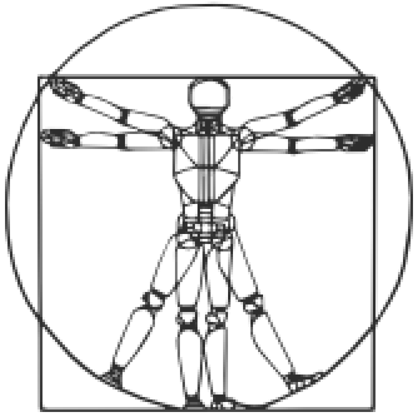


CSC2626
Imitation Learning for Robotics

Florian Shkurti

Week 7: Shared Autonomy and Human-in-the-Loop Learning

Today's agenda

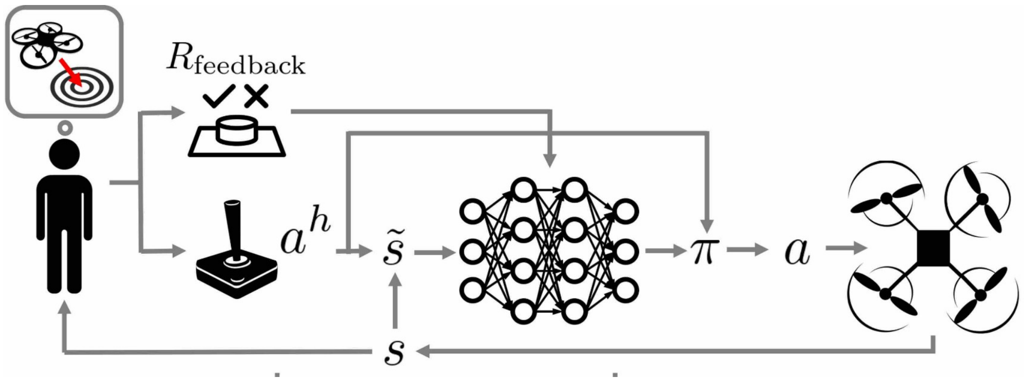
- Shared autonomy with human in the loop in deep RL
- Hindsight optimization and interactive goal prediction
- Relaxed inverse kinematics for fluid interaction with robot arms

Acknowledgments
Today's slides are based on student presentations from 2019 by: Andrei Barsan, Bin Yang, and Tingwu Wang

Shared Autonomy via Deep Reinforcement Learning

Siddharth Reddy, Anca Dragan, Sergey Levine UC Berkeley

Presented by Ioan Andrei Bârsan on February 22, 2019
iab@cs.toronto.edu



Key Question

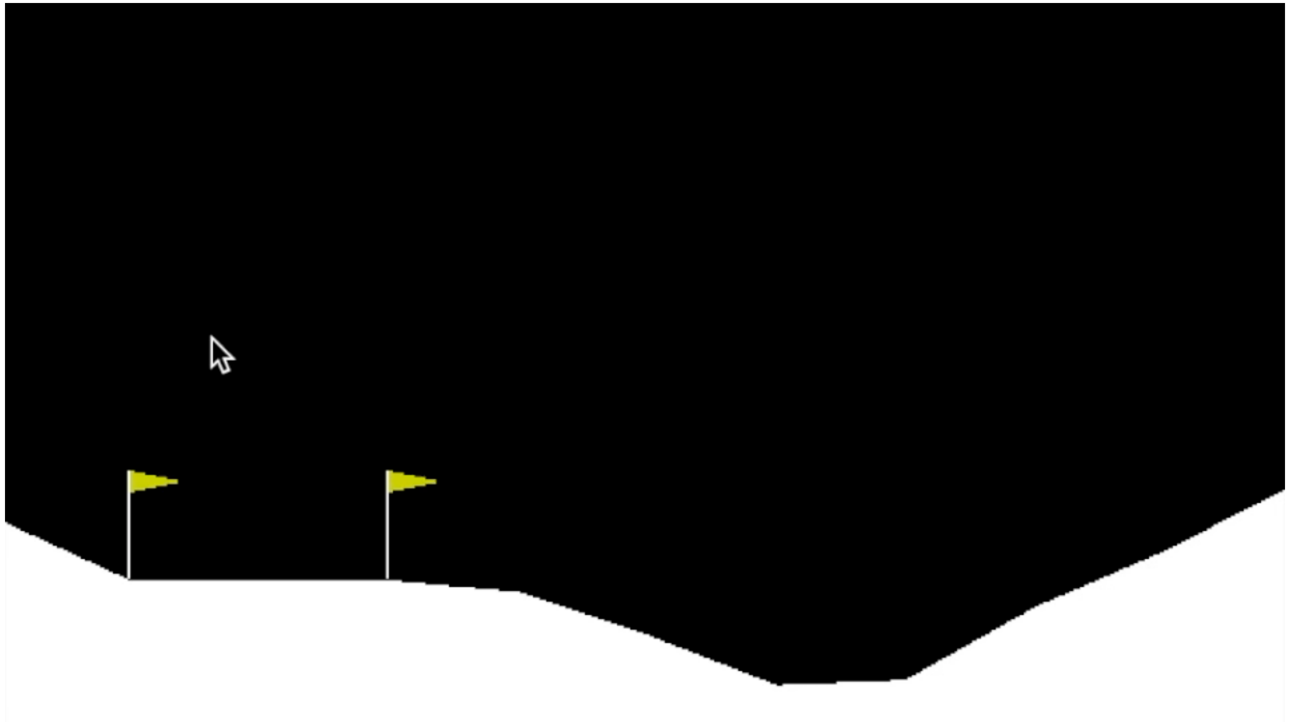
How can a robot **collaborating** with a human infer the human’s goals with as few **assumptions** as possible?

Motivation

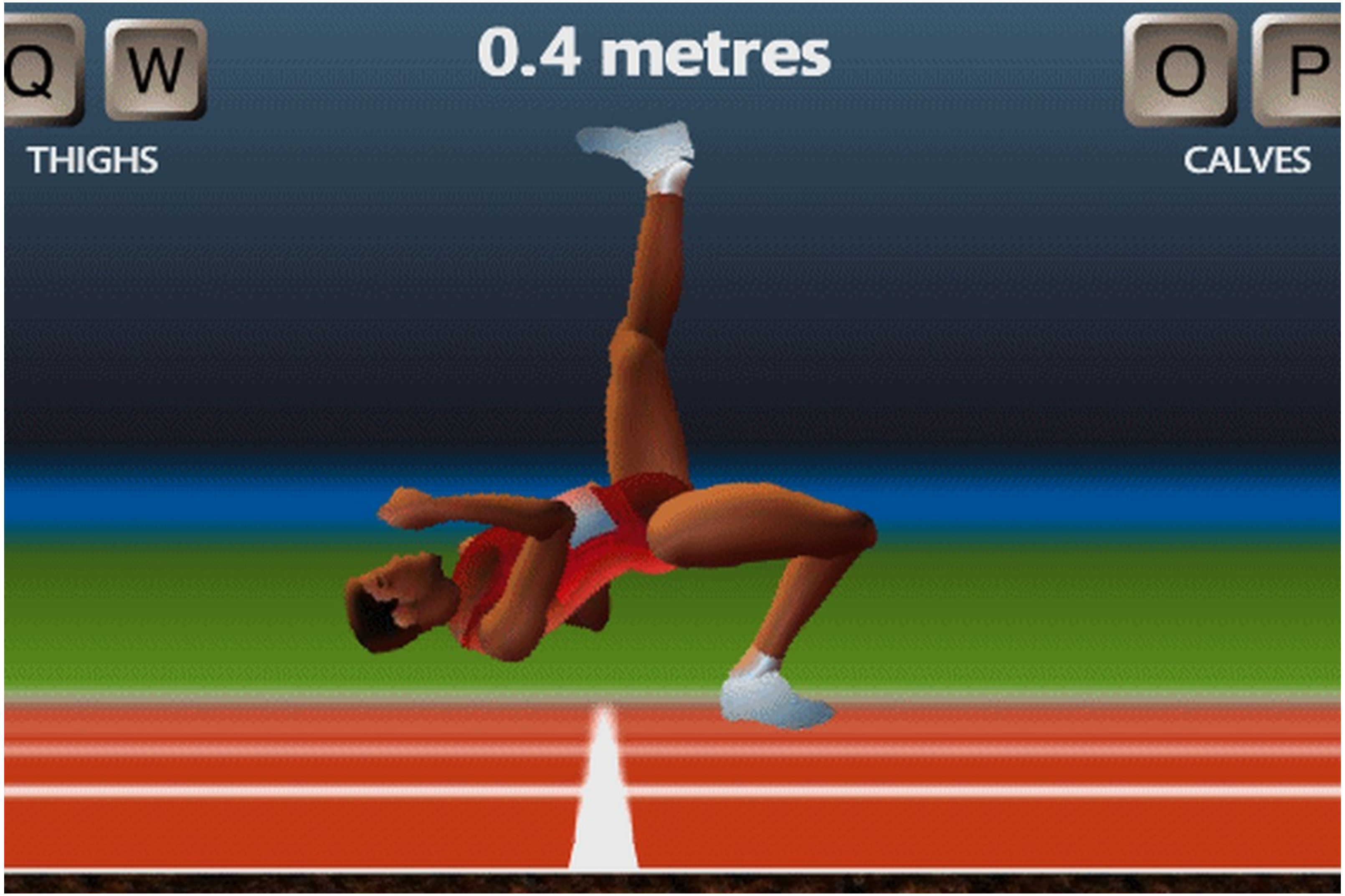
- **Hard:** Actuating a robot with many DoF and/or unfamiliar dynamics.
- **Hard:** Specifying a goal formally (e.g., coordinates).
- **Easy:** Demonstrating the goal indirectly.
  - …let the machine figure out what I want!

**Image source: "Multihierarchical Interactive Task Planning. Application to Mobile Robotics"
Galindo et al., 2008**

Motivation: Unknown Dynamics are Hard for Humans



It can get even worse than Lunar Lander…



www.foddy.net/Athletics.html
or
Google "qwop"

Challenges

- **Recall:** Want to demonstrate the goal indirectly with **minimal assumptions**.
  - → We expect the computer to start helping **while it is still learning**.
- **Challenge #1:** How to actually infer user's goal?
- **Challenge #2:** How can we learn this online with low latency?

Main Hypothesis

Shared autonomy can improve human performance without any assumptions about:

(1) dynamics,

(2) the human's policy,

(3) the nature of the goal.

Formulation: Reward

$$R(s, a, s') = \underbrace{R_{\text{general}}(s, a, s')}_{\text{known}} + \underbrace{R_{\text{feedback}}(s, a, s')}_{\text{unknown, but observed}}$$

Agent's reward (what we want to maximize)

Handcrafted "common sense" knowledge: do not crash, do not tip, etc.

Stuff inferred from the human (Main focus of this paper!)

Formulation

$$\underbrace{R_{\text{feedback}}(s, a, s')}_{\text{unknown, but observed}}$$

- The authors introduce three variants of their method:

Needs virtual "user"!

1. Known goal space, known user policy.
2. Known goal space, unknown user policy.
3. Unknown goal space, unknown user policy.

Fewer Assumptions

The Method

- Based on Q-Learning.
- User input has **two** roles:
    1. A **prior policy** we should fine-tune.
    2. A sensor which can be used to decode the **goal**.
- Short version: Like Q-Learning, but execute closest high-value action to the user's input, instead of highest-value action.

The Method (Continued)

Interesting part!

$$\pi_\alpha(a \mid \tilde{s}, a^h) = \delta\left(a = \underset{\{a: Q'(\tilde{s},a) \geq (1-\alpha) Q'(\tilde{s},a^*)\}}{\arg\max} f(a, a^h)\right),$$

Algorithm 1 Human-in-the-loop deep Q-learning

Standard Q-Learning Initialization

Initialize target action-value function $\hat{Q}$ with weights $\theta^- = \theta$
for episode $= 1, M$ **do**
  for $t = 1, T$ **do**
    Sample action $a_t \sim \pi_\alpha(a_t \mid \tilde{s}_t, a_t^h)$ using equation 3
    Execute action a_t and observe $(\tilde{s}_{t+1}, a_{t+1}^h, r_t)$
    Store transition $(\tilde{s}_t, a_t, r_t, \tilde{s}_{t+1})$ in $\mathcal{D}$
    if $\tilde{s}_{t+1}$ is terminal **then**
      for $k = 1$ to K **do** ▷ training loop

Standard (Double) Q-Learning Training

        $\theta \leftarrow \theta - \eta \nabla_\theta \sum_j (y_j - Q(\tilde{s}_j, a_j; \theta))^2$
      end for
    end if
    Every C steps reset $\hat{Q} = Q$
  end for
end for

The Method (Continued)

$$\pi_\alpha(a \mid \tilde{s}, a^h) = \delta\left(a = \underset{\{a: Q'(\tilde{s},a) \geq (1-\alpha) Q'(\tilde{s},a^*)\}}{\arg\max} f(a, a^h)\right),$$

Algorithm 1 Human-in-the-loop deep Q-learning

Standard Q-Learning Initialization

Initialize target action-value function $\hat{Q}$ with weights $\theta^- = \theta$

for episode $= 1, M$ **do**

  for $t = 1, T$ **do**

    Sample action $a_t \sim \pi_\alpha(a_t \mid \tilde{s}_t, a_t^h)$ using equation 3

    Execute action a_t and observe $(\tilde{s}_{t+1}, a_{t+1}^h, r_t)$

    Store transition $(\tilde{s}_t, a_t, r_t, \tilde{s}_{t+1})$ in $\mathcal{D}$

    if $\tilde{s}_{t+1}$ is terminal **then**

      for $k = 1$ to K **do** ▷ training loop

        Sample minibatch $(\tilde{s}_j, a_j, r_j, \tilde{s}_{j+1})$ from $\mathcal{D}$

Standard Training

      end for

    end if

    Every C steps reset $\hat{Q} = Q$

  end for

end for

Maximize similarity to user action

…ensuring action is “close enough” to optimal one.

But where is $R_{feedback}$?

- The choice of $R_{feedback}$ determines what kind of **input** we give to the Q- Learning agent in addition to state!

1. Known goal space & user policy → exact goal.
2. Known goal space & unknown policy → predicted goal (pretrained LSTM).
3. Unknown goal space & policy → the user's input **(main focus)**

Input to RL Agent

Observed State

$$\tilde{s}_t = \begin{bmatrix} s_t \\ u_t \end{bmatrix}$$

(1) User Goal

User Information **(2) Predicted User Goal**

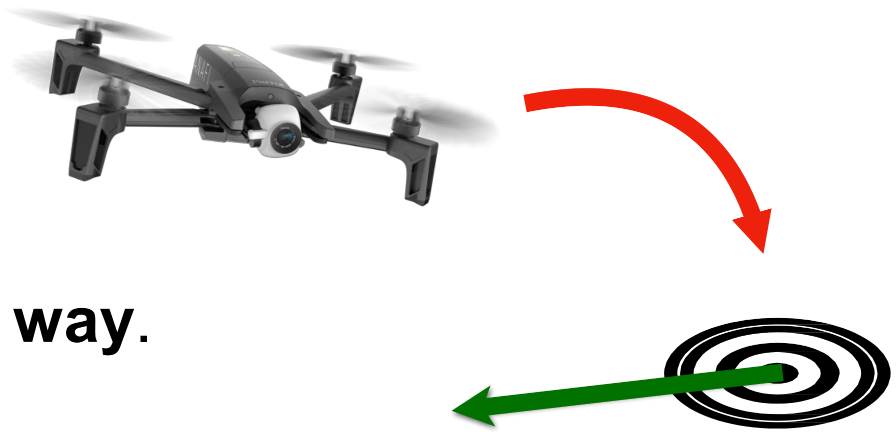
Combined Observation

(3) Raw User Input

Experiments

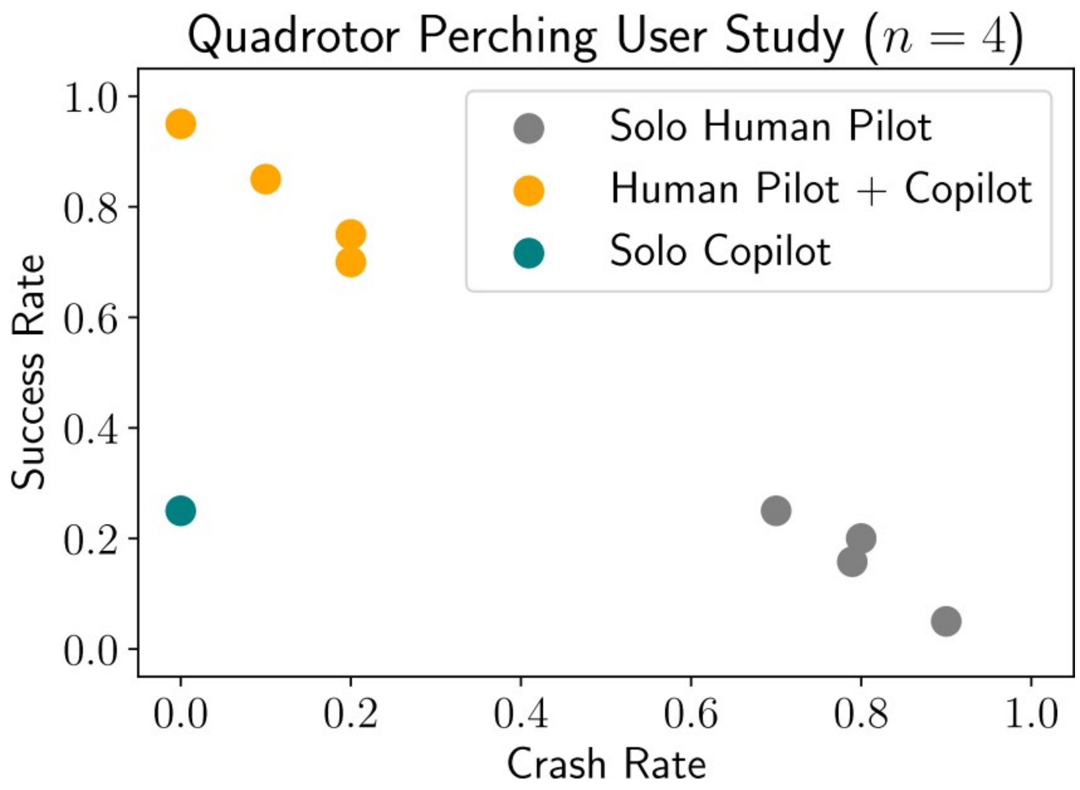
- **Virtual** experiments with Lunar Lander in OpenAI gym.
- **Physical** experiments with an actual drone.

Real-World Experiments



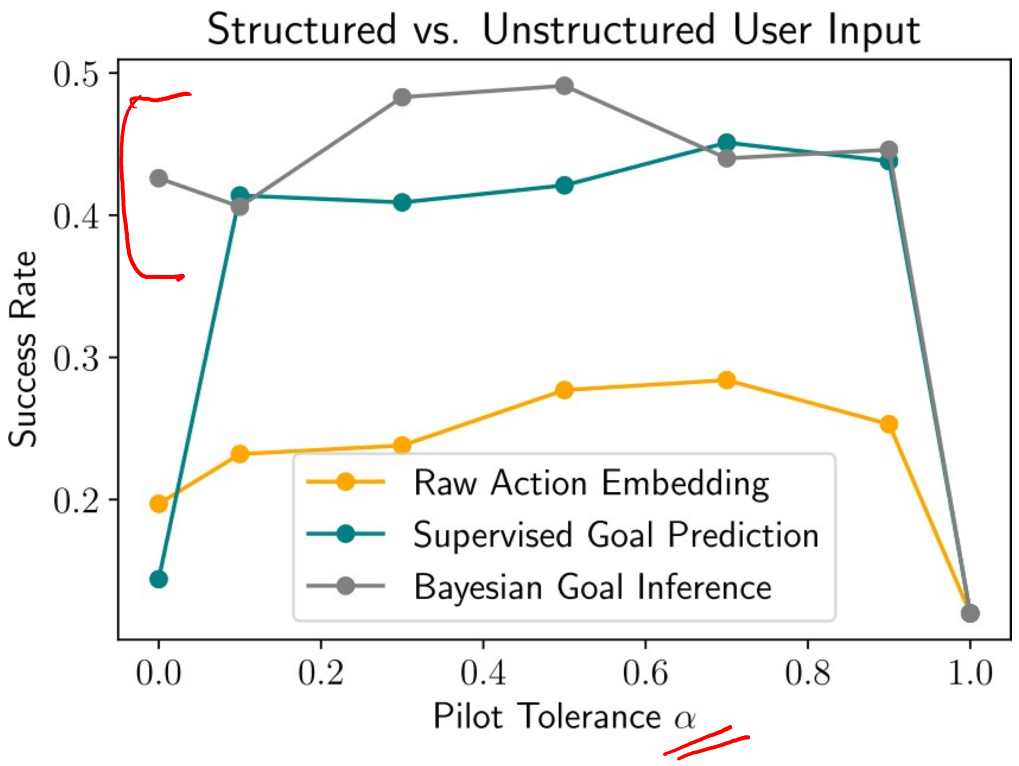
- **Goal:** Land drone on pad **facing a certain way**.
- **Pilot:** Human, knows target orientation.
- **Copilot:** Our Agent, knows where pad is, but not target orientation.

Real-World Results



Important observation: Only n = 4 humans in drone study. ☹

Experimental Results: Assumptions



- Higher alpha means we take any action. α = 1.0 means we ignore the pilot.
- Experimented in virtual environment.

Recap: Strengths

- Good results even when making no assumptions about user/goal.
- Writing is very clear!
- Possible applications in many fields, including e.g., **prosthetics, wheelchairs**.
- Source code released on GitHub!

Recap: Weaknesses

- User studies could have had more participants.
- Could have shown results on more Gym environments.
- Solution does not generalize to sophisticated long-term goals.

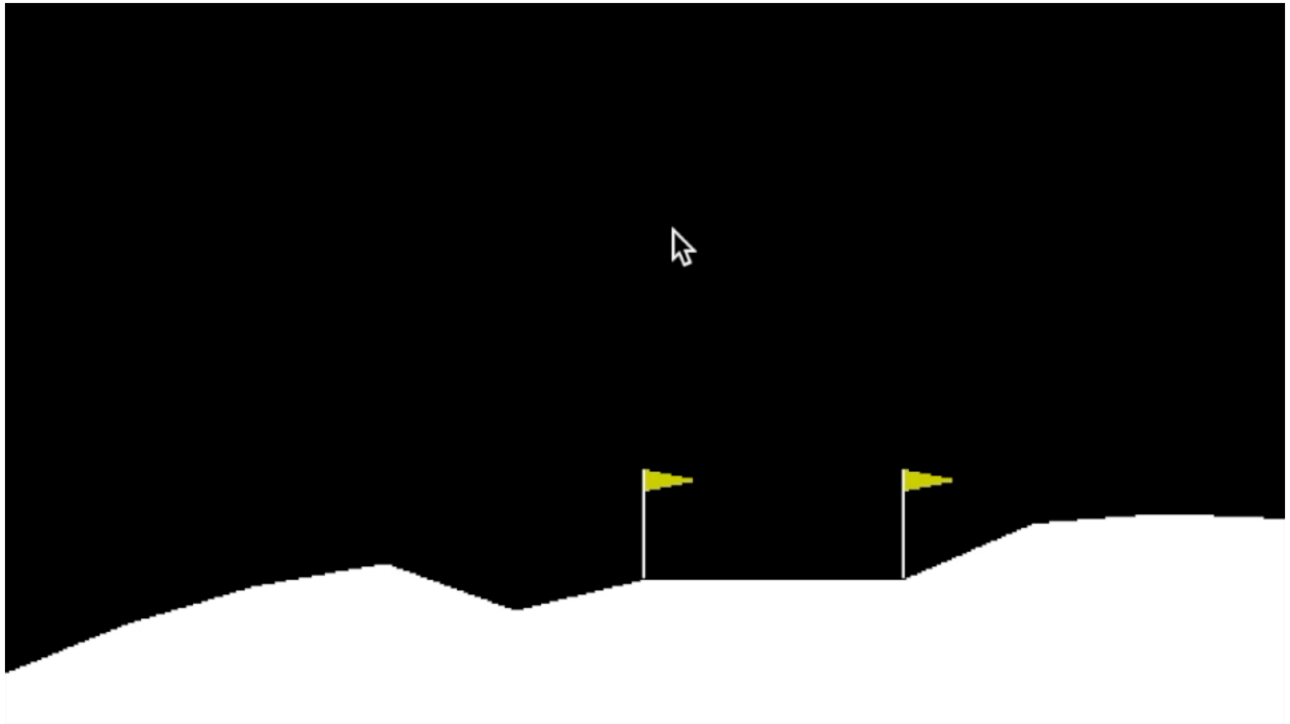
Conclusion

- Can do shared autonomy with minimal assumptions!
- Idea: Q-Learning & pick high-value action most similar to user's action.
- Works well in virtual environments (real humans).
- Seems to work well in real environments, too.

Thanks for your attention!

Q&A, if time permits it.

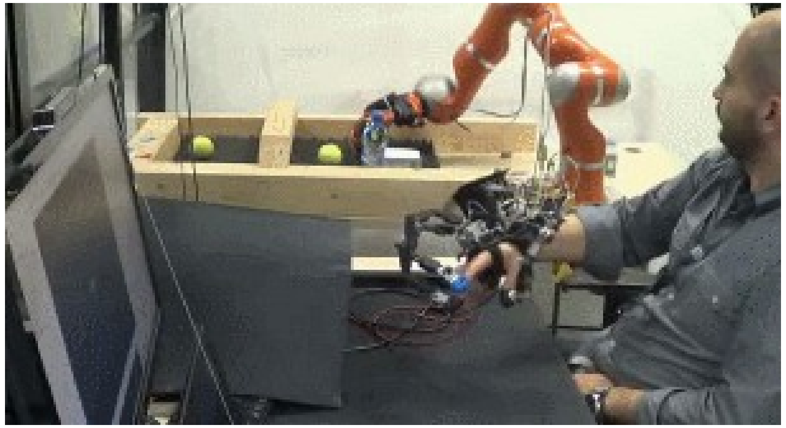
Project website: https://sites.google.com/view/deep-assist



Video of computer-assisted human piloting the lander.

Shared Autonomy via Hindsight Optimization

Teleoperation

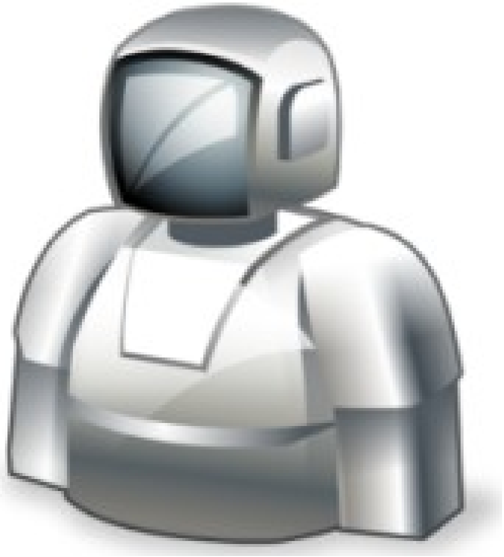


Noisy, insufficient degrees of freedom, tedious

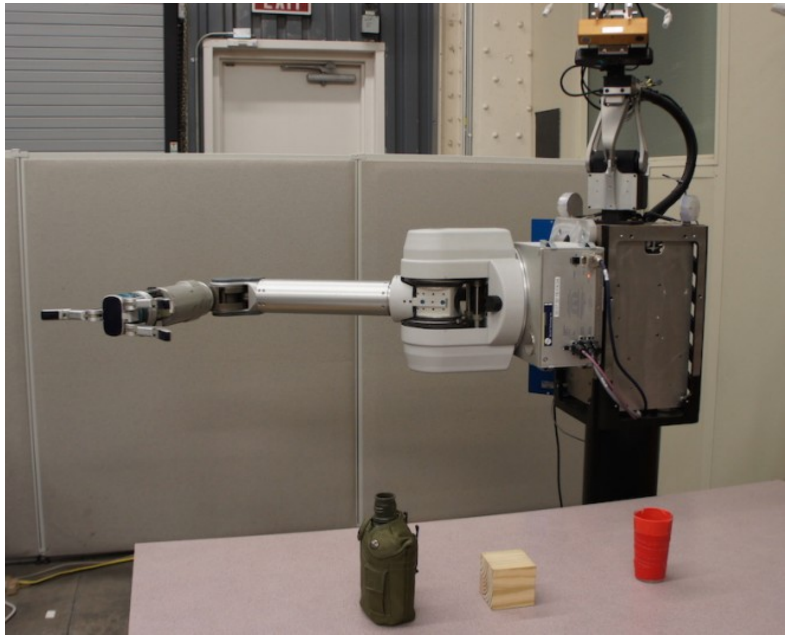
Image credit: Javdani RSS2015 talk

Shared Autonomy

+



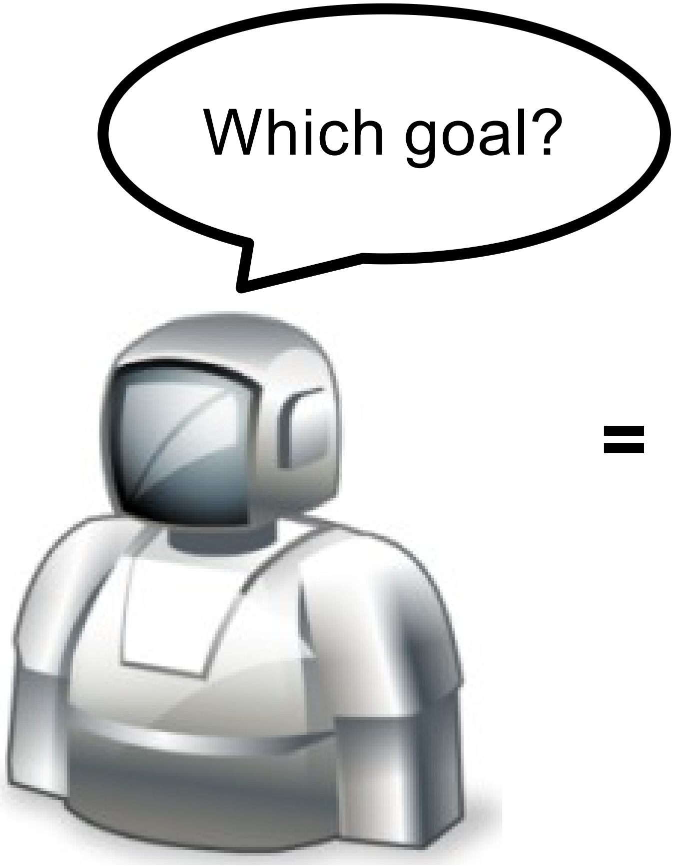
=



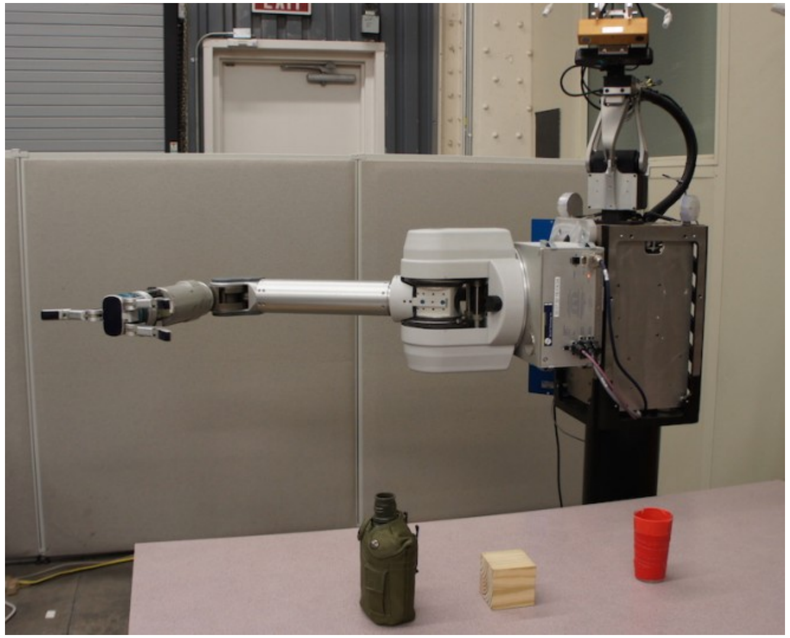
User Input **Autonomous Assistance** **Achieve Goal**

Shared Autonomy

+



=



User Input

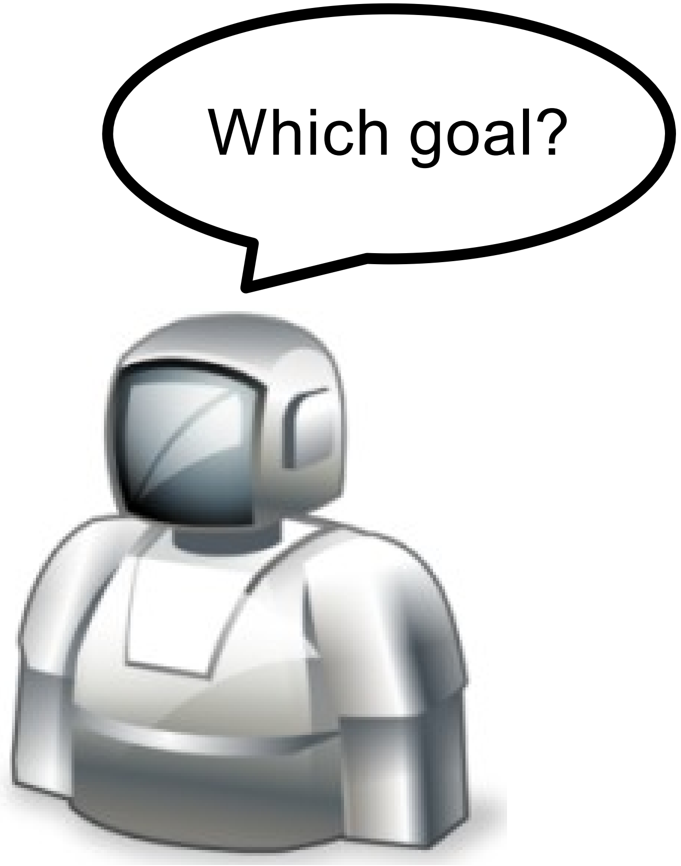
Autonomous Assistance

Achieve Goal

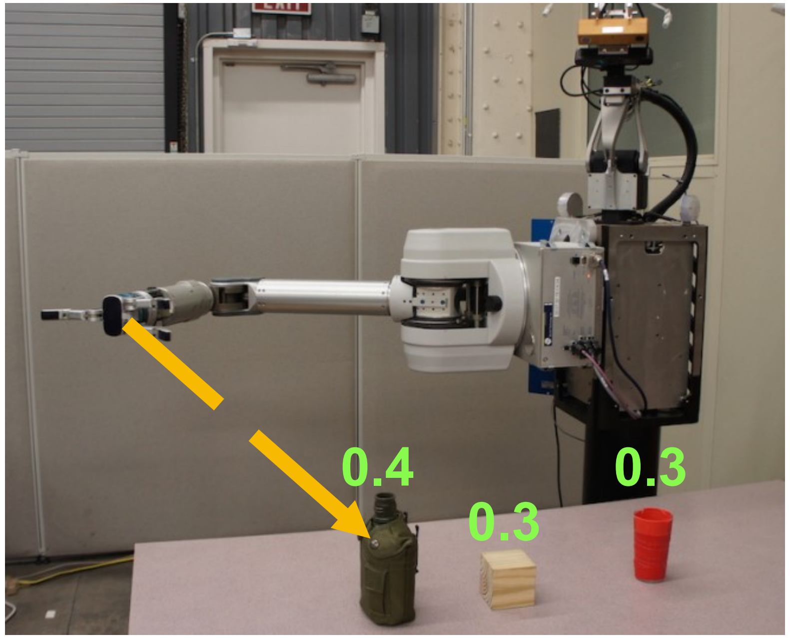
Shared Autonomy

Predict goal
Assist for single goal

[Dragan and Srinivasa 13]
[Kofman et al. 05]
[Kragic et al. 05]
[Yu et al. 05]
[McMullen et al. 14]
…



Autonomous Assistance



Achieve Goal

Shared Autonomy

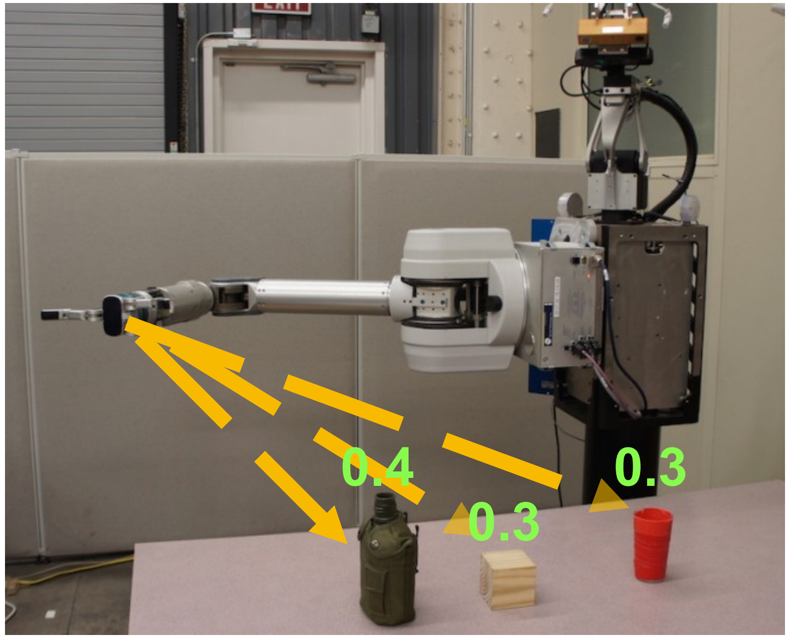
Predict goal
Assist for single goal

[Dragan and Srinivasa 13]
[Kofman et al. 05]
[Kragic et al. 05]
[Yu et al. 05]
[McMullen et al. 14]
…

Predict goal distribution
Assist for distribution

[Hauser 13]
This work!

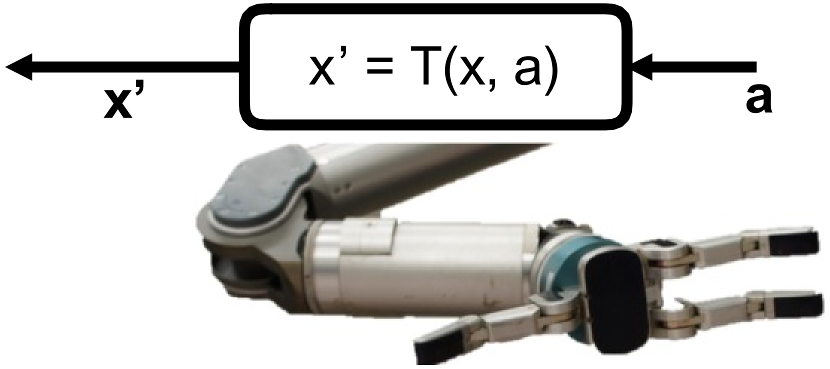
Autonomous Assistance



Achieve Goal

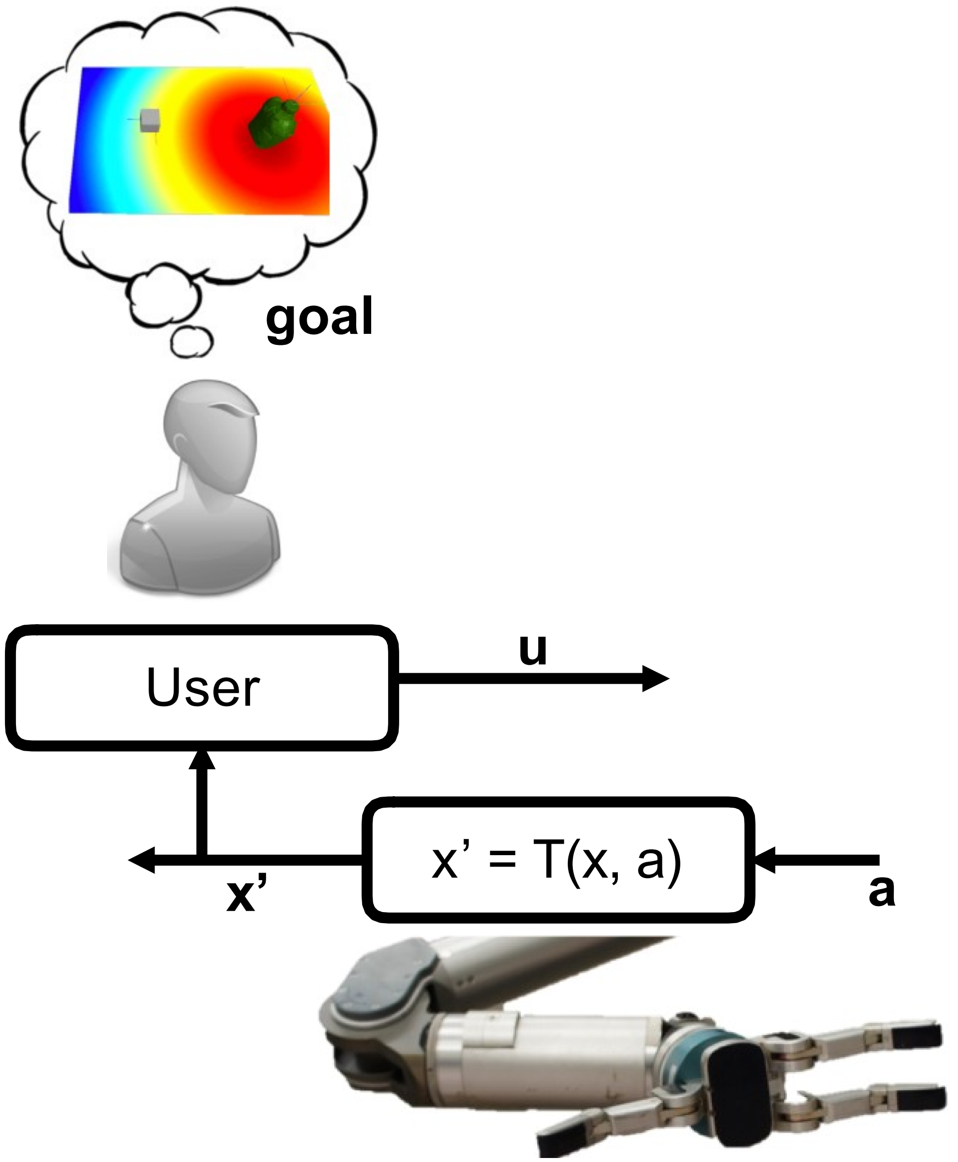
Method

- System dynamics: x' = T(x, a)



Method

- System dynamics: x' = T(x, a)
- User (MDP) as $(X, U, T, C_g^{\text{usr}})$
  - User policy: $\pi_g^{\text{usr}}(x) = p(u|x, g)$
  - MaxEnt IOC: $C_g^{\text{usr}} : X \times U \to \mathcal{R}$



Method

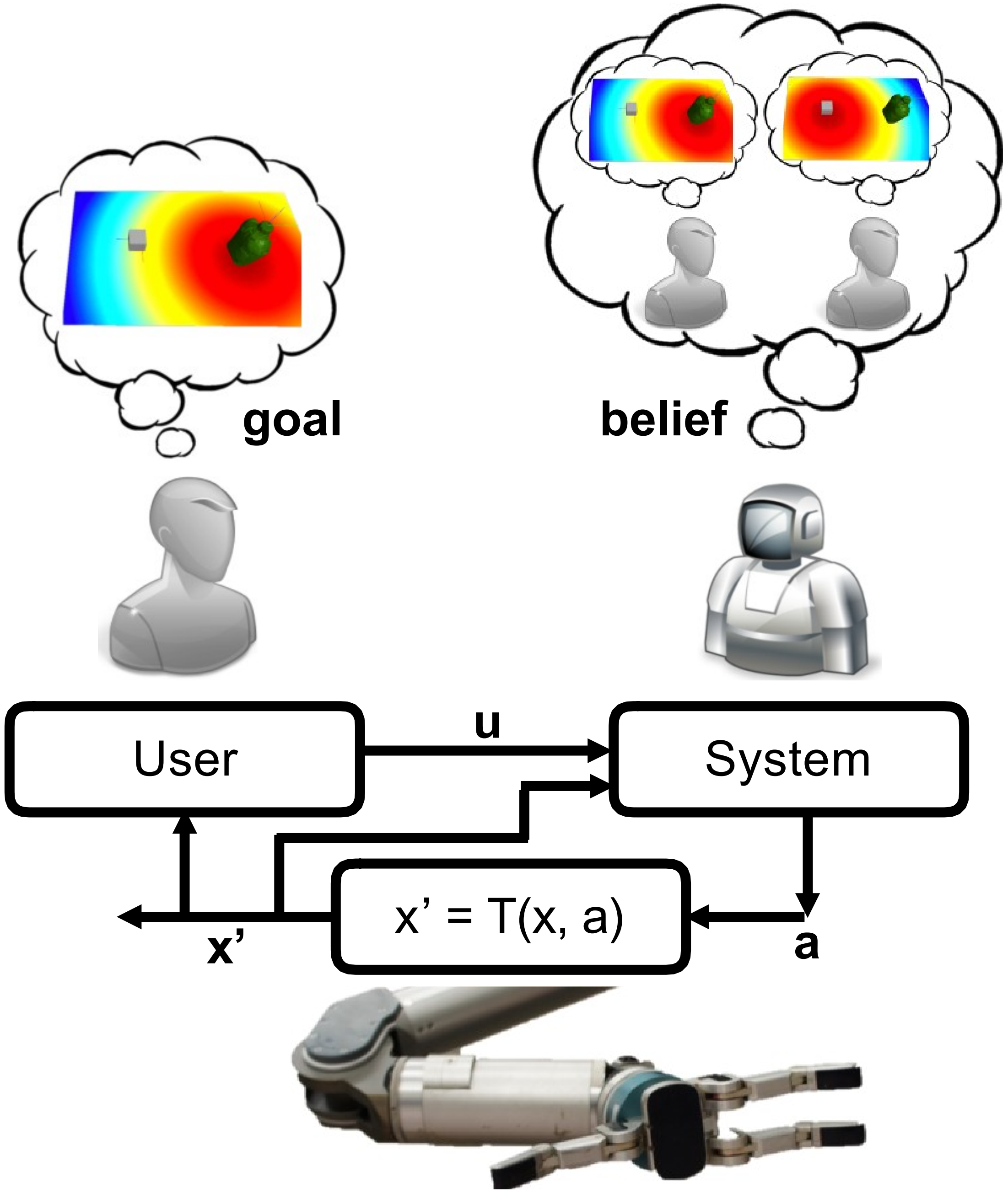
- System dynamics: x' = T(x, a)

- User (MDP) as $(X, U, T, C_g^{\text{usr}})$
  - User policy: $\pi_g^{\text{usr}}(x) = p(u|x, g)$
  - MaxEnt IOC: $C_g^{\text{usr}} : X \times U \to \mathcal{R}$

- System (POMDP) as $(S, A, T, C^{\text{rob}}, U, \Omega)$
  - Uncertainty over user's goal
  - System state: $S = X \times G$
  - Observation: user inputs U
  - Observation model Ω

$$p(g|\xi^{0\to t}) = \frac{p(\xi^{0\to t}|g)p(g)}{\sum_{g'} p(\xi^{0\to t}|g')p(g')}$$

  - Cost function $C^{\text{rob}} : S \times A \times U \to \mathcal{R}$



Hindsight Optimization

- MDP solution:

$$V^{\pi^{\mathrm{r}}}(s) = \mathbb{E}\left[\sum_t C^{\mathrm{r}}(s_t, u_t, a_t) \mid s_0 = s\right]$$

$$V^*(s) = \min_{\pi^{\mathrm{r}}} V^{\pi^{\mathrm{r}}}(s)$$

Hindsight Optimization

- MDP solution:

$$V^{\pi^{\mathrm{r}}}(s) = \mathbb{E}\left[\sum_t C^{\mathrm{r}}(s_t, u_t, a_t) \mid s_0 = s\right]$$

$$V^*(s) = \min_{\pi^{\mathrm{r}}} V^{\pi^{\mathrm{r}}}(s)$$

- POMDP solution:

$$V^{\pi^{\mathrm{r}}}(b) = \mathbb{E}\left[\sum_t C^{\mathrm{r}}(s_t, u_t, a_t) \mid b_0 = b\right]$$

$$V^*(b) = \min_{\pi^{\mathrm{r}}} V^{\pi^{\mathrm{r}}}(b)$$

Hindsight Optimization

- MDP solution:

$$V^{\pi^{\mathrm{r}}}(s) = \mathbb{E}\left[\sum_t C^{\mathrm{r}}(s_t, u_t, a_t) \mid s_0 = s\right]$$

$$V^*(s) = \min_{\pi^{\mathrm{r}}} V^{\pi^{\mathrm{r}}}(s)$$

- POMDP solution:

$$V^{\pi^{\mathrm{r}}}(b) = \mathbb{E}\left[\sum_t C^{\mathrm{r}}(s_t, u_t, a_t) \mid b_0 = b\right]$$

$$V^*(b) = \min_{\pi^{\mathrm{r}}} V^{\pi^{\mathrm{r}}}(b)$$

- HOP approximation:

$$\begin{aligned} V^{\mathrm{HS}}(b) &= \mathbb{E}_b\left[\min_{\pi^{\mathrm{r}}} V^{\pi^{\mathrm{r}}}(s)\right] \\ &= \mathbb{E}_g[V_g(x)] \end{aligned}$$

Hindsight Optimization

- MDP solution:

$$V^{\pi^{\mathrm{r}}}(s) = \mathbb{E}\left[\sum_t C^{\mathrm{r}}(s_t, u_t, a_t) \mid s_0 = s\right]$$

$$V^*(s) = \min_{\pi^{\mathrm{r}}} V^{\pi^{\mathrm{r}}}(s)$$

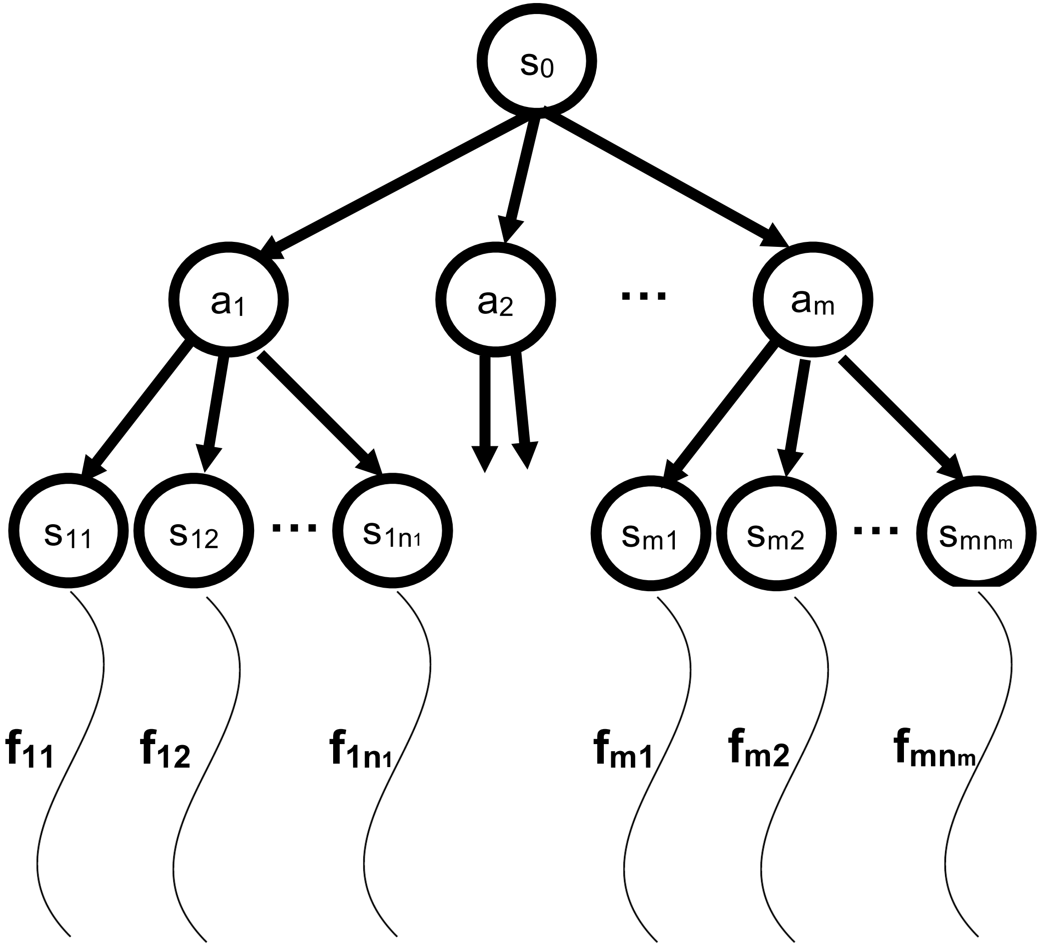
- POMDP solution:

$$V^{\pi^{\mathrm{r}}}(b) = \mathbb{E}\left[\sum_t C^{\mathrm{r}}(s_t, u_t, a_t) \mid b_0 = b\right]$$

$$V^*(b) = \min_{\pi^{\mathrm{r}}} V^{\pi^{\mathrm{r}}}(b)$$

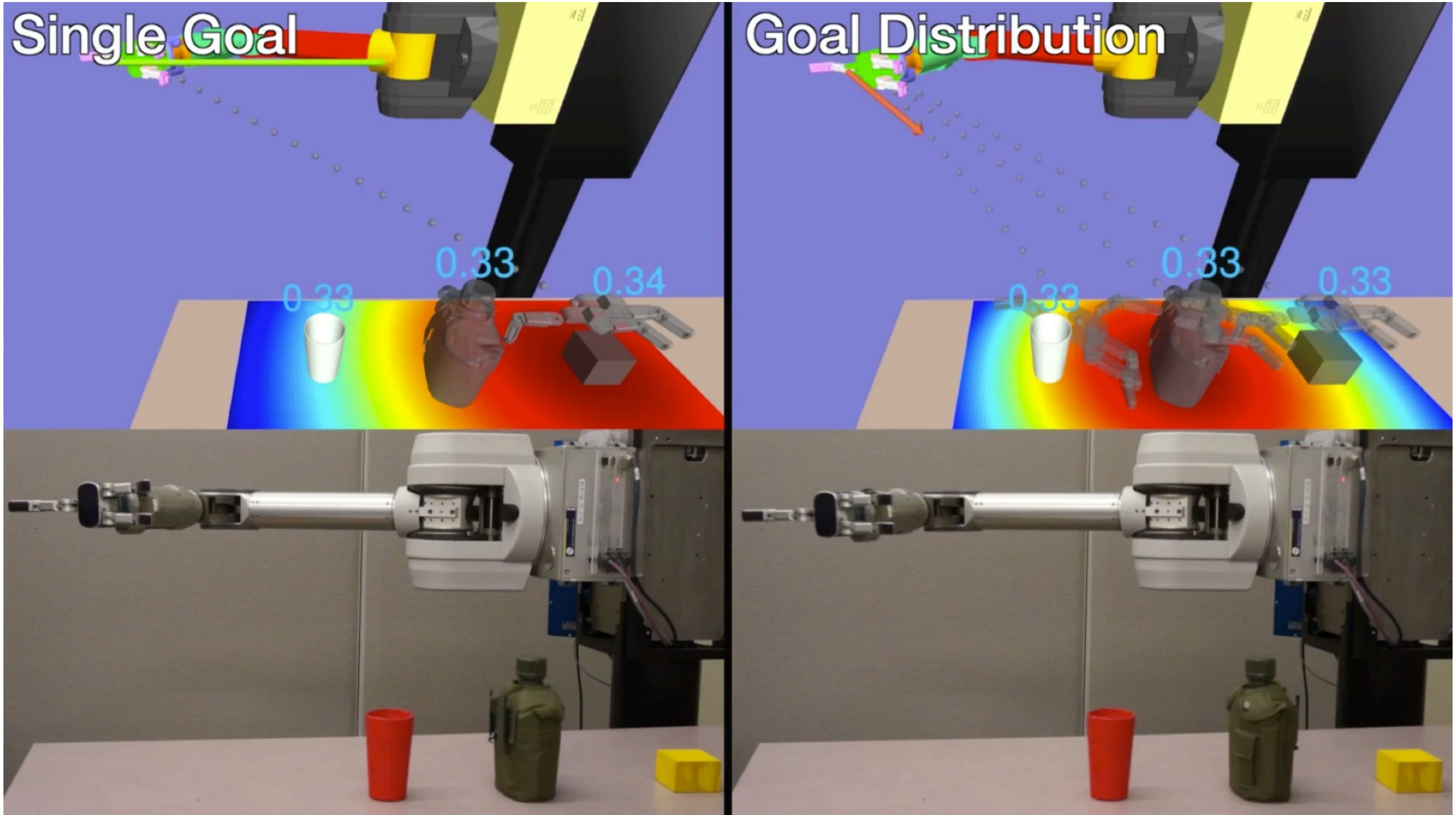
- HOP approximation:

$$\begin{aligned} V^{\mathrm{HS}}(b) &= \mathbb{E}_b\left[\min_{\pi^{\mathrm{r}}} V^{\pi^{\mathrm{r}}}(s)\right] \\ &= \mathbb{E}_g[V_g(x)] \end{aligned}$$



Deterministic problem for each future

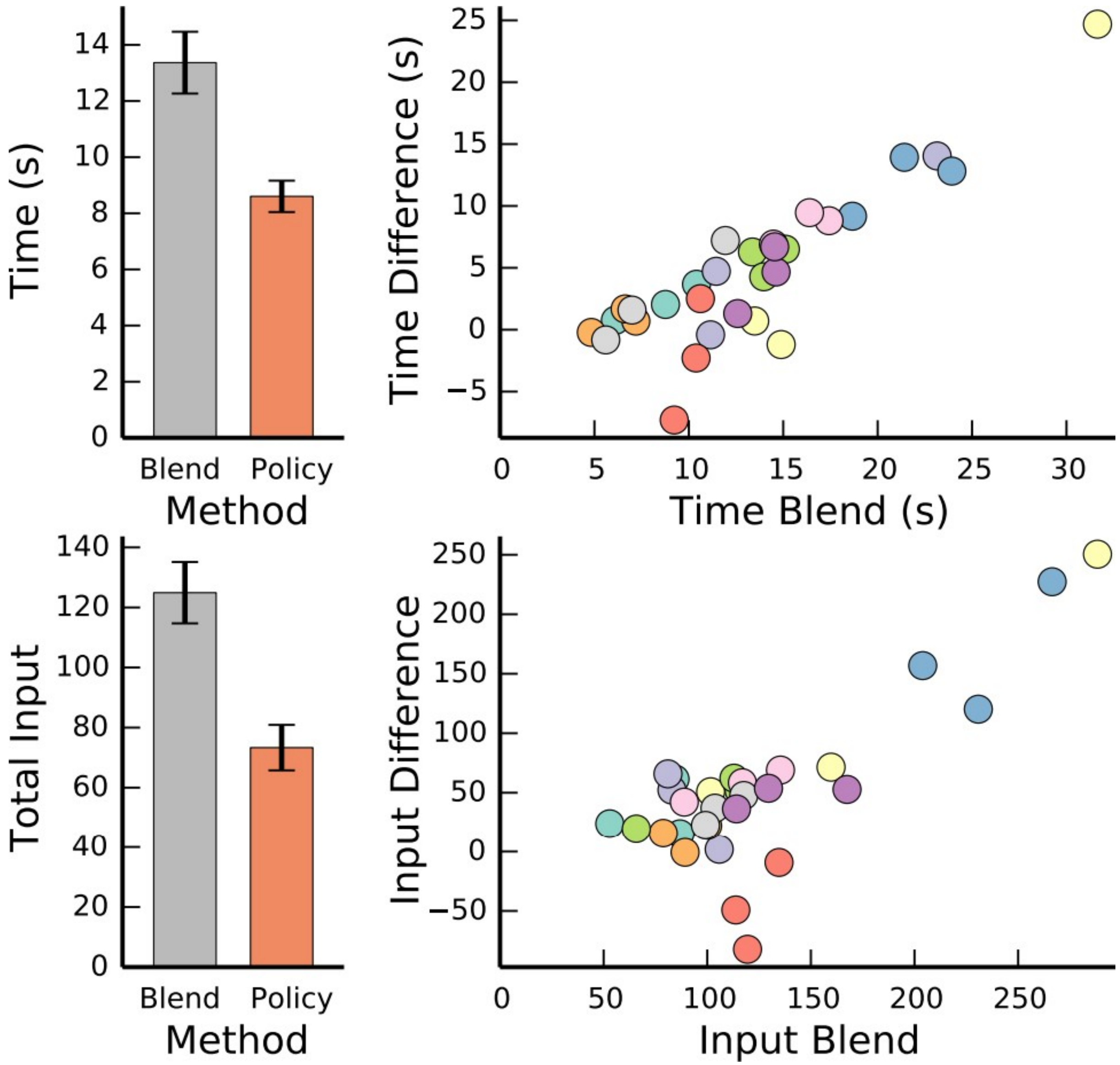
Results (video)



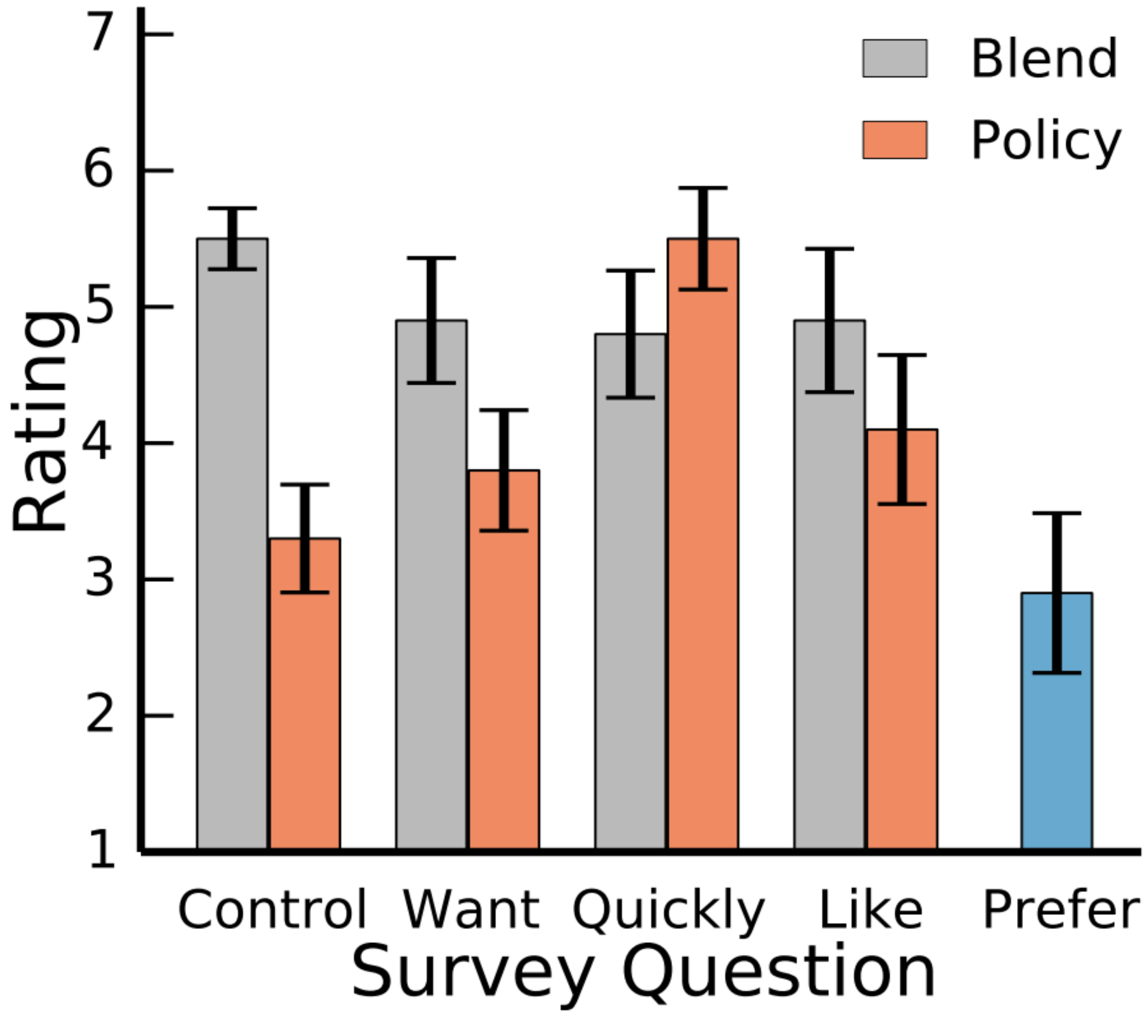
Results

Compare with method that predicts one goal, the proposed method has:

- Faster execution time
- Fewer user inputs



User Study



Rating
7
6
5
4
3
2
1
Blend
Policy
Control
Want
Quickly
Like
Prefer
Survey Question

Limitations

- Requires prior knowledge about the world:
  - a dynamics model that predicts the consequences of taking a given
    action in a given state of the environment;
  - the set of possible goals for the user;
  - the user's control policy given their goal.
- Suitable in constrained domains where where this knowledge can be directly hard-coded or learned.
- Unsuitable for unstructured environments with ill-defined goals and unpredictable user behavior.

References

- Javdani, S., Srinivasa, S. S., & Bagnell, J. A. (2015). Shared autonomy via hindsight optimization. *Robotics science and systems: online proceedings*, *2015*.
- RSS2015 talk: “Shared autonomy via hindsight optimization”
- Javdani, S., Admoni, H., Pellegrinelli, S., Srinivasa, S. S., & Bagnell, J. A. (2018). Shared autonomy via hindsight optimization for teleoperation and teaming. *The International Journal of Robotics Research*, *37*(7), 717-742.
- ICAPS 2015 talk: "Hindsight Optimization for Probabilistic Planning with Factored Actions”

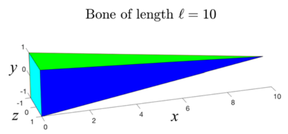
RelaxedIK: Real-time Synthesis of Accurate and Feasible Robot Arm Motion

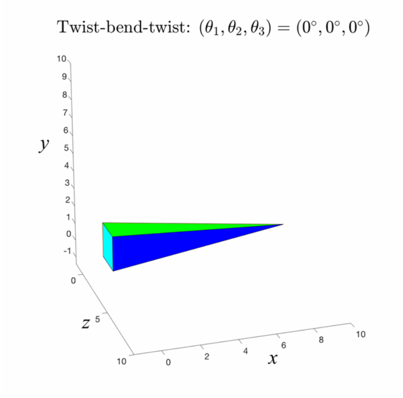
Tingwu Wang

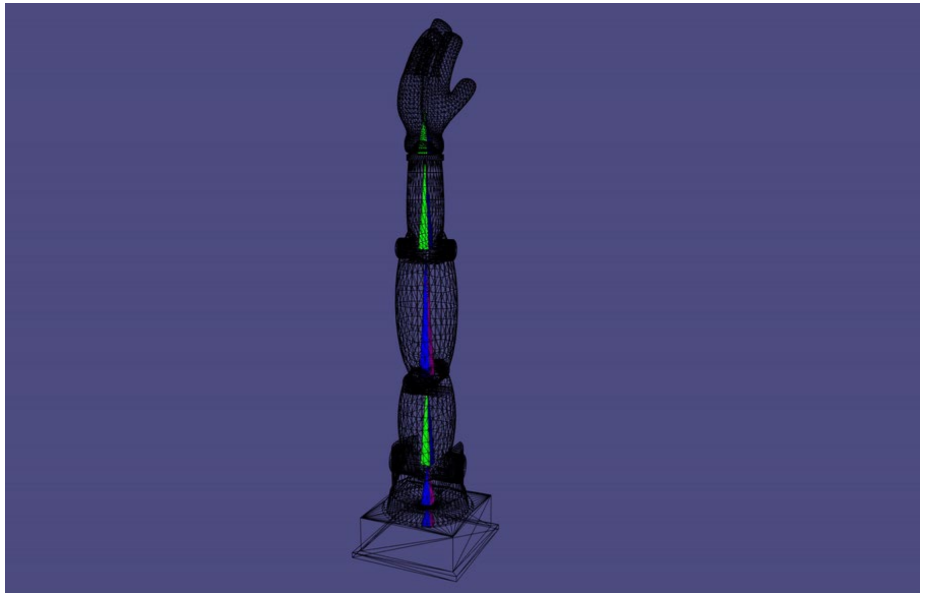
University Of Toronto, CSC-2621, Paper Reading Seminar

Recap: Forward Kinematics (FK)

1. Forward Kinematics
   a. A common robotic skeleton is a tree of rigid bones
   b. The relative Euler angles of all the bones determine the end-effectors
      i. End-effectors? A tool that's connected to the end of a robot arm







Recap: Inverse Kinematics (IK)

1. Inverse Kinematics
   a. The indirect control of forward kinematics makes it hard to use in application
      i. Achieve certain poses?
      ii. Achieve certain velocities (reward)?
   b. We formulate the inverse kinematics function as: $\Theta = IK(\mathbf{p})$, which can be easily written in an analytic form for a simple tree s
      i. Pose contains velocity?
      ii. Hard to find feasible state space?
   c. In reality, IK is often treated as an optimization problem

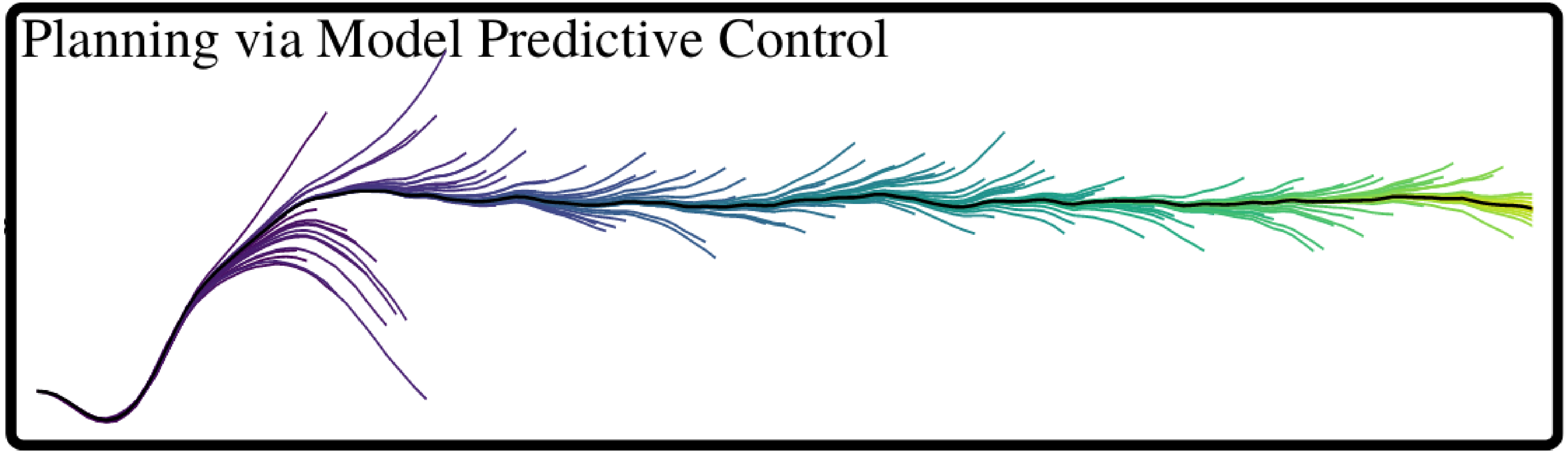
$$\chi_p(\Theta) = || \; \mathbf{p}_g - FK(\Theta) \; ||_2$$

Imitation Learning

1. Imitation learning has been studied by different communities
    a. Motion synthesis in character animation?
    b. Inverse optimal control?
    c. Imitation learning?

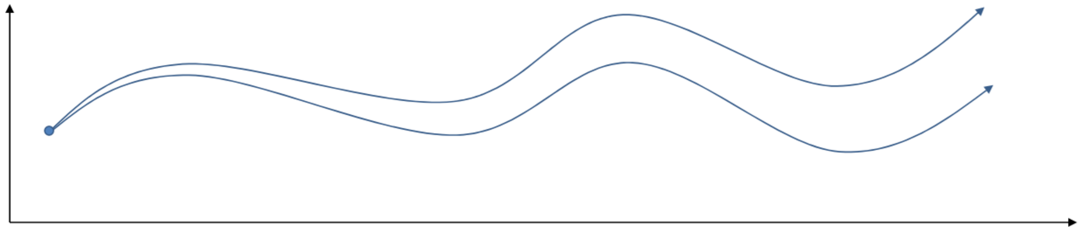
Imitation Learning

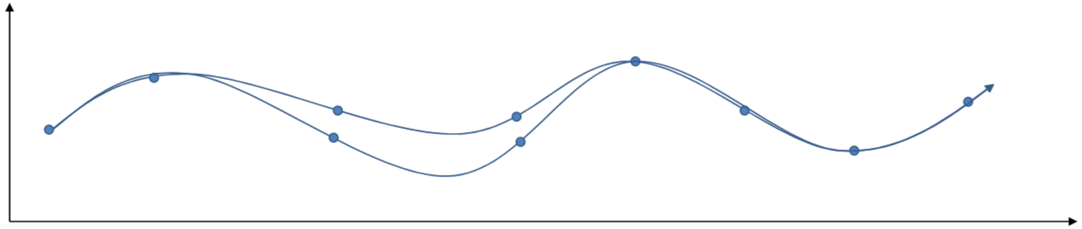
1. Imitation learning has been studied by different communities
2. Within the focus of this course, people worked on imitation learning using
   a. Forward dynamics
      i. Shooting method (optimize the actions)

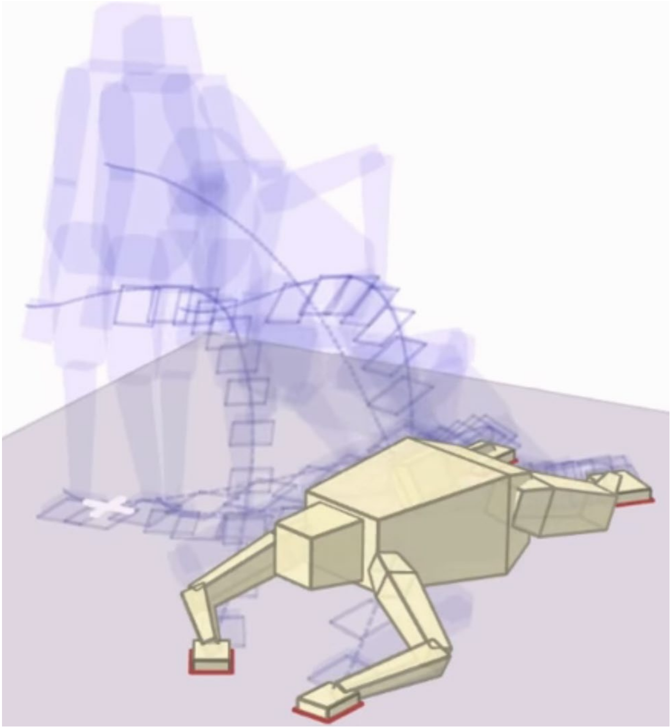


Imitation Learning

1. Imitation learning has been studied by different communities
2. Within the focus of this course, people worked on imitation learning using
    a. Forward dynamics
    b. Inverse dynamics
        i. Collocation method (optimize states)







Imitation Learning

1. Imitation learning has been studied by different communities
2. Within the focus of this course, people worked on imitation learning using
   a. Forward dynamics
      i. Shooting method
   b. Inverse dynamics
      i. Collocation method
   c. Model-free method
      i. GAIL
   d. Motion synthesis with IK
      i. Today's paper
      ii. Old school but with new techniques

Imitation Learning

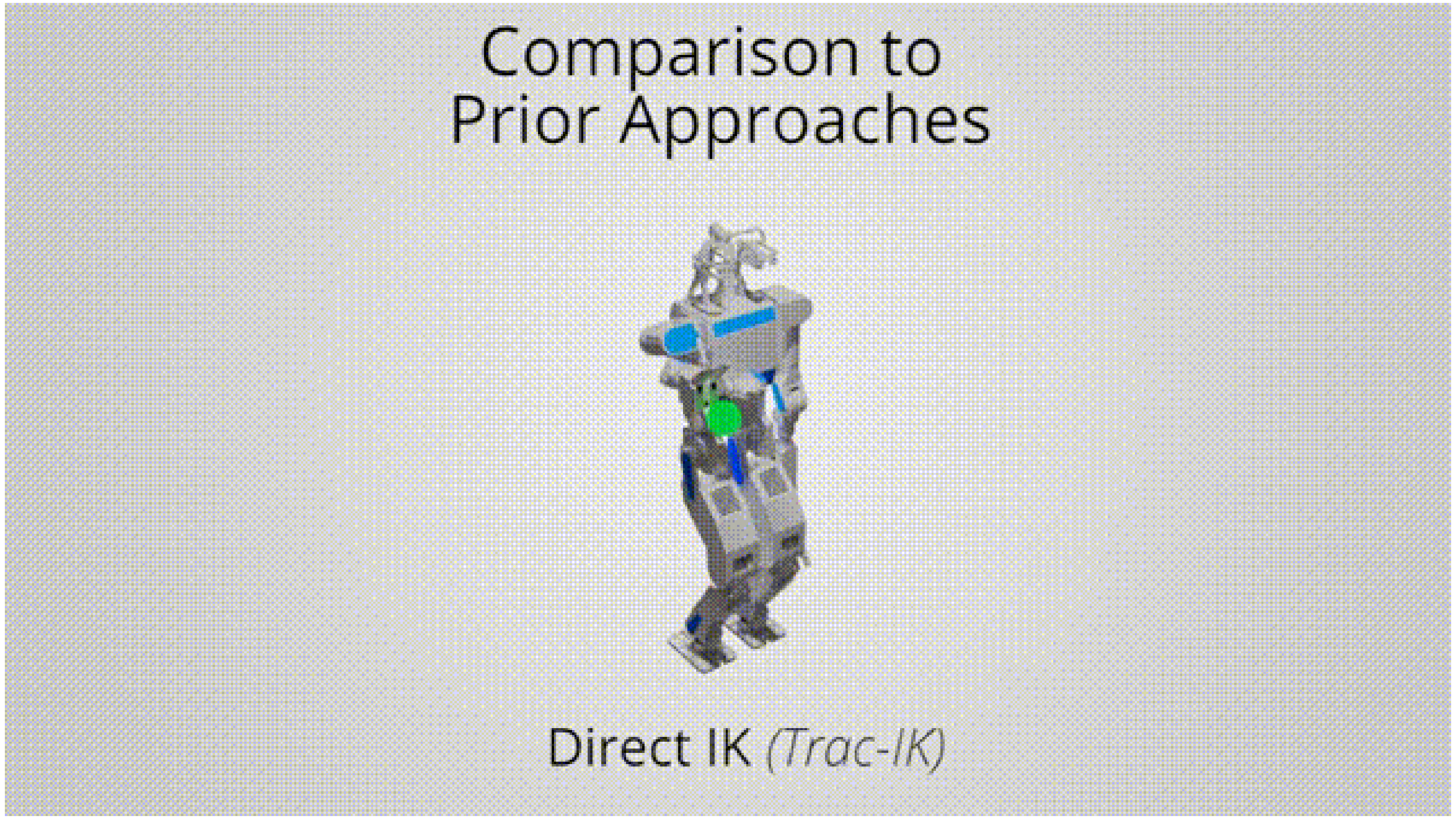
1. Imitation learning using IK
   a. Basic idea: Using IK to bridge between target pose and agent’s angles
   b. Input: M (consecutive) expert (goal) poses
   c. Output: M (consecutive) frames of agent’s euler joints
   d. Constraints:
      i. IK constraints (goal constraints)
      ii. Between-frames constraints
      iii. Etc.

Imitation Learning

1. Imitation learning using IK
    a. Basic idea: minimize the difference of target pose and agent pose
2. Direct point-to-point approach
    a. TRAC-IK (previous state-of-the-art)
    b. Pose2pose / frame2frame imitation learning
        i. Ignore most of the constraints between frames
    c. Problems
        i. Self-collision
            1. Time constraints

Imitation Learning

1. Self-collision

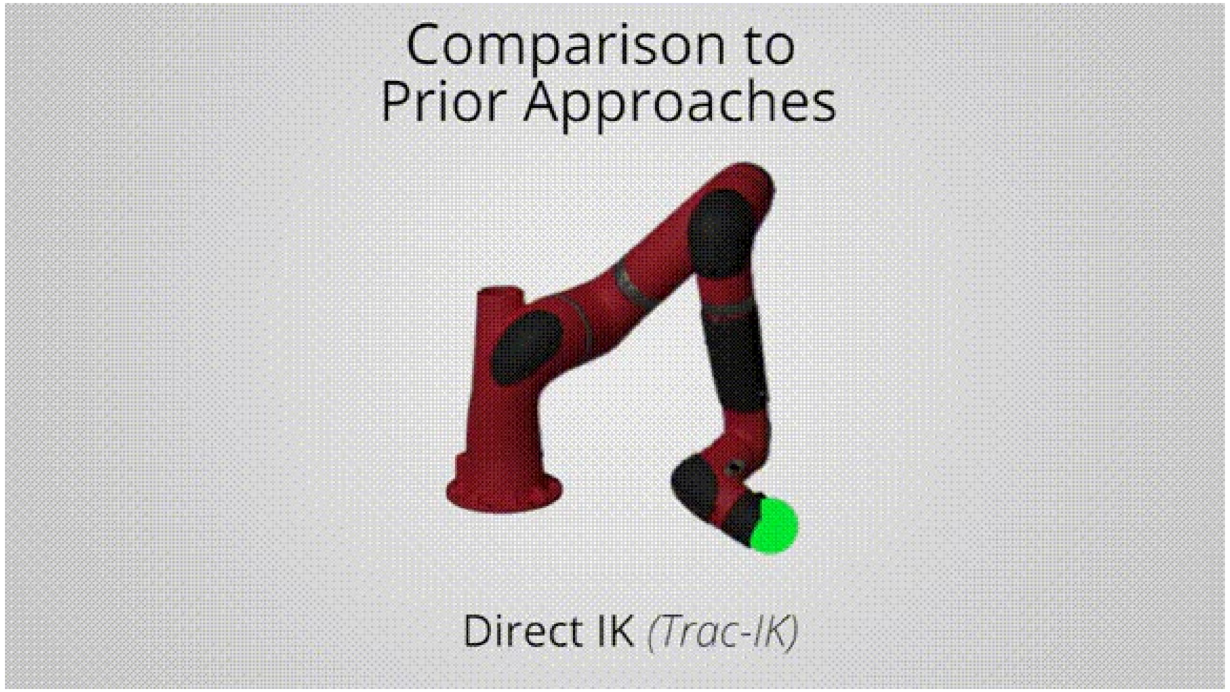


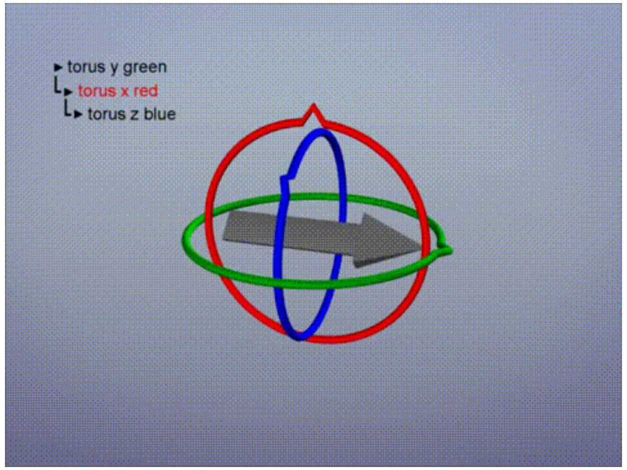
Imitation Learning

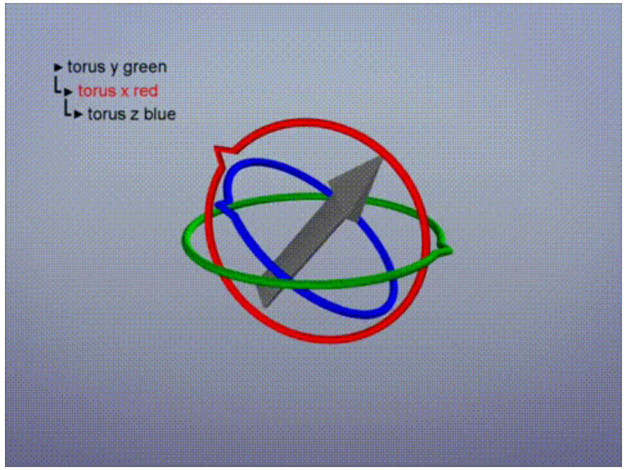
1. Imitation learning using IK
    a. Basic idea: minimize the difference of target pose and agent pose
2. Direct point-to-point approach
    a. TRAC-IK (previous state-of-the-art)
    b. Pose2pose / frame2frame imitation learning
        i. Ignore most of the constraints between frames
    c. Problems
        i. Self-collision
        ii. Singularities

Imitation Learning

1. Singularities
   a. E.g. Losing a DoF
   b. Infinite control signals





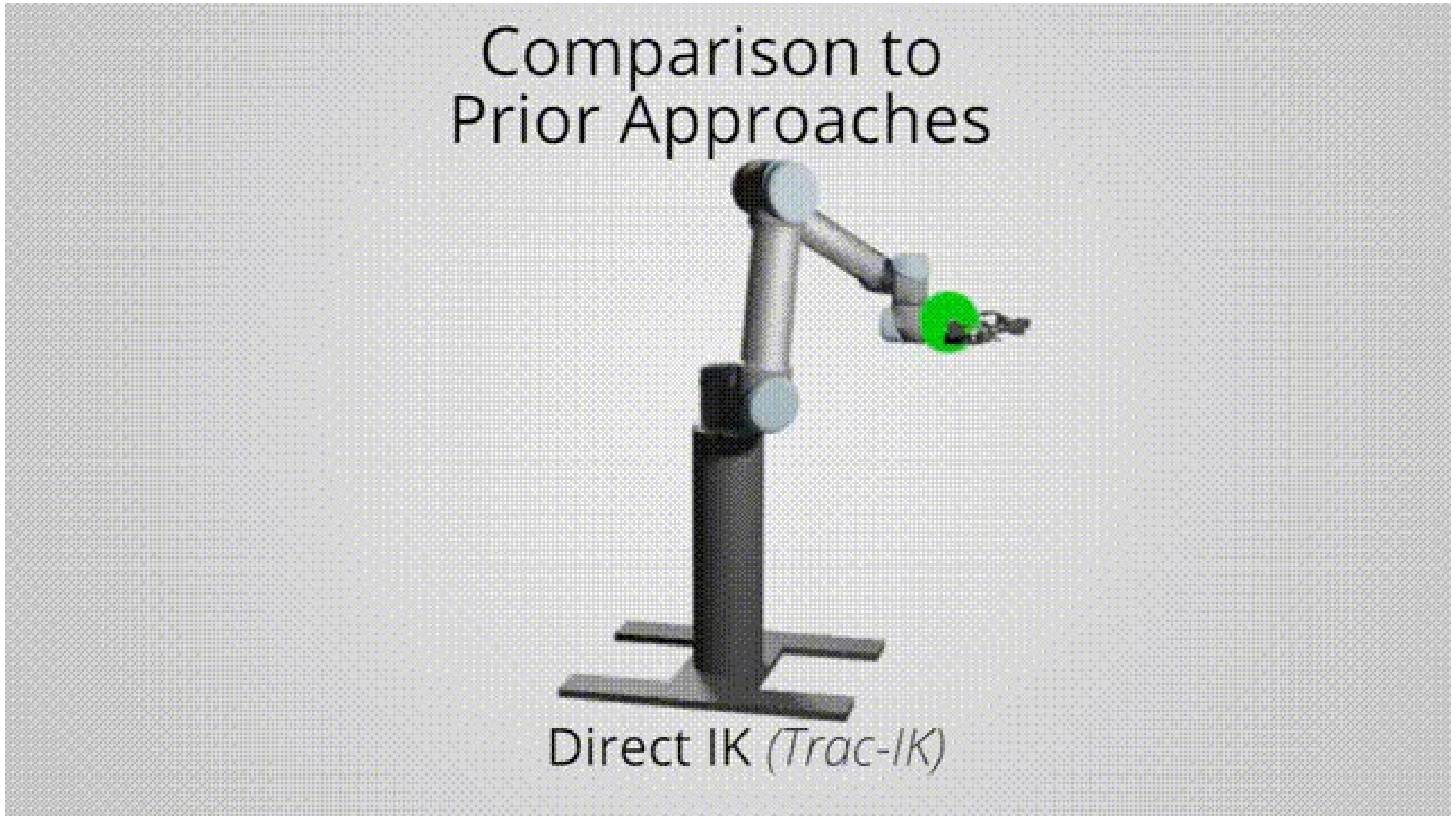


Imitation Learning

1. Imitation learning using IK
    a. Basic idea: minimize the difference of target pose and agent pose
2. Direct point-to-point approach
    a. TRAC-IK (previous state-of-the-art)
    b. Pose2pose / frame2frame imitation learning
        i. Ignore most of the constraints between frames
    c. Problems
        i. Self-collision
        ii. Singularities
        iii. Discontinuity

Imitation Learning

1. Discontinuity

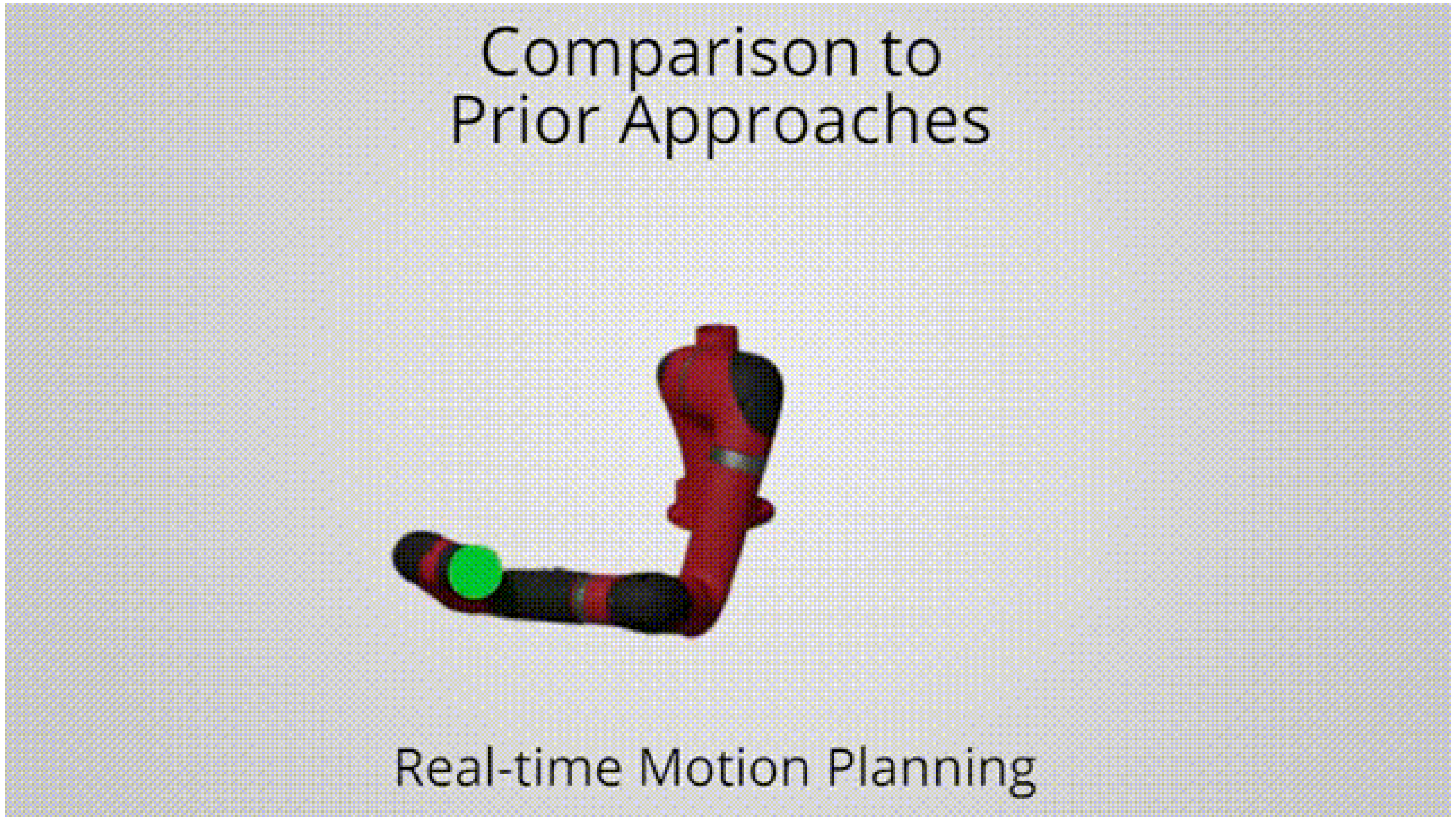


Imitation Learning

1. Imitation learning using IK
    a. Basic idea: minimize the difference of target pose and agent pose
2. Direct point-to-point approach
3. Real-time motion planning approach
    a. Output the (conservative) solutions real-time
        i. Always meet the control & collision constraints
        ii. Soft goal constraints
    b. Problems
        i. Goal mistracking

Imitation Learning

1. Goal mistracking

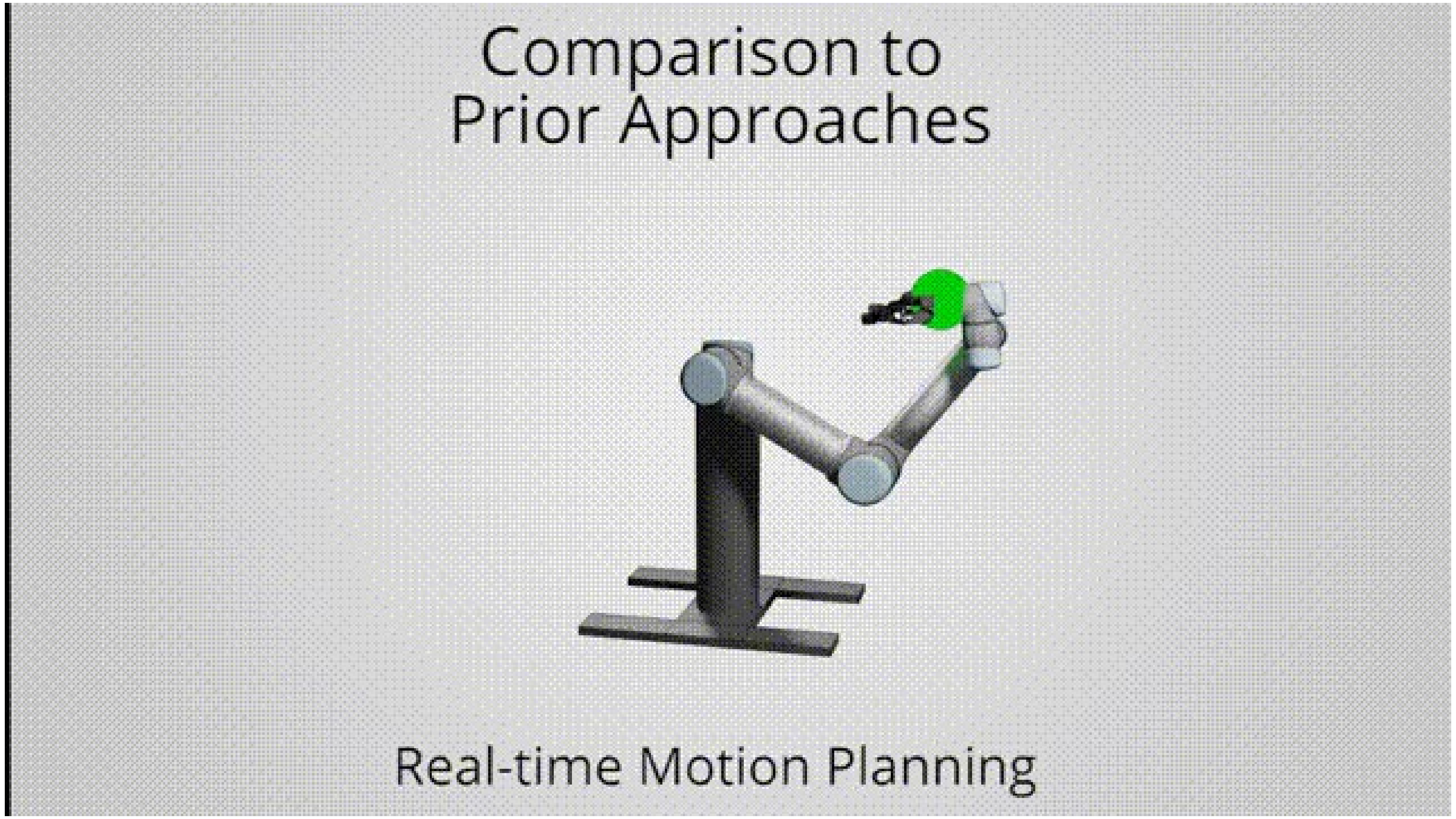


Imitation Learning

1. Imitation learning using IK
    a. Basic idea: minimize the difference of target pose and agent pose
2. Direct point-to-point approach
3. Real-time motion planning approach
    a. Output the (conservative) solutions real-time
        i. Always meet the control & collision constraints
        ii. Soft goal constraints
    b. Problems
        i. Goal mistracking
        ii. Unpredictable behaviors

Imitation Learning

1. Unpredictable behaviors



Relaxed IK

1. Basic Idea: Using soft (Relaxed) IK loss that considers self-collision and singularity for faster optimization

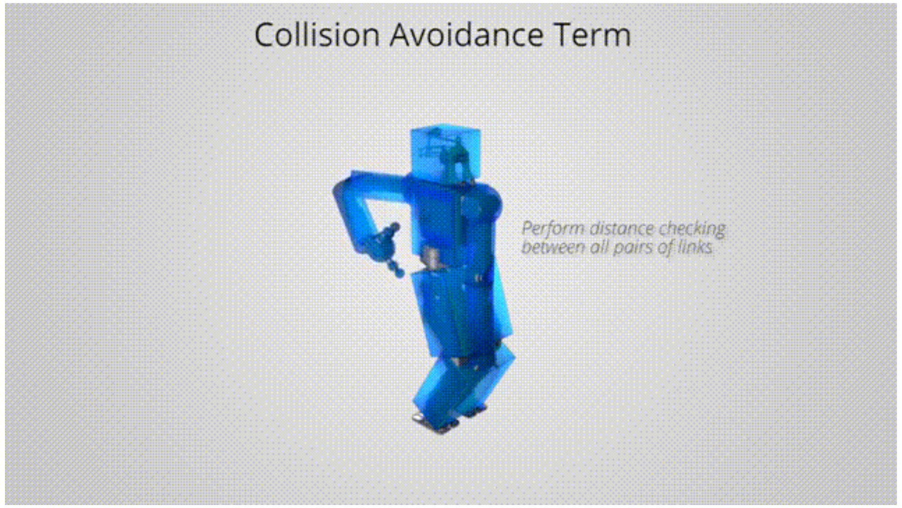
$$\mathbf{f}(\Theta) = \sum_{i=1}^{k} w_i f_i(\Theta, \Omega_i)$$

1. Loss functions
   a. End-effector position & orientation matching
   b. Minimize joint velocity, acceleration, jerk
   c. Self-collision loss (fast)
   d. Singularity loss

Relaxed IK

1. Self-collision loss
    a. Common approach: very slow
    b. Relaxed IK:
        i. Approximate how imminent the robot is to a collision state
        ii. Using simulated data to train a network to predict the distances between links

$$col(\Theta) = \sum_{i,j} b * exp(\frac{-dis(l_i, l_j)^2}{2c^2})$$



Relaxed IK

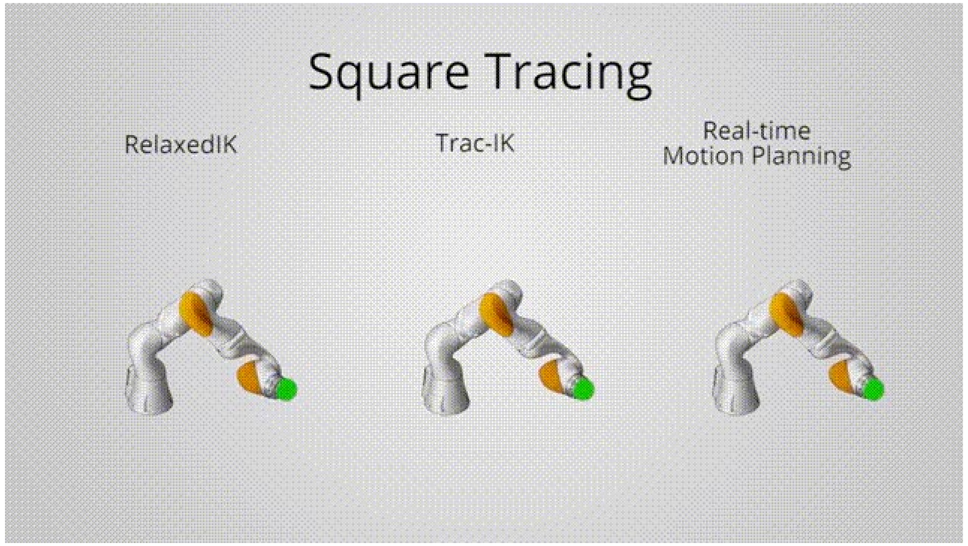
1. Singularity loss
    a. Kinematic singularities are well studied in robotics
    b. Relaxed IK:
        i. Find a metric that can approximate distance to a singularity
        ii. Jacobian's condition number is used as a proxy distance to singularity
            1. Why?

$$\dot{\mathbf{x}} = \mathbf{J}(\Theta)\dot{\Theta}$$

        i. Penalize condition values less than mean – b * std
            1. Estimate mean, std from simulated data

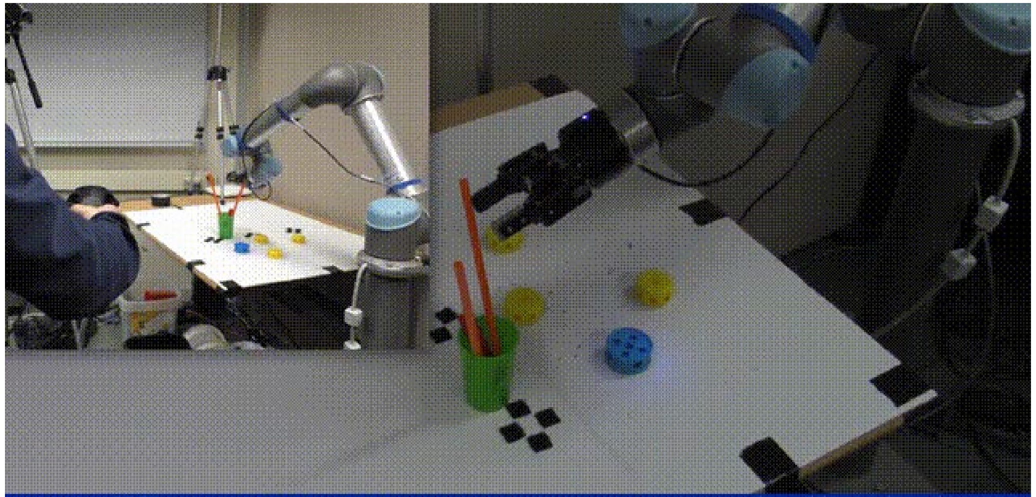
Relaxed IK

1. Pros
   a. Much faster and smoother performance
      i. combining neural network and traditional robotics
   b. Data driven, less human-engineering
      i. Novel singularity metric
   c. Easy to deploy
      i. Sim2Real



Relaxed IK

1. Pros
2. Cons
    a. No safety / convergence guarantee
    b. Weak experiments section
        i. Under-tuned baseline
        ii. Limited ablation study
    c. Slower than point2point methods
    d. Hyper-parameter sensitive



Rakita, D., Mutlu, B., & Gleicher, M. (2017, March).
A motion retargeting method for effective mimicry-based teleoperation of robot arms. HRI '17