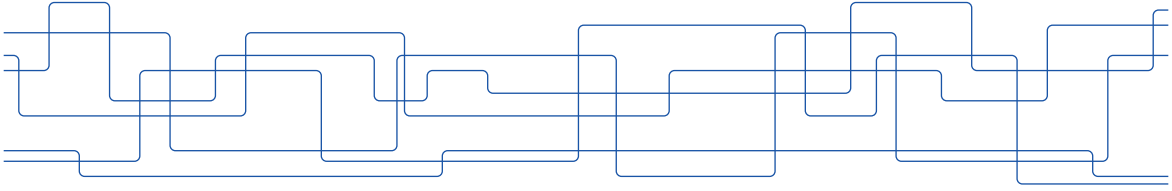
Reinforcement Learning

PhD Course FDD3359 – 2022

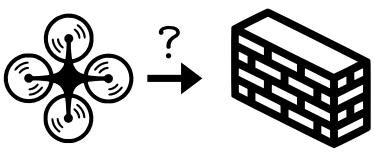
—Safe RL for Control Problems—

Chris Pek



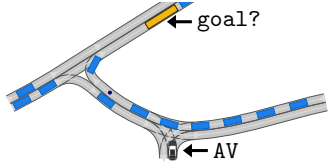
RL and Safe Control

- RL is an important technique to control real robotic systems
- Dynamical systems pose various challenges



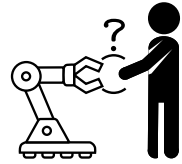
Will the drone hit the wall?

(a)



Can the AV reach the goal?

(b)



Can the human reach the robot?

(c)

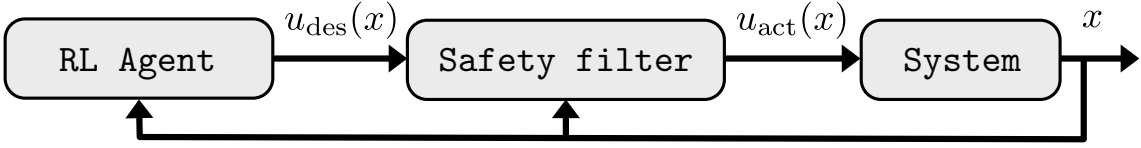
Shielding for continuous domains (state and action space)

1. How can we determine if an action leads to unsafe situations?
2. How can we correct unsafe actions?

Shielding for continuous domains

Shielding for continuous domains (state and action space)

1. How can we determine if an action leads to unsafe situations?
2. How can we correct unsafe actions?



Learning Outcomes of This Session

By the end of this session, you will be able to

1. explain and define safety problems in control;
2. understand the differences between various control techniques;
3. explain basic algorithms used in reachability analysis and control invariance;
4. understand the safety implications of your system;
5. formulate safe control problems for different RL applications;
6. apply shielding to various problems;
7. analyse and evaluate safety problems in your RL applications.

Reachability Analysis

Naive Definition

Set of states that a system can reach over time starting from an initial set of states

- What do we require for reachability analysis?
  - Model of the system, i.e., state space
  - State transition function
  - Representation of (reachable) sets
  - Utility functions and operators

Sets

- Sets are well-defined distinct collections of objects
- Can be described by enumerating all objects, e.g., $\mathcal{S} = \{0, 1, 2, 3, 4\}$
- Set-builder notation can be used to define sets using predicates Φ:

$$S_1 = \{x \,|\, \Phi(x)\} \qquad S_1 = \{x \,|\, \Phi_1(x) \wedge \Phi_2(x)\}$$

- Specifying domains: $S = \{x \in \mathbb{R} \,|\, x \leq 2\}$
- Advanced notations using quantifiers: $S = \{x \in \mathbb{R} \,|\, \exists y \in \mathbb{N} : x = 2y\}$
- Left side may also contain complex notations:

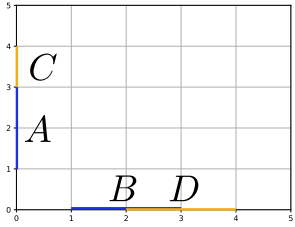
$$S = \{2t + 1 \,|\, t \in \mathbb{Z}\} = \{u \,|\, (u - 1)/2 \in \mathbb{Z}\}$$

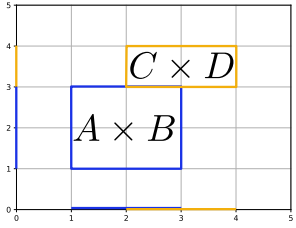
Cartesian Product

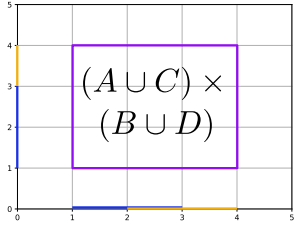
- Cartesian product of two sets:

$$A \times B = \{(a, b) \mid a \in A \wedge b \in B\}$$

- Example: $B = \{x \in \mathbb{R} \mid 1 \leq x \leq 3\}$ $\quad A = \{y \in \mathbb{R} \mid 1 \leq y \leq 3\}$
  $D = \{y \in \mathbb{R} \mid 2 \leq x \leq 4\}$ $\quad C = \{y \in \mathbb{R} \mid 3 \leq y \leq 4\}$







- Important: $(A \times B) \cup (C \times D) \neq (A \cup C) \times (B \cup D)$

Minkowski Sum

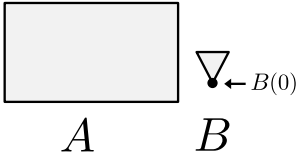
- Minkowsi sum of two sets:

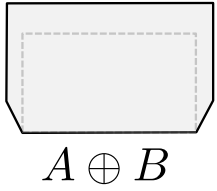
$$A \oplus B = \{a + b \,|\, a \in A \wedge b \in B\}$$

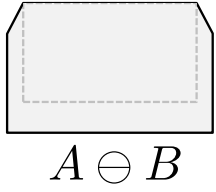
- Minkowski difference of two sets:

$$A \ominus B = (A^{\complement} \oplus -B)^{\complement}$$

- Note: $A \ominus B \neq A \oplus (-B) \Rightarrow$ Morphology vs. motion planning
- Used in motion planning to inflate obstacles







Set Representations – Polytopes

- Set bounded by flat faces; we consider *convex* polytopes
- Possible representations:
  1. VRep: (ordered) list of vertices, i.e.,

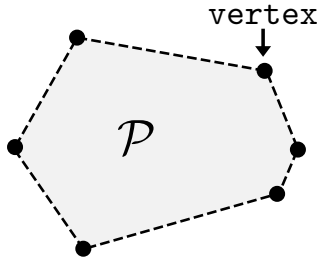
     $$\text{hull}(\mathcal{P}) = (p_0, \ldots, p_n), \forall i \leq n : p_i \in \mathbb{R}^n$$

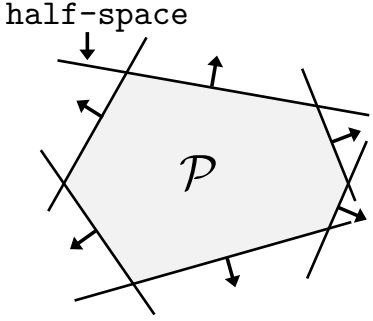
     (set is formed by all convex combinations of vertices)
  2. HRep: intersection of halfspaces $a_0 x_0 + \cdots + a_n x_n \leq b, b \in , \forall i \leq n : a_i \in \mathbb{R}$:

     $$\mathcal{P} = \{x \in \mathbb{R}^n \mid \mathcal{H}x \leq \hat{b}\},$$

     where $\mathcal{H} \in \mathbb{R}^{k \times n}$ and $\hat{b} \in \mathbb{R}^k$ describe k halfspace constraints
- Each representation has pros and cons depending on the use
  - VRep: large number of vertices for complex polytopes
  - HRep: visualization and some operations not straightforward

Set Representations - Polytopes



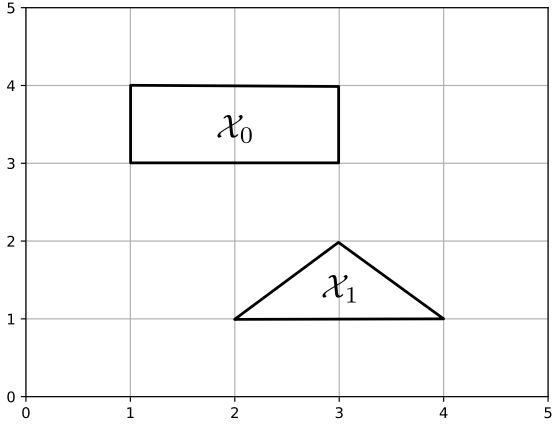


$p =$
$\big((0,0)^T, (1,2)^T, (4,1.4)^T, (4.5,0)^T,$
$(4.2,-0.8)^T, (1.1,-1.8)^T\big)$

$$\mathcal{H} = \begin{pmatrix} -0.89443 & 0.44721 \\ -0.85328 & -0.52145 \\ 0.94174 & 0.33634 \\ 0.19612 & 0.98058 \\ 0.93633 & -0.35112 \\ 0.307 & -0.95171 \end{pmatrix}, \hat{b} = \begin{pmatrix} 0. \\ 0. \\ 4.23784 \\ 2.15728 \\ 4.21348 \\ 2.05078 \end{pmatrix}$$

Your Task: Polytope Creation

- Determine the (minimal) VRep and HRep of the sets $\mathcal{X}_0$ and $\mathcal{X}_1$

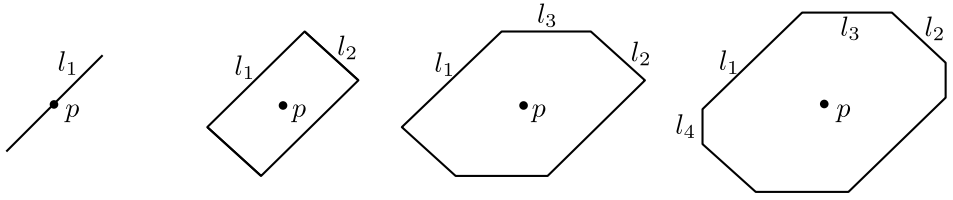


Set Representations – Zonotopes

- Minkowski sum of line segments forms a polytope $\Rightarrow$ Zonotope
- Zonotope with center p is defined using d generators g_i:

$$\mathcal{Z} = \left\{ x \in \mathbb{R}^n \,|\, x = p + \sum_i \lambda_i g_i, \forall i \leq d : g_i \in \mathbb{R}^n \wedge \lambda_i \in [-1, 1] \right\}$$

- Symmetric to center p by construction
- Line segments $l_i = [-1, 1]g_i$



Recap: Dynamical System

- Let $\mathcal{X} \subseteq \mathbb{R}^q$ be the set of feasible states x
- Let $\mathcal{U} \subseteq \mathbb{R}^r$ be the set of feasible inputs u
- The dynamics of the disturbance-free system are described by

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

- Initial state is $x_0 = x(t_0)$
- An input trajectory is denoted as $u\big([t_0, t_h]\big), u(t) \in \mathcal{U}$
- The operator $\chi\big(t, x_0, u([t_0, t_h])\big)$ describes solution of f at time t with respect to trajectory $u([t_0, t_h])$

Recap: Linear Systems

- A system is linear if the superposition principle holds:

$$F(x_1 + x_2) = F(x_1) + F(x_2) \quad \text{and} \quad F(ax) = aF(x)$$

- General state space representation:

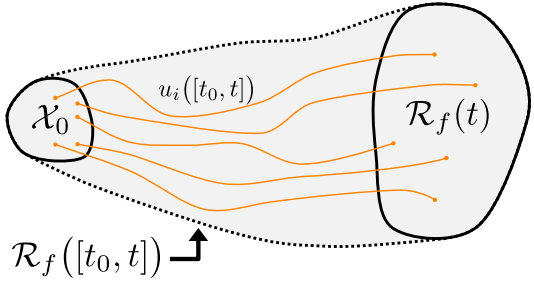
$$\dot{x}(t) = A(t)x(t) + B(t)u(t) \qquad \blacktriangleright \text{ Cont. time}$$
$$x_{k+1} = A_k x_k + B_k u_k \qquad \blacktriangleright \text{ Discr. time}$$

Forward Reachable Sets

- Forward reachable set $\mathcal{R}_f$ is the set of states the system can reach at time t when starting from an initial set of states $\mathcal{X}_0$ and considering all possible inputs $\mathcal{U}$:

$$\mathcal{R}_f(t) = \left\{ \chi\big(t, x_0, u(\cdot)\big) \,\middle|\, x_0 \in \mathcal{X}_0 \wedge \forall \tau \in [t_0, t] : \chi\big(\tau, x_0, u(\cdot)\big) \in \mathcal{X} \wedge u(\tau) \in \mathcal{U} \right\}$$

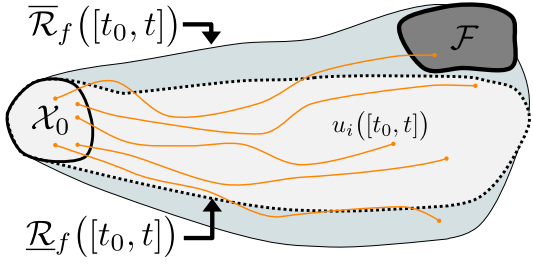
- Reachable set for time interval: $\mathcal{R}_f\big([t_0, t]\big) = \bigcup_{t \in [t_0, t]} \mathcal{R}_f(t)$



Purpose: is the system able to reach certain states?

Over- and Under-approximation of Reachable Sets

- In general, we cannot compute exact forward/backward reachable sets
- Instead, we are interested in (tight) over- and under-approximations
- OA $\overline{\mathcal{R}}_{\square} \supseteq \mathcal{R}_{\square}$ through over-approximative models or computations
- UA $\underline{\mathcal{R}}_{\square} \subseteq \mathcal{R}_{\square}$ through under-approximative models or computations (more difficult)



However, we can still find a collision-free motion

Computing Forward Reachable Sets for Linear Systems

- Different approaches: Hamilton-Jacobi or Propagation-based
- Discretize time horizon into K equal time intervals with step size Δt
- By exploiting superposition, we obtain $\mathcal{R}_f$ as:

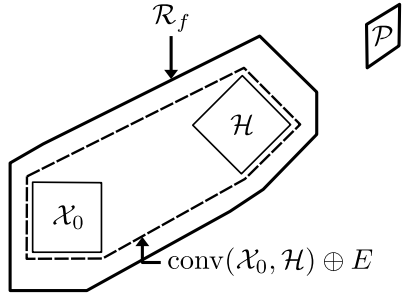
$$\mathcal{R}_f = \mathcal{H} \oplus \mathcal{P},$$

  where $\mathcal{H}$ and $\mathcal{P}$ are the homogeneous and particular solutions, respectively

- Compute $\mathcal{R}_f\big([t_0, t_h]\big) = \bigcup_{k=0}^{K-1} \mathcal{R}_f\big([k\Delta t, (k+1)\Delta t]\big)$
- In general, we cannot obtain exact reachable sets

Computing Forward Reachable Sets for Linear Systems

1. Compute representation of initial set
2. Propagate homogeneous solution $\mathcal{H}$
3. Convex hull
4. Add error term E
5. Compute particular solution $\mathcal{P}$
6. Compute reachable set $\mathcal{R}_f$



Note (see [Althoff, 2010]):

- Use resulting $\mathcal{R}_f$ for next propagation step
- Check if input set contains origin
- Zonotopes are a well-performing set representation for reachability analysis
- Order reduction techniques vs. wrapping effect

Computing Forward Reachable Sets for Linear Systems

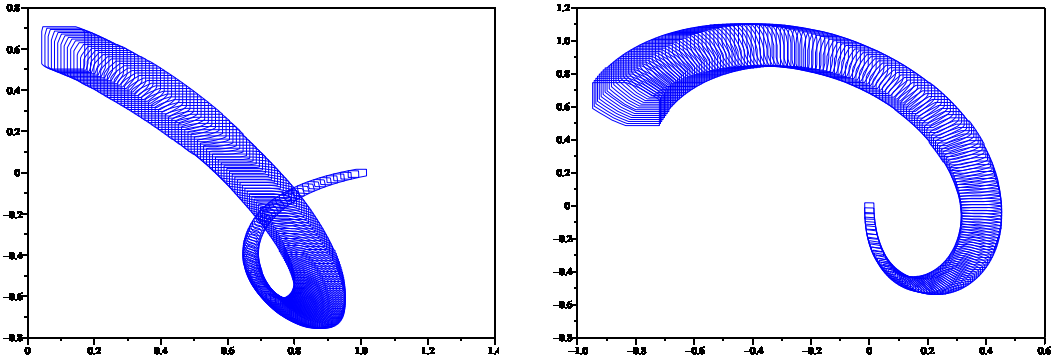
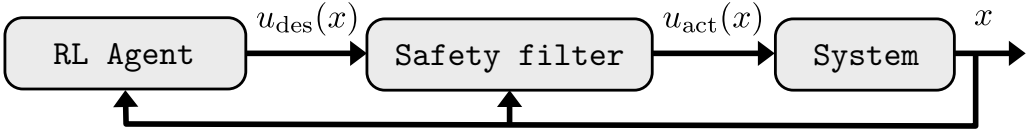


Figure: Projections of reachable set. Reproduced from [Girard, 2005].

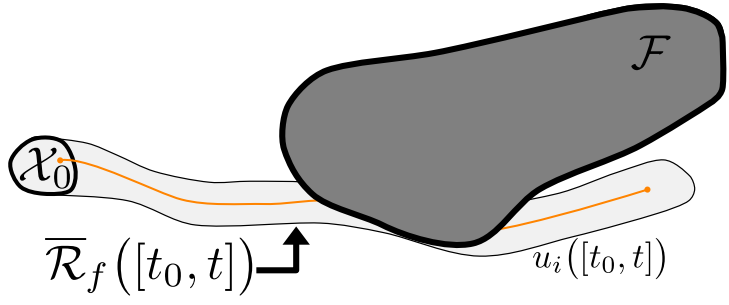
Tools

- **CORA**
  M. Althoff. "An Introduction to CORA 2015". In: Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems. 2015, pp. 120–151
- **JuliaReach**
  S. Bogomolov and et al. "JuliaReach: a toolbox for set-based reachability". In: Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control. ACM. 2019, pp. 39–44
- **Flow***
  X. Chen and et al. "Flow*: An Analyzer for Non-Linear Hybrid Systems". In: Proc. of Computer-Aided Verification. LNCS 8044. Springer, 2013, pp. 258–263
- **SpaceEx**
  G. Frehse and et al. "SpaceEx: Scalable Verification of Hybrid Systems". In: Proc. of the 23rd International Conference on Computer Aided Verification. LNCS 6806. Springer, 2011, pp. 379–395

Going Back to Our Safety Filter



How to Ensure Safe Actions of the RL Agent?



Dynamic obstacles: use reachability to compute forbidden set $\mathcal{F}$ over time

Safe RL for Lane Changes of Autonomous Vehicles

- Continuous state space and discrete action space (lane change left/right, continue)
- Goal: maximize velocity on three lane highway
- Use reachable sets to predict future positions of surrounding vehicles
- Actions safe if agent always in free space $\mathcal{F}^t$

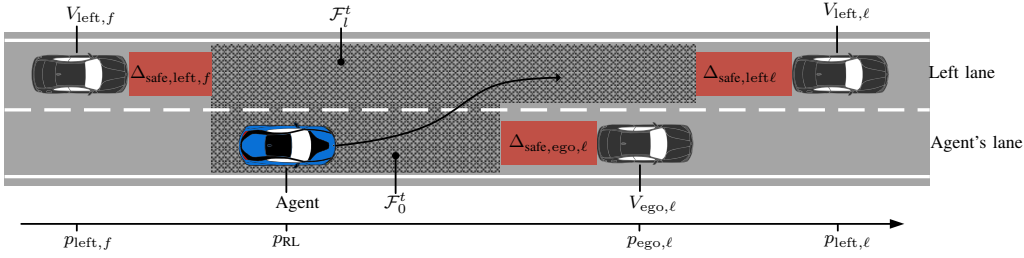
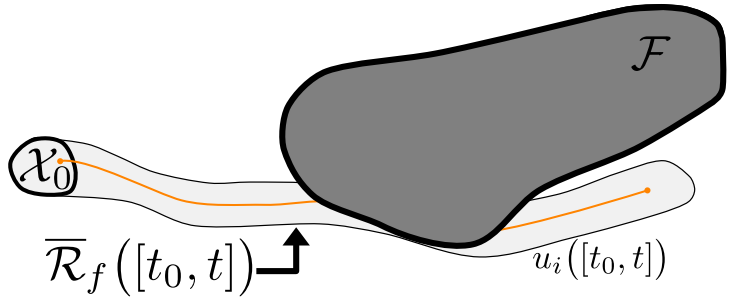


Figure: Reproduced from [Mirchevska et al., 2018].

VIDEO

How to Ensure Safe Actions of the RL Agent?

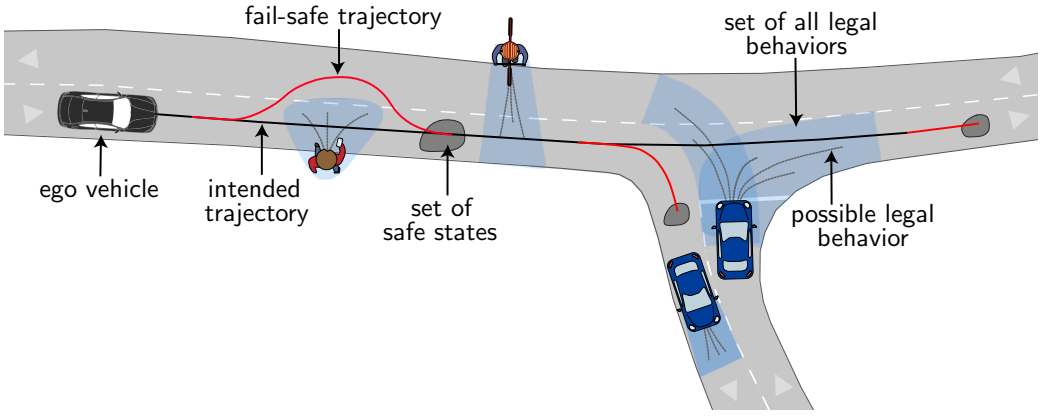


Dynamic obstacles: use reachability to compute forbidden set $\mathcal{F}$ over time

How to synthesize motions that are safe at all times?

Fail-safe Planning/Control

- Fail-safe trajectories are provably safe motions that keep the system safe when the nominal motion becomes (potentially) unsafe [Pek et al., 2020]

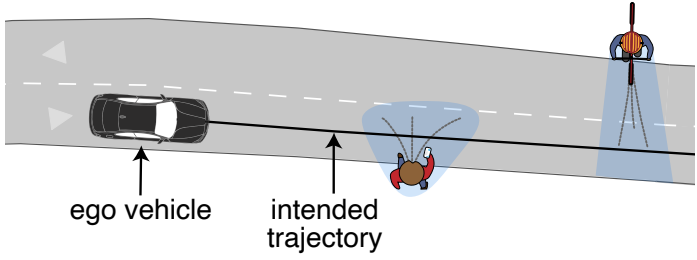


Some Preliminaries

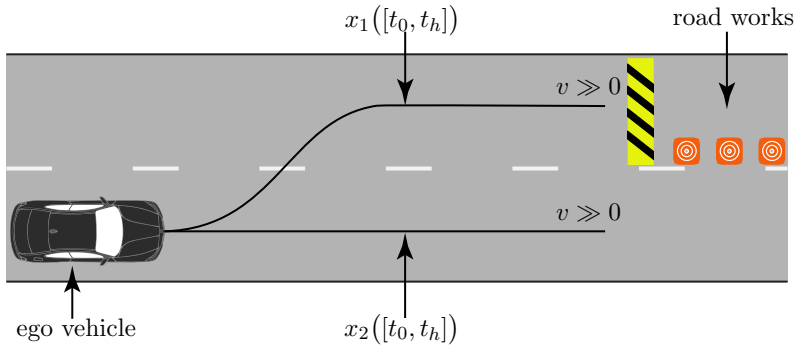
- State space $\mathcal{X} \subseteq \mathbb{R}^q$ and input space $\mathcal{U} \subseteq \mathbb{R}^r$
- Input trajectory $u\big([t_0, t_h]\big), u(t) \in \mathcal{U}$ and state trajectory $x\big([t_0, t_h]\big), x(t) \in \mathcal{X}$
- Workspace / environment $\mathcal{W} \subseteq \mathbb{R}^w$ occupied with obstacles $b \in \mathcal{B}$
- Obstacle footprint (over time) is given by occupancy sets $\mathcal{O}_b(t) \subseteq (\mathcal{W})$
- Operator $\mathrm{occ} : \mathcal{X} \to \mathrm{Pow}(\mathcal{W})$ maps states to occupancy set
- Set of collision-free states $\mathcal{X}_{\mathrm{cf}}(t) := \{x \in \mathcal{X} \,|\, \forall b \in \mathcal{B} : \mathrm{occ}(x) \cap \mathcal{O}_b(t)\}$

Some Preliminaries

- State space $\mathcal{X} \subseteq \mathbb{R}^q$ and input space $\mathcal{U} \subseteq \mathbb{R}^r$
- Input trajectory $u\big([t_0, t_h]\big), u(t) \in \mathcal{U}$ and state trajectory $x\big([t_0, t_h]\big), x(t) \in \mathcal{X}$
- Workspace / environment $\mathcal{W} \subseteq \mathbb{R}^w$ occupied with obstacles $b \in \mathcal{B}$
- Obstacle footprint (over time) is given by occupancy sets $\mathcal{O}_b(t) \subseteq (\mathcal{W})$
- Operator $\mathrm{occ} : \mathcal{X} \to \mathrm{Pow}(\mathcal{W})$ maps states to occupancy set
- Set of collision-free states $\mathcal{X}_{\mathrm{cf}}(t) := \{x \in \mathcal{X} \,|\, \forall b \in \mathcal{B} : \mathrm{occ}(x) \cap \mathcal{O}_b(t)\}$



Persistent Feasibility



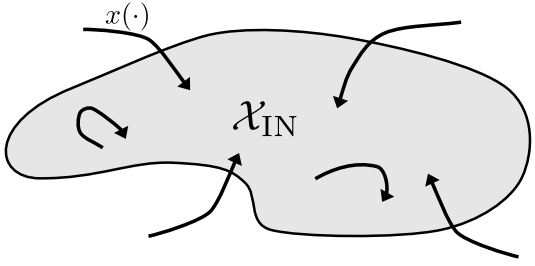
- Both trajectories $x_1(\cdot)$ and $x_2(\cdot)$ are collision-free trajectories!
- However, only trajectory $x_2(\cdot)$ remains collision-free when performing replanning

What are Invariant Sets?

- Set invariance is an important concept in control (related to stability)
- Let us consider an arbitrary autonomous system $\dot{x} = f(x)$ with $x(0) = x_0$
- A set $\mathcal{X}_{\text{IN}} \subseteq \mathcal{X}$ is said to be an invariant (w.r.t. f) if:

$$x(t_0) \in \mathcal{X}_{\text{IN}} \Rightarrow \forall \tau \geq t_0 : x(\tau) \in \mathcal{X}_{\text{IN}}$$

- If trajectory enters $\mathcal{X}_{\text{IN}}$, it will remain in $\mathcal{X}_{\text{IN}}$ indefinitely



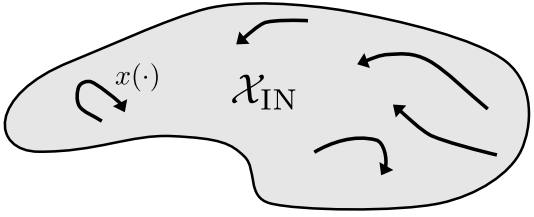
Importance of Invariant Sets for Infinite Horizon Planning

- Let us consider a more general system $\dot{x} = f(x, u)$
- We want to ensure that x remains in a certain set $\mathcal{X}_\mathrm{d}$
- Control invariant set $\mathcal{X}_\mathrm{IN} \subseteq \mathcal{X}_\mathrm{d}$:

$$\mathcal{X}_\mathrm{IN} := \{x(t_0) \in \mathcal{X}_\mathrm{d} \,|\, \exists \Phi : \forall \tau \geq t_0 : \chi(\tau, x, \Phi(x)) \in \mathcal{X}_\mathrm{d}\},$$

  where Φ is a control law.

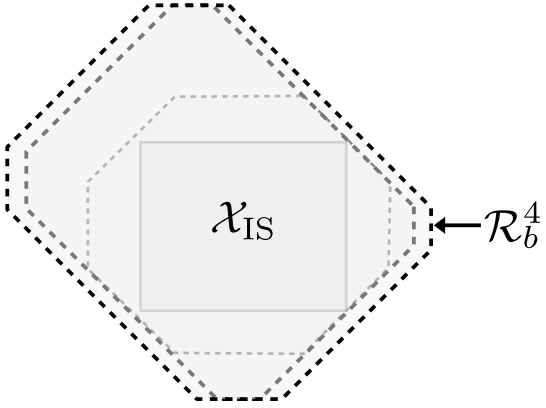
- If system is in $\mathcal{X}_\mathrm{IN}$, we can find a trajectory to remain in $\mathcal{X}_\mathrm{IN}$ indefinitely



Computation of Invariant Sets

Without loss of generality, we consider the system $x_{k+1} = F(x_k, u_k)$

- General observations:
  1. if $x_k \in \mathcal{X}_{\mathrm{IS}} \Rightarrow F(x_k, u_k) \in \mathcal{X}_{\mathrm{IS}}$
  2. what about $\mathcal{X}' = \{x_k \in \mathcal{X}_{\mathrm{d}} | F(x_k, u_k) \in \mathcal{X}_{\mathrm{IS}}\}$?
  3. $\mathcal{X}' \supseteq \mathcal{X}_{\mathrm{IS}}$
  4. $\mathcal{X}' = \mathcal{R}_b^1$ with $(\cap \mathcal{X}_{\mathrm{d}}, \mathcal{X}_f = \mathcal{X}_{\mathrm{IS}})$
  5. Compute $\mathcal{R}_b^{i+1}$ with $\mathcal{X}_f = \mathcal{R}_b^i$
  6. $\mathcal{R}_b^\infty = \mathcal{R}_b^{i+1} = \mathcal{R}_b^i$
  7. $\mathcal{X}'_{\mathrm{IS}} = \mathcal{R}_b^\infty$

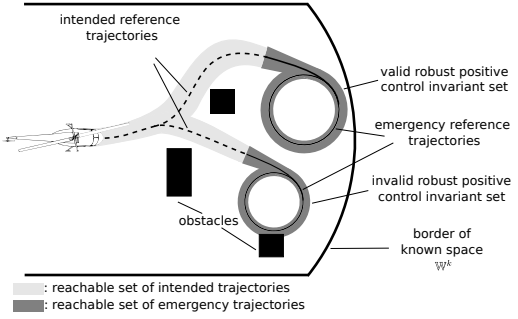


$\Rightarrow$ requires us to find a suitable terminal set to start with

Example – Motion Planning for a Helicopter

Experiment from [Althoff et al., 2015]

- Use emergency maneuvers (Loiter circles) to compute invariant set
- Consider disturbances



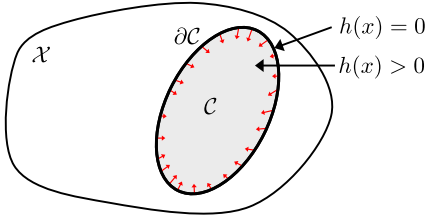
Barrier Certificates for Ensuring Safety

- Very similar to invariant sets
- Originates from the field of optimization theory
- Idea: barrier $B(x) = \infty, x \to \partial C$ cannot be crossed by system trajectories
- Lets consider a safe set
  $\mathcal{C} := \{x \,|\, h(x) \geq 0\}, h : \mathbb{R}^n \to \mathbb{R}$
- Nagumo's theorem [Nagumo, 1942]:

  $\mathcal{C}$ is invariant $\Leftrightarrow \dot{h}(x) \geq 0, \forall x \in \partial\mathcal{C}$

- Controlled system:

  $\mathcal{C}$ is invariant $\Leftrightarrow \exists u : \dot{h}(x, u) \geq 0, \forall x \in \partial\mathcal{C}$



Barrier Certificates for Ensuring Safety

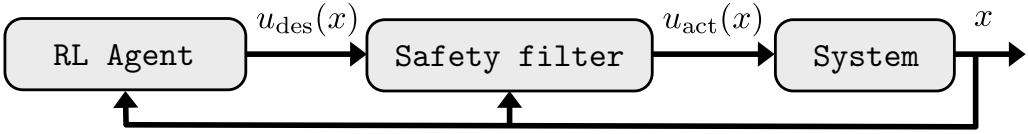
- We are interested in defining the barrier property over the whole set
- Naive idea: $\dot{h}(x) \geq 0, \forall x \in \partial\mathcal{C} \Rightarrow \dot{h}(x) \geq 0, \forall x \in \mathcal{C}$
- Less conservative idea using scaling:

$$\dot{h}(x, u) \geq -\gamma(h(x)), \forall x \in \mathcal{C},$$

  where γ is a class $\mathcal{K}$ function with $\gamma(0) = 0$
- Sufficient condition to show safety
- Can be formulated as a quadratic program (QP):

$$\begin{aligned} u^\star = &\arg\min_{u \in \mathcal{U}} ||u - u_{\text{des}}(x)||^2 \\ &\text{s.t. } \dot{h}(x, u) \geq -\gamma(h(x)) \end{aligned}$$

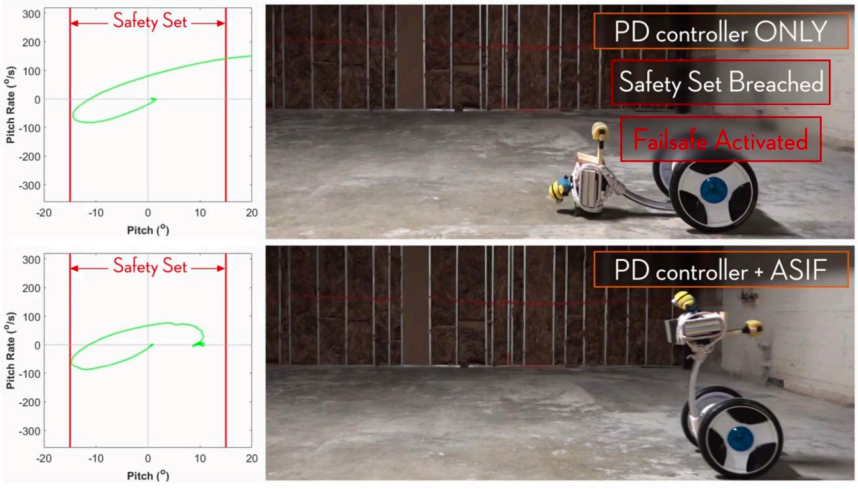
Notes on Barrier Certificates



- Usually, point-wise optimal
- Closed form controllers exist
- Control Lyapunov function is special case of CBF
- Artifical potential fields are special case of CBF
- Theory can be extended to discrete systems as well
- More information in [Ames et al., 2019]

Example – Control of a Segway

- Idea: safety is to keep segway upright [Gurriet et al., 2018]



Example – Control of a Segway

VIDEO

Gaussian Processes

- GPs represent distributions over functions
- Any state x is assigned a random variable $r(x)$
- Distribution is modeled as prior using dataset X:

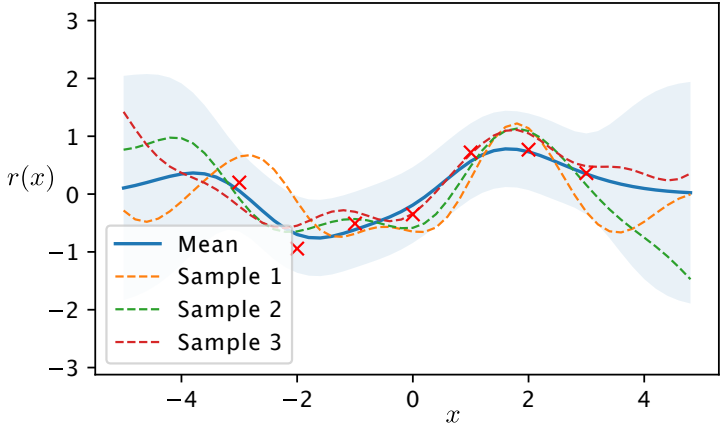
$$
\begin{aligned}
p(\boldsymbol{r}|X) &= \mathcal{N}(\boldsymbol{r}|\boldsymbol{\mu}, K), \text{where} \\
\boldsymbol{r} &= (r(x_1), \ldots, r(x_n)) && \blacktriangleright \text{ random vars.} \\
\boldsymbol{\mu} &= (\mu_1, \ldots, \mu_n) && \blacktriangleright \text{ mean} \\
K_{i,j} &= \kappa(x_i, x_j) && \blacktriangleright \text{ kernels}
\end{aligned}
$$

- Example kernel: $\kappa(x_i, x_j) = \sigma_r^2 \exp\left(-\frac{1}{2\ell^2}(x_i - x_j)^T(x_i - x_j)\right)$
- Given dataset of samples, we can compute the posterior:

$$p(\boldsymbol{r}_\star|X_\star, X, \boldsymbol{r}) = \mathcal{N}(\boldsymbol{r}_\star|\boldsymbol{\mu}_\star, \Sigma_\star)$$

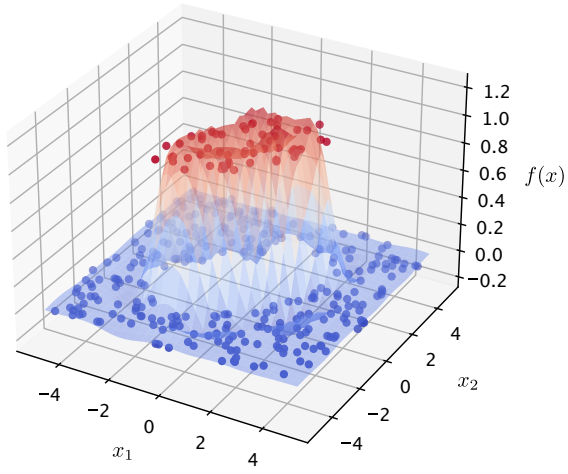
Gaussian Processes

- Posterior given noisy data samples X



GPs for Safe Set Approximation

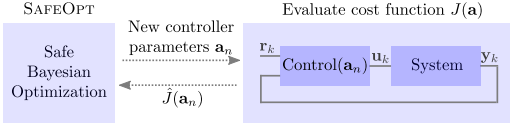
- Use GPs to approximate safe set
- Input: set of samples x_i and value function $g(x_i)$
- Optimize safe set approximation
- Get estimates for unseen parts of the state space
- Only a probabilistic estimate
- Can be applied in online learning

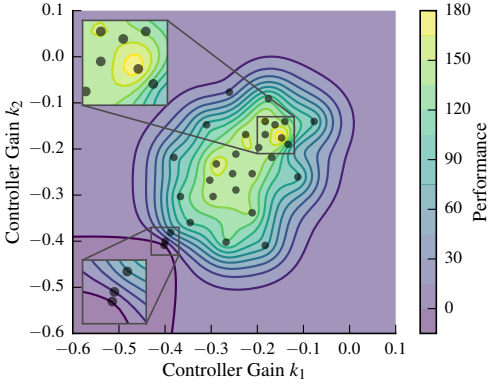


- More infos about GPs: [Reece and Roberts, 2010]

Examples of GPs for Safety

Experiment from [Berkenkamp et al., 2016]





VIDEO

Conclusions

- Shields for continuous domains
- Reachability analysis can be used to detect future unsafe situations
- Persistent feasibility important to keep system safe
- Invariant sets allow system to remain safe
- Restrict actions so that system remains in safe set
- Control barrier functions and Gaussian processes to compute safe sets
- Synthesize of least restrictive shields, model-free vs. model-based world, uncertainty and partial-observability

References I


Althoff, D., Althoff, M., and Scherer, S. (2015).
Online safety verification of trajectories for unmanned flight with offline computed robust invariant sets.
In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 3470–3477.


Althoff, M. (2010).
Reachability analysis and its application to the safety assessment of autonomous cars.
PhD thesis, Technische Universität München.


Ames, A. D., Coogan, S., Egerstedt, M., Notomista, G., Sreenath, K., and Tabuada, P. (2019).
Control barrier functions: Theory and applications.
In *Proc. of the IEEE European Control Conference*, pages 3420–3431.


Berkenkamp, F., Schoellig, A. P., and Krause, A. (2016).
Safe controller optimization for quadrotors with gaussian processes.
In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 491–496.

References II

Girard, A. (2005).
Reachability of uncertain linear systems using zonotopes.
In *Int. Workshop on Hybrid Systems: Computation and Control*, pages 291–305. Springer.

Gurriet, T., Singletary, A., Reher, J., Ciarletta, L., Feron, E., and Ames, A. (2018).
Towards a framework for realizable safety critical control through active set invariance.
In *Proc. of the ACM/IEEE Int. Conf. on Cyber-Physical Systems*, pages 98–106.

Mirchevska, B., Pek, C., Werling, M., Althoff, M., and Boedecker, J. (2018).
High-level decision making for safe and reasonable autonomous lane changing using reinforcement learning.
In *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*, pages 2156–2162.

Nagumo, M. (1942).
Über die lage der integralkurven gewöhnlicher differentialgleichungen.
Proc. of the Physico-Mathematical Society of Japan. 3rd Series, 24:551–559.

References III


Pek, C., Manzinger, S., Koschi, M., and Althoff, M. (2020).
Using online verification to prevent autonomous vehicles from causing accidents.
Nature Machine Intelligence, 2(9):518–528.


Reece, S. and Roberts, S. (2010).
An introduction to gaussian processes for the kalman filter expert.
In *Proc. of the IEEE Int. Conf. on Information Fusion*, pages 1–9.