Reliable SLAM in ambiguous environments

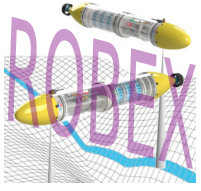
SWIM

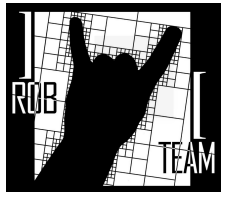
30th June 2025, Rennes, France

Simon Rohou

ENSTA, Lab-STICC, Brest, France

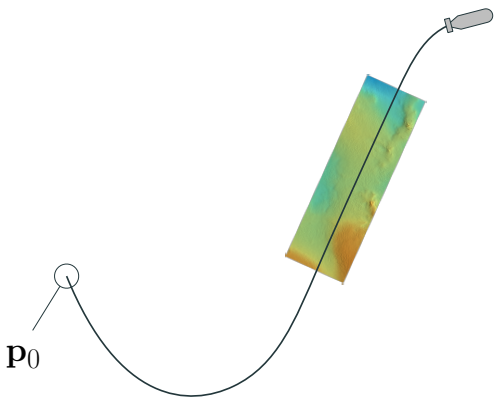
Joint work with Luc Jaulin, Peter Franek and Michel Legris

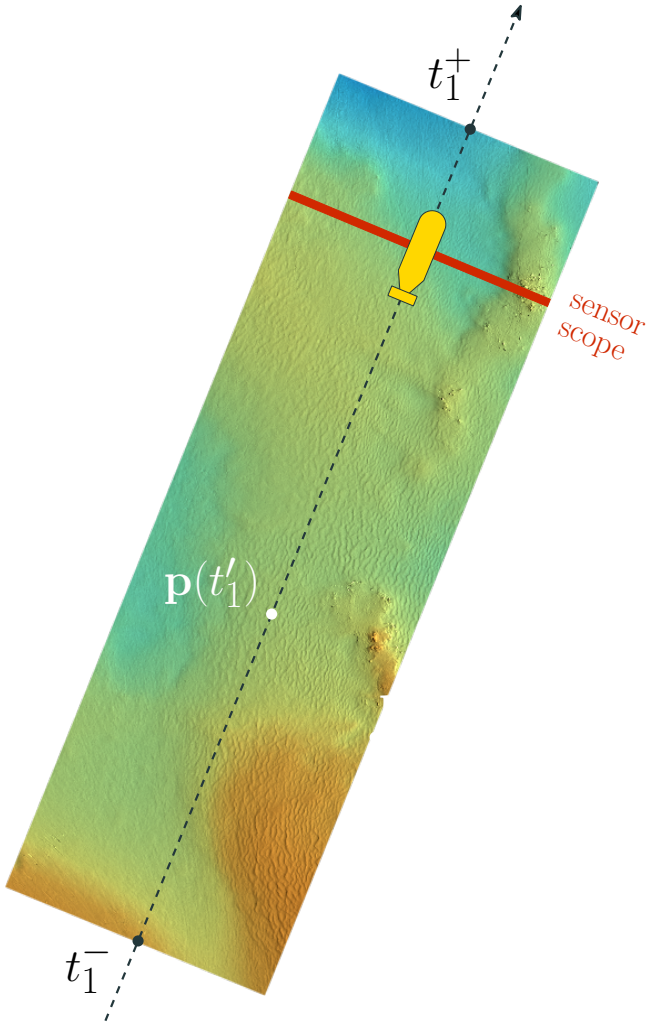




The SLAM problem

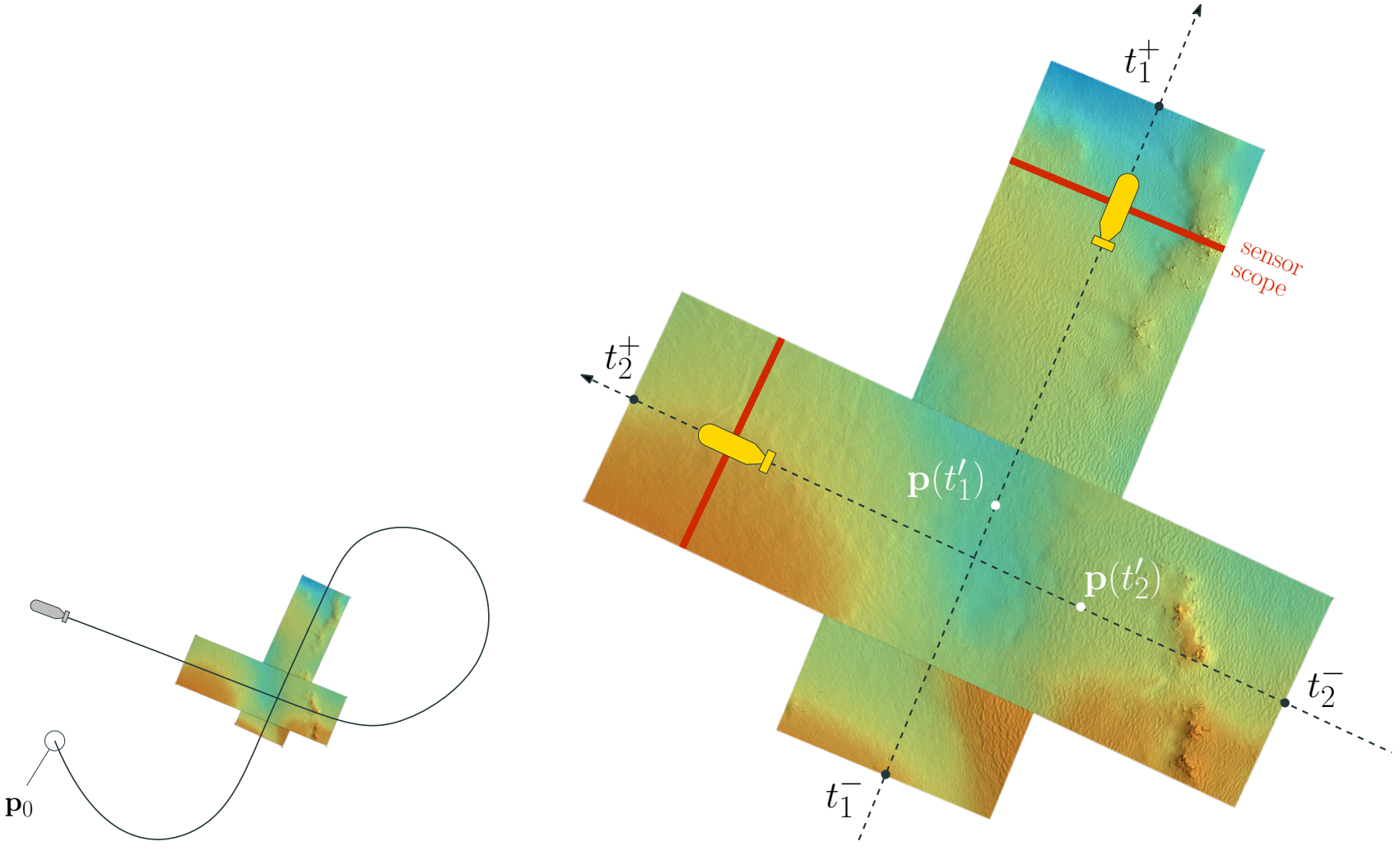
Loop closure with ground perception





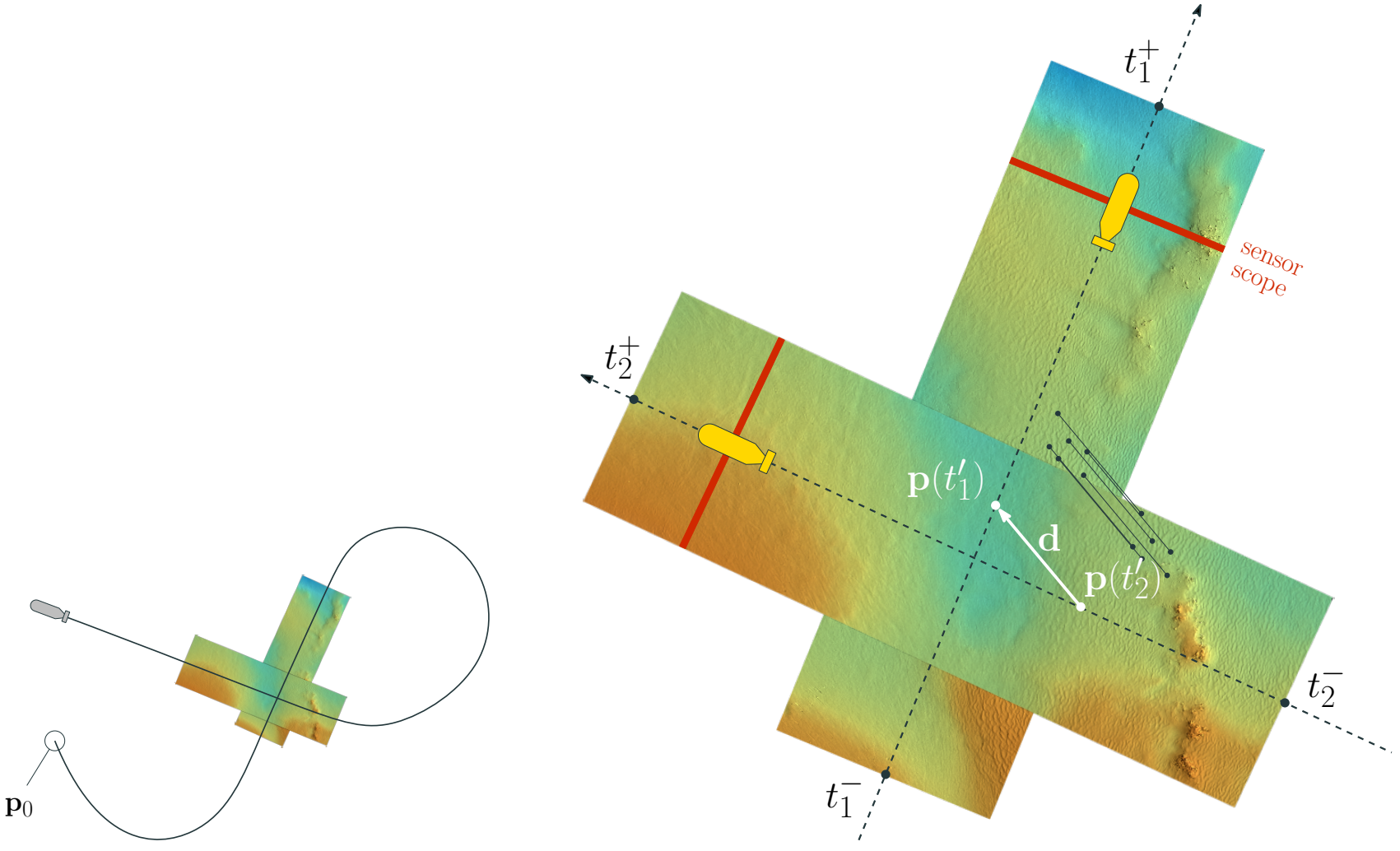
Robot exploring an environment...

Loop closure with ground perception



Robot exploring an environment... coming back to a previous position...

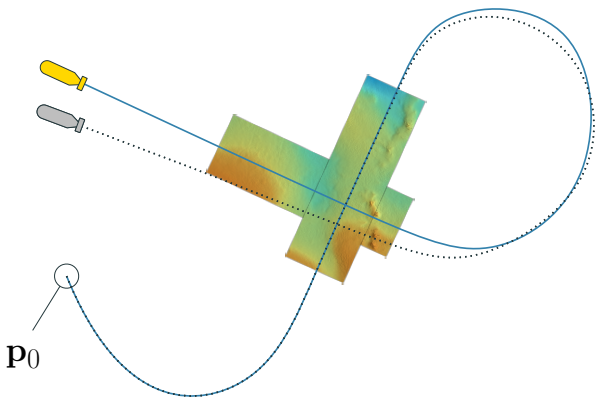
Loop closure with ground perception

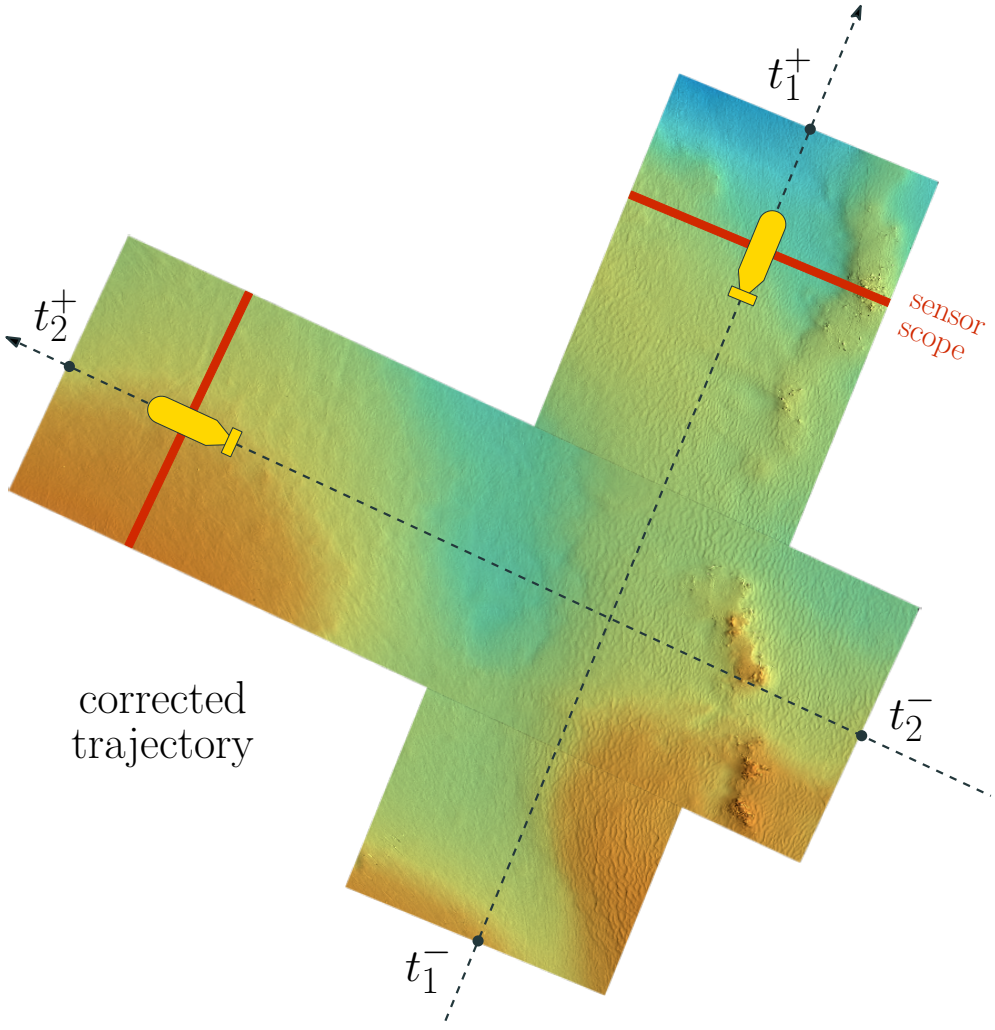


Robot exploring an environment... coming back to a previous position... detecting loop closure...

Loop closure with ground perception

SLAM:
Simultaneous
Localization
And Mapping



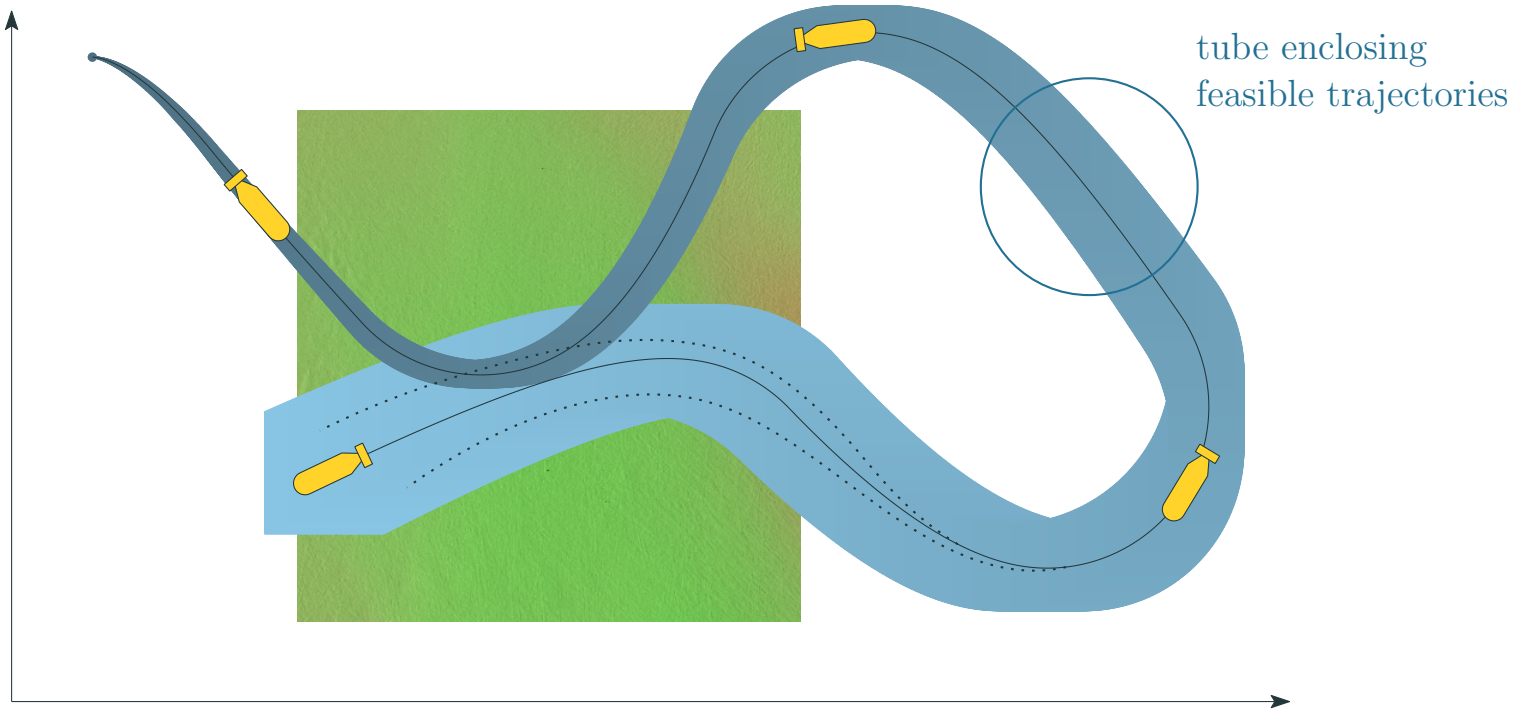


Robot exploring an environment... coming back to a previous position... detecting loop closure... correcting its trajectory.

Problem: doubtful loops and similar environments

What if we recognize a **wrong scene**?

- homogeneous environments $\Longrightarrow$ similar observations
- strong positioning drift $\Longrightarrow$ false loop detections

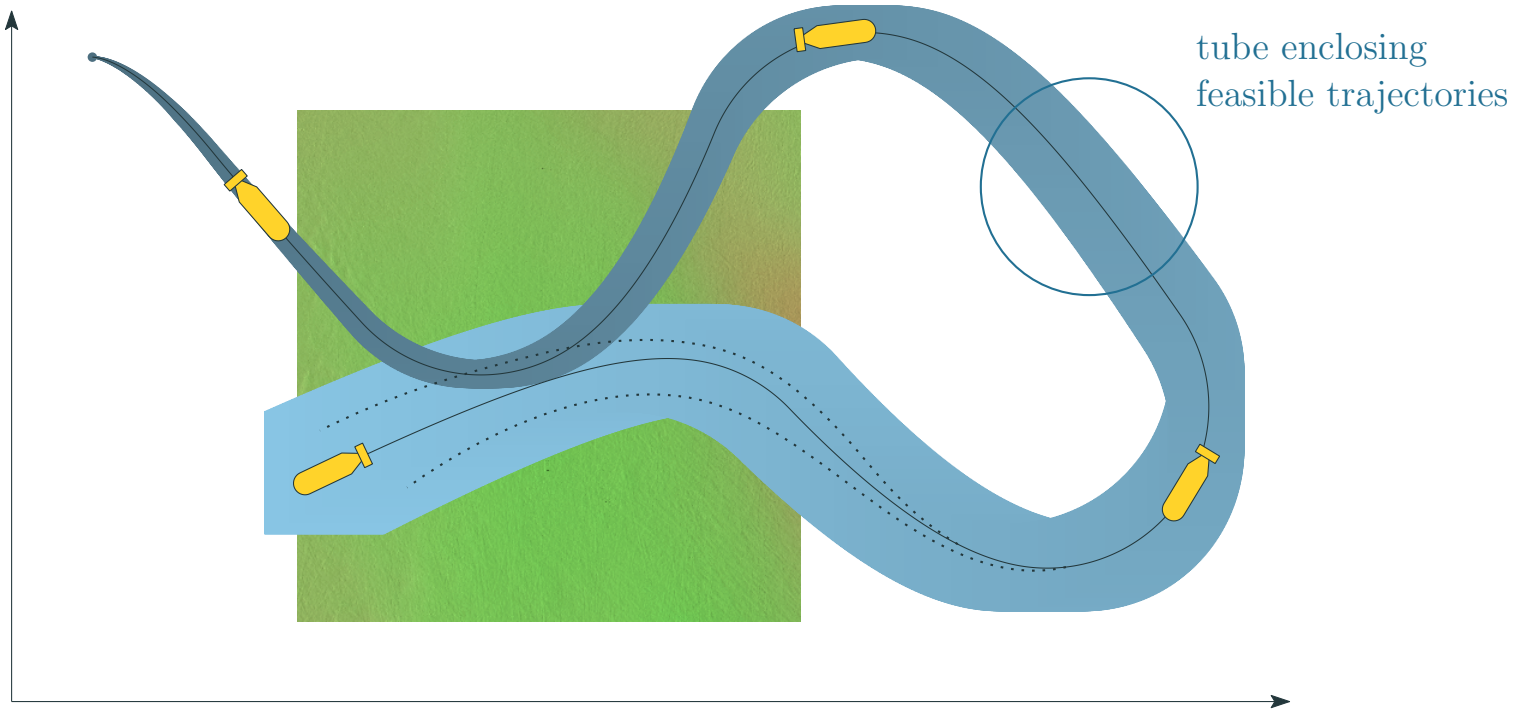


Doubtful loop: the actual trajectory did not loop, but the evolution uncertainties and the homogeneity of the environment may lead to an incorrect correction.

Problem: doubtful loops and similar environments

What if we recognize a **wrong scene**?

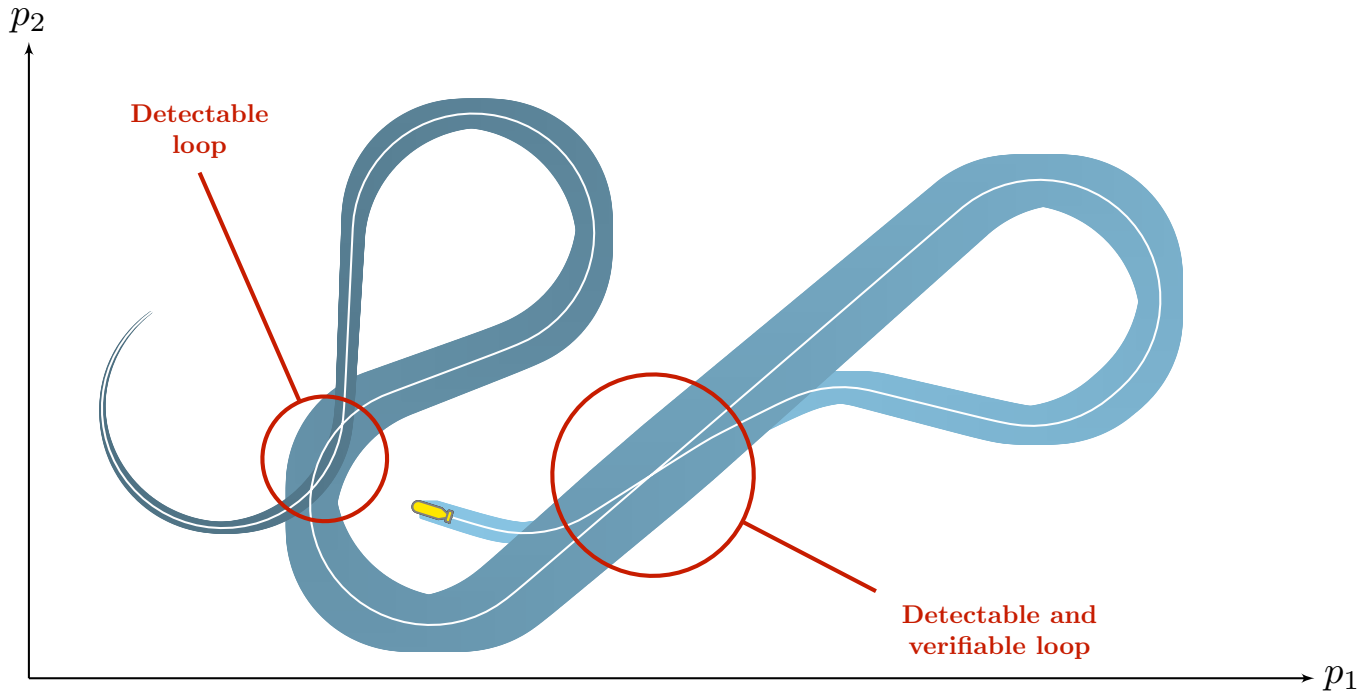
- homogeneous environments $\Longrightarrow$ similar observations
- strong positioning drift $\Longrightarrow$ false loop detections



Doubtful loop: the actual trajectory did not loop, but the evolution uncertainties and the homogeneity of the environment may lead to an incorrect correction. $\Longrightarrow$ *significant impact on the next SLAM iterations.*

Uncertainties: detection vs verification

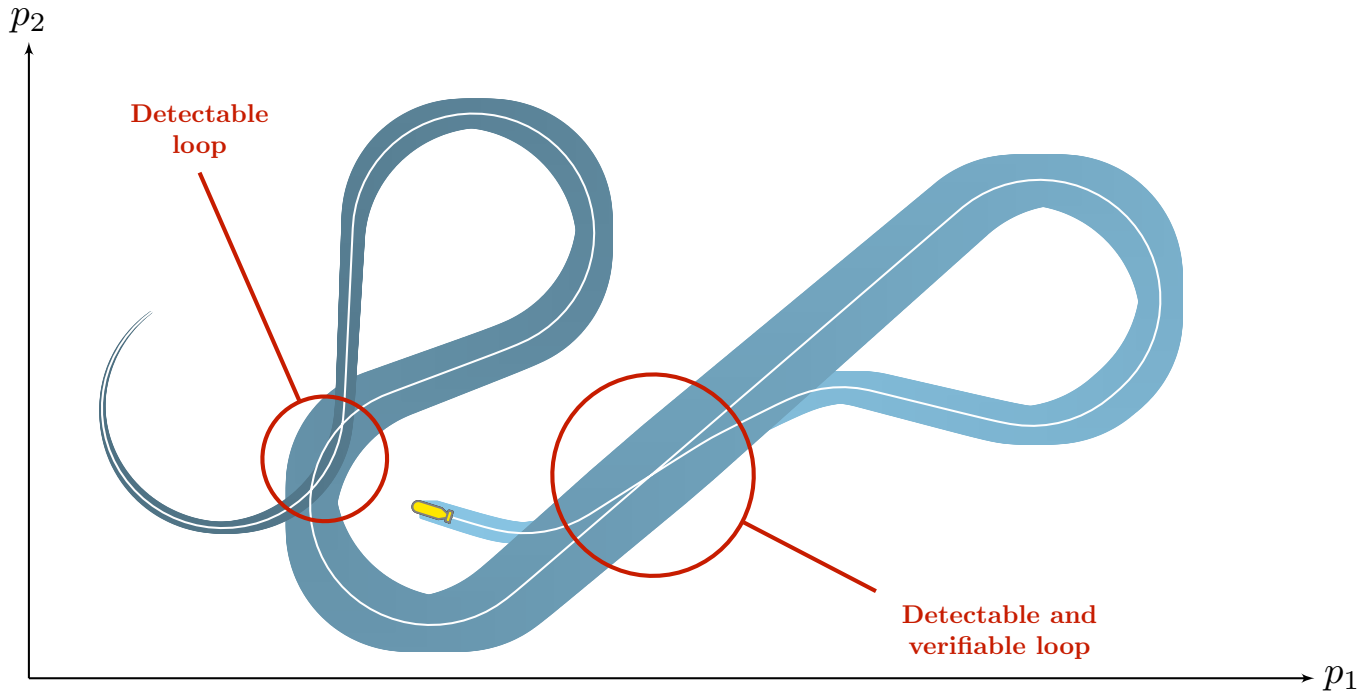
Uncertain trajectories are enclosed by **tubes**.



Only 1 loop can be verified – at least 2 feasible loops are detected.

Uncertainties: detection vs verification

Uncertain trajectories are enclosed by **tubes**.



Only 1 loop can be verified – at least 2 feasible loops are detected.

Need for **loop proof**:

- verify that a trajectory crosses itself at some point
- ..whatever the uncertainties describing this trajectory

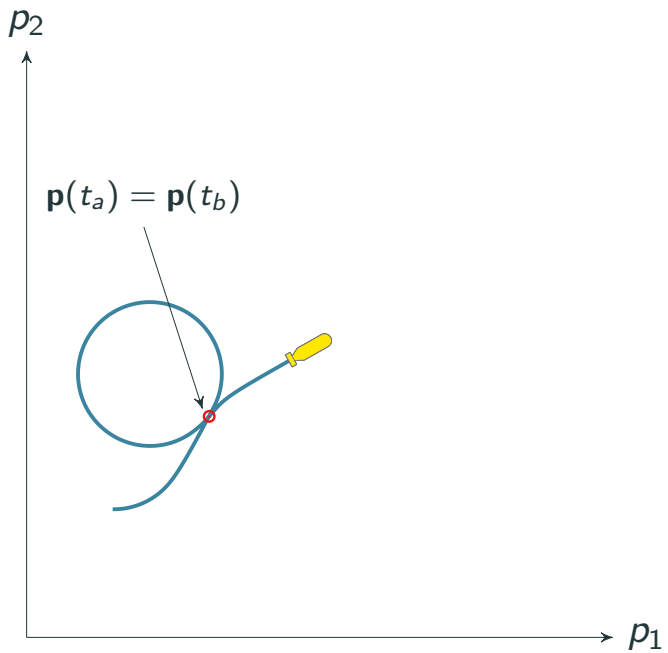
Set of feasible loops

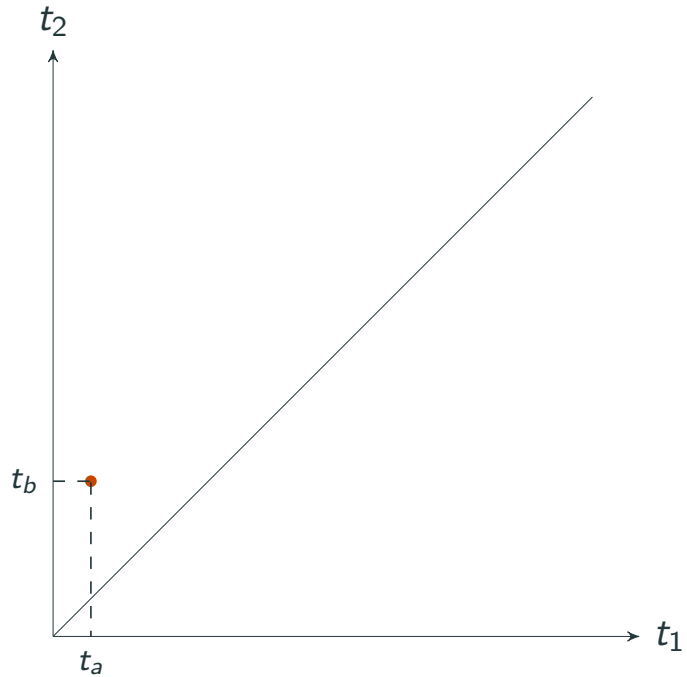
Definitions (Aubry, 2013)

- robot position: $\mathbf{p} = (x, y)^\intercal \in \mathbb{R}^2$
- 2D robot trajectory: $\mathbf{p}(t) : \mathbb{R} \to \mathbb{R}^2,\ t \in [t_0, t_f]$
- looped trajectory $\Leftrightarrow$ trajectory that crosses itself
  - $\mathbf{p}(t_1) = \mathbf{p}(t_2),\ t_1 \neq t_2$
  - 1 loop $\Leftrightarrow$ 1 t-pair (t_1, t_2)

Definitions (Aubry, 2013)

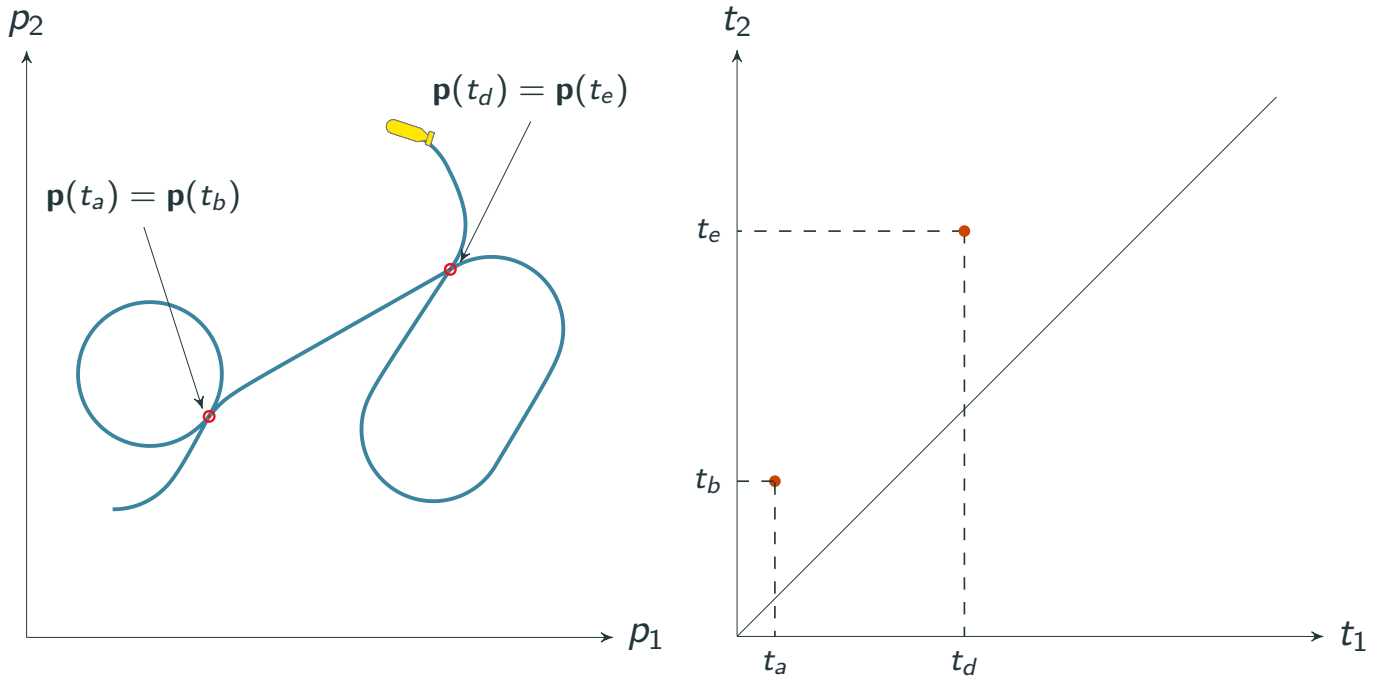
- robot position: $\mathbf{p} = (x, y)^\mathsf{T} \in \mathbb{R}^2$
- 2D robot trajectory: $\mathbf{p}(t) : \mathbb{R} \to \mathbb{R}^2,\ t \in [t_0, t_f]$
- looped trajectory $\Leftrightarrow$ trajectory that crosses itself
  - $\mathbf{p}(t_1) = \mathbf{p}(t_2),\ t_1 \neq t_2$
  - 1 loop $\Leftrightarrow$ 1 t-pair (t_1, t_2)





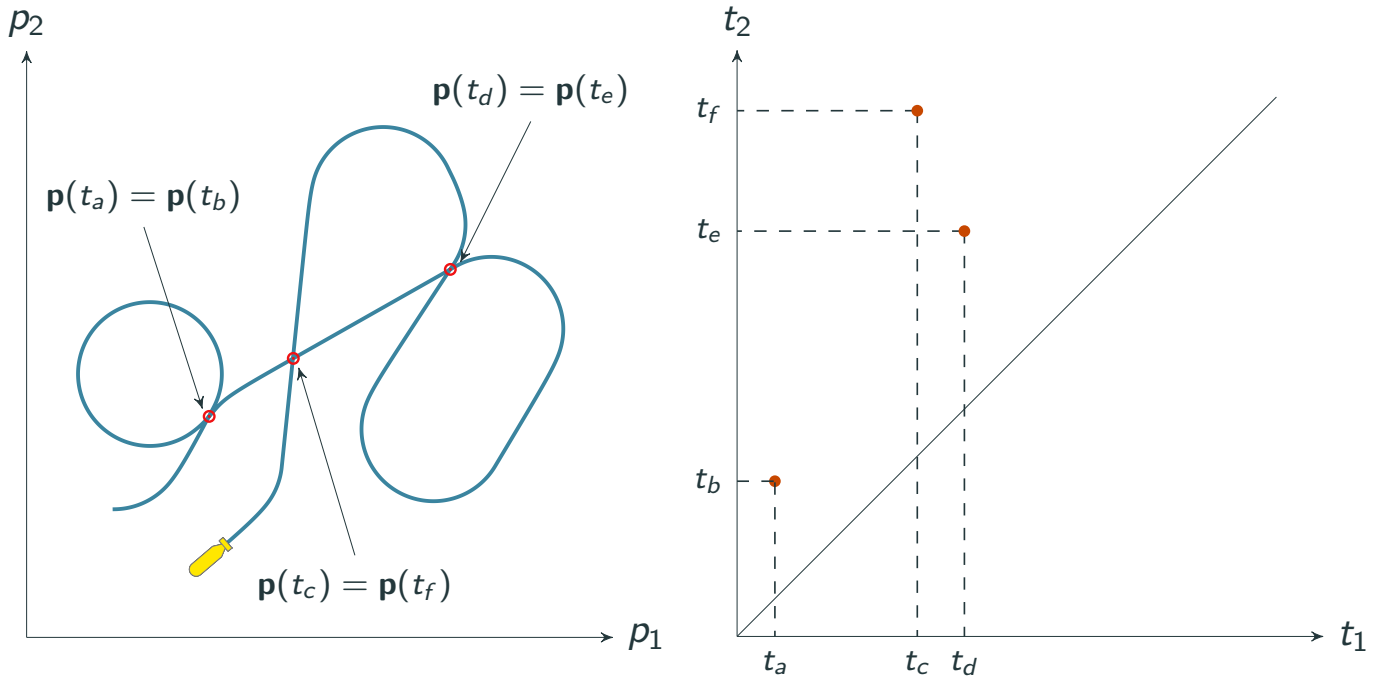
Definitions (Aubry, 2013)

- robot position: $\mathbf{p} = (x, y)^\mathsf{T} \in \mathbb{R}^2$
- 2D robot trajectory: $\mathbf{p}(t) : \mathbb{R} \to \mathbb{R}^2,\ t \in [t_0, t_f]$
- looped trajectory $\Leftrightarrow$ trajectory that crosses itself
  - $\mathbf{p}(t_1) = \mathbf{p}(t_2),\ t_1 \neq t_2$
  - 1 loop $\Leftrightarrow$ 1 t-pair (t_1, t_2)



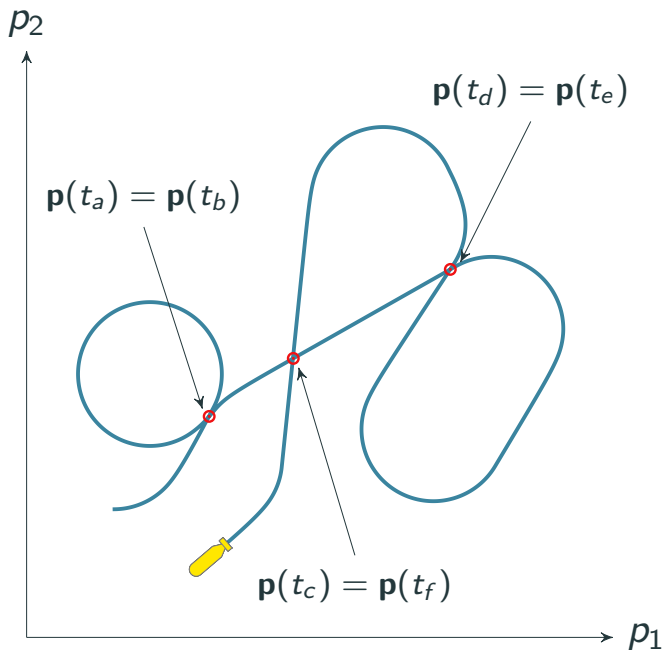
Definitions (Aubry, 2013)

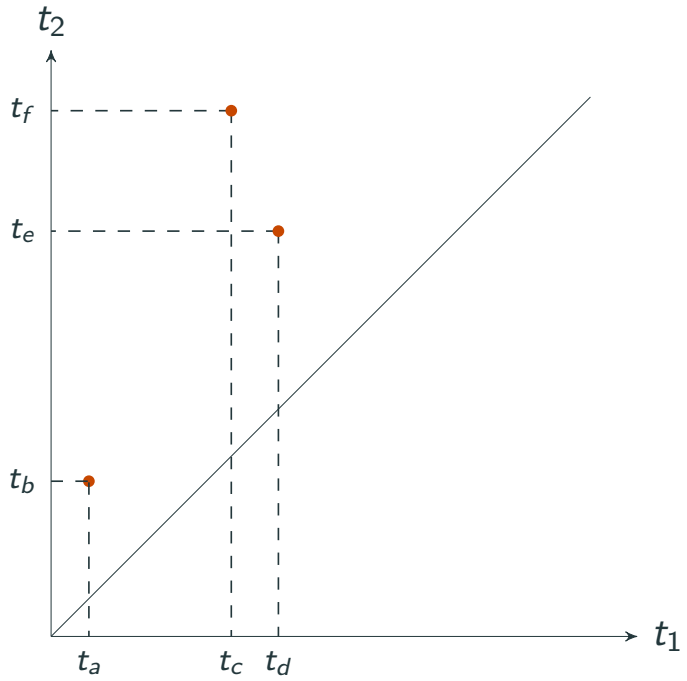
- robot position: $\mathbf{p} = (x, y)^\mathsf{T} \in \mathbb{R}^2$
- 2D robot trajectory: $\mathbf{p}(t) : \mathbb{R} \to \mathbb{R}^2,\ t \in [t_0, t_f]$
- looped trajectory $\Leftrightarrow$ trajectory that crosses itself
  - $\mathbf{p}(t_1) = \mathbf{p}(t_2),\ t_1 \neq t_2$
  - 1 loop $\Leftrightarrow$ 1 t-pair (t_1, t_2)



Definitions (Aubry, 2013)

- t-plane $\Leftrightarrow$ all feasible t-pairs $= [t_0, t_f]^2$
- loop set $\mathbb{T}^*$:
  - $\mathbb{T}^* = \left\{(t_1, t_2) \in [t_0, t_f]^2 \mid \mathbf{p}(t_1) = \mathbf{p}(t_2), t_1 < t_2\right\}$
- loop set of below example:
  - $\mathbb{T}^* = \{(t_a, t_b), (t_c, t_f), (t_d, t_e)\}$





Computing loops from robot sensors

Context: robot trajectory $\mathbf{p}(t)$ cannot be directly sensed.
Computation from speed measurements:

$$\mathbf{p}(t) = \int_{t_0}^{t} \mathbf{v}(\tau) d\tau + \mathbf{p}_0, \tag{1}$$

with $\mathbf{v}(t) \in \mathbb{R}^2$: robot velocity vector at time $t \in [t_0, t_f]$.

Computing loops from robot sensors

Context: robot trajectory $\mathbf{p}(t)$ cannot be directly sensed.
Computation from speed measurements:

$$\mathbf{p}(t) = \int_{t_0}^{t} \mathbf{v}(\tau) d\tau + \mathbf{p}_0, \tag{1}$$

with $\mathbf{v}(t) \in \mathbb{R}^2$: robot velocity vector at time $t \in [t_0, t_f]$.

Loop-set from velocity:

$$\begin{aligned} \mathbb{T}^* &= \left\{ (t_1, t_2) \in [t_0, t_f]^2 \mid \mathbf{p}^*(t_1) = \mathbf{p}^*(t_2), t_1 < t_2 \right\} && (2) \\ &= \left\{ (t_1, t_2) \in [t_0, t_f]^2 \mid \int_{t_1}^{t_2} \mathbf{v}^*(\tau) d\tau = \mathbf{0}, t_1 < t_2 \right\} && (3) \end{aligned}$$

Bounded-error context: $\mathbb{T}$

Actual loop-set $\mathbb{T}^*$ (error free):

$$\mathbb{T}^* = \left\{ (t_1, t_2) \mid \int_{t_1}^{t_2} \mathbf{v}^*(\tau) d\tau = \mathbf{0} \right\} \tag{4}$$

Bounded-error context: $\mathbb{T}$

Actual loop-set $\mathbb{T}^*$ (error free):

$$\mathbb{T}^* = \left\{ (t_1, t_2) \mid \int_{t_1}^{t_2} \mathbf{v}^*(\tau) d\tau = \mathbf{0} \right\} \tag{4}$$

Bounded-error context, assuming $\mathbf{v}^*(\cdot) \in [\mathbf{v}](\cdot)$:

$$\mathbb{T} = \left\{ (t_1, t_2) \mid \exists \mathbf{v}(\cdot) \in [\mathbf{v}](\cdot) \ , \int_{t_1}^{t_2} \mathbf{v}(\tau) d\tau = \mathbf{0} \right\} \tag{5}$$

Bounded-error context: $\mathbb{T}$

Actual loop-set $\mathbb{T}^*$ (error free):

$$\mathbb{T}^* = \left\{ (t_1, t_2) \mid \int_{t_1}^{t_2} \mathbf{v}^*(\tau) d\tau = \mathbf{0} \right\} \tag{4}$$

Bounded-error context, assuming $\mathbf{v}^*(\cdot) \in [\mathbf{v}](\cdot)$:

$$\mathbb{T} = \left\{ (t_1, t_2) \mid \exists \mathbf{v}(\cdot) \in [\mathbf{v}](\cdot) \ , \int_{t_1}^{t_2} \mathbf{v}(\tau) d\tau = \mathbf{0} \right\} \tag{5}$$

Set-membership approach:

$$\mathbb{T}^* \subset \mathbb{T} \subset [t_0, t_f]^2 \tag{6}$$

Algorithm for detecting loops [Aubry 2013]

Using a SIVIA-like algorithm, we define the following inclusion tests:

$$\left.\begin{array}{r} [t_1]-[t_2]\subset\mathbb{R}^- \\ \int_{[t_1]}^{[t_2]}\mathbf{v}^-(\tau)d\tau \leqslant \mathbf{0} \leqslant \int_{[t_1]}^{[t_2]}\mathbf{v}^+(\tau)d\tau \end{array}\right\} \Rightarrow [\mathbf{t}]\subset\mathbb{T}$$

$$\left.\begin{array}{r} [t_1]-[t_2]\subset\mathbb{R}^+ \\ \mathbf{0}\notin \int_{[t_1]}^{[t_2]}[\mathbf{v}](\tau)d\tau \end{array}\right\} \Rightarrow [\mathbf{t}]\cap\mathbb{T}=\varnothing$$

Example of application

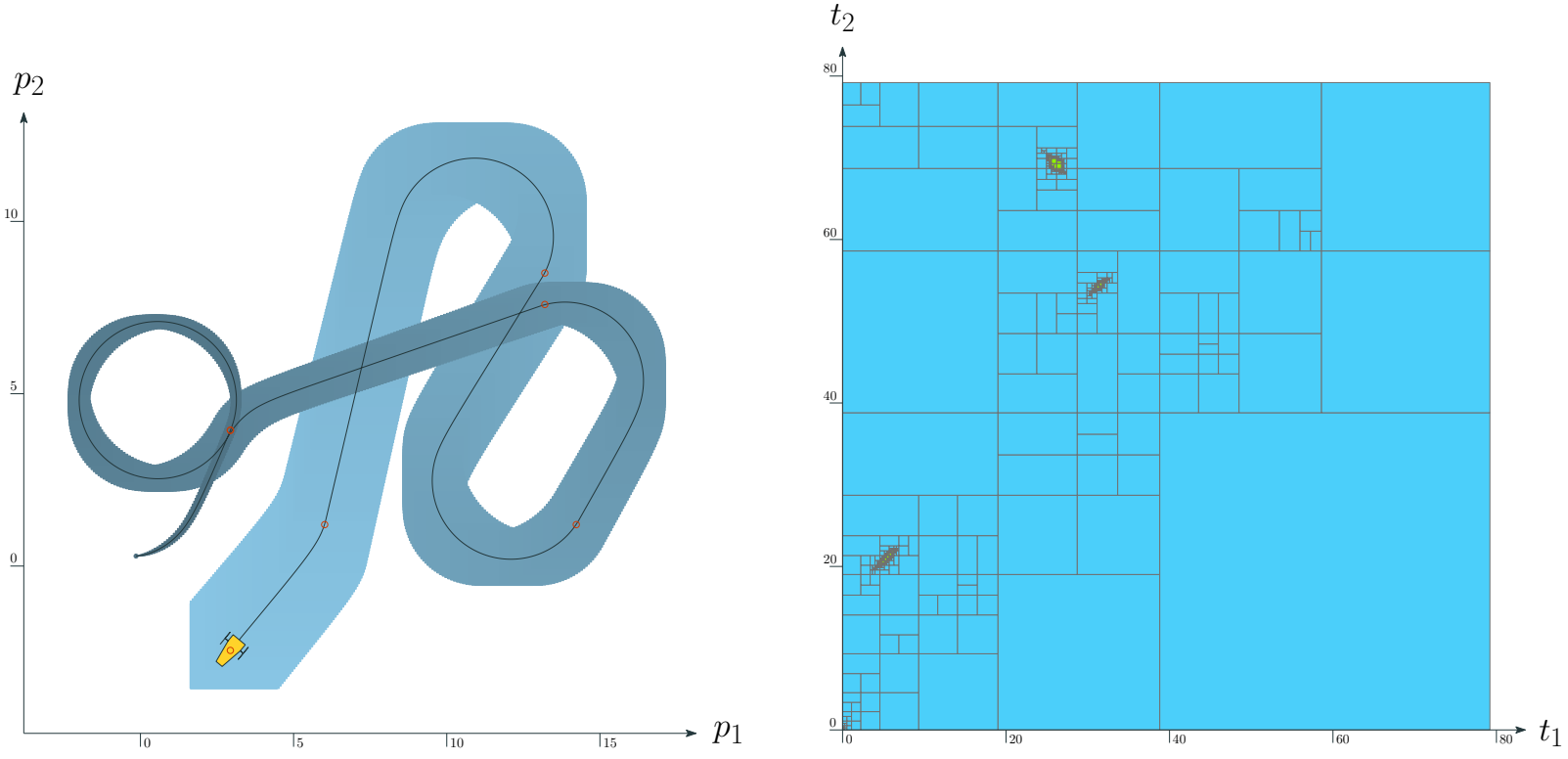
www.codac.io

Computing an enclosure $\mathbb{T}$ of the loop set using Codac:

```
BoolInterval test_loop(const IntervalVector& t)
{
    auto partial_integ = v.partial_integral(t[0],t[1]);

    if((t[0]-t[1]).is_strict_subset({0,oo})
          || !v.integral(t[0],t[1]).interior_contains({0,0})
          || !v(t[0]|t[1]).interior_contains({0,0}))
       return BoolInterval::FALSE;

    else if((t[0]-t[1]).is_strict_subset({-oo,0})
          && partial_integ.first.is_strict_subset({{-oo,0},{-oo,0}})
          && partial_integ.second.is_strict_subset({{0,oo},{0,oo}}))
       return BoolInterval::TRUE;

    else
       return BoolInterval::UNKNOWN;
}

IntervalVector T { {0,10}, {0,10} };
auto loops = regular_pave(T, test_loop, 1e-1);
```

Example of application

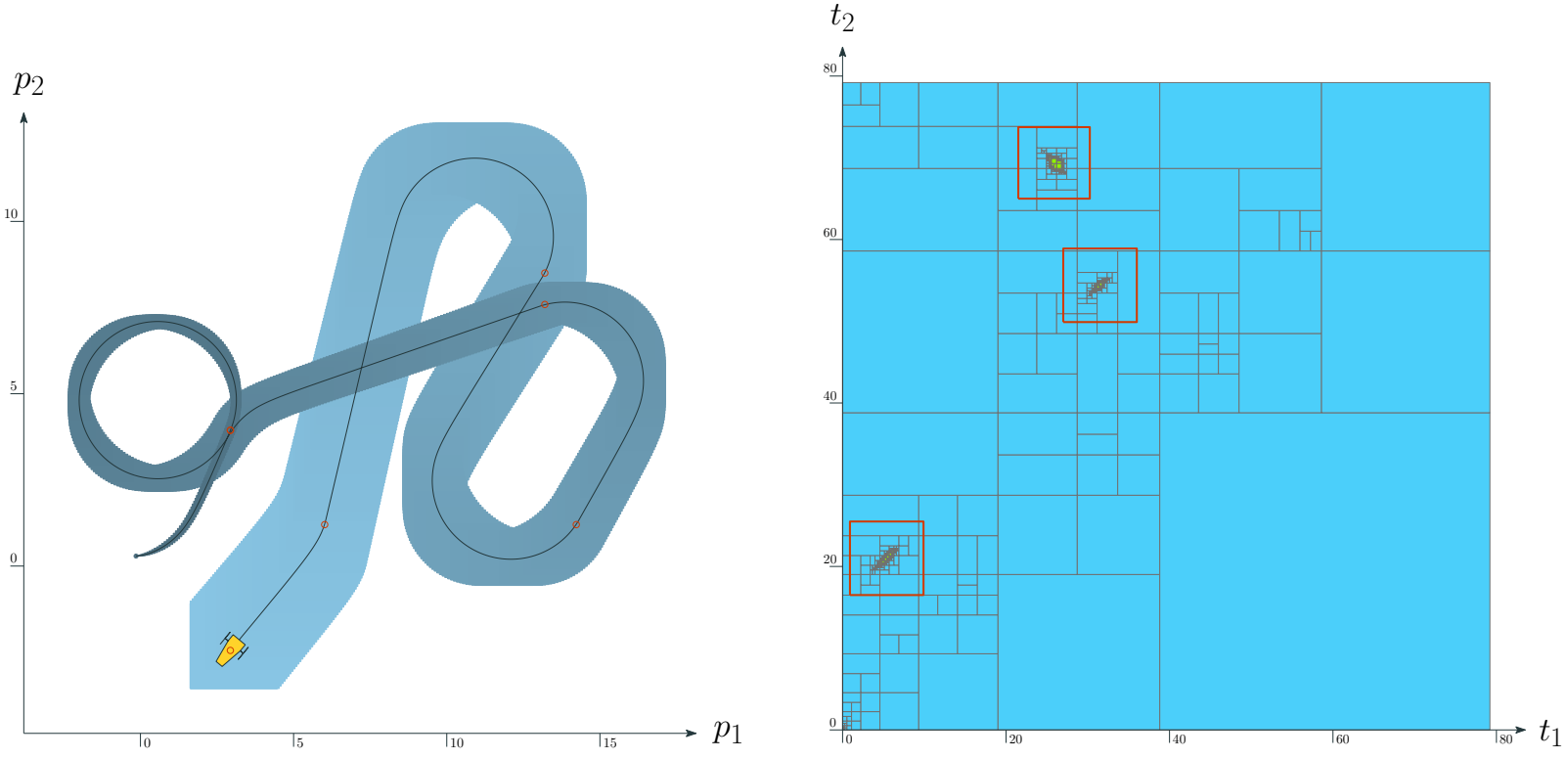
www.codac.io



Example of computed t-plane.

Example of application

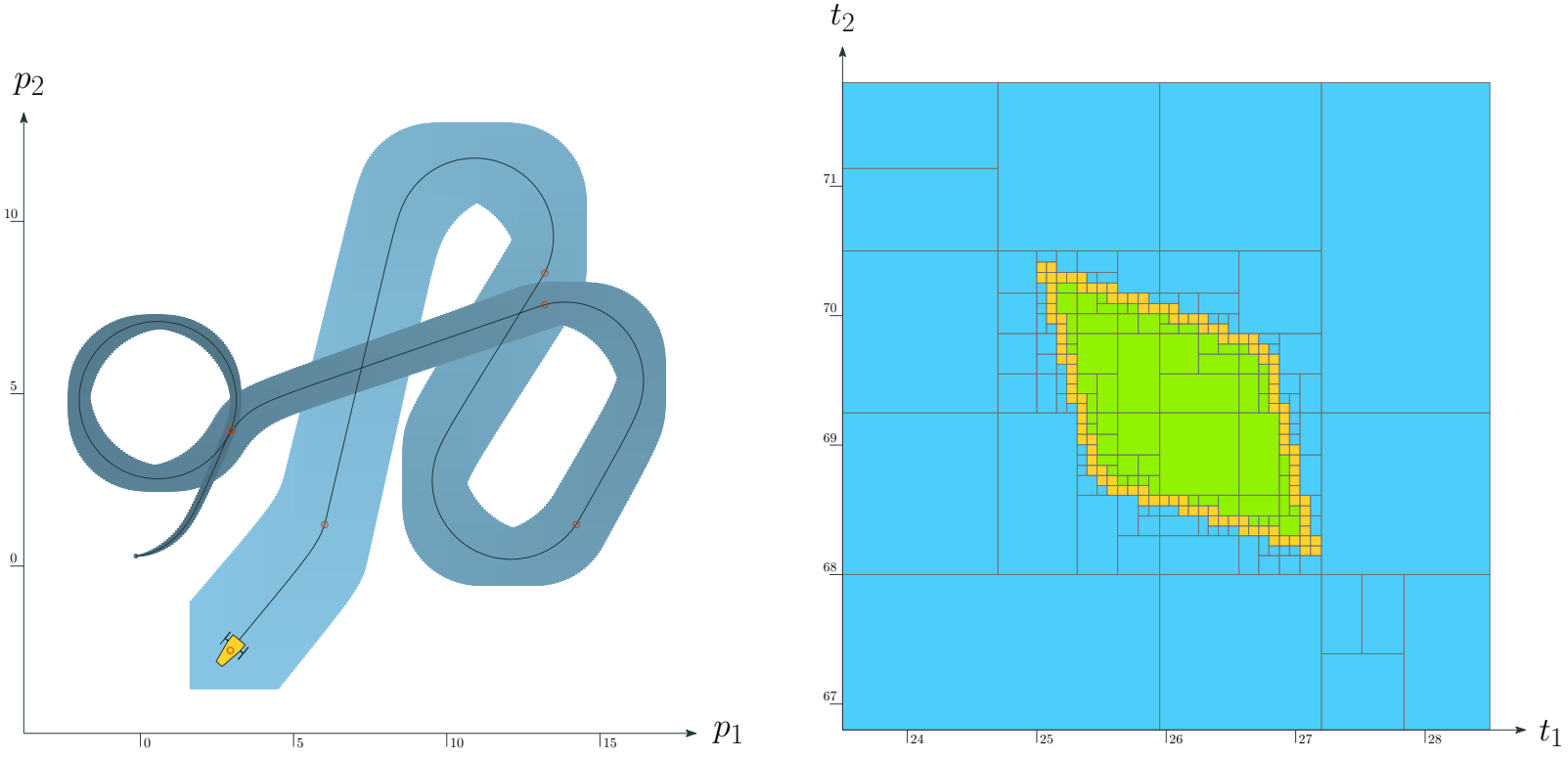
www.codac.io



Example of computed t-plane.

Example of application

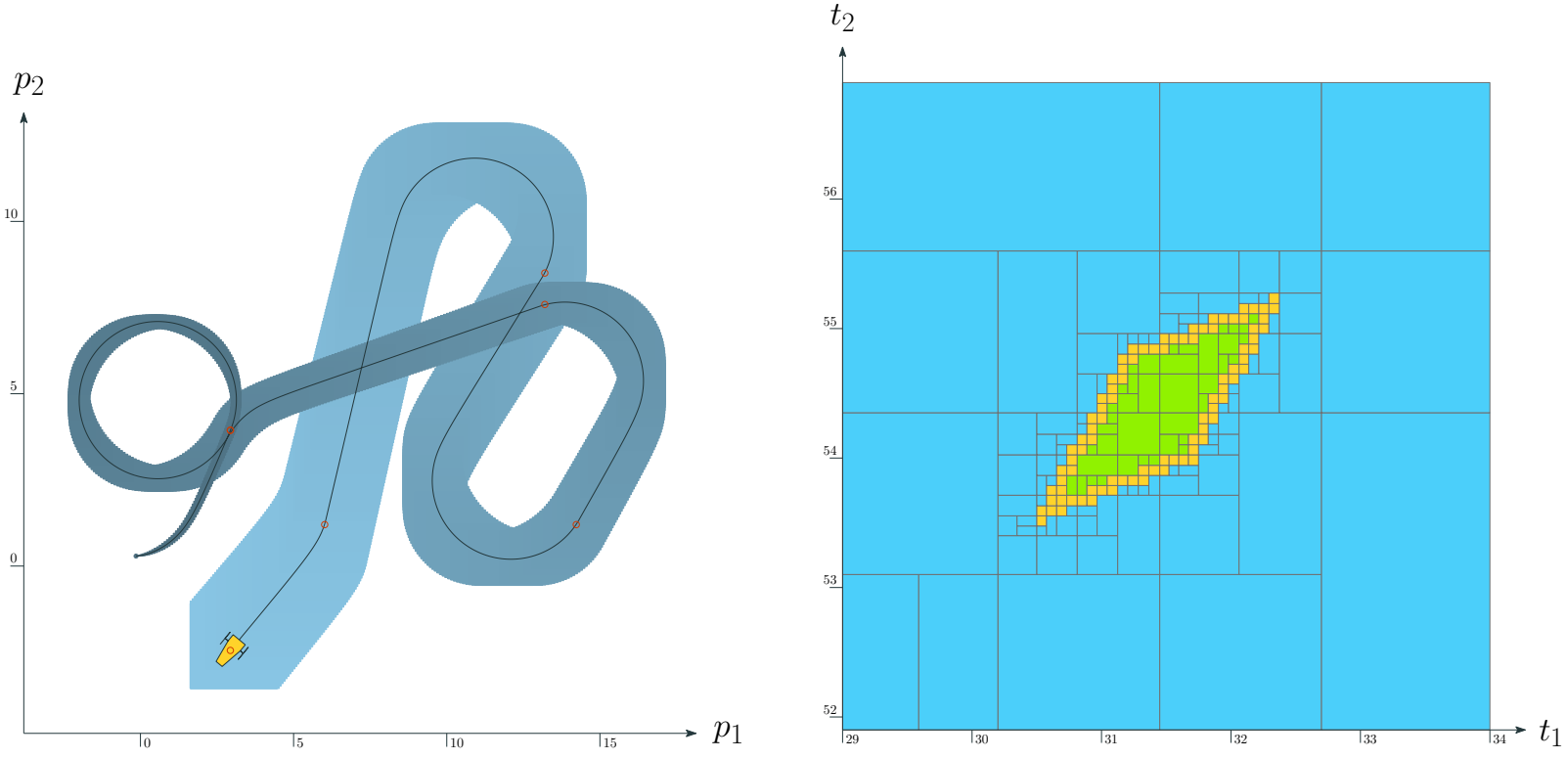
www.codac.io



Example of computed t-plane.

Example of application

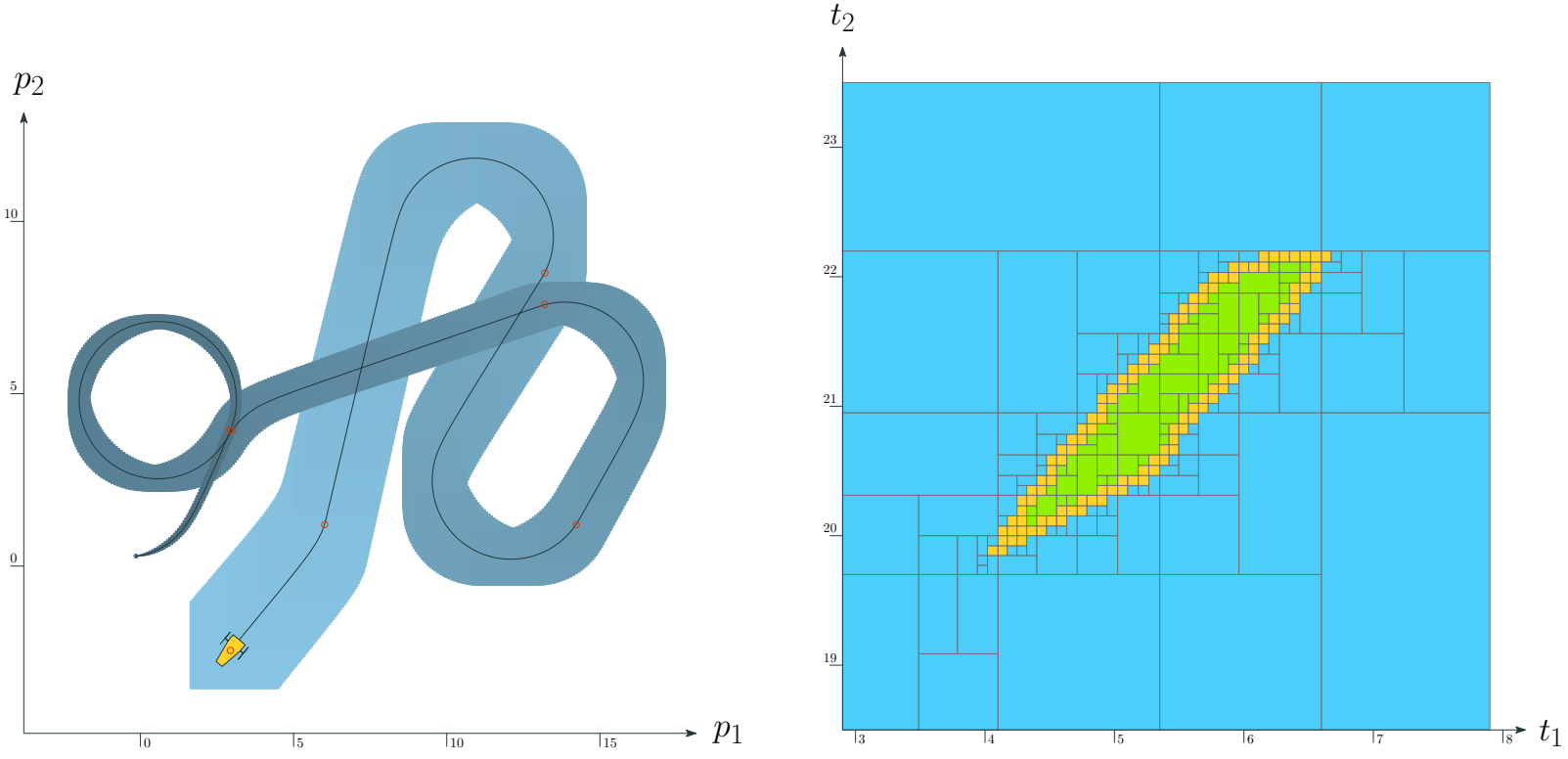
www.codac.io



Example of computed t-plane.

Example of application

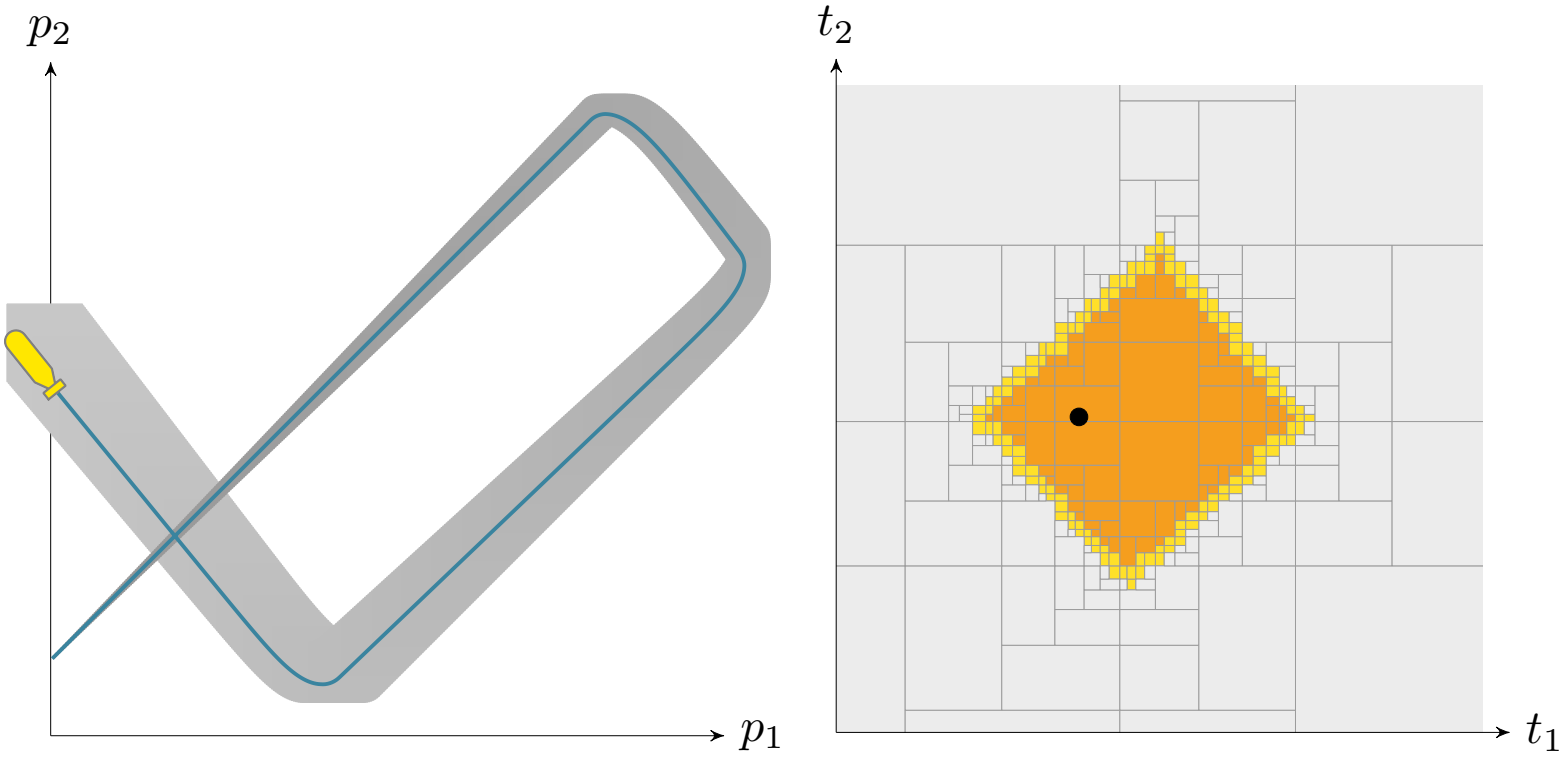
www.codac.io



Example of computed t-plane.

Reliable approximation of a loop set

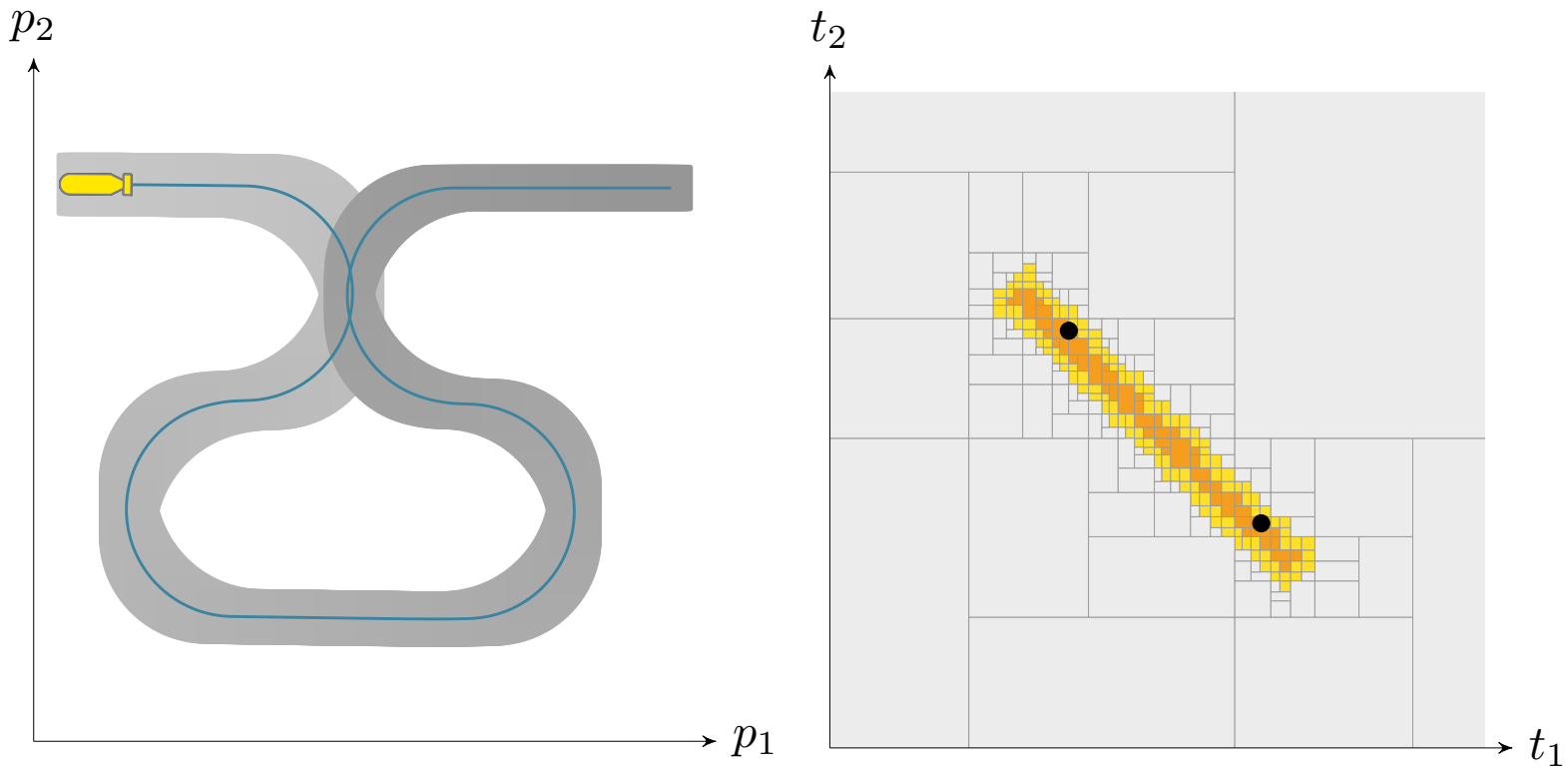
Using a **SIVIA**-like algorithm:



Undeniable looped trajectory

Reliable approximation of a loop set

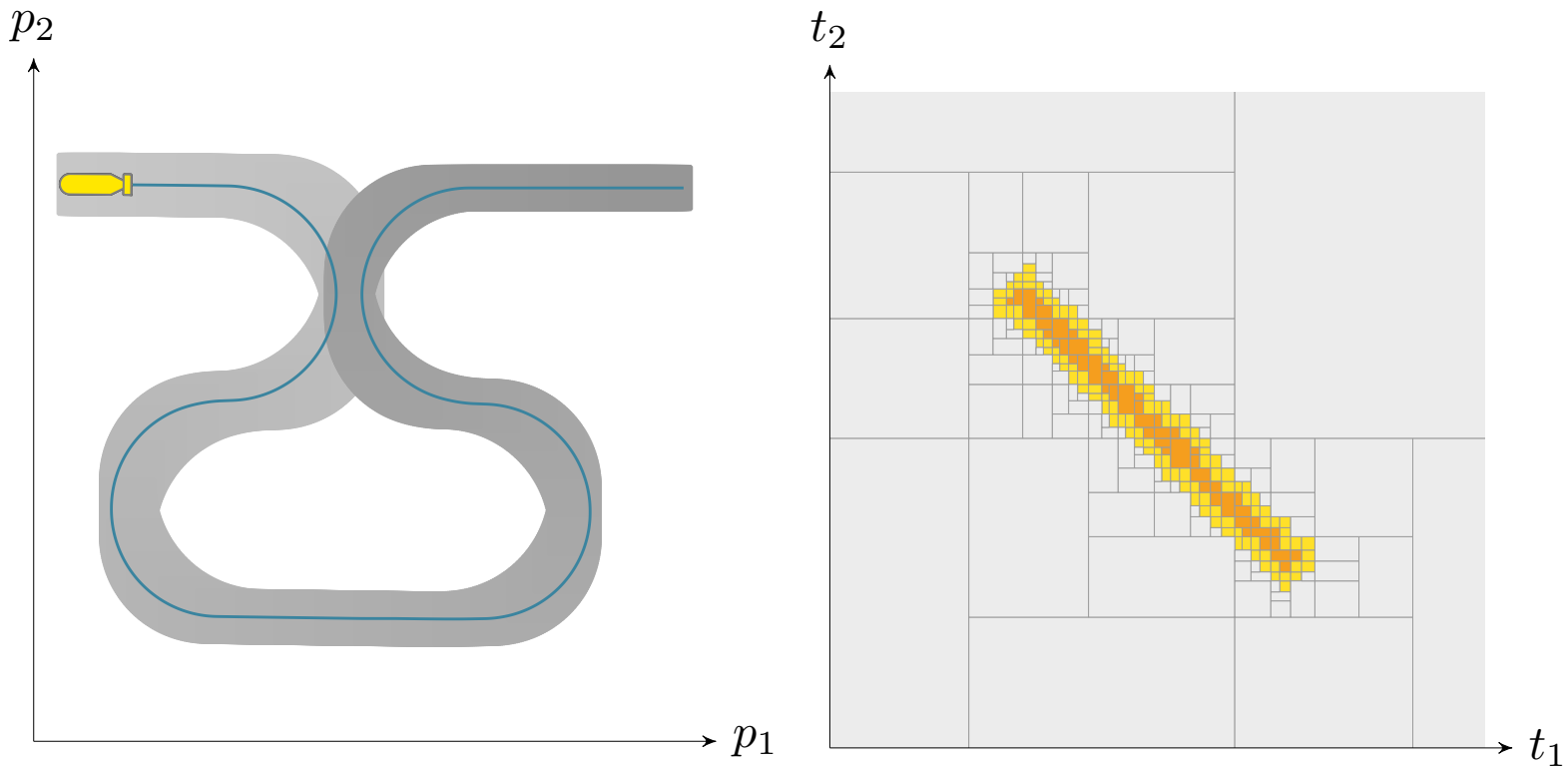
Using a **SIVIA**-like algorithm:



Doubtful looped trajectory

Reliable approximation of a loop set

Using a **SIVIA**-like algorithm:



Doubtful looped trajectory

Verifying the existence of loops

Inclusion function

Simplification:

defining the actual but unknown function $\mathbf{f}^* : \mathbb{R}^2 \to \mathbb{R}^2$

$$\mathbf{f}^*(t_1, t_2) = \int_{t_1}^{t_2} \mathbf{v}^*(\tau) d\tau \tag{7}$$

Inclusion function

Simplification:

defining the actual but unknown function $\mathbf{f}^* : \mathbb{R}^2 \to \mathbb{R}^2$

$$\mathbf{f}^*(t_1, t_2) = \int_{t_1}^{t_2} \mathbf{v}^*(\tau) d\tau \tag{7}$$

Assessed knowledge:

$[\mathbf{f}] : \mathbb{R}^2 \to \mathbb{IR}^2$ is an *interval function* of $\mathbf{f}^*$:

$$\mathbf{f}^*(t_1, t_2) \in [\mathbf{f}](t_1, t_2) = \int_{t_1}^{t_2} [\mathbf{v}](\tau) d\tau \tag{8}$$

Inclusion function

Simplification:

defining the actual but unknown function $\mathbf{f}^* : \mathbb{R}^2 \to \mathbb{R}^2$

$$\mathbf{f}^*(t_1, t_2) = \int_{t_1}^{t_2} \mathbf{v}^*(\tau) d\tau \tag{7}$$

Assessed knowledge:

$[\mathbf{f}] : \mathbb{R}^2 \to \mathbb{IR}^2$ is an *interval function* of $\mathbf{f}^*$:

$$\mathbf{f}^*(t_1, t_2) \in [\mathbf{f}](t_1, t_2) = \int_{t_1}^{t_2} [\mathbf{v}](\tau) d\tau \tag{8}$$

Verification of a loop:

$$\forall \mathbf{f} \in [\mathbf{f}], \exists \mathbf{t} \in \mathbb{T}_i \mid \mathbf{f}(\mathbf{t}) = \mathbf{0} \implies \underbrace{\exists \mathbf{t} \in \mathbb{T}_i \mid \mathbf{f}^*(\mathbf{t}) = \mathbf{0}}_{\text{loop existence proof}} \tag{9}$$

Zero verification: problem statement

Usually we cannot find exactly a zero of an uncertain function, but we can prove it exists within some domain Ω.

Zero verification: problem statement

Usually we cannot find exactly a zero of an uncertain function, but we can prove it exists within some domain Ω.

Assumptions:

- known inclusion function $[\mathbf{f}] : \mathbb{IR}^n \to \mathbb{IR}^m$ of the unknown function $\mathbf{f}^* : \mathbb{R}^n \to \mathbb{R}^m$
- possibly in the form of an algorithm for computing $[\mathbf{f}]([\mathbf{t}])$
- $n = m = 2$
- Ω possibly made of a union of finitely many boxes in $\mathbb{IR}^2$

$\longrightarrow$ need to isolate and verify zeros of $\mathbf{f}^*$

Powerful topological degree

Topological degree $\deg(\mathbf{f}^*, \Omega)$:

- unique integer assigned to $\mathbf{f}^*$ and a compact set $\Omega \subset \mathbb{R}^n$ such that $\mathbf{f}^*(\mathbf{t}) \neq \mathbf{0}$ for all $\mathbf{t} \in \partial\Omega$

Powerful topological degree

Topological degree $\deg(\mathbf{f}^*, \Omega)$:

- unique integer assigned to $\mathbf{f}^*$ and a compact set $\Omega \subset \mathbb{R}^n$ such that $\mathbf{f}^*(\mathbf{t}) \neq \mathbf{0}$ for all $\mathbf{t} \in \partial\Omega$

Most important property of it:

$$\deg(\mathbf{f}^*, \Omega) \neq 0 \implies \exists \mathbf{t} \in \Omega \mid \mathbf{f}^*(\mathbf{t}) = \mathbf{0} \tag{10}$$

- Topological degree theory and applications
  Y. J. Cho, Y. Q. Chen *Mathematical Analysis and Applications*, 2006
- Degree theory in analysis and applications
  I. Fonseca, W. Gangbo *Oxford lecture series*, 1995
- A set of axioms for the degree of a tangent vector field on differentiable manifolds
  M. Furi, M. P. Pera, M. Spadini *Fixed Point Theory and Applications*, 2010

Powerful topological degree

Assets of topological degree:

- can be computed in case where only an inclusion function $[\mathbf{f}]$ of $\mathbf{f}^*$ is given.

  - Effective topological degree computation based on interval arithmetic
    P. Franek, S. Ratschan *CoRR*, 2012

Powerful topological degree

Assets of topological degree:

- can be computed in case where only an inclusion function $[\mathbf{f}]$ of $\mathbf{f}^*$ is given.
  - Effective topological degree computation based on interval arithmetic
    P. Franek, S. Ratschan *CoRR*, 2012
- is in many cases more powerful than more classical verification tools including interval Newton, Miranda's or Borsuk's tests.
  - Quasi-decidability of a fragment of the first-order theory of real numbers
    P. Franek, S. Ratschan, P. Zgliczynski *Journal of Automated Reasoning*, 2015

Powerful topological degree

Assets of topological degree:

- can be computed in case where only an inclusion function $[\mathbf{f}]$ of $\mathbf{f}^*$ is given.
  - Effective topological degree computation based on interval arithmetic
    P. Franek, S. Ratschan *CoRR*, 2012
- is in many cases more powerful than more classical verification tools including interval Newton, Miranda's or Borsuk's tests.
  - Quasi-decidability of a fragment of the first-order theory of real numbers
    P. Franek, S. Ratschan, P. Zgliczynski *Journal of Automated Reasoning*, 2015
- useful to count the number of $\mathbf{0}$.

Powerful topological degree

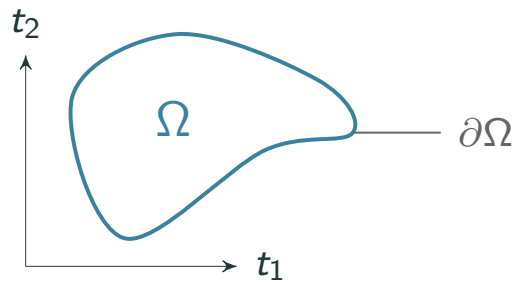
Our application for loop detection:

- deals with the relatively easy case $n = 2$ (t_1, t_2)
- nice geometric interpretation

Powerful topological degree

Our application for loop detection:

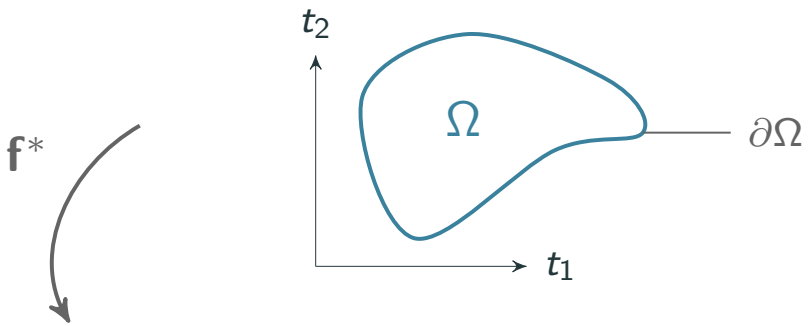
- deals with the relatively easy case $n = 2$ $\quad (t_1, t_2)$
- nice geometric interpretation:
  - *winding number* of the curve $\partial\Omega \xrightarrow{\mathbf{f}^*} \mathbb{R}^2 \setminus \{\mathbf{0}\}$ around $\mathbf{0}$

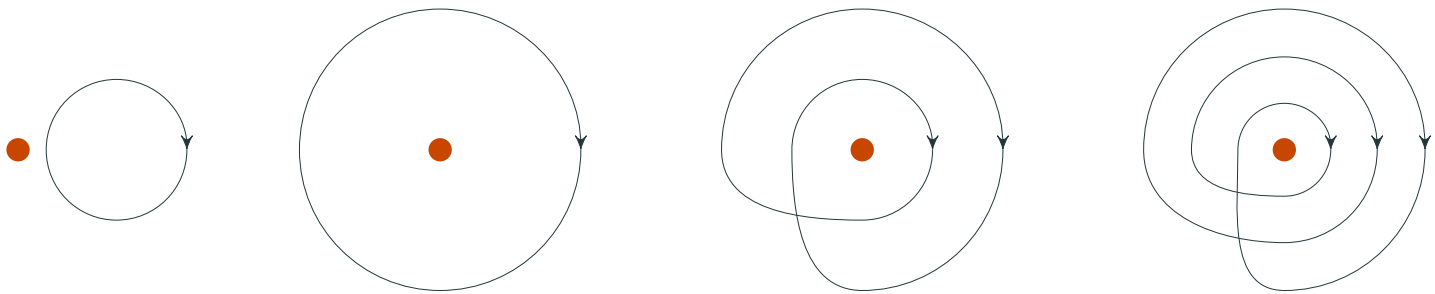


Powerful topological degree

Our application for loop detection:

- deals with the relatively easy case $n = 2$ $\qquad (t_1, t_2)$
- nice geometric interpretation:
  - *winding number* of the curve $\partial\Omega \xrightarrow{\mathbf{f}^*} \mathbb{R}^2 \setminus \{\mathbf{0}\}$ around $\mathbf{0}$

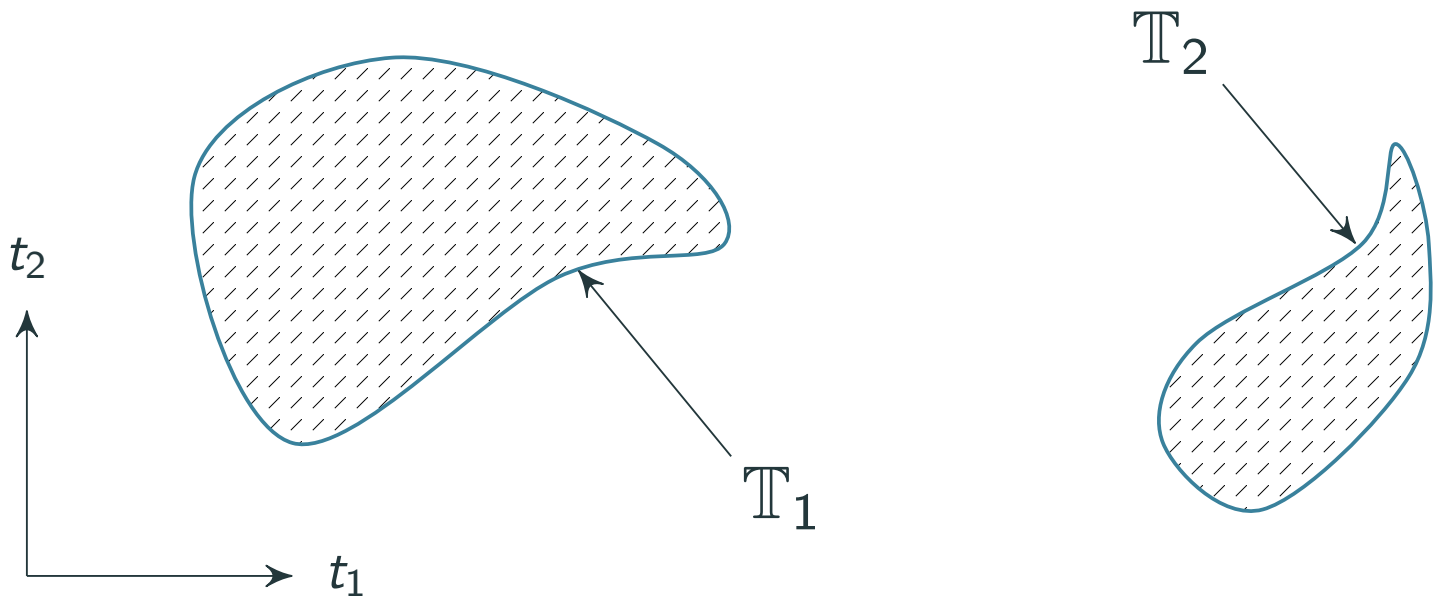




$\deg = 0$ $\qquad$ $\deg = 1$ $\qquad$ $\deg = 2$ $\qquad$ $\deg = 3$

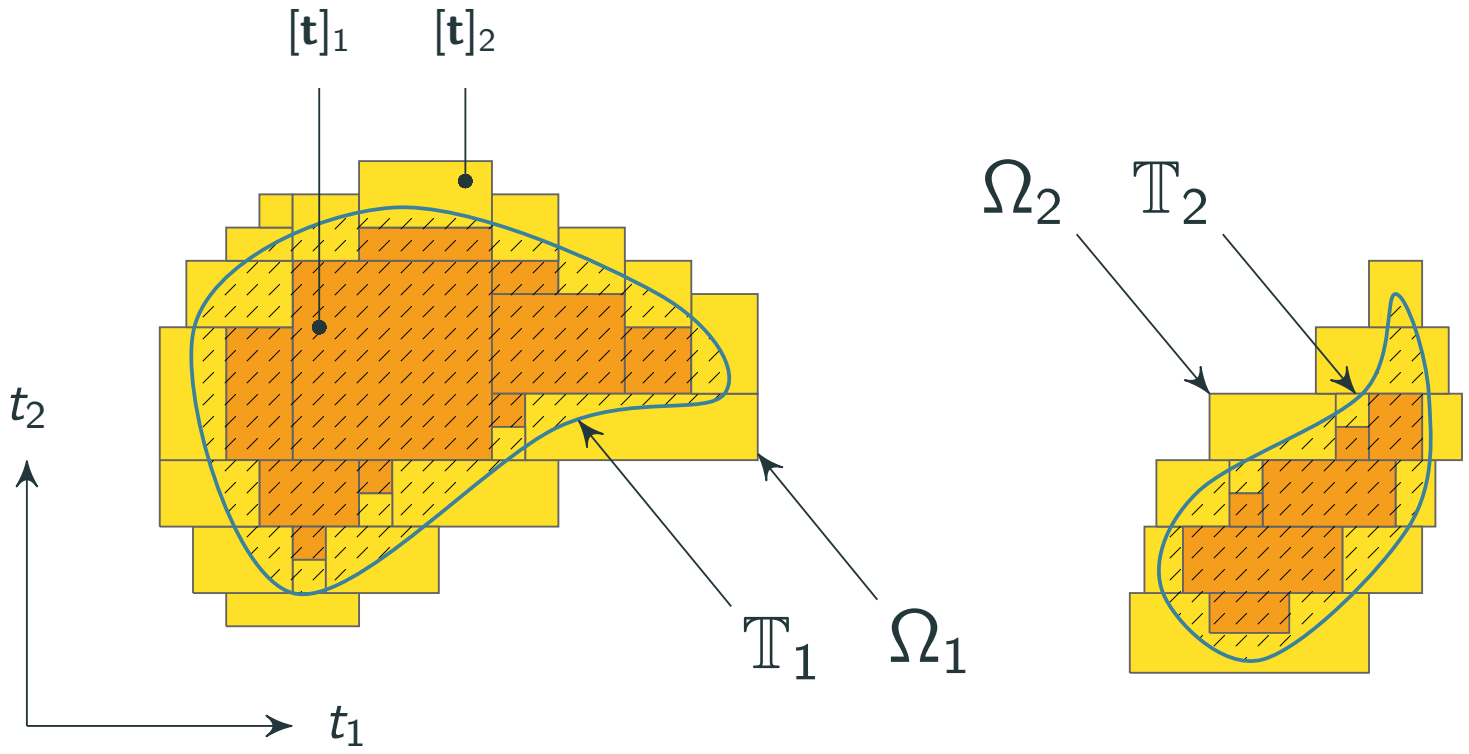
Outer approximation of a set $\mathbb{T}$ with SIVIA

Consider $\mathbb{T} \subset \mathbb{R}^2$ in which we want to find zeros of $\mathbf{f}^*$.



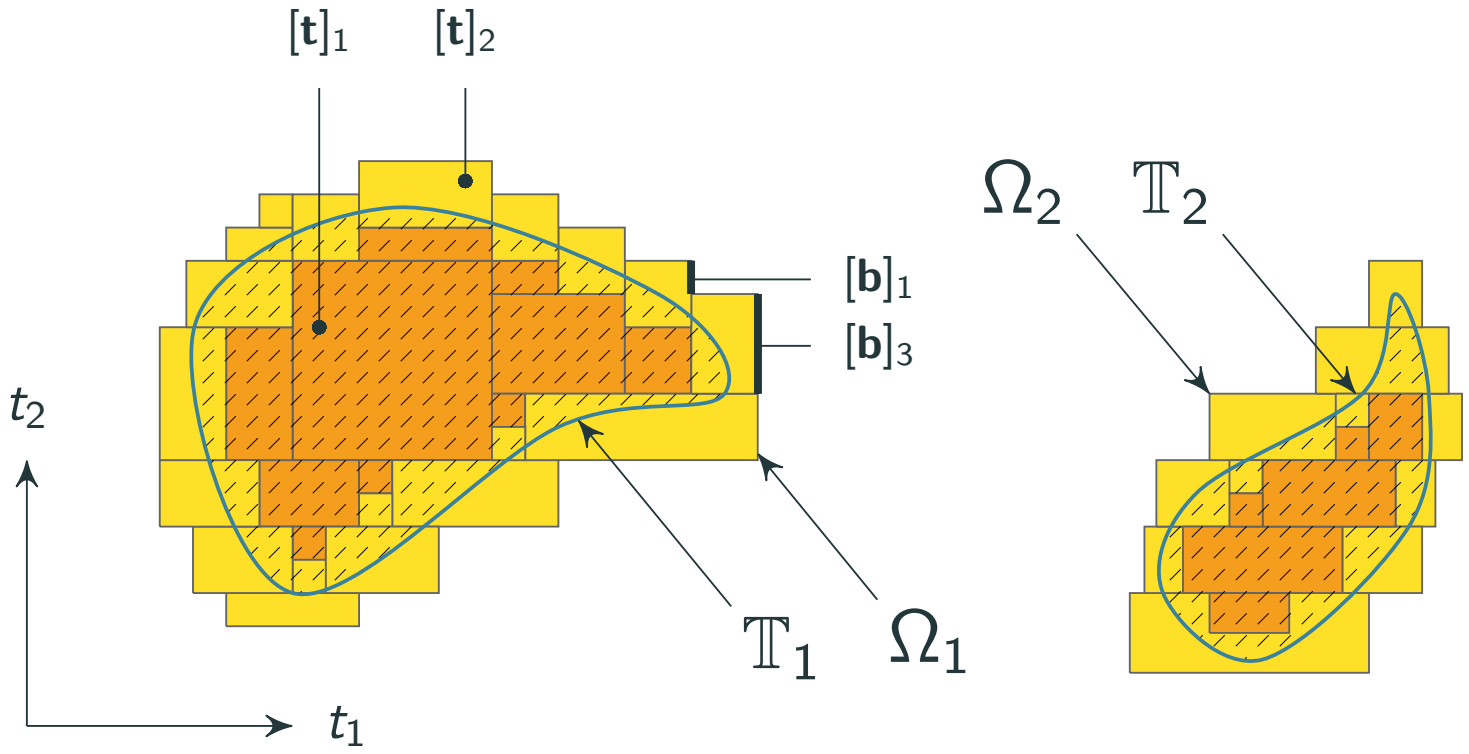
Outer approximation of a set $\mathbb{T}$ with SIVIA

Consider $\mathbb{T} \subset \mathbb{R}^2$ in which we want to find zeros of $\mathbf{f}^*$.



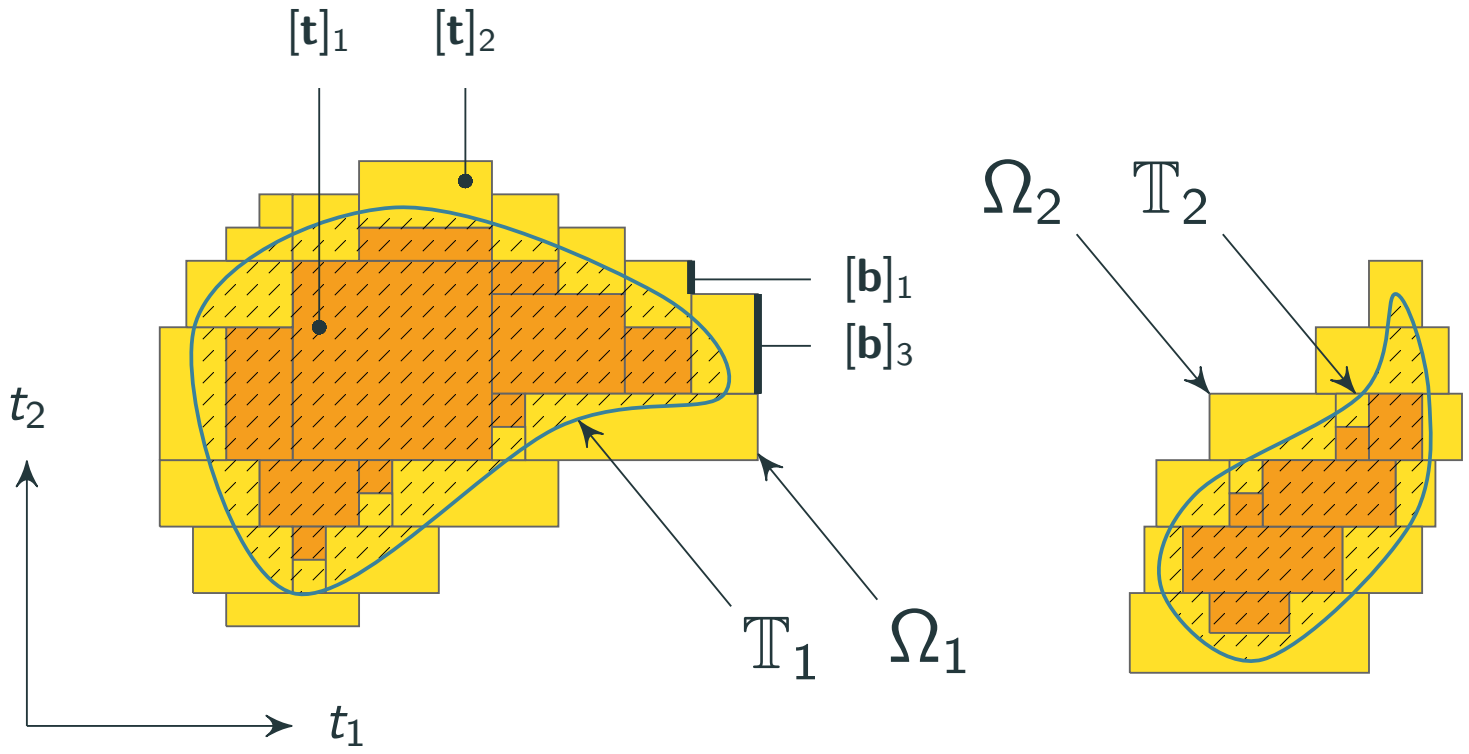
Outer approximation of a set $\mathbb{T}$ with SIVIA

Consider $\mathbb{T} \subset \mathbb{R}^2$ in which we want to find zeros of $\mathbf{f}^*$.



Outer approximation of a set $\mathbb{T}$ with SIVIA

Consider $\mathbb{T} \subset \mathbb{R}^2$ in which we want to find zeros of $\mathbf{f}^*$.

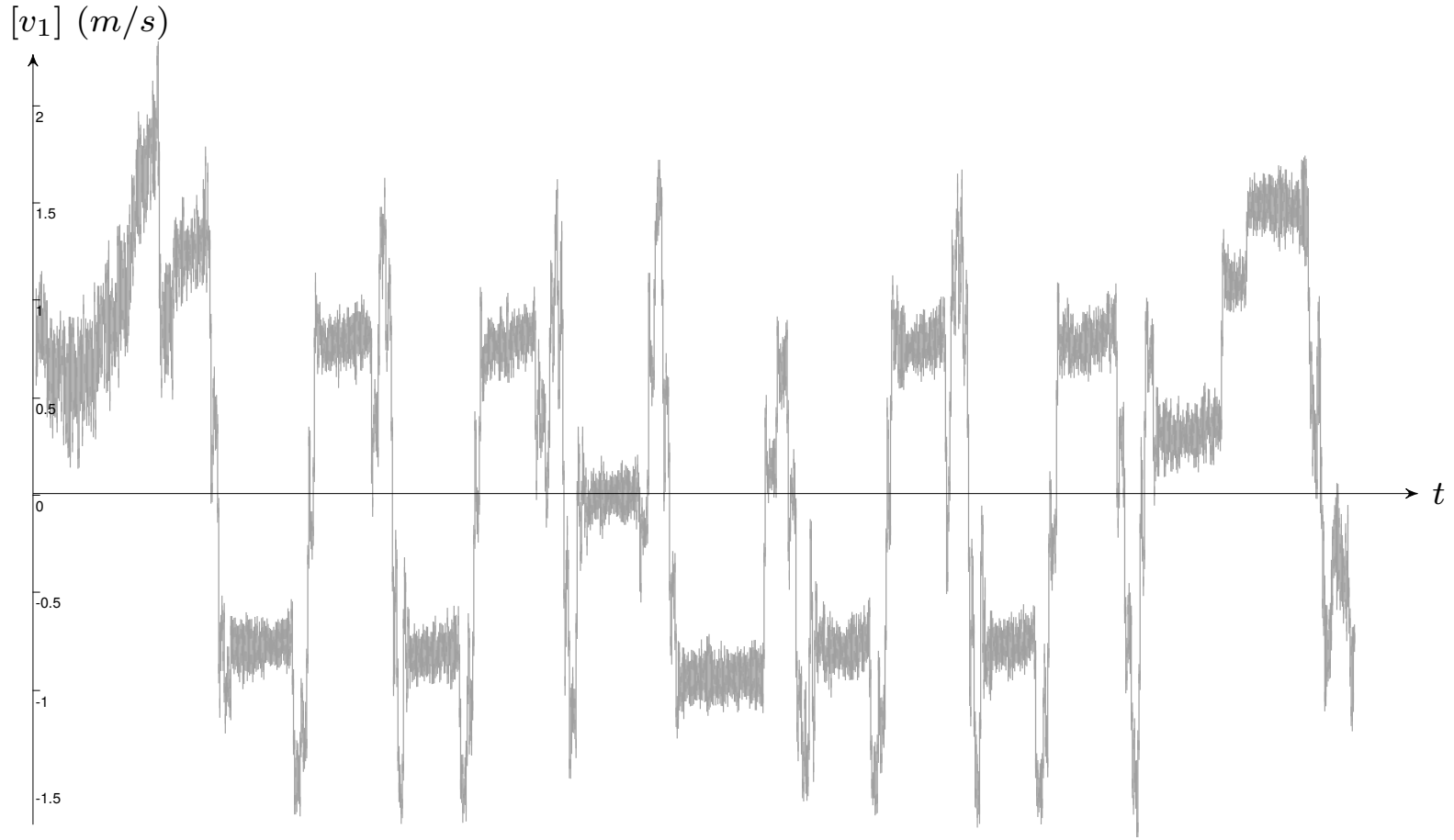


Outer set has the properties required for Ω: $\mathbf{f}^*(\mathbf{t}) \neq \mathbf{0},\ \forall \mathbf{t} \in \partial\Omega$

Verifying loops on actual datasets

Enclosed velocities

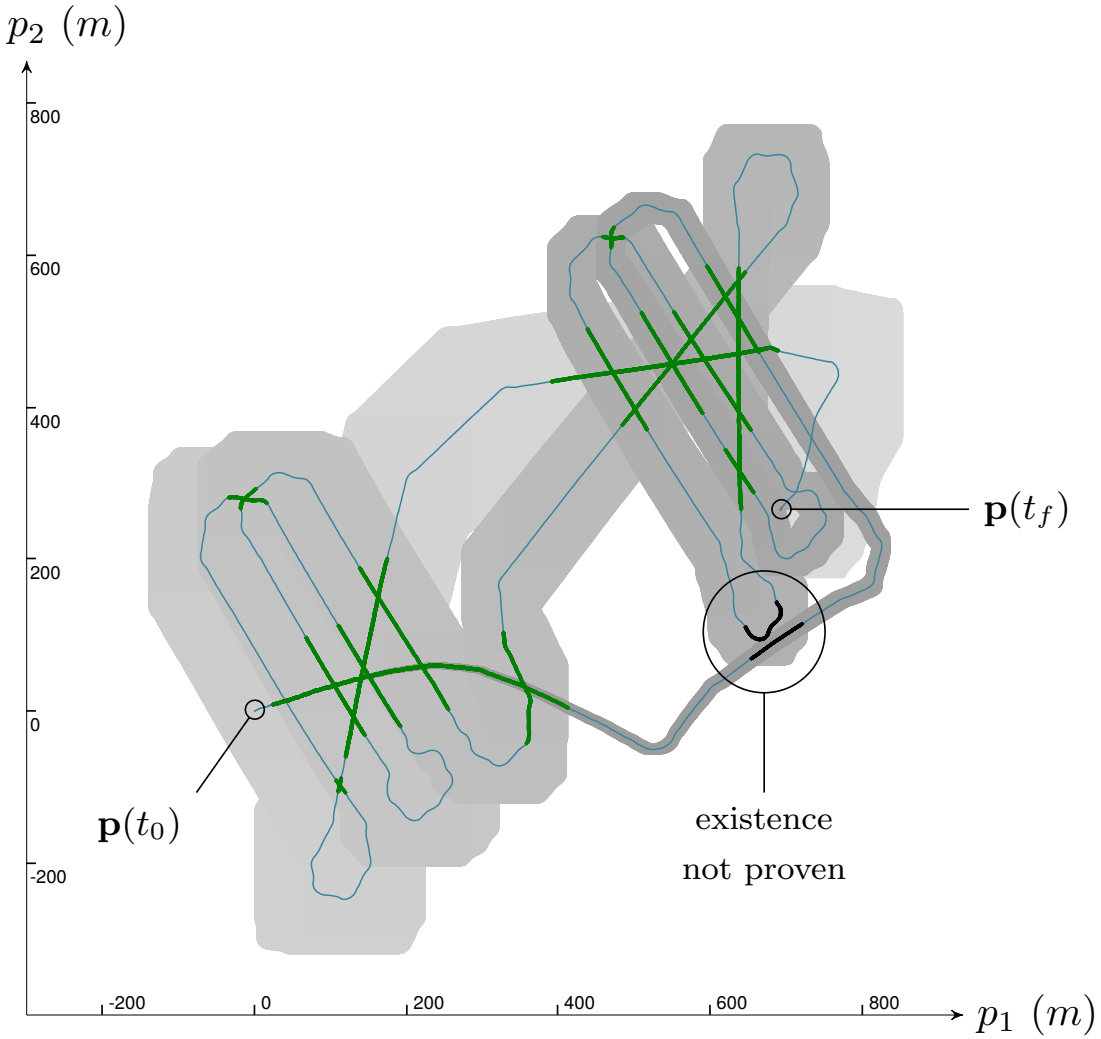
Tube $[\mathbf{v}](\cdot)$ built from sensor data:
(DVL for velocity + AHRS for attitude)



East speed velocity tube $[v_1](\cdot)$.

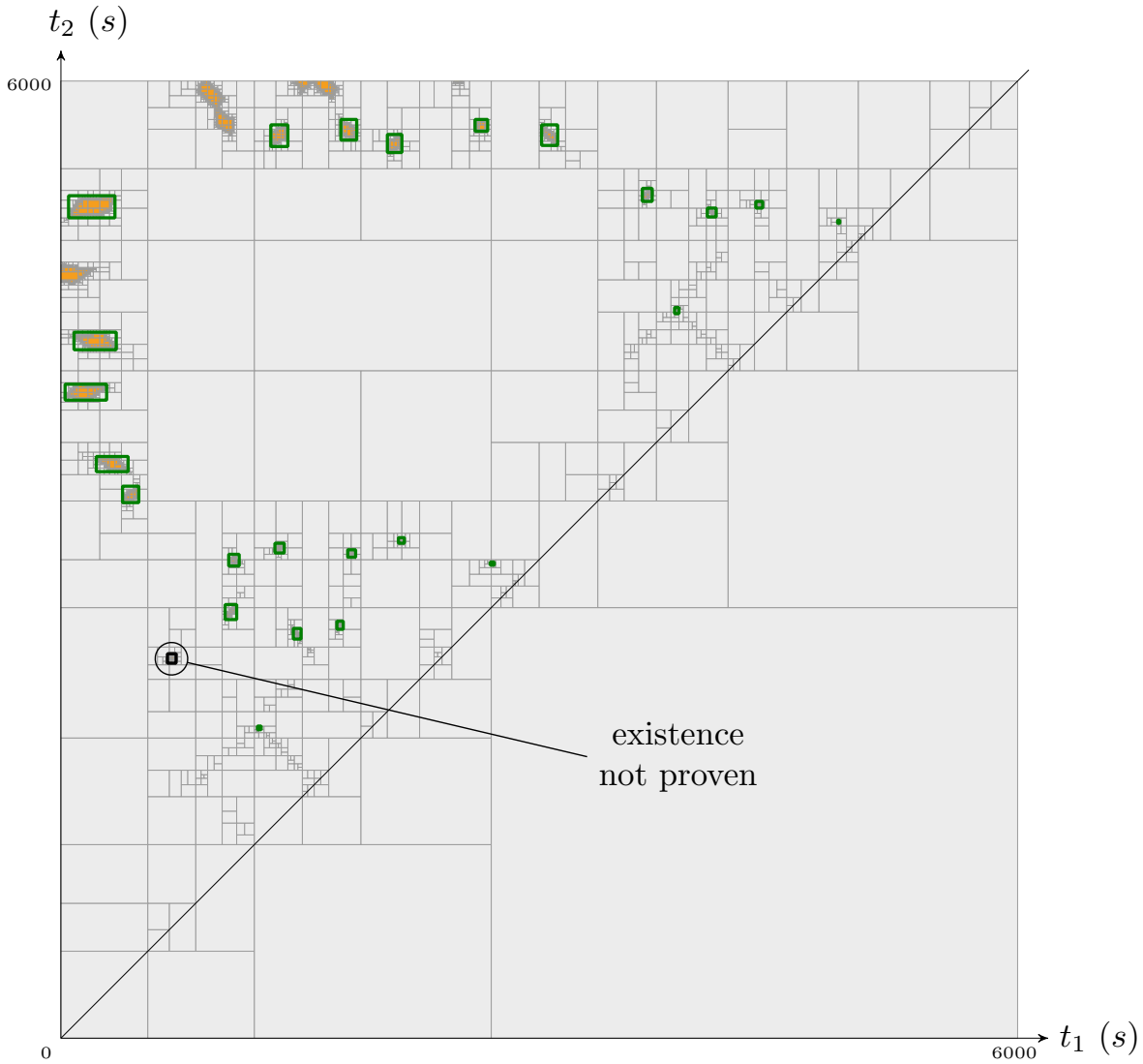
Redermor's mission

Guaranteed enclosure of the trajectory:



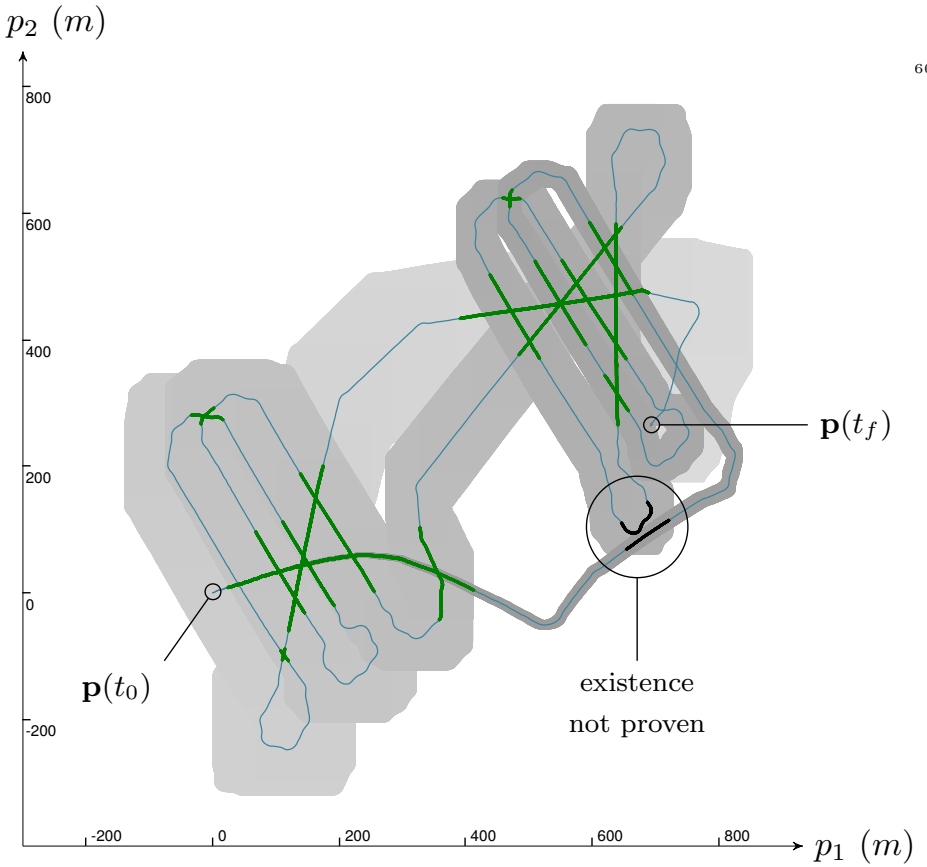
Tube $[\mathbf{p}](\cdot)$ *enclosing the feasible trajectories of Redermor AUV.*

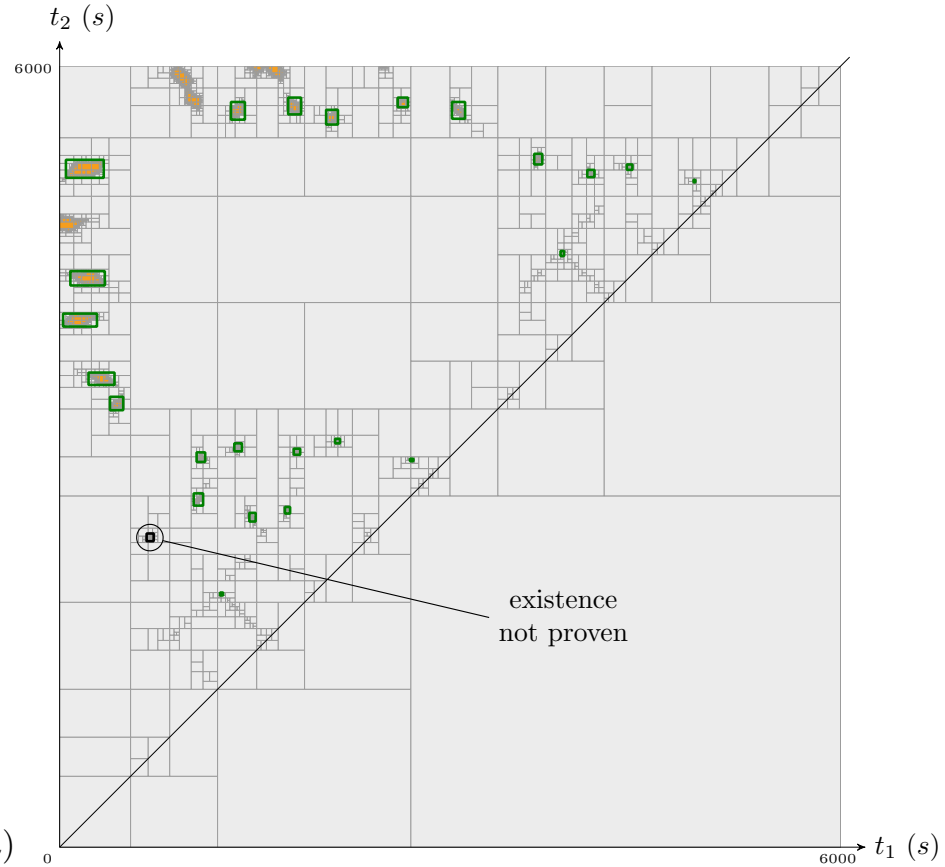
t-plane of the mission: $\mathbb{T} = \{(t_1, t_2) \mid \mathbf{0} \in [\mathbf{f}](t_1, t_2), t_1 < t_2\}$



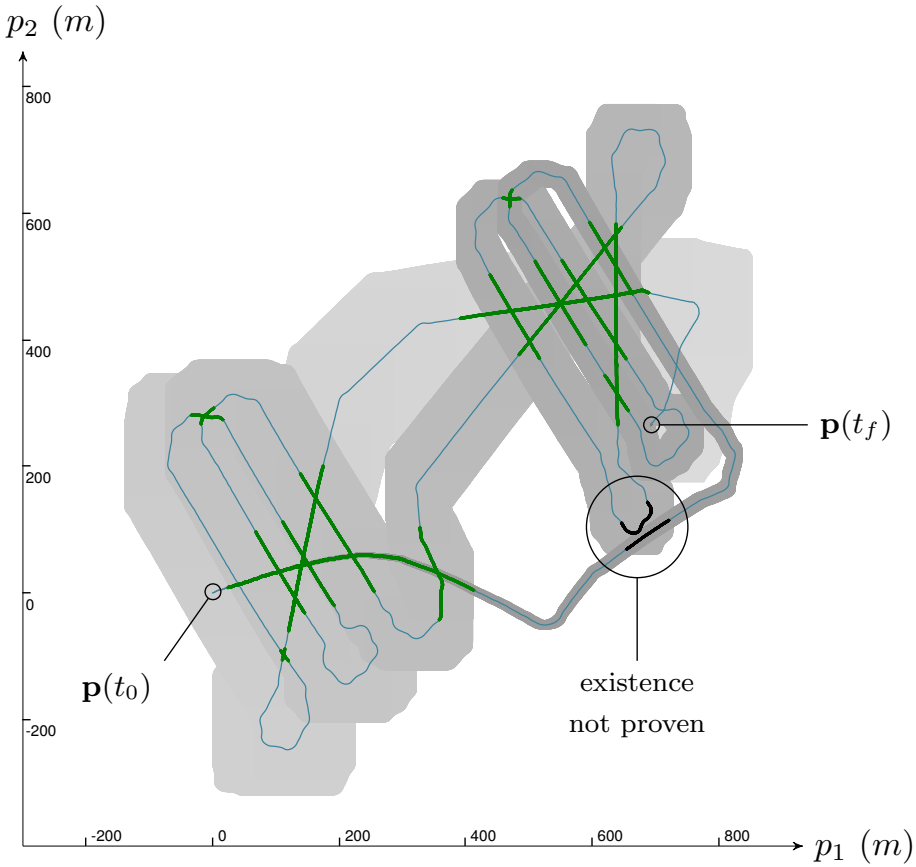
t-plane corresponding to Redermor's mission.

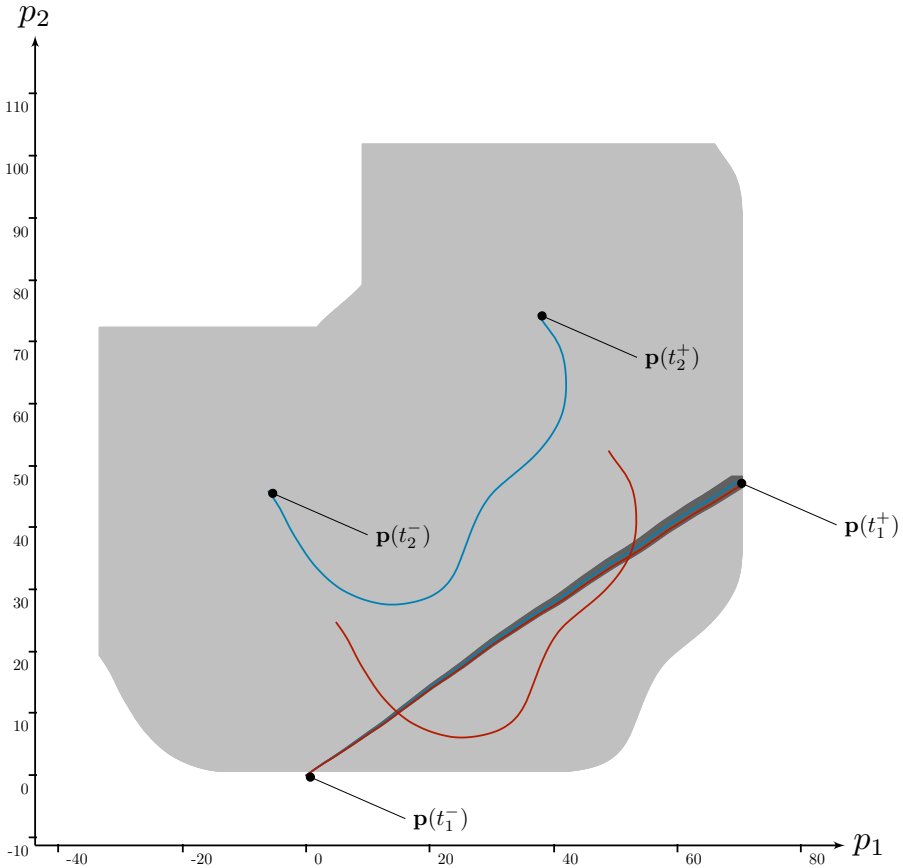
Redermor's mission: overview and results



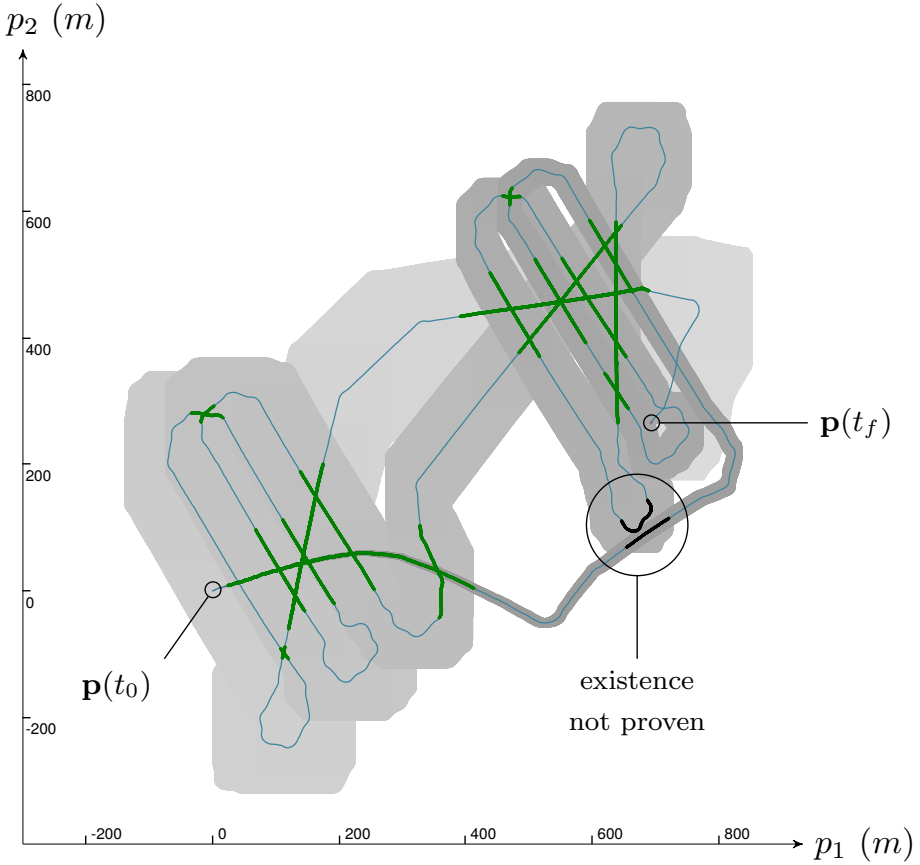


Redermor's mission: overview and results





Redermor's mission: overview and results

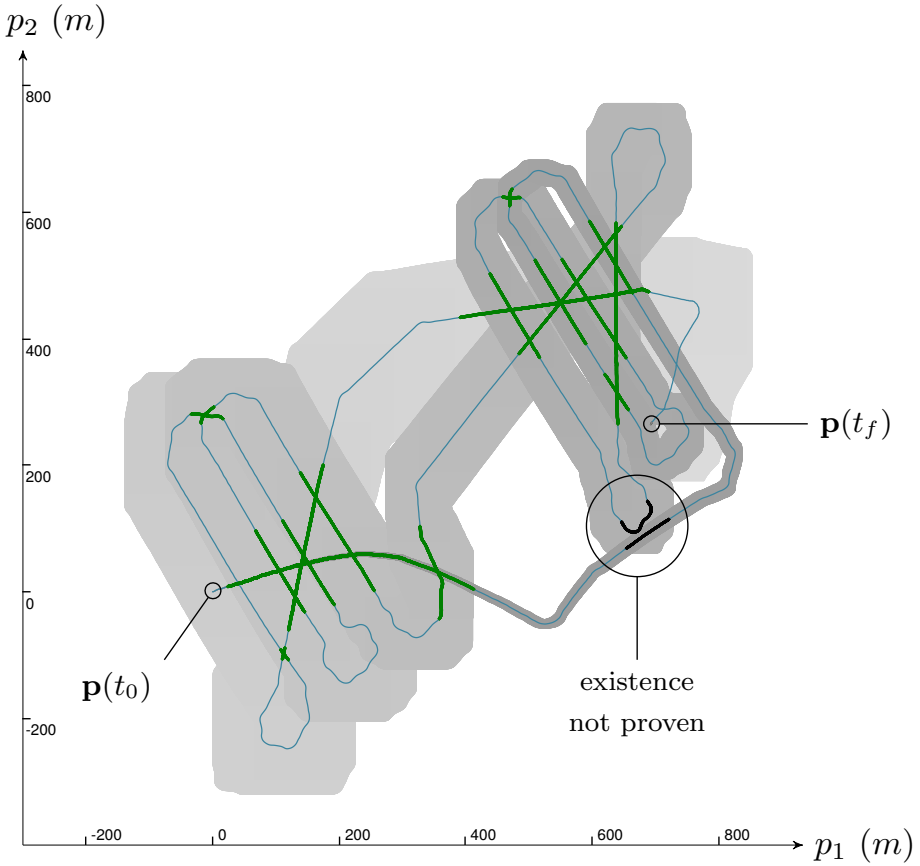


Loop proof number

Without uncertainties:

$$\lambda^* = \#\{\mathbf{t} \mid \mathbf{f}^*(\mathbf{t}) = \mathbf{0}, t_1 < t_2\}$$

Redermor's mission: overview and results



Loop proof number

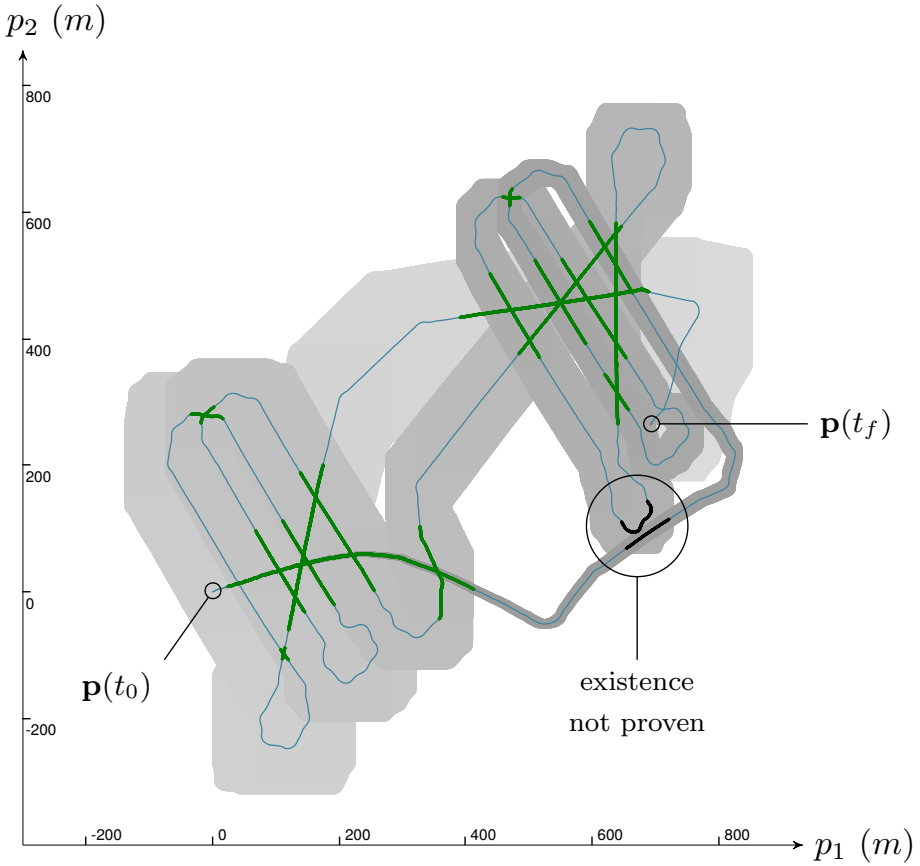
Without uncertainties:

$$\lambda^* = \#\{\mathbf{t} \mid \mathbf{f}^*(\mathbf{t}) = \mathbf{0}, t_1 < t_2\}$$

Results:

Newton operator test: $\lambda_{\mathcal{N}} = 14$

Redermor's mission: overview and results



Loop proof number

Without uncertainties:

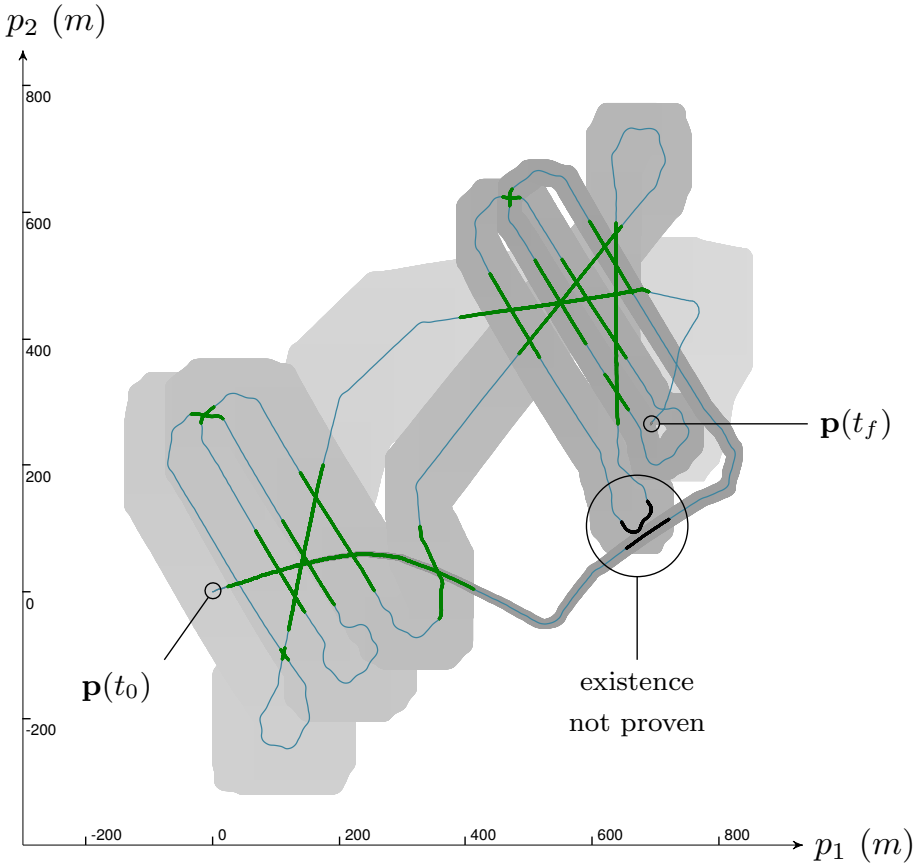
$$\lambda^* = \#\{\mathbf{t} \mid \mathbf{f}^*(\mathbf{t}) = \mathbf{0}, t_1 < t_2\}$$

Results:

Newton operator test: $\lambda_{\mathcal{N}} = 14$

Topological degree test: $\lambda_{\mathcal{T}} = 24$

Redermor's mission: overview and results



Loop proof number

Without uncertainties:

$$\lambda^* = \#\{\mathbf{t} \mid \mathbf{f}^*(\mathbf{t}) = \mathbf{0}, t_1 < t_2\}$$

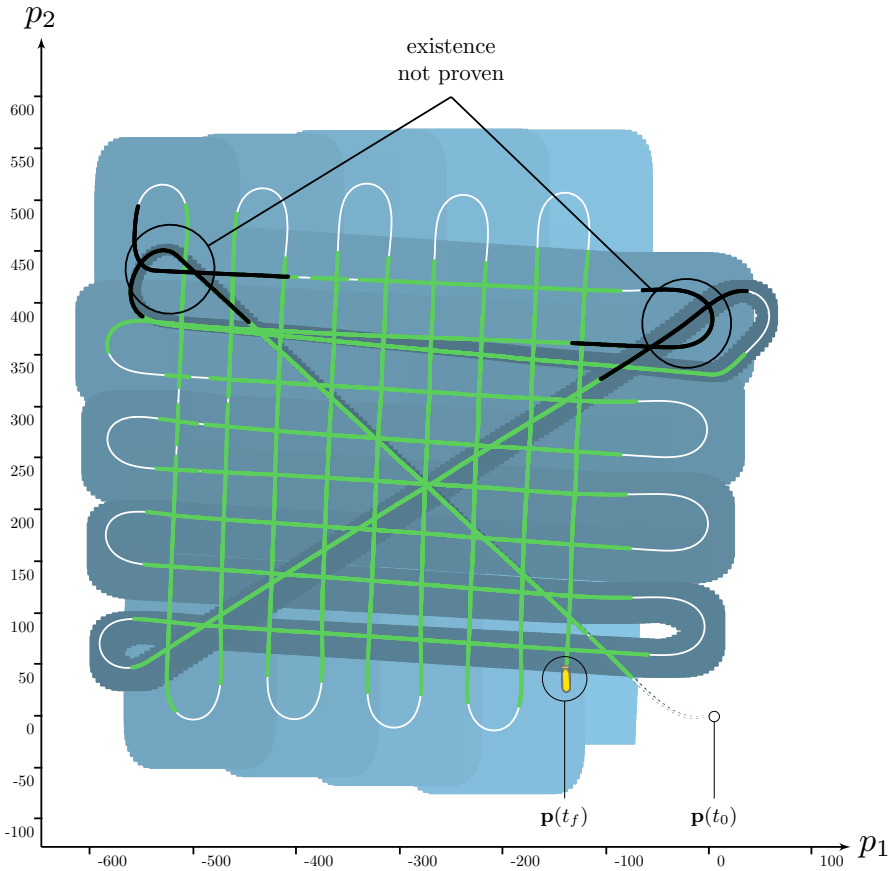
Results:

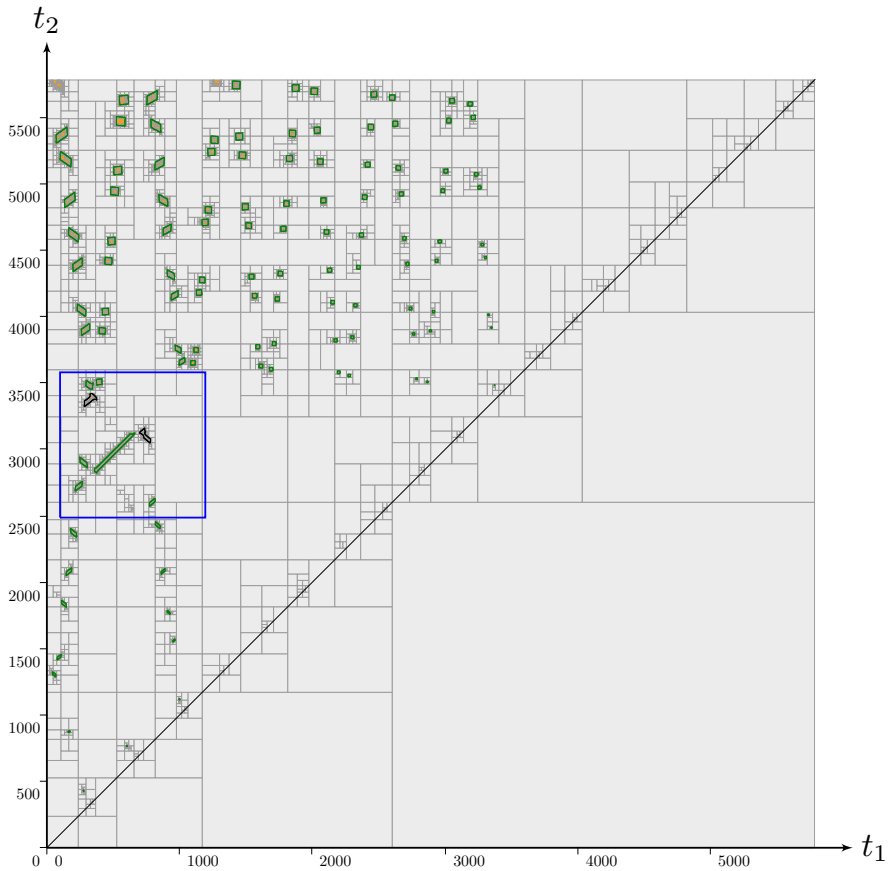
Newton operator test: $\lambda_{\mathcal{N}} = 14$

Topological degree test: $\lambda_{\mathcal{T}} = 24$

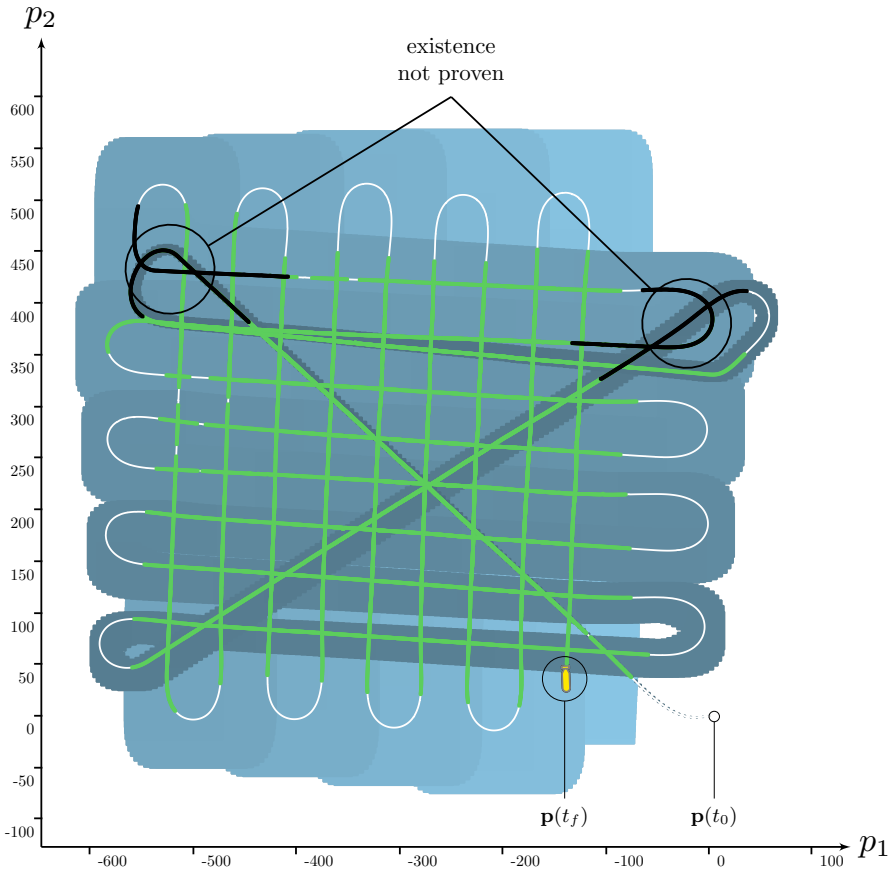
Truth: $\lambda^* = 24$

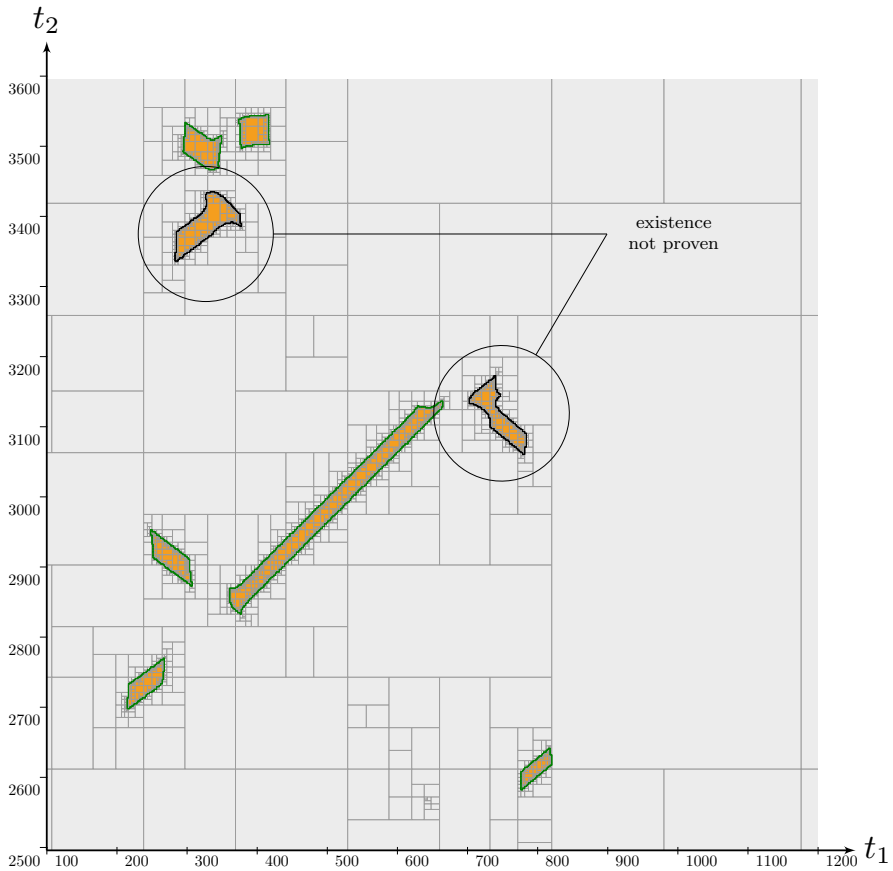
Another experiment: *Daurade*'s mission



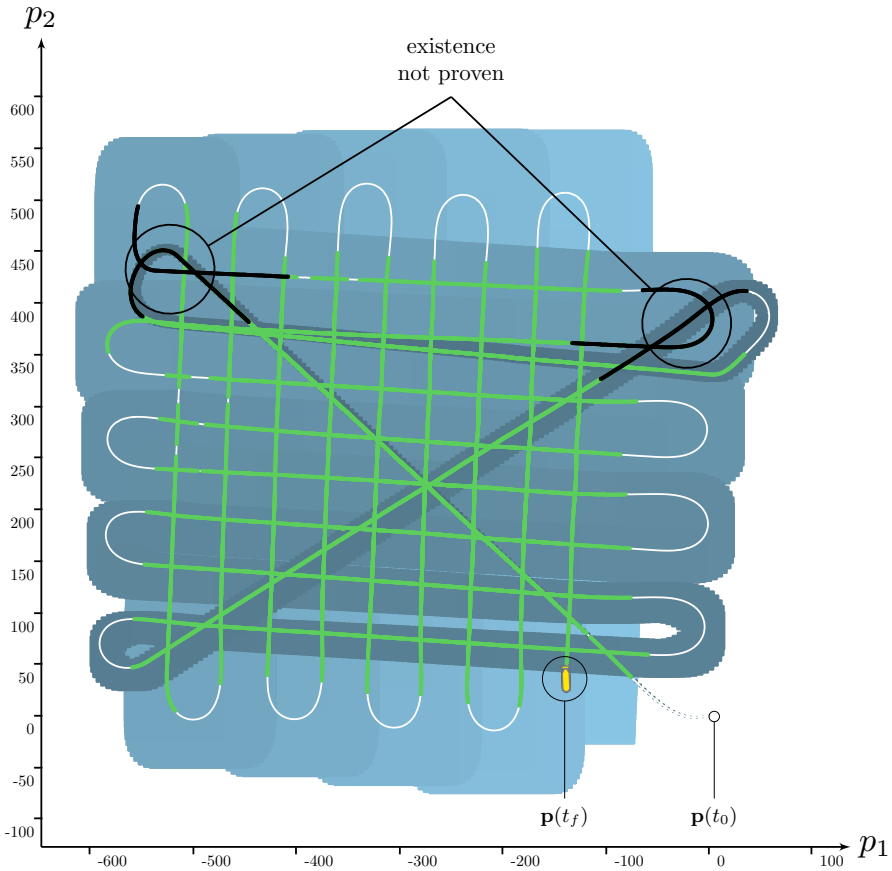


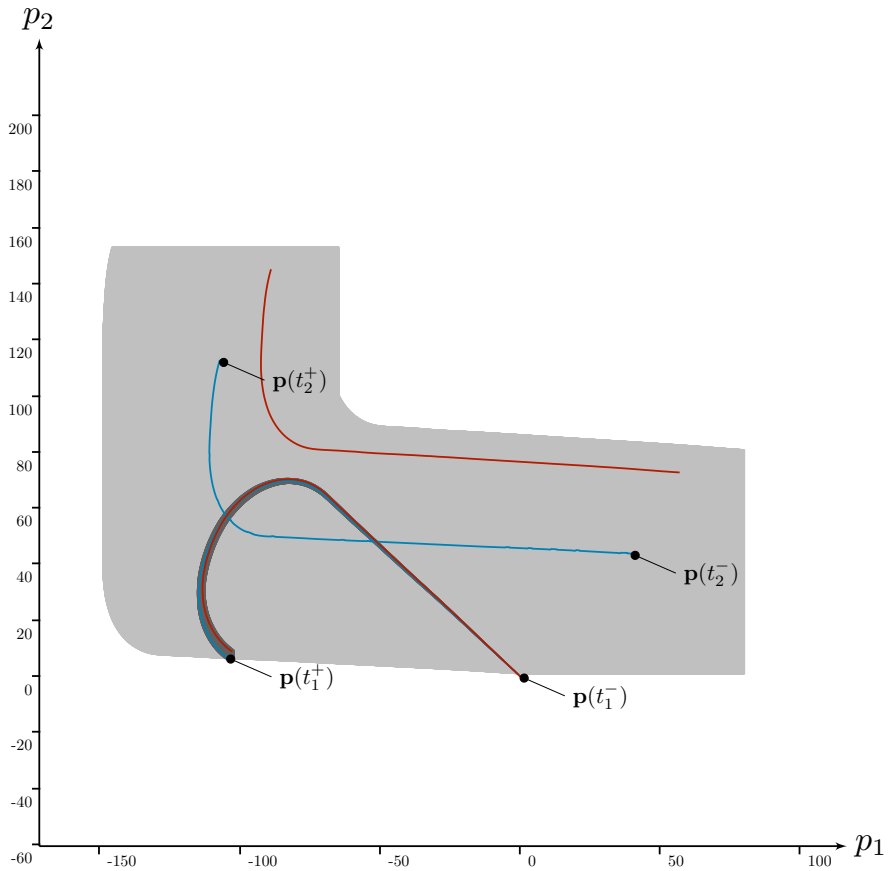
Another experiment: *Daurade*'s mission





Another experiment: *Daurade*'s mission





Proving loops: conclusion

Loop proof $\Leftrightarrow$ **verified existence of a 0** of an uncertain function:

- context where the exact values of the function are not known
- have to deal with a reliable approximation of it

Proving loops: conclusion

Loop proof $\Leftrightarrow$ **verified existence of a 0** of an uncertain function:

- context where the exact values of the function are not known
- have to deal with a reliable approximation of it

Topological degree theory:

- well suited in this case
- applied in a 2d context

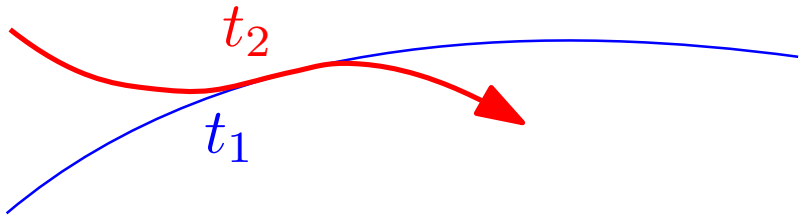
Proving loops: conclusion

Loop proof $\Leftrightarrow$ **verified existence of a 0** of an uncertain function:

- context where the exact values of the function are not known
- have to deal with a reliable approximation of it

Topological degree theory:

- well suited in this case
- applied in a 2d context
- optimal results



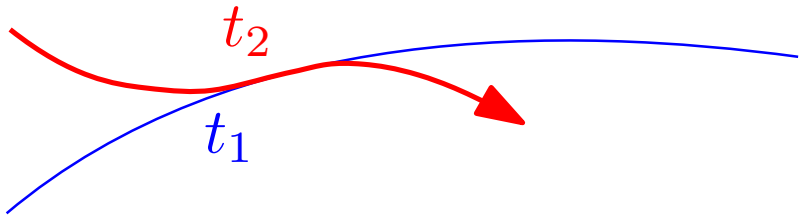
Proving loops: conclusion

Loop proof $\Leftrightarrow$ **verified existence of a 0** of an uncertain function:

- context where the exact values of the function are not known
- have to deal with a reliable approximation of it

Topological degree theory:

- well suited in this case
- applied in a 2d context
- optimal results



Proving the existence of loops in robot trajectories

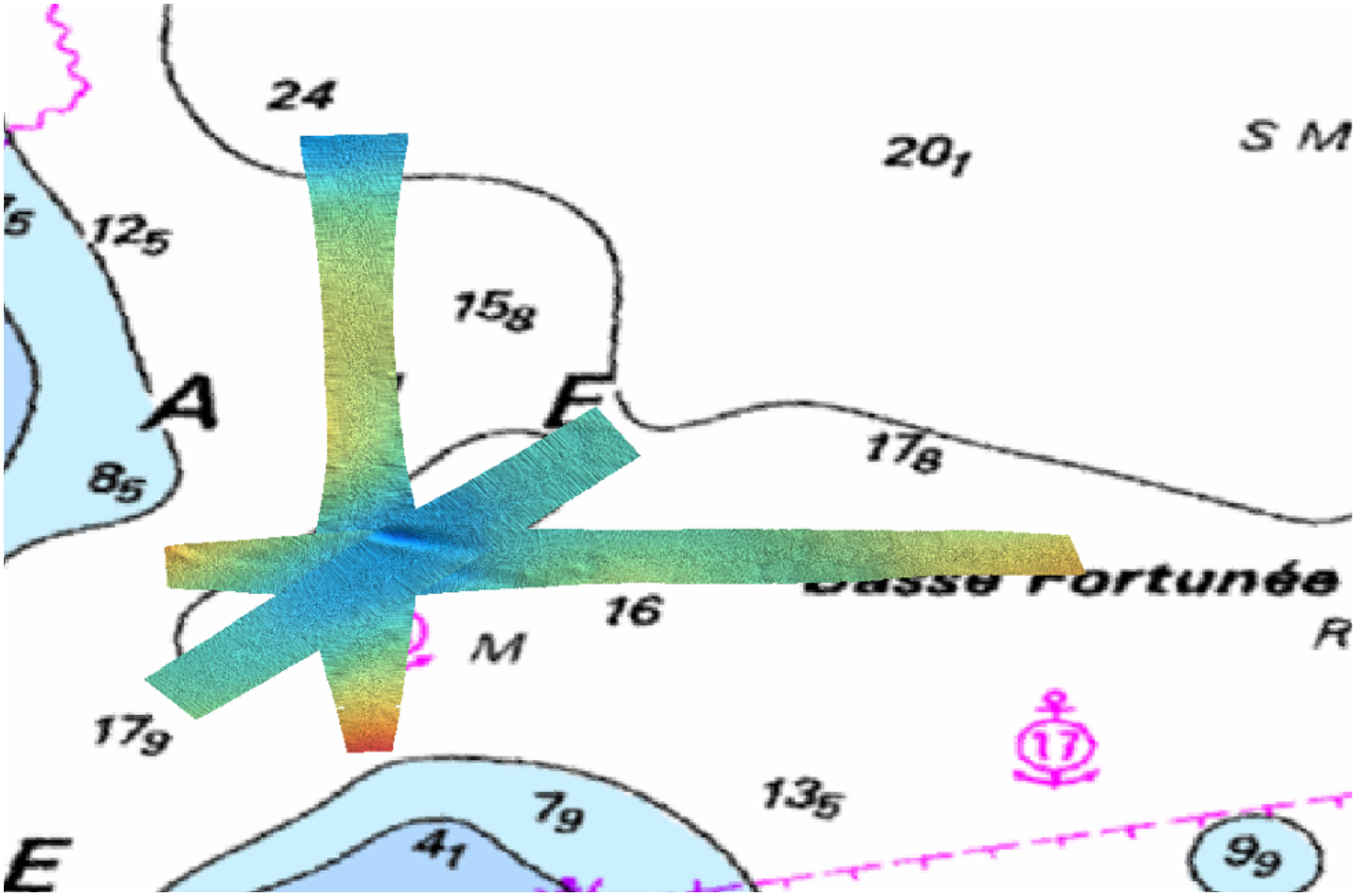
The International Journal of Robotics Research, 2018

Simon Rohou, Peter Franek, Clément Aubry, Luc Jaulin

Robot localization

Ground observations: crossing acquisitions

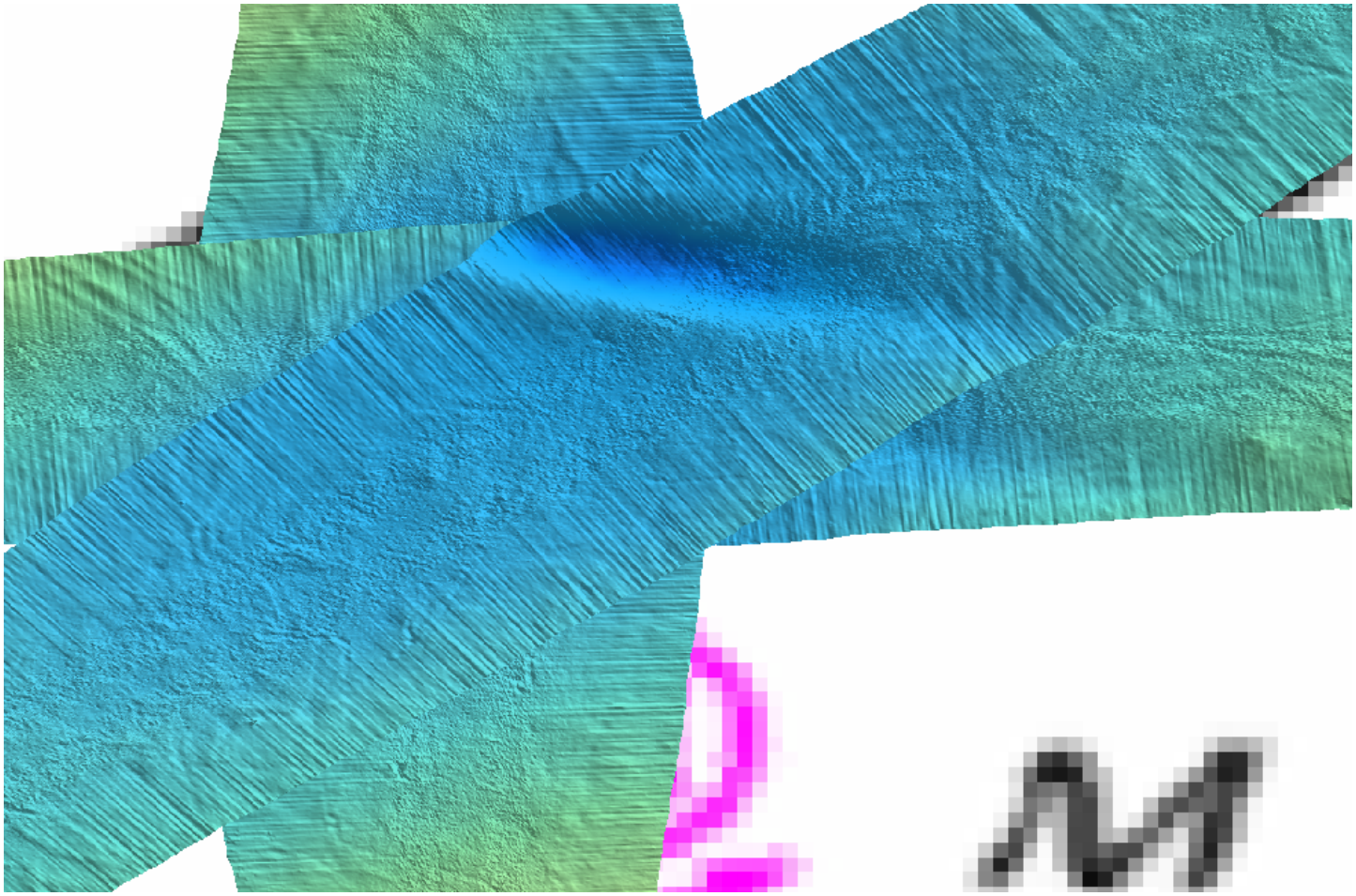
Goal: use bathymetric information for localization purposes



Crossed acquisition tracks (looped trajectories), in case of positioning drift.

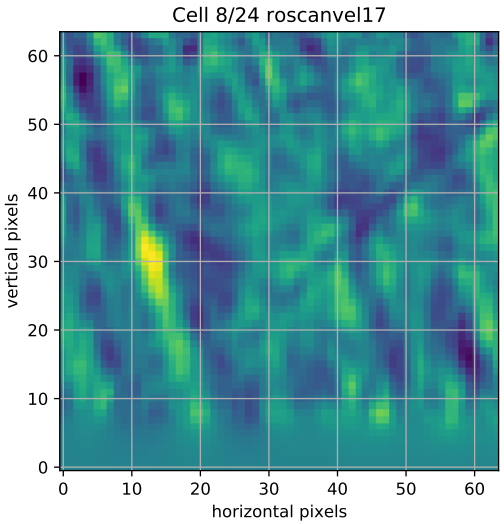
Ground observations: crossing acquisitions

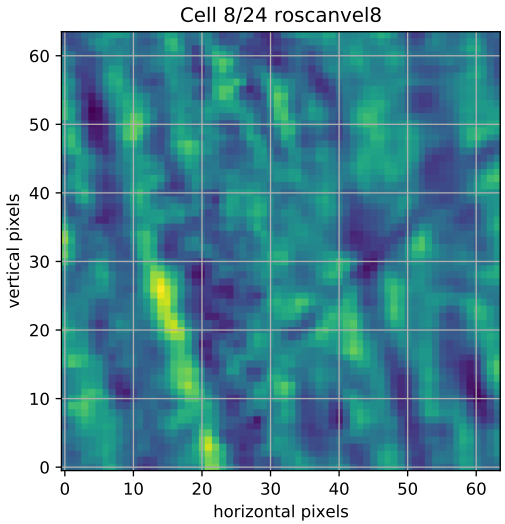
Goal: use bathymetric information for localization purposes

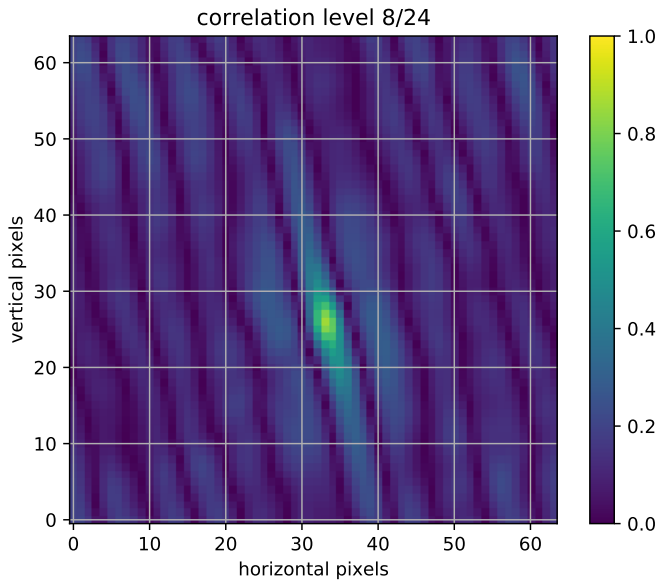


Crossed acquisition tracks (looped trajectories), in case of positioning drift.

Correlation of two ground observations





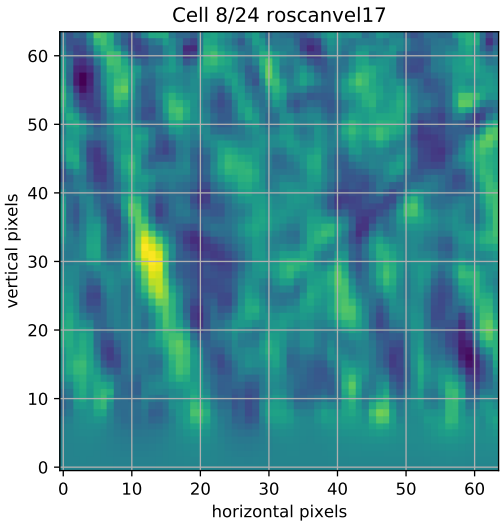


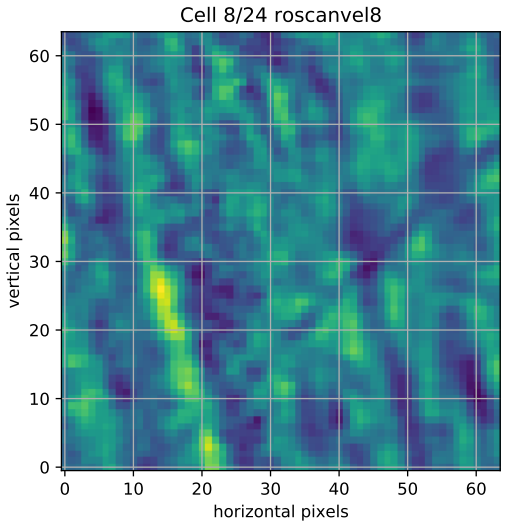
- Correlation: 0.84

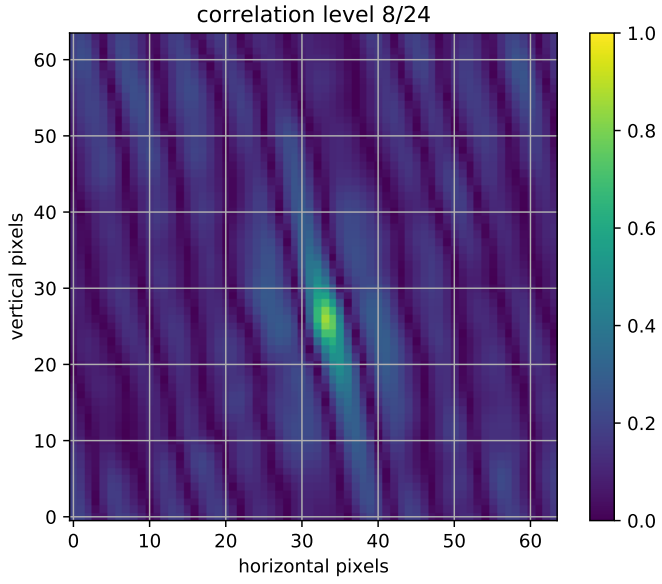
  $d_1 = 0.58\text{m}$

  $d_2 = \text{-}2.53\text{m}$

Correlation of two ground observations







- Correlation: 0.84

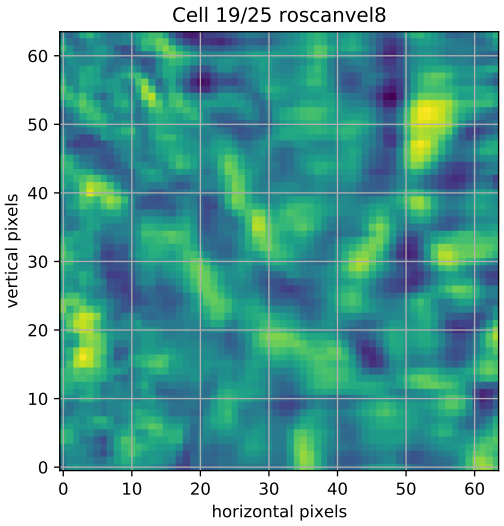
  $d_1 = 0.58$m

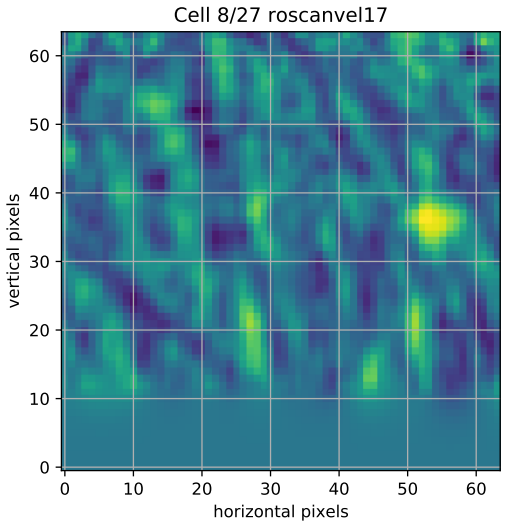
  $d_2 = -2.53$m

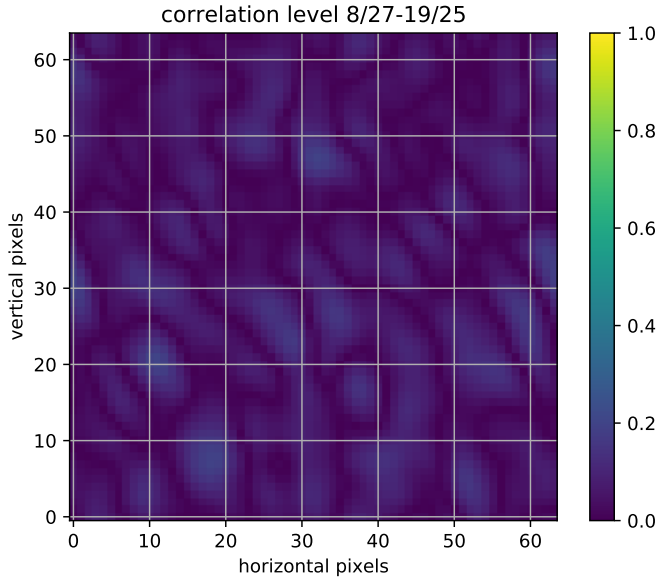
- Enclosure [**d**]

  based on some confidence level

Non-correlation of two ground observations

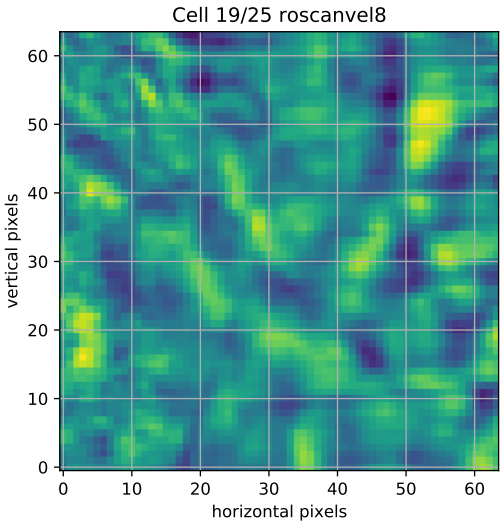


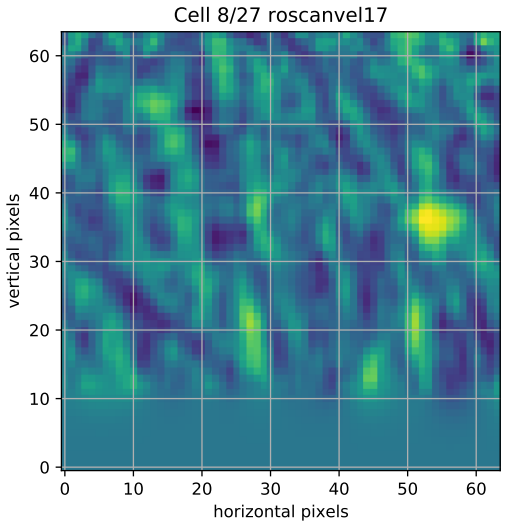


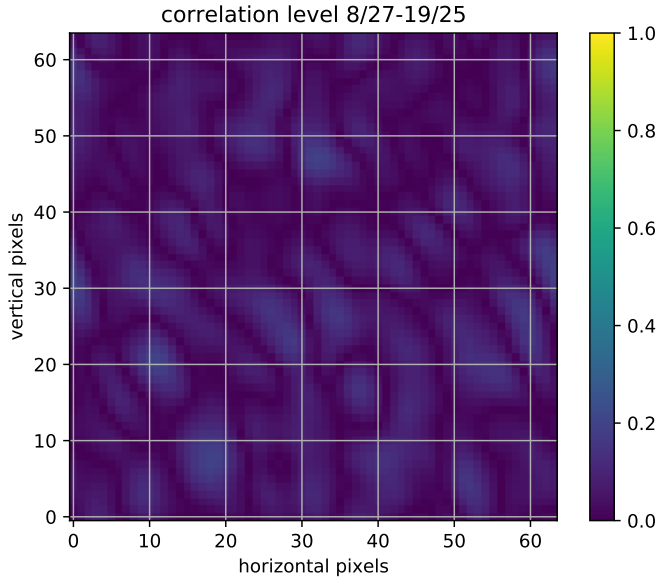


– Correlation: 0.17

Non-correlation of two ground observations







- Correlation: 0.17

- Enclosure $[\mathbf{d}] = [-\infty, \infty]^2$
  (unreliable loop closure)

Localization: constraint programming approach

State estimation associated with one loop.
One t-pair $(\tilde{t_1}, \tilde{t_2})$ is selected randomly in a verified loop-set $\mathbb{T}_i$.

- **Variables:**
  - $\mathbf{p}(\cdot)$, $\mathbf{v}(\cdot) \in [t_0, t_f] \rightarrow \mathbb{R}^2$
  - $\mathbf{d} \in \mathbb{R}^2$
  - $(\tilde{t_1}, \tilde{t_2}) \in \mathbb{R}^2$
- **Constraints:**
  - $\dot{\mathbf{p}}(t) = \mathbf{v}(t)$
  - $\mathbf{p}(\tilde{t_1}) + \mathbf{d} = \mathbf{p}(\tilde{t_2})$

Localization: constraint programming approach

State estimation associated with one loop.

One t-pair $(\tilde{t_1}, \tilde{t_2})$ is selected randomly in a verified loop-set $\mathbb{T}_i$.

Decomposition

- **Variables:**
  - $\mathbf{p}(\cdot)$, $\mathbf{v}(\cdot) \in [t_0, t_f] \rightarrow \mathbb{R}^2$
  - $\mathbf{d}$, $\mathbf{q}_1$, $\mathbf{q}_2 \in \mathbb{R}^2$
  - $(\tilde{t_1}, \tilde{t_2}) \in \mathbb{R}^2$

- **Constraints:**
  - $\dot{\mathbf{p}}(t) = \mathbf{v}(t)$
  - $\mathbf{q}_1 = \mathbf{p}(\tilde{t_1})$
  - $\mathbf{q}_2 = \mathbf{p}(\tilde{t_2})$
  - $\mathbf{d} = \mathbf{q}_2 - \mathbf{q}_1$

Localization: constraint programming approach

State estimation associated with one loop.

One t-pair $(\tilde{t_1}, \tilde{t_2})$ is selected randomly in a verified loop-set $\mathbb{T}_i$.

Decomposition

- **Variables:**
  - $\mathbf{p}(\cdot)$, $\mathbf{v}(\cdot) \in [t_0, t_f] \rightarrow \mathbb{R}^2$
  - $\mathbf{d}$, $\mathbf{q}_1$, $\mathbf{q}_2 \in \mathbb{R}^2$
  - $(\tilde{t_1}, \tilde{t_2}) \in \mathbb{R}^2$
- **Interval domains:**
- **Constraints:**
  - $\dot{\mathbf{p}}(t) = \mathbf{v}(t)$
  - $\mathbf{q}_1 = \mathbf{p}(\tilde{t_1})$
  - $\mathbf{q}_2 = \mathbf{p}(\tilde{t_2})$
  - $\mathbf{d} = \mathbf{q}_2 - \mathbf{q}_1$
- **Contractors:**

Localization: constraint programming approach

State estimation associated with one loop.

One t-pair $(\tilde{t_1}, \tilde{t_2})$ is selected randomly in a verified loop-set $\mathbb{T}_i$.

Decomposition

- **Variables:**
  - $\mathbf{p}(\cdot)$, $\mathbf{v}(\cdot) \in [t_0, t_f] \to \mathbb{R}^2$
  - $\mathbf{d}$, $\mathbf{q}_1$, $\mathbf{q}_2 \in \mathbb{R}^2$
  - $(\tilde{t_1}, \tilde{t_2}) \in \mathbb{R}^2$
- **Constraints:**
  - $\dot{\mathbf{p}}(t) = \mathbf{v}(t)$
  - $\mathbf{q}_1 = \mathbf{p}(\tilde{t_1})$
  - $\mathbf{q}_2 = \mathbf{p}(\tilde{t_2})$
  - $\mathbf{d} = \mathbf{q}_2 - \mathbf{q}_1$
- **Interval domains:**
  - $[\mathbf{p}](\cdot)$, $[\mathbf{v}](\cdot) \in [t_0, t_f] \to \mathbb{IR}^2$
  - $[\mathbf{d}]$, $[\mathbf{q}_1]$, $[\mathbf{q}_2] \in \mathbb{IR}^2$
- **Contractors:**
  - $\mathcal{C}_{\text{deriv}}\left([\mathbf{p}](\cdot), [\mathbf{v}](\cdot)\right)$
  - $\mathcal{C}_{\text{eval}}\left(\tilde{t_1}, [\mathbf{q}_1], [\mathbf{p}](\cdot)\right)$
  - $\mathcal{C}_{\text{eval}}\left(\tilde{t_2}, [\mathbf{q}_2], [\mathbf{p}](\cdot)\right)$
  - $\mathcal{C}_{-}\left([\mathbf{q}_2], [\mathbf{q}_1], [\mathbf{d}]\right)$

Example of application

www.codac.io

Possible implementation of the SLAM algorithm:

Append to a new contractor network:

- $\mathcal{C}_{\text{deriv}}([\mathbf{p}](\cdot), [\mathbf{v}](\cdot))$

While fixpoint is not reached:

- compute/update loop-sets $\mathbb{T}$ from $[\mathbf{p}](\cdot), [\mathbf{v}](\cdot)$
- for each connected subset $\mathbb{T}_i$ of $\mathbb{T}$
  - if zero verification on $\mathbb{T}_i$:
    - select $(\tilde{t_1}, \tilde{t_2}) \in \mathbb{T}_i$
    - init $[\mathbf{q}_2] \leftarrow [-\infty, \infty]^2$, $[\mathbf{q}_1] \leftarrow [-\infty, \infty]^2$
    - compute bounded correlation $[\mathbf{d}]$ between $\mathbf{g}(\hat{\mathbf{x}}(\tilde{t_1}))$, $\mathbf{g}(\hat{\mathbf{x}}(\tilde{t_2}))$

      Append to a contractor network:
    - $\mathcal{C}_{-}([\mathbf{d}], [\mathbf{q}_2], [\mathbf{q}_1])$
    - $\mathcal{C}_{\text{eval}}(\tilde{t_1}, [\mathbf{q}_1], [\mathbf{p}](\cdot), [\mathbf{v}](\cdot))$
    - $\mathcal{C}_{\text{eval}}(\tilde{t_2}, [\mathbf{q}_2], [\mathbf{p}](\cdot), [\mathbf{v}](\cdot))$

  Propagate contractions in the contractor network.

Example of application

Possible implementation of the SLAM algorithm:

Append to a new contractor network:

- $\mathcal{C}_{\text{deriv}}([\mathbf{p}](\cdot), [\mathbf{v}](\cdot))$

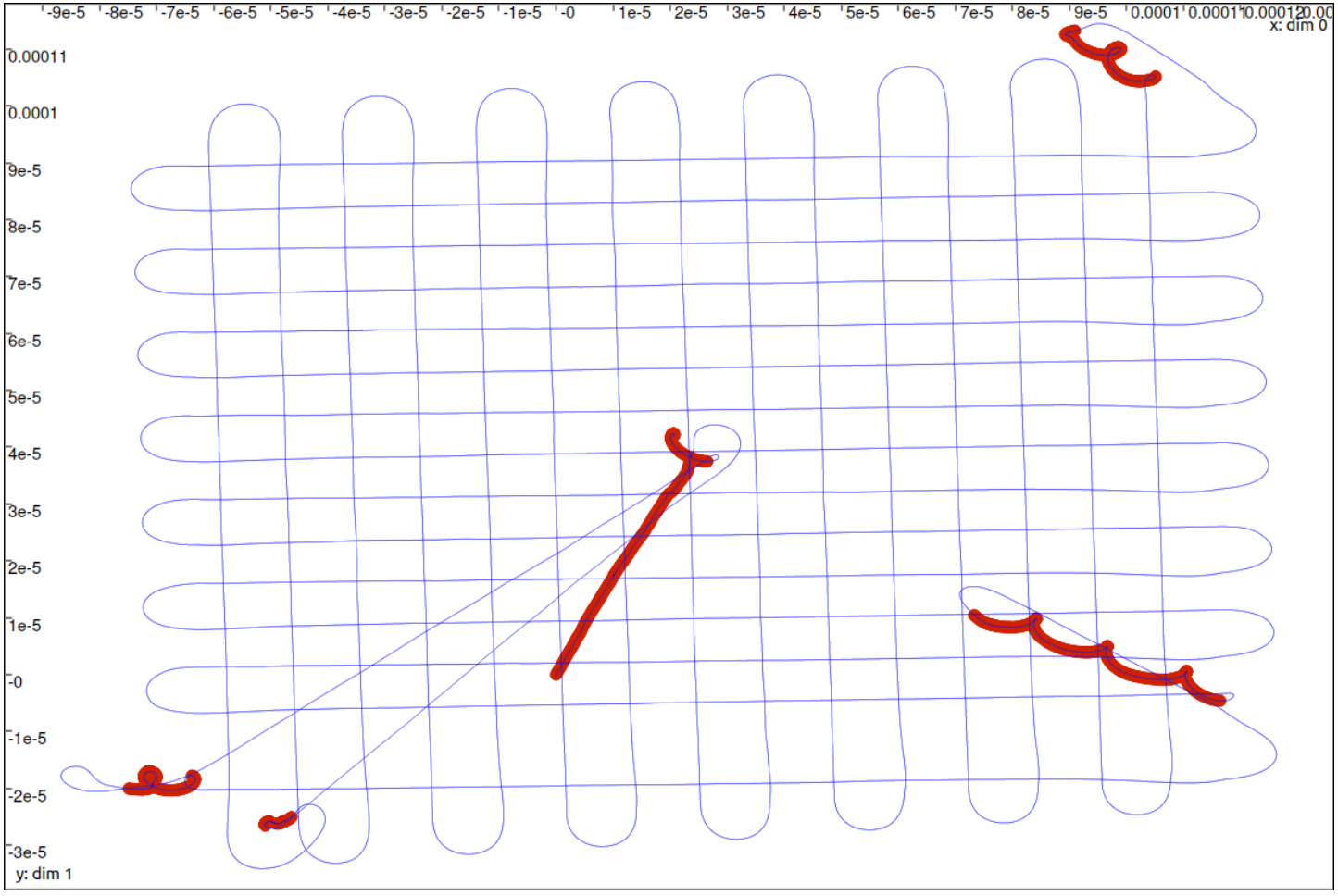
While fixpoint is not reached ($\Leftrightarrow$ while new loops are verified):

- compute/update loop-sets $\mathbb{T}$ from $[\mathbf{p}](\cdot), [\mathbf{v}](\cdot)$
- for each connected subset $\mathbb{T}_i$ of $\mathbb{T}$
  - if zero verification on $\mathbb{T}_i$:
    - select $(\tilde{t_1}, \tilde{t_2}) \in \mathbb{T}_i$
    - init $[\mathbf{q}_2] \leftarrow [-\infty, \infty]^2$, $[\mathbf{q}_1] \leftarrow [-\infty, \infty]^2$
    - compute bounded correlation $[\mathbf{d}]$ between $\mathbf{g}(\hat{\mathbf{x}}(\tilde{t_1}))$, $\mathbf{g}(\hat{\mathbf{x}}(\tilde{t_2}))$

      Append to a contractor network:
    - $\mathcal{C}_{-}([\mathbf{d}], [\mathbf{q}_2], [\mathbf{q}_1])$
    - $\mathcal{C}_{\text{eval}}(\tilde{t_1}, [\mathbf{q}_1], [\mathbf{p}](\cdot), [\mathbf{v}](\cdot))$
    - $\mathcal{C}_{\text{eval}}(\tilde{t_2}, [\mathbf{q}_2], [\mathbf{p}](\cdot), [\mathbf{v}](\cdot))$

Propagate contractions in the contractor network.

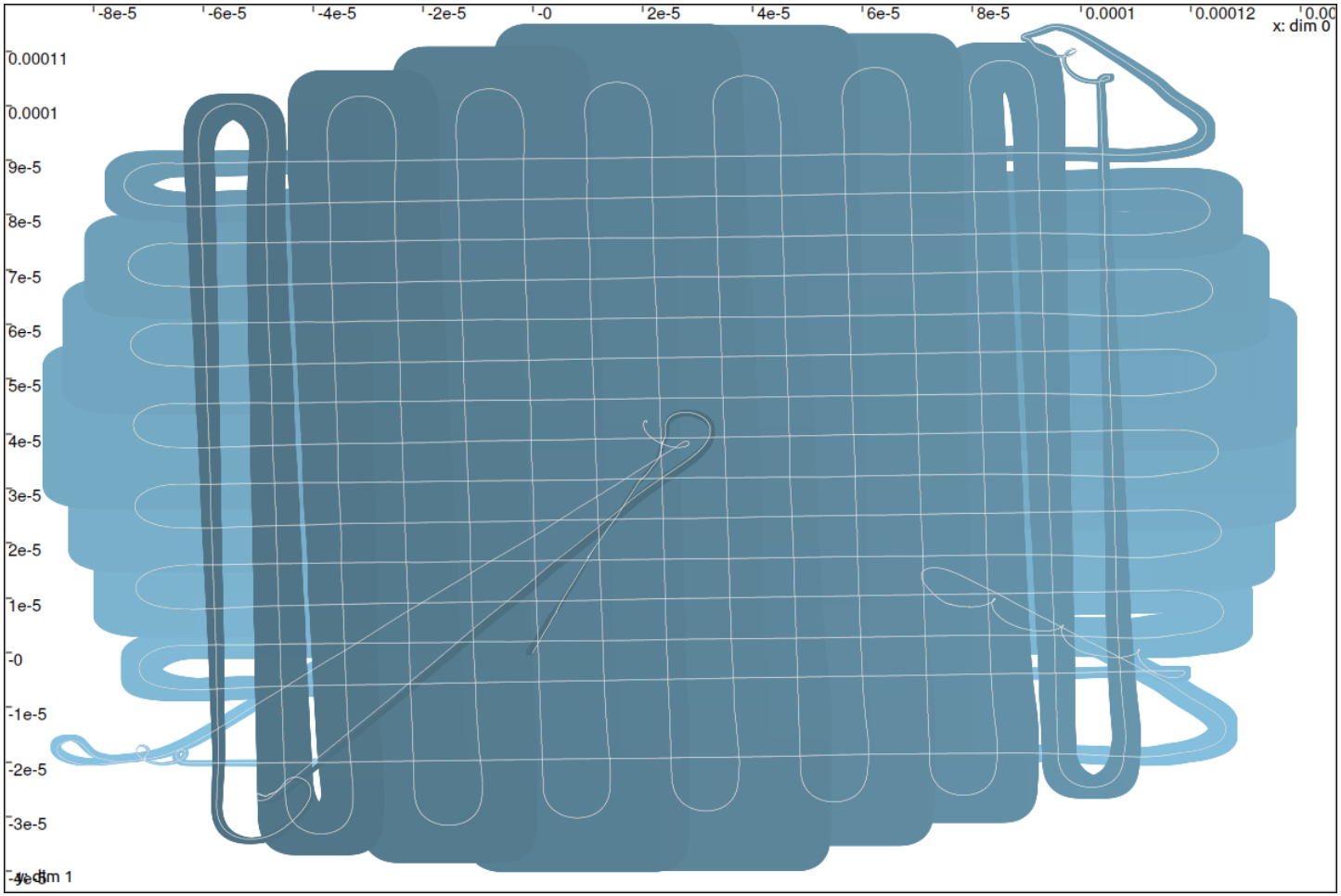
Experiment in the "Rade de Brest" (Boustrophédon)



Multi-boustrophedon pattern (actual trajectory performed by Daurade). Red points are surface positions.

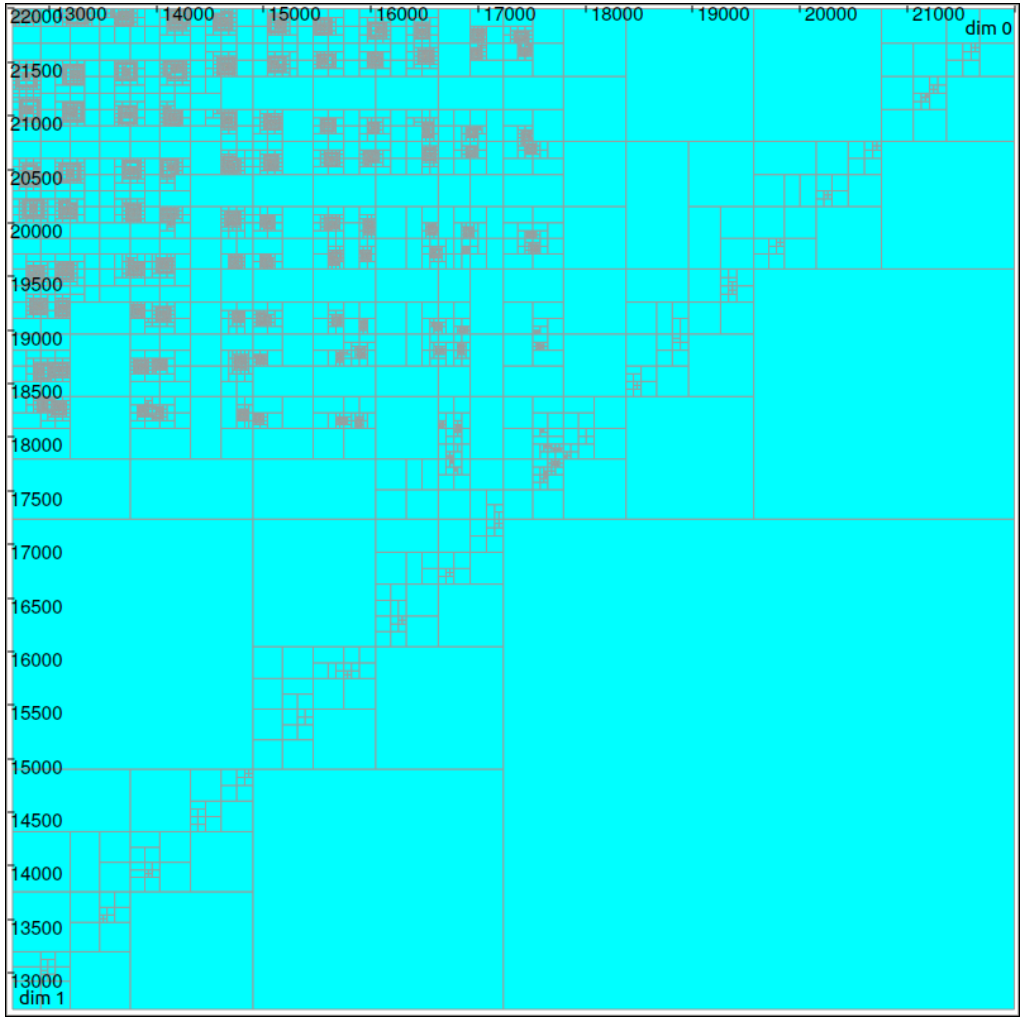
Deadreckoning

Enclosure of trajectories without external measurements:



Initial tube $[\mathbf{p}](\cdot)$ *before SLAM (inertial/DVL odometry + GNSS fixes).*

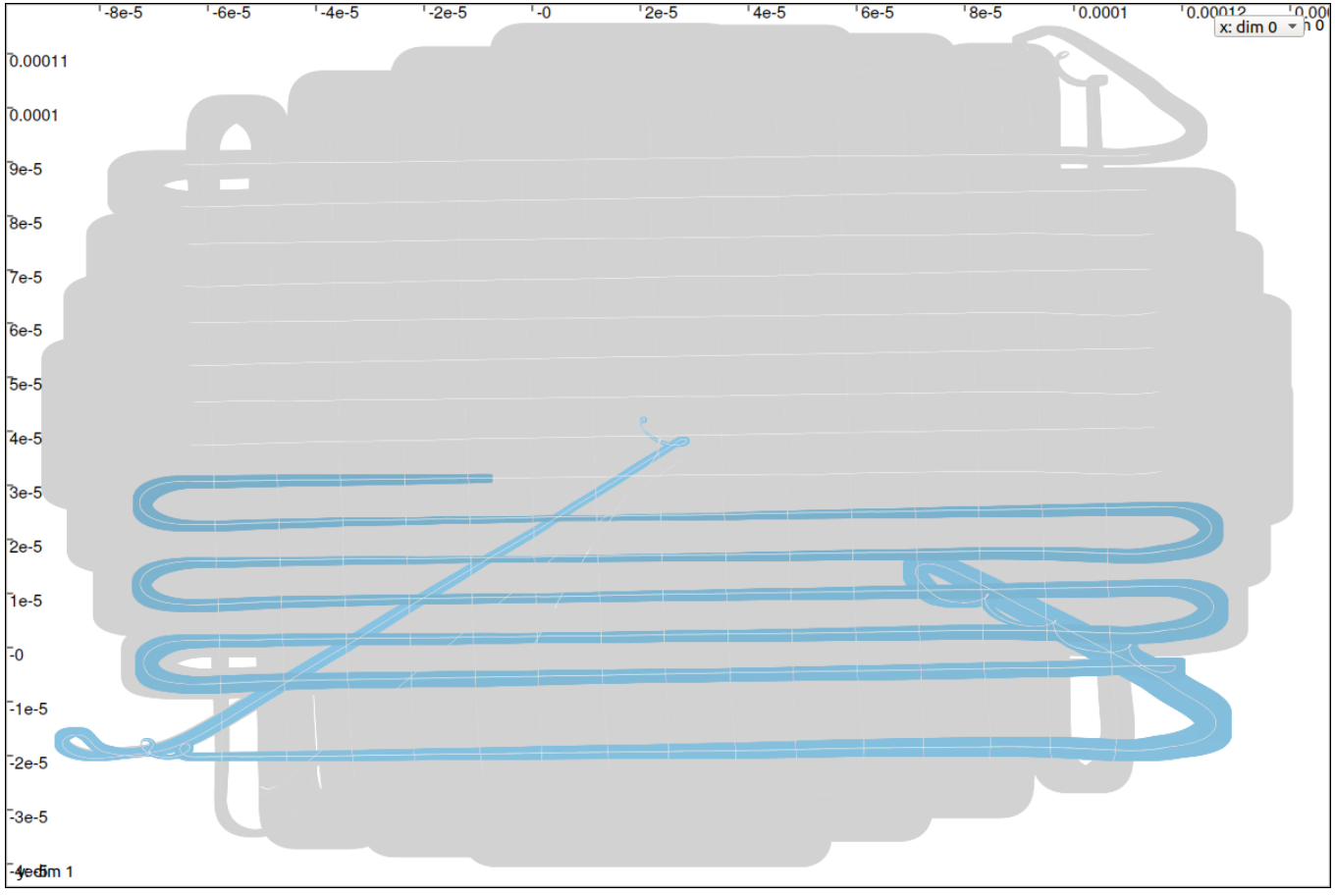
t-plane of the mission (loops summary)



Computed t-plane $[0, t_{\max}]^2$ – hundreds of loops are detected.

SLAM results

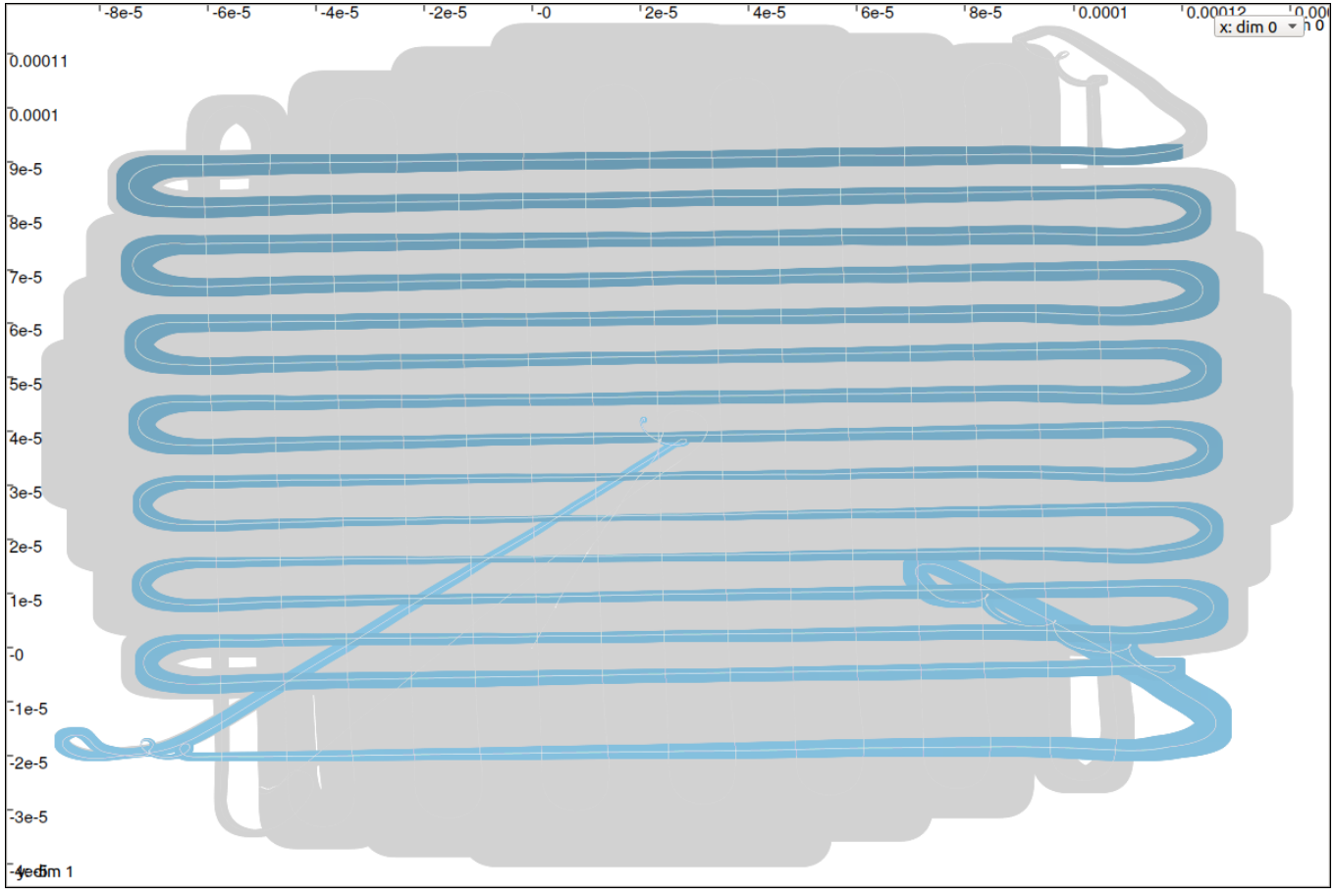
Contracted enclosure of the feasible trajectories,
with external measurements (seafloor perceptions):



Final tube $[\mathbf{p}](\cdot)$ *after SLAM.*

SLAM results

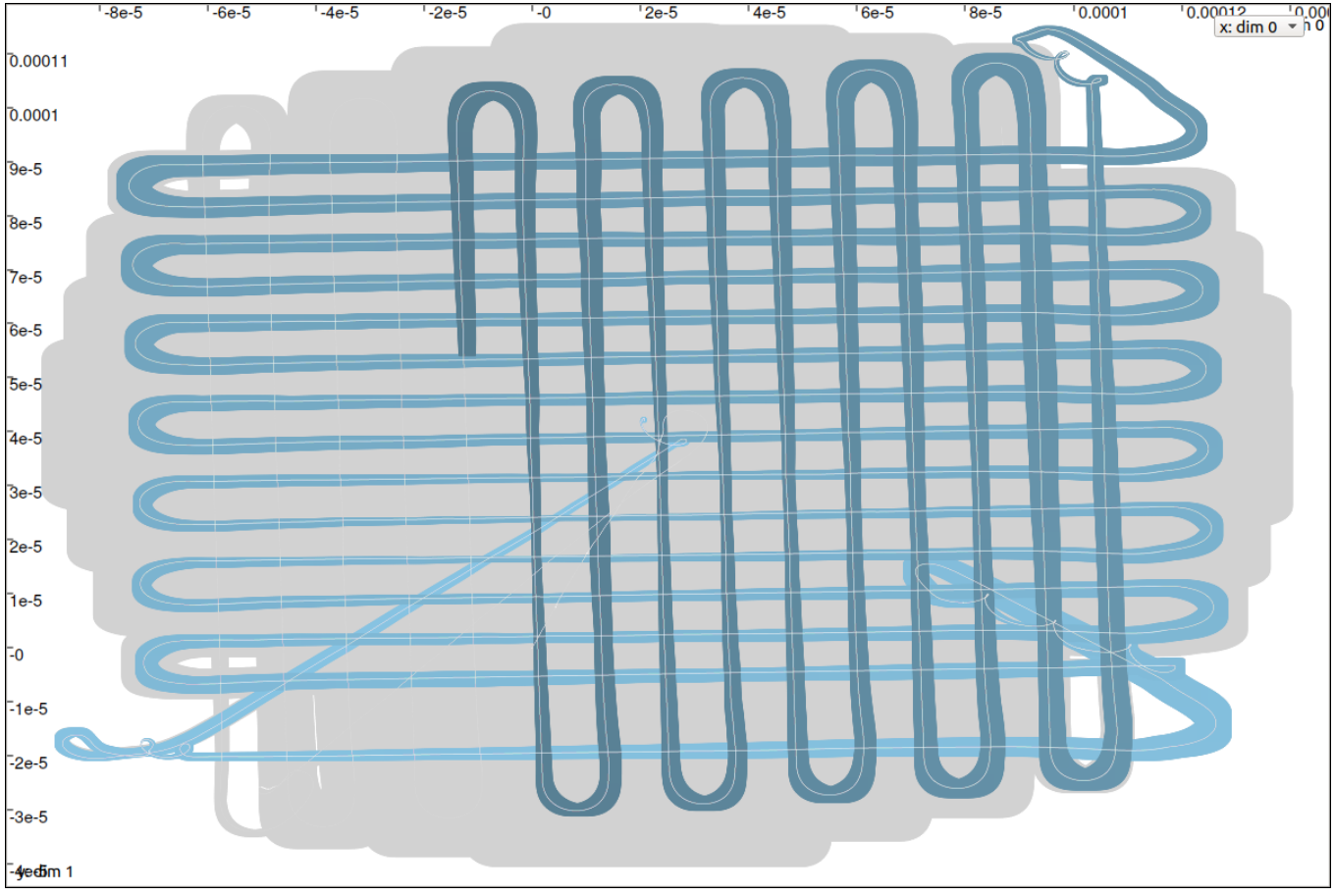
Contracted enclosure of the feasible trajectories,
with external measurements (seafloor perceptions):



Final tube $[\mathbf{p}](\cdot)$ *after SLAM.*

SLAM results

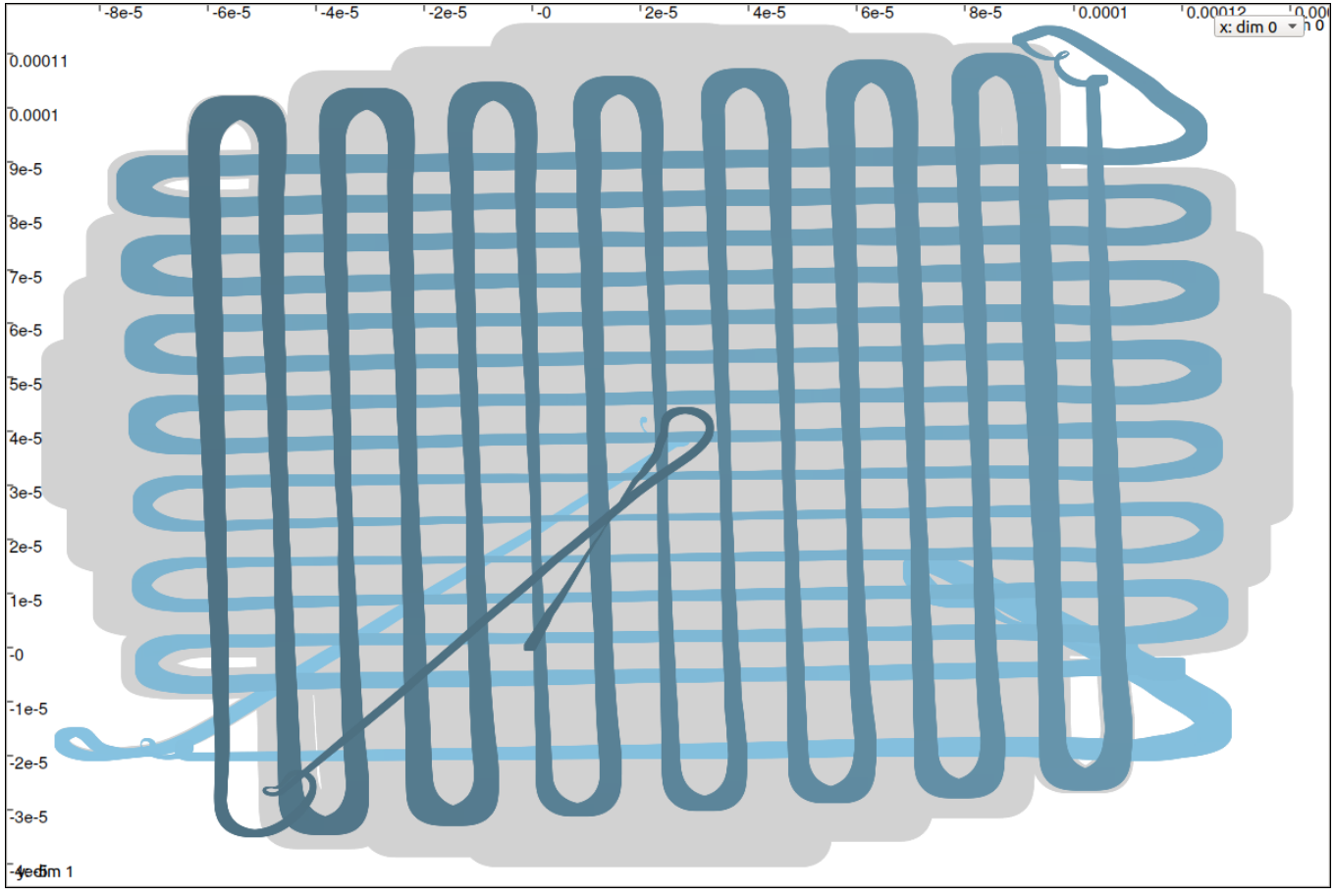
Contracted enclosure of the feasible trajectories,
with external measurements (seafloor perceptions):



Final tube $[\mathbf{p}](\cdot)$ *after SLAM.*

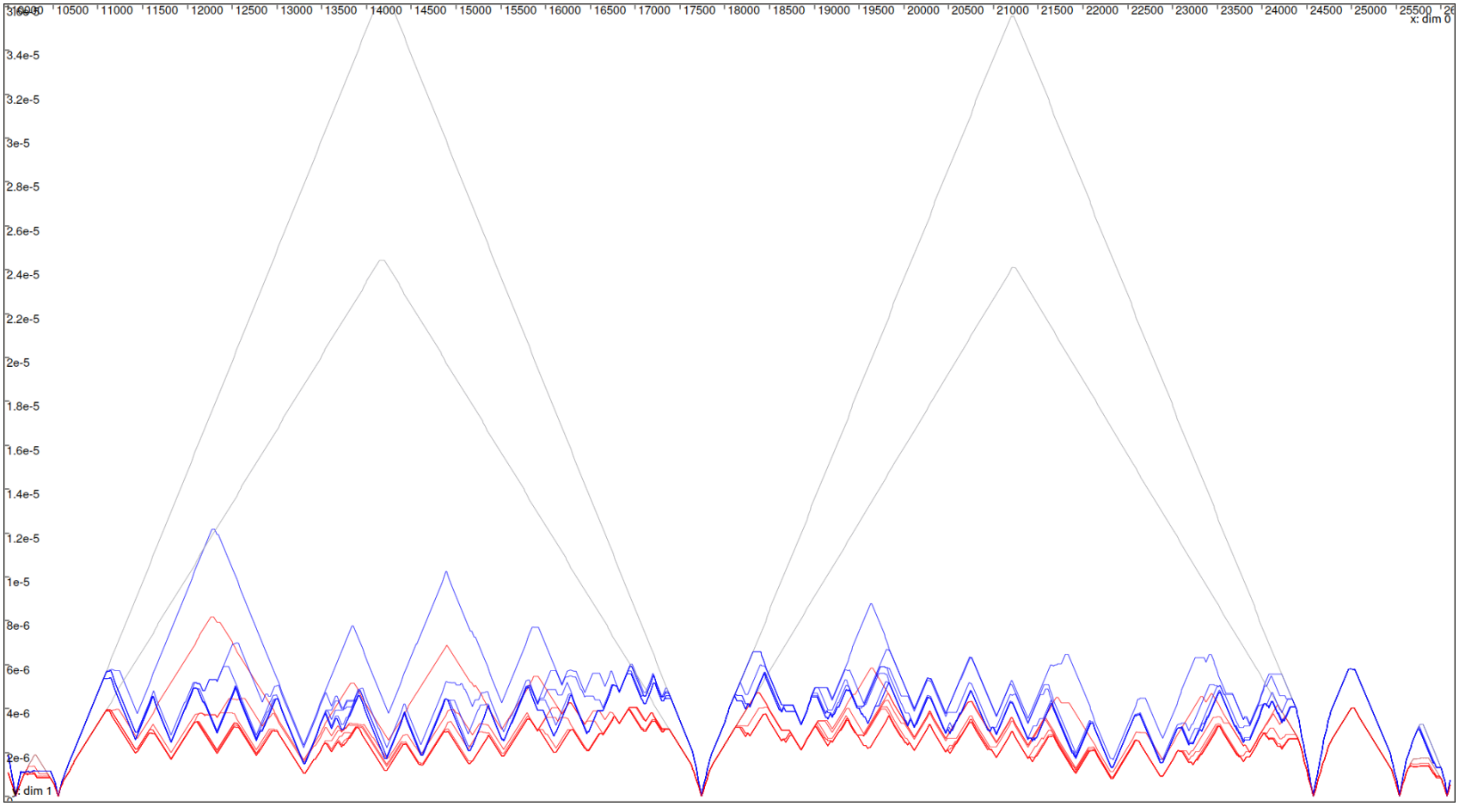
SLAM results

Contracted enclosure of the feasible trajectories,
with external measurements (seafloor perceptions):



Final tube $[\mathbf{p}](\cdot)$ *after SLAM.*

SLAM results

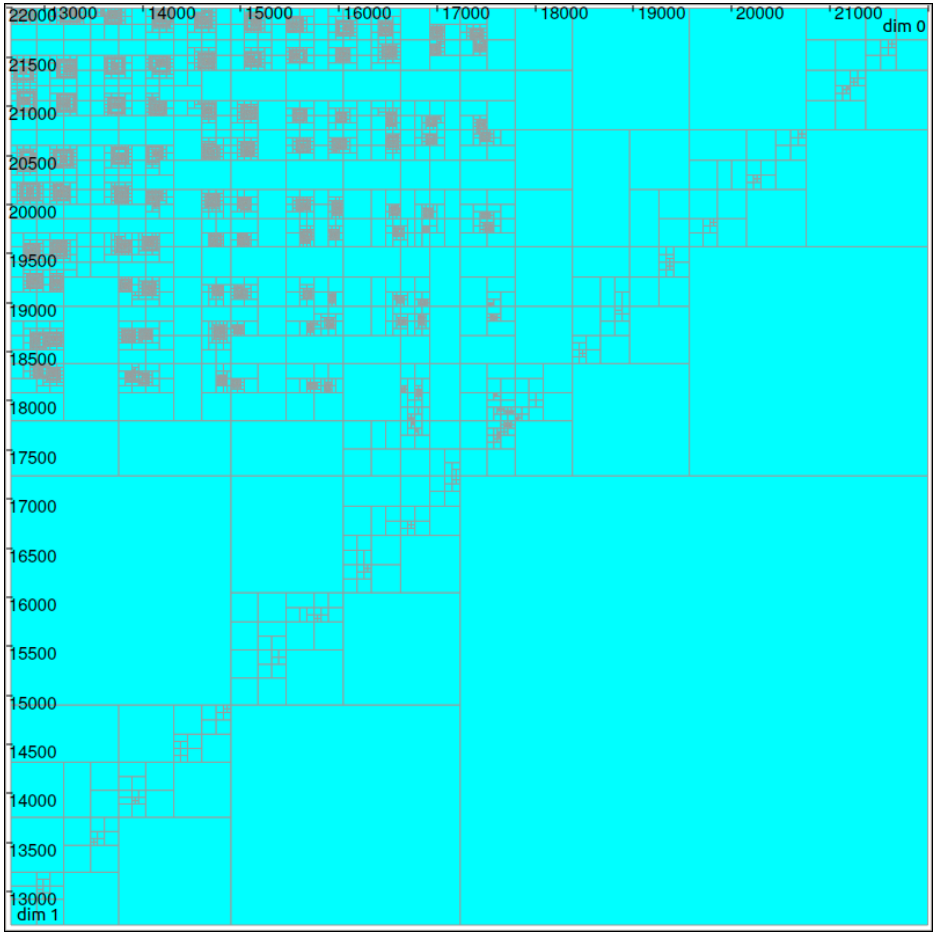


Tubes thickness along time: $\mathrm{width}([\mathbf{p}](t))$.

- *in gray: tubes without SLAM (deadreckoning only = significant drift)*
- *in blue/red: with SLAM in* p_1/p_2

t-plane of the mission (iterative computations)

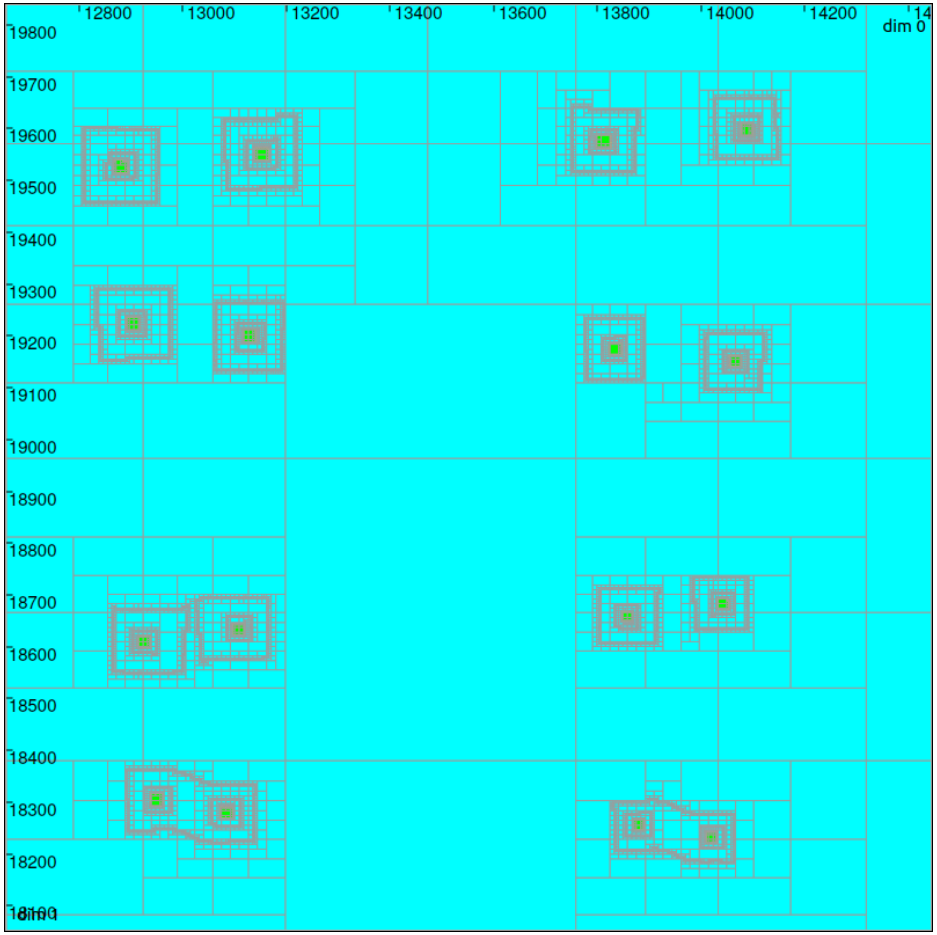
The associated t-plane is iteratively contracted with respect to improvements on $[\mathbf{p}](t)$:



Computed t-plane $[0, t_{\max}]^2$ *– revealing old loop sets boundaries.*

t-plane of the mission (iterative computations)

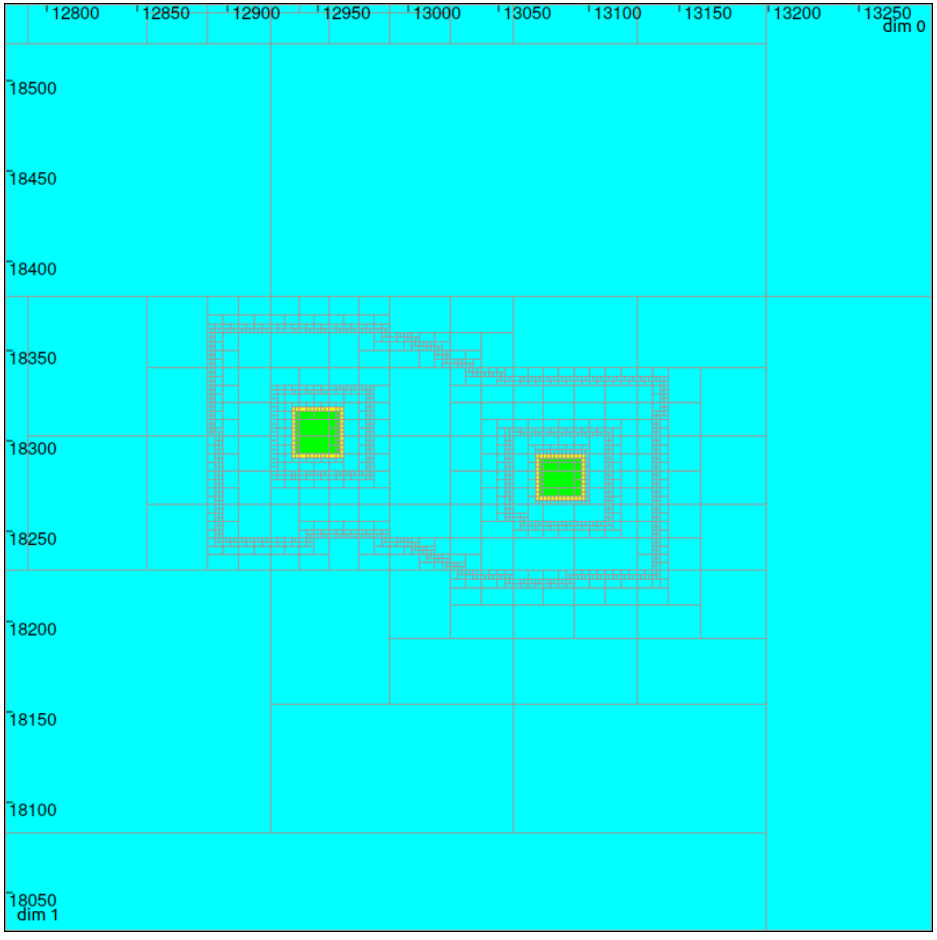
The associated t-plane is iteratively contracted with respect to improvements on $[\mathbf{p}](t)$:



Computed t-plane $[0, t_{\max}]^2$ *– revealing old loop sets boundaries.*

t-plane of the mission (iterative computations)

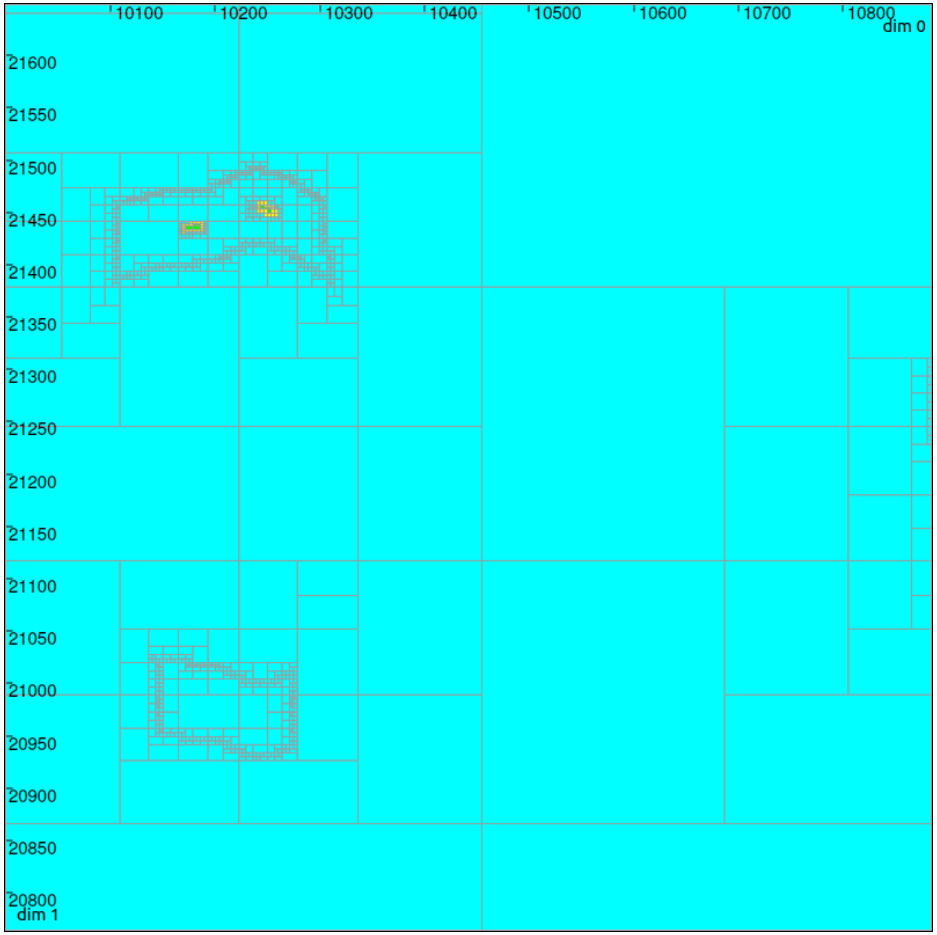
The associated t-plane is iteratively contracted with respect to improvements on $[\mathbf{p}](t)$:



Computed t-plane $[0, t_{\max}]^2$ *– revealing old loop sets boundaries.*

t-plane of the mission (iterative computations)

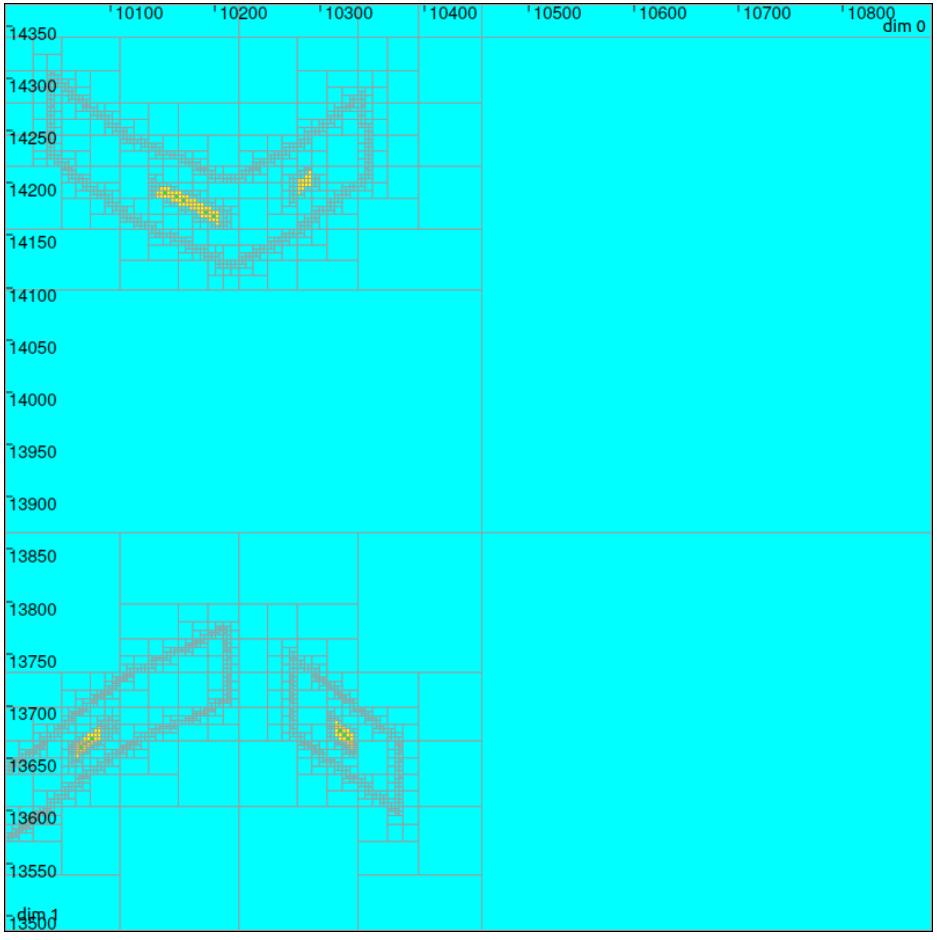
The associated t-plane is iteratively contracted with respect to improvements on $[\mathbf{p}](t)$:



Computed t-plane $[0, t_{\max}]^2$ *– revealing old loop sets boundaries.*

t-plane of the mission (iterative computations)

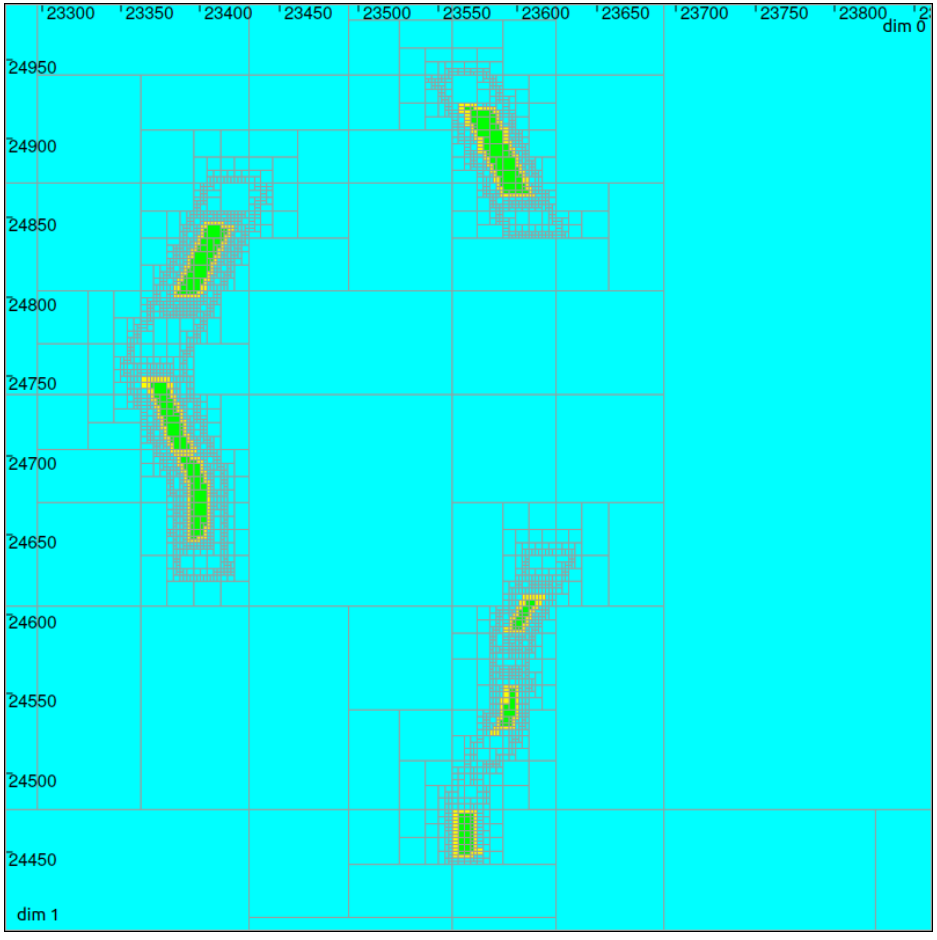
The associated t-plane is iteratively contracted with respect to improvements on $[\mathbf{p}](t)$:



Computed t-plane $[0, t_{\max}]^2$ *– revealing old loop sets boundaries.*

t-plane of the mission (iterative computations)

The associated t-plane is iteratively contracted with respect to improvements on $[\mathbf{p}](t)$:



Computed t-plane $[0, t_{\max}]^2$ *– revealing old loop sets boundaries.*

Conclusion and perspectives

Assets of this approach:

- analyzing ground textures is efficient even over flat seabeds, or during strong tidal changes
- but **requires reliable localization**, whatever:
  - the ground ambiguities/similarities
  - the uncertainties on loops

Future work:

- online SLAM!

Thank you!