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$



Algorithm

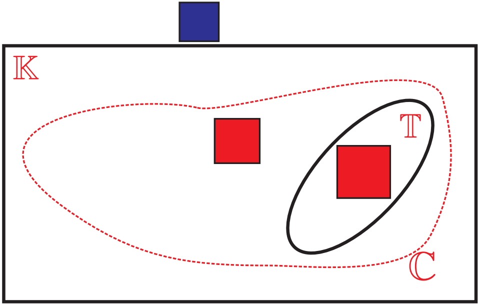
The capture basin algorithm is based on the following properties

(i) $[\boldsymbol{x}] \subset \mathbb{T} \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(ii) $[\boldsymbol{x}] \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$

(iii) $([\boldsymbol{\phi}]^t([\boldsymbol{x}], \mathbf{u}) \subset \mathbb{C}^- \wedge [\boldsymbol{\phi}]^{[0,t]}([\boldsymbol{x}], \mathbf{u}) \subset \mathbb{K}) \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(iv) $[\boldsymbol{\phi}]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{C}^+ = \emptyset \wedge [\boldsymbol{\phi}]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$



Algorithm

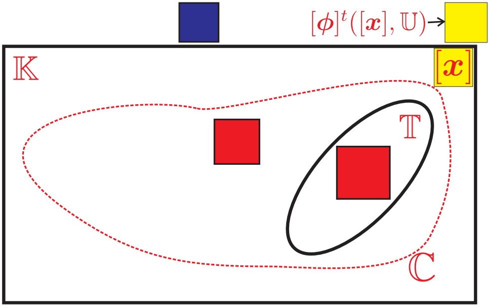
The capture basin algorithm is based on the following properties

(i) $[\boldsymbol{x}] \subset \mathbb{T} \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(ii) $[\boldsymbol{x}] \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$

(iii) $([\boldsymbol{\phi}]^t([\boldsymbol{x}], \mathbf{u}) \subset \mathbb{C}^- \wedge [\boldsymbol{\phi}]^{[0,t]}([\boldsymbol{x}], \mathbf{u}) \subset \mathbb{K}) \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(iv) $[\boldsymbol{\phi}]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{C}^+ = \emptyset \wedge [\boldsymbol{\phi}]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$



Algorithm

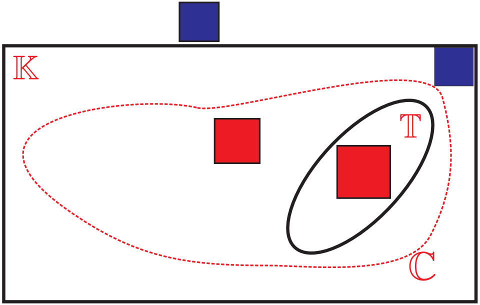
The capture basin algorithm is based on the following properties

(i) $[\boldsymbol{x}] \subset \mathbb{T} \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

(ii) $[\boldsymbol{x}] \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$

(iii) $([\boldsymbol{\phi}]^t ([\boldsymbol{x}], \mathbf{u}) \subset \mathbb{C}^- \wedge [\boldsymbol{\phi}]^{[0,t]} ([\boldsymbol{x}], \mathbf{u}) \subset \mathbb{K}) \Rightarrow [\boldsymbol{x}] \subset \mathbb{C}$

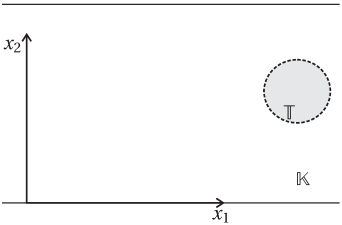
(iv) $[\boldsymbol{\phi}]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{C}^+ = \emptyset \wedge [\boldsymbol{\phi}]^t([\boldsymbol{x}], \mathbb{U}) \cap \mathbb{K} = \emptyset \Rightarrow [\boldsymbol{x}] \cap \mathbb{C} = \emptyset$



Illustration

Zermelo navigation problem

- In control theory, Zermelo has described the problem of a boat which wants to reach an island from the bank of a river with strong currents.
  - The magnitude and direction of the currents are known as a function of position.
- Let $\mathbb{T} \triangleq \mathcal{B}(0, r)$ with $r = 1$ be the island and we set $\mathbb{K} = [-8, 8] \times [-4, 4]$, where $\mathbb{K}$ represents the river.

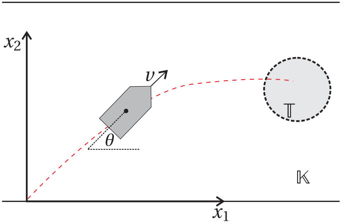


Illustration

Zermelo navigation problem

- Let us consider the following dynamics for the boat

$$\begin{cases} x_1'(t) & = & v\cos(\theta) \\ x_2'(t) & = & v\sin(\theta) \end{cases},$$



where the controls $v \in [0, 0.8]$ and $\theta \in [-\pi, \pi]$.

- The currents are represented by an autonomous vector field, then the global dynamic (boat and currents) is given by

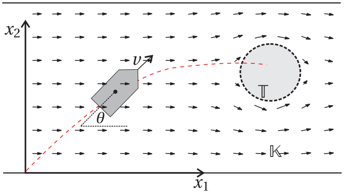
$$\begin{cases} x_1'(t) & = & 1 + \dfrac{x_2^2 - x_1^2}{(x_1^2 + x_2^2)^2} + v\cos(\theta) \\ x_2'(t) & = & \dfrac{-2x_1x_2}{(x_1^2 + x_2^2)^2} + v\sin(\theta) \end{cases}.$$

Illustration

Zermelo navigation problem

- Let us consider the following dynamics for the boat where the controls $v \in [0, 0.8]$ and $\theta \in [-\pi, \pi]$.
- The currents are represented by an autonomous vector field, then the global dynamic (boat and currents) is given by

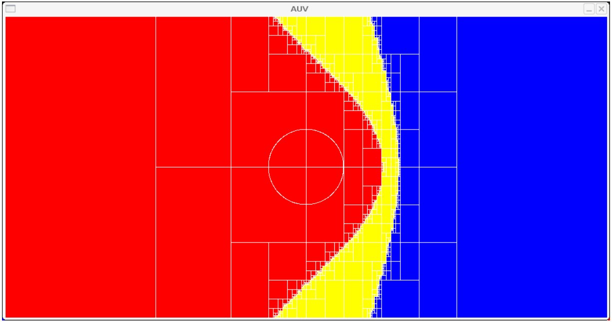
$$\left\{ \begin{array}{rcl} x_1'(t) & = & 1 + \dfrac{x_2^2 - x_1^2}{(x_1^2 + x_2^2)^2} + v\cos(\theta) \\ x_2'(t) & = & \dfrac{-2x_1x_2}{(x_1^2 + x_2^2)^2} + v\sin(\theta) \end{array} \right. .$$



Illustration

Results

The figure shows the result of the capture basin algorithm,



where

- the circle delimits the border of the target $\mathbb{T}$;
- $\mathbb{C}^-$ corresponds to the union of all red boxes ;
- $\mathbb{C}^+$ corresponds to the union of both red and yellow boxes ;
- finally, $\mathbb{C}^- \subset \mathbb{C} \subset \mathbb{C}^+$.

Conclusions

- A new approach to deal with capture basin problems is presented.
  - This approach uses interval analysis to compute an inner an outer approximation of the capture basin for a given target.
- To fill out this work, different perspectives appear. It could be interesting to tackle problems in significantly larger dimensions.
  - Constraint propagation techniques make it possible to push back this frontier and to deal with high dimensional problems (with more than 1000 variables for instance)
- We plan to combine our algorithm with graph theory and guaranteed numerical integration to compute a guaranteed control $\mathbf{u}$.

Conclusions

- A new approach to deal with capture basin problems is presented.
  - This approach uses interval analysis to compute an inner an outer approximation of the capture basin for a given target.
- To fill out this work, different perspectives appear. It could be interesting to tackle problems in significantly larger dimensions.
  - Constraint propagation techniques make it possible to push back this frontier and to deal with high dimensional problems (with more than 1000 variables for instance)
- We plan to combine our algorithm with graph theory and guaranteed numerical integration to compute a guaranteed control $\mathbf{u}$.

Conclusions

- A new approach to deal with capture basin problems is presented.
    - This approach uses interval analysis to compute an inner an outer approximation of the capture basin for a given target.
- To fill out this work, different perspectives appear. It could be interesting to tackle problems in significantly larger dimensions.
    - Constraint propagation techniques make it possible to push back this frontier and to deal with high dimensional problems (with more than 1000 variables for instance)
- We plan to combine our algorithm with graph theory and guaranteed numerical integration to compute a guaranteed control $\mathbf{u}$.

Conclusions

- A new approach to deal with capture basin problems is presented.
  - This approach uses interval analysis to compute an inner an outer approximation of the capture basin for a given target.
- To fill out this work, different perspectives appear. It could be interesting to tackle problems in significantly larger dimensions.
  - Constraint propagation techniques make it possible to push back this frontier and to deal with high dimensional problems (with more than 1000 variables for instance)
- We plan to combine our algorithm with graph theory and guaranteed numerical integration to compute a guaranteed control $\mathbf{u}$.

Conclusions

- A new approach to deal with capture basin problems is presented.
  - This approach uses interval analysis to compute an inner an outer approximation of the capture basin for a given target.
- To fill out this work, different perspectives appear. It could be interesting to tackle problems in significantly larger dimensions.
  - Constraint propagation techniques make it possible to push back this frontier and to deal with high dimensional problems (with more than 1000 variables for instance)
- We plan to combine our algorithm with graph theory and guaranteed numerical integration to compute a guaranteed control $\mathbf{u}$.