Robot Perception and Learning

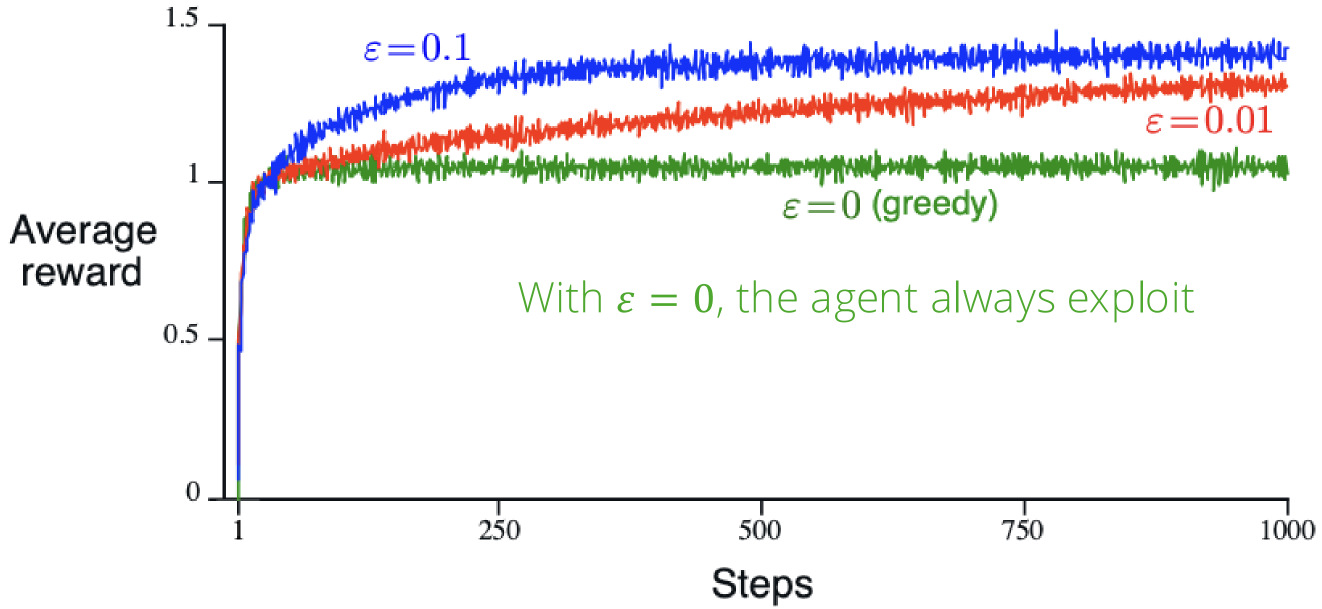
Exploration and Visual Imitation Learning

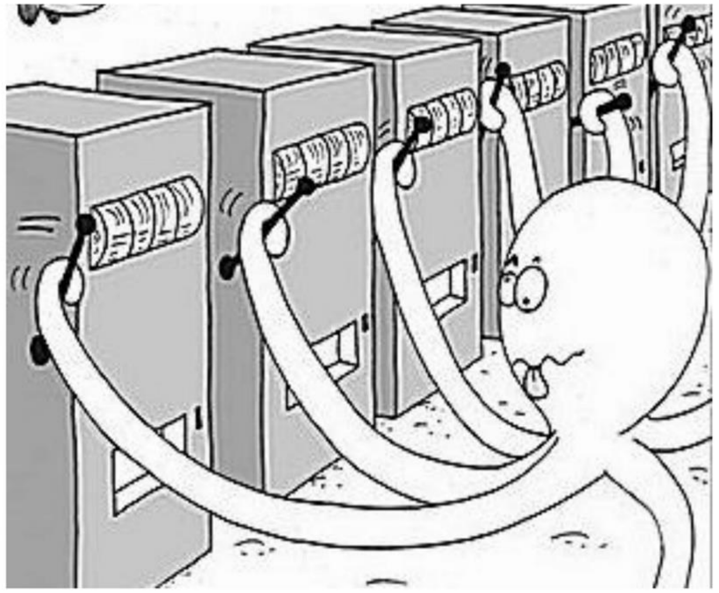
Tsung-Wei Ke

Fall 2025

Exploration is the Core Problem in Reinforcement Learning

- The goal of reinforcement learning is to maximize the return
- Exploration: the agent experiments with novel strategies that may improve returns in the long run
- Exploitation: the agent maximizes rewards through behavior that is known to be successful
- Exploitation makes sense only when the agent had stumbled into high-reward states

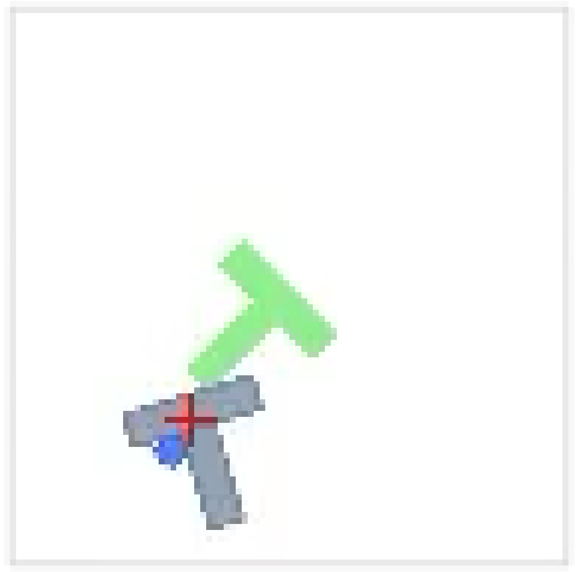




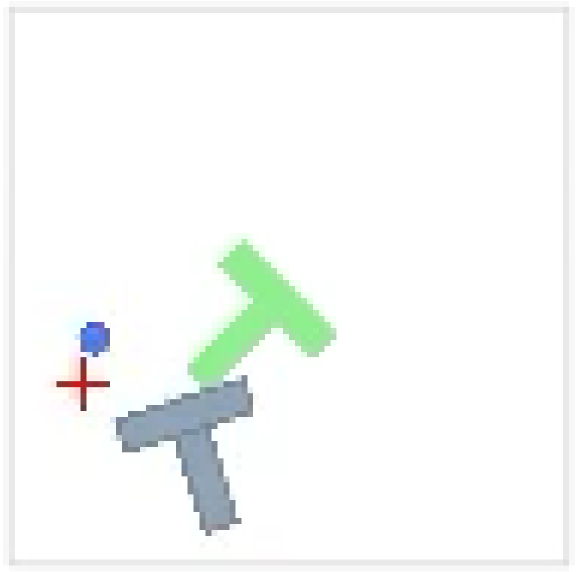
Exploration is the Core Problem in Reinforcement Learning

- The goal of reinforcement learning is to maximize the return
- Exploration: the agent experiments with novel strategies that may improve returns in the long run
- Exploitation: the agent maximizes rewards through behavior that is known to be successful
- Exploitation makes sense only when the agent had stumbled into high-reward states
- Oftentimes, the reward is extremely sparse (e.g. Success vs. Failure). Stumbling into a goal state is futile for all but the simplest of environments.

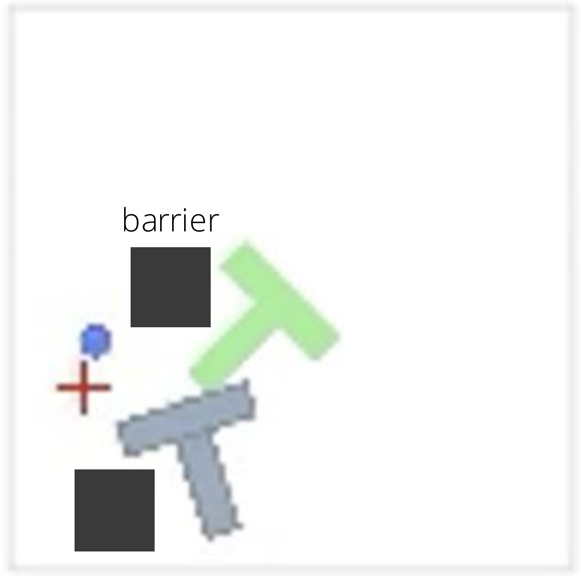
Easy env



Hard env, need more exploration



How about this?

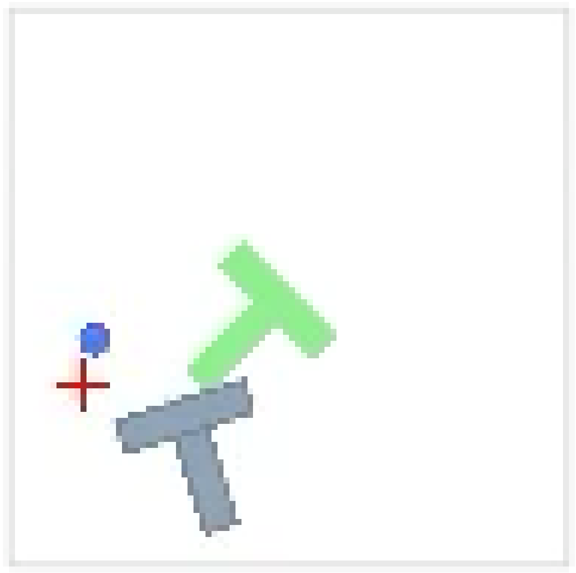


A Naïve Solution: Shaping Rewards

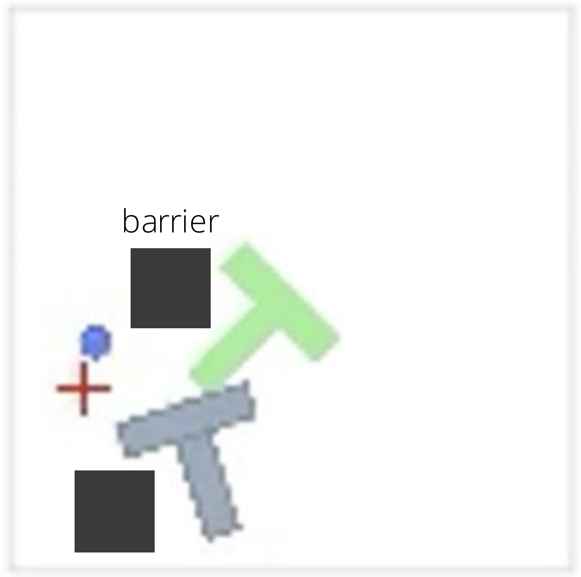
- Rewards in an easy env: success / Failure
- Rewards in a hard env: success / Failure, proximity to the object, overlapping of the object and the target
- Rewards in a very hard env: success / Failure, proximity to the object, overlapping of the object and the target, distance to the barrier, and MORE

Easy env

Hard env, need more exploration

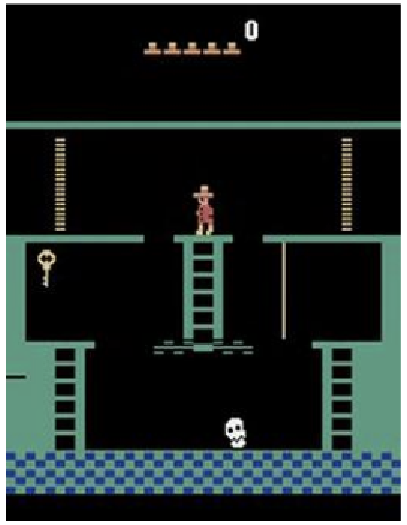


How about this?



Shaping Rewards Require Prior Knowledge on the Task

Montezuma's revenge



- Getting key = reward
- Opening door = reward
- Getting killed by skull = nothing (is it good? bad?)
- Finishing the game only weakly correlates with rewarding events
- We know what to do because we **understand** what these sprites mean!

Shaping Rewards Require Prior Knowledge on the Task What if the Task is Extremely Complex…

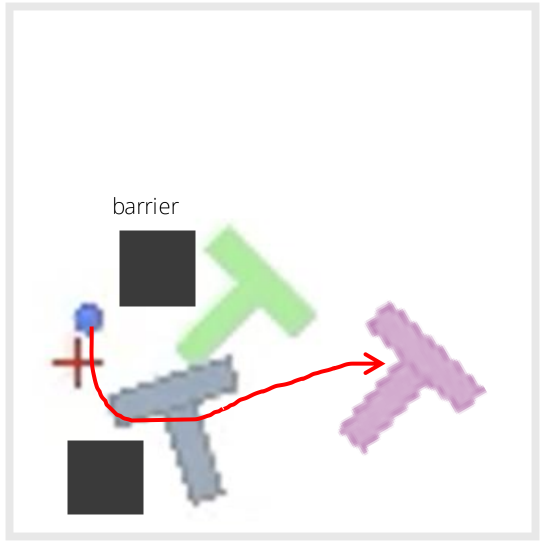
Mao

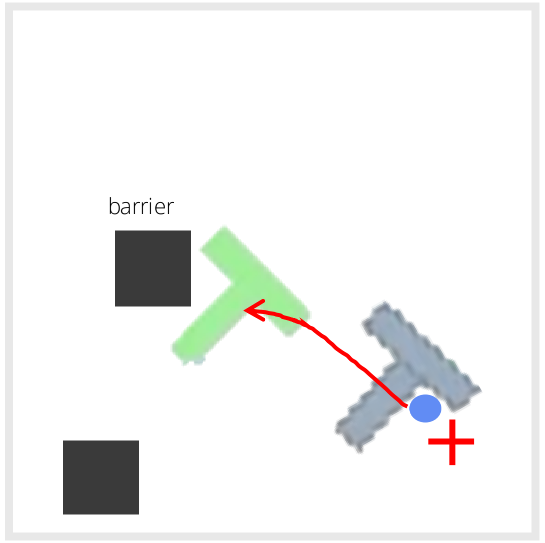
- “the only rule you may be told is this one”
- Incur a penalty when you break a rule
- Can only discover rules through trial and error
- Rules don’t always make sense to you

- Temporally extended tasks like Montezuma’s revenge become increasingly difficult based on
  - How extended the task is
  - How little you know about the rules
- Imagine if your goal in life was to win 50 games of Mao…
- (and you didn’t know this in advance)

Slide adapted from S. Levine

Let's Rethink about the Problem. Can We Discover the Intended Behavior without Rewards?





- Intended behaviors:
  - ➢ Avoid the barrier
  - ➢ Move the object to spacious regions
  - ➢ Move the object to the target location

- Exploration is all you need:
  - ➢ Try out new behaviors with the hope of discovering high rewards (if given)
  - ➢ Explore efficiently once we know what we do not know
  - ➢ With what we learn about the world, we can tackle new tasks more effectively

Exploration is All You Need… No Reward RL

Image credit S. Levine

- Babies are excellent explorers. They try out different body poses. They play with different objects.
  - ➢ Learn skills unsupervisedly
  - ➢ Learn the dynamics of the environment

- What are the factors that drive babies to explore?

- In other words, how can we do exploration more efficiently and effectively?

Intrinsic vs. Extrinsic Motivations

- Motivation: “Forces” that energize an organism to act and that direct its activity

- Extrinsic Motivation: being moved to do something because of some external reward ($$, a prize, etc.).
  - ➢ Problem: such rewards are sparse..

- Intrinsic Motivation: being moved to do something because it is inherently enjoyable (curiosity, exploration, novelty, surprise, incongruity, complexity…)
  - ➢ Gain: Task independent! Free of human supervision, no need to code up reward functions to incentivize the agent. A general loss functions that drives learning.

Slide credit K. Fragkiadaki

Intrinsic Motivation vs. Intrinsic Necessity

- Curiosity vs. Survival

"As knowledge accumulated about the conditions that govern exploratory behavior and about how quickly it appears after birth, it seemed less and *less likely that this behavior could be a derivative of hunger, thirst, sexual appetite, pain, fear of pain, and the like*, or that stimuli sought through exploration are welcomed because they have previously accompanied satisfaction of these drives."

D. E. Berlyne, *Curiosity and Exploration*, Science, 1966

How to Model Exploration?

- We can re-formulate the objective of reinforcement learning:

$$R^t(s,a,s') = r\underbrace{(s,a,s')}_{\substack{\text{extrinsic reward}\\ \text{(related to the task)}}} + \underbrace{\mathcal{B}^t(s,a,}_{\substack{\text{intrinsic reward}\\ \text{(unrelated to the task)}}}s')$$

- Types of intrinsic rewards:
  1. Data-based exploration (Optimistic exploration)
  2. Knowledge-based exploration
  3. Information gain exploration
  4. Reachability exploration
  5. Posterior sampling for exploration

Data-based Exploration

- Idea: Maximize the coverage and minimize the uncertainty of data distribution
- For example, deciding the exploration bonus based by counting the times of visiting the action / state
  - Upper Confidence Bounds (UCB)

$$A_t \doteq \underset{a}{\arg\max} \left[Q_t(a) + c\sqrt{\frac{\log t}{N_t(a)}} \right]$$

- t: how many times I have played any action,
- $N_t(a)$: how many times I have played action a in t interactions

Data-based Exploration

- Idea: Maximize the coverage and minimize the uncertainty of data distribution
- For example, deciding the exploration bonus based by counting the times of visiting the action / state
    - ➢ Upper Confidence Bounds (UCB)

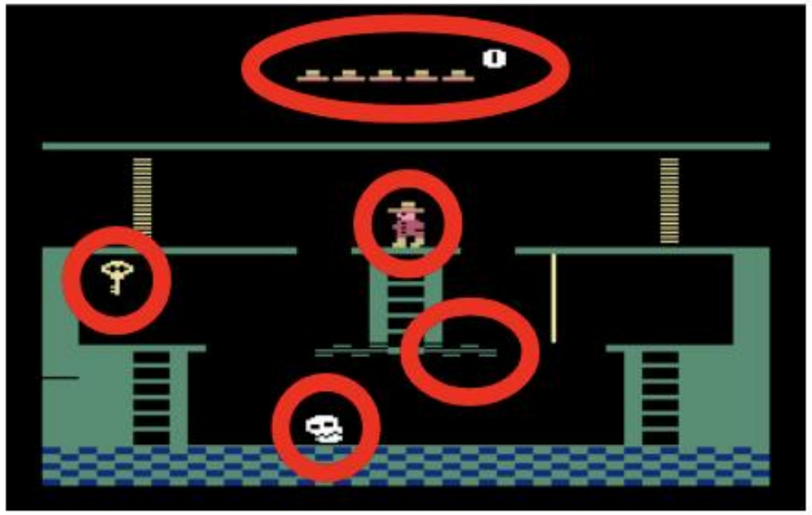
$$A_t \doteq \underset{a}{\operatorname{argmax}} \left[Q_t(a) + c\sqrt{\frac{\log t}{N_t(a)}} \right]$$

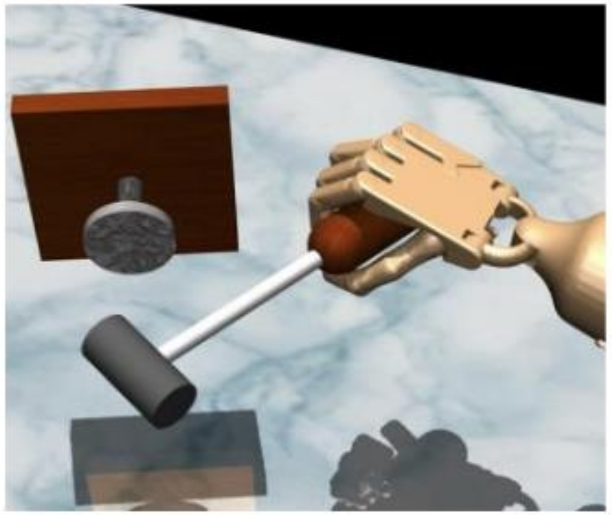
- We can count the times of visiting the state instead (select the action that goes to the least-frequently visited state)
- A strong assumption of discrete action / state space

- t: how many times I have played any action,
- $N_t(a)$: how many times I have played action a in t interactions

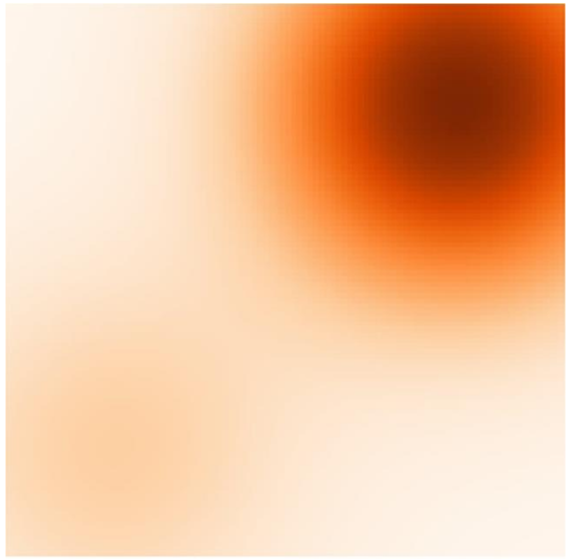
The Trouble with Counts

- How to count continuous states/actions?
  - ➢ We never see the same thing twice
  - ➢ Some states are more similar than others





Idea 1: Fitting Generative Models



- Fit a generative model that captures the distribution of seen data $p_\theta(x)$ (e.g. diffusion models, energy-based models …)
  - ➢ $p_\theta(x)$ is high for seen data or unseen but similar data

- Can we use $p_\theta(x)$ to get a "pseudo-count"?

<u>The true probability:</u>

$$P(x) = \frac{N(x)}{n}$$

count of x

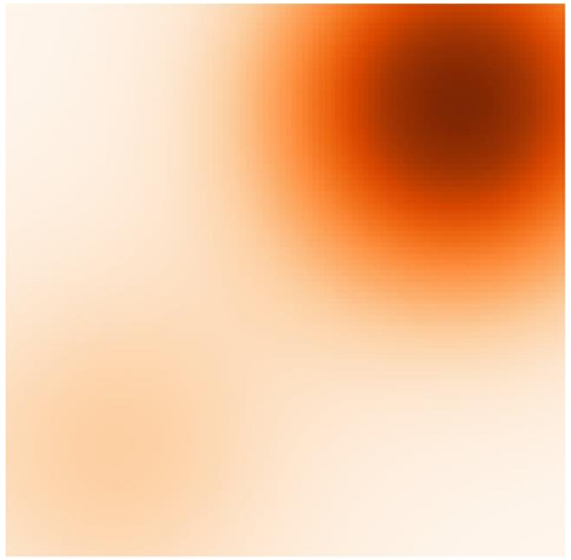
total count

<u>After seeing x:</u>

$$P'(x) = \frac{N(x)+1}{n+1}$$

Slide adapted from S. Levine

Idea 1: Fitting Generative Models



- Fit a generative model that captures the distribution of seen data $p_\theta(x)$ (e.g. diffusion models, energy-based models ...)
  - ➢ $p_\theta(x)$ is high for seen data or unseen but similar data
- Can we use $p_\theta(x)$ to get a “pseudo-count”?

<u>The true probability:</u>

$$p_\theta(x) = \frac{\hat{N}(x)}{\hat{n}}$$

$\hat{N}(x) \rightarrow$ pseudo count of x

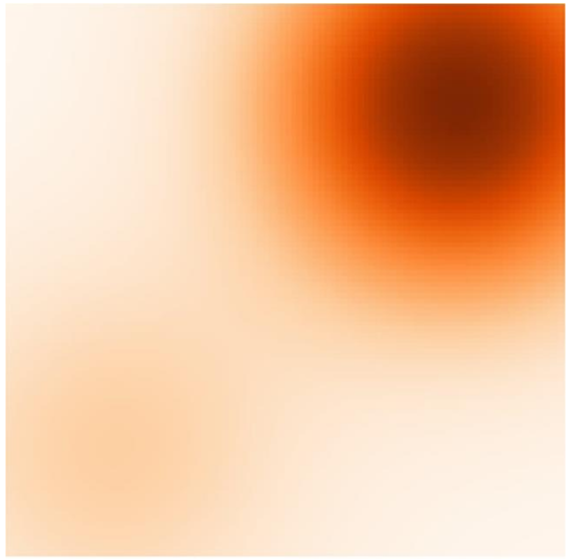
$\hat{n} \rightarrow$ total pseudo count

<u>After seeing x:</u>

$$p_{\theta'}(x) = \frac{\hat{N}(x)+1}{\hat{n}+1}$$

Slide adapted from S. Levine

Idea 1: Fitting Generative Models



- Fit a generative model that captures the distribution of seen data $p_\theta(x)$ (e.g. diffusion models, energy-based models …)
  - ➢ $p_\theta(x)$ is high for seen data or unseen but similar data

- Can we use $p_\theta(x)$ to get a "pseudo-count"?

<u>The true probability:</u>

$$p_\theta(x) = \frac{\hat{N}(x)}{\hat{n}}$$

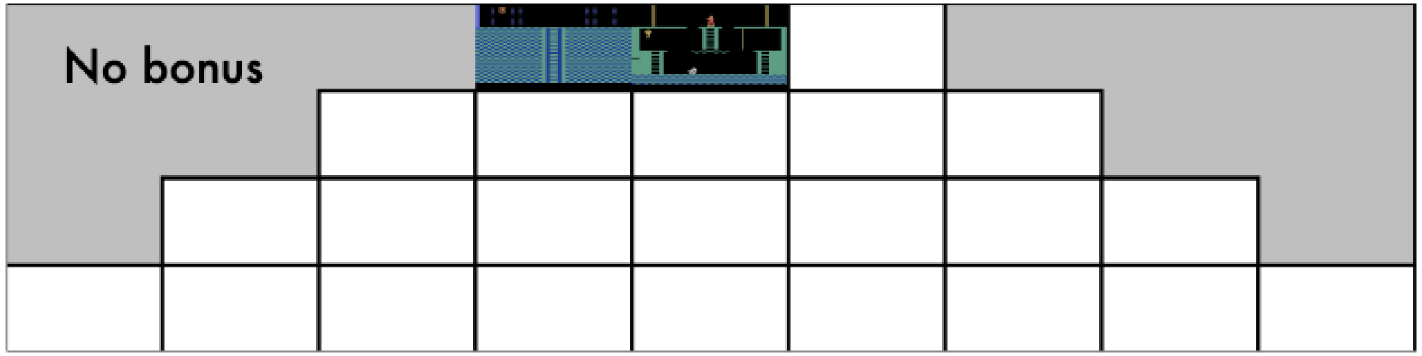
$$\hat{N}(x) = \hat{n}\, p_\theta(x)$$

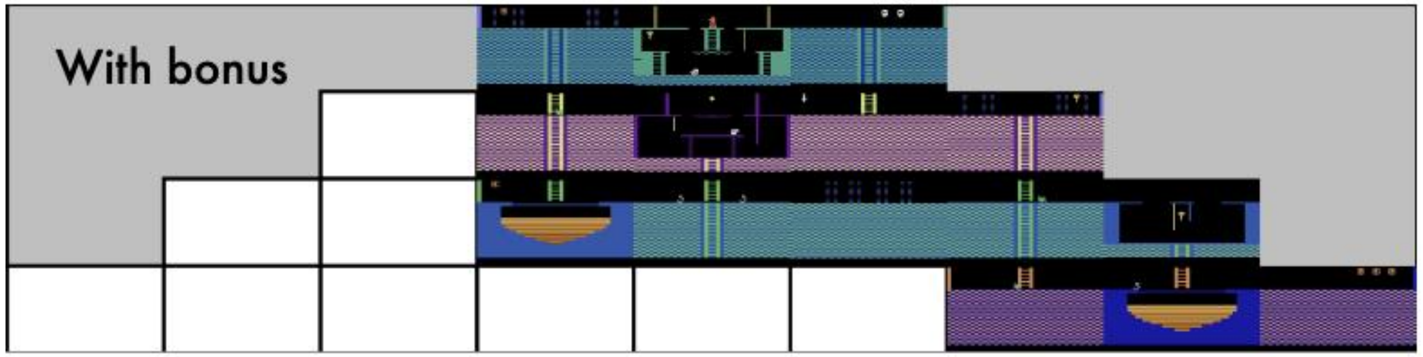
<u>After seeing x:</u>

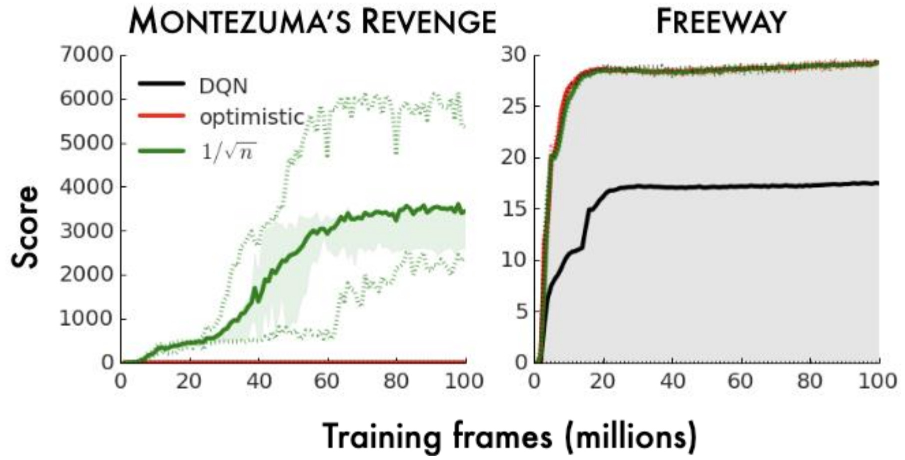
$$p_{\theta'}(x) = \frac{\hat{N}(x) + 1}{\hat{n} + 1}$$

$$\hat{n} = \frac{1 - p_{\theta'}(x)}{p_{\theta'}(x) - p_\theta(x)} p_\theta(x)$$

Does It Work?


No bonus

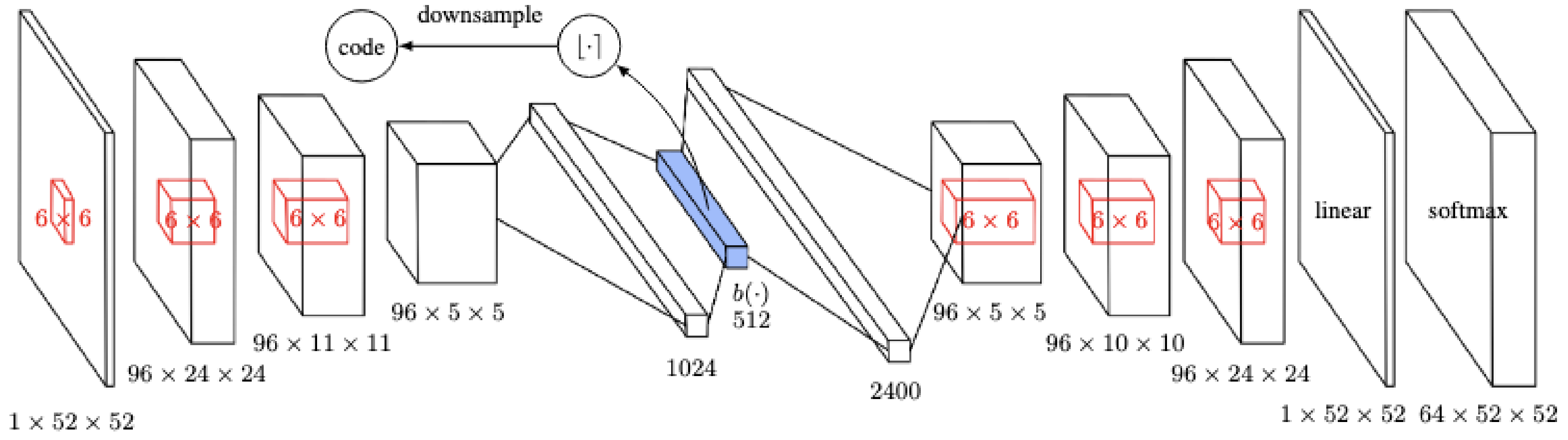

With bonus



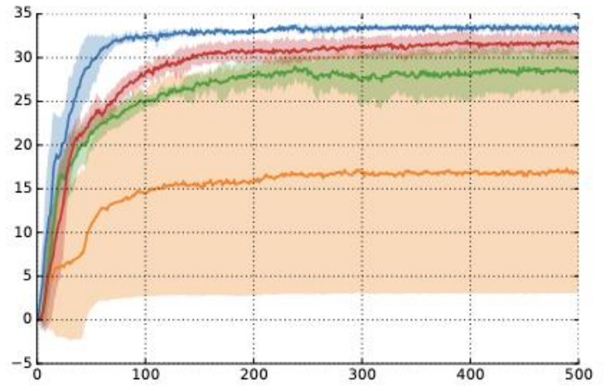
MONTEZUMA'S REVENGE
FREEWAY
DQN
optimistic
$1/\sqrt{n}$
Score
Training frames (millions)

Idea : Counting with Hash Codes

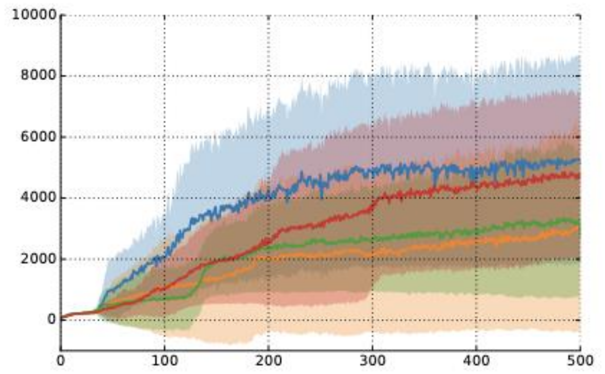
- Learn to convert continuous inputs into discretized hash codes.
- Count the occurrences of the code
- How do we get the image encoding? E.g. using Autoencoders
  - ➢ Do similar states get the same hash? Maybe…



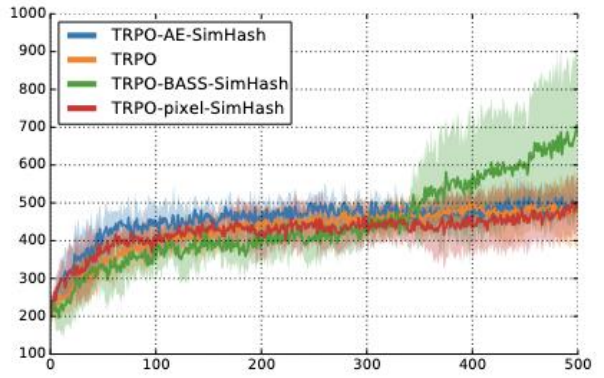
Does It Work?



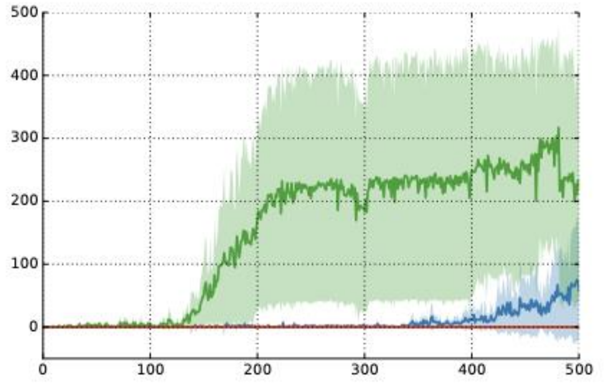
(a) Freeway



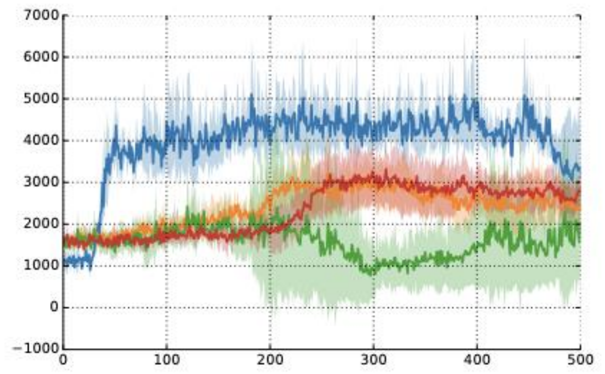
(b) Frostbite



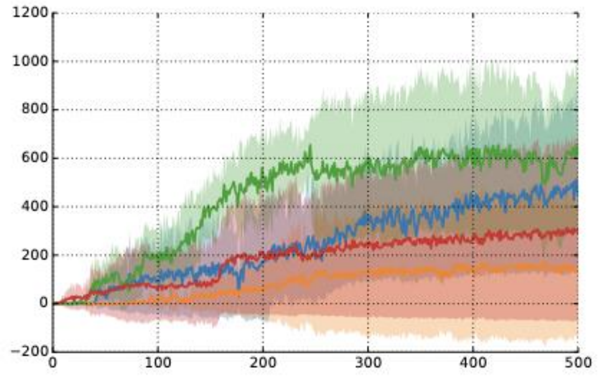
(c) Gravitar



(d) Montezuma's Revenge



(e) Solaris



(f) Venture

How to Model Exploration?

- We can re-formulate the objective of reinforcement learning:

$$R^t(s,a,s') = r\underbrace{(s,a,s')}_{\substack{\text{extrinsic reward} \\ \text{(related to the task)}}} + \underbrace{\mathcal{B}^t(s,a,}_{\substack{\text{intrinsic reward} \\ \text{(unrelated to the task)}}}s')$$

- Types of intrinsic rewards:
  1. Data-based exploration (Optimistic exploration)
  2. Knowledge-based exploration
  3. Information gain exploration
  4. Reachability exploration
  5. Posterior sampling for exploration

Knowledge-based Exploration

- Idea: Maximize the knowledge about the world. The motivation is also known as *curiosity.* Knowledge could be in the form of:
  - ➢ Surprise
  - ➢ Unpredictability
  - ➢ How much learned about the world
  - ➢ Uncertainty

Computational Curiosity

- "The direct goal of curiosity and boredom is to improve the **world model**. The indirect goal is to ease the learning of new goal-directed action sequences."
- "The same complex mechanism which is used for 'normal' goal-directed learning is used for implementing curiosity and boredom. There is no need for devising a separate system which aims at improving the world model."
- "Curiosity Unit": reward is a function of the mismatch between model's current predictions and actuality. There is positive reinforcement whenever the system fails to correctly predict the environment.
- "Thus the usual credit assignment process ... encourages certain past actions in order to repeat situations similar to the mismatch situation." (planning to make your (internal) world model to fail)

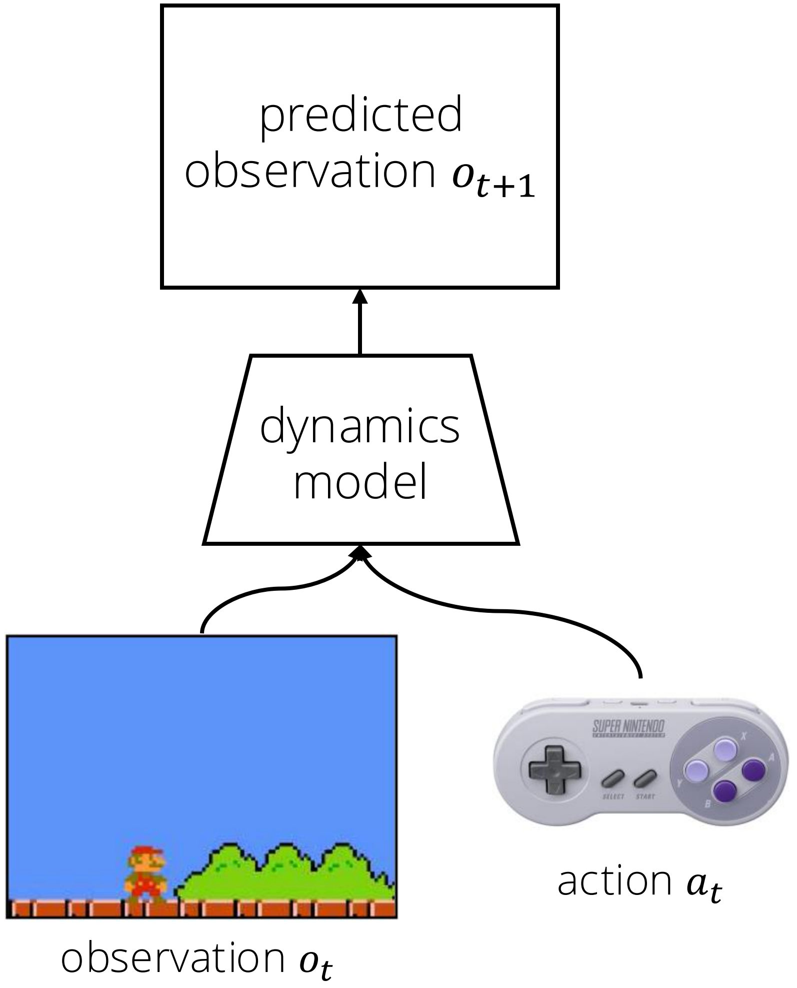
Jurgen Schmidhuber, 1991, 1991, 1997

Computational Curiosity

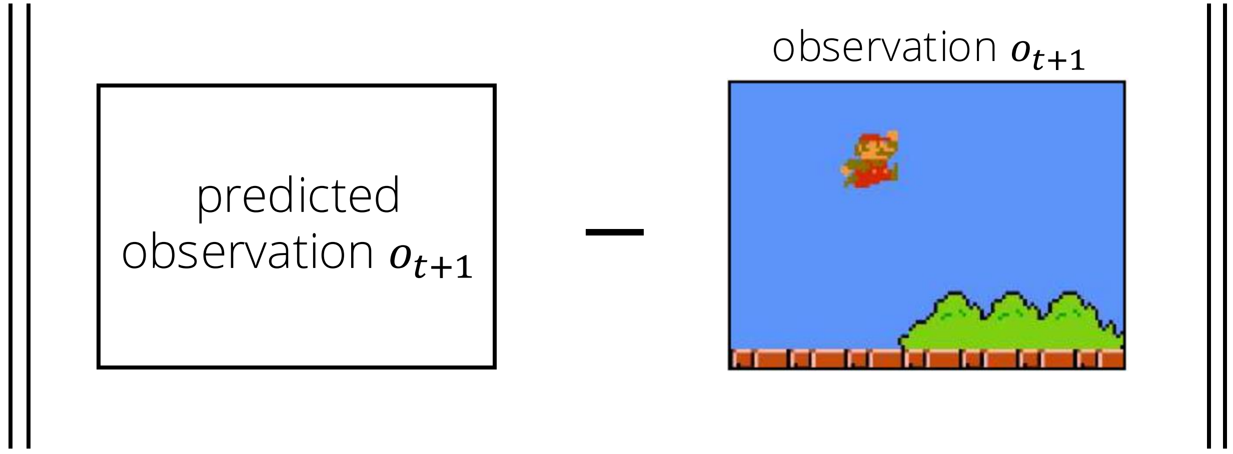
In other words:

- Model learning and model improvement can be cast as the goals of goal-seeking behaviour.
- My goal is not to beat Atari but to improve my Atari model.
- OK. What is my reward then that trying to maximize that reward will lead to fast model learning?

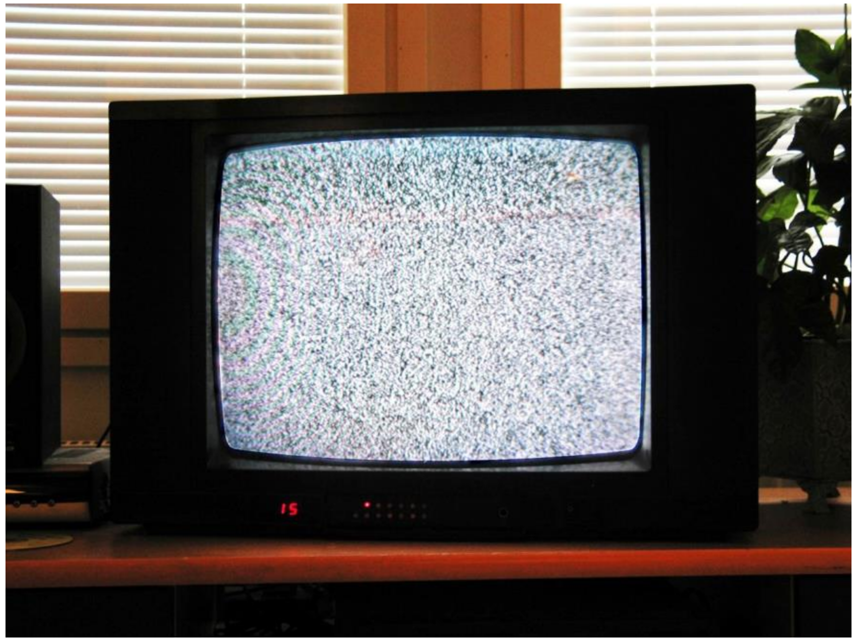
Curiosity Exploration



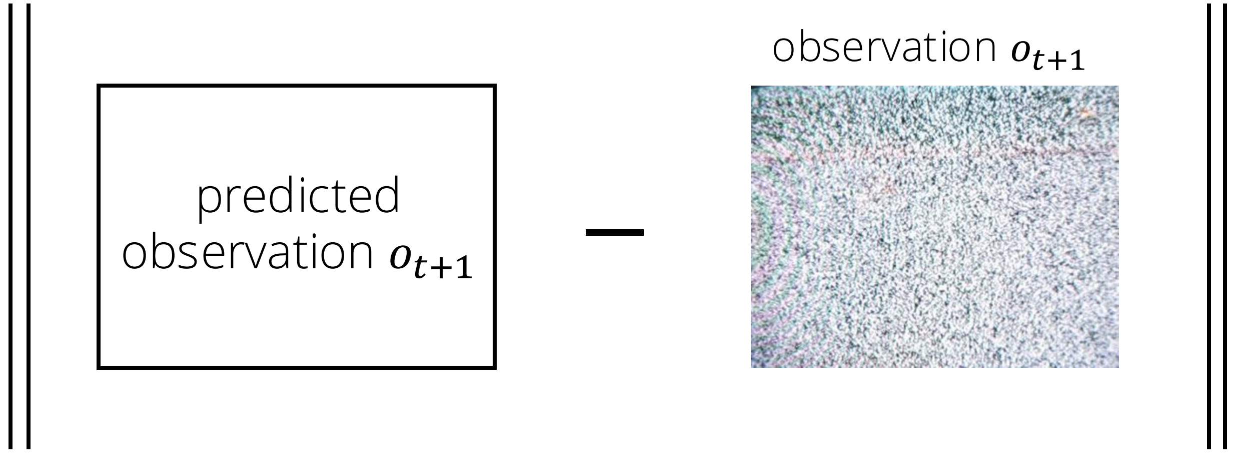
- Set the implicit reward $\mathcal{B}^t(s, a, s')$ as the prediction error of the dynamics model
- Action a is worth exploring if we are uncertain of its outcome



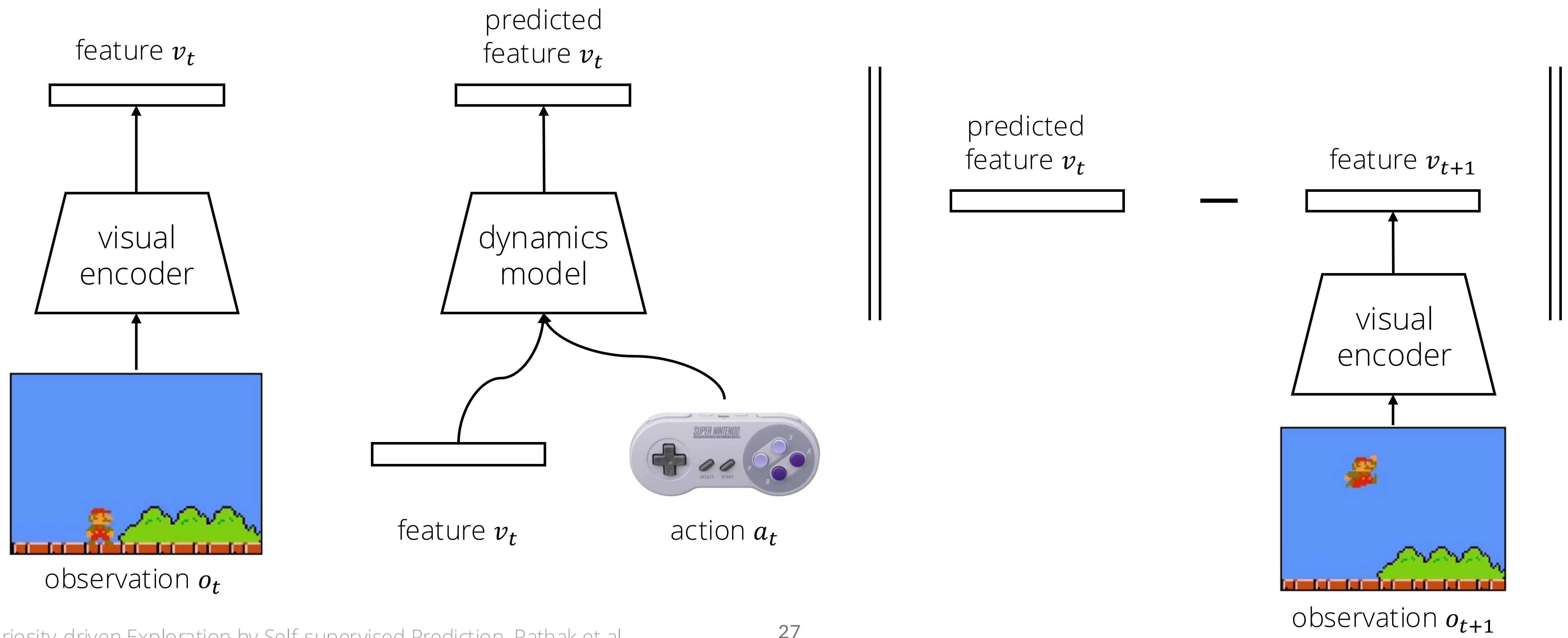
Is Pixel-based Prediction Error Desirable?



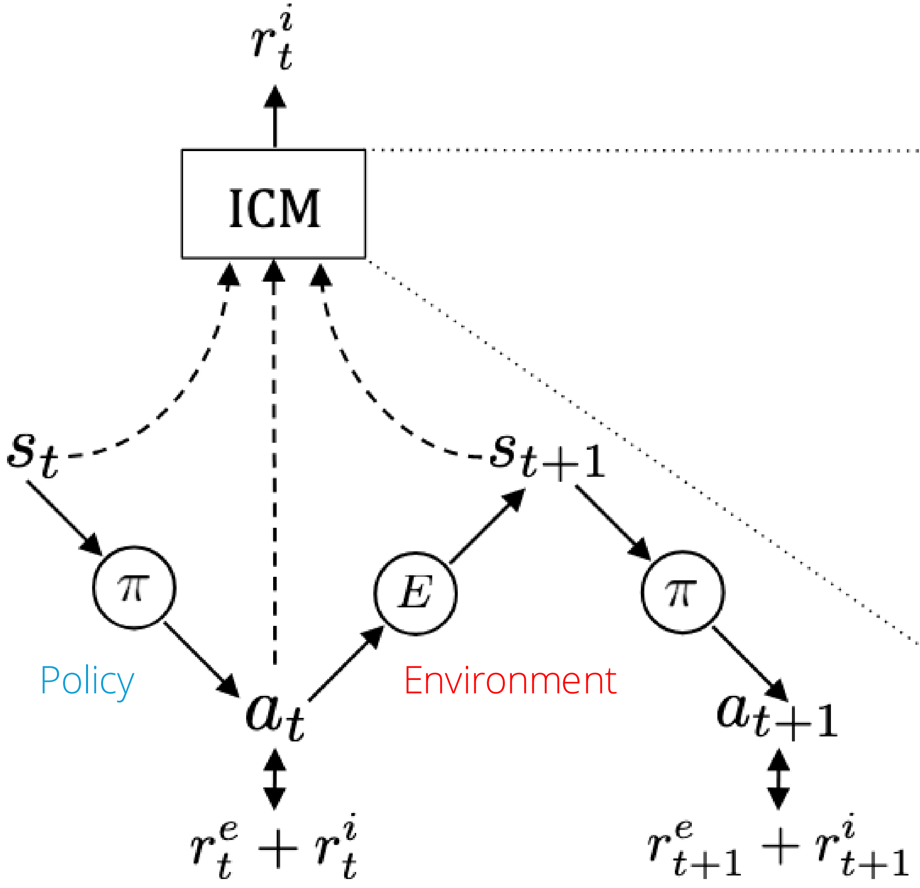
- Suppose the agent control to switch channels, and one channel if offline, showing white noise
- The prediction error on this channel will be very high
- The agent is encouraged to select this channel…

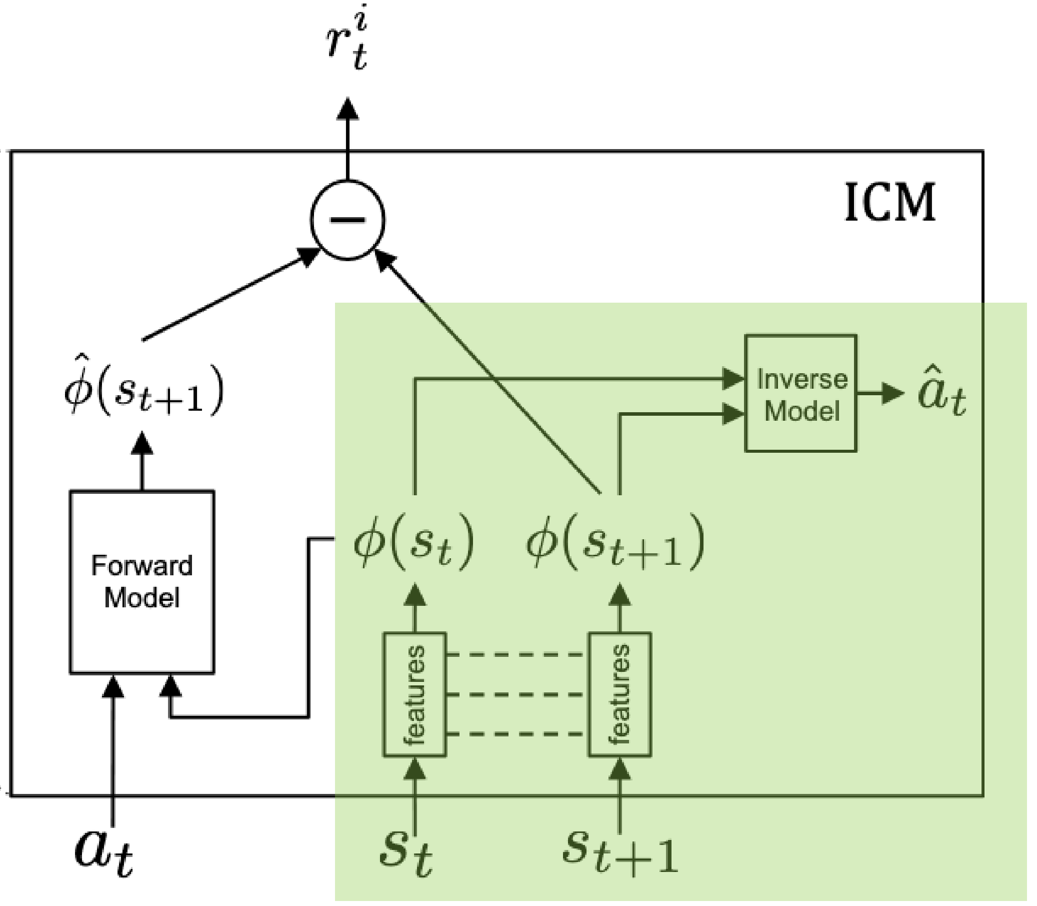


Idea: Curiosity as Prediction Errors in the Latent Feature Space



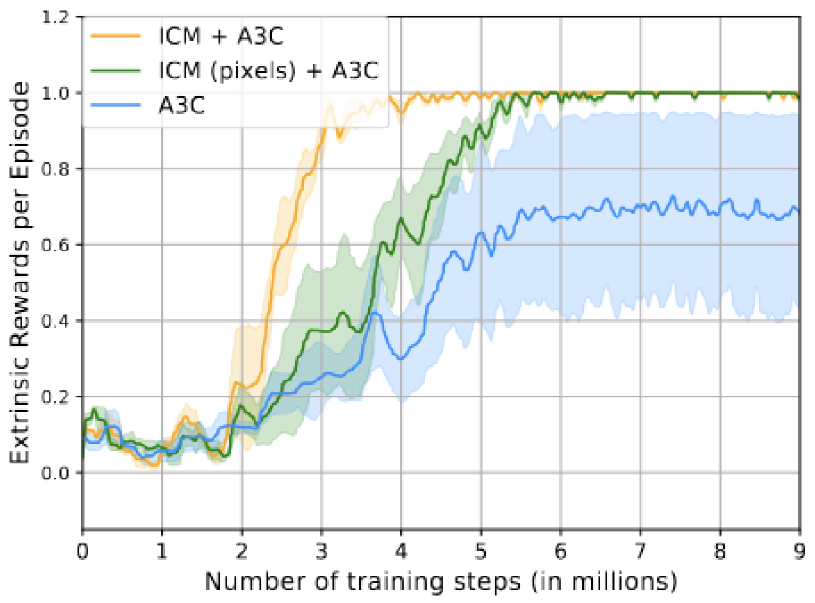
Regularize the Latent Feature Space: Only Encode Action-Related Information



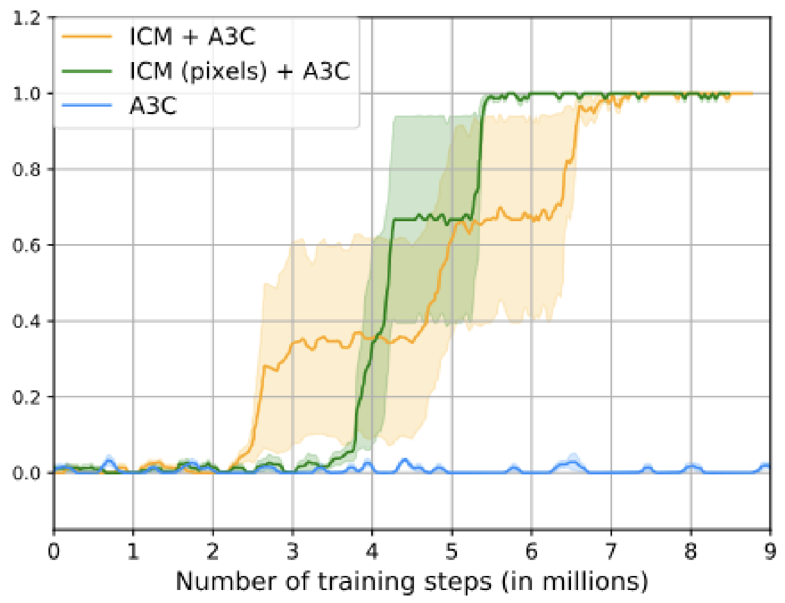


Pre-train visual encoder with inverse model

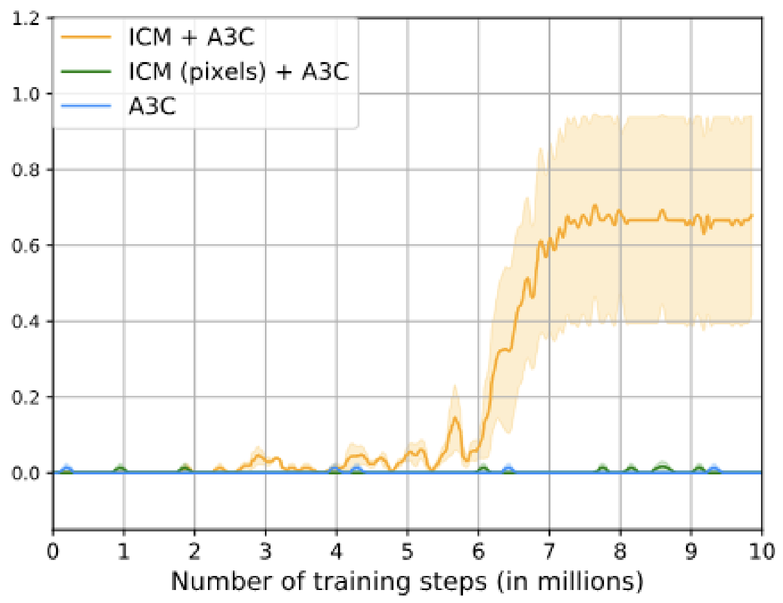
Does It Work?



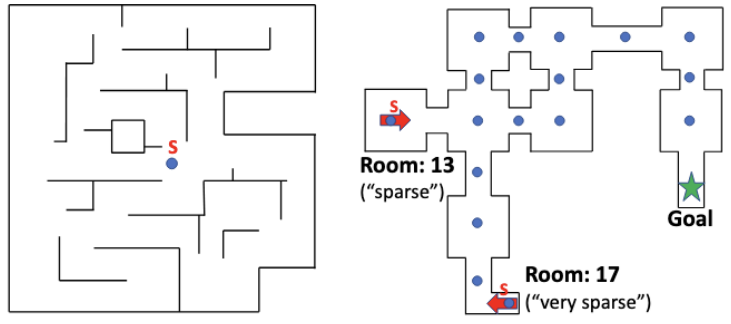
(a) “dense reward” setting



(b) “sparse reward” setting



(c) “very sparse reward” setting



(a) Train Map Scenario (b) Test Map Scenario

Figure 4. Maps for VizDoom 3-D environment: (a) For generalization experiments (c.f. Section 4.3), map of the environment where agent is pre-trained only using curiosity signal without any reward from environment. ‘S’ denotes the starting position. (b) Testing map for VizDoom experiments. Green star denotes goal location. Blue dots refer to 17 agent spawning locations in the map in the “dense” case. Rooms 13, 17 are the fixed start locations of agent in “sparse” and “very sparse” reward cases respectively. Note that textures are also different in train and test maps.

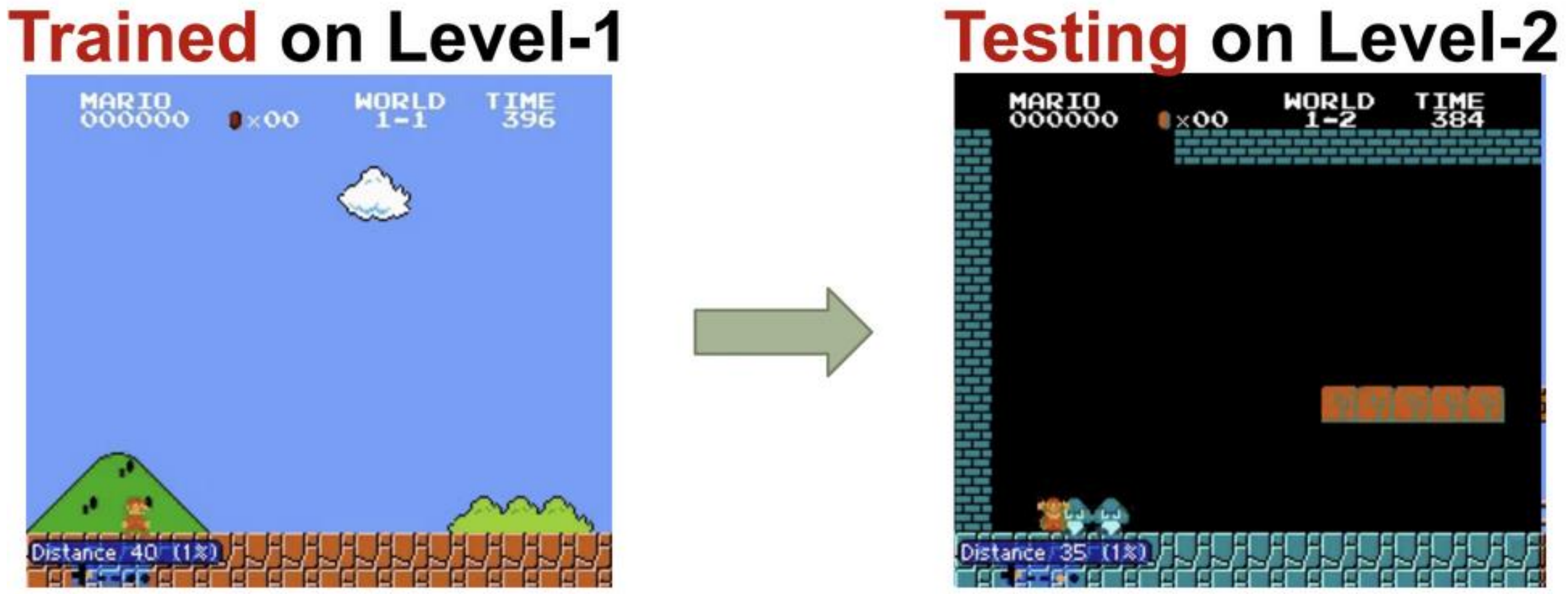
RL Objective:

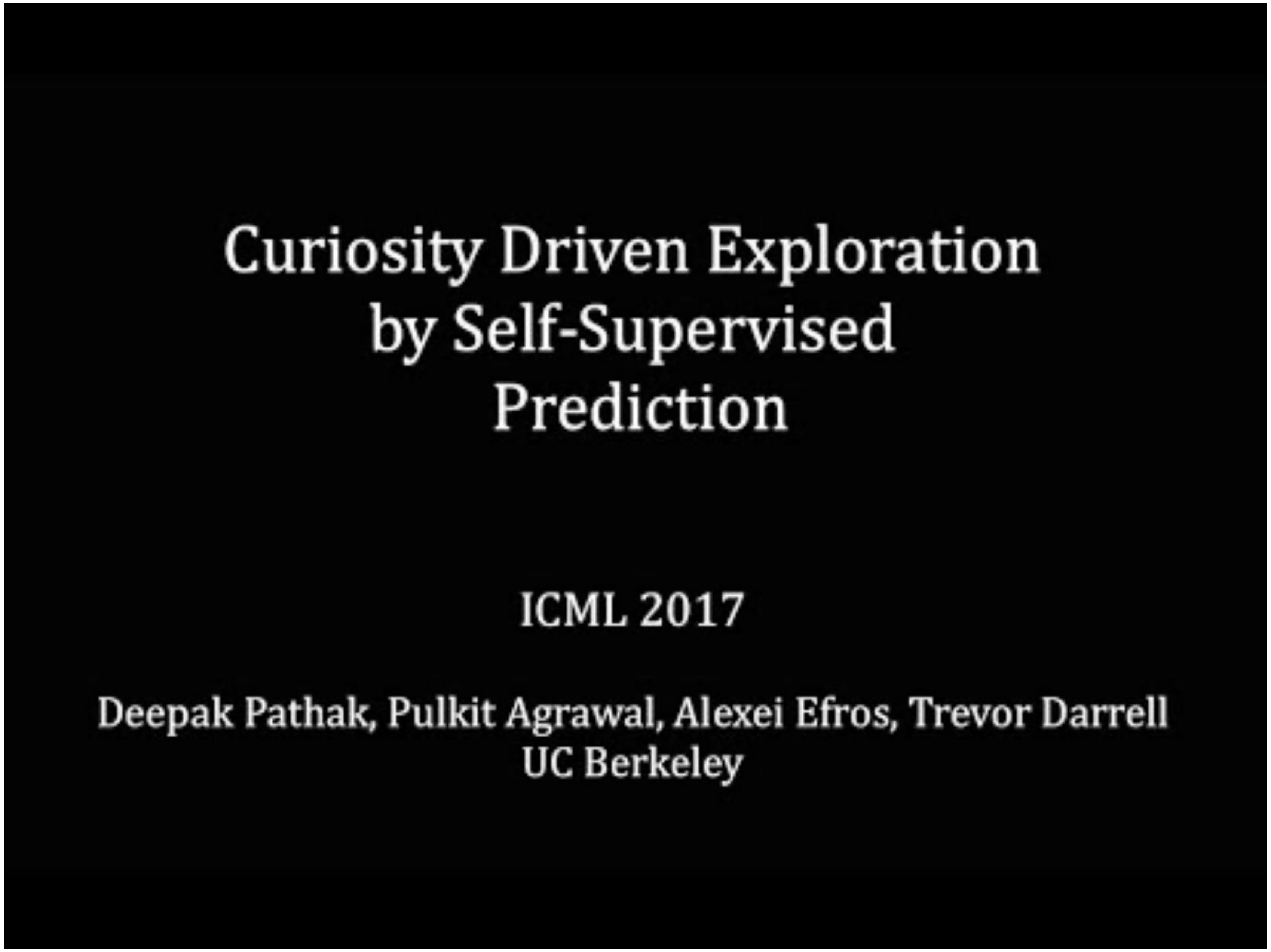
$$R^t(s,a,s') = r(s,a,s') + \mathcal{B}^t(s,a,s')$$

What if We Train the Policy with Only Intrinsic Reward?

- RL Objective:

$$R^t(s,a,s') = \cancel{r(s,a,s')} + \mathcal{B}^t(s,a,s')$$





Curiosity Driven Exploration by Self-Supervised Prediction

ICML 2017

Deepak Pathak, Pulkit Agrawal, Alexei Efros, Trevor Darrell
UC Berkeley

What Could be Wrong?

- The source of high predictions errors result from:
  1. **Amount of training data** (epistemic uncertainty): few similar examples were seen by the predictor
  2. **Stochasticity** (aleatoric uncertainty): stochastic transitions are a source of such error for forward dynamics prediction
  3. **Model misspecification**: necessary information is missing, or the model class is too limited to fit the complexity of the dynamic function
  4. **Learning dynamics**: the optimization process fails to find a predictor in the model class that best approximates the dynamic function

What Could be Wrong?

- The source of high predictions errors result from:

✓ 1. **Amount of training data** (epistemic uncertainty): few similar examples were seen by the predictor

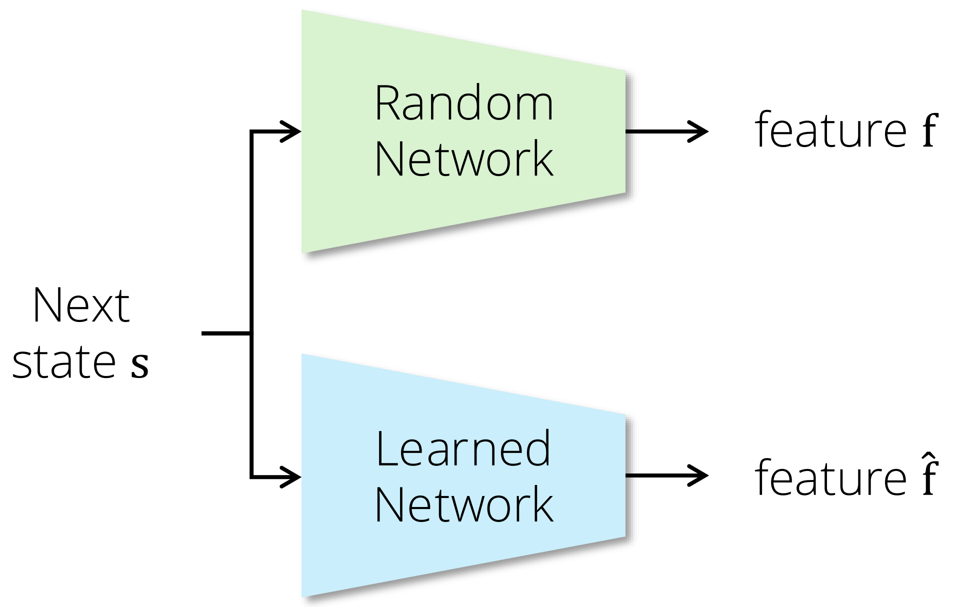
Can we fix them?

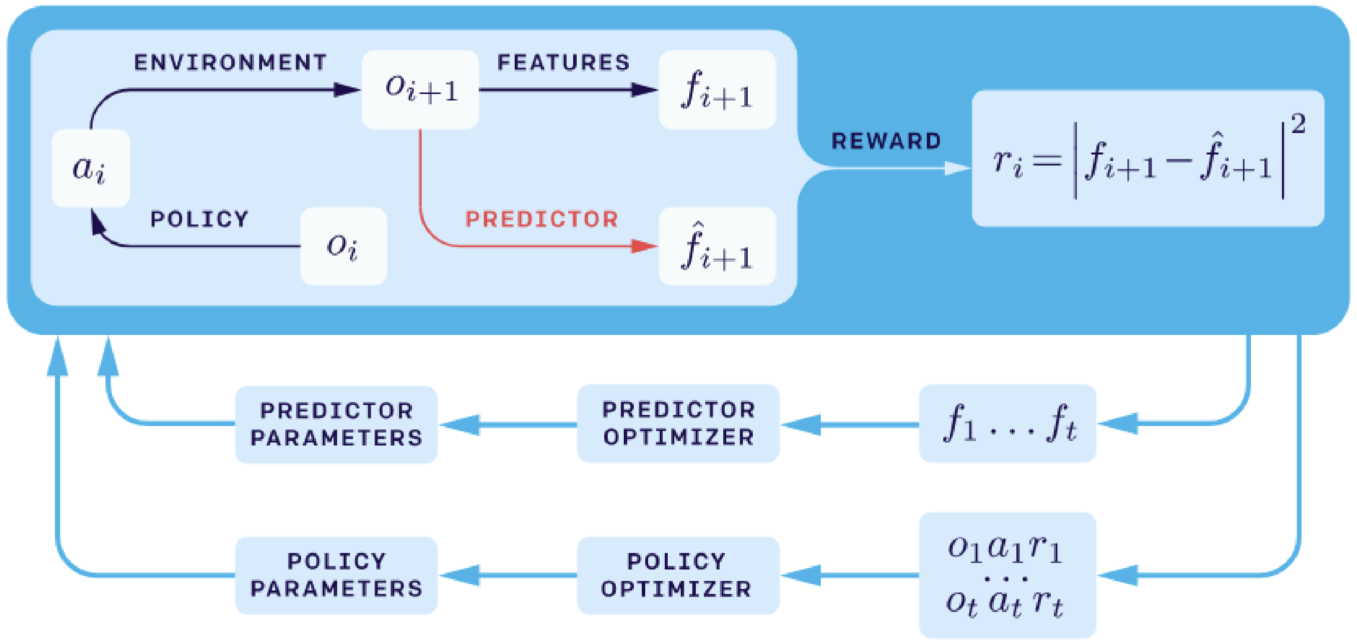
✗ 2. **Stochasticity** (aleatoric uncertainty): stochastic transitions are a source of such error for forward dynamics prediction

✗ 3. **Model misspecification**: necessary information is missing, or the model class is too limited to fit the complexity of the dynamic function

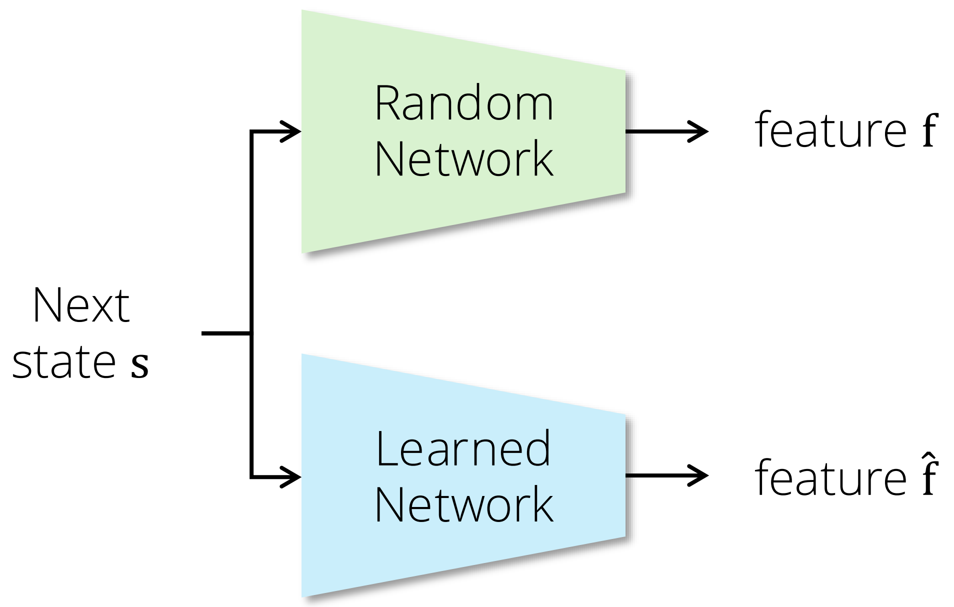
✗ 4. **Learning dynamics**: the optimization process fails to find a predictor in the model class that best approximates the dynamic function

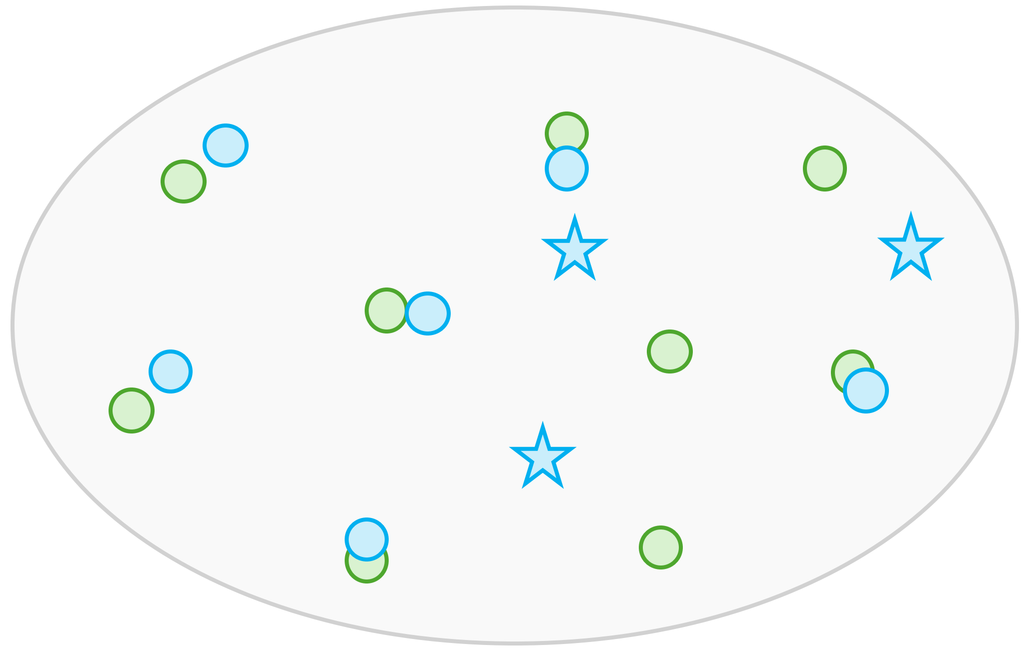
Exploration By Random Network Distillation





Novel States Result in Larger Feature Distance





○ Feature of seen state

☆ Feature of novel state

What Could be Wrong?

Solution: (1) Bypass learning forward dynamics model but deterministically learn to encode future states. (2) Smoothness prior in neural network enhance the robustness of novel-state estimation

✗ 2. **Stochasticity** (aleatoric uncertainty): stochastic transitions are a source of such error for forward dynamics prediction

✗ 3. **Model misspecification**: necessary information is missing, or the model class is too limited to fit the complexity of the dynamic function

Solution: Build the target network and the predictor with similar model class

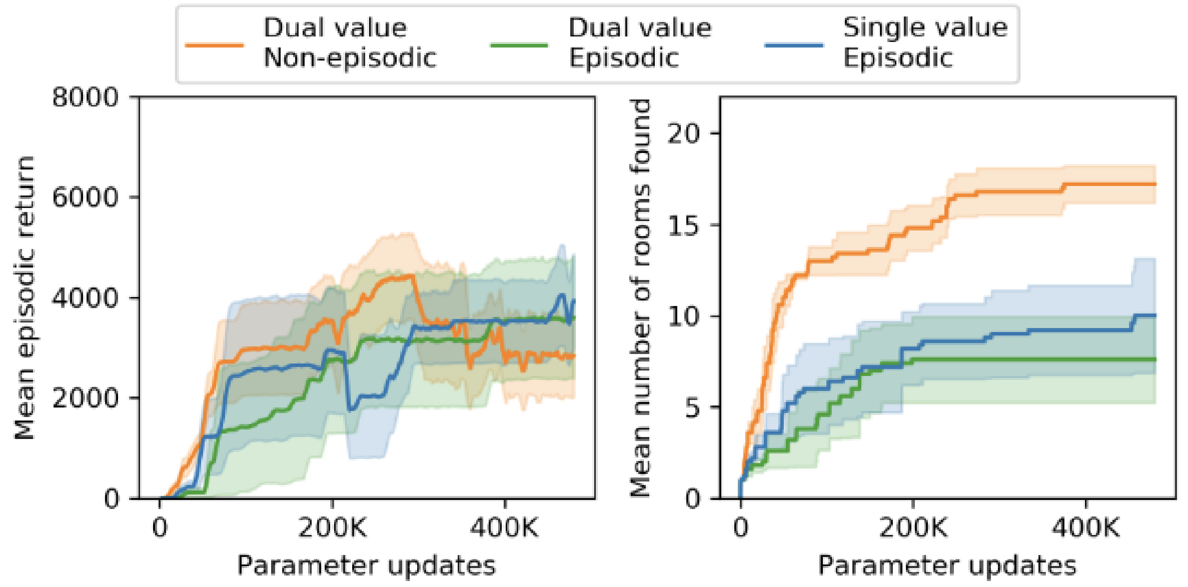
Representation Learning is More Suitable for Non-episodic Learning

- Under episodic setting $\sum_{t=0}^{T-1} \gamma^t R_t$:
  - The return is truncated at "game over".
  - The environment is reset to a fixed initial scene at "game over"
  - Agents prefer risk-free behavior to avoid game over
- Let $\mathbf{r}_{\mathbf{intrinsic}} = \|\mathrm{f} - \hat{\mathrm{f}}\|^2$. This reward decreases for seen states
- If we train the agent under non-episodic return stream $\lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} R_t$:
  - The agent will avoid "game over" behavior, since the reward of initial states become small
  - Episodic intrinsic rewards can leak information about the task to the agent (Burda et al 2018)

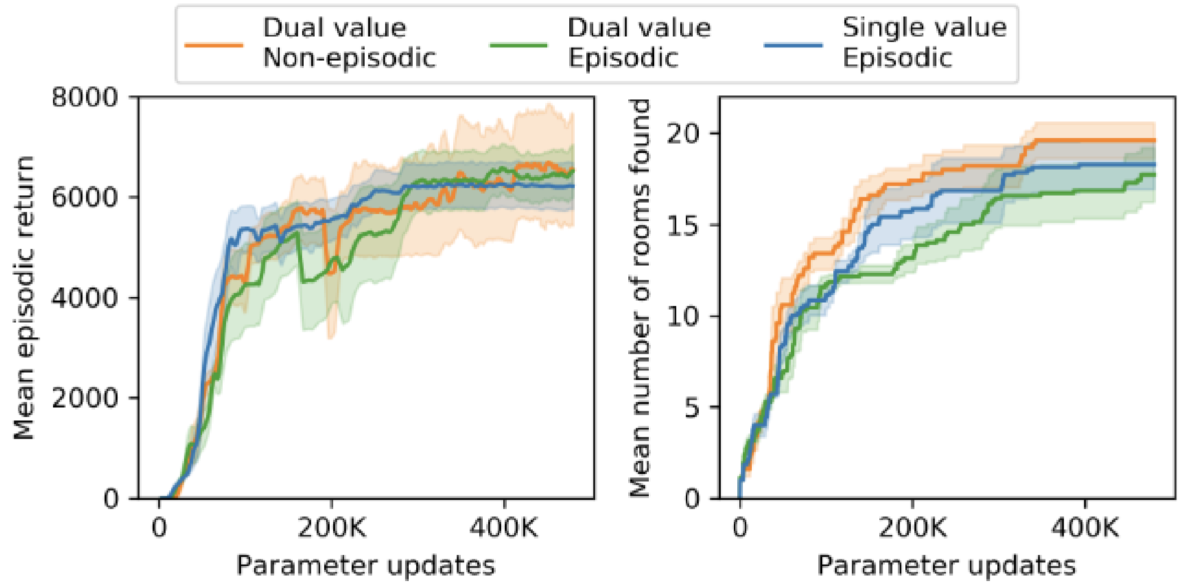
Details Matter: Separate Return Streams

- For extrinsic rewards: episodic return streams and an independent value head $\mathrm{V_E}$
- For extrinsic rewards: non-episodic return streams and an independent value head $\mathrm{V_I}$
- Train the agent with PPO using $\mathrm{V} = \mathrm{V_E} + \mathrm{V_I}$

Non-episodic Intrinsic Return Streams Matter

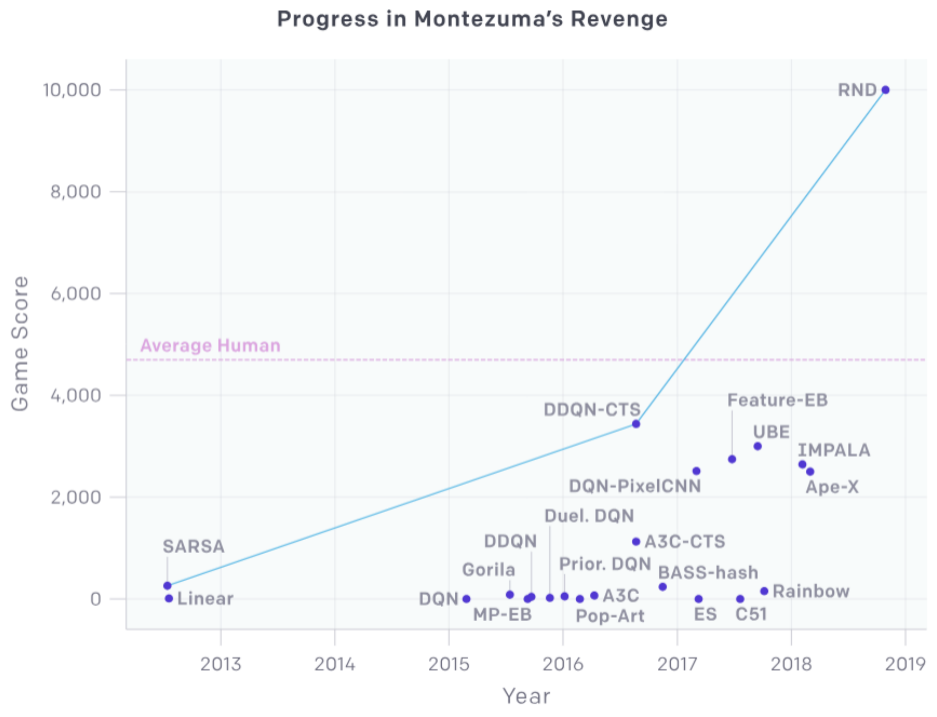


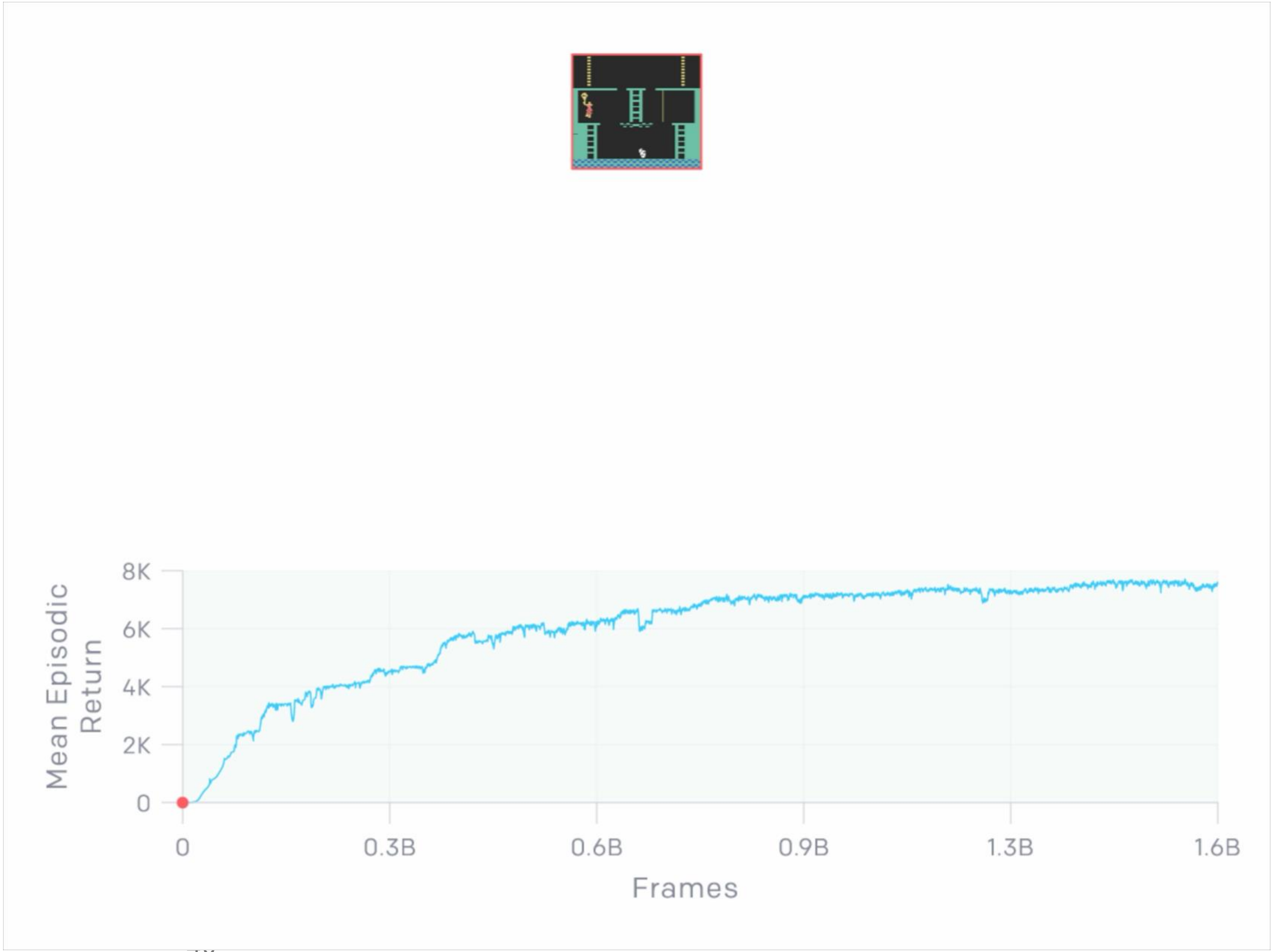
(a) RNN policies



(b) CNN policies

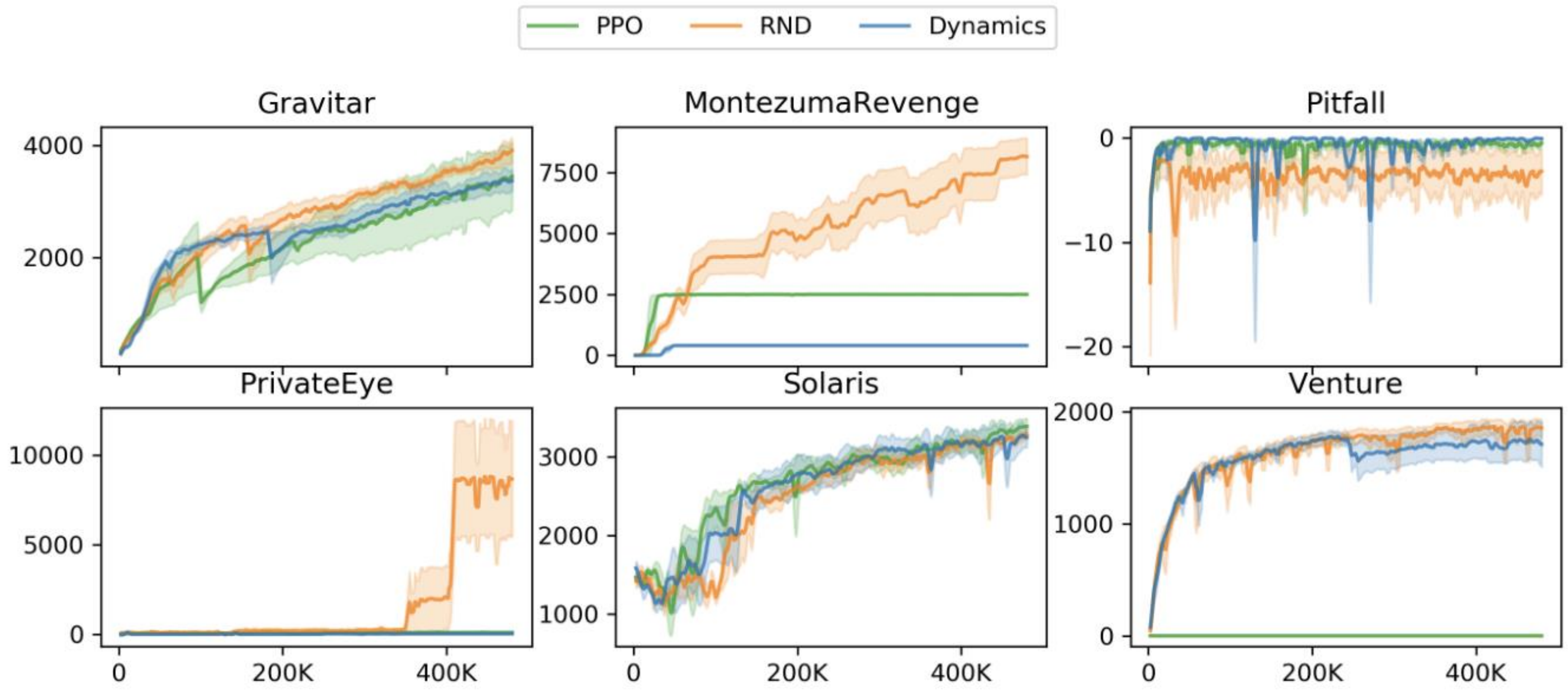
State-of-the-Art in Montezuma's Revenge





Exploration by Random Network Distillation. Burda et al.

Outperforms PPO



How to Model Exploration?

- We can re-formulate the objective of reinforcement learning:

$$R^t(s,a,s') = r\underbrace{(s,a,s')}_{\substack{\text{extrinsic reward} \\ \text{(related to the task)}}} + \underbrace{\mathcal{B}^t(s,a,}_{\substack{\text{intrinsic reward} \\ \text{(unrelated to the task)}}}s')$$

- Types of intrinsic rewards:
  1. Data-based exploration (Optimistic exploration)
  2. Knowledge-based exploration
  3. Information gain exploration
  4. Reachability exploration
  5. Posterior sampling for exploration

Empowerment and Mutual Information

- The concept of *empowerment*: an attempt at formalizing and quantifying the degrees of freedom (or options) that an organism or agent has as a proxy for "*preparedness*"

- *Preparedness* can be seen as:
  - ➢ Keeping options open
  - ➢ Influence over the environment
  - ➢ Resilience to uncertainty

- Empowerment through Mutual Information Maximization: Find policy $\pi(a|y,x)$ that maximizes mutual information $\mathcal{I}(x;y)$

- Mutual information:

$$\mathcal{I}(x;y) = \mathcal{H}(x) - H(x|y)$$

where $\mathcal{H}$ denotes entropy (the higher entropy the higher uncertainty is)

Empowerment and Mutual Information

- The concept of *empowerment*: an attempt at formalizing and quantifying the degrees of freedom (or options) that an organism or agent has as a proxy for "*preparedness*"

- *Preparedness* can be seen as:
  - Keeping options open
  - Influence over the environment
  - Resilience to uncertainty

- Empowerment through Mutual Information Maximization: Find policy $\pi(a|y, x)$ that maximizes mutual information $\mathcal{I}(x; y)$

- Mutual information:

How much information you know about $\boldsymbol{x}$ when you perform action $\boldsymbol{y}$

$$\boxed{\mathcal{I}(x; y)} = \mathcal{H}(x) - H(x|y)$$

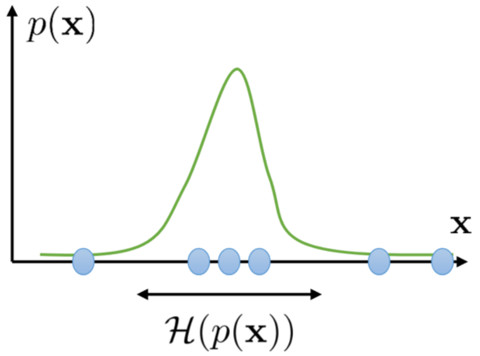
where $\mathcal{H}$ denotes entropy (the higher entropy the higher uncertainty is)

Some Useful Identities

$p(\mathbf{x})$ distribution (e.g., over observations $\mathbf{x}$)

$$\mathcal{H}(p(\mathbf{x})) = -E_{\mathbf{x}\sim p(\mathbf{x})}[\log p(\mathbf{x})]$$

entropy – how "broad" $p(\mathbf{x})$ is

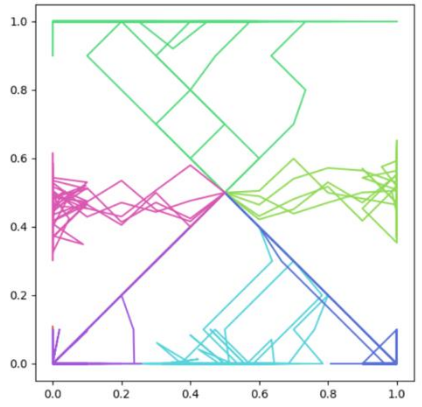


$$\mathcal{I}(\mathbf{x};\mathbf{y}) = D_{\mathrm{KL}}(p(\mathbf{x},\mathbf{y}) \| p(\mathbf{x})p(\mathbf{y}))$$

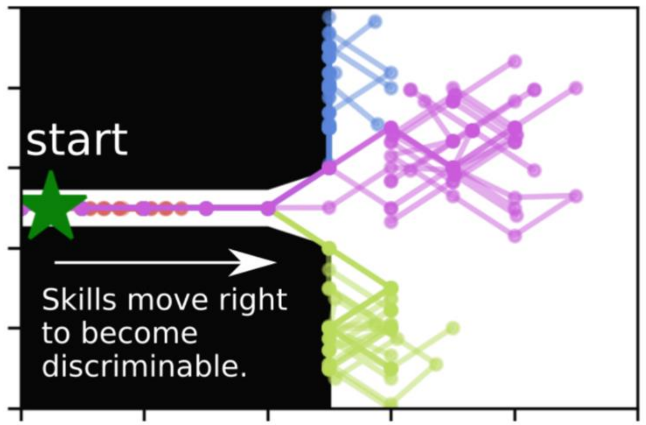
$$= E_{(\mathbf{x},\mathbf{y})\sim p(\mathbf{x},\mathbf{y})}\left[\log \frac{p(\mathbf{x},\mathbf{y})}{p(\mathbf{x})p(\mathbf{y})}\right]$$

$$= \mathcal{H}(p(\mathbf{y})) - \mathcal{H}(p(\mathbf{y}|\mathbf{x}))$$

Skill-based Exploration: Diversity is All You Need (DAIYN)



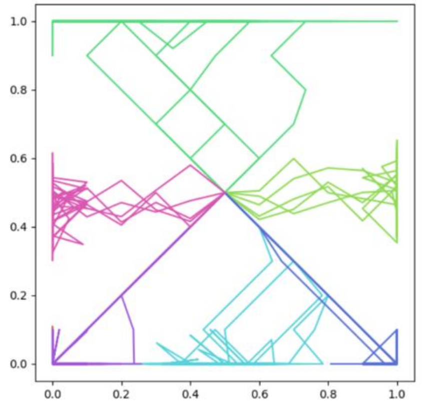
(a) 2D Navigation



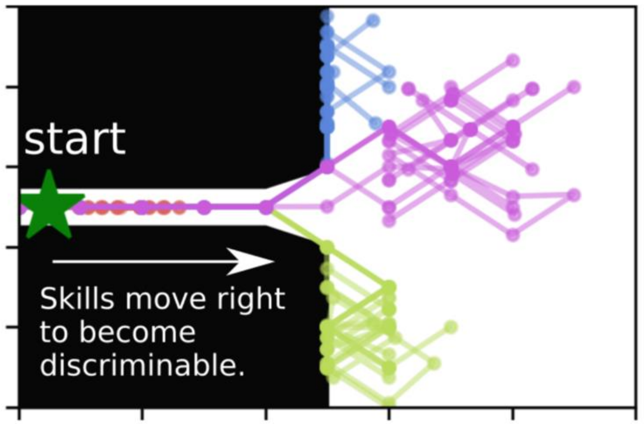
(b) Overlapping Skills

- This paper defines *skill* as the states that the agent visits
- The core idea is to find *skill* that induces actions to visit different states. These states should be distinguishable
- How to formulate the problem?

Skill-based Exploration: Diversity is All You Need (DAIYN)



(a) 2D Navigation



(b) Overlapping Skills

- Let $q_\phi(z|s)$ be the discriminator probability of skill z given state s parameterized by ϕ.
- Intrinsic reward as the discriminability of skill z given state s:

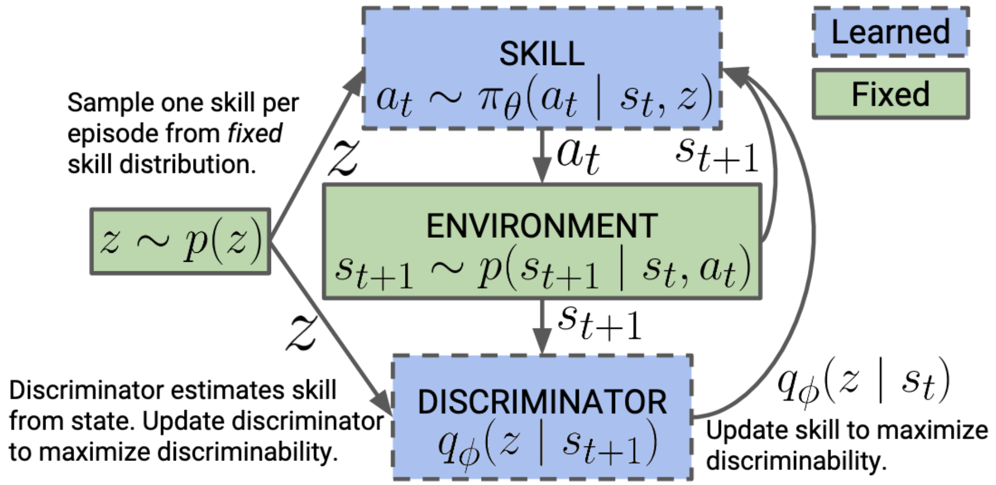
$$r_z(s,a) := \log q_\phi(z|s) - \log p(z)$$

- The objective of the policy:

$$\pi_\theta(a|s,z) = \underset{\pi}{\operatorname{argmax}} \sum_z \mathbb{E}_{s\sim\pi(s|z)}[r(s,z)]$$

reward states that are unlikely for other $z' \neq z$

Skill-based Exploration: Diversity is All You Need (DAIYN)

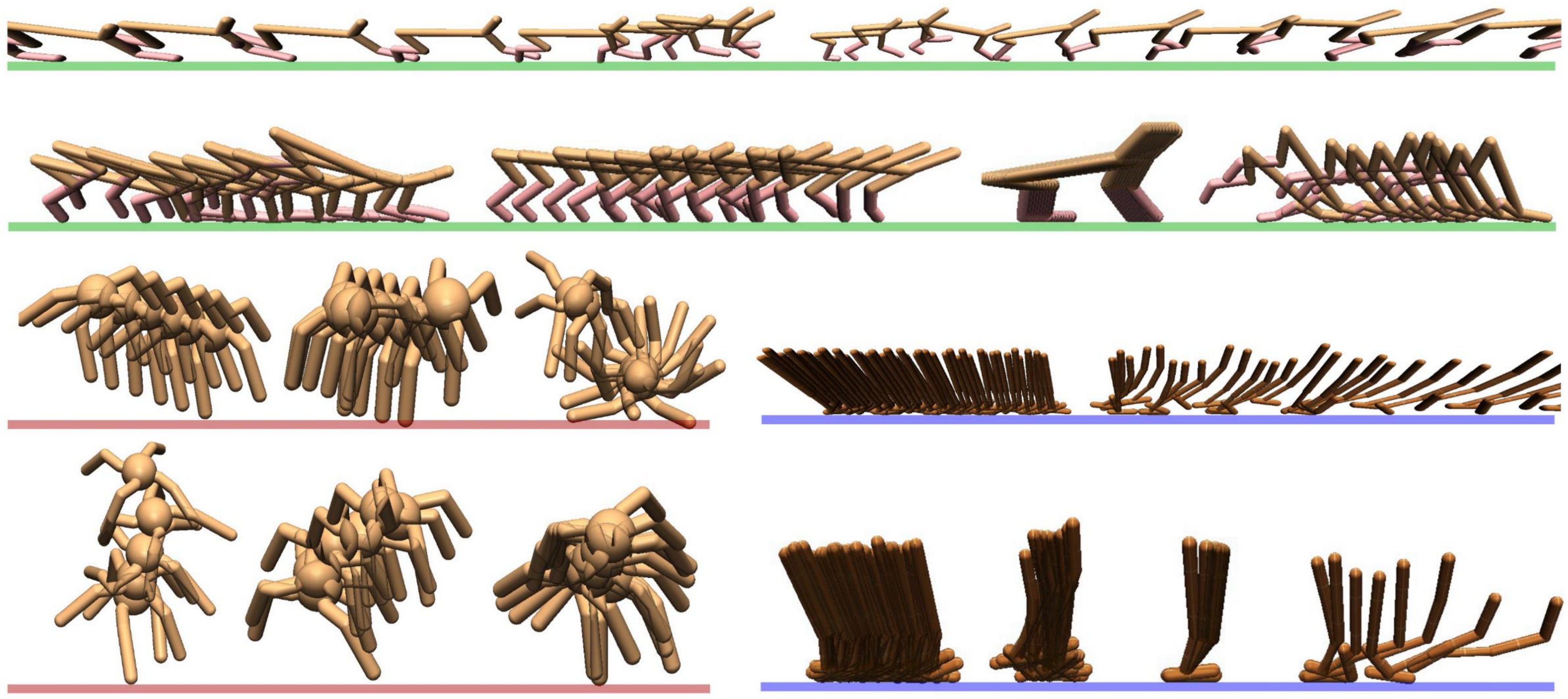

SKILL
$a_t \sim \pi_\theta(a_t \mid s_t, z)$
Learned
Fixed
Sample one skill per episode from *fixed* skill distribution.
$z \sim p(z)$
z
a_t
s_{t+1}
ENVIRONMENT
$s_{t+1} \sim p(s_{t+1} \mid s_t, a_t)$
s_{t+1}
z
Discriminator estimates skill from state. Update discriminator to maximize discriminability.
DISCRIMINATOR
$q_\phi(z \mid s_{t+1})$
$q_\phi(z \mid s_t)$
Update skill to maximize discriminability.

$p(z)$ is fixed so that model won't collapse

Algorithm 1: DIAYN

```
while not converged do
    Sample skill $z \sim \boxed{p(z)}$ and initial state $s_0 \sim p_0(s)$
    for $t \leftarrow 1$ to steps_per_episode do
        Sample action $a_t \sim \pi_\theta(a_t \mid s_t, z)$ from skill.
        Step environment: $s_{t+1} \sim p(s_{t+1} \mid s_t, a_t)$.
        Compute $q_\phi(z \mid s_{t+1})$ with discriminator.
        Set skill reward $r_t = \log q_\phi(z \mid s_{t+1}) - \log p(z)$
        Update policy ($\theta$) to maximize $r_t$ with SAC.
        Update discriminator ($\phi$) with SGD.
```

Example of Learned Skills



The Connection to Mutual Information

$$\pi(\mathbf{a}|\mathbf{s}, z) = \arg\max_{\pi} \sum_{z} E_{\mathbf{s}\sim\pi(\mathbf{s}|z)}[r(\mathbf{s}, z)]$$

$$r(\mathbf{s}, z) = \log p(z|\mathbf{s})$$

$$I(z, \mathbf{s}) = \underbrace{H(z)}_{\text{maximized by using uniform prior } p(z)} - \underbrace{H(z|s)}_{\text{minimized by maximizing } \log p(z|\mathbf{s})}$$

How to Model Exploration?

- We can re-formulate the objective of reinforcement learning:

$$R^t(s,a,s') = r\underbrace{(s,a,s')}_{\substack{\text{extrinsic reward} \\ \text{(related to the task)}}} + \underbrace{\mathcal{B}^t(s,a,}_{\substack{\text{intrinsic reward} \\ \text{(unrelated to the task)}}}s')$$

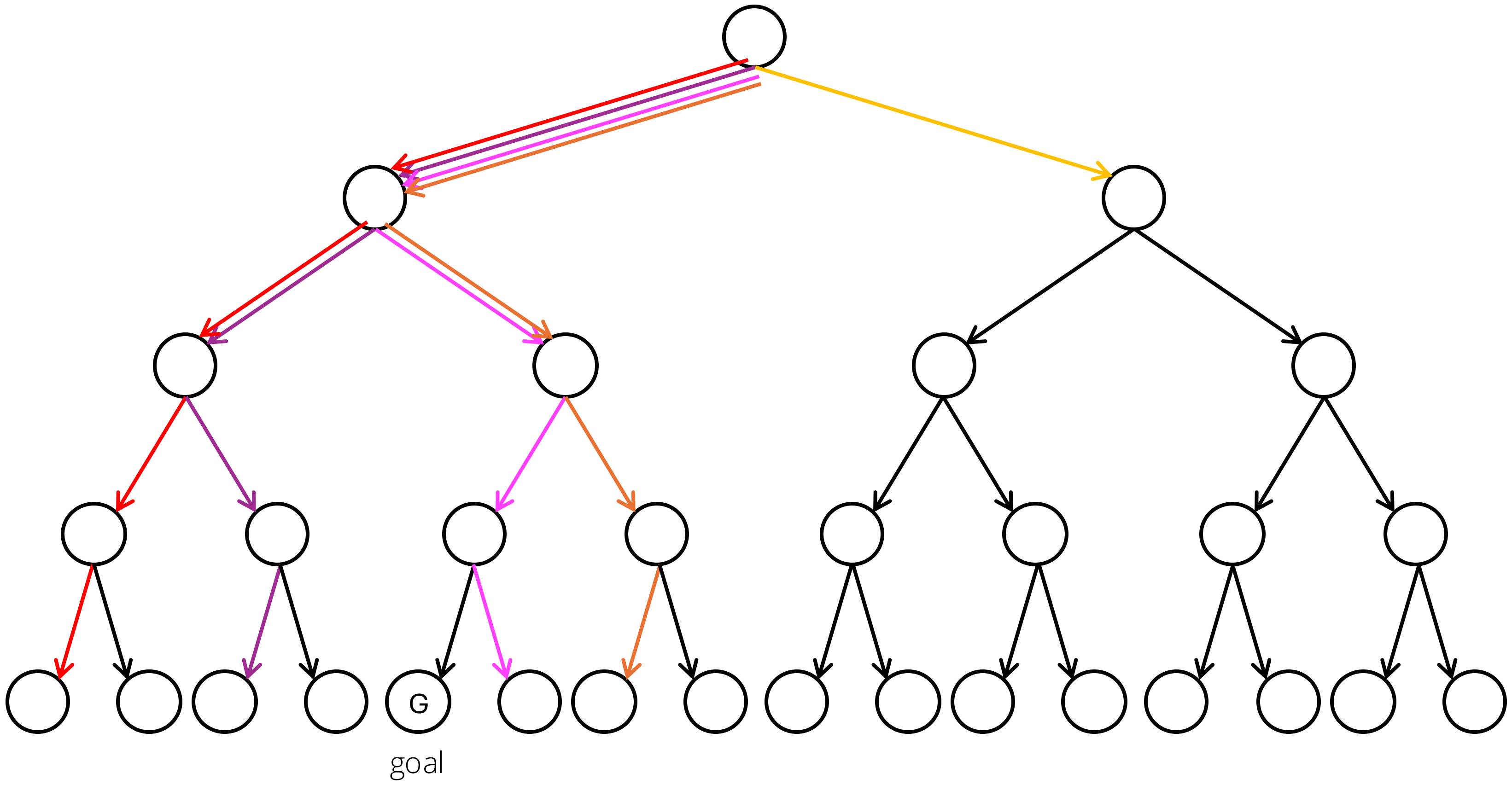
- Types of intrinsic rewards:
  1. Data-based exploration (Optimistic exploration)
  2. Knowledge-based exploration
  3. Information gain exploration
  4. Reachability exploration
  5. Posterior sampling for exploration

Exploration is Hard…

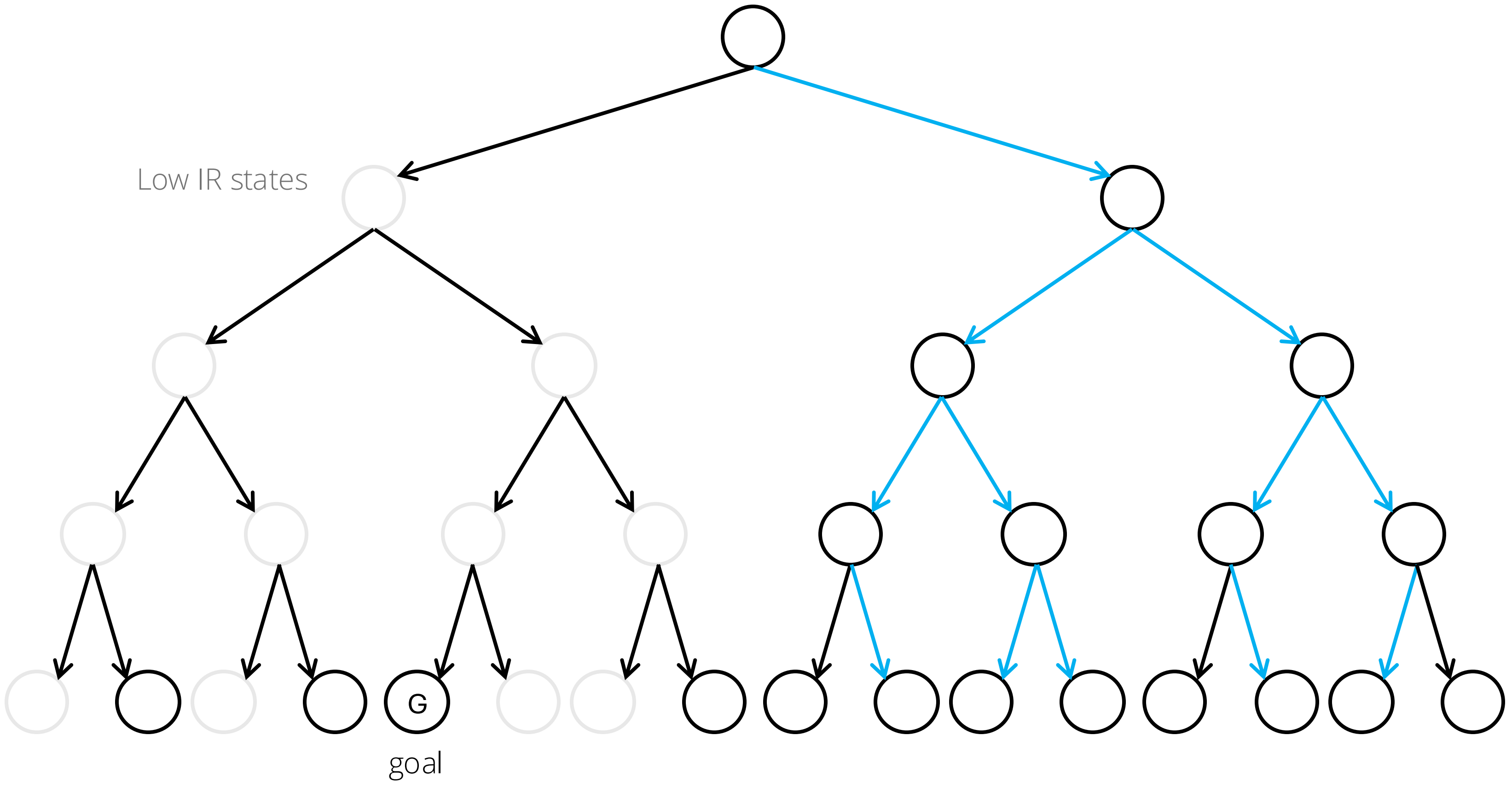
- Hard exploration problem:
  - ➢ When the reward is sparse, precise sequences of many (e.g. hundreds or more) actions must be taken between obtaining rewards
  - ➢ When the reward is deceptive, it actually provides misleading feedback for reaching the overall global objective, which can lead to getting stuck on local optima

- Exploration methods based intrinsic motivations have two problems:
  - ➢ <u>Detachment</u>: an agent become detached from the frontiers of high intrinsic reward
  - ➢ <u>Derailment</u>: an agent derail from the discovered high-reward state

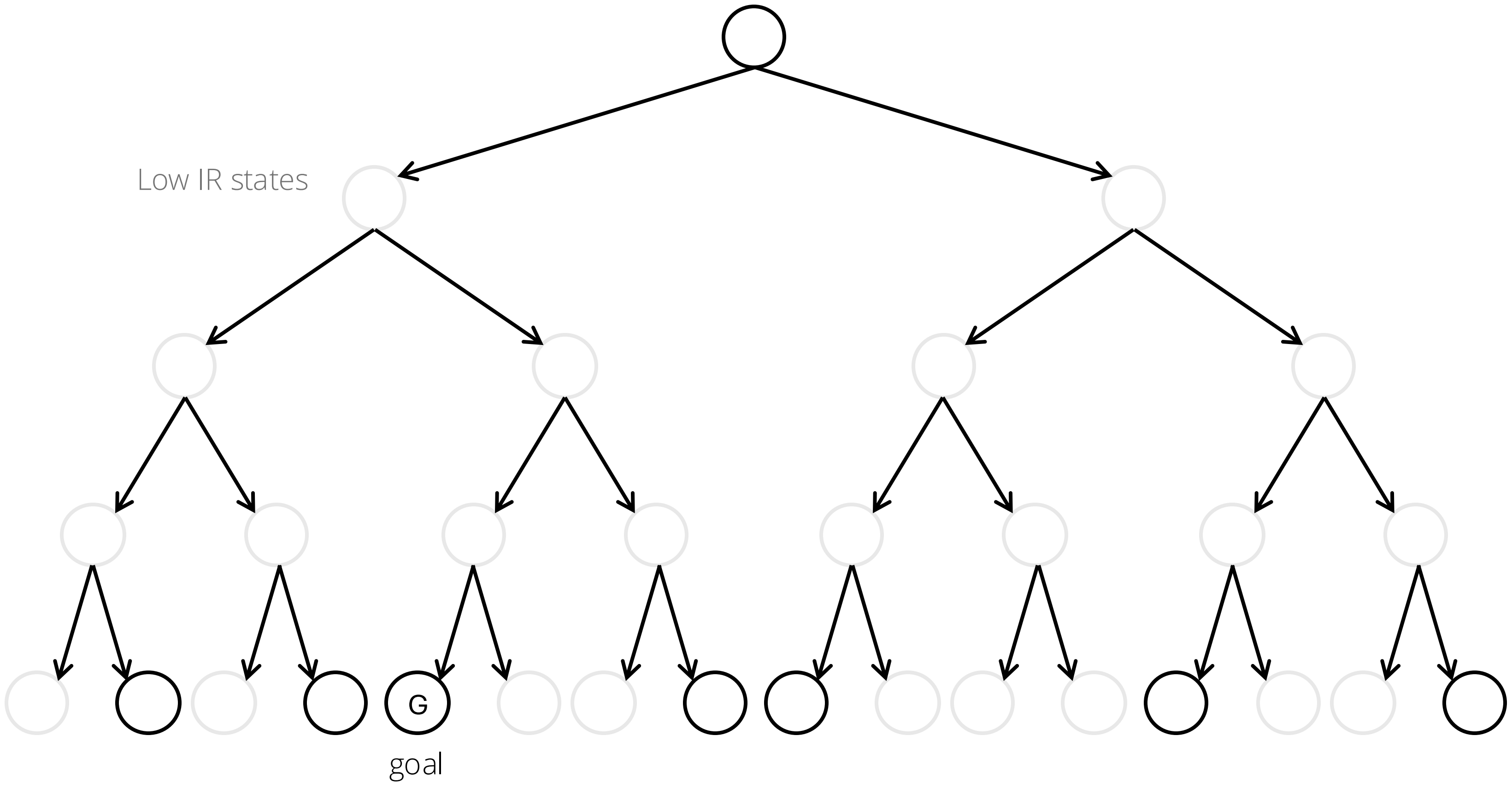
Previous Exploration Strategy



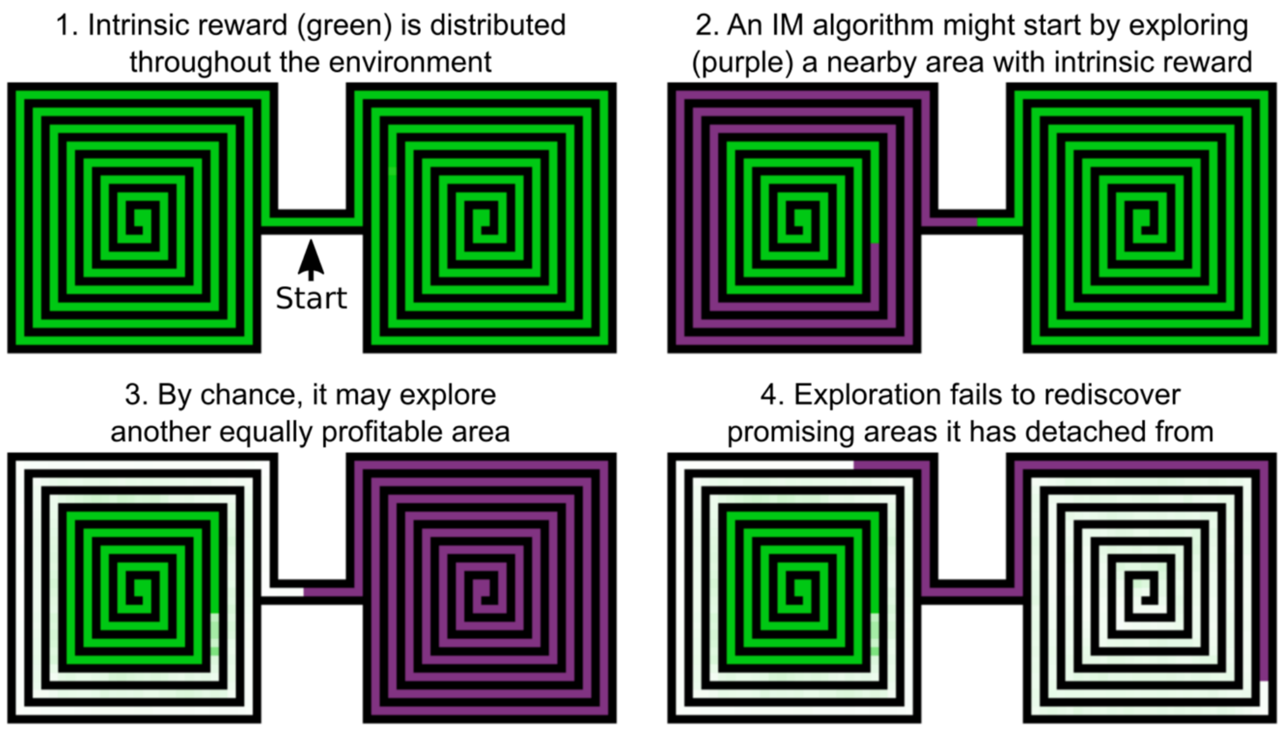
Previous Exploration Strategy



Previous Exploration Strategy: Little Incentive to Explore Visited States



Detachment

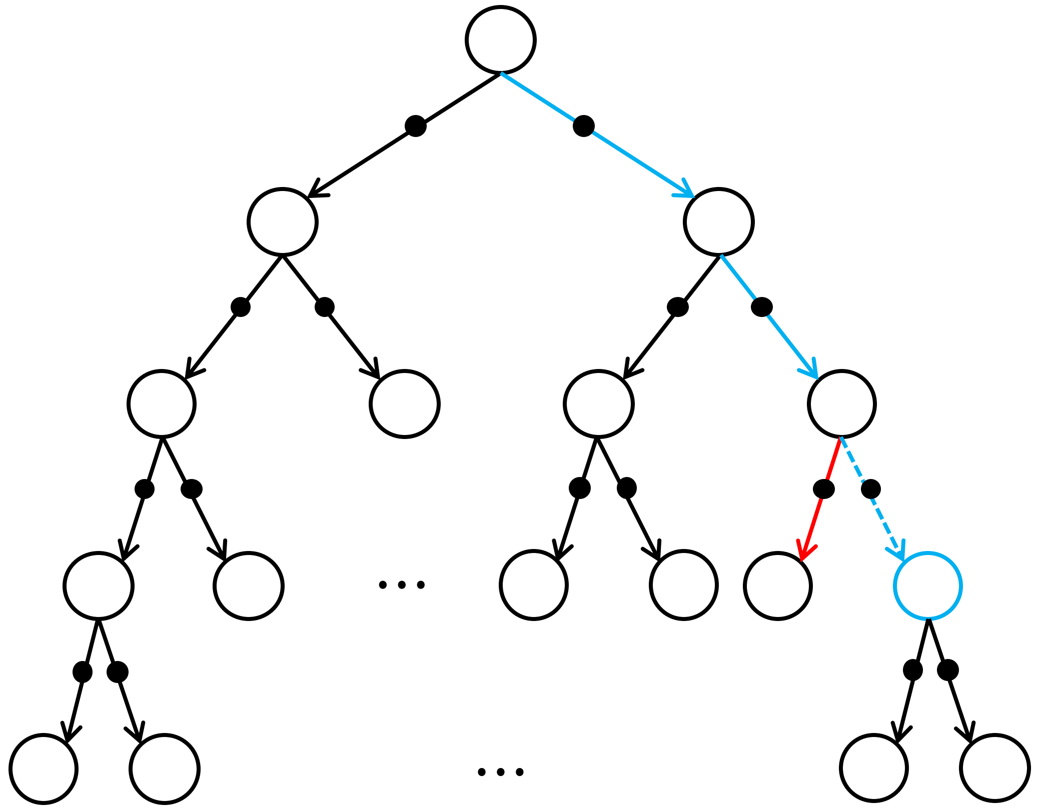

1. Intrinsic reward (green) is distributed throughout the environment
Start
2. An IM algorithm might start by exploring (purple) a nearby area with intrinsic reward
3. By chance, it may explore another equally profitable area
4. Exploration fails to rediscover promising areas it has detached from

- IM-based exploration occurs as follows:

1. An agent is encouraged by discover regions with high intrinsic rewards (IR)
2. After exhausting IRs, the agent may by chance start consuming IR in another regions
3. Even if the IR in 2. is consumed, it's still hard to rediscover region of 1., where IR is also exhausted

- This phenomena is called *detachment*
- Large replay buffer may solve this issue, but why not explicitly store these previously discovered states?

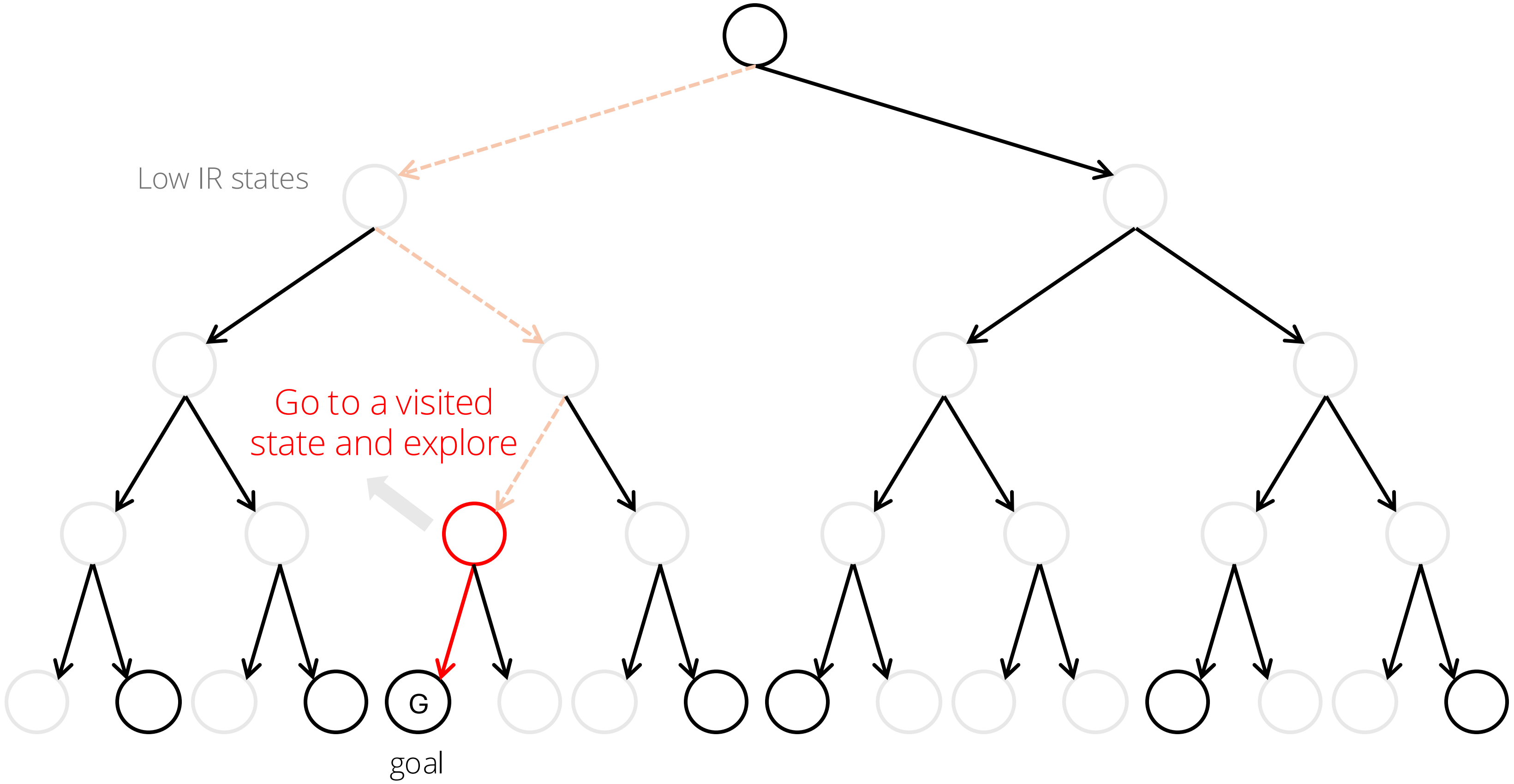
Derailment



- An agent has two types of exploration mechanism:
    1. high-level exploration: IR incentive that rewards when new states are reached
    2. low-level exploration: epsilon-greedy exploration, action-space noise, or parameter-space noise

- The longer, more complex, and more precise a sequence of actions needs to be in order to reach a previously-discovered high-IR state, the more likely it is that such stochastic perturbations will “*derail*” the agent from ever returning to that state

Go-Explore: **Go** to a State You Have Visited without adding exploration and Randomly **Explore** from There



- Go-Explore assumes the environment can be reset to any state; in reality, you may need to train a goal-conditioned policy

Go-Explore: **Go** to a State You Have Visited without adding exploration and Randomly **Explore** from There

- State representations of Go-Explore: (1) convert an image to grayscale, (2) down-scale to 11×8 resolution, and (3) rescale pixel intensities between 0 and 8, instead of the original 0 to 255

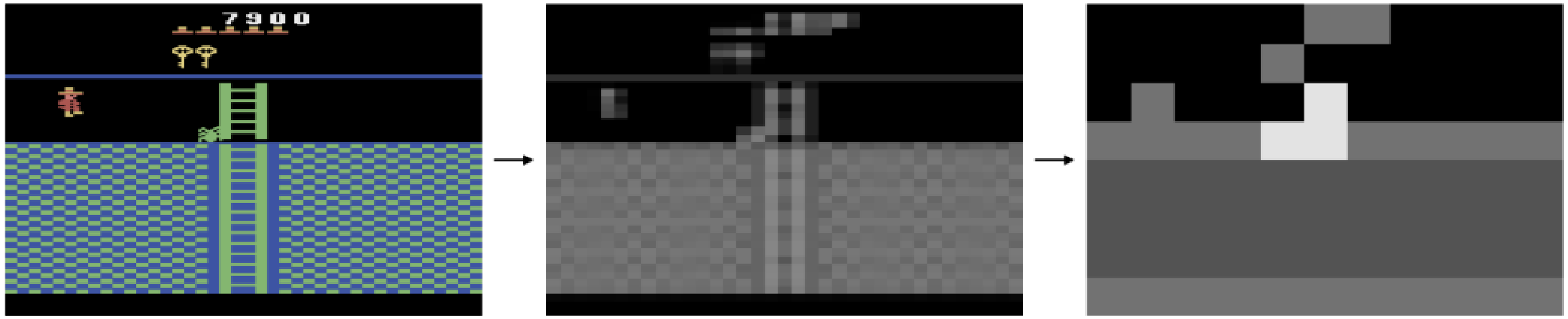


Figure 3: **Example cell representation without domain knowledge, which is simply to down-sample each game frame.** The full observable state, a color image, is converted to grayscale and downscaled to an 11×8 image with 8 possible pixel intensities.

Go-Explore: **Go** to a State You Have Visited without adding exploration and Randomly **Explore** from There

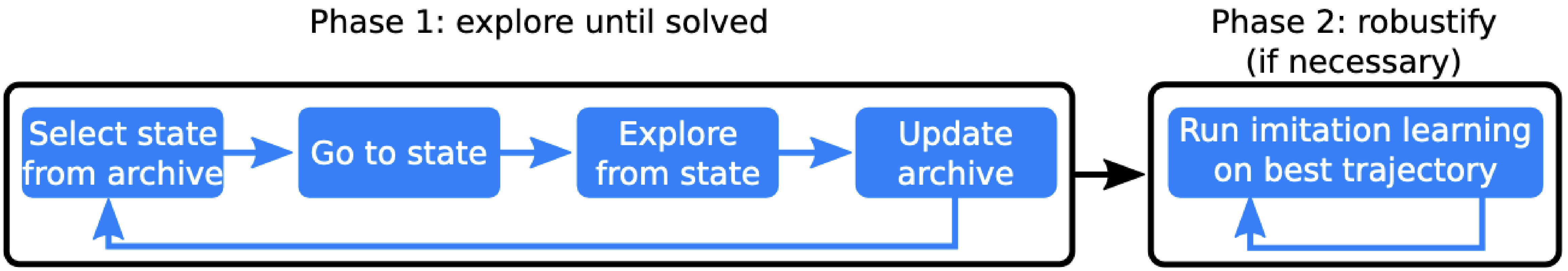
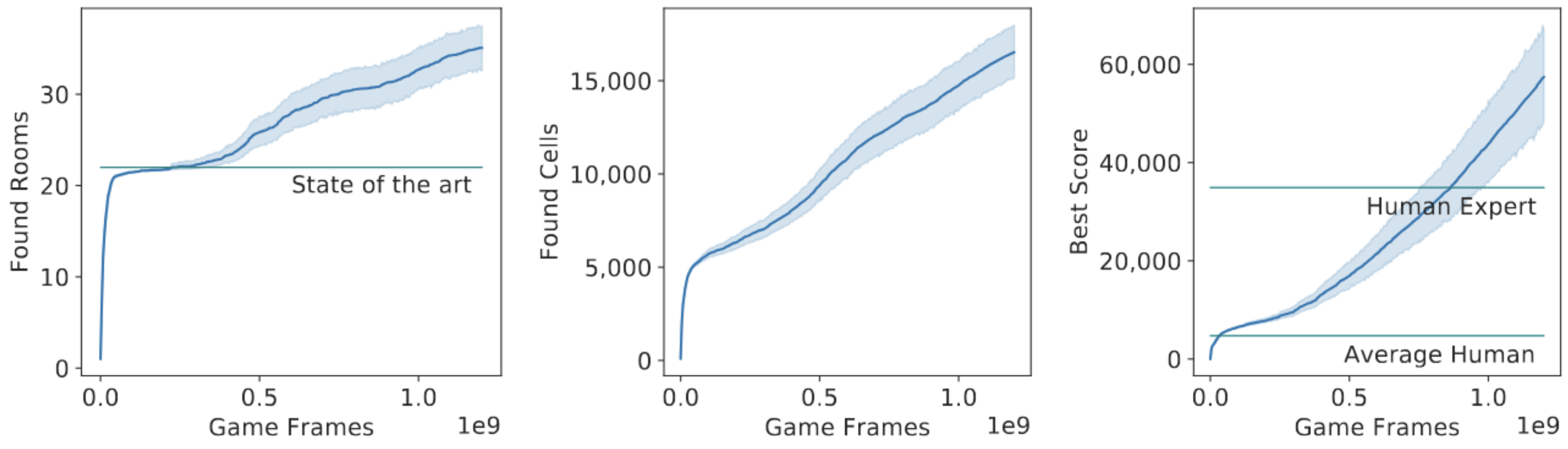

Phase 1: explore until solved
Select state from archive
Go to state
Explore from state
Update archive
Phase 2: robustify (if necessary)
Run imitation learning on best trajectory

Figure 2: **A high-level overview of the Go-Explore algorithm.**

Breakthrough in Long-horizon Tasks



(a) Number of rooms found (b) Number of cells found (c) Maximum score in archive

Figure 4: **Performance of the exploration phase of Go-Explore with downscaled frames on Montezuma's Revenge.** Lines indicating human and the algorithmic state of the art are for comparison, but recall that the Go-Explore scores in this plot are on a deterministic version of the game (unlike the post-Phase 2 scores presented in this section).

Breakthrough in Long-horizon Tasks

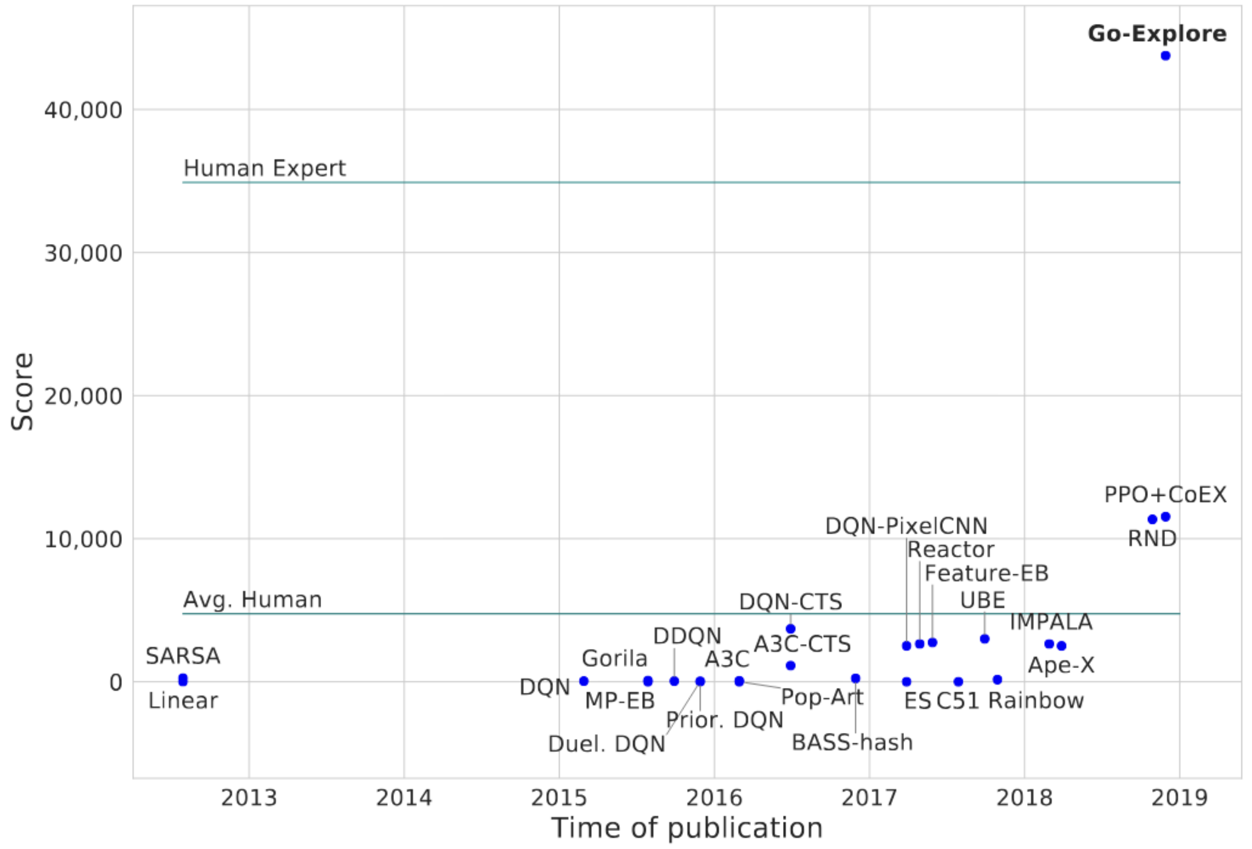


Figure 6: **History of progress on Montezuma's Revenge vs. the version of Go-Explore that does not harness domain knowledge.** Go-Explore significantly improves on the prior state of the art. These data are presented in tabular form in Appendix A.9.

How to Model Exploration?

- We can re-formulate the objective of reinforcement learning:

$$R^t(s,a,s') = r\underbrace{(s,a,s')}_{\substack{\text{extrinsic reward} \\ \text{(related to the task)}}} + \underbrace{\mathcal{B}^t(s,a,}_{\substack{\text{intrinsic reward} \\ \text{(unrelated to the task)}}}s')$$

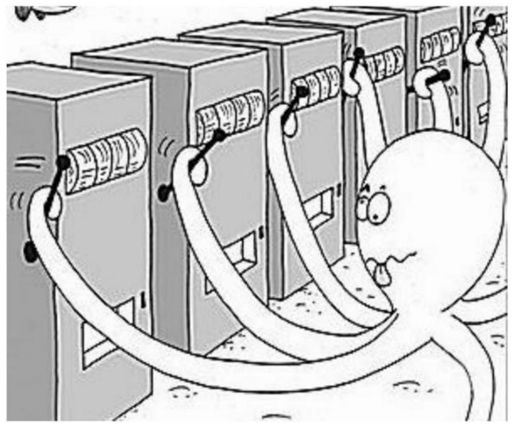
- Types of intrinsic rewards:
  1. Data-based exploration (Optimistic exploration)
  2. Knowledge-based exploration
  3. Information gain exploration
  4. Reachability exploration
  5. Posterior sampling for exploration

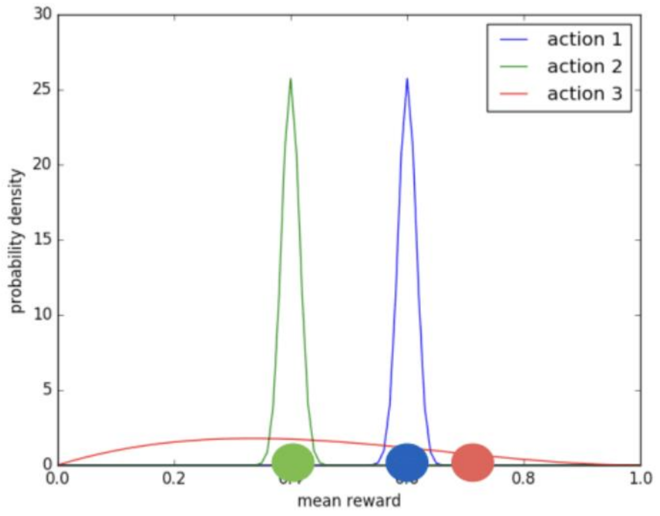
Posterior Sampling for Exploration

- Exploration often waste interaction budget on low-reward states
- Idea:
  - ➢ To explore more efficiently, an agent should explore only when there are valuable learning opportunities
  - ➢ Since any action may have long term consequences, the agent should reason about the informational value of possible observation sequence

- **Thompson sampling automatically balances exploitation and exploration** : it chooses the action that maximizes the expected reward with respect to a randomly drawn belief, and updating the belief with respected to received reward

Thompson Sampling in Multi-arm Bandits





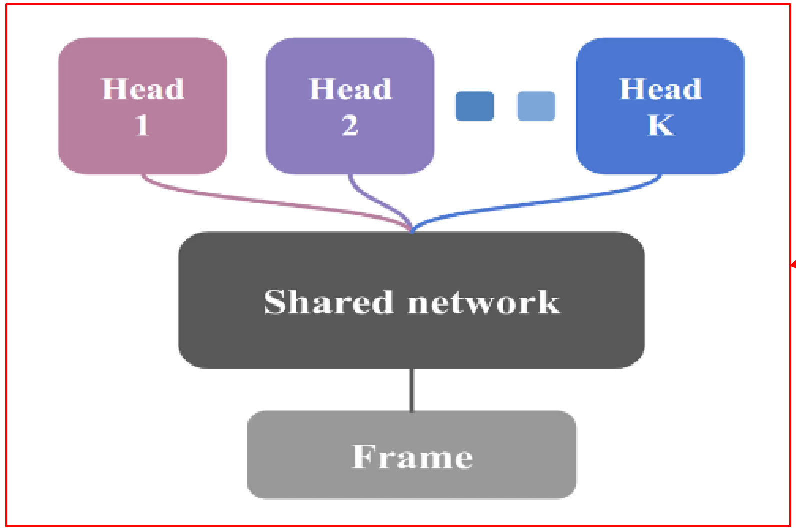
- Posterior distribution: $P(\theta|D) \propto P(\theta)P(D|\theta)$
- Thompson sampling: Represent a posterior distribution of mean rewards of the arms, as opposed to point estimates
  1. Sample from it: $\theta_t \sim P(\theta)$
  2. Choose action: $a = \underset{a}{\operatorname{argmax}} \mathbb{E}_{\theta_t}[r(a;\theta_t)]$
  3. Play action, observe reward
  4. Update the posterior distribution
  $$P(\theta) \leftarrow P(\theta|D) \propto P(\theta)P(D|\theta)$$
- A simple explanation on stack overflow: https://stats.stackexchange.com/questions/187059/what-is-thompson-sampling-in-laymans-terms

Thompson Sampling for General MDP

- Represent a posterior distribution of Q, instead of a point estimate:
  1. Sample from $\mathrm{Q} \sim P(Q|D)$
  2. Choose action according to this Q function for an episode: $a = \underset{a}{\operatorname{argmax}}\, Q(a, s)$
  3. Play action, observe reward
  4. Update the Q distribution

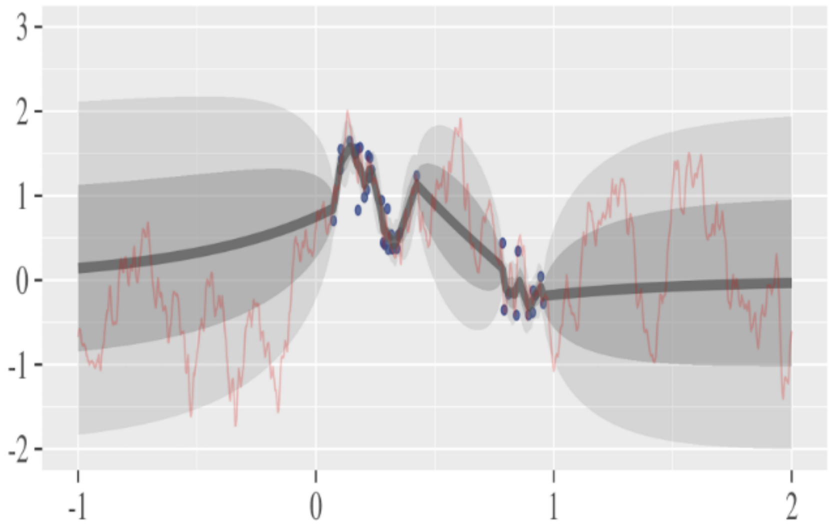
- There's no need for ϵ-greedy for exploration!
- How can we learn a distribution of Q functions $P(Q|D)$ if Q function is a deep neural network?

The Same Idea of Bootstrap!

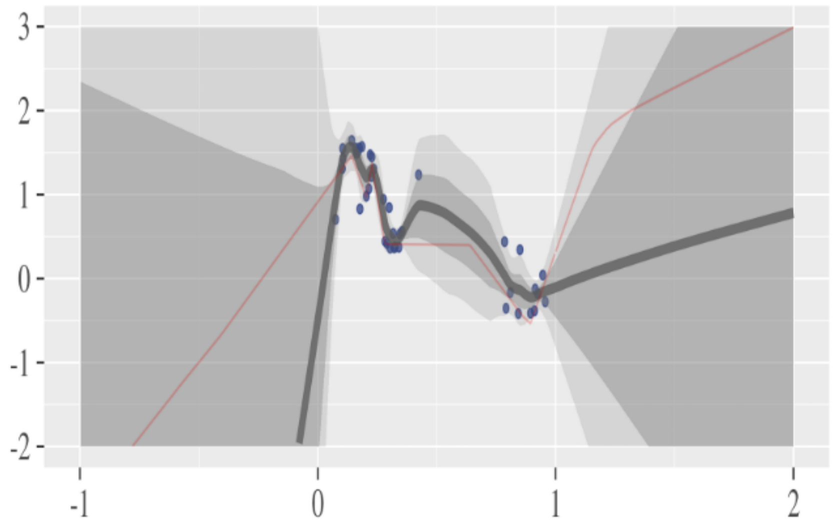


- How to represent a posterior distribution of Q?
  1. **Bayesian neural networks**. Estimate posteriors for the neural weights, as opposed to point estimates.
  2. **Neural network ensembles**. Train multiple Q-function approximations each on using different subset of the data. A reasonable approximation to 1.
  3. Neural network ensembles with shared backbone. Only the heads are trained with different subset of the data. A reasonable approximation to 2 with less computation.
  4. **Ensembling by dropout**. Randomly mask-out (zero out)neural network weights, to create different neural nets, both at train and test time. reasonable approximation to 2.

Bootstrap Captures the Model Uncertainty

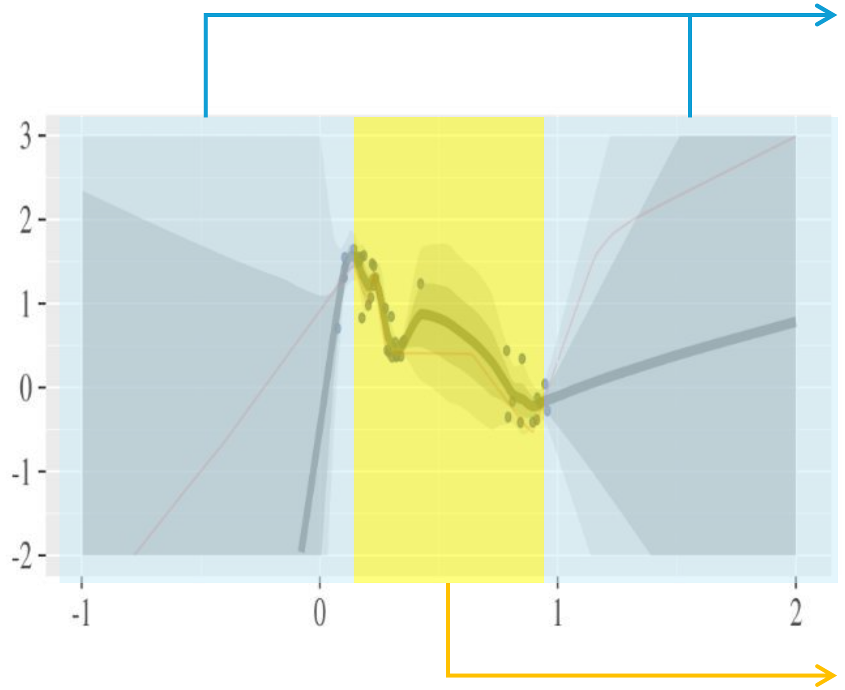


(b) Gaussian process posterior



(c) Bootstrapped neural nets

Bootstrap Captures the Model Uncertainty



In no data regime, Q function values will have high entropy and encourage exploration

When Q values have low entropy, the agent exploit, no need for exploration

Results

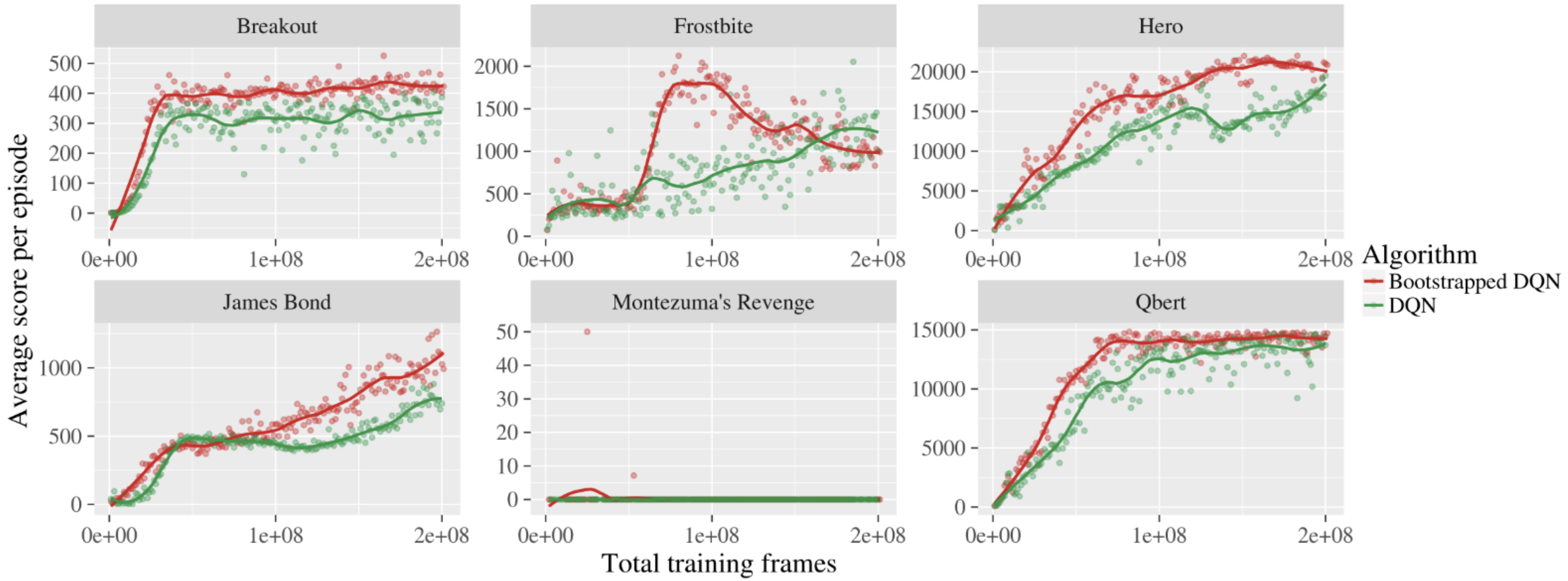
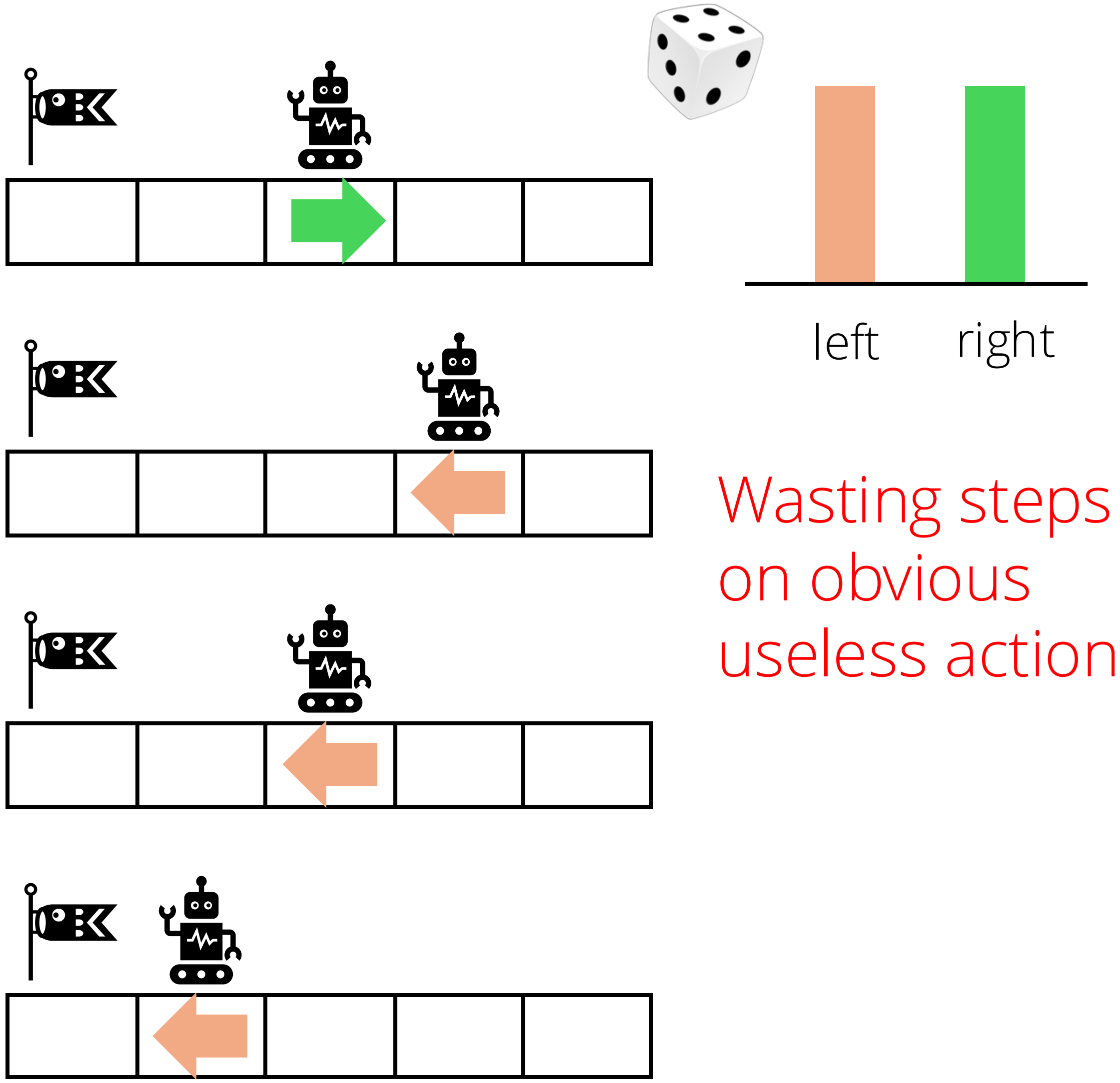


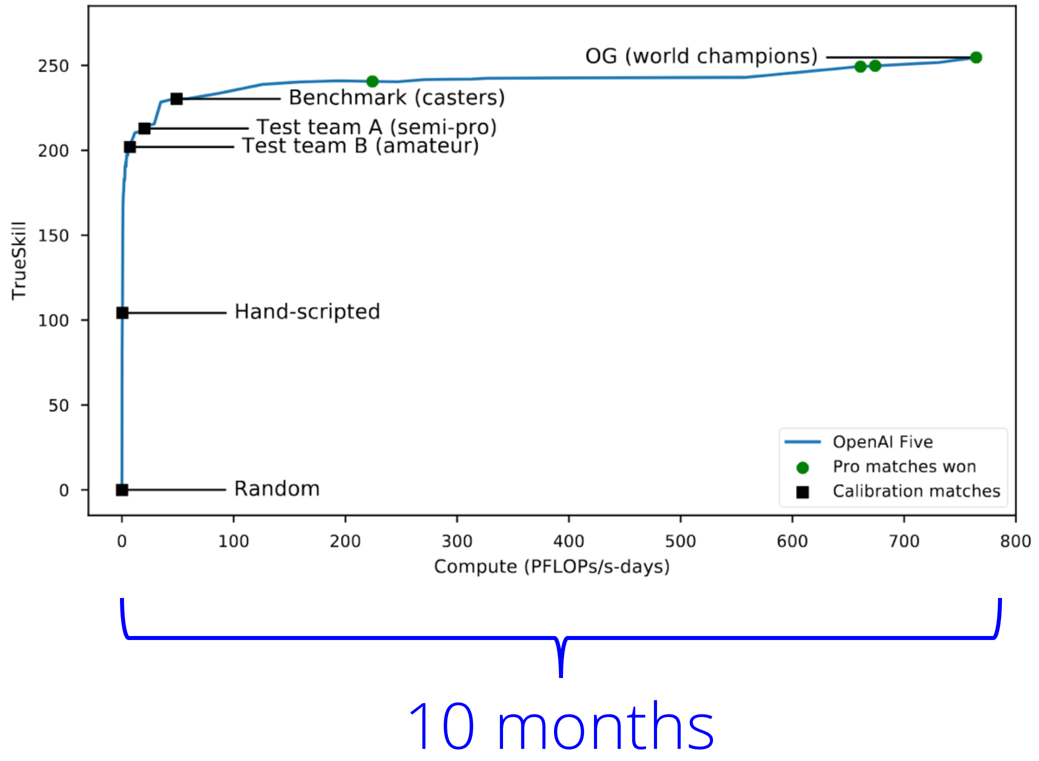
Figure 6: Bootstrapped DQN drives more efficient exploration.

Is Exploration Solved?

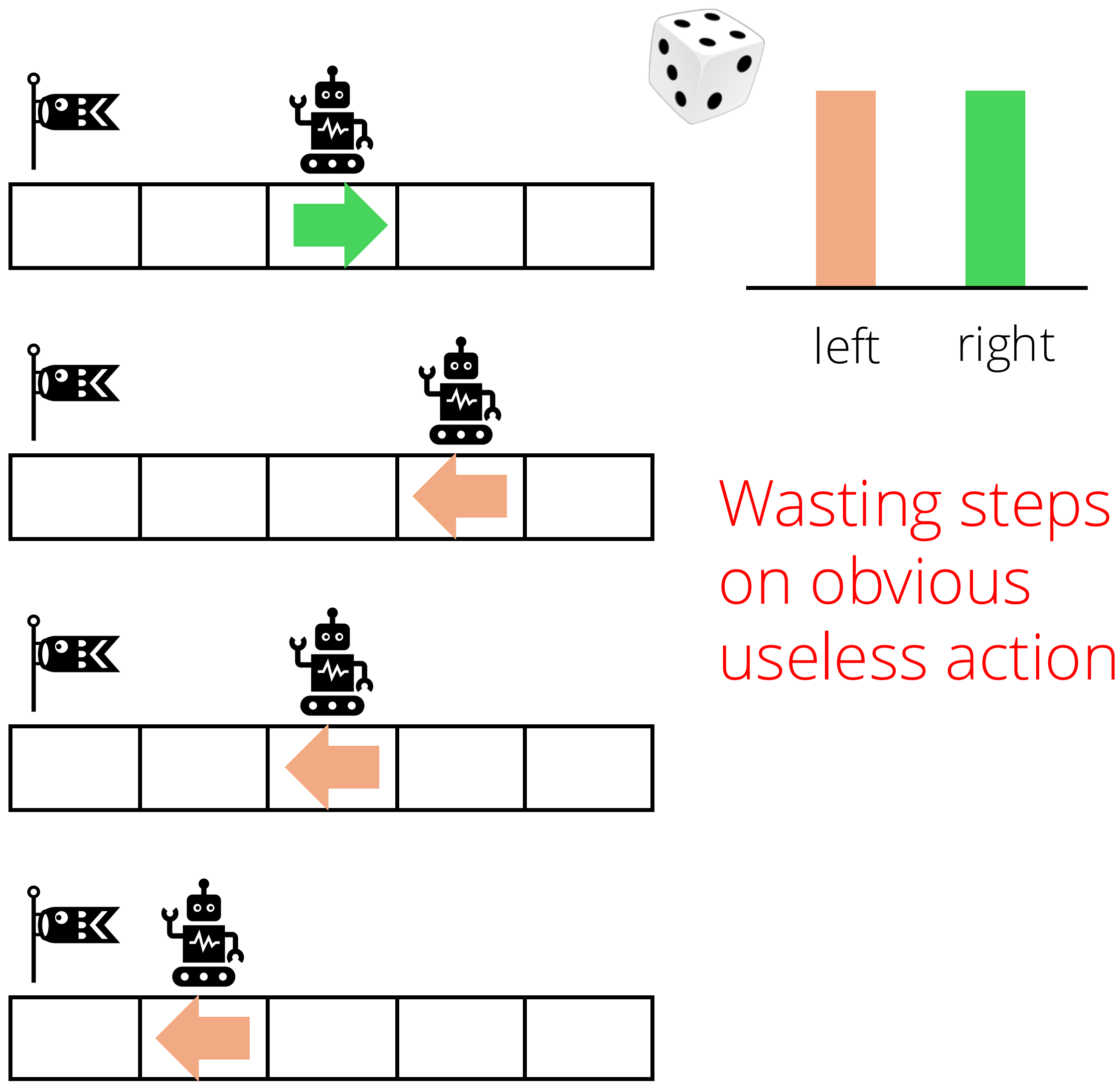
Exploration Based on Inductive Bias are Inefficient



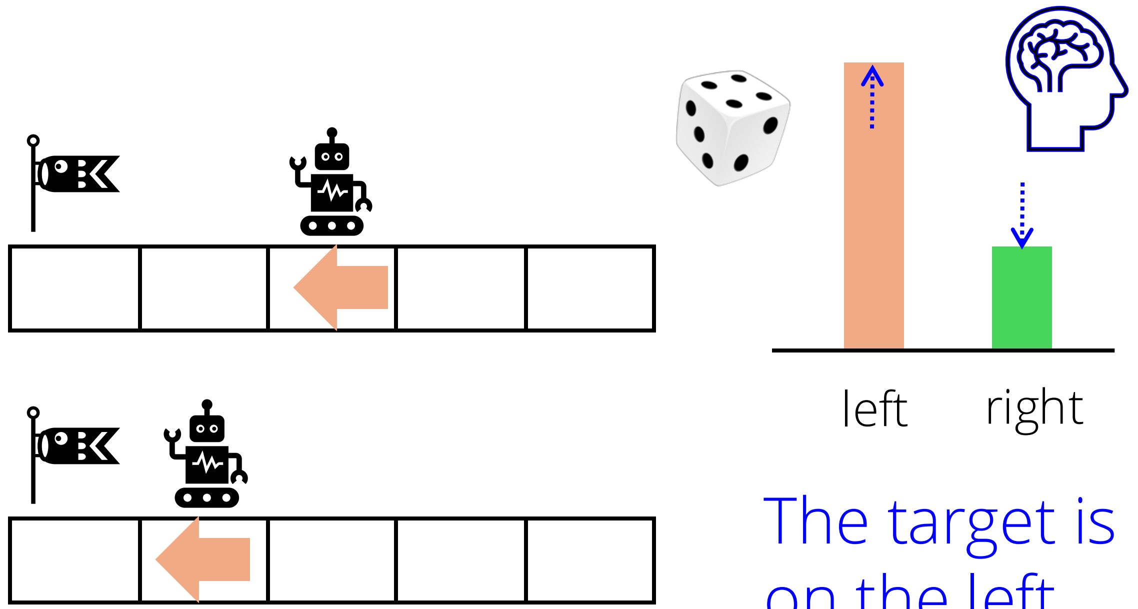
Wasting steps on obvious useless actions



Exploration Based on Inductive Bias vs. Semantic Priors



Wasting steps on obvious useless actions



The target is on the left

Where to Obtain Semantic Priors? Internet!

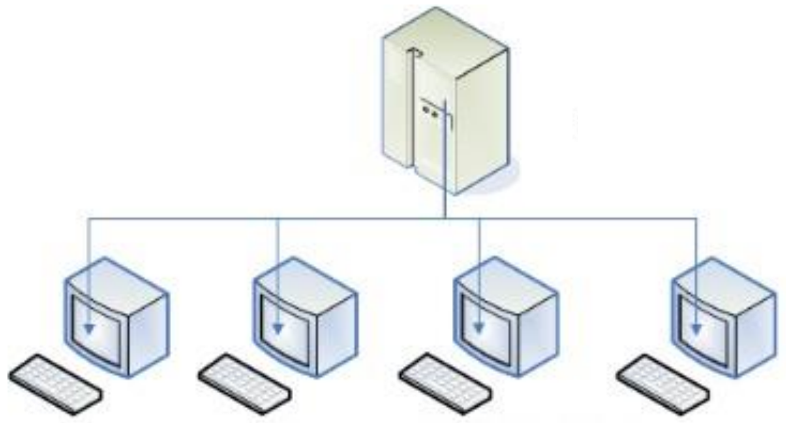


Image credit 鳥哥的首頁

Gemini

Claude

Visual Imitation Learning: Robot Learning From Human Demonstration Videos with Action Labels

HOT3D by Meta

- Pros:
  - ➢ No need for collecting new data
  - ➢ We already have a lot of videos available

- Cons:
  - ➢ The environment state is unknown (e.g. object pose, velocity)
  - ➢ Action labels do not exist
  - ➢ Reward is not available

What information should we extract from human demonstration videos?

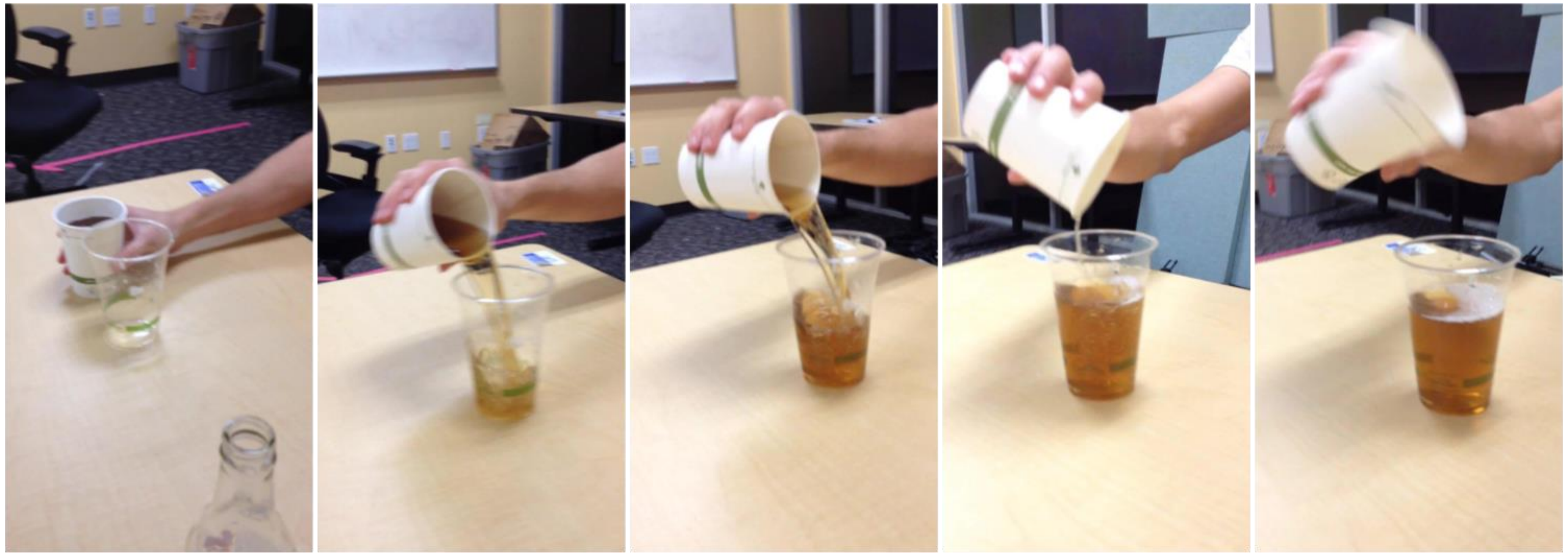
- Types of visual imitation learning:
  1. Knowledge distillation via representation learning
  2. Knowledge distillation via video generation
  3. Knowledge distillation with correspondence
  4. Knowledge distillation with affordance
  5. Knowledge distillation with 3D hand modeling
  6. Knowledge distillation with digital twins

Knowledge distillation via representation learning

- Can we learn robot policies from 3rd person human demonstration videos?

Knowledge distillation via representation learning

- What information do we need to know to imitate the following actions?
  - Object attributes: liquid level in cups, pouring stages, hand contact, liquid is flowing, texture, viscosity, presence of foam ...
  - Object 3D pose / motion
  - Hand 3D motion
  - Lighting / layout of the room, who is the demonstrator ...



Knowledge distillation via representation learning

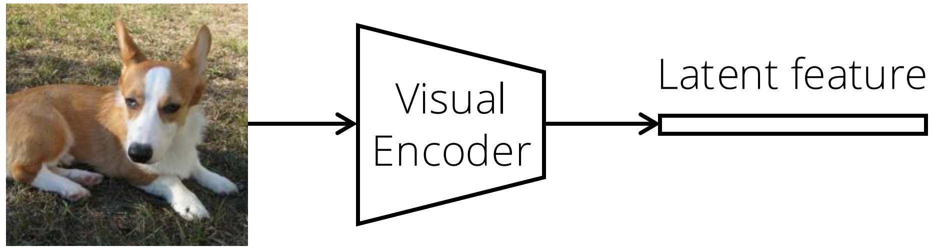
- What information do we need to know to imitate the following actions?
  - ➢ Object attributes: liquid level in cups, pouring stages, hand contact, liquid is flowing, texture, viscosity, presence of foam ...
  - ➢ Object 3D pose / motion
  - ➢ Hand 3D motion
  - ➢ Lighting / layout of the room, who is the demonstrator ...

- An excess of information:
  - ➢ Some are redundant, while others are costly to obtain...
  - ➢ We might still miss useful ones...

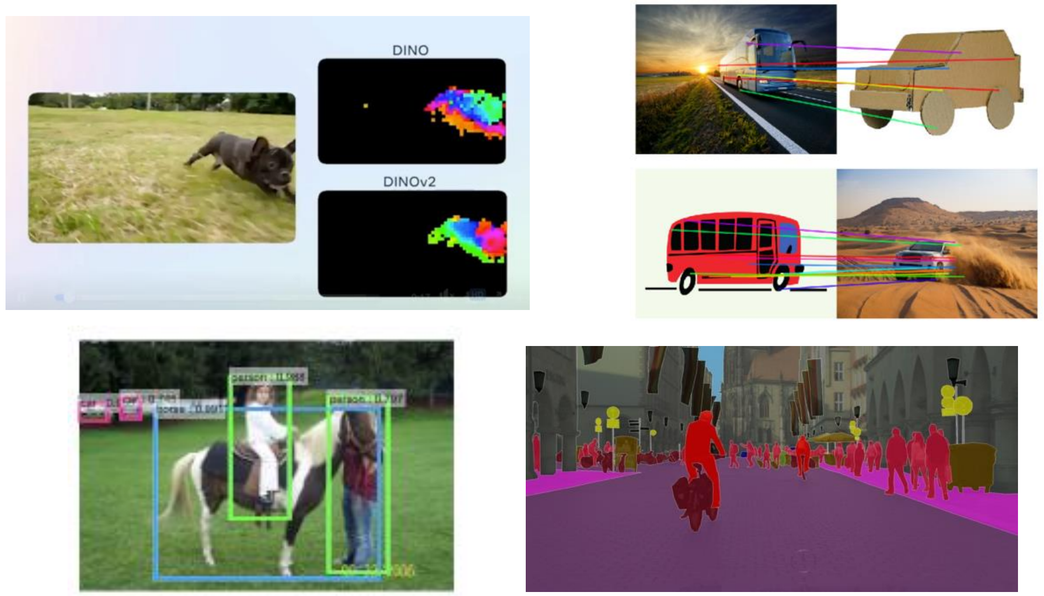
- Can we uncover and disentangle dimensions of the world without any priors?

What We Have Learned from Computer Vision

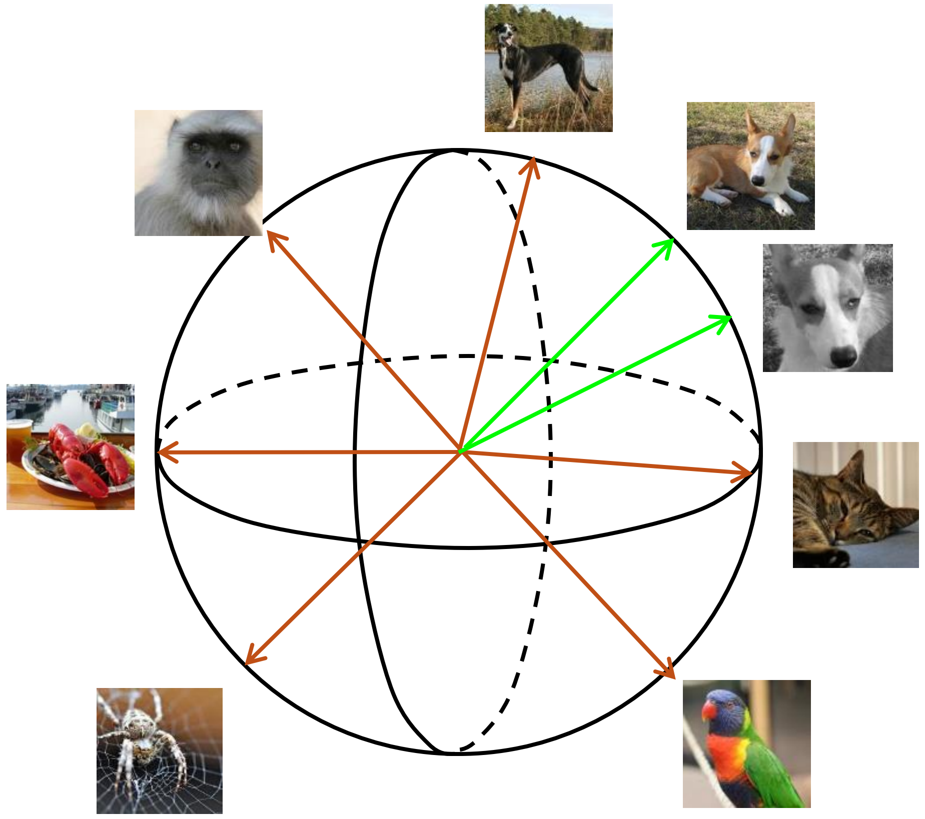
Goal: learn good features without labels



Testing: different downstream tasks



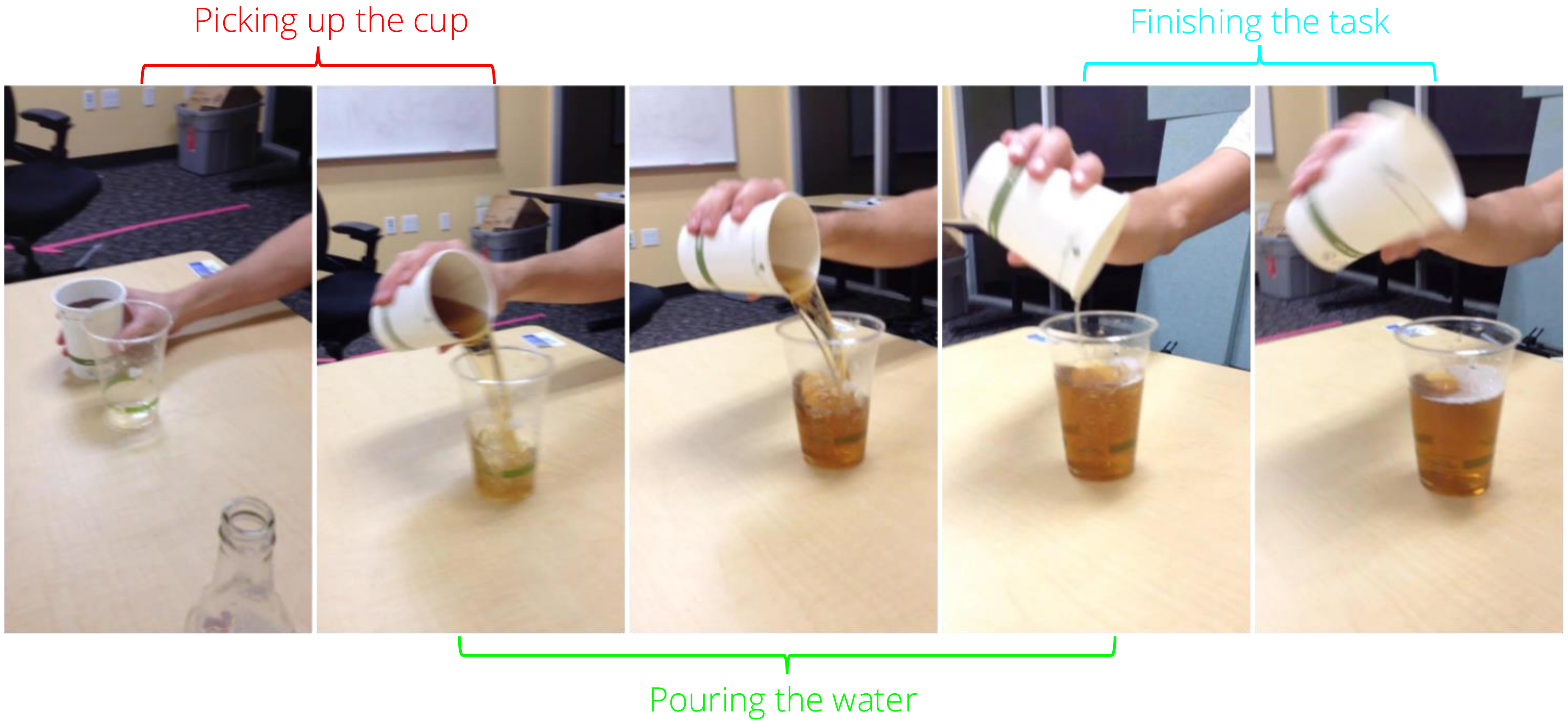
Training: contrast image instances



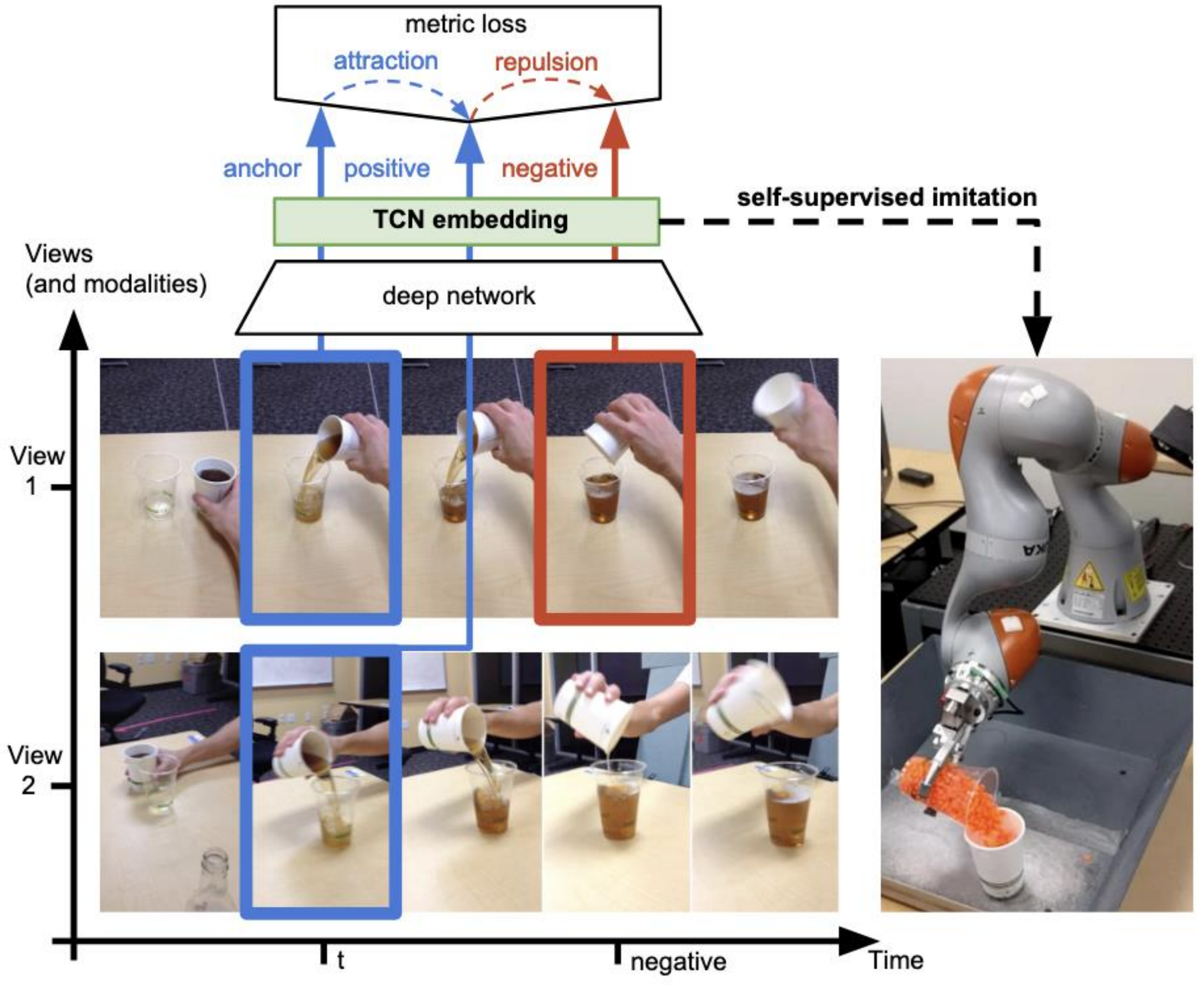
Can We Use the Same Idea for Robot Learning?

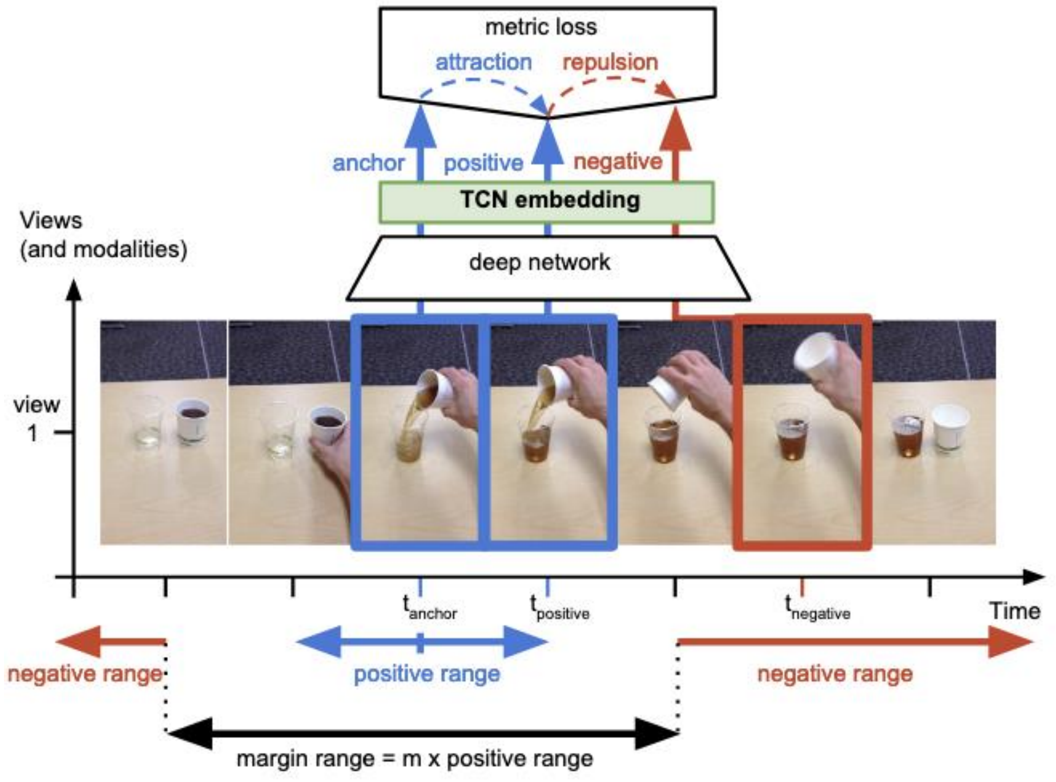
- Unlike conventional computer vision, every image instance are independent
- Video frames are temporally dependent

Intuition: Visual Difference between Neighboring Frames Reveals Change of the World State

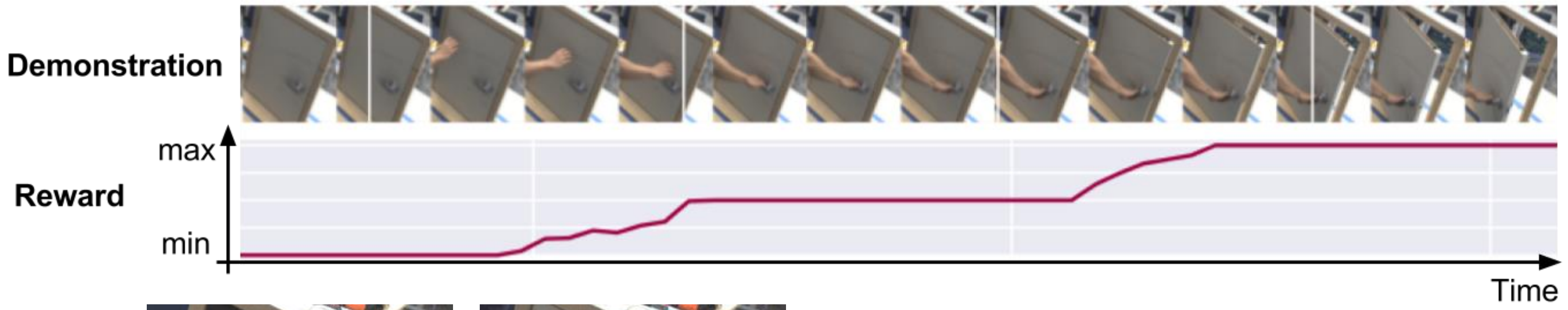


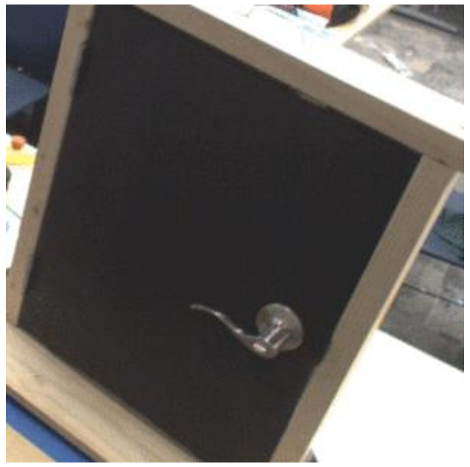
Idea: Contrastive Learning between Video Frames

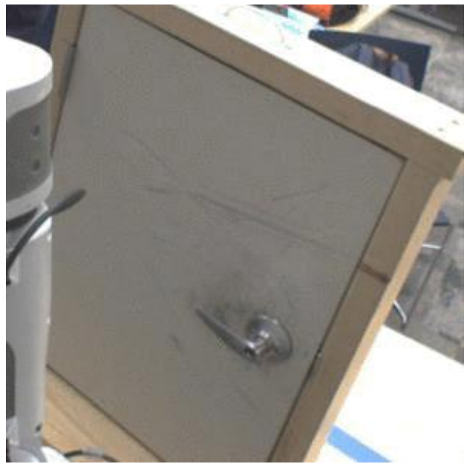




Define Rewards with Learned Features to Follow the Provided Video Demonstration



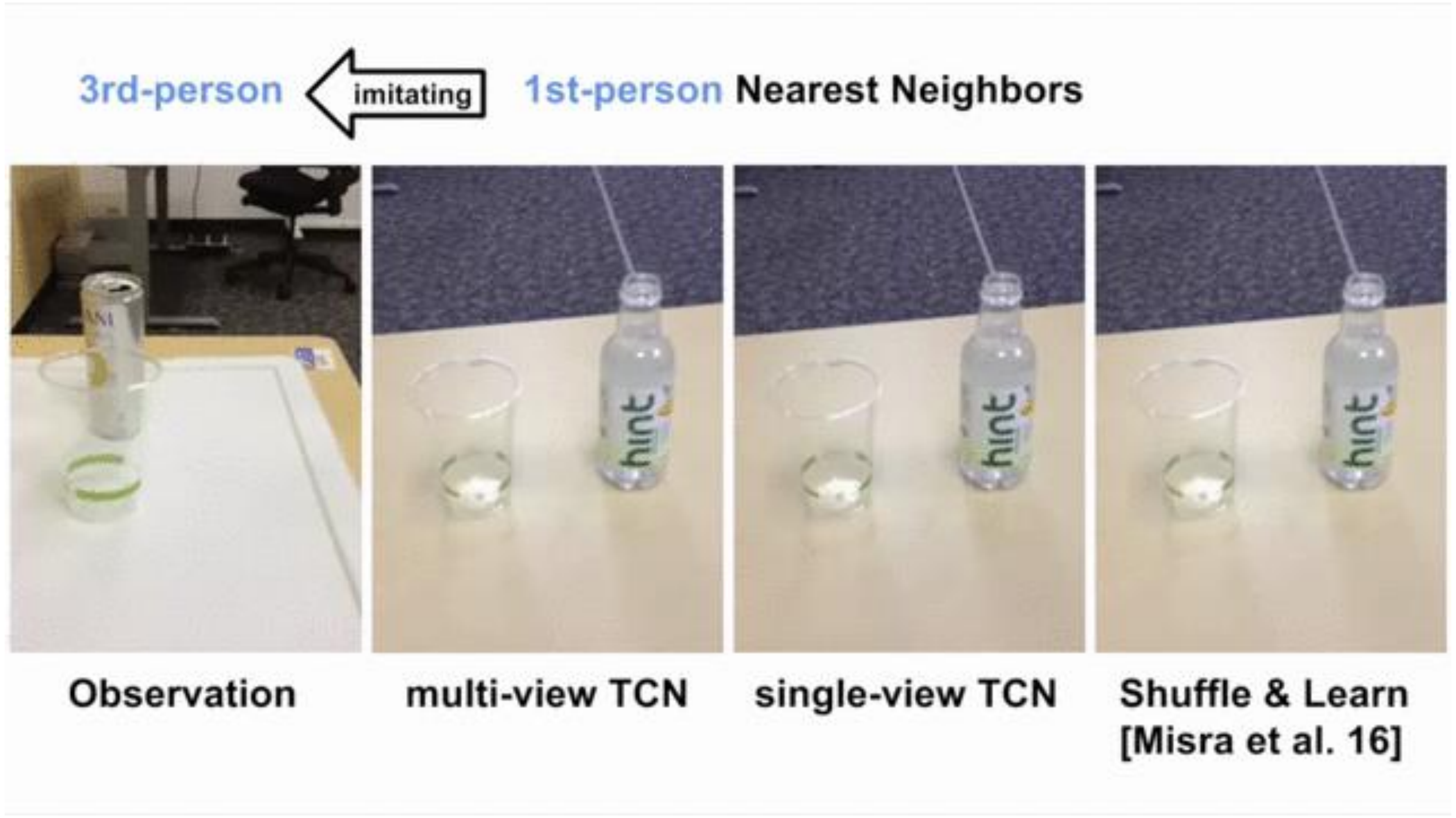




$$R(\mathbf{v}_t, \mathbf{w}_t) = -\alpha \|\mathbf{w}_t - \mathbf{v}_t\|_2^2 - \beta \sqrt{\gamma + \|\mathbf{w}_t - \mathbf{v}_t\|_2^2}$$

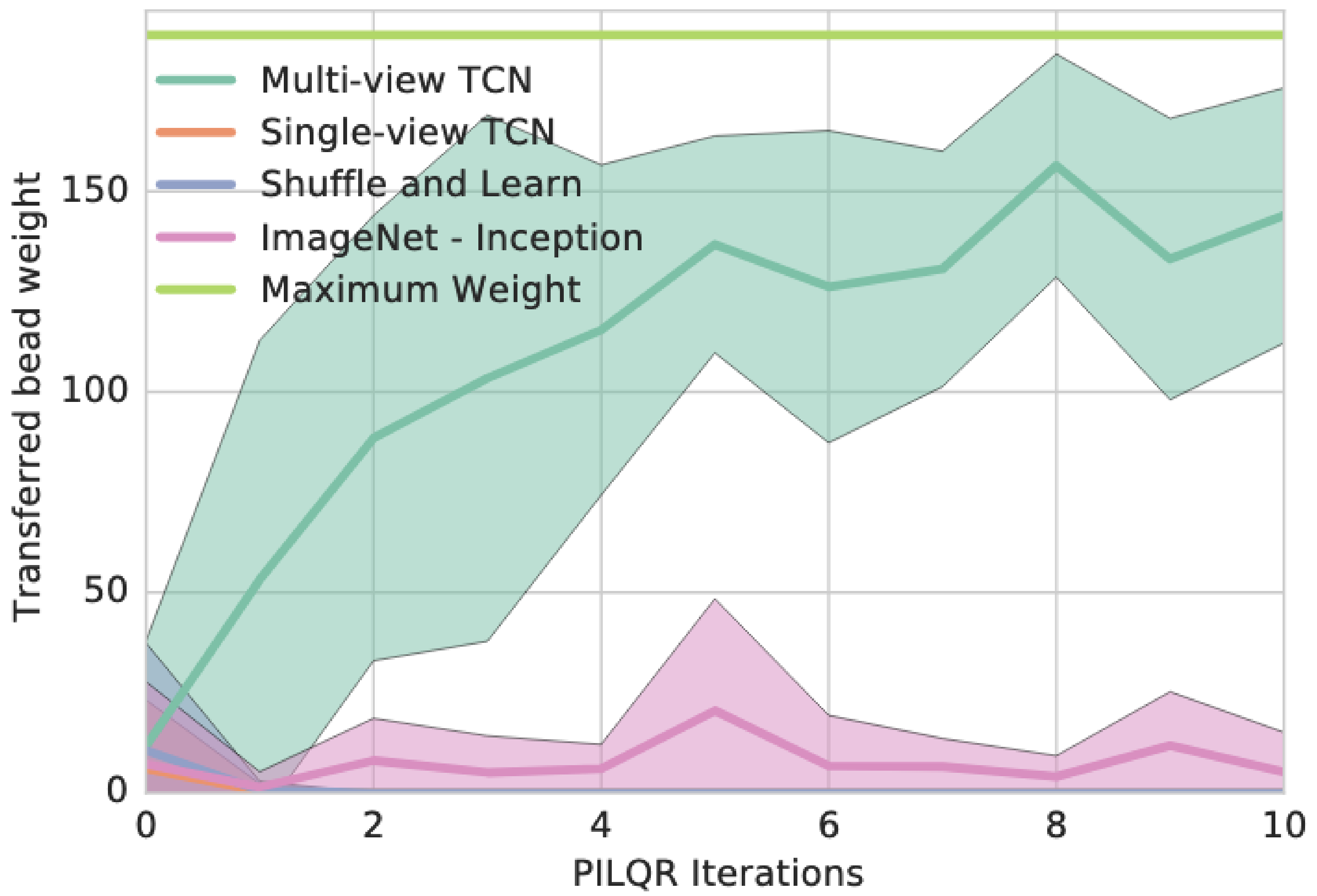
- $V = (v_1, \ldots, v_T)$ is the embeddings of each frame in a video demonstration sequence
- $W = (w_1, \ldots, w_T)$ is the embeddings of image observed during a robot task execution

Per-Frame Nearest Neighbor Retrievals



3rd-person
imitating
1st-person Nearest Neighbors
Observation
multi-view TCN
single-view TCN
Shuffle & Learn [Misra et al. 16]

Learning Progress of the Pouring Task

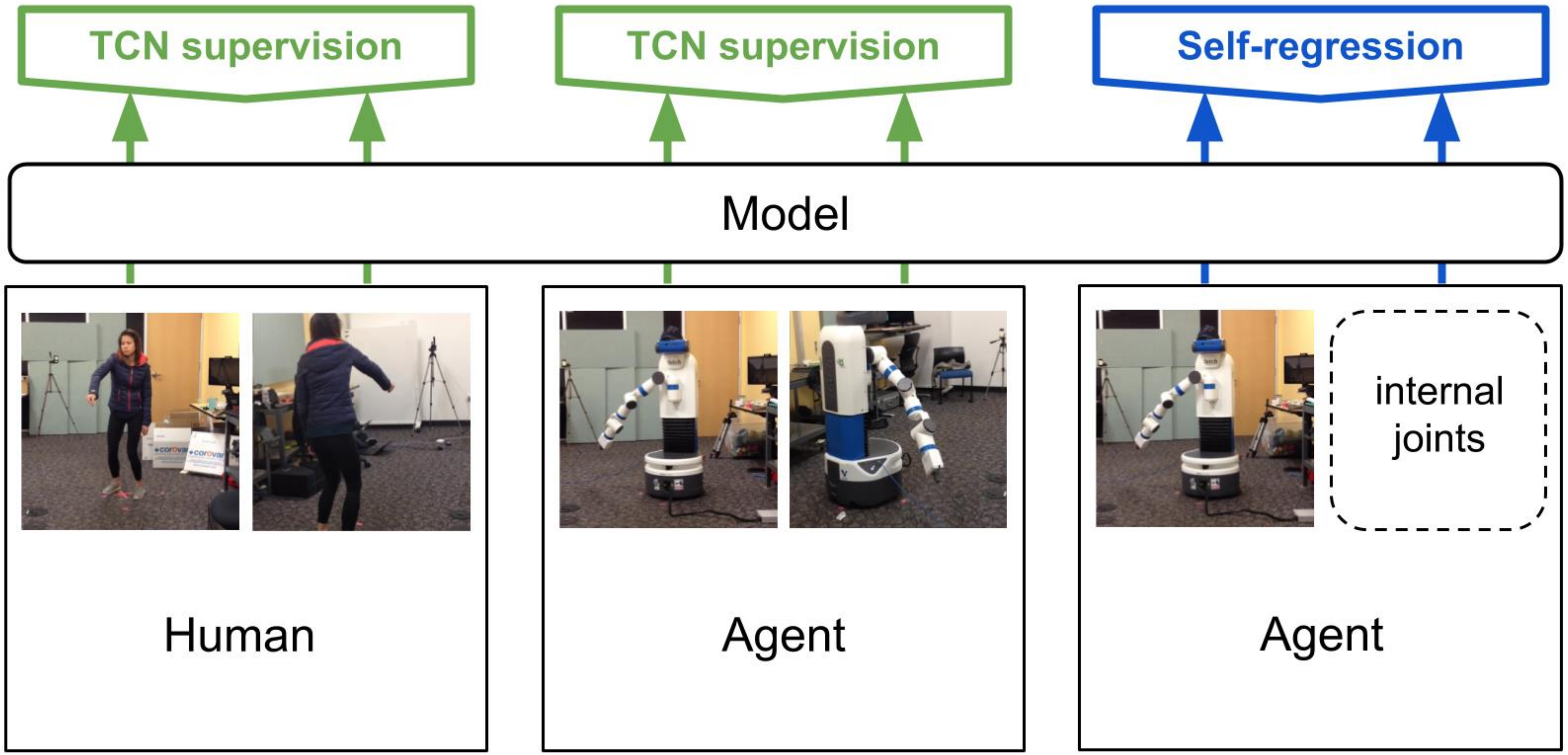


Can We Learn to Follow Demonstration More Closely?

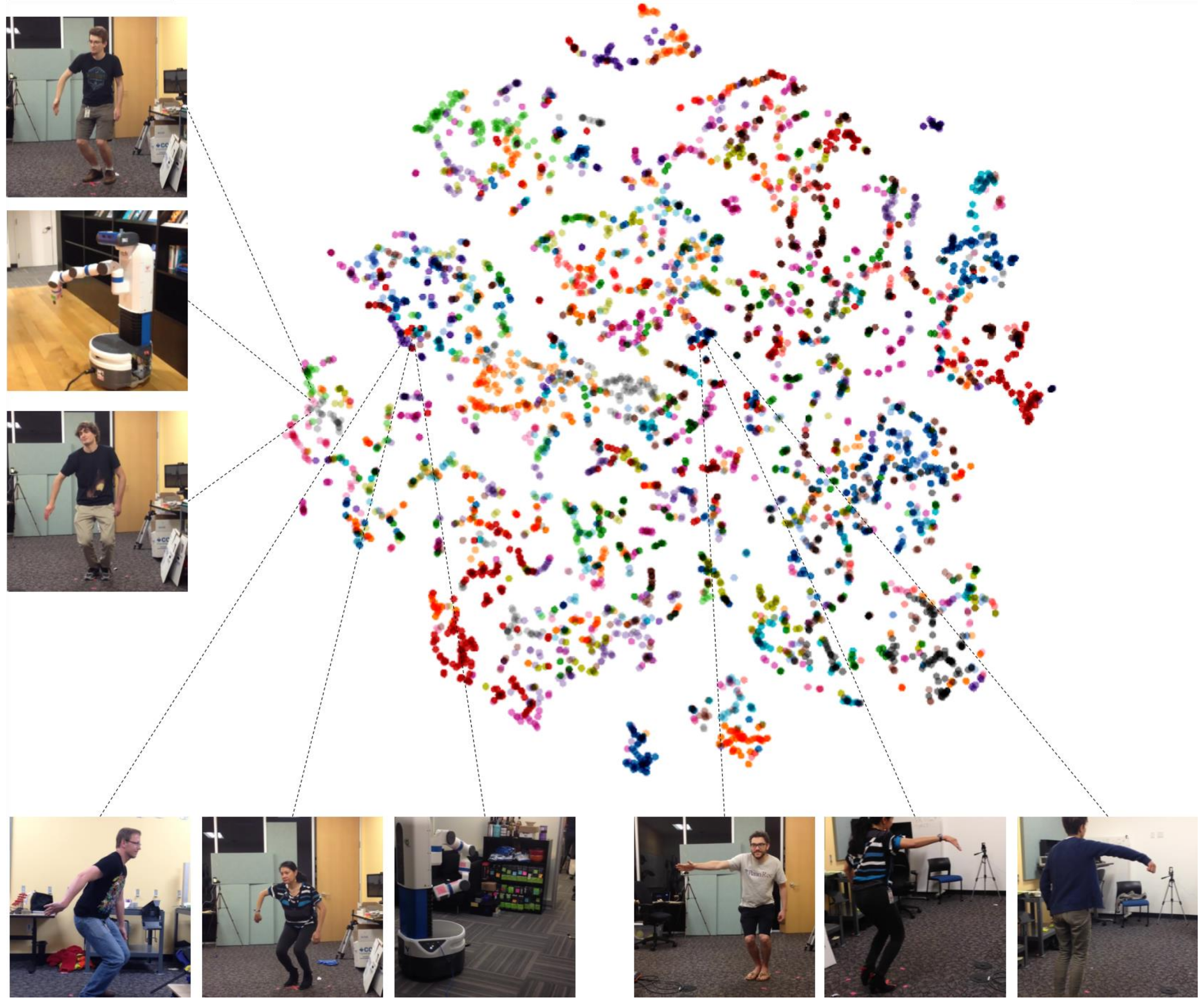
- We hoped to obtain the correct robot motion using RL, however, the task is hard…
- We can learn to predict the robot motion from the encoded features, however, annotating action labels is expensive…
- Idea: Annotate cheap action labels by playing (e.g. random motion, task-agnostic motion …)

$$\xrightarrow{\quad\quad\quad}\ \boxed{\text{Robot joint angles}}$$

Time Contrastive Learning + Action Regression



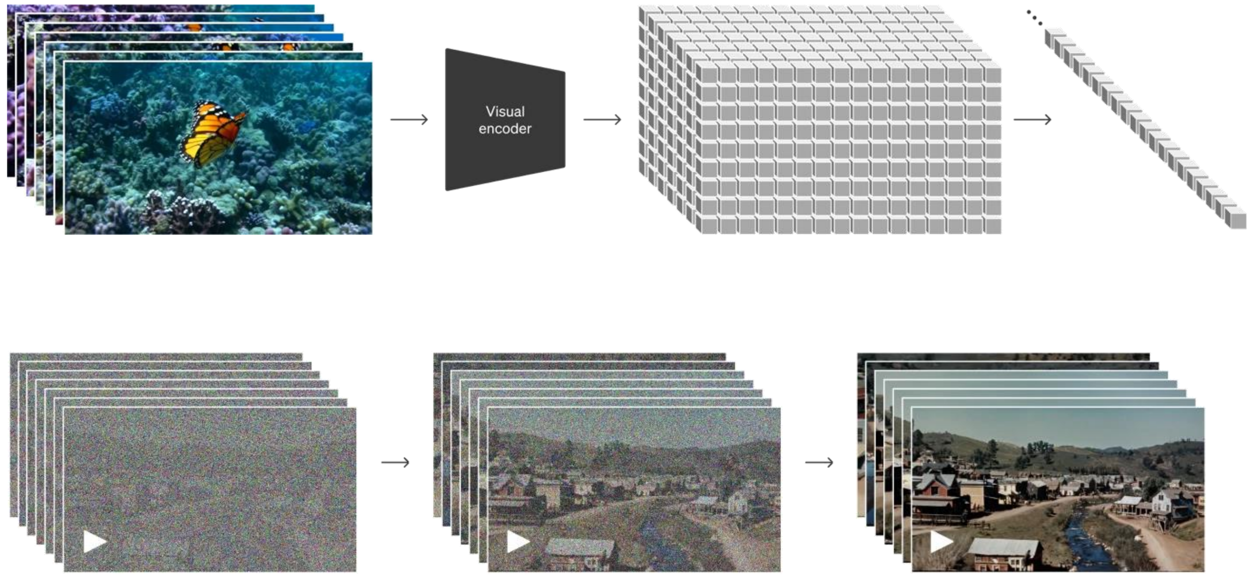
Learning to imitate, from video, without supervision
fetch
3rd-person observation
Imitating
Self-regression



What information should we extract from human demonstration videos?

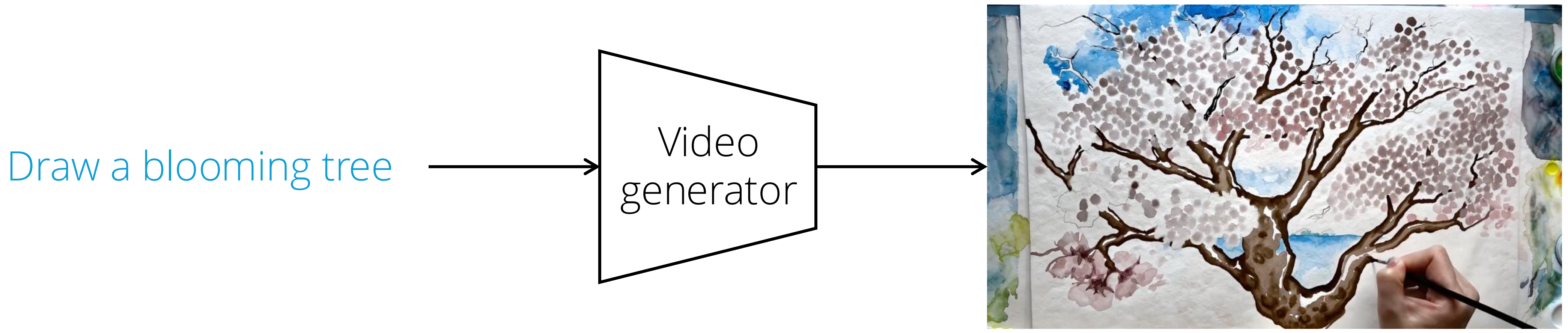
- Types of visual imitation learning:
  1. Knowledge distillation via representation learning
  2. Knowledge distillation via video generation
  3. Knowledge distillation with correspondence
  4. Knowledge distillation with affordance
  5. Knowledge distillation with 3D hand modeling
  6. Knowledge distillation with digital twins

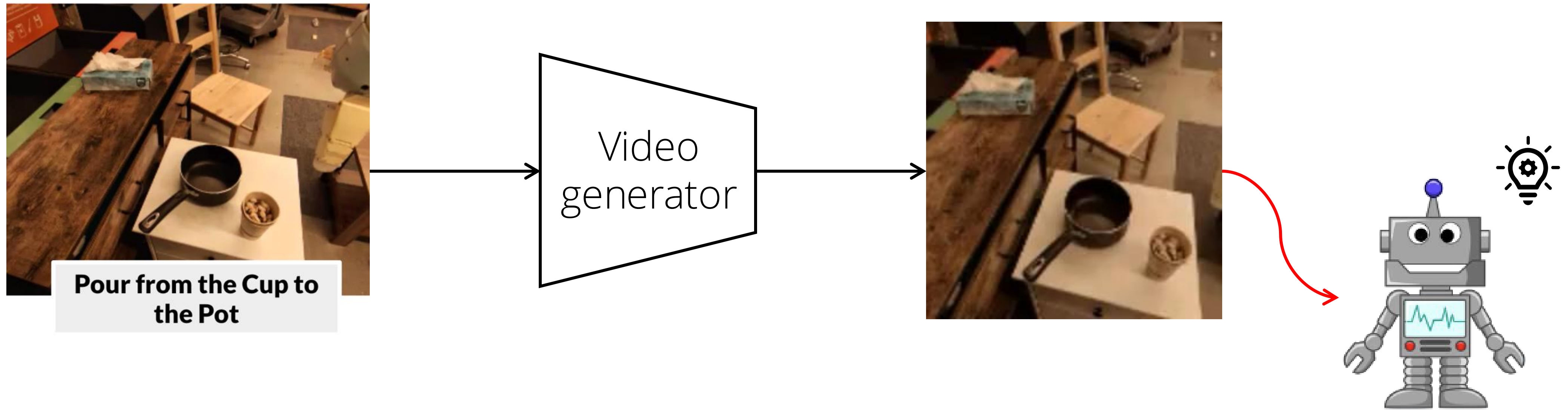
Knowledge distillation via Video Generation



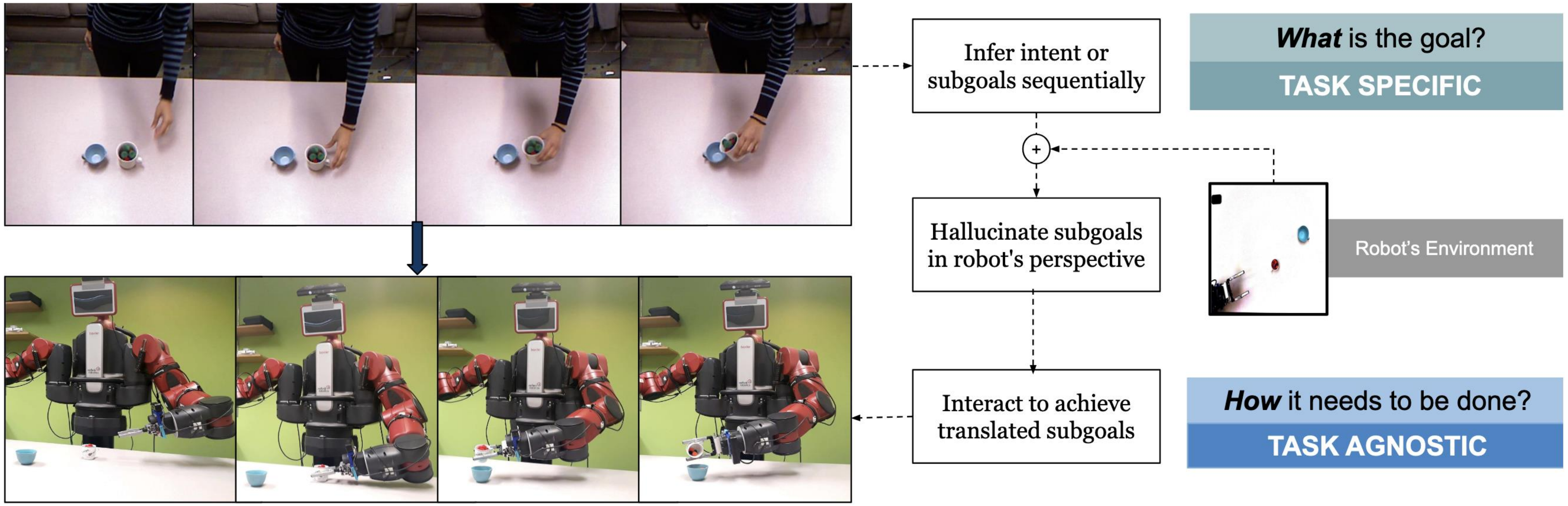
https://openai.com/index/video-generation-models-as-world-simulators/

Video Generator as a Language-conditioned Subgoal Predictor



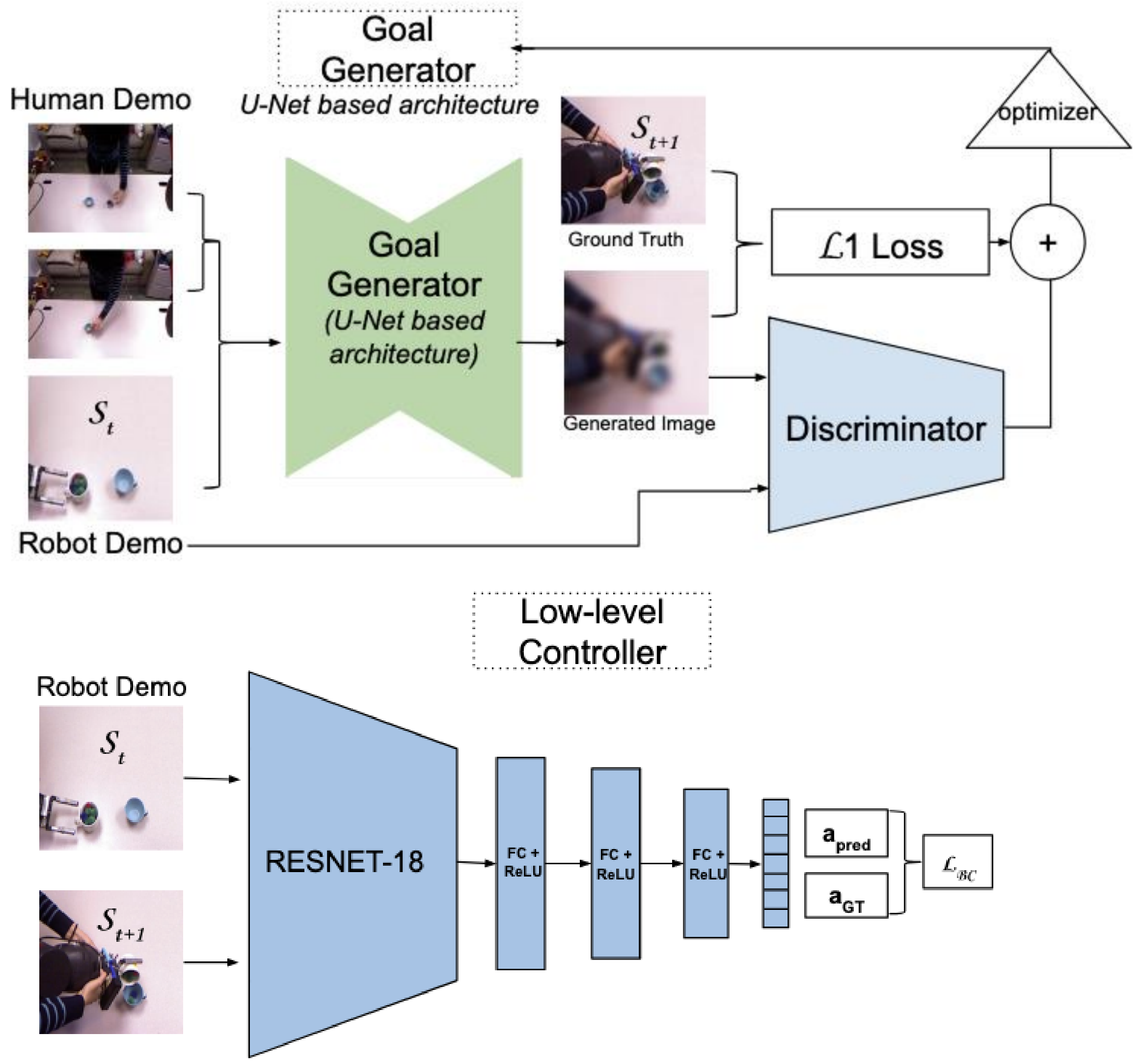


Video Generator as a Language-conditioned Subgoal Predictor


Infer intent or subgoals sequentially
What is the goal?
TASK SPECIFIC
+
Hallucinate subgoals in robot's perspective
Robot's Environment
Interact to achieve translated subgoals
How it needs to be done?
TASK AGNOSTIC

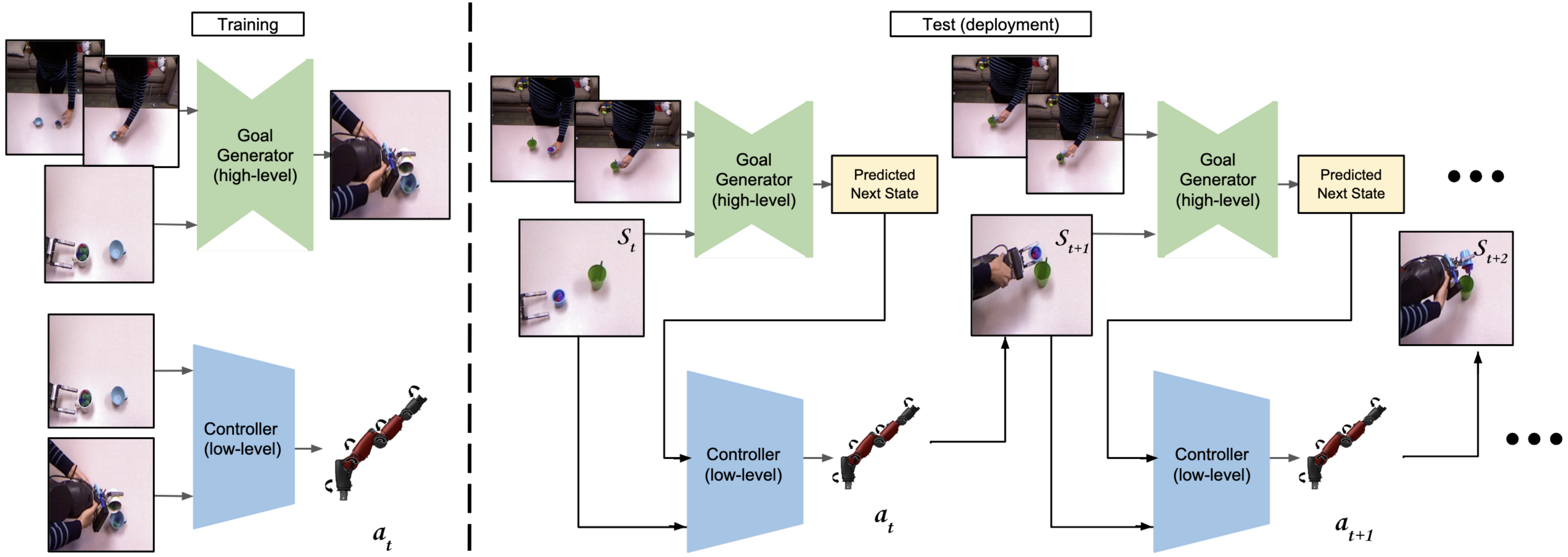
How to solve the cross-embodiment problem: inferring robot action from human video?

Hierarchical Control: A High-level Goal Predictor and a Low-level Goal-conditioned Controller



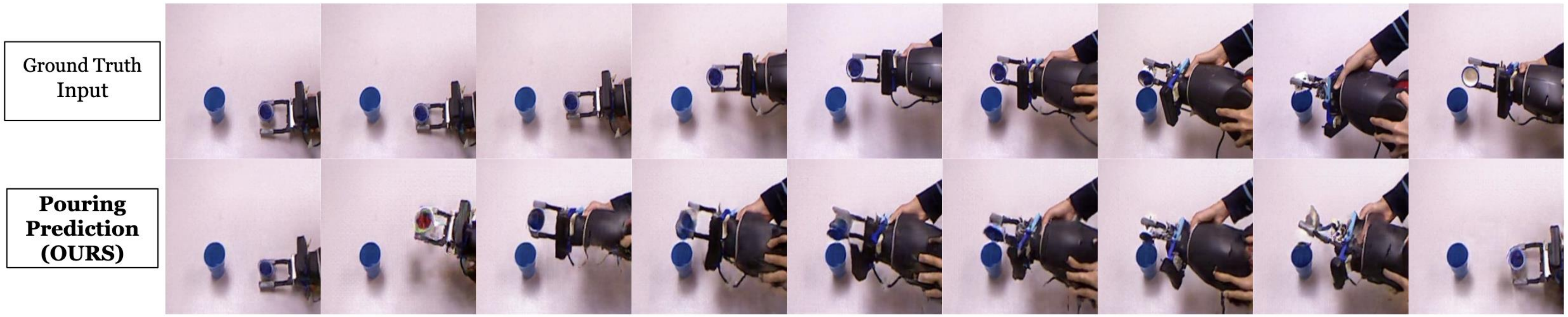
- We need paired dataset of `<robot action, 1`st`-person-view robot observation, and 3`rd`-person-view human observation>`
- High-level goal predictor: translate change in human demonstration to robot observation
- Low-level controller: predict joint angles from robot observation at time t and t + n

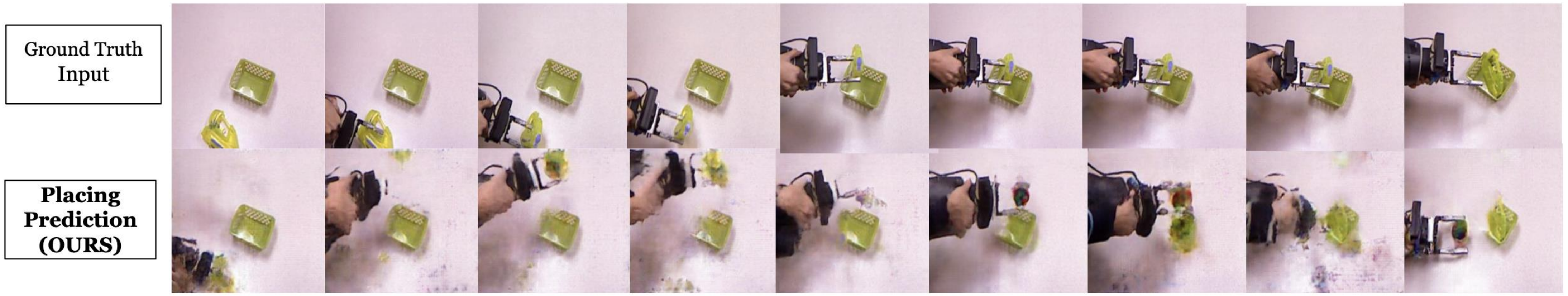
Hierarchical Control: A High-level Goal Predictor and a Low-level Goal-conditioned Controller



Third-Person Visual Imitation Learning via Decoupled Hierarchical Controller. Sharma et al.

Imagined Goals





Method	Pouring		Placing	
	Reaches	Pours	Reaches	Drops
End to End [27]	20%	8%	20%	10%
DAML [30]	25%	15%	20%	10%
Hierarchy (Ours)	**75%**	**60%**	**70%**	**50%**

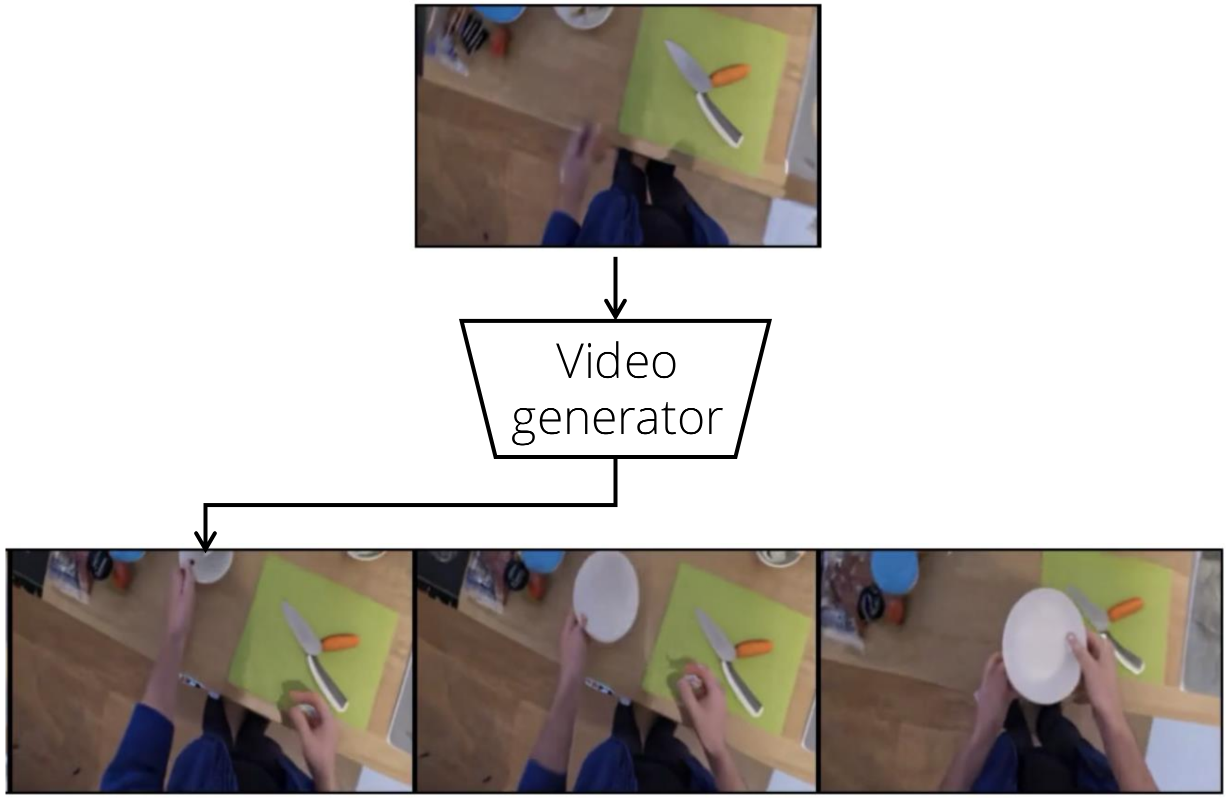
Third-Person Visual Imitation Learning via Decoupled Hierarchical Controller. Sharma et al.

A Bigger Story: Video Generator is a World Model

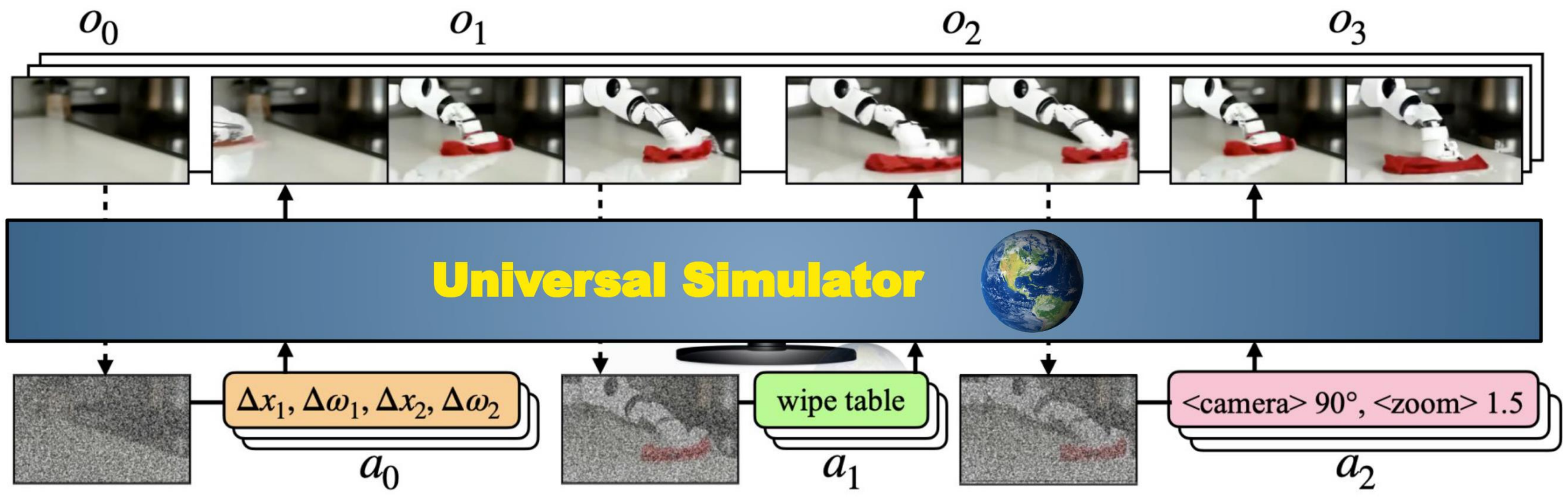
The real world: expensive to interact with

The simulator world: expensive to build and almost impossible to be identical to the real world

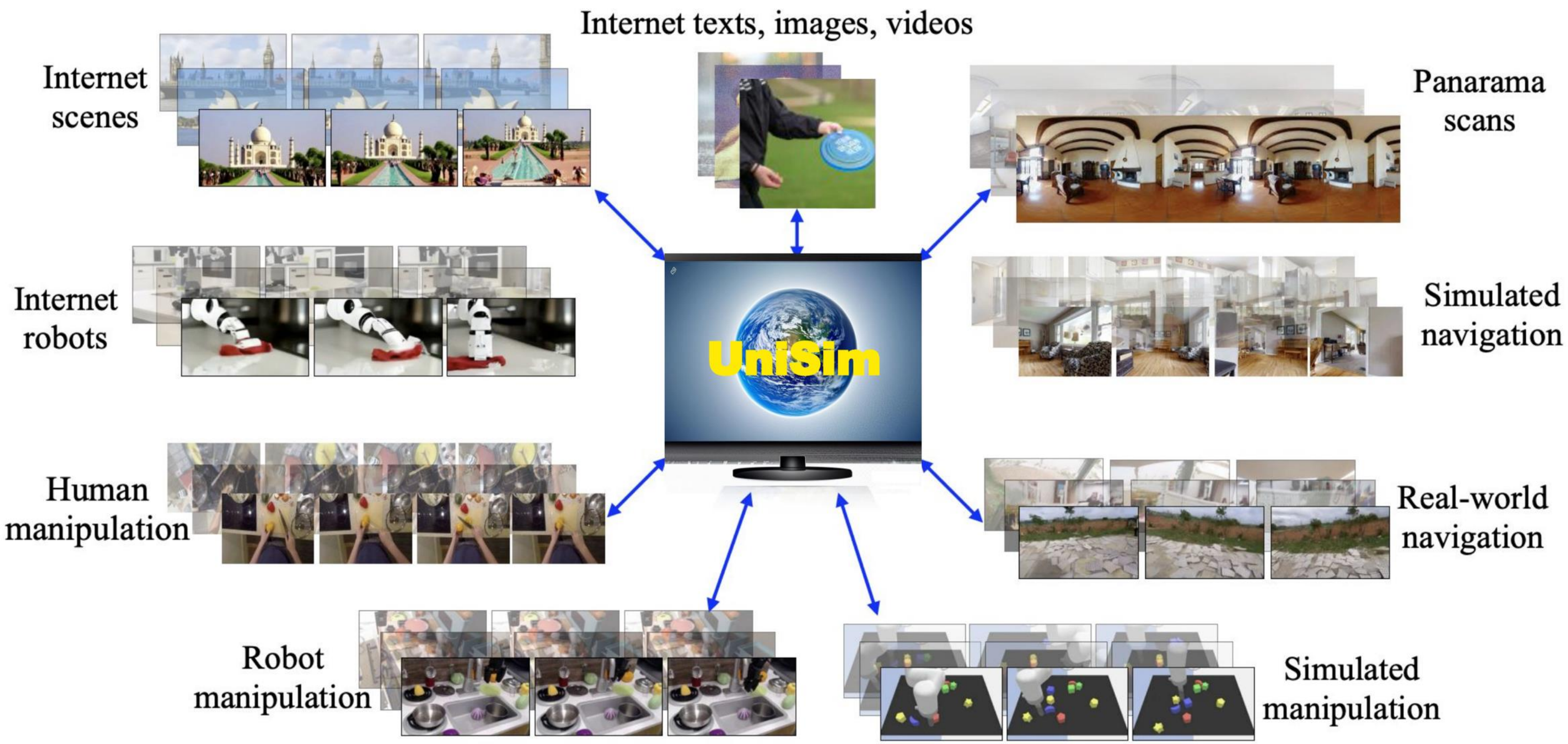
Isaac Sim by Nvidia



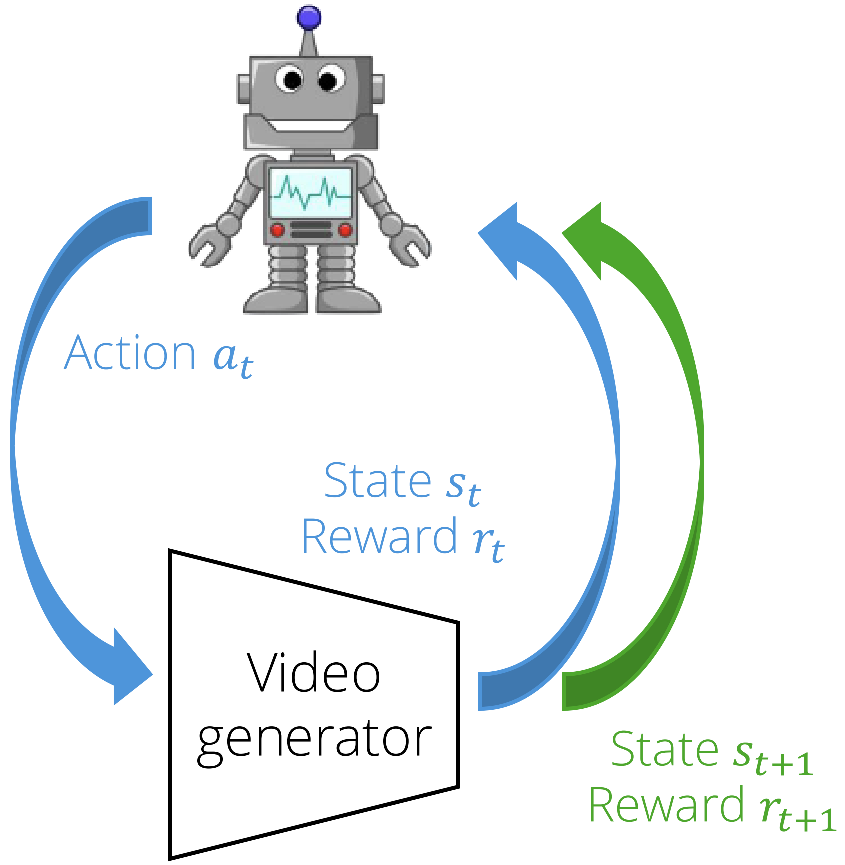
Conditional Video Generator is a World Model with Different Kinds of Interfaces

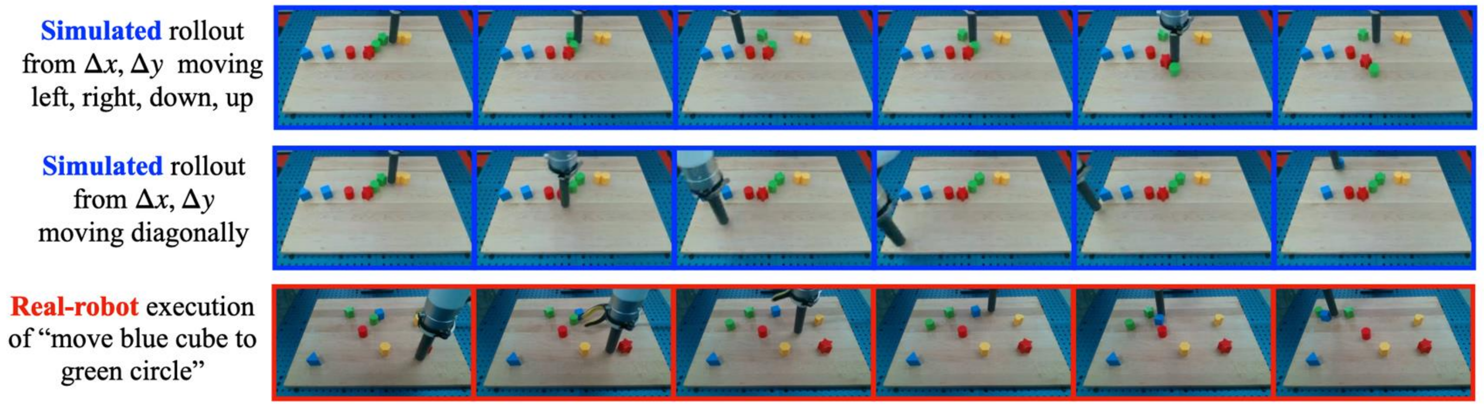


Use All Videos on the Internet



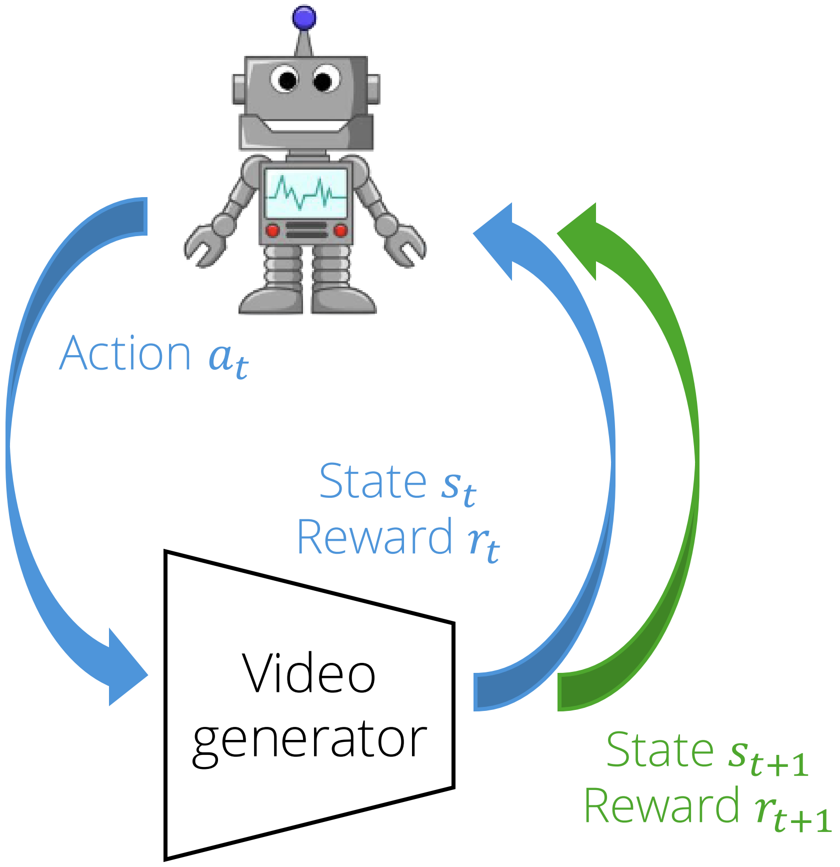
Training RL Policy with the Video-based Simulator

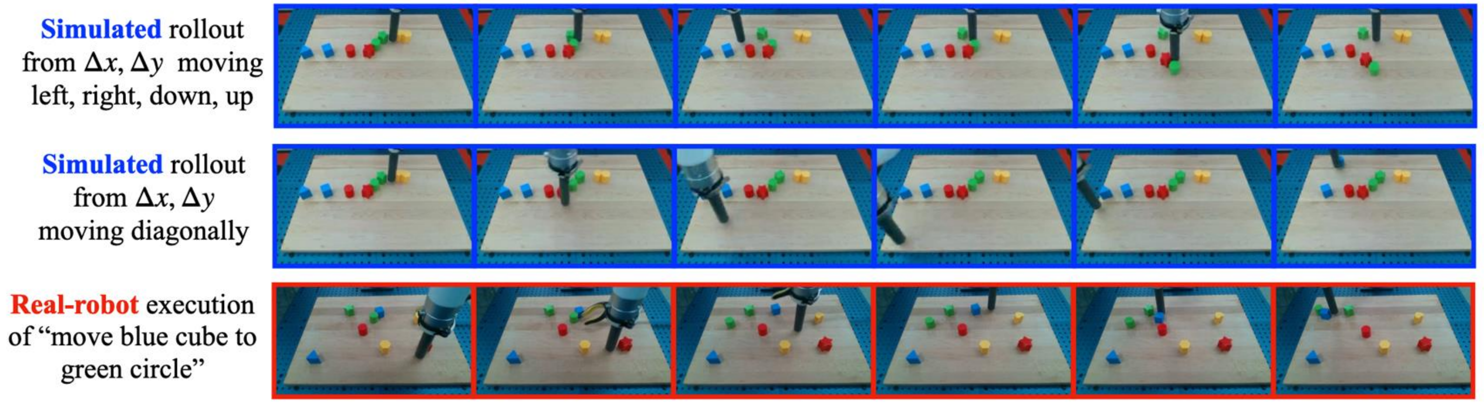



Simulated rollout from $\Delta x, \Delta y$ moving left, right, down, up
Simulated rollout from $\Delta x, \Delta y$ moving diagonally
Real-robot execution of "move blue cube to green circle"

- Stage 1:
  - Train UniSim on action-labeled Language Table dataset (and all the other datasets)
  - Train a policy π with behavior cloning
- Stage 2:
  - Clone the policy π to predict reward (if from timestep t to t + 1 the policy arrived closer to the desired goal)
  - Clone the policy π and fine-tune it with RL

Training RL Policy with the Video-based Simulator Works!




Simulated rollout from $\Delta x, \Delta y$ moving left, right, down, up
Simulated rollout from $\Delta x, \Delta y$ moving diagonally
Real-robot execution of "move blue cube to green circle"

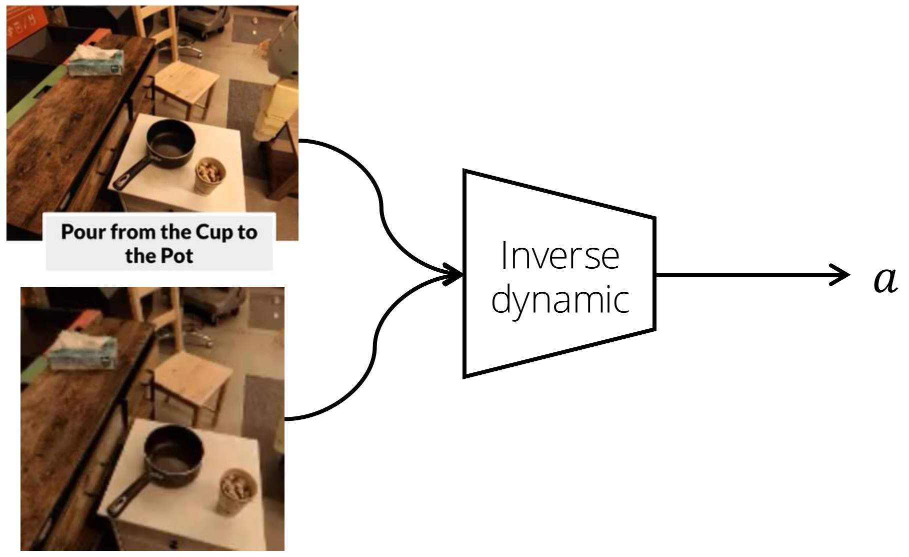
	Succ. rate (all)	Succ. rate (pointing)
VLA-BC	0.58	0.12
Simulator-RL	**0.81**	**0.71**

Table 3: **Evaluation of RL policy.** Percentage of successful simulated rollouts (out of 48 tasks) using the VLA policy with and without RL finetuning on Language Table (assessed qualitatively using video rollouts in the simulator). Simulator-RL improves the overall performance, especially in pointing-based tasks which contain limited expert demonstrations.

Wait, we still need large-scale robot dataset with action labels!

Can We Reduce the Need of Action Labels?

- Idea: Train an inverse dynamic model with a small-scale action-labeled dataset, and use the model to generate pseudo labels on the larger-scale unlabeled dataset



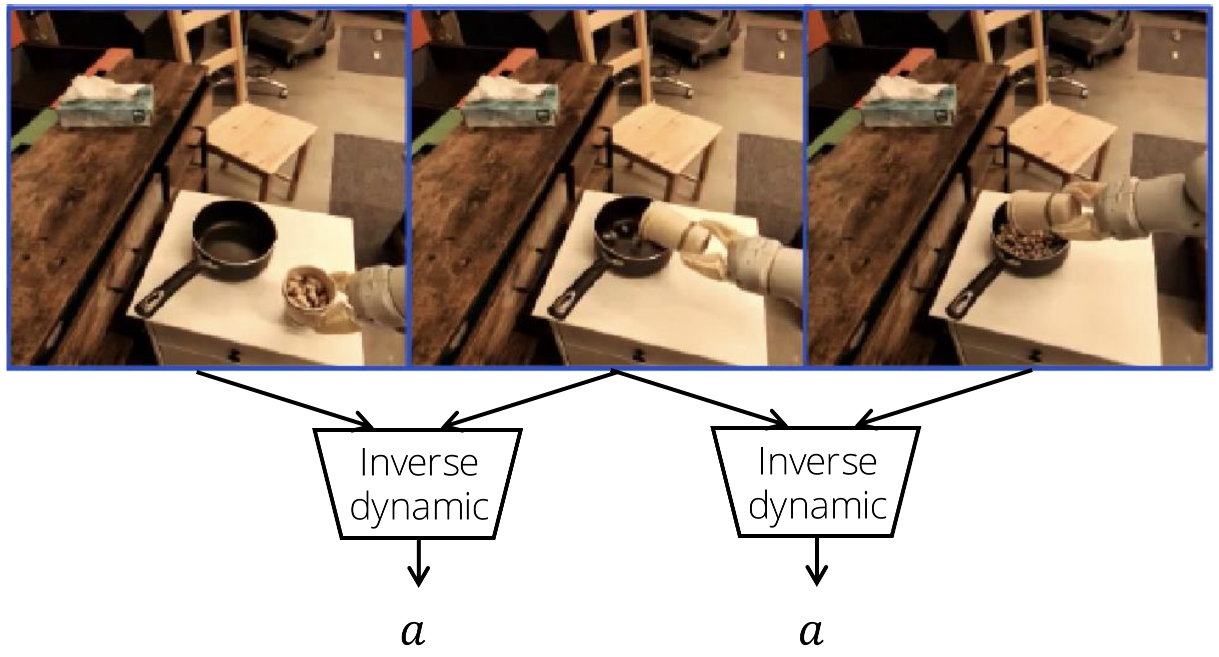
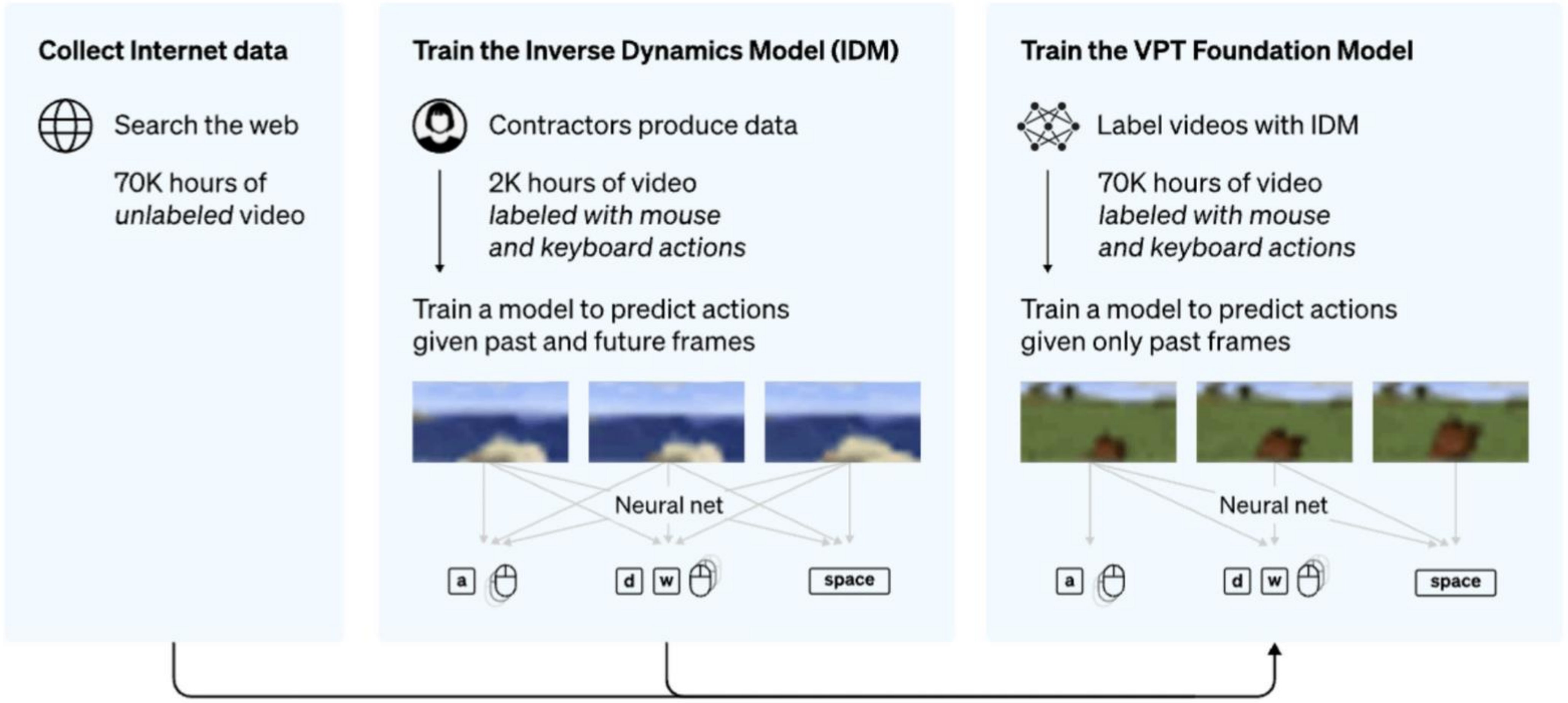


Image credit: Gen2Act: Human Video Generation in Novel Scenarios enables Generalizable Robot Manipulation

Learning to Act by Watching Unlabeled Online Videos

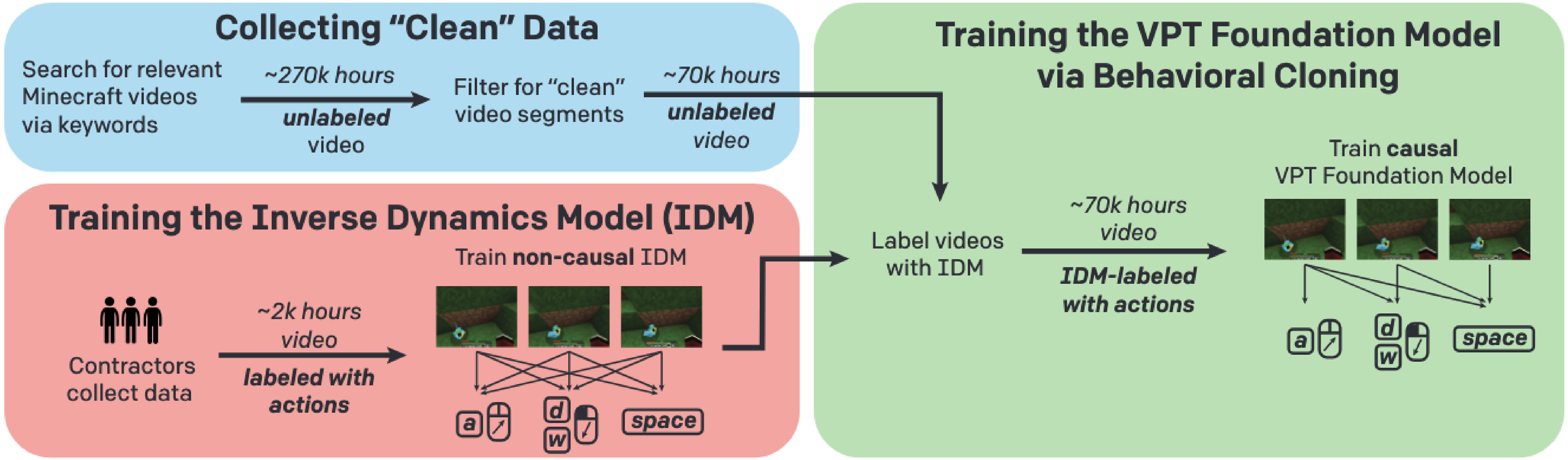
- Goal: train a policy in Minecraft from enormous actionless videos on the Internet


Collect Internet data
Search the web
70K hours of *unlabeled* video
Train the Inverse Dynamics Model (IDM)
Contractors produce data
2K hours of video *labeled with mouse and keyboard actions*
Train a model to predict actions given past and future frames
Neural net
a
d
w
space
Train the VPT Foundation Model
Label videos with IDM
70K hours of video *labeled with mouse and keyboard actions*
Train a model to predict actions given only past frames
Neural net
a
d
w
space

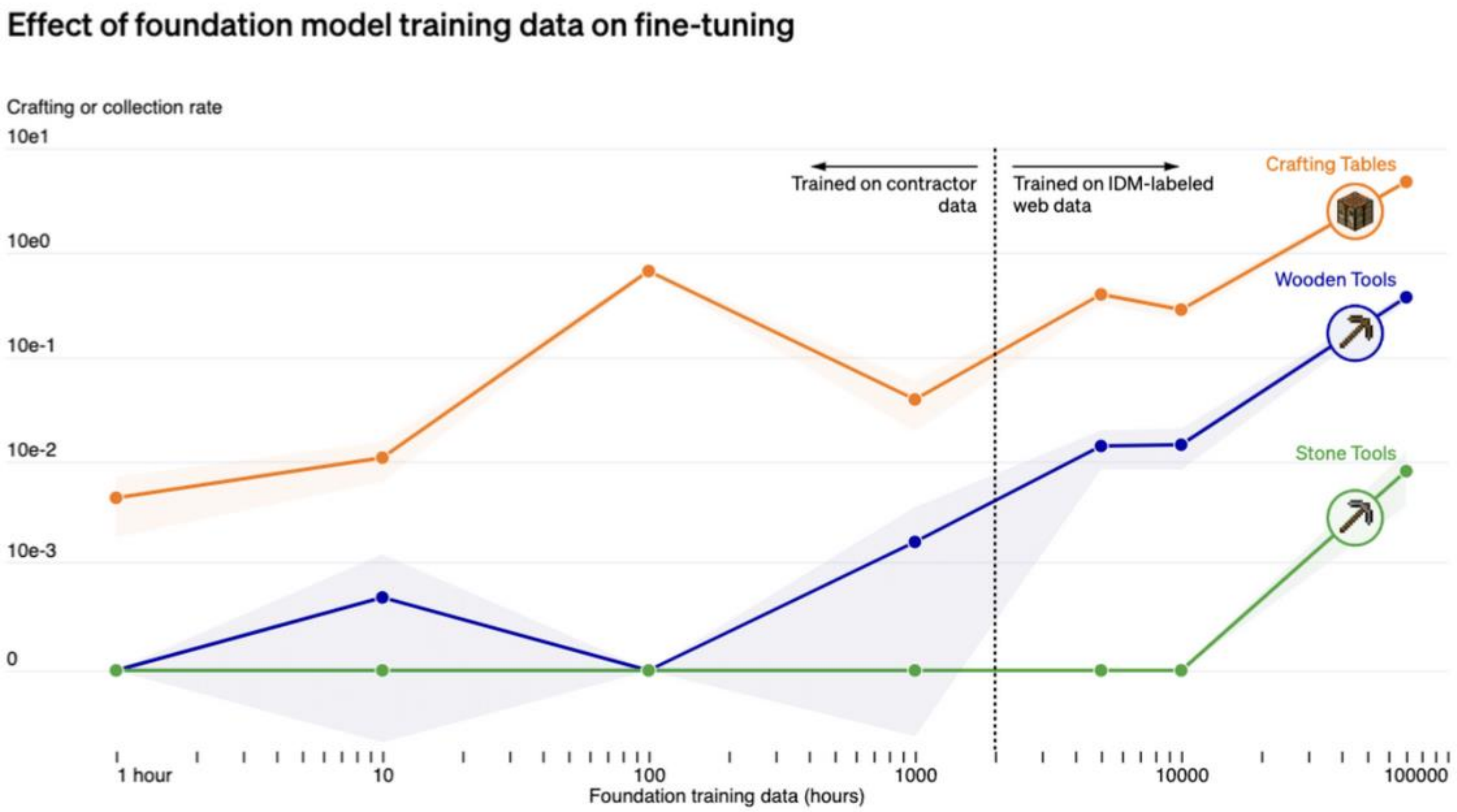
Video PreTraining (VPT): Learning to Act by Watching Unlabeled Online Videos. Open AI

Learning to Act by Watching Unlabeled Online Videos

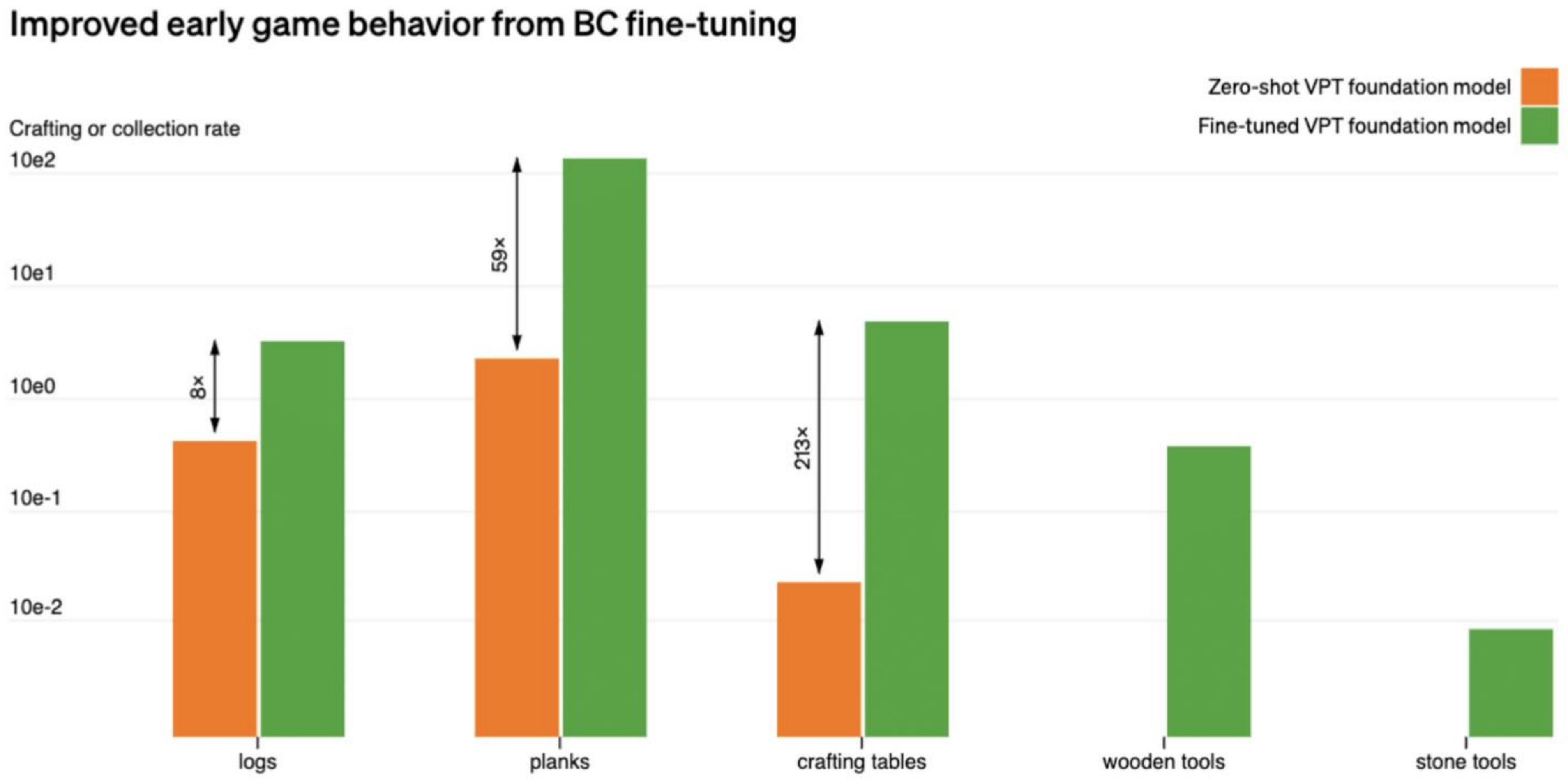
- A model that can use both the past and future of an action to infer what action was taken, is more accurate than a policy model that maps the past and present to the action that needs to be taken


Collecting "Clean" Data
Search for relevant Minecraft videos via keywords
~270k hours unlabeled video
Filter for "clean" video segments
~70k hours unlabeled video
Training the Inverse Dynamics Model (IDM)
Contractors collect data
~2k hours video labeled with actions
Train non-causal IDM
a
d
w
space
Training the VPT Foundation Model via Behavioral Cloning
Label videos with IDM
~70k hours video IDM-labeled with actions
Train causal VPT Foundation Model
a
d
w
space

Scaling Law with Large-scale Unlabeled Videos



Bootstrap the Model by Fine-tuning on Clean Dataset

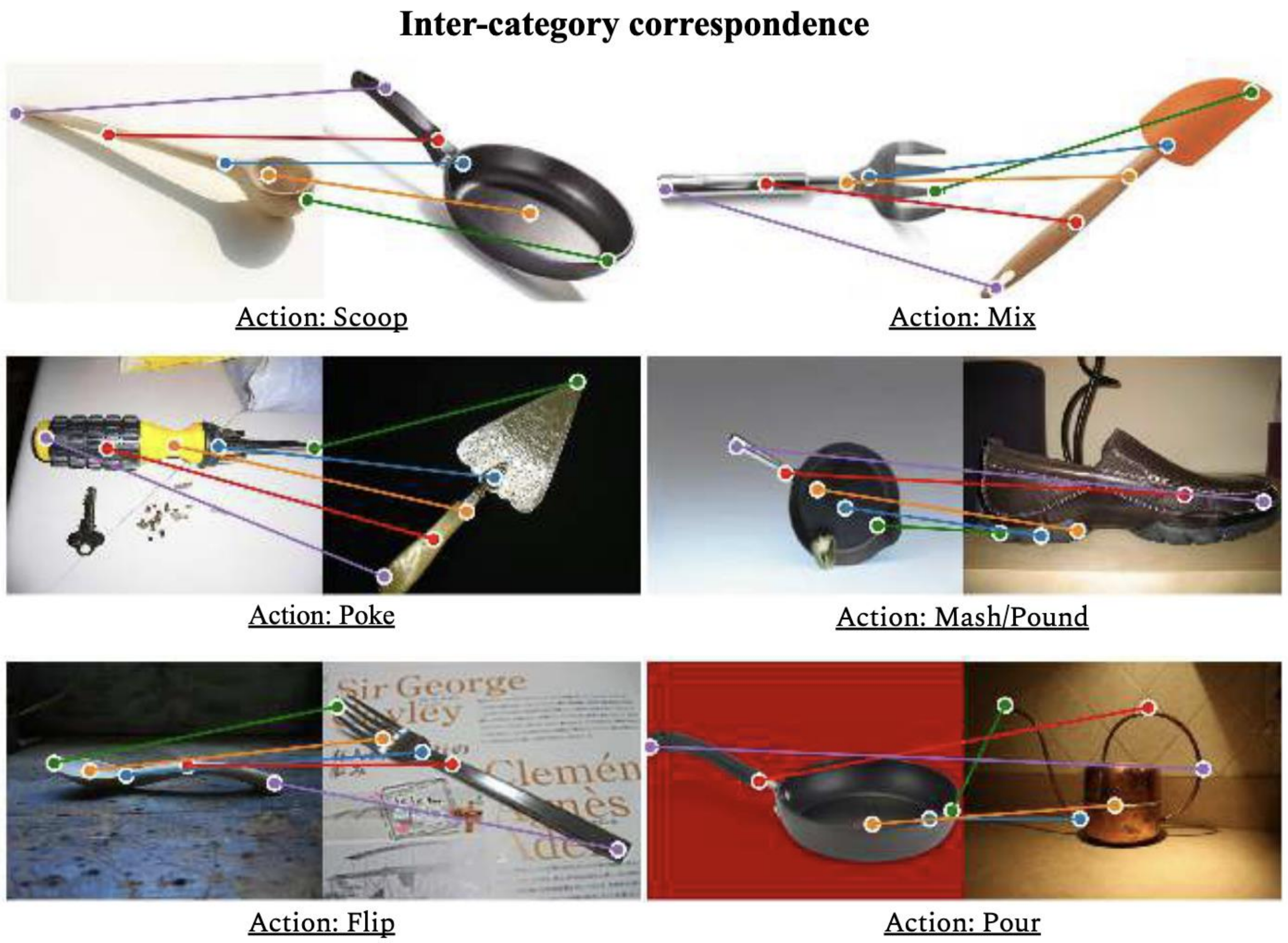


What information should we extract from human demonstration videos?

- Types of visual imitation learning:
  1. Knowledge distillation via representation learning
  2. Knowledge distillation via video generation
  3. Knowledge distillation with correspondence
  4. Knowledge distillation with affordance
  5. Knowledge distillation with 3D hand modeling
  6. Knowledge distillation with digital twins

Knowledge distillation with Correspondence: Finding "Where" and "How" to Interact with an Object

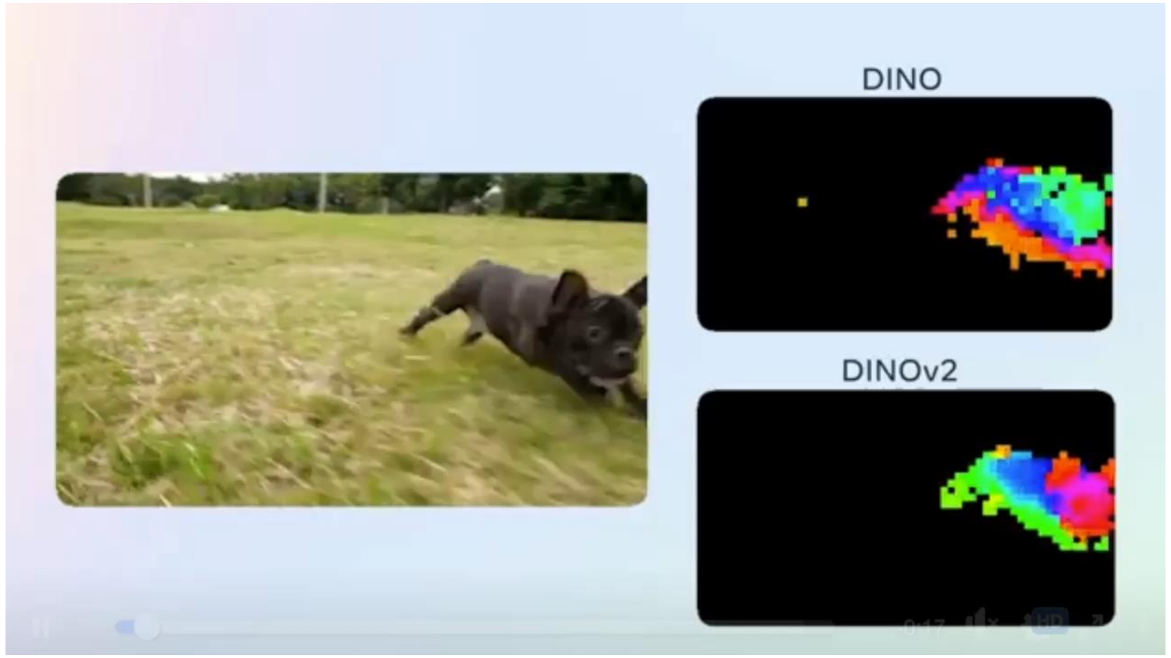
https://www.youtube.com/watch?v=F9Ow8yW0c3E



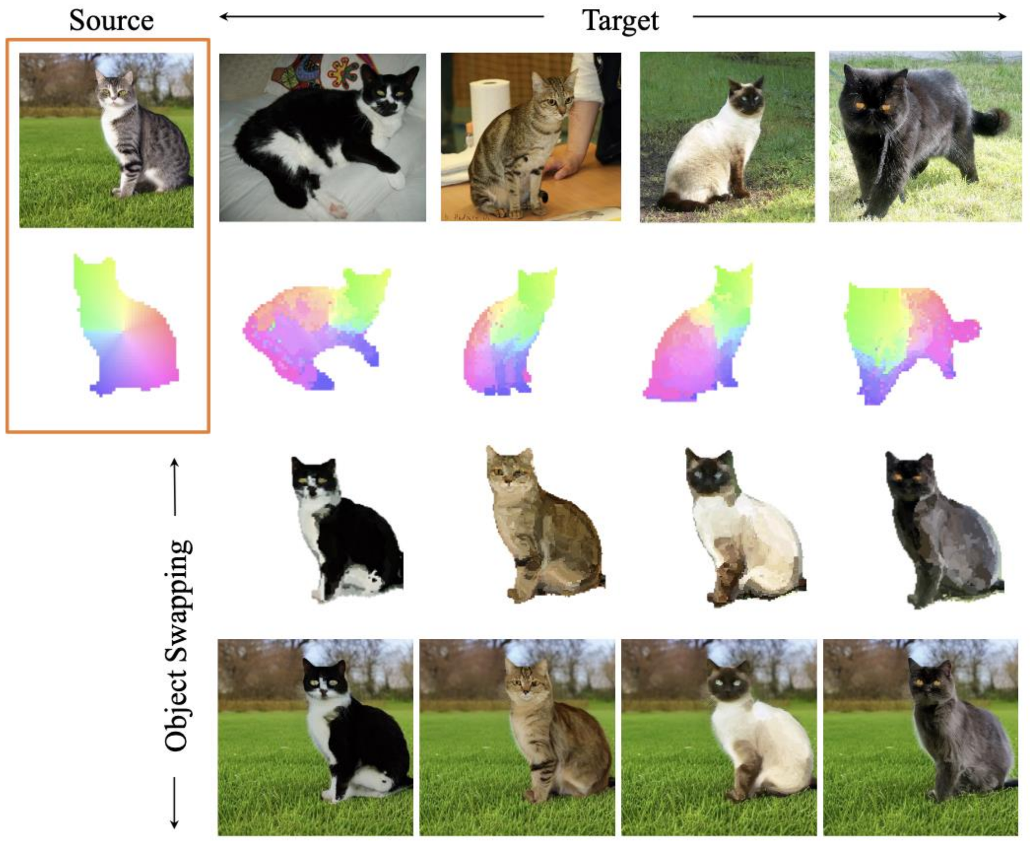
The Functional Correspondence Problem. Lai et al.

How to Obtain Correspondence? Use Existing Vision Foundation Models!

Correspondence from DINO.v2



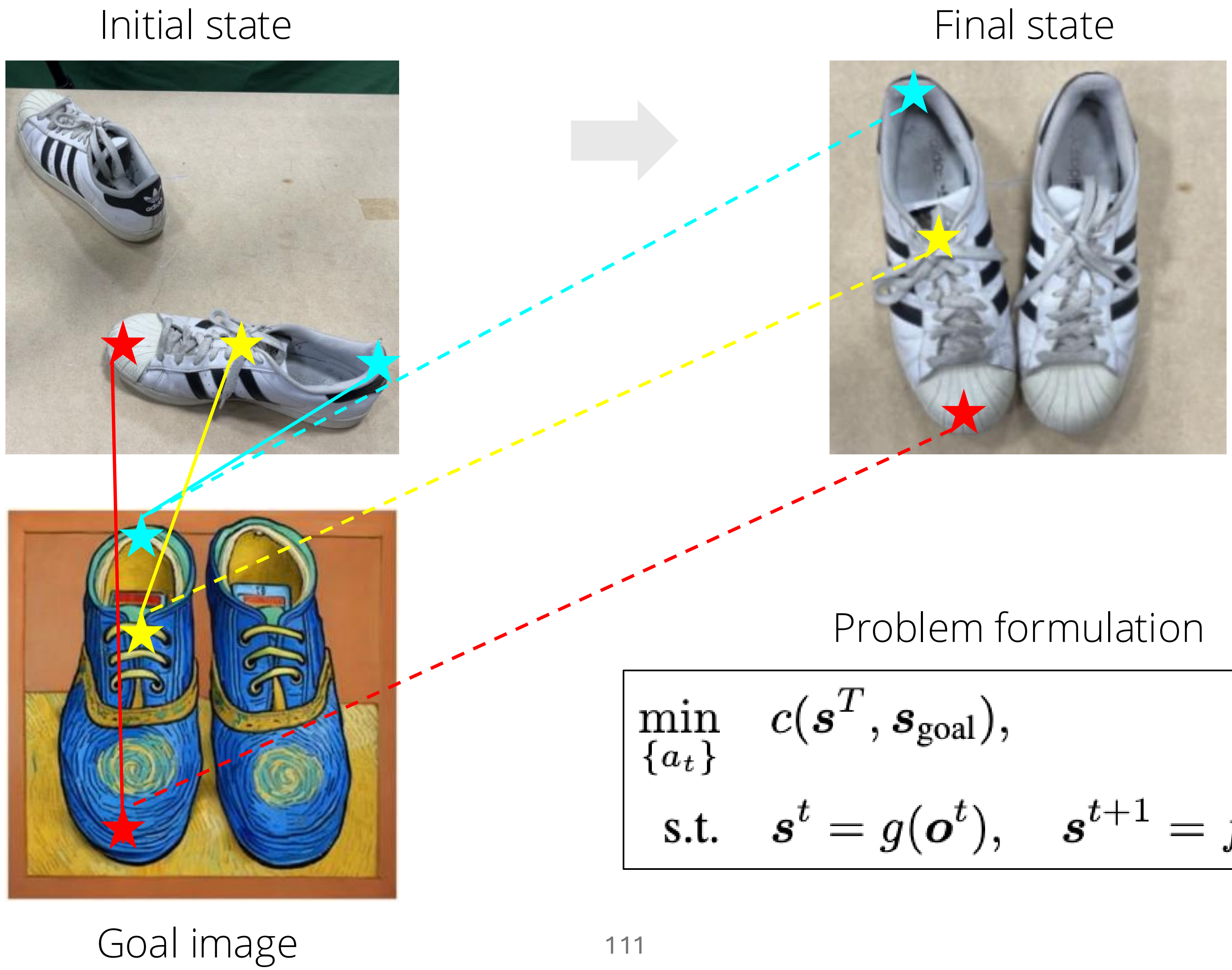
Correspondence from DINO.v2 and Stable Diffusion



DINOv2: Learning Robust Visual Features without Supervision. Oquab et al.

A Tale of Two Features: Stable Diffusion Complements DINO for Zero-Shot Semantic Correspondence. Zhang et al.

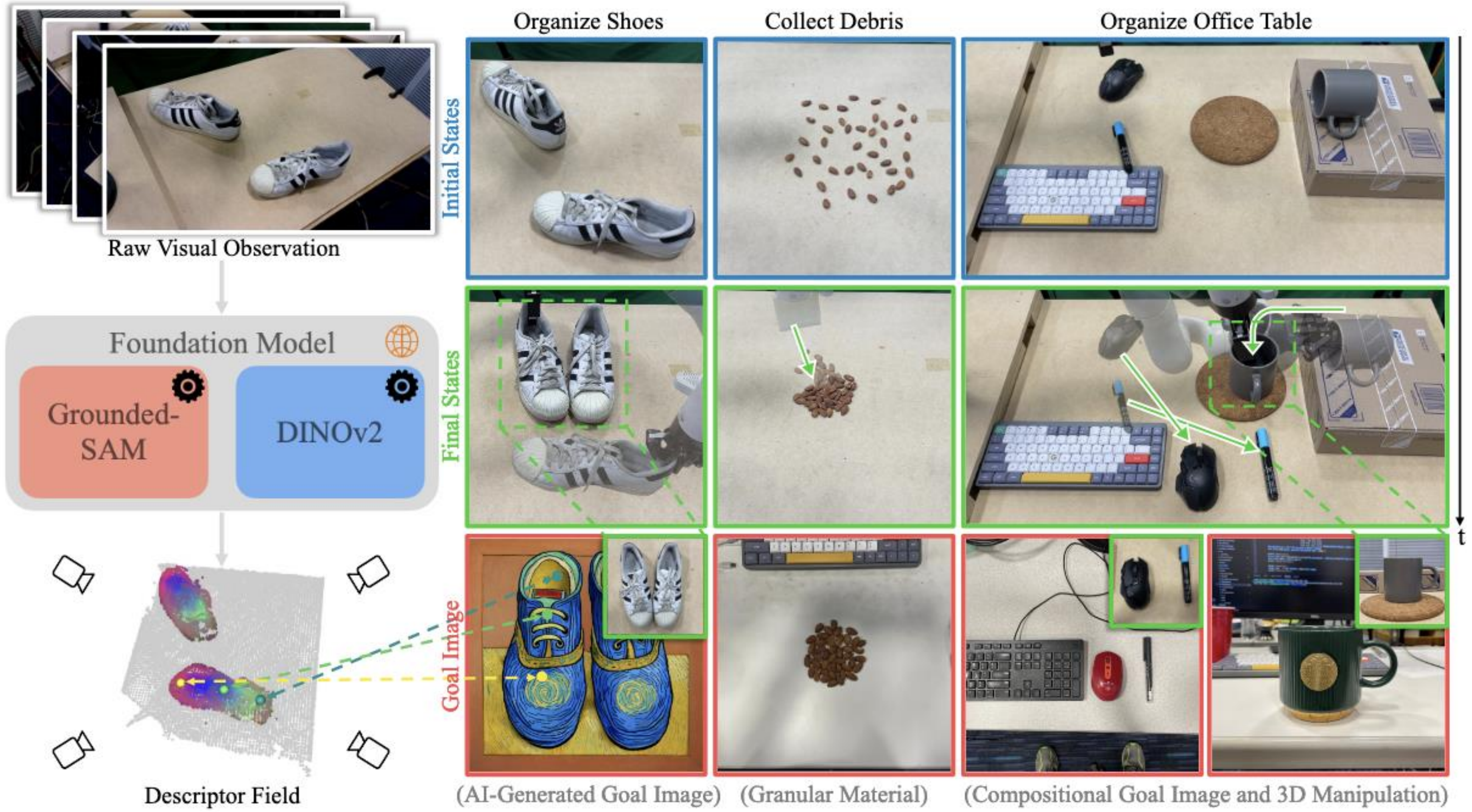
Define the Goal with Correspondence



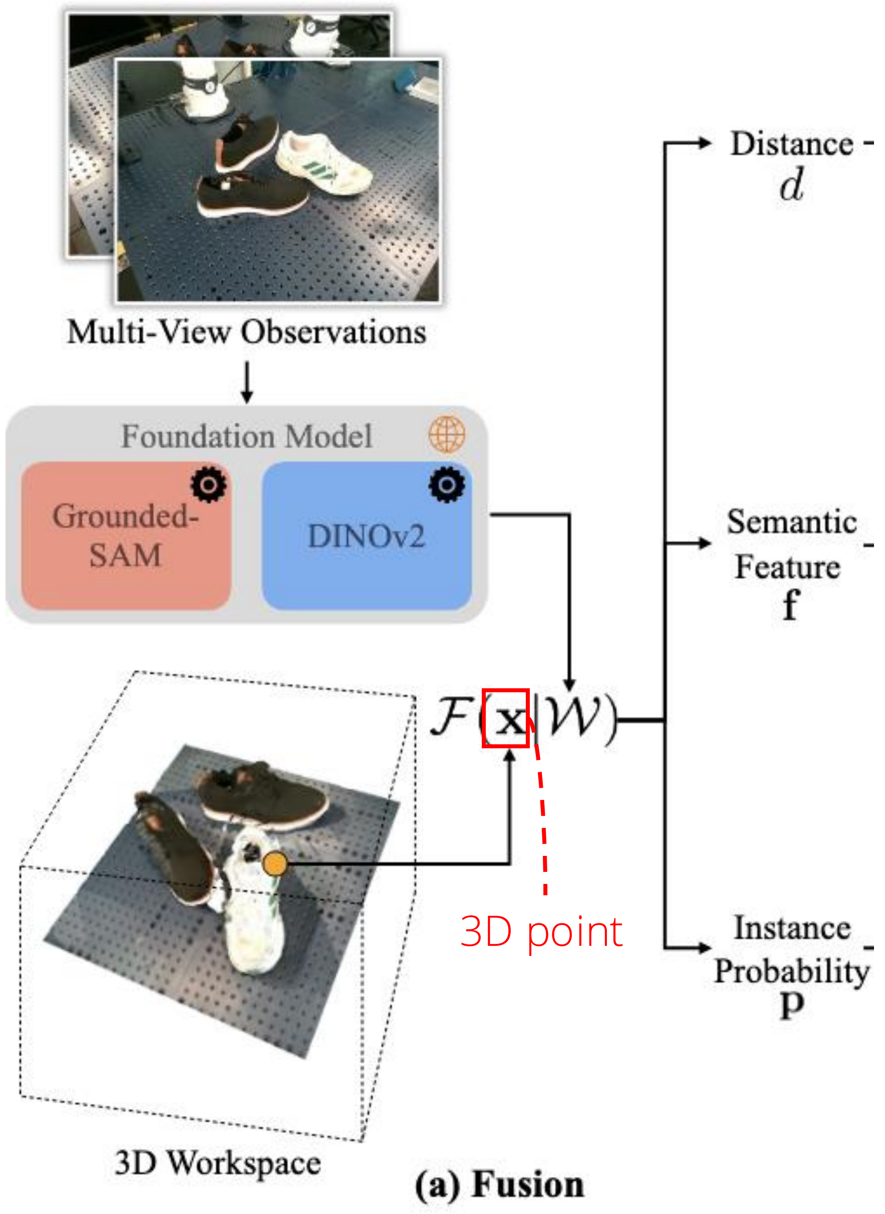
Problem formulation

$$\begin{aligned} \min_{\{a_t\}} \quad & c(\boldsymbol{s}^T, \boldsymbol{s}_{\text{goal}}), \\ \text{s.t.} \quad & \boldsymbol{s}^t = g(\boldsymbol{o}^t), \quad \boldsymbol{s}^{t+1} = f(\boldsymbol{s}^t, a^t), \end{aligned}$$

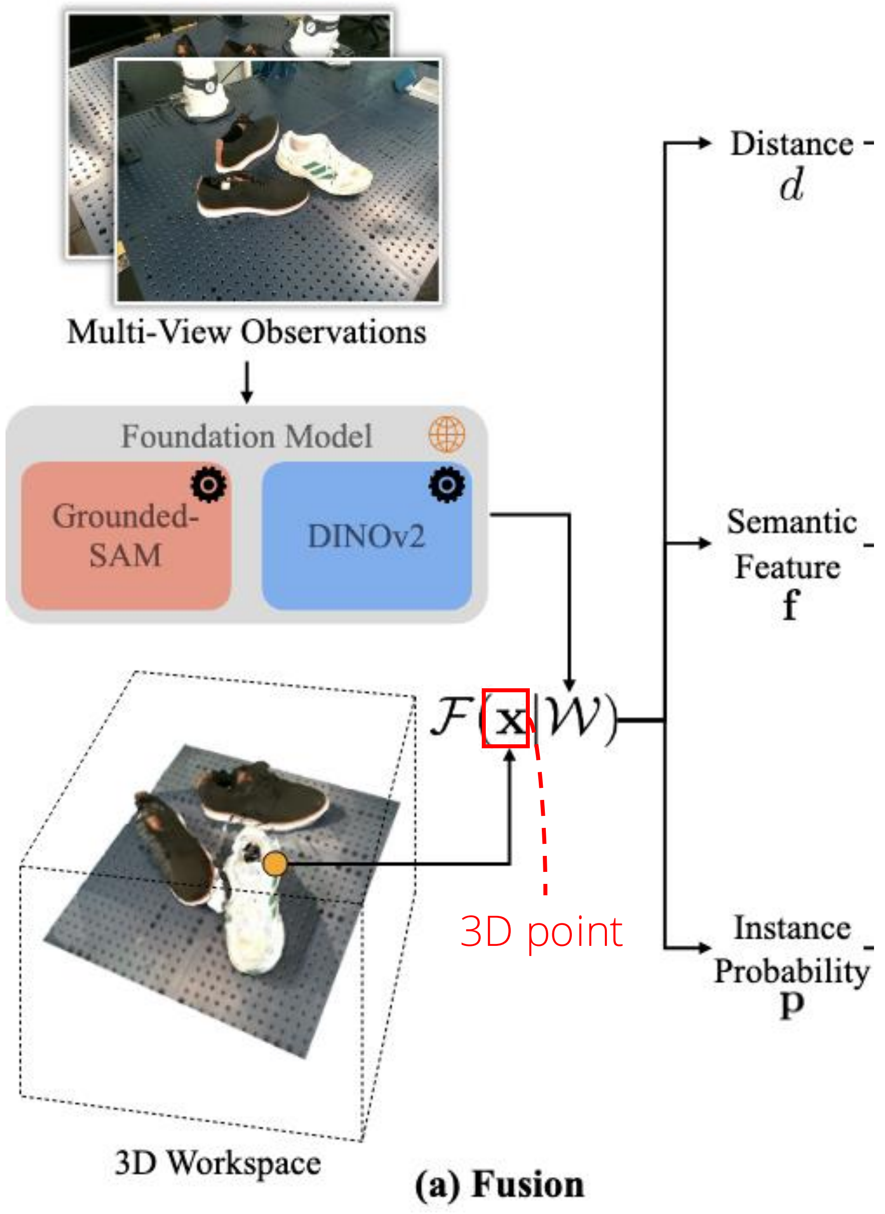
Building a 3D Feature Field for Correspondence



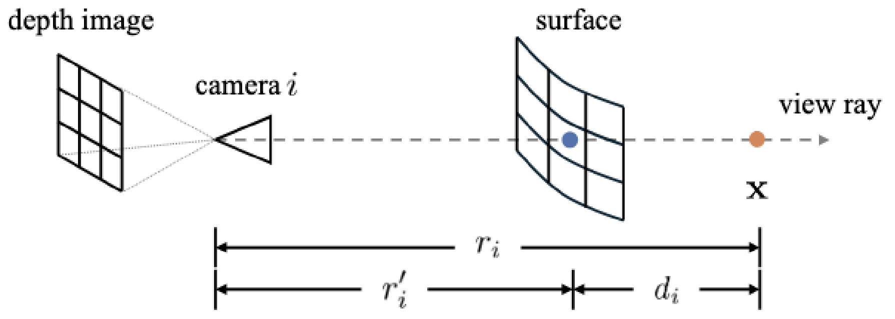
D^{3}Fields: Dynamic 3D Descriptor Fields for Zero-Shot Generalizable Rearrangement. Wang et al.


Multi-View Observations
Foundation Model
Grounded-SAM
DINOv2
$\mathcal{F}(\mathbf{x}|\mathcal{W})$
3D point
Distance d
Semantic Feature $\mathbf{f}$
Instance Probability $\mathbf{p}$
3D Workspace
(a) Fusion

- Goal: Build a 3D feature field from multiple camera views
  1. Extract 2D feature maps and instance masks from each camera view
  2. Retrieve feature, instance probability and distance to the object surface for a 3D point from each camera view
     a. To obtain distance, project 3D point x to 2D point u_i and compare to the sensed depth

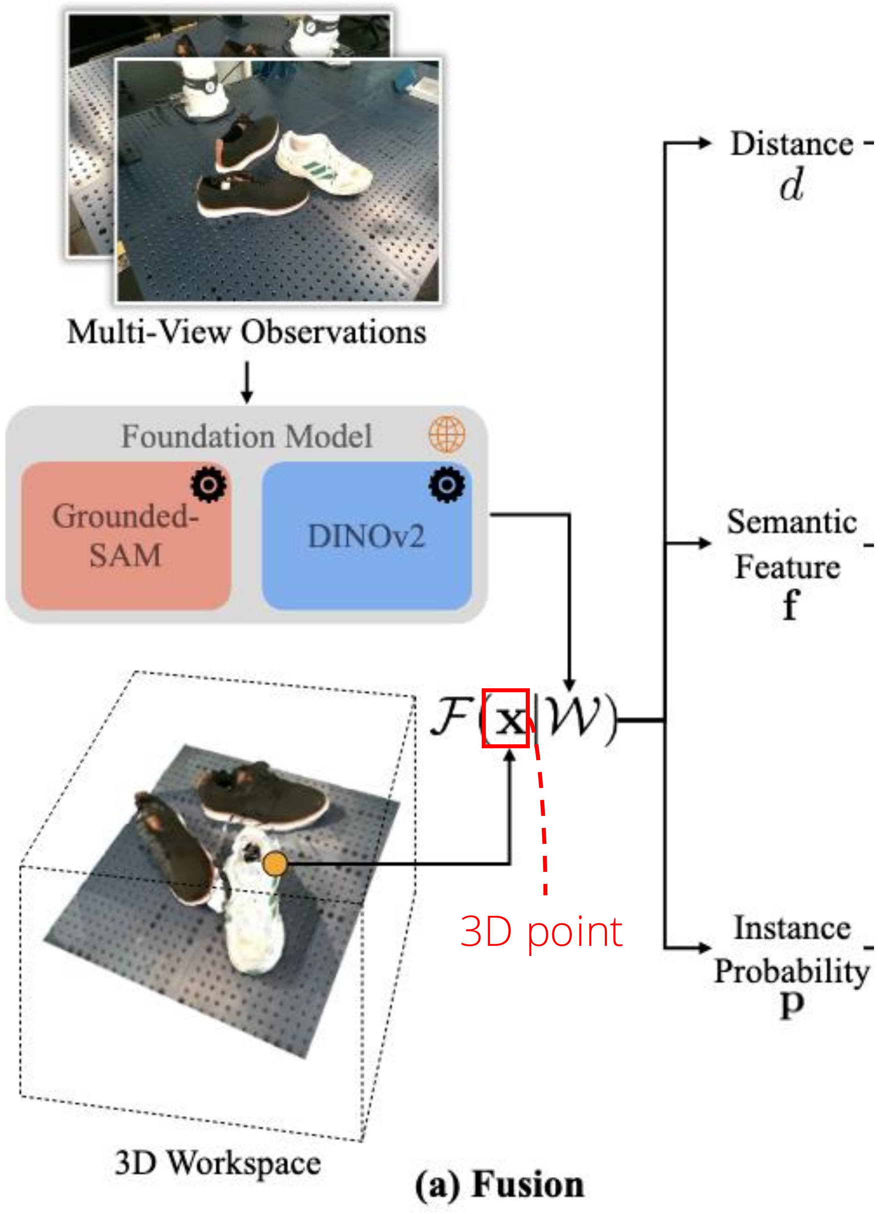

Multi-View Observations
Foundation Model
Grounded-SAM
DINOv2
$\mathcal{F}(\mathbf{x}|\mathcal{W})$
3D point
3D Workspace
Distance d
Semantic Feature $\mathbf{f}$
Instance Probability $\mathbf{p}$
(a) Fusion

- Goal: Build a 3D feature field from multiple camera views
  1. Extract 2D feature maps and instance masks from each camera view
  2. Retrieve feature, instance probability and distance to the object surface for a 3D point from each camera view
     a. To obtain distance, project 3D point x to 2D point u_i and compare to the sensed depth r_i'



$$d_i = r_i - r_i',$$

- Let μ be the threshold, we define:
  - ➢ Visibility $v_i = \mathbb{I}_{d_i<\mu}$
  - ➢ Weight $w_i = \exp(\frac{\min(\mu-|d_i|,0)}{\mu})$

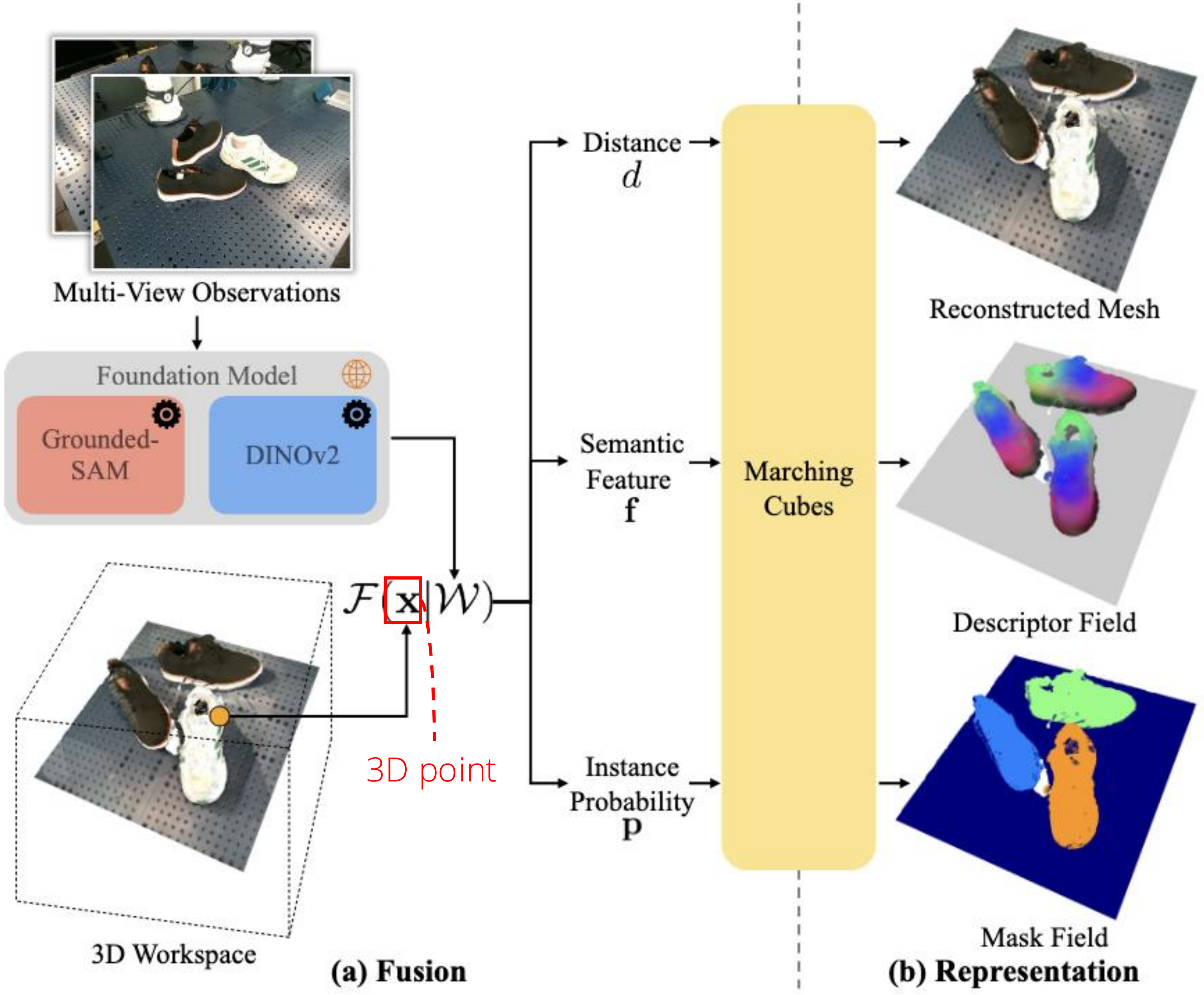

Multi-View Observations
Foundation Model
Grounded-SAM
DINOv2
$\mathcal{F}(\mathbf{x}|\mathcal{W})$
3D point
Distance d
Semantic Feature $\mathbf{f}$
Instance Probability $\mathbf{p}$
3D Workspace
(a) Fusion

- Goal: Build a 3D feature field from multiple camera views
  1. Extract 2D feature maps and instance masks from each camera view
  2. Retrieve feature, instance probability and distance to the object surface for a 3D point from each camera view
     a. To obtain distance, project 3D point x to 2D point u_i and compare to the sensed depth r_i'
     b. Retrieve the 2D feature map at u_i

$$\mathbf{f}_i = \mathcal{W}_i^{\mathbf{f}}[\mathbf{u}_i].$$

     c. Retrieve the instance mask at u_i

$$\mathbf{p}_i = \mathcal{W}_i^{\mathbf{p}}[\mathbf{u}_i]$$



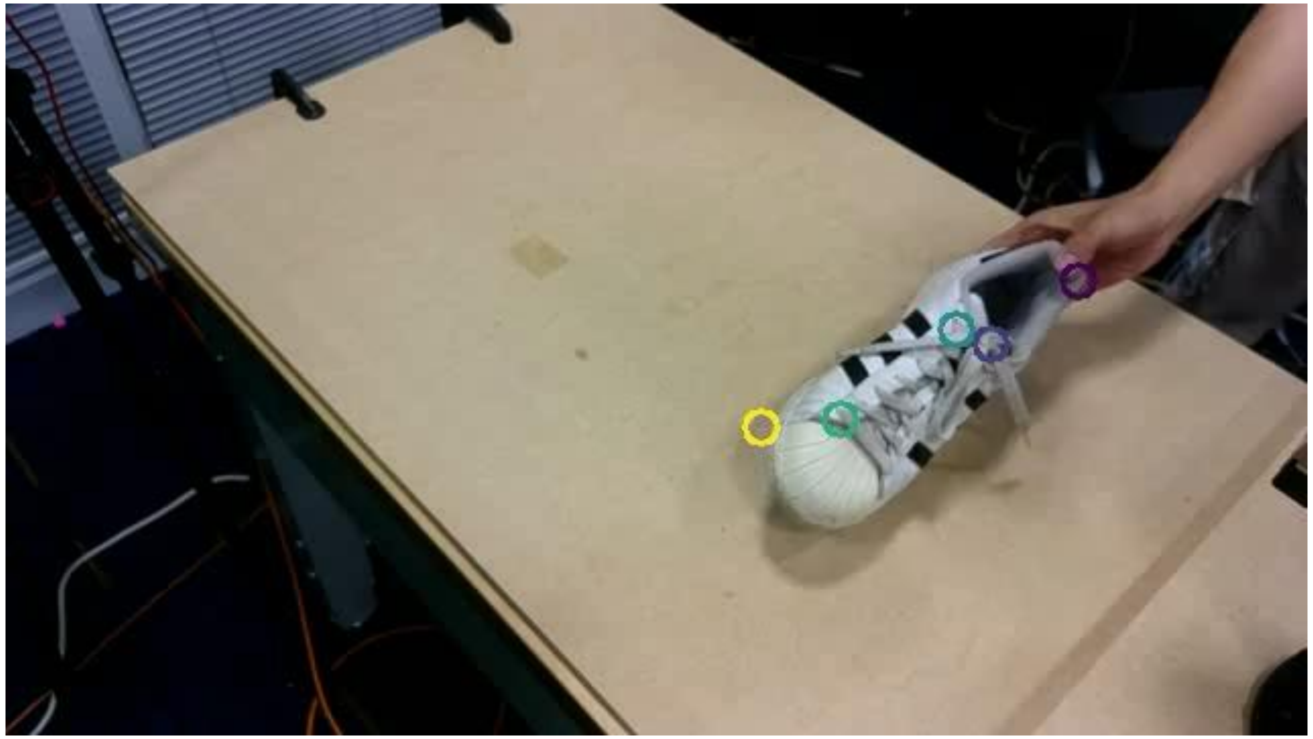
Aggregate across camera views

$$\boxed{d = \frac{\sum_{i=1}^{K} v_i d_i'}{\delta + \sum_{i=1}^{K} v_i}}$$

$$\boxed{\mathbf{f} = \frac{\sum_{i=1}^{K} v_i w_i \mathbf{f}_i}{\delta + \sum_{i=1}^{K} v_i}}$$

$$\boxed{\mathbf{p} = \frac{\sum_{i=1}^{K} v_i w_i \mathbf{p}_i}{\delta + \sum_{i=1}^{K} v_i}}$$

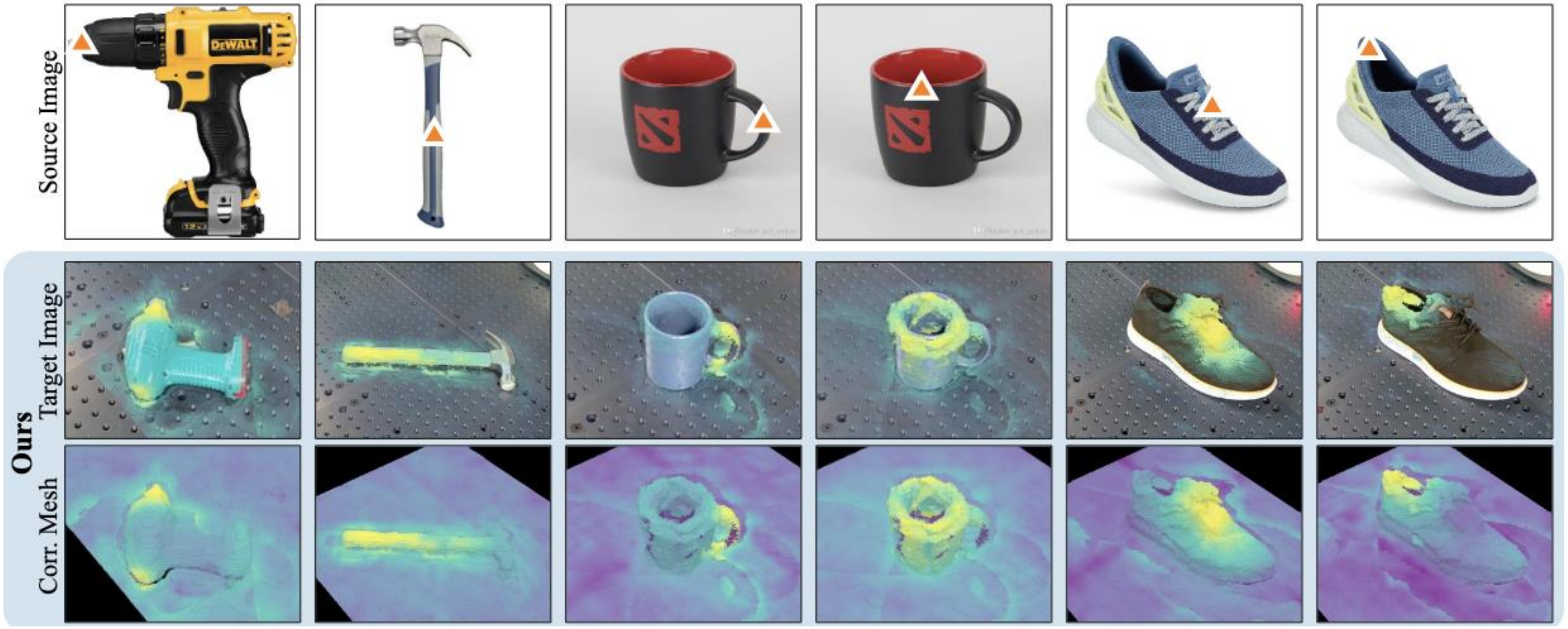
The 3D Feature Field Facilitate Tracking



- $\mathcal{F}_f(s^t|\mathcal{W})$ denotes the 3D feature field at time t

$$\min_{\boldsymbol{s}^{t+1}} \quad ||\mathcal{F}_{\mathbf{f}}(\boldsymbol{s}^{t+1}|\mathcal{W}) - \mathcal{F}_{\mathbf{f}}(\boldsymbol{s}^{0}|\mathcal{W})||_2.$$

The 3D Feature Field Facilitate Correspondence



D^{3}Fields: Dynamic 3D Descriptor Fields for Zero-Shot Generalizable Rearrangement. Wang et al.

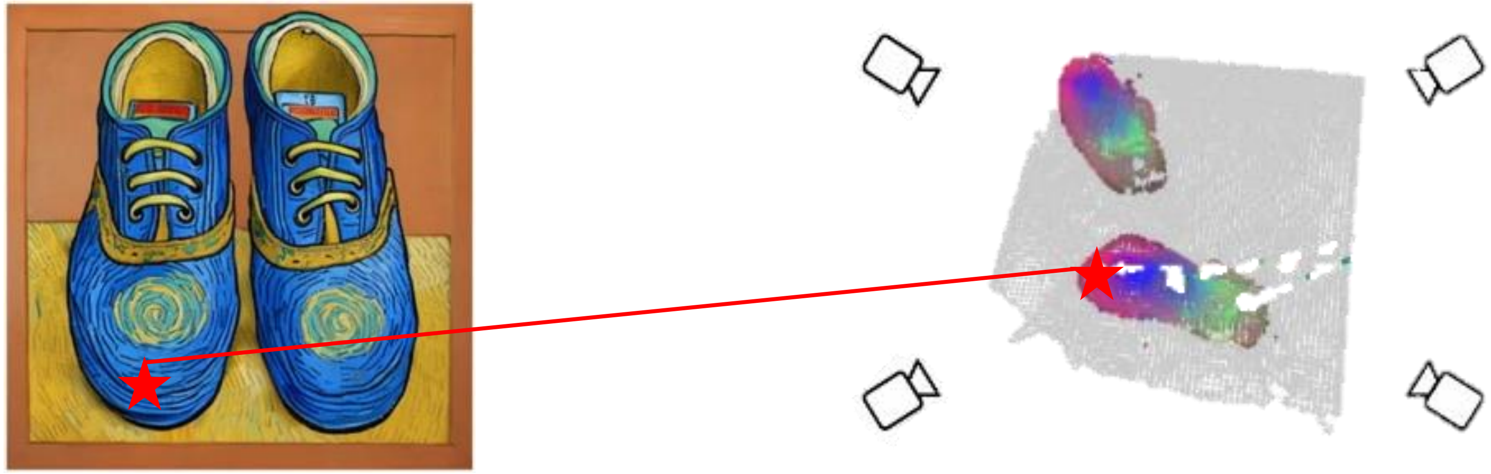
Predict Actions to Follow the Provided Goal

Initial state

1. Sample 3D points, and find their corresponding pixels in the goal image.

- Calculate feature similarity of each pixel in the goal image and obtain weighted average pixel location based on the similarity

$$\alpha_{ij} = ||\mathcal{W}^{\mathbf{f}}_{\text{goal}}[\mathbf{u}_i] - \mathbf{f}^0_j||_2, \quad \beta_{ij} = \frac{\exp(-s\alpha_{ij})}{\sum_{i=1}^{H\times W}\exp(-s\alpha_{ij})}, \quad \mathbf{s}_{\text{goal},j} = \sum_{i=1}^{H\times W}\beta_{ij}\mathbf{u}_i,$$



Goal image

Predict Actions to Follow the Provided Goal

Initial state

Goal image

1. Sample 3D points, and find their corresponding pixels in the goal image.
2. Sample multiple action trajectories, predict the outcome of each trajectory with a learned dynamic model (or MPC), and pick the best action trajectory

$$c(\boldsymbol{s}^t, \boldsymbol{s}_{\text{goal}}) = ||\boldsymbol{s}^t_{\text{2D}} - \boldsymbol{s}_{\text{goal}}||^2_2.$$

Predict Actions to Follow the Provided Goal

Initial state

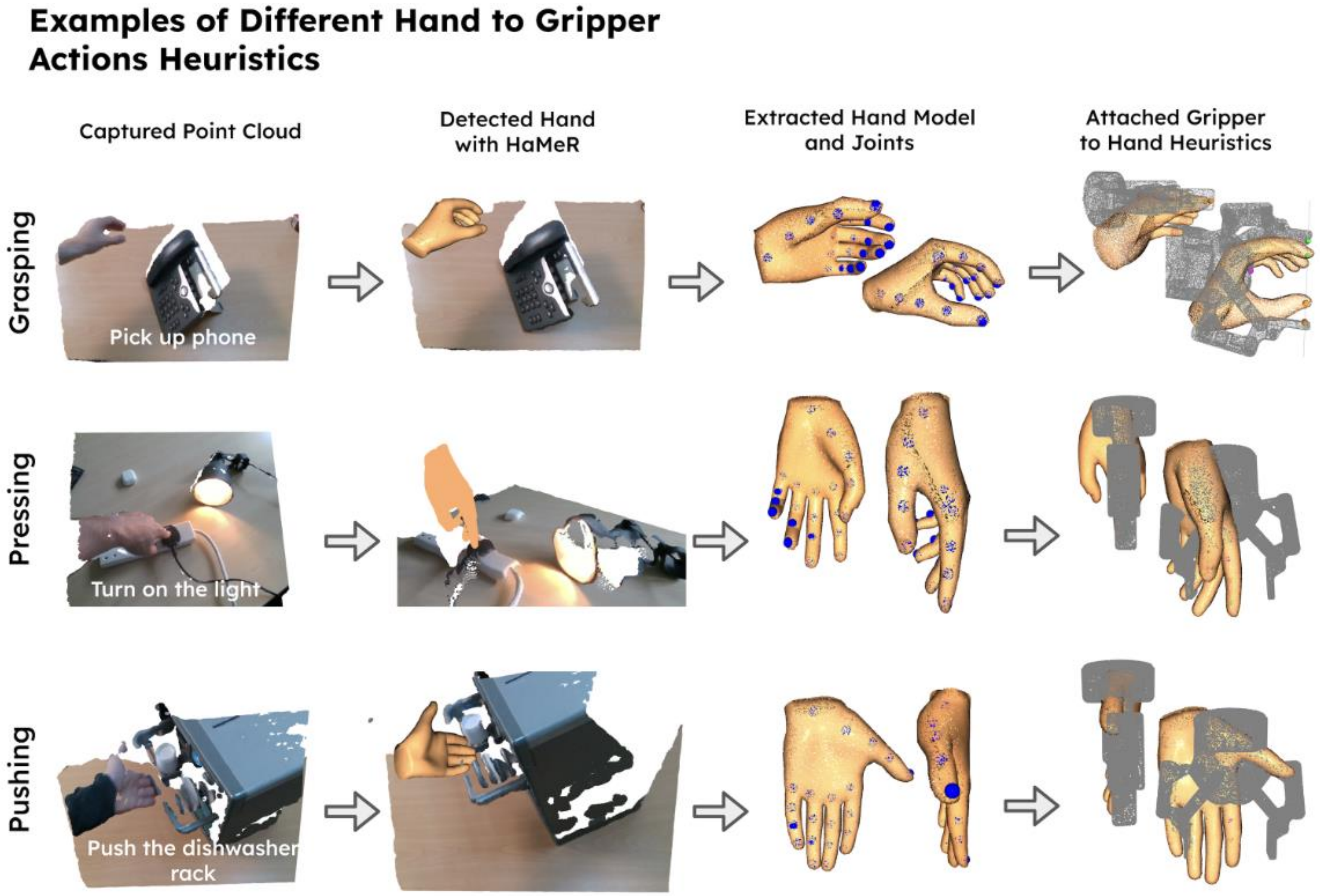
Goal image

1. Sample 3D points, and find their corresponding pixels in the goal image.
2. Sample multiple action trajectories, predict the outcome of each trajectory with a learned dynamic model (or MPC), and pic

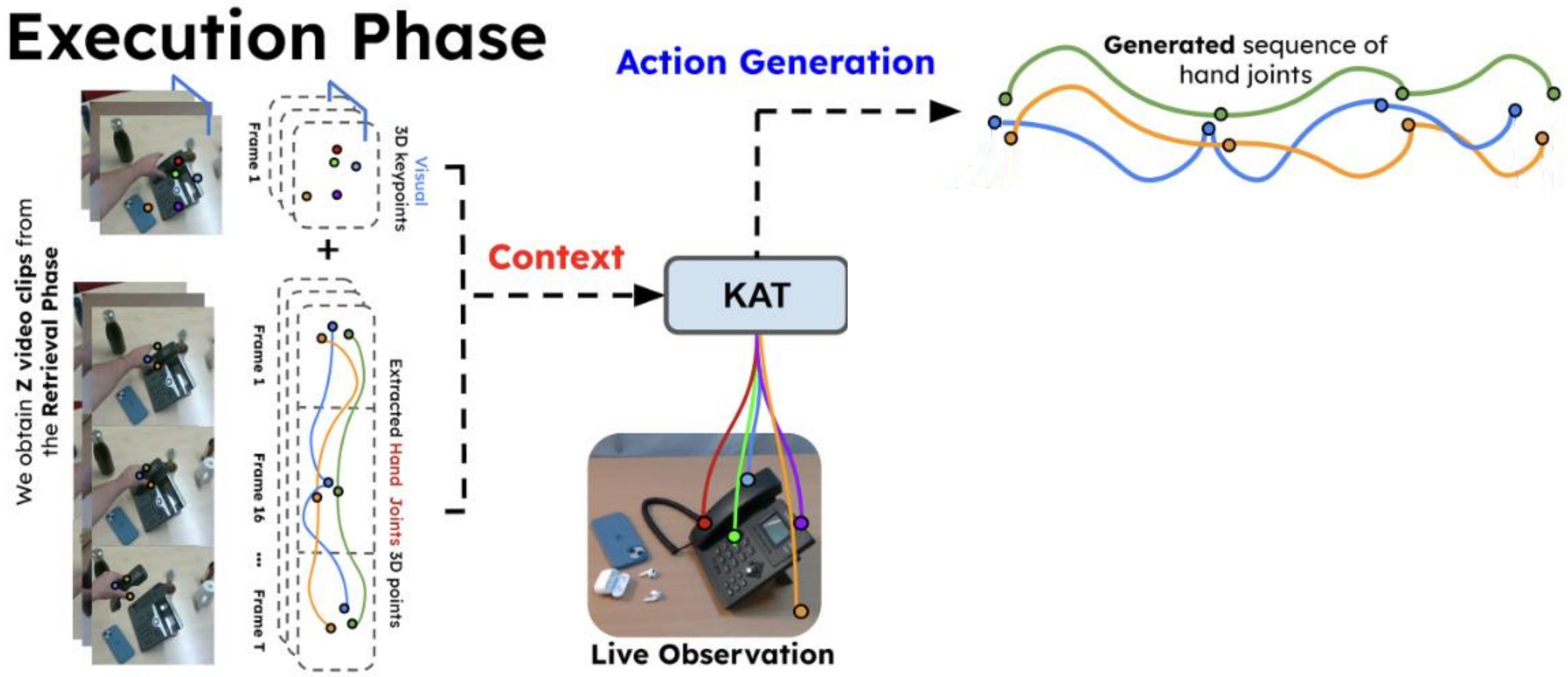
Again, this require action labels!

$$c(\boldsymbol{s}^t, \boldsymbol{s}_{\text{goal}}) = ||\boldsymbol{s}^t_{2\text{D}} - \boldsymbol{s}_{\text{goal}}||^2_2.$$

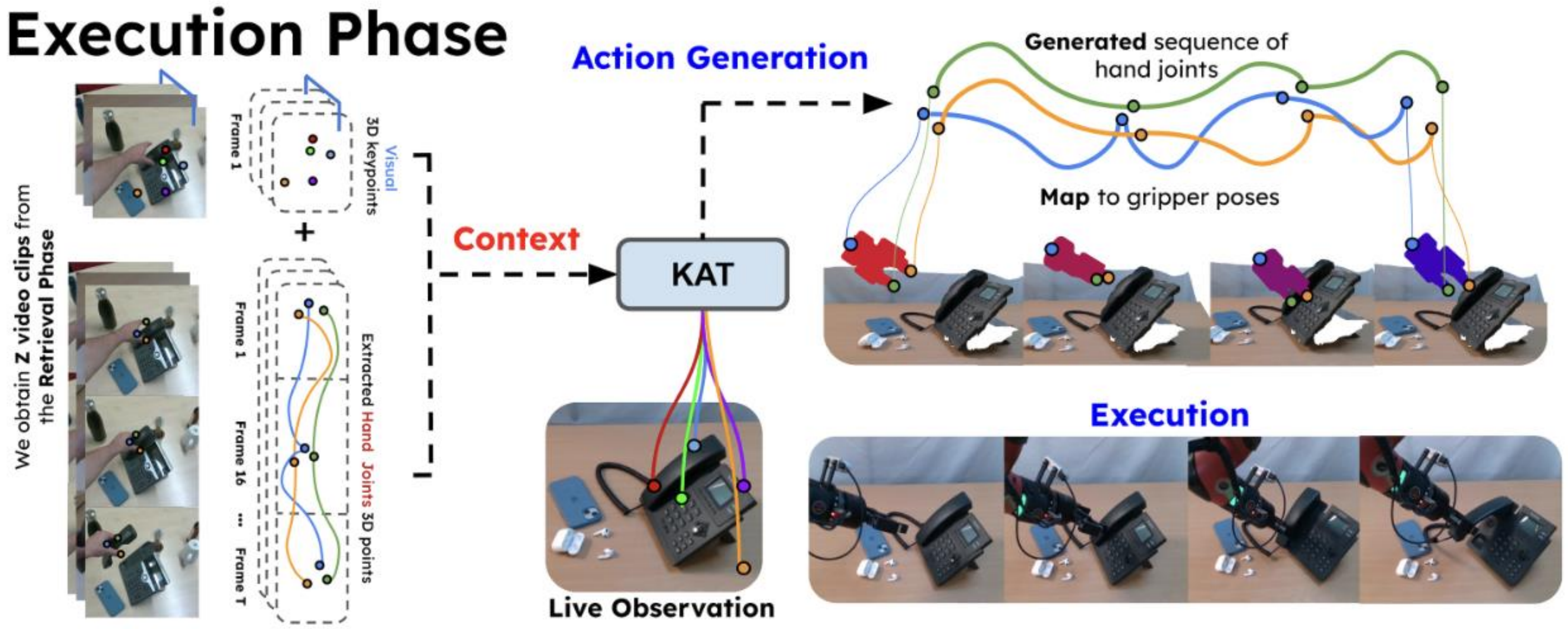
Can We Obtain Actions without Using Any Labels?


Examples of Different Hand to Gripper Actions Heuristics
Captured Point Cloud
Detected Hand with HaMeR
Extracted Hand Model and Joints
Attached Gripper to Hand Heuristics
Grasping
Pick up phone
Pressing
Turn on the light
Pushing
Push the dishwasher rack

Retrieve Human Demonstration and Infer 3D Hand Motion of the Current Scene


Execution Phase
We obtain Z video clips from the Retrieval Phase
Frame 1
Visual 3D keypoints
+
Frame 1
Frame 16
...
Frame T
Extracted Hand Joints 3D points
Context
KAT
Live Observation
Action Generation
Generated sequence of hand joints

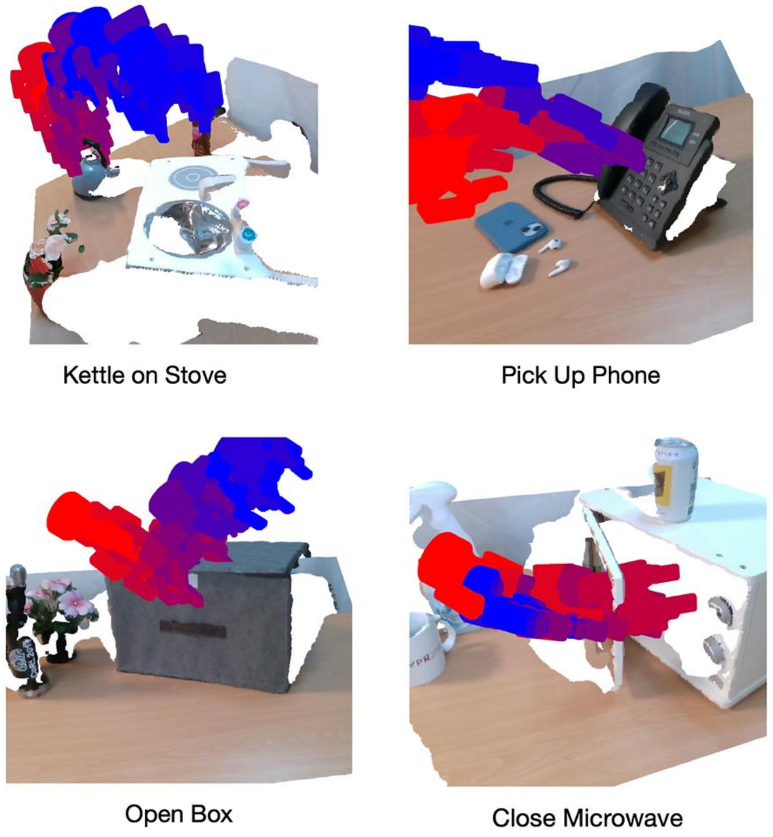
Map the Predicted 3D Hand Motion to Robot Action



Results: Few-Shot Imitation Learning without Action Labels

Human demonstrations

Robot execution



What information should we extract from human demonstration videos?

- Types of visual imitation learning:
  1. Knowledge distillation via representation learning
  2. Knowledge distillation via video generation
  3. Knowledge distillation with correspondence
  4. Knowledge distillation with affordance
  5. Knowledge distillation with point trajectories
  6. Knowledge distillation with 3D hand modeling
  7. Knowledge distillation with digital twins

Let's continue next week