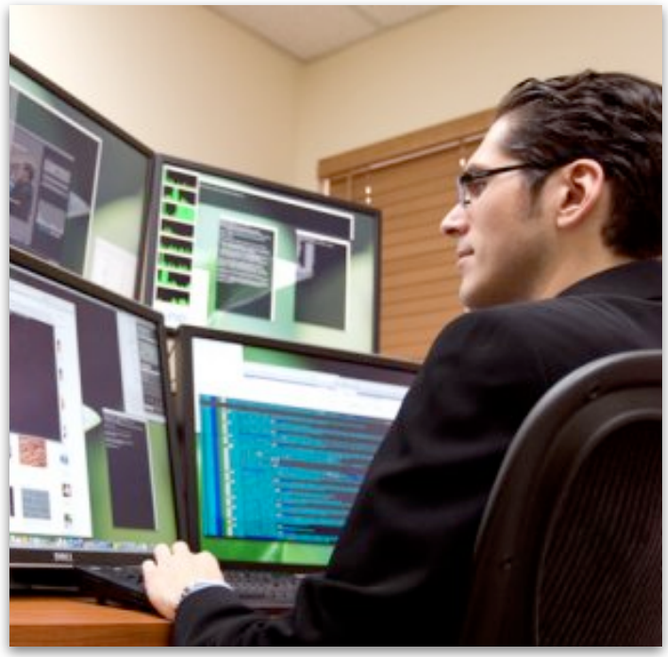
Introduction: Why learn from demonstration?



General purpose robot → Specific task → Expert engineer

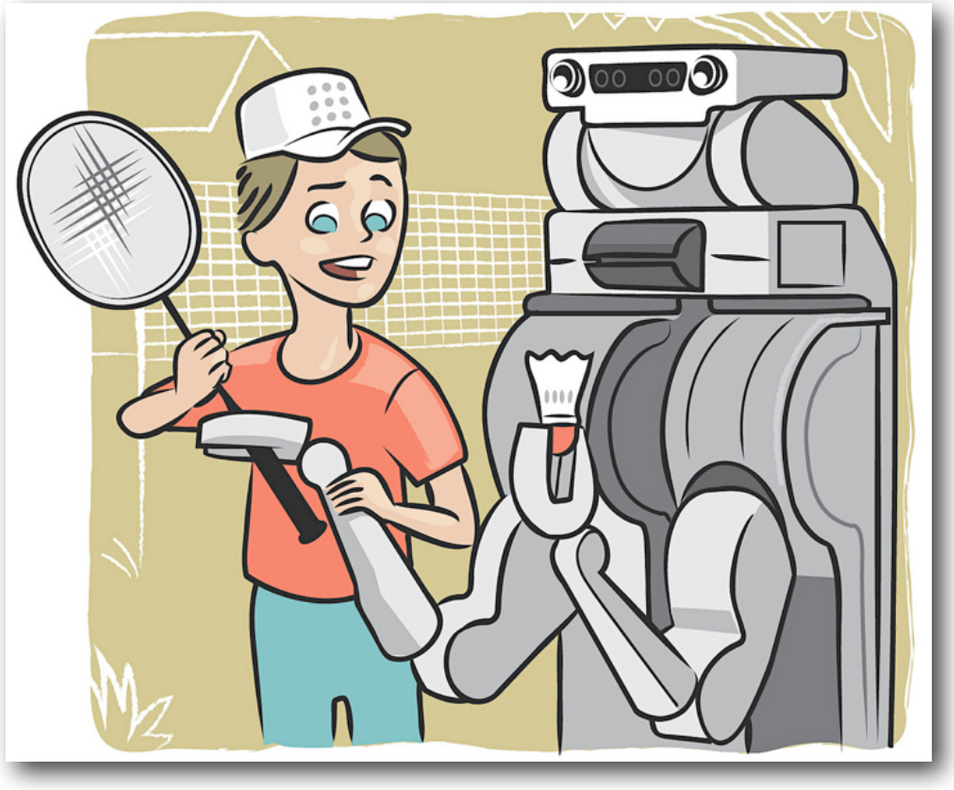
Introduction: Why learn from demonstration?

Programming robots is hard!

- Huge number of possible tasks
- Unique environmental demands
- Tasks difficult to describe formally
- Expert engineering impractical

Introduction: Why learn from demonstration?

- Natural, expressive way to program
- No expert knowledge required
- Valuable human intuition
- Program new tasks as-needed



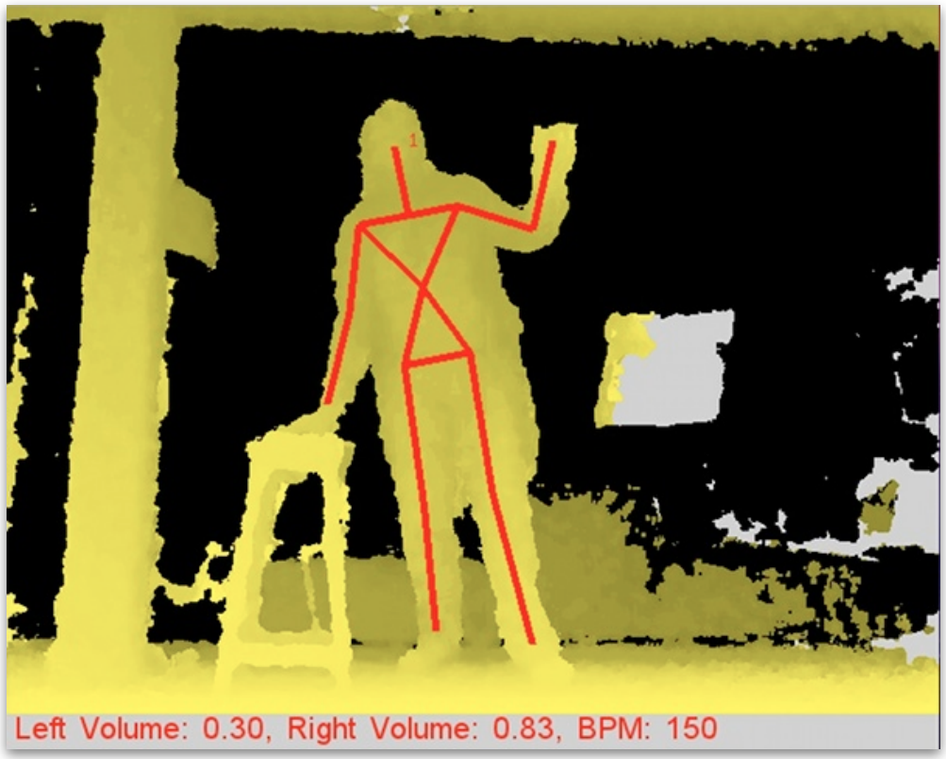
How can robots be shown how to perform tasks?

Introduction

Sensing

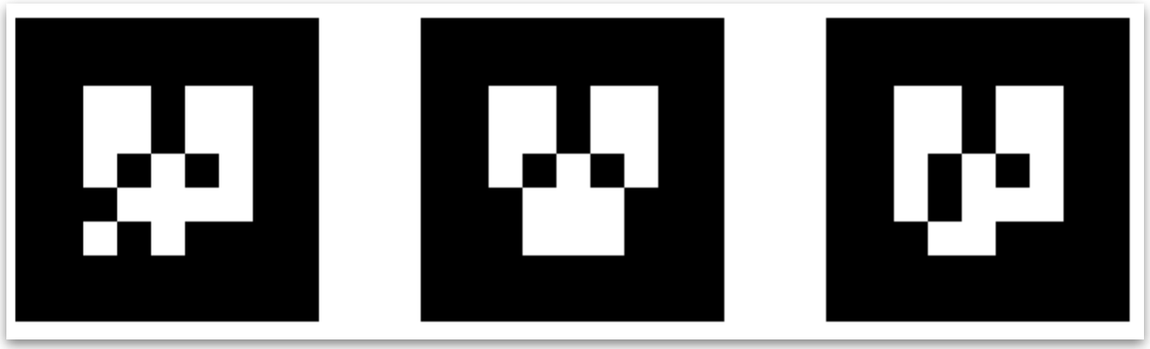
Modes of input

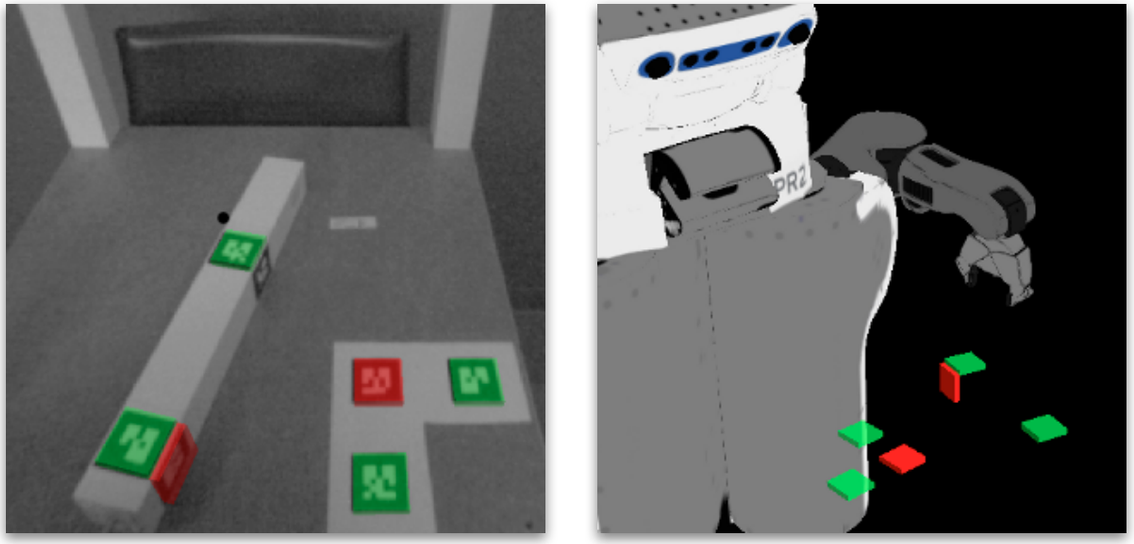
Sensing: RGB(D) cameras, depth sensors



- Standard RGB cameras
- Stereo: Bumblebee
- RGB-D: Microsoft Kinect
- Time of flight: Swiss Ranger
- LIDAR: SICK

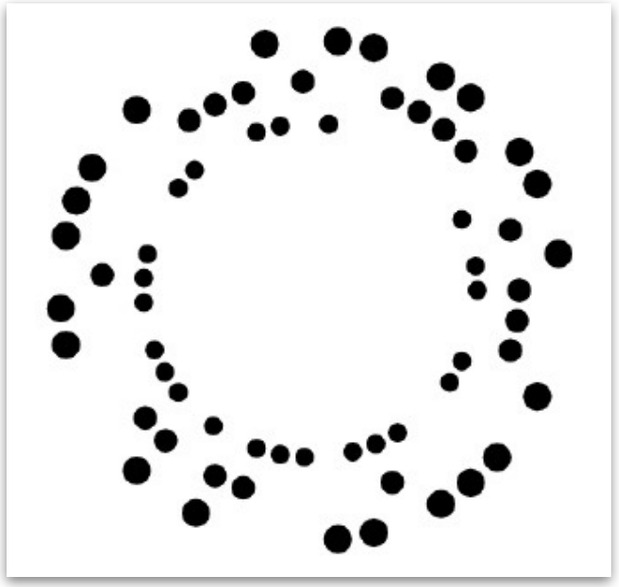
Sensing: Visual fiducials

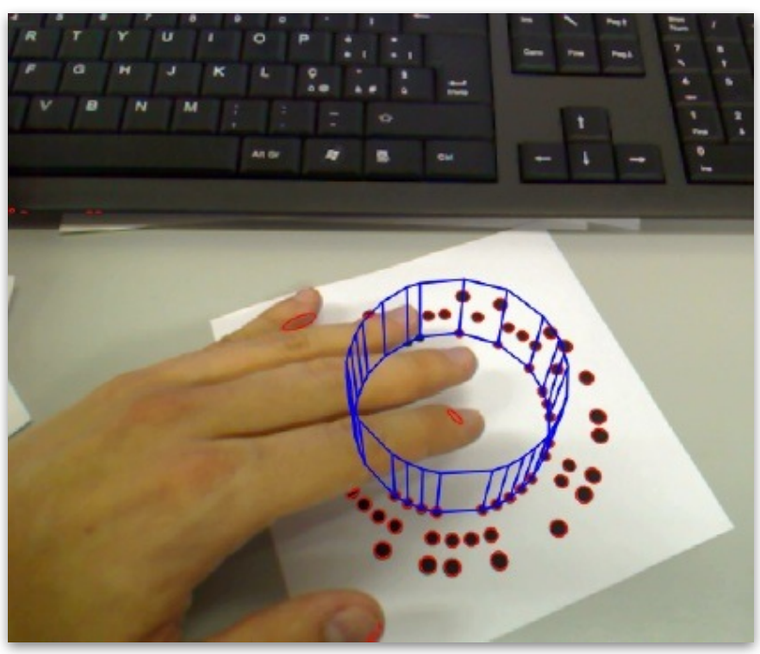




AR tags

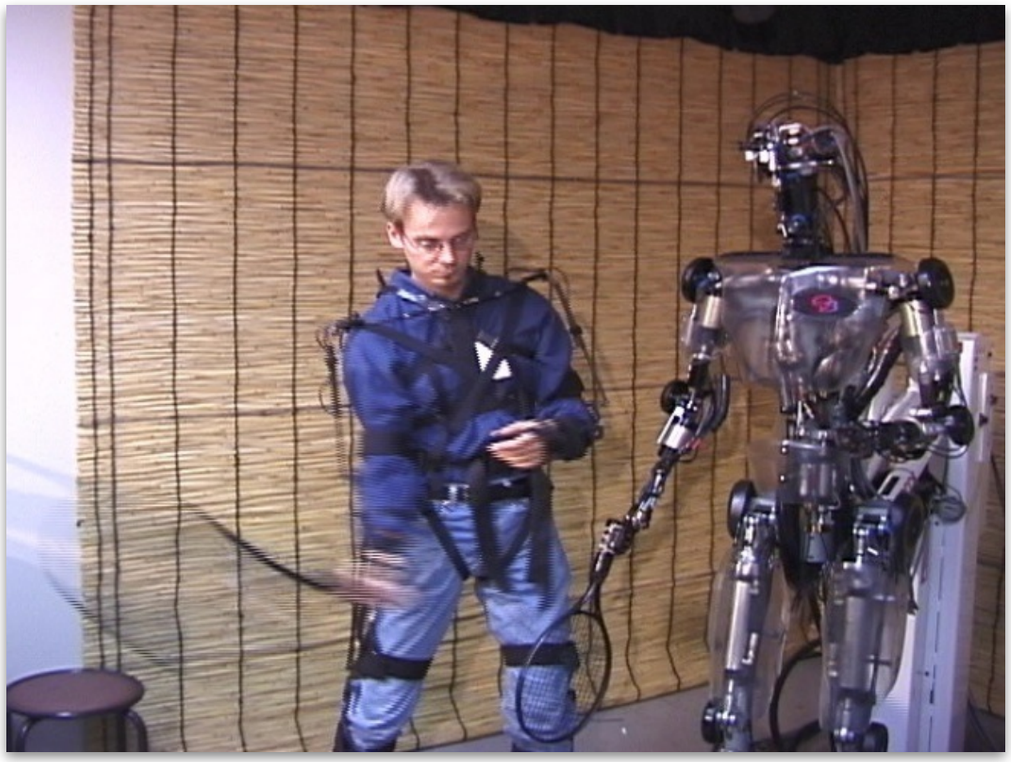
http://wiki.ros.org/ar_track_alvar





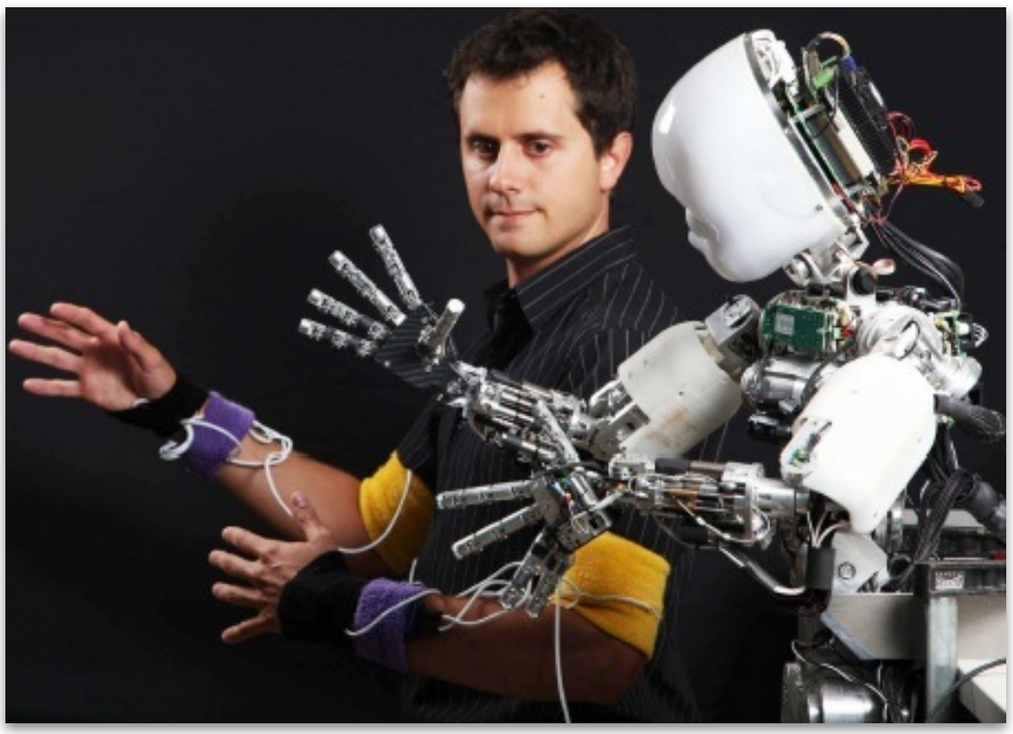
RUNE-129 tags

Sensing: Wearable sensors



SARCOS Sensuit:

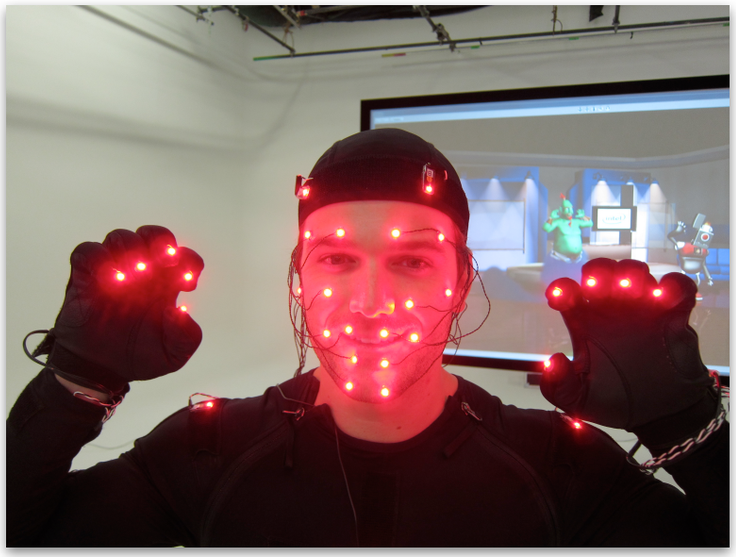
Record 35-DOF poses
at 100 Hz



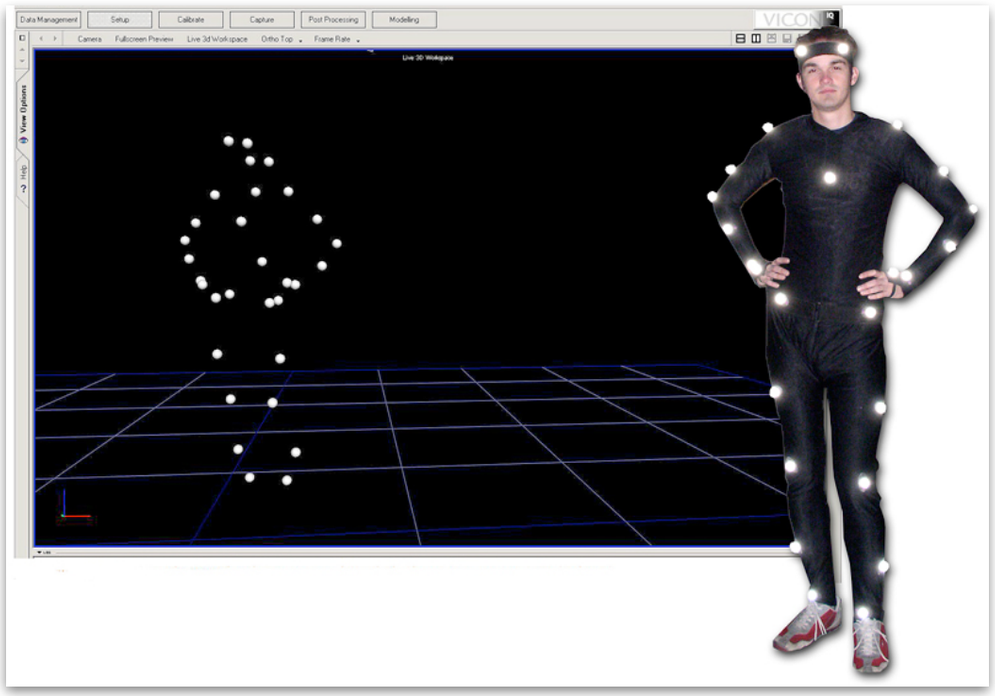
Other wearables:

- Accelerometers
- Pressure sensors
- First-person video

Sensing: Motion capture



Phasespace



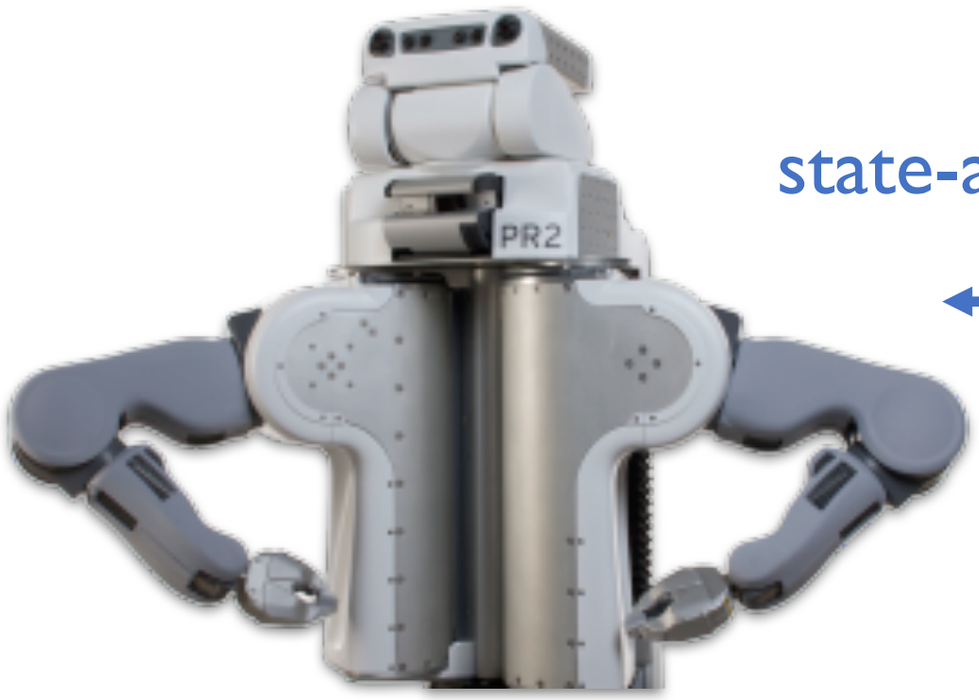
Vicon

Introduction

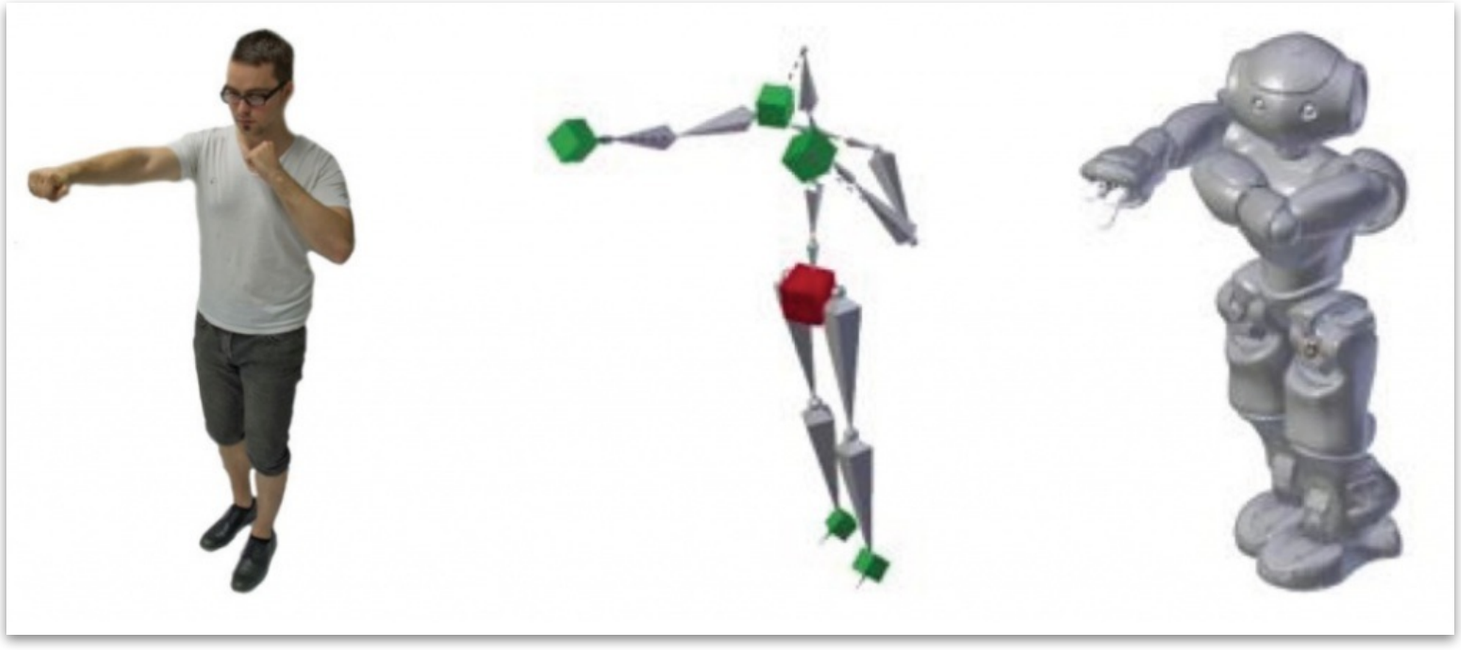
Sensing

Modes of input

The correspondence problem



state-action mapping?



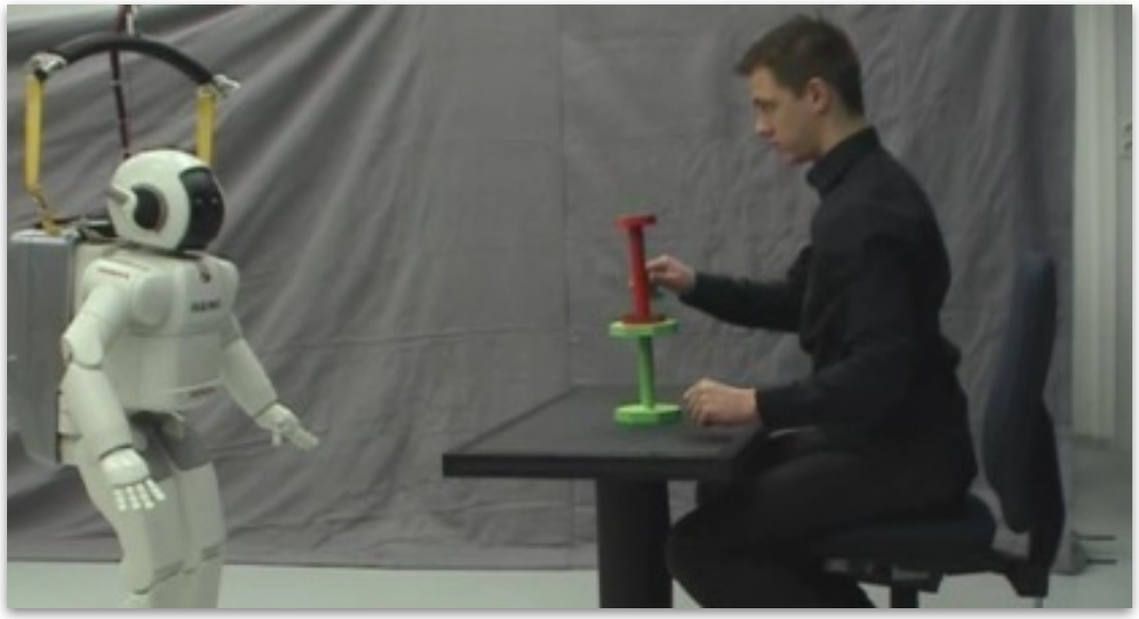
The correspondence problem

How to provide demonstrations?
Two primary modes of input:

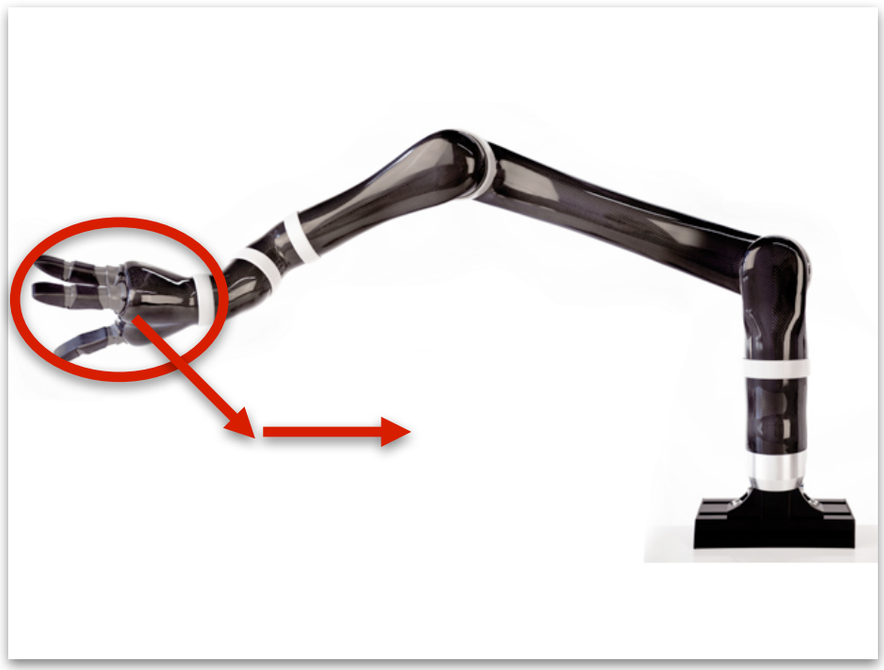
Learning by watching (imitation): Define a correspondence

Learning by doing (demonstration): Avoid correspondence entirely

Learning by watching: Simplified mimicry

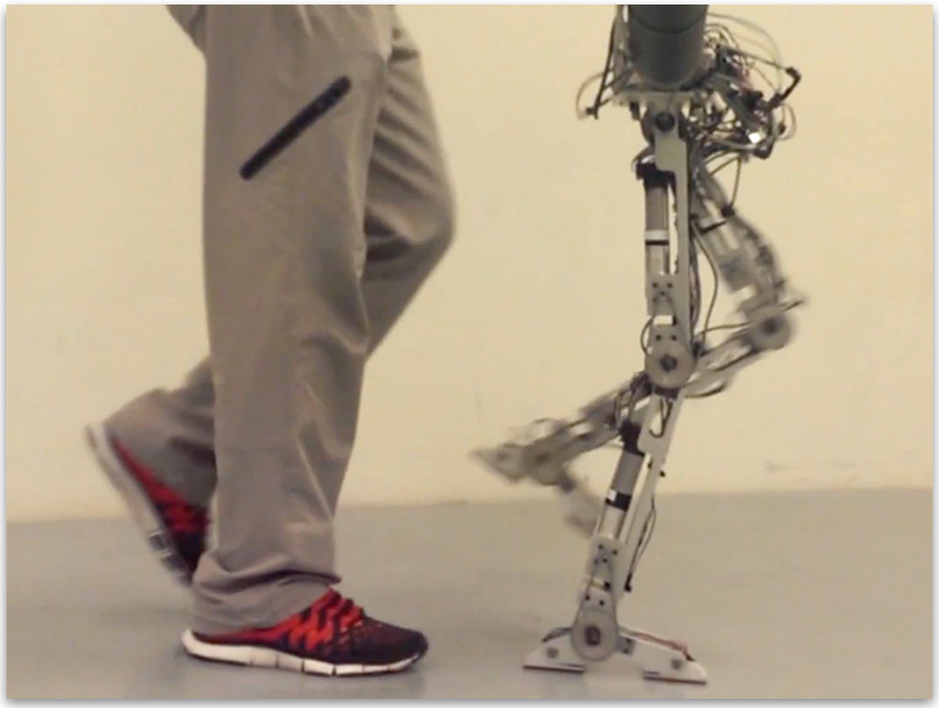


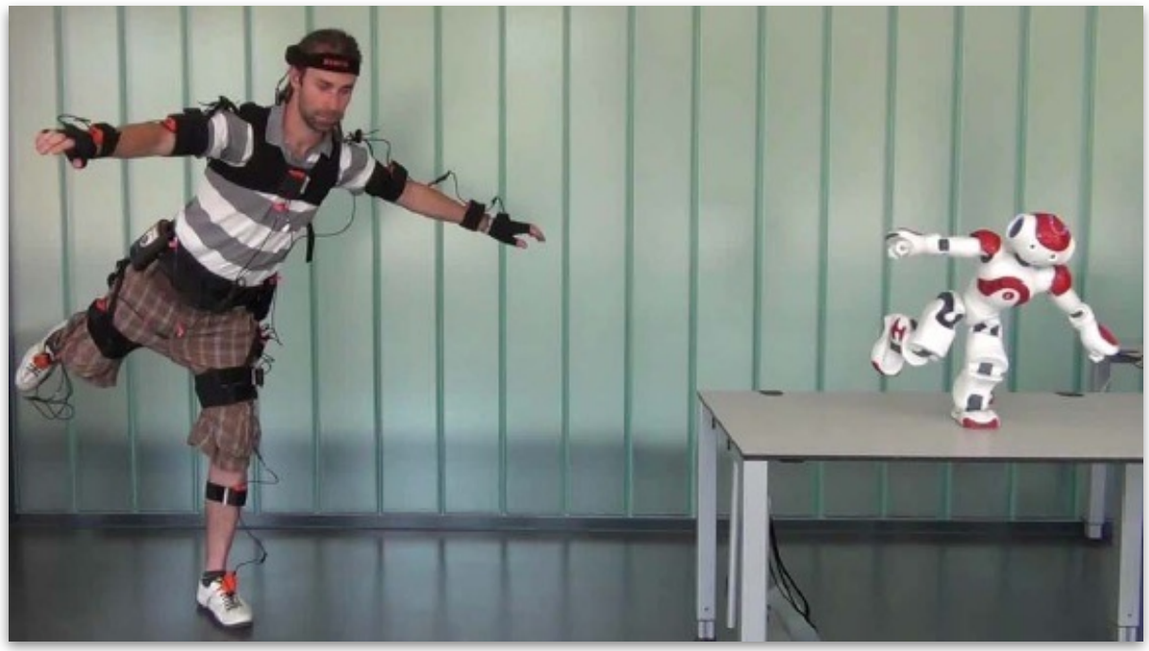
Object-based

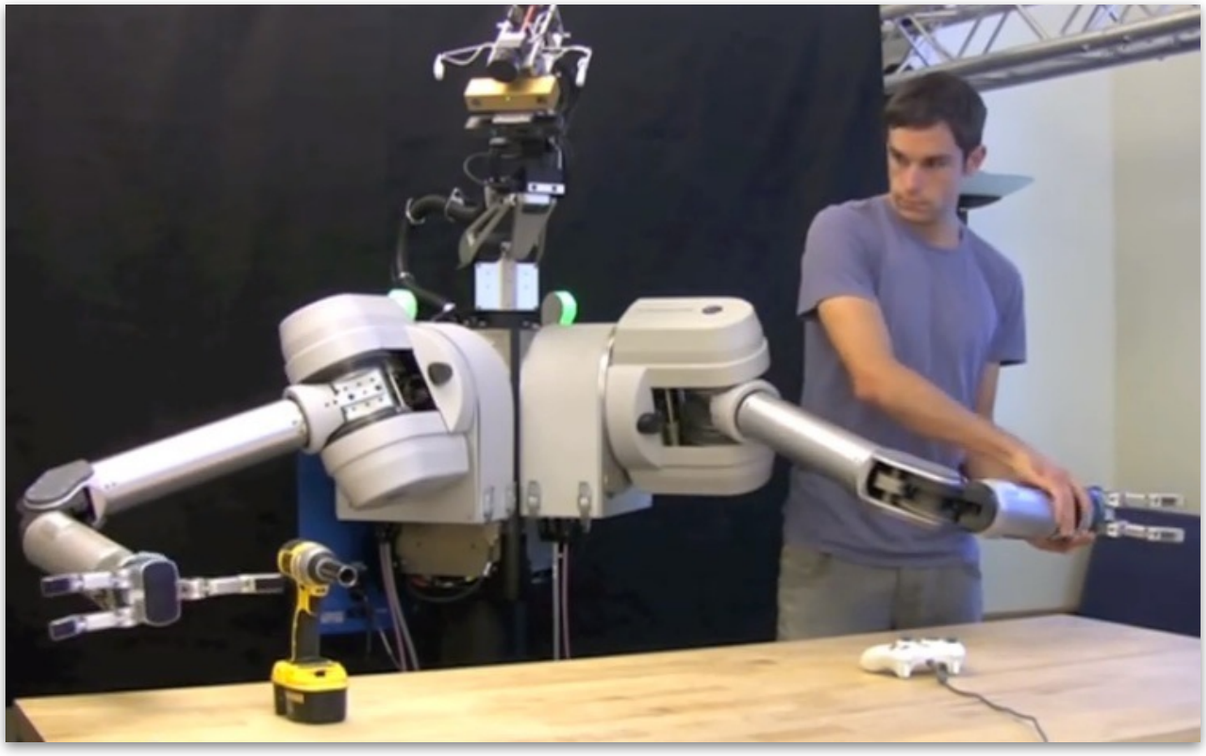


End effector-based

Learning by watching: Shadowing



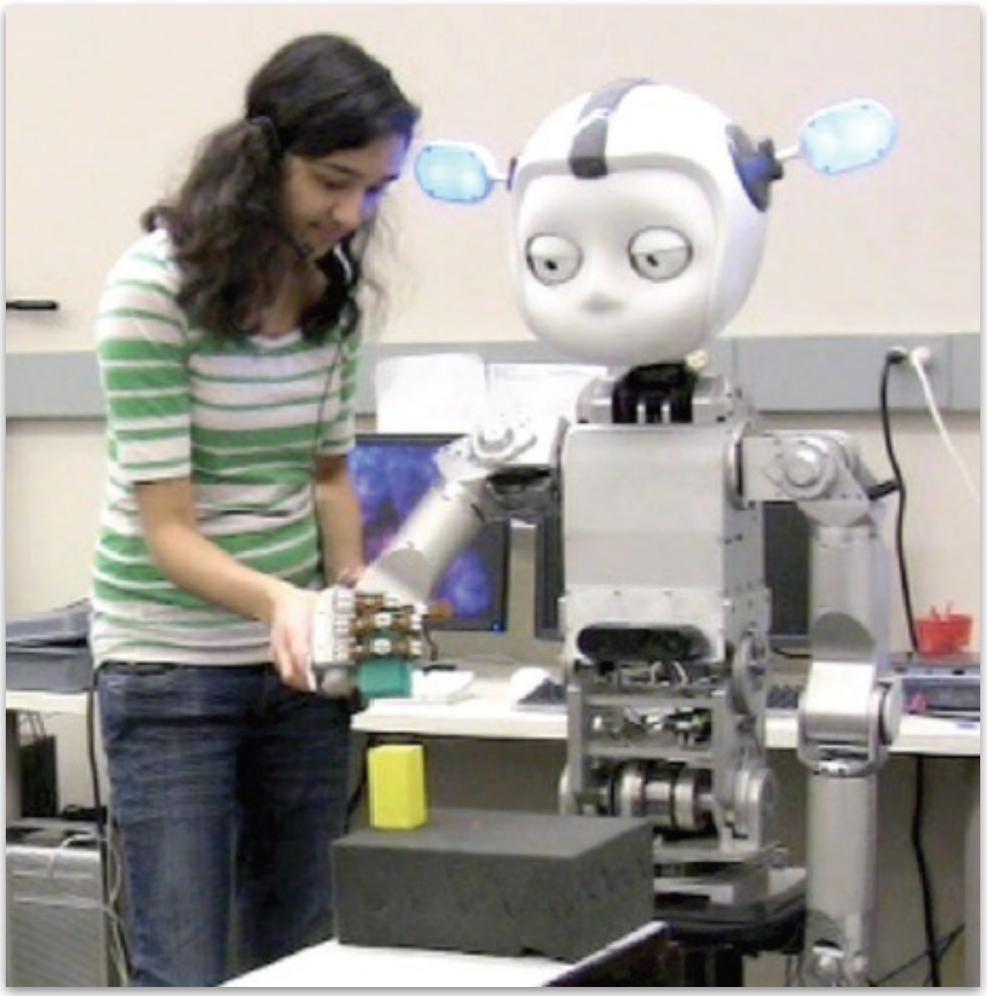


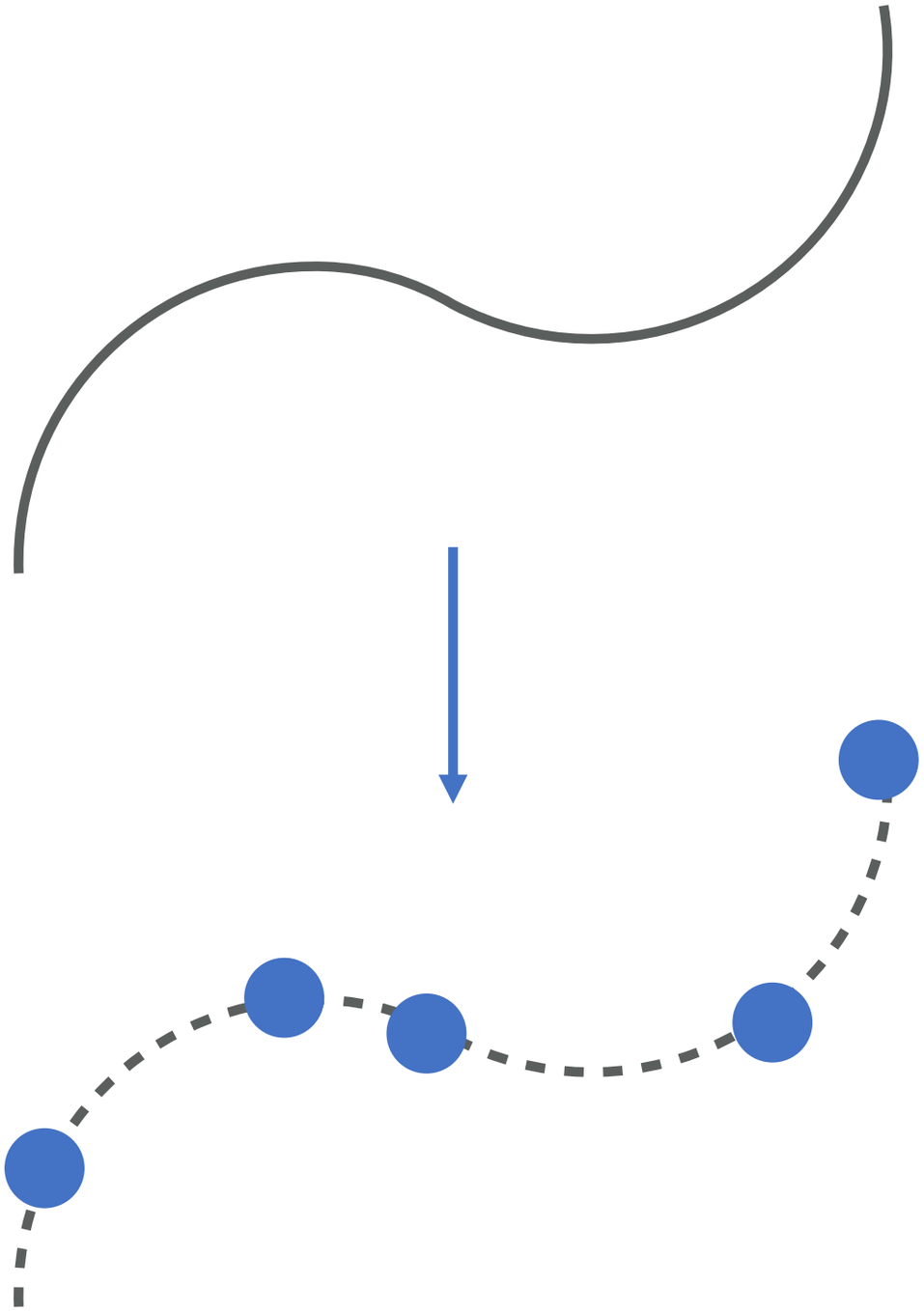


Learning by doing: Teleoperation

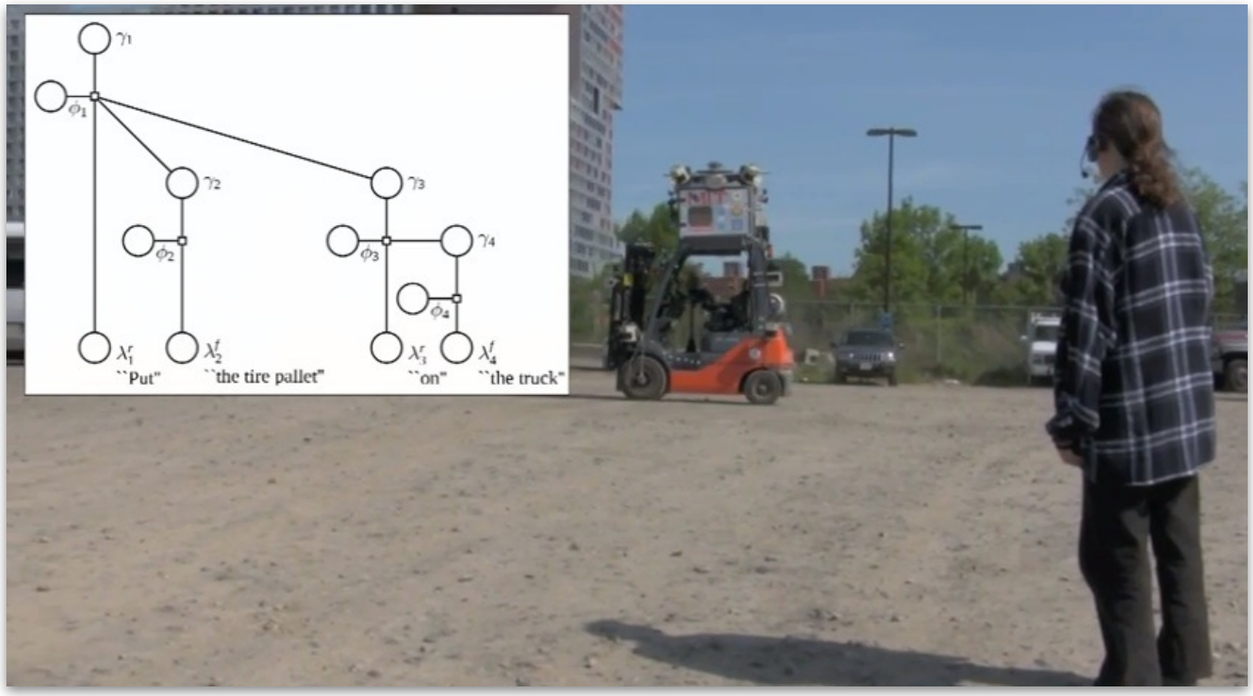
Learning by doing: Kinesthetic demonstration

Learning by doing: Keyframe demonstration

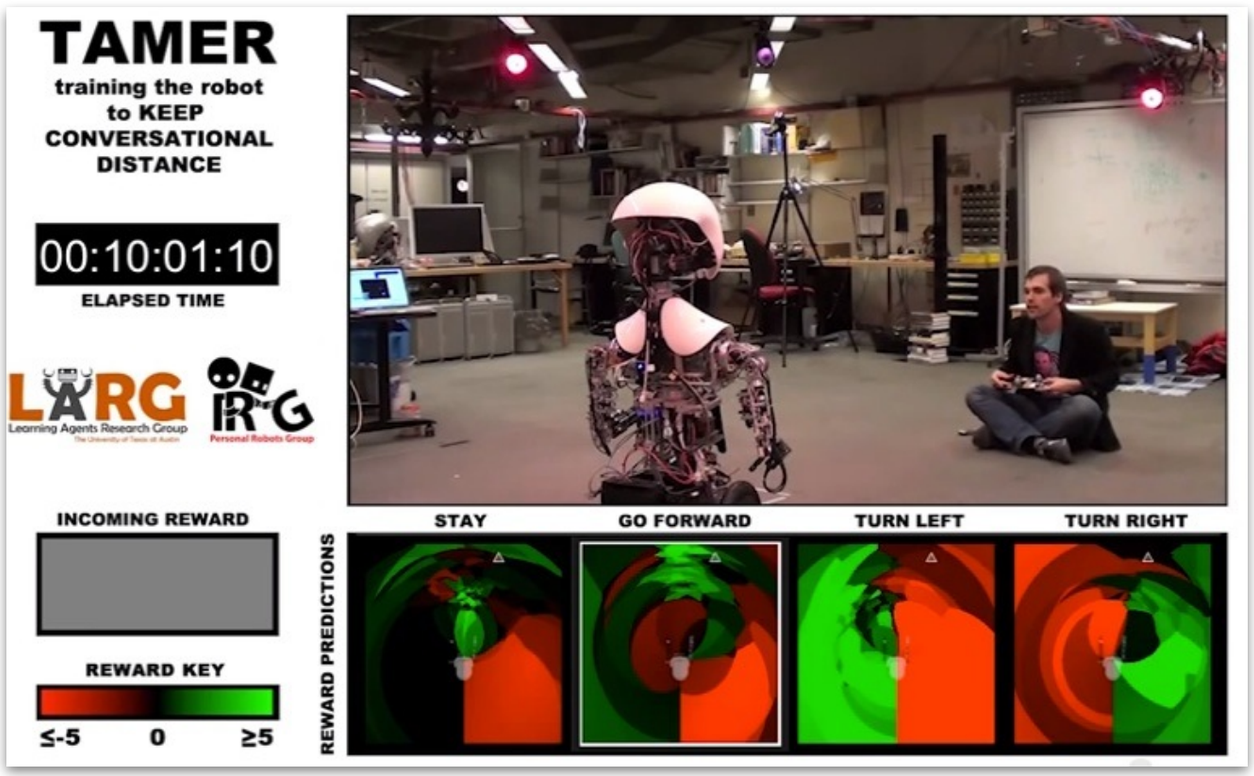




Supplementary information: Speech and critique



Interpreting and grounding natural language commands



Realtime user feedback given to RL system

Learning task features

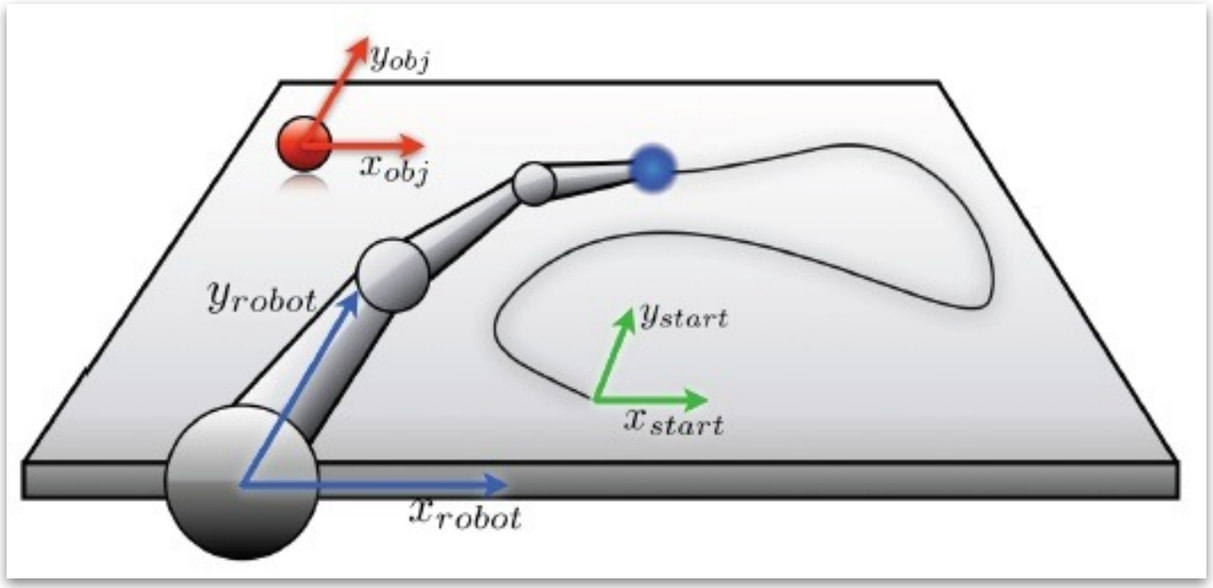
Learning a task plan

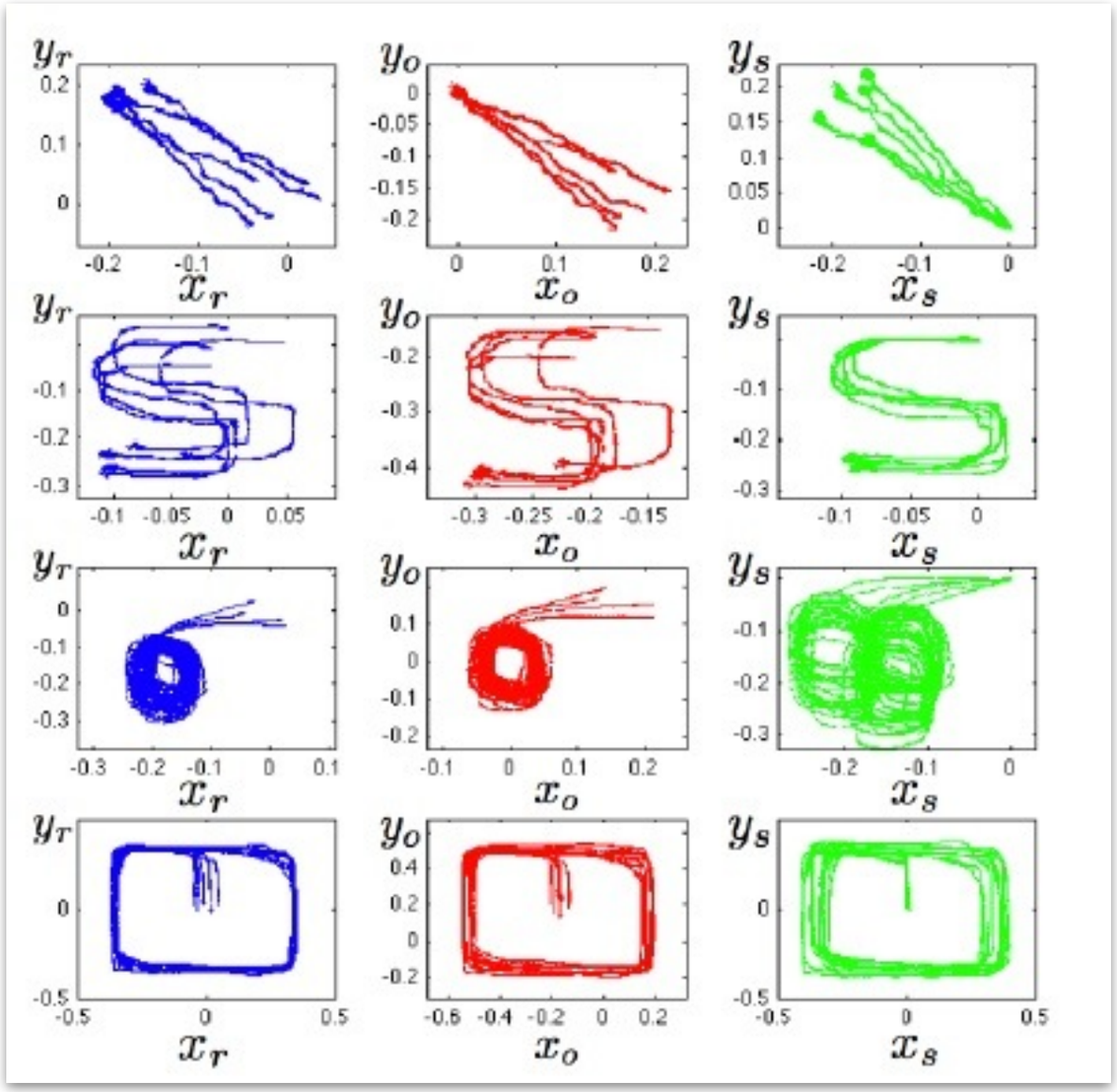
Learning task objectives

Learning object affordances

Learning task features: Reference frame inference

Controllers generalize better when in correct reference frame





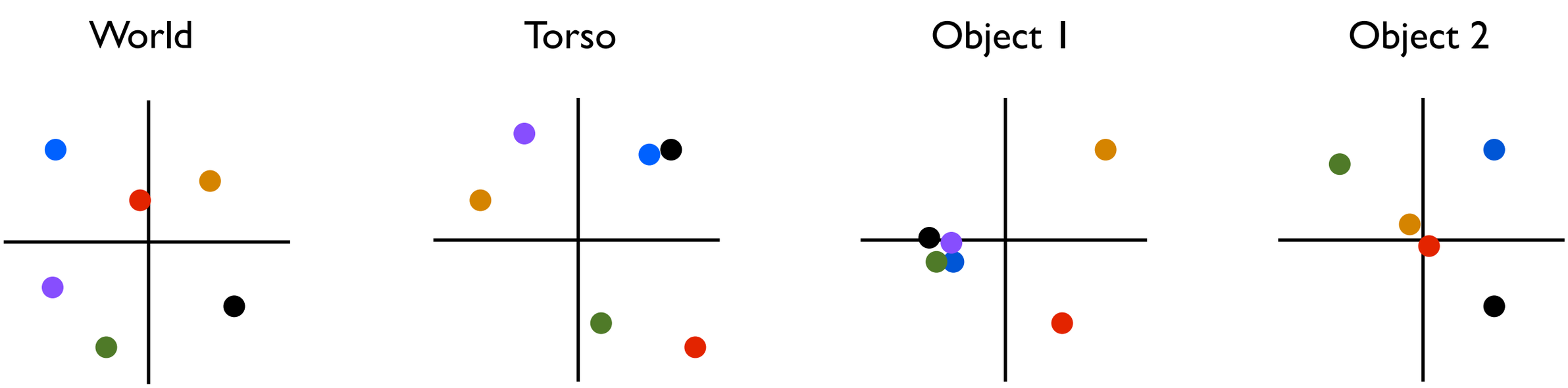
1. Weight each reference frame by total distance error of trajectories in frame

2. Generate velocity profile by GMR with weighted reference frames

[Cederborg et al. 2010]

Learning task features: Reference frame inference

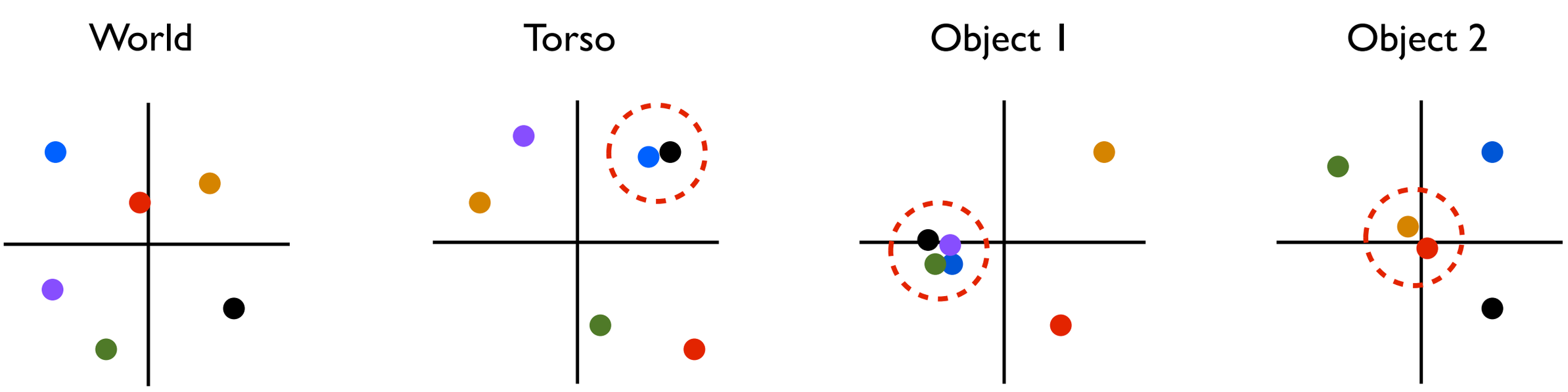
Graph endpoint of each trajectory w.r.t. each coordinate frame:



[Niekum et al. 2012]

Learning task features: Reference frame inference

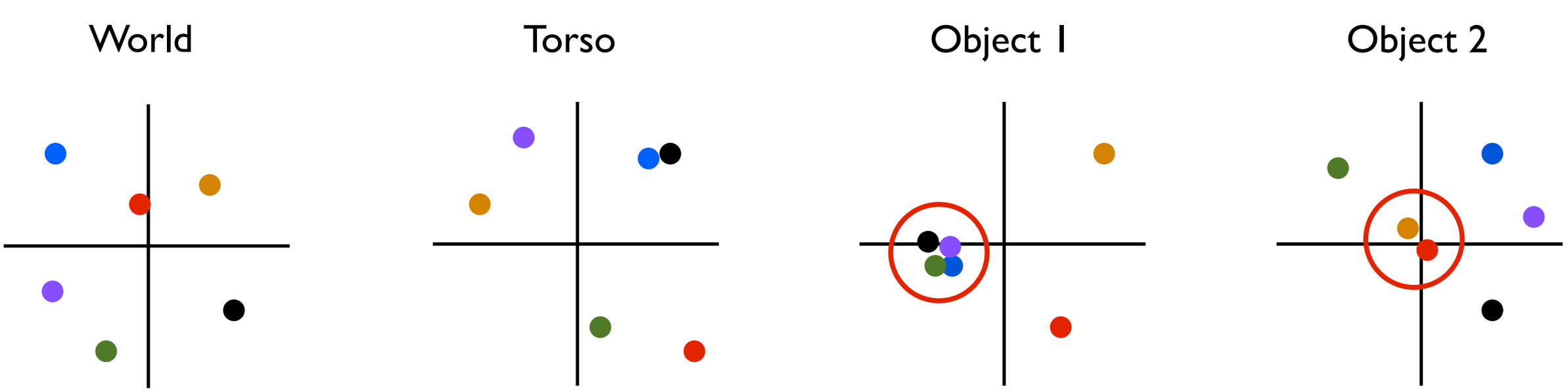
Identify possible clusters:



[Niekum et al. 2012]

Learning task features: Reference frame inference

Choose best point-wise cluster assignments:



[Niekum et al. 2012]

Learning task features: Abstraction from demonstration

Can we do better than original demonstrations?
Use RL and learn in abstracted lower-dimensional feature space.

1. Create abstraction by selecting features that are good predictors of demonstrated actions.

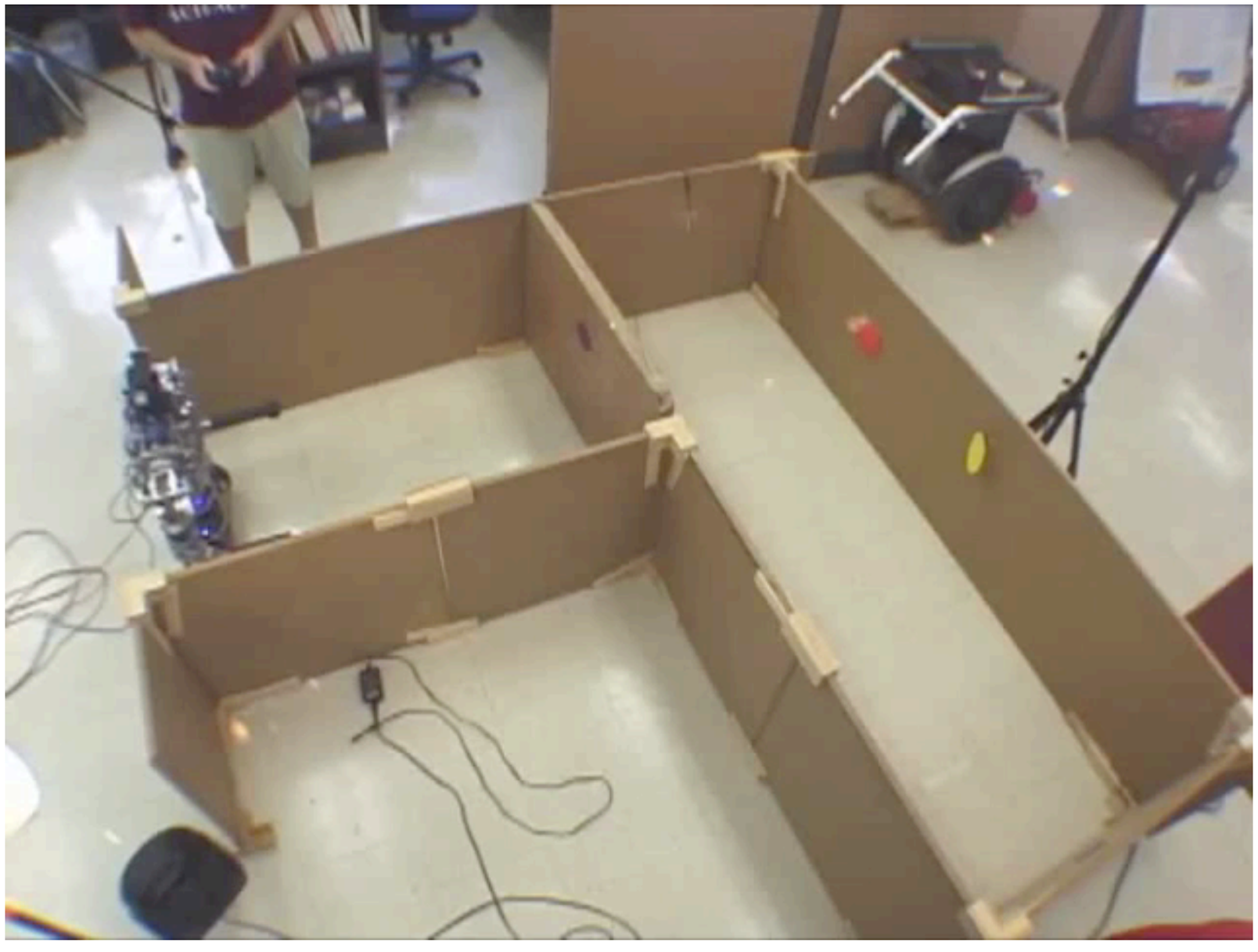
2. Use reinforcement learning in abstracted feature space to learn improved policy.

3. Iteratively remove features that minimally affect return.

[Cobo et al. 2009]

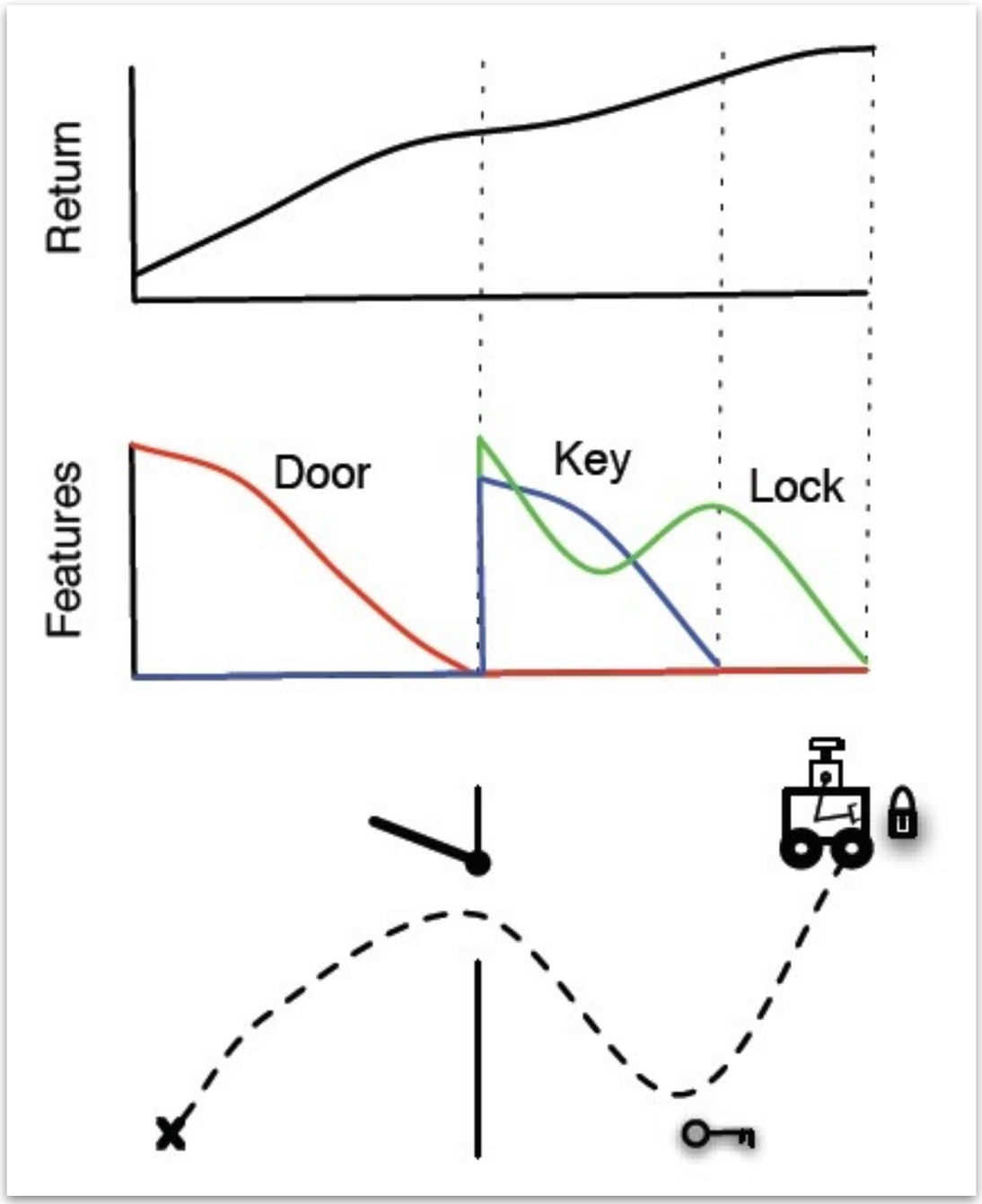
Learning task features: Abstraction segmentation

Some tasks are comprised of skills that each have their own abstraction



[Konidaris et al. 2012]

Learning task features: Abstraction segmentation

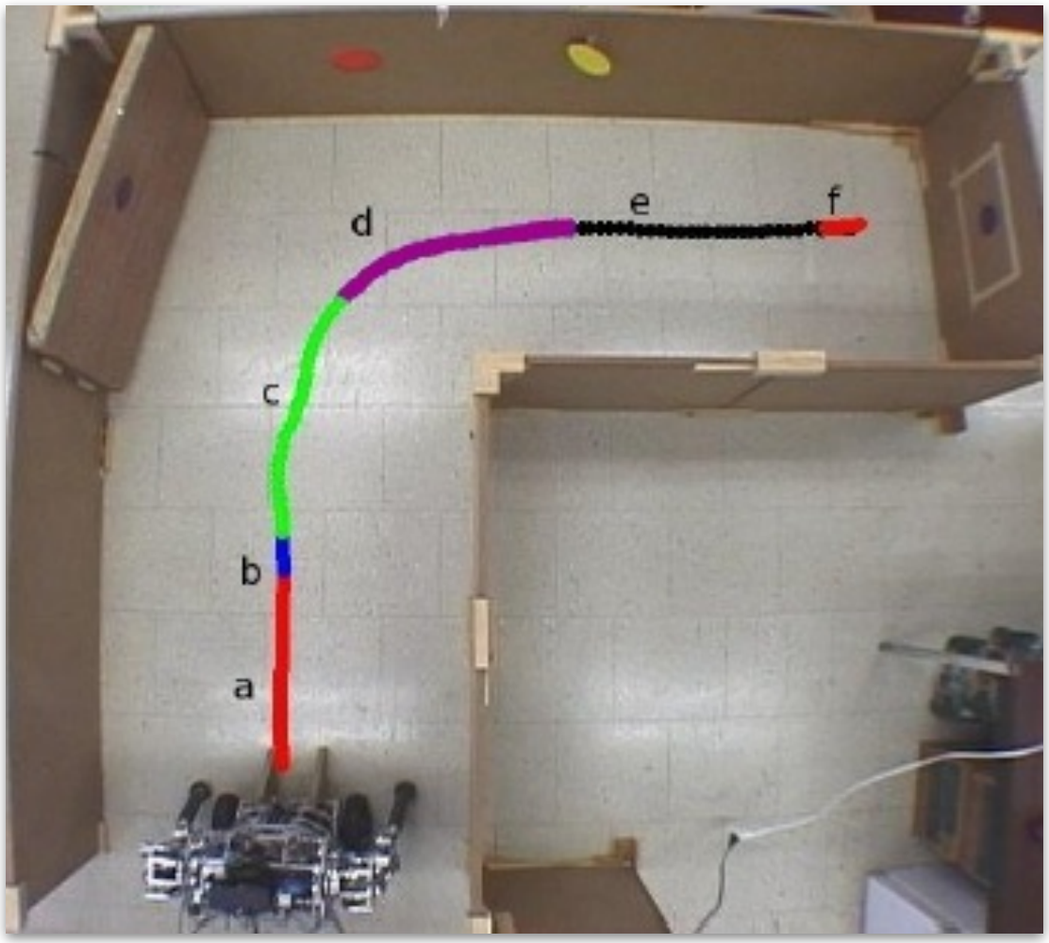


Identify changes in the abstraction that best explains the robot's observed returns.

Use this info to segment demonstrations into skills

[Konidaris et al. 2012]

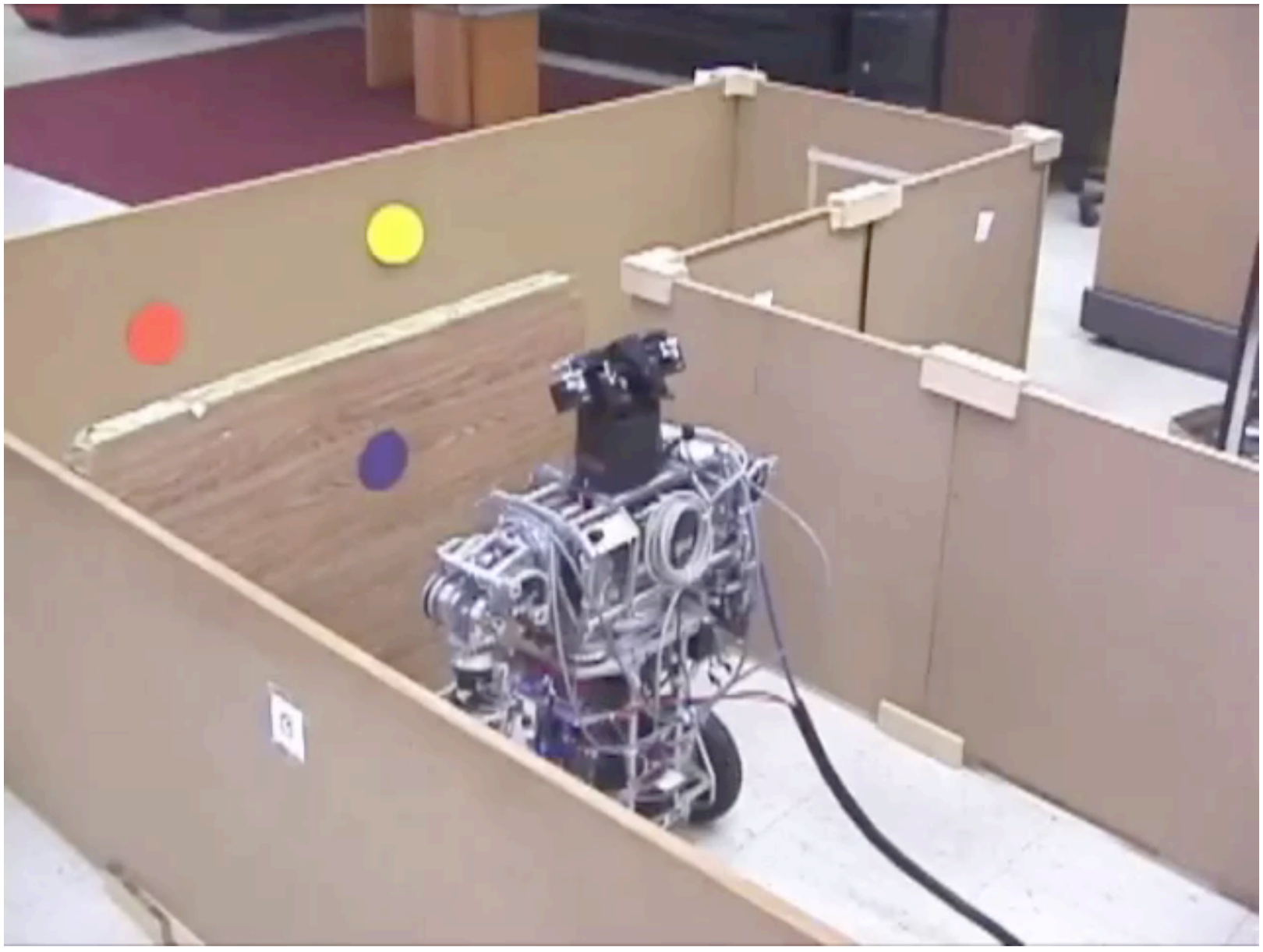
Learning task features: Abstraction segmentation



#	Abstraction	Description	Trajectories Required
a	torso-purple	Drive to door.	2
b	hand-purple	Push the door open.	1
c	torso-orange	Drive toward wall.	1
d	torso-yellow	Turn toward the end wall.	2
e	torso-purple	Drive to the panel.	1
f	hand-purple	Press the panel.	3

Trajectory segmented into skills and abstractions

[Konidaris et al. 2012]



Learning task features

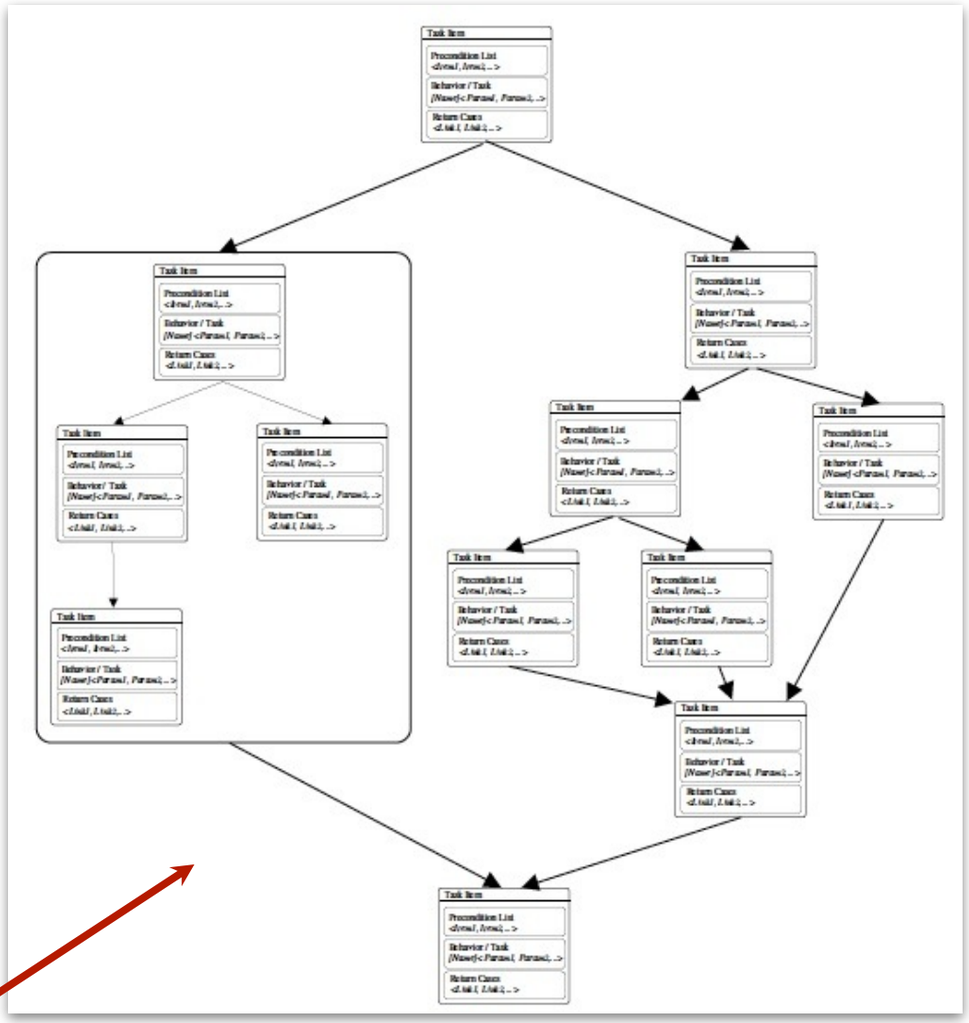
Learning a task plan

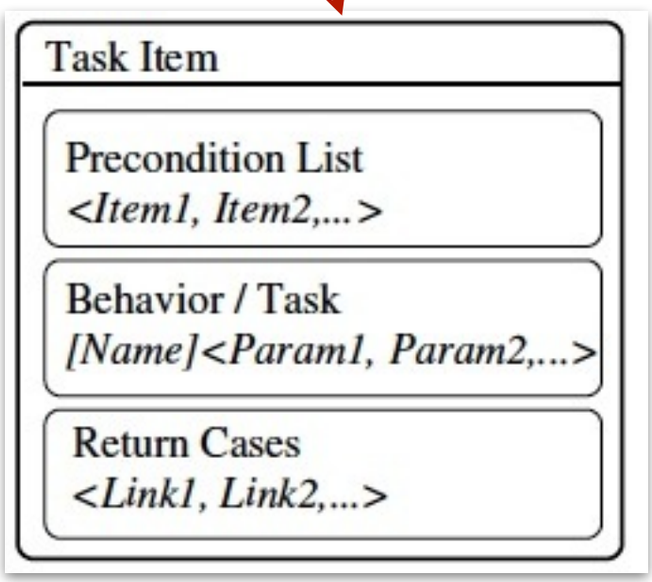
Learning task objectives

Learning object affordances

Learning a task plan: STRIPS-style plans

```
>> When I say deliver message
>> If Person1 is present
>>   Give message to Person1
>> Otherwise
>>   If Person2 is present
>>     Give message to Person2
>>   Otherwise
>>     Report message delivery failure
>>   Before
>> Before
>> Goto home
```



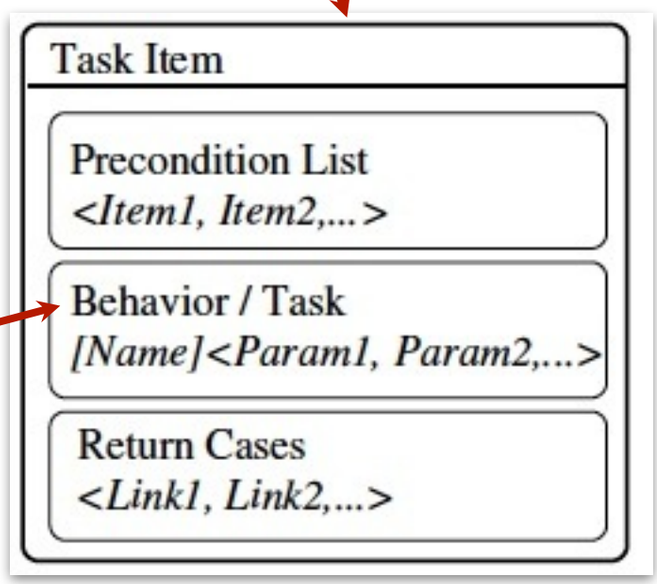


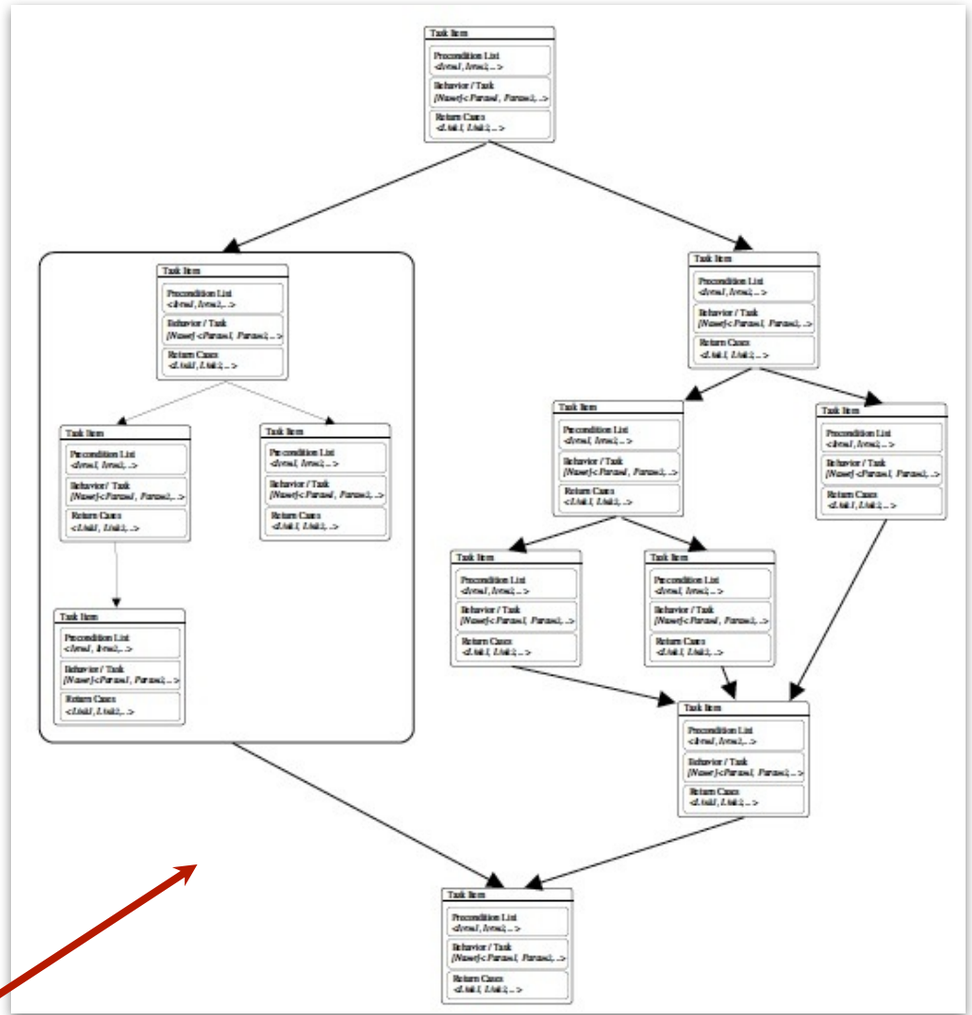
[Rybski et al. 2007]

Learning a task plan: STRIPS-style plans

```
>> When I say deliver message
>> If Person1 is present
>>   Give message to Person1
>> Otherwise
>>   If Person2 is present
>>     Give message to Person2
>>   Otherwise
>>     Report message delivery failure
>>   Before
>> Before
>> Goto home
```

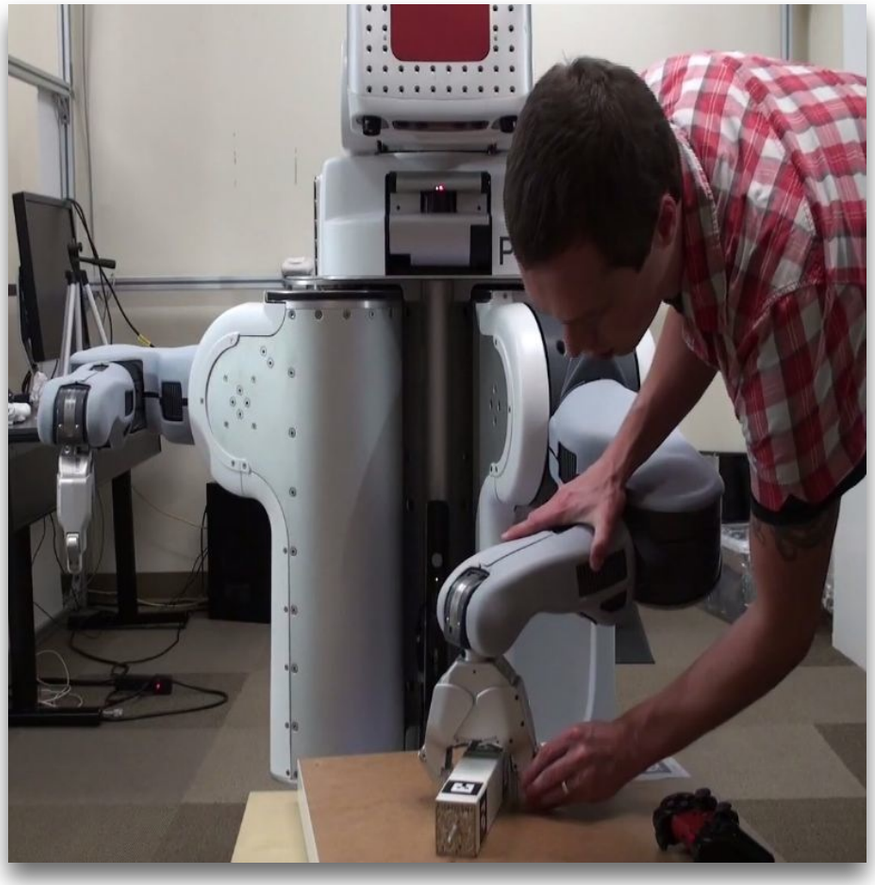
Demonstrated behavior





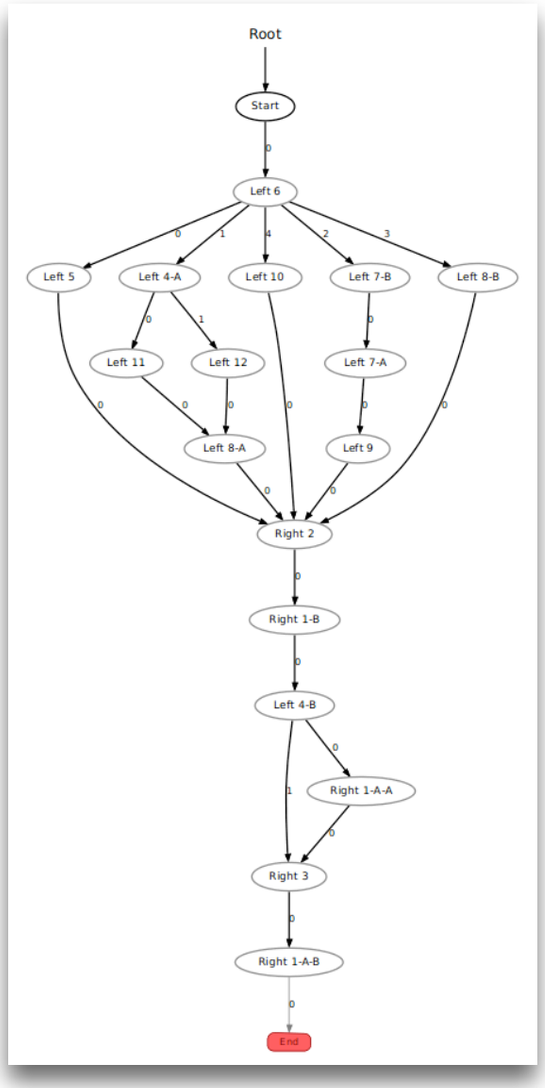
[Rybski et al. 2007]

Learning a task plan: Finite state automata



Unsegmented demonstrations of multi-step tasks

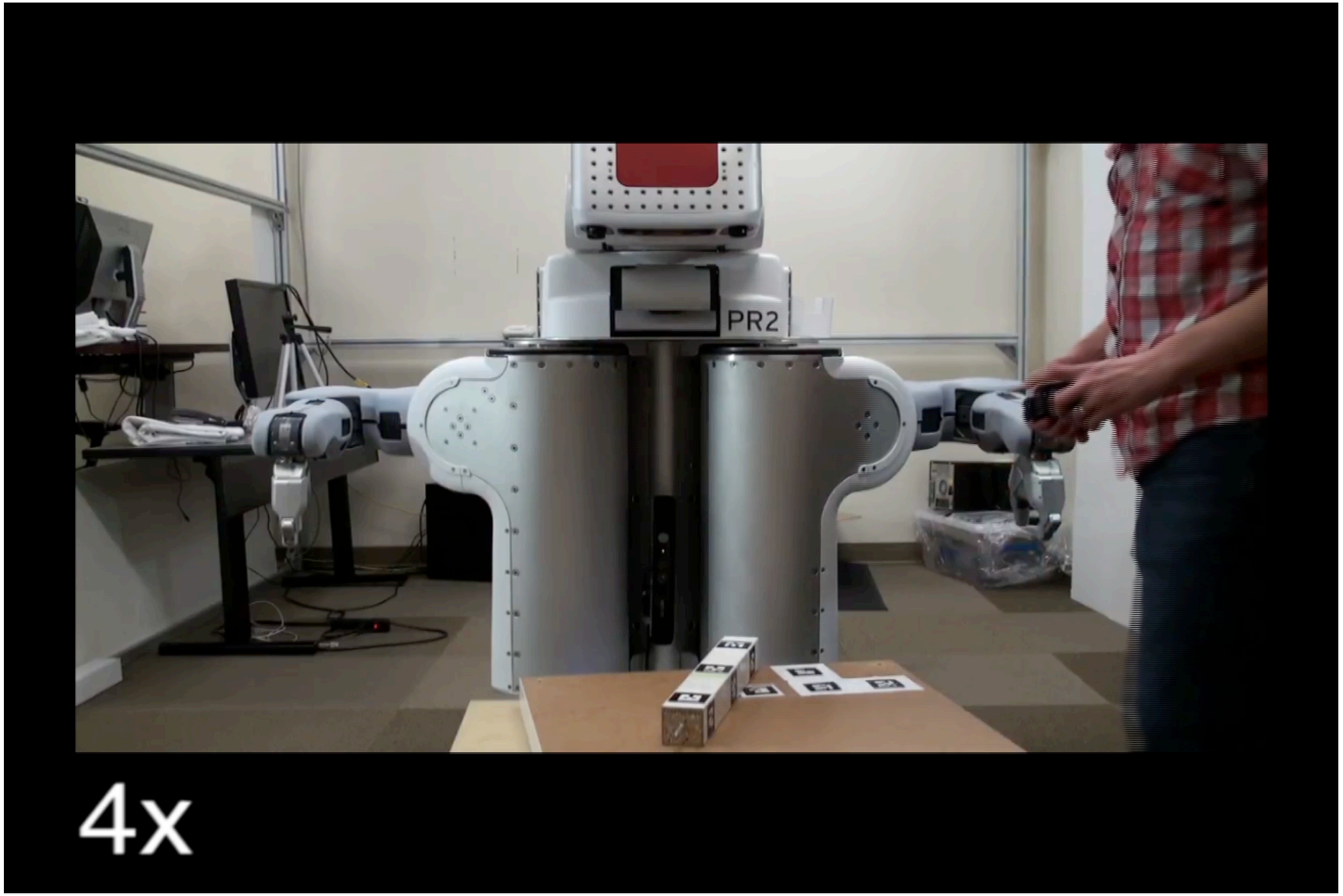
?



Finite-state task representation

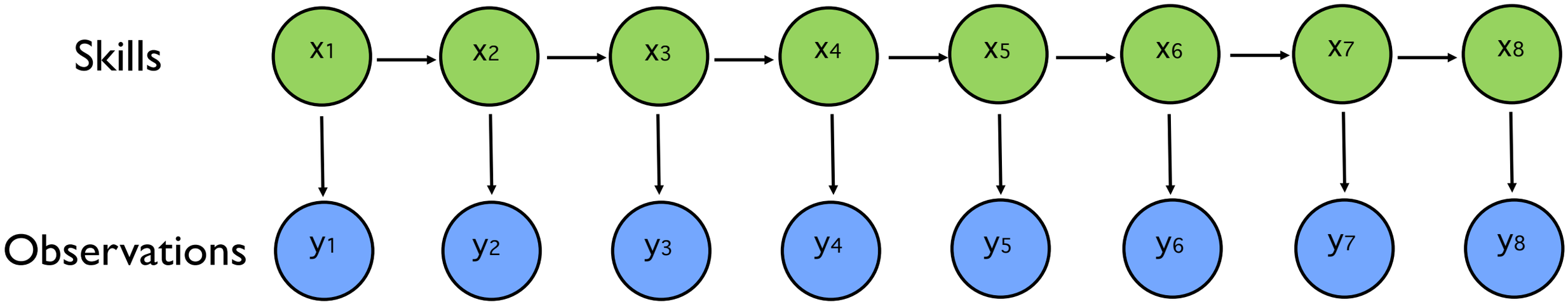
[Niekum et al. 2013]

Learning a task plan: Finite state automata



[Niekum et al. 2013]

Learning a task plan: Finite state automata

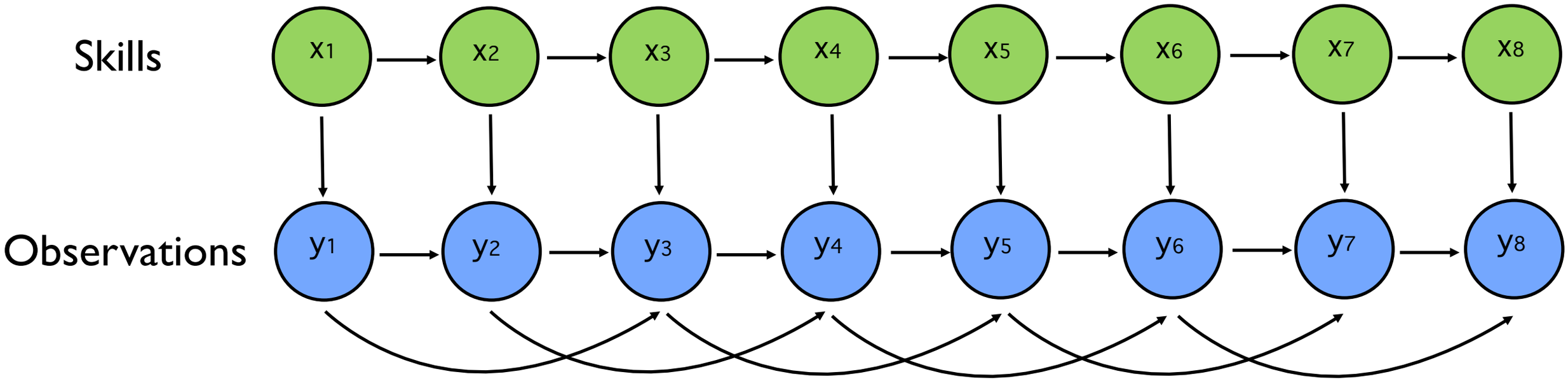


Standard Hidden Markov Model

[Niekum et al. 2013]

Learning a task plan: Finite state automata

$$\mathbf{y}_t^{(i)} = \sum_{j=1}^{r} A_{j,z_t^{(i)}} \mathbf{y}_{t-j}^{(i)} + \mathbf{e}_t^{(i)}(z_t^{(i)})$$

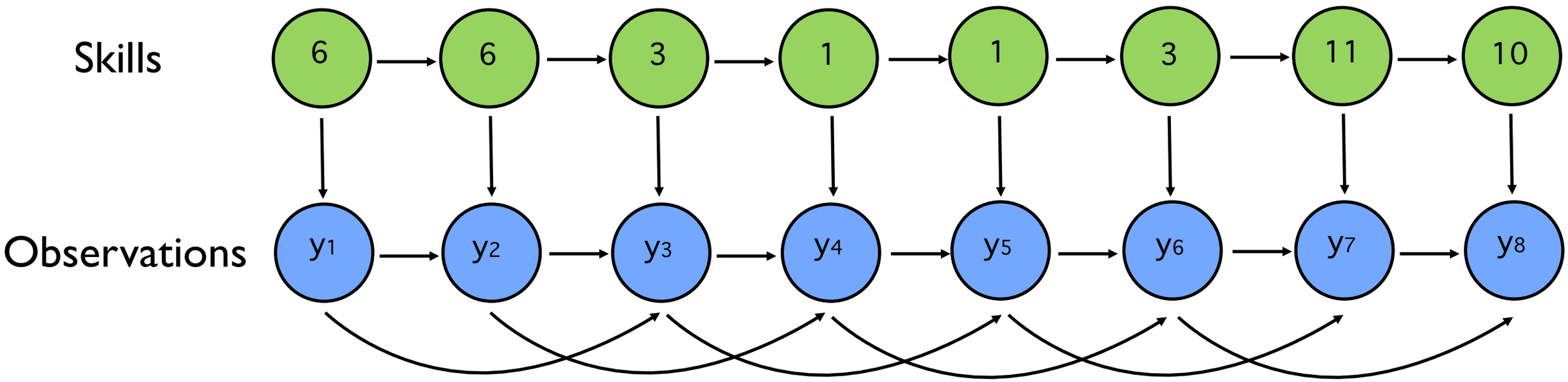


Autoregressive Hidden Markov Model

[Niekum et al. 2013]

Learning a task plan: Finite state automata

$$\mathbf{y}_t^{(i)} = \sum_{j=1}^{r} A_{j,z_t^{(i)}} \mathbf{y}_{t-j}^{(i)} + \mathbf{e}_t^{(i)}(z_t^{(i)})$$

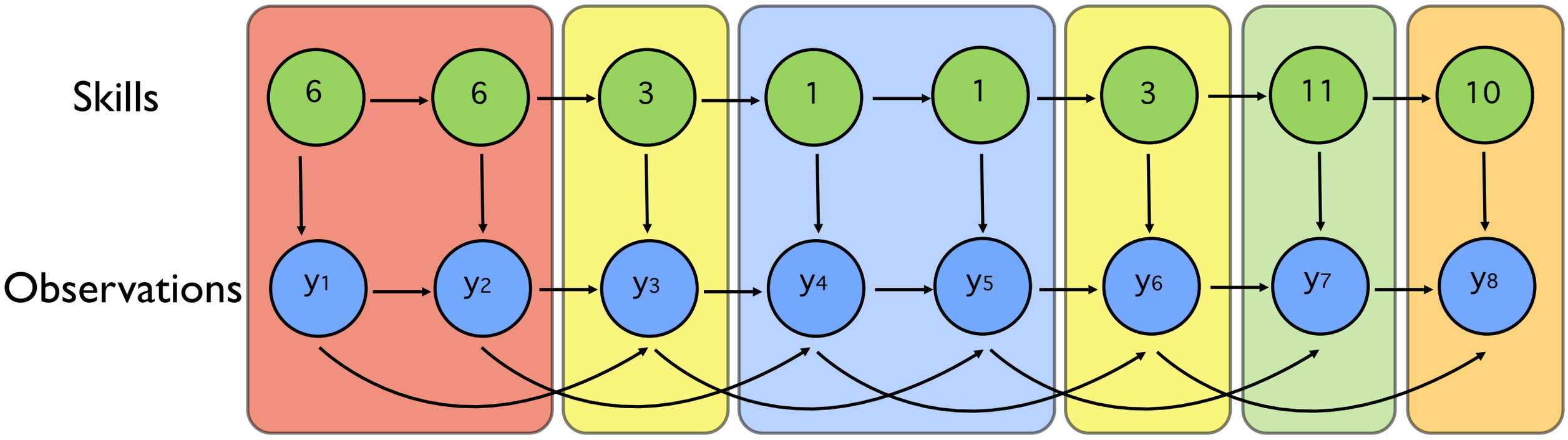


Autoregressive Hidden Markov Model

[Niekum et al. 2013]

Learning a task plan: Finite state automata

$$\mathbf{y}_t^{(i)} = \sum_{j=1}^{r} A_{j,z_t^{(i)}} \mathbf{y}_{t-j}^{(i)} + \mathbf{e}_t^{(i)}(z_t^{(i)})$$



Autoregressive Hidden Markov Model

[Niekum et al. 2013]

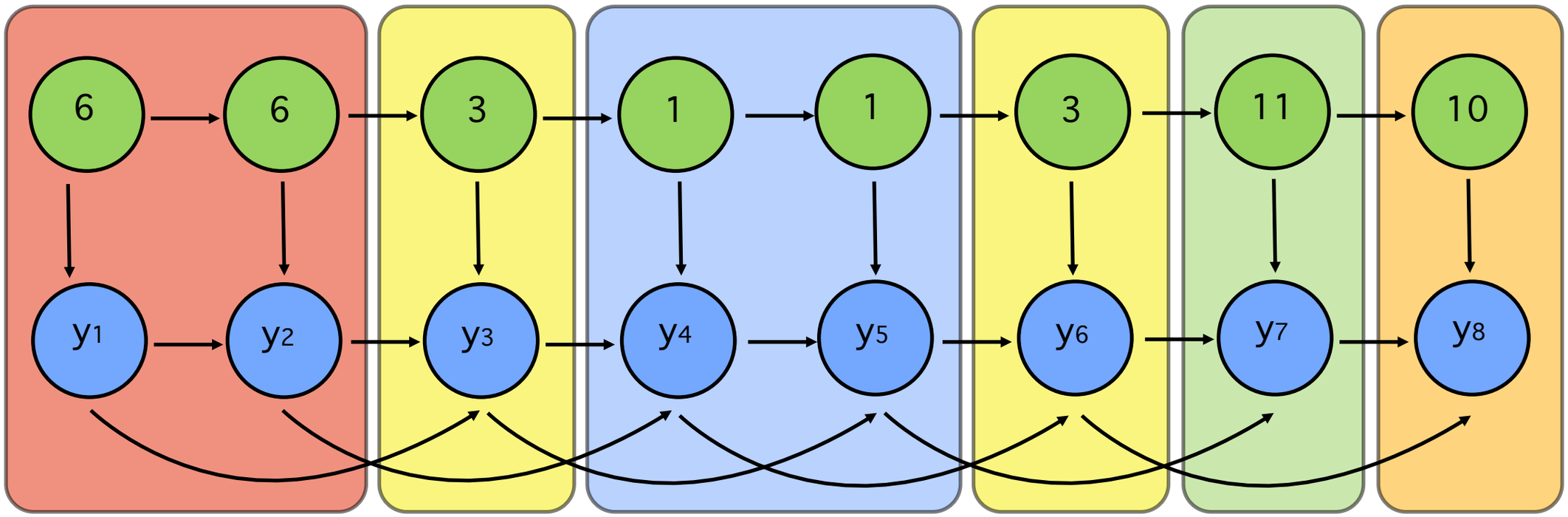
Learning a task plan: Finite state automata

$$\mathbf{y}_t^{(i)} = \sum_{j=1}^{r} A_{j,z_t^{(i)}} \mathbf{y}_{t-j}^{(i)} + \mathbf{e}_t^{(i)}(z_t^{(i)})$$

unknown number!

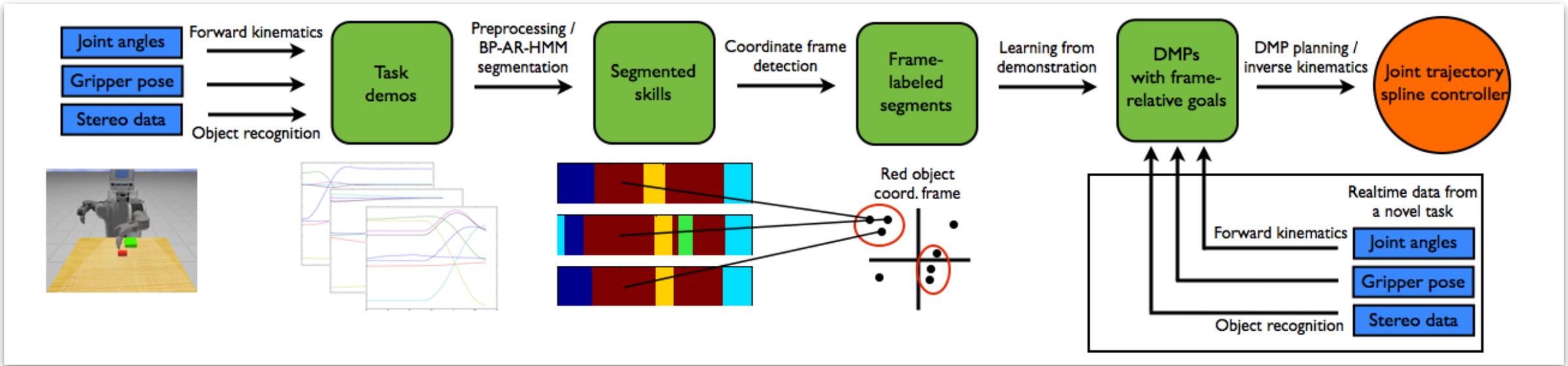
Skills

Observations



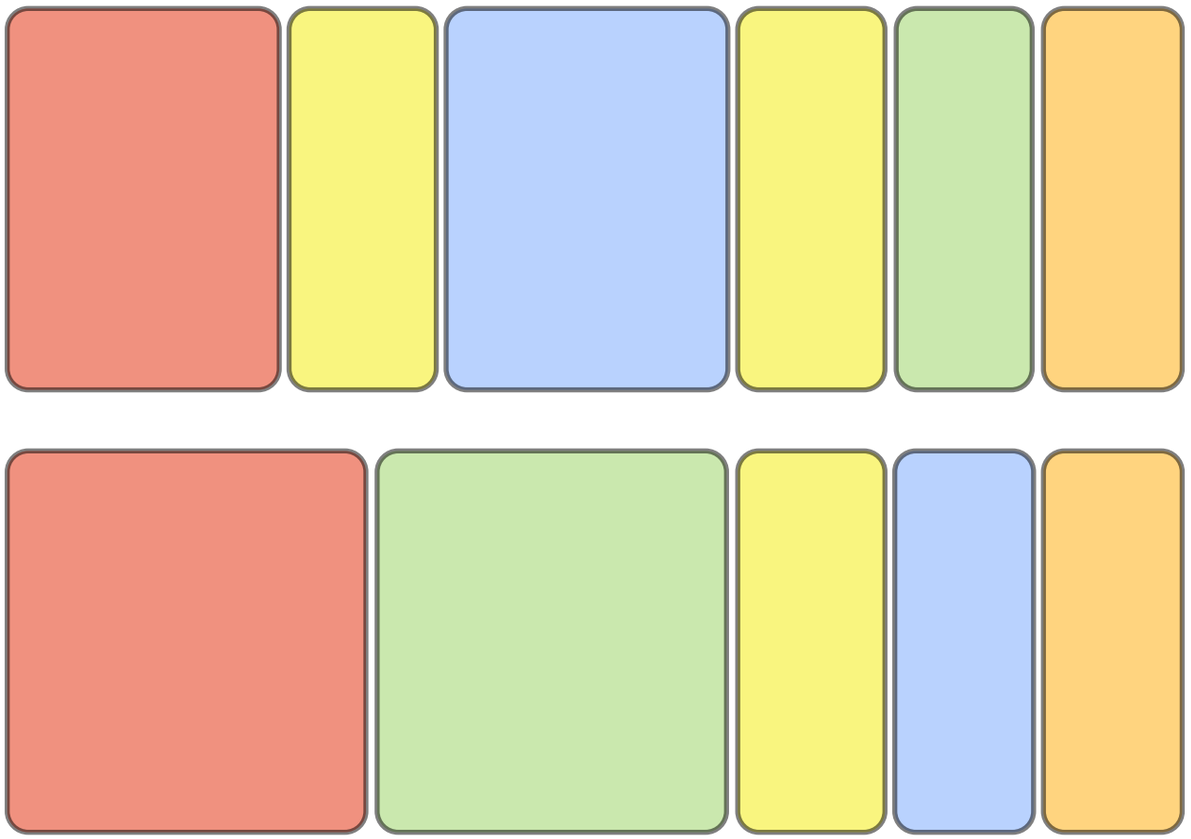
Beta Process Autoregressive Hidden Markov Model

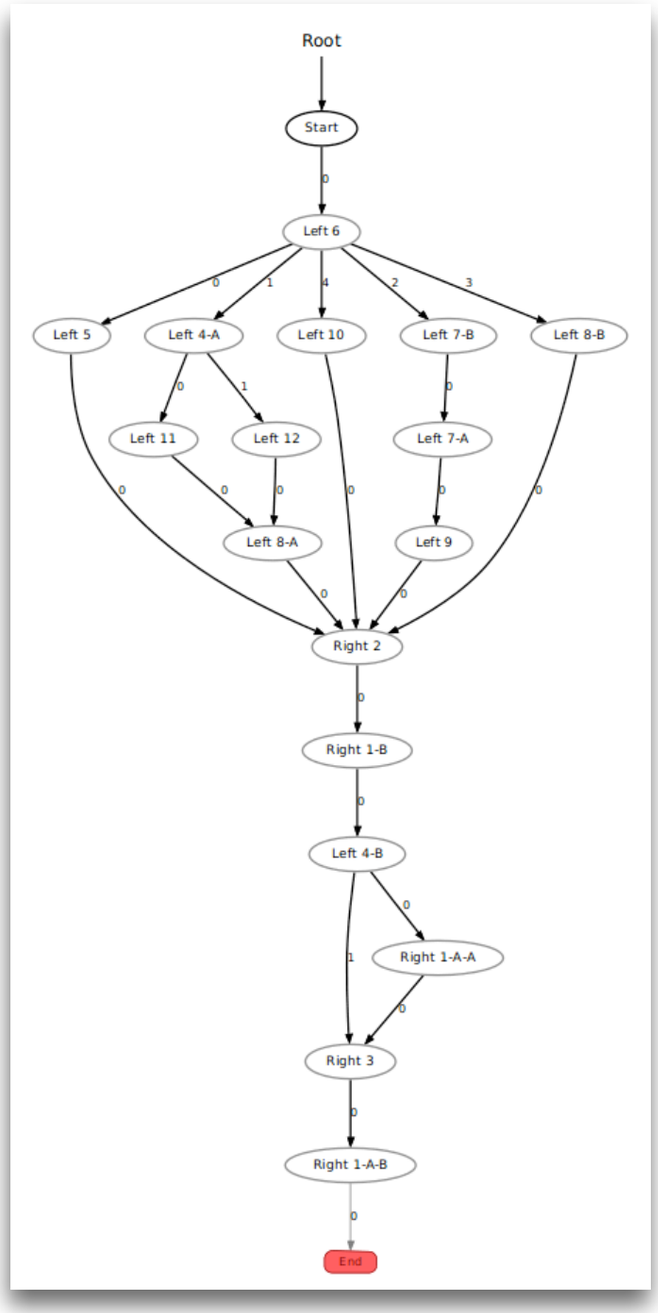
[Niekum et al. 2013]



Learning multi-step tasks from unstructured demonstrations

Learning a task plan: Finite state automata



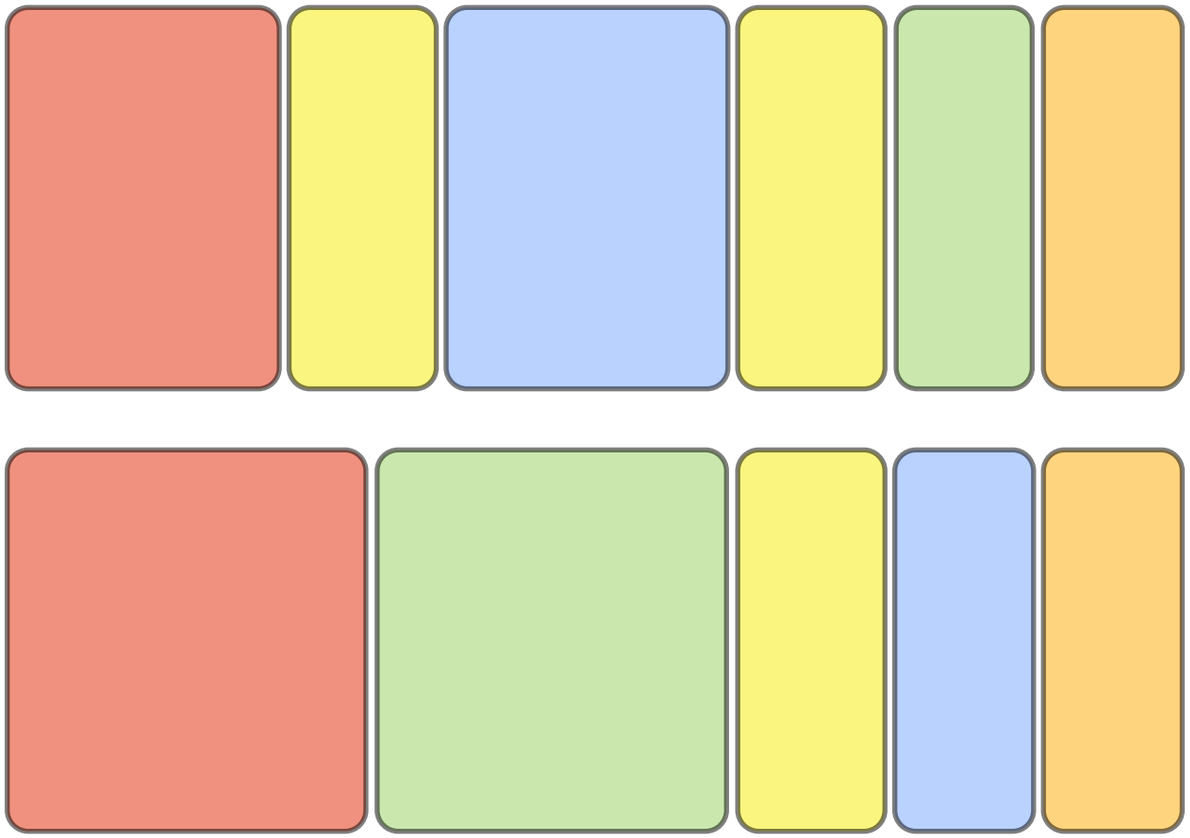


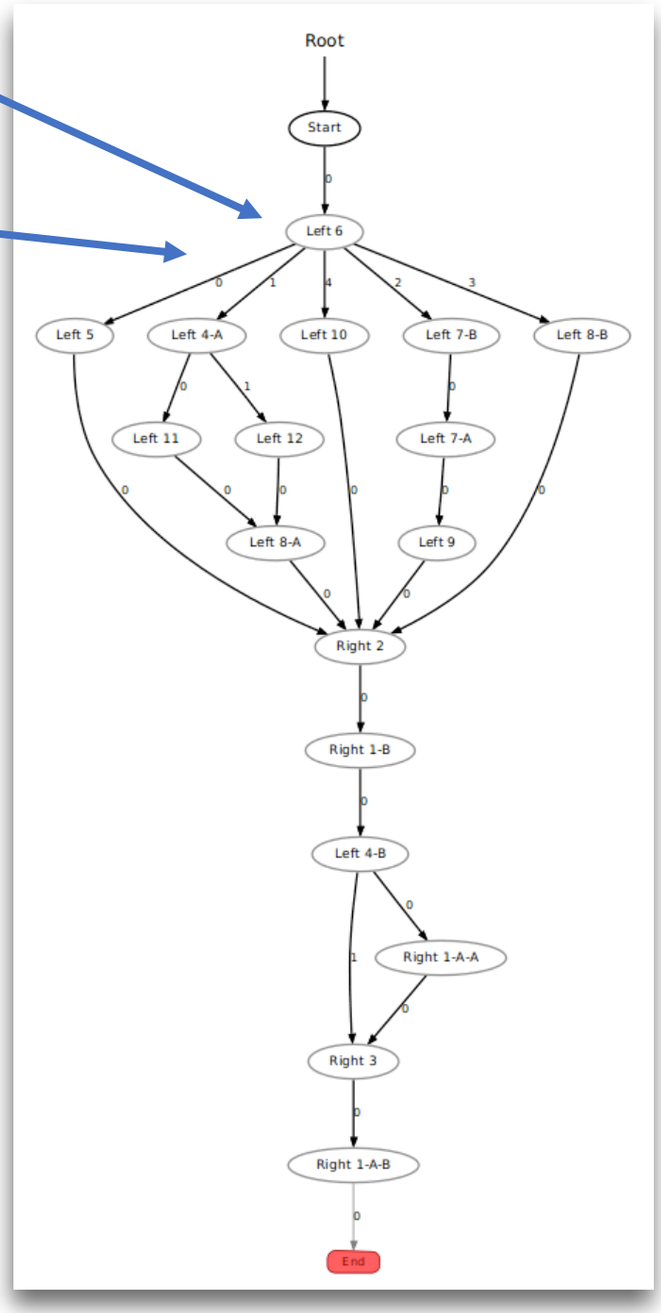
[Niekum et al. 2013]

Learning a task plan: Finite state automata

Controller built from motion category examples

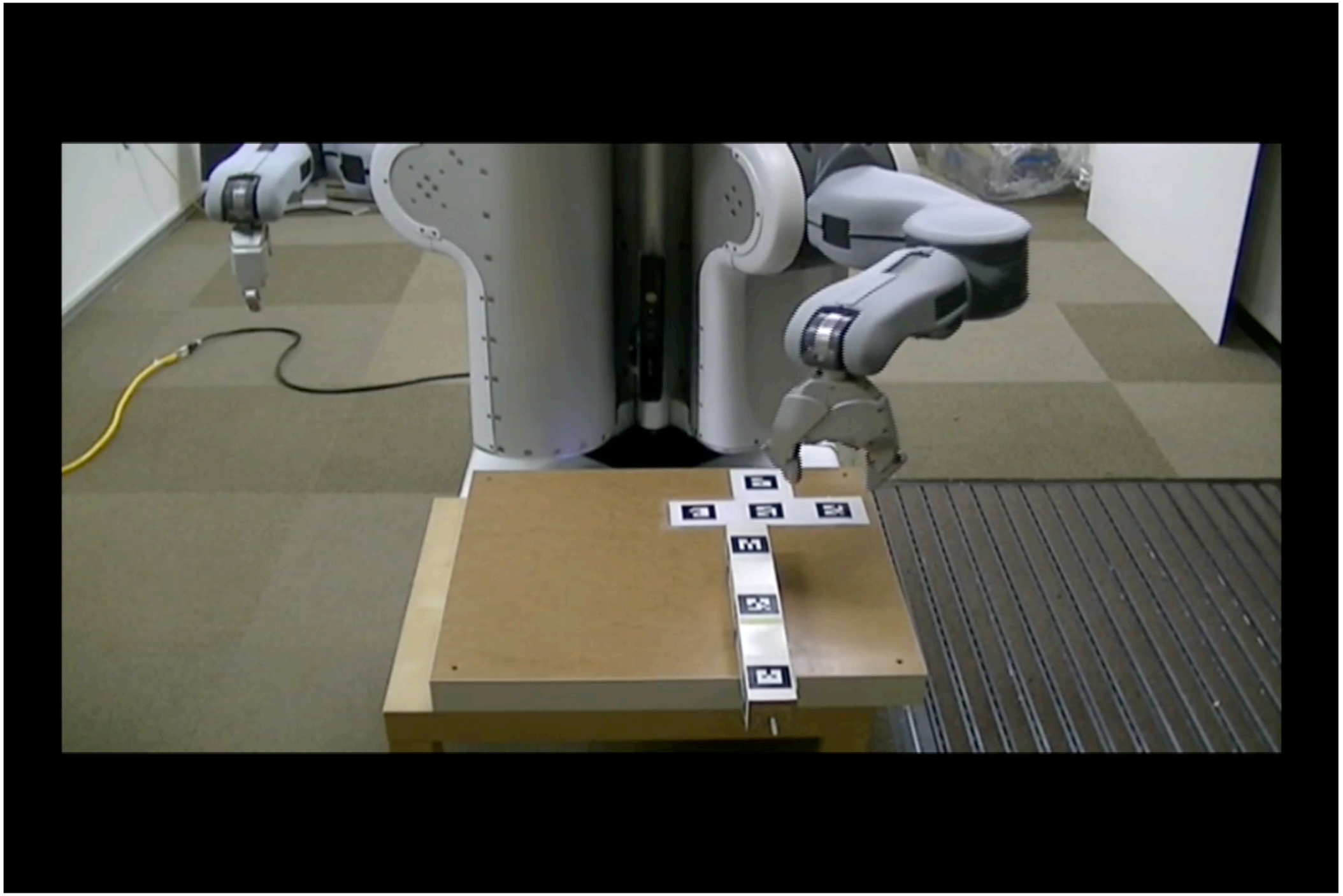
Classifier built from robot percepts





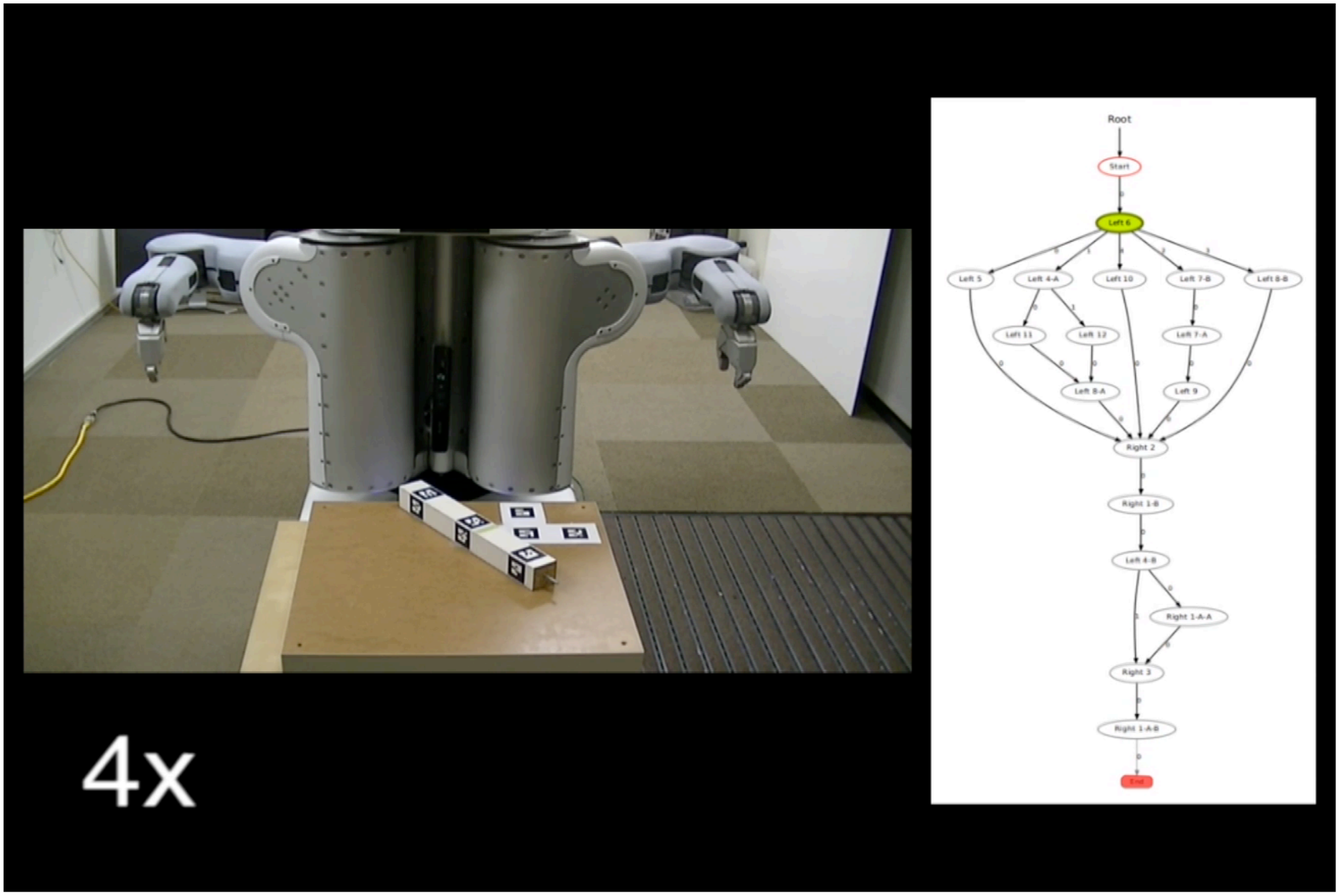
[Niekum et al. 2013]

Interactive corrections



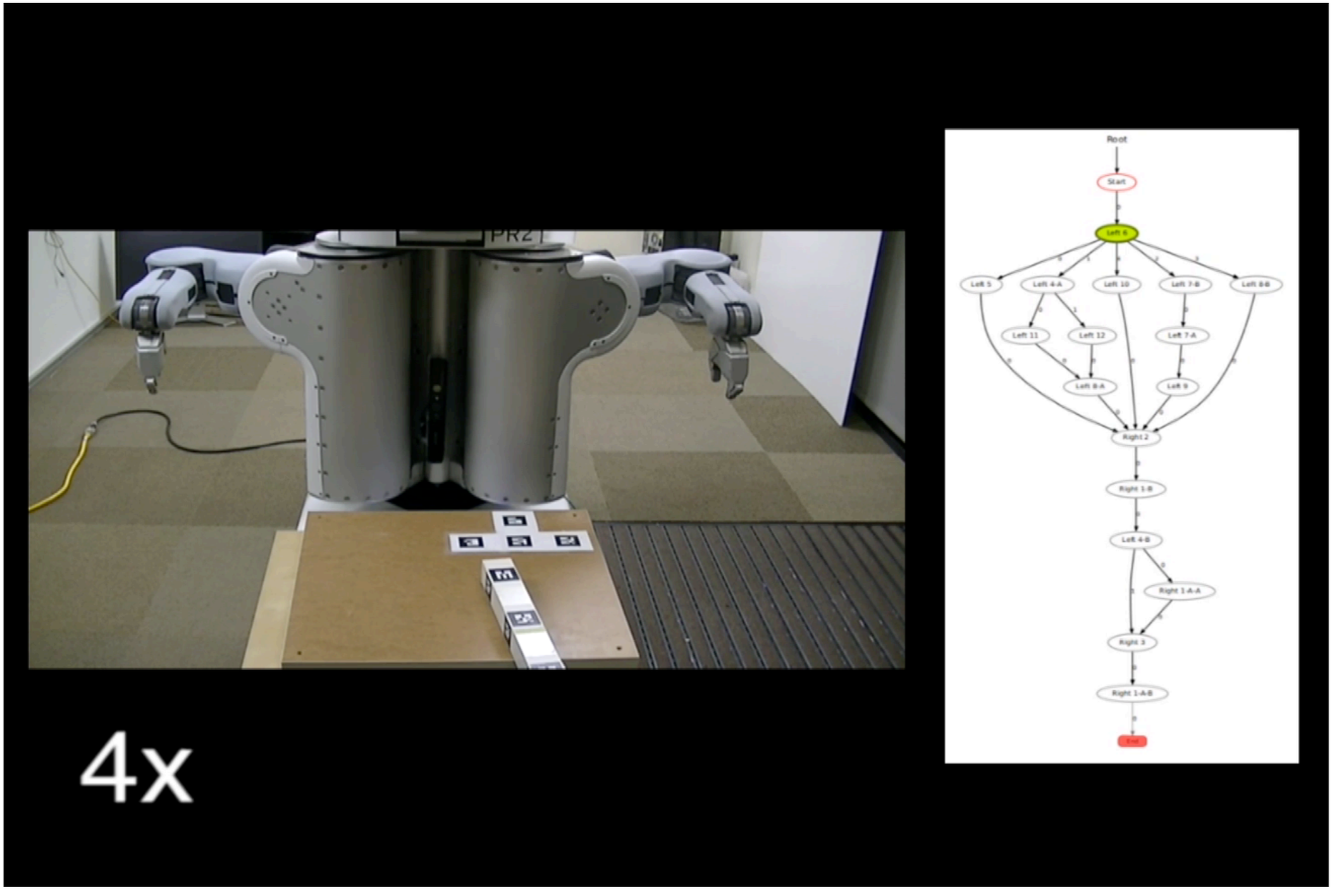
[Niekum et al. 2013]

Replay with corrections: missed grasp



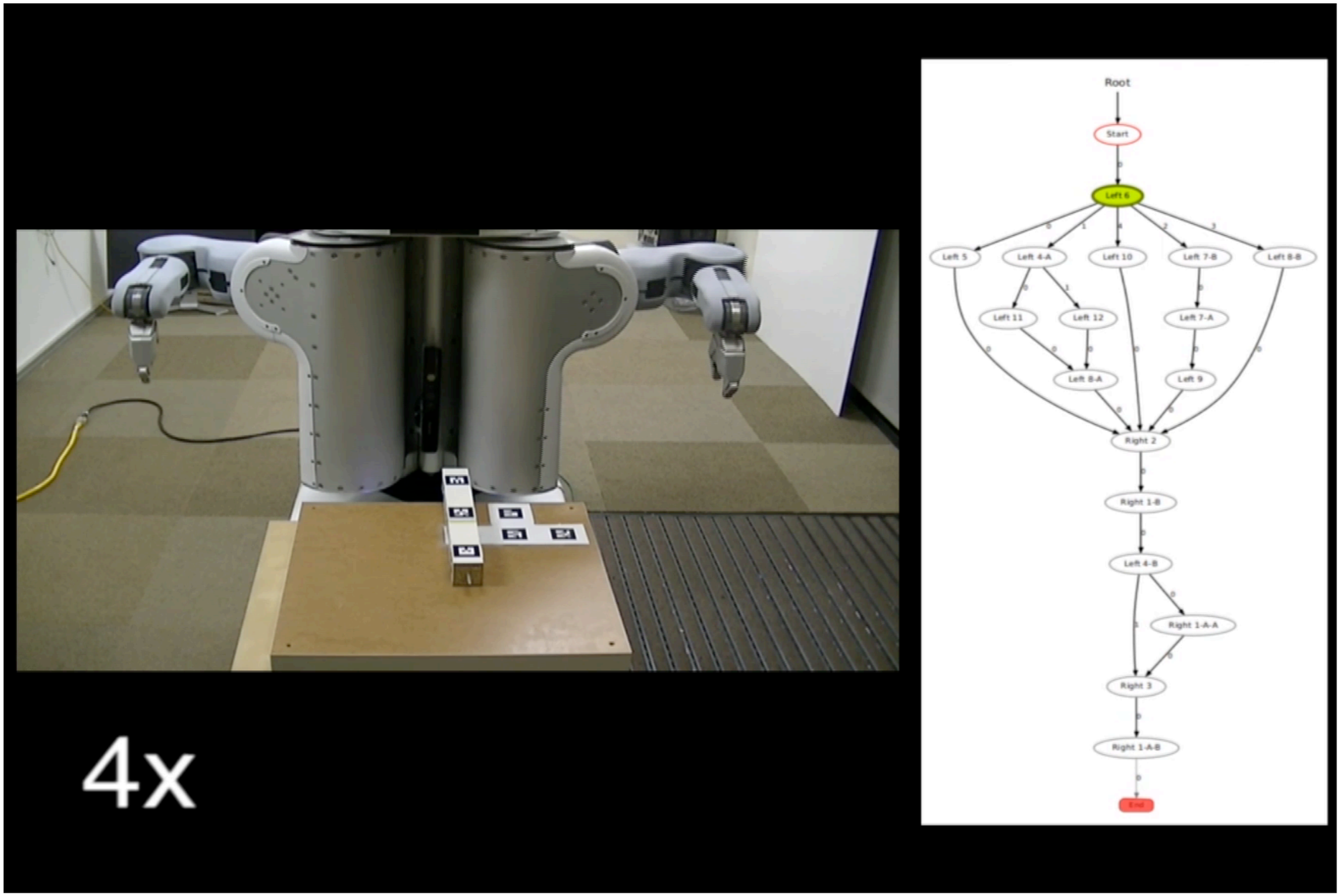
[Niekum et al. 2013]

Replay with corrections: too far away



[Niekum et al. 2013]

Replay with corrections: full run



[Niekum et al. 2013]

Learning task features

Learning a task plan

Learning task objectives

Learning object affordances

Learning task objectives: Inverse reinforcement learning

Using IRL + RL for super-human performance

Helicopter tricks
[Abbeel et al. 2007]

Littledog walking
[Kolter et al. 2007]

Learning task objectives: Inverse reinforcement learning

Reinforcement learning basics:

MDP: (S, A, T, γ, D, R)

states — actions — transition dynamics — discount rate — start state — reward function — start state distribution

Policy: $\pi(s, a) \rightarrow [0, 1]$

Value function: $V^{\pi}(s_0) = \sum_{t=0}^{\infty} \gamma^t R(s_t)$

IRL is an **MDP/R**.

Learning task features

Learning a task plan

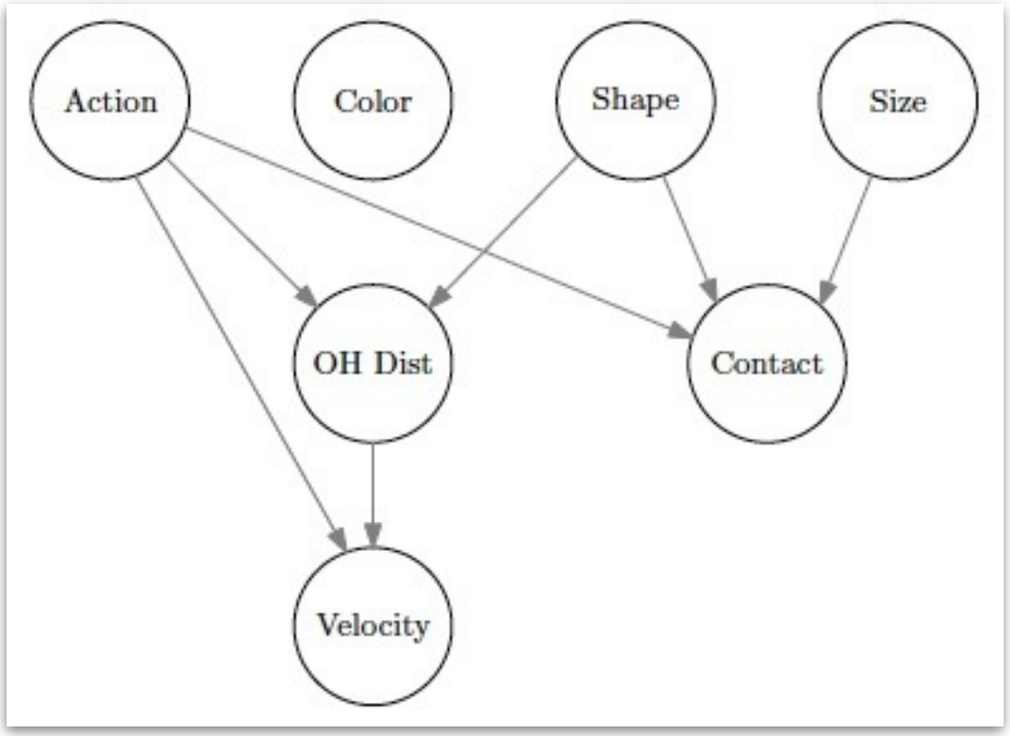
Learning task objectives

Learning object affordances

Learning object affordances: Action + object

Can we learn to recognize actions based on their effects on objects?

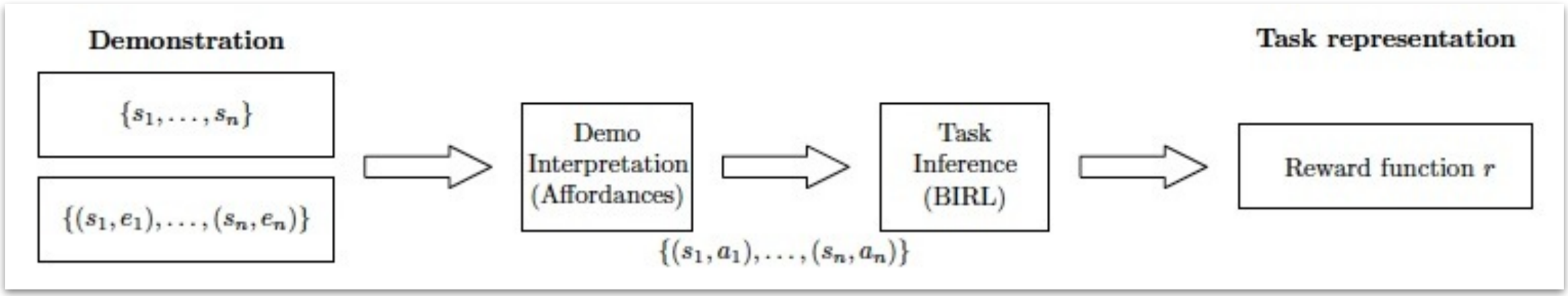
Random exploration



Object features: Color, shape, size
Actions: Grasp, tap, touch
Effects: Velocity, contact, object-hand distance

[Lopes et al. 2007]

Learning object affordances: Action + object



1. Interpret demonstrations using learned affordance Bayes net
   (based only on observed effects, which action is most likely at each step?)

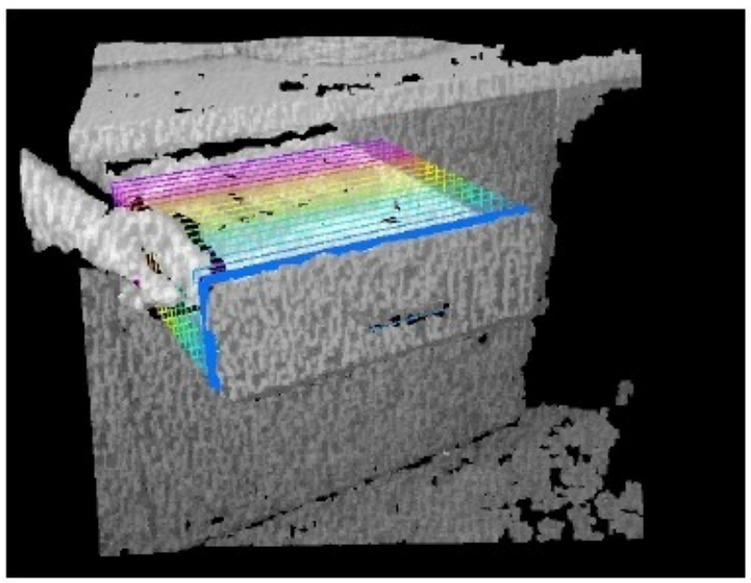
2. Use Bayes net to generate transition model
   (for each state, what does each action/object combo result in?)

3. Use transition model with Bayesian inverse reinforcement learning to infer task goals via a reward function
   (what is the likelihood of a demonstration under a particular reward function?)

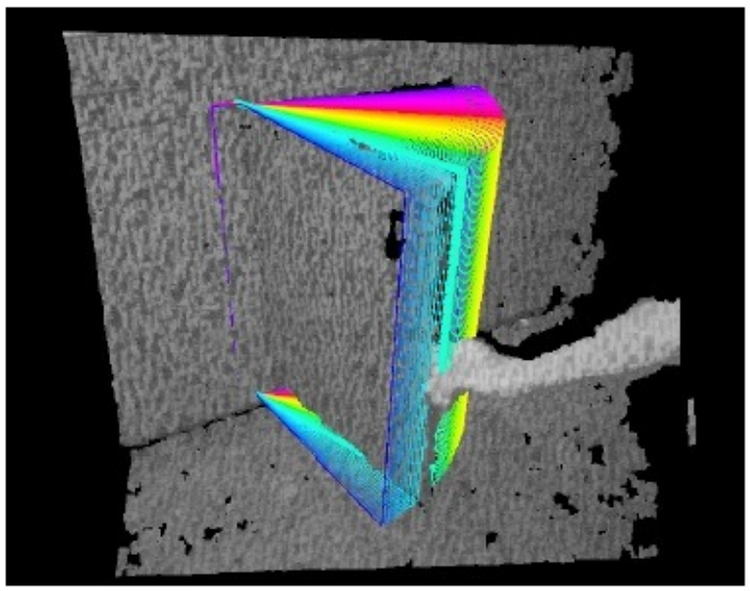
4. Use standard RL to improve task performance

[Lopes et al. 2007]

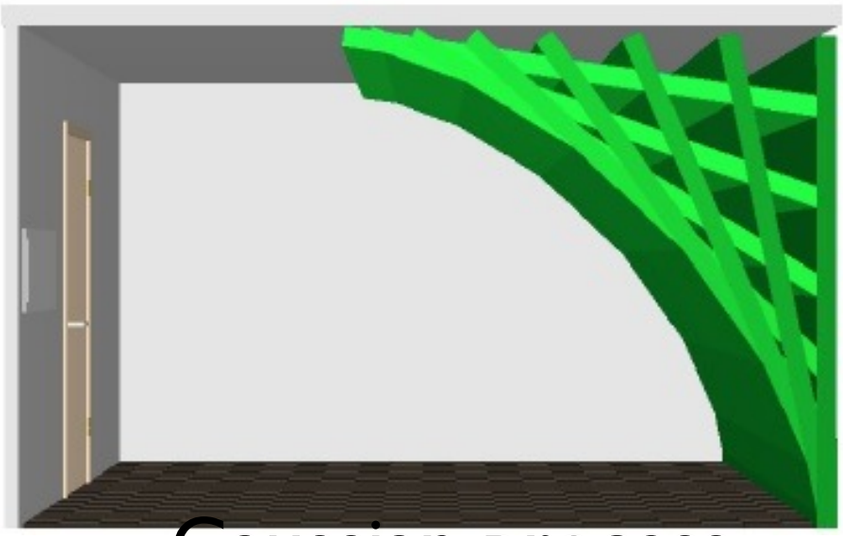
Learning object affordances: Articulation models



Prismatic - drawer

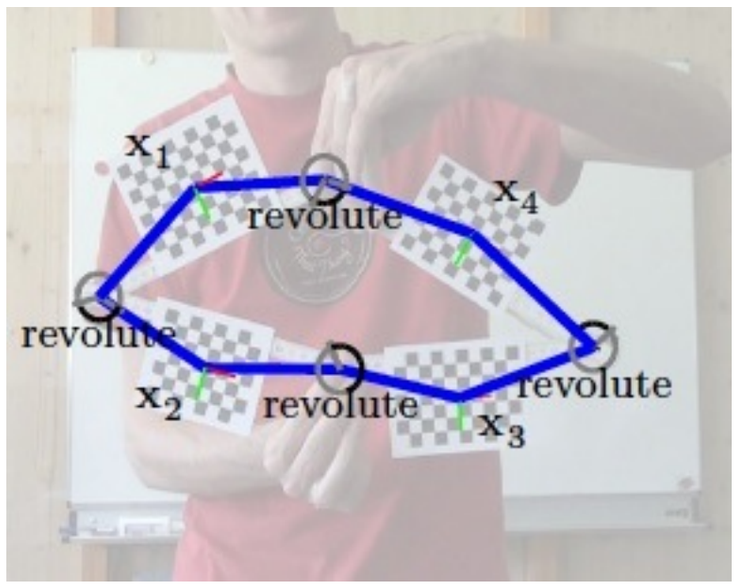


Revolute - cabinet



Gaussian process

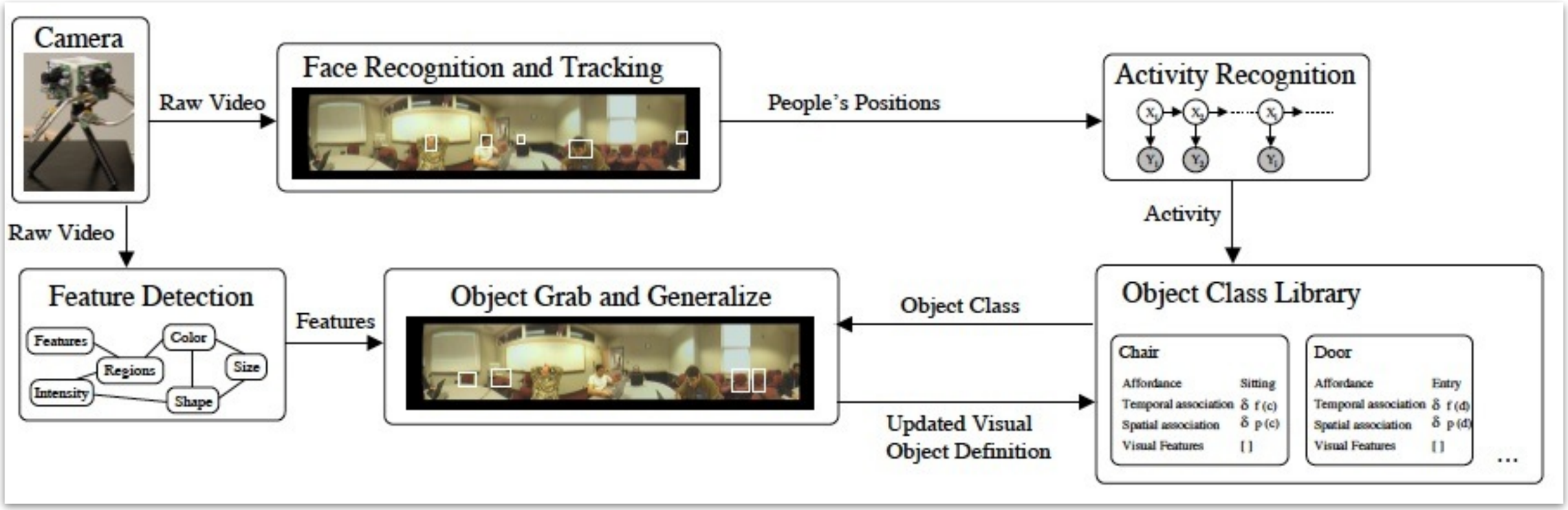
garage door



Infer full kinematic chain via Bayes net

[Sturm et al. 2011]

Learning object affordances: Functional identification

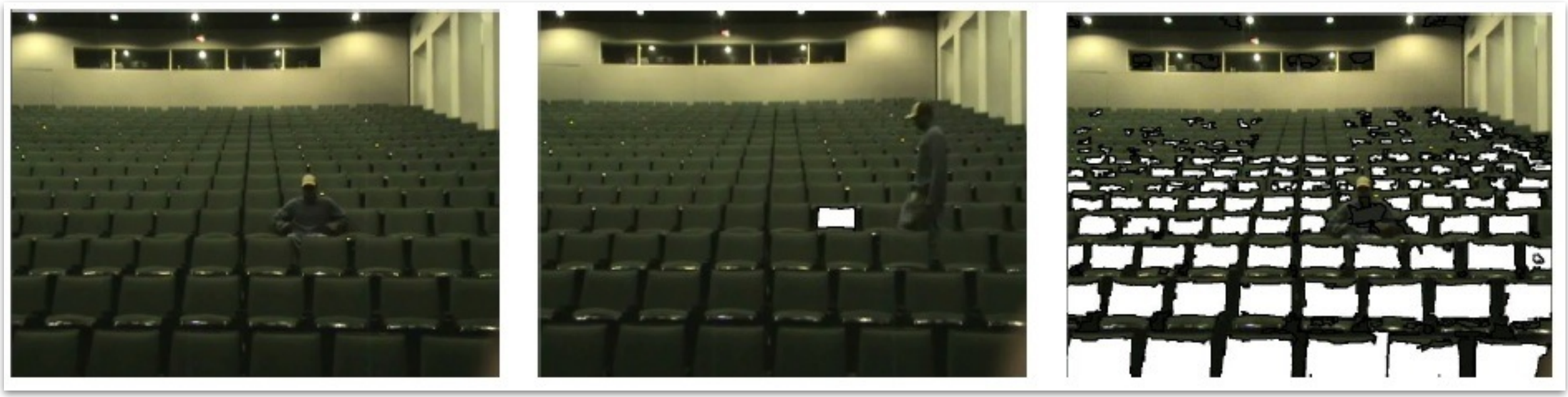


FOCUS (Finding Object Classification through Use and Structure):

Combine high-level activity recognition with low-level vision to learn how to recognize novel examples of known object classes.

[Veloso et al. 2005]

Learning object affordances: Functional identification



Recognize activity: sitting down

Predict object location and capture pixels

Generalize learned description

[Veloso et al. 2005]

Future directions

- Multiple tasks, libraries of skills, skill hierarchies
- Parameterized skills (pick up any object, hit ball to any location, etc.)
- ‘Common sense’ understanding of physics, actions, etc.
- Bridge the gap between low-level observations and high-level concepts
- Novel ways to leverage human insight (natural language + demonstrations, learning to ‘play’, etc.)

P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng. **An application of reinforcement learning to aerobatic helicopter flight**. In Neural Information Processing (NIPS'07), 2007.

P. Abbeel and A. Ng. **Apprenticeship learning via inverse reinforcement learning**. In Proceedings of the 21st International Conference on Machine Learning, 2004.

T. Cederborg, M. Li, A. Baranes, and P.-Y. Oudeyer. **Incremental local online gaussian mixture regression for imitation learning of multiple tasks**. In 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010.

Luis C Cobo, Peng Zang, Charles L Isbell Jr, Andrea L Thomaz, and Charles L Isbell Jr. **Automatic state abstraction from demonstration**. In Twenty-Second International Joint Conference on Artificial Intelligence, 2009.

G. Konidaris, S. Kuindersma, R. Grupen, and A. Barto. **Robot learning from demonstration by constructing skill trees**. The International Journal of Robotics Research, 31(3):360–375, December 2011.

M. V. Lent and J. E. Laird. **Learning procedural knowledge through observation**. In K-CAP '01: Proceedings of the 1st International Conference on Knowledge Capture, 2001.

M. Lopes, F. S. Melo, and L. Montesano. **Affordance-based imitation learning in robots**. 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, October 2007.

S. Niekum, S. Osentoski, C.G. Atkeson, A.G. Barto. **Online Bayesian Changepoint Detection for Articulated Motion Models**. IEEE International Conference on Robotics and Automation (submitted), May 2015.

S. Niekum, S. Chitta, B. Marthi, S. Osentoski, and A. G Barto. **Incremental semantically grounded learning from demonstration**. In Robotics Science and Systems, 2013.

P. E Rybski, K. Yoon, J. Stolarz, and M. Veloso. **Interactive robot task training through dialog and demonstration**. In HRI '07: Proceedings of the ACM/IEEE international conference on Human-robot interaction, 2007.

M. Veloso, F. V. Hundelshausen, and P. E Rybski. **Learning visual object definitions by observing human activities**. In 5th IEEE-RAS International Conference on Humanoid Robots, 2005.

B. Akgun, M. Cakmak, J. Yoo, and A. L. Thomaz. **Trajectories and Keyframes for Kinesthetic Teaching: A Human-Robot Interaction Perspective**. In Proceedings of the International Conference on Human-Robot Interaction, 2012.

W. B. Knox and P. Stone. **Tamer: Training an agent manually via evaluative reinforcement**. In Proc. of the 7th IEEE International Conference on Development and Learning, 2008.

S. Tellex, T. Kollar, S. Dickerson, M. R. Walter, A. G. Banerjee, S. J. Teller, and N. Roy. **Understanding Natural Language Commands for Robotic Navigation and Mobile Manipulation**. AAAI, 2011.