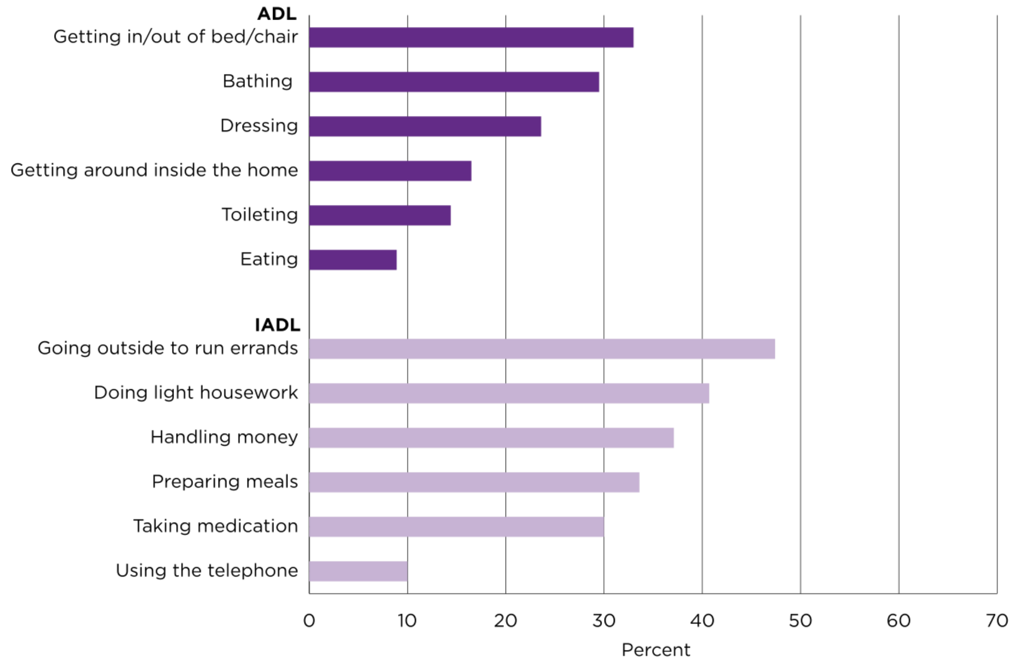
collab

collab.me.vt.edu

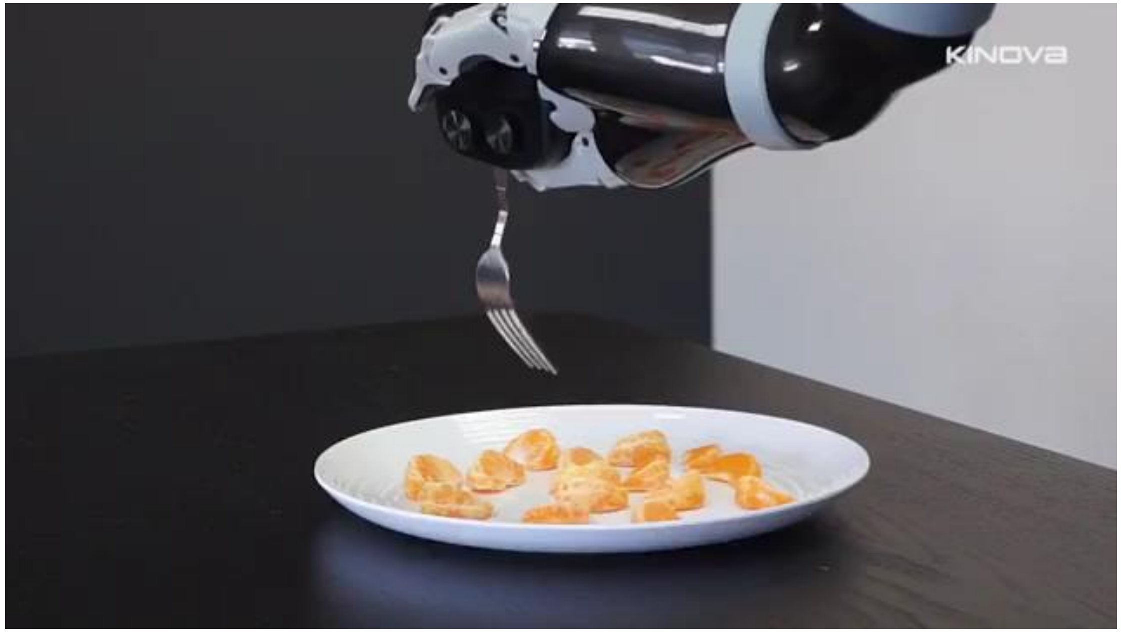
Shared Autonomy

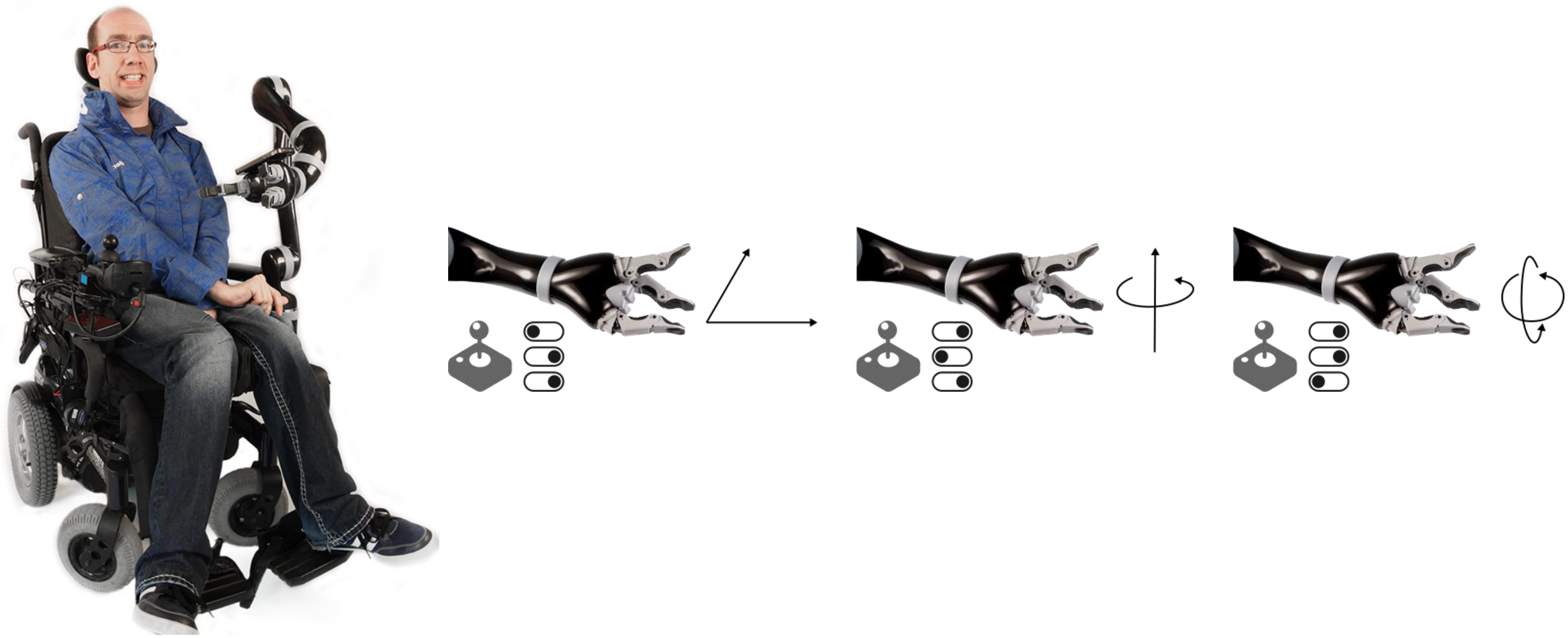
Dylan Losey
Virginia Tech | Fall 2024

Prevalence of Difficulty Performing ADLs and IADLs in Adults 18 Years and Older With One or More Selected Symptoms That Interfere With Everyday Activities: 2014



Source: U.S. Census Bureau, Social Security Administration Supplement to the 2014 Panel of the Survey of Income and Program Participation, September–November 2014.

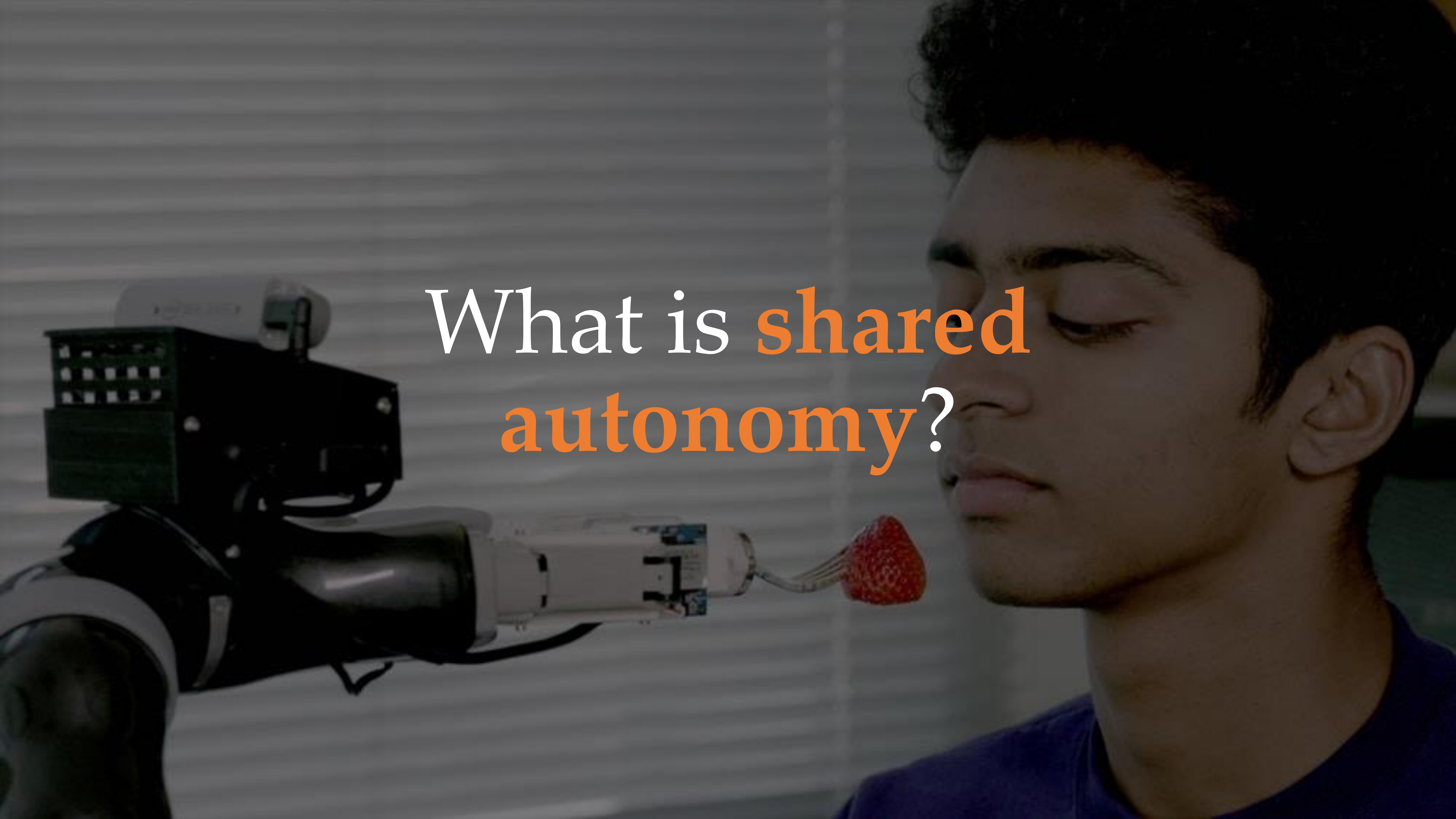




This Lecture

- Introducing shared autonomy
- One flavor of solutions: predicting goals
- Another flavor of solutions: predicting actions

What is **shared autonomy**?



Shared Autonomy

Consider a system with:

- State s
- Human input u_H

The system maps the human's input into a commanded action:

$$a_H^t = \phi(s^t, u_H^t)$$

Shared Autonomy

Shared Autonomy

Consider a system with:

- State s
- Human input u_H
- Assistive action a_R

The dynamics depend on both the human's input and the robot's assistance:

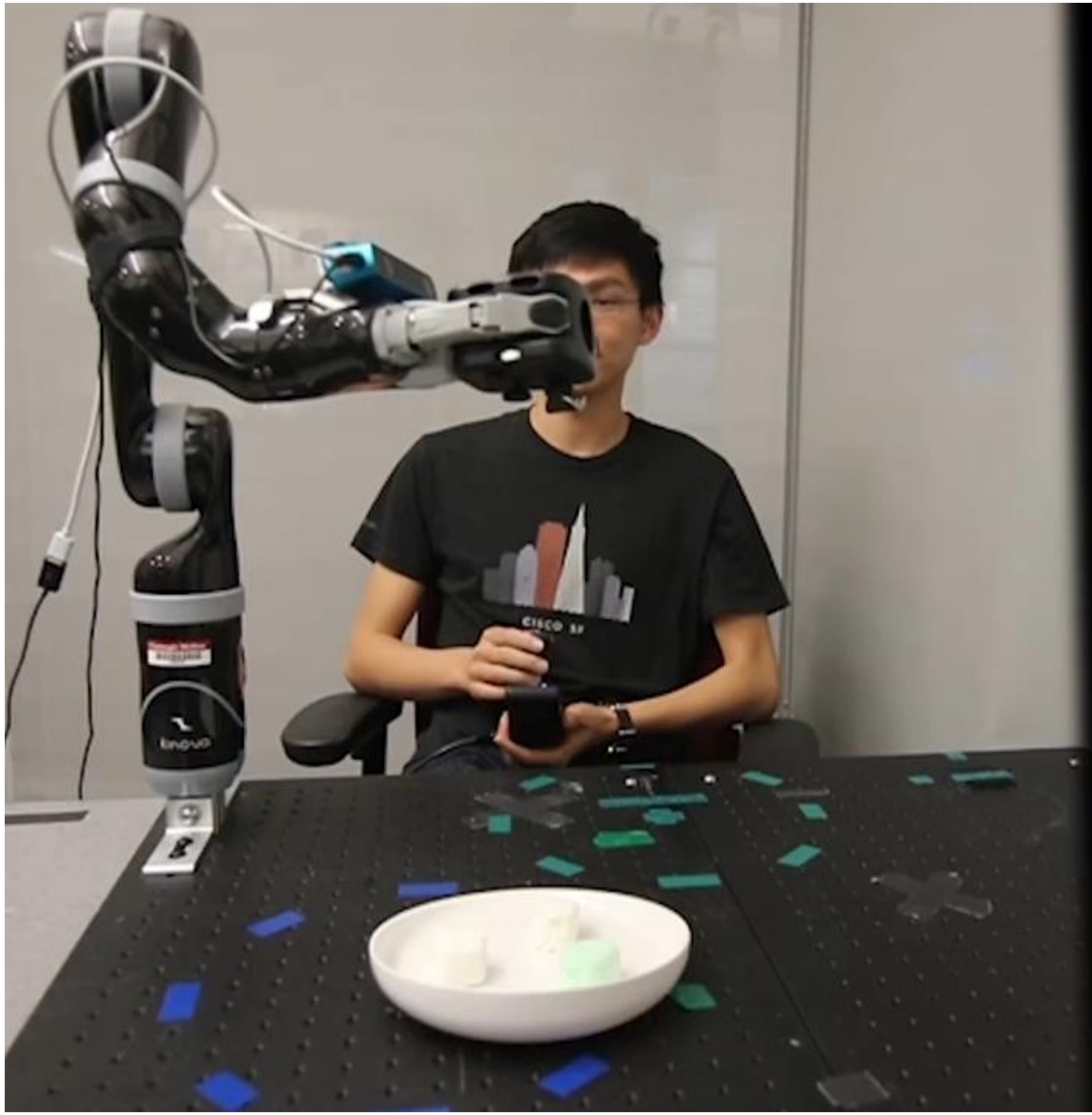
$$s^{t+1} = f\left(s^t, a_R^t, \phi(s^t, u_H^t)\right)$$

Shared Autonomy

Shared Autonomy

Shared Autonomy

Shared Autonomy: Predicting Goals



Assistive arms allow people with disabilities and robots to work together to perform tasks, like eating.

Predicting Goals

One approach is to think of shared autonomy as an **optimization problem** under uncertainty. The human knows the reward (i.e., the goal), and the robot needs to predict that goal, and take assistive actions towards the goal.

$$\mathcal{M} = \langle S, A_R, U_H, f, \phi, \theta \rangle$$

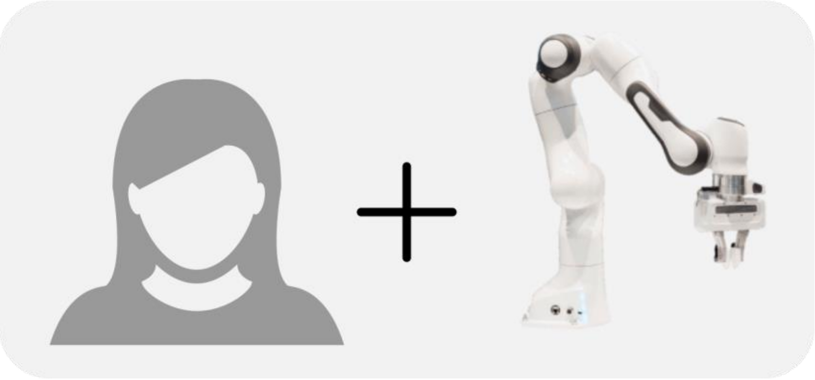
Shared autonomy written as a Markov decision process (can be extended to POMDP)

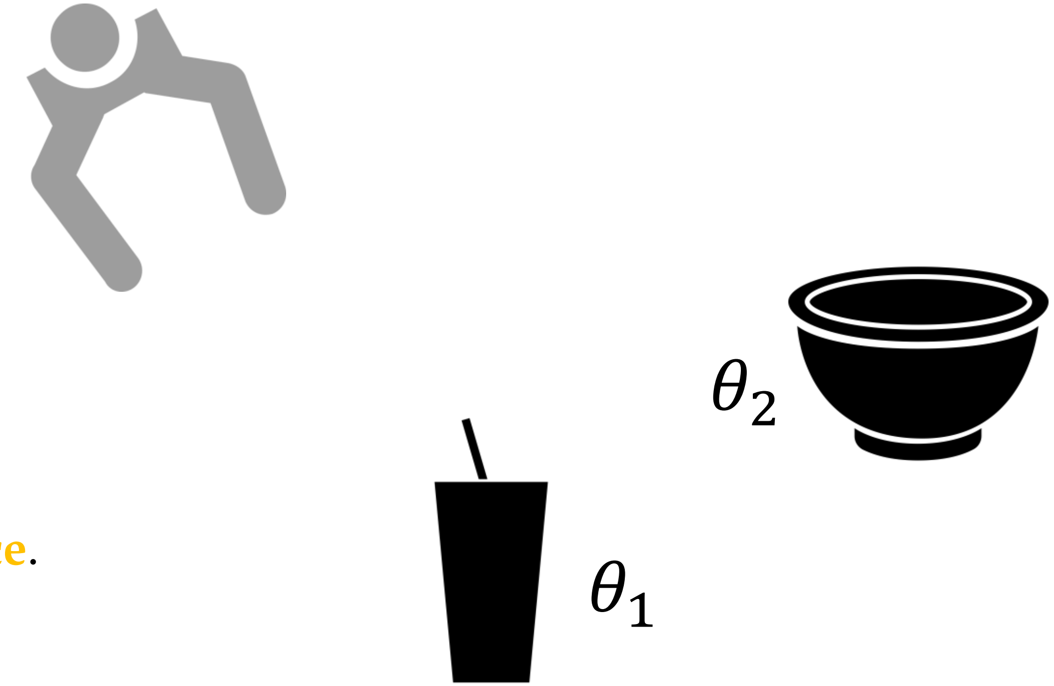
Predicting Goals

$$\mathcal{M} = \langle S, A_R, U_H, f, \phi, \theta \rangle$$

- S is the set of states
- A_R and U_H are the set of assistive actions and human inputs
- f and ϕ are the known dynamics (including the input mapping)
- $r(s, \theta)$ is the **reward function** that the robot should optimize for
- θ is the human's goal, which the robot does not know *a priori*

Predict & Blend



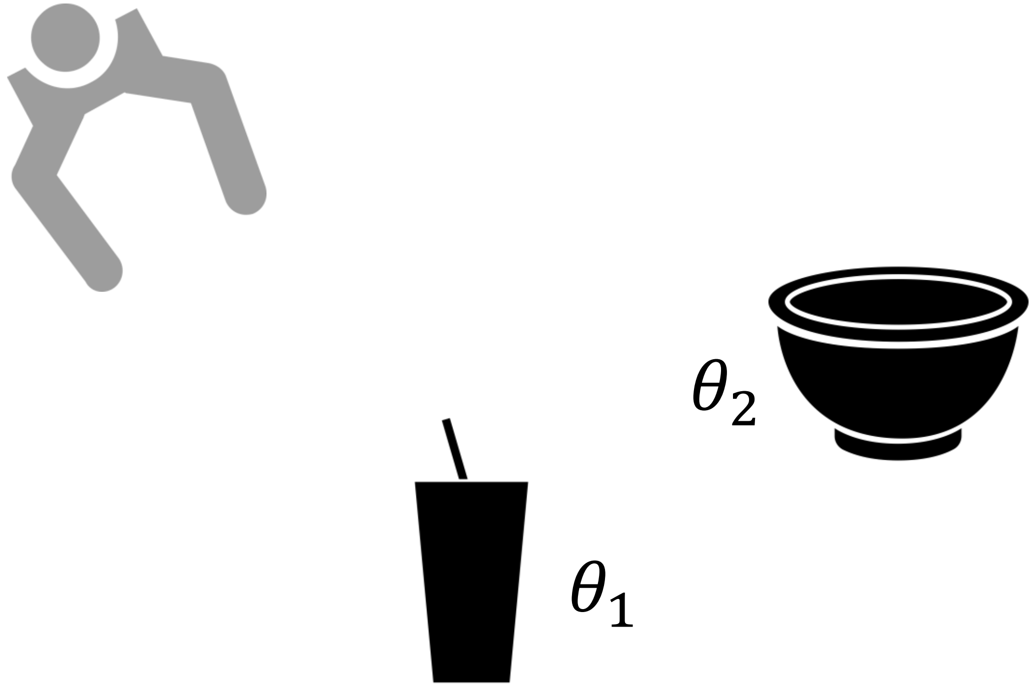


Imagine the human and robot are trying to reach for a **goal** in **free space**.

Predict & Blend

Step 1 (Predict): Infer which goal the human is trying to reach

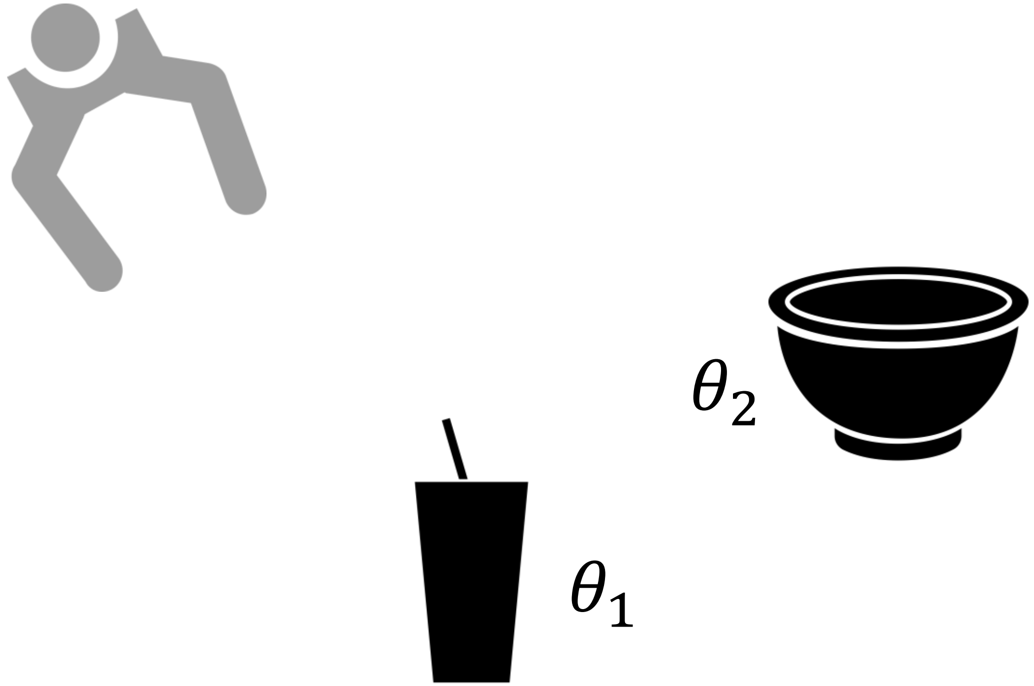
Step 2 (Blend): Blend the human's commanded action with assistive action



Predict & Blend

Step 1 (Predict): Infer which goal the human is trying to reach

Step 2 (Blend): Blend the human's commanded action with assistive action



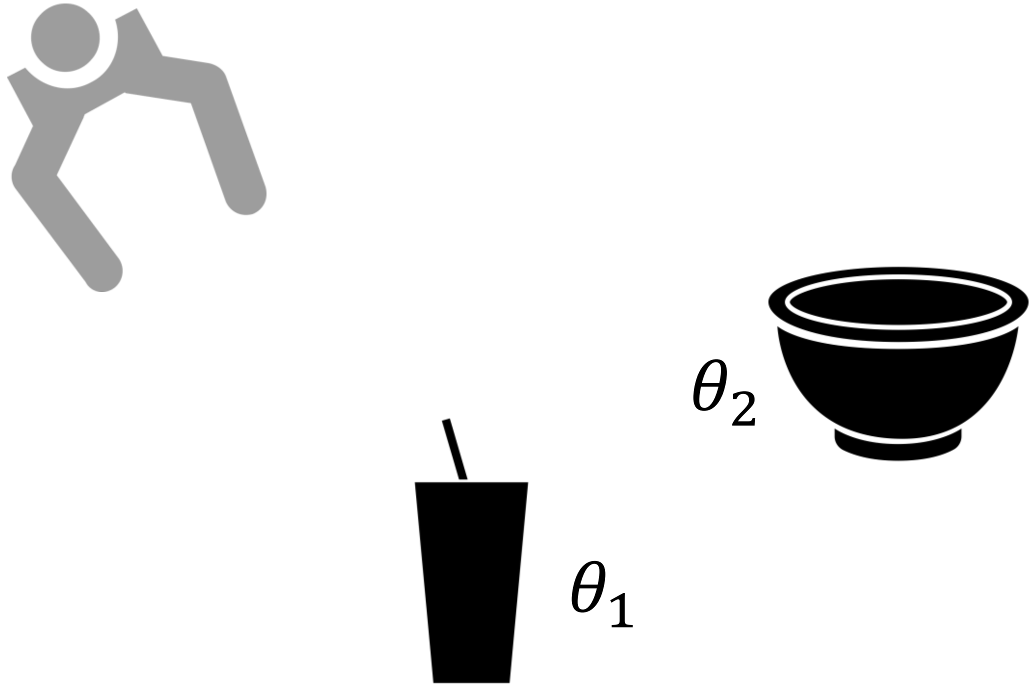
Predict & Blend

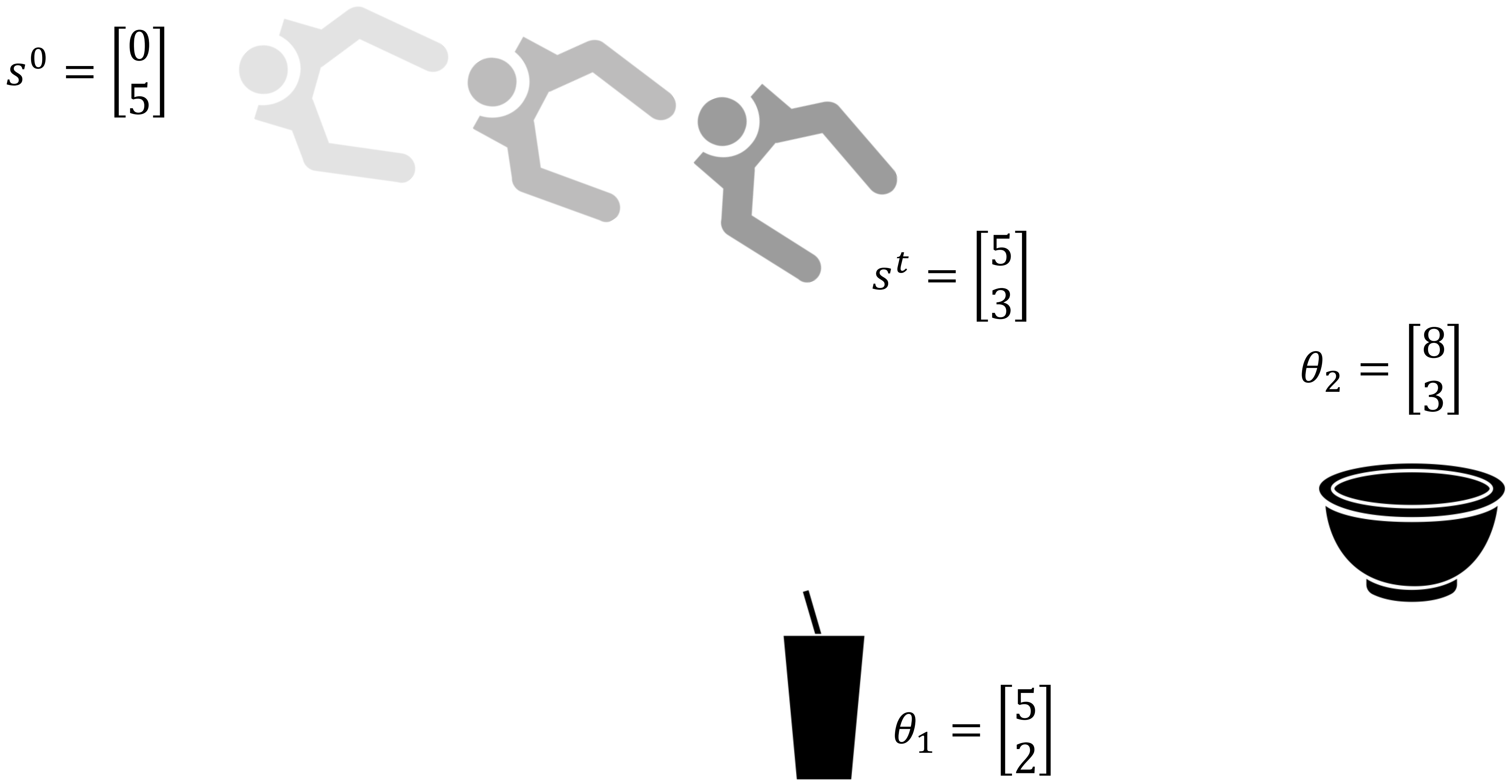
Step 1 (Predict): Infer which goal the human is trying to reach

- Start with prior $P(\theta)$
- At each timestep get (s, a_H)
- Update posterior:

$$P(\theta \mid D) \propto P(\theta) \prod_{(s,a_H)\in D} P(a_H \mid s, \theta)$$

Often people simplify this...




$s^0 = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$
$s^t = \begin{bmatrix} 5 \\ 3 \end{bmatrix}$
$\theta_2 = \begin{bmatrix} 8 \\ 3 \end{bmatrix}$
$\theta_1 = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$

$$s^0 = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$$

$$s^t = \begin{bmatrix} 5 \\ 3 \end{bmatrix}$$

$$\theta_2 = \begin{bmatrix} 8 \\ 3 \end{bmatrix}$$

Naïve approach

Probability human wants a goal is inverse prop. to distance from goal

$$P(\theta \mid D) \propto \frac{1}{\underbrace{\|\theta - s^t\|}_{\textit{Distance left to } \theta}}$$

$$\theta_1 = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$$

$$s^0 = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$$

$$s^t = \begin{bmatrix} 5 \\ 3 \end{bmatrix}$$

Naïve approach

Probability human wants a goal is inverse prop. to distance from goal

$$P(\theta \mid D) \propto \frac{1}{\|\theta - s^t\|}$$

$P(\theta_2 | s^t) = 0.25$

$P(\theta_1 | s^t) = 0.75$

$$s^0 = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$$

$$s^t = \begin{bmatrix} 5 \\ 3 \end{bmatrix}$$

Better approach

Consider how efficiently human is moving towards the goal

$$P(\theta \mid D) \propto \frac{\|\theta - s^0\|}{\underbrace{\|s^t - s^0\|}_{\text{Distance gone so far}} + \|\theta - s^t\|}$$

Distance gone so far

Distance left to θ

$$\theta_2 = \begin{bmatrix} 8 \\ 3 \end{bmatrix}$$

$$\theta_1 = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$$

$$s^0 = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$$

$$s^t = \begin{bmatrix} 5 \\ 3 \end{bmatrix}$$

Better approach

Consider how efficiently human is moving towards the goal

$$P(\theta \mid D) \propto \frac{\|\theta - s^0\|}{\|s^t - s^0\| + \|\theta - s^t\|}$$

$$P(\theta_2 | s^t) = 0.52$$

$$P(\theta_1 | s^t) = 0.48$$

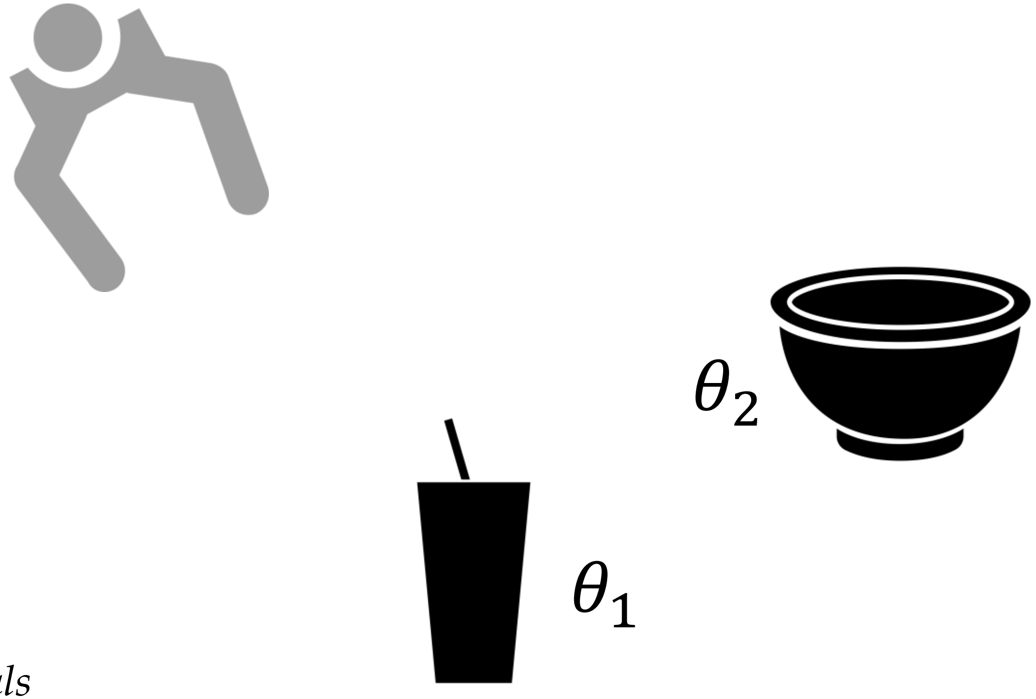
Predict & Blend

Step 1 (Predict): Infer which goal the human is trying to reach

- Start with prior $P(\theta)$
- At each timestep get (s, a_H)
- Update posterior:

$$P(\theta \mid D) \propto P(\theta) \frac{\exp(\beta \cdot \|\theta - s^0\|)}{\exp(\beta\|s^t - s^0\| + \beta\|\theta - s^t\|)}$$

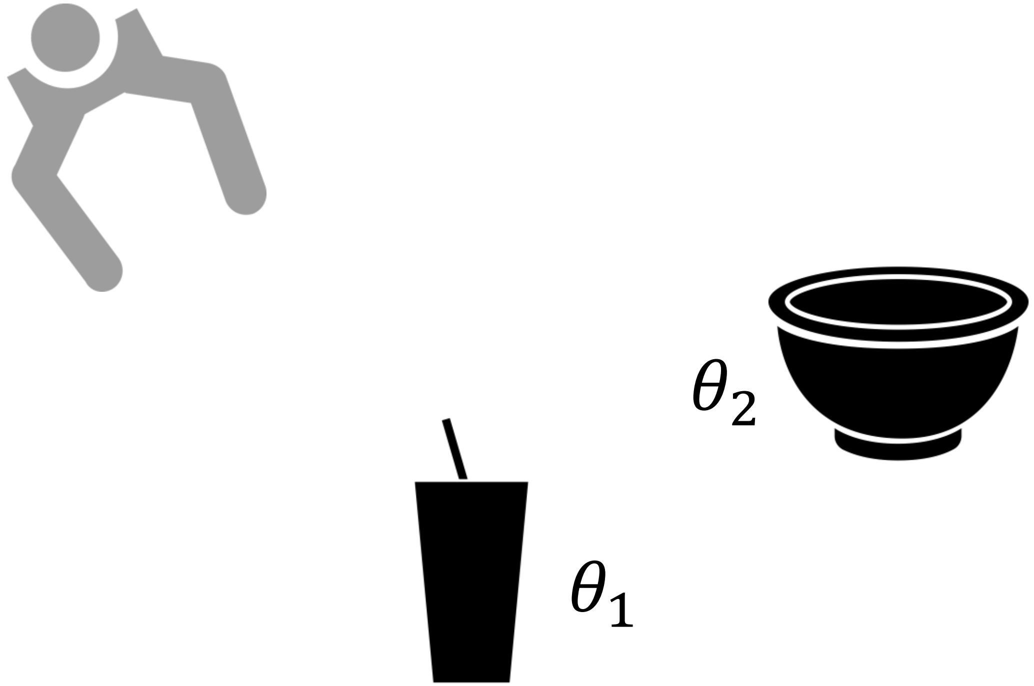
One common simplification for free space goals



Predict & Blend

Step 1 (Predict): Infer which goal the human is trying to reach

Step 2 (Blend): Blend the human's commanded action with assistive action

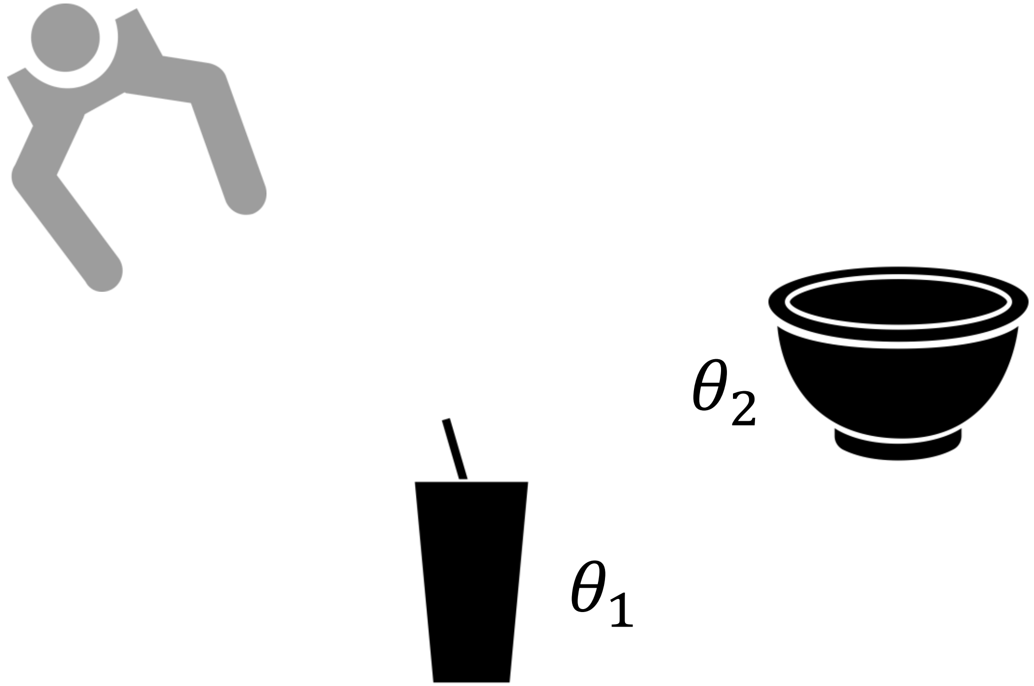


Predict & Blend

Step 2 (Blend): Blend the human's commanded action with assistive action

$$a = (1 - \alpha) \cdot a_H + \alpha \cdot a_R$$

Linearly blend the human and robot actions, the robot executes the overall action a



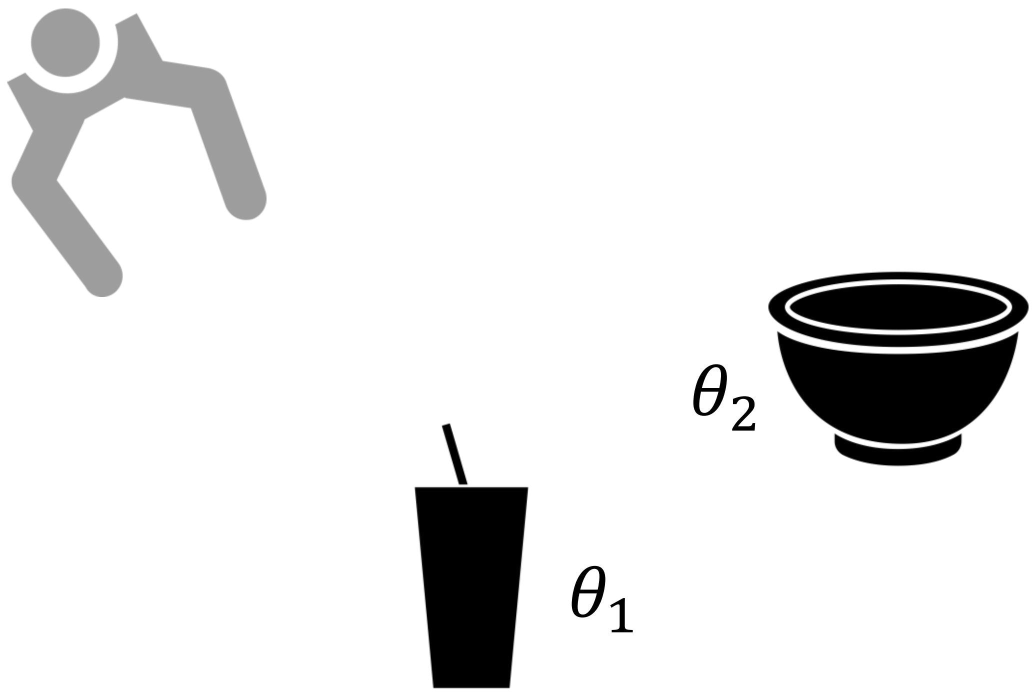
Predict & Blend

Step 2 (Blend): Blend the human's commanded action with assistive action

$$a = (1 - \alpha) \cdot a_H + \alpha \cdot a_R$$

We know this…

What about the assistive robot action?



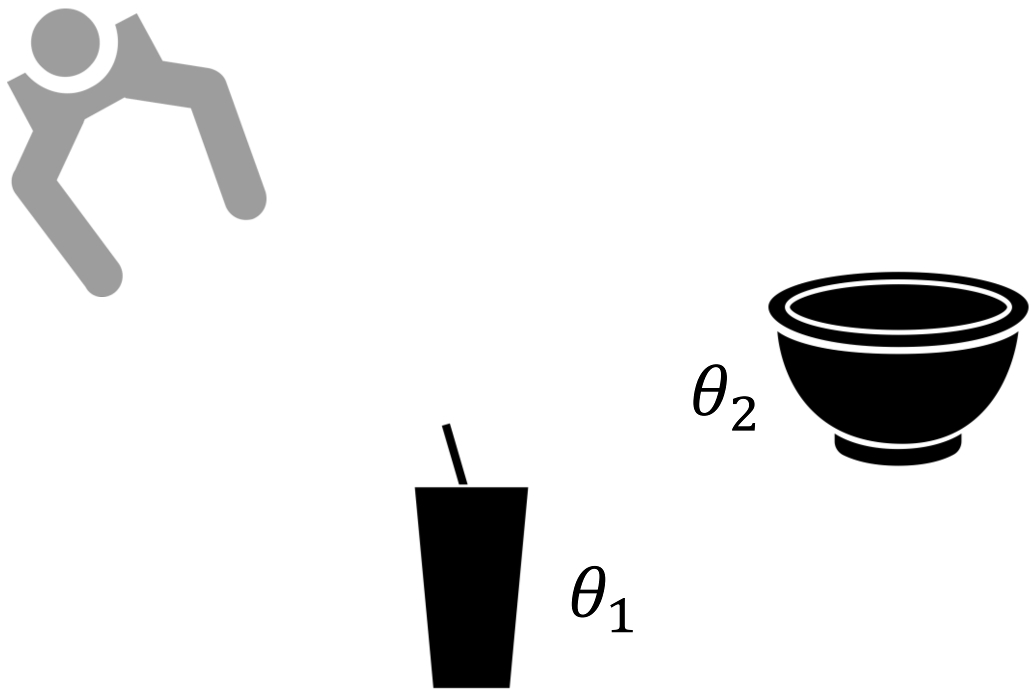
Predict & Blend

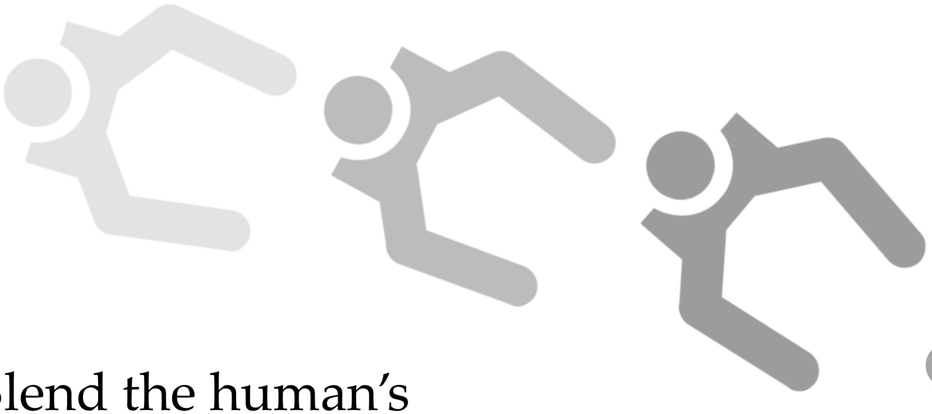
Step 2 (Blend): Blend the human's commanded action with assistive action

$$\boxed{a = (1-\alpha)\cdot a_H + \alpha \cdot a_R}$$

$$a_R = \sum_{\theta \in \Theta} P(\theta|D)\cdot(\theta - s^t)$$

Assist towards weighted average goal



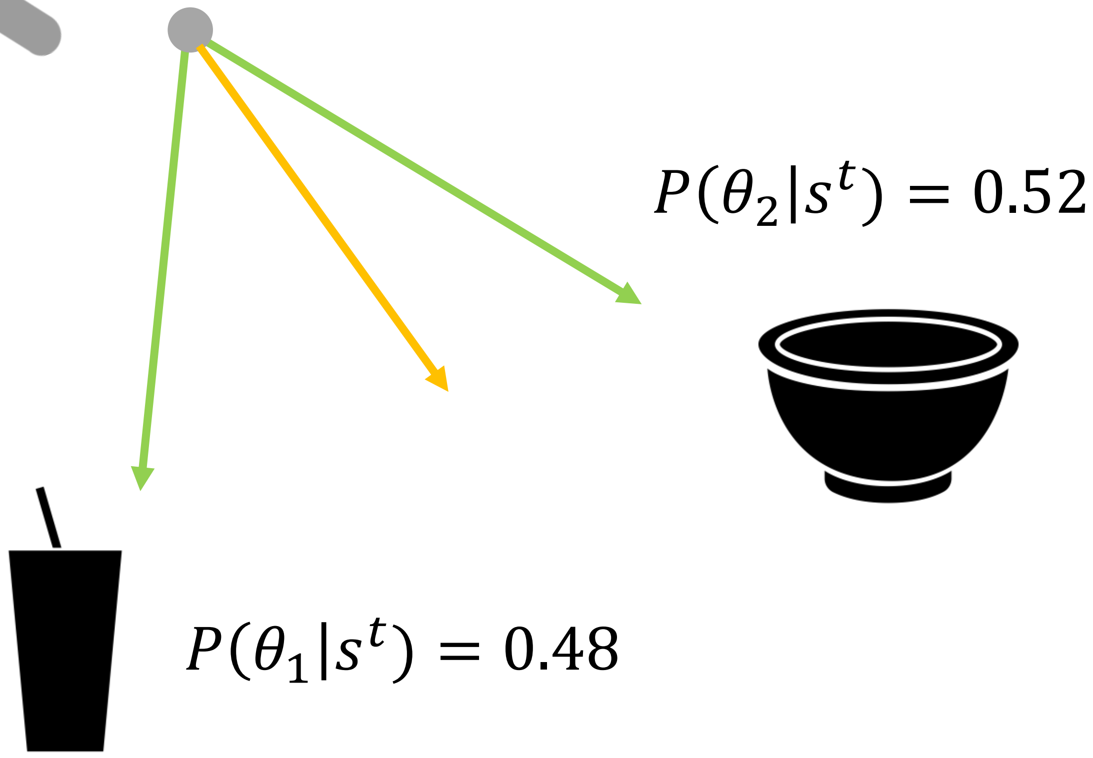


Step 2 (Blend): Blend the human's commanded action with assistive action

$$a = (1 - \alpha) \cdot a_H + \alpha \cdot a_R$$

$$a_R = \sum_{\theta \in \Theta} \underline{P(\theta | D) \cdot (\theta - s^t)}$$

Assist towards weighted average goal



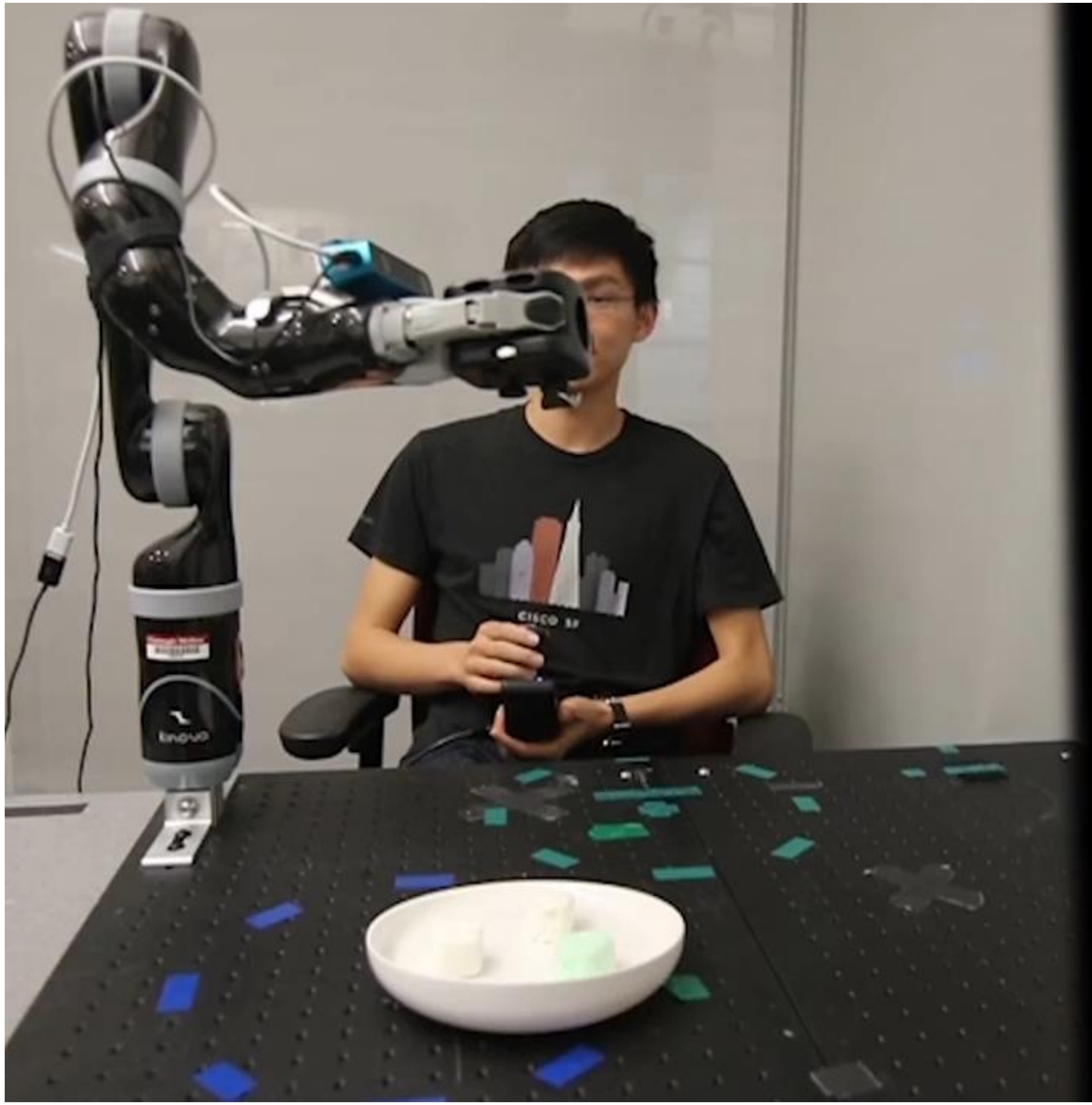
Predict & Blend

Given set of possible goals $\theta \in \Theta$

Given prior over goals $P(\theta)$

For each timestep t

- Measure state s and human input a_H
- **Predict** $P(\theta \mid D) \propto P(\theta) \prod_{(s,a_H)\in D} P(a_H \mid s, \theta)$
- Compute assistance $a_R = \sum_{\theta\in\Theta} P(\theta|D) \cdot (\theta - s^t)$
- Take **blended** action $a = (1-\alpha) \cdot a_H + \alpha \cdot a_R$

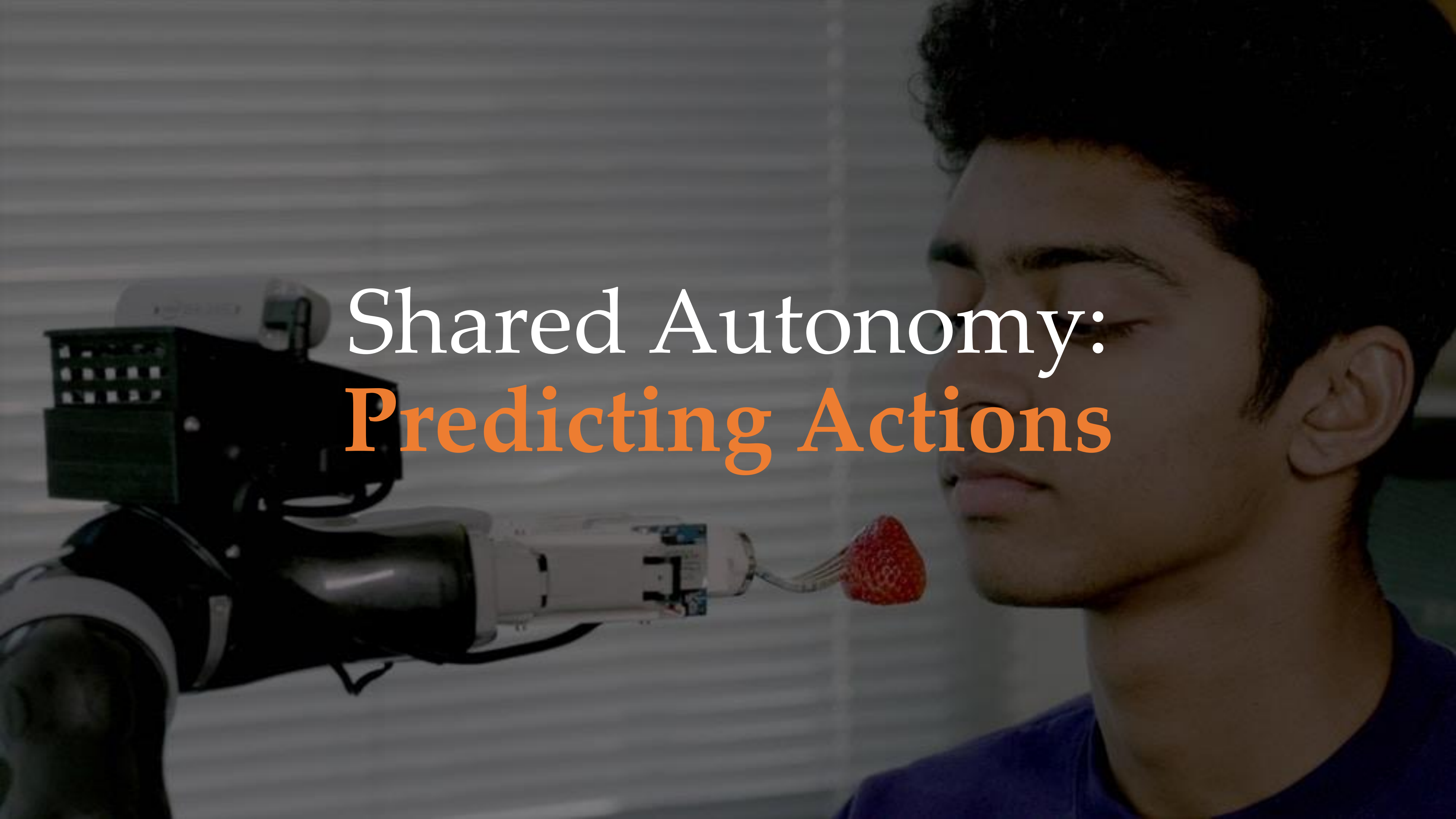


Assistive arms allow people with disabilities and robots to work together to perform tasks, like eating.

Related Papers

- https://journals.sagepub.com/doi/full/10.1177/0278364913490324
- https://journals.sagepub.com/doi/full/10.1177/0278364918776060
- https://dl.acm.org/doi/pdf/10.1145/3359614

Shared Autonomy: **Predicting Actions**



Shared Autonomy

Consider a system with:

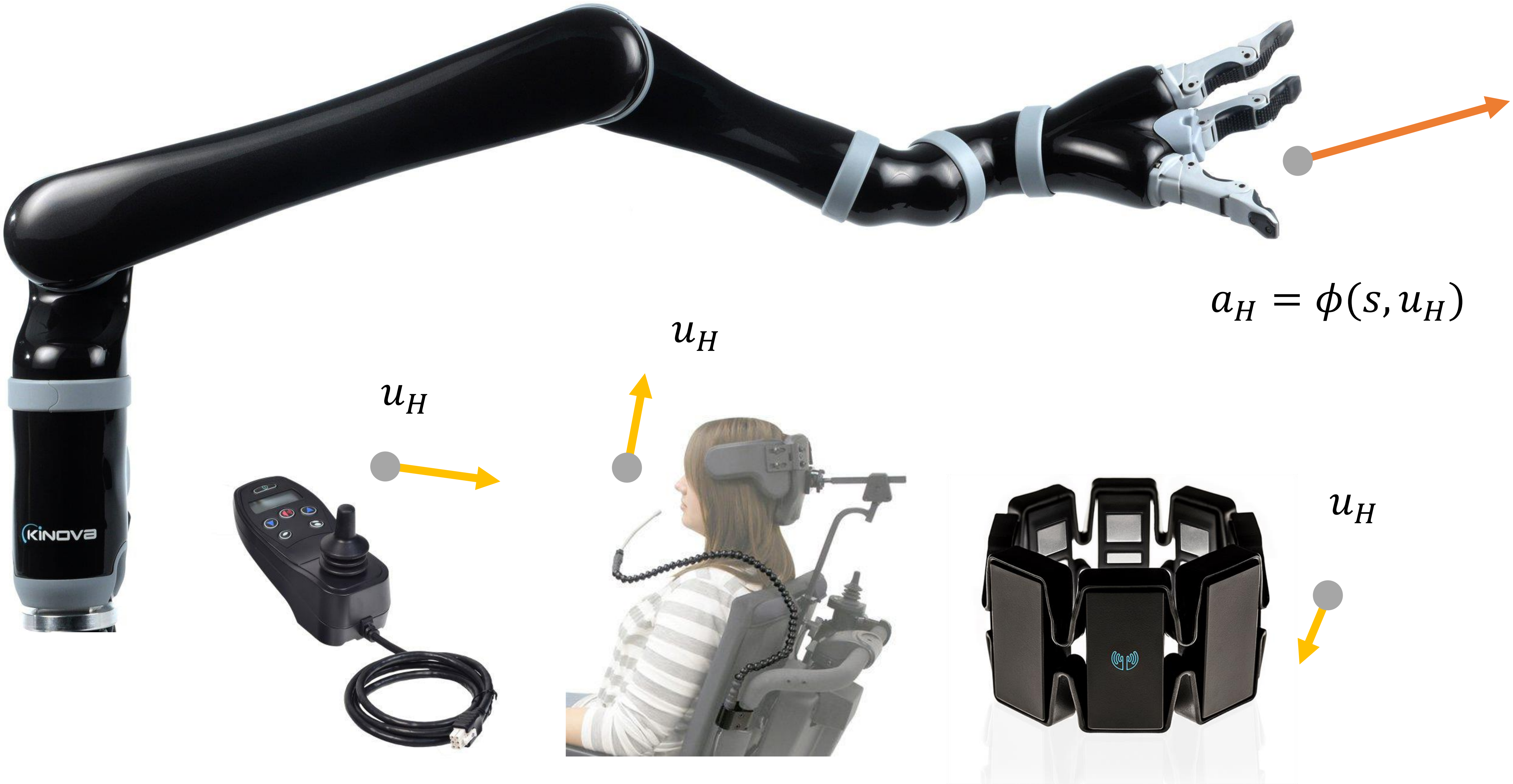
- State s
- Human input u_H

The system maps the human's input into a commanded action:

$$a_H^t = \phi(s^t, u_H^t)$$

Shared Autonomy

The robot autonomy generates trajectory segments



$a_H = \phi(s, u_H)$
u_H
u_H
u_H
KINOVA

Predicting Actions

With assistive applications in mind, the human's input is often **low-dimensional**. But the robot the human is trying to control is **high-dimensional**. Instead of assuming access to a discrete set of goals, can we enable the human to seamlessly control their complex and dexterous robot arm?

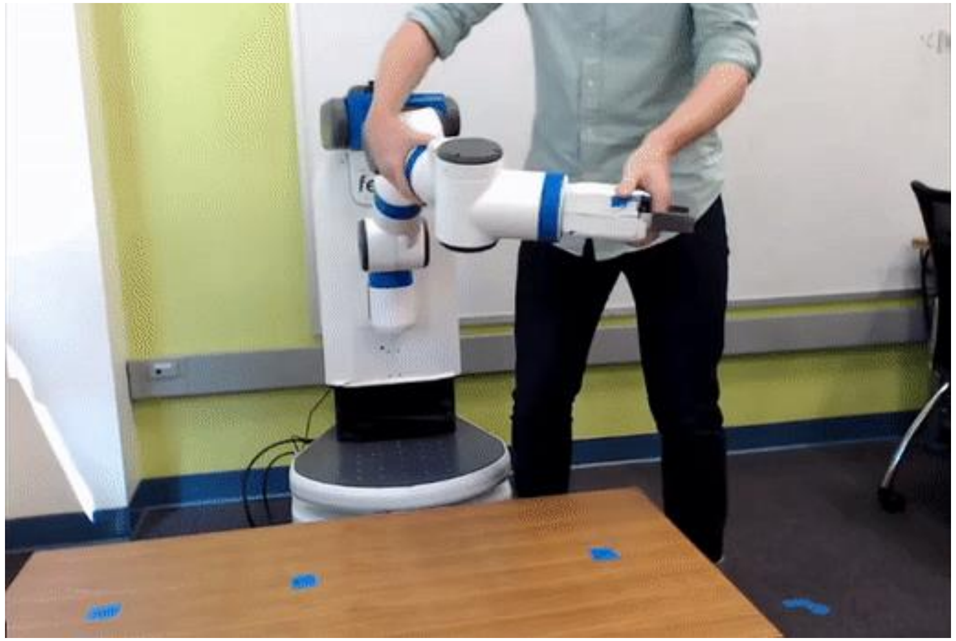
$$a_H = \phi(s, u_H)$$

These approaches learn a mapping from states and inputs to commanded robot actions

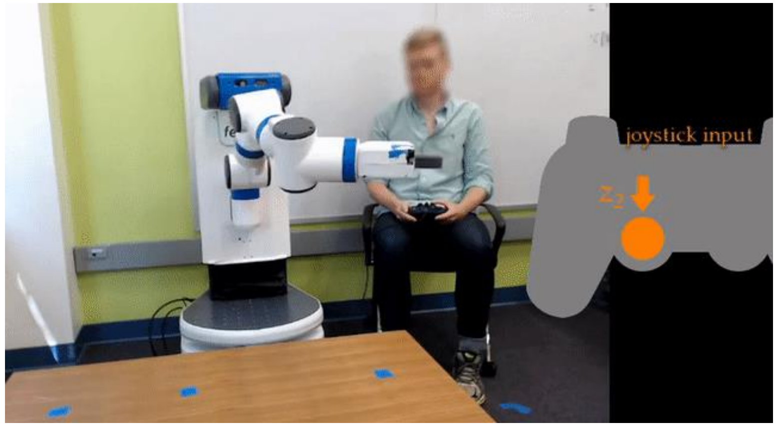
Predicting Actions

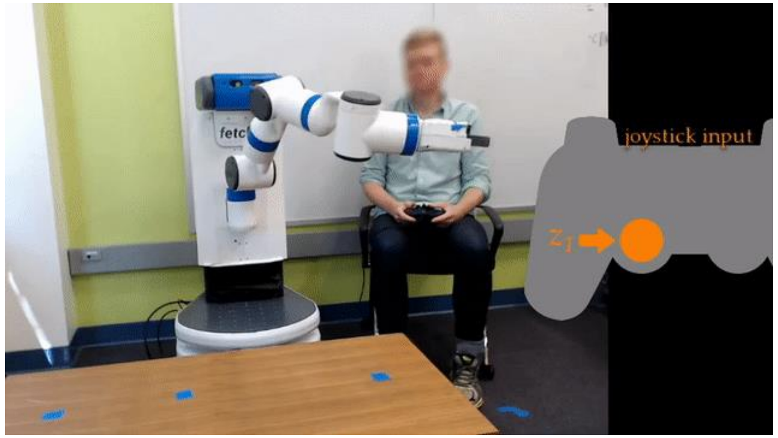
$$a_H = \phi(s, u_H)$$

- s is the current state (i.e., joint position + any camera images)
- u_H is the human's low-dimensional input (i.e., 2-DoF joystick)
- ϕ is the unknown teleoperation mapping from inputs to actions
- a_H is the **high-dimensional** action the human wants the robot to take



(s, a)

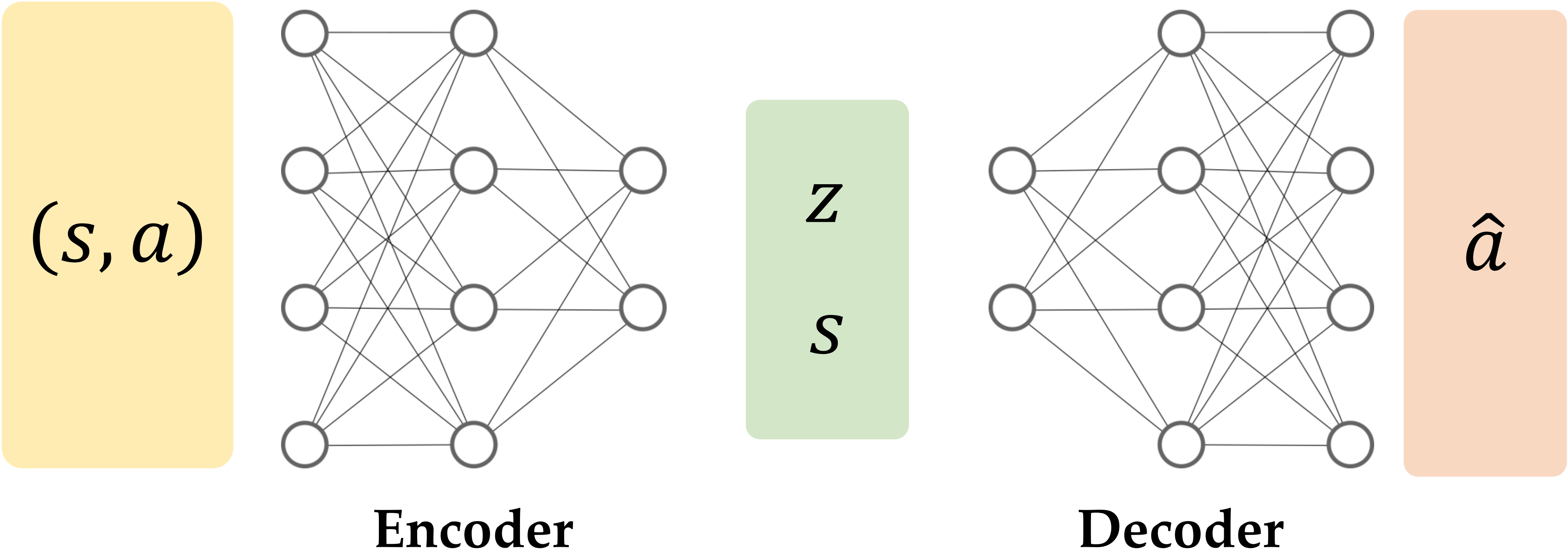




(s, u_H)

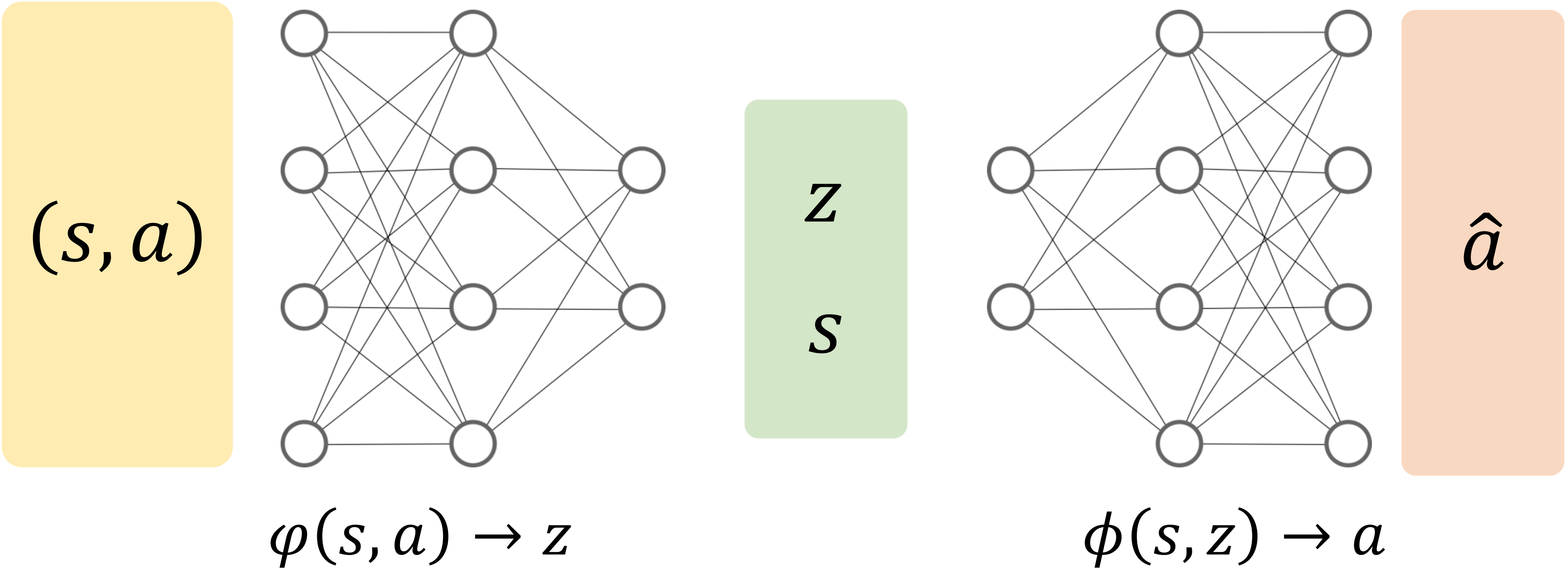
Latent Actions

To learn the mapping $\phi(s, u_H) \rightarrow a_H$ we will use a **conditional autoencoder**.



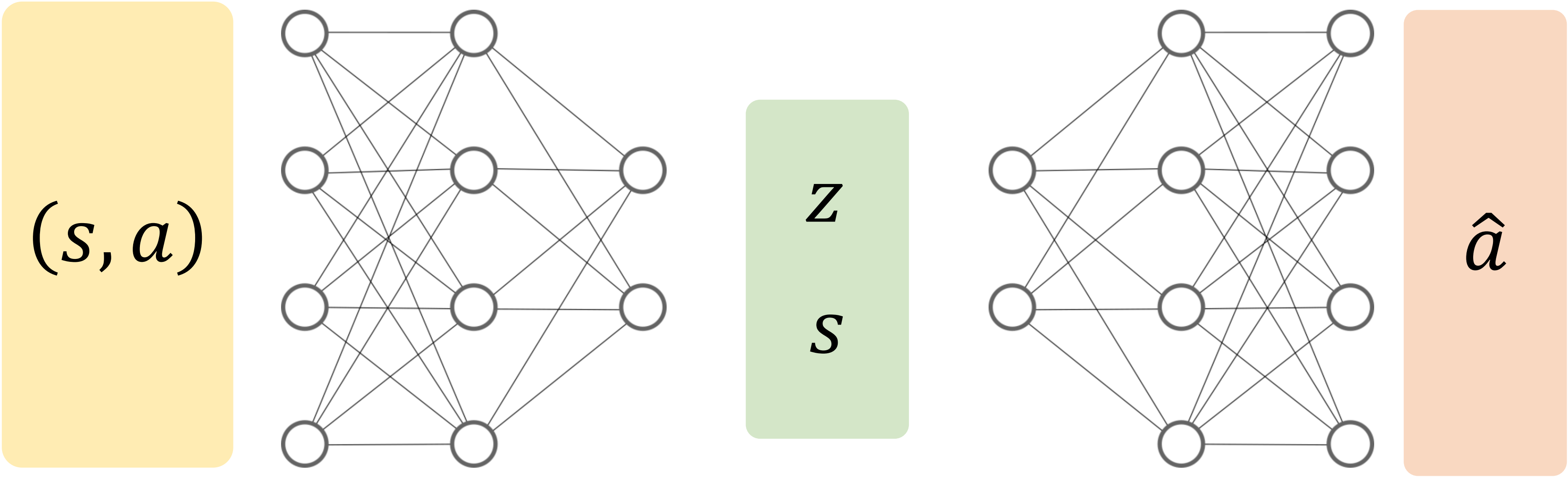
Latent Actions

To learn the mapping $\phi(s, u_H) \rightarrow a_H$ we will use a **conditional autoencoder**.



Latent Actions

To learn the mapping $\phi(s, u_H) \rightarrow a_H$ we will use a **conditional autoencoder**.



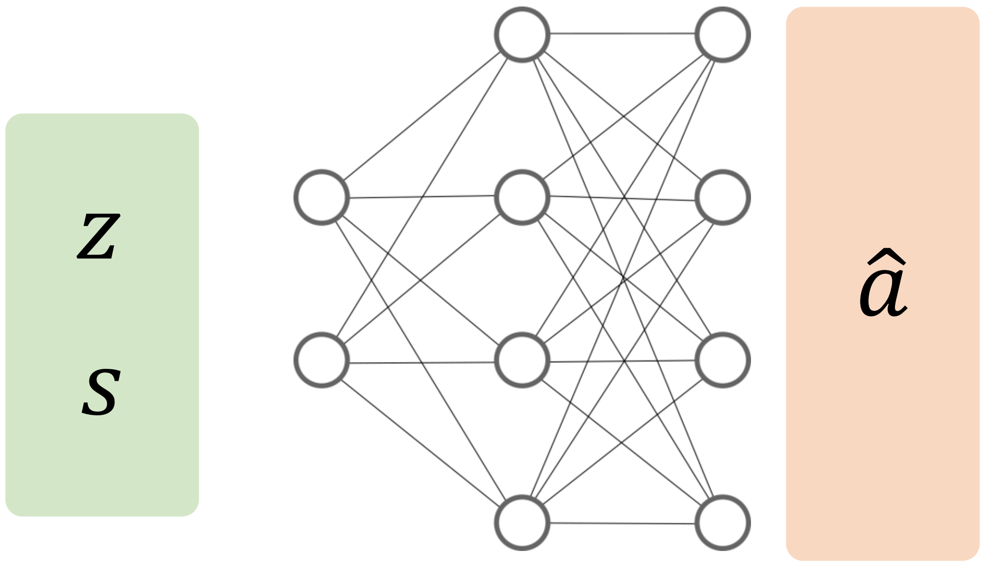
$\varphi(s, a) \rightarrow z$

The decoder is *conditioned* on state s. Take s from input and pass directly to decoder.

Latent Actions

To learn the mapping $\phi(s, z) \rightarrow a$ we will use a **conditional autoencoder**.

Once trained, we control the robot using only the decoder to get action a



Latent Actions

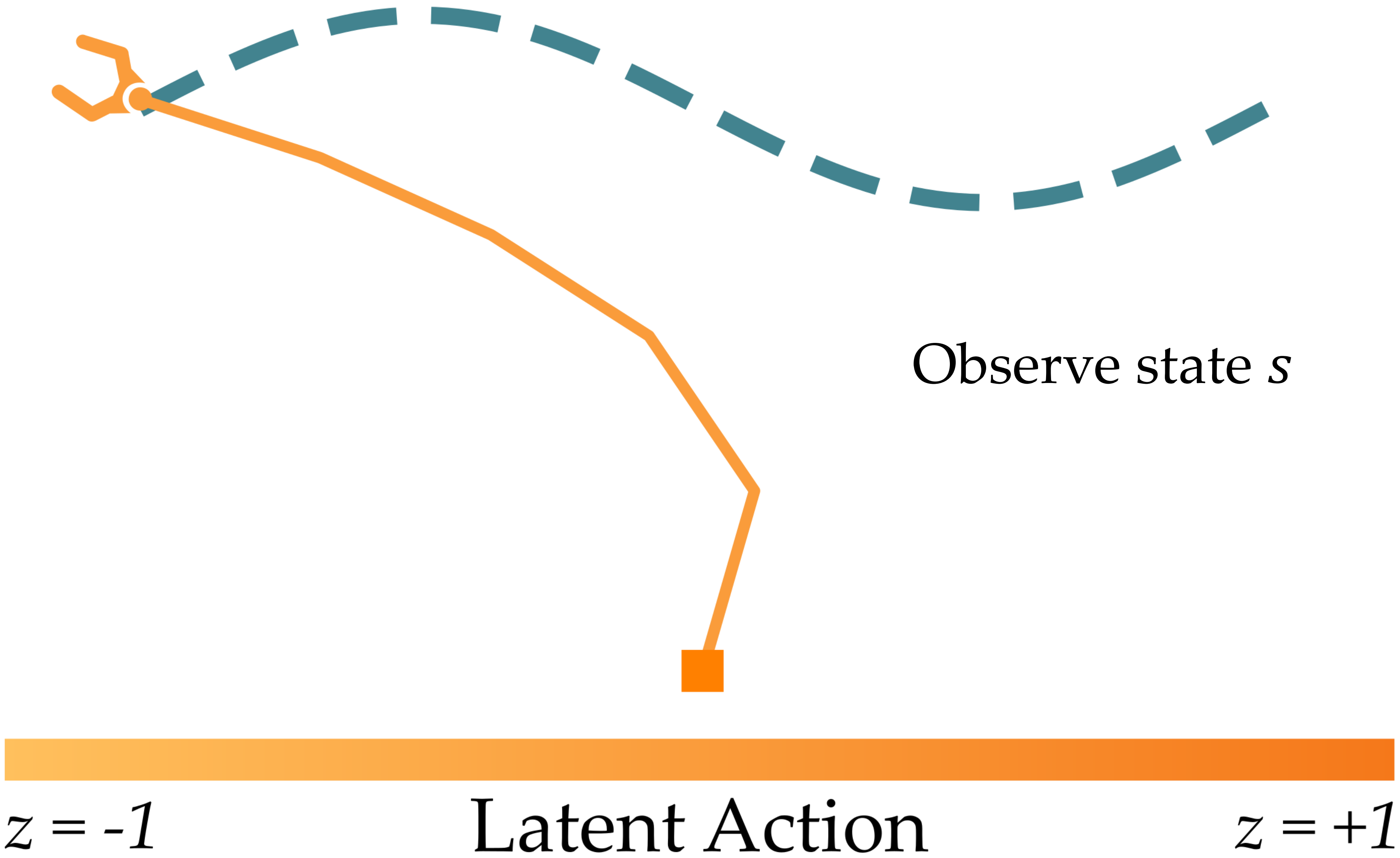
Our idea. We embed high-dimensional and complex tasks to low-dimensional **latent representations**. At runtime users select the latent representation with a joystick, which then maps to high-dimensional and meaningful behaviors.

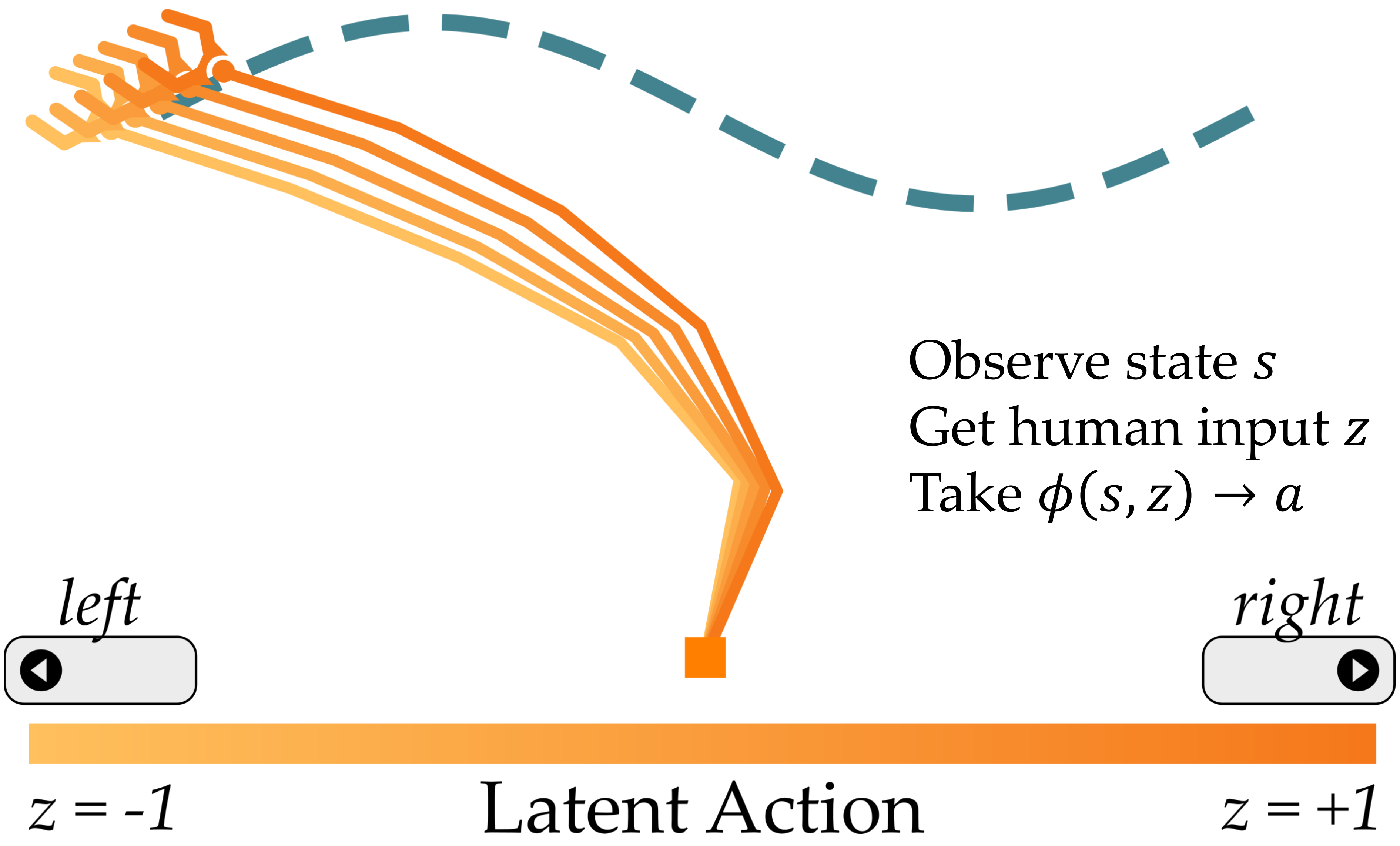
When user presses the joystick to the ***right****, the robot decodes this latent input to help reach the spatula*

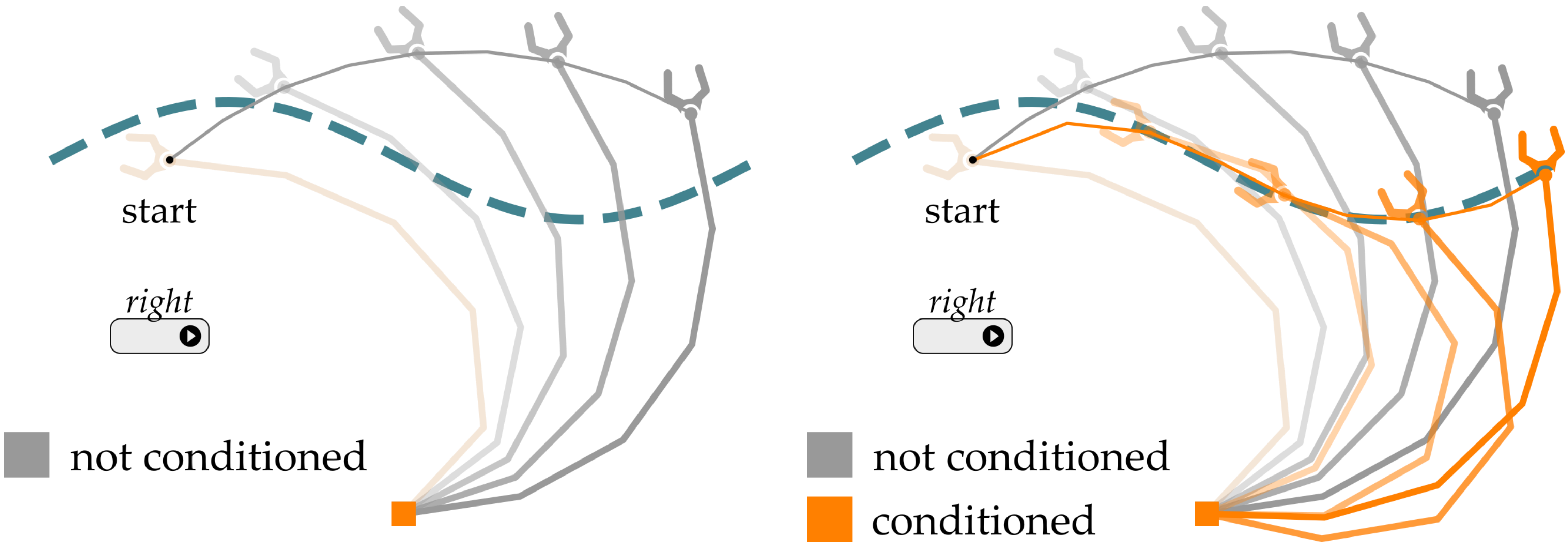
Latent Actions

Our idea. We embed high-dimensional and complex tasks to low-dimensional **latent representations**. At runtime users select the latent representation with a joystick, which then maps to high-dimensional and meaningful behaviors.

*When robot is holding the spatula and user presses the joystick **down**, robot decodes this latent input to help automate a stirring motion*


Observe state s
z = -1
Latent Action
z = +1


Observe state s
Get human input z
Take $\phi(s, z) \rightarrow a$
left
right
z = -1
Latent Action
z = +1


start
right
not conditioned
start
right
not conditioned
conditioned

Latent Actions

Offline collect dataset $D = \{(s^1, a^1), \ldots, (s^N, a^N)\}$

Initialize models:

- Encoder $\varphi(s, a) \rightarrow z$
- Decoder $\phi(s, z) \rightarrow a$

Train encoder and decoder to minimize loss:

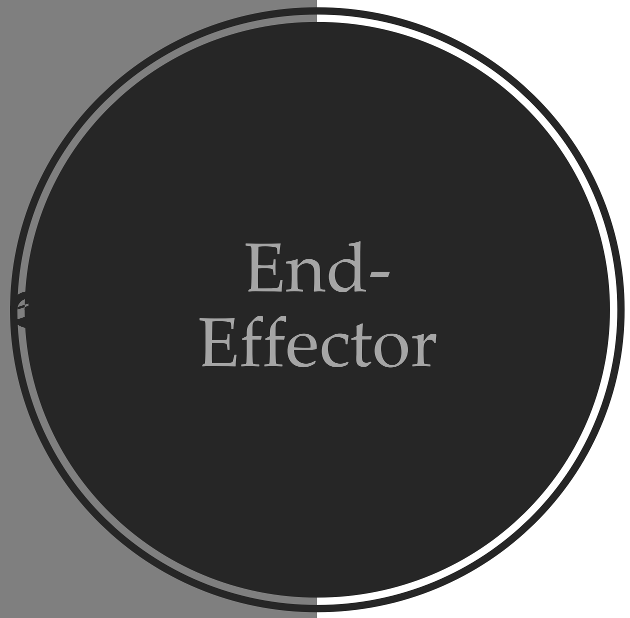
$$\underbrace{\mathcal{L}(\theta)}_{\text{Models have weights } \theta} = \frac{1}{N} \sum_{(s,a) \in D} \underbrace{\left\| a - \phi\big(s, \varphi(s, a)\big) \right\|^2}_{\text{Error between actual and predicted}}$$

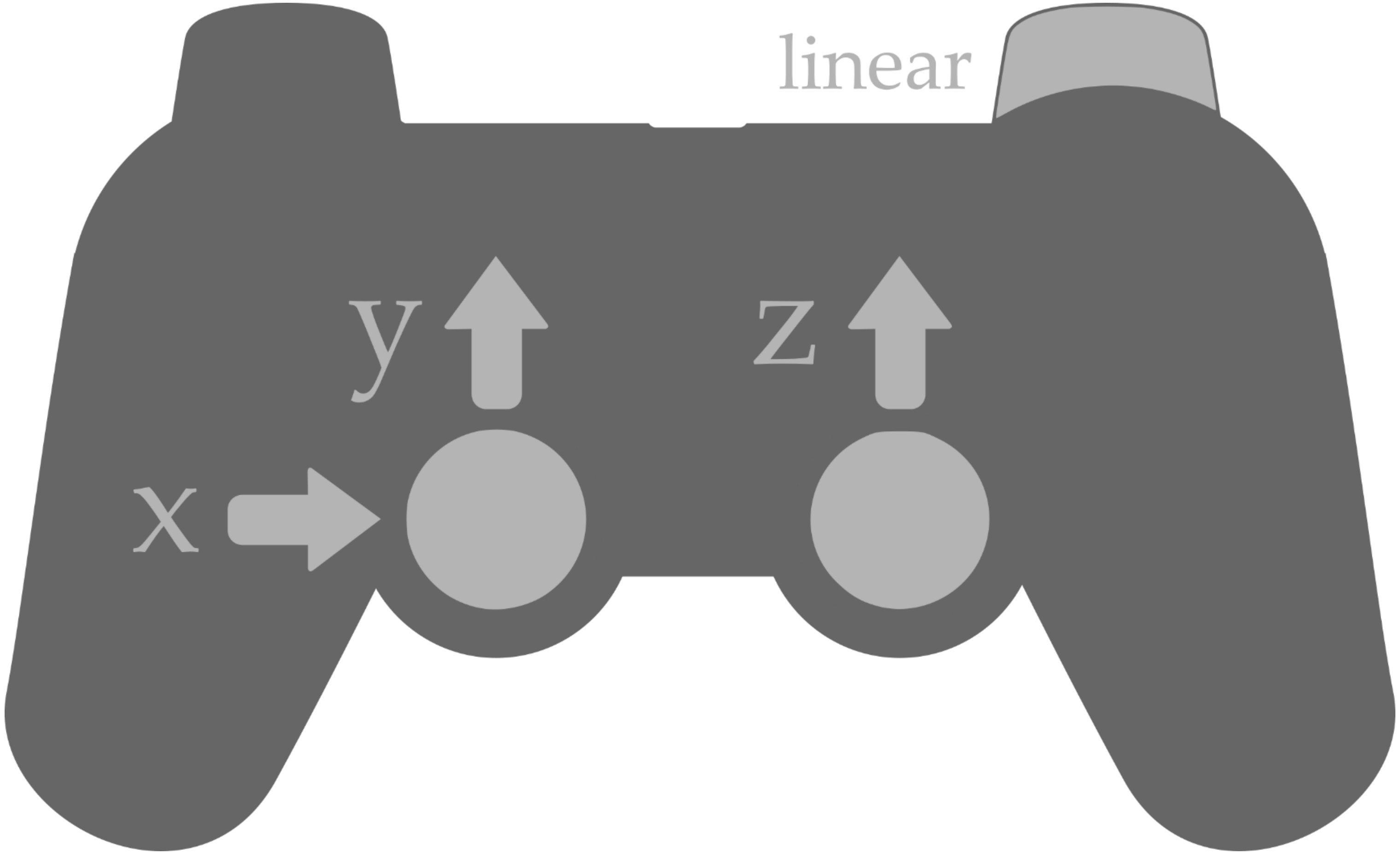
Latent Actions

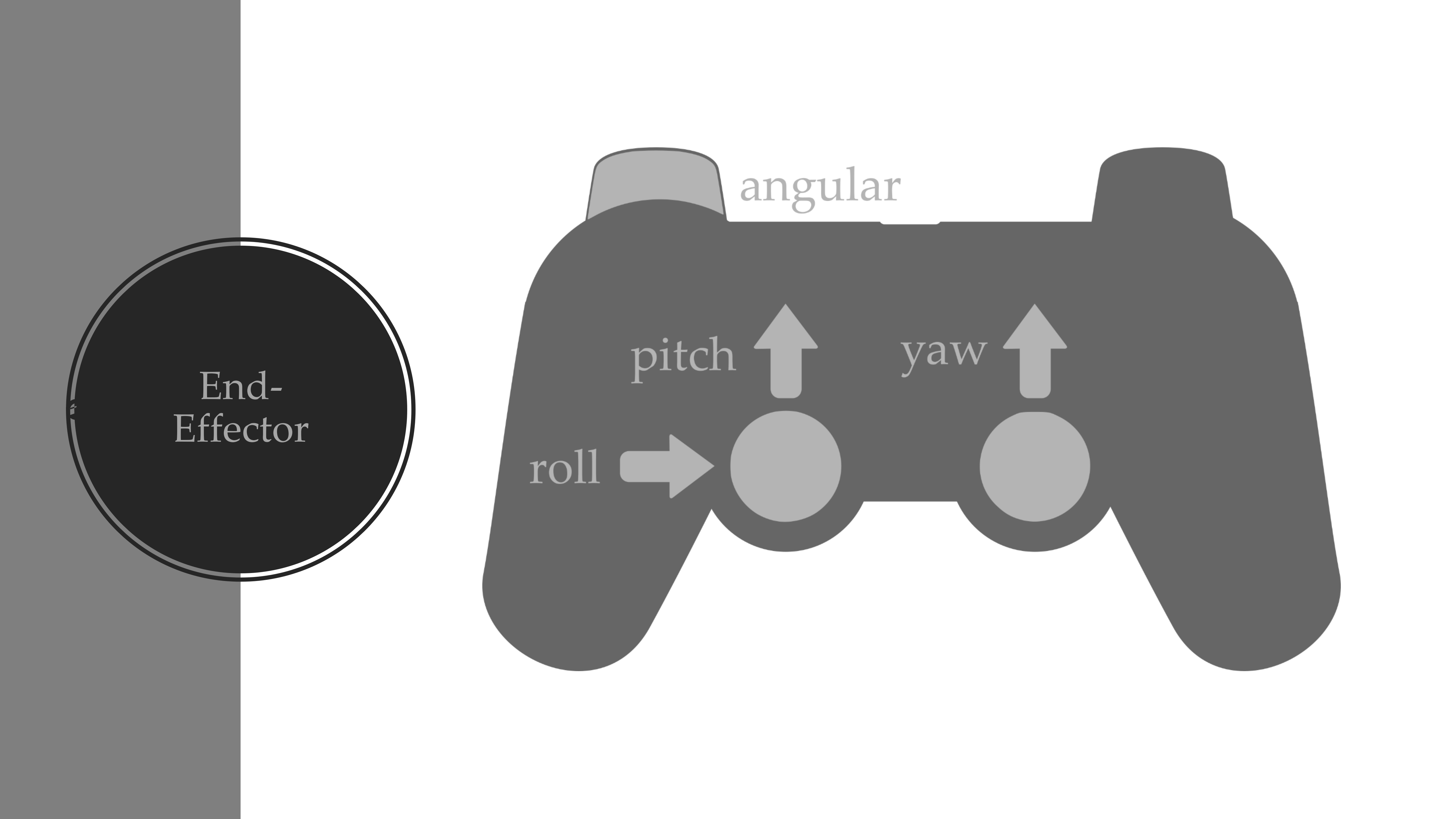
Online take trained decoder $\phi(s, z) \rightarrow a$

At each timestep t:

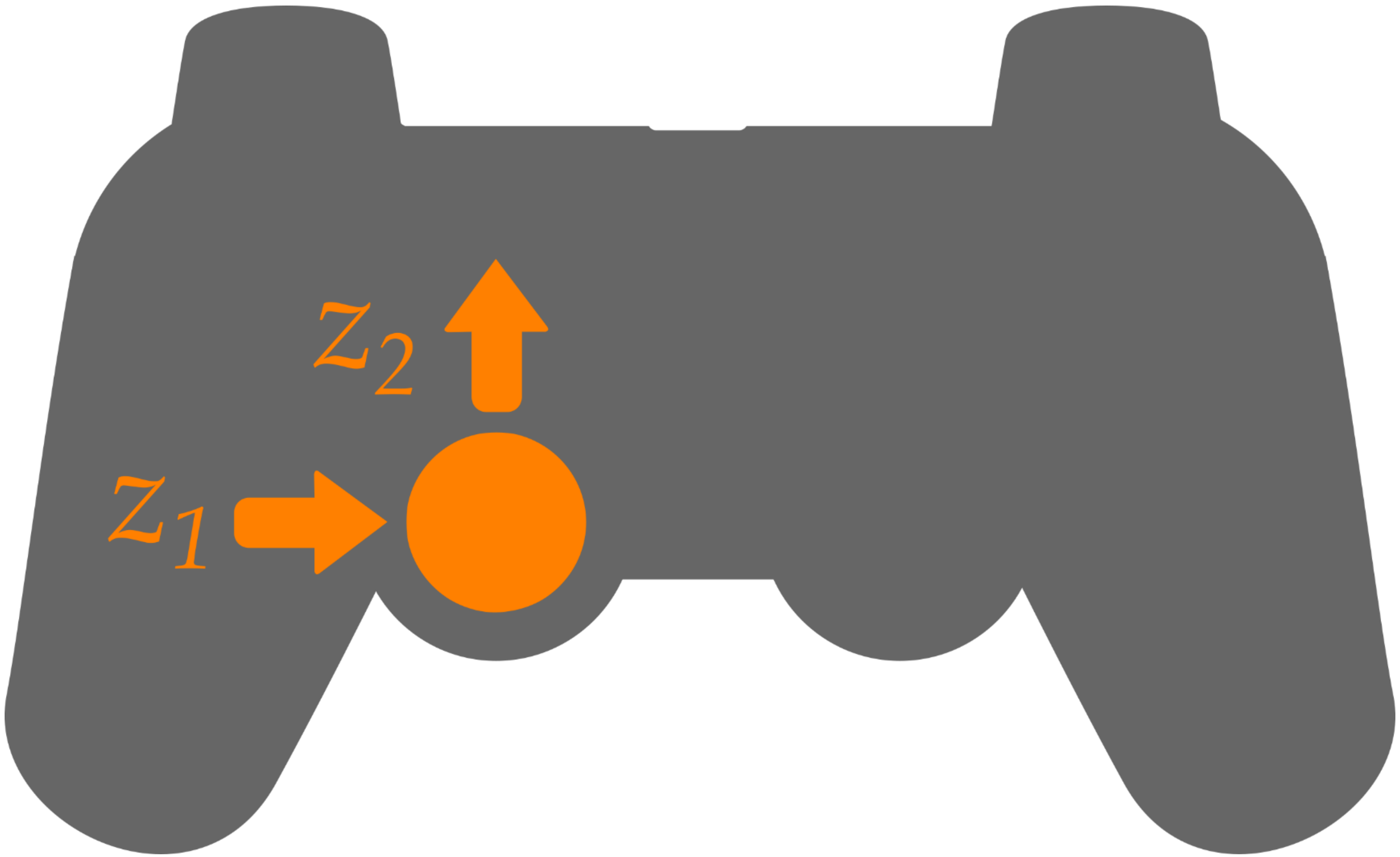
- Observe state s and human input u_H
- Treat z = u_H as the latent action
- Get decoded action $\phi(s, z) \rightarrow a_H$
- [*Optional*] Use predict and blend to assist human
- Transition to next state


End-
Effector


linear
y
z
x


End-
Effector
angular
pitch
yaw
roll

Latent Actions



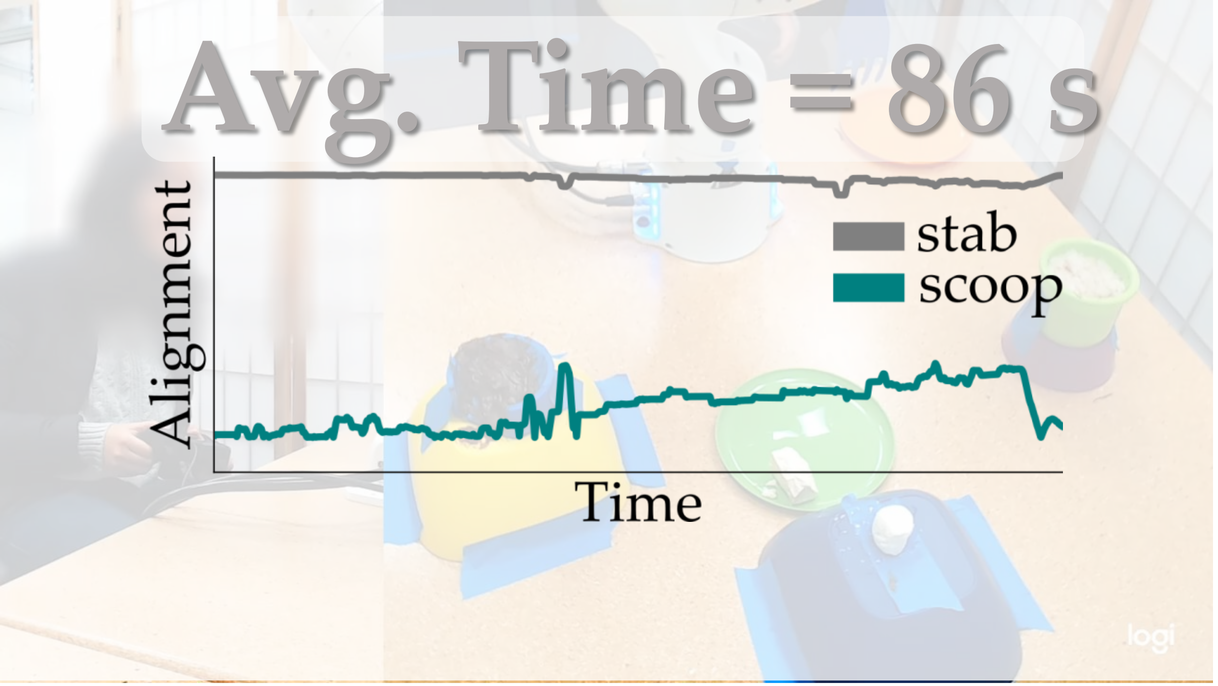
4x Speed

(1) add eggs

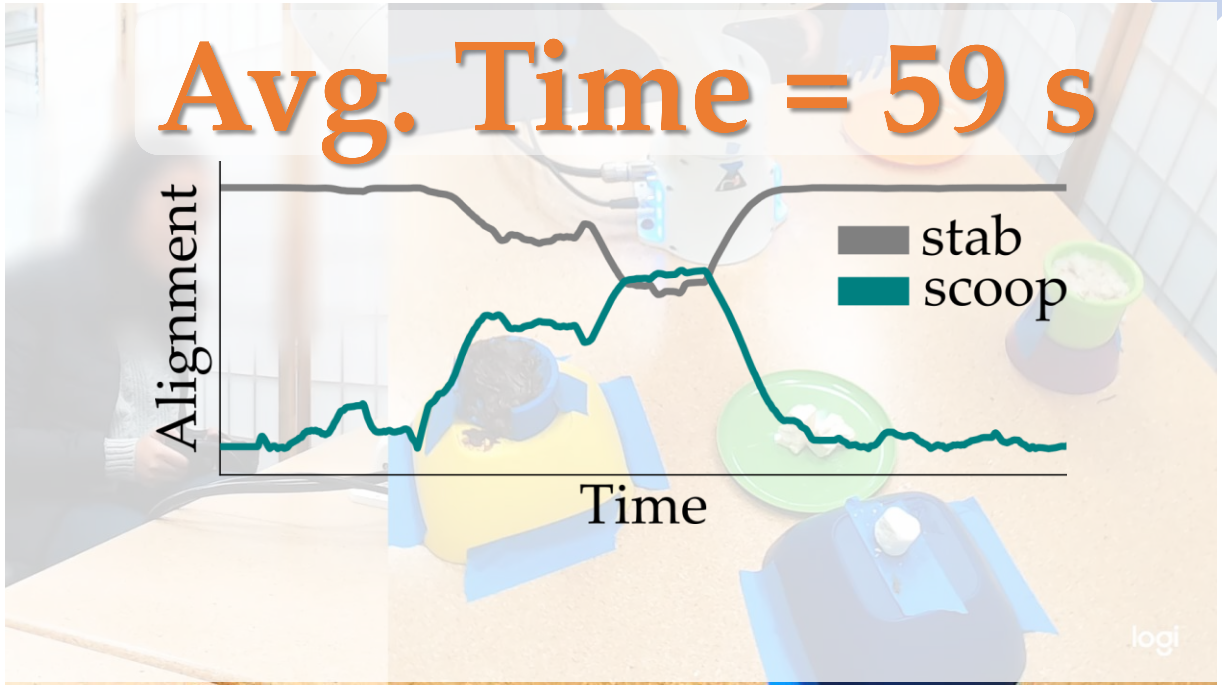
End-Effector

(1) add eggs

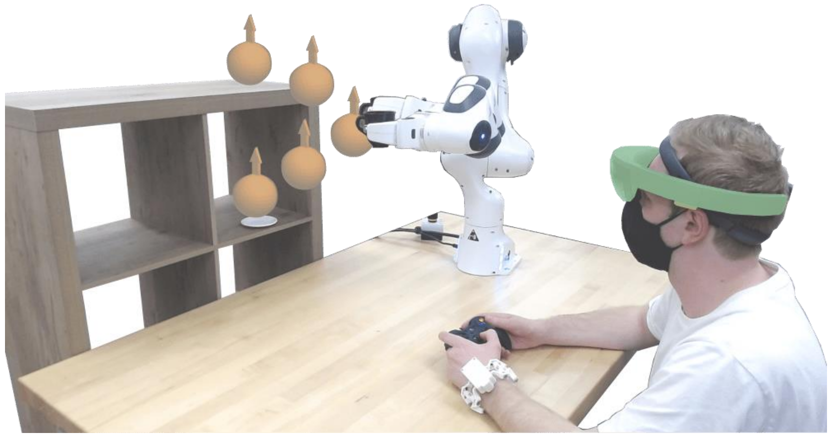
Latent Action

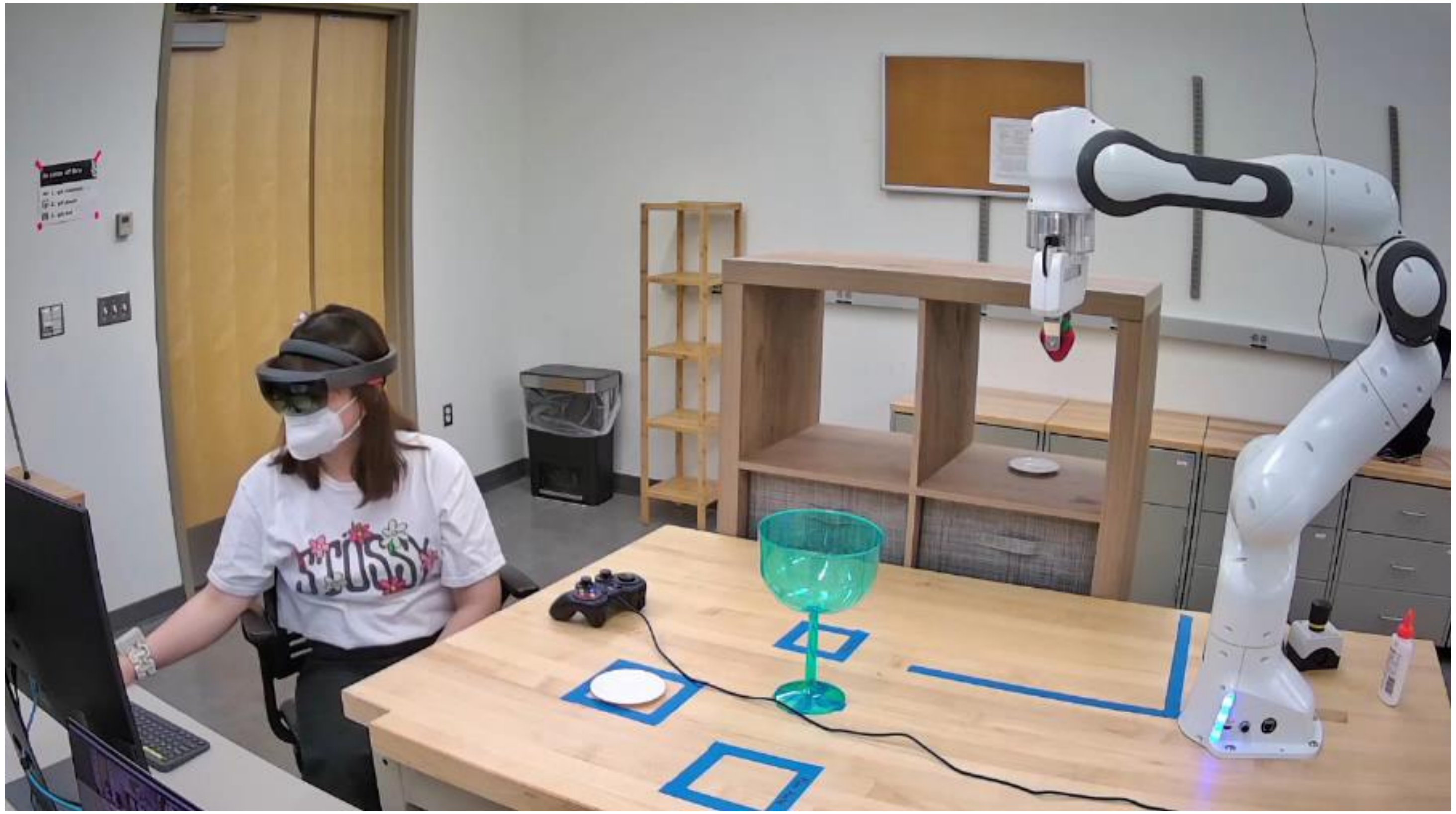

Avg. Time = 86 s
Alignment
stab
scoop
Time
logi

R


Avg. Time = 59 s
Alignment
stab
scoop
Time
logi

LA + SA (ours)





Related Papers

- https://link.springer.com/article/10.1007/s10514-021-10005-w
- https://dl.acm.org/doi/full/10.1145/3651994

This Lecture

- Introducing shared autonomy
- One flavor of solutions: predicting goals
- Another flavor of solutions: predicting actions




NSF

USDA
NIFA

UNIVERSITY OF
ILLINOIS
URBANA-CHAMPAIGN

Carnegie Mellon University

PURDUE
UNIVERSITY



collab

collab.me.vt.edu