Safe Reinforcement Learning for Autonomous Vehicles through Parallel Constrained Policy Optimization

William Dawkins,
3/17/22

AUTHORED BY: LU WEN, JINLIANG DUAN, SHENGBO EBEN LI, SHAOBING XU, AND HUEI PENG

Content

Introduction

- This work applies Reinforcement Learning (RL) to autonomous vehicles
- RL algorithms applied to real vehicles have safety concerns
- This paper presents a new safe RL algorithm, Parallel Constrained Policy Optimization (PCPO)

Background - Problem

- Autonomous driving has two categories: rule based or learning based
- Rule based methods are limited by difficulty to account for all situations
- Learning based can imitate and learn driving habits implicitly
- This work seeks to develop an improved learning based method

Background - Problem

- Previous work has applied RL to autonomous driving
- Predominately developed on simulation platforms due to safety concerns
- Back propagation driven process may lead to unforeseen accidents
- Safety is the most basic requirement for autonomous driving

Background – Safe RL

- Safe RL: *"Process of learning policies that maximizes the expectation of accumulated rewards, while respecting security constraints in the learning and deployment process"*

- General safe RL approaches: 1) modifying optimization criterion, 2) modifying exploration process [1]

- The purpose of this work is to introduce a new safe RL algorithm, Parallel Constraint Policy optimization applied to real autonomous vehicles

[1] J. Garcıa and F. Fernández, "A comprehensive survey on safe reinforcement learning," *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 1437–1480, 2015.

Parallel Constrained Policy Optimization (PCPO) Methodology – Preliminaries

- Problem is formalized as MDP with $(S, A, r, P, \rho_0, \gamma)$
- Define Value and Q functions: $V^{\pi}(s) = E_{\pi}[R_t | s_t = s]$, $Q^{\pi}(s, a) = E_{\pi}[R_t | s_t = s, a_t = a]$
- Wish to find policy that maximizes objective function: $\eta(\pi) = E_{\tau,\pi}[\sum_{t=0}^{\infty} \gamma^t r(s_t)]$

PCPO Methodology – Preliminaries

$$\begin{aligned} A^\pi(s,a) &= Q^\pi(s,a) - V^\pi(s) \\ &= \mathbb{E}_{s'}\left[r(s) + \gamma V^\pi(s') - V^\pi(s)\right]. \end{aligned}$$

PCPO Methodology – Preliminaries

$$A^{\pi}(s,a) = Q^{\pi}(s,a) - V^{\pi}(s)$$
$$= \mathbb{E}_{s'}\left[r(s) + \gamma V^{\pi}(s') - V^{\pi}(s)\right].$$

$$\boxed{\eta(\pi) = \eta(\pi_{\text{old}}) + \mathbb{E}_{\tau,\pi}\left[\sum_{t=0}^{\infty}\gamma^t A^{\pi_{\text{old}}}(s_t, a_t)\right],}$$

PCPO Methodology – Preliminaries

$$A^{\pi}(s,a) = Q^{\pi}(s,a) - V^{\pi}(s) \\ = \mathbb{E}_{s'}\left[r(s) + \gamma V^{\pi}(s') - V^{\pi}(s)\right].$$

$$\eta(\pi) = \eta(\pi_{\text{old}}) + \mathbb{E}_{\tau,\pi}\left[\sum_{t=0}^{\infty} \gamma^t A^{\pi_{\text{old}}}(s_t, a_t)\right],$$

$$\begin{aligned}
&\eta(\pi) \\
&= \eta(\pi_{\text{old}}) + \sum_s \rho_\pi(s) \sum_a \pi(a|s) A^{\pi_{\text{old}}}(s,a) \\
&\approx \eta(\pi_{\text{old}}) + \sum_s \rho_{\pi_{\text{old}}}(s) \sum_a \pi(a|s) A^{\pi_{\text{old}}}(s,a) \\
&= \eta(\pi_{\text{old}}) + \mathbb{E}_{s,a\sim\pi_{\text{old}}}\left[\frac{\pi(a|s)}{\pi_{\text{old}}(a|s)}\left(Q^{\pi_{\text{old}}}(s,a) - V^{\pi_{\text{old}}}(s)\right)\right] \\
&= \eta(\pi_{\text{old}}) + \mathop{\mathbb{E}}_{s,a\sim\pi_{\text{old}}}\left[\frac{\pi(a|s)}{\pi_{\text{old}}(a|s)} Q^{\pi_{\text{old}}}(s,a)\right] - \mathop{\mathbb{E}}_{s\sim\pi_{\text{old}}}\left[V^{\pi_{\text{old}}}(s)\right].
\end{aligned}$$

PCPO Methodology – Preliminaries

$$\begin{aligned} A^\pi(s,a) &= Q^\pi(s,a) - V^\pi(s) \\ &= \mathbb{E}_{s'}\left[r(s) + \gamma V^\pi(s') - V^\pi(s)\right]. \end{aligned}$$

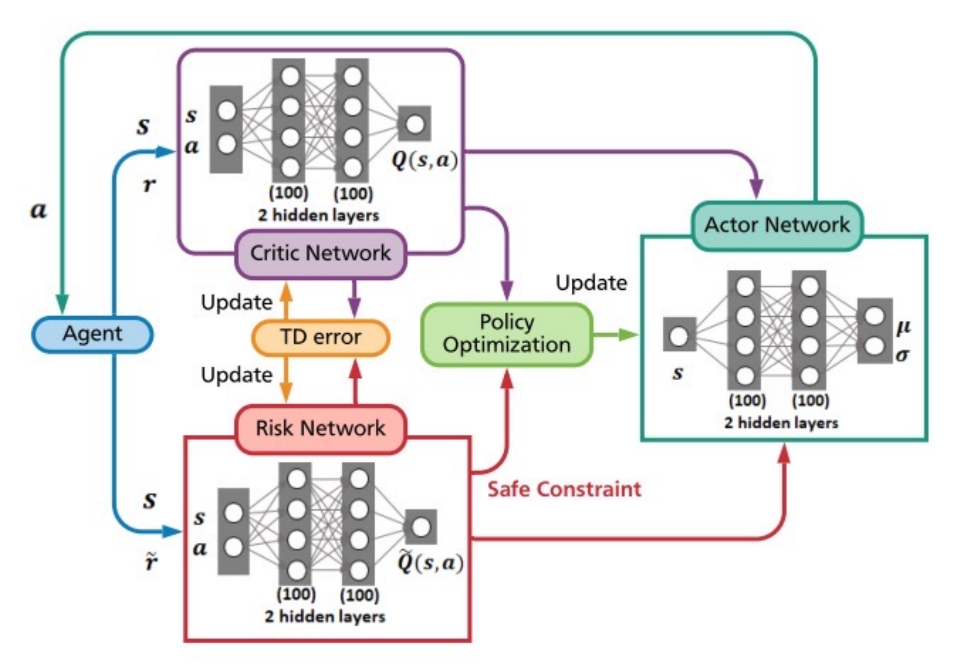
$$\eta(\pi) = \eta(\pi_{\text{old}}) + \mathbb{E}_{\tau,\pi}\left[\sum_{t=0}^{\infty}\gamma^t A^{\pi_{\text{old}}}(s_t,a_t)\right],$$

$$\begin{aligned} &\eta(\pi) \\ &= \eta(\pi_{\text{old}}) + \sum_s \rho_\pi(s) \sum_a \pi(a|s) A^{\pi_{\text{old}}}(s,a) \\ &\approx \eta(\pi_{\text{old}}) + \sum_s \rho_{\pi_{\text{old}}}(s) \sum_a \pi(a|s) A^{\pi_{\text{old}}}(s,a) \\ &= \eta(\pi_{\text{old}}) + \mathbb{E}_{s,a\sim\pi_{\text{old}}}\left[\frac{\pi(a|s)}{\pi_{\text{old}}(a|s)}\left(Q^{\pi_{\text{old}}}(s,a) - V^{\pi_{\text{old}}}(s)\right)\right] \\ &= \eta(\pi_{\text{old}}) + \mathop{\mathbb{E}}_{s,a\sim\pi_{\text{old}}}\left[\frac{\pi(a|s)}{\pi_{\text{old}}(a|s)} Q^{\pi_{\text{old}}}(s,a)\right] - \mathop{\mathbb{E}}_{s\sim\pi_{\text{old}}}\left[V^{\pi_{\text{old}}}(s)\right]. \end{aligned}$$

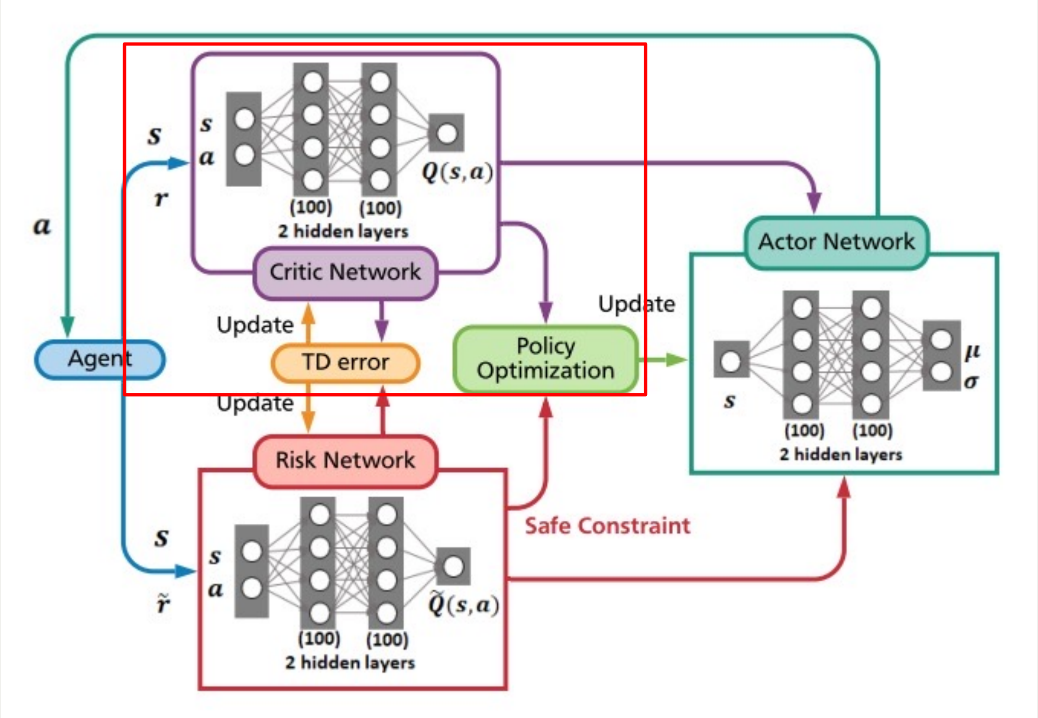
$$\boxed{J(\pi) = \mathbb{E}_{s,a\sim\pi_{\text{old}}}\left[\frac{\pi(a|s)}{\pi_{\text{old}}(a|s)} Q^{\pi_{\text{old}}}(s,a)\right].}$$

PCPO Methodology – Actor-Critic-Risk architecture

- PCPO utilizes so-called Actor-Critic-Risk architecture
- Similar to Actor-Critic methods, use neural networks to approximate policy (actor) and value (critic)
- Third NN approximates risk function, ensures safe policy



PCPO – Critic Network



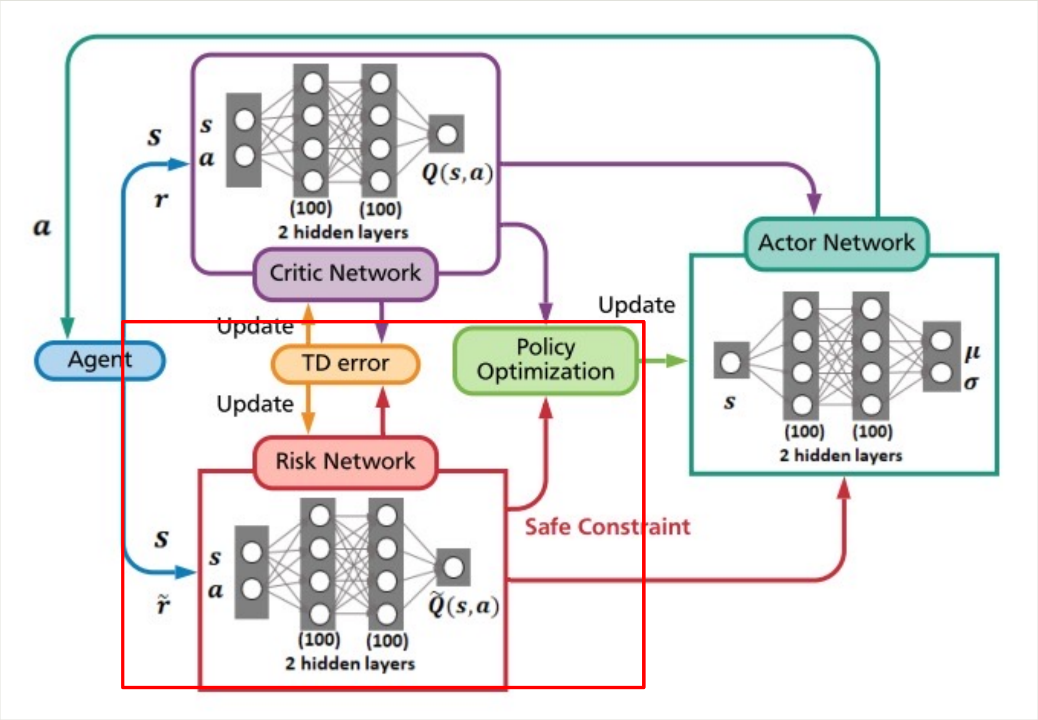
- Min. Temporal Difference (TD) squared:

$$L_{\text{critic}} = (R_t - Q^{\omega}(s_t, a_t))^2/2,$$

- Update parameters with gradient:

$$\mathrm{d}\omega = (R_t - Q^{\omega}(s_t, a_t))\nabla_{\omega} Q^{\omega}(s_t, a_t).$$

PCPO – Risk Network



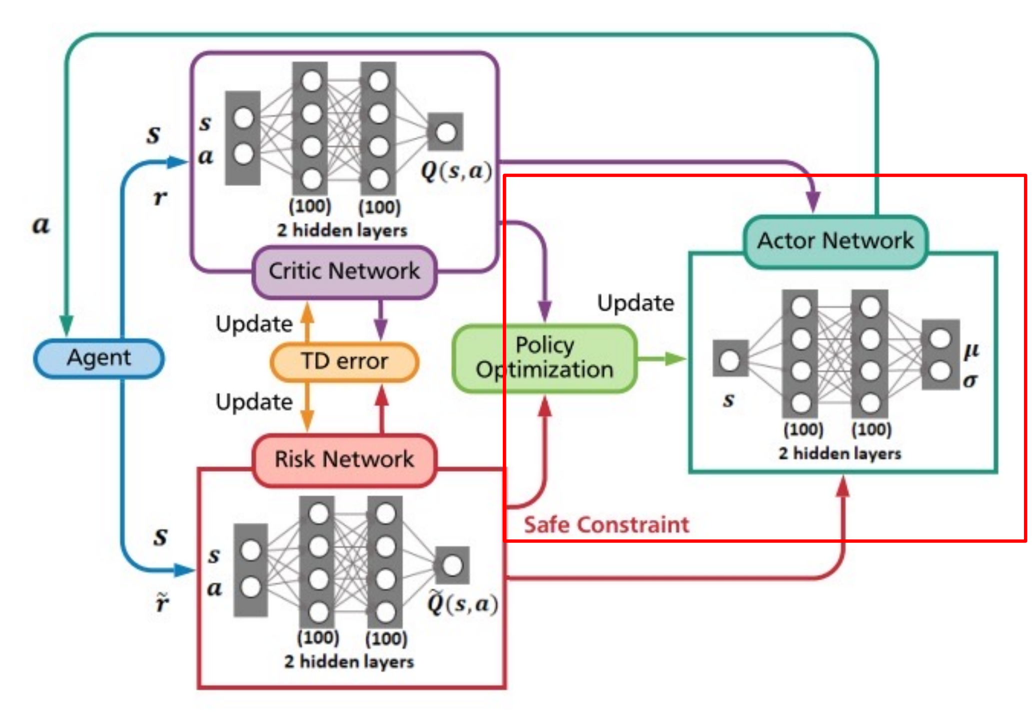
- Introduce risk signal $\tilde{r}$ observed at every step
- Define risk function analogous to Q function:

$$\tilde{Q}^{\pi}(s,a) = \mathbb{E}_{\pi}\left[\tilde{R}_t | s_t = s, a_t = a\right]$$

- Update risk network via TD:

$$\mathrm{d}\phi = (\widetilde{R}_t - \widetilde{Q}^{\phi}(s_t,a_t))\nabla_{\phi}\widetilde{Q}^{\phi}(s_t,a_t)$$

PCPO – Actor Network



- Update Actor Network by gradient of $J(\pi)$:

$$\mathrm{d}\boldsymbol{\theta} = \nabla_{\boldsymbol{\theta}} J(\pi^{\boldsymbol{\theta}}) = \nabla_{\boldsymbol{\theta}} \mathbb{E}_{s,a\sim\pi^{\boldsymbol{\theta}_{\text{old}}}} \left[\frac{\pi^{\boldsymbol{\theta}}(a|s)}{\pi^{\boldsymbol{\theta}_{\text{old}}}(a|s)} Q^{\pi^{\boldsymbol{\theta}_{\text{old}}}}(s,a) \right].$$

- Inspired by [1], define objective function wrt. Risk function:

$$\tilde{J}(\pi) = \mathbb{E}_{s,a\sim\pi_{\text{old}}} \left[\frac{\pi(a|s)}{\pi_{\text{old}}(a|s)} \tilde{Q}^{\pi_{\text{old}}}(s,a) \right].$$

- Add policy security constraint:

$$\tilde{J}(\pi) \leq \delta.$$

- This method is called Constrained Policy Optimization (CPO)

[1]: J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," *arXiv preprint arXiv:1705.10528*, 2017.

PCPO – Trust Region Constraint

- Since the risk and reward functions are approximated by NN monotonic improvements can only be guaranteed for small policy changes
- Add a policy constraint: $\mathbb{E}_{s\sim\pi^{\boldsymbol{\theta}_{\text{old}}}}\left[D_{\text{KL}}(\pi^{\boldsymbol{\theta}}(s),\pi^{\boldsymbol{\theta}_{\text{old}}}(s))\right]\leq\delta,$
- Total optimization problem:

$$\begin{aligned}\boldsymbol{\theta}^{k+1} &= \arg\max_{\boldsymbol{\theta}} J(\pi^{\boldsymbol{\theta}})\\ \textbf{s.t.}\quad & \widetilde{J}(\pi^{\boldsymbol{\theta}})\leq d\\ & \mathbb{E}_{s\sim\pi^{\boldsymbol{\theta}_{\text{old}}}}\left[D_{\text{KL}}(\pi^{\boldsymbol{\theta}}(s),\pi^{\boldsymbol{\theta}_{\text{old}}}(s))\right]\leq\delta.\end{aligned}\tag{2}$$

PCPO – Linear Approximation

- The optimization problem is non-linear and difficult to solve, but can be approximated around θ^k:

$$\begin{aligned}\boldsymbol{\theta}^{k+1} &= \arg\max_{\boldsymbol{\theta}} g^T(\boldsymbol{\theta} - \boldsymbol{\theta}^k) \\ \textbf{s.t.} \quad & c + b^T(\boldsymbol{\theta} - \boldsymbol{\theta}^k) \le 0 \\ & \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}^k)^T H(\boldsymbol{\theta} - \boldsymbol{\theta}^k) \le \delta,\end{aligned} \tag{4}$$

- g is the gradient of $J(\pi^k)$, b is the gradient of $\tilde{J}(\pi^k)$, H is the Hessian of the KL divergence and $c \ := \tilde{J}(\pi^k) - d$
- This can be solved with Lagrange multipliers, λ and ν, yielding the update rule:

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k + \frac{1}{\lambda^*} H^{-1}(g - b\nu^*). \tag{6}$$

PCPO – Infeasible Solutions

- It is possible to be unable to find a feasible solution to (4)
- Occurs when the risk function is very high due to being in unsafe state, or a bad update that produces an unsafe action due to approximation errors in (4)
- Previous work [1] with CPO dealt with bad updates with a recovery rule:

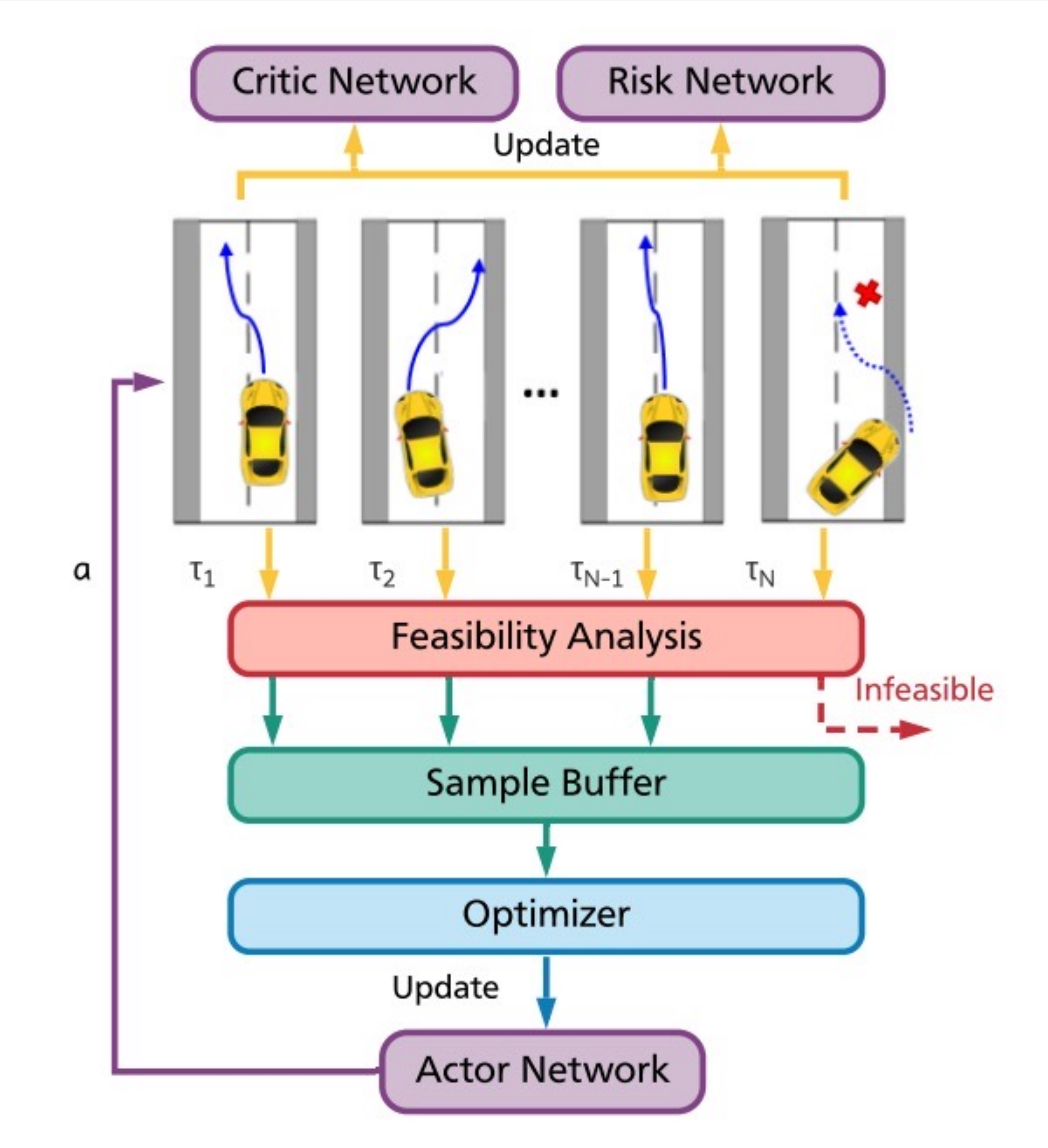
$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^{k} - \sqrt{\frac{2\delta}{b^T H^{-1} b}} H^{-1} b. \quad (7)$$

- This does not help the case where the risk function is high because π^{θ^k} may work well in safe states, if so the recovery rule leads to slower convergence

[1] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," *arXiv preprint arXiv:1705.10528*, 2017.

PCPO – Parallel Learners

- To deal with this issue, this work introduces parallel learners
- Each learner generates samples synchronously, τ_i
- All samples used to update value and risk networks
- Only feasible samples used to update policy network
- Increases convergence speed
- Combining parallel learners with CPO is the final PCPO algorithm



PCPO- Algorithm

Algorithm 1 Parallel Constrained Policy Optimization

Initialization:

Initial with arbitrary $\boldsymbol{\theta}$, $\boldsymbol{\omega}$ and $\boldsymbol{\phi}$ and state $s_0 \in S$

Iteration:

for $k = 1, 2, \ldots, n$ **do**

Explore samples set $\tau = \{s\} \sim \pi(\boldsymbol{\theta}^k)$

Update the Value Network with d$\boldsymbol{\omega}$ in (1)

Update Risk Network with:

$$\mathrm{d}\boldsymbol{\phi} = (\widetilde{R}_t - \widetilde{Q}^{\phi}(s_t, a_t))\nabla_{\boldsymbol{\phi}}\widetilde{Q}^{\phi}(s_t, a_t)$$

Estimate g, b, H, c in (4) with τ

Store feasible τ in buffer D

end for

if $D \neq \varnothing$ **then**

Solve (5) for λ^*, ν^*

Update policy network using (6)

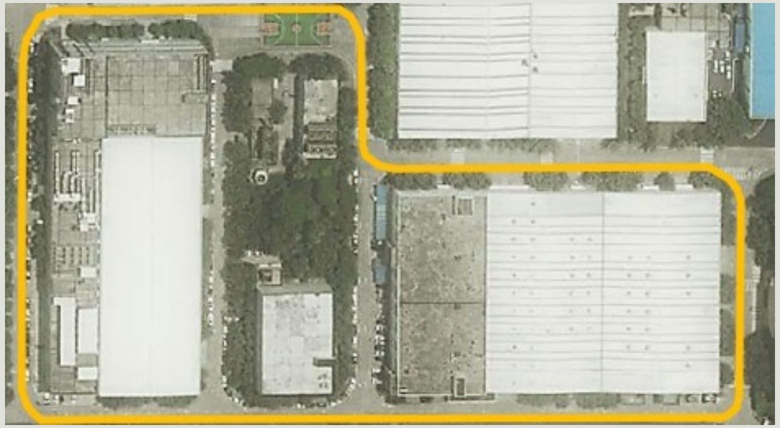
else

Recovery policy using (7)

end if

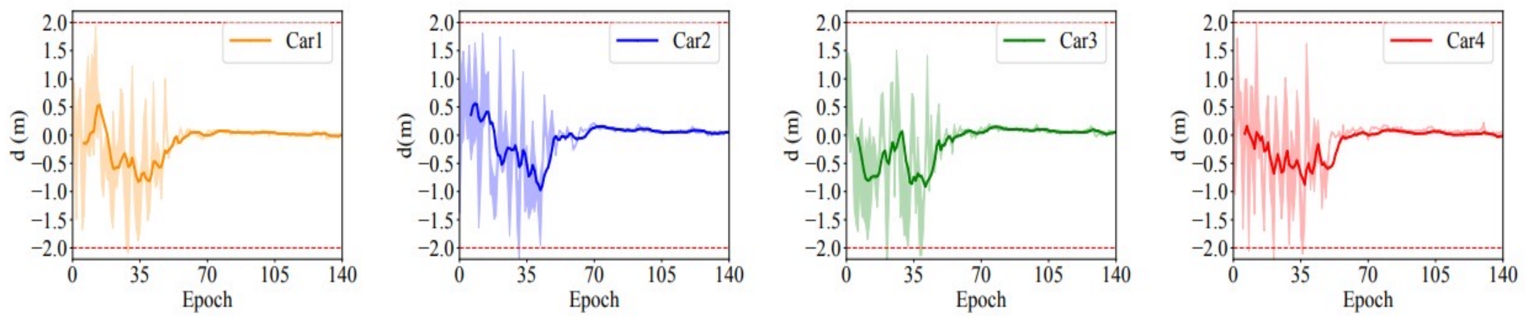
Experiment 1 – Lane Keeping

- Goal: Keep car as close to center of lane as possible while not deviating from road throughout learning process
- State space: $S = \{d[m], \beta[rad]\}$, distance from center line, angle between vehicles heading angle and direction of current trajectory
- Action space: $A = \{\delta[rad]\}$, referring to the front wheel angle
- Define reward function: $r = -\frac{100}{9}d^2 - \beta^2$, risk of 100 if car leaves lane

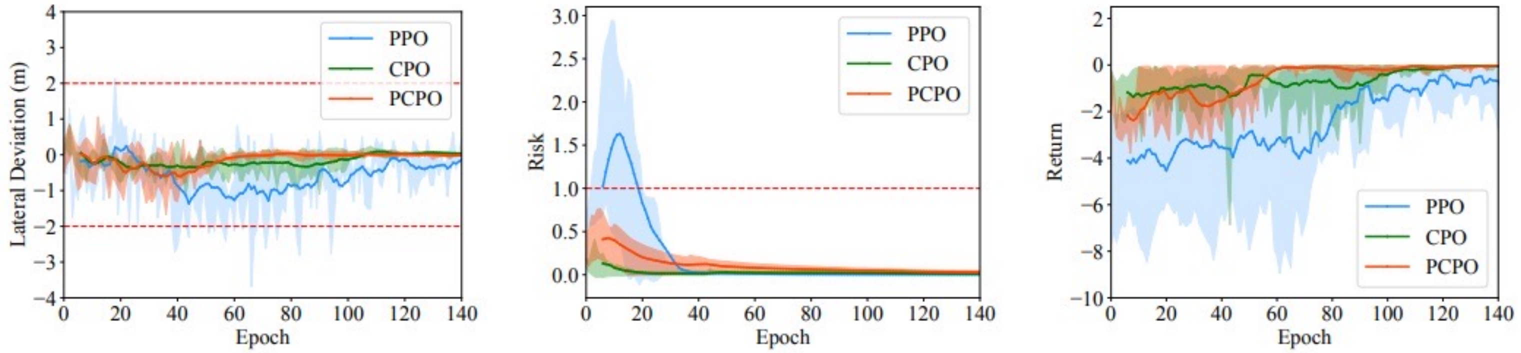


Experiment 1 – Lane Keeping

- PCPO used 4 parallel learners
- Compare PCPO to parallel policy optimization (PPO) and constrained policy optimization (CPO)
- The safety constraint is set to 1 and the trust region constraint is set to 10^{-3}
- First figure shows average lateral deviation of 4 learners over 5 runs
- Second figure shows training performances of all three algorithms



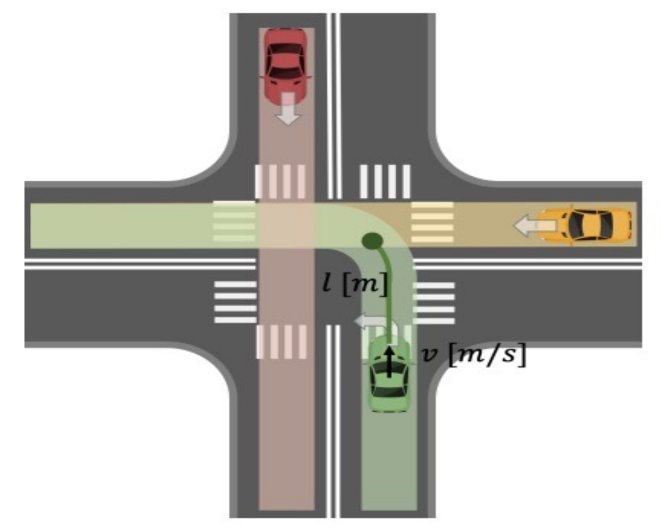
(a) Car-learner 1 (b) Car-learner 2 (c) Car-learner 3 (d) Car-learner 4



(a) Lateral Deviation Comparison (b) Risk Comparison (c) Return Comparison

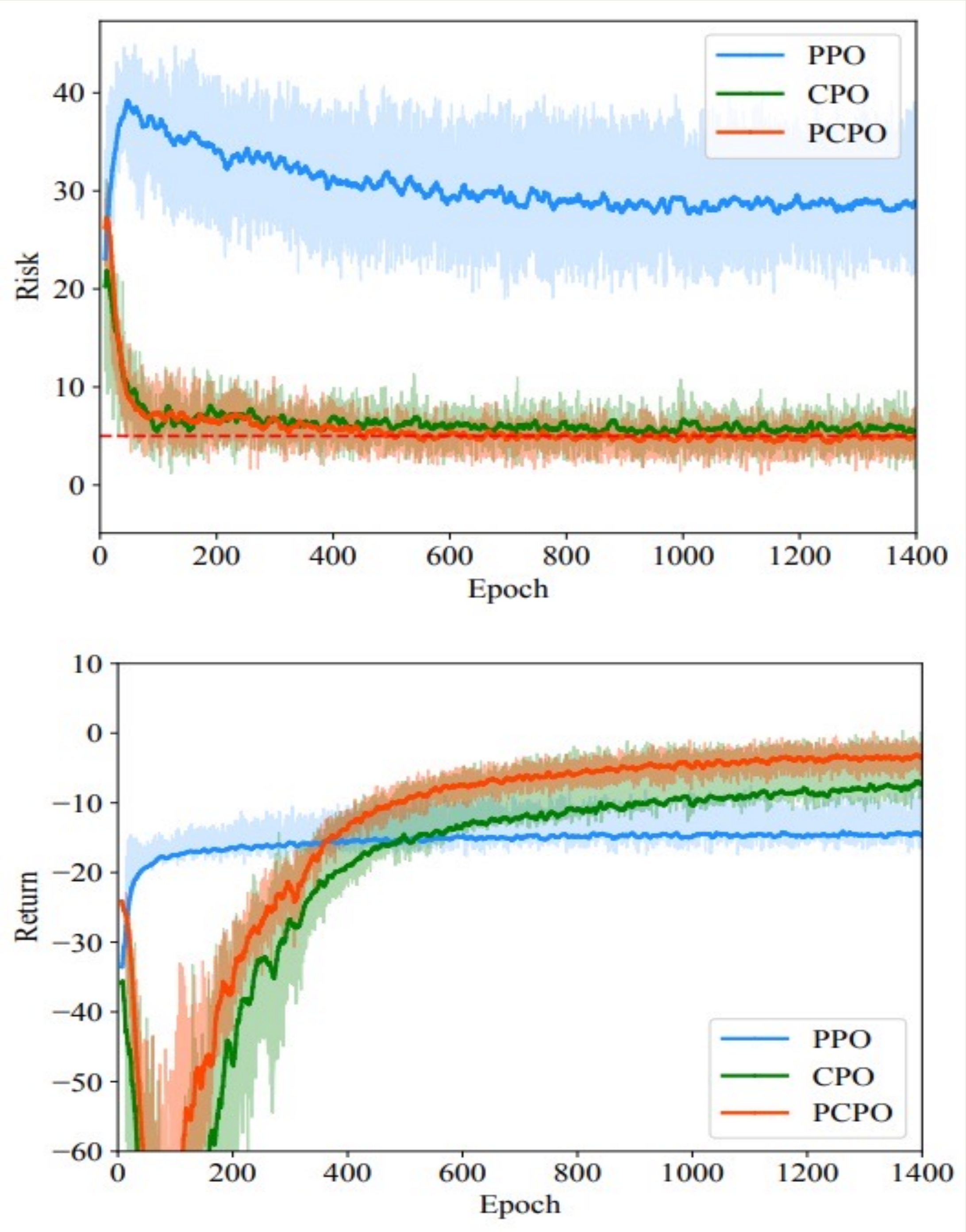
Experiment 2 – Intersection decision-making

- Goal: Three cars approach unsignalized intersection, randomly assign velocity and position along each track, learn policy for all vehicles to pass through as fast as possible with no collisions
- State: $S = \{l_1, v_1, l_2, v_2, l_3, v_3\}$, positions of vehicles from middle of their track and velocities
- Action space: $A = \{a_1, a_2, a_3\}$, accelerations of each vehicle where $a \in [-3,3]$
- Reward: +10 for each passing vehicle, -1 every time step, +10 for terminal success, risk +50 for collision



Experiment 2 – Intersection Decisioning

- Safe limit set a 5 and trust region constraint set to 10^{-3}
- Again compare PPO and CPO to PCPO
- Top figure is Risk over learning process
- Bottom figure is return over learning process



Conclusions

- This work presents a new Safe RL algorithm, PCPO, for automated driving tasks
- PCPO uses actor-critic-risk architecture with newly introduced risk function
- Introduced parallel learning
- Through experiments have shown:
  - PCPO guarantees safety constraints during learning for general autonomous driving tasks
  - Improved learning speed
  - Prevents learners being stuck at a sub-optimal policy