Robustness Metrics for Motion Planning and Control with STL Specifications

Joris Verhagen, KTH Royal Institute of Technology
Division of Robotics, Perception and Learning

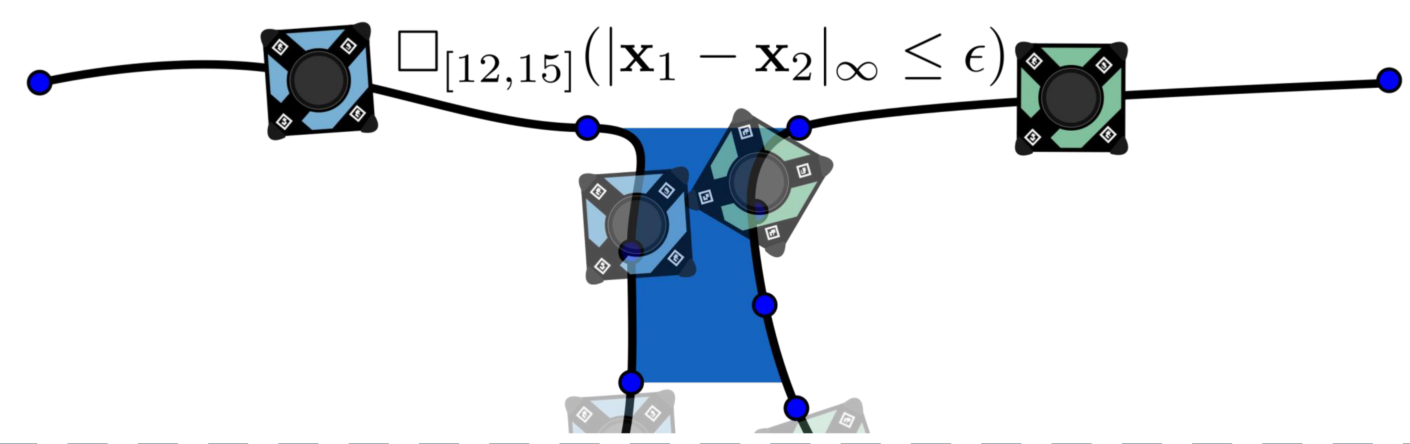
Robust STL Satisfaction: Time-robustness and Disturbance-robustness

Robots may not be able to follow a pre-specified trajectory
- SLAM mapping errors
- External Disturbances
- On-board timing issues
- Parameter uncertainty

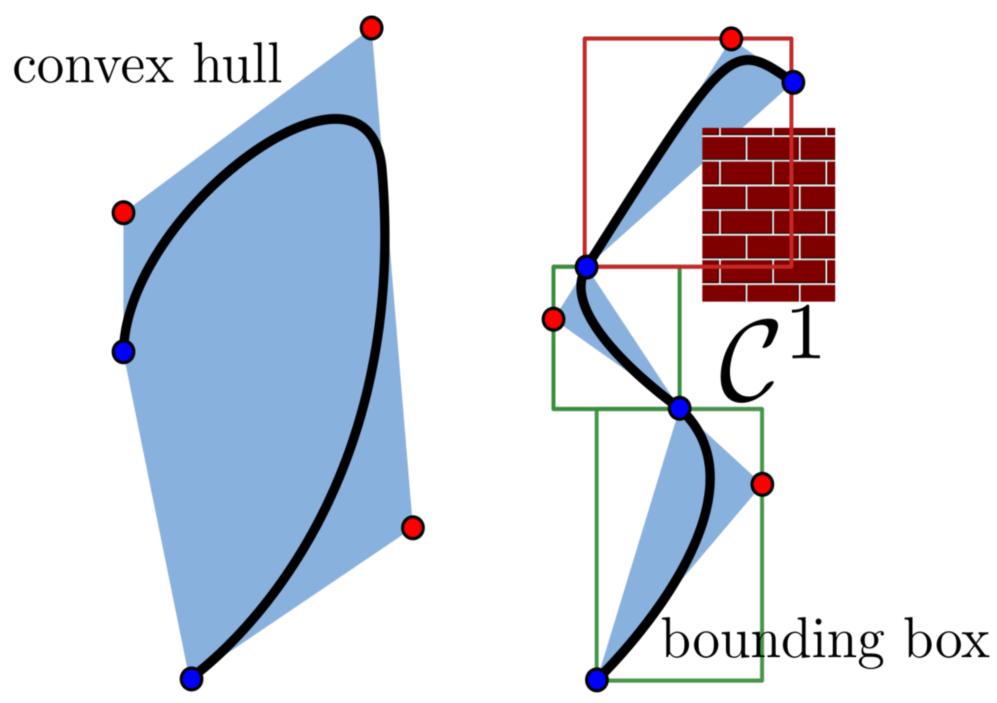
But robots should satisfy a specification as best as possible!

Goal: Maximize robustness metrics

"Robot 1 and 2 should be close enough to hand over a product between 12 and 15 seconds"



Asynchronous Time Robustness

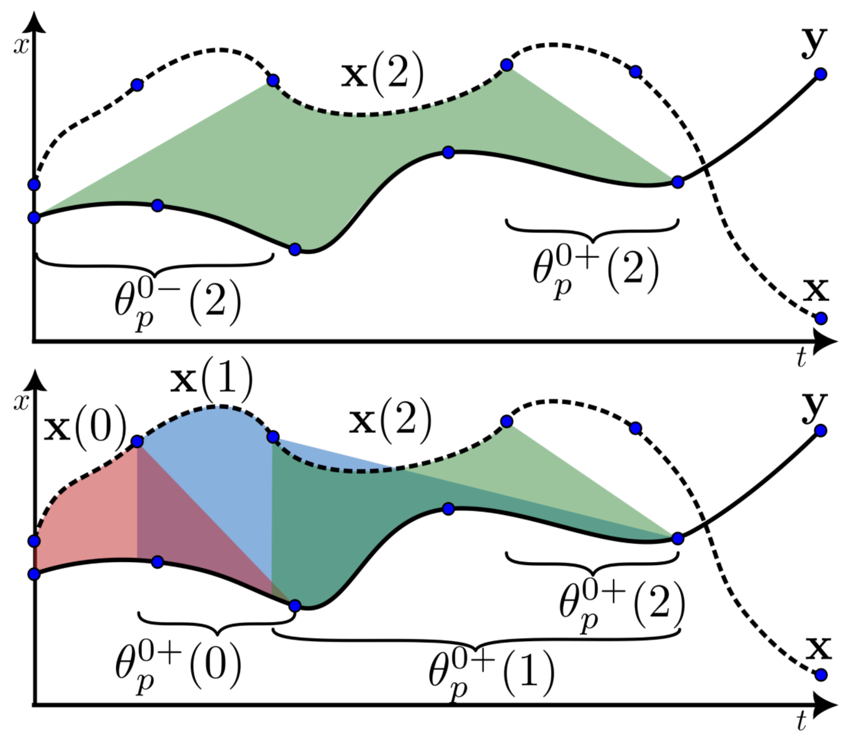


Bezier curves for curvature and time
Continuous-time trajectory and speed-up!

Each state of a robot can be shifted by a unique time-shift

$$\mathbf{x}^p_{\bar{\kappa}}(t) := \{\mathbf{x}_k(t+\kappa_k) | k \in \mathcal{S}_p\}$$

How far can we shift signals such that the predicate still holds

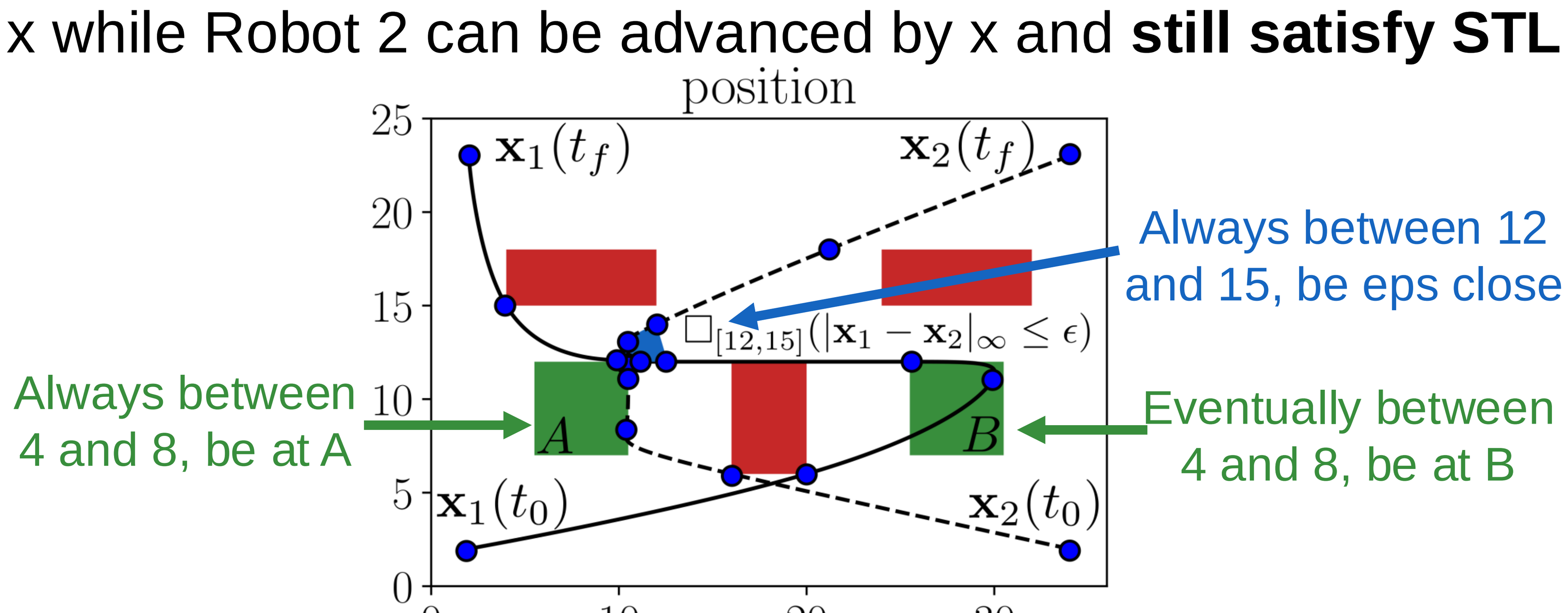


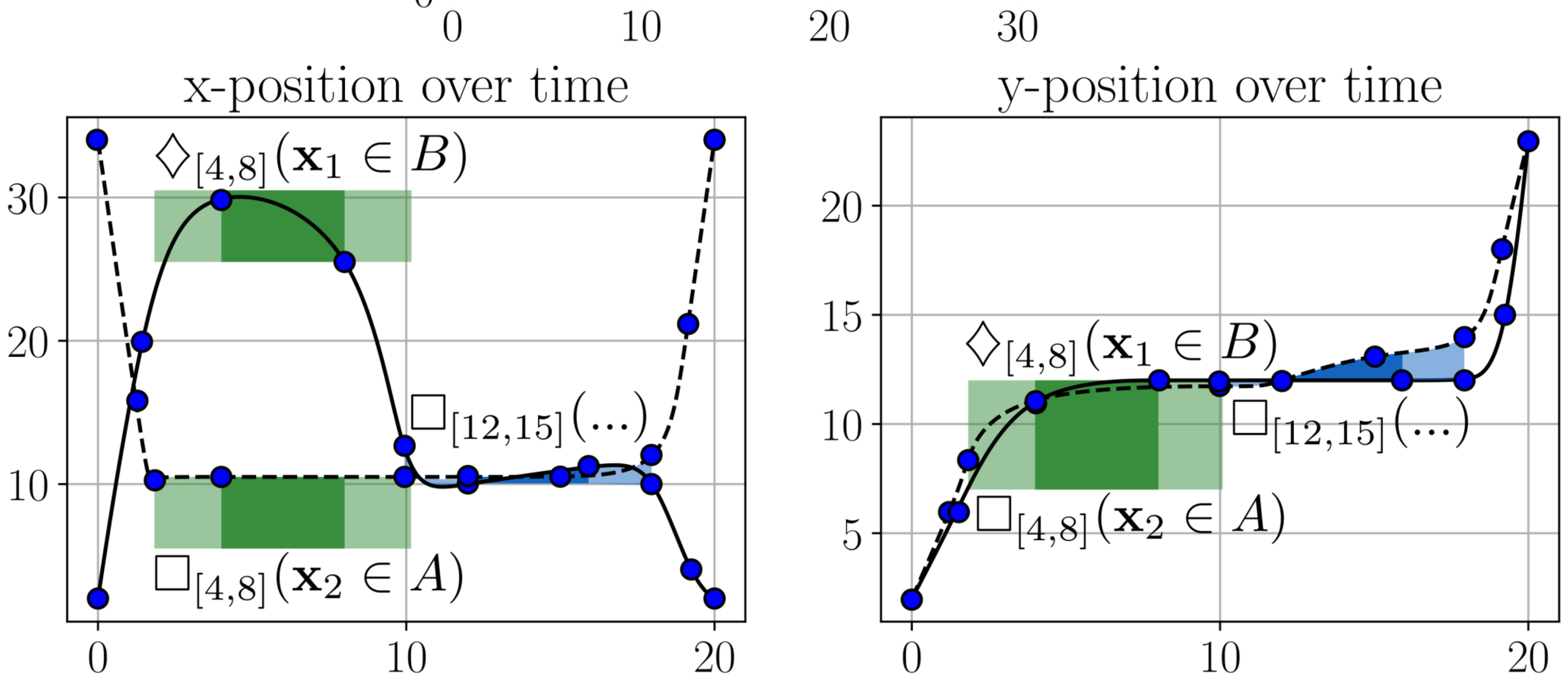
Predicate

$$p := \mathbf{y} \leq \mathbf{x}$$

We need to look at overlap between segments

Asynchronous time robustness: Robot 1 can be delayed by x while Robot 2 can be advanced by x and **still satisfy STL**



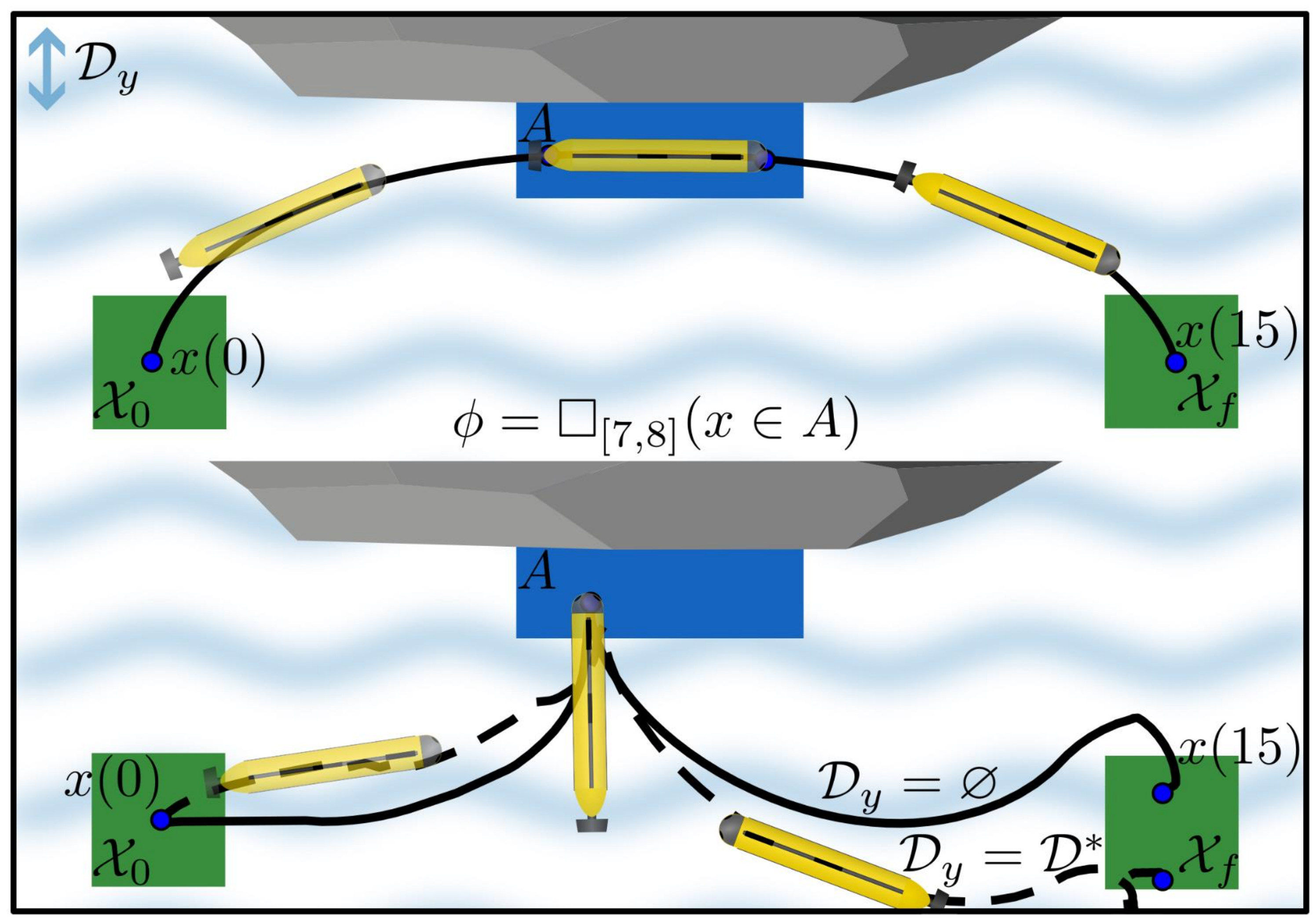


Disturbance Robustness

An underwater robot has some STL mission

$$\dot{x} = f(x, u, d)$$

- What is most time-robust?
- What is most space-robust?
- But what kind of behavior do we want?



$$\delta_\phi = \max_{K,\mathcal{D}} \quad |\mathcal{D}|,$$
$$\text{s.t.} \quad \Phi_f(t; \mathcal{X}_0, K, \mathcal{D}) \models \phi, \quad \forall t \in [t_0, t_f]$$

Maximize the 'size' of the disturbance set while the closed-loop evolution of the system satisfies STL

A joint controller and disturbance optimization: **intractible**!

We take an algorithmic search approach
1. Rewrite the STL spec in disjunctive normal form (this and that) OR (this and that) OR (this and that)
2. Take a subspecification (this and this) and go backwards in time, from operator to operator
   - Check whether the BRS from one predicate intersects the other predicate
   - No? Decrease |D|

But, this is tricky due to temporal flexibitility in Eventually and Until operators!

References

[1]: Verhagen, J., Lindemann, L., & Tumova, J. (2024). Temporally robust multi-agent stl motion planning in continuous time. ACC IEEE
[2]: Verhagen, J., Lindemann, L., & Tumova, J. (2024). Robust STL Control Synthesis under Maximal Disturbance Sets. *CDC IEEE*

WALLENBERG AI, AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM