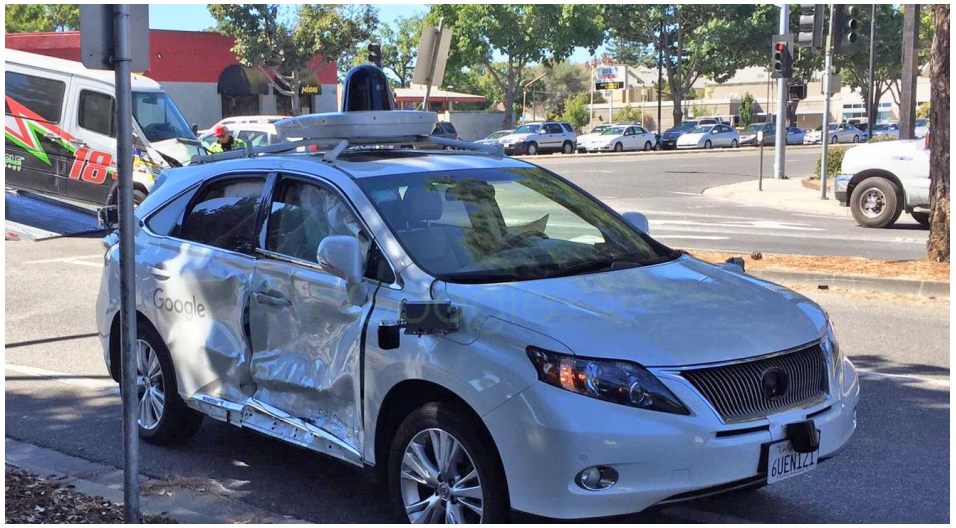
Safe Reinforcement Learning through Barrier Functions for Safety-Critical Control Tasks

Richard Cheng, Gabor Orosz, Richard Murray, Joel W. Burdick

Caltech

Motivation

Machine learning is promising to enable new technologies ranging from autonomous cars to robot manipulation



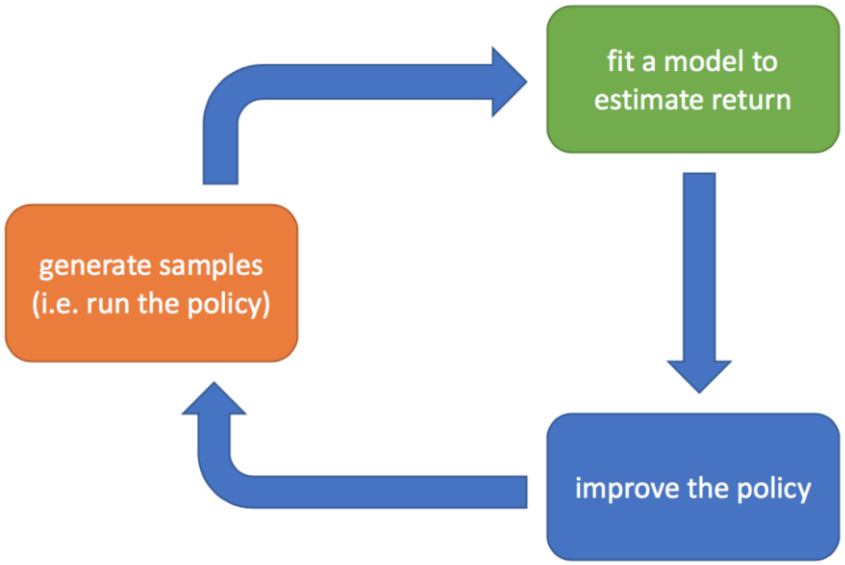
Safety is critical

We must ensure that robots either remain away from obstacles or within a workspace, even when it is still learning

In this work, we introduce model information (through barrier functions) into the reinforcement learning framework to **guarantee safety** during learning

Model-Free Reinforcement Learning

Reinforcement learning (RL) learns an optimal policy through interactions with the environment:



$$\pi^* = \max_{\pi} J(\pi) = \max_{\pi} \mathbb{E}_{\tau \sim \pi}[\gamma^t \, r(s_t, a_t)]$$

Policy gradient-based optimization with no prior information:

$$\nabla_\theta J(\theta) = \int \nabla_\theta \pi_\theta(\tau) r(\tau) d\tau$$

$$= \mathbb{E}_{\tau \sim \pi_\theta} \left[\nabla_\theta \log \pi_\theta(\tau) Q^\pi(\tau) \right]$$

$$\tau: (s_t, a_t, \ldots, s_{t+N}, a_{t+N})$$

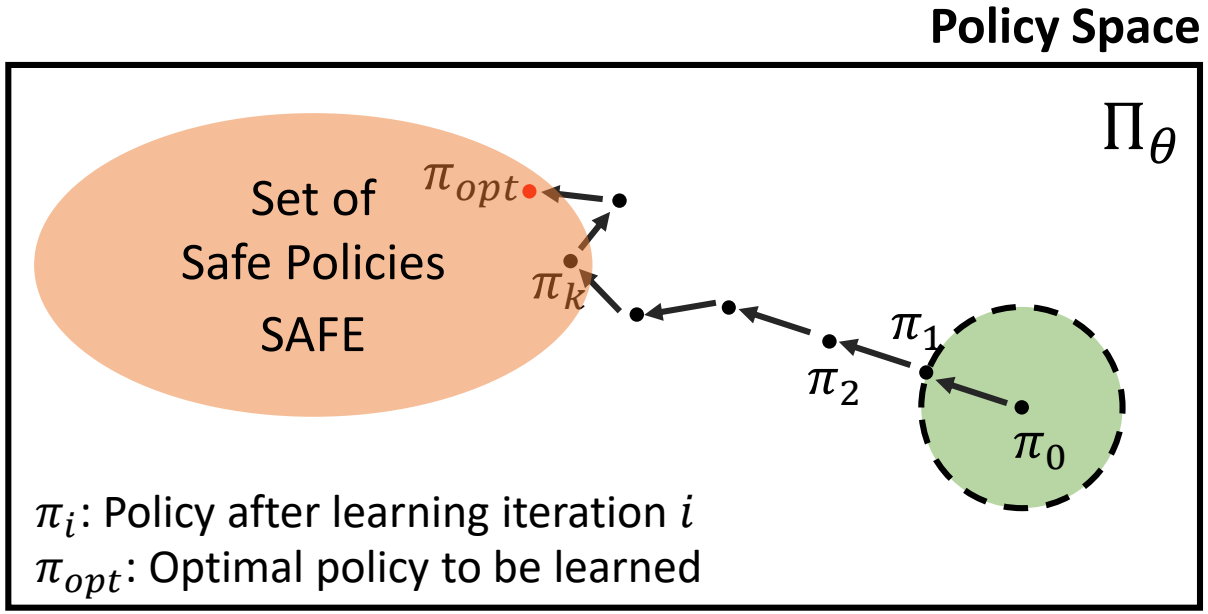
Silver et al. (2014)
Lillicrap et al. (2015)
Schulman et al. (2015)

Allows us to optimize policy with no model information (only sampled trajectories from interactions)

Safety during Policy Search

Throwing away any model information we have – learning from scratch…

- Slow Learning, high variance in policies
- No guarantees on system safety! Usually no replay in real world.



$$\pi^* = \max_{\pi} \mathbb{E}_{\tau\sim\pi}[\gamma^t\, R(s_t, a_t, s_{t+1})] \qquad \text{s.t.} \qquad \pi \in SAFE$$

Defining Safety of Policy

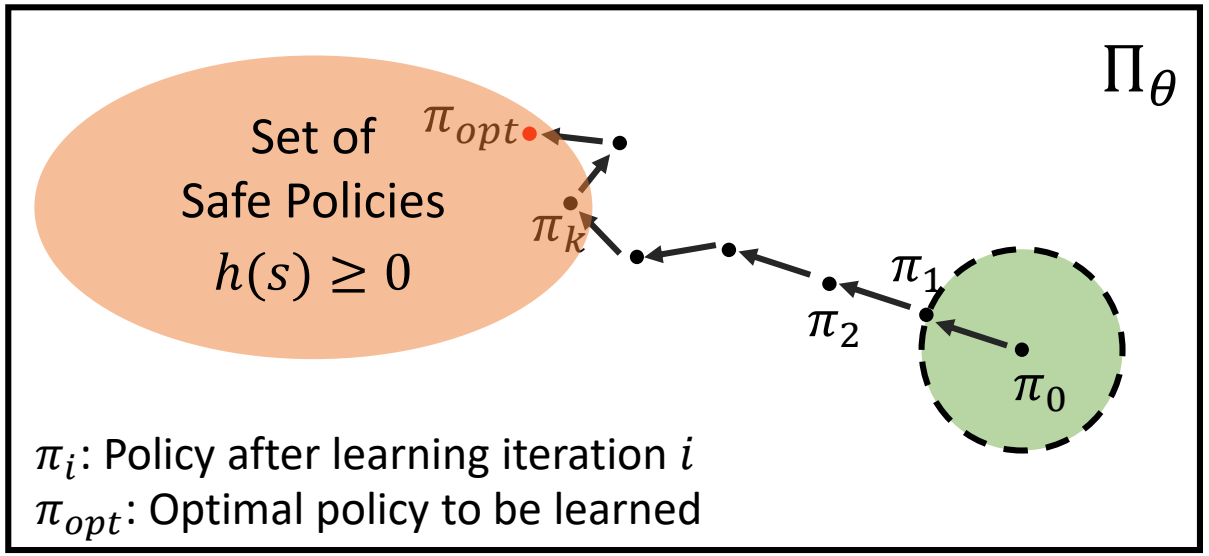
Consider an arbitrary safe set, $\mathcal{C}$, defined by the super-level set of a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$,

$$\mathcal{C} : \{s \in \mathbb{R}^n : h(s) \geq 0\}.$$

To maintain safety during the learning process, the system state must remain within the safe set $\mathcal{C}$

We want to restrict policy search to policies π such that for all $s_t \in C$, the next state $s_{t+1} \in C$ under policy π

Policy Space



$$\pi^* = \max_{\pi} \mathbb{E}_{\tau \sim \pi}[\gamma^t \, R(s_t, a_t, s_{t+1})] \qquad \text{s.t.} \qquad h(s_t) \geq 0 \;\; \forall t$$

Enforcing Safety through Barrier Functions

Assume control-affine dynamics:

$$s_{t+1} = f(s_t) + g(s_t)a_t + d(s_t)$$

d is unknown, but either (1) bounded, or (2) described by a mean/variance.

Given the safe set C defined previously, the continuously differentiable function h is a discrete-time control barrier function (CBF) for the dynamical system if there exists $\eta \in [0,1]$ such that for all $s_t \in C$,

$$\sup_{a_t \in A} \left[h\Big(f(s_t) + g(s_t)a_t + d(s_t) \Big) + (\eta - 1)h(s_t) \right] \geq 0.$$

If we can find an a_t that satisfies this safety condition for all $s \in C$, then we are guaranteed a safe controller.

Gaussian Process model parameterizes $d(s)$ with mean $\mu_d(s)$ and variance $\sigma_d(s)$

- Bounds model error with given probability

$$\sup_{a_t \in A} \left[h\big(f(s_t) + g(s_t)a_t + \mu_d(s_t) \pm k_\delta \sigma_d(s_t) \big) + (\eta - 1)h(s_t) \right] \geq 0$$

If we can find action a_t that satisfies this condition, then safety certified with probability $1 - \delta$

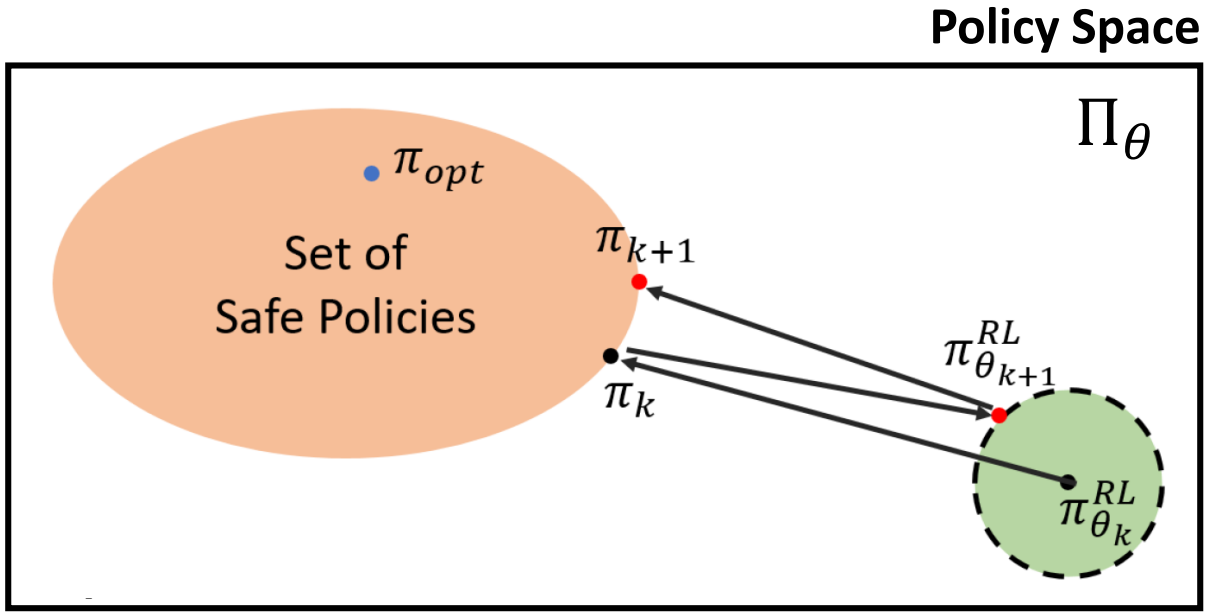
Agrawal and Sreenath (2017)
Rasmussen (2004)

Constraining the Policy Search

We now have a barrier function condition to ensure safety

- Lets use it to compensate for unsafe controllers proposed in learning

$$u_k(s) = u_{\theta_k}^{RL}(s) + u_k^{CBF}(s, u_{\theta_k}^{RL})$$



Define safe set by set of affine barrier functions:

$$h(s) = p^T s + q$$

$$\begin{aligned}(a_t, \epsilon) = \ & \underset{a_t, \epsilon}{\text{argmin}} \ \|a_t\|_2 + K_\epsilon \epsilon \\ \text{s.t.} \ & p^T f(s_t) + p^T g(s_t)\left(u_{\theta_k}^{RL}(s_t) + a_t\right) + p^T \mu_d(s_t) \\ & - k_\delta |p|^T \sigma_d(s_t) + q \geq (1-\eta) h(s_t) - \epsilon \\ & a_{low}^i \leq a_t^i + u_{\theta_k}^{RL(i)}(s_t) \leq a_{high}^i \text{ for } i = 1, \ldots, M\end{aligned}$$

What if actuation limitations prevent us from satisfying the safety condition for all $s \in C$?

Graceful Degradation of Safety

Assume safety condition cannot be satisfied for all $s \in C$

Potential safety violation of ε

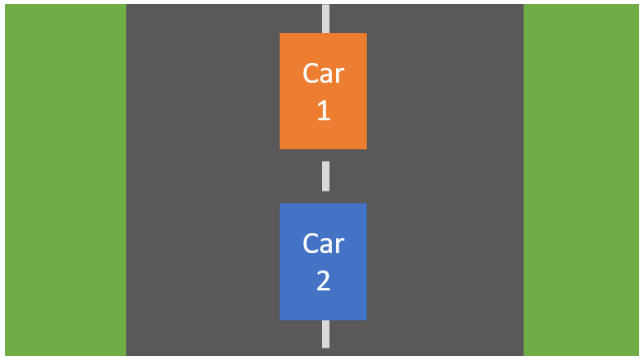
$$\sup_{a_t \in A} \left[h\left(f(s_t) + g(s_t)a_t + \mu_d(s_t) \pm k_\delta \sigma_d(s_t) \right) + (\eta - 1)h(s_t) \right] = -\varepsilon(s_t)$$

$$C : \{ s \in \mathbb{R}^n : h(s) \geq 0 \}$$

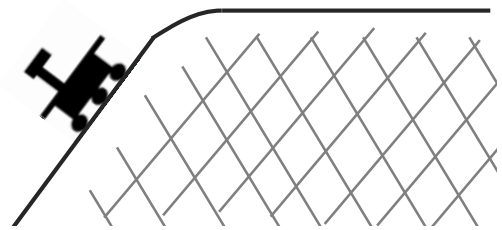
$$C_\varepsilon : \{ s \in \mathbb{R}^n : h(s) \geq -\varepsilon^{\max} \}$$

Suppose that for all $s \in C$, the safety condition above is satisfied with $\varepsilon(s) \leq \varepsilon^{\max}$. If for all $s \in C_\varepsilon$, the safety condition is satisfied with $\varepsilon(s) \leq \varepsilon^{\max}$, then the larger set C_ε is forward invariant with probability $(1 - \delta)$.

Scenario 1:



Scenario 2:

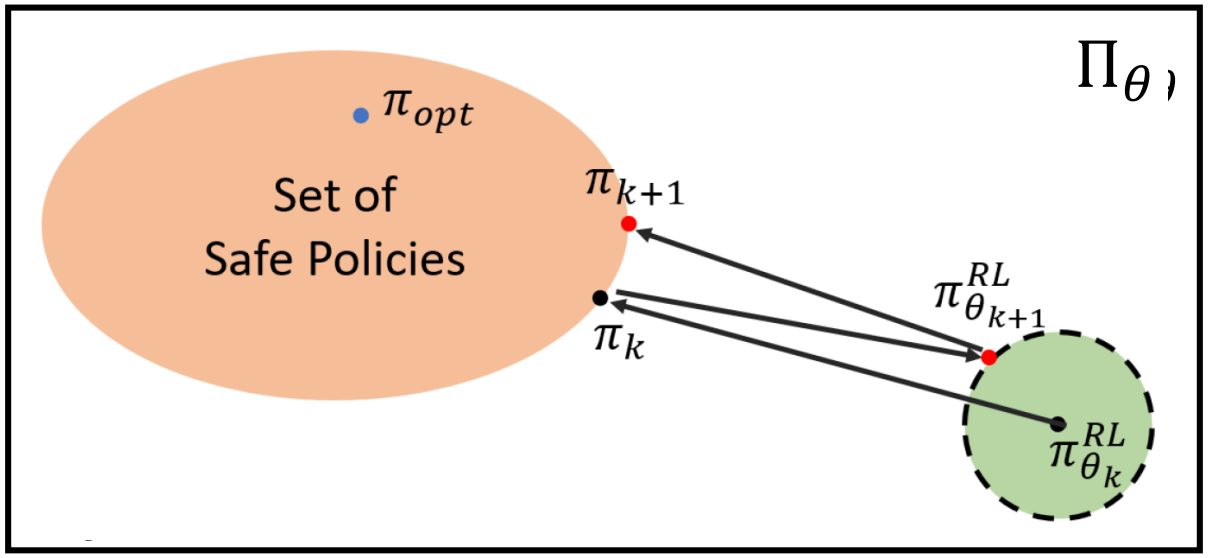


Best solution: Define maximal safe set such that safety condition can be always satisfied

Next best solution: Ensure that we stay as close as possible to defined safety set

Constraining the Policy Search

Anytime RL controller proposes an unsafe action, project that control action to the “closest” safe action

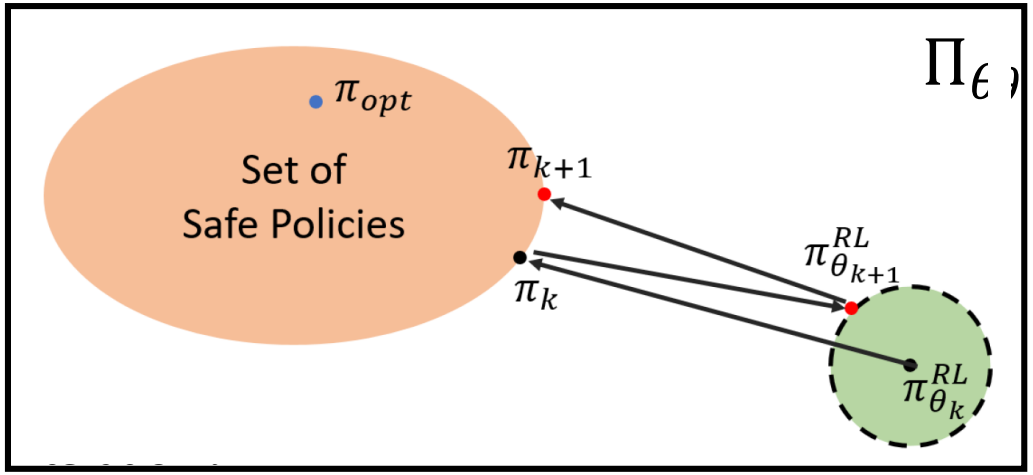


Quite inefficient, especially if the RL policy continues to “search” near unsafe policies

- Adding a compensatory mechanism can distort the learning process

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\nabla_\theta \log \pi_\theta(\tau) Q^\pi(\tau)\right]$$

Guiding the Policy Search with Barrier Functions

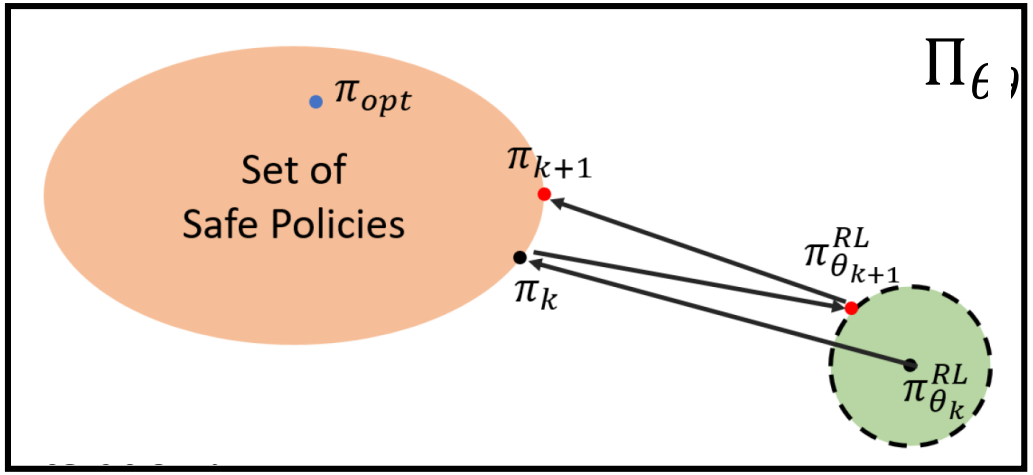


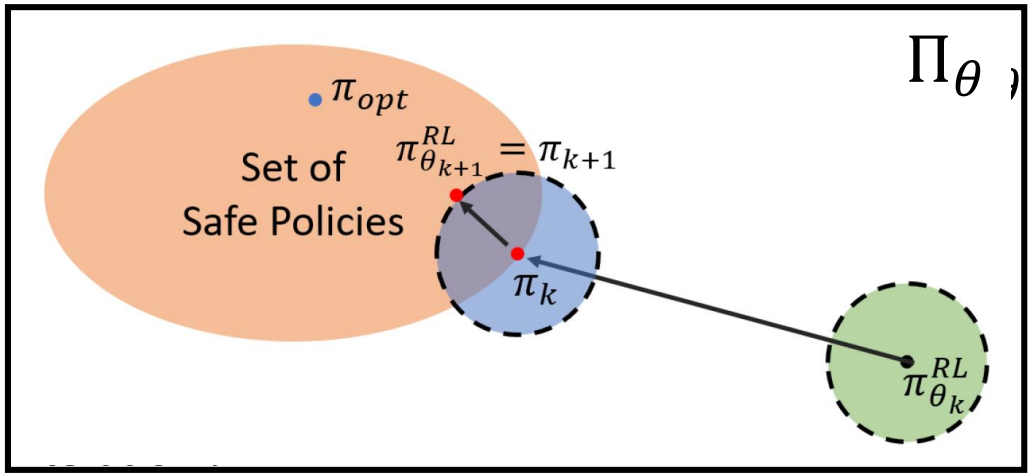
$$u_k(s) = u^{RL}_{\theta_k}(s) + u^{CBF}_k(s, u^{RL}_{\theta_k}).$$

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\nabla_\theta \log \pi_\theta(\tau) Q^\pi(\tau)\right]$$

RL algorithm learns the association, (s_t, u^{RL}_t, r_t) , when the association is really, $(s_t, u^{RL}_t + u^{CBF}_t, r_t)$

Guiding the Policy Search with Barrier Functions





$$u_k(s) = \overbrace{u_{\theta_k}^{RL}(s) + u_k^{CBF}(s, u_{\theta_k}^{RL})}^{u^{RL}}.$$

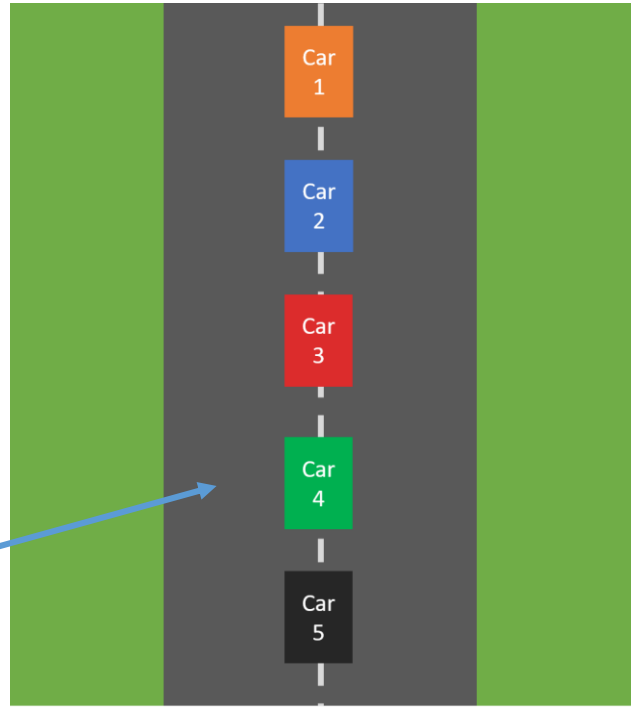
$$u_k(s) = \overbrace{u_{\theta_k}^{RL}(s) + \sum_{j=0}^{k-1} u_j^{CBF}(s, u_{\theta_0}^{RL}, ..., u_{\theta_{j-1}}^{RL}) + u_k^{CBF}(s, u_{\theta_k}^{RL})}^{u^{RL}}.$$

Incorporate compensatory barrier functions from previous policy iterations

Guarantee safety and guide policy search towards the "safe" region

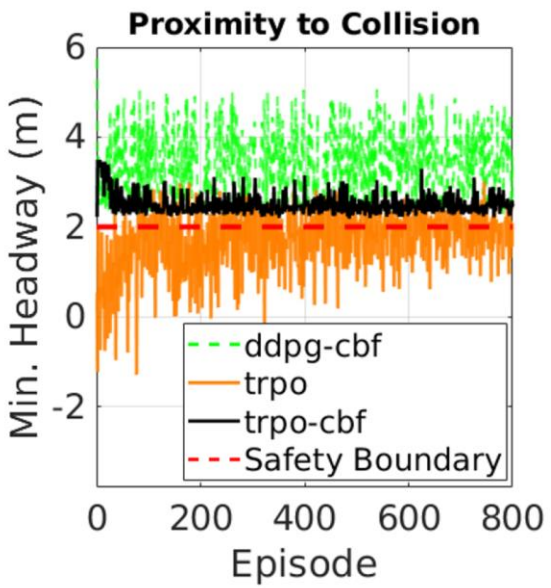
Simulations

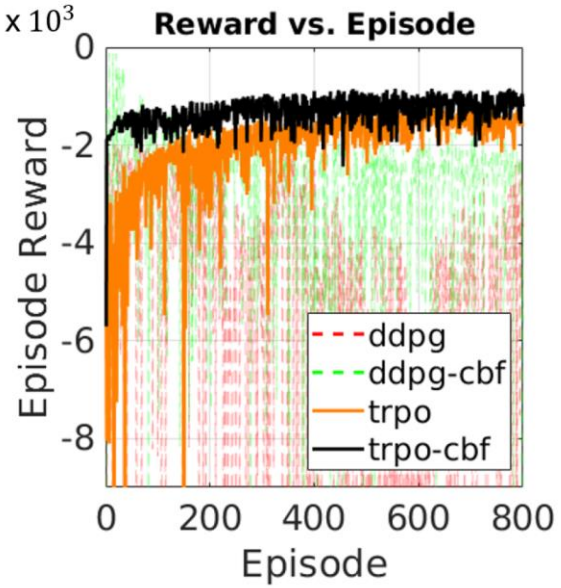
1-D Car Following



Control this car

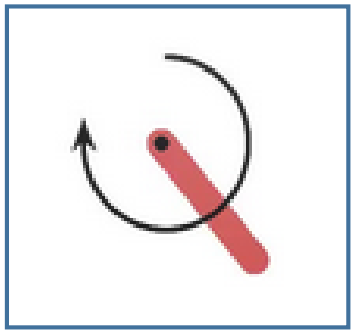
Try to learn an optimal (max fuel efficiency) controller for acceleration of car 4, while maintaining minimum headway from other cars.



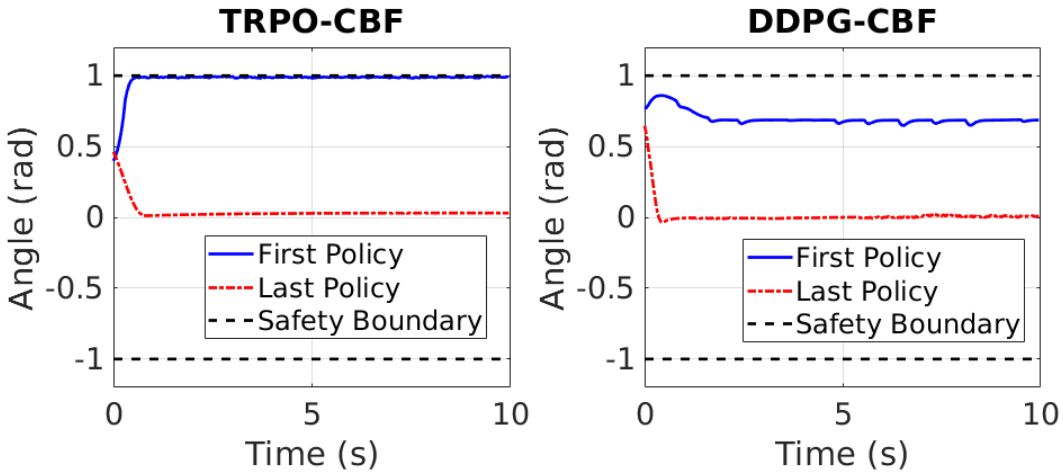


Simulations

Inverted Pendulum



Try to swing/maintain the pendulum upright, with small penalty on actuation. Safety region is between ± 1 rad from the upright position



No safety violations. Obviously this application is contrived, since often the optimal solution is to swing *through* the unsafe region. This method would *not* be good for these types of applications.

Conclusion

- Reinforcement learning can be a powerful tool for controller synthesis
- Current RL techniques are slow at learning and do not include safety *constraints*
- Using barrier functions, we can constrain the system to safe behaviors while learning, and improve learning efficiency

<u>Challenges/Next Steps</u>

- How can we learn/expand a valid safe set $h(s)$ online?
- How can we further leverage crude model information to improve learning of controllers?
  - Lot of recent interest in model-based RL, but no clear view on how to best incorporate this model information