UC San Diego

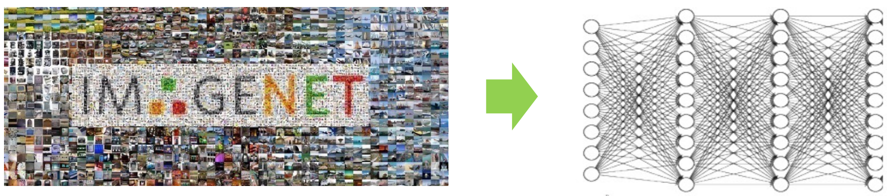
CSE-276F

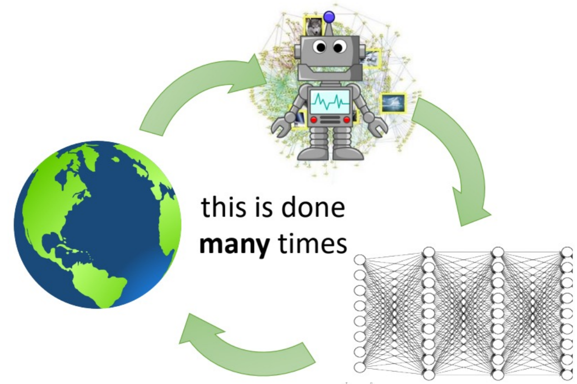
Boosting Reinforcement Learning and Planning with Demonstrations: A Survey

Stone Tao
Slides prepared by Tongzhou Mu and Stone Tao

RL Can Be Costly

- Deep RL algorithms usually require tremendous training samples
  - Impractical or inefficient in complex environments





Supervised Learning

- Dataset is collected beforehand
- Fit the labeled data

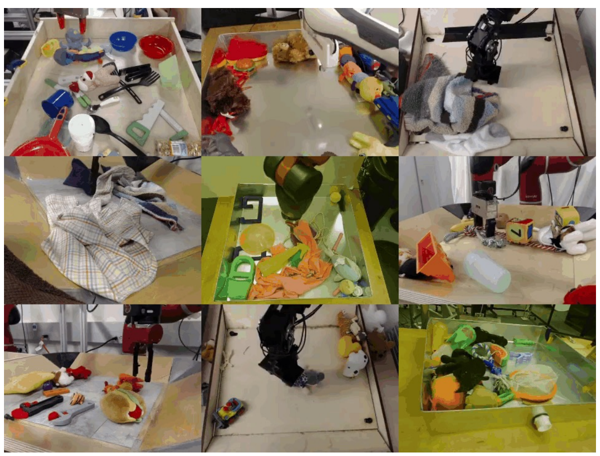
Reinforcement Learning

- Dataset is collected during interaction
- Find a good policy by trial-and-error

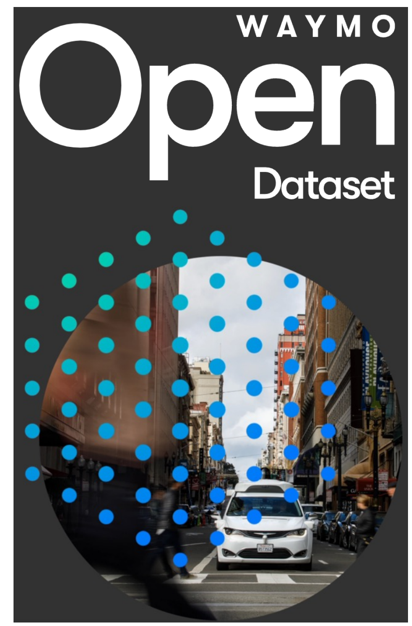
Figures from Offline RL Tutorial at NeurIPS 2020

Datasets for Decision Making

- Pre-collected demonstrations in many domains
  - Robotics
  - Autonomous Driving
  - …



RoboNet



Waymo Open Dataset

Figures from RoboNet and Waymo

Topics of This Talk

- Key questions to discuss
  - How to utilize demonstrations in RL and planning?
  - How to collect demonstrations that are useful for RL and planning?

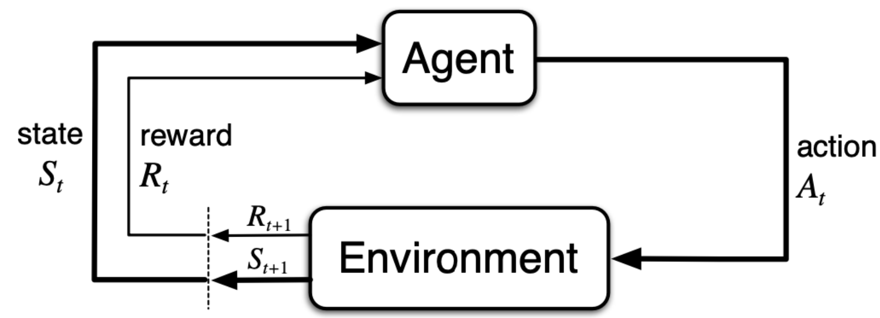
Outline

- **Use Demonstrations**
  - Offline – without interactions with environments
  - Online – with interactions with environments
- **Collect Demonstrations**
- **Future Directions**

Outline

- **Use Demonstrations**
  - Offline – without interactions with environments
  - Online – with interactions with environments
- **Collect Demonstrations**
- **Future Directions**

Demo Use – Offline & Online

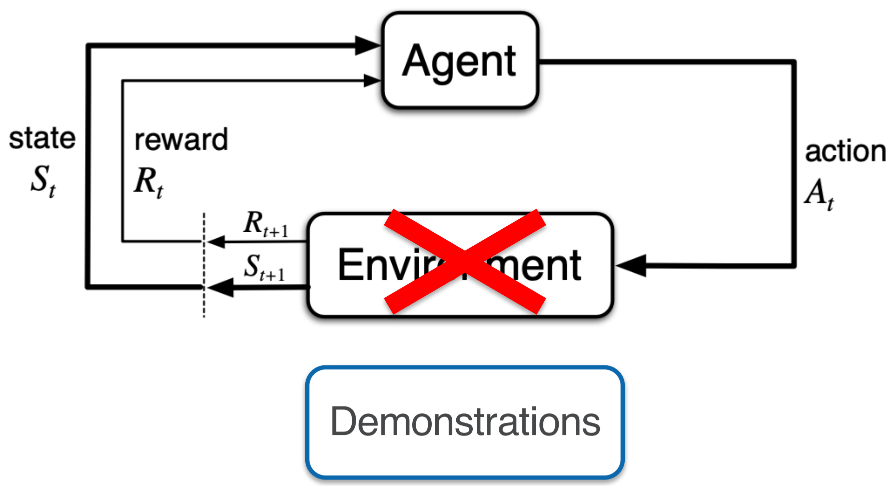


- Offline Stage
  - Agent cannot access the environment
  - Only learn from the demonstrations
- Online Stage
  - Agent can interact with the environment
  - Learn from the environment while still leveraging demonstrations
- Note
  - Both stages are optional
  - They can also be combined together

Figures from Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction.

Use Demo Offline

- **What to learn** from the demo?



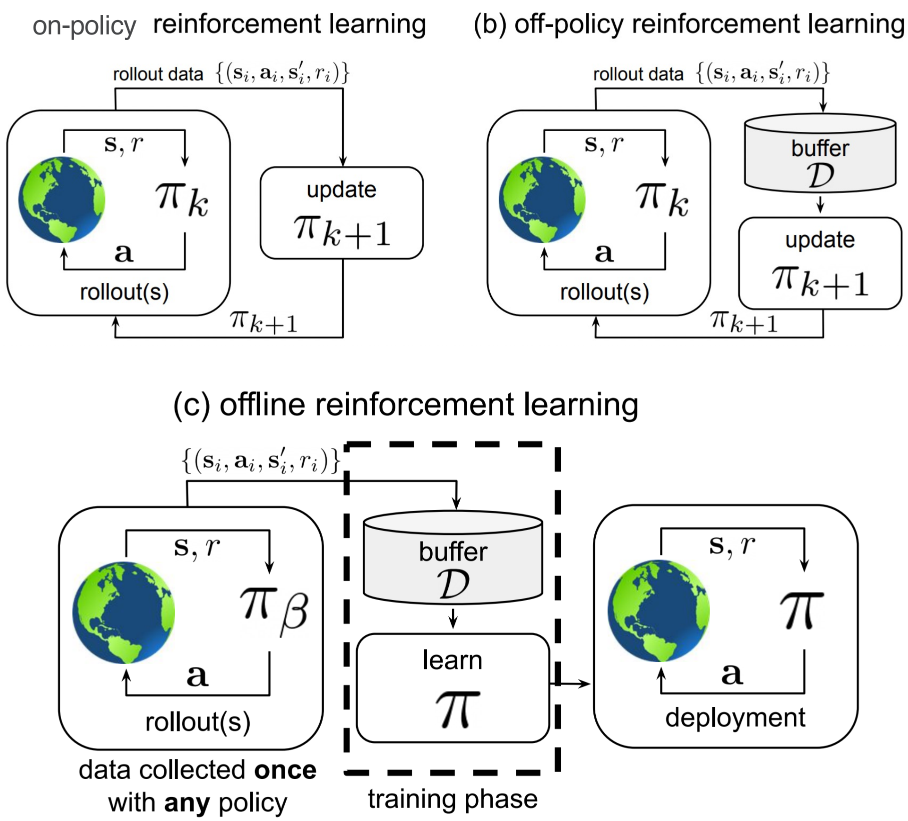
Figures adapted from Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction.

Learn Policy

- Imitation Learning
  - Imitate the behaviors in demonstrations
  - Two types of methods: behavior cloning and inverse RL
- Behavior Cloning (BC)
  - Supervised learning, i.e., clone the expert's actions at each state in demo
  - $$\underset{\theta}{\text{maximize}} \sum_{(s,a)\in\rho_D} \ln \pi_\theta(a|s)$$
- Inverse RL
  - Recover a reward function that can induce the behaviors in demonstrations
  - We will talk more about this later
- Limitations of Imitation Learning
  - Requires expert demonstrations

Learn Policy

- Offline RL



Figures from Levine, Sergey, et al. "Offline reinforcement learning: Tutorial, review, and perspectives on open problems."

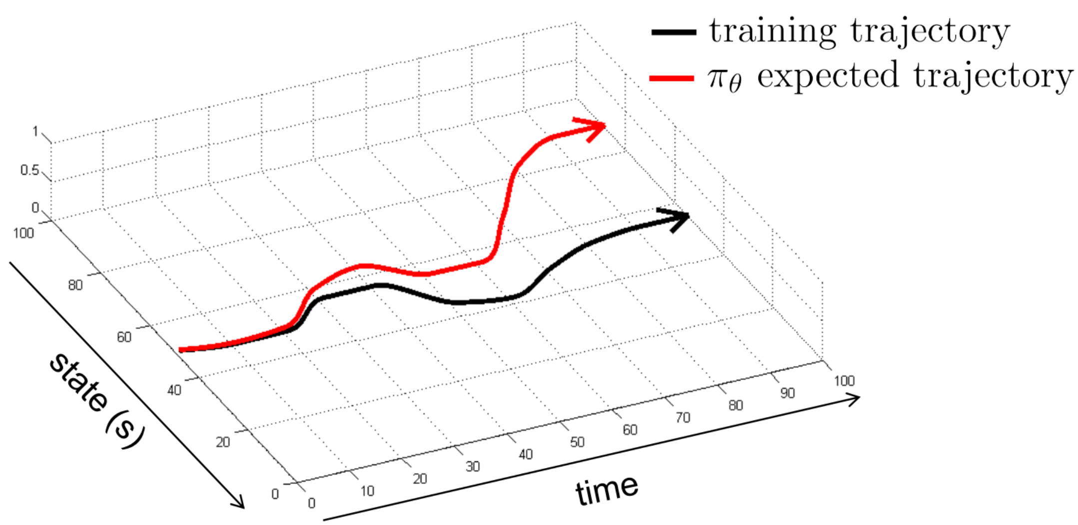
Learn Policy

- Is offline learned policy good enough?
  - No, due to the distribution shift
  - When we use a learned policy to act, it changes what we see

Simple example:

behavioral cloning
train π_θ to *copy*
assume data is *optimal*

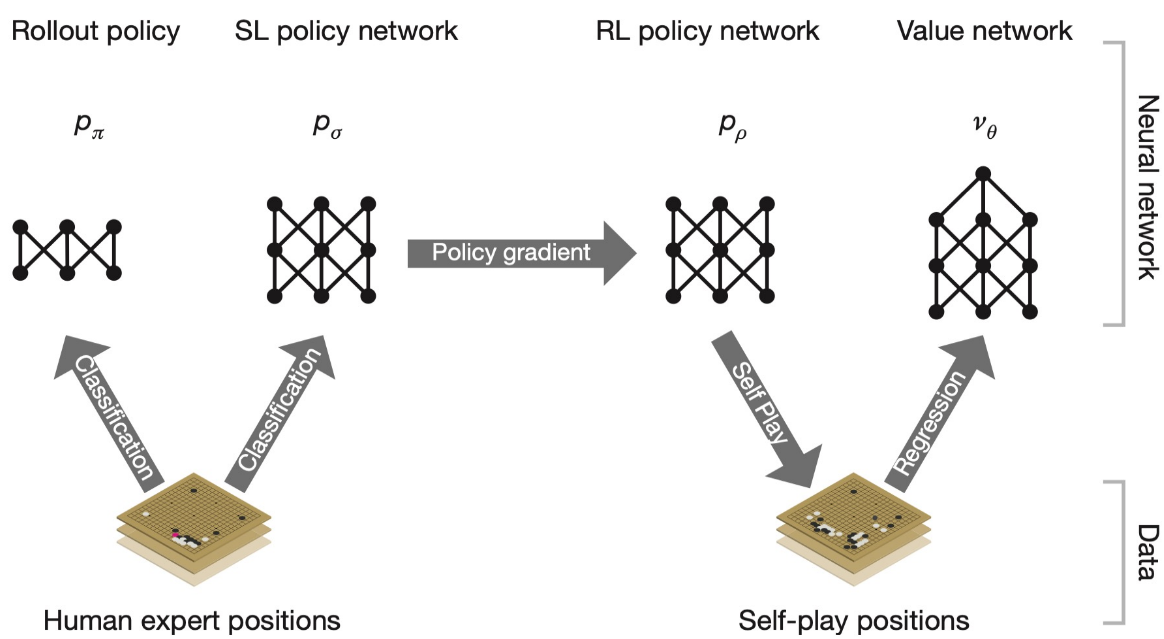
error scales as $O(T^2)$



Figures from Offline RL Tutorial at NeurIPS 2020
Ross et al. A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning. '11

Learn Policy

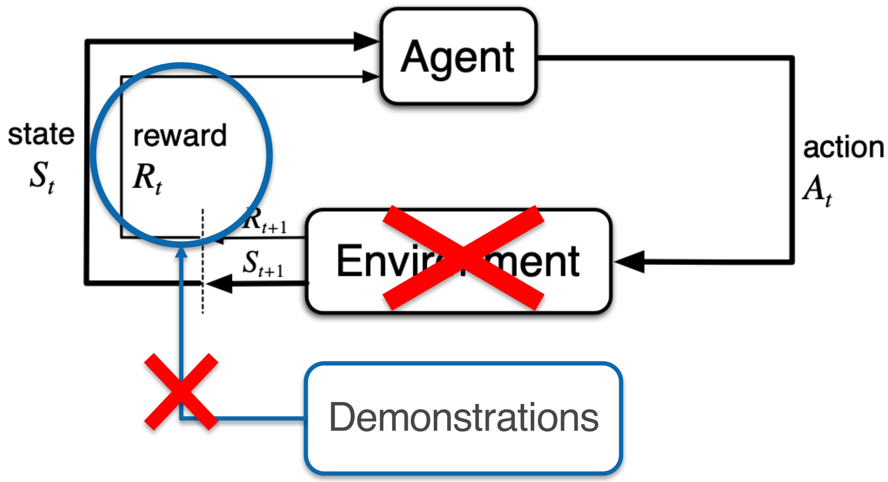
- Common Practice: Fine-tune it by online RL
  - Example: AlphaGo



Figures from Silver, David, et al. "Mastering the game of Go with deep neural networks and tree search."

Use Demo Offline

- **What to learn** from the demo?



Reward is not present in demo?

We can still infer the reward offline if we have some trajectory-level labels, e.g., success and failure

Figures adapted from Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction.

Infer Reward / Goal

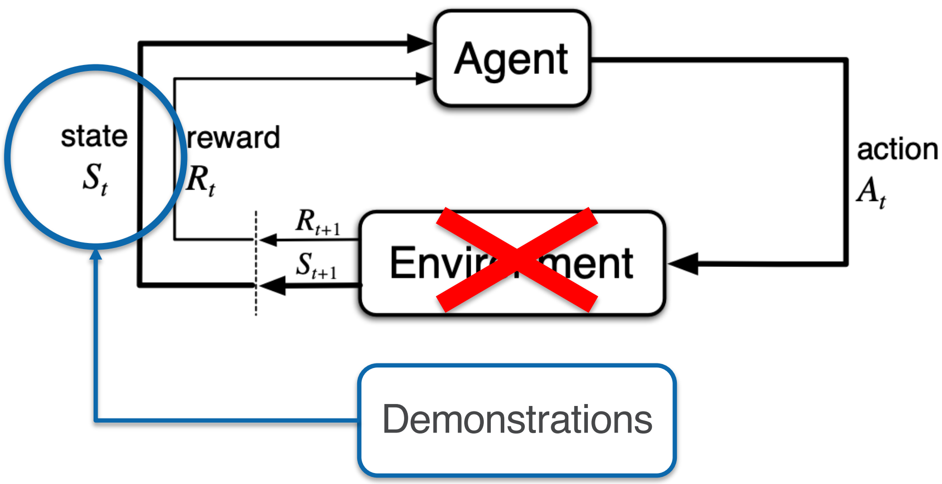
- Learn a goal classifier from demo, and regard it as a reward
  - Positive samples: the last observation in each success trajectory
  - Negative samples: observations from failure trajectories

Failure Success Success

Figures from Singh, Avi, et al. "End-to-end robotic reinforcement learning without reward engineering."

Use Demo Offline

- **What to learn** from the demo?



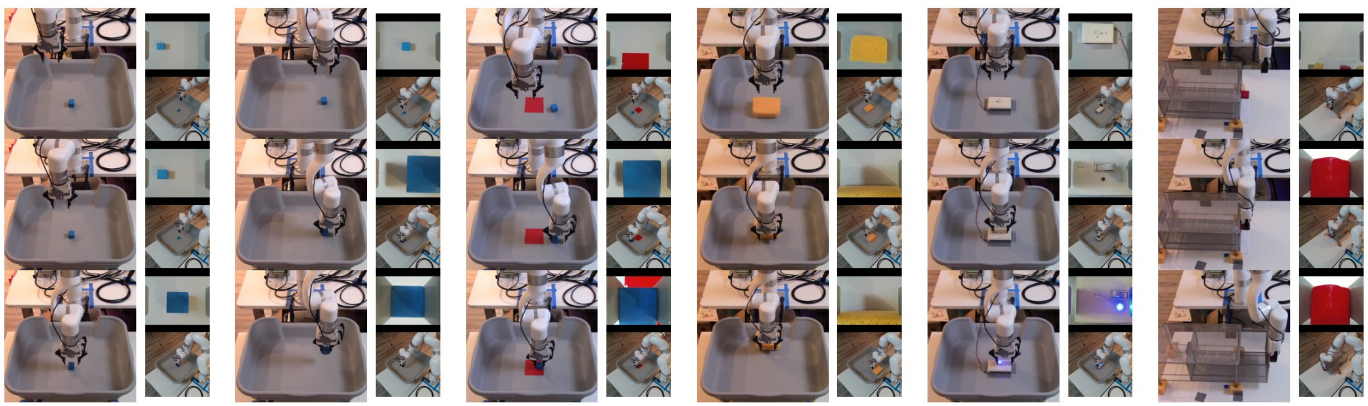
In visual RL, the observations / states usually very high-dimensional

Representation Learning

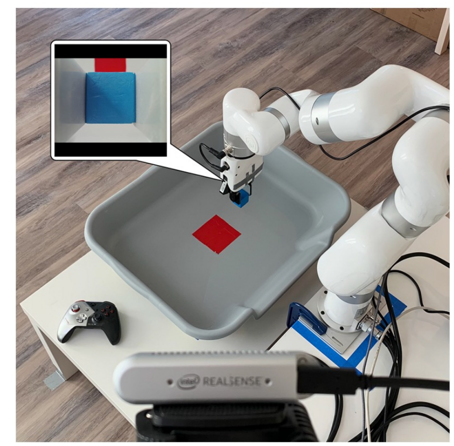
Figures adapted from Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction.

Learn Representation

- Learn representation in in-domain dataset
  - E.g., pre-train the visual encoder by contrastive learning
    - Minimize distance between positive samples and maximize distance of negative samples



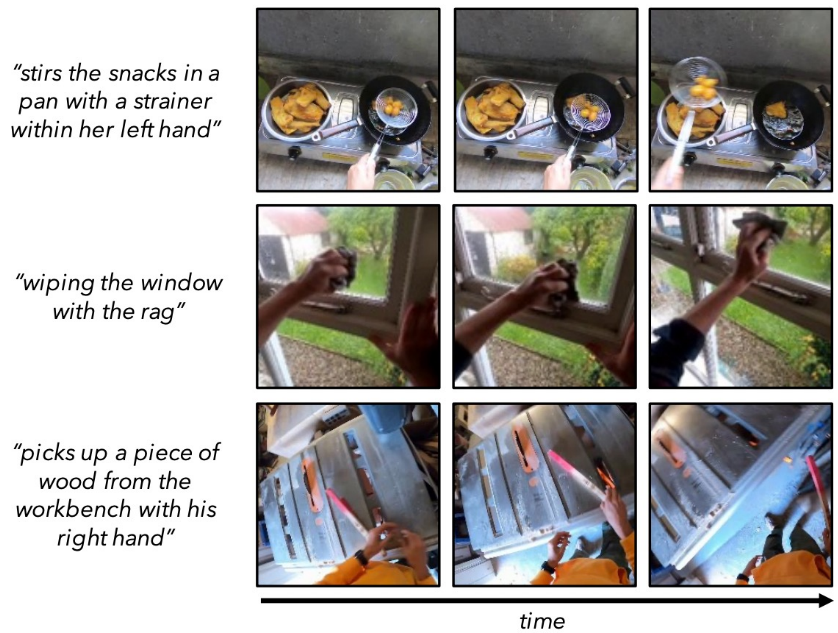
(a) Reach (b) Pickup (c) Move (d) Pull (e) Light Switch (f) Drawer Open

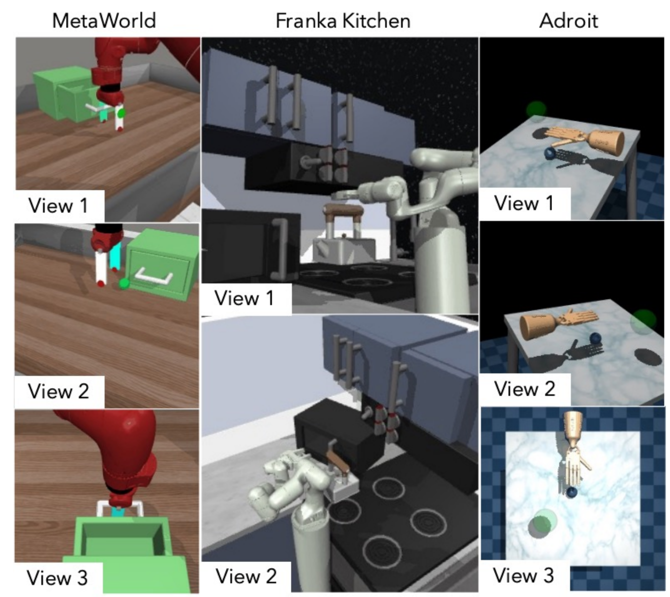


Figures from Zhan, Albert, et al. "Learning Visual Robotic Control Efficiently with Contrastive Pre-training and Data Augmentation."

Learn Representation

- Learn representation in large external datasets



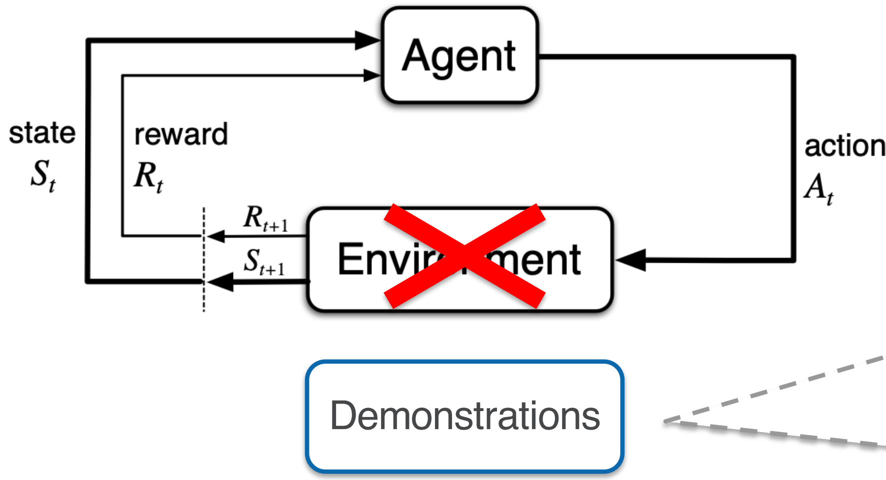


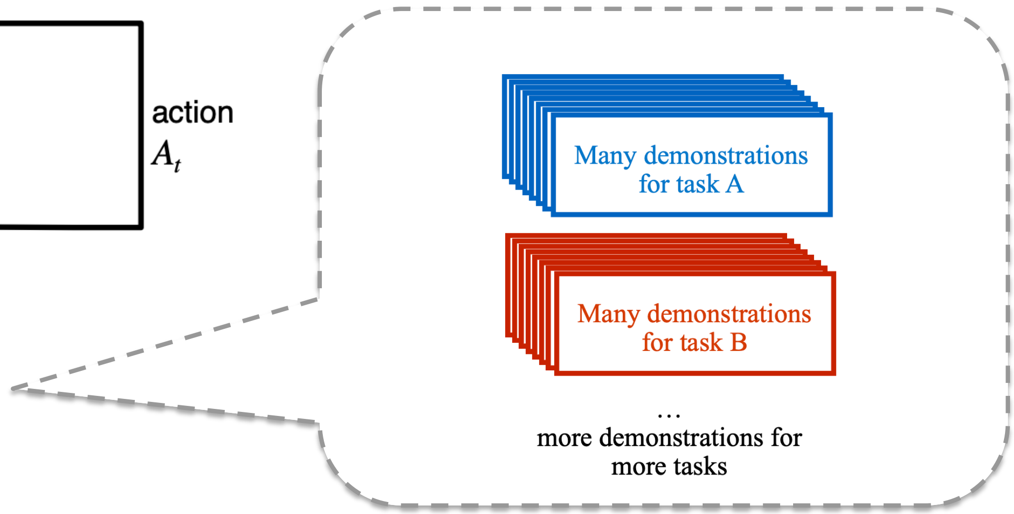
Figures from Nair, Suraj, et al. "R3m: A universal visual representation for robot manipulation."

Use Demo Offline

- **What to learn** from the demo?

What if we have demos from
many different tasks?

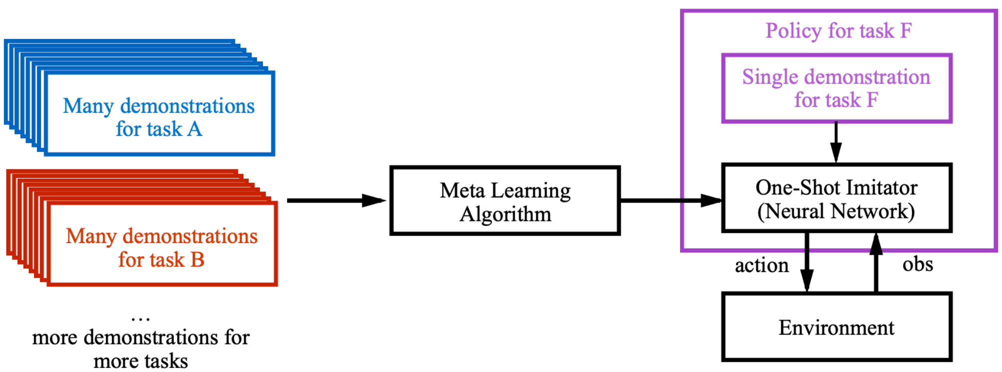




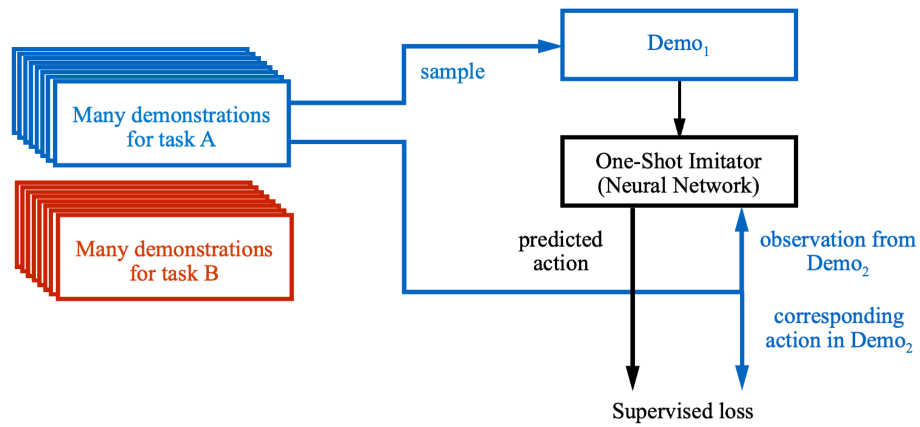
Figures adapted from Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction.

Learn Trajectory Imitator

- One-Shot Imitation Learning
  - Given a trajectory at the test time
  - Trained to imitate the trajectory, instead of completing the tasks



(b) One-Shot Imitation Learning

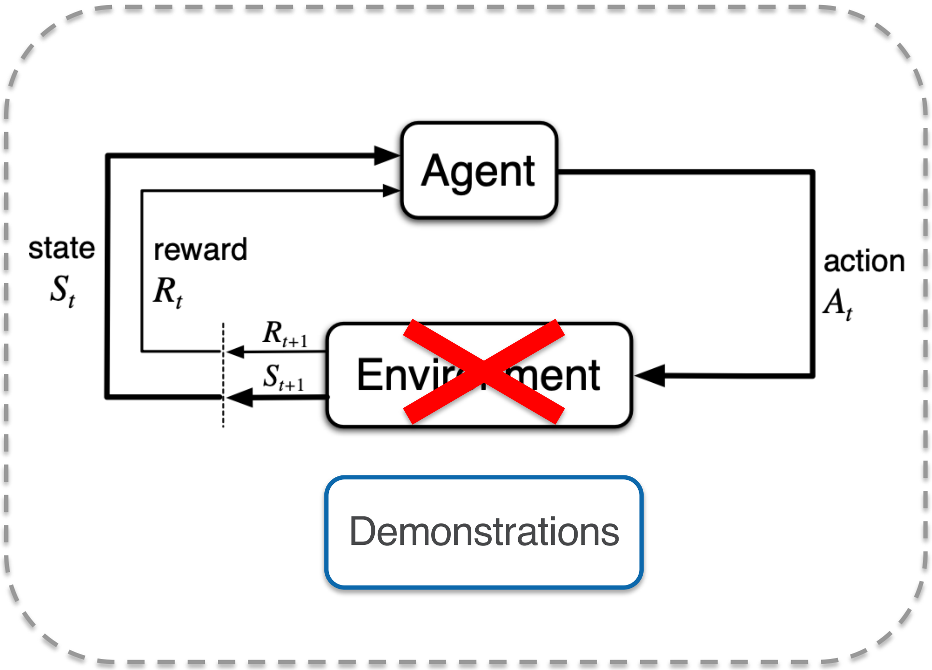


(c) Training the One-Shot Imitator

Figures from Duan, Yan, et al. "One-shot imitation learning."

Use Demo Offline

- **What to learn** from the demo?

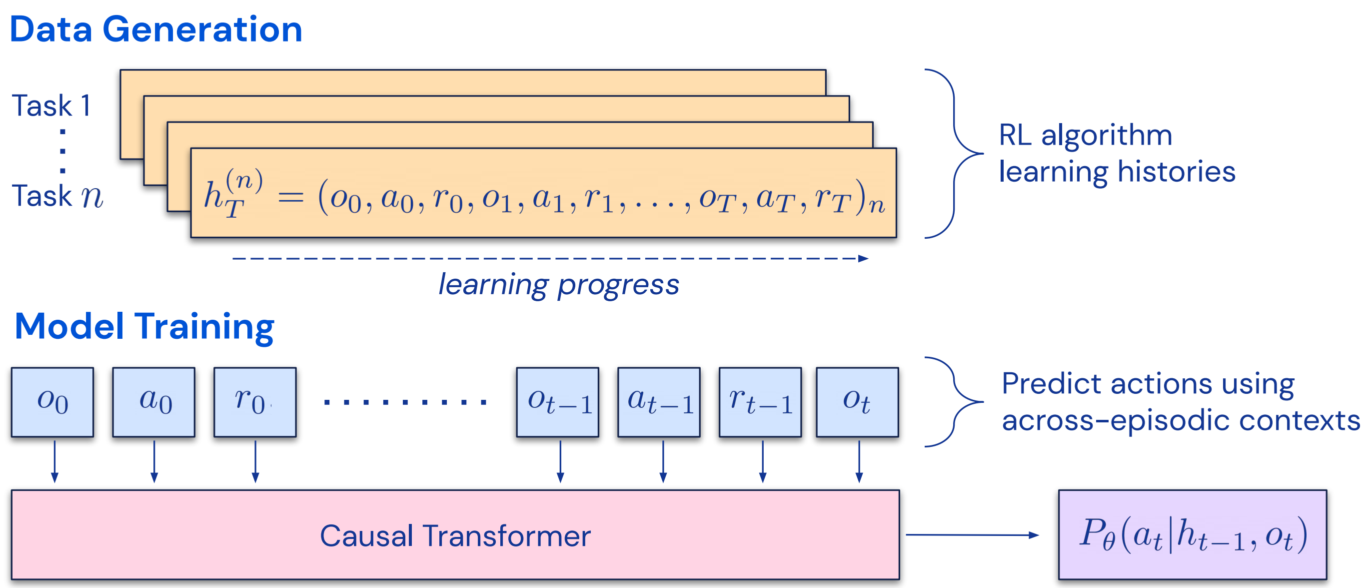


Learn the RL algorithm itself?

Figures adapted from Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction.

Learn Algorithm

- Algorithm Distillation
  - Assume the demo dataset contains the whole learning history of an agent
  - Train a transformer to predict actions given the preceding learning history



Figures from Laskin, Michael, et al. "In-context Reinforcement Learning with Algorithm Distillation."

Use Demo Offline

- **What to learn** from the demo?
  - Policy
  - Skill
  - World Model
  - Reward / Goal
  - Representation
  - Trajectory Imitator
  - Algorithm
  - …
  - Maybe there will be more creative ways to use demo offline in the future?

Outline

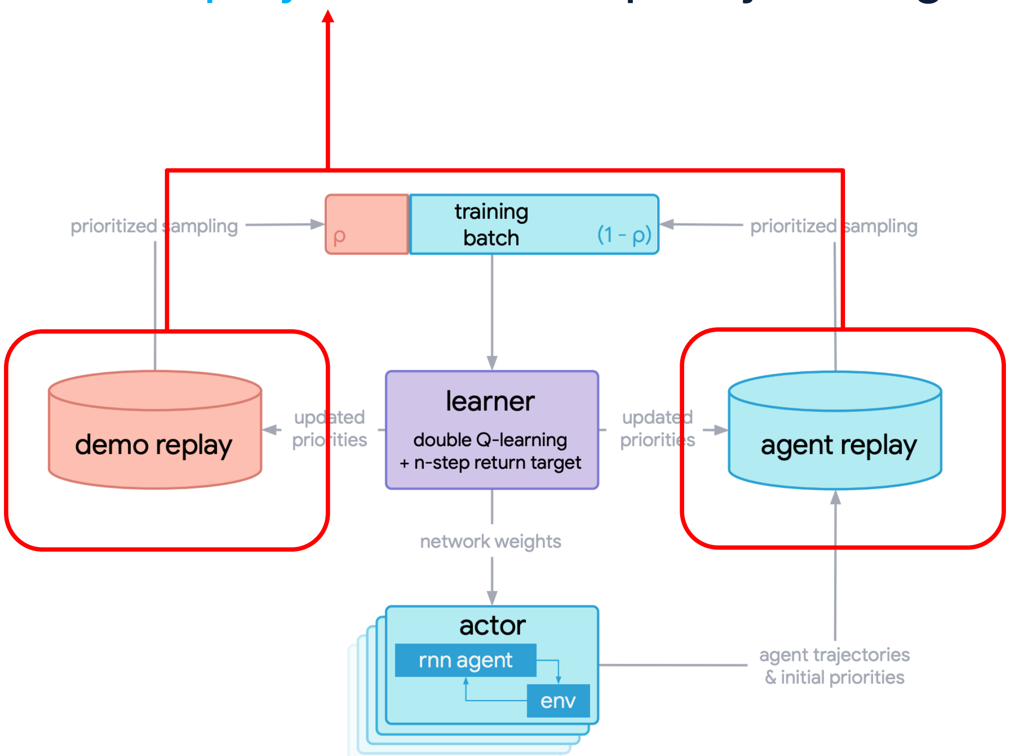
- **Use Demonstrations**
  - Offline – without interactions with environments
  - Online – with interactions with environments
- **Collect Demonstrations**
- **Future Directions**

Use Demo Online

- Utilize demo during online learning directly
- A preceding offline learning stage is optional

Demo as Off-Policy Experience

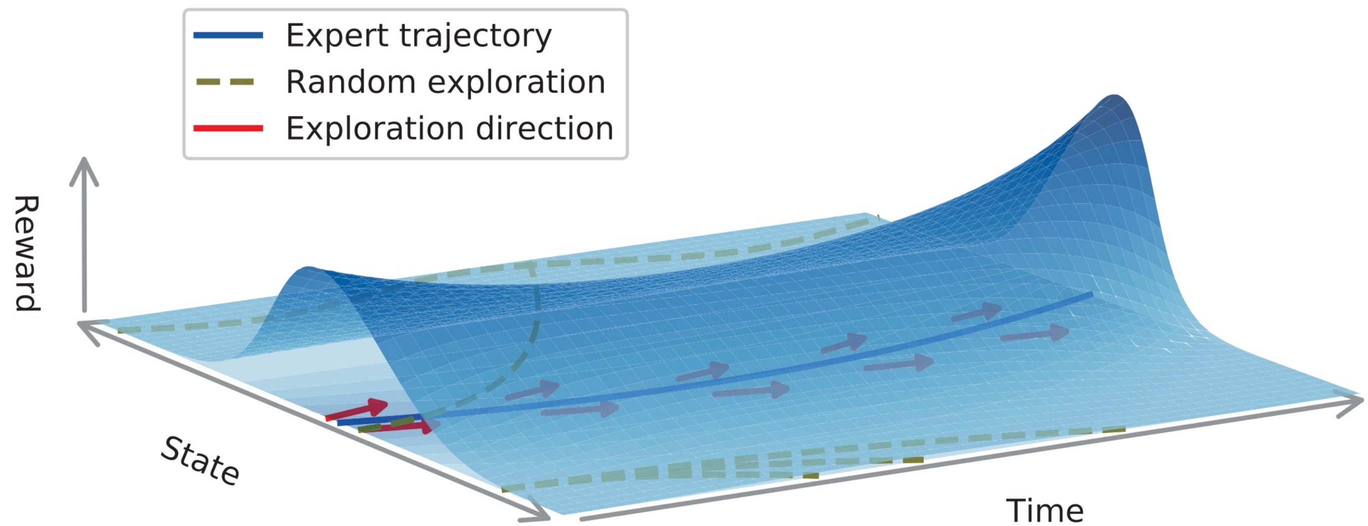
- Add demo into the replay buffer of off-policy RL algorithms



Figures from Paine, Tom Le, et al. "Making efficient use of demonstrations to solve hard exploration problems."

Demo as On-Policy Regularization

- Augment the RL objective with a regularization term
- Encourages the agent to keep the behavior close to the demo



Figures from Kang, Bingyi, Zequn Jie, and Jiashi Feng. "Policy optimization with demonstrations."

Demo as On-Policy Regularization

- Augment the RL objective with a regularization term
- Encourages the agent to keep the behavior close to the demo
- POfD (Policy Optimization from Demo)

$$\mathcal{L}\left(\pi_\theta\right)=-\eta\left(\pi_\theta\right)+\lambda_1 D_{JS}\left(\rho_\theta, \rho_E\right)$$

, where the D_{JS} denotes the Jensen-Shannon divergence, the ρ_θ and ρ_E are the occupancy measures of agent policy and expert policy, respectively.

Kang et. al, “Policy Optimization with Demonstrations”

Demo as On-Policy Regularization

- Augment the RL objective with a regularization term
- Encourages the agent to keep the behavior close to the demo
- DAPG (Demo Augmented Policy Gradient)

$$g_{aug} = \sum_{(s,a)\in\rho_\pi} \nabla_\theta \ln \pi_\theta(a|s) A^\pi(s,a) + \sum_{(s,a)\in\rho_D} \nabla_\theta \ln \pi_\theta(a|s) w(s,a)$$

,where $w(s,a)$ is a weighting function. In practice, $w(s,a)$ is implemented as a heuristic weighting function:

$$w(s,a) = \lambda_0 \lambda_1^k \max_{(s',a')\in\rho_\pi} A^\pi(s',a') \quad \forall (s,a) \in \rho_D$$

,where λ_0 and λ_1 are hyperparameters, and k is the iteration counter.

Rajeswaran et. al, “Demo Augmented Policy Gradient for Dextrous Hand Manipulation”

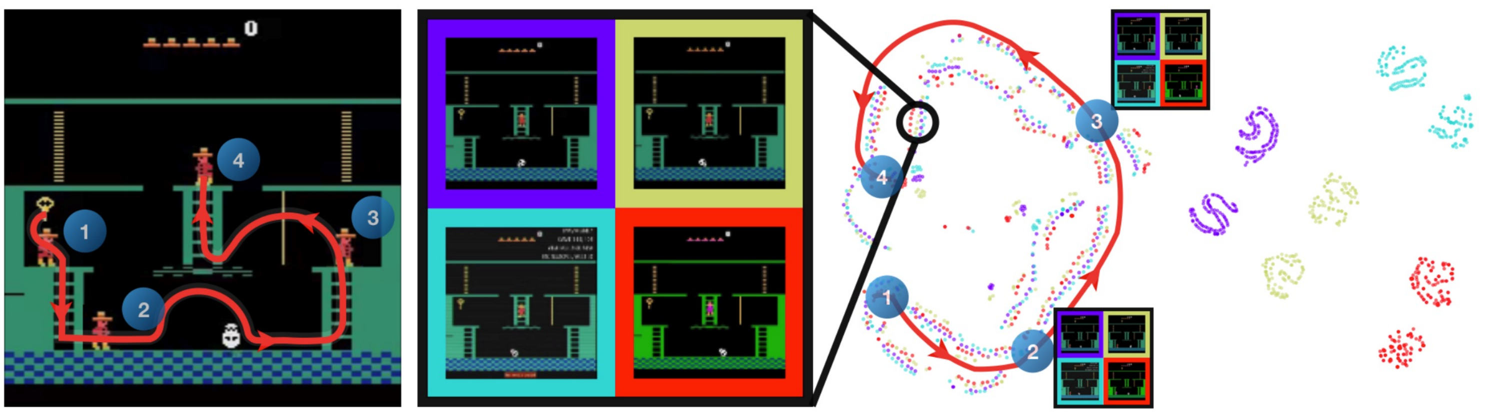
Demo as Reference for Reward

- Regularization modifies the RL learning objective
- Another natural way is to convert demo into reward, then the demo is automatically incorporated into RL objective
- Two kinds of ideas:
  - Directly define reward based on a single demo trajectory
  - Match the distribution of demonstrations, and use the divergence as the reward

Define Reward with a Single Demo

- Reward: the distance to the demo trajectory in an embedding space
- The embedding space can be learned

$$r_{\text{imitation}} = \begin{cases} 0.5 & \text{if } \bar{\phi}(v_{\text{agent}}) \cdot \bar{\phi}(v_{\text{checkpoint}}) > \alpha \\ 0.0 & \text{otherwise} \end{cases}$$

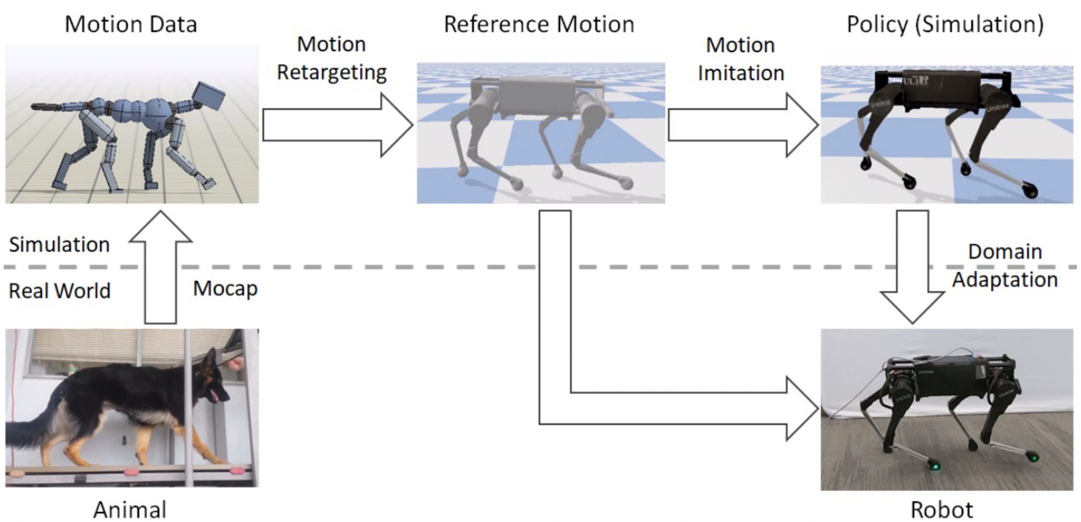


(a) An example path (b) Aligned frames (c) Our embedding (d) Pixel embedding

Figures from Playing hard exploration games by watching YouTube

Define Reward with a Single Demo

- Reward: the distance to the demo trajectory in an embedding space
- The embedding space can be manually defined



$$r_t^{\mathrm{p}} = \exp\left[-5 \sum_j ||\hat{\mathbf{q}}_t^j - \mathbf{q}_t^j||^2\right] \qquad r_t^{\mathrm{v}} = \exp\left[-0.1 \sum_j ||\hat{\dot{\mathbf{q}}}_t^j - \dot{\mathbf{q}}_t^j||^2\right]$$

$$r_t = w^{\mathrm{p}} r_t^{\mathrm{p}} + w^{\mathrm{v}} r_t^{\mathrm{v}} + w^{\mathrm{e}} r_t^{\mathrm{e}} + w^{\mathrm{rp}} r_t^{\mathrm{rp}} + w^{\mathrm{rv}} r_t^{\mathrm{rv}}$$

$$r_t^{\mathrm{e}} = \exp\left[-40 \sum_e ||\hat{\mathbf{x}}_t^e - \mathbf{x}_t^e||^2\right]$$

$$r_t^{\mathrm{rp}} = \exp\left[-20||\hat{\mathbf{x}}_t^{\mathrm{root}} - \mathbf{x}_t^{\mathrm{root}}||^2 - 10||\hat{\mathbf{q}}_t^{\mathrm{root}} - \mathbf{q}_t^{\mathrm{root}}||^2\right]$$

$$r_t^{\mathrm{rv}} = \exp\left[-2||\hat{\dot{\mathbf{x}}}_t^{\mathrm{root}} - \dot{\mathbf{x}}_t^{\mathrm{root}}||^2 - 0.2||\hat{\dot{\mathbf{q}}}_t^{\mathrm{root}} - \dot{\mathbf{q}}_t^{\mathrm{root}}||^2\right]$$

Figures from Peng, Xue Bin, et al. "Learning agile robotic locomotion skills by imitating animals."

Match the Distribution of Demo

- Reward: the divergence between agent trajectories and demo
  - Usually needs to be approximated due to the computation cost
- This is actually the idea behind most inverse RL methods

Algorithm 1: Template for IRL

Input: $\mathcal{M}\backslash_{R_E} = \langle S, A, T, \gamma \rangle$,
Set of trajectories demonstrating desired behavior:
$\mathcal{D} = \{\langle (s_0, a_0), (s_1, a_1), \ldots, (s_t, a_t) \rangle, \ldots\}$, $s_t \in S$, $a_t \in A$, $t \in \mathbb{N}$,
or expert's policy: π_E, and reward function features
Output: $\hat{R}_E$

1 Model the expert's observed behavior as the solution of an MDP whose reward function is not known;
2 Initialize the parameterized form of the reward function using any given features (linearly weighted sum of feature values, distribution over rewards, or other);
3 Solve the MDP with current reward function to generate the learned behavior or policy;
4 Update the optimization parameters to minimize the divergence between the observed behavior (or policy) and the learned behavior (policy);
5 Repeat the previous two steps till the divergence is reduced to a desired level.

Algorithm from Arora, Saurabh, and Prashant Doshi. "A survey of inverse reinforcement learning: Challenges, methods and progress."

Match the Distribution of Demo

- GAIL (Generative adversarial imitation learning)
  - Reward is Jensen-Shannon divergence, implemented similar to GANs
  - Generator: policy $\pi(a|s)$
  - Discriminator: predicts (s, a) from agent or demo
  - Discriminator has to predict source of (s, a), generator tries to fool discriminator by generating actions that look like the demo distribution

Algorithm 1 Generative adversarial imitation learning

1: **Input:** Expert trajectories $\tau_E \sim \pi_E$, initial policy and discriminator parameters θ_0, w_0
2: **for** $i = 0, 1, 2, \ldots$ **do**
3: Sample trajectories $\tau_i \sim \pi_{\theta_i}$
4: Update the discriminator parameters from w_i to w_{i+1} with the gradient

$$\hat{\mathbb{E}}_{\tau_i}[\nabla_w \log(D_w(s,a))] + \hat{\mathbb{E}}_{\tau_E}[\nabla_w \log(1 - D_w(s,a))] \tag{17}$$

5: Take a policy step from θ_i to θ_{i+1}, using the TRPO rule with cost function $\log(D_{w_{i+1}}(s,a))$. Specifically, take a KL-constrained natural gradient step with

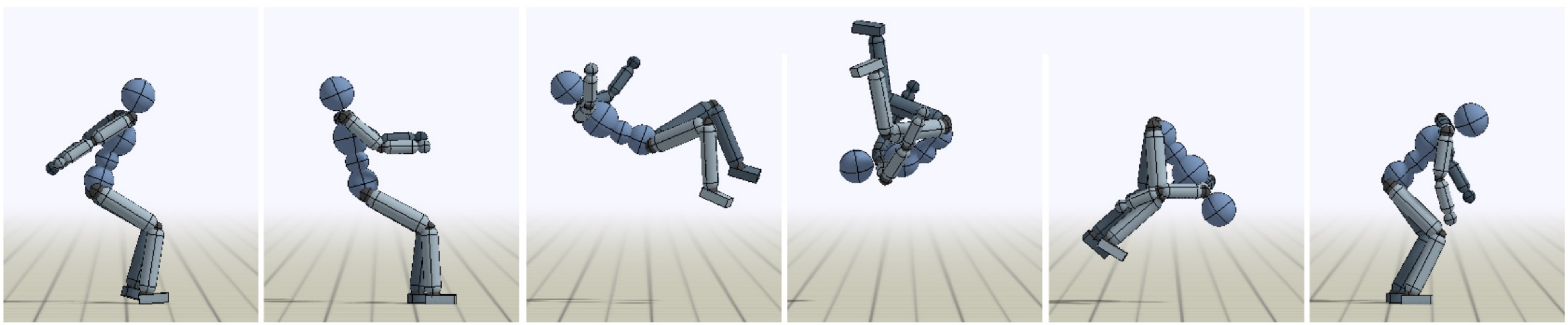
$$\begin{aligned} &\hat{\mathbb{E}}_{\tau_i}[\nabla_\theta \log \pi_\theta(a|s) Q(s,a)] - \lambda \nabla_\theta H(\pi_\theta), \\ &\text{where } Q(\bar{s}, \bar{a}) = \hat{\mathbb{E}}_{\tau_i}[\log(D_{w_{i+1}}(s,a)) \mid s_0 = \bar{s}, a_0 = \bar{a}] \end{aligned} \tag{18}$$

6: **end for**

Algorithm from Ho, Jonathan, and Stefano Ermon. "Generative adversarial imitation learning."

Demo as Curriculum of Start States

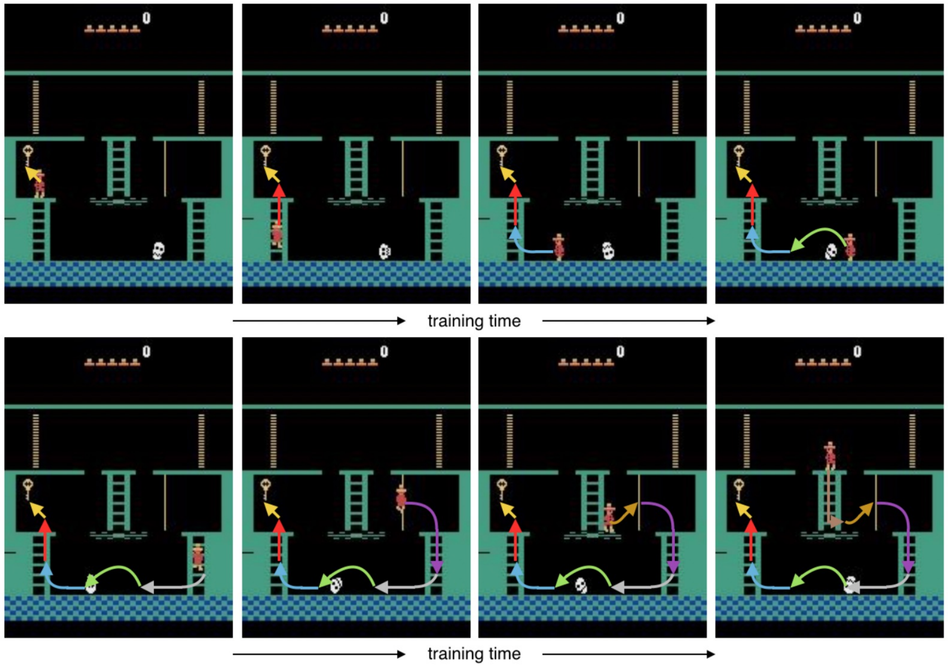
- Assume the environment is a simulator that we can fully control
- Reset to the states in demo, and start to explore from there
  - Uniformly sample states in demo as the start states



Figures from Peng, Xue Bin, et al. "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills."

Demo as Curriculum of Start States

- Assume the environment is a simulator that we can fully control
- Reset to the states in demo, and start to explore from there
  - Or start from the end of the demo, and gradually go backwards



Figures from Salimans, Tim, and Richard Chen. "Learning montezuma's revenge from a single demonstration."

Outline

- **Use Demonstrations**
  - Offline – without interactions with environments
  - Online – with interactions with environments
- **Collect Demonstrations**
- **Future Directions**

Collect Demo

- Use Embodied AI as an example domain
  - Demo can be collected in various ways
    - By human or by robots
    - In simulators or in the real world
  - One of the most popular domains to use demonstrations

- Focus on acquiring expert demonstrations
  - Non-expert demo are relatively easy to get

Figures from WALL·E

Collect Demo

- Embodiment: the "physical body" of the agent
- Operator: the "brain" to control the agent

Embodiment \ Operator	Robot	Human
Robot	✓	✓
Human	✗	✓

Figures from WALL·E

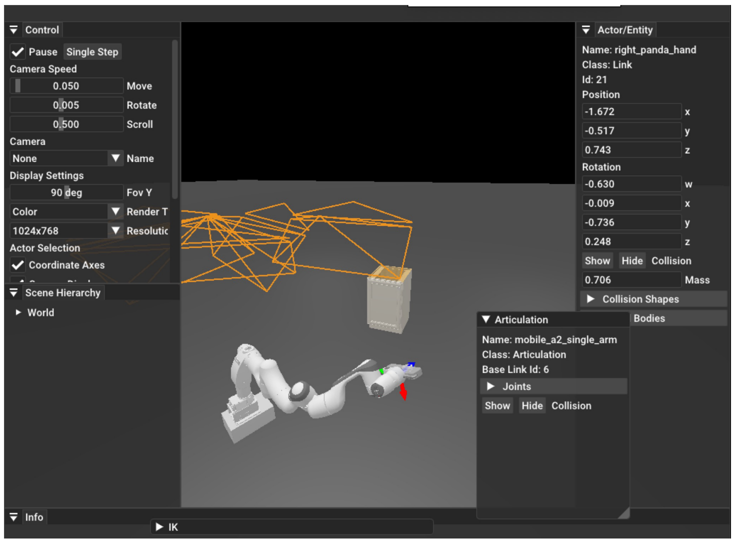
Collect Demo

- Embodiment: the “physical body” of the agent
- Operator: the “brain” to control the agent

Embodiment \ Operator	Robot	Human
Robot	✓	✓
Human	✗	✓

Usually done by teleoperation (remote control)

Teleoperation – Basic Devices



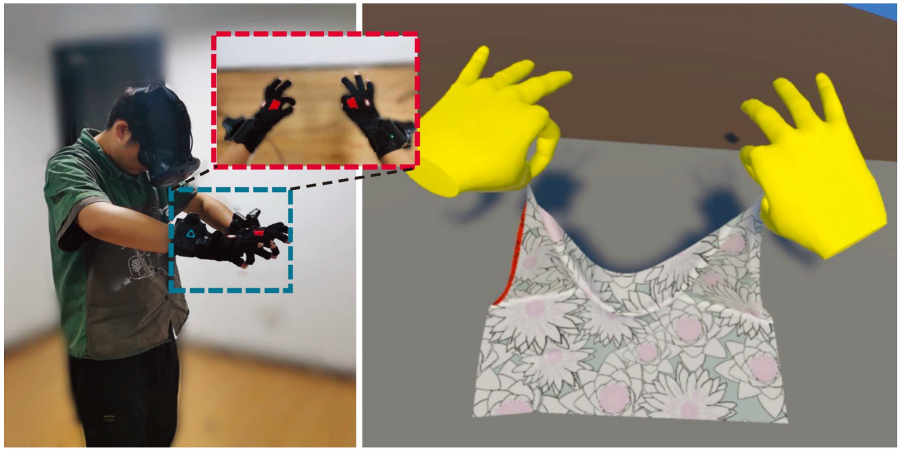
Keyboard + Mouse
(Many Simulation Environments)

Smartphone (RoboTurk)

Teleoperation – Virtual Reality

- VR is widely adopted in many simulation environments

Headset + Regular Hand Controllers
(iGibson 2)



Headset + Motion Capture Gloves
(RFUniverse)

Teleoperation – Virtual Reality

- VR is also used when collecting robot demo in the real world

VR remotes + joystick
(RT-1)

Video from Brohan, Anthony, et al. "RT-1: Robotics Transformer for Real-World Control at Scale."

Collect Demo

- Embodiment: the "physical body" of the agent
- Operator: the "brain" to control the agent

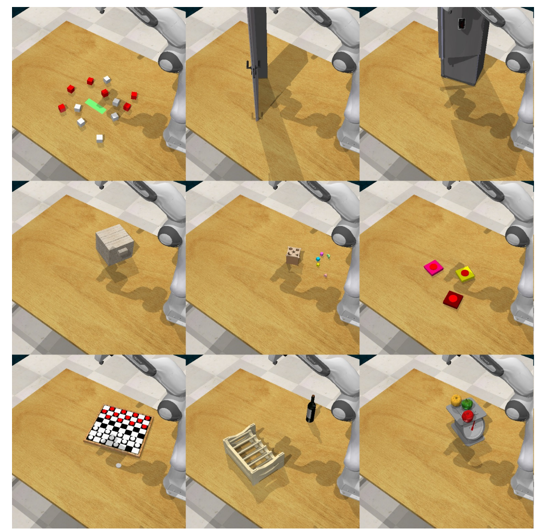
Embodiment \ Operator	Robot	Human
Robot	✓	
Human	✗	✓

Build an autonomous system to control the robot

Autonomous System – Planning

- Use planners to generate demo (assume world model is known)

Symbolic Planner for High-level Tasks (ALFRED)

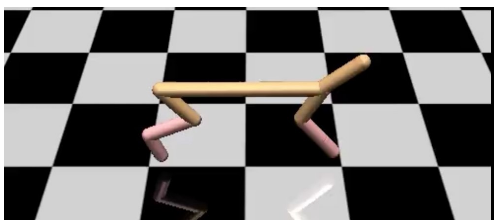


Motion Planner for Low-level Tasks (RLBench)

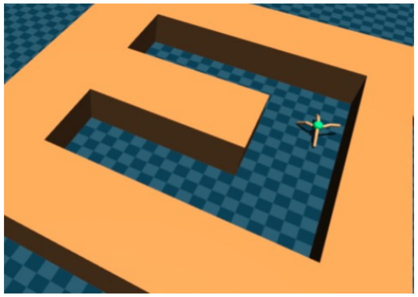
Figures from

Autonomous System – Learning

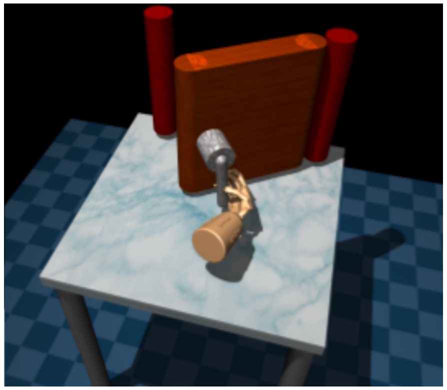
- Learning-based methods
- Design different methods for different tasks



RL alone is already enough for many tasks



Waypoint generator + Goal-reaching policy from RL



DAPG + a few human demo

Figures from Fu, Justin, et al. "D4rl: Datasets for deep data-driven reinforcement learning."

Autonomous System – Self-supervised

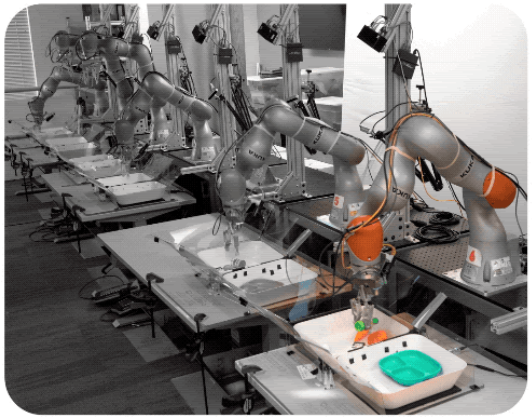
- Self-supervised system in the real world

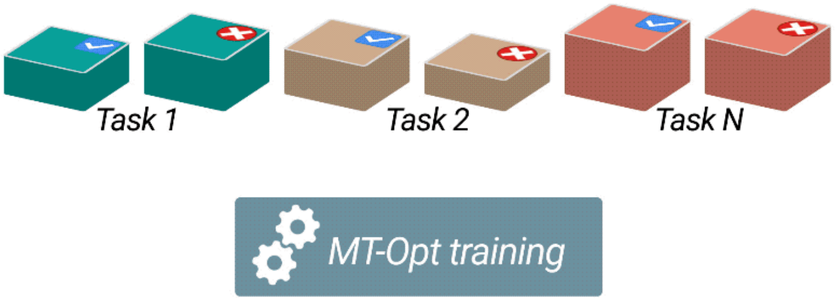
- **QT-opt**
- Run RL in the real world
- Vision system to get a sparse reward

Video from Kalashnikov, Dmitry, et al. "Scalable deep reinforcement learning for vision-based robotic manipulation."

Autonomous System – Self-supervised

- MT-Opt
  - Reset by special boxes
  - Success detectors trained on data from all tasks
  - Use the solutions to easier tasks to bootstrap learning of more complex tasks





Video from Kalashnikov, Dmitry, et al. "Mt-opt: Continuous multi-task robotic reinforcement learning at scale."

Collect Demo

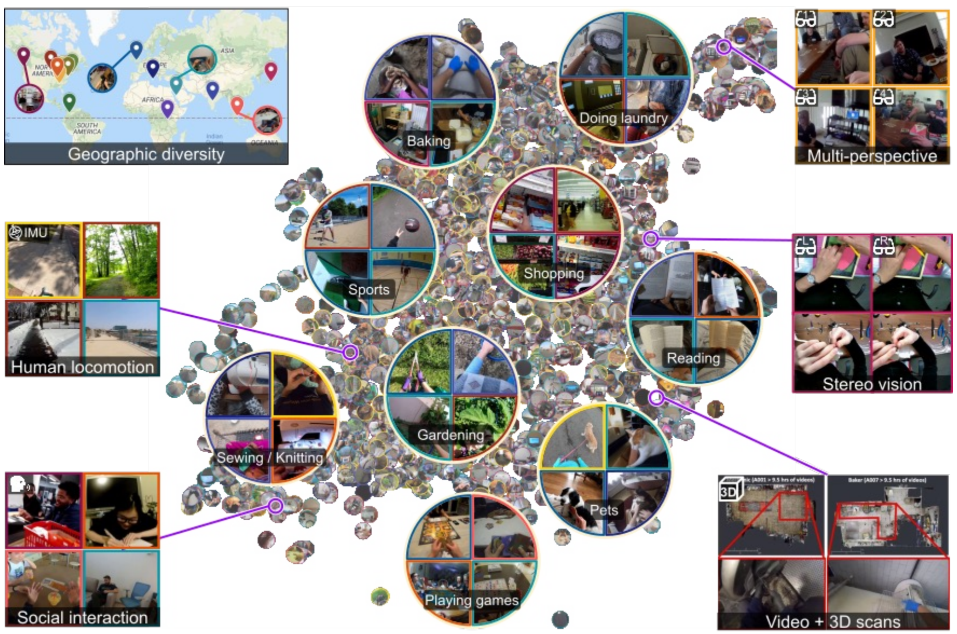
- Embodiment: the "physical body" of the agent
- Operator: the "brain" to control the agent

Embodiment \ Operator	Robot	Human
Robot	✓	✓
Human	✗	✓

Robot can also learn from datasets of human activities

Human Demo Datasets

- Ego4D
  - Ego-centric, multi-modal dataset of human activities



Figures from Grauman, Kristen, et al. "Ego4d: Around the world in 3,000 hours of egocentric video."

Human Demo Datasets

- RoboTube
  - Human video dataset + its digital twin in simulation environment

Figures from Fu, Haoyuan, et al. "RoboTube: Learning Household Manipulation from Human Videos with Simulated Twin Environments."

Outline

- **Use Demonstrations**
  - Offline – without interactions with environments
  - Online – with interactions with environments
- **Collect Demonstrations**
- **Future Directions**

Outline

- **Background**
- **Use Demonstrations**
  - Offline – without interactions with environments
  - Online – with interactions with environments
- **Future Directions**

Future Directions

- Scale up demo collection
  - Teleoperation-based approaches
    - Pros: provides high-quality and diverse demo
    - Cons: very costly, hard to scale up
  - Learning-based autonomous data collection pipelines
    - Pros: generates unlimited data, easier to scale up
    - Cons: not strong enough to solve some complex tasks, quality of demo is an issue
  - Combine them together?
    - Human-in-the-loop autonomous system which improves itself over time?
    - Still a long way to go…

Future Directions

- What kinds of demo do we really need?
  - Quality: always need near-optimal demo?
  - Modality: learn robot policies from videos and language descriptions?
  - Embodiment: learn robot policies from human?
  - Content: always need actions? rewards?
  - …

Future Directions

- Combine offline learning from demo with online learning
  - Though there have been several preliminary attempts, the problem is still not solved yet
  - Demo can come in different forms and different qualities
  - Solutions might need to be designed for each different scenario
  - An interesting problem to study
    - Low to learn from non-optimal, cross-domain, partially observed demonstrations
    - This kind of demo is what we usually get in the real world

Thank you!