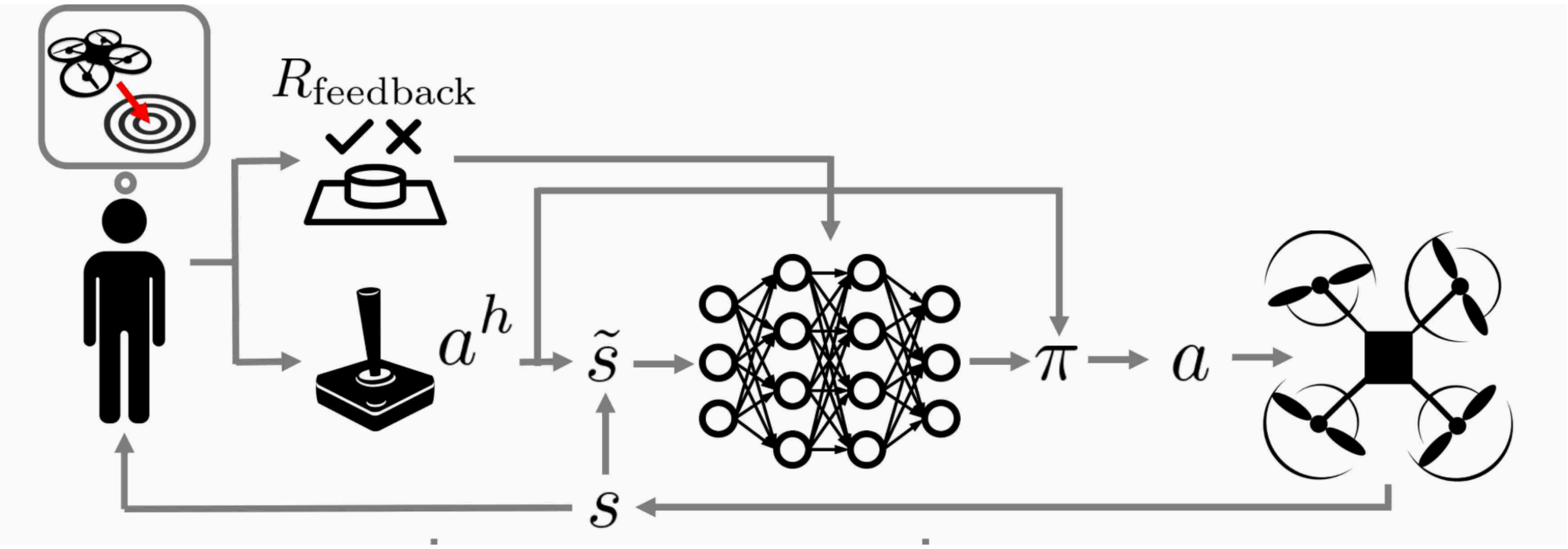
Shared Autonomy via Deep Reinforcement Learning

Siddharth Reddy, Anca Dragan, Sergey Levine
UC Berkeley

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



Key Question

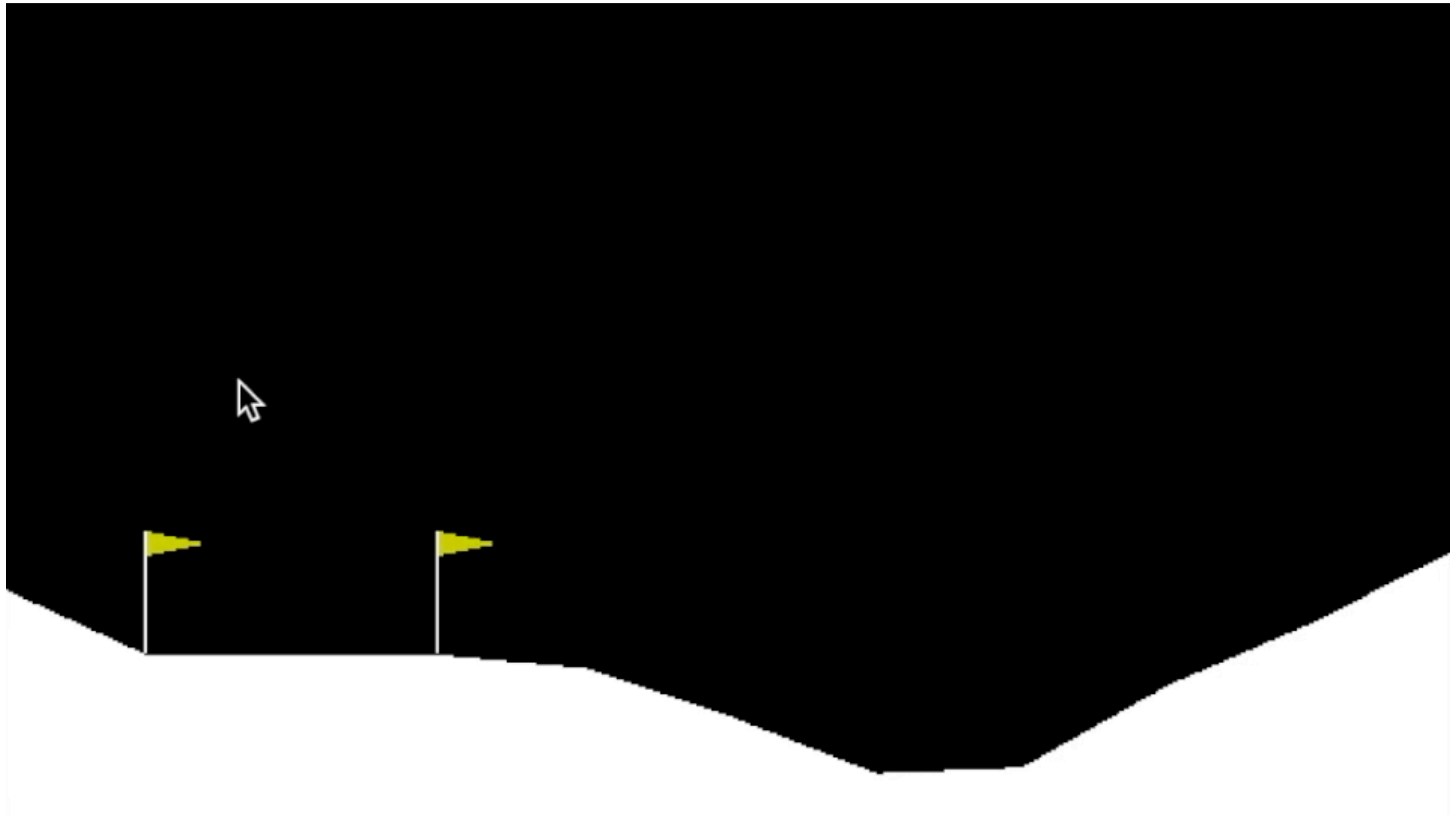
How can a robot **collaborating** with a human infer the human's goals with as few **assumptions** as possible?

Motivation

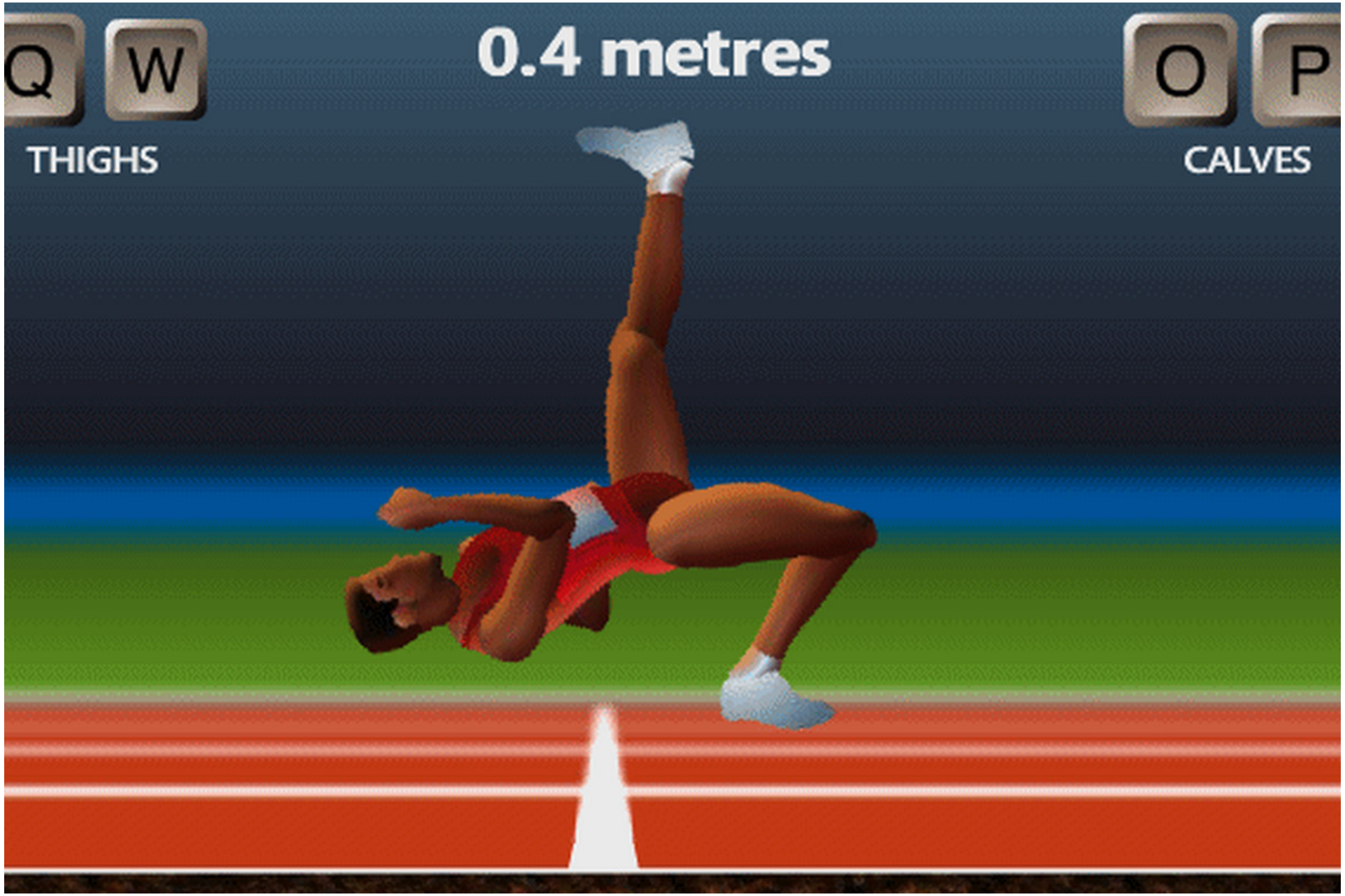
- **Hard:** Actuating a robot with many DoF and/or unfamiliar dynamics.
- **Hard:** Specifying a goal formally (e.g., coordinates).
- **Easy:** Demonstrating the goal indirectly.
    - …let the machine figure out what I want!

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

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
      end for
    end if
    Every C steps reset $\hat{Q} = Q$
  end for
end for

The Method (Continued)

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

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

Standard Q-Learning Initialization

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

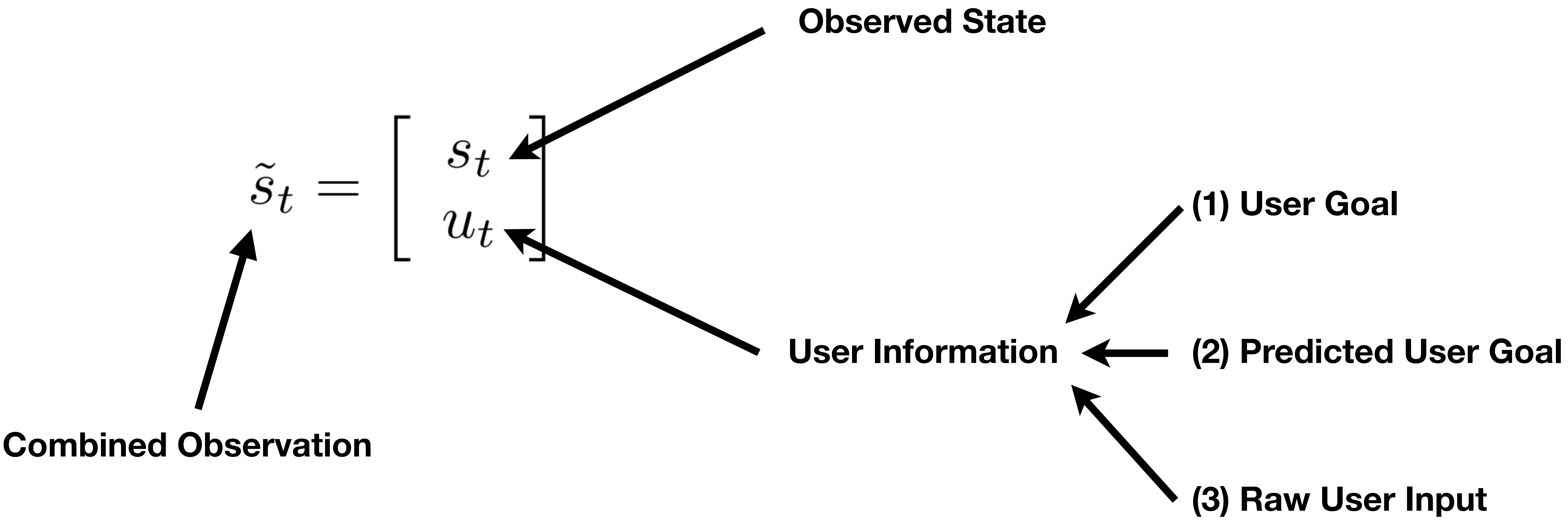
...ensuring action is "close enough" to optimal one.

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

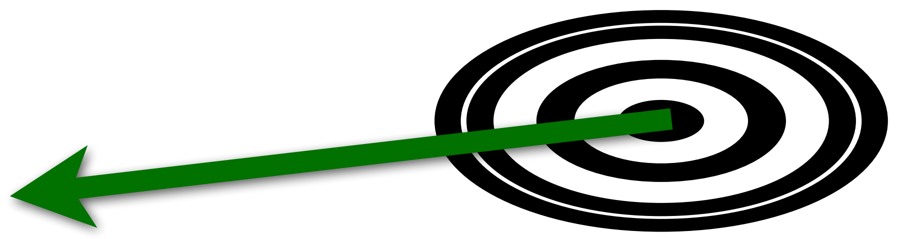


Experiments

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

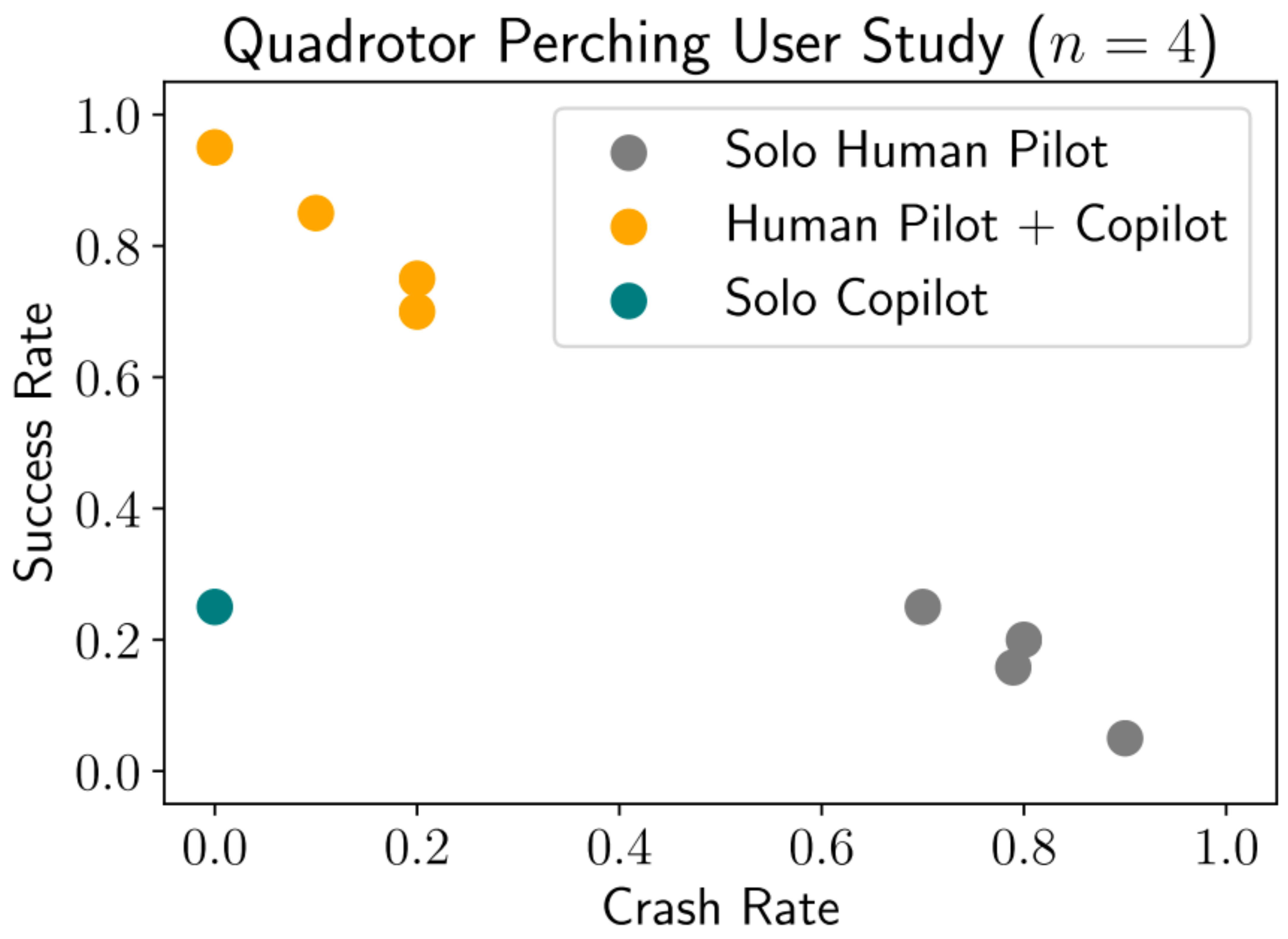
Real-World Experiments

- **Goal:** Land drone on pad **facing a certain way**.



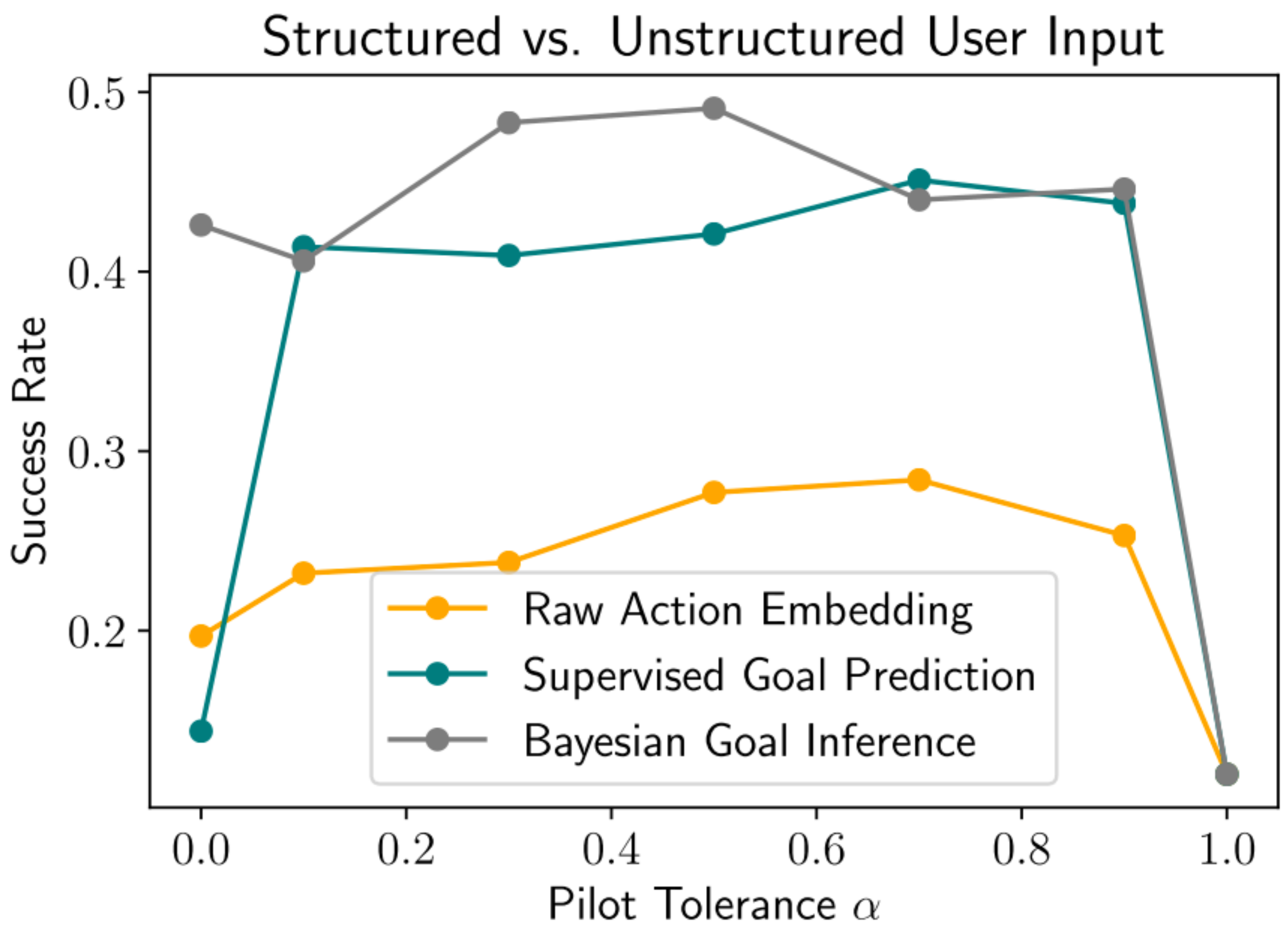
- **Pilot:** Human, knows target orientation.
- **Copilot:** Our Agent, knows where pad is, but not target orientation.

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.

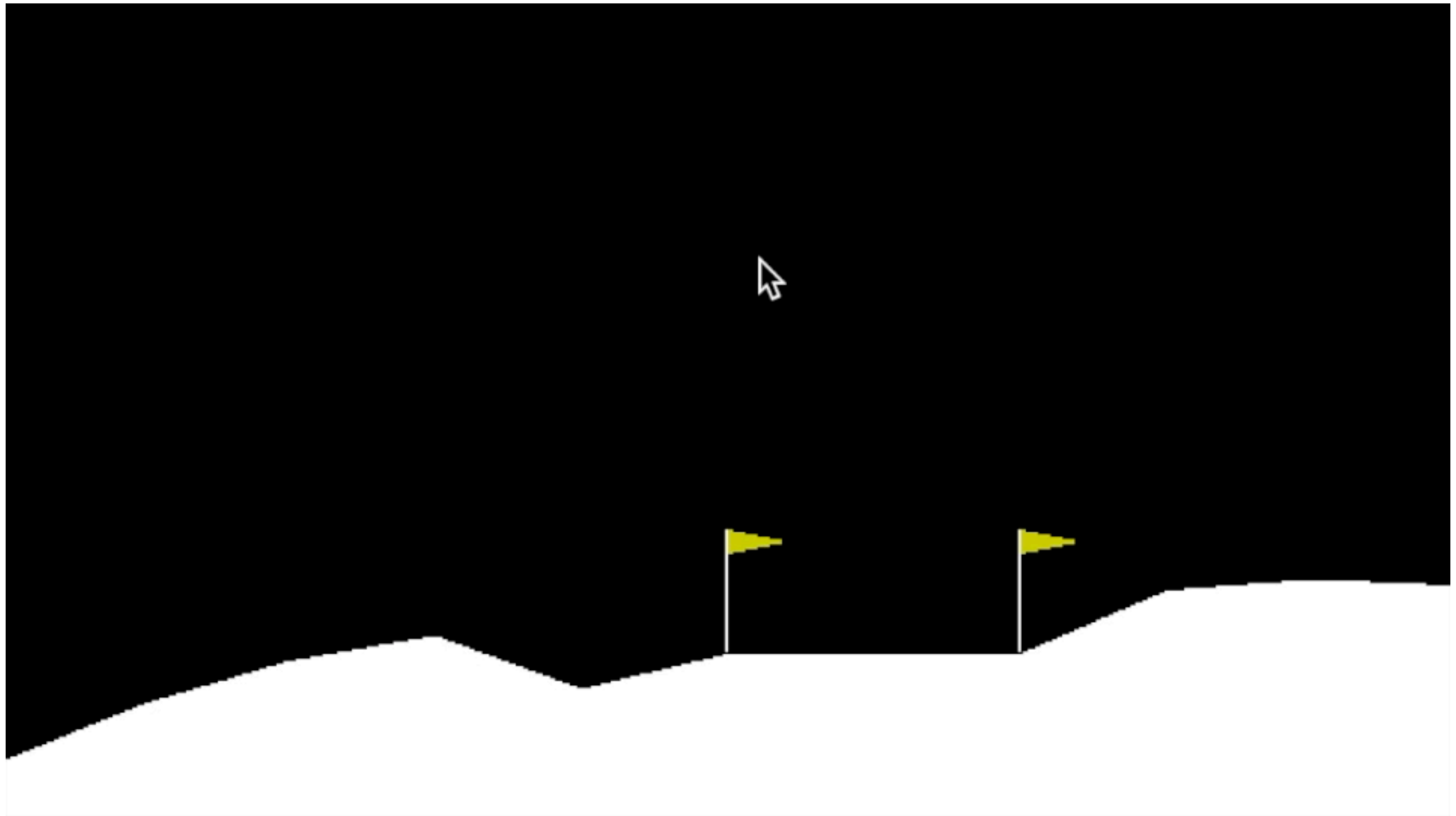
Conclusion

- 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.