Learning-Based Predictive Control

Chapter 8
Invariance-Based Safety Filters

Prof. Melanie Zeilinger

ETH Zurich

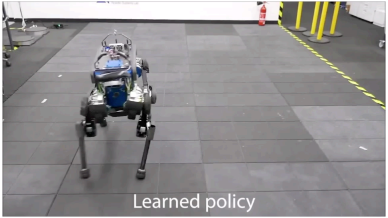
2023

Outline

1. Motivation & Examples
2. Invariance-based Safety Filter
3. Semidefinite Programming Approach for Linear Systems
4. Extension to Uncertain Systems (robust case)

Other Successful Control Policies (often) Without Safety Properties

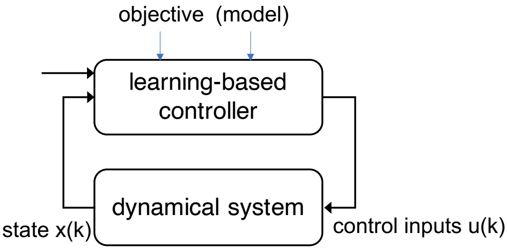
- Human control
- Reinforcement learning

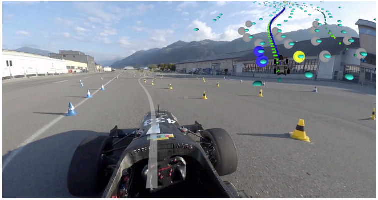


[*Robotic Systems Lab, ETH Zurich*]
`https://www.youtube.com/watch?v=aTDkYFZFWug&feature=emb_logo`

[*UZH Robotics and Perception Group*]
`https://www.youtube.com/watch?v=2N_wKXQ6MXA`

Challenge of Learning-based Control: Safety





Dilemma:

Need to act to collect data & explore

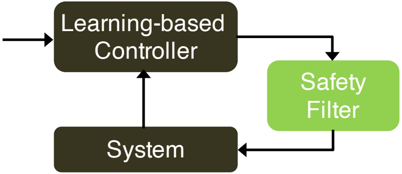
But: Every action can be safety-critical

Goal:

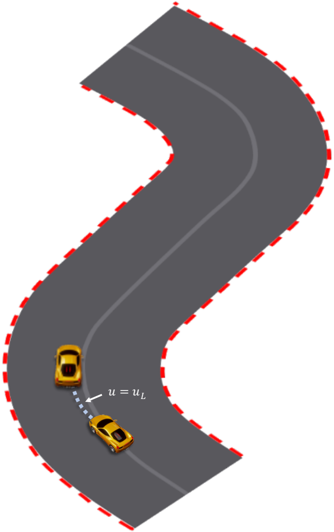
Augment any learning-based controller with safety guarantees.

Safety Filter - Main Idea

Add safety verification module to any controller

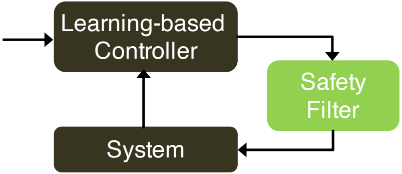


Goal: Verify control input and modify if required to ensure safety at all future times

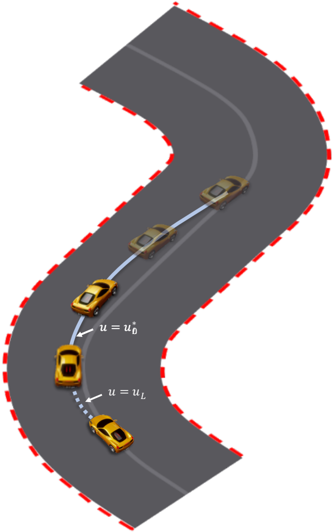


Safety Filter - Main Idea

Add safety verification module to any controller

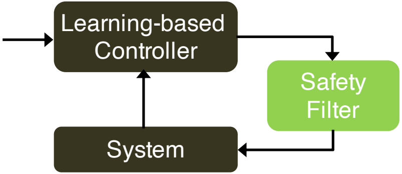


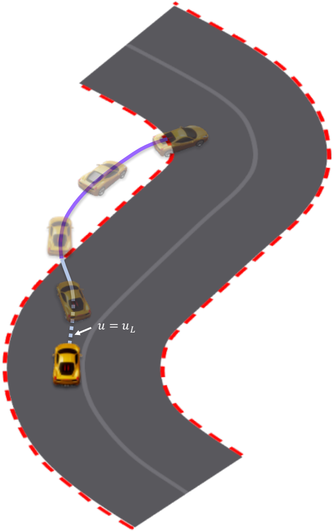
Goal: Verify control input and modify if required to ensure safety at all future times



Safety Filter - Main Idea

Add safety verification module to any controller

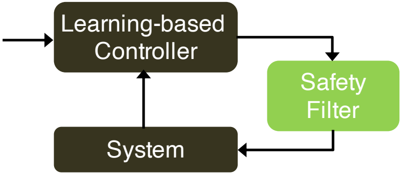




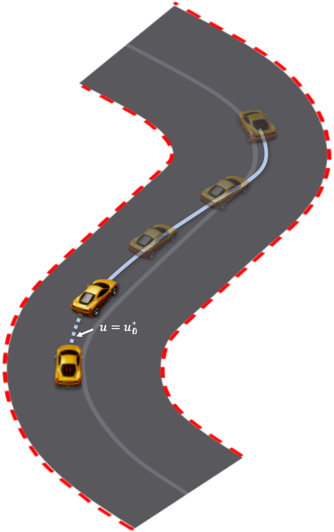
Goal: Verify control input and modify if required to ensure safety at all future times

Safety Filter - Main Idea

Add safety verification module to any controller



Goal: Verify control input and modify if required to ensure safety at all future times



Problem Definition

Consider controlled system $x(k+1) = f(x(k), u(k))$ under control policy $u(k) = \pi(x(k))$

$$x(k+1) = f(x(k), \pi(x(k)))$$

$\rightarrow$ Given state $x(k)$ (forward) trajectory is fully determined
(uncertainties will be considered later)

Example (Point mass):
To determine trajectory both position $p(k)$ and velocity $v(k)$ need to be known
$\rightarrow x(k) = [p(x),\ v(k)]^\top$

Safety as Constraint Satisfaction

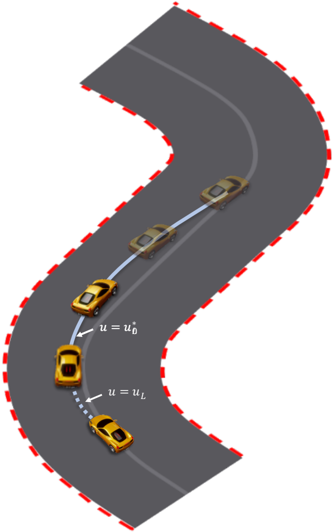
Safety constraints: $x \in \mathcal{X} \subset \mathbb{R}^n$

(or $\Pr(x \in \mathcal{X}) \geq p$)

Example (Point mass):
$p(k) \leq 0$ may represent crash with a wall
$\rightarrow \mathcal{X} := \{x(k) \,|\, [1,\, 0]x(k) \geq 0\}$

Note: Physical systems have actuator limitations

$\rightarrow$ Input constraints: $u \in \mathcal{U}$

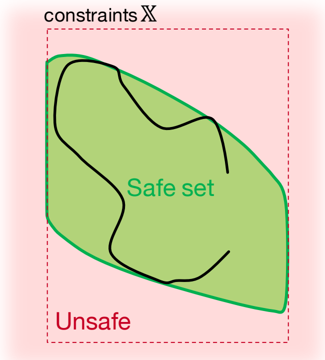


Safe Set

Set $\mathcal{S}$ is a **safe set** for a dynamical system with respect to constraints $\mathcal{X}$ if for all $x(0) \in \mathcal{S}$ we have $x(k) \in \mathcal{X}\, \forall k \geq 0$

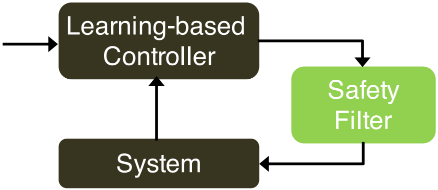
Note: $\mathcal{S} \neq \mathcal{X}$ and finding (the maximum) $\mathcal{S}$ can be very involved, even for autonomous systems

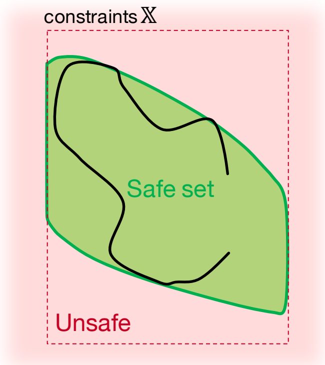
Example (Point mass):
Any position $p(k)$ may be safe or unsafe, depending on velocity $v(k)$



Application to Safety Filters

Consider now the controlled system $x(k+1) = f(x(k), u(k))$





Proposed (learning) input u_L which an (RL) controller intends to apply

Goal: Design safety filter $\pi_f(u_L, x) \in \mathcal{U}$ to

- Maximize safe set $\mathcal{S}$ for system $f(x, \pi_f(u_L, x))$
- Interfere as 'as little as possible': $\pi_f(u_L, x) \approx u_L$

Example: Quadrotor Learning to Fly Up and Down

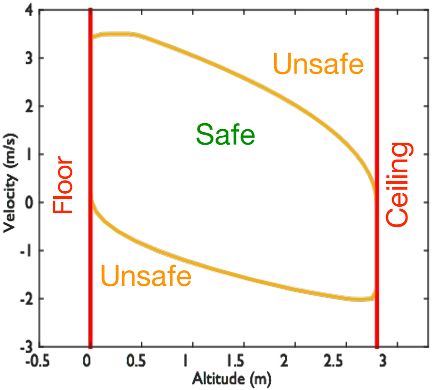
Model:

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

states: position velocity

control: thrust

model uncertainty



Example: Quadrotor Learning to Fly Up and Down - Video

Learning Objectives: Invariance-based Safety Filter

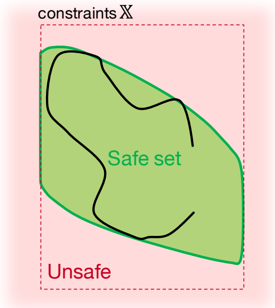
- Understand the general concept & theory of an invariance based safety filter (nominal case)
- Derive an invariance-based safety filter for linear systems based on semidefinite programming
- Extend concept to account for model uncertainty
- Understand limitations and benefits

Outline

1. Motivation & Examples
2. Invariance-based Safety Filter
3. Semidefinite Programming Approach for Linear Systems
4. Extension to Uncertain Systems (robust case)

Safe Set Based on Control Invariance

Consider system $x(k+1) = f(x(k), u(k))$ with constraints $x \in \mathcal{X}, u \in \mathcal{U}$.



Maximum control invariant set provides largest safe set.

Control Invariant Set

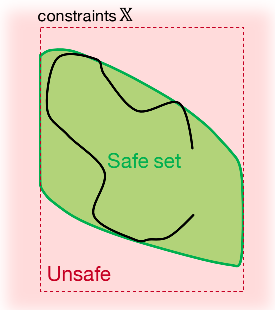
A set $\mathcal{C} \subseteq \mathcal{X}$ is said to be a control invariant set if

$$x(k) \in \mathcal{C} \quad \Rightarrow \quad \exists u(k) \in \mathcal{U} \text{ such that } f(x(k), u(k)) \in \mathcal{C} \quad \text{for all } k \in \mathbb{N}^+$$

Defines the states for which there exists a **controller** that will satisfy constraints for **all time**.

Safe Set Based on Control Invariance

Consider system $x(k+1) = f(x(k), u(k))$ with constraints $x \in \mathcal{X}, u \in \mathcal{U}$.



Maximum control invariant set provides largest safe set.

→ But: Difficult to compute

Control Invariant Set

A set $\mathcal{C} \subseteq \mathcal{X}$ is said to be a control invariant set if

$$x(k) \in \mathcal{C} \quad \Rightarrow \quad \exists u(k) \in \mathcal{U} \text{ such that } f(x(k), u(k)) \in \mathcal{C} \quad \text{for all } k \in \mathbb{N}^+$$

Defines the states for which there exists a **controller** that will satisfy constraints for **all time**.

Safe Set Based on Invariance

Positive Invariant set

A set $\mathcal{O}$ is said to be a positive invariant set for system $x(k+1) = f(x(k), \pi(x(k)))$ if

$$x(k) \in \mathcal{O} \;\; \Rightarrow \;\; x(k+1) \in \mathcal{O} \;\; , \;\; \forall k \in \{0, 1, \dots\}$$

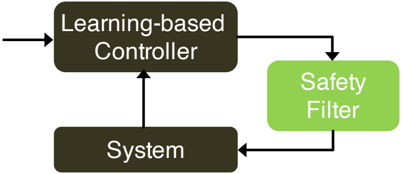
If the invariant set is within the constraints, it provides a set of initial states from which the trajectory will never violate the system constraints $\rightarrow$ $\mathcal{O} \subseteq \mathcal{X}$ is **safe set**

Design **safety controller** $\pi_s(x)$ with large corresponding invariant set $\mathcal{O} \subseteq \mathcal{X}$ for controlled system $f(x, \pi_s(x))$, while ensuring $\pi_s(x) \in \mathcal{U}$ for all $x \in \mathcal{O}$

$$\textbf{Safety filter: } \pi_f(u_L, x) = \begin{cases} u_L & \text{if } f(x, u_L) \in \mathcal{O} \text{ and } u_L \in \mathcal{U} \\ \pi_s(x) & \text{otherwise (maintains invariance property of } \pi_s) \end{cases}$$

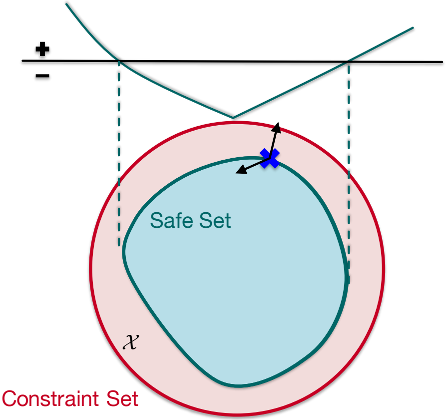
Level Set Based Safety Filter I

Add safety verification module to any controller



Formulate safe set as level set of safety value function:

$$\mathcal{S} = \{x \in \mathbb{R}^n \mid V(x) \leq 1\}$$



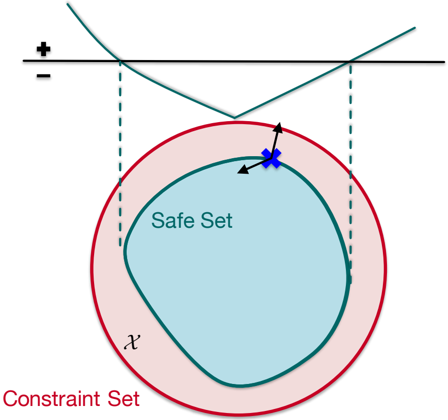
Invariant Sets from Lyapunov Function Level Sets

Invariant set from Lyapunov function

Let V be such that $V(f(x)) - V(x) \leq 0$, then

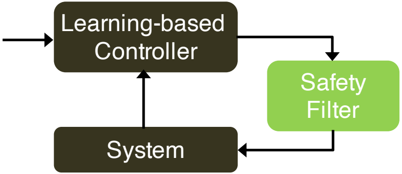
$$Y := \{x \mid V(x) \leq \alpha\}$$

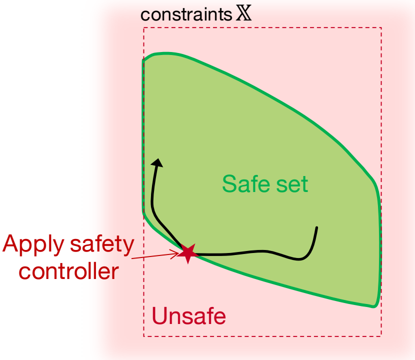
(known as level set of function V) is an invariant set for all $\alpha \geq 0$.



Level Set Based Safety Filter I

Add safety verification module to any controller





Formulate safe set as level set of safety value function:

$$\mathcal{S} = \{x \in \mathbb{R}^n \mid V(x) \leq 1\}$$

$$\pi_f(u_L, x) = \begin{cases} \text{learning input } u_L(x) & \text{if } V(f(x, u_L)) \leq 1, u_L(x) \in \mathcal{U} \\ \text{safe input } \pi_s(x) & \text{otherwise} \end{cases}$$

Outline

1. Motivation & Examples
2. Invariance-based Safety Filter
3. Semidefinite Programming Approach for Linear Systems
4. Extension to Uncertain Systems (robust case)

Safe Set Based on Invariance

Consider the system model $x(k+1) = f(x(k), u(k))$ with constraints $x \in \mathcal{X}$, $u \in \mathcal{U}$.

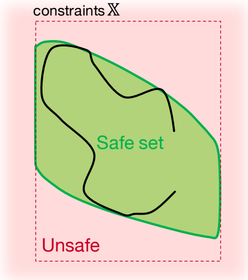
Approach: Select control law $u = \pi_s(x)$, define safe set as invariant set for controlled system

$$x(k+1) = f(x(k), \pi_s(x(k))).$$

Here: Define safe set as level set of safety value function $\mathcal{S} = \{x \in \mathbb{R}^n \mid V(x) \leq 1\}$.

Goal: We want to maximize 'size' of $\mathcal{S}$,
subject to the conditions

1. Invariance: $V(x(k)) \leq 1 \Rightarrow V(x(k+1)) \leq 1$
2. Constraint satisfaction: $\mathcal{S} \subseteq \mathcal{X}$
3. Feasibility of safety controller: $\pi_s(x) \in \mathcal{U}\ \forall x \in \mathcal{S}$



Safe Set Based on Invariance for Linear System

Consider the system model $x(k+1) = Ax(k) + Bu(k)$
with polytopic constraints $x \in \mathcal{X} := \{x \,|\, H_x x \leq b_x\}$, $u \in \mathcal{U} := \{u \,|\, H_u u \leq b_u\}$

Approach: Select control law $\pi_s(x) = Kx$, define safe set as invariant set for controlled system

$$x(k+1) = Ax(k) + BKx(k) = (A + BK)x(k).$$

Here: Define safe set as level set of safety value function $\mathcal{S} = \{x \in \mathbb{R}^n \mid x^T P x \leq 1\}$.

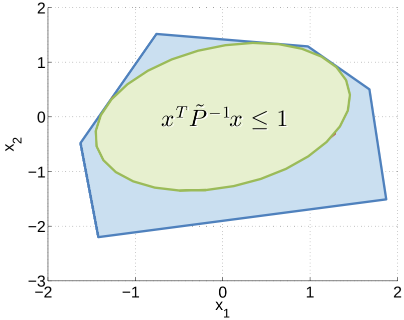
Goal: We want to maximize 'size' of $\mathcal{S}$,
subject to the invariance-based safe set conditions:

1. Invariance: $x^\mathsf{T}(A+BK)^\mathsf{T} P (A+BK) x \leq x^\mathsf{T} P x$

   Note: Only a sufficient condition for invariance

2. Constraint satisfaction: $\mathcal{S} \subseteq \mathcal{X}$

3. Feasibility of safety controller: $\pi_s(x) = Kx \in \mathcal{U}\ \forall x \in \mathcal{S}$



Computation via Semi-definite Programming: Objective

Semi-definite programs (SDPs) are convex optimization problems encoding positive definiteness conditions on matrix-valued decision variables
$\rightarrow$ allows to express design problem as a constrained convex optimization problem

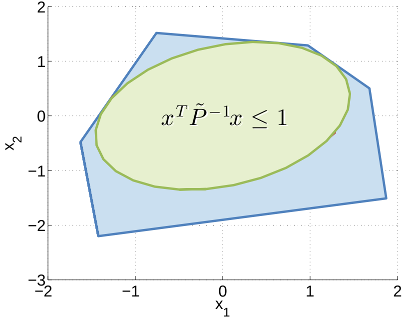
$$\mathcal{S} = \{x \in \mathbb{R}^n \mid x^T P x \leq 1\}$$

Objective: maximize 'size'

- Volume of ellipse $\propto \det P^{-\frac{1}{2}}$
- Maximizes volume by minimizing

$$-\log\det P^{-1} = -2\log\det P^{-\frac{1}{2}}$$

which is convex in $P^{-1} > 0$
(matrix-valued decision variable)



Semi-definite Programming Approach: 1. Invariance

$$
\begin{aligned}
& x^\mathsf{T}(A+BK)^\mathsf{T}P(A+BK)x - x^\mathsf{T}Px \le 0 \\
\Leftrightarrow\quad & x^\mathsf{T}Px - x^\mathsf{T}(A+BK)^\mathsf{T}P(A+BK)x \ge 0 \\
\Leftrightarrow\quad & P - (A+BK)^\mathsf{T}P(A+BK) \ge 0 \\
\Leftrightarrow\quad & P^{-1} - P^{-1}(A+BK)^\mathsf{T}P(A+BK)P^{-1} \ge 0 \\
\Leftrightarrow\quad & \begin{bmatrix} P^{-1} & P^{-1}(A+BK)^\mathsf{T} \\ (A+BK)P^{-1} & P^{-1} \end{bmatrix} \ge 0
\end{aligned}
$$

(pre-post multiply by P^{-1}, apply Schur complement)

which is a linear matrix inequality (LMI) in shape matrix P^{-1} which can be optimized in an SDP

Problem: We would like to additionally optimize over controller gain K

Solution: Introduce additional optimization variable $Y = KP^{-1}$
(since $P^{-1} > 0$ one can always reconstruct $K = YP$)

Semi-definite Programming Approach: 2. State Constraints

Consider each half space j in $\mathcal{X} = \{x \mid H_x x \leq b_x\}$ individually

$$\mathcal{S} \subseteq \mathcal{X} \Leftrightarrow \forall j \ \left(\max_x [H_x]_j x \text{ s.t. } x^\top P x \leq 1 \right) \leq [b_x]_j$$

where the solution to the inner optimization problem is analytically available

$$\max_x [H_x]_j x \text{ s.t. } x^\top P x \leq 1 = \sqrt{[H_x]_j P^{-1} [H_x]_j^\top}$$

$\rightarrow$ we can express the constraint as an LMI in P^{-1}:

$$\mathcal{S} \subseteq \mathcal{X} \Leftrightarrow [H_x]_j P^{-1} [H_x]_j^\top \leq [b_x]_j^2 \ \forall j$$

Semi-definite Programming Approach: 3. Input Constraints

Following the same procedure for $\mathcal{U} = \{u \,|\, H_u u \leq b_u\}$ we have

$$
\begin{aligned}
K\mathcal{S} \subseteq \mathcal{U} \Leftrightarrow & [H_u]_j K P^{-1} K [H_x]_j^\mathsf{T} \leq [b_u]_j^2 \ \forall j \\
\Leftrightarrow & [H_u]_j K P^{-1} P P^{-1} K^\mathsf{T} [H_u]_j^\mathsf{T} \leq [b_u]_j^2 \ \forall j \\
\Leftrightarrow & [b_u]_j^2 - [H_u]_j Y P Y^\mathsf{T} [H_u]_j^\mathsf{T} \geq 0 \ \forall j \\
\Leftrightarrow & \begin{bmatrix} [b_u]_j^2 & [H_u]_j Y \\ Y^\mathsf{T} [H_u]_j^\mathsf{T} & P^{-1} \end{bmatrix} \geq 0 \ \forall j
\end{aligned}
$$

which is again an LMI in Y and P^{-1}

Using dedicated SDP solvers, these problems can be efficiently and reliably solved. Additionally, there exist powerful interfaces such as Yalmip[1] and CVXPY[2] facilitating implementation.

[1] yalmip.github.io
[2] www.cvxpy.org

Outline

1. Motivation & Examples

2. Invariance-based Safety Filter

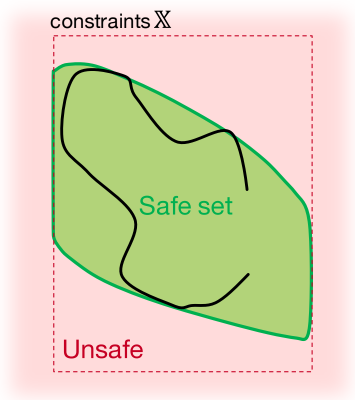
3. Semidefinite Programming Approach for Linear Systems

4. Extension to Uncertain Systems (robust case)

Safe Set Based on Robust (Control) Invariance

Consider system $x(k+1) = f(x(k), u(k), w(k))$ with bounded noise $w \in \mathcal{W}$ and constraints $x \in \mathcal{X}$, $u \in \mathcal{U}$.

$\rightarrow$ The discussed safety filter concept can be extended to this problem class using robust (control) invariant sets.

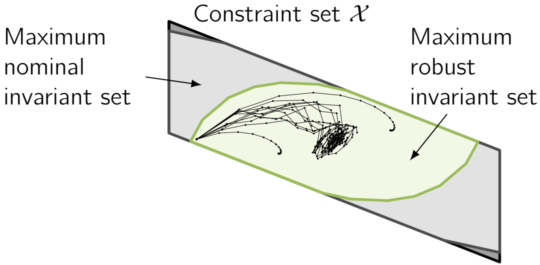


Recall: Robust Invariance

Robust Positive Invariant set

A set $\mathcal{O}^{\mathcal{W}}$ is said to be a robust positive invariant set for the autonomous system $x(k+1) = f(x(k), w(k))$ if

$$x \in \mathcal{O}^{\mathcal{W}} \quad \Rightarrow \quad f(x,w) \in \mathcal{O}^{\mathcal{W}} \quad , \quad \text{for all } w \in \mathcal{W}$$



Recall: Robust Invariance

Robust Positive Invariant set

A set $\mathcal{O}^{\mathcal{W}}$ is said to be a robust positive invariant set for the autonomous system $x(k+1) = f(x(k), w(k))$ if

$$x \in \mathcal{O}^{\mathcal{W}} \;\;\Rightarrow\;\; f(x, w) \in \mathcal{O}^{\mathcal{W}} \;\;,\;\; \text{for all } w \in \mathcal{W}$$

Control Invariant Set

A set $\mathcal{C}^{\mathcal{W}} \subseteq \mathcal{X}$ is said to be a robust control invariant set for the **controlled** system $x(k+1) = f(x(k), u(k), w(k))$, if

$$x(k) \in \mathcal{C}^{\mathcal{W}} \quad \Rightarrow \quad \exists u(k) \in \mathcal{U} \text{ such that } f(x(k), u(k), w(k)) \in \mathcal{C}^{\mathcal{W}} \quad \text{for all } w \in \mathcal{W}, k \in \mathbb{N}^{+}$$

Defines the states for which there exists a **controller** that will robustly satisfy constraints for **all time**.

Safe Set Based on Robust Invariance

Consider the system model $x(k+1) = f(x(k), u(k), w(k)$, with $x \in \mathcal{X}, u \in \mathcal{U}, w \in \mathcal{W}$

Approach: Select control law $u = \pi_s(x)$, define safe set as invariant set for controlled system
$x(k+1) = f(x(k), \pi_s(x(k)), w(k)$.

Here: Define safe set as level set of safety value function $\mathcal{S} = \{x \in \mathbb{R}^n \mid V(x) \leq 1\}$.

Conditions on safe set:

1. Robust invariance: $V(x(k)) \leq 1 \Rightarrow V(x(k+1)) \leq 1 \; \forall w(k) \in \mathcal{W}$
2. Constraint satisfaction: $\mathcal{S} \subseteq \mathcal{X}$
3. Feasibility of safety controller: $\pi_s(x) \in \mathcal{U} \; \forall x \in \mathcal{S}$

Safe Set Based on Robust Invariance for Linear System

Consider the system model $x(k+1) = Ax(k) + Bu(k) + w(k)$, with $x \in \mathcal{X}, u \in \mathcal{U}, w \in \mathcal{W}$

Approach: Select control law $\pi_s(x) = Kx$, define safe set as invariant set for controlled system $x(k+1) = (A + BK)x(k) + w(k)$.

Here: Define safe set as level set of safety value function $\mathcal{S} = \{x \in \mathbb{R}^n \mid x^T P x \leq 1\}$.

Conditions on safe set:

1. Robust invariance: $x(k)^T P x(k) \leq 1 \Rightarrow x(k+1)^T P x(k+1) \leq 1 \; \forall w(k) \in \mathcal{W}$
2. Constraint satisfaction: $\mathcal{S} \subseteq \mathcal{X}$
3. Feasibility of safety controller: $\pi_s(x) = Kx \in \mathcal{U} \; \forall x \in \mathcal{S}$ (linear control law)

- Simple approach: Compute stabilizing K and then robust positively invariant set.
- Optimize over K and P
  - Condition 2 and 3 can again be written as Linear Matrix Inequalities (LMIs) in P^{-1}
  - Condition 1 can be written as bilinear matrix inequality in P^{-1}, $Y = KP^{-1}$ and scalar variables using S-procedure (see e.g. S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, "Linear Matrix Inequalities in System and Control Theory", 1994.)

Remark: Effects of Safety Filter on Performance

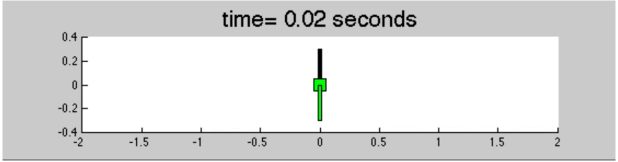
- Filter interventions can negatively affect performance of learning (or other) algorithm
- Common approach: Inclusion of safety metric in learning controller
  - Based on safety value function
    Example: Add barrier term to cost used in learning algorithm

$$C_s(x, u) = C(x, u) - \gamma log(1 - V(x))$$

(barrier goes to infinity at boundary of safe set)

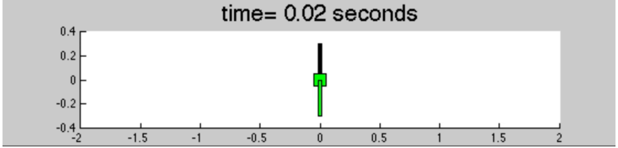
Example: Including Barrier based on Safety Value Function

Example: Learning to control pendulum swingup



Add barrier term to cost used in learning algorithm

$$C_s(x, u) = C(x, u) - \gamma log(1 - V(x))$$



Related Work & Extensions

- Invariance-based approach for distributed (e.g. [4]) or nonlinear systems (e.g. [3])
- Reachability analysis (e.g. [1])
- Control barrier functions (e.g. [2])
- Extensions updating the model and disturbance bound from data (e.g. [7]).

(often for continuous-time systems)

Note:

- There are various other safety concepts (safe RL [6], RL with temporal logic constraints,...)

Main limitation of related techniques: Explicit computation of safe set and controller can result in

- conservativeness
- challenging design or computation
- scalability issues

Idea: Approximate maximum control invariant set with MPC

References

[1] J. Gillula and C.J. Tomlin, “Guaranteed Safe Online Learning via Reachability: tracking a ground target using a quadrotor”, ICRA 2012

[2] A. Ames et al., “Control Barrier Functions: Theory and Applications”, ECC 2019

[3] K. Wabersich and M.N. Zeilinger, “Scalable synthesis of safety certificates from data with application to learning-based control”, ECC 2018

[4] A. Carron, K. P. Wabersich, and M. N. Zeilinger, “Plug-and-play distributed safety verification for linear control systems with bounded uncertainties,” IEEE Transactions on Control of Network Systems, vol. 8, no. 3, pp. 1501-1512, 2021.

[5] F. Berkenkamp et al., “Safe model-based reinforcement learning with stability guarantees”, NIPS 2017

[6] J. Garcia and F. Fernandez, “A comprehensive survey on safe reinforcement learning”, 2015.

[7] Fisac et al., “A General Safety Framework for Learning-Based Control in Uncertain Robotic Systems”, TAC, 2019.