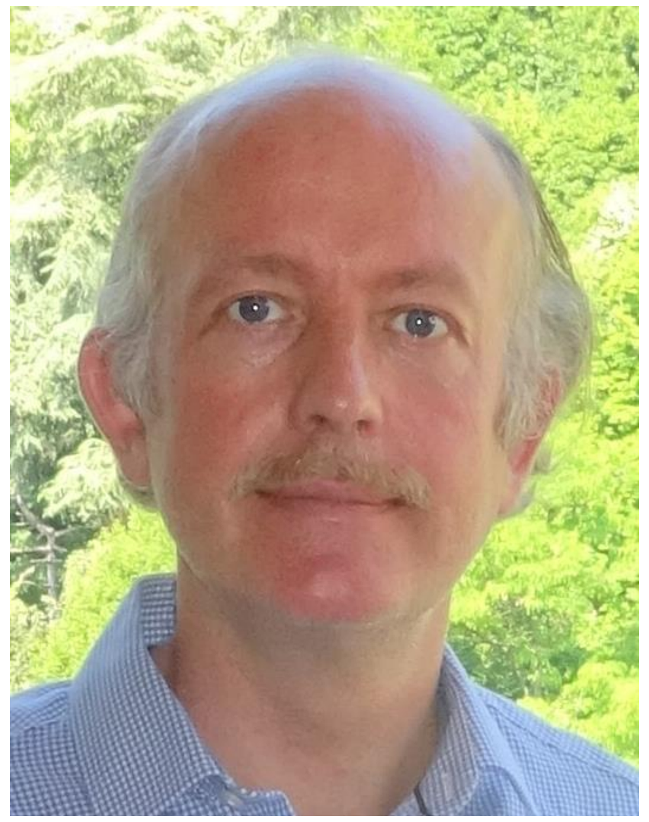
Efficient Imitation for Robotics

DEVIEW 2021

Theo Cachet, Christopher Dance, Julien Perez, NAVER LABS Europe

DEVIEW 2021





Théo Cachet Julien Perez Christopher Dance

Research at NAVER LABS Europe

Computer Vision

3D Vision

Machine Learning and Optimization

Systemic AI

Search and Recommendation

Natural Language Processing

UX and Ethnography

26

nationalities

Contents

1. Motivation
2. The few-shot imitation problem
3. Demonstration-conditioned reinforcement learning (DCRL)
4. Benchmarking few-shot imitation performance
5. Why it works and what’s next?

1. Robots & Diverse Tasks

1. How to get robots to perform diverse tasks?

DEVIEW 2021

Options

Classical planning and control → Manual choice of objectives and constraints
Uncertainty and partial observation
Planning through multiple contact modes

Multi-task reinforcement learning → Manual choice of reward function per task

Natural language → Interesting but needs data relating words to physical states

Imitation learning → Often requires too many demonstrations
Brittle if the state deviates from the demonstrated states

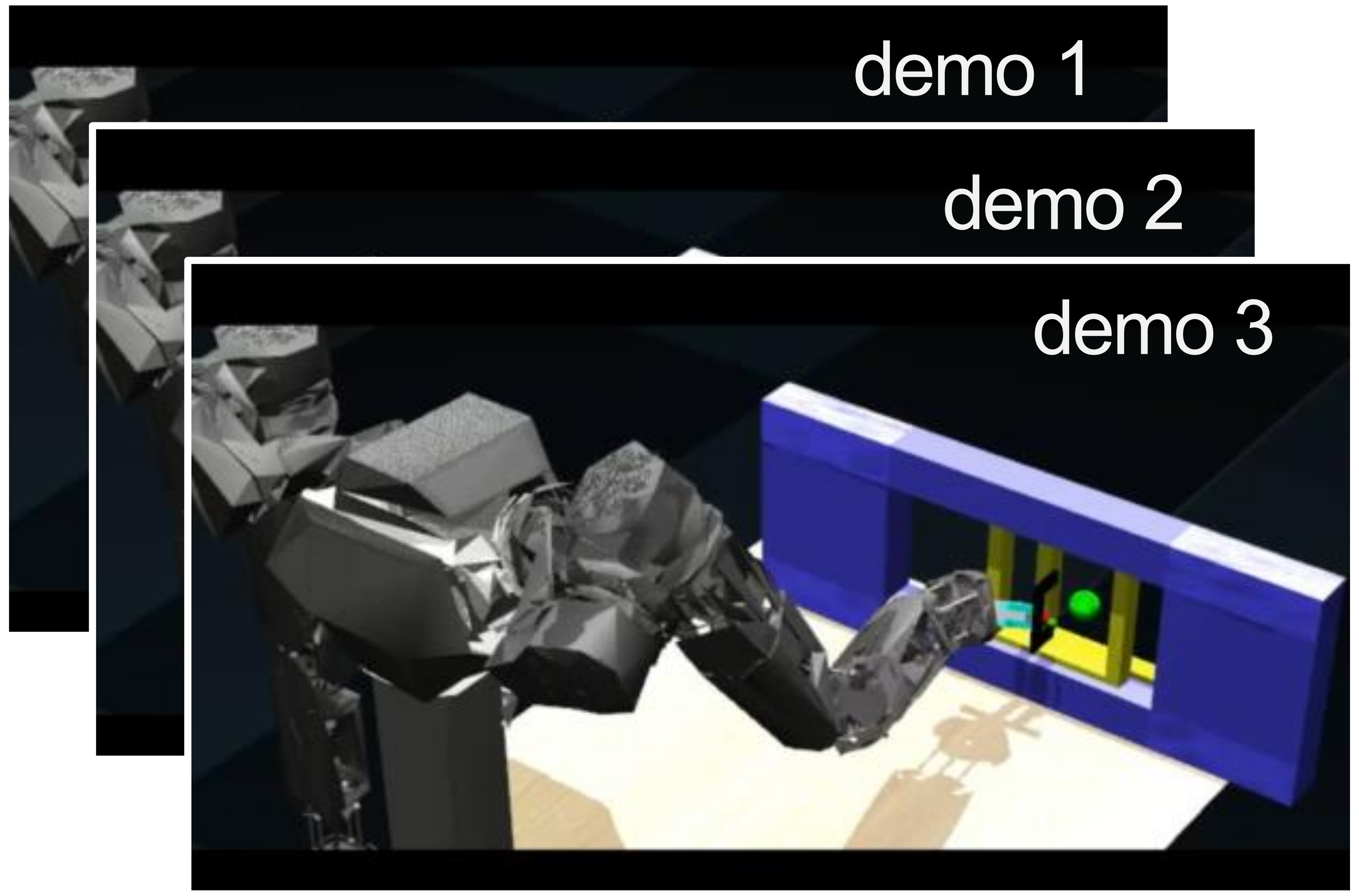
Few-shot imitation → This talk

2. Few-Shot Imitation

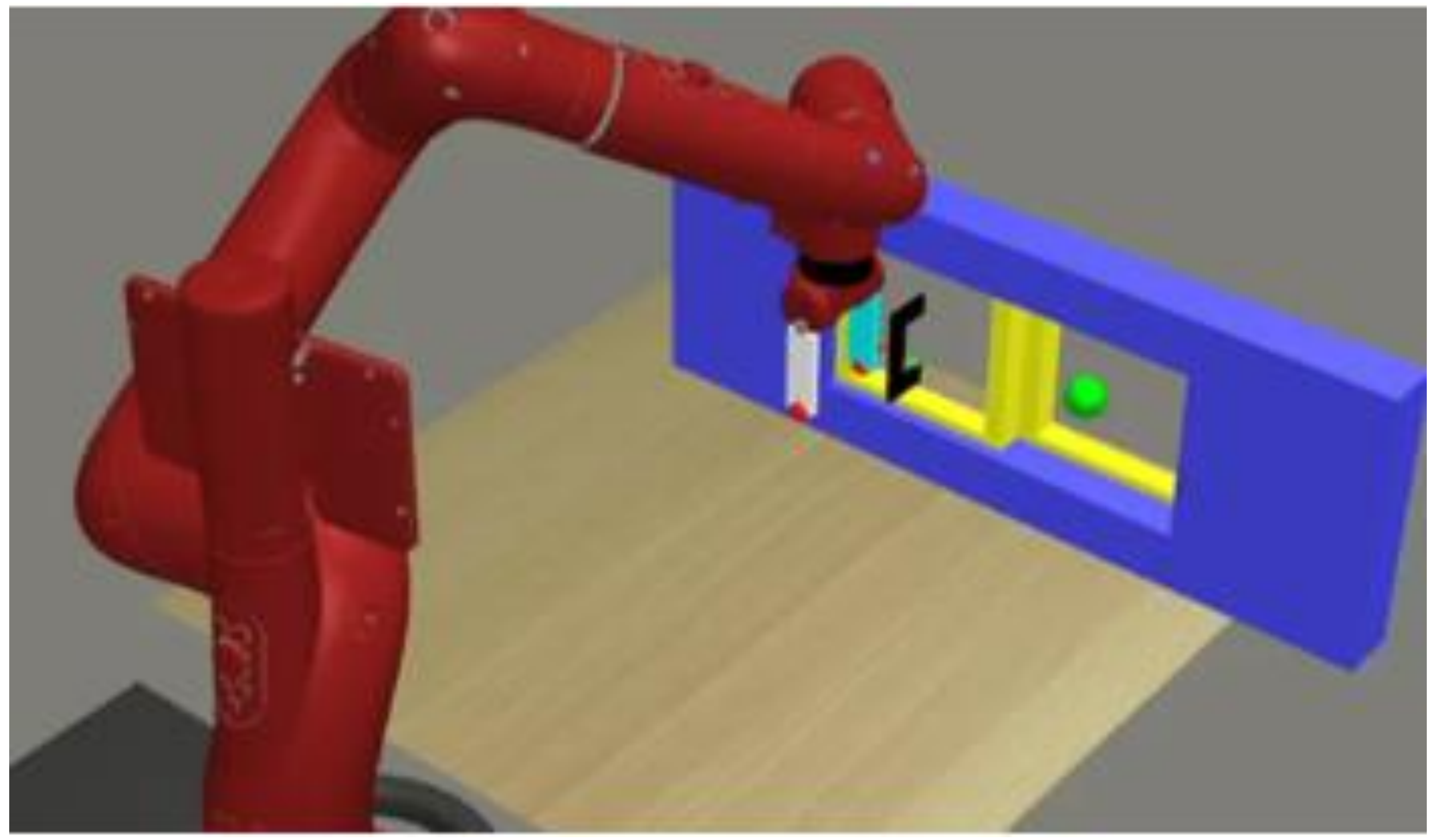
2.1 Few-Shot Imitation Problem

DEVIEW 2021

Given a few demonstrations of a new, previously unseen task → **Find** a policy which performs that task effectively.

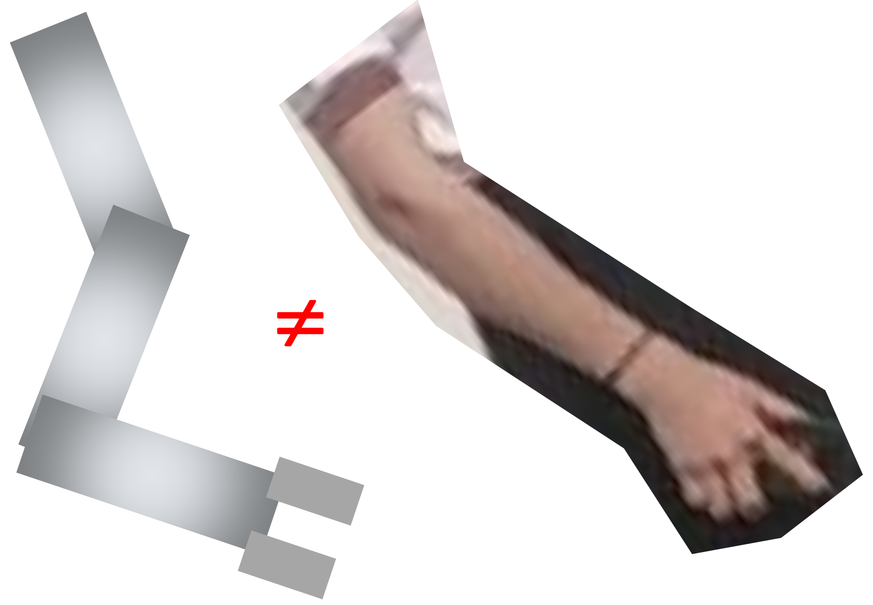


Must ***generalize*** to
✓ new tasks
✓ new states



$\text{action} \sim \pi(\cdot \mid \text{history}, \text{demonstrations})$

Demonstration is flexibly defined:
✓ noisy, incomplete, sub-optimal
✓ no actions
✓ human demonstrator + robot agent



2.2 Problem Formulation

Each task μ is an MDP.
$J_\mu(\pi)$ is the return for policy π.

Ingredients

η Distribution over tasks $\mu \sim \eta$

D_μ Distribution over collections $\mathbf{d}$ of demonstrations of task μ

few ↔ 1 to 10

$\pi(\cdot \mid h, \mathbf{d})$ *Demonstration-conditioned policy* given history h and demonstrations $\mathbf{d}$

partially observed

Objective

$$\max_{\pi} \; \mathbb{E}_{\mu \sim \eta} \; \mathbb{E}_{\mathbf{d} \sim D_\mu} \; J_\mu(\pi(\cdot \mid h, \mathbf{d}))$$

tasks

return

demonstration-conditioned policy

collections of demonstrations

2.2 New Idea

Demonstration-Conditioned Reinforcement Learning (DCRL)

DEVIEW 2021

DCRL maximizes the return of a demonstration-conditioned policy, averaged over a set of training tasks and corresponding demonstrations.

Train

$\mu^0, \ldots, \mu^{N-1}$ may not be distinct

Input Pairs $(\mathbf{d}^0, \mu^0), \ldots, (\mathbf{d}^{N-1}, \mu^{N-1})$ where $\mathbf{d}^i$ is a collection of demonstrations of task μ^i

Output A demonstration−conditioned policy $\boldsymbol{\pi}$ attaining $\max_{\pi} \sum_{i=0}^{N-1} J_{\mu^i}\left(\boldsymbol{\pi}(\cdot \mid \cdot, \mathbf{d}^i)\right)$

Test

Input
- $\mathbf{d}$ Collection of demonstrations of new, previously unseen task
- $\boldsymbol{\pi}$ Demonstration−conditioned policy given by DCRL

Repeat Observe history h_t and take action $a_t \sim \boldsymbol{\pi}(\cdot \mid h_t, \mathbf{d})$

No need for reward function

No need for to explore the test env't

2.3 Related Work

Behaviour Cloning

DEVIEW 2021

Idea

Behaviour cloning uses *supervised learning* to learn a policy

input demonstrations

$(\text{state}_0, \text{action}_0, \text{state}_1, \text{action}_1, \ldots)$

policy$_\theta$: state → action

learning

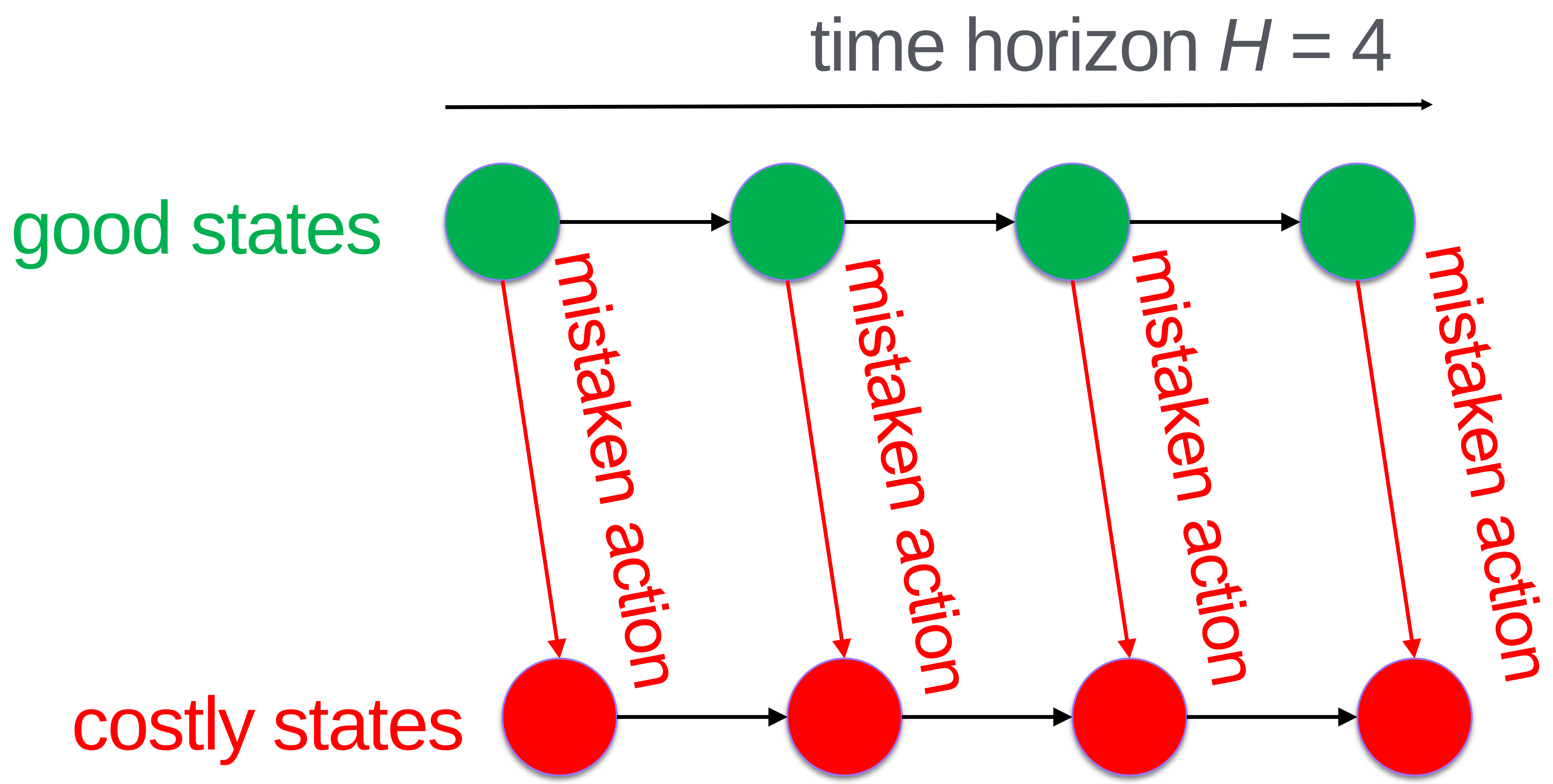
$\min_\theta \sum_t \text{prediction\_loss}(\text{action}_t, \text{policy}_\theta(\text{state}_t))$

2.3 Related Work

Behaviour Cloning

Issues

- Assumes actions in demonstrations
- Error compounding ⇒ loss is $O(H^2)$ on horizon H



- Rajaraman *et al.* (2020) showed that all behaviour cloning algorithms have this defect

2.3 Related Work

Behaviour Cloning

DEVIEW 2021

Few-Shot Imitation

- Earliest work on few-shot imitation (Duan *et al.*, 2017) relied on behaviour cloning
- Learned a demonstration-conditioned policy

$$\underbrace{\boldsymbol{\pi}(\cdot\,|s, \mathbf{d}^{i,1})}_{\text{policy taking first demo' } \mathbf{d}^{i,1} \text{ as input}}$$

2.3 Related Work

Behaviour Cloning

Few-Shot Imitation

- Earliest work on few-shot imitation (Duan *et al.*, 2017) relied on behaviour cloning
- Learned a demonstration-conditioned policy

$$\min_{\boldsymbol{\pi}} \sum_{i=0}^{N-1} \sum_{(s,a)\in\mathbf{d}^{i,2}} \textbf{loss}\Big(a, \underbrace{\boldsymbol{\pi}(\cdot\,|s, \mathbf{d}^{i,1})}_{\text{policy taking first demo' } \mathbf{d}^{i,1} \text{ as input}}\Big)$$

Train to predict actions a in second demo' $\mathbf{d}^{i,2}$

2.3 Related Work

Inverse Reinforcement Learning

Example. Look for a reward for which the demo’s would be optimal

Idea

- Infer reward function from demonstrations
- Train a policy to optimize that reward function

demo’s → [infer] → *reward fn.* → [optimize] → *policy*

Issues

- Reward is **non-unique**
  - May be many reward functions for which given trajectories are optimal
- Hard to improve if demo’s are **suboptimal**

Few-Shot Imitation

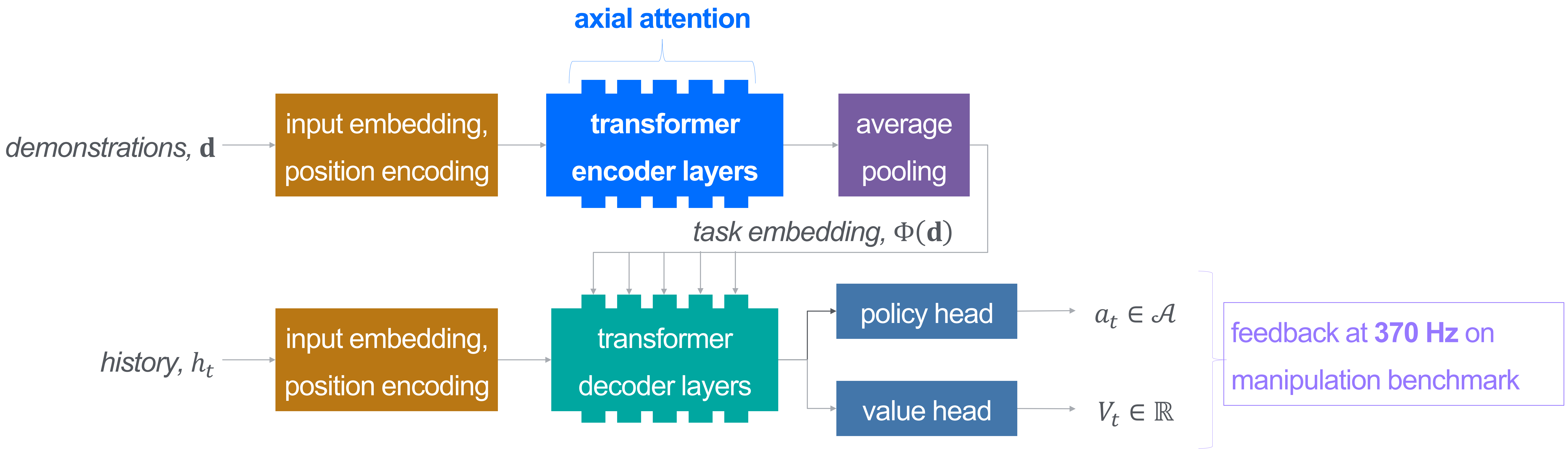
- Yu et al. (2019) extended inverse reinforcement learning to few-shot imitation
- **Assumption**: *structure and dynamics of the environment do not change with task*
- 40 hours of exploration of each test environment to overcome this assumption

2.3 Related Work

Comparison of few-shot imitation approaches

Desideratum	Behaviour Cloning	Inverse RL	DCRL (ours)
Copes without actions in demonstrations?	✗	✓	✓
No need to explore the test environment?	✓	✗	✓
Improves on suboptimal demonstrations?	✗	✗	✓
Copes when demonstrator is physically different?	✗	✗	✓
No need for rewards for training tasks?	✓	✓	✗

2.4 Implementation



Literature

Attention and transformers already in use for few-shot imitation (Duan *et al.* '17, Mishra '18, James '18, Dasari '20)

Novelty

1. **Cross-demonstration attention.** Process multiple demonstrations *jointly*.
2. **Axial attention.** Attend to one dimension of the input at a time (Ho *et al.* '18).
   Reduces time and memory from $O(T^2n^2)$ to $O\big(Tn(T+n)\big)$ for n time series of length T

3. Performance of Few-Shot Imitation Methods

3.0 Overview

- Introduce two benchmarks

… then demonstrate our claims that DCRL …

- Consistently outperforms behaviour cloning
- Learns error-recovery skills that transfer to new tasks
- Copes when the demonstrator has a different physical structure to the agent
- Outperforms suboptimal demonstrators

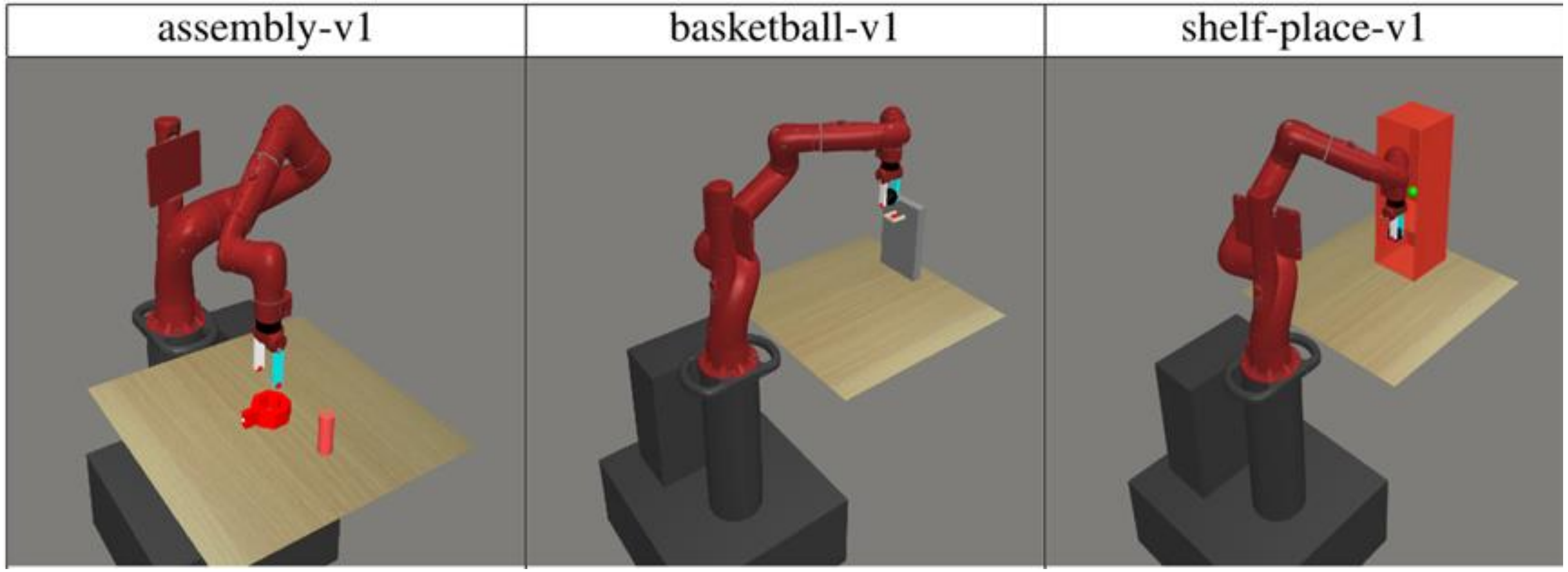
… and that cross-demonstration attention …

- Effectively resolves ambiguity, when a single demonstration is insufficient to identify a task.

3.1 Meta-World Benchmark

Meta-World (Yu *et al.*, 2019) consist of 50 diverse robotic manipulation tasks

Example. Score goal, remove peg, open window, close door

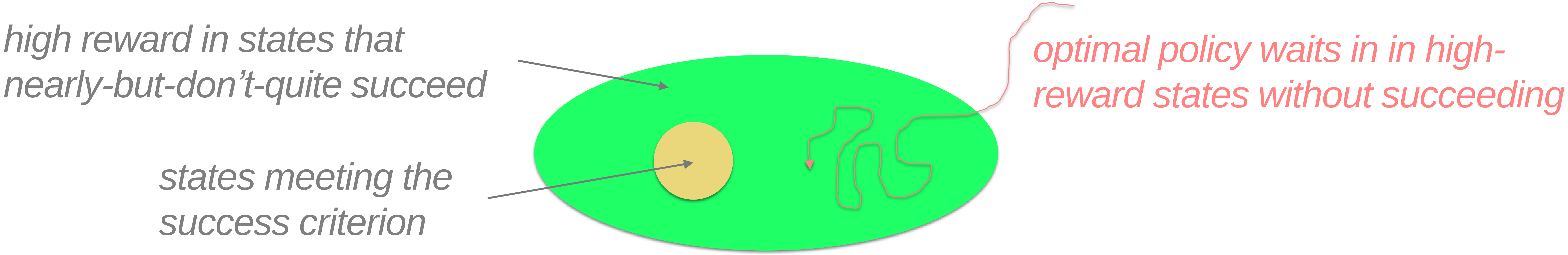


Task. Reward function, success criterion, MuJoCo model with Sawyer robot

3.1 Meta-World Benchmark

Step 1. Train one policy per task

Finding. Cumulative reward (= return) is maximized by ***failing*** on some tasks!



Solution. Modified reward which acts like the time-derivative of the original reward

Step 2. Train one policy for all tasks with no demonstration input.

$\Rightarrow$ Agent gets no information about the nature of the task at hand.

Finding. This policy has a ***48% success rate***!

$\Rightarrow$ Devise a second benchmark where demonstrations are more critical to succeeding at a task.

3.1 Navigation Benchmark

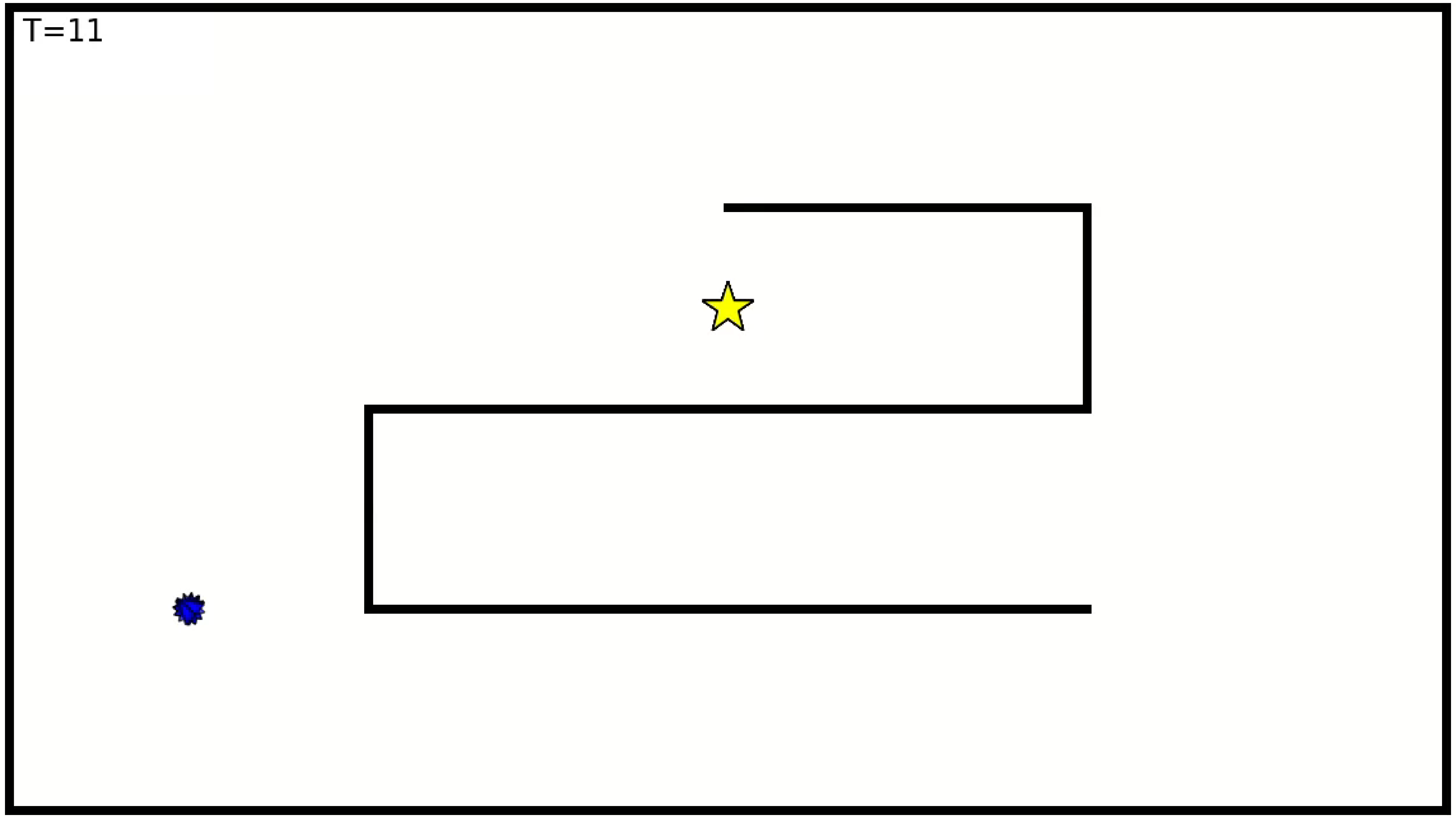
- Consists of 60 mazes, each of which corresponds to a single task.
- The task remains **ambiguous** even if we supply a single demonstration.

Aim. Get from start to goal state within a time limit.
Observe. Agent position, start and goal positions, the demonstrated paths.
Penalty. For hitting the walls.

Challenge. Agent must infer the layout of the walls from the demonstrations, which is challenging as the start and goal states are ***randomized***.

3.2 Navigation Benchmark

Several successful realizations of a single task



3.1 Benchmarks

***Meta-World**: 5-fold cross-validation*

***Navigation**: fixed split of 50 training mazes and 10 test mazes*

Generalization

- In all experiments, the test and training tasks were distinct
- Thus, the results truly measure *generalization to new tasks*

Demonstrations

- We trained per-task expert policies and used their trajectories as demonstrations
- As input to DCRL, we provided only state information in the demonstrations
  - *no action information*

Natural if videos of *human* used as demonstrations

3.2 Comparison with Behaviour Cloning

We compare with ***demonstration-conditioned behavioural cloning (DCBC)***

DCBC has the same architecture as our DCRL implementation

But DCBC is trained with a behaviour-cloning loss, rather than with RL
- Meta-World ↔ real-valued actions ↔ quadratic loss
- Navigation ↔ discrete actions ↔ cross-entropy loss

Thus DCBC is similar to the seminal work of Duan *et al.* (2017) which first studied one-shot imitation, except we have improved it by:
- Using a transformer, rather than a “soft attention network with convolutions”
- Doing few-shot rather than just one-shot imitation, with cross-demonstration attention
- Using history-dependent policies rather than attending only to the current state

3.2 Comparison with Behaviour Cloning

	Success rates			
	Meta-World		Navigation	
Method	**1 demo input**	**5 demos input**	**1 demo input**	**5 demos input**
DCBC	17%	24%	68%	68%
DCRL	48%	48%	77%	85%

win

win

3.2 Comparison with Behaviour Cloning

Success rates

Method	Meta-World: 1 demo input	Meta-World: 5 demos input	Navigation: 1 demo input	Navigation: 5 demos input
DCBC	17%	24%	68%	68%
DCRL	48%	48%	77%	85%
	little change		improves	

What if we reuse the demonstrations of test tasks to *finetune* DCRL or DCBC?

3.2 Comparison with Behaviour Cloning

	Success rates			
	Meta-World		Navigation	
Method	1 demo input	5 demos input	1 demo input	5 demos input
DCBC	17%	24%	68%	68%
DCRL	48%	48%	77%	85%
DCBC+finetune	52%	68%	58%	58%
DCRL+finetune	70%	90%	73%	80%

What if we reuse the demonstrations of test tasks to ***finetune*** DCRL and DCBC?

3.2 Comparison with Behaviour Cloning

DEVIEW 2021

Success rates

Method	Meta-World: 1 demo input	Meta-World: 5 demos input	Navigation: 1 demo input	Navigation: 5 demos input
DCBC	17%	24%	68%	68%
DCRL	48%	48%	77%	85%
DCBC+finetune	52%	68%	58%	58%
DCRL+finetune	70%	90%	73%	80%

helps!

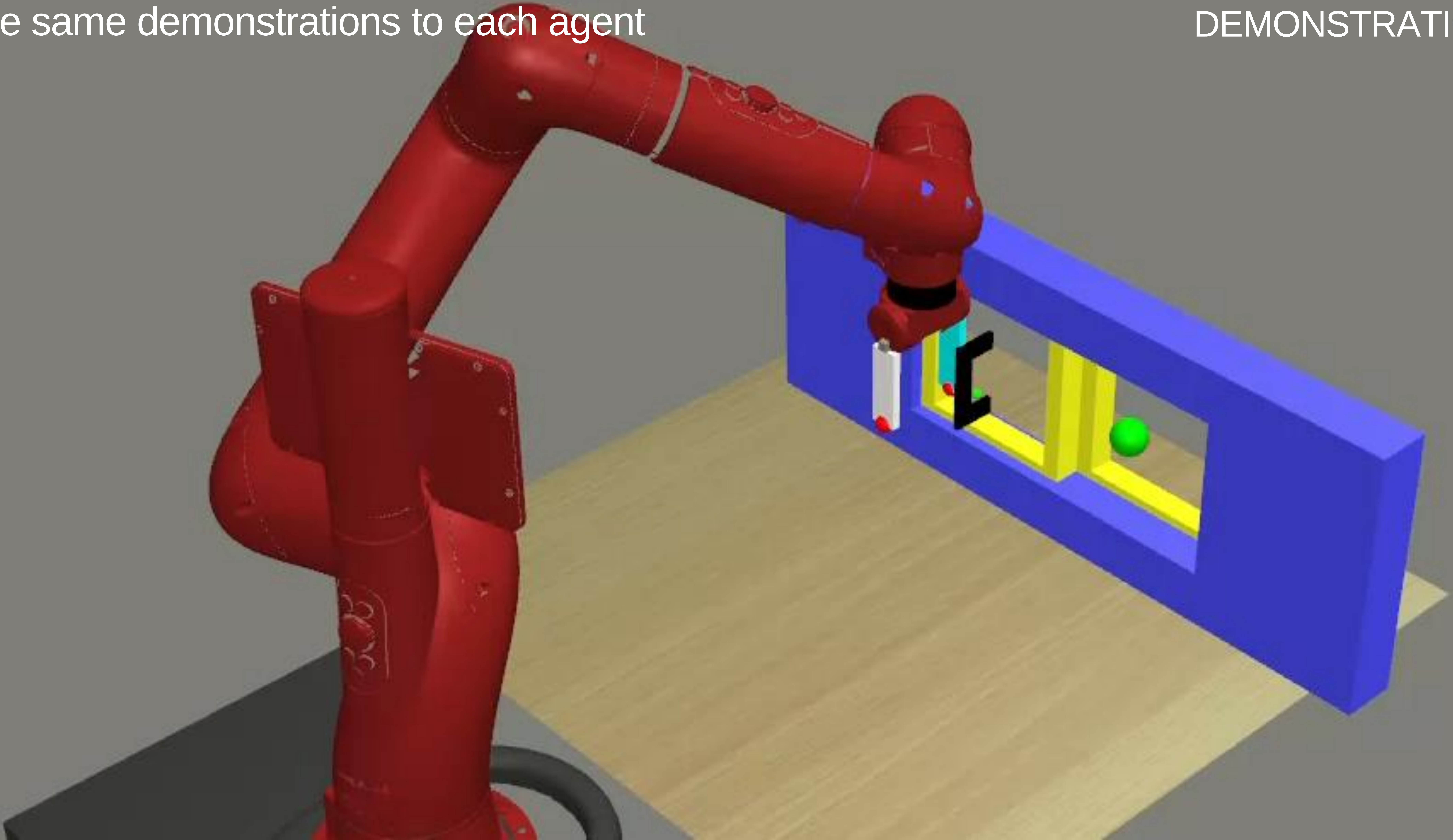
degrades

What if we reuse the demonstrations of test tasks to ***finetune*** DCRL and DCBC?

Why does DCLR outperform DCBC?

Present the same demonstrations to each agent

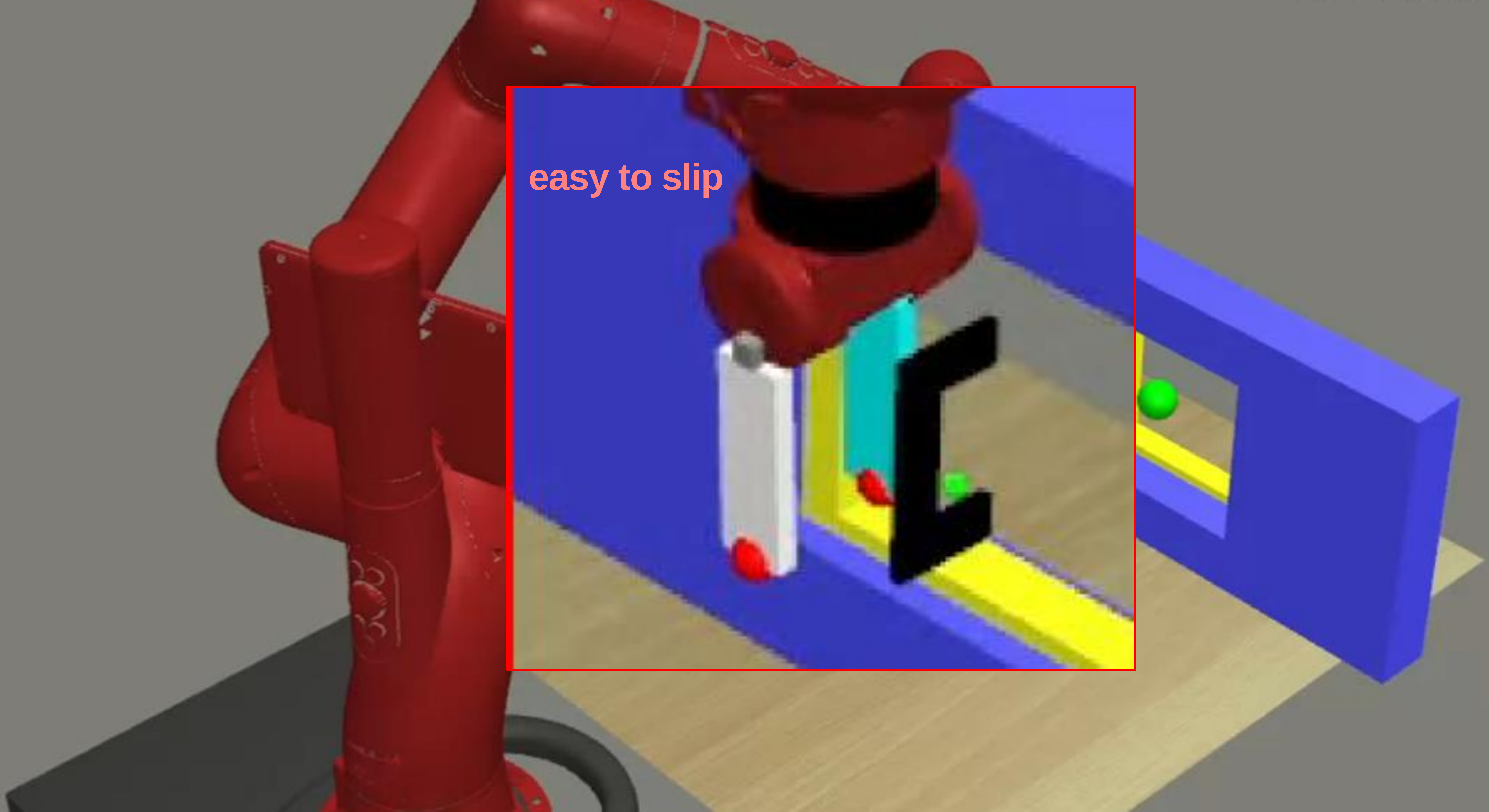
DEMONSTRATIONS



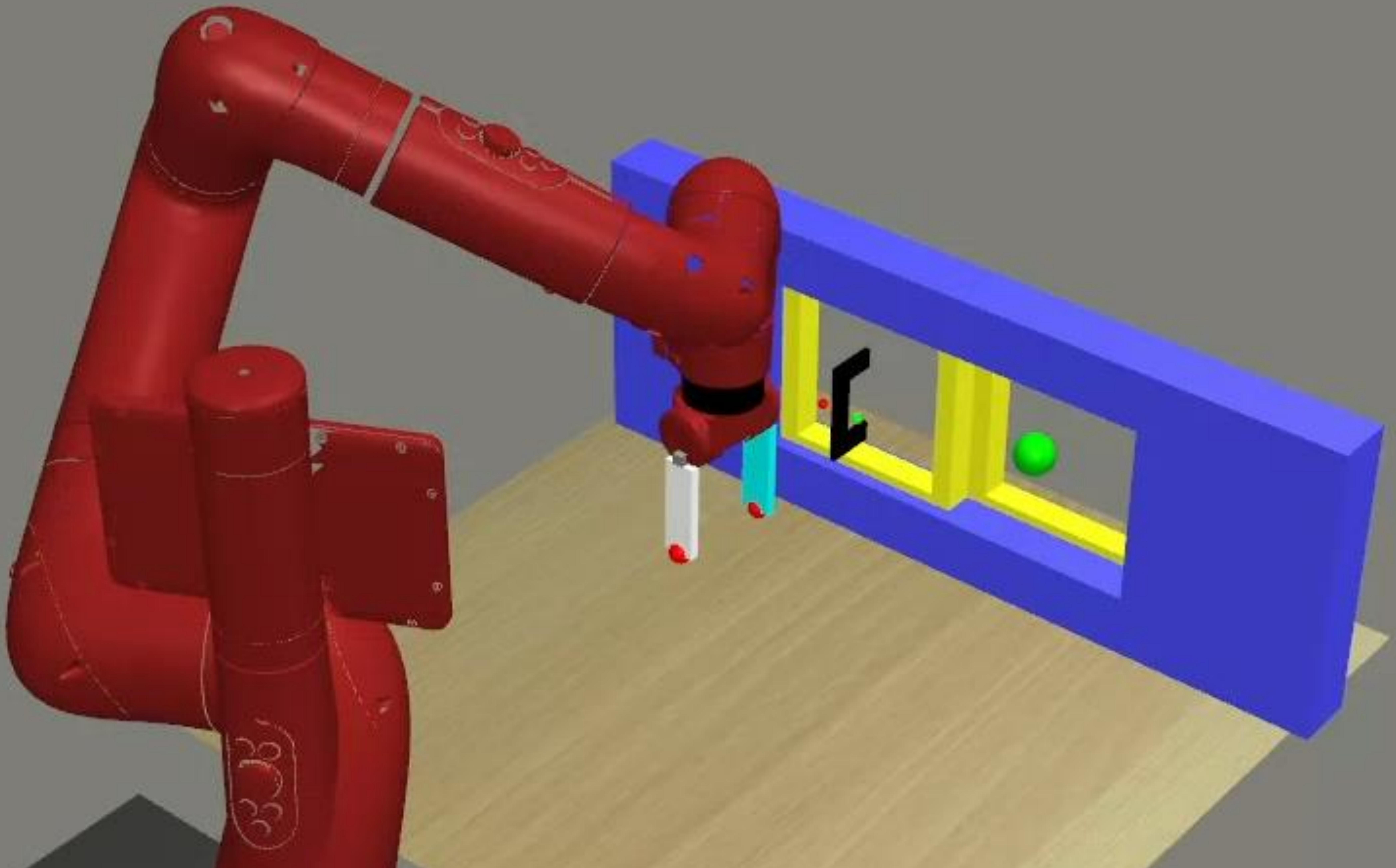
Why does DCLR outperform DCBC?

Present the same demonstrations to each agent

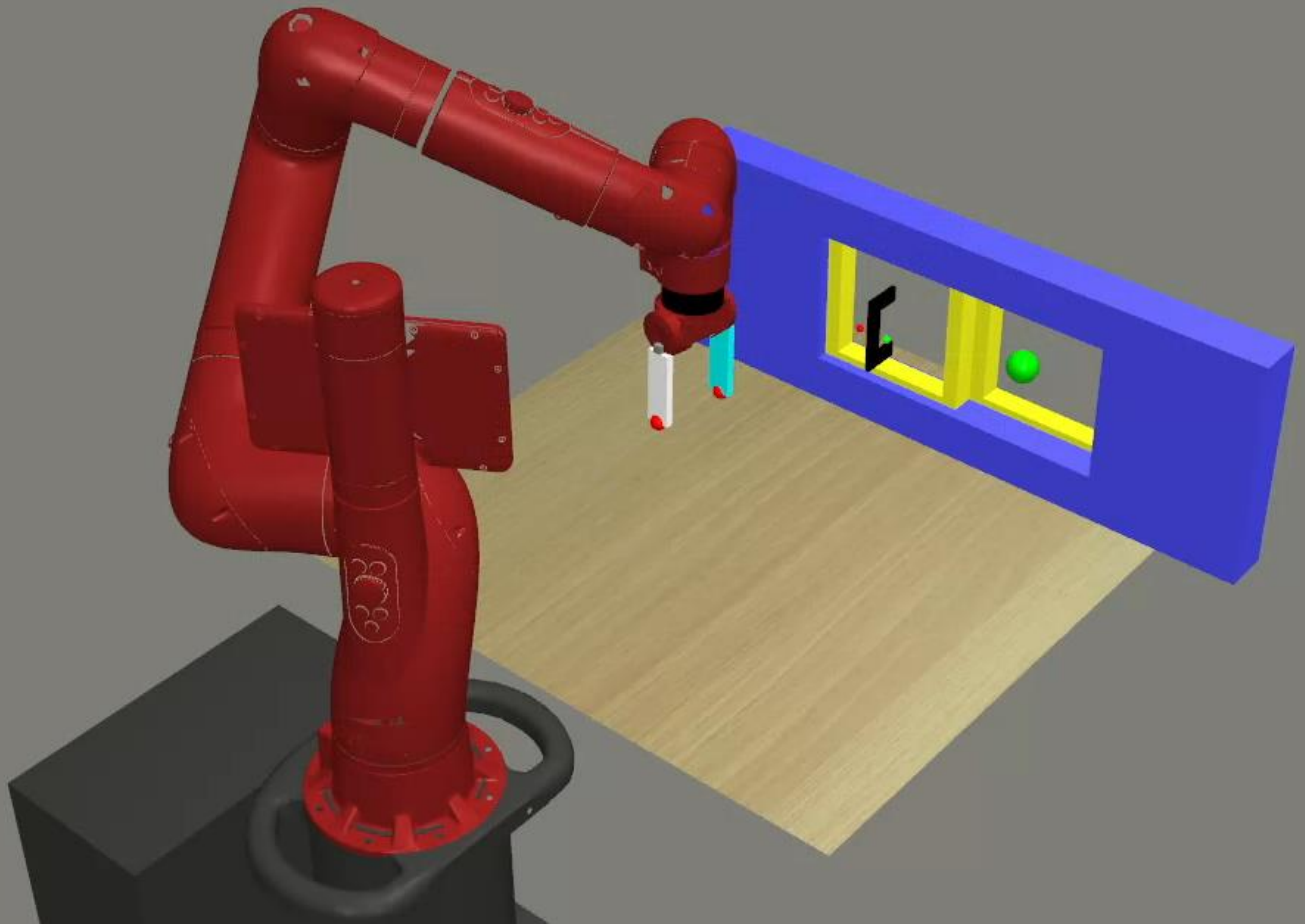
DEMONSTRATIONS



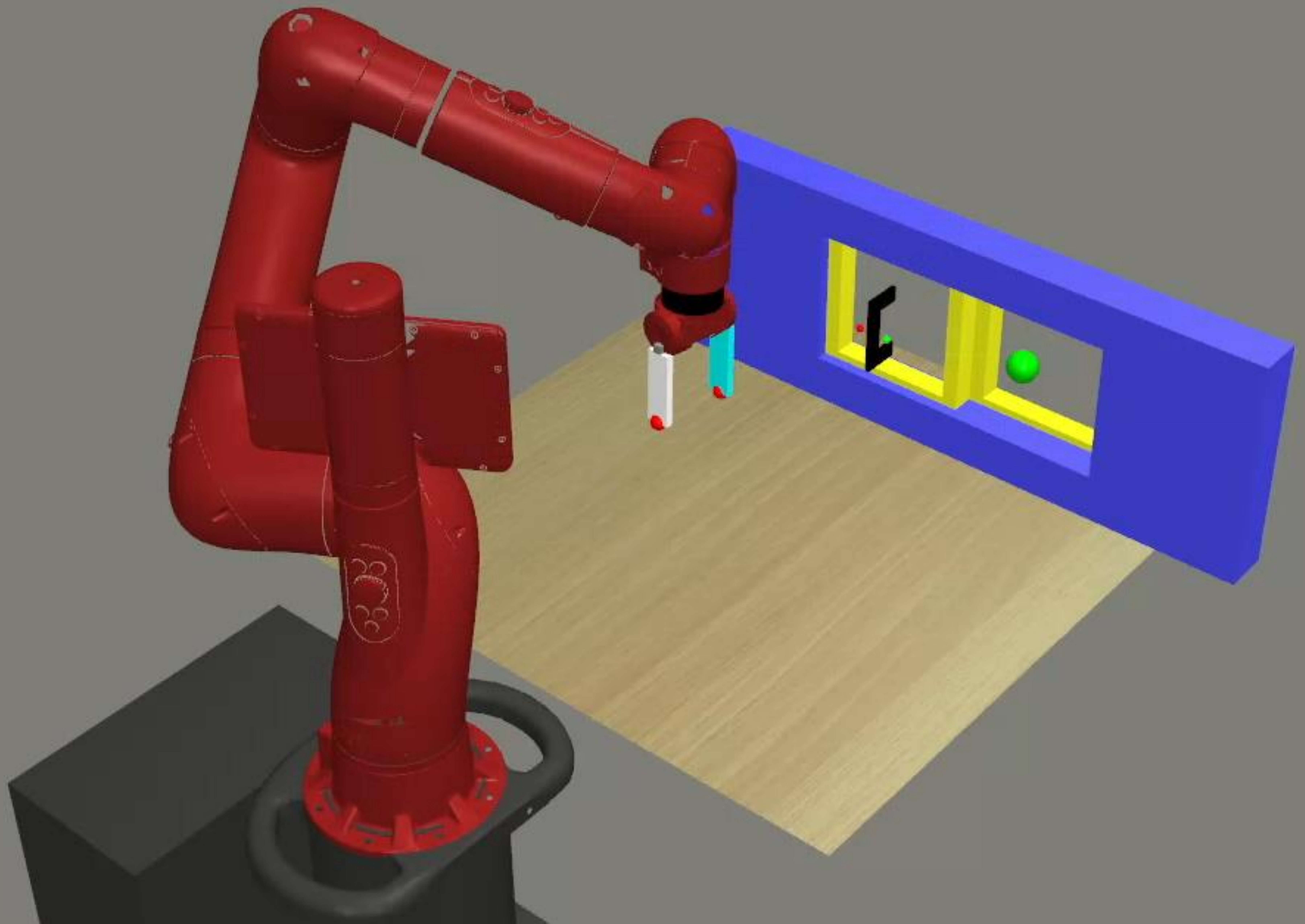
DCBC slips and fails



DCRL slips too, but then it recovers!



3.3 DCRL learns error-recovery skills which it can transfer to new tasks!

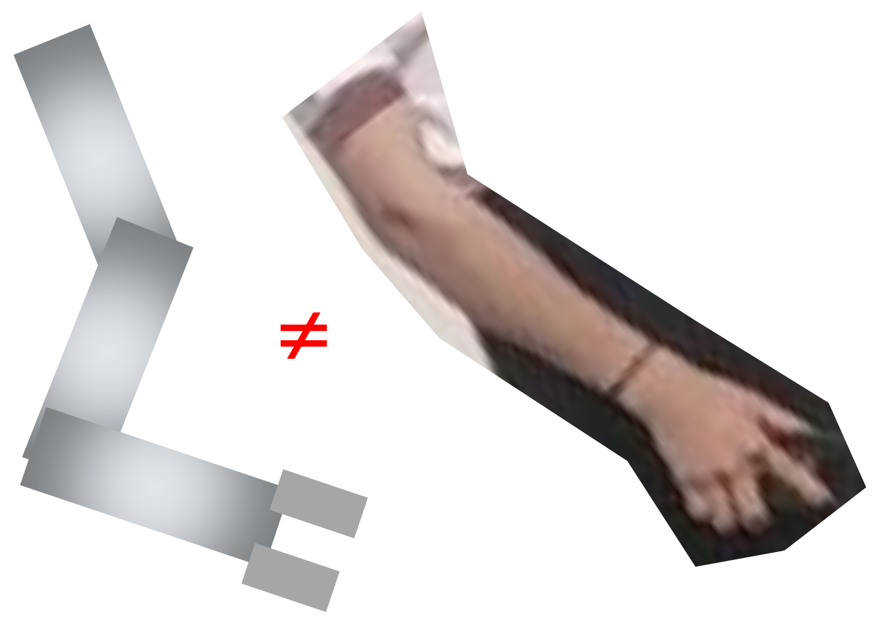


3.4 Demonstrator Domain Shift

DEVIEW 2021

Motivation

- We would like our agent to control a robot given demonstrations from a human demonstrator.
- But a human might have a rather different physical structure to the robot.
- In that case, cloning a human's actions would make little sense.



3.4 Demonstrator Domain Shift

Motivation

- We would like our agent to control a robot given demonstrations from a human demonstrator.
- But a human might have a rather different physical structure to the robot.
- In that case, cloning a human's actions would make little sense.

Experimentally Compare

1. Controlling a Sawyer robot given demonstrations from an AMBIDEX robot

with

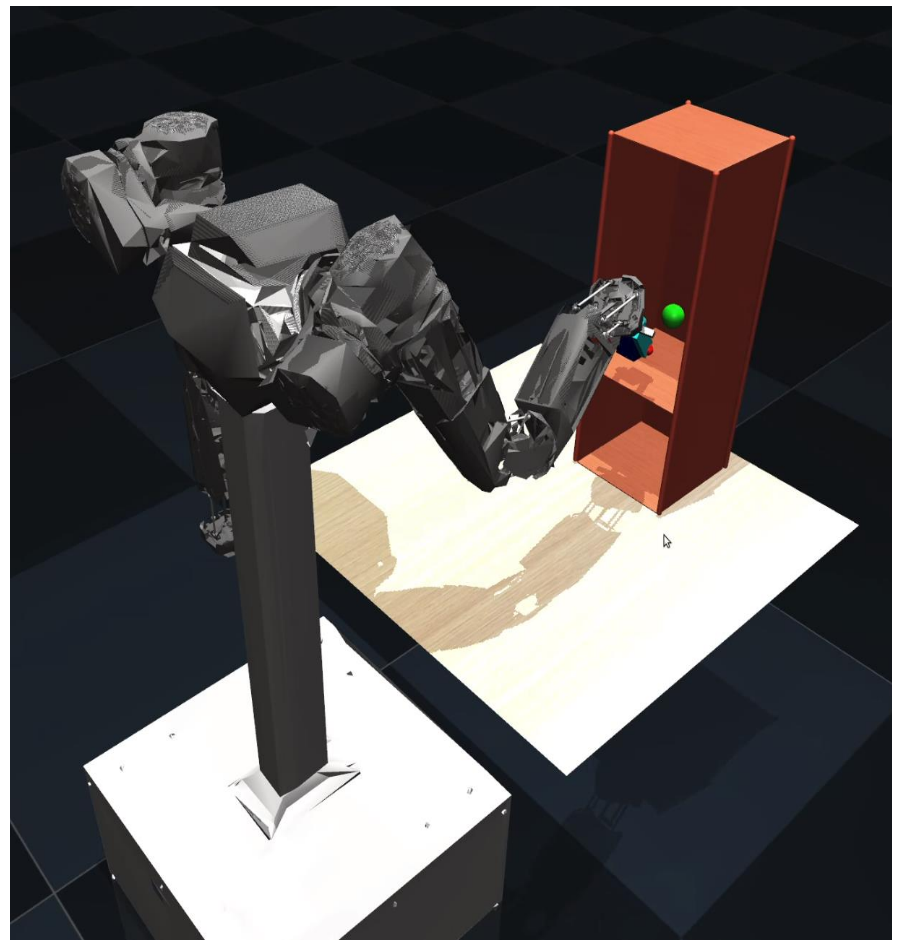
2. Controlling a Sawyer robot given demonstrations from another Sawyer robot

3.4 Demonstrator Domain Shift

DEVIEW 2021

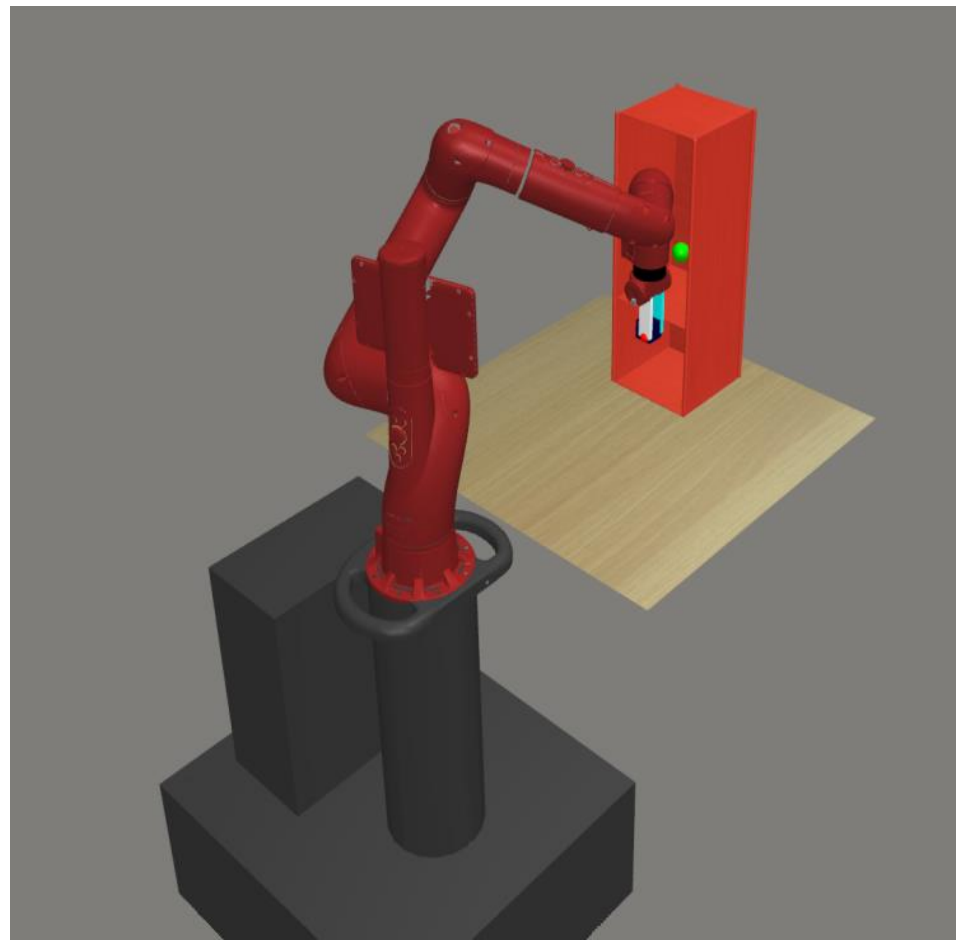
Example

AMBIDEX robot demonstrates a new task



→ DCRL agent →

Sawyer robot performs the new task



Results

Which robot demonstrates new tasks

	success rate		average return	
# demo's	1	5	1	5
Sawyer	**51%**	**51%**	**316**	323
AMBIDEX	45%	48%	308	**329**

6% lower at worst

nearly identical

3.5 Suboptimal Demonstrators

Motivation

Demonstrations from humans may be suboptimal:

- Clumsiness, natural variability, noisy perception

Question

Can DCRL perform tasks better than a suboptimal demonstrator?

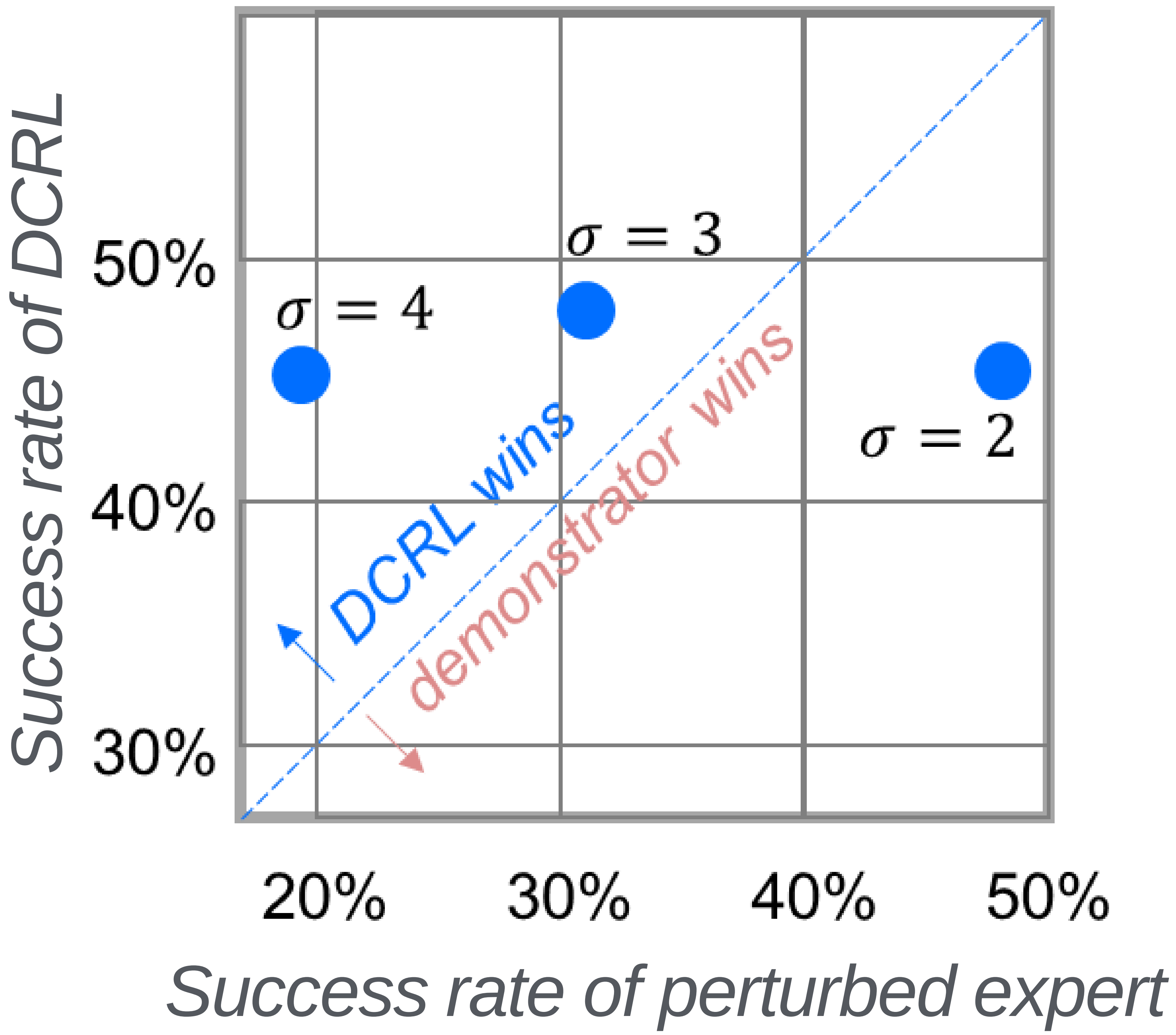
Experiment

- Add noise $\sim \mathcal{N}(0, \sigma^2 I_{4\times 4})$ to the expert's actions (only at test)
- Compare the success rates of
  - This perturbed expert; and
  - DCRL using demonstrations from this perturbed expert.

3.5 Suboptimal Demonstrators

Results

- Success rates on Meta-World are shown
- For $\boldsymbol{\sigma}>\mathbf{2}$, DCRL outperforms the noisy demonstrator



Remark on Interpretation

- We only added noise at test time.
- We would surely do better in practice if we also train with demonstrations having typical "clumsiness" and perceptual noise characteristics

3.6 Benefit of Cross-Demonstration Attention

DEVIEW 2021

Motivation

Previous authors only considered ***one-shot*** imitation

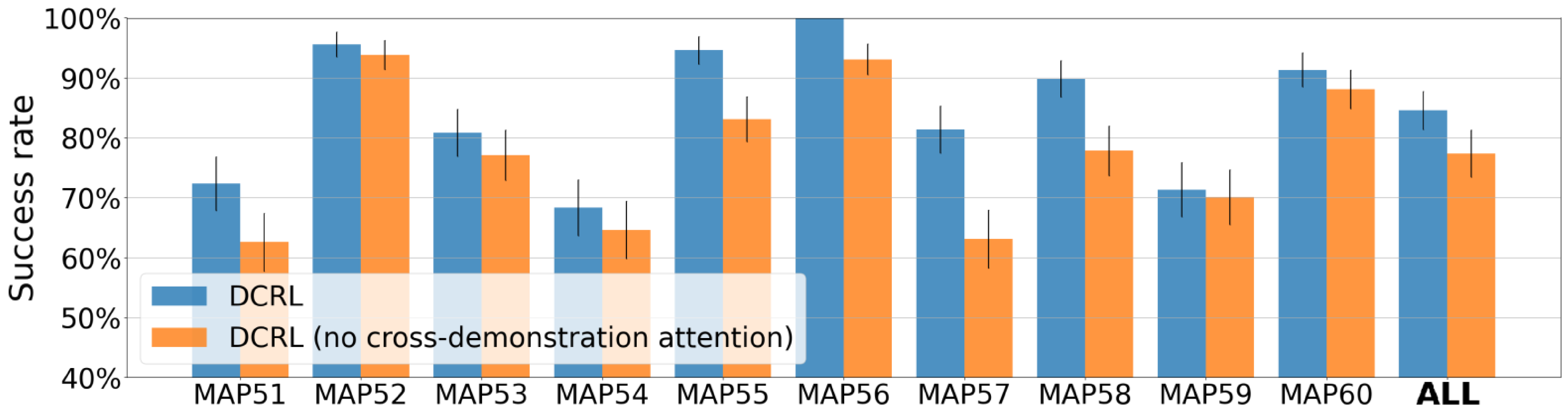
Except James *et al.* (2018) who fed one demonstration at a time to their network

What if a single demonstration leaves a lot of ambiguity about the nature of the task?

Comparison on navigation benchmark

Cross-demo' attention: feed 5 demonstration to the network simultaneously

No cross-demo' baseline: feed 1 demonstration at a time to the network, average the resulting action probabilities over the 5 demonstrations



4. Why does it work and what's next?

4.1 Why does it work?

Question. *How does our DCRL implementation generalize to new tasks?*

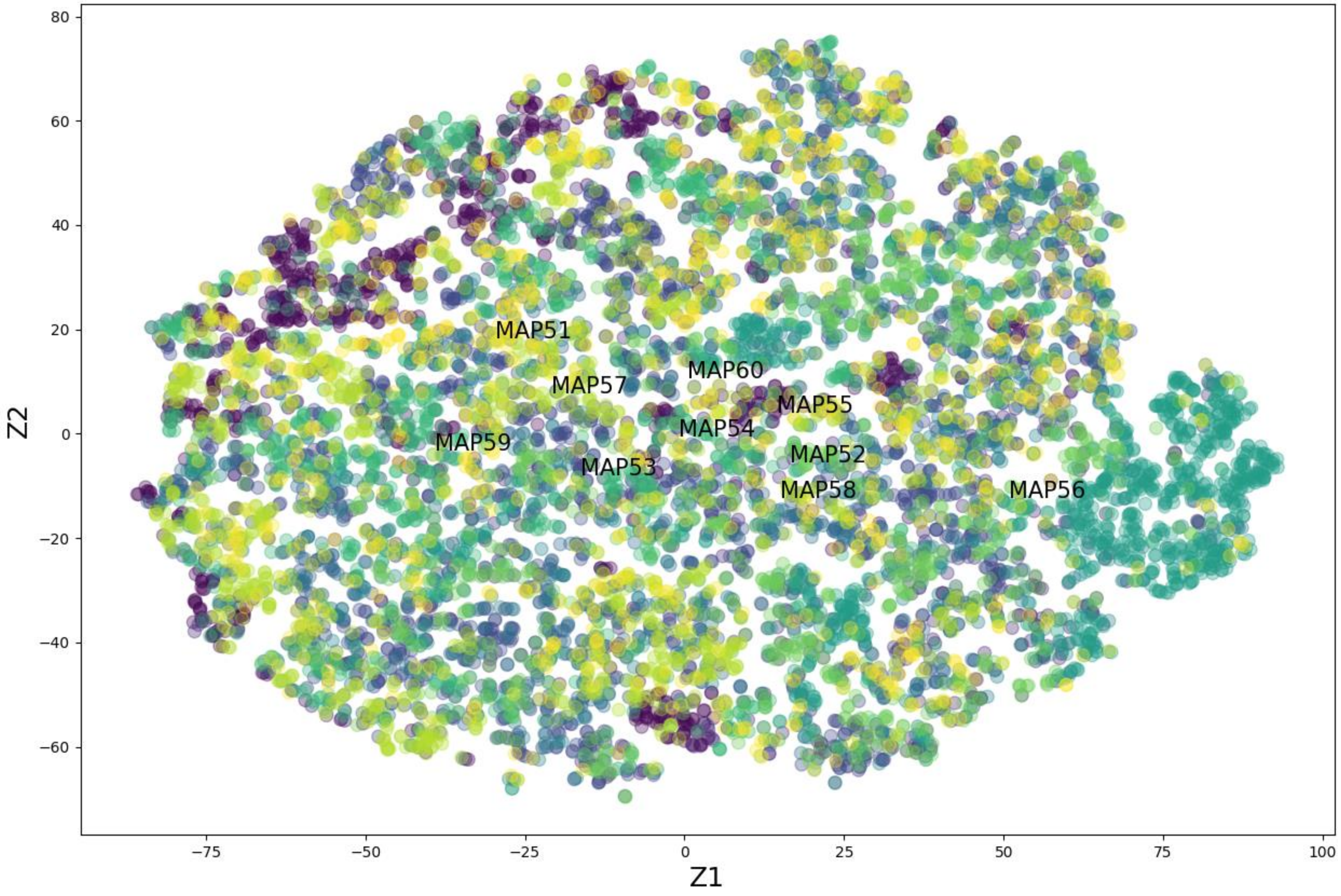
Intuition. Collections of demonstrations are close under the encoder mapping if and only if they correspond to tasks with similar optimal policies.

t-SNE. Visualize high-dimensional data while preserving clustering
(van der Maaten and Hinton, 2008).

t-SNE(demonstrations)

DEVIEW 2021

for collections of 4 demonstrations of the 10 navigation test tasks

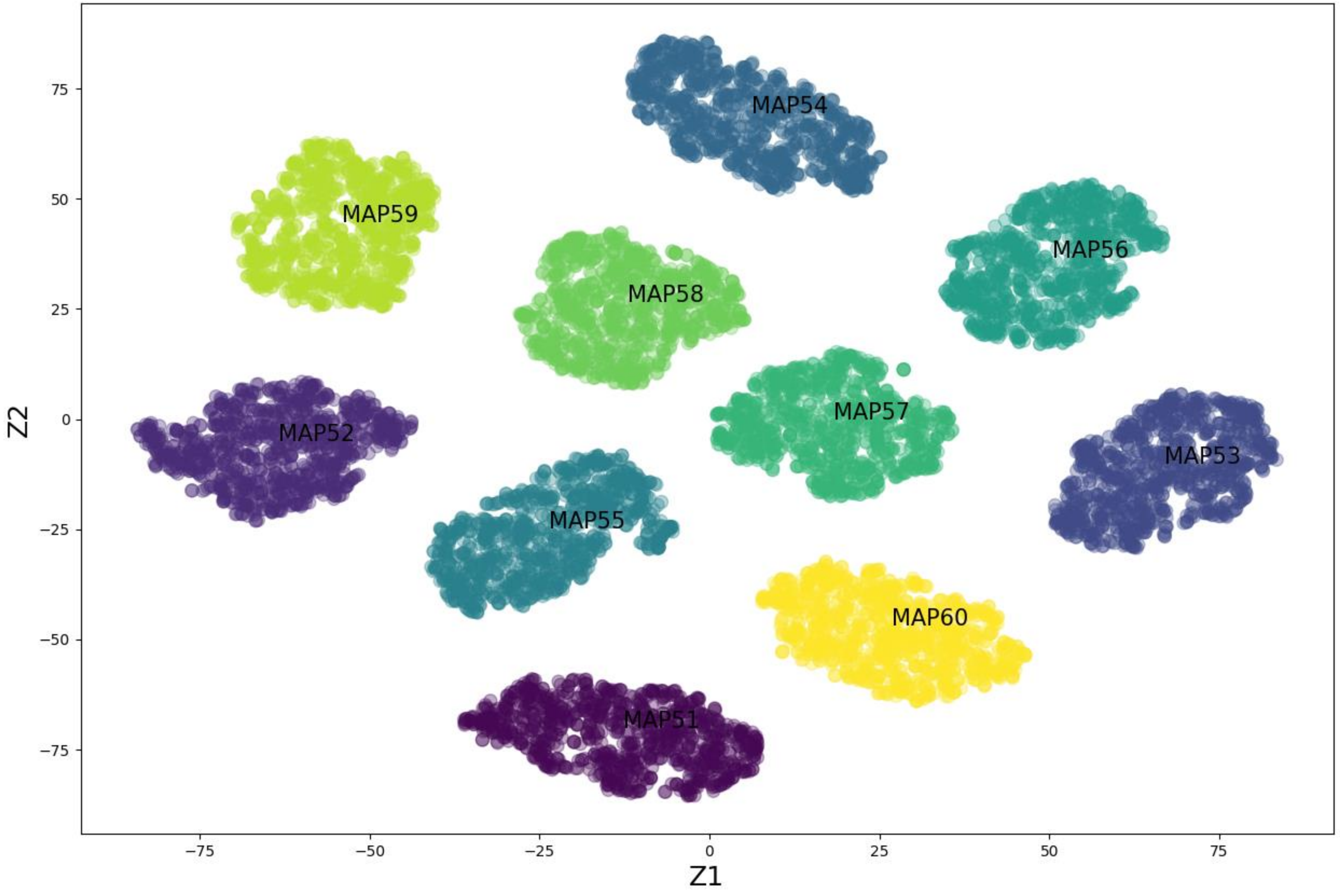


Different colours correspond to different mazes.

Z1 and Z2 are just arbitrary names for the axes of the t-SNE plot.

t-SNE(randomly_initialized_embedding(demonstrations))

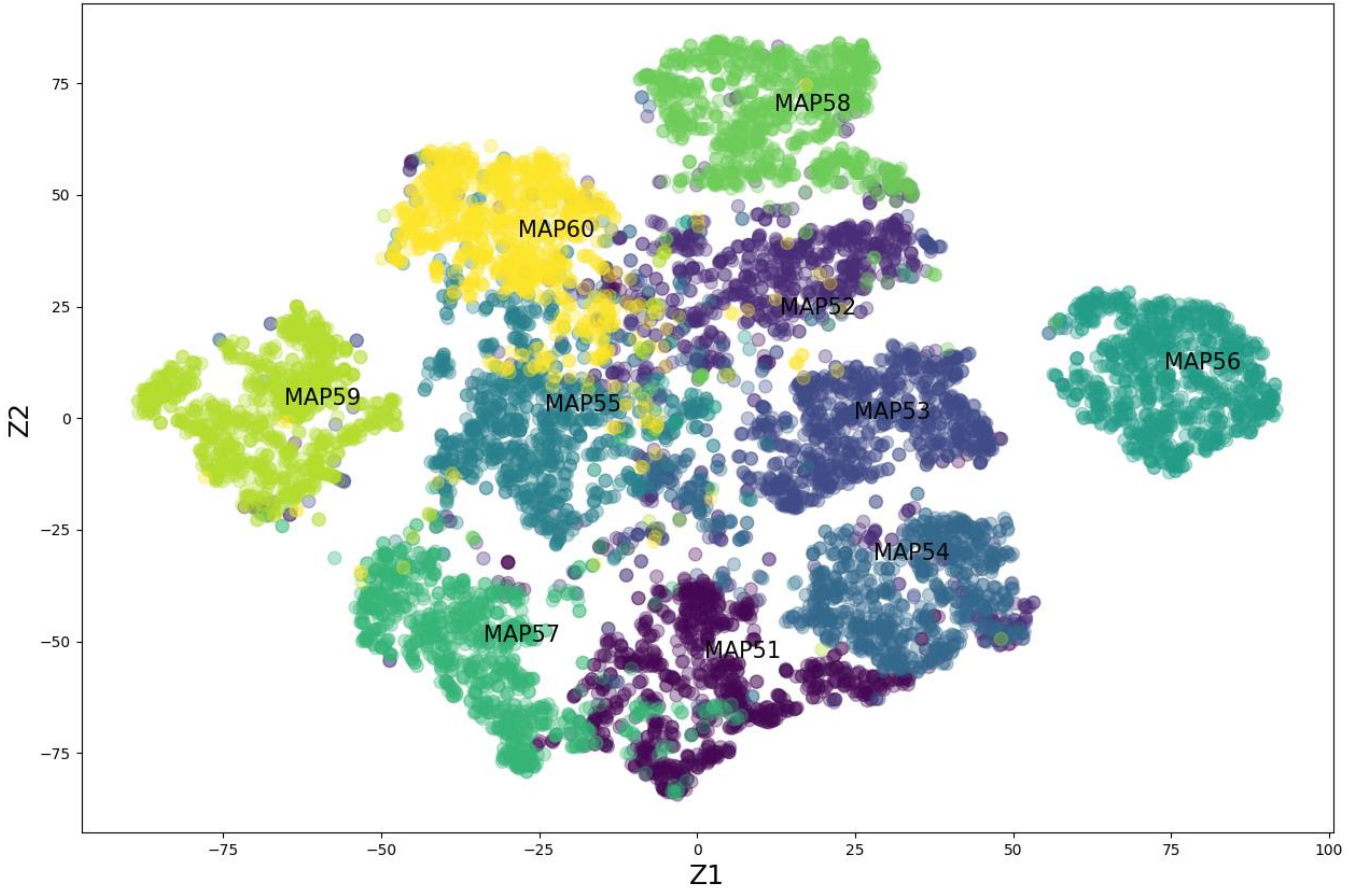
for collections of 4 demonstrations of the 10 navigation test tasks



Even though this is a random embedding, the data is surprisingly clustered!

t-SNE(learned_embedding(demonstrations))

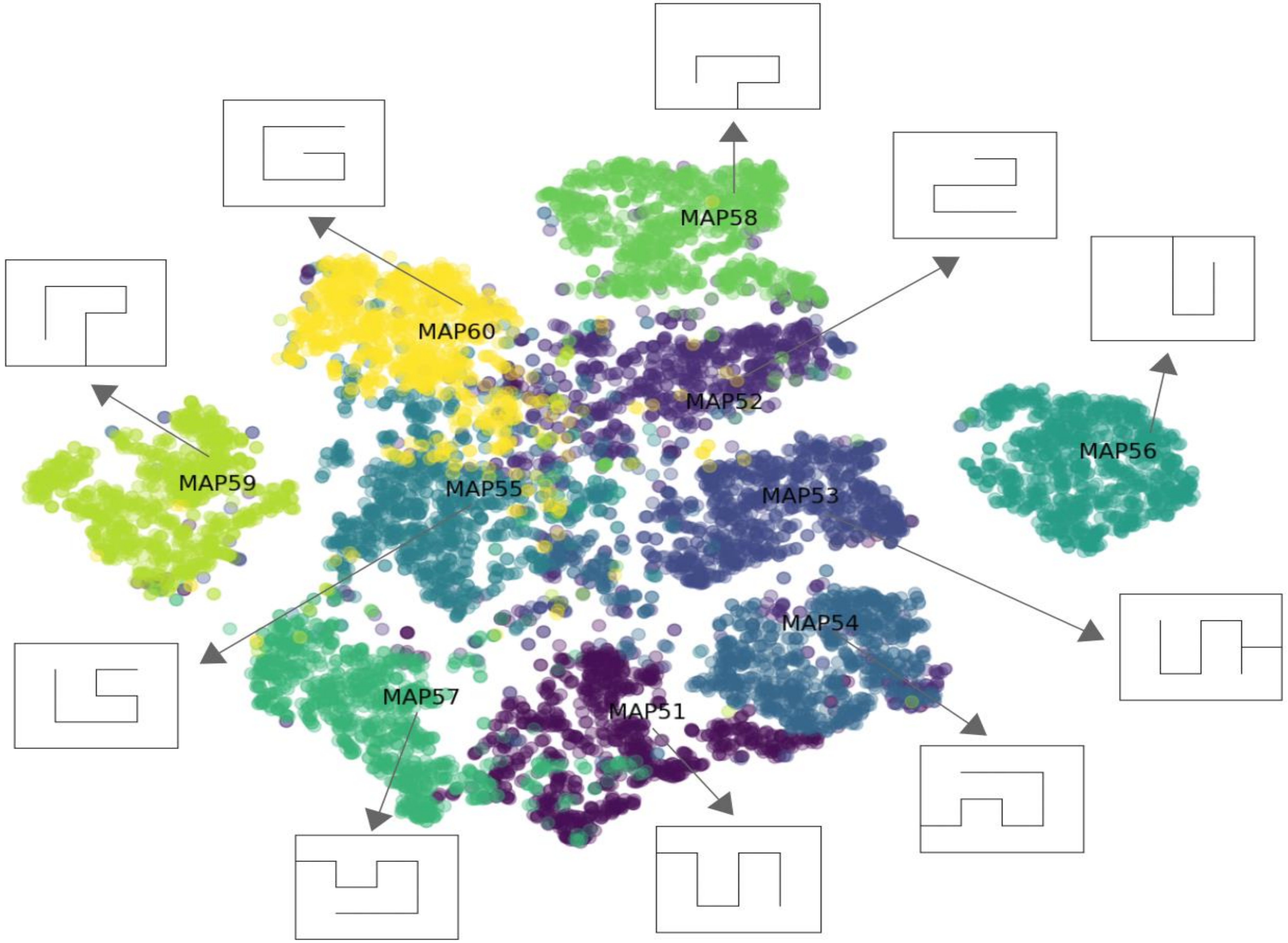
for collections of 4 demonstrations of the 10 navigation test tasks



Learning degrades the clustering! Why?

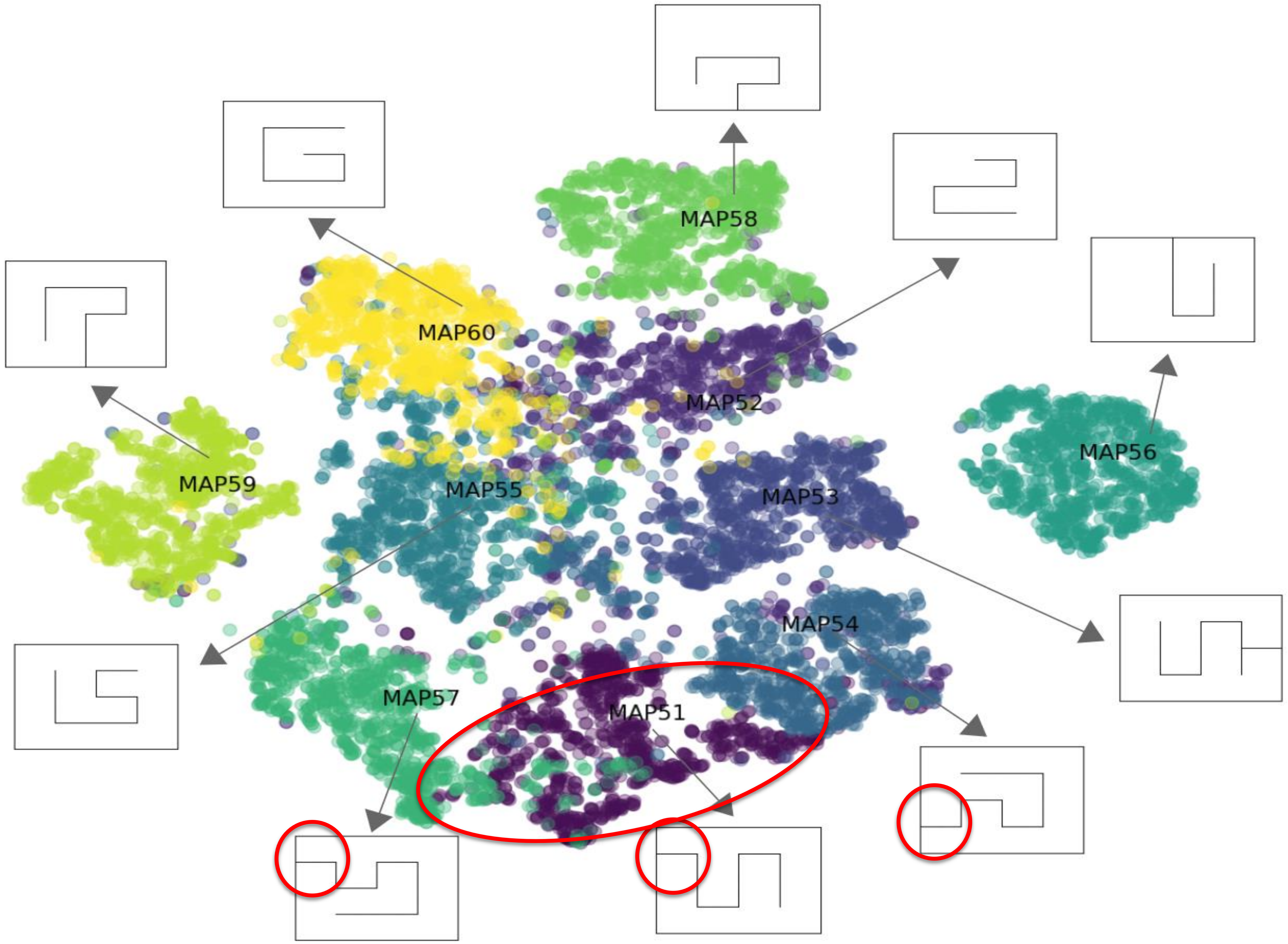
t-SNE(learned_embedding(demonstrations))

for collections of 4 demonstrations of the 10 navigation test tasks



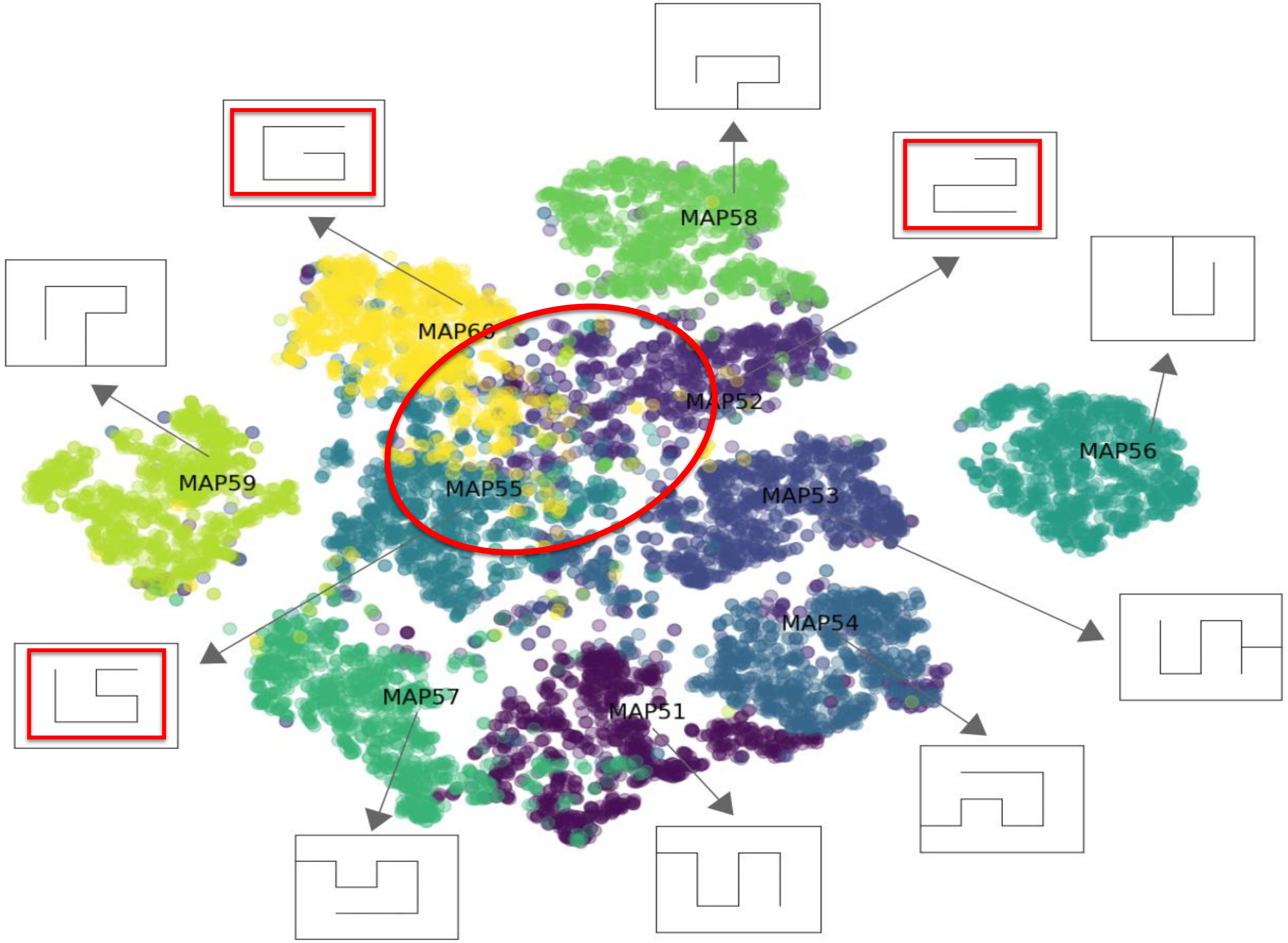
t-SNE(learned_embedding(demonstrations))

for collections of 4 demonstrations of the 10 navigation test tasks



t-SNE(learned_embedding(demonstrations))

for collections of 4 demonstrations of the 10 navigation test tasks



Can't hope to draw conclusions by looking at 10 mazes.

But interesting to see how learning brings clusters with similar optimal policies together.

4.2 In Future

Better generalization:

☼ more training tasks

- DCRL was trained with only 40 or 50 tasks!
- Can we automatically generate 100s of diverse, realistic but solvable tasks?

☼ better actor-critic training

Phasic policy gradient (Cobbe et al., 2021)

Real-world application:

☼ videos of human demonstrators

☼ real robot rather than simulations

Sim-to-real and offline RL

Richer input:

☼ success / failure feedback

Watch, Try, Learn (Zhou *et al*., 2019)

☼ natural language input

4.3 Conclusion

DCRL is a new, third family of approaches to few-shot imitation

$$\{\text{ inverse RL, behaviour cloning }\} \cup \{\text{ DCRL }\}$$

Advantages

+ learns error-recovery skills, which transfer to new tasks
+ can improve on suboptimal demonstrations
+ can cope with demonstrator domain shift
+ does not need to explore the test environment

Disadvantage

- requires reward functions for training tasks – but maybe we can automatically generate them?

See also: Cachet, Dance and Perez, Demonstration-Conditioned Reinforcement Learning for Few-Shot Imitation, *ICML* 2021

Q & A

Thank You!