Reinforcement learning

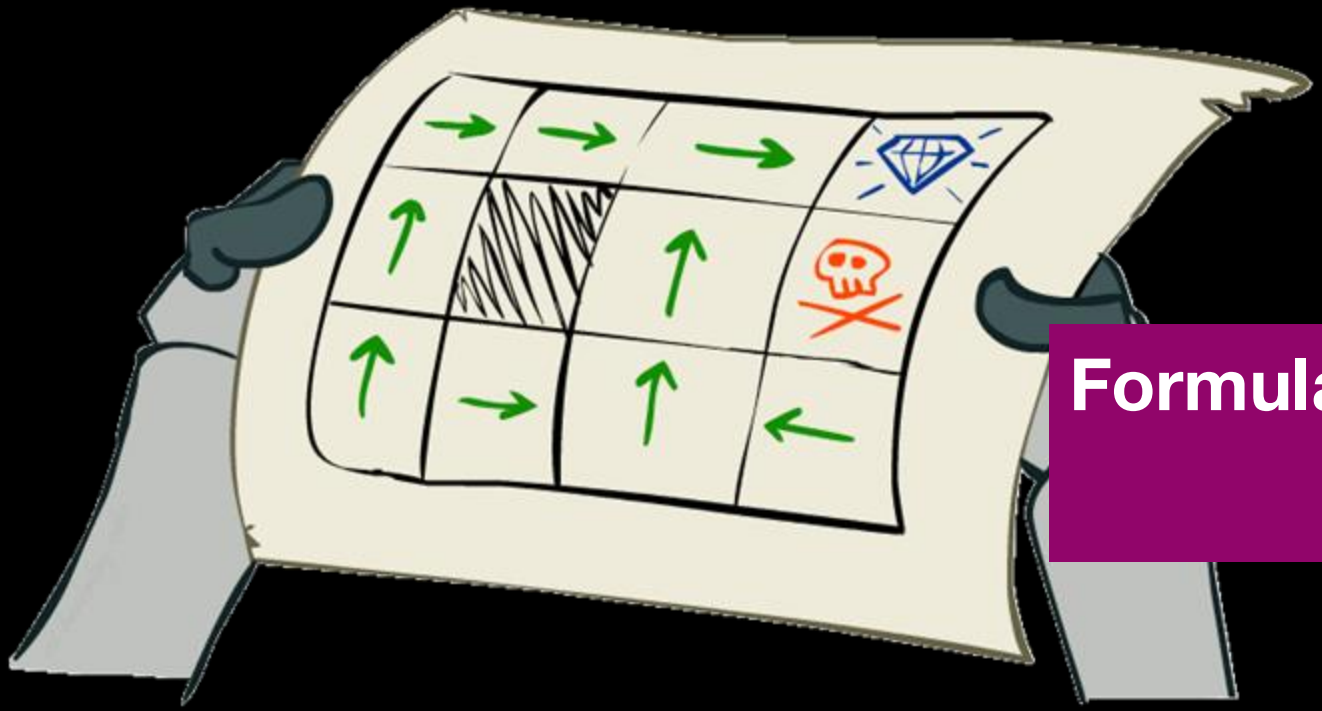
Chenyu Yang
Robert Katzschmann

Soft Robotics Lab 27/10/2025

Toshimitsu et al., Getting the ball rolling, Humanoids (2023)

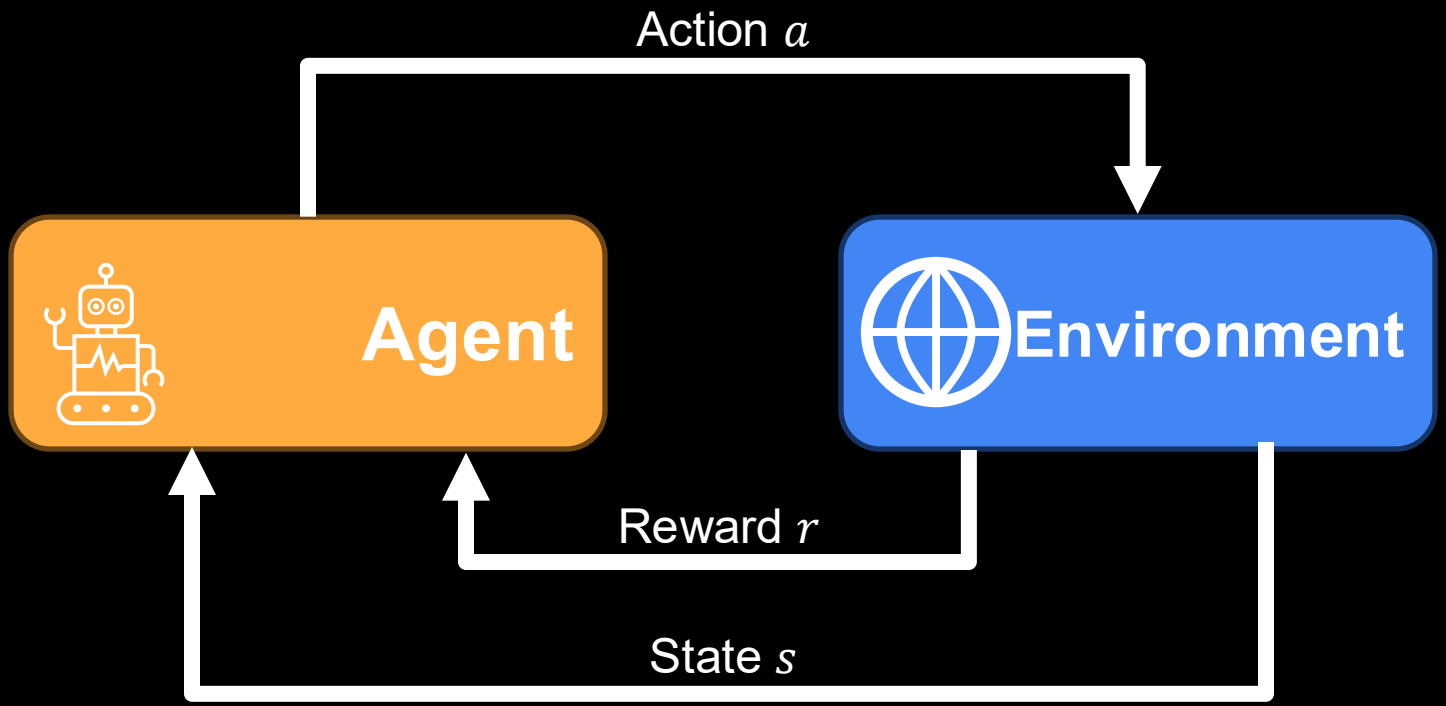
Plan for Today

- Formulation
- Algorithms
- Practice in Robotics



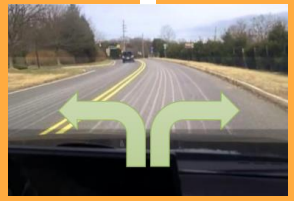
Formulation

Definitions

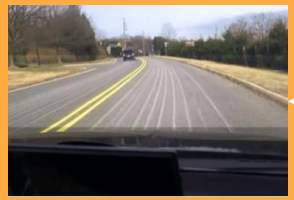


Terminology & notation

Action a



Policy
$\pi_\theta(a|o)$



Agent

Environment

Observation o

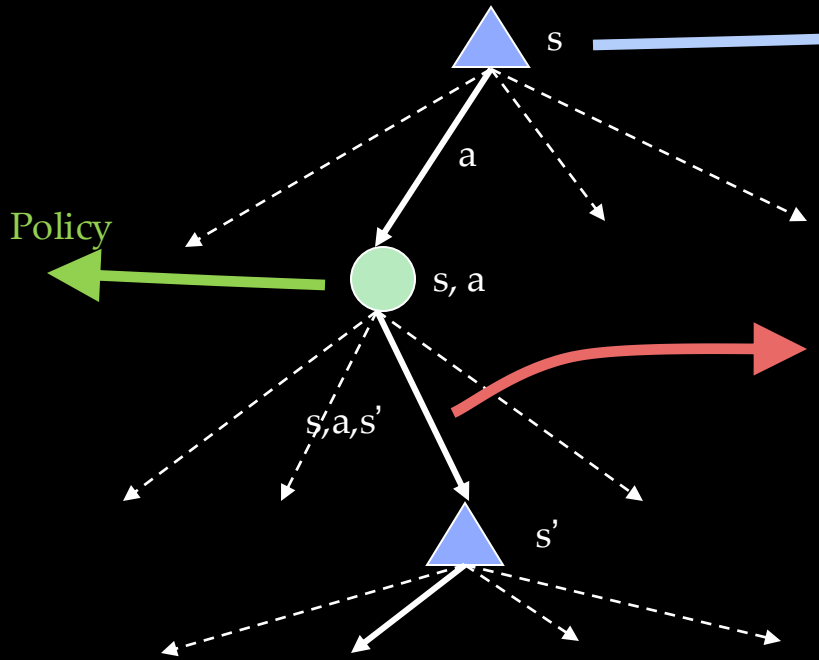
High reward

Reward r

Low reward (penalty)

The Goal of RL

- **Maximize reward** in the interaction with the environment (Markov decision process)

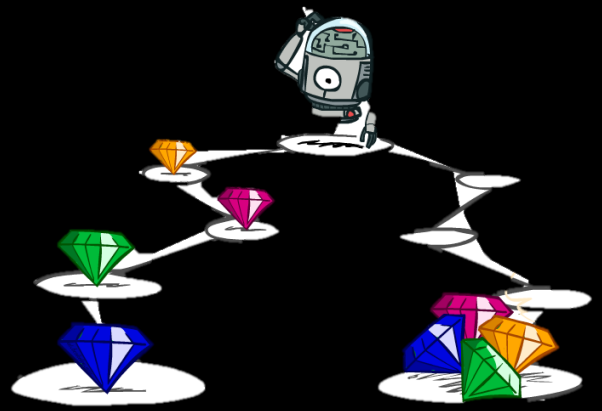


s is a *state*

(s,a,s') called a *transition*

$T(s,a,s') = P(s' \mid s,a)$

$R(s,a,s')$



Discount with γ

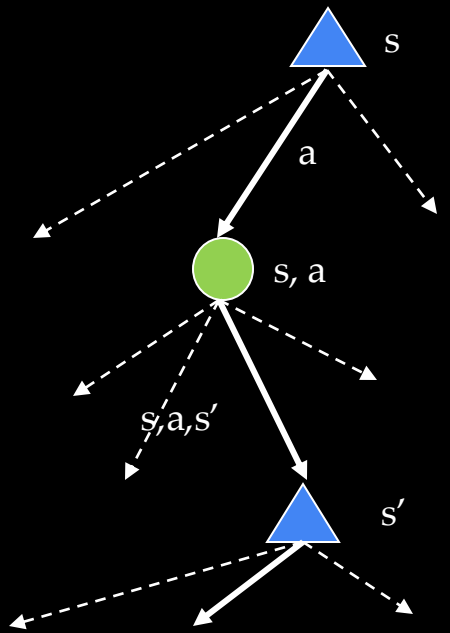
ETH zürich SoftRobotics Laboratory

Value and Bellman Equations

- **The value (utility) of a state s:**
  $V^*(s)$ = expected utility starting in s and acting optimally

- **The value (utility) of a q-state (s,a):**
  $Q^*(s,a)$ = expected utility starting out having taken action a from state s and (thereafter) acting optimally

- **The optimal policy:**
  $\pi^*(s)$ = optimal action from state s



s is a *state*

(s, a) is a *q-state*

(s,a,s') is a *transition*

Value and Bellman Equations

- **The value (utility) of a state s:**

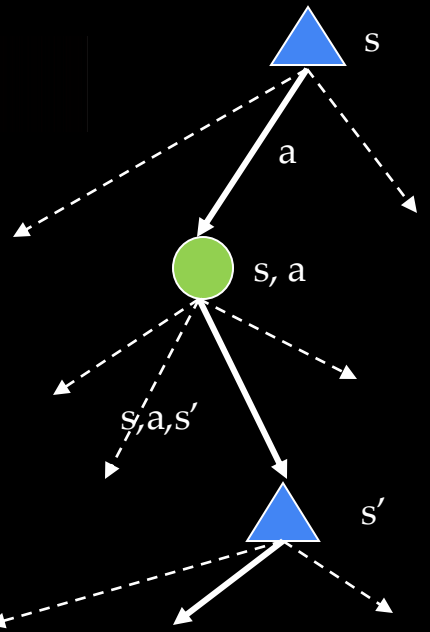
$$V^*(s) = \max_a Q^*(s,a)$$

- **The value (utility) of a q-state (s,a):**

$$Q^*(s,a) = \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^*(s')]$$

- **The optimal policy:**

$$V^*(s) = \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^*(s')]$$



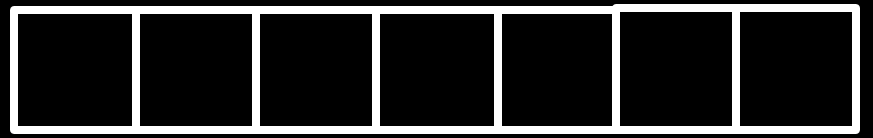
s is a *state*

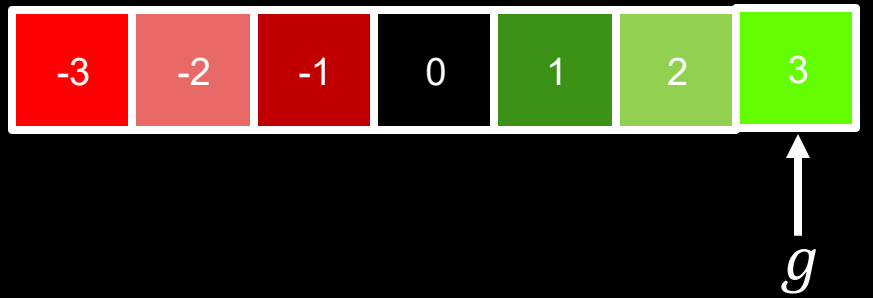
(s, a) is a *q-state*

(s,a,s′) is a *transition*

Example 1d grid world

- State space:
  - Position of agent $s \in \{-3, -2, -1, 0, 1, 2, 3\}$
  - Target position $g \in \{-3, 3\}$
- Action space:
  - Left / Stay / Right $a \in \{-1, 0, 1\}$
- Dynamics:
  - $s' = \text{clip}(s + a, -3, 3)$ with probability 0.9
  - $s' = s$ with probability 0.1
- Rewards:
  - $r(s, a) = 3 - |g - s| - |a|$





Example 1d grid world

Suppose fixed goal g=3

- Bootstrap on value function

Value Iteration: Step 0

To learn about value iteration, check Dynamic Programming and Optimal Control

- Q values

Example 1d grid world

What happens if we set new target once it's achieved?

g' = - g if g=s

Value Iteration (Iteration 0)

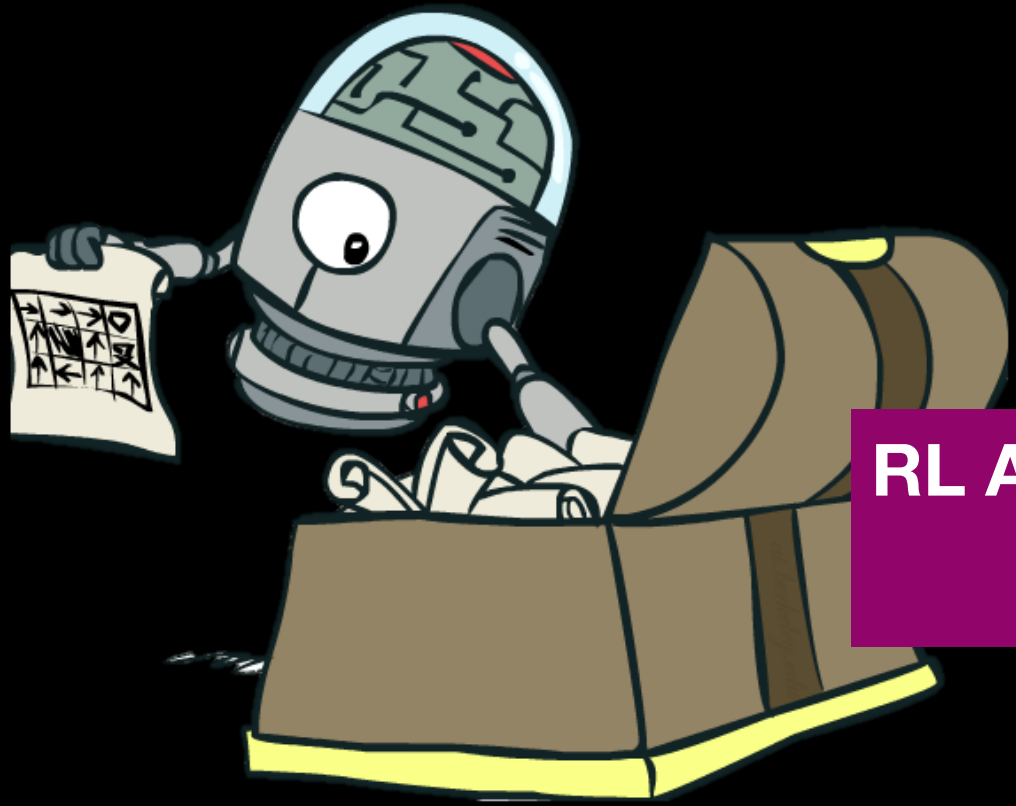
g=+3	0.00	0.00	0.00	0.00	0.00	0.00	0.00
g=-3	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Does optimal policy reach the goal?

No

How to fix the problem?

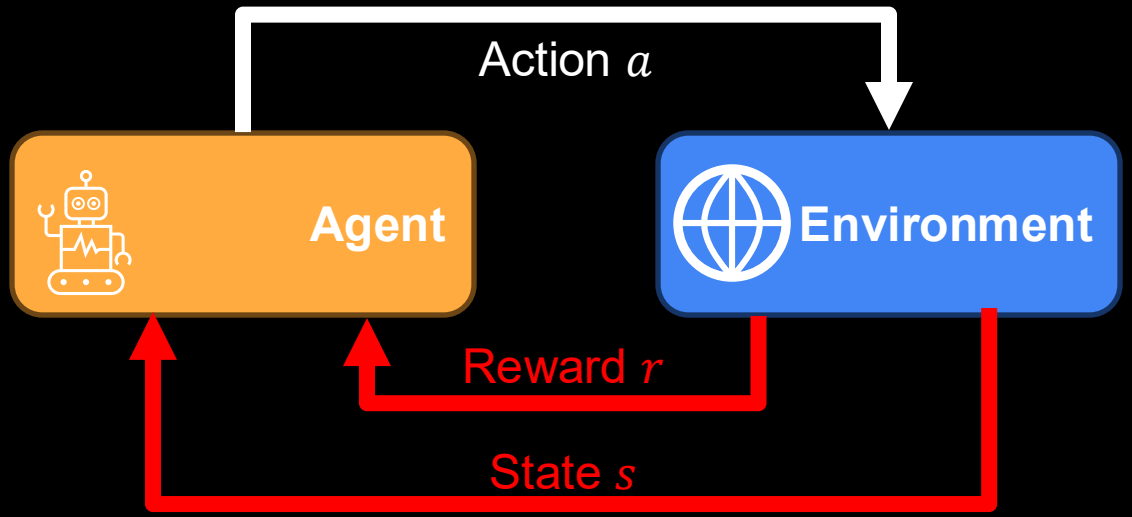
1. Success reward
2. End episode



RL Algorithms

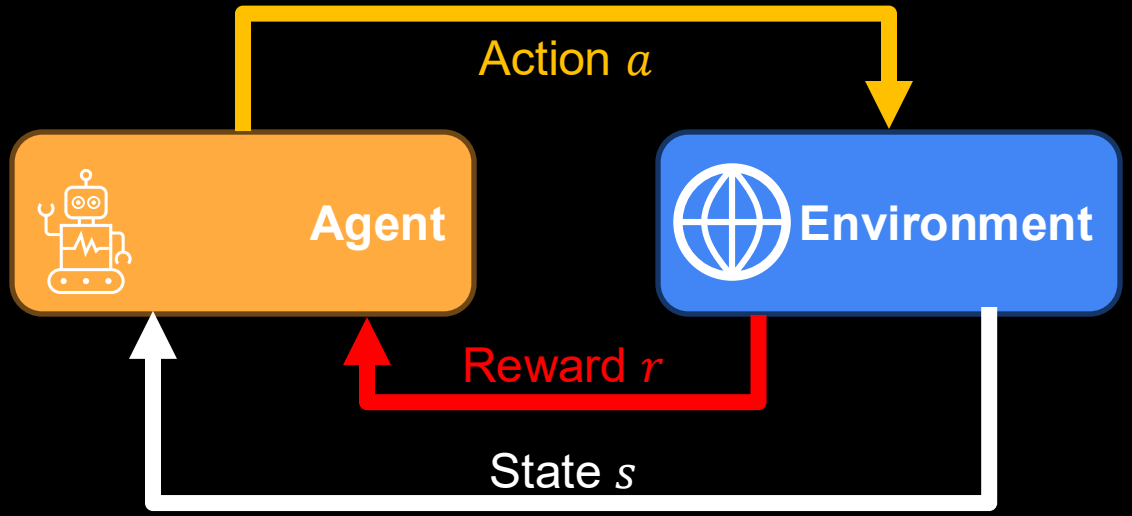
What's special about RL problem

Stochasticity: Rewards and state transitions may be random and unknown



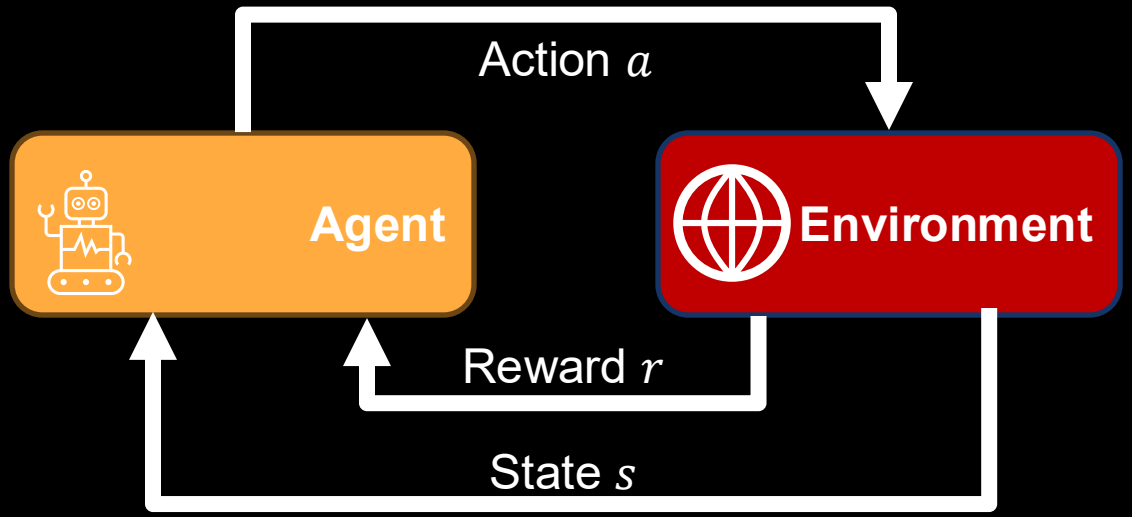
What's special about RL problem

Credit assignment: Reward r_t may not directly depend on action a_t



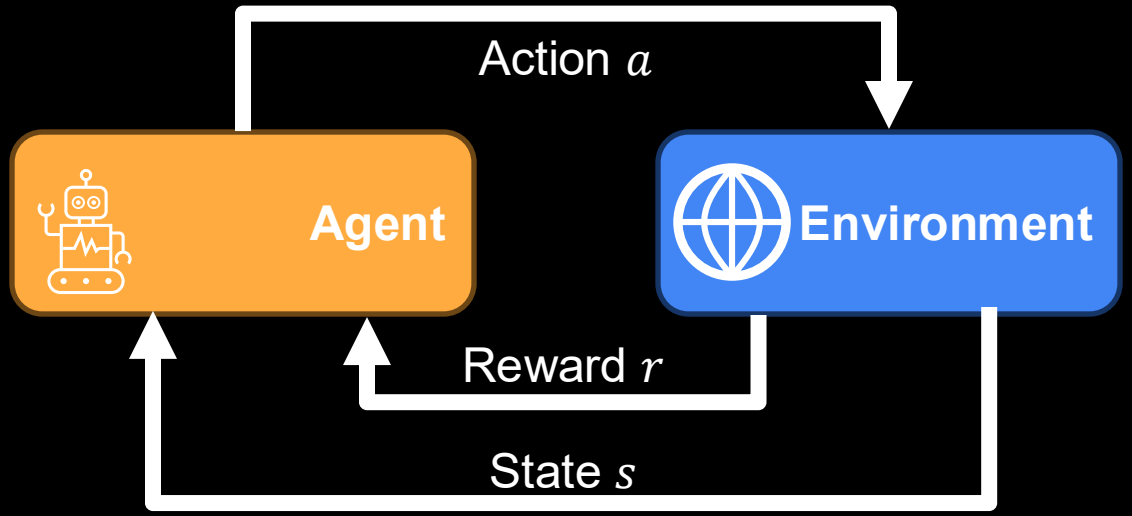
What's special about RL problem

Nondifferentiable: Can't backprop through world; can't compute $\frac{\partial r}{\partial a}$ or $\frac{\partial x}{\partial a}$

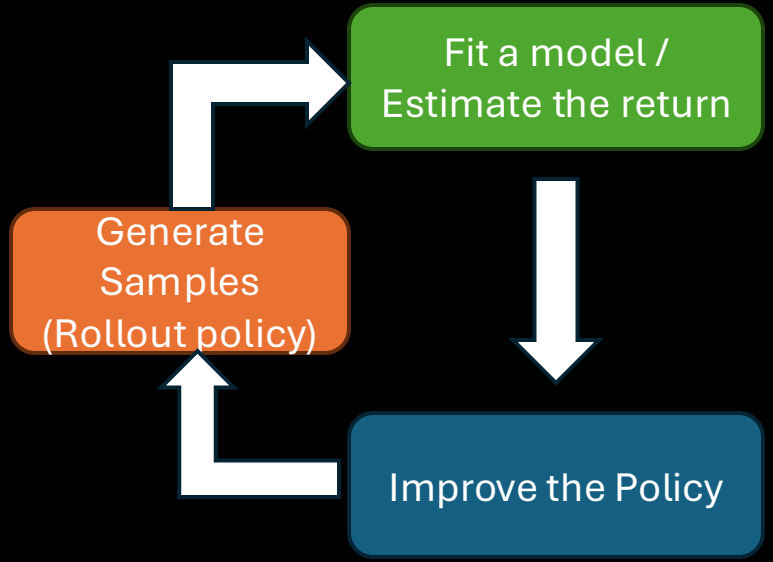


What's special about RL problem

Nonstationary: What the agent experiences depends on how it acts



The anatomy of a reinforcement learning algorithm

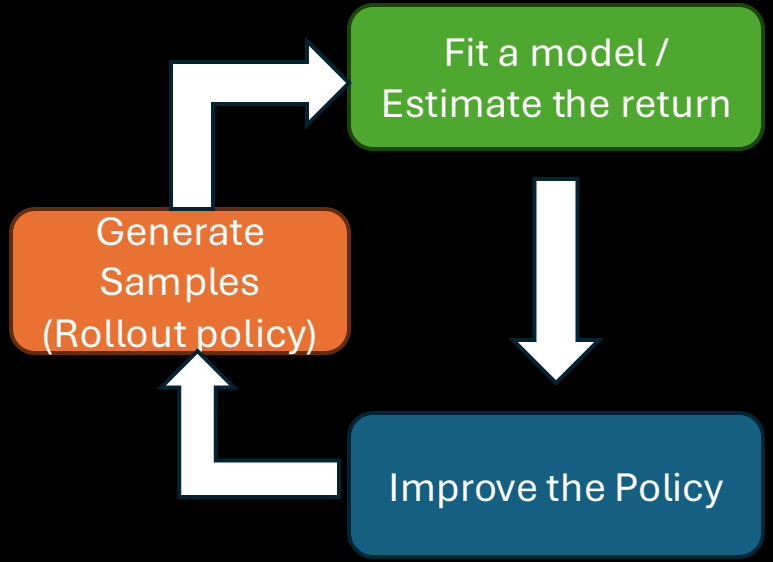


Learn the dynamics model

- $p(s'|s,a)$
- $r(s,a)$

Optimal control from the model

Value-function-based algorithms



Fit $V(s)$ or $Q(s,a)$.

$\pi(s) = \text{argmax}_a \, Q(s,a)$.

Policy Gradient

Fit a model / Estimate the return

Evaluate total returns of episodes τ

$$R_\tau = \sum r(s, a)$$

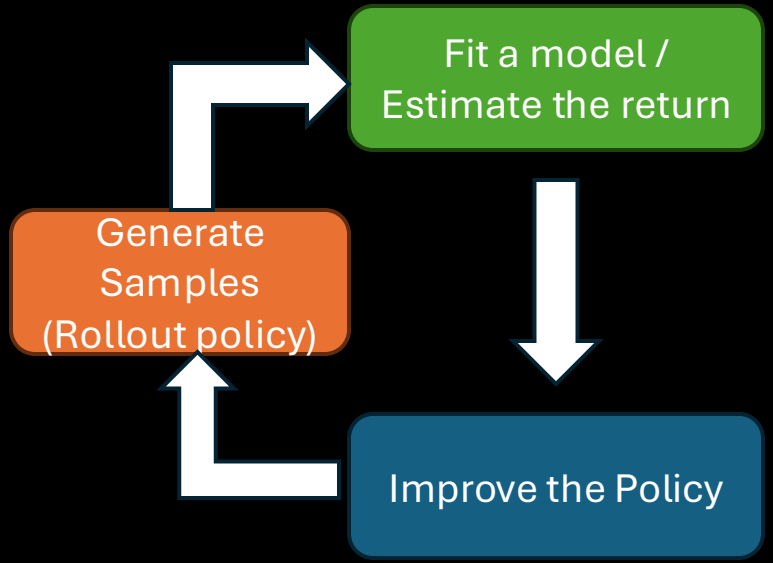
To learn about value iteration, check Probabilistic AI class

Improve the Policy

Estimate the gradient to the policy

$$\theta \leftarrow \theta + \alpha \nabla_\theta E\left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t)\right]$$

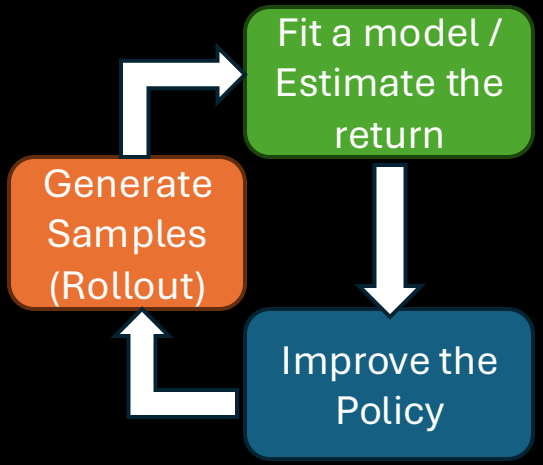
Actor-critic



A deep model (the “critic”)

Another deep model (the “actor”)

Actor-critic: PPO vs SAC



PPO (Policy Gradient)	DDPG / SAC (Value Based)
Fit $V(s)$ Evaluate Advantages	Fit $\mathrm{Q}(s, \mathrm{a})$
Estimate the **clipped** gradient to the policy $\theta \leftarrow \theta + \alpha \nabla_\theta E\left[\sum_t A(\mathbf{s}_t, \mathbf{a}_t)\right]$	$\text{policy} = \text{argmax}\ \mathrm{Q}$ $\theta \leftarrow \theta + \alpha \nabla_\theta E\left[\sum_t A(\mathbf{s}_t, \mathbf{a}_t)\right]$

ETH zürich SoftRobotics Laboratory

Tradeoffs Between Algorithms

Stability

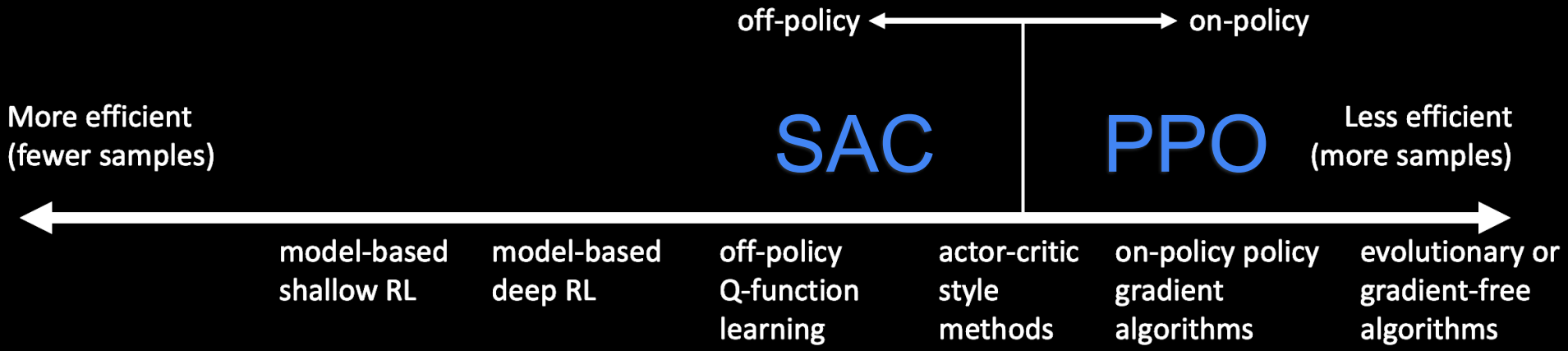
Policy Optimization

Biases in Bellman error optimization.

- Bootstrapping Bias (Baird's counterexample)
- Overestimation Bias
- Nonstationary Targets

Tradeoffs Between Algorithms

Sample Efficiency



resource: https://rail.eecs.berkeley.edu/deeprlcourse/deeprlcourse/static/slides/lec-4.pdf page37

Why does everyone still use PPO?

Practices in robotics

Curriculum Learning

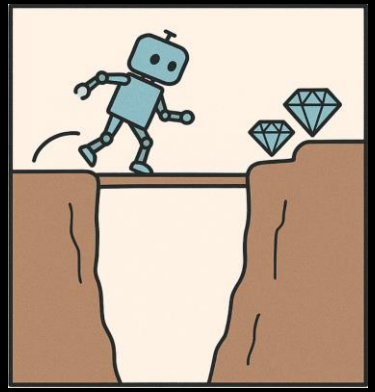
Examples:

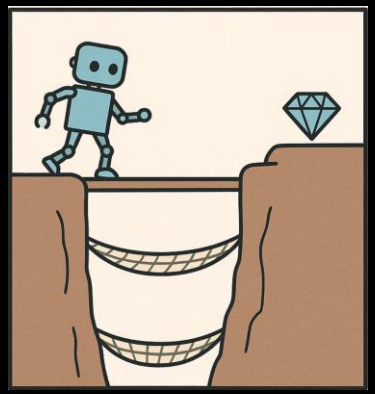
- Action smoothness:
  Low penalty weight -> high penalty weight

- Perception:
  Low noise -> high Noise

- Locomotion:
  Easy terrain -> hard terrain

- Manipulation:
  Easy success threshold -> hard success threshold





Sim or Real

resource: Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection
Levine et al., IJRR 2018
https://youtu.be/V05SuCSRAtg?si=Vx7IrS02e0FTGro4

Sim or Real

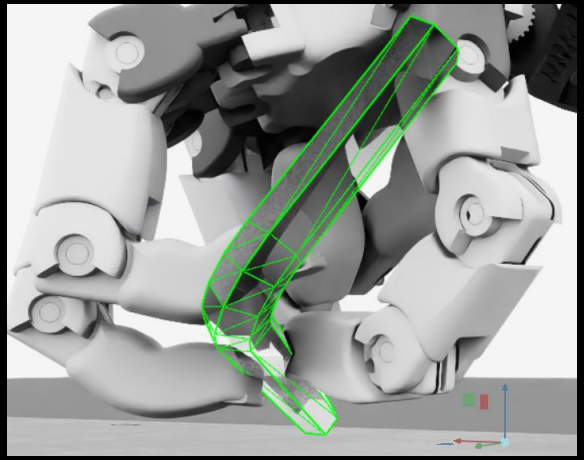
resource: Balancing Efficiency and Comfort in Robot-Assisted Bite-Transfer
Belkhale et al., ICRA 2022
https://sites.google.com/view/comfortbitetransfer-icra22/home

Simulated data

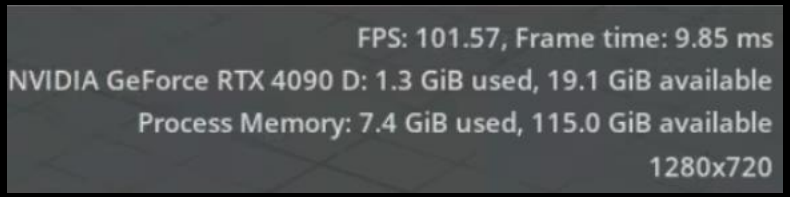
- Cheap
- Fast
- Scalable
- Safe
- Labeled
- Not beholden to real-world probability distributions

Difficulty of using simulated data

Physics simulators make big assumptions to run faster



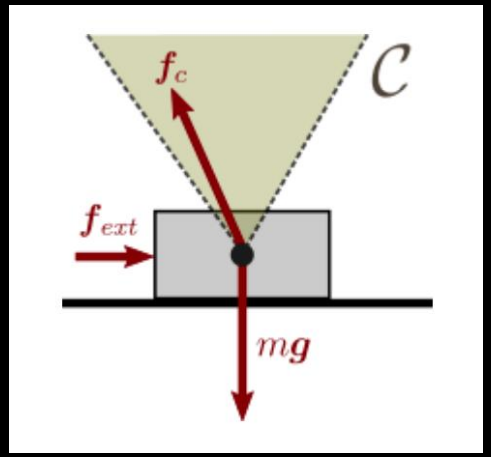
Convex Collision Check



Discrete time

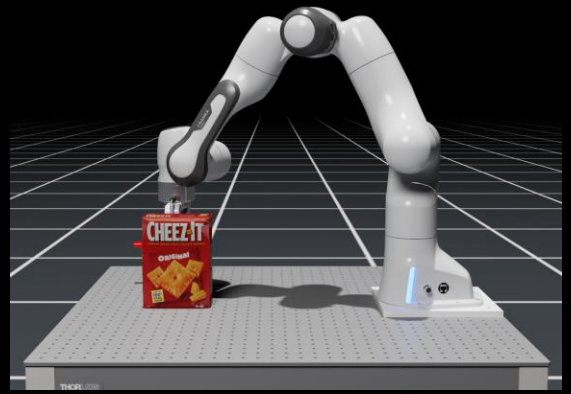
Difficulty of using simulated data

Physics simulators make big assumptions to run faster
Accurate physics model means more parameters to identify



Coulomb friction

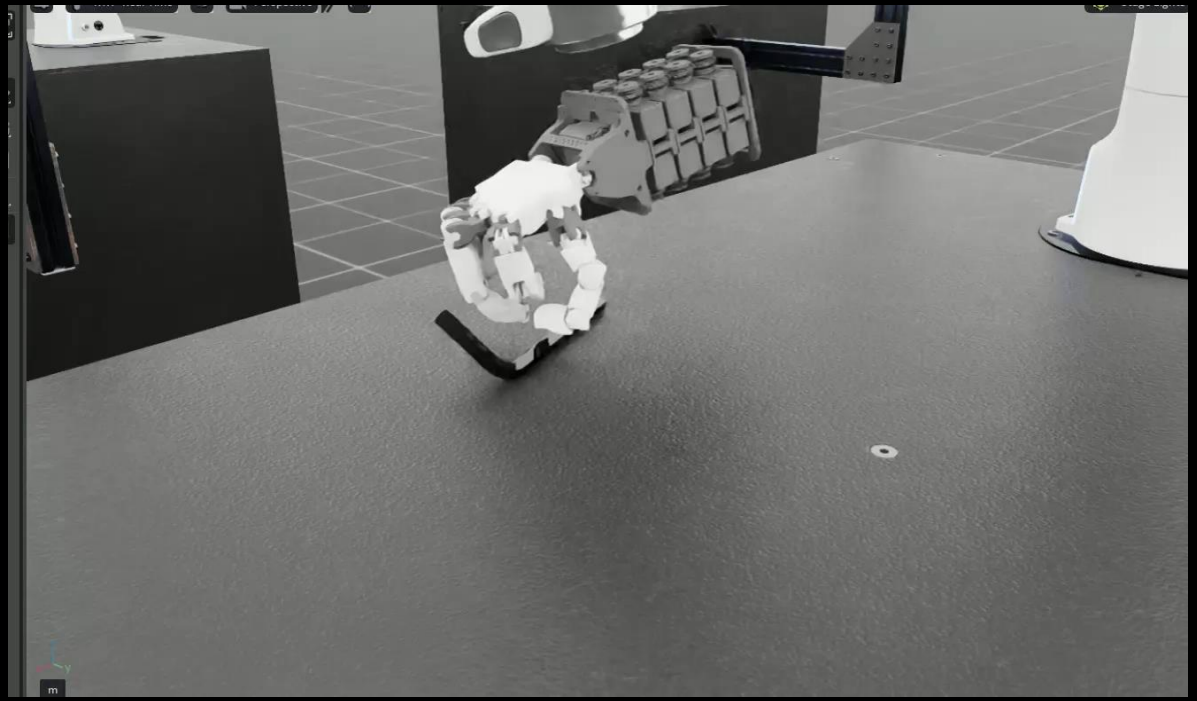
- Friction Coefficient?
- Inertia?
- Damping?
- Spring constants?



Rigid bodies

Difficulty of using simulated data

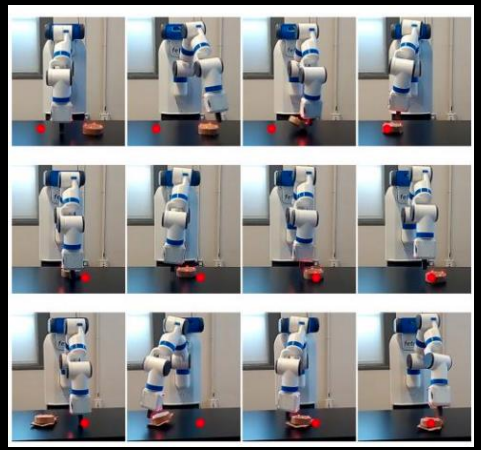
- Neural nets will exploit/overfit to differences in data distributions



Domain randomization

Increase the diversity in simulation domains so that the real world **may** look like another simulator.

For dynamics



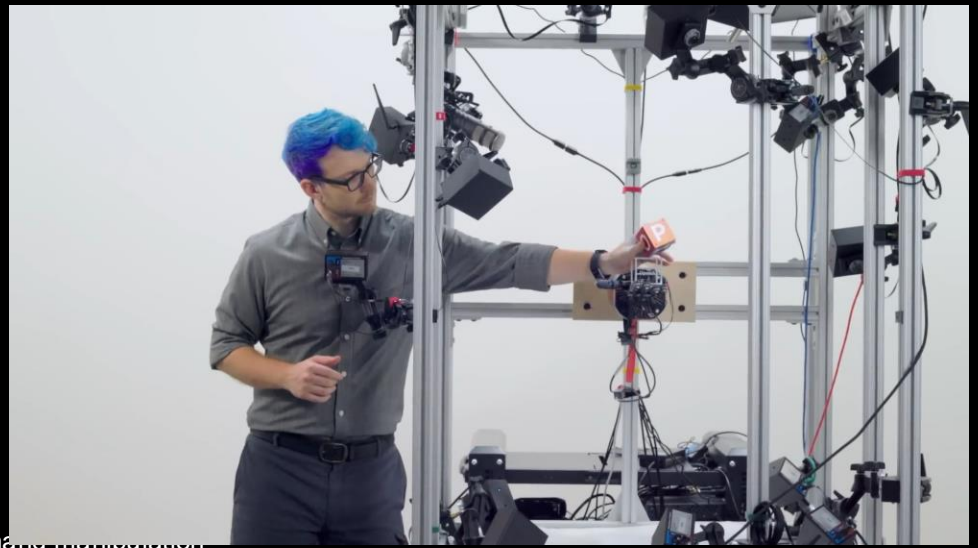
Parameter	Range
Link Mass	$[0.25, 4]\times$ default mass of each link
Joint Damping	$[0.2, 20]\times$ default damping of each joint
Puck Mass	$[0.1, 0.4]kg$
Puck Friction	$[0.1, 5]$
Puck Damping	$[0.01, 0.2]Ns/m$
Table Height	$[0.73, 0.77]m$
Controller Gains	$[0.5, 2]\times$ default gains
Action Timestep λ	$[125, 1000]s^{-1}$

resource: Sim-to-real transfer of robotic control with dynamics randomization
Peng et al., ICRA 2018

Domain randomization

Increase the diversity in simulation domains so that the real world **may** look like another simulator.

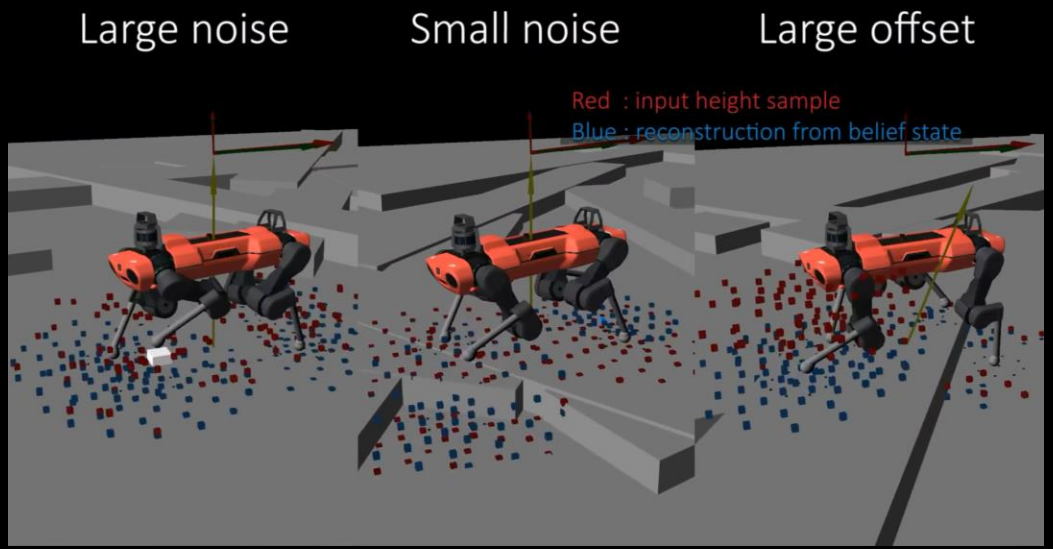
For Dexterity



resource: Learning dexterous in-hand manipulation
OpenAI, IJRR 2020
https://openai.com/index/learning-dexterity/

Domain randomization

Increase the diversity in simulation domains so that the real world **may** look like another simulator.
For height scans (point clouds)



ETH zürich
SoftRobotics Laboratory

resource: Learning robust perceptive locomotion for quadrupedal robots in the wild
Miki et al, Science Robotics 2022
https://youtu.be/zXbb6KQ0xV8?si=iKArSVbpijYKJB00

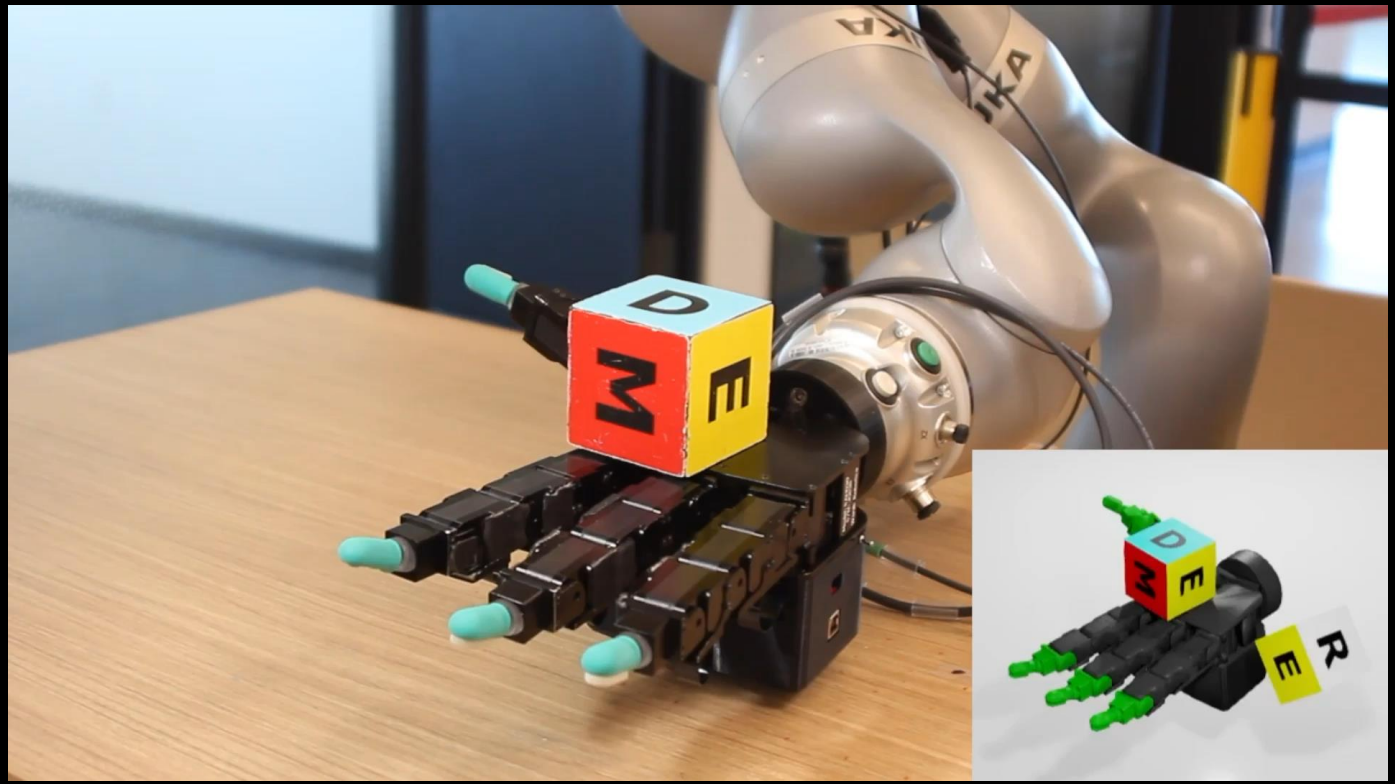
Domain randomization

Increase the diversity in simulation domains so that the real world **may** look like another simulator.

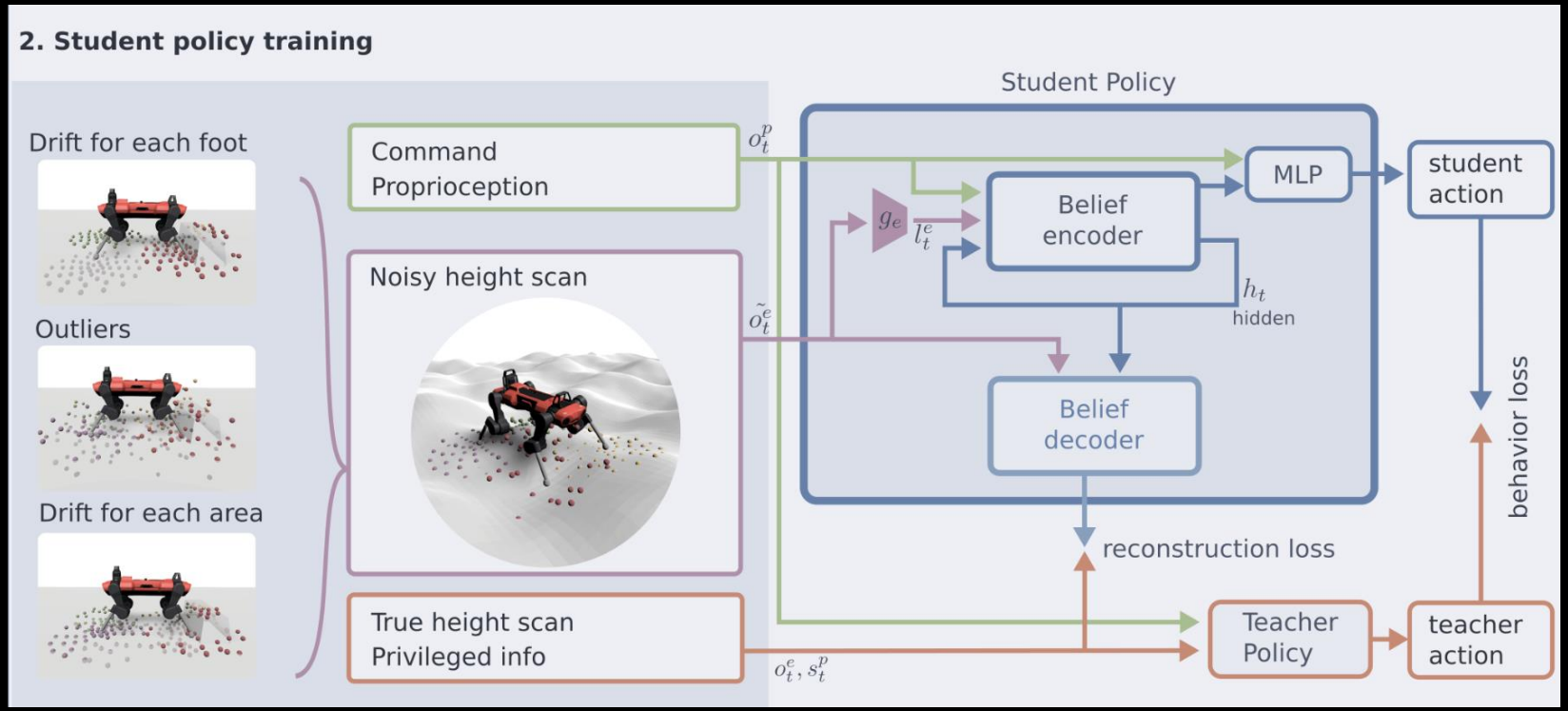
For RGB images

resource: Visuomotor Policies to Grasp Anything with Dexterous Hands
Singh et al, CORR 2024
https://dextrah-rgb.github.io/

Transfer of Agile In-Hand Manipulation from Simulation to Reality



Teacher student distillation



resource: Learning robust perceptive locomotion for quadrupedal robots in the wild
Miki et al, Science Robotics 2022
https://youtu.be/zXbb6KQ0xV8?si=iKArSVbpijYKJB00

Acknowledgements and Useful links

Randomization and the reality gap: how to transfer robotic policies from sim to real
https://youtu.be/ac_W9IgKX2c?si=kYlix7K6aDBd5a0E

Acknowledgements

- Berkeley CS285 Sergey Levine
- Berkeley CS188 Summer 2024