Inverse Reinforcement Learning

Chelsea Finn

3/5/2017

Course Reminders:
March 22nd: Project group & title due
April 17th: Milestone report due & milestone presentations
April 26th: Beginning of project presentations

Inverse RL: Outline

1. Motivation & Definition
2. Early Approaches
3. Maximum Entropy Inverse RL
4. Scaling inverse RL to deep cost functions

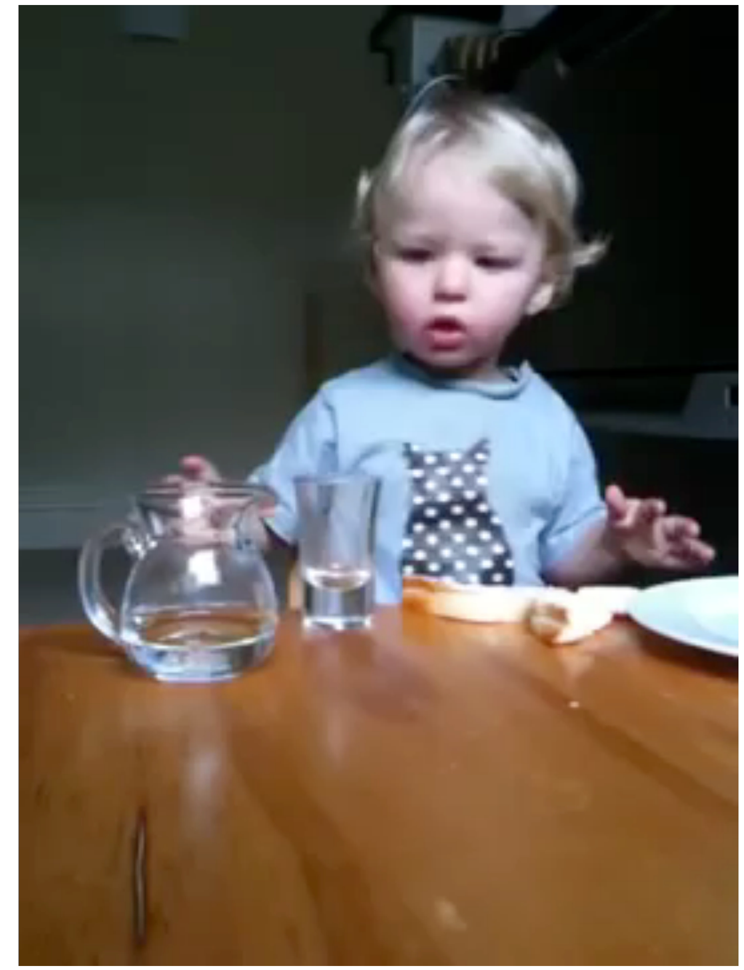
Inverse RL: Outline

1. **Motivation & Definition**
2. Early Approaches
3. Maximum Entropy Inverse RL
4. Scaling inverse RL to deep cost functions

reward

Mnih et al. ’15

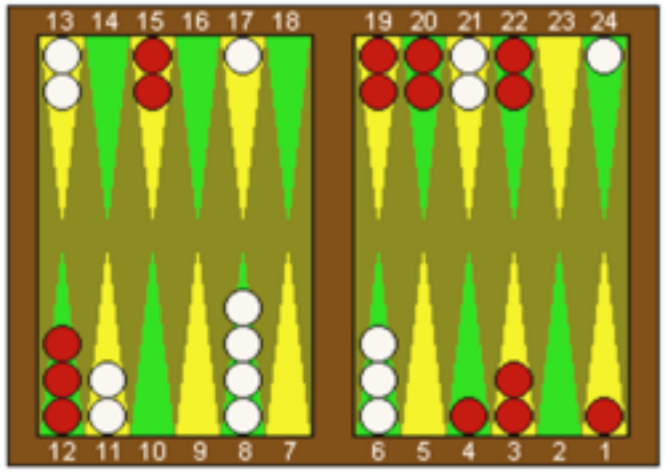
reinforcement learning agent



what is the reward?

In the real world, humans don’t get a score.

video from Montessori New Zealand



Tesauro '95

Kohl & Stone, '04

Mnih et al. '15

AlphaGo

Silver et al. '16

reward function is essential for RL

real-world domains: reward/cost often difficult to specify

- robotic manipulation
- autonomous driving
- dialog systems
- virtual assistants
- and more…

Motivation

Behavioral Cloning: Mimic actions of expert

- but no reasoning about outcomes or dynamics
- the expert might have different degrees of freedom

Can we reason about what the expert is trying to achieve?

Inverse Optimal Control / Inverse Reinforcement Learning:
infer cost/reward function from expert demonstrations

(Kalman '64, Ng & Russell '00)

(IOC/IRL)

Inverse Optimal Control / Inverse Reinforcement Learning:

infer cost/reward function from demonstrations

given:
- state & action space
- roll-outs from π*
- dynamics model [sometimes]

goal:
- recover reward function
- then use reward to get policy

Compare to DAgger: no direct access to π*

Challenges

underdefined problem

difficult to evaluate a learned cost

demonstrations may not be precisely optimal

Early IRL Approaches

All: alternate between solving MDP w.r.t. cost and updating cost

Ng & Russell '00: expert actions should have higher value than other actions, larger gap is better

Abbeel & Ng '04: expert policy w.r.t. cost should match feature counts of expert trajectories

Ratliff et al. '06: max margin formulation between value of expert actions and other actions

How to handle ambiguity? What if expert is not perfect?

Inverse RL: Outline

1. Motivation & Examples
2. Early Approaches
3. **Maximum Entropy Inverse RL**
4. Scaling inverse RL to deep cost functions

Maximum Entropy Inverse RL

(Ziebart et al. '08)

Notation:

$$\tau = \{s_1, a_1, ..., s_t, a_t, ..., s_T\}$$

c_θ: cost with parameters θ [linear case $c_\theta(\tau) = \theta^T \mathbf{f}_\tau = \sum_{s \in \tau} \theta^T \mathbf{f}_s$]

$\mathcal{D}$: dataset of demonstrations $M = |\mathcal{D}|$

T: transition dynamics

Whiteboard

Maximum Entropy Inverse RL

(Ziebart et al. '08)

0. Initialize θ, gather demonstrations $\mathcal{D}$
1. Solve for optimal policy $\pi(a|s)$ w.r.t. c_θ with value iteration
2. Solve for state visitation frequencies $p(s \mid \theta, T)$
3. Compute gradient $\nabla_\theta \mathcal{L} = \frac{1}{M} \sum_{\tau_d \in \mathcal{D}} \mathbf{f}_{\tau_d} + \sum_s p(s \mid \theta, T) \mathbf{f}_s$
4. Update θ with one gradient step using $\nabla_\theta \mathcal{L}$

Inverse RL: Outline

1. Motivation & Examples
2. Early Approaches
3. Maximum Entropy IRL
4. **Scaling IRL to deep cost functions**

Case Study: MaxEnt Deep IRL

MaxEnt IRL with known dynamics (tabular setting), neural net cost

Maximum Entropy Deep Inverse Reinforcement Learning

Markus Wulfmeier MARKUS@ROBOTS.OX.AC.UK
Peter Ondrúška ONDRUSKA@ROBOTS.OX.AC.UK
Ingmar Posner INGMAR@ROBOTS.OX.AC.UK
Mobile Robotics Group, Department of Engineering Science, University of Oxford

NIPS Deep RL workshop 2015

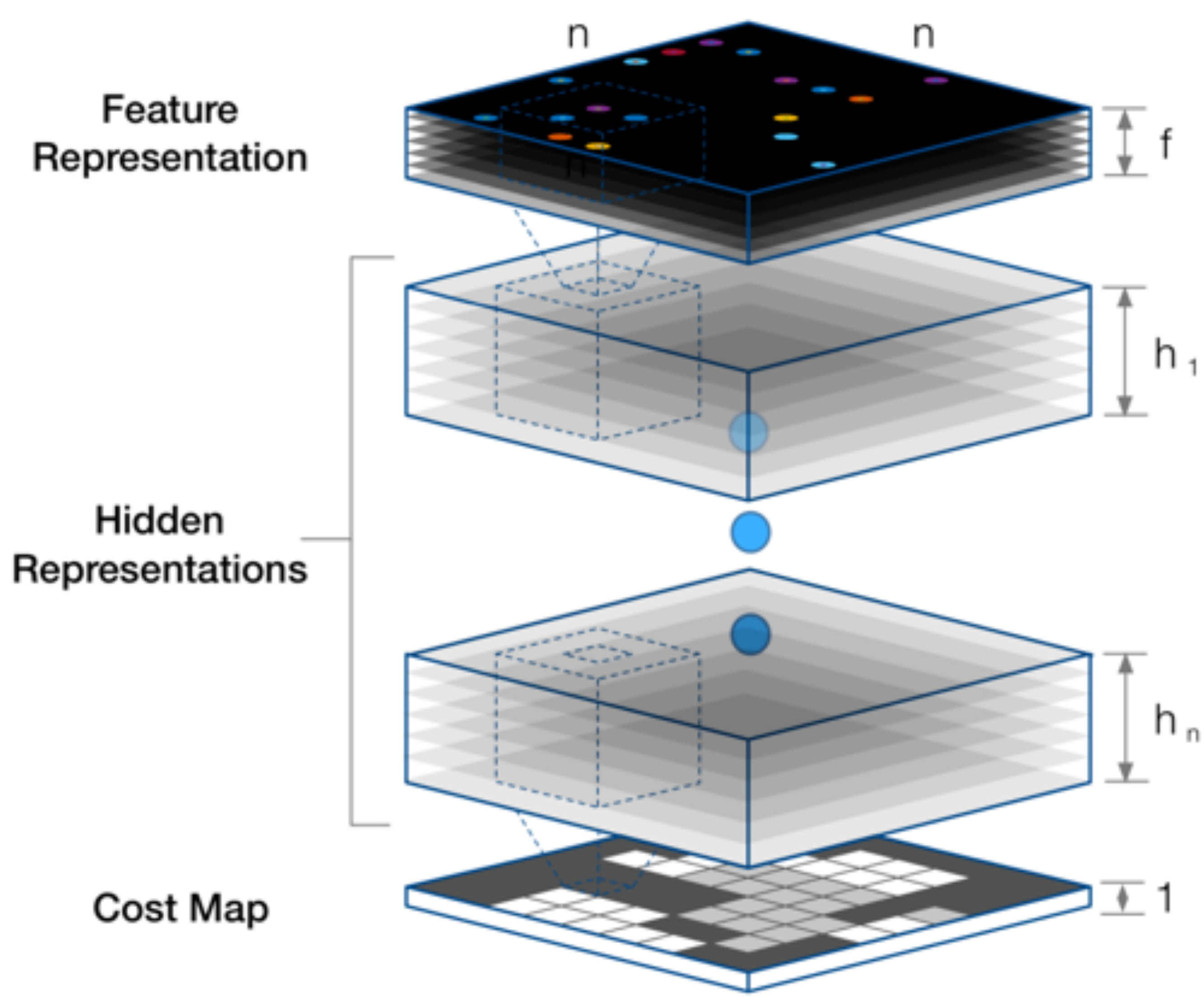
Watch This: Scalable Cost-Function Learning for Path Planning in Urban Environments

Markus Wulfmeier[1], Dominic Zeng Wang[1] and Ingmar Posner[1]

IROS 2016

Case Study: MaxEnt Deep IRL

MaxEnt IRL with known dynamics (tabular setting), neural net cost



Algorithm 1 Maximum Entropy Deep IRL

Input: $\mu_D^a, f, S, A, T, \gamma$

Output: optimal weights θ^*

$\theta^1 = \text{initialise\_weights}()$

Iterative model refinement

for n = 1 : N **do**

$\quad r^n = \text{nn\_forward}(f, \theta^n)$

$\quad$ **Solution of MDP with current reward**

$\quad \pi^n = \text{approx\_value\_iteration}(r^n, S, A, T, \gamma)$

$\quad \mathbb{E}[\mu^n] = \text{propagate\_policy}(\pi^n, S, A, T)$

$\quad$ **Determine Maximum Entropy loss and gradients**

$\quad \mathcal{L}_D^n = \log(\pi^n) \times \mu_D^a$

$\quad \frac{\partial \mathcal{L}_D^n}{\partial r^n} = \mu_D - \mathbb{E}[\mu^n]$

$\quad$ **Compute network gradients**

$\quad \frac{\partial \mathcal{L}_D^n}{\partial \theta_D^n} = \text{nn\_backprop}(f, \theta^n, \frac{\partial \mathcal{L}_D^n}{\partial r^n})$

$\quad \theta^{n+1} = \text{update\_weights}(\theta^n, \frac{\partial \mathcal{L}_D^n}{\partial \theta_D^n})$

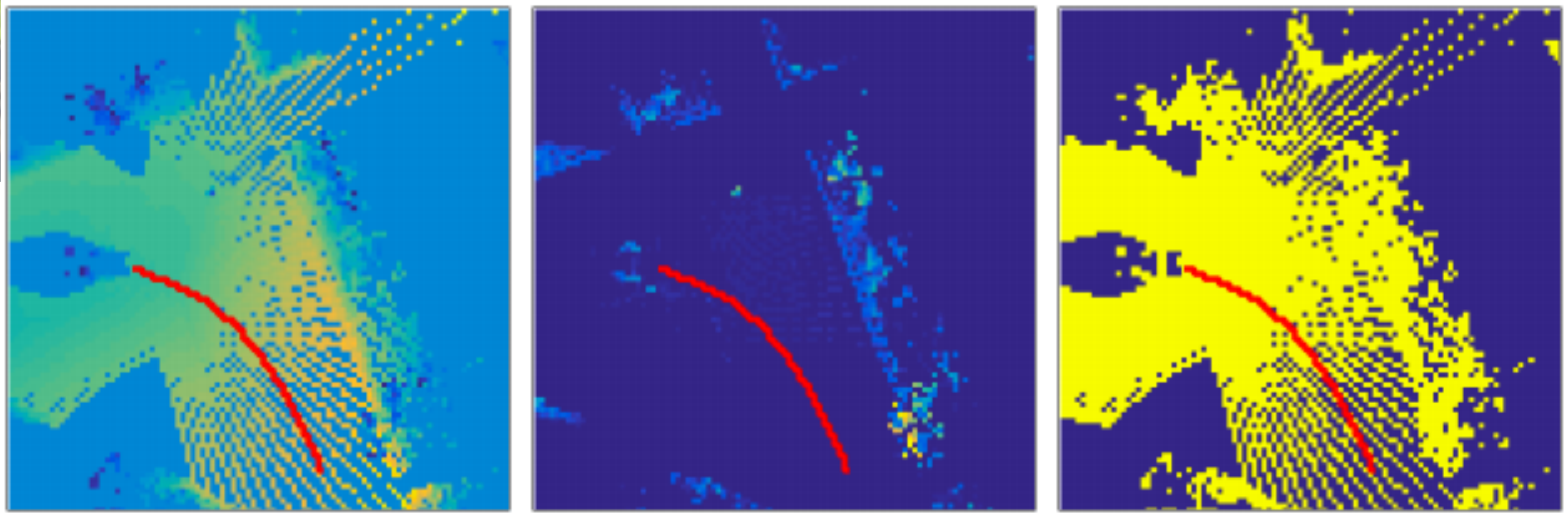
end for

Need to iteratively solve MDP for every weight update

Case Study: MaxEnt Deep IRL

MaxEnt IRL with known dynamics (tabular setting), neural net cost

demonstrations



mean height | height variance | cell visibility

120km of demonstration data

manually designed cost:

test-set trajectory prediction:

Prediction metrics	Standard FCN	Pooling FCN	MS FCN	Manual CF
NLL	69.35	69.73	65.39	78.13
MHD	0.221	0.230	0.200	0.284

MHD: modified Hausdorff distance

Case Study: MaxEnt Deep IRL

MaxEnt IRL with known dynamics (tabular setting), neural net cost

Strengths

- scales to neural net costs
- efficient enough for real robots

Limitations

- still need to repeatedly solve the MDP
- assumes known dynamics

What about unknown dynamics?

Whiteboard

Case Study: Guided Cost Learning

Guided Cost Learning: Deep Inverse Optimal Control via Policy Optimization

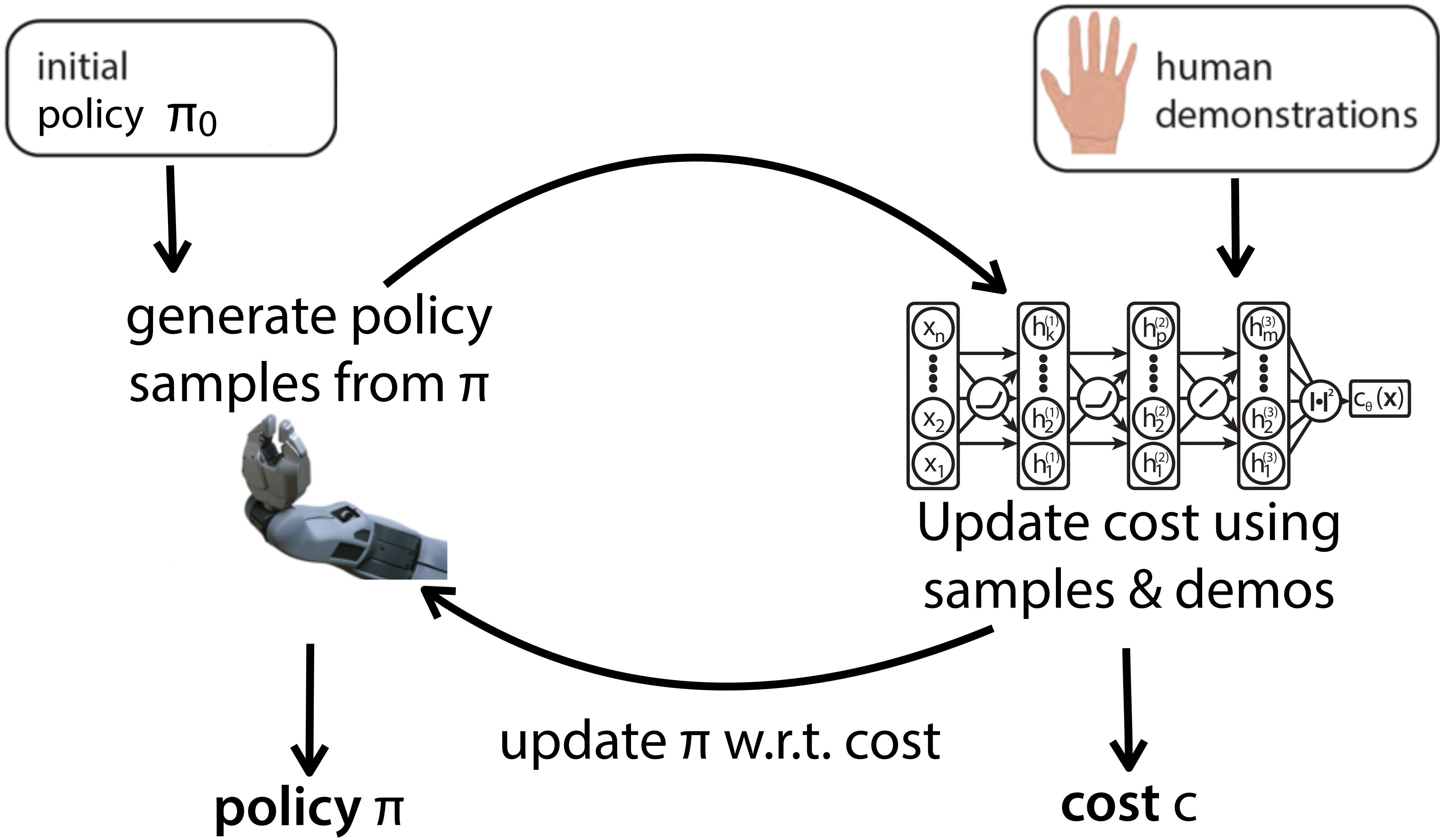
Chelsea Finn CBFINN@EECS.BERKELEY.EDU
Sergey Levine SVLEVINE@EECS.BERKELEY.EDU
Pieter Abbeel PABBEEL@EECS.BERKELEY.EDU

ICML 2016

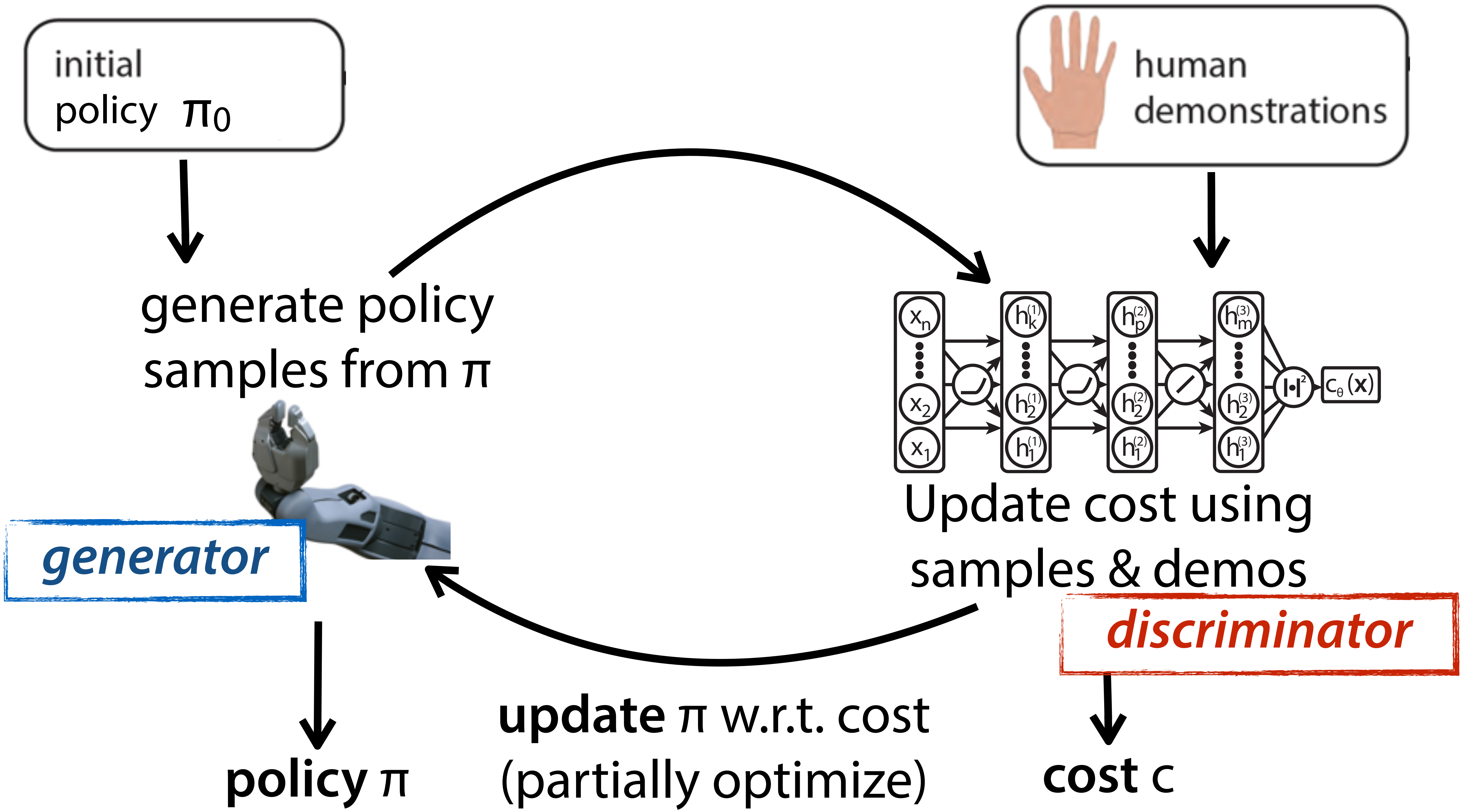
Goals:

- remove need to solve MDP in the inner loop
- be able to handle unknown dynamics
- handle continuous state & actions spaces

guided cost learning algorithm



guided cost learning algorithm



update cost in inner loop of policy optimization

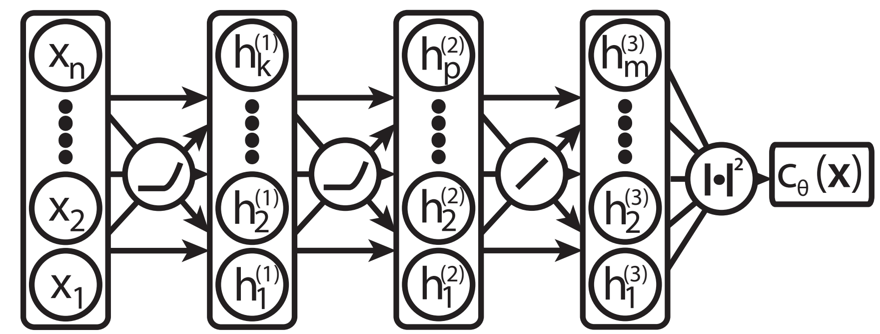
guided cost learning algorithm

initial
policy π_0

human
demonstrations

generate policy
samples from π

generator



Update cost using
samples & demos

discriminator

policy π

update π w.r.t. cost
(partially optimize)

cost c

Ho et al., ICML '16, NIPS '16

What about unknown dynamics?

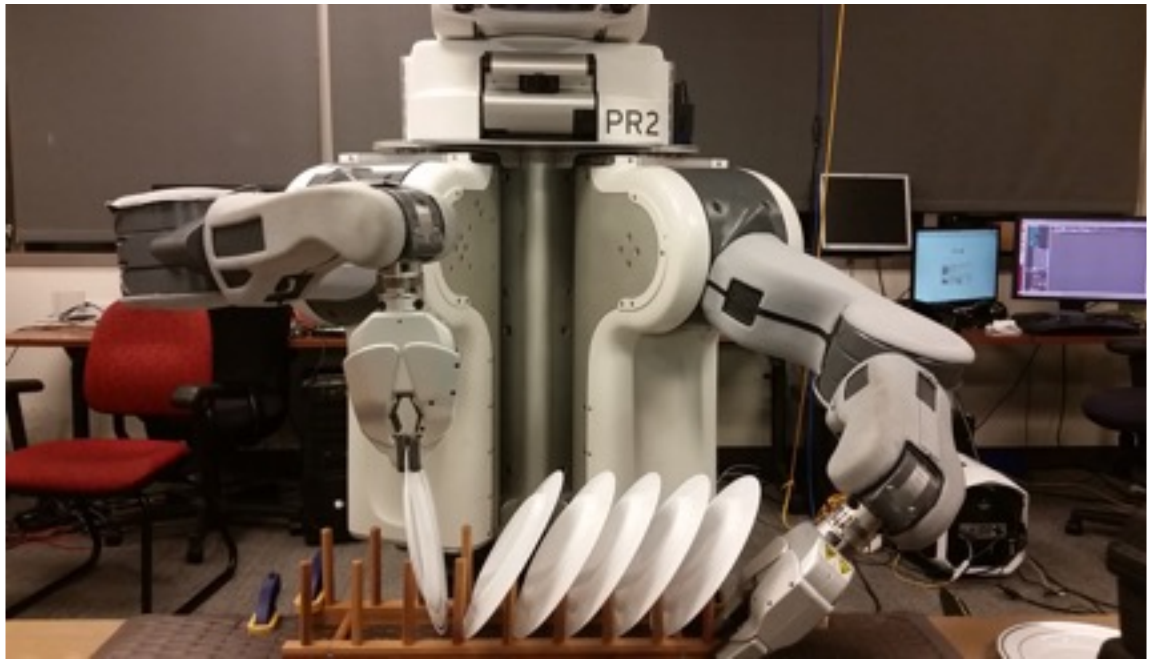
Adaptive importance sampling

1: Initialize $q_k(\tau)$ as either a random initial controller or from demonstrations
2: **for** iteration $i = 1$ to I **do**
3: Generate samples $\mathcal{D}_{\text{traj}}$ from $q_k(\tau)$
4: Append samples: $\mathcal{D}_{\text{samp}} \leftarrow \mathcal{D}_{\text{samp}} \cup \mathcal{D}_{\text{traj}}$
5: Use $\mathcal{D}_{\text{samp}}$ to update cost c_θ using gradient descent
6: Update $q_k(\tau)$ using $\mathcal{D}_{\text{traj}}$ and the method from (Levine & Abbeel, 2014) to obtain $q_{k+1}(\tau)$
7: **end for**
8: **return** optimized cost parameters θ and trajectory distribution $q(\tau)$

GCL Experiments

Real-world Tasks

dish placement



state includes goal plate pose

pouring almonds

state includes unsupervised visual features [Finn et al. '16]

action: joint torques

Comparisons

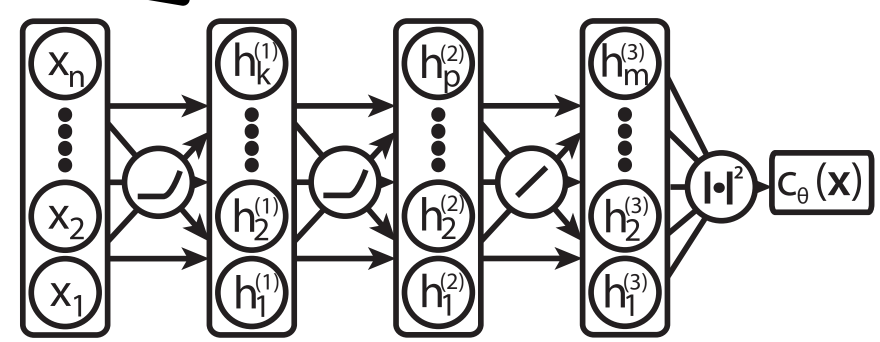
Path Integral IRL
(Kalakrishnan et al. '13)

Relative Entropy IRL
(Boularias et al. '11)

initial
distribution q_0

generate policy
samples from q

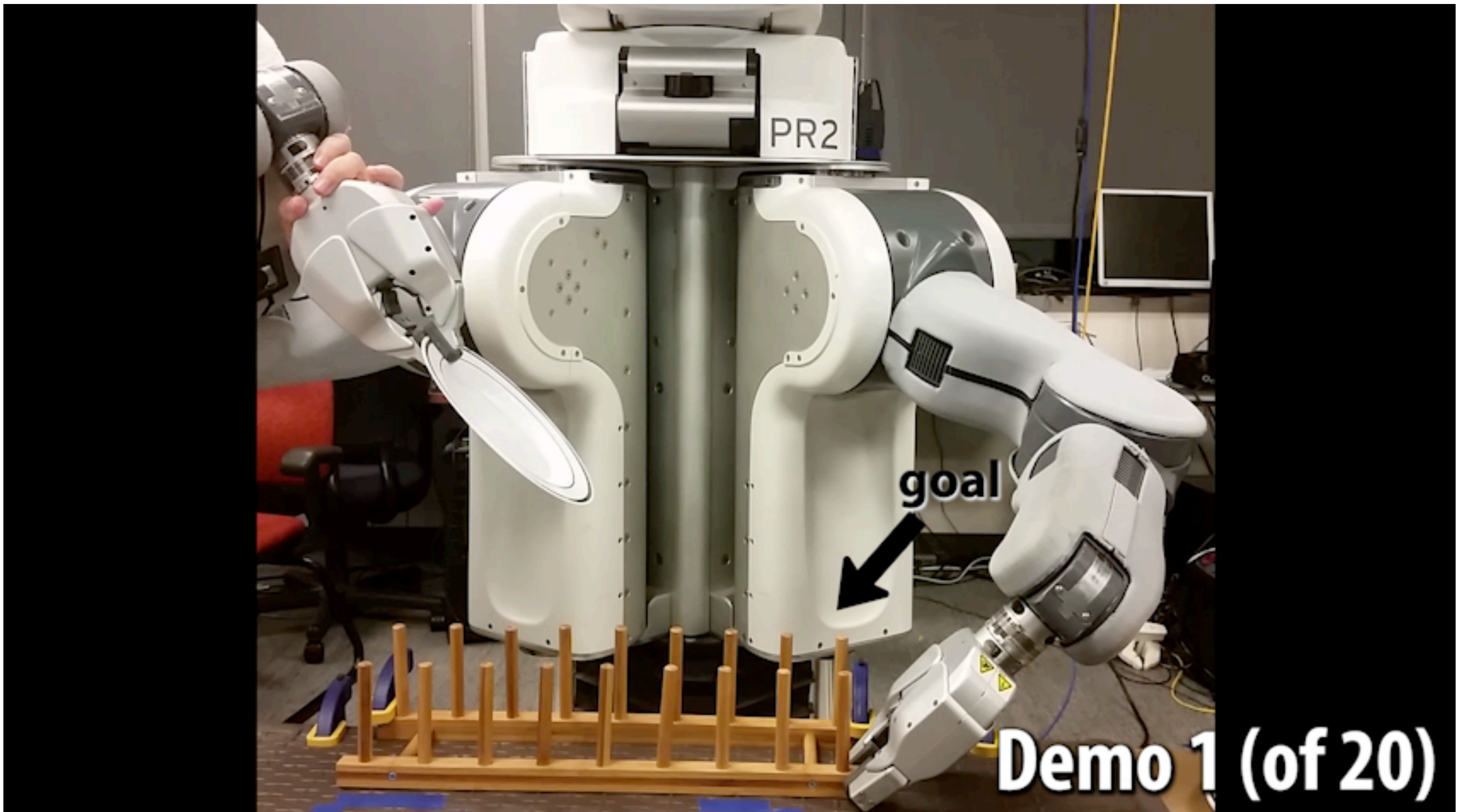
human
demonstrations



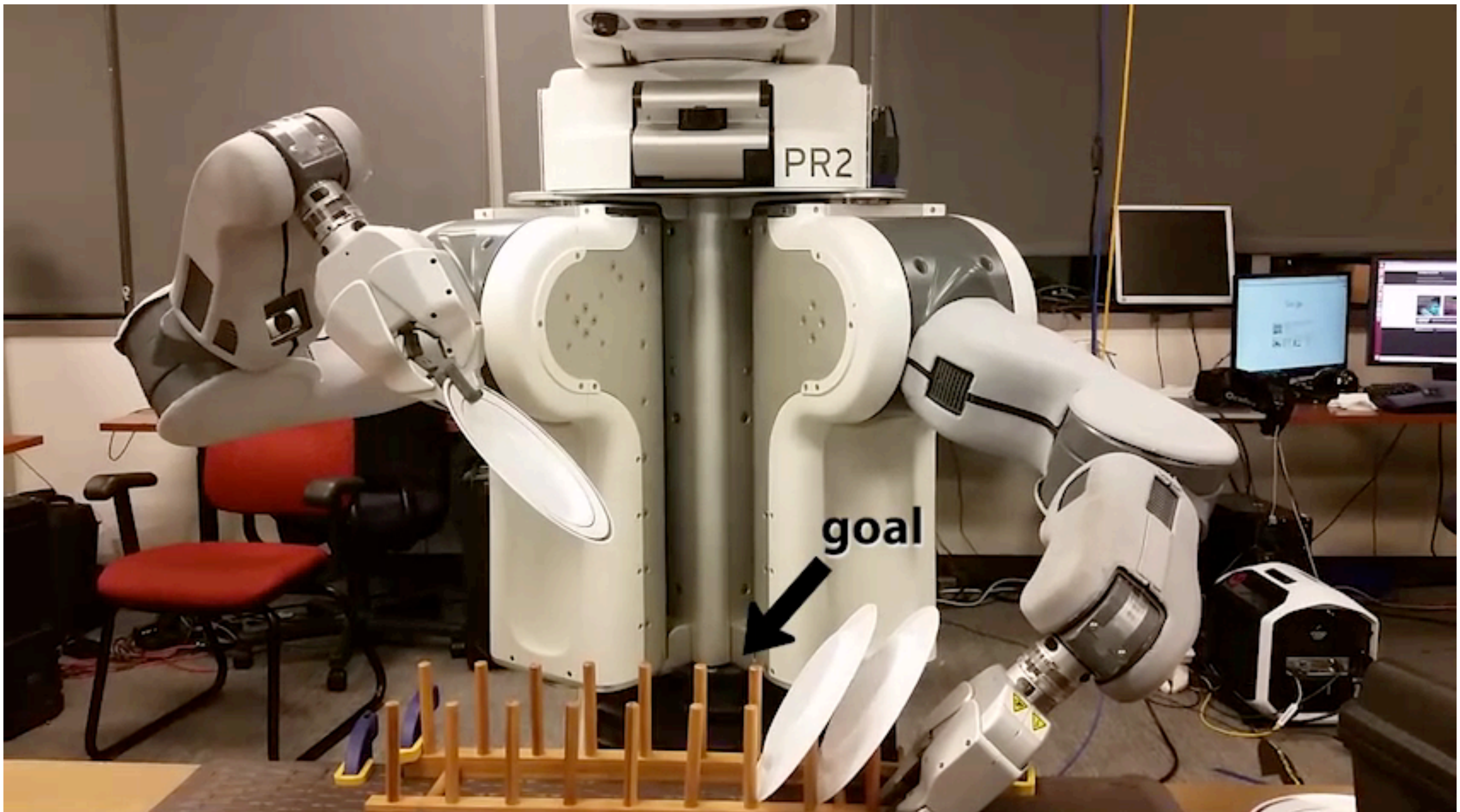
Update cost using
samples & demos

cost c

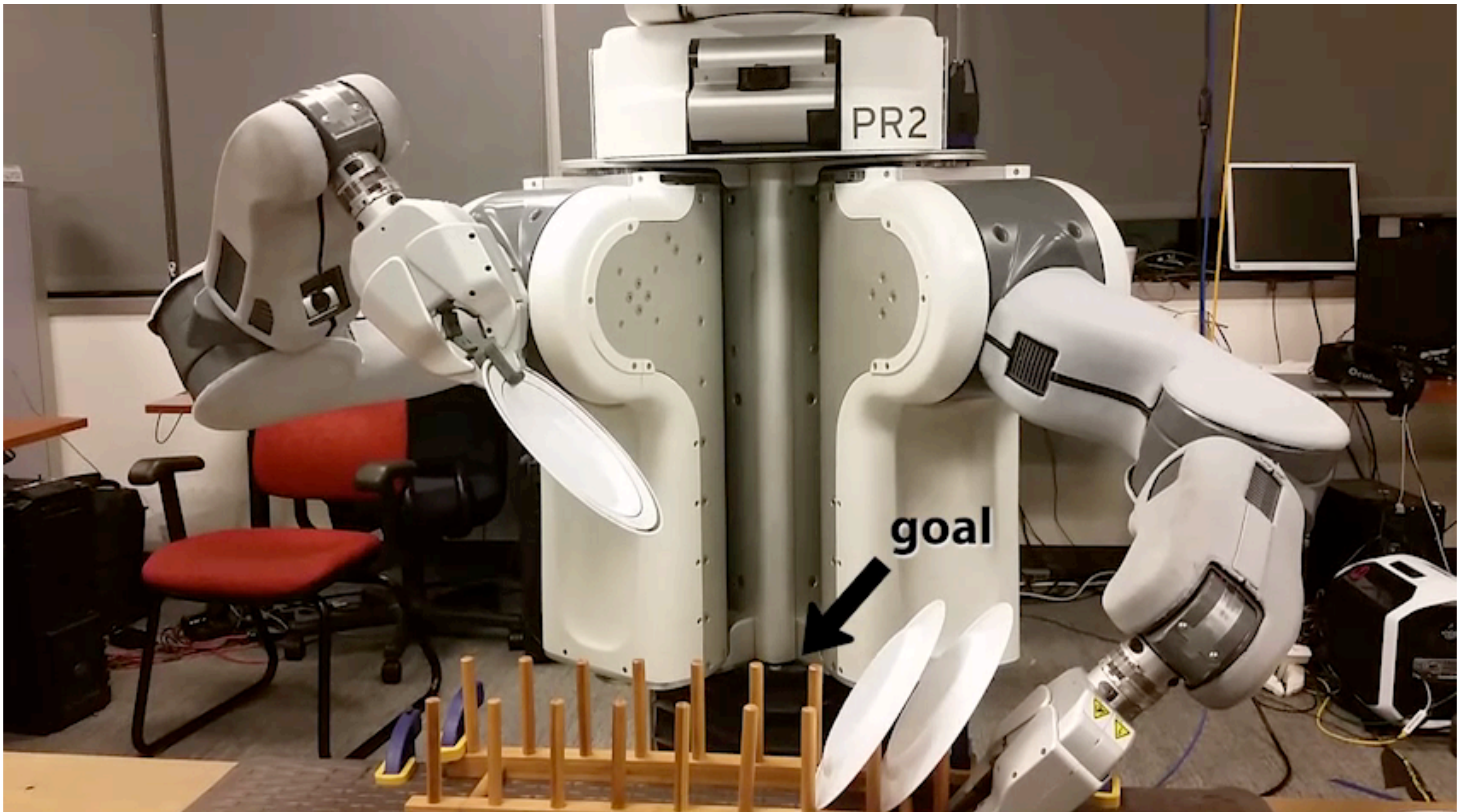
Dish placement, demos



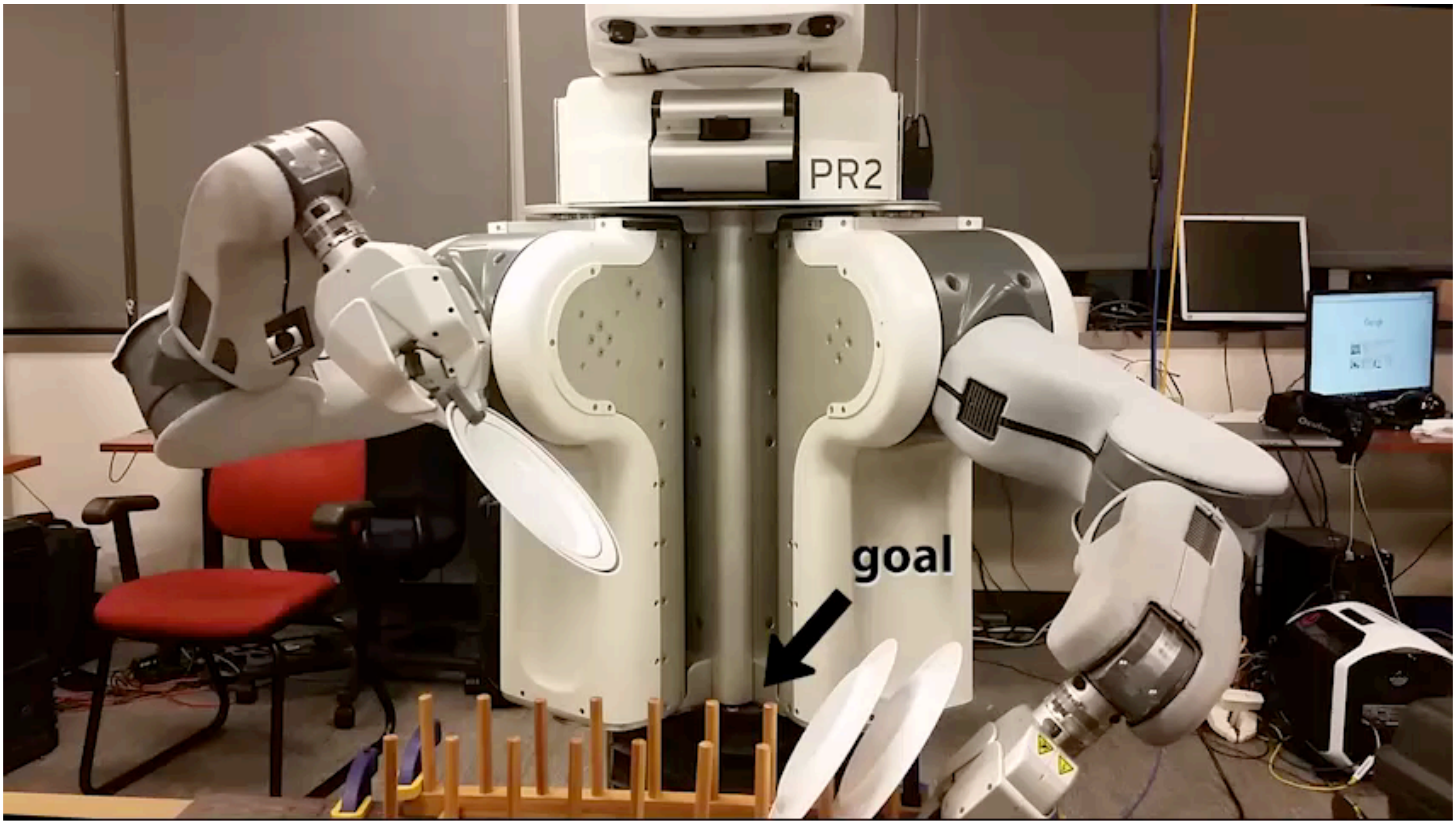
Dish placement, standard cost



Dish placement, RelEnt IRL

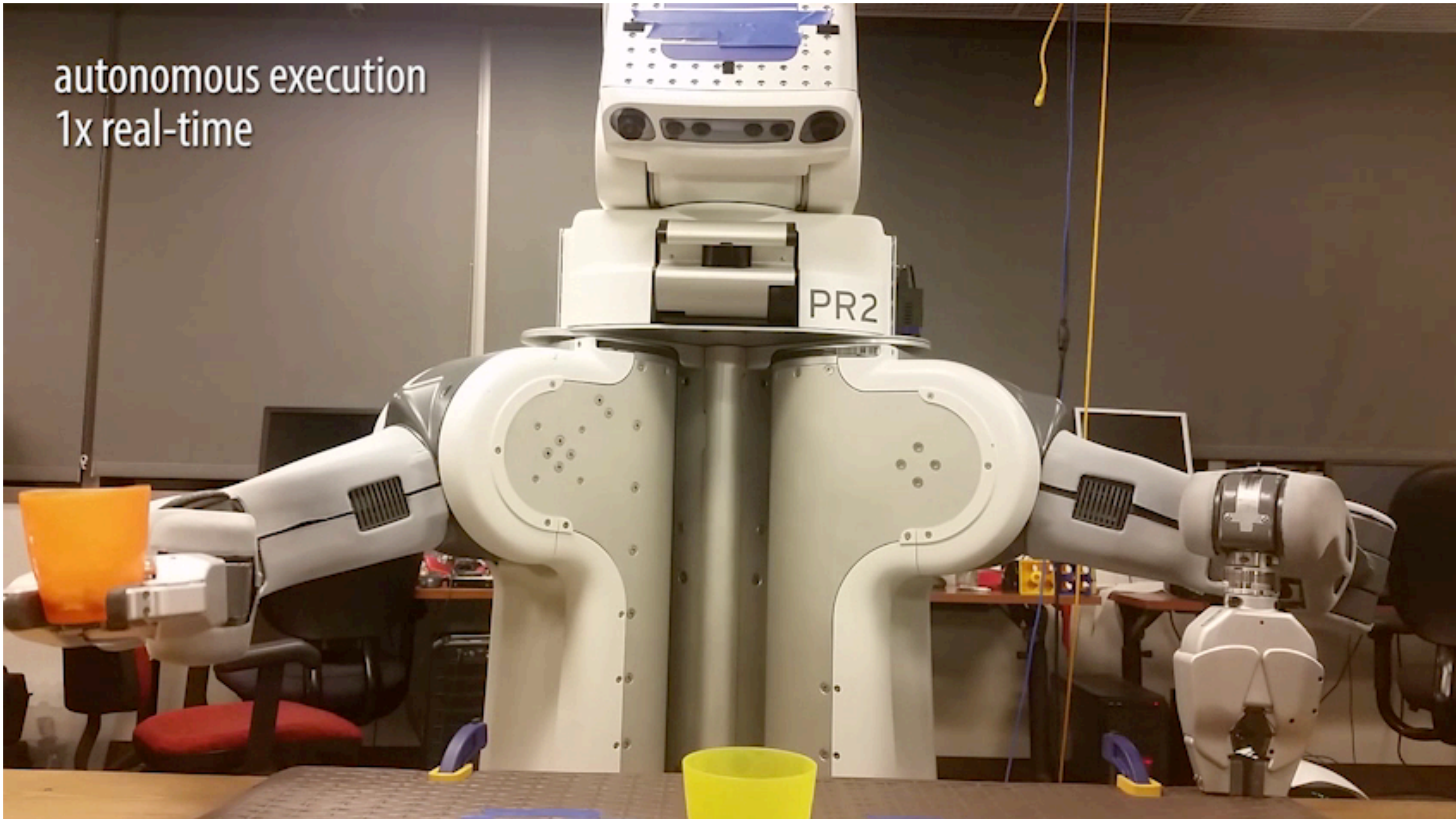


Dish placement, GCL policy

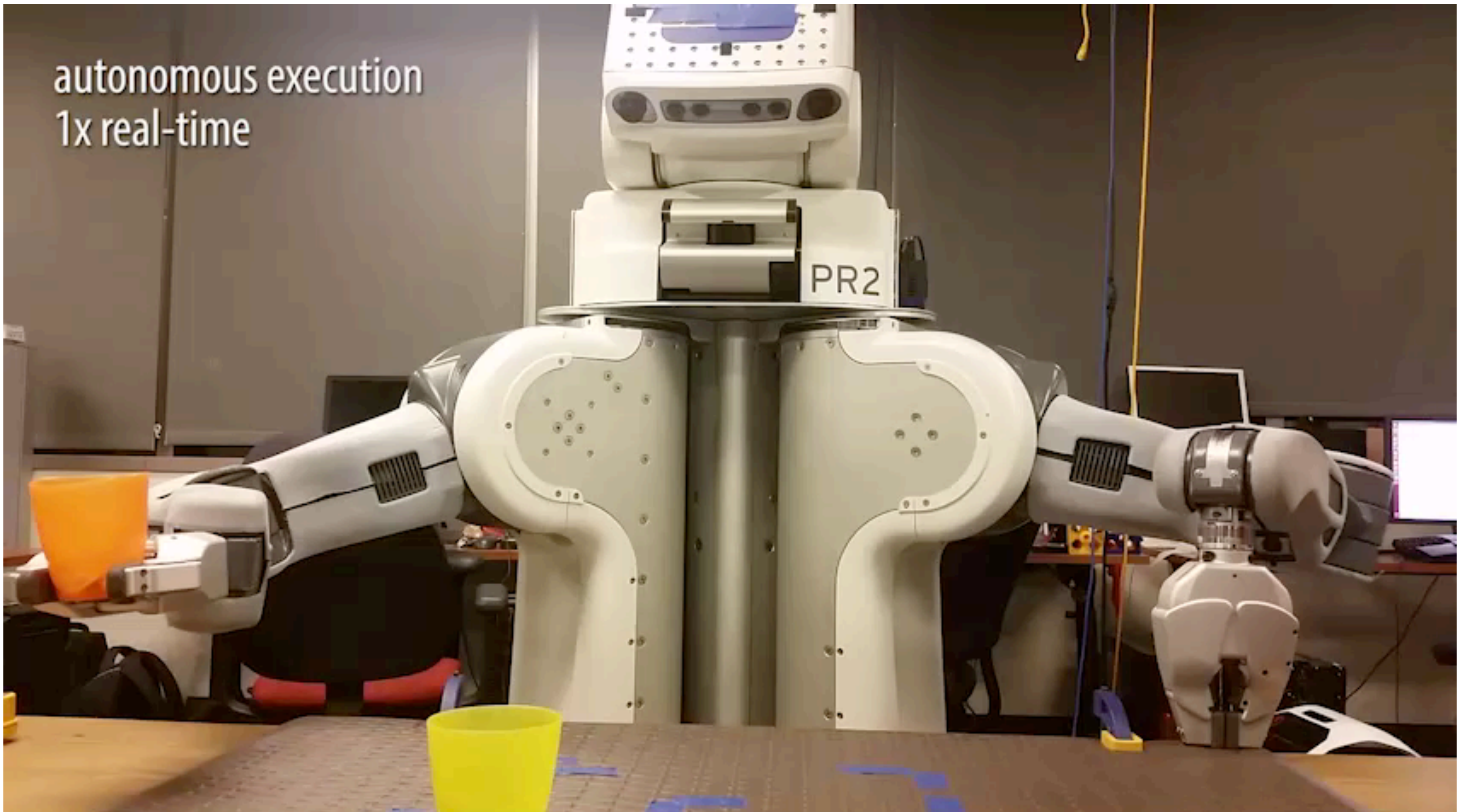


Pouring, demos

Pouring, RelEnt IRL



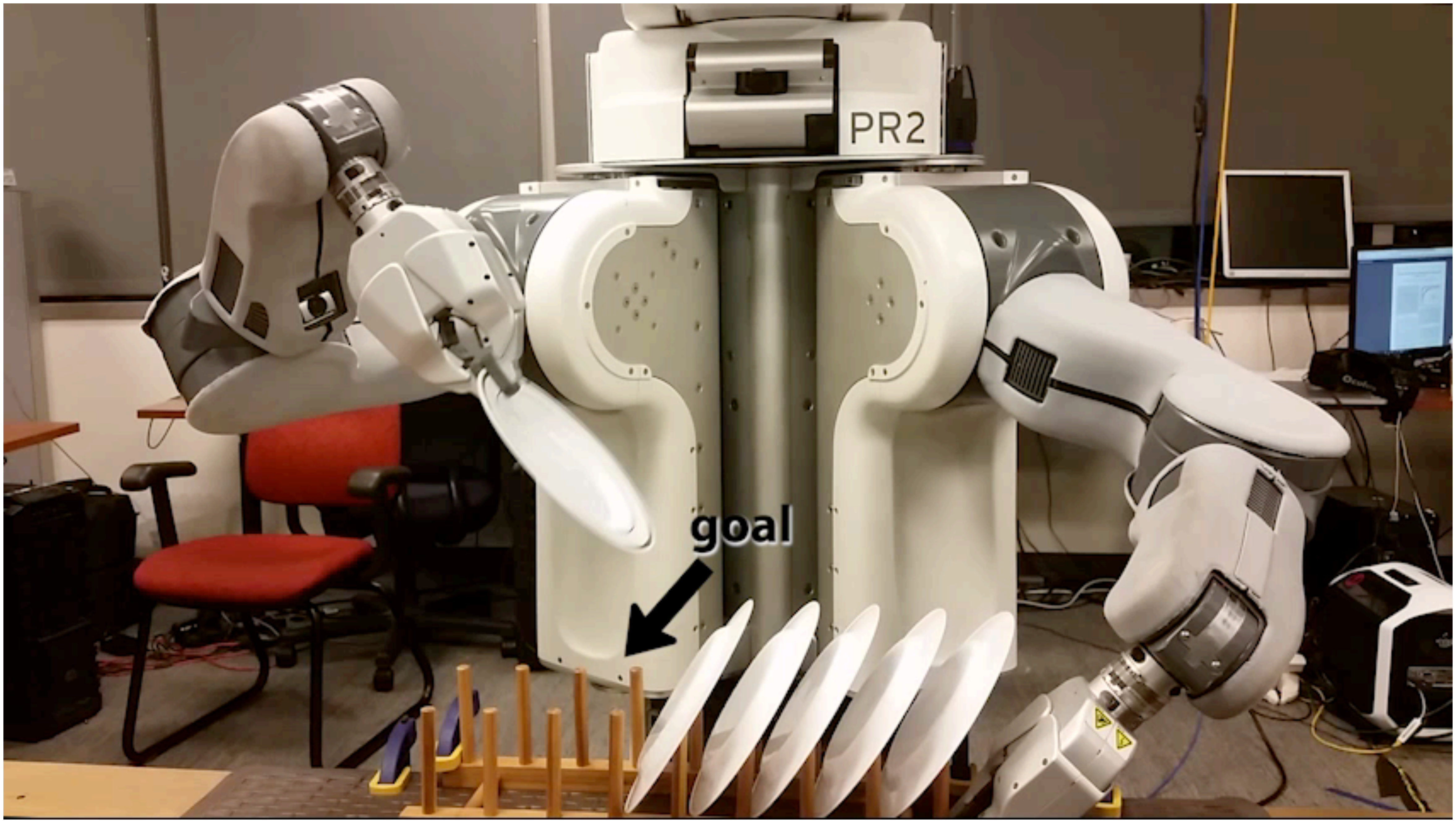
Pouring, GCL policy



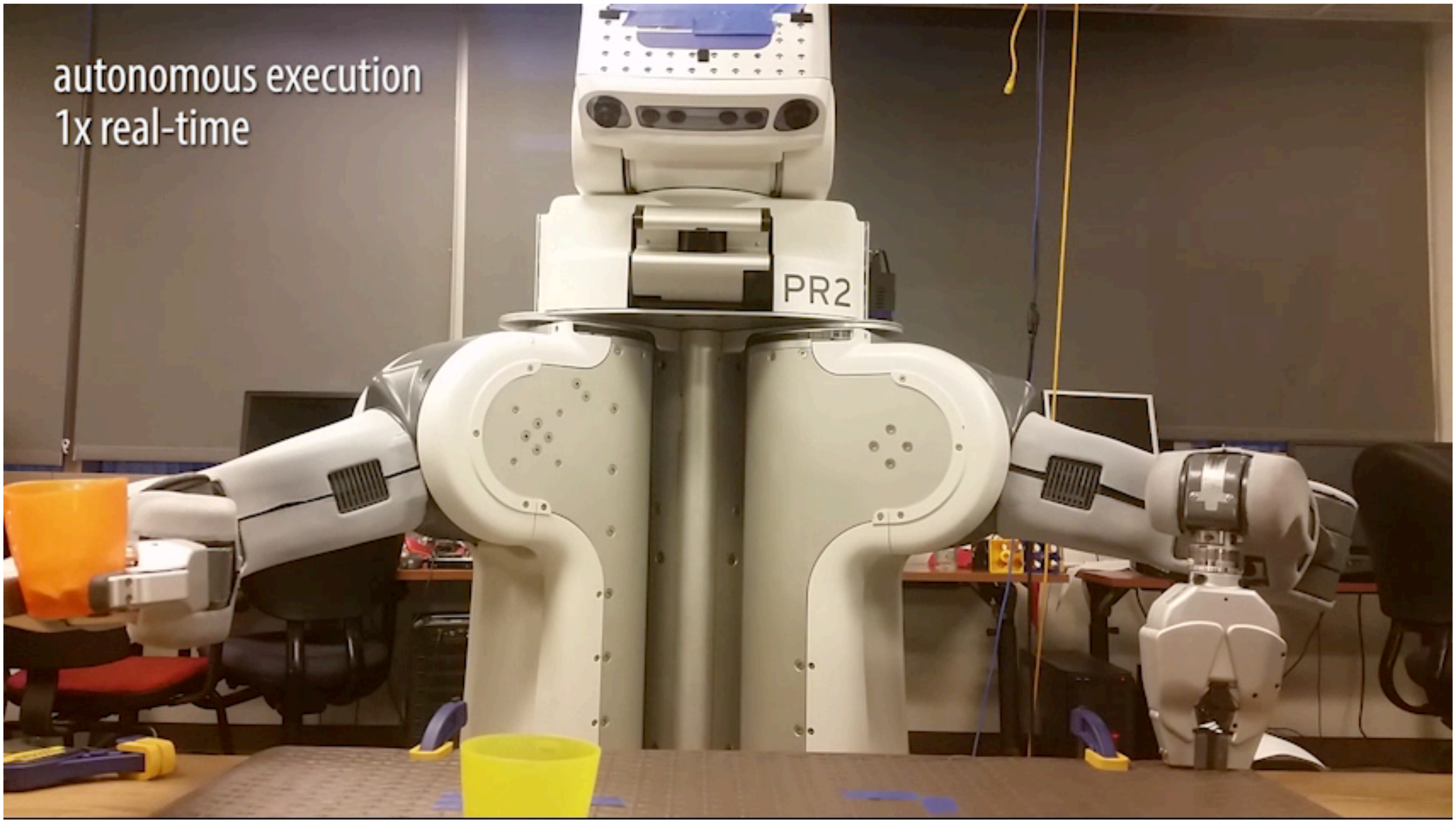
Conclusion: We can recover successful policies for new positions.

Is the cost function also useful for new scenarios?

Dish placement - GCL reopt.



Pouring - GCL reopt.



Note: normally the GAN discriminator is discarded

Case Study: Generative Adversarial Imitation Learning

Generative Adversarial Imitation Learning

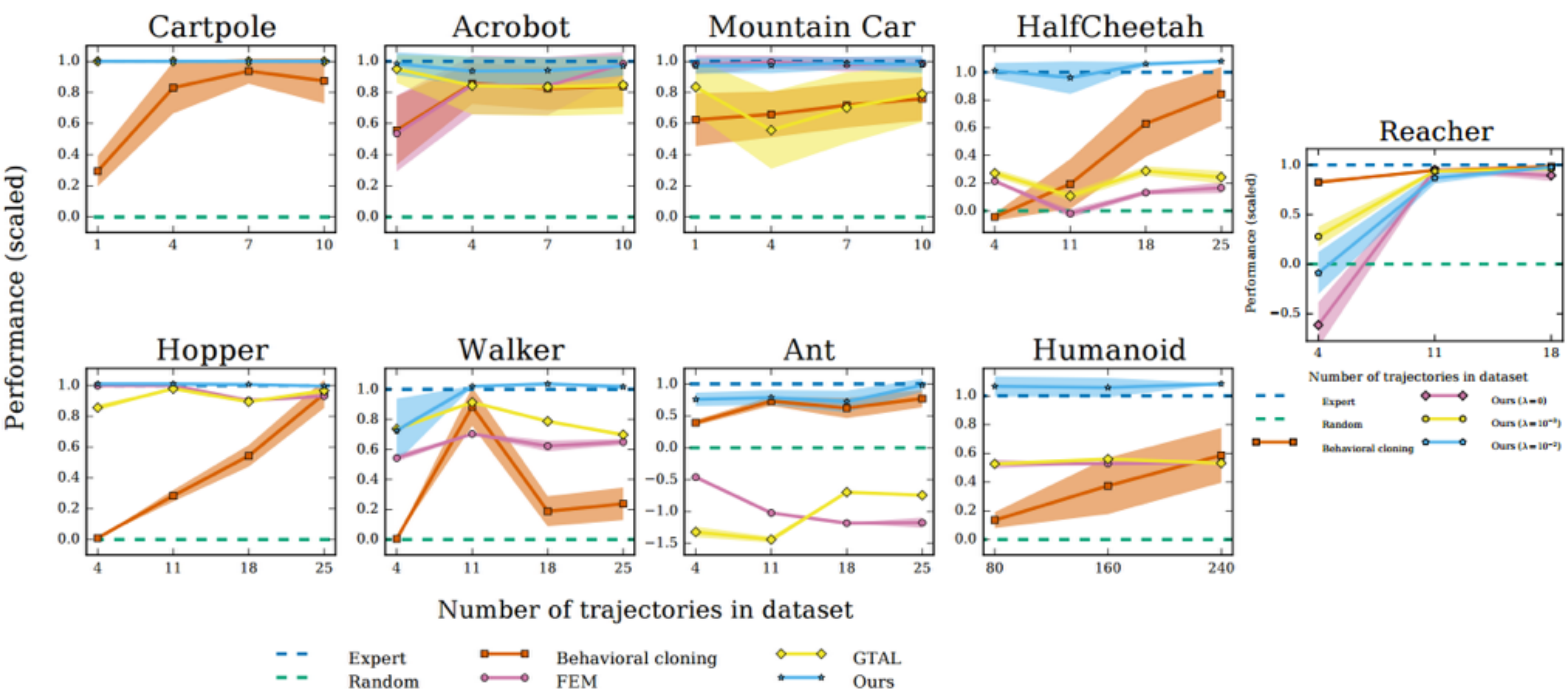
Jonathan Ho
Stanford University
hoj@cs.stanford.edu

Stefano Ermon
Stanford University
ermon@cs.stanford.edu

NIPS 2016

Case Study: Generative Adversarial Imitation Learning

- demonstrations from TRPO-optimized policy
- use TRPO as a policy optimizer
- OpenAI gym tasks



Guided Cost Learning & Generative Adversarial Imitation Learning

Strengths

- can handle unknown dynamics
- scales to neural net costs
- efficient enough for real robots

Limitations

- adversarial optimization is hard
- can't scale to raw pixel observations of demos
- demonstrations typically collected with kinesthetic teaching or teleoperation (first person)

Next Time:

Back to forward RL (advanced policy gradients)

IOC is under-defined

need regularization:

- encourage slowly changing cost

$$g_{\mathrm{lcr}}(\tau) = \sum_{x_t \in \tau} \left[(c_\theta(x_{t+1}) - c_\theta(x_t)) - (c_\theta(x_t) - c_\theta(x_{t-1})) \right]^2$$

- cost of demos decreases strictly monotonically in time

$$g_{\mathrm{mono}}(\tau) = \sum_{x_t \in \tau} \left[\max(0, c_\theta(x_t) - c_\theta(x_{t-1}) - 1) \right]^2$$

Regularization ablation