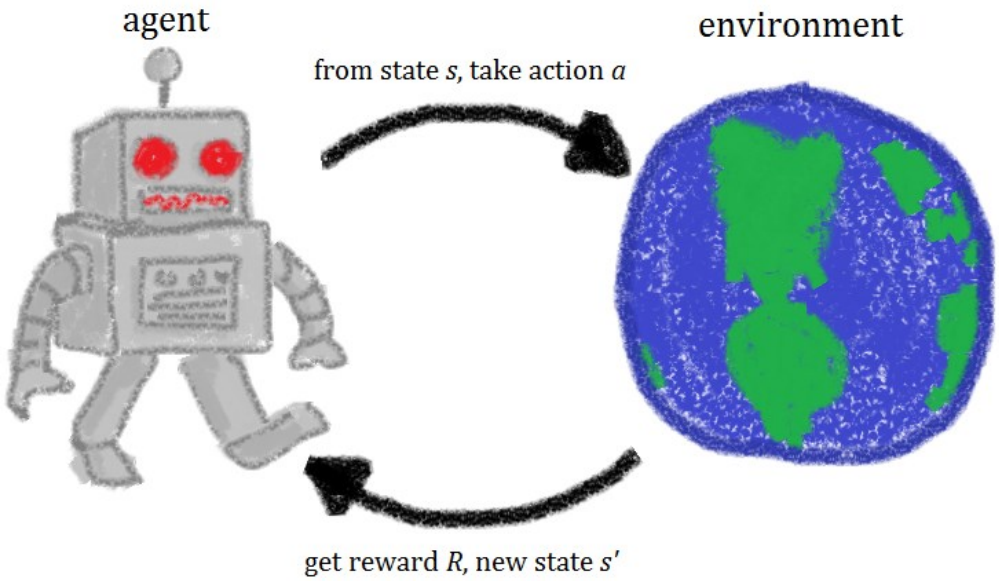
Locomotion Part 4: Robot Learning

Maren Bennewitz, Murad Dawood
Humanoid Robots Lab, University of Bonn

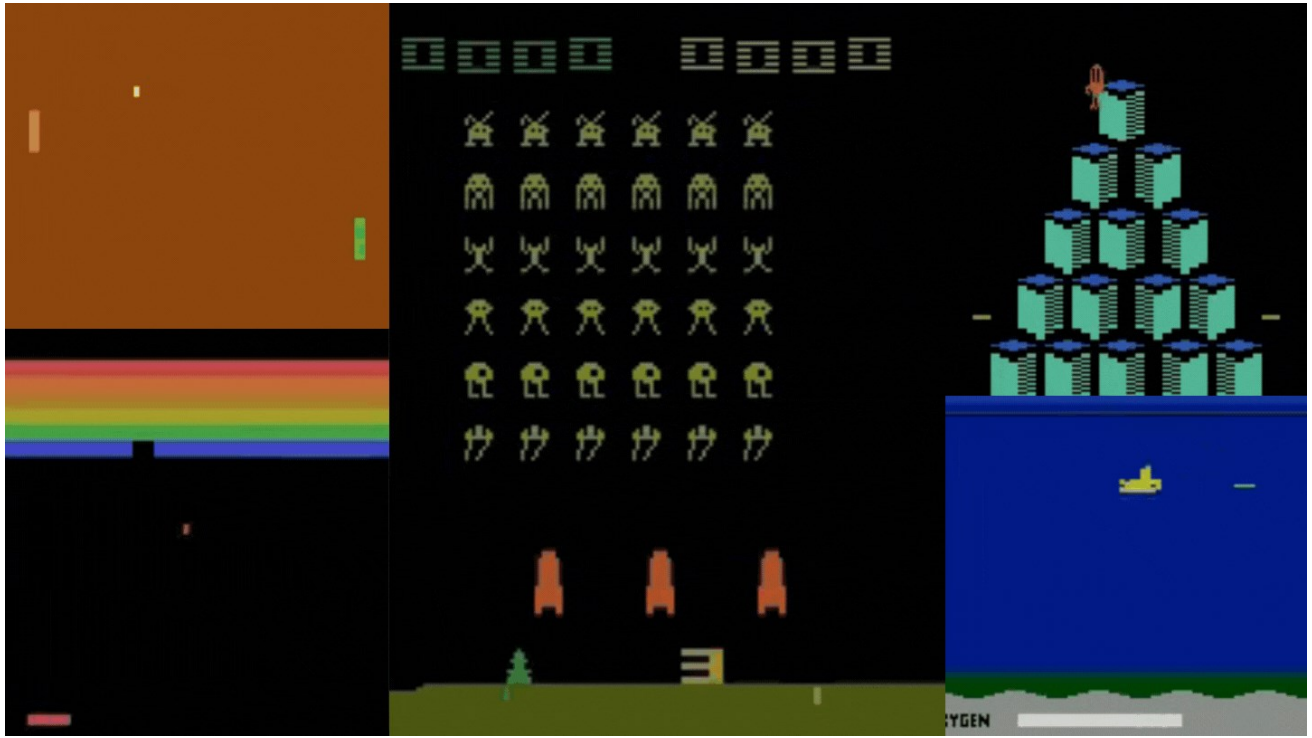
Goals of this Lecture

- Understand the basic **structure** and **objective** of reinforcement learning
- Describe the components of a **Markov Decision Process** (MDP)
- Explain the role of **policy** and **value** networks in decision-making
- **Compare** key RL approaches: on-policy, off-policy, offline, model-free, and model-based
- Learn how imitation learning is used in robot learning
- Reinfrocement learning for **Quadruped Locomotion**

Reinforcement learning (RL)



Atari Games using RL



Mnih, V., Kavukcuoglu, K., Silver, D. *et al.* Human-level control through deep reinforcement learning. *Nature* 518, 529–533 (2015).

Playing Badminton

Yuntao Ma et al. ,Learning coordinated badminton skills for legged manipulators.Sci. Robot.10, 2025

Drone Racing

Kaufmann, E., Bauersfeld, L., Loquercio, A. et al. Champion-level drone racing using deep reinforcement learning. Nature 620, 982–987 (2023)

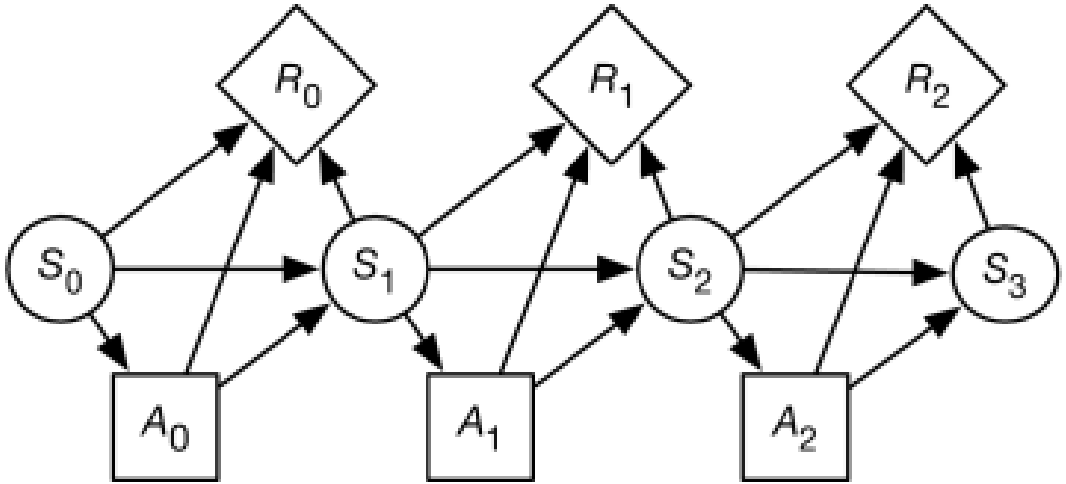
Markov Decision Process (MDP)

An MDP consists of the following components:

- S: A set of **states** the agent can be in
- A: A set of **actions** the agent can take
- P(s'|s, a): The **transition probability function** – the probability of reaching state s' after taking action a in state s
- R(s, a): The **reward** function – the immediate reward received after taking action a in state s
- γ (gamma): The **discount factor** – a number between 0 and 1 that determines how much future rewards are valued compared to immediate rewards

Markov Decision Process (MDP)

- The MDP satisfies the **Markov property,** which means the **next state and reward** depend only on the **current state and action**, not on the history of previous states or actions.



Reinforcement Learning Objective

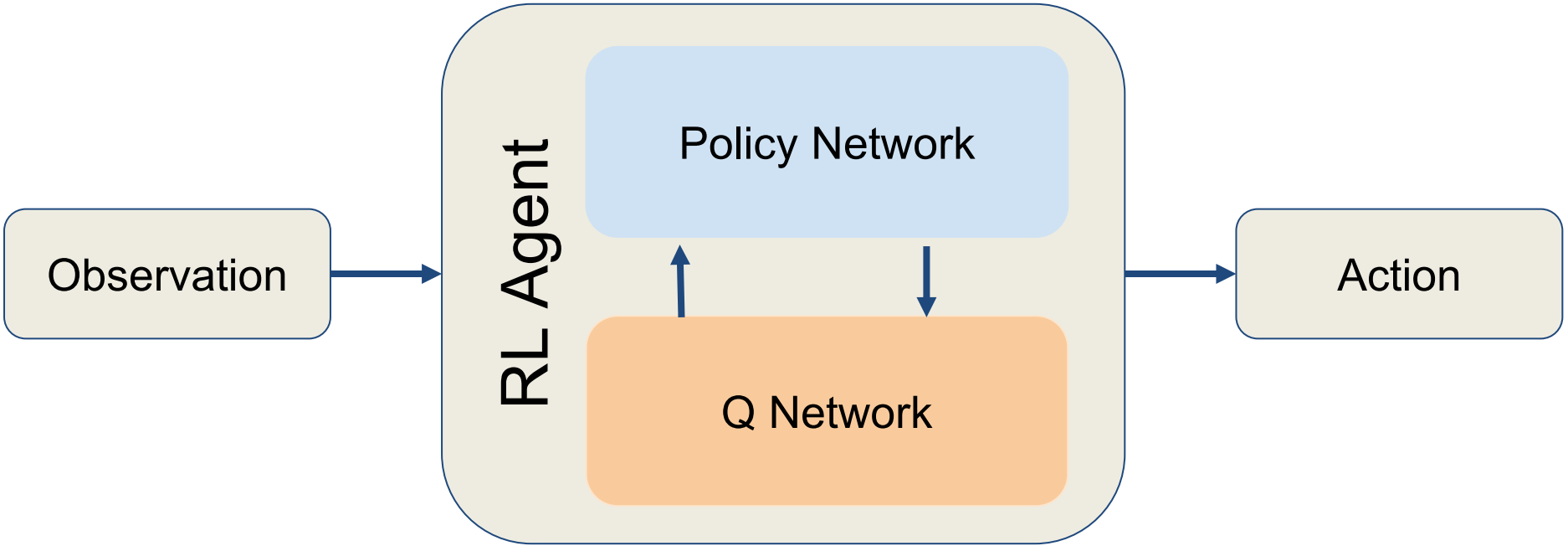
- Reinforcement learning aims to maximize the expected cumulative reward over time within an MDP.

$$E_{\tau \sim p_\theta(\tau)} \left[\sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t) \right]$$

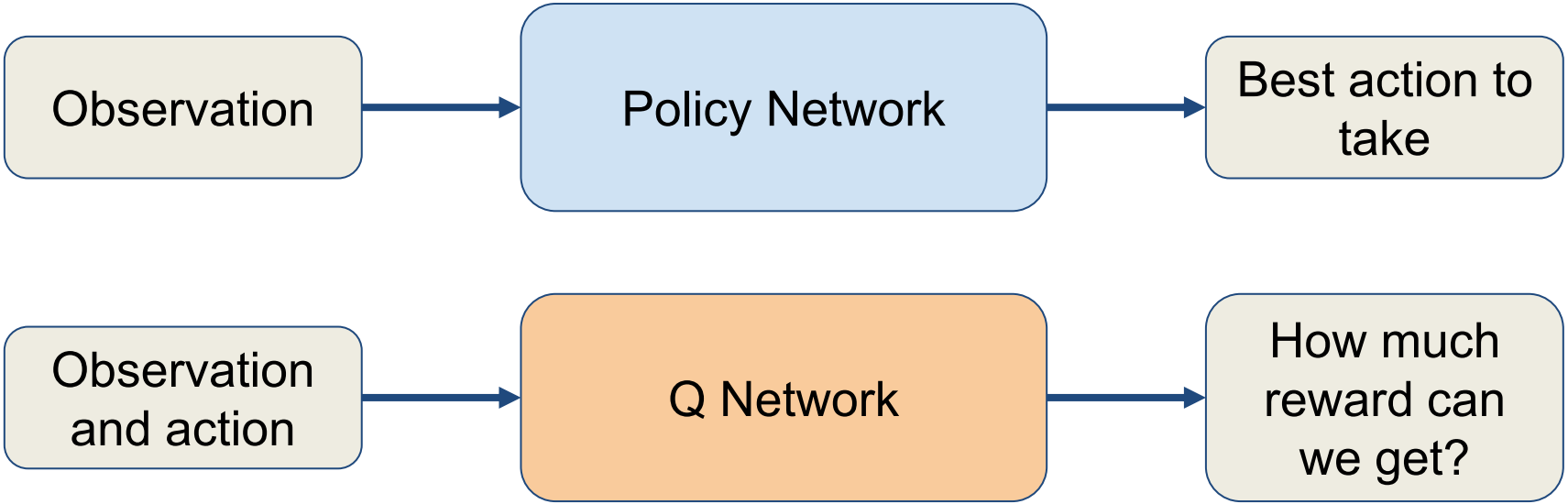
Expectation over Trajectories

Cumulative Reward / Return

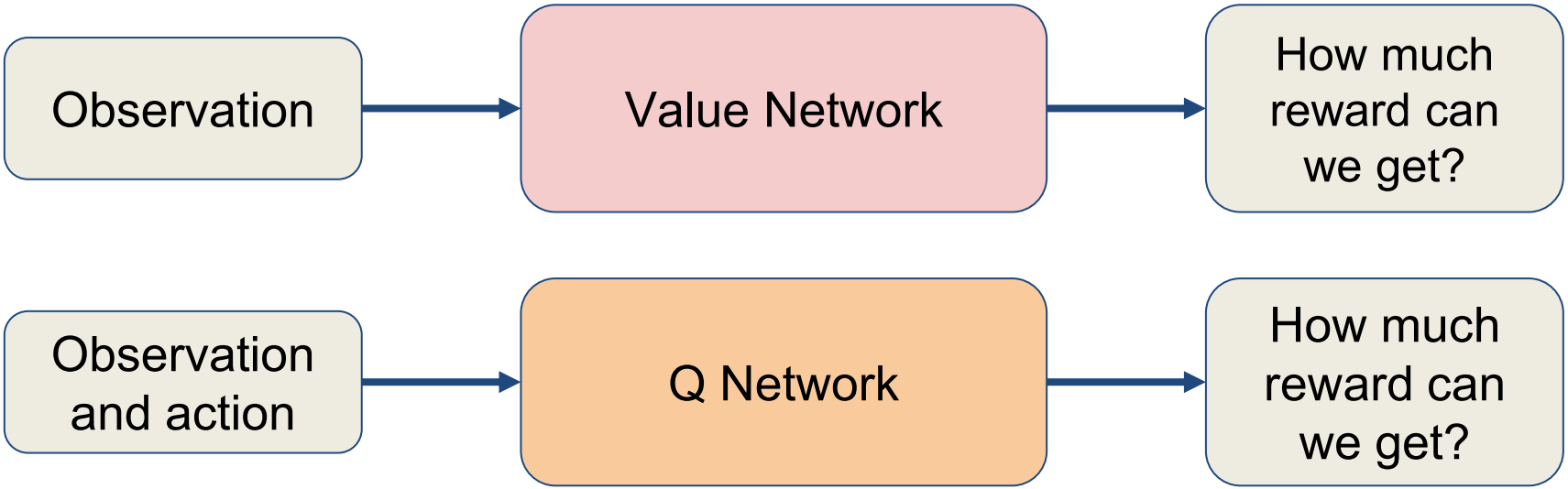
Policy and Value Networks



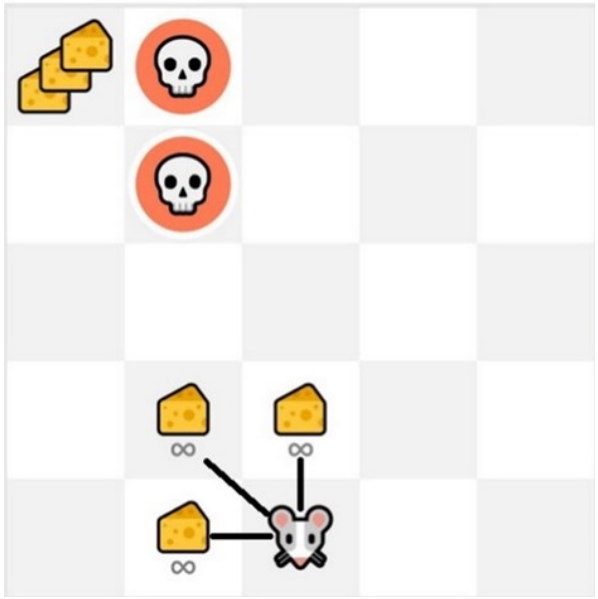
Policy and Value Networks



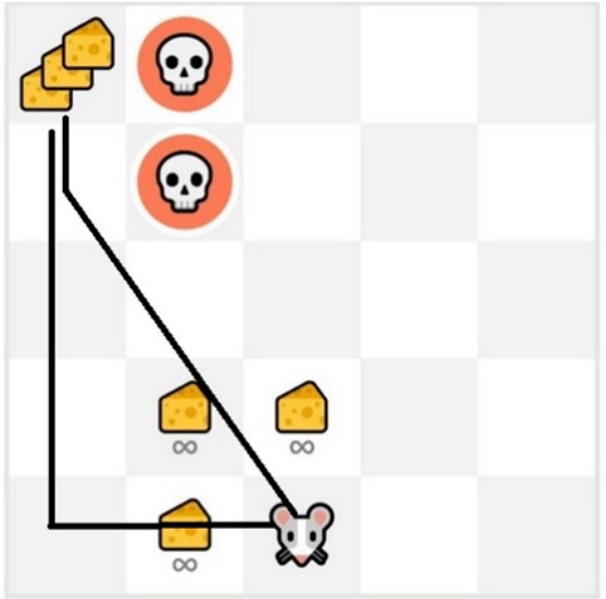
Policy and Value Networks



Exploitation vs Exploration



Take the best action we know



Try new actions to find better rewards

Exploitation vs Exploration

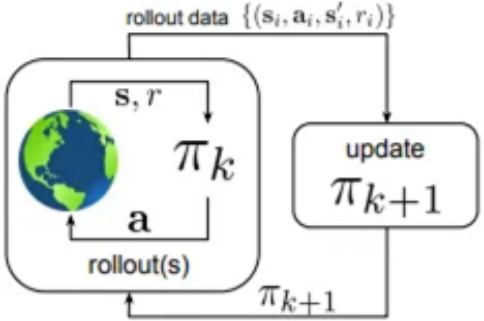
- **Epsilon-Greedy**
  - Commonly used in value-based methods like **Q-learning and DQN**
  - With **probability ε**, the agent selects a random action (**exploration**)
  - With **probability 1 − ε**, the agent selects the action with the highest estimated value (**exploitation**)
- **Noise-based Exploration**
  - Typically used in continuous action spaces (e.g., **DDPG, TD3**)
  - The agent selects actions by **adding noise** (e.g., Gaussian or Ornstein-Uhlenbeck process) to the output of the policy
  - This promotes local exploration around the current best action

Exploitation vs Exploration

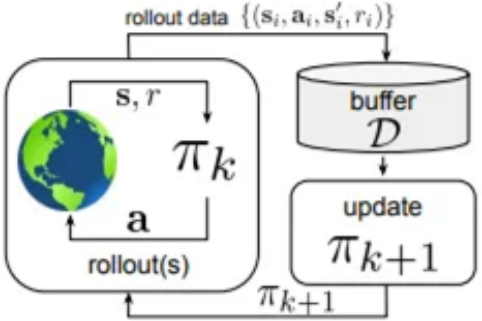
- **Maximum Entropy Method**
  - Used in entropy-regularized RL algorithms like **Soft Actor-Critic (SAC)**
  - The policy is encouraged to **maximize** both expected **return** and action **entropy**
  - This results in stochastic policies that naturally explore more diverse actions

Classification of RL Approaches: On-policy vs Off-policy vs Offline

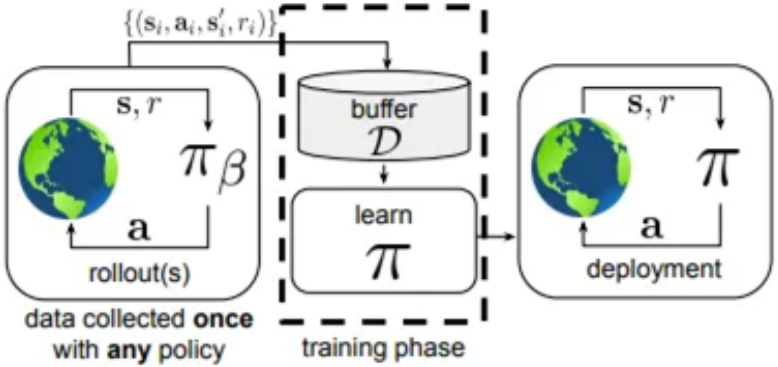
- On-policy RL



- Off-policy RL



- Offline RL

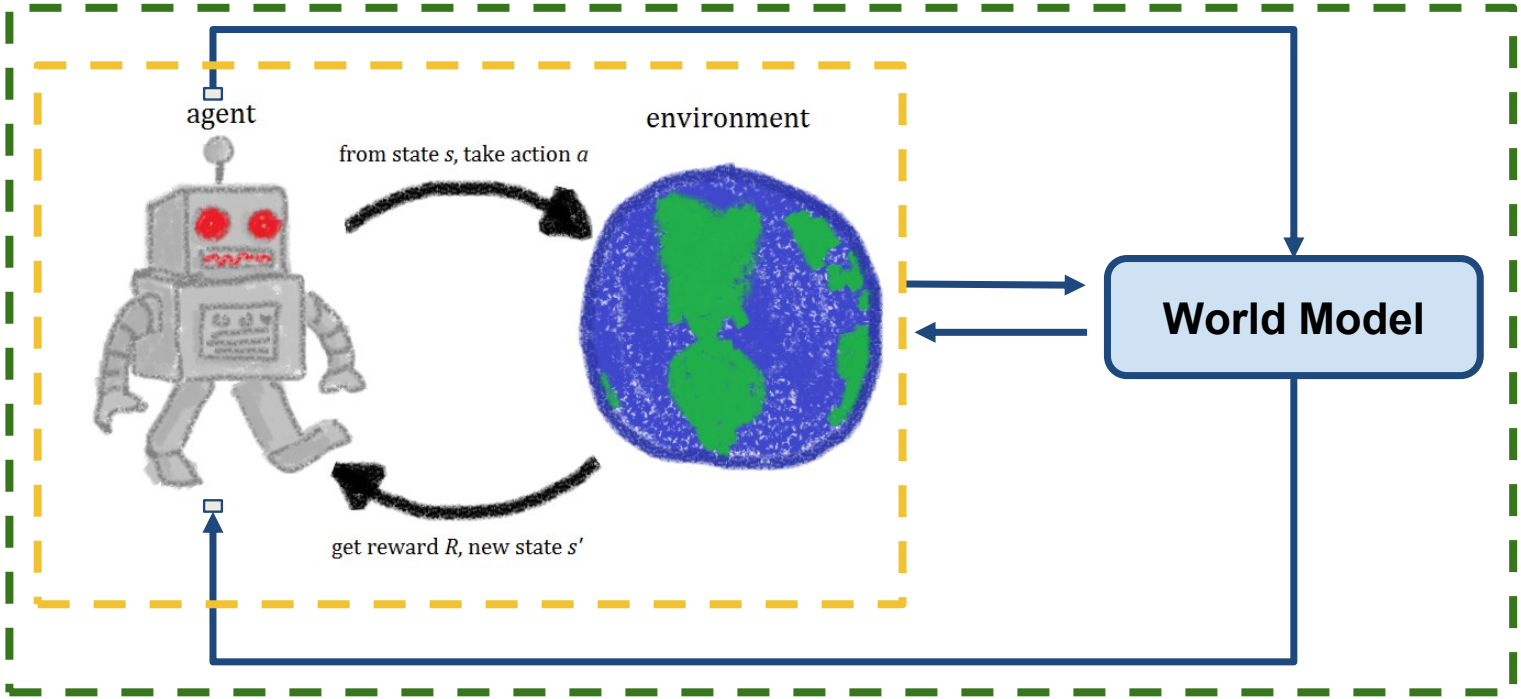


- **Examples**: TRPO, PPO
- **Examples** : DDPG, TD3, SAC
- **Examples**: CQL, Decision Transformers

On-policy vs Off-policy vs Offline RL

On-policy	Off-policy	Offline
Learns from **data** generated by the **current policy** itself	Learns from **data** that may come from a **different behavior policy**	Learns entirely from a fixed, pre-collected dataset with **no online interaction**
More stable	**Higher sample efficiency** via data reuse	Safe and practical when **environment interaction is expensive or risk**
Less sample-efficient	Can suffer from stability issues due to **distributional gap** between the data and what the current policy would actually experience	**Vulnerable to distributional shift**—the learned policy may encounter states/actions not covered by the dataset

Classification of RL Approaches: Model-based vs Model-free



Model-based vs Model-free

Model-based	Model-free
Learns a model of the environment (transition dynamics and/or reward) to plan or improve policies	Learns a policy or value function directly from experience **without modeling dynamics**
Higher Sample Efficiency — can generate synthetic data	**Lower Sample Efficiency** — needs large amounts of real experience
Higher Computational Cost — planning (e.g., MPC) or model rollouts are often done online	**Lower Computational Cost** — policy is typically a feed-forward network, fast to execute
Model bias, difficulty of learning accurate long-horizon dynamics	Poor sample efficiency, harder to guarantee safety

Model-based reinforcement learning (MBRL)

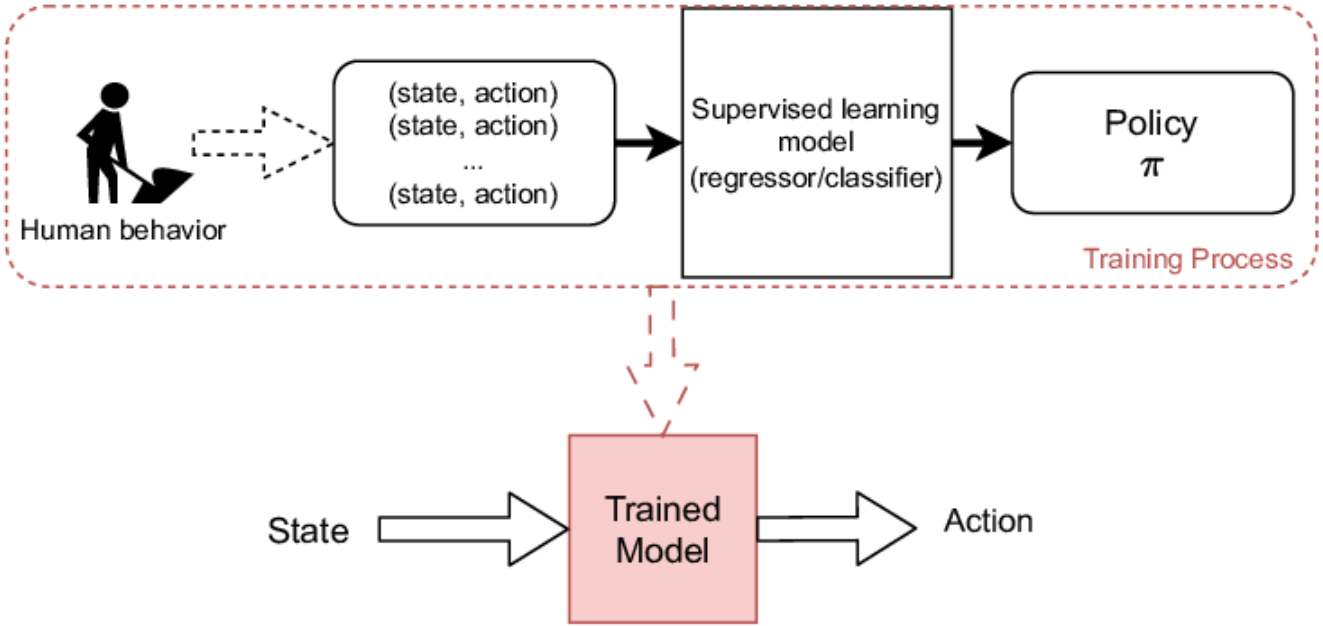
- Learn world model
- Additionally, learn rewards model and termination model
- Examples:
  - Hafner, Danijar, et al. "Dream to control: Learning behaviors by latent imagination." arXiv preprint arXiv:1912.01603 (2019). "***Dreamer***"
  - Janner, Michael, et al. "When to trust your model: Model-based policy optimization." Advances in neural information processing systems 32 (2019). ***"MBPO"***

MBRL for improving decision making

- **Planning:**
  - Use the learned dynamics model to plan future a trajectory
  - Choose the trajectory with the highest rewards
  - Apply the first step of the trajectory as in MPC
  - Example: ***"Deep Dynamics Models for Learning Dexterous Manipulation" (Nagabandi et al., 2019)***
- **Policy Training with Synthetic Rollouts:**
  - Use the world, reward, and termination models to generate synthetic data.
  - Train the policy and value networks in on the generated data
  - Apply the trained policy to the real world. ***(Dreamer)***

Imitation Learning

- Leverages expert data to train the robot to perform the task.



Imitation Learning

Imitation Learning: DAgger

Input:

- Expert policy (provides correct actions)
- Initial policy (e.g., from behavior cloning)
- Number of training iterations

Initialize:

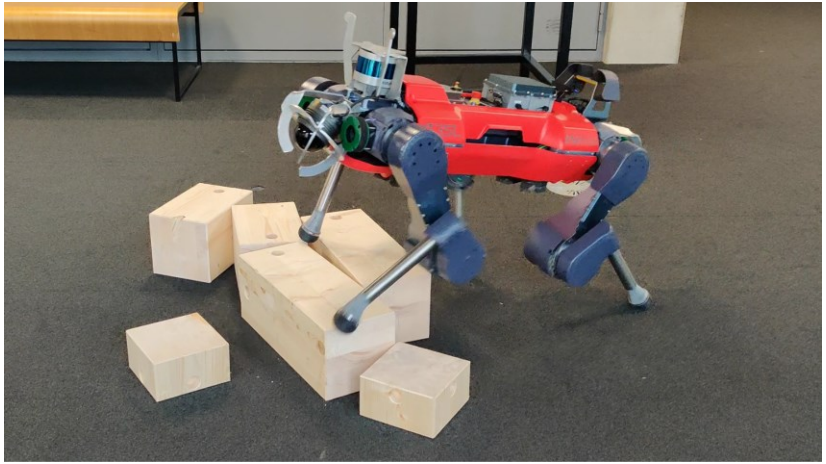
- An empty dataset to store training examples

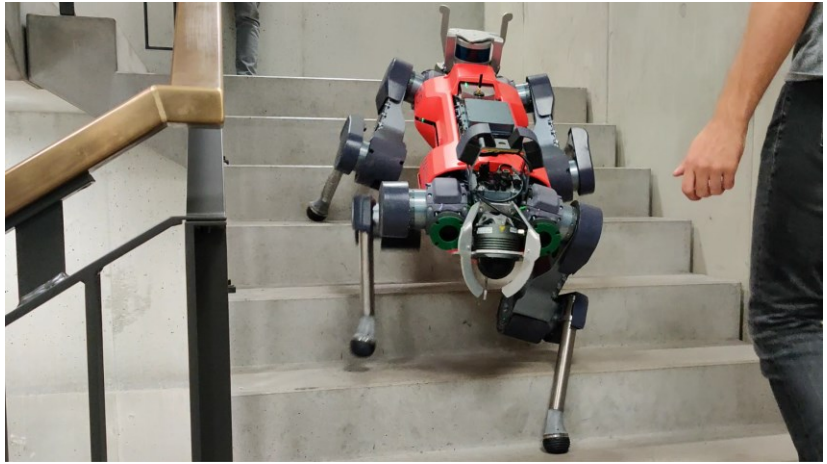
For each iteration:

1. Run the **current policy** in the environment to **collect states**
2. For each visited state, ask the **expert** for the **correct action**
3. **Add the (state, expert action)** pairs to the dataset
4. Retrain the policy on the **updated dataset** using supervised learning

Return the final trained policy

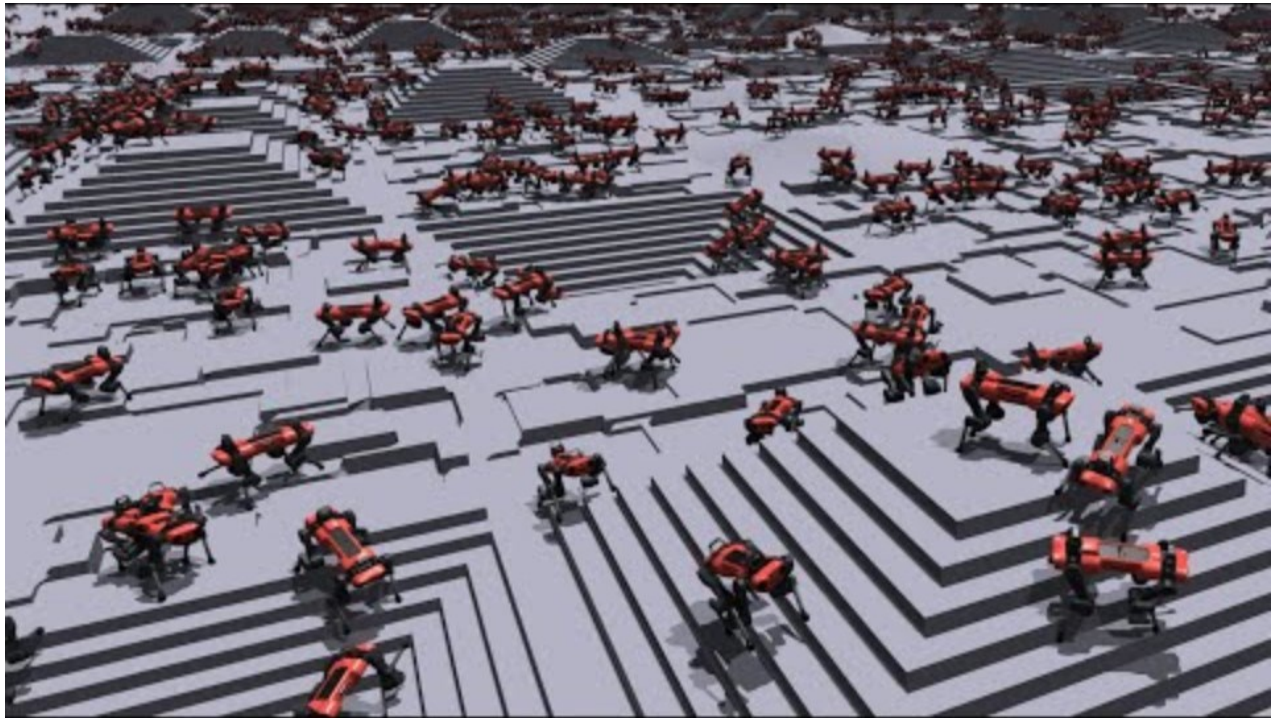
Learning Locomotion of Quadrupeds





Rudin, Nikita, et al. "Learning to walk in minutes using massively parallel deep reinforcement learning." Conference on Robot Learning. PMLR, 2022.

Learning Locomotion of Quadrupeds



Rudin, Nikita, et al. "Learning to walk in minutes using massively parallel deep reinforcement learning." Conference on Robot Learning. PMLR, 2022.

Algorithm: On-policy PPO

- Uses parallel environments to handle sample inefficiency
- Reduces training time

Observations

- Base linear and angular velocities
- Measurement of the gravity vector
- Joint positions and velocities
- The previous actions selected by the policy
- The 108 measurements of the terrain sampled from a grid around the robot’s base. Each measurement is the distance from the terrain surface to the robot’s base height.

Actions

- Desired joint positions sent to the motors. There, a PD controller produces motor torques.

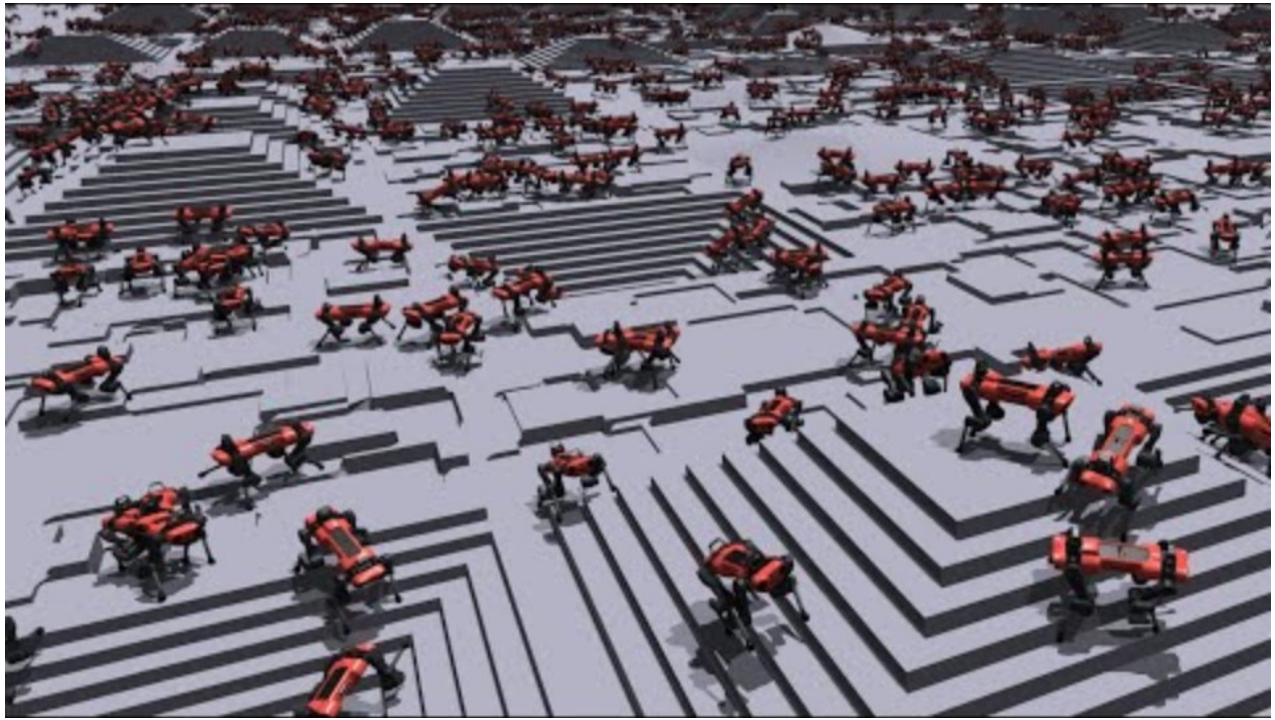
Rewards

- Encourage the robot to follow the commanded velocities.
- Penalize joint torques, joint accelerations, joint target changes, and collisions, to create a smoother, more natural motion.
- Contacts with the knees, shanks or between the feet and a vertical surface are considered collisions.
- Contacts with the base are considered crashes and lead to resets .
- Reward term encouraging the robot to take longer steps, which results in a more visually appealing behavior.

Sim to Real Transfer

- Randomize the friction of the ground
- Add noise to the observations
- Randomly push the robots during the episode to teach them a more stable stance

Sim to Real Transfer



Rudin, Nikita, et al. "Learning to walk in minutes using massively parallel deep reinforcement learning." Conference on Robot Learning. PMLR, 2022.

Summary

- Reviewed the core idea of Reinforcement Learning and its formalization through Markov Decision Processes (MDPs)
- Explained the RL objective: maximizing expected cumulative reward
- Introduced policy and value networks as key components of decision-making
- Compared RL approaches: on-policy, off-policy, offline, model-free, and model-based
- Discussed the role of supervised learning in RL, including behavior cloning
- Highlighted applications of RL in real-world tasks like quadruped locomotion

Literature

- *Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. Vol. 1. No. 1. Cambridge: MIT press, 1998.*
- *Mnih, V., Kavukcuoglu, K., Silver, D. et al. Human-level control through deep reinforcement learning. Nature 518, 529–533 (2015).*
- *Yuntao Ma et al. ,Learning coordinated badminton skills for legged manipulators.Sci. Robot.10, 2025*
- *Ross, Stéphane, Geoffrey Gordon, and Drew Bagnell. "A reduction of imitation learning and structured prediction to no-regret online learning." Proceedings of the fourteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings, 2011.*
- *Kaufmann, E., Bauersfeld, L., Loquercio, A. et al. Champion-level drone racing using deep reinforcement learning. Nature 620, 982–987 (2023)*
- *Hafner, Danijar, et al. "Dream to control: Learning behaviors by latent imagination." arXiv preprint arXiv:1912.01603 (2019).*
- *Janner, Michael, et al. "When to trust your model: Model-based policy optimization." Advances in neural information processing systems 32 (2019).*
- *Rudin, Nikita, et al. "Learning to walk in minutes using massively parallel deep reinforcement learning." Conference on Robot Learning. PMLR, 2022.*