Safe Reinforcement Learning

Mayank S JHA*,
Didier THEILLOL

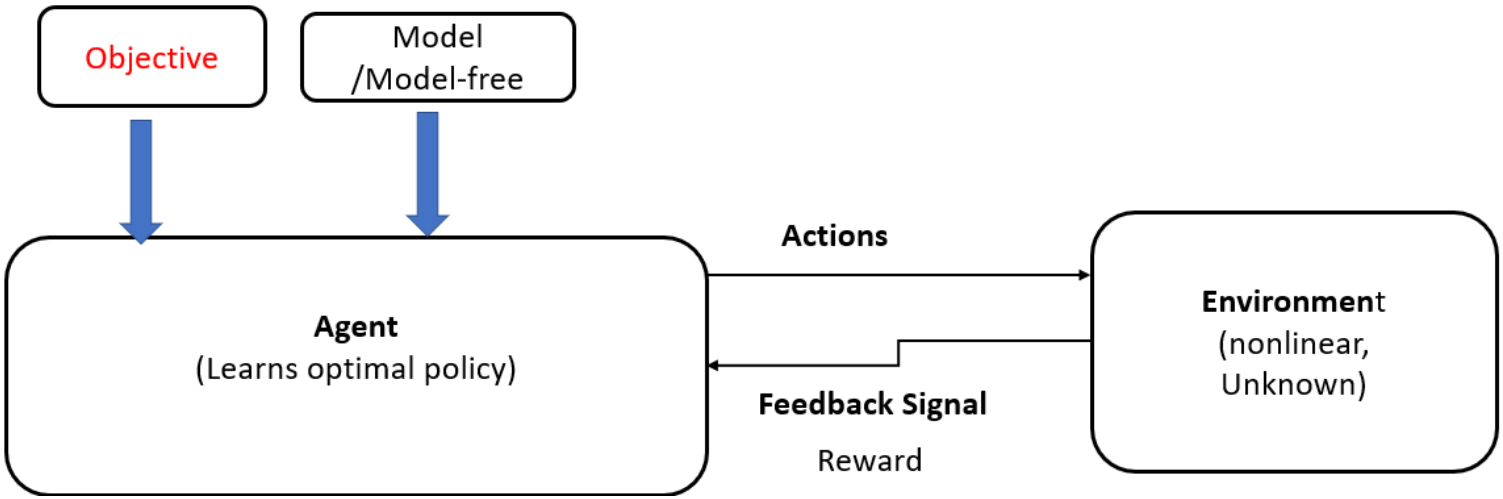
Le Centre de Recherche en Automatique de Nancy (CRAN),
UMR 7039, CNRS,
Université de Lorraine

SAUTOS: Safe AUTOnomous Systems,
Université Paris-Saclay

Table of Contents

Table of Contents

Reinforcement Learning Architecture



Reinforcement Learning: Automatic Control

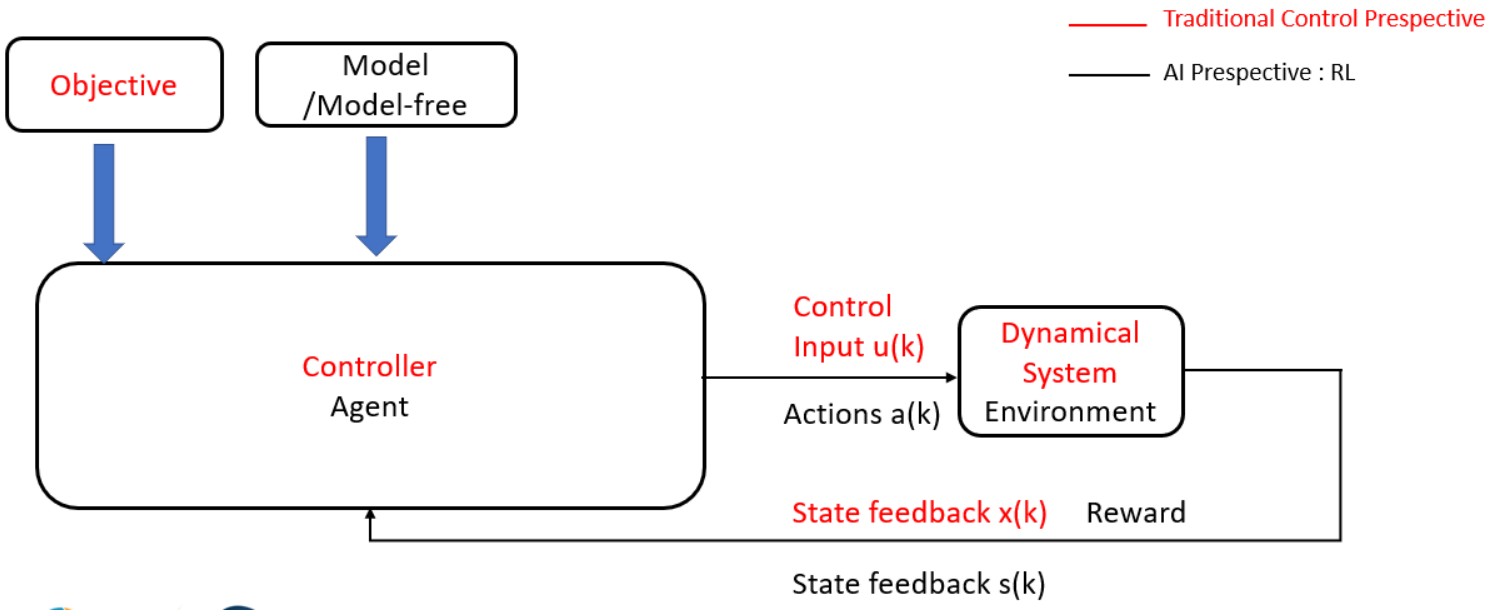


Table of Contents

RL: Discrete time optimal control

System

$$x_{k+1} = f(x_k) + g(x_k)u(x_k) \tag{1}$$

- $x_k \in \Omega \subset \mathbb{R}^n$ is the state variable vector
- Ω being a compact set
- $u(x_k) \in U \subset \mathbb{R}^m$ is the control input vector
- $f(x)$ is C^1 and $x = 0$ is an equilibrium state such that $f(0) = 0$ and $g(0) = 0$.

Note: $u(x_k)$ will be denoted as u_k.

RL: Discrete time optimal control

Control law/ Policy

A control policy is a function from state space to control space $\pi(\cdot) : \mathbb{R}^n \to \mathbb{R}^m$, that defines for every state x_k , a control action:

$$u_k = \pi(x_k) \tag{2}$$

- Such mappings $\to$ feedback controllers.
- Example: linear state-variable feedback $u_k = \pi(x_k) = -Kx_k$

RL: Discrete time optimal control

Goal directed performance

Cost-to-go is a sum of (discounted) future costs from the current time k into the infinite horizon future under a prescribed control law $u_k = \pi(x_k)$:

$$J(x_k, u_k) = \sum_{n=k}^{\infty} r(x_n, u_n) \tag{3}$$

where $r(x_n, u_n)$ is the utility function defined as:
$r(x_n, u_n) = x_n^T Q x_n + u_n^T R\, u_n$

- Q symmetric positive semi-definite matrix $Q = Q^T \geqslant 0$
- R is a symmetric positive definite matrix $R = R^T > 0$.

RL: Discrete time optimal control

Assumption: Stabilizable system

System (1) is *stabilizable* on the prescribed set $\Omega \in \mathbb{R}^n$.

$\Rightarrow$ There is a control policy $u_k^1 = \pi(x)$ such that closed loop system $x_{k+1} = f(x_k) + g(x_k)u_k^1$ is asymptotically stable over Ω i.e. $u_k^1 = (u^1(x_k), u^1(x_{k+1}), u^1(x_{k+2}), ...u^1(x_\infty))$ exists that

- that stabilises the system (1)
- associated cost $J(x_k, u_k^1)$ is finite.

U denotes the set of all admissible control inputs.

RL: Discrete time optimal control

For a given admissible prescribed policy $\pi(x)$,
the cost associated is called as it *value* denoted as
$V_\pi(x_k) = J(x_k, \pi(x))$

RL: Discrete time optimal control

Objective: Optimal Cost

To find a control policy $\pi^*(x_k)$ that minimizes the infinite horizon cost function,

$$V^*(x_k) = \min_{u_k \in U} \sum_{n=k}^{\infty} r(x_n, u_n), \forall x_k \tag{4}$$

or, $V^*(x_k) = \min_{\pi(\cdot)} \sum_{n=k}^{\infty} r(x_n, \pi(x_n)), \forall x_k$

$V^*(x_k) ==>$ optimal cost or optimal value.

Optimal policy

Optimal control policy: $\pi^*(x_k) = \arg\min_{\pi(\cdot)} \sum_{n=k}^{\infty} r(x_n, \pi(x_n)), \forall x_k$

RL: Discrete time optimal control

Cost (given a prescribed policy $u_k = \pi(x_k)$)

$$V_\pi(x_k) = \sum_{n=k}^{\infty} r(x_n, u_n), \forall x_k$$
$$V_\pi(x_k) = r(x_k, u_k) + V_\pi(x_{k+1})$$

Bellman Eq/ Nonlinear Lyapunov Eq (Recursive): Hamiltonian:

$$H(x_k, u_k, V_\pi) = r(x_k, u_k) + V_\pi(x_{k+1}) - V_\pi(x_k)$$

Optimal Cost:

$$V^*(x_k) = \min_{u_k \in U} \left(r(x_k, u_k) + V_\pi(x_{k+1})\right)$$

RL: Discrete time optimal control

Bellman Principle Bellman, 1957

"An optimal policy has the property that no matter what the previous decisions (i.e. controls) have been, the remaining decisions must constitute an optimal policy with regard to the state resulting from those previous decisions

RL: Discrete time optimal control

Cost (given a prescribed policy $u_k = \pi(x_k)$)

$$V_\pi(x_k) = \sum_{n=k}^{\infty} r(x_n, u_n), \forall x_k$$
$$V_\pi(x_k) = r(x_k, u_k) + V_\pi(x_{k+1})$$

Bellman Eq/ Nonlinear Lyapunov Eq (Recursive): Hamiltonian:

$$H(x_k, u_k, V_\pi) = r(x_k, u_k) + V_\pi(x_{k+1}) - V_\pi(x_k)$$

Optimal Cost:

$$V^*(x_k) = \min_{u_k \in U} \left(r(x_k, u_k) + V_\pi(x_{k+1})\right)$$

Bellman principle:

$$V^*(x_k) = \min_{u_k \in U} \left(r(x_k, u_k) + V^*(x_{k+1})\right)$$

Backwards in Time!!

Optimal control (policy):

$$\pi^*(x_k) = \arg\min_{u_k \in U} \left(r(x_k, u_k) + V^*(x_{k+1})\right)$$

RL: Discrete time optimal control

Cost (given a prescribed policy $u_k = \pi(x_k)$)

$$V_\pi(x_k) = \sum_{n=k}^{\infty} r(x_n, u_n), \forall x_k$$
$$V_\pi(x_k) = r(x_k, u_k) + V_\pi(x_{k+1})$$

Bellman Eq/ Nonlinear Lyapunov Eq (Recursive): Hamiltonian:

$$H(x_k, u_k, V_\pi) = r(x_k, u_k) + V_\pi(x_{k+1}) - V_\pi(x_k)$$

Optimal Cost:

$$V^*(x_k) = \min_{u_k \in U} \left(r(x_k, u_k) + V_\pi(x_{k+1})\right)$$

Bellman principle:

$$V^*(x_k) = \min_{u_k \in U} \left(r(x_k, u_k) + V^*(x_{k+1})\right)$$

Optimal control (policy):

$$\pi^*(x_k) = \arg\min_{u_k \in U} \left(r(x_k, u_k) + V^*(x_{k+1})\right)$$

Only data required!!

RL: Discrete time optimal control

Bellman principle: *(DT Hamilton-Jacobi-Bellman Equation)*

$$V^*(x_k) = \min_{u_k \in U} \left(r(x_k, u_k) + V^*(x_{k+1}) \right)$$
$$= \min_{u_k \in U} \left(x_k^T Q x_k + u_k^T R\, u_k + V^*(x_{k+1}) \right)$$
$$= \min_{u_k \in U} \left(x_k^T Q x_k + u_k^T R\, u_k + V^*(f(x_k) + g(x_k) u_k) \right)$$

Optimal control (policy):

$$\pi^*(x_k) = \arg\min_{u_k \in U} \left(r(x_k, u_k) + V^*(x_{k+1}) \right)$$
$$\pi^*(x_k) = u_k^* = (-1/2) R^{-1} g^T(x_k) \frac{\partial V^*(x_{k+1})}{\partial x_{k+1}}$$

RL: Discrete time optimal control

Cost (given a prescribed policy $u_k = \pi(x_k)$)

$$V_\pi(x_k) = \sum_{n=k}^{\infty} r(x_n, u_n), \forall x_k$$

Bellman Eq/ Nonlinear Lyapunov Eq (Recursive):

$$V_\pi(x_k) = r(x_k, u_k) + V_\pi(x_{k+1})$$

Optimal Cost:

$$V^*(x_k) = \min_{u_k \in U} \left(r(x_k, u_k) + V_\pi(x_{k+1})\right)$$

Bellman principle:

$$V^*(x_k) = \min_{u_k \in U} \left(r(x_k, u_k) + V^*(x_{k+1})\right)$$

Optimal control (policy):

$$\pi^*(x_k) = \arg\min_{u_k \in U} \left(r(x_k, u_k) + V^*(x_{k+1})\right)$$

DT Policy Iteration

Initialization

Select any *stabilizing* /admissible control policy: $\pi_j(x_k)$

Policy Evaluation

Determine the *Value* under the current policy using Bellman Equation/Nonlinear Lyapunov Eq.

$$V_{j+1}(x_k) = r(x_k, \pi_j(x_k)) + V_{j+1}(x_{k+1}) \; ; \; V_{j+1}(0) = 0$$

Policy Improvement

Determine an improved policy

$$\pi_{j+1}(x_k) = \underset{u_k \in U}{\arg\min} \; (r(x_k, u_k) + V_{j+1}(x_{k+1}))$$

DT Policy Iteration

Initialization

Select any *stabilizing* /admissible control policy: $\pi_j(x_k)$

Policy Evaluation

Determine the *Value* under the current policy using Bellman Equation/Nonlinear Lyapunov Eq.

$V_{j+1}(x_k) = r(x_k, \pi_j(x_k)) + V_{j+1}(x_{k+1})$; $V_{j+1}(0) = 0$

Policy Improvement

Determine an improved policy

$$\pi_{j+1}(x_k) = \arg\min_{u_k \in U} \left(r(x_k, u_k) + V_{j+1}(x_{k+1})\right)$$

DT Policy Iteration

Initialization

Select any *stabilizing* /admissible control policy: $\pi_j(x_k)$

Policy Evaluation

Determine the *Value* under the current policy using Bellman Equation/Nonlinear Lyapunov Eq.

$V_{j+1}(x_k) = r(x_k, \pi_j(x_k)) + V_{j+1}(x_{k+1})$; $V_{j+1}(0) = 0$

Policy Improvement

Determine an improved policy

$$\pi_{j+1}(x_k) = \arg\min_{u_k \in U} \left(r(x_k, u_k) + V_{j+1}(x_{k+1})\right)$$

DT Policy Iteration

Initialization

$\pi_j(x_k)$

Policy Evaluation

$V_{j+1}(x_k) = r(x_k, \pi_j(x_k)) + V_{j+1}(x_{k+1})$

Policy Improvement

$\pi_{j+1}(x_k) = \underset{u_k \in U}{\arg\min}\ (r(x_k, u_k) + V_{j+1}(x_{k+1}))$

When $r(x_k, u_k) = x_k^T Q x_k + u_k^T R\ u_k$,

$\pi_{j+1}(x_k) = (-1/2) R^{-1} g^T(x_k) \frac{\partial V_{j+1}(x_{k+1})}{\partial x_{k+1}}$

DT Policy Iteration: Observations

- Initial policy must be stabilizing.
- Policy Iteration (Howard, 1960; Leake and Liu, 1967) $\Rightarrow$
  - $V_{j+2}(x_k) \leq V_{j+1}(x_k)$
- As $j \to \infty$:
  - $V_j(x_k) \to V^*(x_k)$
  - $\pi_j \to \pi^*$
- Convergence to optimal cost and thus, optimal control policy.

DT Policy Iteration: Observations

- $V_{j+1}(x_k) = r(x_k, \pi_j(x_k)) + V_{j+1}(x_{k+1}); \forall x_k \in \Omega$
  - value of using a given policy starting in all current states possible.
  - Several states $\Rightarrow$ Significant computations!
- Called *full backup* (Sutton and Barto, 2018)$\Rightarrow$ Massive computational load
- Bellman Eq $\rightarrow$ *fixed point equation*
  - Given admissible policy π_j, $V^{i+1}(x_k) = r(x_k, \pi_j(x_k)) + V^i(x_{k+1})$ is a *contraction map*
  - Upon iterated starting from $V^0(x_k)$, $V^i(x_k) \rightarrow V_{j+1}(x_k)$ as $i \rightarrow \infty$.

DT Policy Iteration: Observations

- $V_{j+1}(x_k) = r(x_k, \pi_j(x_k)) + V_{j+1}(x_{k+1}); \forall x_k \in \Omega$
  - value of using a given policy starting in all current states possible.
  - Several states $\Rightarrow$ Significant computations!
- Called *full backup* (Sutton and Barto, 2018)$\Rightarrow$ Massive computational load
- Bellman Eq $\rightarrow$ *fixed point equation*
  - Given admissible policy π_j, $V^{i+1}(x_k) = r(x_k, \pi_j(x_k)) + V^i(x_{k+1})$ is a *contraction map*
  - Upon iterated starting from $V^0(x_k)$, $V^i(x_k) \rightarrow V_{j+1}(x_k)$ as $i \rightarrow \infty$.

Issues

- This strategy $\Rightarrow$ *backward in time procedure*
- Good for:
  - Off-line planning, Offline optimization, Offline control synthesis.
  - NOT online leanring (optimla control synthesis using real time data measured along system trajectories.
- Exact solutions: very difficult
  - Large state space
  - Highly nonlinear dynamics

Issues

- This strategy $\Rightarrow$ *backward in time procedure*
- Good for:
  - Off-line planning, Offline optimization, Offline control synthesis.
  - NOT online leanring (optimla control synthesis using real time data measured along system trajectories.
- Exact solutions: very difficult
  - Large state space
  - Highly nonlinear dynamics

Issues

- This strategy $\Rightarrow$ *backward in time procedure*
- Good for:
  - Off-line planning, Offline optimization, Offline control synthesis.
  - NOT online learning (optimal control synthesis using real time data measured along system trajectories.
    Temporal Difference (TD) or *forward in time learning*
- Exact solutions: very difficult
  - Large state space
  - Highly nonlinear dynamics
    Value Function approximation (VFA): Neural Networks

Forward-in-time Learning

Temporal Difference Error (TD error):
$e_k = r(x_k, \pi_{x_k}) + V_\pi(x_{k+1}) - V_\pi(x_k)$

- RHS is DT Hamiltonian
- If Bellman Eq holds, e_k is zero.
- Linear in x.
- Thus, given a policy $\pi(x)$, Least Square based solution at each time k for $e_k = 0$.

NN based approximation

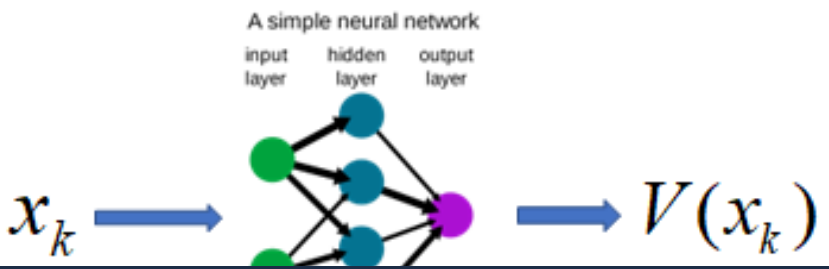
Value Function approximation (VFA): Neural Networks

- Value function is sufficiently smooth over compact space
- Consider dense basis set $\{\phi_i(x)\}$ with basis vector (Weierstrass Theorem):
  $\phi(x) = [\varphi_1(x)\varphi_2(x)...\varphi_L(x)] : \mathbb{R}^n \rightarrow \mathbb{R}^L$

$$V_\pi(x) = \sum_{i=1}^{L} w_i \varphi_i(x) = W^T \phi(x)$$

Substituting in Bellman TD equation:

$$e_k = r(x_k, \pi_{x_k}) + W^T \phi(x_{k+1}) - W^T \phi(x_k)$$



Online DT Policy Iteration

Initialization Choose an initial stabilizing policy (admissible): $\pi_0(x_k)$

Policy Evaluation

$V_{j+1}(x_k) = r(x_k, \pi_j(x_k)) + V_{j+1}(x_{k+1})$

$$r(x_k, \pi_{x_k}) = W_{j+1}^T(\phi(x_k) - \phi(x_{k+1}))$$

Policy Improvement

$$\pi_{j+1}(x_k) = \arg\min_{u_k \in U} \left(r(x_k, u_k) + W_{j+1}^T(\phi(x_{k+1})) \right)$$

When $r(x_k, u_k) = x_k^T Q x_k + u_k^T R\, u_k$,

$\pi_{j+1}(x_k) = (-1/2) R^{-1} g^T(x_k) \nabla \phi^T(x_{k+1}) W_{j+1}$

Online Policy Iteration

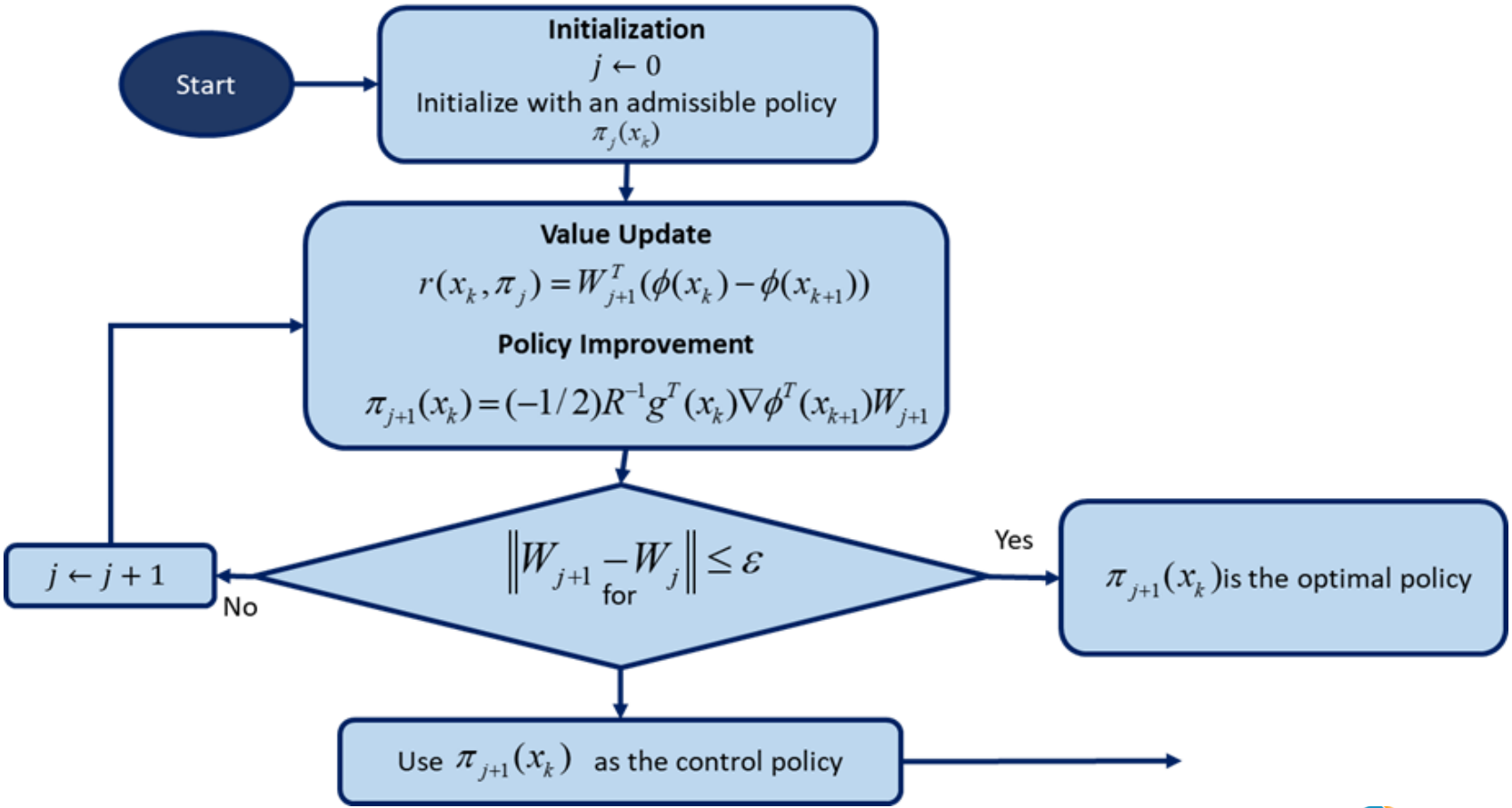


Figure: Online PI

Online DT Policy Iteration: Observations

- At $k+1$: observe $x_k, u_k = \pi_j(x_k), x_{k+1}$
- Calculate $r(x_k, u_k)$ One scalar Equation in $r(x_k, \pi_{x_k}) = W_{j+1}^T(\phi(x_k) - \phi(x_{k+1}))$
- Use same policy $u_k = \pi_j(x_k)$, collect L data $\Rightarrow$ L equations $(!!\phi(x) = \mathbb{R}^n \rightarrow \mathbb{R}^L)$.
- Determine LS based solution $\hat{W}_{j+1}$
- Repeat till $\hat{W}_{j+1} \equiv \hat{W}_{j+2} \rightarrow W^*$ Apply Improved control

Execution: Adaptive Critic Structures

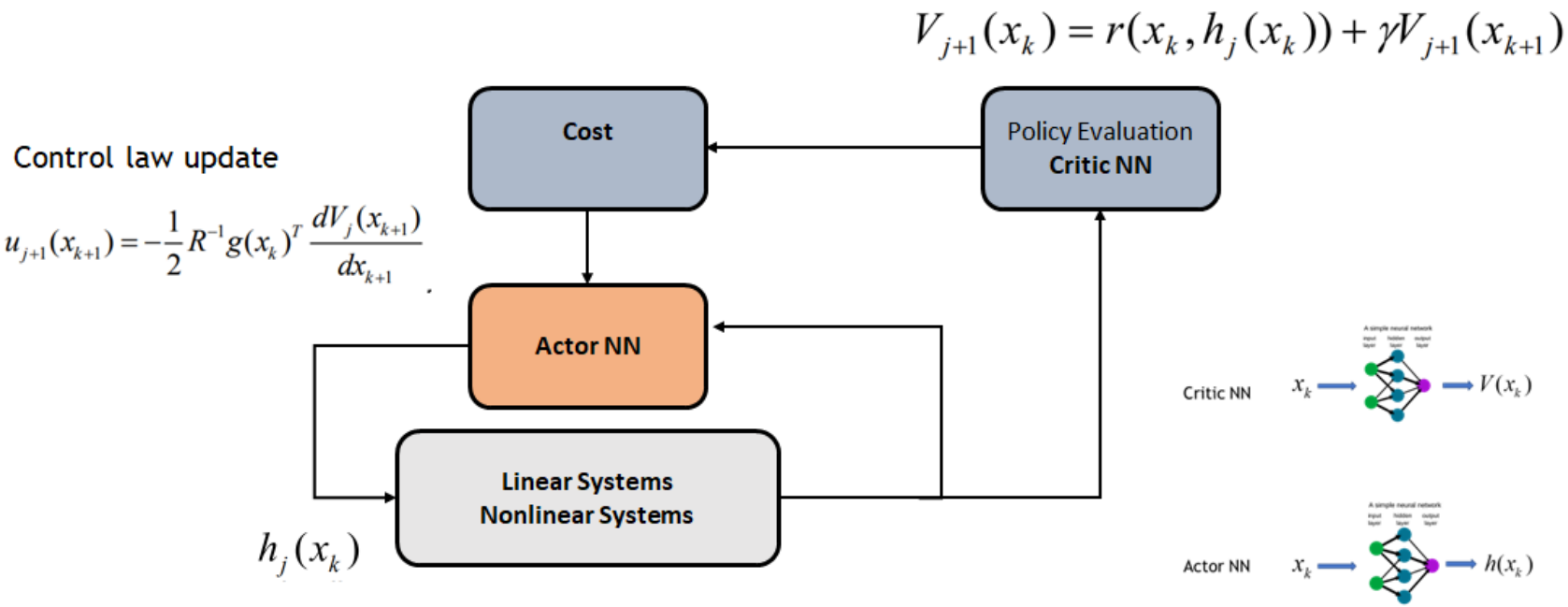


Figure: Actor Critic Structure

Execution: Adaptive Critic Structures: Two time Scale!

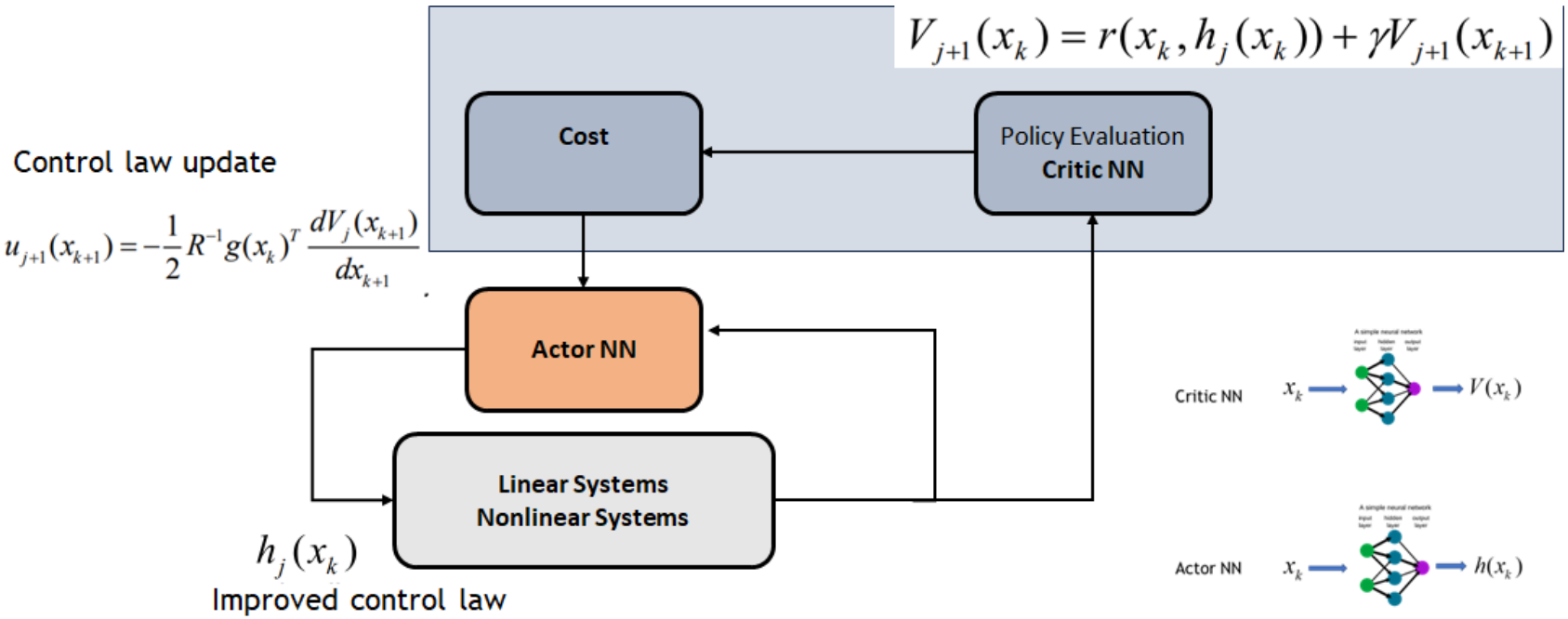


Figure: Actor Critic Structure

Execution: Adaptive Critic Structures

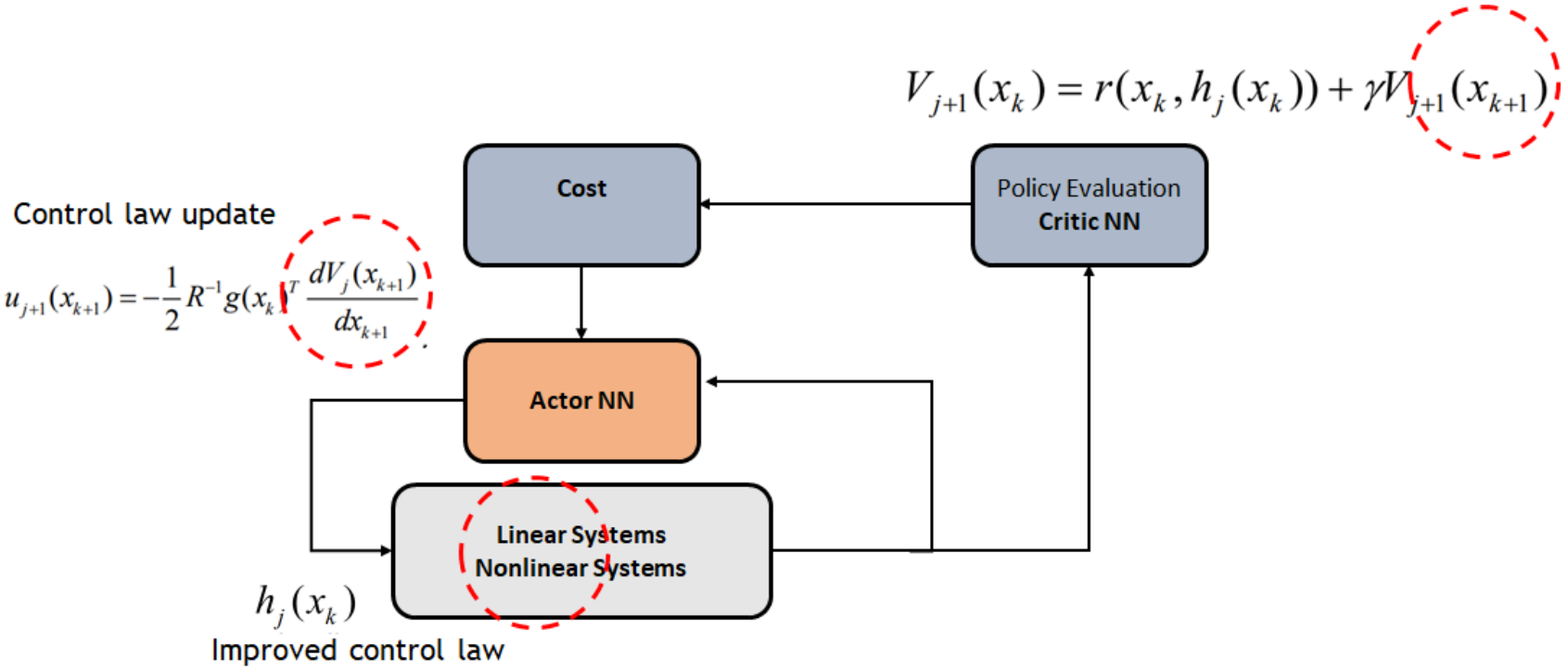


Figure: Actor Critic Structure

Execution: Adaptive Critic Structures

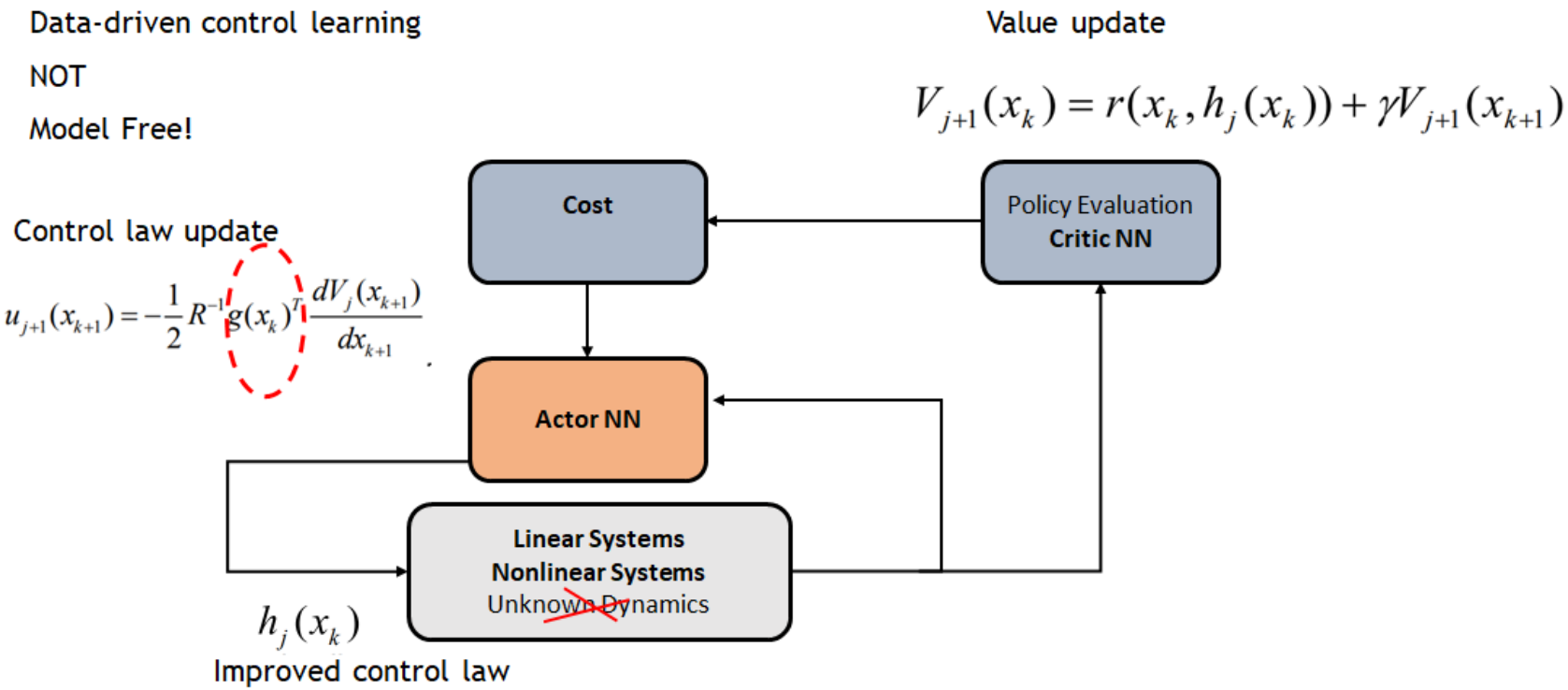


Figure: Actor Critic Structure

Execution: Adaptive Critic Structures

RL: Model free approach ➔ Q-function

System $x_{k+1} = f(x_k) + g(x_k)u_k$

$V_h(0) = 0$

$$Q_h(x_k, u_k) = r(x_k, u_k) + \gamma V_h(x_{k+1})$$

$$Q_h(x_k, u_k) = r(x_k, u_k) + \gamma Q_h(x_{k+1}, h(x_{k+1}))$$

$$V^*(x_k) = \min_{u_k}(Q^*(x_k, u_k))$$

$$h^*(x_k) = \arg\min_{u_k}(Q^*(x_k, u_k))$$

$$u^*(x_k) = -\frac{1}{2} R^{-1} g(x_k)^T \frac{dV^*(x_{k+1})}{dx_{k+1}}$$

Table of Contents

Conventional RL

Conventional RL:

Does:

- Stability
- Optimality: Performance, energy consumption etc.

Does NOT:

- ensure SAFETY.

Poses "Threat"

- during Exploration: Learning phase.
- during Exploitation: **Operational phase.**

Conventional RL

Conventional RL:
Does:

- Stability
- Optimality: Performance, energy consumption etc.

Does NOT:

- ensure SAFETY.

Poses "Threat"

- during Exploration: Learning phase.
- during Exploitation: **Operational phase.**

Conventional RL

Conventional RL:
Does:

- Stability
- Optimality: Performance, energy consumption etc.

Does NOT:

- ensure SAFETY.

Poses "Threat"

- during Exploration: Learning phase.
- during Exploitation: **Operational phase.**

Conventional RL

Conventional RL:
Does:

- Stability
- Optimality: Performance, energy consumption etc.

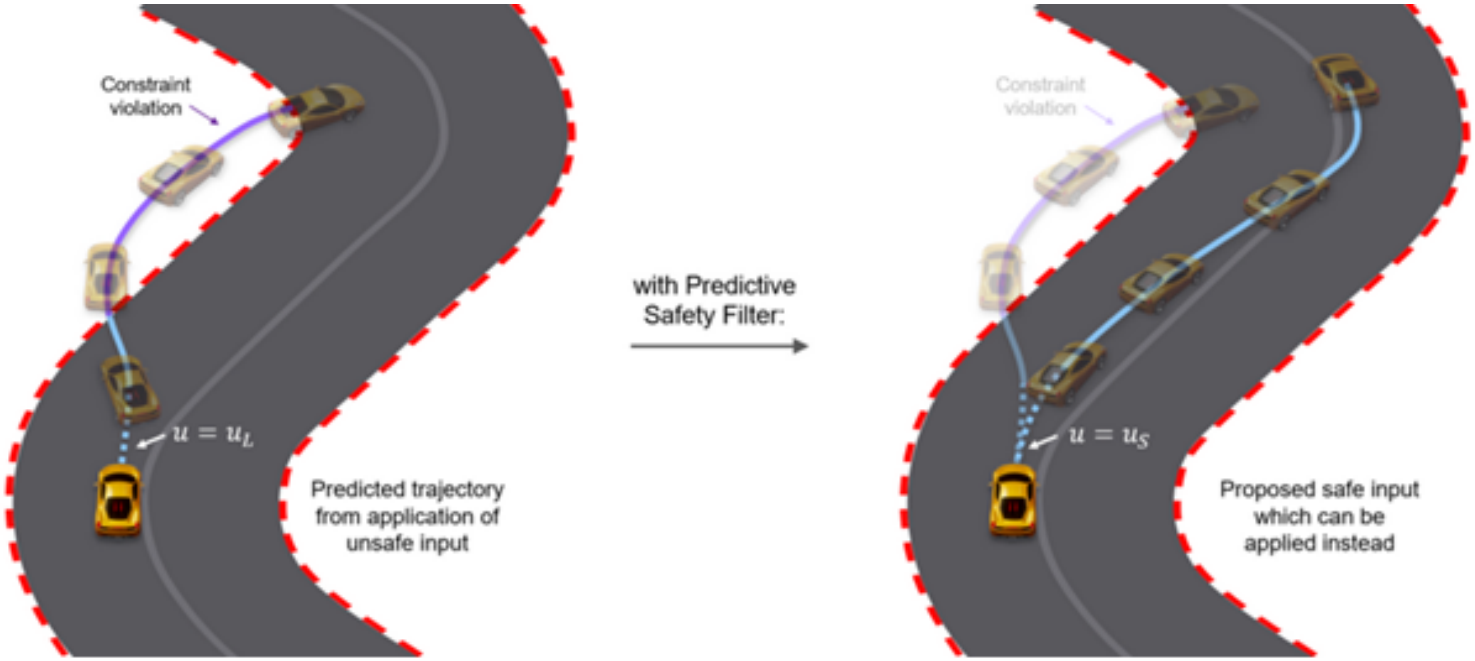
Does NOT:

- ensure SAFETY.

Poses "Threat"

- during Exploration: Learning phase.
- during Exploitation: **Operational phase.**

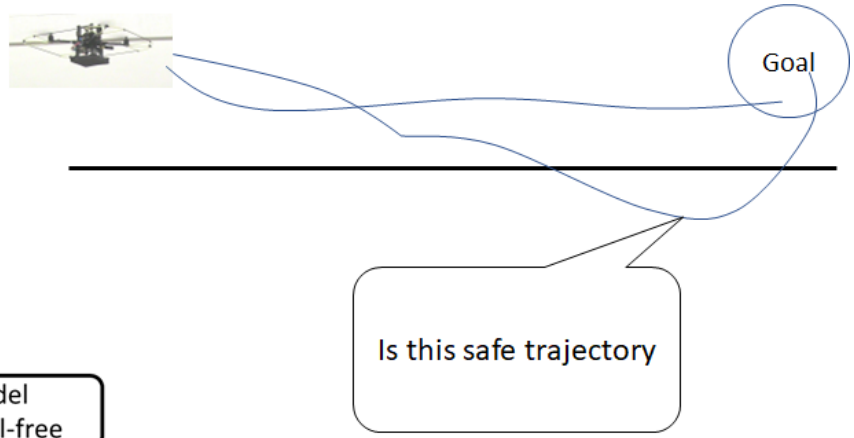
Safe Learning

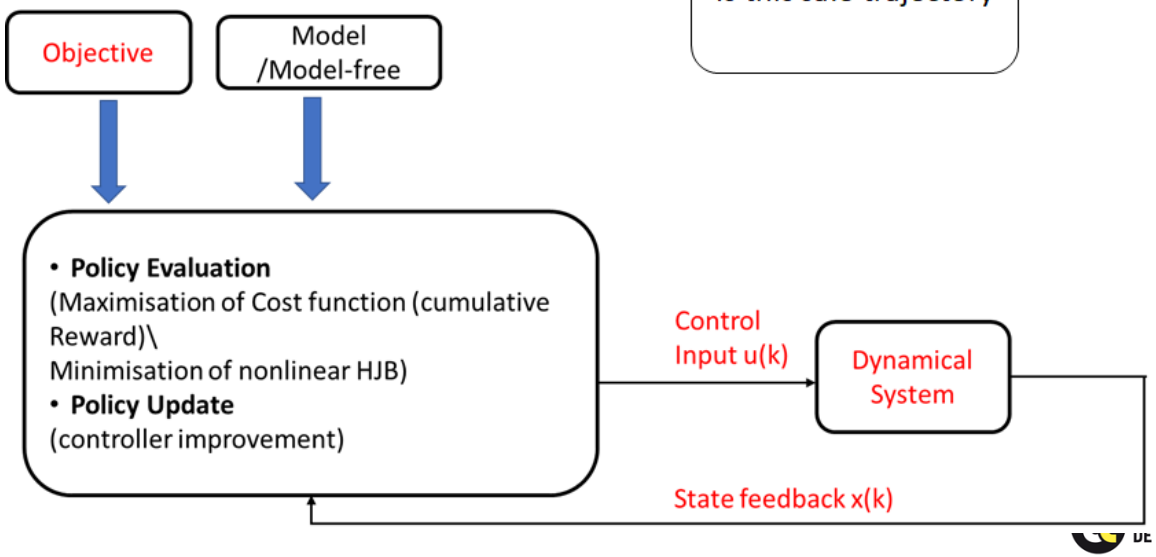


Wabersich, K. P., & Zeilinger, M. N. (2021). A predictive safety filter for learning-based control of constrained nonlinear dynamical systems. ***Automatica,*** *129*, 109597.

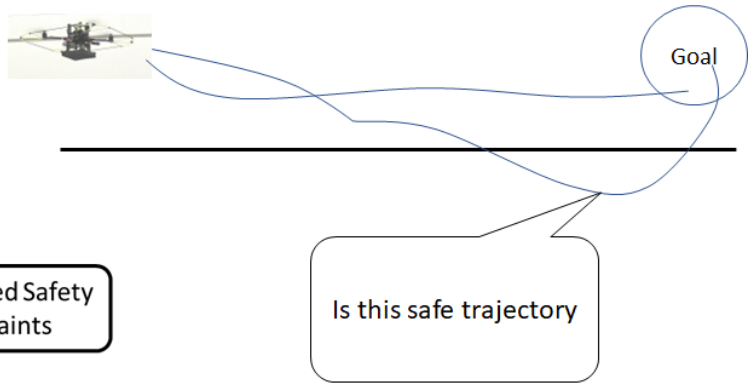
- Sequence of speed control inputs leads to DANGER!
- Hard constraints:
  - conservative performance (optimality not guaranteed).

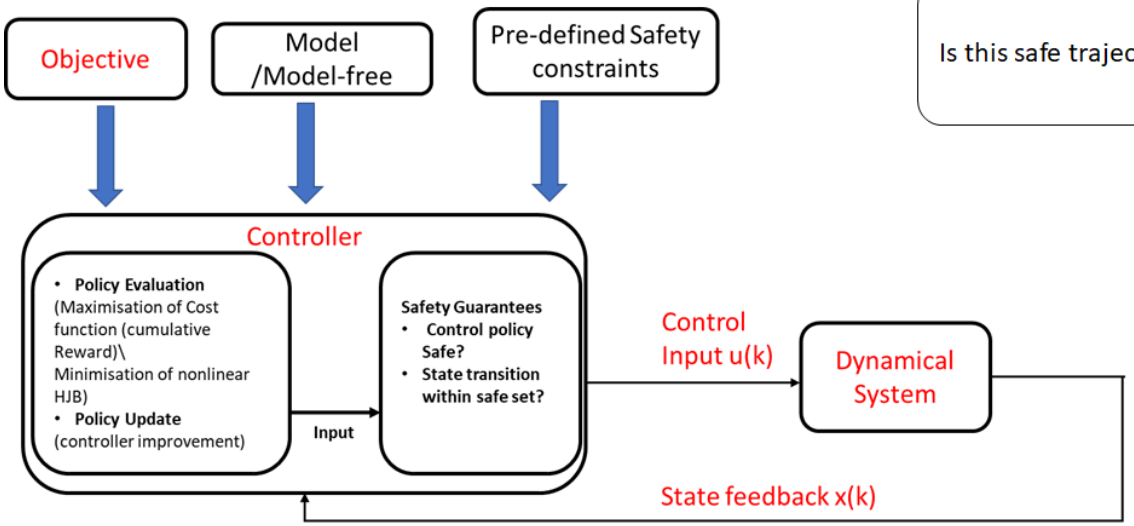
Safe Learning



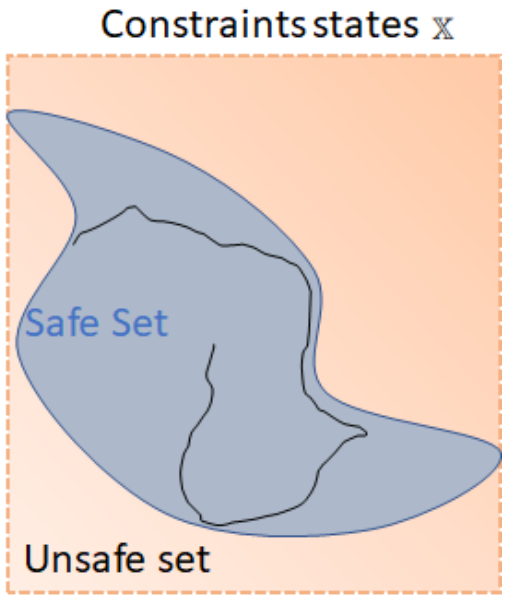


Safe Learning





Safe RL: Objectives



Learn control policy that ensures:

- Safety (states are within a "Safe set")
- Stability
- Optimal performance

Table of Contents

System

$$x_{k+1} = f(x_k) + g(x_k)u(x_k) \tag{1}$$

where:

- $x_k \in \Omega \subset \mathbb{R}^n$ states of the system
- $u(x_k) \in U \subset \mathbb{R}^m$ are the and the control input
- U denotes the set of all admissible control inputs
- $f(x_k) \in \mathbb{R}^n$represents the drift dynamics
- $g(x_k) \in \mathbb{R}^{n \times m}$ is the input dynamics.
- $f(x_k)$ is C^1 and $x = 0$ is an equilibrium state such that $f(0) = 0$ and $g(0) = 0$.
  It is assumed that system (1) is stabilizable on a prescribed set $\Omega \in \mathbb{R}^n$.

Safe Set

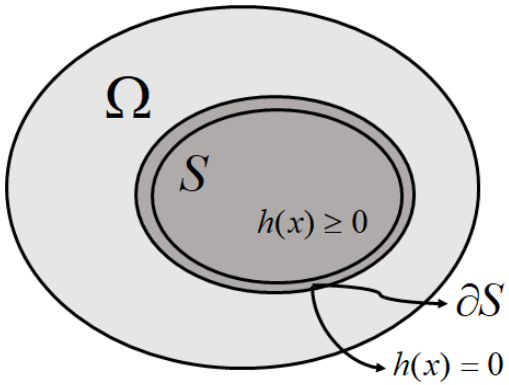
Definition

The safe set $\mathcal{S}$ and its boundary $\partial\mathcal{S}$ can be mathematically defined as:

$\mathcal{S} = \{x \in \Omega | h(x) \geqslant 0\}$

$\partial\mathcal{S} = \{x \in \Omega | h(x) = 0\}$

where $h(x) : \mathbb{R}^n \to \mathbb{R}$ belongs to C^1 and $h(x) > 0$ represents the admissible state space that respects the safety requirements.



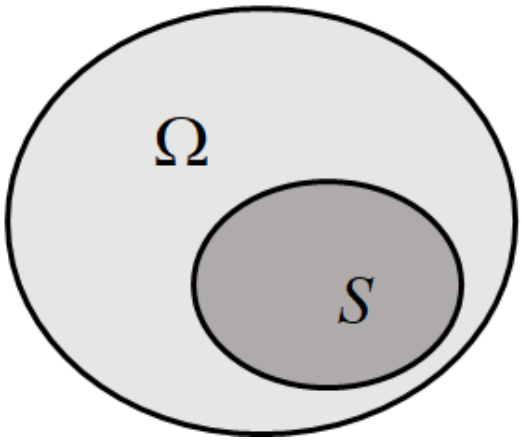
Strategy

Definition. *A set* $\mathcal{S} \in \Omega$ *is control invariant set if*
$x_k \in \mathcal{S} \Rightarrow \exists u_k \in U \quad | \quad x_{k+1} \in \mathcal{S} \quad \forall k \in \mathbb{Z}^+$
where $x_{k+1} = f(x_k) + g(x_k)u_k$
with $x_k \in \Omega \subset \mathbb{R}^n$ *and* $u_k \in U \subset \mathbb{R}^m$

Strategy:

Learning control law (sequence of control actions)

- *that ensures positive invariant property of safe set S,*
- *Optimality : performance + energy consumption etc.*



Safe Input set

Definition

Safe input set: The set of safe inputs for a current state x_k can be defined as the set of input that results in keeping next system's state within the interior of the safe set defined in (**??**):

$$\mathcal{U}^s = \{u \in \mathbb{R}^m | x_{k+1} \in int\mathcal{S}\} \tag{5}$$

where $x_{k+1} = f(x_k) + g(x_k)\, u$ is the state evolved with the input u.

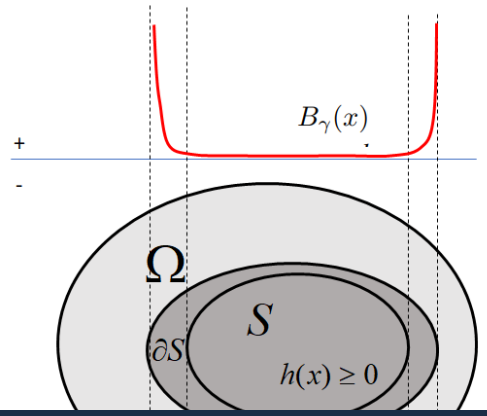
Class $\mathcal{K}$ function: A continuous function $\alpha : [0, \alpha) \to [0, \infty)$ is a class $\mathcal{K}$ function if it is strictly increasing and $\alpha(0) = 0$.

Barrier Function

Definition

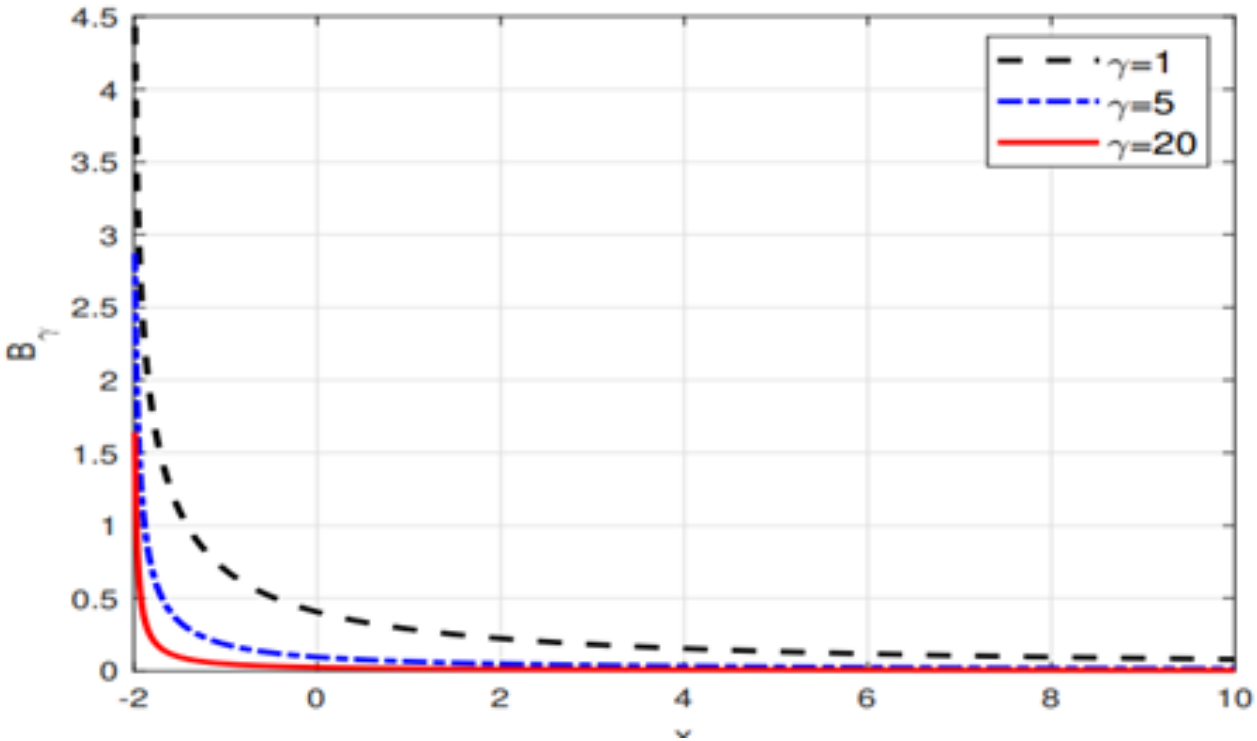
CBF properties (Ames et al., 2016; Brunke et al., 2022; Wabersich et al., 2023) : The BF candidate $B_\gamma(x) : \mathcal{S} \to \mathbb{R}$ satisfies the following properties:

1. $B_\gamma(x) > 0 \ \ \forall x \in \mathcal{S}$
2. $B_\gamma(x) \to \infty \ \ \forall x \in \partial\mathcal{S}$
3. $B_\gamma(x)$ is monotonically decreasing $\forall x \in \mathcal{S}$



Barrier Function Candidate

$$B_\gamma(x_k) = -\log\left(\frac{\gamma h(x_k)}{\gamma h(x_k)+1}\right)$$



Rate of Damping

Control Barrier Function CBF

Definition

Control Barrier functions for DT systems Agrawal and Sreenath, 2017: A function $B_\gamma(x) : \mathcal{S} \to \mathbb{R}$ is a CBF on the safe set $\mathcal{S}$ and for the nonlinear DT control system (1), if there exists:

1. locally Lipschitz class $\mathcal{K}$ functions α_1 and α_2 such that

$$\frac{1}{\alpha_1\left(h\left(x_k\right)\right)} \leqslant B_\gamma(x_k) \leqslant \frac{1}{\alpha_2\left(h\left(x_k\right)\right)}, \quad \forall x \in int\mathcal{S} \tag{6}$$

2. a safe control input $u_k \in \mathcal{U}^s$, $\forall x \in int\mathcal{S}$ such that

$$\Delta B_\gamma\left(x_{k+1}, x_k\right) := B_\gamma\left(f(x_k) + g(x_k)\, u_k\right) - B_\gamma\left(x_k\right) \leqslant \alpha_3(h(x_k)) \tag{7}$$

Control Barrier Function CBF

These conditions imply:

- u_k maintains the barrier function $B_\gamma(x_k) \geqslant 0$, $\forall k \in \mathbb{Z}^+$ given $B_\gamma(x_0) \geqslant 0$
- safe input maintains the trajectory of system within the safe set $\mathcal{S}$ if the initial state x_0 is within $\mathcal{S}$.

Safety Aware Control design

Classical cost-to-go modified and augmented with a CBF candidate as:

$$\min_{u \in U} \quad J_s(x_k, u) = \sum_{n=k}^{\infty} r_s(x_n, u_n) = \sum_{n=k}^{\infty} x_n^T Q x_n + u_n^T R\, u_n + B_\gamma(x_n)$$

$B_\gamma(x) : \mathcal{S} \to \mathbb{R}$ is augmented utility function $r_s(x_k, u_k)$ as:

$$r_s(x_k, u_k) = x_k^T Q x_k + u_k^T R\, u_k + B_\gamma(x_k) \tag{8}$$

The candidate CBF $B_\gamma(x)$ is sensitive to a coefficient γ that models the relative importance of the CBF to the utility function.

Safe Admissible policy and strict interiority

Definition

Safe admissible control policy: $\mathcal{U}^a = U \cap \mathcal{U}^s$

Definition

Strict interiority of initial condition:
The initial condition of system (1) remains strictly in the interior of the safe set $\mathcal{S}$, i.e. $x_0 \in int\mathcal{S}$.

Assumption

$\mathcal{U}^a = U \cap \mathcal{U}^s \neq \emptyset$

Simulations

Car model

$$\begin{bmatrix} y_{k+1} \\ v_{k+1} \\ \phi_{k+1} \\ \psi_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & Ts & v_{l0}.Ts & 0 \\ 0 & 1+(-\frac{C_f+C_r}{Mv_{l0}})Ts & 0 & (\frac{bC_r-aC_f}{Mv_{l0}} - v_{l0})Ts \\ 0 & 0 & 1 & Ts \\ 0 & (\frac{bC_r-aC_f}{I_z v_{l0}})Ts & 0 & 1 \end{bmatrix} \begin{bmatrix} y_k \\ v_k \\ \phi_k \\ \psi_k \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{C_f}{M} \\ 0 \\ a\frac{C_f}{I_z} \end{bmatrix} .Ts.u_k + \begin{bmatrix} 0 \\ 0 \\ -1 \\ 0 \end{bmatrix} .Ts.d_k \quad (18)$$

Simulations

Safety aware Reward/Utility function

$$r_s(x_k, u_k) = x_k^T Q x_k + u_k^T R u_k - m(log(\frac{\gamma(x_{1,k} + y_{max})}{\gamma(x_{1,k} + y_{max}) + 1}) + log(\frac{\gamma(-x_{1,k} + y_{max})}{\gamma(-x_{1,k} + y_{max}) + 1}))$$

- y_k and v_k are lateral displacement and its velocity
- y_{max} expresses the absolute value of maximum safe displacement from the center of the road.
- ϕ_k is error yaw angel and ψ_k is its derivative,
- u_k is the steering angle,
- d_k is the desired yaw rate obtained from the curvature of the road as $d_k = \frac{v_{l0}}{R}$;

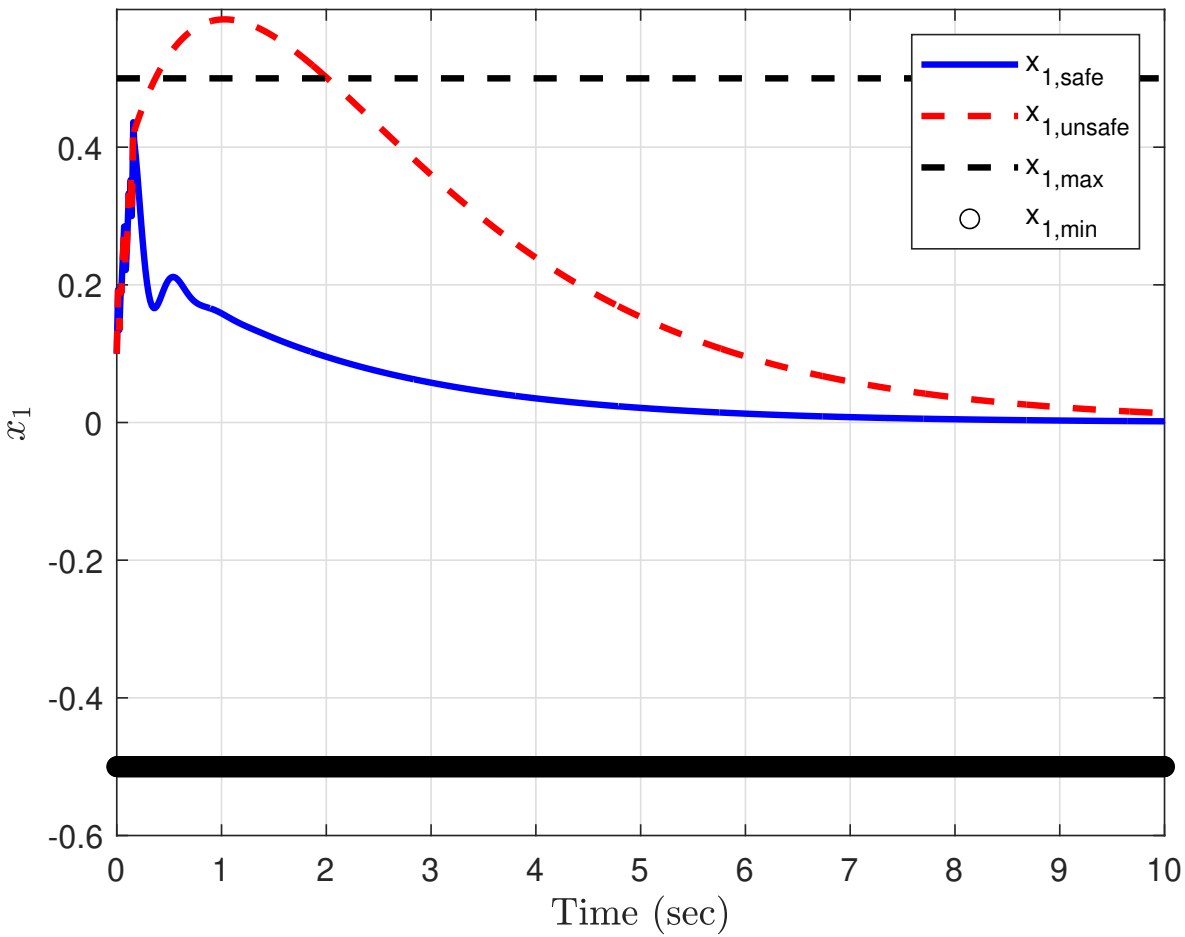
Simulations

Actor and Critic NNs

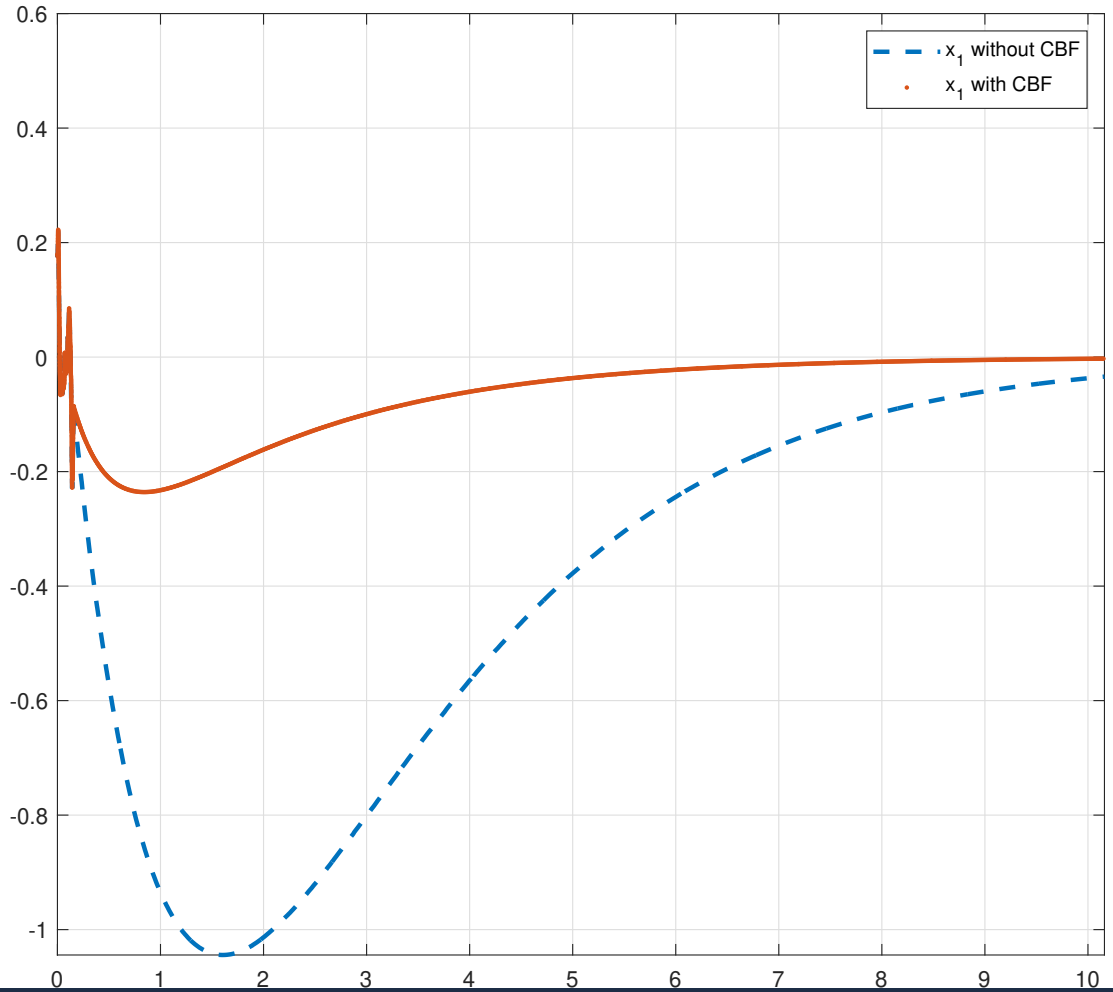
$$\Phi(x) = [x_1^2 \; x_2^2 \; x_3^2 \; x_4^2 \; x_1x_2 \; x_1x_3, \; x_1x_4 \; x_2x_3 \; x_2x_4 \; x_3x_4 \; (x_1 - y_{max})^2 \; {x_1}^4, {x_2}^4]$$

$$\Psi(x) = [x_1 \; x_2 \; x_3 \; x_4]^T$$

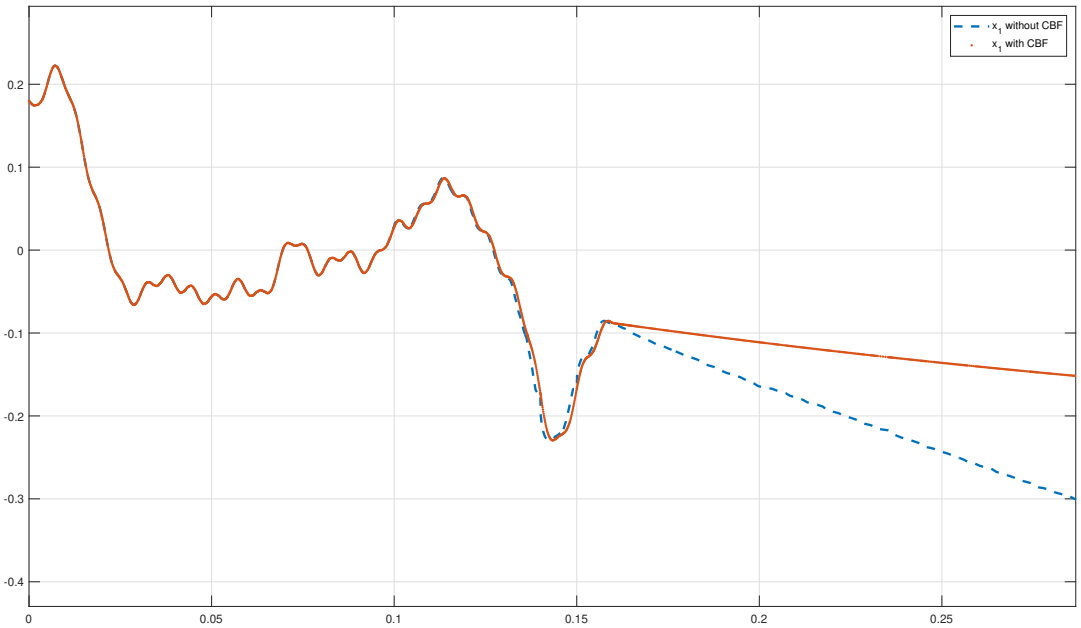
Lateral displacement zoomed



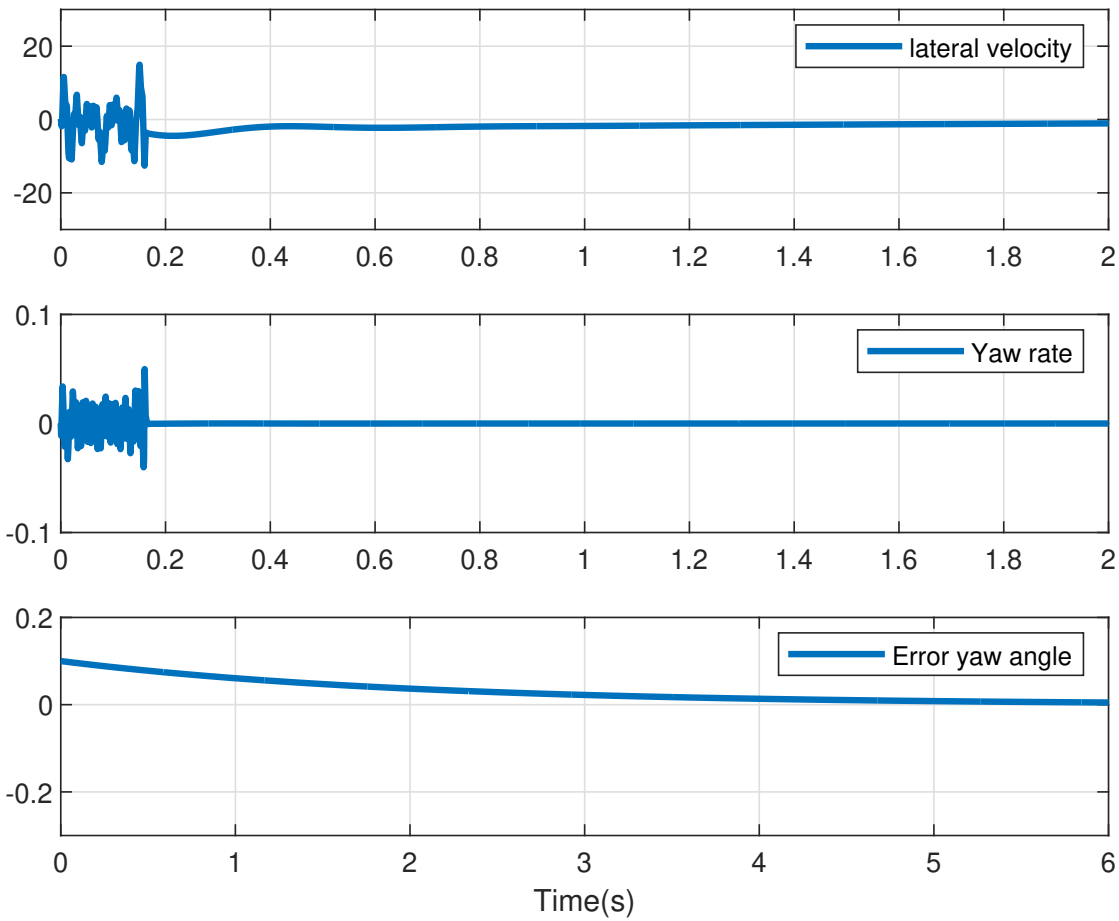
Lateral displacement



Lateral displacement zoomed



Other states



Conclusions

- Model free approach (data based)
- Optimality
- Stability
- Safety during operation—OK!
- Safety during EXPLORATION ???
- Initial admissible policy ?????

JHA Mayank S, Kiumarsi Bahare, Didier Theilliol, *Safe Reinforcement Learning based on Off-policy approach for Nonlinear Discrete-Time Systems, Submitted to 2024 American Control Conference.*

JHA, Mayank S, Bahare Kiumarsi and Didier Theilliol, "*Off-Policy Safe Reinforcement Learning for Nonlinear Discrete-Time Systems.*" *Under Revision, Neurocomputing.*

Thanks to contributors

- Dr. Bahare Kiumarsi, University of Michigan, USA.
- Prof. Didier Theilliol, Univ Lorraine, France.

JHA Mayank S, Kiumarsi Bahare, Didier Theilliol, *Safe Reinforcement Learning based on Off-policy approach for Nonlinear Discrete-Time Systems, Submitted to 2024 American Control Conference.*

JHA, Mayank S, Bahare Kiumarsi and Didier Theilliol, "*Off-Policy Safe Reinforcement Learning for Nonlinear Discrete-Time Systems.*" *Under Revision, Neurocomputing.*

Table of Contents

Example

Exploration Phase

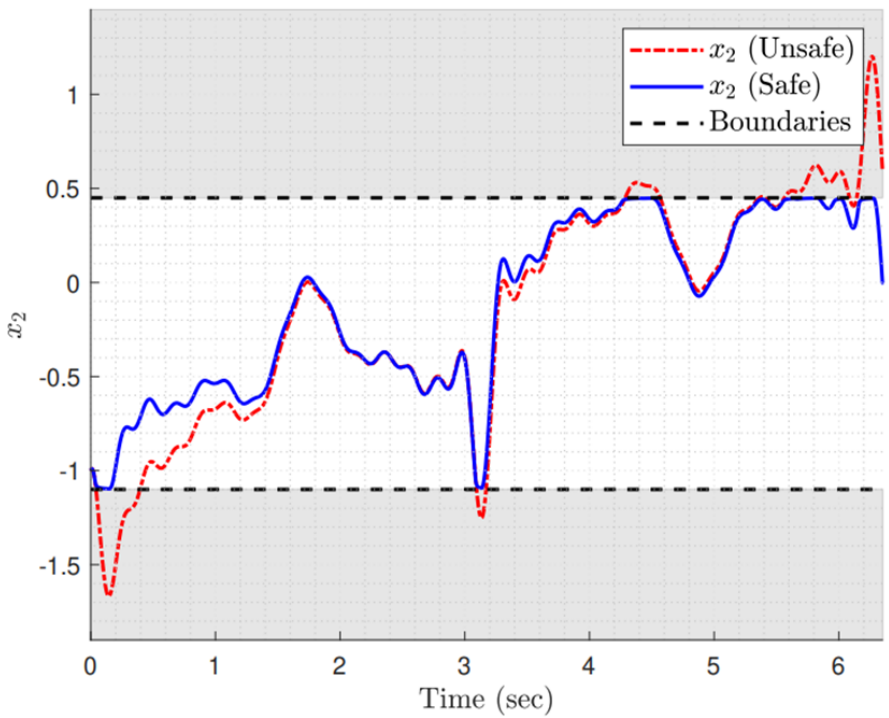


Fig.1 Trajectory of x_2 during exploration

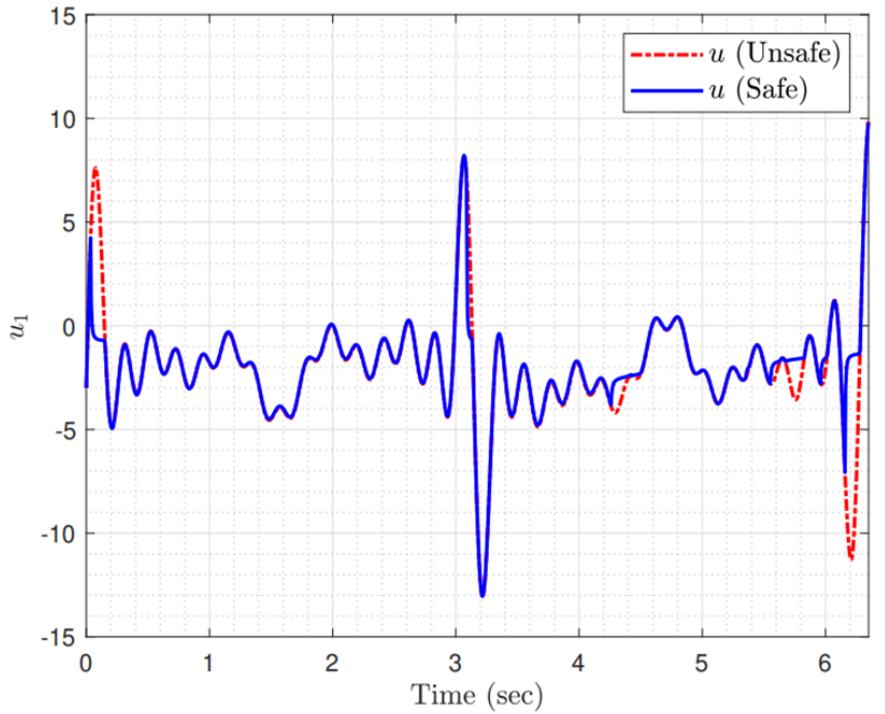
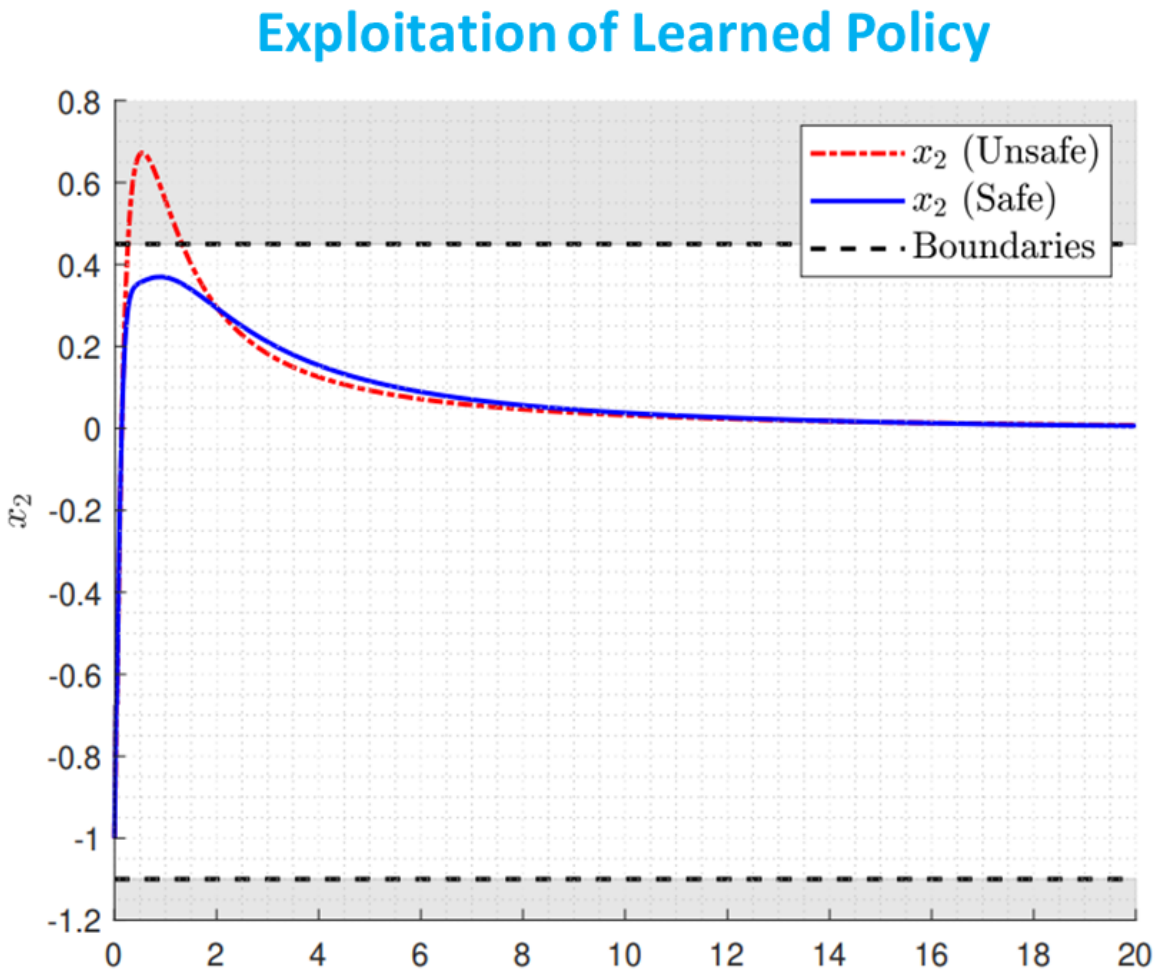


Fig.2 Exploration policy under probing noise

Example


Exploitation of Learned Policy
x_2 (Unsafe)
x_2 (Safe)
Boundaries
x_2
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1.2
0
2
4
6
8
10
12
14
16
18
20

Conclusions

- Optimality
- Stability
- Safety during operation—OK!
- Safety during EXPLORATION –OK!
- Initial admissible policy –OK!
- BUT, Model BAsed!

Continuous Time CONTINUES!

Principle Work by: Soha KANSO, 3rd Year PhD, CRAN
"Safe RL for Safety critical systems under degradation'

Kanso S, Jha MS, Theilliol D. **Off-Policy Model-Based End-to-End Safe Reinforcement Learning.** International Journal of Robust and Nonlinear Control, Under second revision.

References I


Agrawal, A., & Sreenath, K. (2017).Discrete Control Barrier Functions for Safety-Critical Control of Discrete Systems with Application to Bipedal Robot Navigation.. *Robotics: Science and Systems*, *13*.


Ames, A. D., Xu, X., Grizzle, J. W., & Tabuada, P. (2016).Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, *62*(8), 3861–3876.


Bellman, R. (1957).A markovian decision process. *Journal of mathematics and mechanics*, 679–684.

References II


Brunke, L., Greeff, M., Hall, A. W., Yuan, Z., Zhou, S., Panerati, J., & Schoellig, A. P. (2022).Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, *5*, 411–444.


Howard, R. A. (1960).Dynamic programming and markov processes..


Leake, R., & Liu, R.-W. (1967).Construction of suboptimal control sequences. *SIAM Journal on Control*, *5*(1), 54–63.


Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

References III

Wabersich, K. P., Taylor, A. J., Choi, J. J., Sreenath, K., Tomlin, C. J., Ames, A. D., & Zeilinger, M. N. (2023).Data-driven safety filters: Hamilton-jacobi reachability, control barrier functions, and predictive methods for uncertain systems. *IEEE Control Systems Magazine*, *43*(5), 137–177.