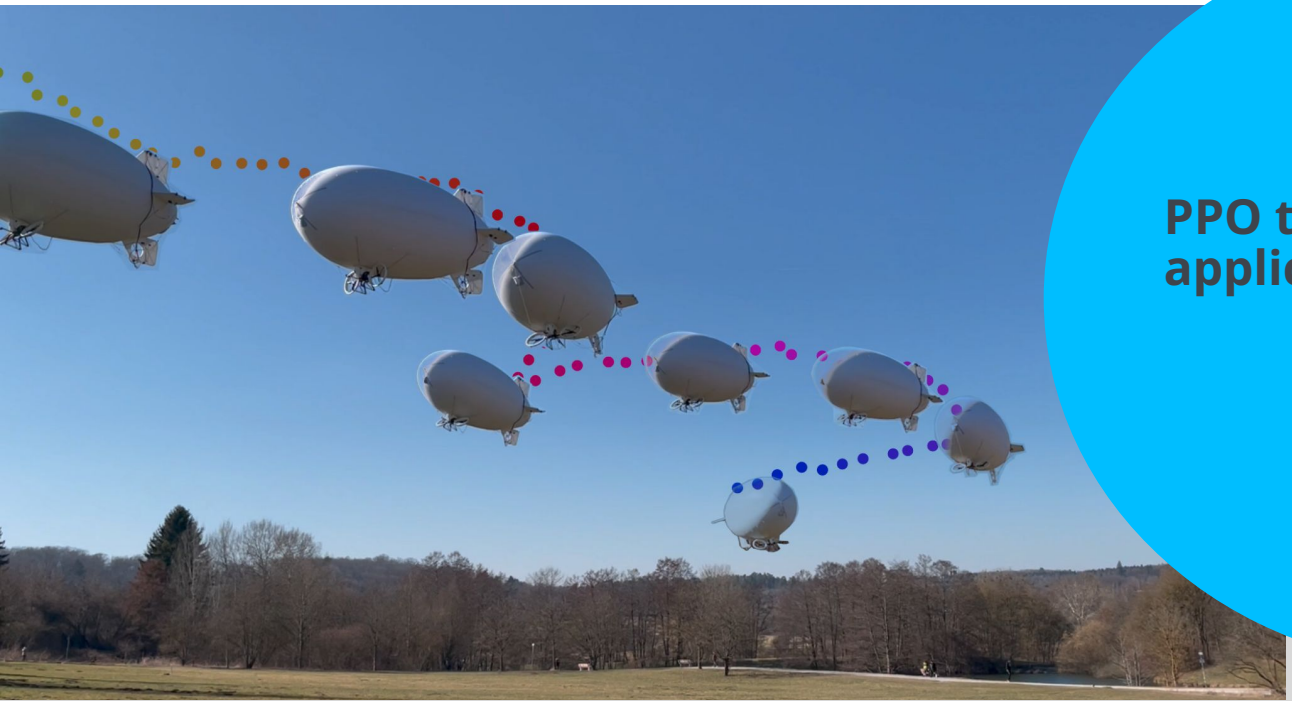
University of Stuttgart
Institute of Flight Mechanics and Controls



PPO tutorial and its application in Robotics

Yu-Tang Liu

Flight Robotics and Perception Group

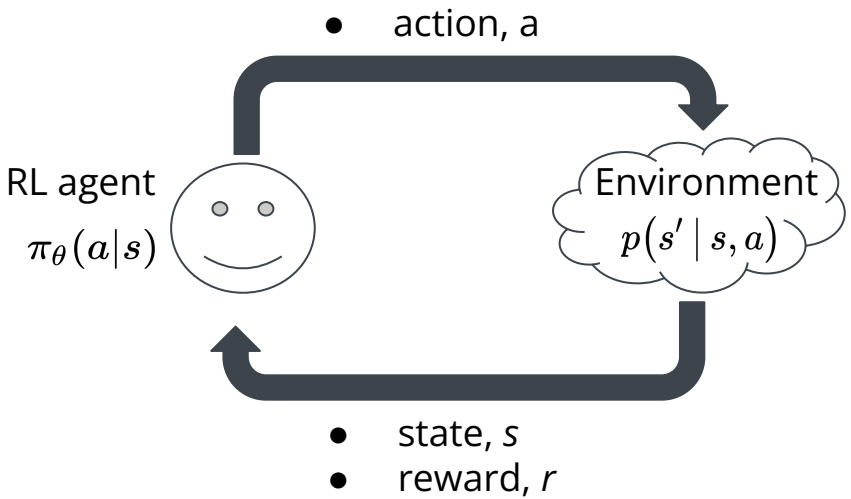
PERCEIVING SYSTEMS
MAX PLANCK INSTITUTE FOR INTELLIGENT SYSTEMS

iFR

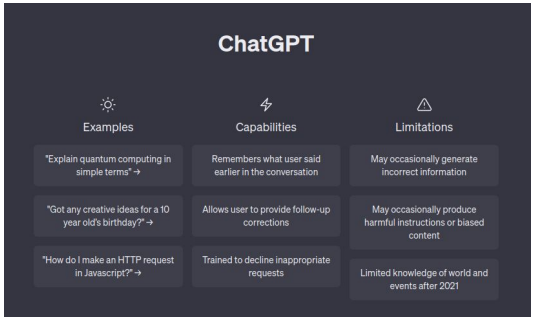
Topics

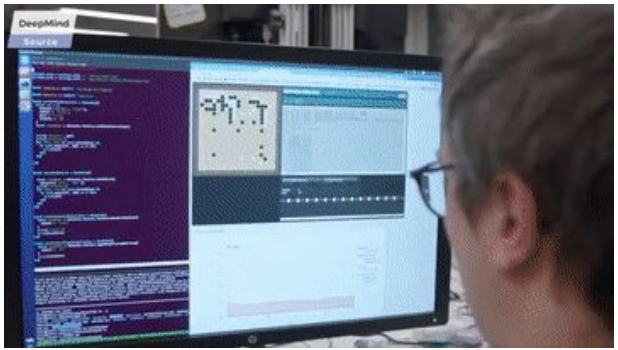
- **Introduction to Reinforcement Learning**
  - **what can RL do?**
- PPO algorithm + hands-on
- Blimp Control with Residual RL

Reinforcement Learning - learning from interaction



Goal: maximize cumulative reward
step1: collect data from the env
step2: update control policy
repeat



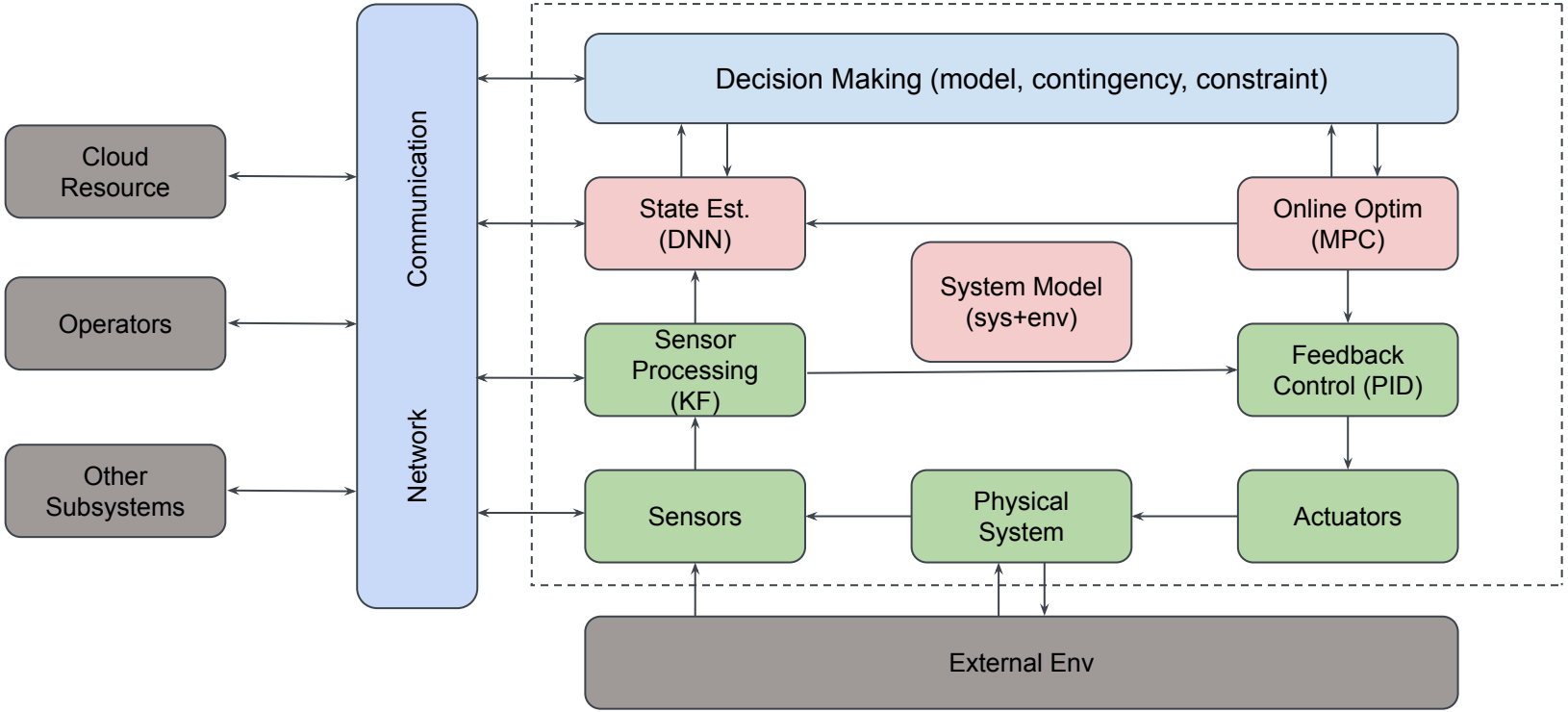


- Silver, 2017, Mastering the game of Go without human knowledge
- Vollenweider, 2022, Advanced Skills through Multiple Adversarial Motion Priors in Reinforcement Learning
- OpenAI, 2023, ChatGPT, https://ai.com
- Vinyals, 2019, AlphaStar: Grandmaster level in StarCraft II using multi-agent reinforcement learning
- Liu, 2022, From motor control to team play in simulated humanoid football

Topics

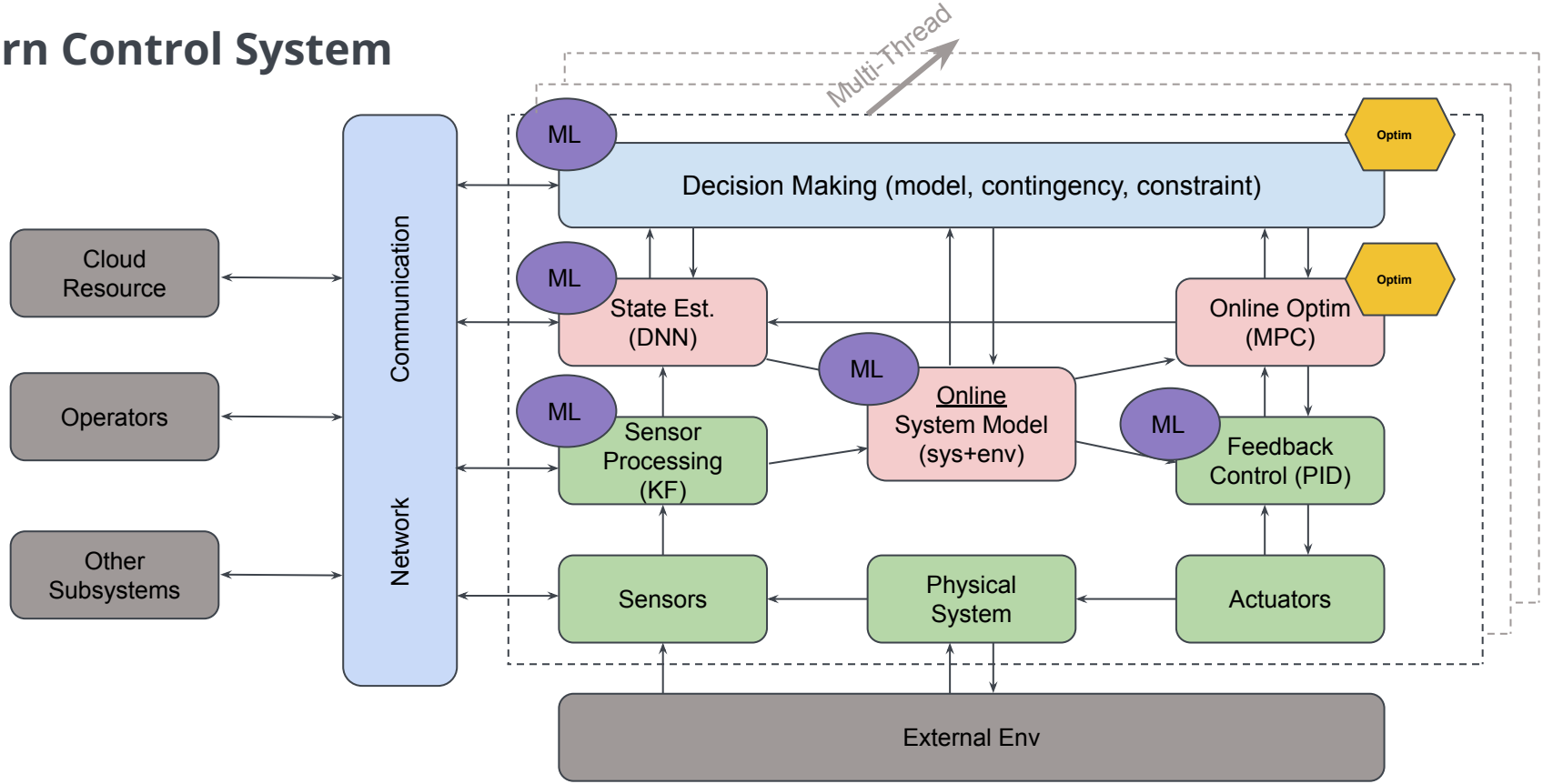
- **Introduction to Reinforcement Learning**
  - what can RL do?
  - **why using RL in robotics?**
- PPO algorithm + hands-on
- Blimp Control with Residual RL

Modern Control System



Murray, 2020, Can We Really Use Machine Learning in Safety Critical Systems?

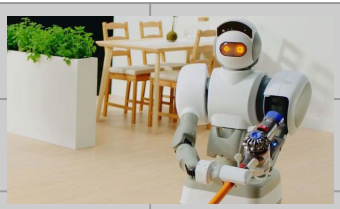
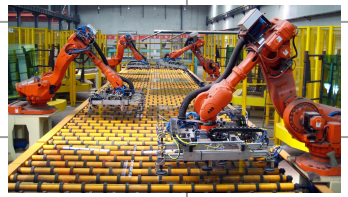
Modern Control System



Murray, 2020, Can We Really Use Machine Learning in Safety Critical Systems?

Task Difficulties

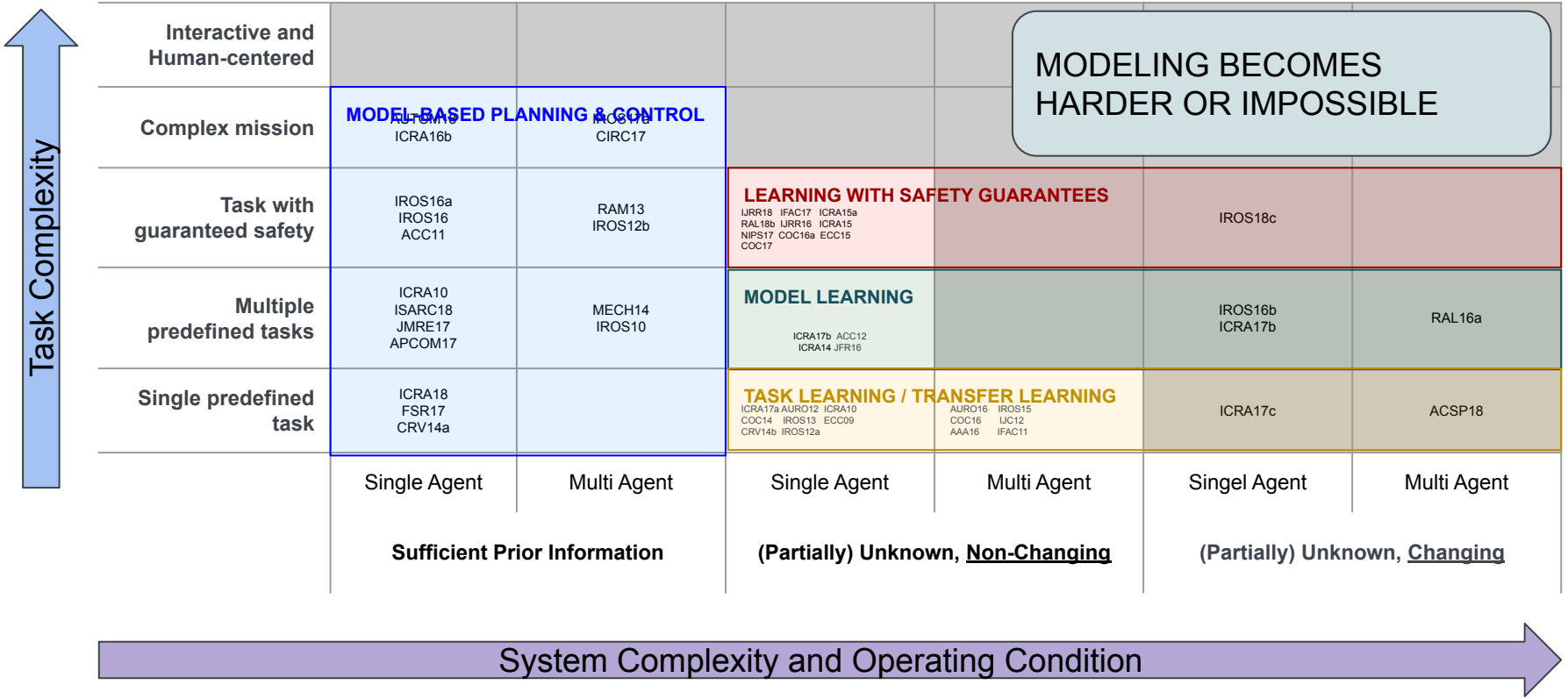
Task Complexity

	Single Agent	Multi Agent	Single Agent	Multi Agent	Singel Agent	Multi Agent
Interactive and Human-centered						
Complex mission	AUTOM18 ICRA16b	IROS17a CIRC17				
Task with guaranteed safety	IROS16a IROS16 ACC11	RAM13 IROS12b	IJRR18 IFAC1 RAL18b IJRR1 NIPS17 COC1 COC17		IROS18c	
Multiple predefined tasks			ICRA17b ACC12 ICRA14 JFR16			RAL16a
Single predefined task	CRV14a		ICRA17a AURO12 ICRA10 COC14 IROS13 ECC09 CRV14b IROS12a	AURO16 IROS15 COC16 IJC12 AAA16 IFAC11		ACSP18
	Sufficient Prior Information		**(Partially) Unknown, <u>Non-Changing</u>**		**(Partially) Unknown, <u>Changing</u>**	

System Complexity and Operating Condition

Schoellig, 2020, Machine Learning for Robotics: Achieving Safety, Performance and Reliability by Combining Models and Data in a Closed-Loop System Architecture

Task Difficulties

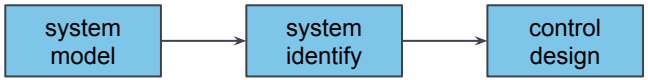


Schoellig, 2020, Machine Learning for Robotics: Achieving Safety, Performance and Reliability by Combining Models and Data in a Closed-Loop System Architecture

Conclusion

- Modern Control System:
  - problem: complex system, environment, and task formulation
  - goal: active decision making / expect safe and high performance behavior
  - given: large prior uncertainty (unknown system and env)

- Design Framework
  - classic control:



    - pros:
      - well-defined
      - safety / worst-case
      - parametric models
    - cons:
      - performance limited by system understanding
      - hard to scale up

  - RL:

    - pros:
      - scale with data
      - avg. case
      - high capacity models
    - cons:
      - no safety
      - low explainability

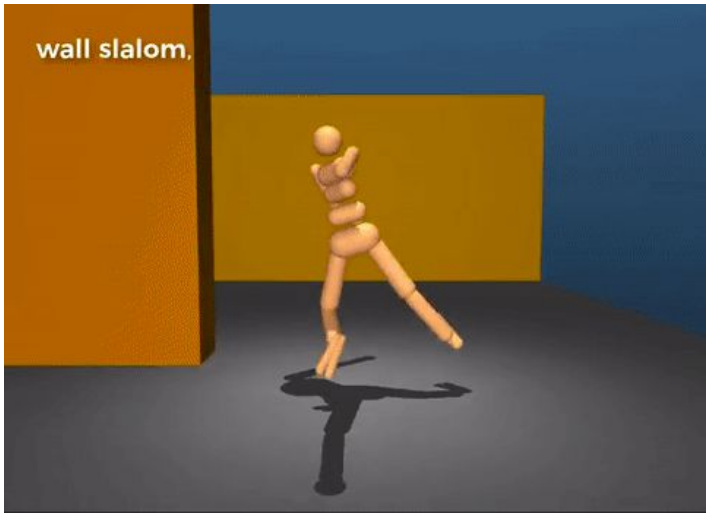
Schoellig, 2020, Machine Learning for Robotics: Achieving Safety, Performance and Reliability by Combining Models and Data in a Closed-Loop System Architecture

Topics

- **Introduction to Reinforcement Learning**
  - what can RL do?
  - why using RL in robotics?
  - **challenges with RL**
- PPO algorithm + hands-on
- Blimp Control with Residual RL

Unsolved Challenges for RL + Robotics

- safety
  - do some weird stuff
  - no control stability and safety guarantee
- training
  - no training stability guarantee
  - low sample efficiency
  - exploration is still difficult
- practicality
  - partial observation & disturbance
  - sim-to-real transfer
  - multi-task ability
  - hard to define reward



Heess, 2017, Emergence of Locomotion Behaviours in Rich Environments

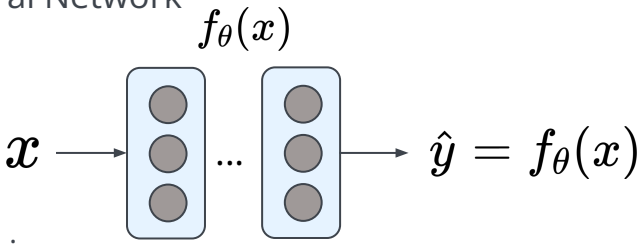
Q&A I

Topics

- Introduction to Reinforcement Learning
- **PPO algorithm + hands-on**
  - **Deep Learning**
  - Quick Fact about PPO
  - Policy Gradient
  - Tricks
  - PPO = Policy Gradient + Tricks
- Blimp Control with Residual RL

Deep Learning Overview

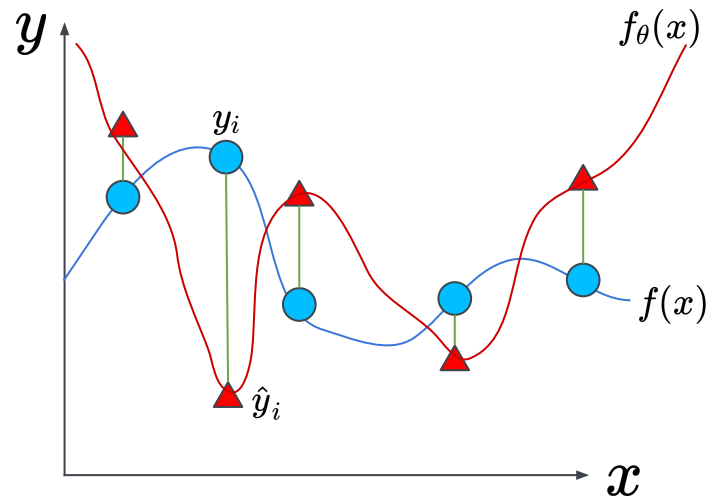
- True Unknown Function $y = f(x)$
- Neural Network

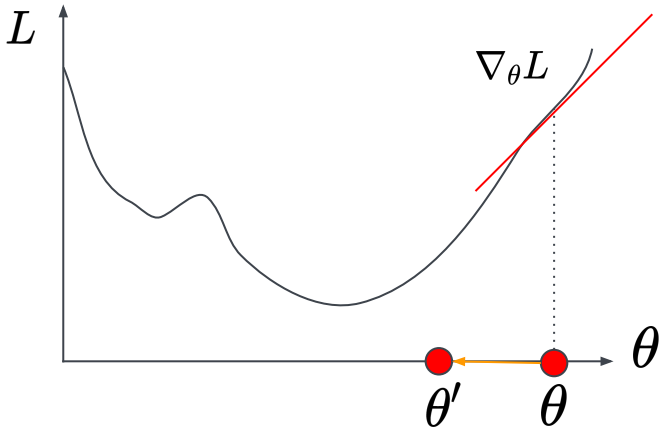


- Training
  - Step1: Collect Samples (x_i, y_i)
  - Step2: Compute Loss $\hat{y}_i = f_\theta(x_i)$

    $L = \sum_i ||y_i - \hat{y}_i||$
  - Step3: Optimization $\nabla_\theta L$

    $\theta' \leftarrow \theta - \alpha \nabla_\theta L$

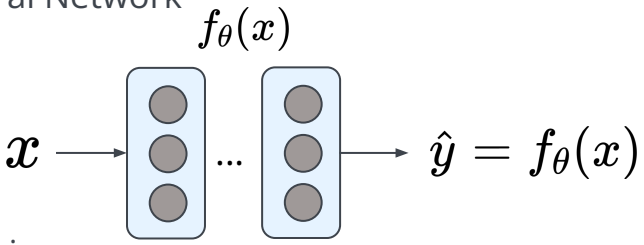




- Hung-Yi Lee, https://www.youtube.com/watch?v=bHcJCp2Fyxs

Deep Learning Overview

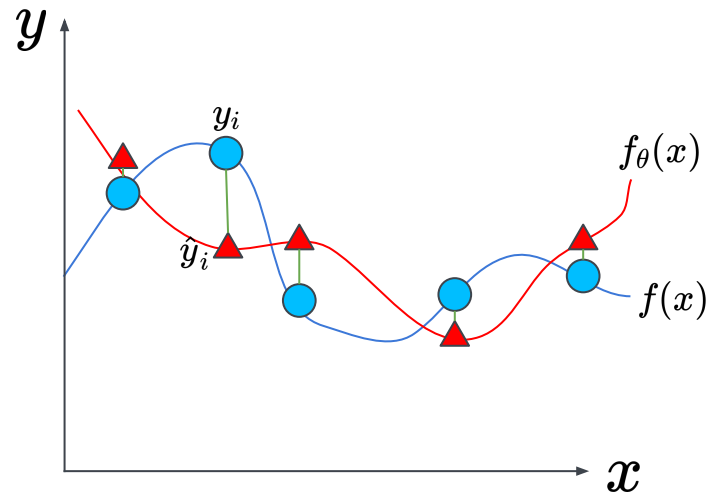
- True Unknown Function $y = f(x)$
- Neural Network

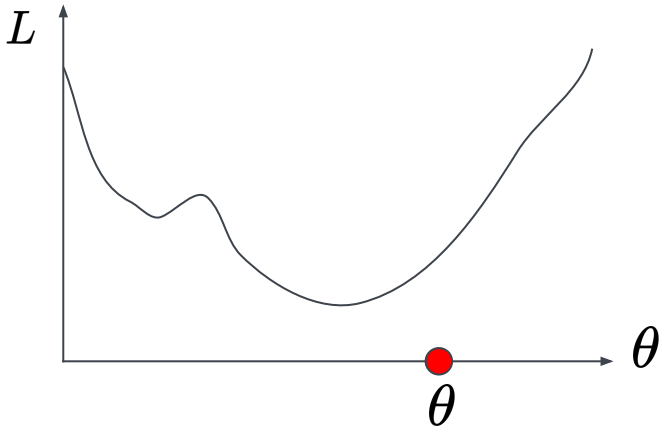


- Training
  - Step1: Collect Samples (x_i, y_i)
  - Step2: Compute Loss $\hat{y}_i = f_\theta(x_i)$

    $L = \sum_i ||y_i - \hat{y}_i||$
  - Step3: Optimization $\nabla_\theta L$

    $\theta' \leftarrow \theta - \alpha \nabla_\theta L$

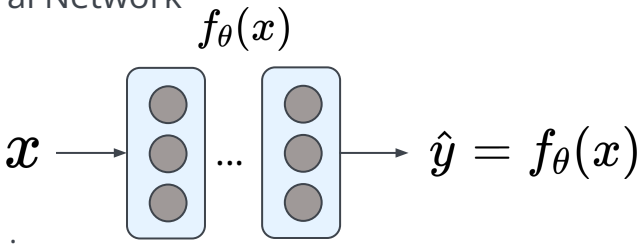




- Hung-Yi Lee, https://www.youtube.com/watch?v=bHcJCp2Fyxs

Deep Learning Overview

- True Unknown Function $y = f(x)$
- Neural Network

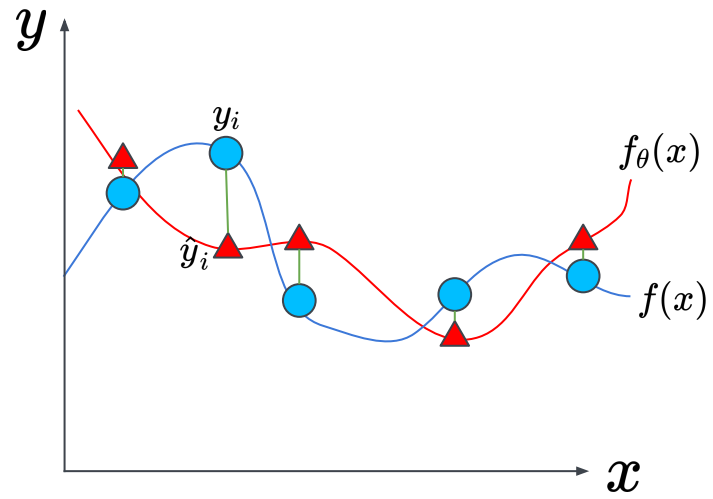


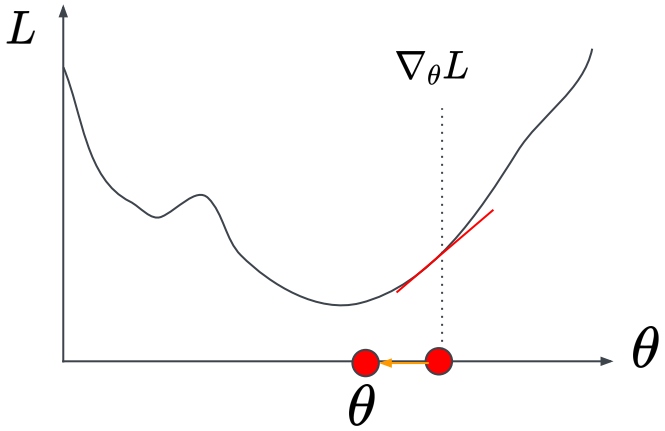
- Training
  - Step1: Collect Samples (x_i, y_i)
  - Step2: Compute Loss $\hat{y}_i = f_\theta(x_i)$

$$L = \sum_i ||y_i - \hat{y}_i||$$

  - Step3: Optimization $\nabla_\theta L$

$$\theta' \leftarrow \theta - \alpha \nabla_\theta L$$

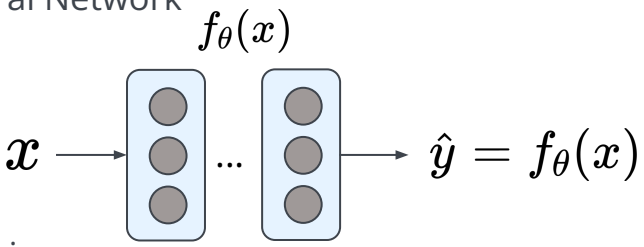




- Hung-Yi Lee, https://www.youtube.com/watch?v=bHcJCp2Fyxs

Deep Learning Overview

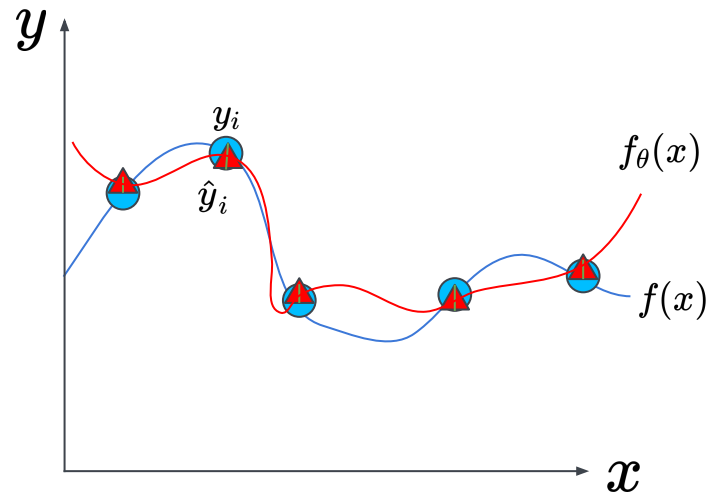
- True Unknown Function $y = f(x)$
- Neural Network

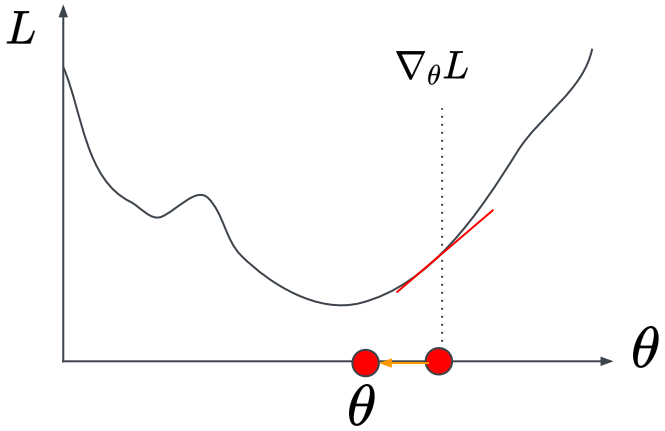


- Training
  - Step1: Collect Samples (x_i, y_i)
  - Step2: Compute Loss $\hat{y}_i = f_\theta(x_i)$

    $L = \sum_i ||y_i - \hat{y}_i||$
  - Step3: Optimization $\nabla_\theta L$

    $\theta' \leftarrow \theta - \alpha \nabla_\theta L$



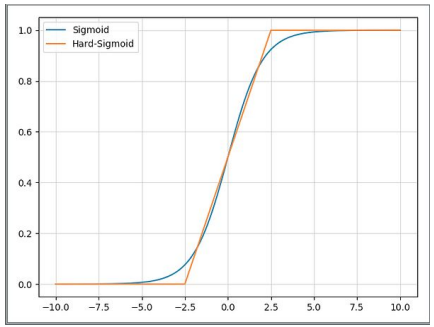


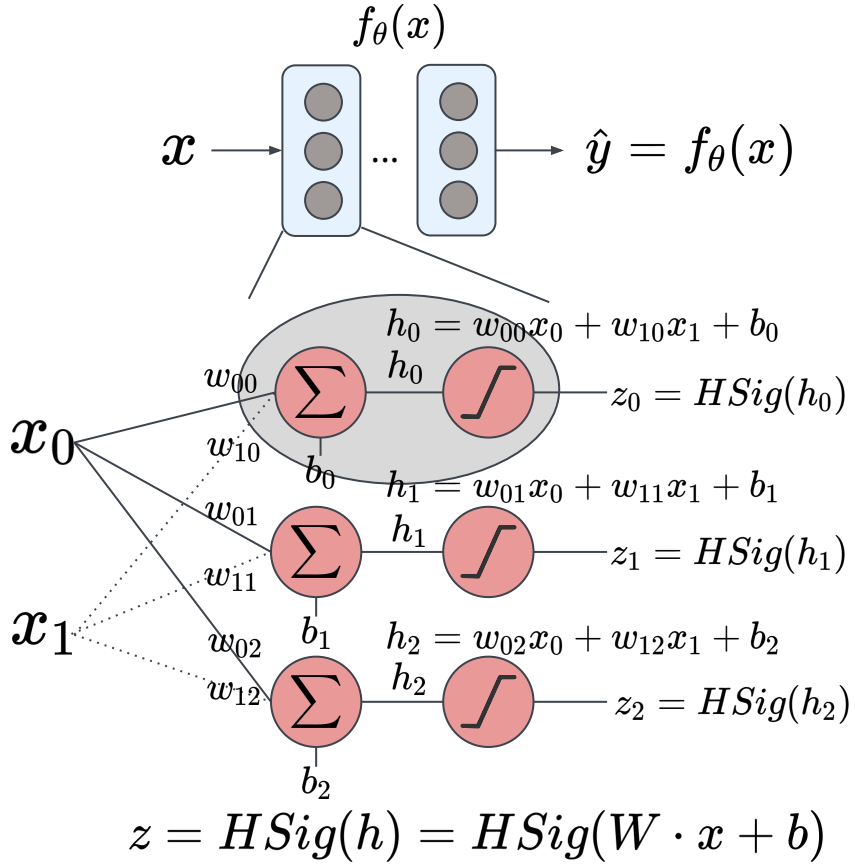
- Hung-Yi Lee, https://www.youtube.com/watch?v=bHcJCp2Fyxs

Neural Network (NN)

- hard sigmoid 

$$HSig(x) = \max\left(0, \min\left(1, \frac{x+1}{2}\right)\right)$$



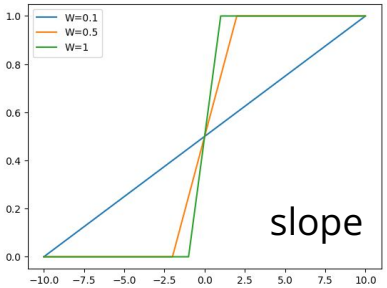


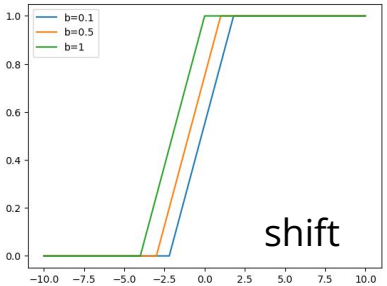
$$z = HSig(h) = HSig(W \cdot x + b)$$

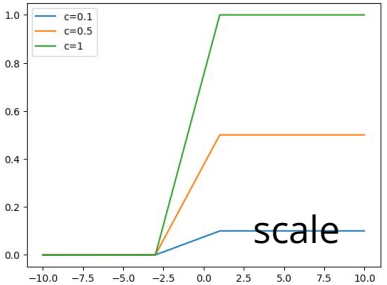
- Hung-Yi Lee, https://www.youtube.com/watch?v=bHcJCp2Fyxs

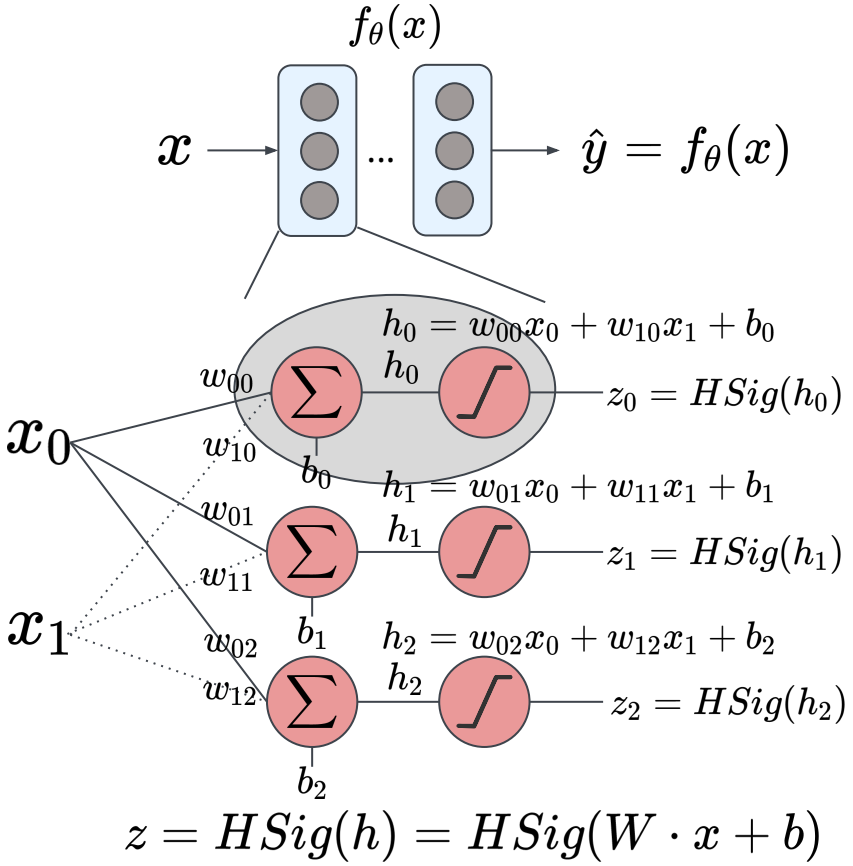
Neural Network (NN)

$$z_0 = HSig(w_{00}x_0 + \cancel{w_{10}x_1} + b_0)$$







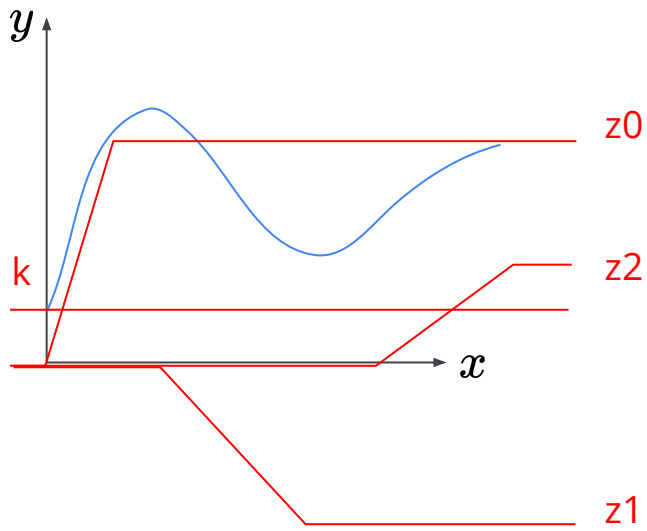


$$z = HSig(h) = HSig(W \cdot x + b)$$

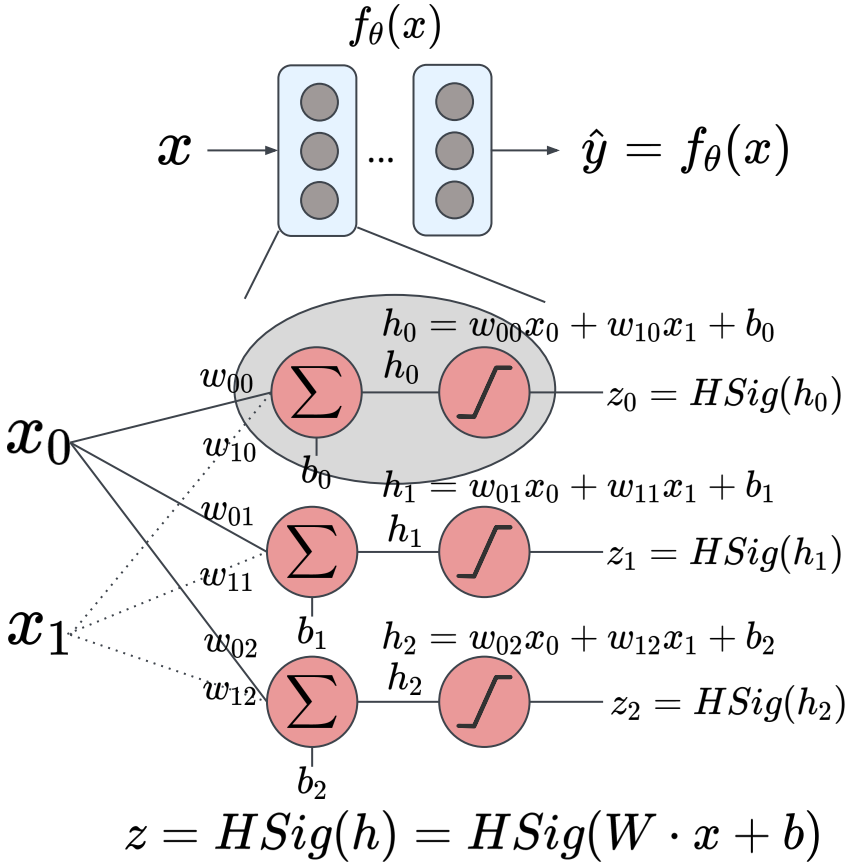
- Hung-Yi Lee, https://www.youtube.com/watch?v=bHcJCp2Fyxs

Neural Network (NN)

- blue = k + multiple 

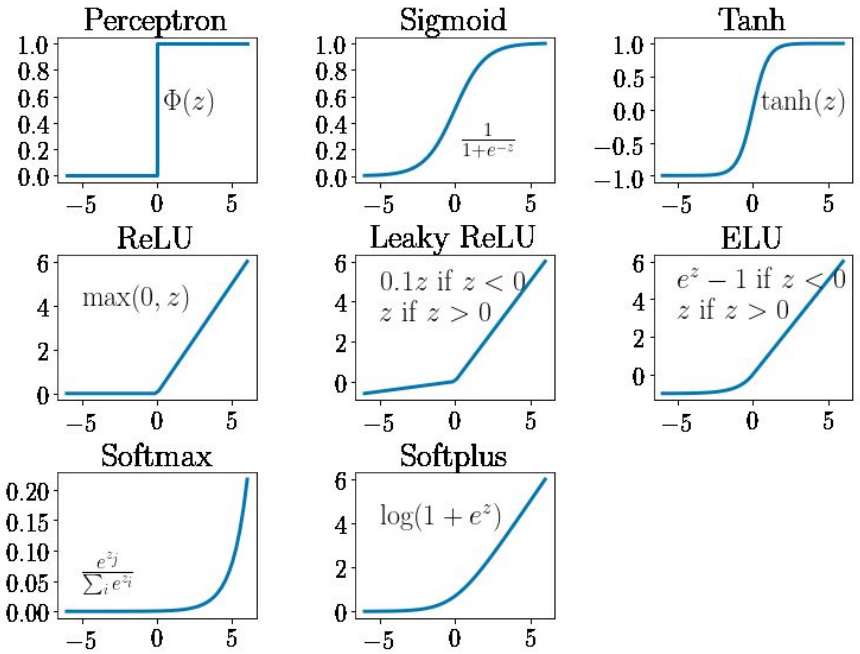


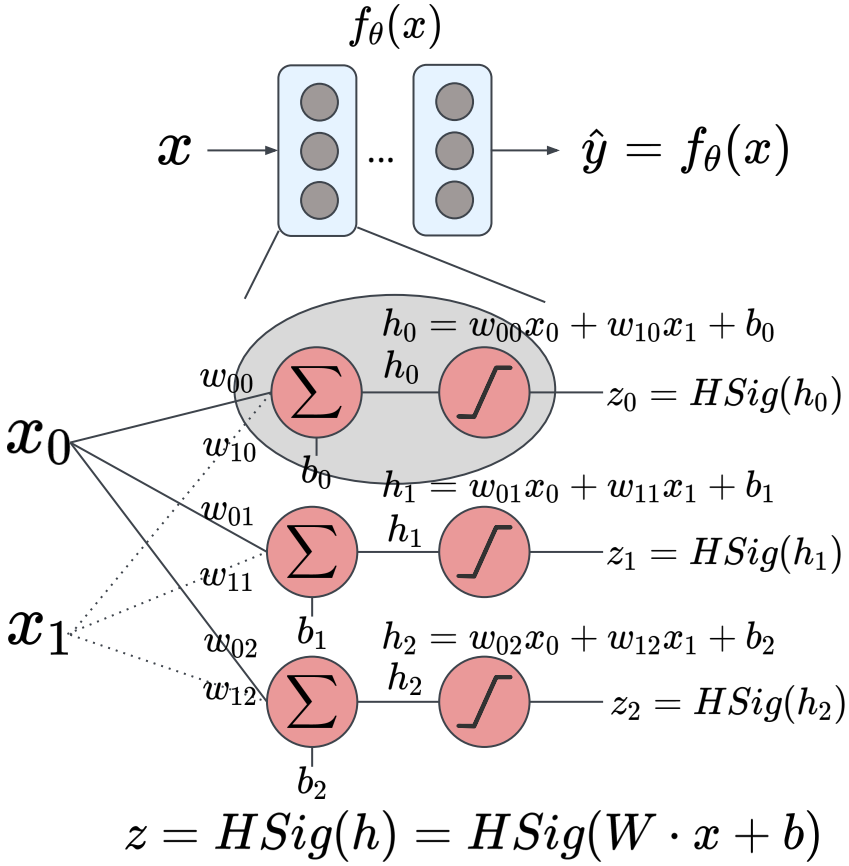
- training a NN → tune W and b to put sigmoid in the right spot



- Hung-Yi Lee, https://www.youtube.com/watch?v=bHcJCp2Fyxs

Neural Network (NN)

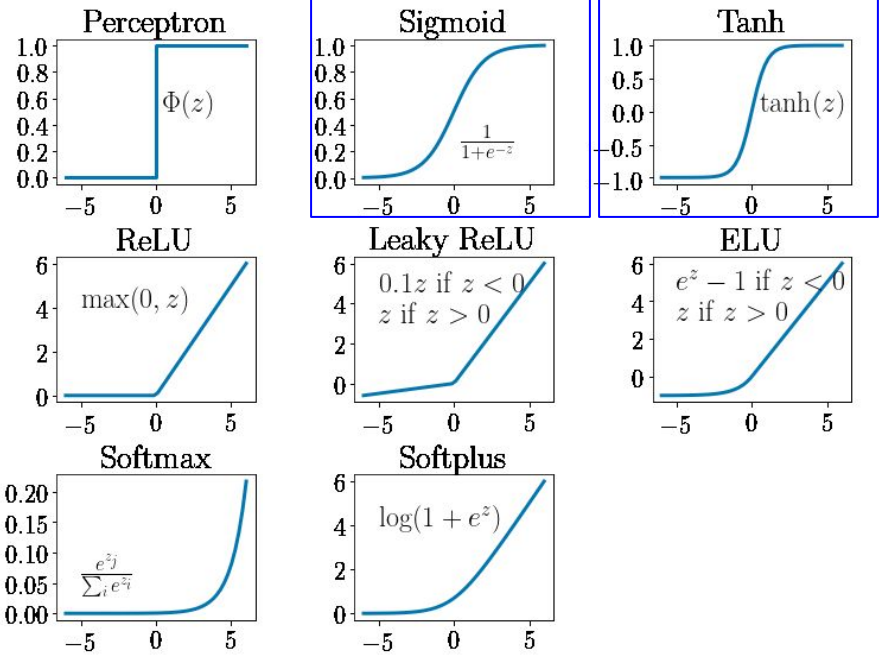


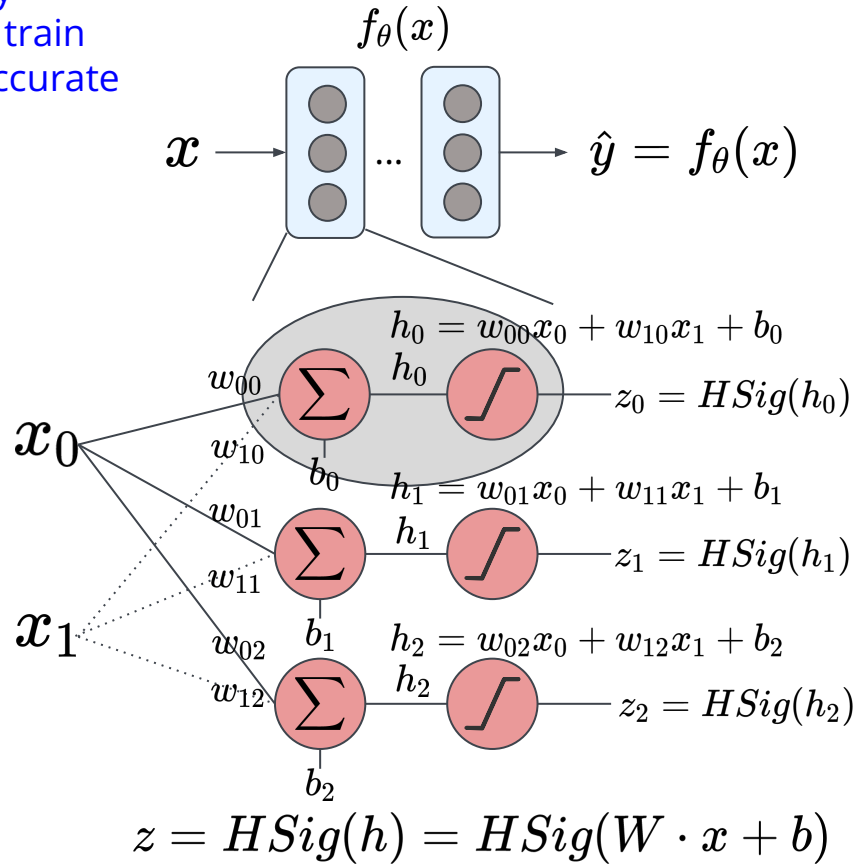


- Hung-Yi Lee, https://www.youtube.com/watch?v=bHcJCp2Fyxs

Neural Network (NN)

- logistic Family
  - hard to train
  - more accurate

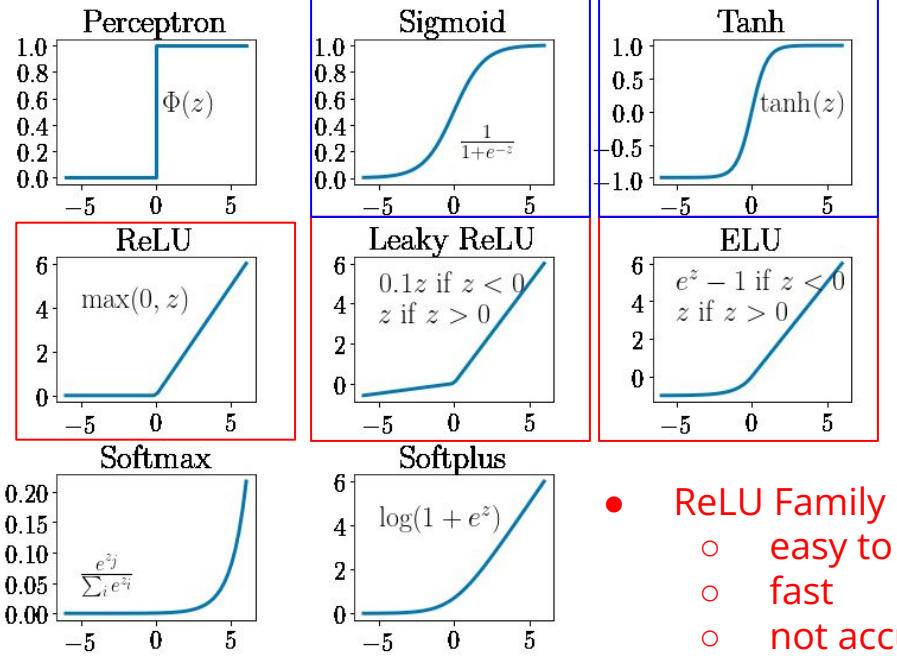




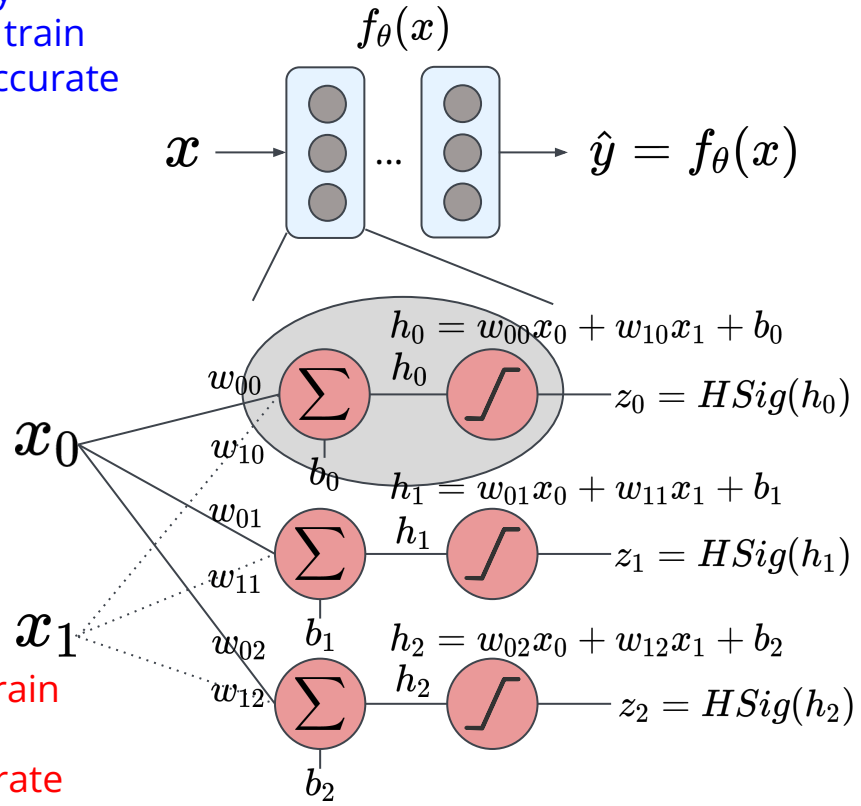
- Hung-Yi Lee, https://www.youtube.com/watch?v=bHcJCp2Fyxs

Neural Network (NN)

- logistic Family
  - hard to train
  - more accurate



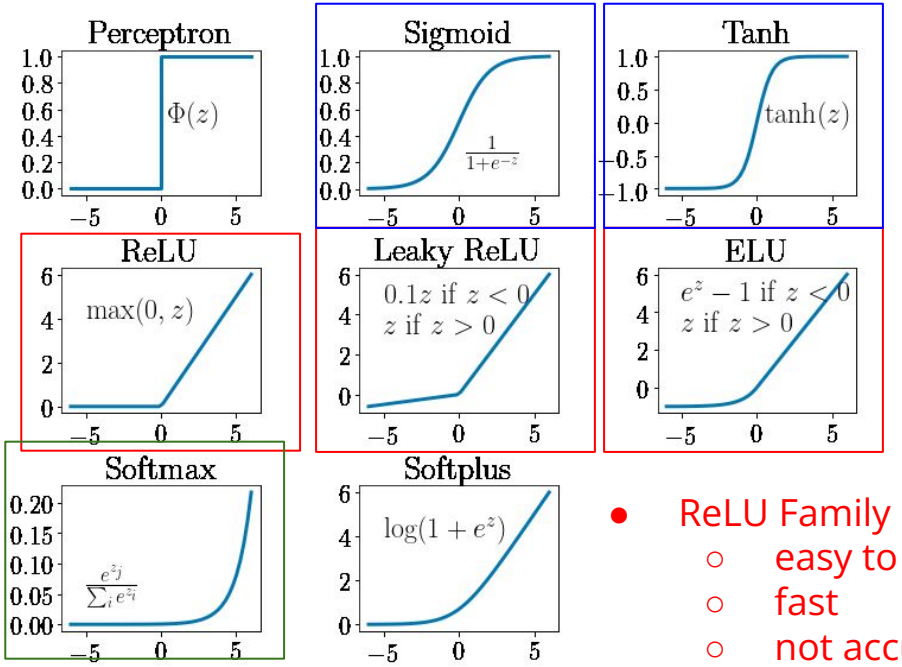
- ReLU Family
  - easy to train
  - fast
  - not accurate



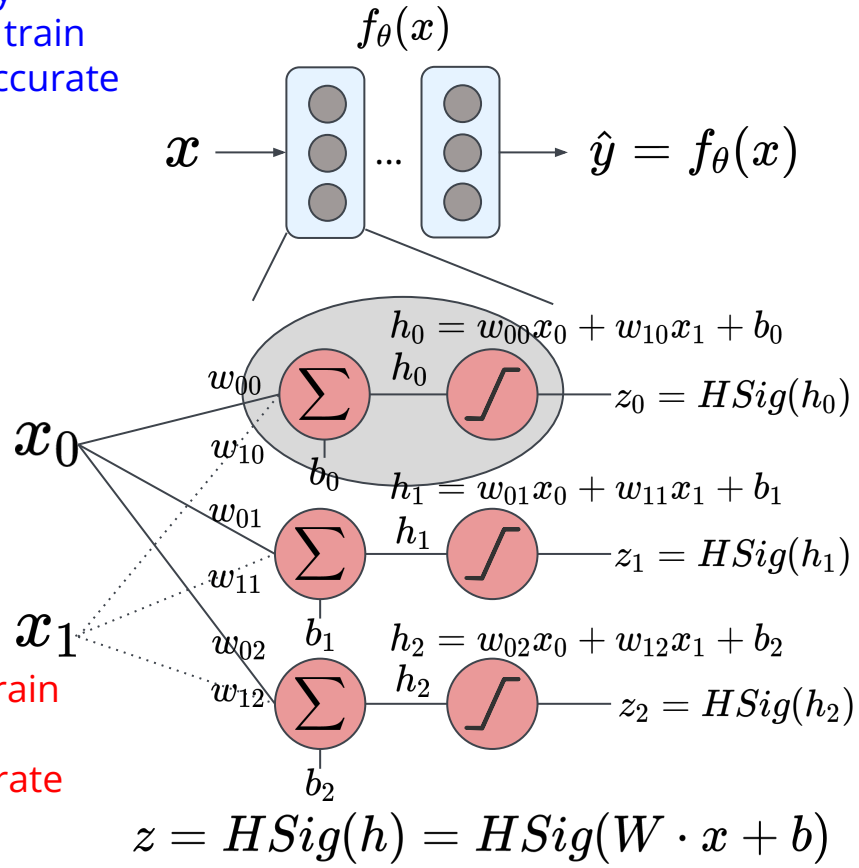
- Hung-Yi Lee, https://www.youtube.com/watch?v=bHcJCp2Fyxs

Neural Network (NN)

- logistic Family
  - hard to train
  - more accurate



- ReLU Family
  - easy to train
  - fast
  - not accurate



- Output layer: output [0,1] e.g. probability

- Hung-Yi Lee, https://www.youtube.com/watch?v=bHcJCp2Fyxs

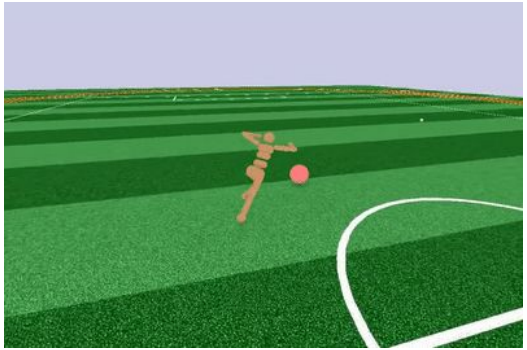
Q&A II

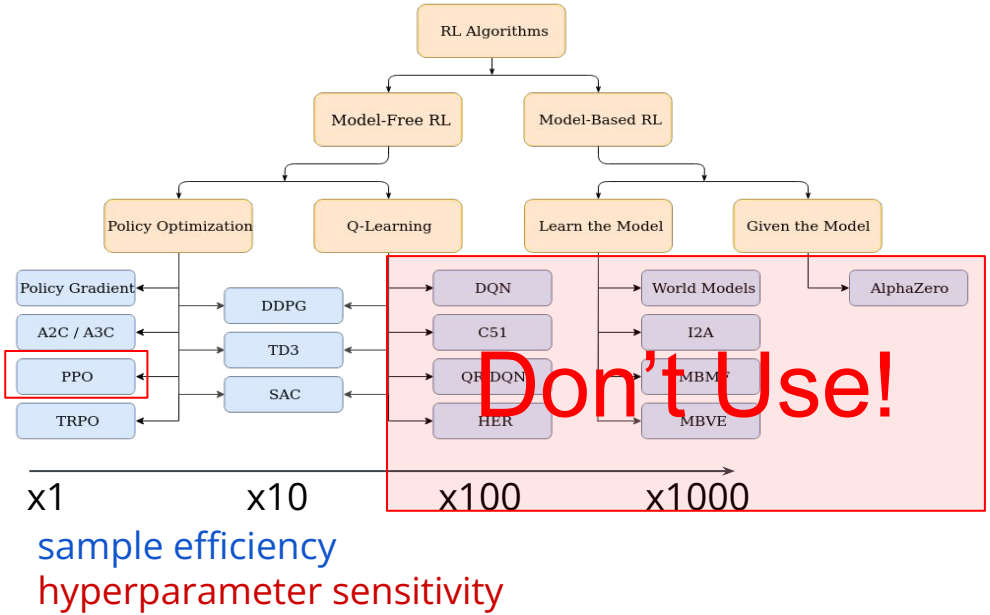
Topics

- Introduction to Reinforcement Learning
- **PPO algorithm + hands-on**
  - Deep Learning
  - **Quick Fact about PPO**
  - Policy Gradient
  - Tricks
  - PPO = Policy Gradient + Tricks
- Blimp Control with Residual RL

PPO (Proximal Policy Optimization)

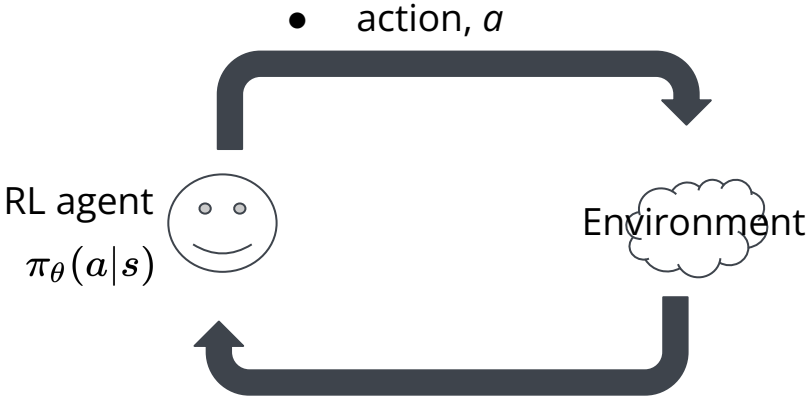
- default RL algorithm at OpenAI
- training speed
  - poor sample efficiency
  - fastest on wall-clock time
    - low hyper-param sensitivity
      - little tuning
    - highest training stability





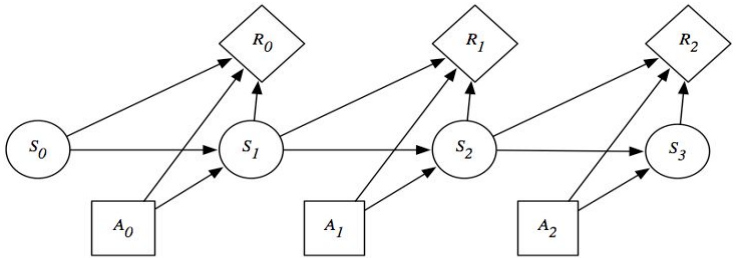
- https://openai.com/research/openai-baselines-ppo

Reinforcement Learning



Given some data, how to update policy?

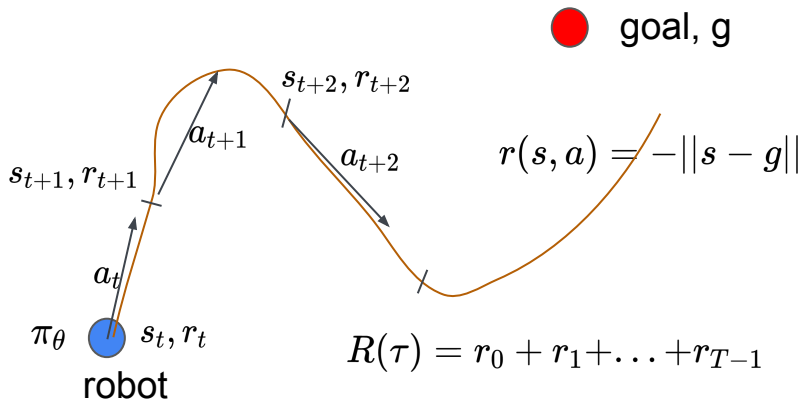
Markov Decision Process (MDP)



- trajectory: $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \dots)$
- dynamics: $p(s_{t+1} \mid s_t, a_t)$
- total reward: $R(\tau) = \sum_{t=0}^{\infty} r_t$
- goal of RL: $\pi^\star = \arg\max_\pi [\ \mathbb{E}_{p,\pi}[R(\tau)]\ ]$

Generate Trajectories with a Policy

- policy:
  - input: state
  - output: action distribution
- parameterized by any function approximator, e.g. a NN



$\pi_\theta(\cdot \mid s_t)$

$$s_t = \begin{bmatrix} x \\ y \\ \psi \end{bmatrix}$$

continuous: normal distribution

$[\mathcal{N}(\mu_x, \sigma_x), \mathcal{N}(\mu_y, \sigma_y)]$

[(.8, .1), (-.3,1.2)]

$$a_t = [F_x, F_y]$$

discrete: categorical distribution

.2	↑
.0	↓
.3	←
.5	→

sampling

$a_t \sim \pi(\cdot \mid s_t)$

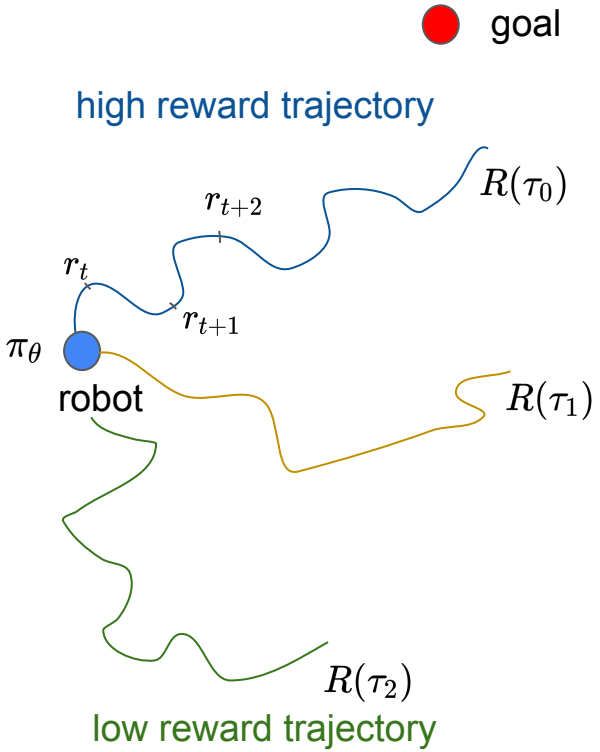
$$a_t = [\uparrow]$$

Topics

- Introduction to Reinforcement Learning
- **PPO algorithm + hands-on**
  - Deep Learning
  - Quick Fact about PPO
  - **Policy Gradient**
  - Tricks
  - PPO = Policy Gradient + Tricks
- Blimp Control with Residual RL

Policy Gradient

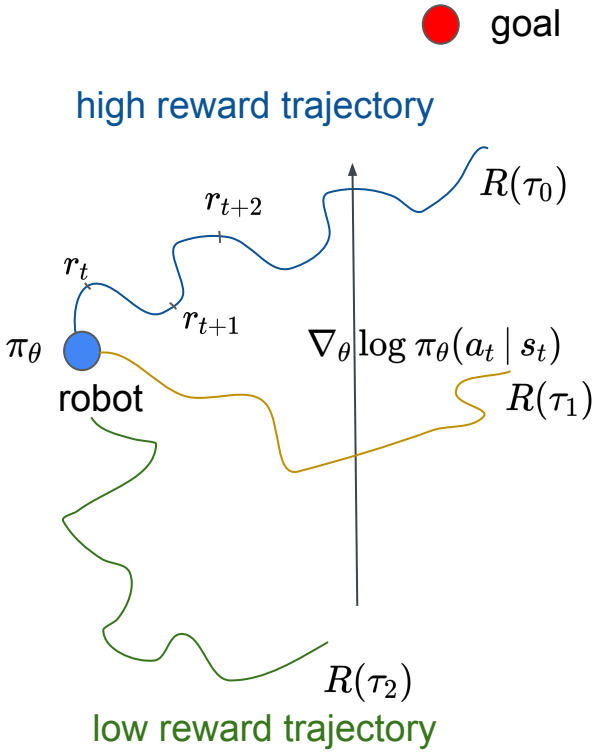
- given the objective function
  - $J(\pi_\theta) = \mathbb{E}_{\pi,p}[R(\tau)]$
- we can improve the policy by gradient ascent
  - $\theta' \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta)$
- with some math, we find
  - $$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\pi,p}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t) R(\tau)\right]$$
  - so called *policy gradient*
  - increase the probability of high reward trajectory



- spinningup, https://spinningup.openai.com/en/latest/algorithms/vpg.html

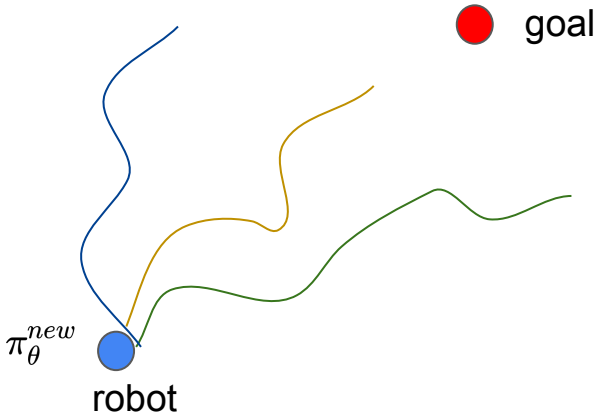
Policy Gradient

- given the objective function
  - $J(\pi_\theta) = \mathbb{E}_{\pi,p}[R(\tau)]$
- we can improve the policy by gradient ascent
  - $\theta' \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta)$
- with some math, we find
  - $$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\pi,p}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t) R(\tau)\right]$$
  - so called *policy gradient*
  - increase the probability of high reward trajectory



- spinningup, https://spinningup.openai.com/en/latest/algorithms/vpg.html

Policy Gradient



- given the objective function
  - $J(\pi_\theta) = \mathbb{E}_{\pi,p}[R(\tau)]$
- we can improve the policy by gradient ascent
  - $\theta' \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta)$
- with some math, we find
  - $$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\pi,p}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t) R(\tau)\right]$$
  - so called *policy gradient*
  - increase the probability of high reward trajectory

- spinningup, https://spinningup.openai.com/en/latest/algorithms/vpg.html

Derivation of Policy Gradient

- probability of a trajectory

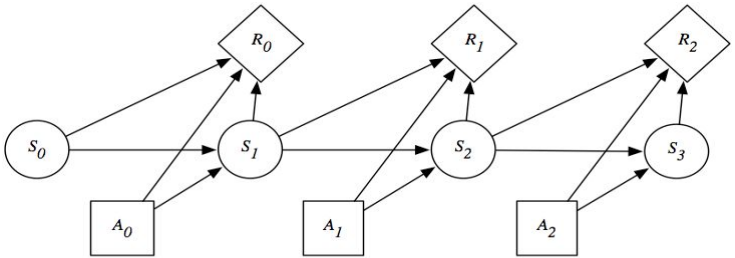
$$p_\theta(\tau) = p(s_0)\pi_\theta(a_0 \mid s_0)p(s_1 \mid s_0, a_0)\dots$$
$$= p(s_0)\prod_{t=0}^{T-1}\pi_\theta(a_t \mid s_t)p(s_{t+1} \mid s_t, a_t)$$

- Expected Total Reward

$$J(\pi) = \mathbb{E}_{p,\pi}[R(\tau)] = \sum_\tau R(\tau)p_\theta(\tau)$$

- Policy Gradient

$$\nabla J(\pi) = \sum_\tau R(\tau)\nabla p_\theta(\tau) = \sum_\tau R(\tau)p_\theta(\tau)\frac{\nabla p_\theta(\tau)}{p_\theta(\tau)}$$
$$= \sum_\tau R(\tau)p_\theta(\tau)\nabla \log p_\theta(\tau) \qquad \nabla f(x) = f(x)\nabla \log f(x)$$
$$= \mathbb{E}_{p_{\theta(\tau)}}[R(\tau)\nabla \log p_\theta(\tau)]$$
$$= \mathbb{E}_{p,\pi}\left[\sum_{t=0}^{T-1}\nabla_\theta \log \pi_\theta(a_t \mid s_t)R(\tau)\right]$$
$$\approx \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}R(\tau_n)\nabla \log \pi_\theta(a_t \mid s_t)$$



$$\tau = \{s_0, a_0, s_1, a_1, \dots, s_T, a_T\}$$

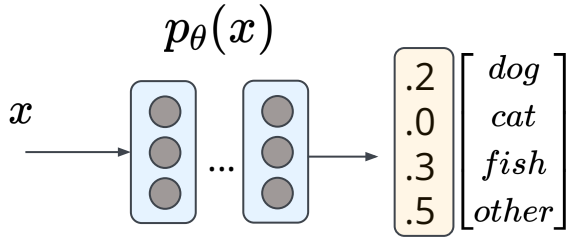
$$R(\tau) = r_0 + r_1 + \dots + r_{T-1}$$

- Hung-Yi Lee, DRL Lecture: Policy Gradient, https://www.youtube.com/watch?v=z95ZYgPgXOY&t

Policy Gradient

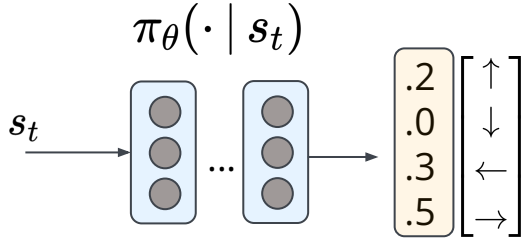
- ML Classification
  cross entropy loss

$$\nabla_\theta L \approx \frac{1}{D}\sum_{i=0}^{D-1} T_i \nabla \log p_\theta(x_i)$$



- RL
  policy gradient

$$\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1} R(\tau_n)\nabla \log \pi_\theta(a_t \mid s_t)$$



- Hung-Yi Lee, DRL Lecture: Policy Gradient, https://www.youtube.com/watch?v=z95ZYgPgXOY&t

REINFORCE

repeat

- collect data by policy π_θ

$$\mathcal{D} = (s_t, a_t, r_t, s_{t+1}, \log \pi_\theta(a_t \mid s_t)) \mid_{t=0:T-1}$$

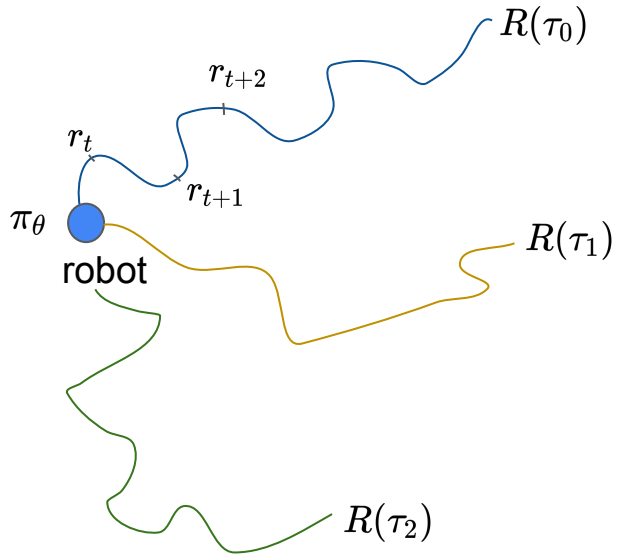
- compute policy gradient

$$J(\pi_\theta) \approx \frac{1}{N} \sum_{n=0}^{N-1} \sum_{t=0}^{T-1} R(\tau_n) \log \pi_\theta(a_t \mid s_t)$$

$$\nabla_\theta J(\pi_\theta)$$

- update policy and discard used data

$$\theta' \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta)$$



- Hung-Yi Lee, DRL Lecture: Policy Gradient, https://www.youtube.com/watch?v=z95ZYgPgXOY&t

Reinforcemeant Learning

repeat

- collect data by policy π_θ

$$\mathcal{D} = (s_t, a_t, r_t, s_{t+1}, \log \pi_\theta(a_t \mid s_t)) \mid_{t=0:T-1}$$

- compute policy gradient

$$J(\pi_\theta) \approx \frac{1}{N} \sum_{n=0}^{N-1} \sum_{t=0}^{T-1} R(\tau_n) \log \pi_\theta(a_t \mid s_t)$$

$$\nabla_\theta J(\pi_\theta)$$

- update policy and discard used data

$$\theta' \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta)$$

Supervised Learning

- prepare data

$$(x_i, y_i)$$

repeat

- compute loss

$$L = \sum_i ||y_i - \hat{y}_i|| \qquad \hat{y}_i = f_\theta(x_i)$$

$$\nabla_\theta L$$

- update model

$$\theta' \leftarrow \theta - \alpha \nabla_\theta L$$

- Hung-Yi Lee, DRL Lecture: Policy Gradient, https://www.youtube.com/watch?v=z95ZYgPgXOY&t

Reinforcemeant Learning

repeat

- collect data by policy π_θ

$$\mathcal{D} = (s_t, a_t, r_t, s_{t+1}, \log \pi_\theta(a_t \mid s_t)) \mid_{t=0:T-1}$$

- compute policy gradient

$$J(\pi_\theta) \approx \frac{1}{N} \sum_{n=0}^{N-1} \sum_{t=0}^{T-1} R(\tau_n) \log \pi_\theta(a_t \mid s_t)$$

$$\nabla_\theta J(\pi_\theta)$$

- update policy and discard used data

$$\theta' \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta)$$

- data come from interaction
  - poor policy → poor data → worse policy
- no correct label → loss is less meaningful
  → check reward instead

Supervised Learning

- prepare data

$$(x_i, y_i)$$

repeat

- compute loss

$$L = \sum_i ||y_i - \hat{y}_i|| \qquad \hat{y}_i = f_\theta(x_i)$$

$$\nabla_\theta L$$

- update model

$$\theta' \leftarrow \theta - \alpha \nabla_\theta L$$

- data are pre-collected
  - poor model → improve data quality
- ground true y_i → loss reflect performance

- Hung-Yi Lee, DRL Lecture: Policy Gradient, https://www.youtube.com/watch?v=z95ZYgPgXOY&t

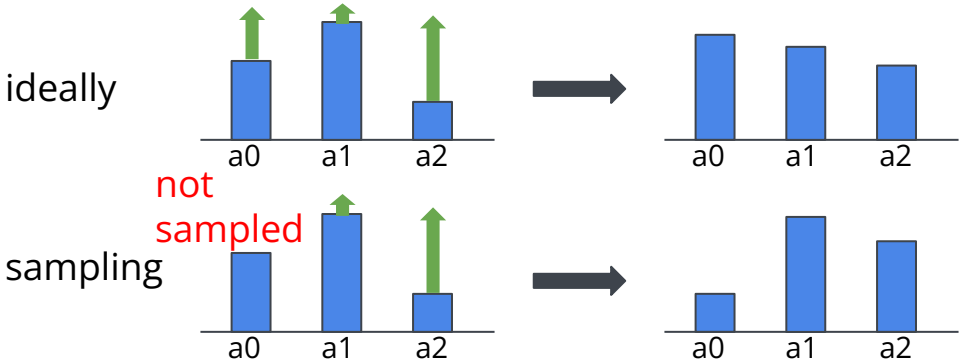
Q&A III

Topics

- Introduction to Reinforcement Learning
- **PPO algorithm + hands-on**
  - Deep Learning
  - Quick Fact about PPO
  - Policy Gradient
  - **Tricks**
  - PPO = Policy Gradient + Tricks
- Blimp Control with Residual RL

Trick 1: Add a Baseline

- Problem: limited number of sampled trajectories



$$p(a_0) + p(a_1) + p(a_2) = 1$$

probability of unsampled action decreases

- Add a baseline

$$\nabla_\theta J(\pi_\theta) \approx \frac{1}{N} \sum_{n=0}^{N-1} \sum_{t=0}^{T-1} [R(\tau_n) - b] \nabla \log \pi_\theta(a_t \mid s_t)$$

$$b \equiv \mathbb{E}[R(\tau)]$$

- Hung-Yi Lee, DRL Lecture: Policy Gradient, https://www.youtube.com/watch?v=z95ZYgPgXOY&t

Trick 1: Add a Baseline

- Problem: limited number of sampled trajectories

ideally

$$p(a_0) + p(a_1) + p(a_2) = 1$$

not sampled

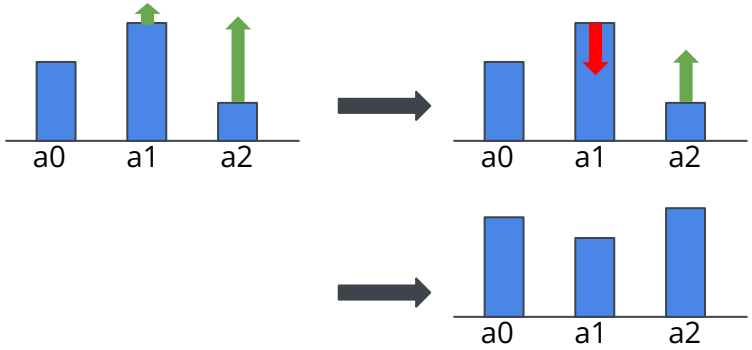
sampling

probability of unsampled action decreases

- Add a baseline

$$\nabla_\theta J(\pi_\theta) \approx \frac{1}{N} \sum_{n=0}^{N-1} \sum_{t=0}^{T-1} [R(\tau_n) - b] \nabla \log \pi_\theta(a_t \mid s_t)$$

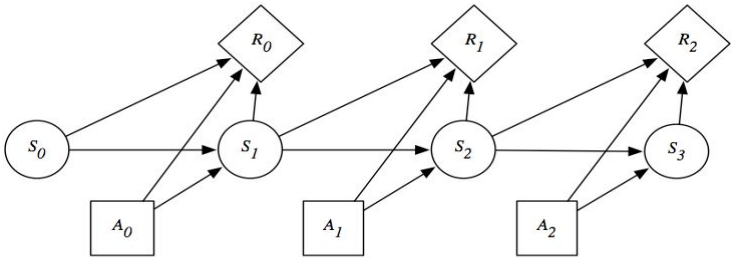
$$b \equiv \mathbb{E}[R(\tau)]$$



- Hung-Yi Lee, DRL Lecture: Policy Gradient, https://www.youtube.com/watch?v=z95ZYgPgXOY&t

Trick 2: Reward-to-Go

- Problem: Credit Assignment
  - a reward is only related to prior actions
  - e.g.
    - *a1* → *r1*, *r2*
    - *a1* ~~→~~ *r0*
- Reward-to-Go (RTG)



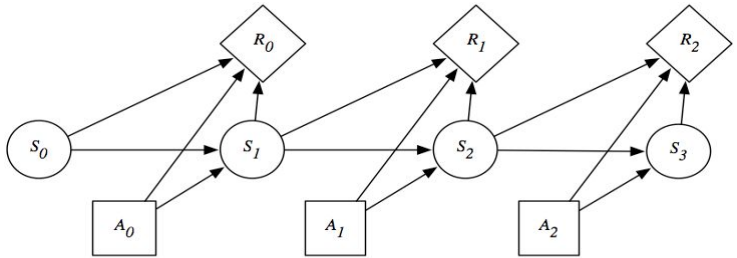
$$\nabla_\theta J(\pi_\theta) \approx \frac{1}{N} \sum_{n=0}^{N-1} \sum_{t=0}^{T-1} [R(\tau_n) - b] \nabla \log \pi_\theta(a_t \mid s_t)$$

$$\sum_{t'=t}^{T-1} r_{t'}^n$$

- Hung-Yi Lee, DRL Lecture: Policy Gradient, https://www.youtube.com/watch?v=z95ZYgPgXOY&t

Trick 2: Reward-to-Go

- Problem: Credit Assignment
  - a reward is only related to prior actions
  - e.g.
    - *a1* → *r1*, *r2*
    - *a1* ~~↛~~ *r0*
- Reward-to-Go (RTG)



$$\nabla_\theta J(\pi_\theta) \approx \frac{1}{N} \sum_{n=0}^{N-1} \sum_{t=0}^{T-1} [R(\tau_n) - b] \nabla \log \pi_\theta(a_t \mid s_t)$$

$$\sum_{t'=t}^{T-1} r_{t'}^n \quad \sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'}^n$$

- a reward is more related to recent actions
  - e.g.
    - *a0* → *r0* (direct)
    - *a0* ⇢ *r2* (indirect)
- add a discount factor
  - $\gamma < 1$

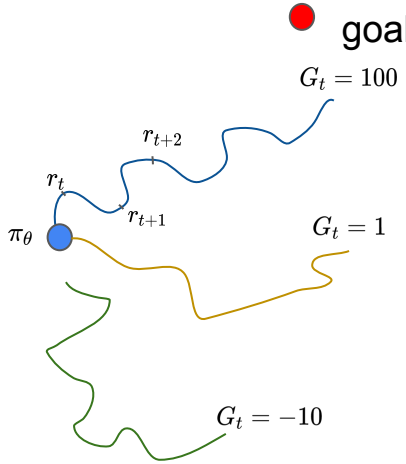
- Hung-Yi Lee, DRL Lecture: Policy Gradient, https://www.youtube.com/watch?v=z95ZYgPgXOY&t

Trick 3: Actor-Critic

- Problem: $\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}[G_t - b]\nabla \log \pi_\theta(a_t \mid s_t)$

  high variance → unstable

  $G_t = \sum_{t'=t}^{T-1}\gamma^{t'-t}r_{t'}^n$ discounted RTG



- Value function (Critic):
  - approximate G_t by $Q(s_t, a_t) \equiv \mathbb{E}[G_t \mid s_t, a_t]$
  - approximate b by $V(s_t) \equiv \mathbb{E}[G_t \mid s_t]$
  - $\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}[Q^\pi(s_t, a_t) - V^\pi(s_t)]\nabla \log \pi_\theta(a_t \mid s_t)$

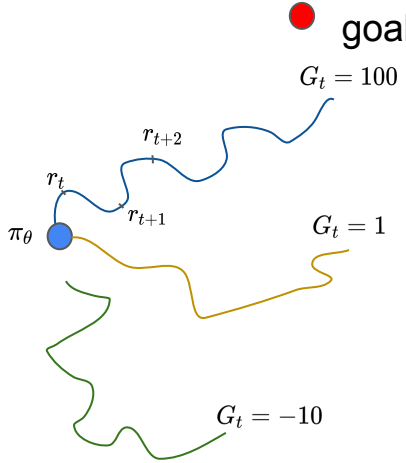
- Hung-Yi Lee, DRL Lecture: Policy Gradient, https://www.youtube.com/watch?v=z95ZYgPgXOY&t
- Mnih 2016, Asynchronous Methods for Deep Reinforcement Learning

Trick 3: Actor-Critic

- Problem: $\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}[G_t - b]\nabla \log \pi_\theta(a_t \mid s_t)$

  high variance → unstable

  $G_t = \sum_{t'=t}^{T-1}\gamma^{t'-t}r_{t'}^n$ discounted RTG



- Value function (Critic):
  - approximate G_t by $Q(s_t, a_t) \equiv \mathbb{E}[G_t \mid s_t, a_t]$
  - approximate b by $V(s_t) \equiv \mathbb{E}[G_t \mid s_t]$
  - $\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}[Q^\pi(s_t, a_t) - V^\pi(s_t)]\nabla \log \pi_\theta(a_t \mid s_t)$

    $= \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}[r_t + V^\pi(s_{t+1}) - V^\pi(s_t)]\nabla \log \pi_\theta(a_t \mid s_t)$

$$Q^\pi(s_t, a_t) = r_t + V^\pi(s_{t+1})$$

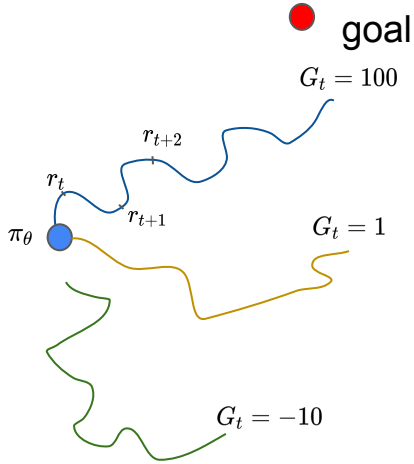
- Hung-Yi Lee, DRL Lecture: Policy Gradient, https://www.youtube.com/watch?v=z95ZYgPgXOY&t
- Mnih 2016, Asynchronous Methods for Deep Reinforcement Learning

Trick 3: Actor-Critic

- Problem: $\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}[G_t - b]\nabla \log \pi_\theta(a_t \mid s_t)$

high variance → unstable

$G_t = \sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'}^n$ discounted RTG



- Value function (Critic):
  - approximate G_t by $Q(s_t, a_t) \equiv \mathbb{E}[G_t \mid s_t, a_t]$
  - approximate b by $V(s_t) \equiv \mathbb{E}[G_t \mid s_t]$
  - $\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}[Q^\pi(s_t, a_t) - V^\pi(s_t)]\nabla \log \pi_\theta(a_t \mid s_t)$

$$= \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}[r_t + V^\pi(s_{t+1}) - V^\pi(s_t)]\nabla \log \pi_\theta(a_t \mid s_t)$$

$$Q^\pi(s_t, a_t) = r_t + V^\pi(s_{t+1})$$

Just need to train V

$$\phi' = \arg\min_\phi \frac{1}{N \cdot T}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}\left(V_\phi^\pi(s_t) - G_t\right)^2$$

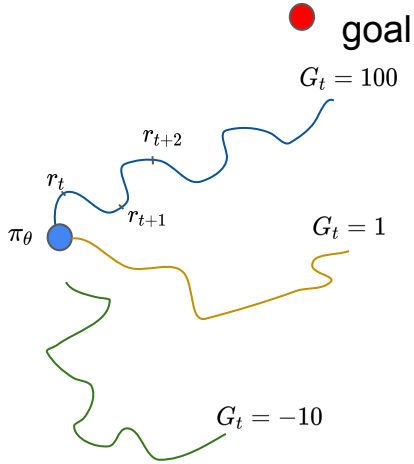
- Hung-Yi Lee, DRL Lecture: Policy Gradient, https://www.youtube.com/watch?v=z95ZYgPgXOY&t
- Mnih 2016, Asynchronous Methods for Deep Reinforcement Learning

Trick 3: Actor-Critic

- Problem: $\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}[G_t - b]\nabla \log \pi_\theta(a_t \mid s_t)$

high variance → unstable

$G_t = \sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'}^n$ discounted RTG



- Value function (Critic):
  - approximate G_t by $Q(s_t, a_t) \equiv \mathbb{E}[G_t \mid s_t, a_t]$
  - approximate b by $V(s_t) \equiv \mathbb{E}[G_t \mid s_t]$
  - $\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}[Q^\pi(s_t, a_t) - V^\pi(s_t)]\nabla \log \pi_\theta(a_t \mid s_t)$

advantage $A^\pi(s_t)$

$= \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}[r_t + V^\pi(s_{t+1}) - V^\pi(s_t)]\nabla \log \pi_\theta(a_t \mid s_t)$

$Q^\pi(s_t, a_t) = r_t + V^\pi(s_{t+1})$

Just need to train V

$\phi' = \arg\min_\phi \frac{1}{N \cdot T}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}\left(V_\phi^\pi(s_t) - G_t\right)^2$

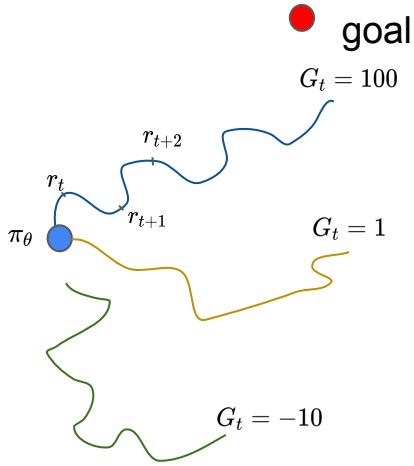
- Hung-Yi Lee, DRL Lecture: Policy Gradient, https://www.youtube.com/watch?v=z95ZYgPgXOY&t
- Mnih 2016, Asynchronous Methods for Deep Reinforcement Learning

Trick 3: Actor-Critic

- Problem: $\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}[G_t - b]\nabla \log \pi_\theta(a_t \mid s_t)$

high variance → unstable

$G_t = \sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'}^n$ discounted RTG



- Value function (Critic):
  - approximate G_t by $Q(s_t, a_t) \equiv \mathbb{E}[G_t \mid s_t, a_t]$
  - approximate b by $V(s_t) \equiv \mathbb{E}[G_t \mid s_t]$
  - $\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}[Q^\pi(s_t, a_t) - V^\pi(s_t)]\nabla \log \pi_\theta(a_t \mid s_t)$

advantage $A^\pi(s_t)$

$= \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}[r_t + V^\pi(s_{t+1}) - V^\pi(s_t)]\nabla \log \pi_\theta(a_t \mid s_t)$

$= \frac{1}{N}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1} A^\pi(s_t)\nabla \log \pi_\theta(a_t \mid s_t)$

where $A = Q - V$
$= r + V' - V$

$Q^\pi(s_t, a_t) = r_t + V^\pi(s_{t+1})$

Just need to train V

$\phi' = \arg\min_\phi \frac{1}{N \cdot T}\sum_{n=0}^{N-1}\sum_{t=0}^{T-1}\left(V_\phi^\pi(s_t) - G_t\right)^2$

- Hung-Yi Lee, DRL Lecture: Policy Gradient, https://www.youtube.com/watch?v=z95ZYgPgXOY&t
- Mnih 2016, Asynchronous Methods for Deep Reinforcement Learning

Trick 3: Actor-Critic

$Q(s,a)$ - $V(s) = \mathbb{E}_{\mathcal{A}}[Q(s,a)]$ = $A(s,a)$

Q(s,a)	a0	a1	a2
s0	8	-21	4
s1	-15	-10	1
s2	15	4	8

-

V(s)	
s0	-3
s1	-8
s2	9

=

A(s,a)	a0	a1	a2
s0	11	-18	7
s1	-7	-2	9
s2	6	-5	-1

Trick 3: Actor-Critic

$Q(s,a)$ - $V(s) = \mathbb{E}_{\mathcal{A}}[Q(s,a)]$ = $A(s,a)$

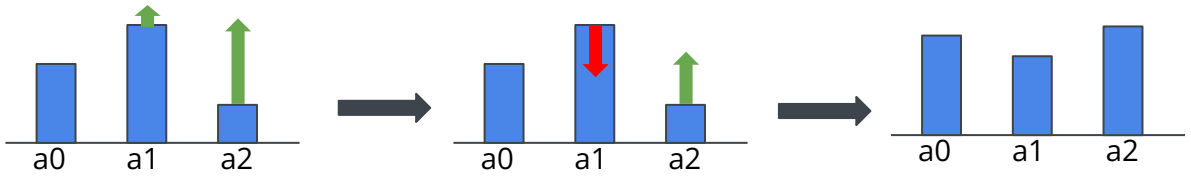
Q(s,a)	a0	a1	a2
s0	8	-21	4
s1	-15	-10	1
s2	15	4	8

-

V(s)	
s0	-3
s1	-8
s2	9

=

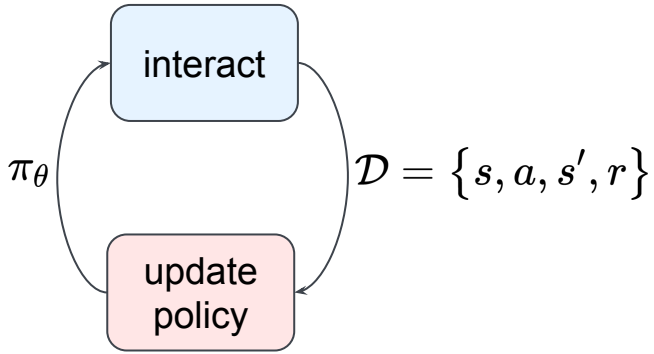
A(s,a)	a0	a1	a2
s0	11	-18	7
s1	-7	-2	9
s2	6	-5	-1



$Q(s,a)$ $A(s,a)$

Trick 3: Actor-Critic

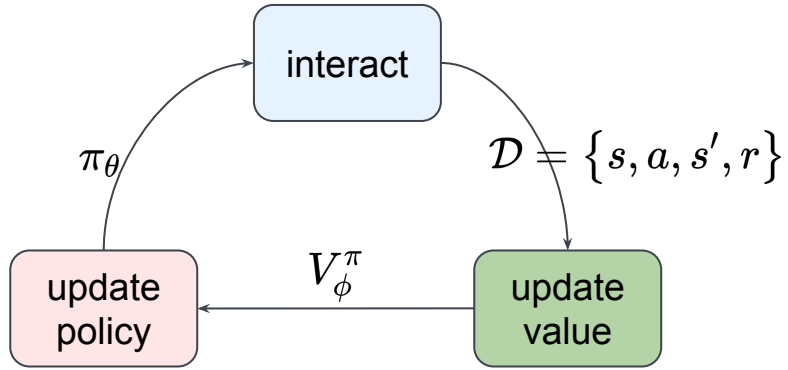
REINFORCE



$$J(\pi_\theta) \approx \frac{1}{N} \sum_{n=0}^{N-1} \sum_{t=0}^{T-1} R(\tau_n) \log \pi_\theta(a_t \mid s_t)$$

$$\theta' \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta)$$

A2C (Advantage Actor Critic)



$$A_\phi^\pi(s_t) = r + V_\phi^\pi(s_{t+1}) - V_\phi^\pi(s_t)$$

$$J(\pi_\theta) \approx \frac{1}{N} \sum_{n=0}^{N-1} \sum_{t=0}^{T-1} A_\phi^\pi(s_t) \log \pi_\theta(a_t \mid s_t)$$

$$\theta' \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta)$$

$$G_t = \sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'}^n$$

$$\mathcal{L} = \frac{1}{N \cdot T} \sum_{n=0}^{N-1} \sum_{t=0}^{T-1} \left(V_\phi^\pi(s_t) - G_t \right)^2$$

$$\phi' \leftarrow \phi - \alpha \nabla_\phi \mathcal{L}$$

- Hung-Yi Lee, DRL Lecture: Policy Gradient, https://www.youtube.com/watch?v=z95ZYgPgXOY&t
- Mnih 2016, Asynchronous Methods for Deep Reinforcement Learning

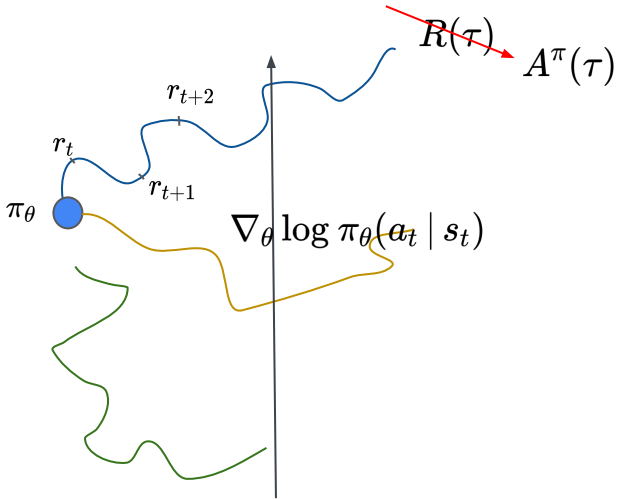
Trick 3: Actor-Critic

- REINFORCE objective
  - maximize **total reward**

    $J(\pi_\theta) = \mathbb{E}_{\pi,p}[R(\tau)]$

- A2C objective
  - maximize **total advantage**

    $J(\pi_\theta) = \mathbb{E}_{\pi,p}[A^\pi(\tau)]$



Q&A IV

Trick 4: Importance Sampling

- problem: A2C is an on-policy method
  - on-policy: only use self-collected data → cannot use old data to update policy
  - off-policy: use data collected by others → can use old data to update policy
- Importance Sampling: on-policy → off-policy
  - $\mathbb{E}_{x\sim p}[f(x)] \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i)$

- Hung-Yi Lee, DRL Lecture: PPO, https://www.youtube.com/watch?v=OAKAZhFmYoI

Trick 4: Importance Sampling

- problem: A2C is an on-policy method
  - on-policy: only use self-collected data → cannot use old data to update policy
  - off-policy: use data collected by others → can use old data to update policy
- Importance Sampling: on-policy → off-policy
  - $$\mathbb{E}_{x\sim p}[f(x)] \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$
    $$= \int f(x)p(x)dx = \int f(x)\frac{p(x)}{q(x)}q(x)dx = \mathbb{E}_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right]$$

    importance sampling ratio

- Hung-Yi Lee, DRL Lecture: PPO, https://www.youtube.com/watch?v=OAKAZhFmYoI

Trick 4: Importance Sampling

- problem: A2C is an on-policy method
  - on-policy: only use self-collected data → cannot use old data to update policy
  - off-policy: use data collected by others → can use old data to update policy
- Importance Sampling: on-policy → off-policy
  - $$\mathbb{E}_{x\sim p}[f(x)] \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$
    $$= \int f(x)p(x)dx = \int f(x)\frac{p(x)}{q(x)}q(x)dx = \mathbb{E}_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right]$$
    importance sampling ratio
  - $$\nabla_\theta J(\pi_\theta) = \mathbb{E}_\pi\left[\sum_{t=0}^{T-1} A_\phi^\pi(s_t)\nabla \log \pi_\theta(a_t \mid s_t)\right]$$
    $$= \mathbb{E}_{\pi^{old}}\left[\sum_{t=0}^{T-1} A_\phi^{\pi^{old}}(s_t)\nabla \log \pi_\theta(a_t \mid s_t)\frac{\pi_\theta(a_t \mid s_t)}{\pi_\theta^{old}(a_t \mid s_t)}\right]$$

- Hung-Yi Lee, DRL Lecture: PPO, https://www.youtube.com/watch?v=OAKAZhFmYoI

Trick 4: Importance Sampling

- problem: A2C is an on-policy method
  - on-policy: only use self-collected data → cannot use old data to update policy
  - off-policy: use data collected by others → can use old data to update policy
- Importance Sampling: on-policy → off-policy
  - $$\mathbb{E}_{x\sim p}[f(x)] \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$
    $$= \int f(x)p(x)dx = \int f(x)\frac{p(x)}{q(x)}q(x)dx = \mathbb{E}_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right]$$
    importance sampling ratio
  - $$\nabla_\theta J(\pi_\theta) = \mathbb{E}_\pi\left[\sum_{t=0}^{T-1} A_\phi^\pi(s_t)\nabla \log \pi_\theta(a_t \mid s_t)\right]$$
    $$= \mathbb{E}_{\pi^{old}}\left[\sum_{t=0}^{T-1} A_\phi^{\pi^{old}}(s_t)\nabla \log \pi_\theta(a_t \mid s_t)\frac{\pi_\theta(a_t \mid s_t)}{\pi_\theta^{old}(a_t \mid s_t)}\right]$$
    $$= \mathbb{E}_{\pi^{old}}\left[\sum_{t=0}^{T-1} A_\phi^{\pi^{old}}(s_t)\frac{\nabla \pi_\theta(a_t \mid s_t)}{\pi_\theta^{old}(a_t \mid s_t)}\right]$$

$$\nabla f(x) = f(x)\nabla \log f(x)$$

- Hung-Yi Lee, DRL Lecture: PPO, https://www.youtube.com/watch?v=OAKAZhFmYoI

Trick 4: Importance Sampling

- problem: A2C is an on-policy method
  - on-policy: only use self-collected data → cannot use old data to update policy
  - off-policy: use data collected by others → can use old data to update policy
- Importance Sampling: on-policy → off-policy
  - $$\mathbb{E}_{x\sim p}[f(x)] \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$
    $$= \int f(x)p(x)dx = \int f(x)\frac{p(x)}{q(x)}q(x)dx = \mathbb{E}_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right]$$
    importance sampling ratio
  - $$\nabla_\theta J(\pi_\theta) = \mathbb{E}_\pi\left[\sum_{t=0}^{T-1} A_\phi^\pi(s_t)\nabla\log\pi_\theta(a_t\mid s_t)\right]$$
    $$= \mathbb{E}_{\pi^{old}}\left[\sum_{t=0}^{T-1} A_\phi^{\pi^{old}}(s_t)\nabla\log\pi_\theta(a_t\mid s_t)\frac{\pi_\theta(a_t\mid s_t)}{\pi_\theta^{old}(a_t\mid s_t)}\right]$$
    $$= \mathbb{E}_{\pi^{old}}\left[\sum_{t=0}^{T-1} A_\phi^{\pi^{old}}(s_t)\frac{\nabla\pi_\theta(a_t\mid s_t)}{\pi_\theta^{old}(a_t\mid s_t)}\right]$$
    $$\nabla f(x) = f(x)\nabla\log f(x)$$
  - objective: $$J^{IS}(\pi_\theta) = \mathbb{E}_{\pi^{old}}\left[\sum_{t=0}^{T-1} A^{\pi^{old}}(s_t)\frac{\pi_\theta(a_t\mid s_t)}{\pi_\theta^{old}(a_t\mid s_t)}\right]$$

- Hung-Yi Lee, DRL Lecture: PPO, https://www.youtube.com/watch?v=OAKAZhFmYoI

Trick 4: Importance Sampling

- problem: A2C is an on-policy method
  - on-policy: only use self-collected data → cannot use old data to update policy
  - off-policy: use data collected by others → can use old data to update policy
- Importance Sampling: on-policy → off-policy
  - $$\mathbb{E}_{x\sim p}[f(x)] \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$
    $$= \int f(x)p(x)dx = \int f(x)\frac{p(x)}{q(x)}q(x)dx = \mathbb{E}_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right]$$
    importance sampling ratio
  - $$\nabla_\theta J(\pi_\theta) = \mathbb{E}_\pi\left[\sum_{t=0}^{T-1} A_\phi^\pi(s_t)\nabla\log\pi_\theta(a_t\mid s_t)\right]$$
    $$= \mathbb{E}_{\pi^{old}}\left[\sum_{t=0}^{T-1} A_\phi^{\pi^{old}}(s_t)\nabla\log\pi_\theta(a_t\mid s_t)\frac{\pi_\theta(a_t\mid s_t)}{\pi_\theta^{old}(a_t\mid s_t)}\right]$$
    $$= \mathbb{E}_{\pi^{old}}\left[\sum_{t=0}^{T-1} A_\phi^{\pi^{old}}(s_t)\frac{\nabla\pi_\theta(a_t\mid s_t)}{\pi_\theta^{old}(a_t\mid s_t)}\right]$$
    $$\nabla f(x) = f(x)\nabla\log f(x)$$
  - objective: $$J^{IS}(\pi_\theta) = \mathbb{E}_{\pi^{old}}\left[\sum_{t=0}^{T-1} A^{\pi^{old}}(s_t)\frac{\pi_\theta(a_t\mid s_t)}{\pi_\theta^{old}(a_t\mid s_t)}\right]$$

With IS, we can
- use old data
- update policy multiple times

- Hung-Yi Lee, DRL Lecture: PPO, https://www.youtube.com/watch?v=OAKAZhFmYoI

Trick 5: KL Penalty

- Problem: Importance Sampling has high variance

$$\mathbb{E}_{x\sim p}[f(x)] = \mathbb{E}_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right]$$

$$Var_{x\sim p}[f(x)] = \mathbb{E}_{x\sim p}\left[f(x)^2\right] - \left(\mathbb{E}_{x\sim p}[f(x)]\right)^2$$

$$\begin{aligned} Var_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right] &= \mathbb{E}_{x\sim q}\left[\left(f(x)\frac{p(x)}{q(x)}\right)^2\right] - \left(\mathbb{E}_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right]\right)^2 \\ &= \mathbb{E}_{x\sim p}\left[f(x)^2\frac{p(x)}{q(x)}\right] - \left(\mathbb{E}_{x\sim p}[f(x)]\right)^2 \end{aligned}$$

- Hung-Yi Lee, DRL Lecture: PPO, https://www.youtube.com/watch?v=OAKAZhFmYoI

Trick 5: KL Penalty

- Problem: Importance Sampling has high variance

$$\mathbb{E}_{x\sim p}[f(x)] = \mathbb{E}_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right]$$

$$Var_{x\sim p}[f(x)] = \mathbb{E}_{x\sim p}\left[f(x)^2\right] - (\mathbb{E}_{x\sim p}[f(x)])^2$$

$$\begin{aligned} Var_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right] &= \mathbb{E}_{x\sim q}\left[\left(f(x)\frac{p(x)}{q(x)}\right)^2\right] - \left(\mathbb{E}_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right]\right)^2 \\ &= \mathbb{E}_{x\sim p}\left[f(x)^2\frac{p(x)}{q(x)}\right] - (\mathbb{E}_{x\sim p}[f(x)])^2 \end{aligned}$$

when p and q are different → introduce variance

- PPO (proximal policy optimization)

$$J^{PPO}(\pi_\theta) = J^{IS}(\pi_\theta) - \beta D_{KL}(\pi_\theta, \pi_\theta^{old})$$

penalty for the distance between new policy and old policy distribution

- Hung-Yi Lee, DRL Lecture: PPO, https://www.youtube.com/watch?v=OAKAZhFmYoI

Trick 5: KL Penalty

$$J^{PPO}(\pi_\theta) = J^{IS}(\pi_\theta) - \beta D_{KL}(\pi_\theta, \pi_\theta^{old})$$

tricky to implement

$$J^{IS}(\pi_\theta) \approx \sum_{(s_t,a_t)} A^{\pi^{old}}(s_t) \frac{\pi_\theta(a_t \mid s_t)}{\pi_\theta^{old}(a_t \mid s_t)}$$

$$J^{PPO2}(\pi_\theta) \approx \sum_{(s_t,a_t)} \min\left(A^{\pi^{old}} \frac{\pi}{\pi_{old}}, A^{\pi^{old}} \cdot clip\left(\frac{\pi}{\pi_{old}}, 1-\epsilon, 1+\epsilon \right) \right)$$

1. clip the ratio to a small range
2. choose a smaller one

- Hung-Yi Lee, DRL Lecture: PPO, https://www.youtube.com/watch?v=OAKAZhFmYoI

Trick 6: GAE Lambda

- Problem:

- GAE Lambda

- Schulman et al., HIGH-DIMENSIONAL CONTINUOUS CONTROL USING GENERALIZED ADVANTAGE ESTIMATION

Topics

- Introduction to Reinforcement Learning
- **PPO algorithm + hands-on**
  - Deep Learning
  - Quick Fact about PPO
  - Policy Gradient
  - Tricks
  - **PPO = Policy Gradient + Tricks**
- Blimp Control with Residual RL

Summary

- PPO = policy gradient + baseline + RTG + actor-critic + importance sampling + KL penalty + (GAE-Lambda)

Algorithm 1 PPO-Clip (= PPO2)

1: Input: initial policy parameters θ_0, initial value function parameters ϕ_0
2: **for** $k = 0, 1, 2, \ldots$ **do**
3: Collect set of trajectories $\mathcal{D}_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment.
4: Compute rewards-to-go $\hat{R}_t$.
5: Compute advantage estimates, $\hat{A}_t$ (using any method of advantage estimation) based on the current value function V_{ϕ_k}.
6: Update the policy by maximizing the PPO-Clip objective:

$$\theta_{k+1} = \arg\max_{\theta} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \min\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} A^{\pi_{\theta_k}}(s_t, a_t), \quad g(\epsilon, A^{\pi_{\theta_k}}(s_t, a_t)) \right),$$

typically via stochastic gradient ascent with Adam.
7: Fit value function by regression on mean-squared error:

$$\phi_{k+1} = \arg\min_{\phi} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \left(V_\phi(s_t) - \hat{R}_t \right)^2,$$

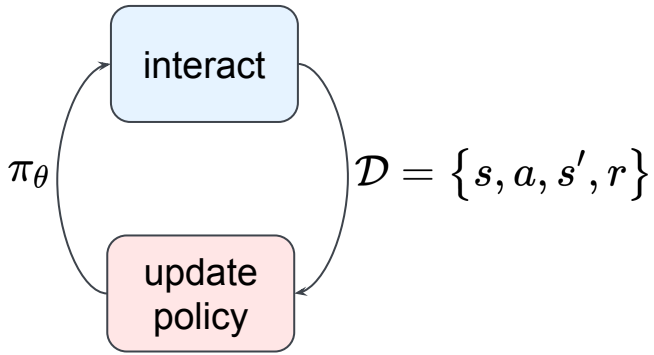
typically via some gradient descent algorithm.
8: **end for**

- spinningup, https://spinningup.openai.com/en/latest/algorithms/ppo.html

Summary

REINFORCE

$$J(\pi_\theta) = \mathbb{E}_{\pi,p}[R(\tau)]$$

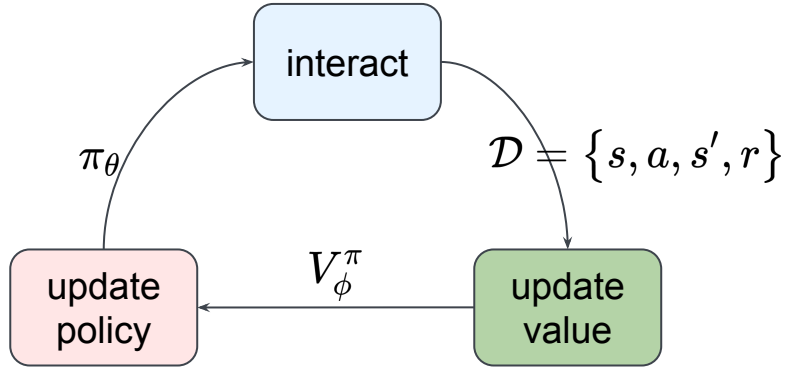


$$J(\pi_\theta) \approx \frac{1}{N} \sum_{n=0}^{N-1} \sum_{t=0}^{T-1} R(\tau_n) \log \pi_\theta(a_t \mid s_t)$$

$$\theta' \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta)$$

PPO2

$$J(\pi_\theta) = \mathbb{E}_{\pi^{old},p(\tau)} \left[\hat{A}(s_t) \frac{\pi_\theta(a_t \mid s_t)}{\pi_\theta^{old}(a_t \mid s_t)} \right] - \beta D_{KL}\left(\pi_\theta, \pi_\theta^{old}\right)$$



repeat K times

$$A_\phi^\pi(s_t) = r + V_\phi^\pi(s_{t+1}) - V_\phi^\pi(s_t)$$

$$J(\pi_\theta) \approx \sum_{(s_t,a_t)} \min\left(A^{\pi^{old}} \frac{\pi}{\pi_{old}}, A^{\pi^{old}} \cdot clip\left(\frac{\pi}{\pi_{old}}, 1-\epsilon, 1+\epsilon \right) \right)$$

$$\theta' \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta)$$

$$G_t = \sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'}^n$$

$$\mathcal{L} = \frac{1}{N \cdot T} \sum_{n=0}^{N-1} \sum_{t=0}^{T-1} \left(V_\phi^\pi(s_t) - G_t \right)^2$$

$$\phi' \leftarrow \phi - \alpha \nabla_\phi \mathcal{L}$$

- Hung-Yi Lee, DRL Lecture: PPO, https://www.youtube.com/watch?v=OAKAZhFmYoI
- Schulman 2017, Proximal Policy Optimization Algorithms

Q&A V

Hands-On: PPO tutorial

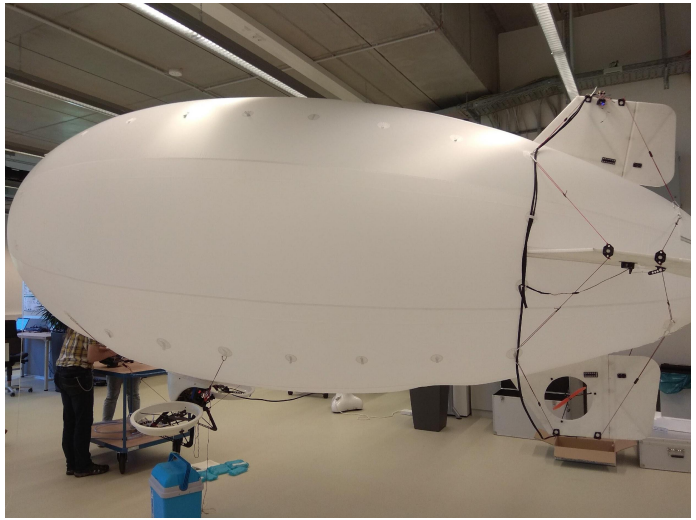
https://github.com/Ootang2019/deepfield2023

Topics

- Introduction to Reinforcement Learning
- PPO algorithm + hands-on
- **Blimp Control with Residual RL**

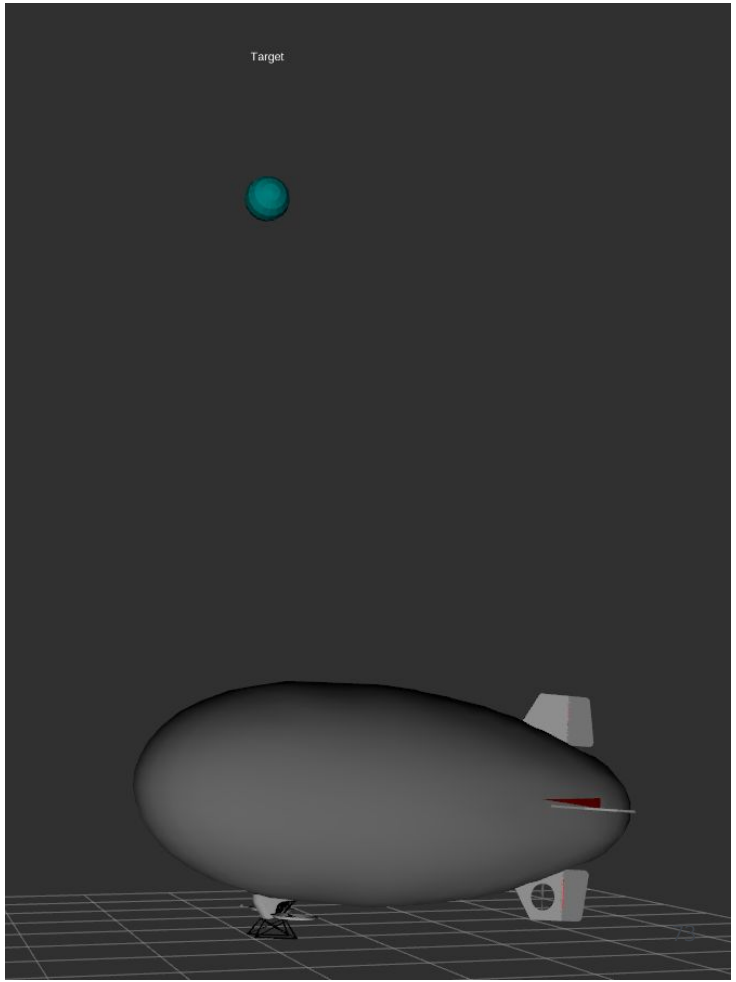
Compare to Copters

- Pros
  - Long Endurance
  - Safe
- Cons
  - Deformation
  - Time Delay
  - Disturbance
  - Complex Dynamics (nonlinearity, coupling)



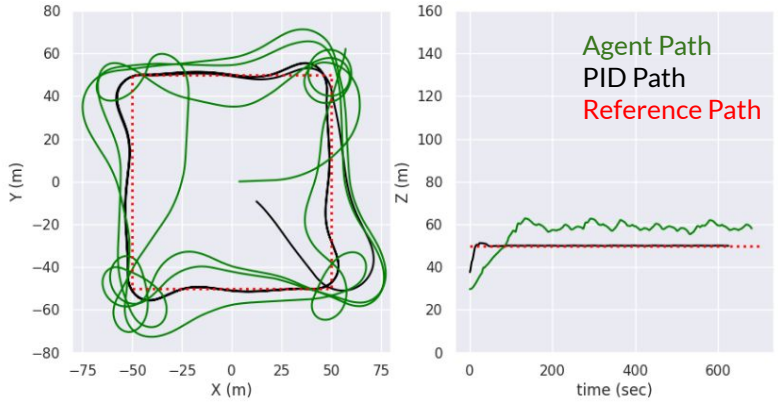
RL formulation

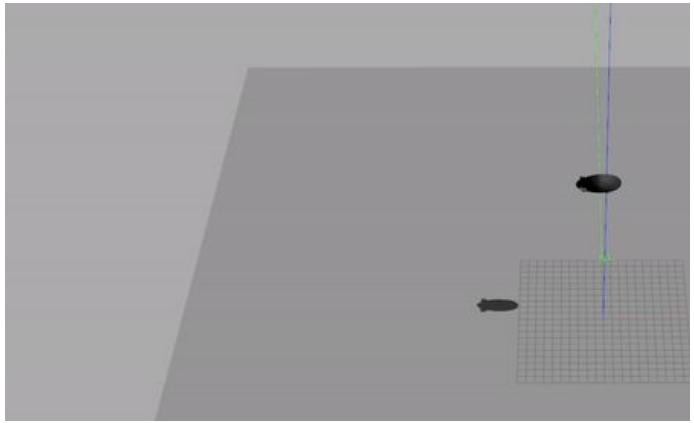
- Observation Space (position, velocity, orientation, angVel,...)
  - $s_t : \{p, v, \psi, w\}$
- Action Space (motor0, motor1, motor2, servo, fin0, fin1,.....)
  - $a_t : \{m0, m1, m2, s, f0, f1, f2, f3\}$
- Control Law, or policy
  - $\pi_\theta(a|x)$
- Target Space (position, velocity, orientation, angVel,...)
  - $g_t : \{p, v, \psi, w\}$
- Reward
  - $r_t = -|s_t - g_t|$
- Solve optimization problem by RL algorithm:
  - $max_{\pi_\theta} E[\sum_t \gamma^t r_t]$
  - where γ is a discount factor



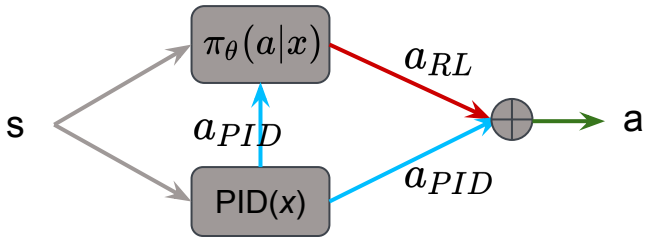
DRL issues

- significant training time
  - QRDQN agent 7 days
  - mediocre performance
- training stability
  - only half of the agents can find a working solution
- unexpected behavior
  - hover above reference
  - backward flight
- robustness
  - cannot deal with wind disturbance
  - altitude control impaired by buoyancy change

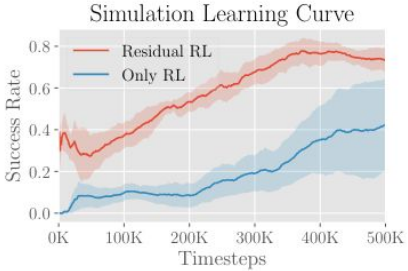


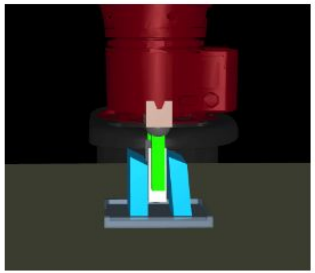


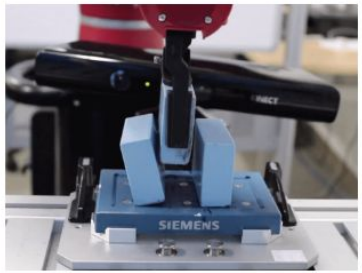
Residual Reinforcement Learning

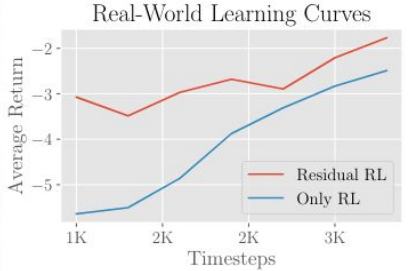


1. stability:
   a. agent action is extra disturbance to PID
   b. stability property depend on mixers
2. guidance:
   a. constraint policy search space
   b. start exploration in high reward region







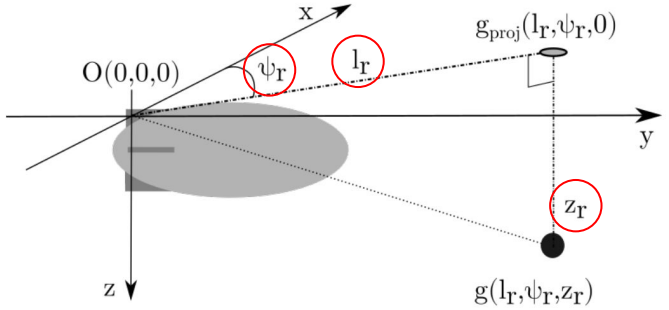


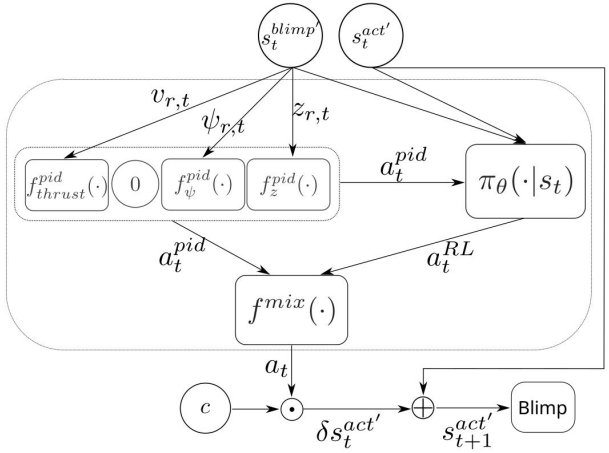
- Tobias Johannink, ICRA2019, Residual Reinforcement Learning for Robot Control
- Tom Staessens, TIE2022, Adaptive control of a mechatronic system using constrained residual reinforcement learning

DRRL formulation

- state: $s_t^{blimp} = (l_r, \psi_r, z_r, v_r, v_o, w)_t.$

$$s_t^{blimp'} = (s^{blimp}, \omega_\psi, v_{air}, \psi'_r)_t.$$

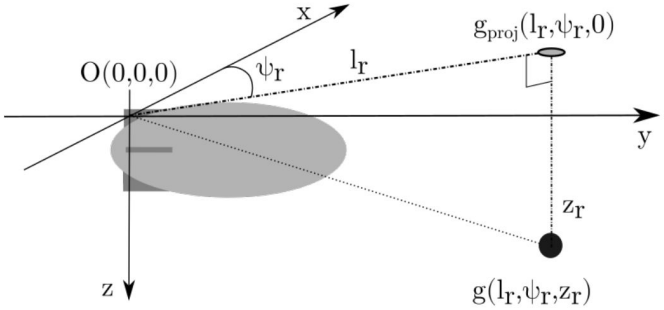


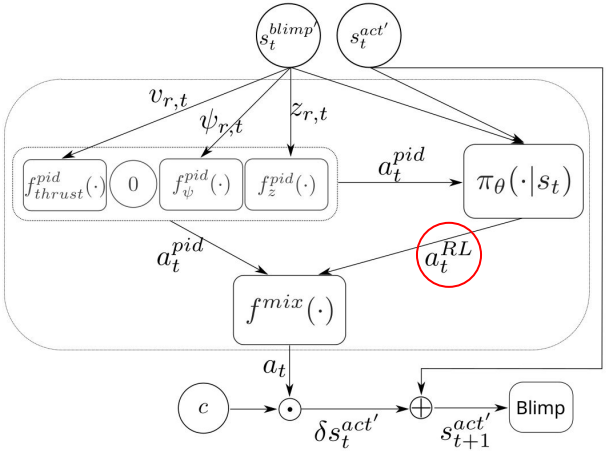


DRRL formulation

- state: $s_t^{blimp} = (l_r, \psi_r, z_r, v_r, v_o, w)_t.$

  $s_t^{blimp'} = (s^{blimp}, \omega_\psi, v_{air}, \psi'_r)_t.$
- action: $a_t^{RL} = (m_1, n_0, f_{(0,2)})_t$





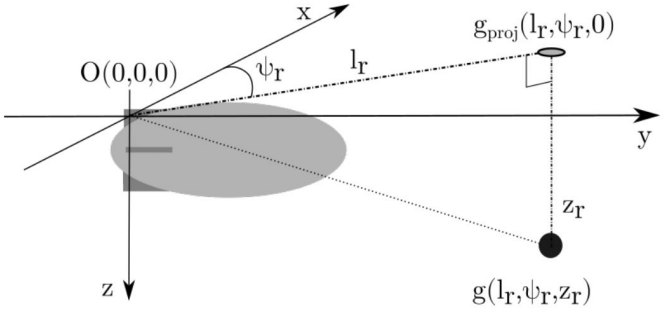
DRRL formulation

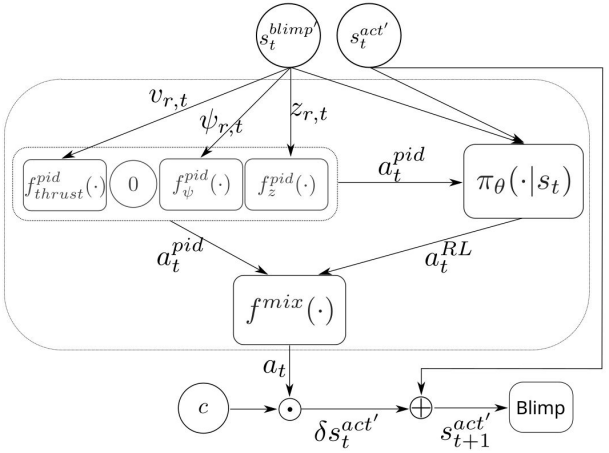
- state: $s_t^{blimp} = (l_r, \psi_r, z_r, v_r, v_o, w)_t.$

  $s_t^{blimp'} = (s^{blimp}, \omega_\psi, v_{air}, \psi'_r)_t.$
- action: $a_t^{RL} = (m_1, n_0, f_{(0,2)})_t$
- reward: $r_t = w_0 r_t^{success} + w_1 r_t^{track} + w_2 r_t^{act} + w_3 r_t^{bonus}$

$$r_t^{success} = \begin{cases} 1 & \text{if } d(s_{blimp}, s_{target}) \leq \epsilon \\ 0 & \text{otherwise} \end{cases}$$

$$r_t^{track} = -i_0|z_r| - i_1|l_r| - i_2|\psi_r| - i_3|v_r|,$$
$$r_t^{act} = -j_0|m_0| - j_1|m_1| - j_1|m_2|,$$
$$r_t^{bonus} = -k_0|\psi'_r|/(1 + l_r),$$





DRRL formulation

- state: $s_t^{blimp} = (l_r, \psi_r, z_r, v_r, v_o, w)_t.$

  $s_t^{blimp'} = (s^{blimp}, \omega_\psi, v_{air}, \psi'_r)_t.$
- action: $a_t^{RL} = (m_1, n_0, f_{(0,2)})_t$
- reward: $r_t = w_0 r_t^{success} + w_1 r_t^{track} + w_2 r_t^{act} + w_3 r_t^{bonus}$

  $$r_t^{success} = \begin{cases} 1 & \text{if } d(s_{blimp}, s_{target}) \leq \epsilon \\ 0 & \text{otherwise} \end{cases}$$

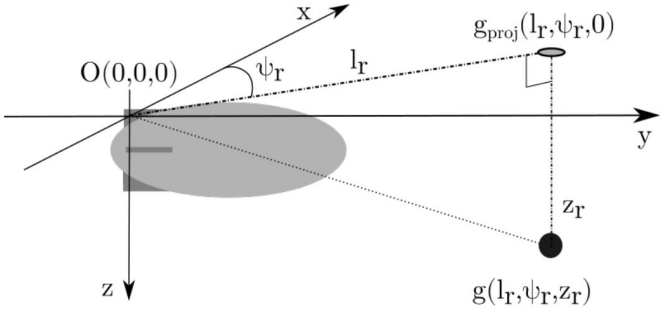
  $r_t^{track} = -i_0|z_r| - i_1|l_r| - i_2|\psi_r| - i_3|v_r|,$

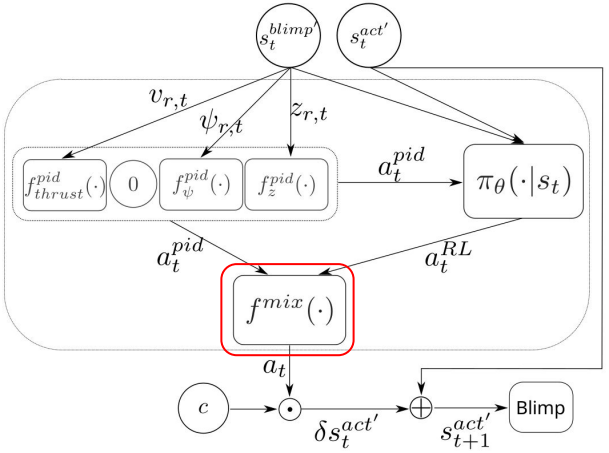
  $r_t^{act} = -j_0|m_0| - j_1|m_1| - j_1|m_2|,$

  $r_t^{bonus} = -k_0|\psi'_r|/(1 + l_r),$
- mixer: $f_{abs}^{mix}(x, y) = (1 - \beta)x + \beta y$

  $f_{rel}^{mix}(x, y) = x(1 + \beta y)$

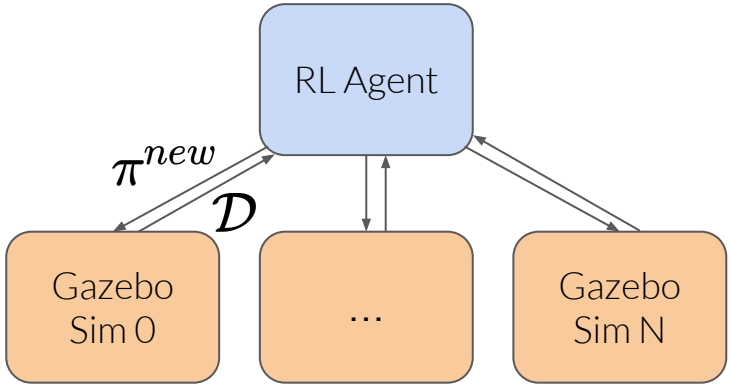
  $f_{hyb}^{mix}(x, y) = (1 - \alpha) f_{abs}^{mix}(x, y) + \alpha f_{rel}^{mix}(x, y),$

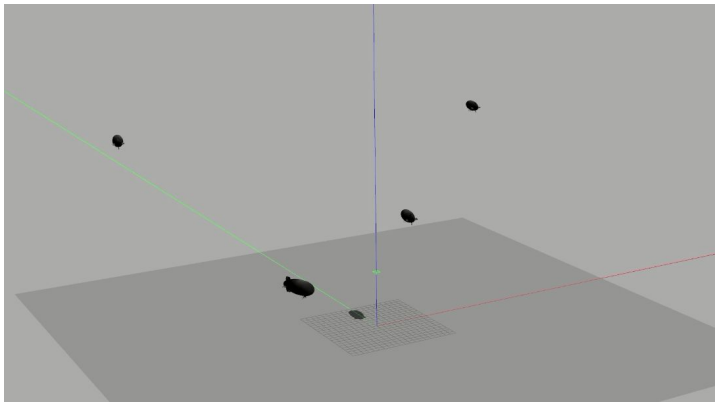




RL training in robotics

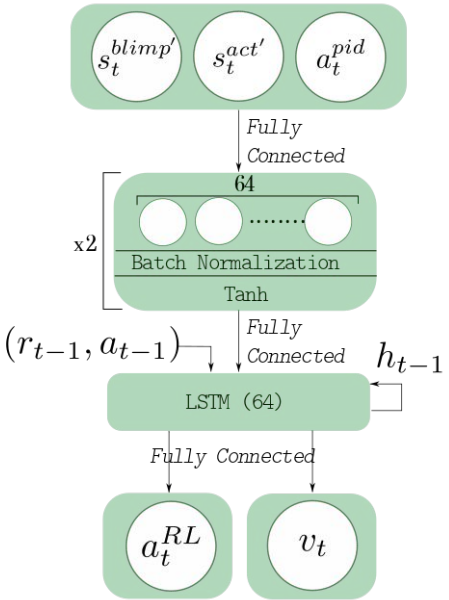
- RL require significant amount of data
  - parallelize simulation
- Partial observation
  - LSTM policy
- Sim-to-Real transfer
  - domain randomization
  - e.g. wind field, buoyancy, waypoint sequence
- Jerky command, or chattering
  - accumulative action space





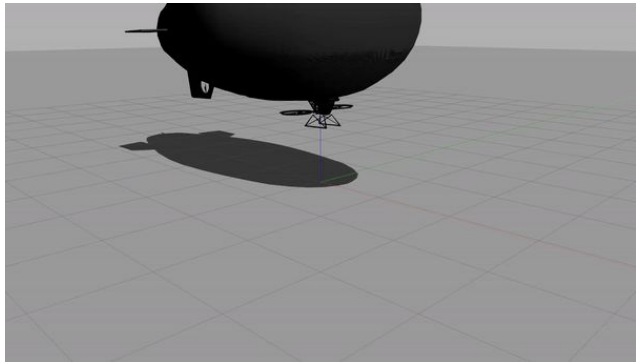
RL training in robotics

- RL require significant amount of data
  - parallelize simulation
- Partial observation
  - LSTM policy
- Sim-to-Real transfer
  - domain randomization
  - e.g. wind field, buoyancy, waypoint sequence
- Jerky command, or chattering
  - accumulative action space



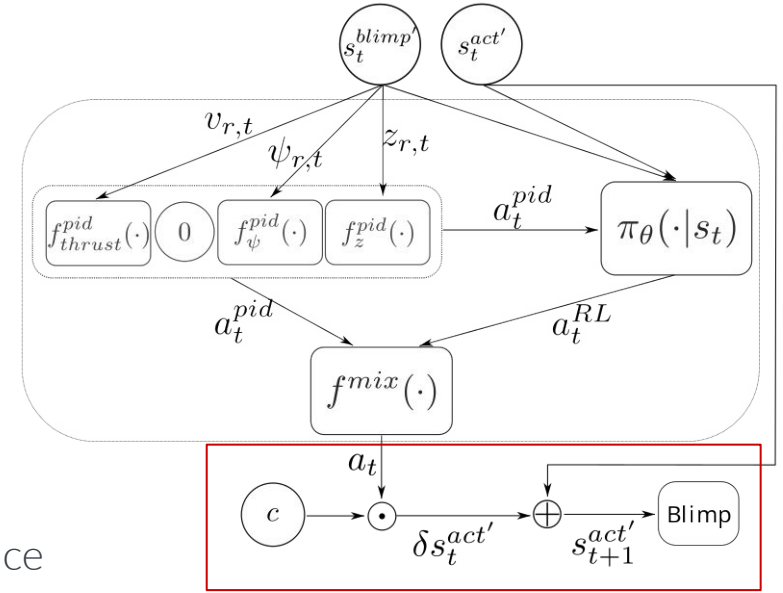
RL training in robotics

- RL require significant amount of data
  - parallelize simulation
- Partial observation
  - LSTM policy
- Sim-to-Real transfer
  - domain randomization
  - e.g. wind field, buoyancy, waypoint sequence
- Jerky command, or chattering
  - accumulative action space



RL training in robotics

- RL require significant amount of data
  - parallelize simulation
- Partial observation
  - LSTM policy
- Sim-to-Real transfer
  - domain randomization
  - e.g. wind field, buoyancy, waypoint sequence
- Jerky command, or chattering
  - accumulative action space

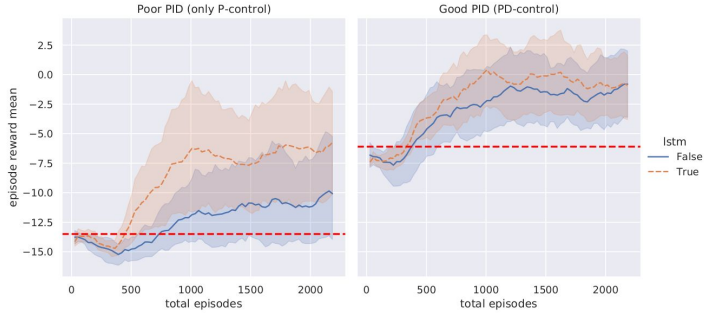


e.g.
when agent command *a=1*
- x *100%* throttle
- o *(x+10*a)%* throttle

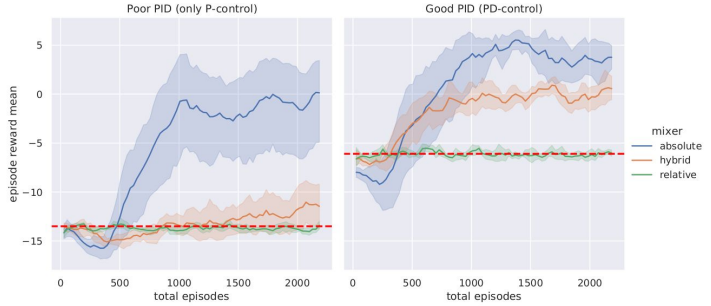
Training using PPO

Training progress

Simpler Task: *Yaw Control* (1 day)

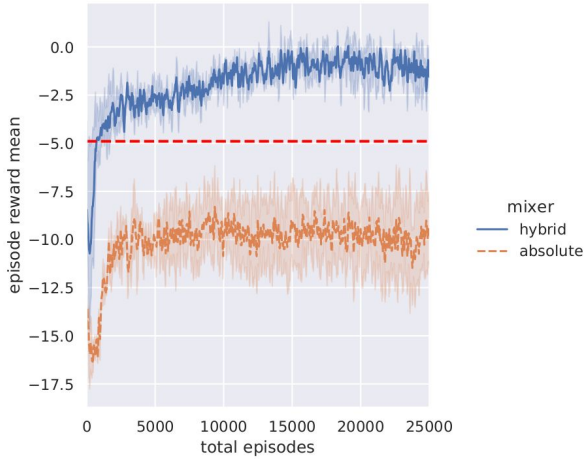


(a) LSTM policy stabilizes and accelerates the training progress and reduces the performance drop.



(b) Absolute mixer is the most aggressive as it provides the most performance growth as well as performance drop during exploration, and vice versa.

Complex Task: *Blimp Control* (28 days)



1. lstm layer is critical for training stability
2. hybrid mixer scales better with task complexity
3. DRRL agent reaches 60% final performance in 2 days (~3.5 hours in parallel simulation)

Testing

square trajectory

- long edge ~80m
- sharp corner
- no ascend

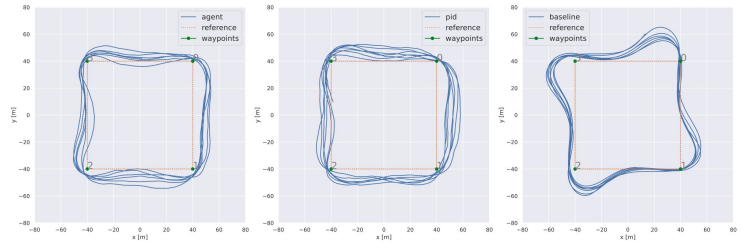
coil trajectory

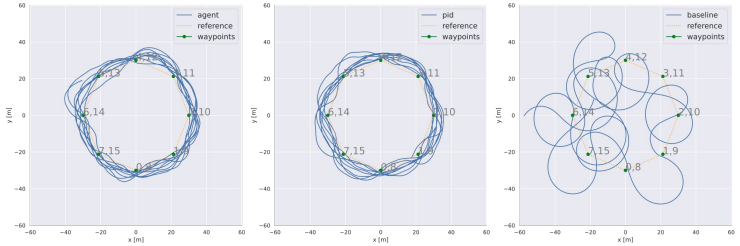
- short edge ~22.5m
- great ascend ~5m

Robustness Test

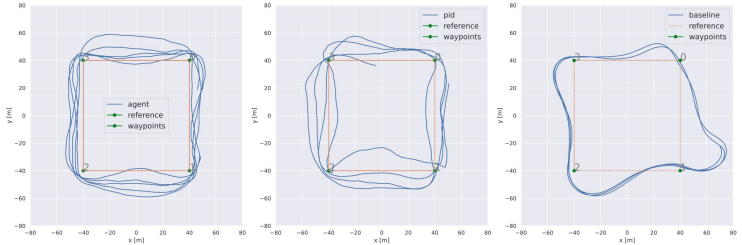
DRRL(Ours) PID Baseline DRRL(Ours) PID Baseline

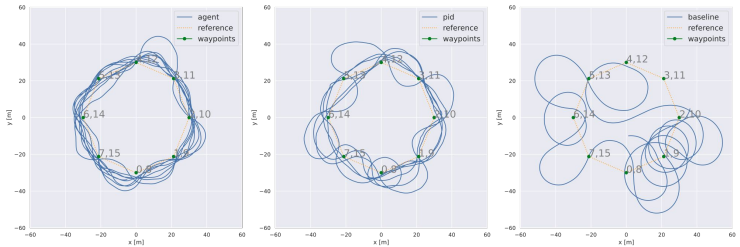
No wind



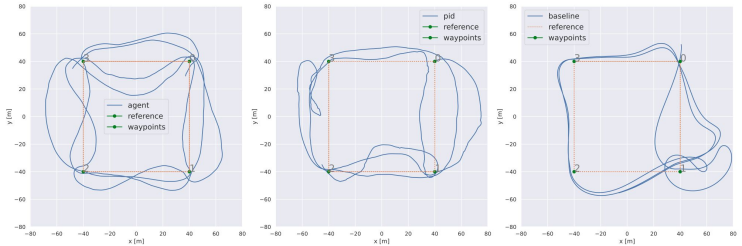


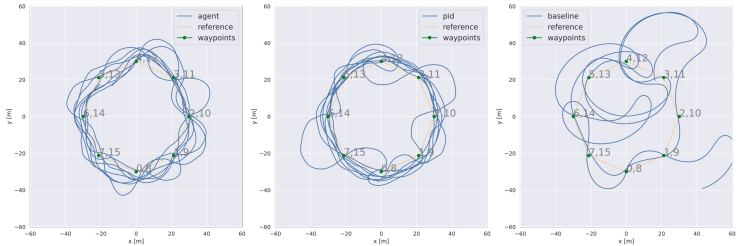
0.5m/s wind





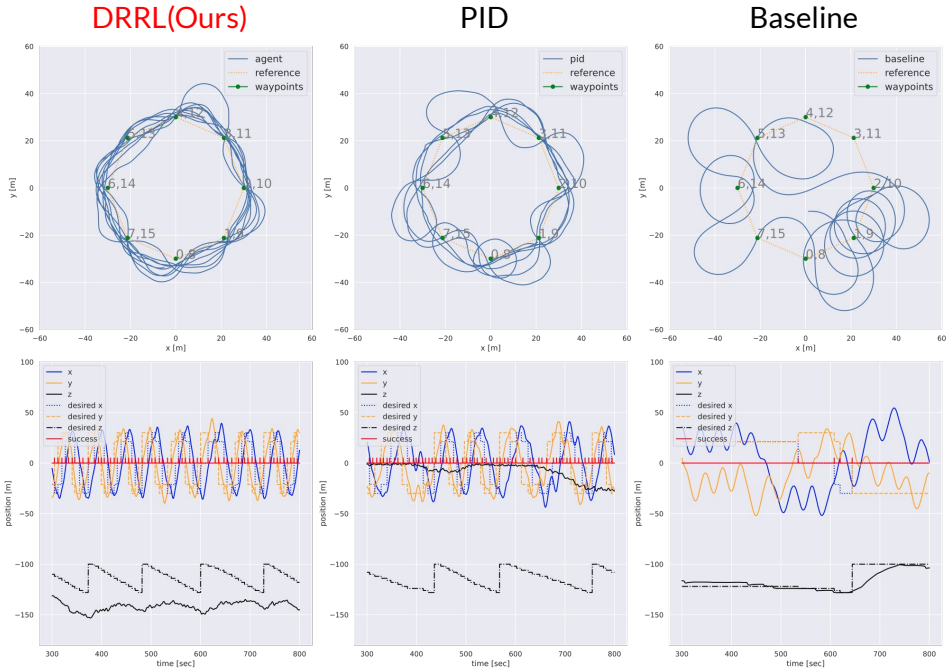
1.0m/s wind





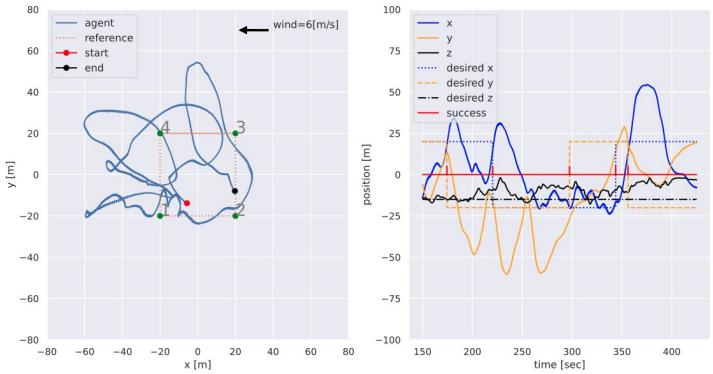
Robustness Test

- Coil Trajectory
  - top row:
    planar trajectory
  - bottom row black lines:
    altitude trajectory
- Reason
  - maintaining altitude require velocity → turning reduces velocity → altitude losses
  - **PID** only focus on planar tracking and cannot hold altitude at all
  - **Baseline** prevents altitude loss by limiting the turn radius



Real-World Test

1. successfully complete a square
2. under 6m/s wind
3. huge sim-to-real gap: e.g.
    a. max speed of the blimp:
        i. simulation ~2m/s
        ii. real-world ~8m/s



Conclusion

- The first data-driven approach to blimp control in the real world
  - DRRL: PID controller + PPO agent
    - accelerate training
    - safe exploration
    - controller design process remain model-free
  - Challenges
    - training data gathering → parallel simulation
    - sim-to-real gap → domain randomization
    - partial observation → LSTM policy
    - jerky actuator command → accumulative action space

Future Work

- increase sample efficiency
  - multi-task learning → Liu 2023, Multi-Task Reinforcement Learning in Continuous Control with Successor Feature-Based Concurrent Composition
  - higher sample efficiency RL agent, e.g. DSAC, REDQ, MBRL
- increase robustness
  - using a robust controller as base policy, e.g. H-inf → Yang and Liu 2023, Autonomous Blimp Control via H-infinity Robust Deep Residual Reinforcement Learning
- increase long-term performance
  - the current agent is still quite short-sighted and did not plan very well
    - e.g. learn to reduce overshoot without explicit reward term
- increase memory and context identification ability
  - lstm memory is relatively short
  - meta-learning / adaptation
    - e.g. learn to identify different wind context and adapt

Q&A VI

Take Away:

Don’t do RL

from scratch

Thank You!

Yu-Tang Liu

Aamir Ahmad

University of Stuttgart
Institute of Flight Mechanics and Controls

Flight Robotics and Perception Group

PERCEIVING SYSTEMS
MAX PLANCK INSTITUTE FOR INTELLIGENT SYSTEMS

Source code https://github.com/robot-perception-group/AutonomousBlimpDRL