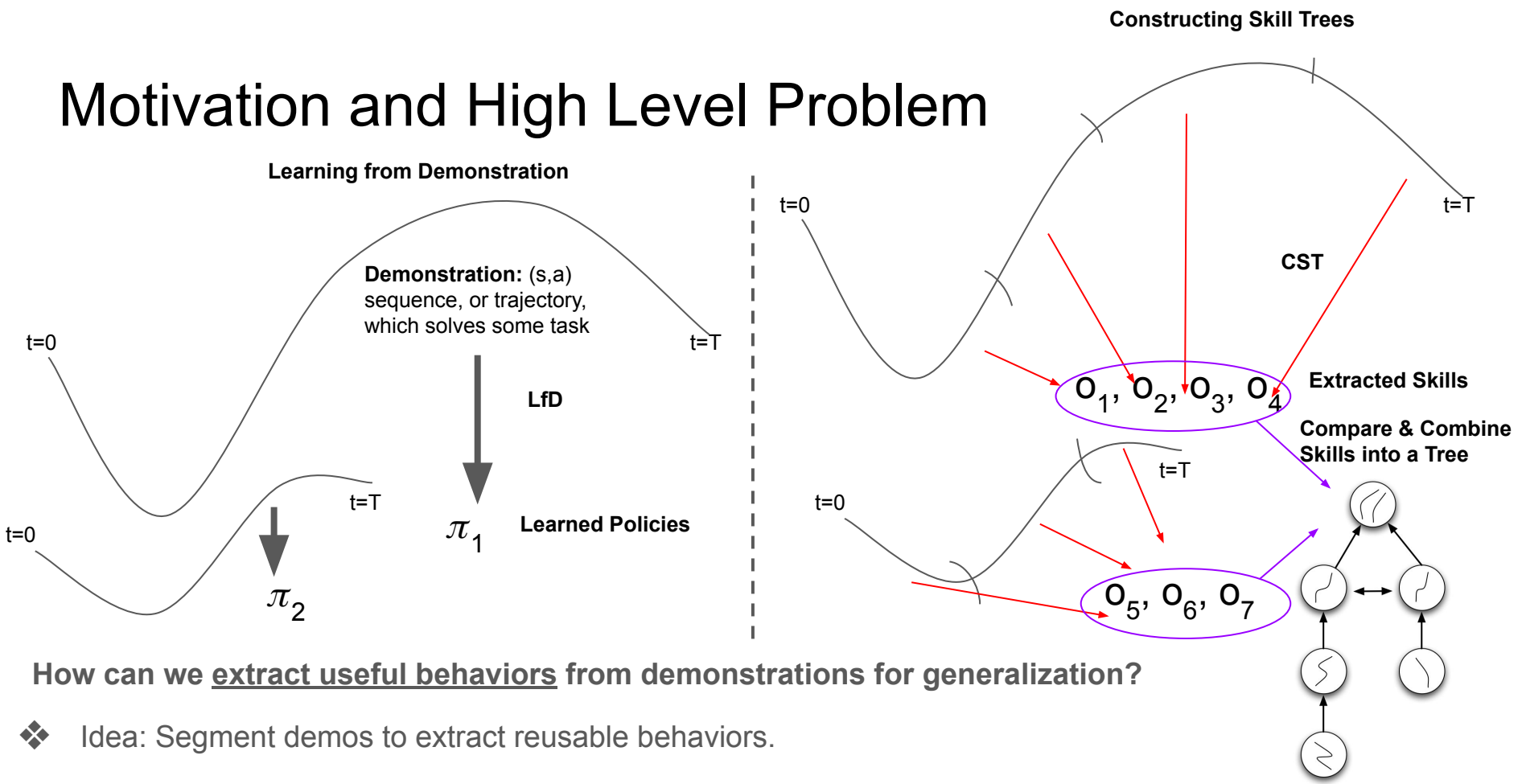
Robot Learning from Demonstration by Constructing Skill Trees

George Konidaris, Scott Kuindersma, Roderic Grupen, Andrew Barto

Presenter: Eric Hsiung

Thursday October 20th 2022

Motivation and High Level Problem



How can we <u>extract useful behaviors</u> from demonstrations for generalization?

- Idea: Segment demos to extract reusable behaviors.

Problem Setting

Given:

- Multiple demonstrations in a given RL environment
  - Assume demonstrations gathered from same MDP (S, A, T, γ, R)

Goal:

- Extract skills from each trajectory, and determine if pairs of skills are similar enough to be combined
- This leads to constructing a skill tree

Related Work

Learning from Demonstration (LfD) [Schaal, NeurIPS 1996] [Argall et al., 2009]

- Typically want to learn optimal policies which can reproduce such trajectories

Skill Acquisition, Abstraction [Konidaris and Barto, NeurIPS 2009][Konidaris and Barto, IJCAI 2009]

- Skill chaining – find a sequence of skills to achieve a goal (by intersecting initiation sets together). **Skill trees introed by this paper.**
- State Abstraction – make decisions in a lower dimensional space, or “smaller” state space.

Segmentation [Dixon and Khosla, 2004a, 2004b]

- Distinction – segmented trajectories into policies that are linear robot state variables (vs inferring value functions in this current paper)

Key Contributions

Components:

- Segmenting demonstrations leads to skill sequences
- Skills in skill sequences have their own goals; demonstrations have subgoals
- Each skill is defined in terms of skill-specific abstractions
- Merge similar skills together (from separate sequences) into a tree

Simultaneously perform statistical trajectory segmentation to extract skills with goals, and associate skill-specific abstractions.

Background

Reinforcement Learning

Value functions ←→ discounted returns

Options

Temporally extended actions

Abstraction Selection

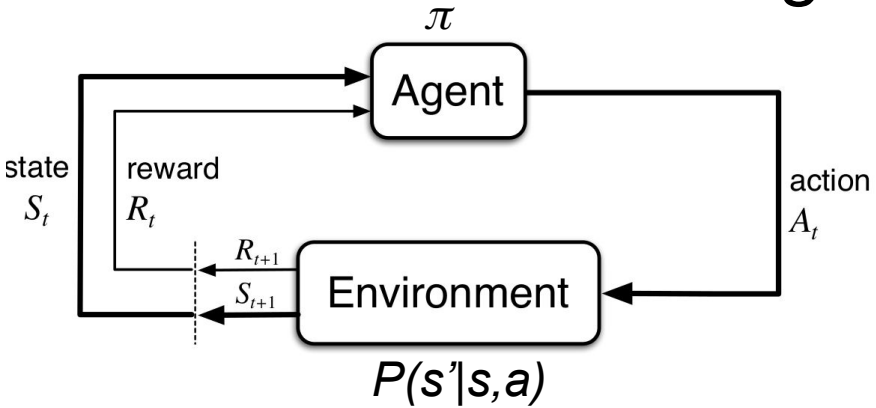
Use only relevant features / dimensionality reduction

Changepoint Detection

Were two consecutive state-action subsequences generated differently?

Reinforcement Learning - RL

Markov Decision Process (MDP) –
fully observable + Markov assumption



Typical RL diagram in fully observable environments (i.e. MDPs).

$$M = (S, A, P, \mathcal{R}, \gamma)$$

Set of possible states (state space)
Set of possible actions (action space)
Transition model
Reward model generates a reward
Discount factor

Objective: $\underset{\pi}{\text{argmax}} \quad R_\pi(s) = E\left[\sum_{i=0}^{\infty} \gamma^i r_i \,\middle|\, s_0 = s\right]$

Agent must learn the optimal policy π which maximizes the expected discounted return from each state.
However, the agent only has access to samples from the environment.

Approximating Value Functions

Key Insight: Which basis sets are best for representing value functions for different parts of the demonstration?

$$\hat{V}(\mathbf{x}) = \sum_{i=1}^{k} w_i \phi_i(\mathbf{x})$$

Linear value function approximation – approximation w/ set of basis functions

- This paper uses a Fourier basis (inspired by Fourier series)

$$\phi_i(\mathbf{x}) = \cos\left(\pi \mathbf{c}^i \cdot \mathbf{x}\right) \text{ where } \mathbf{c}^i = [c_1, ..., c_d],\ c_j \in \{0, ..., k\},\ 1 \leq j \leq d$$

Options + Abstraction Selection

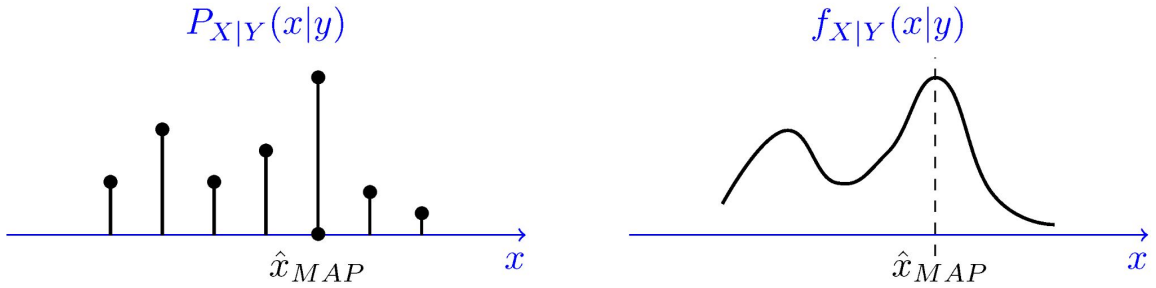
Key insight: Assume abstractions are associated with basis functions and options.

Options (skills)

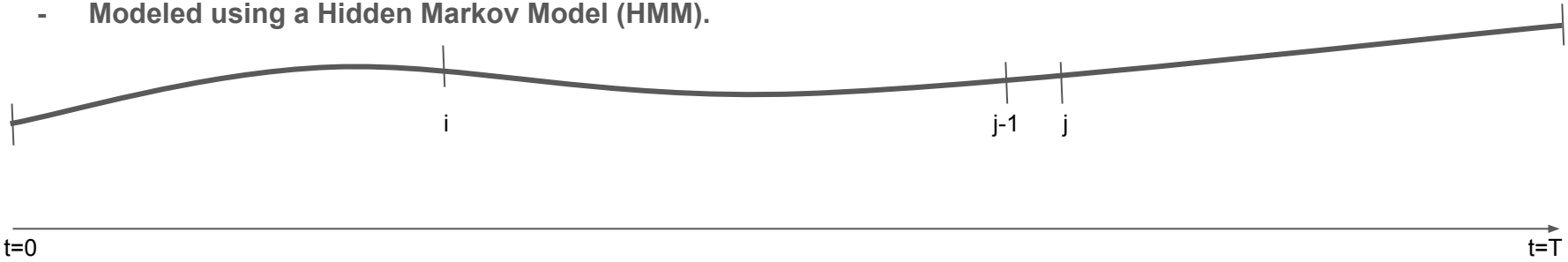
- Option Policy – operating with a specific state abstraction
- Initiation Set – set of states over which an option can executed
- Termination Condition – describes probability where an option ends

Changepoint Detection

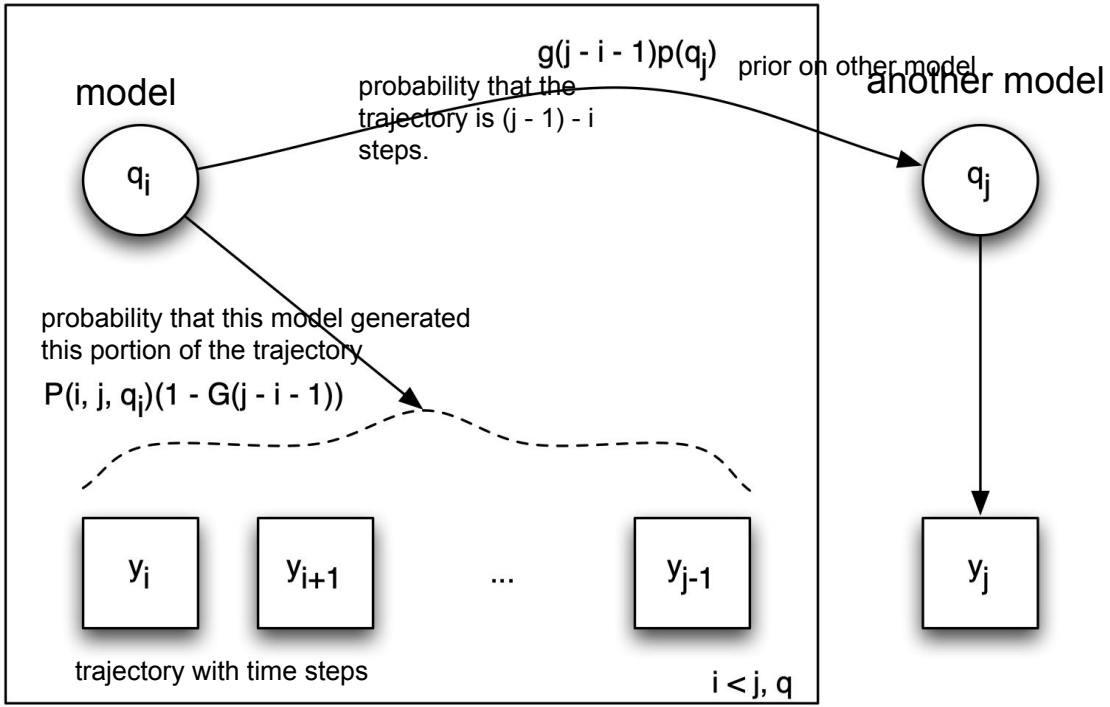
Maximum a posteriori estimation – which *future* timestep in the trajectory has the highest probability density of being generated by a different model?



- **Used a Viterbi algorithm to obtain MAP changepoints**
- **Modeled using a Hidden Markov Model (HMM).**



Hidden Markov Model of Changepoint Detection



Model (q):
Linear value function approximation (via basis set) + the abstraction

Priors:
segment length ~ Geom

Stratified Optimal Resampling [Fearnhead and Liu 2007] to filter particles.

They use Fearhead and Liu's online MAP changepoint detection algorithm.

Online Trajectory Segmentation into Skill Chain

Input: Demonstration, Fourier Basis, Abstractions

1. Compute MAP estimates using conjugate priors in closed form.

2. Compute sufficient statistics and Viterbi path in an online fashion.
   a. Future timestep statistics **depend only on previous timestep statistics**
   b. Each particle stores:

Changepoint	Viterbi Path (likely sequence of models)
Value Function Model	Basis functions evaluated at current state
MAP Estimate	Discounted future return from current state

3. Target is the sample return

   Which Viterbi path results in predicted return closest to target?

Output: Skill Chain ← Changepoints + Viterbi Path (model sequence)

Merging Skills

Merge statistically similar segments via conditional probability of future changepoint from time t, given a model q

$$P(j,t,q) = \frac{\pi^{-\frac{n}{2}}}{\delta^m} |(\mathbf{A}_q + \mathbf{D})^{-1}|^{\frac{1}{2}} \frac{u^{\frac{v}{2}}}{(y_q + u)^{\frac{n+v}{2}}} \frac{\Gamma(\frac{n+v}{2})}{\Gamma(\frac{v}{2})}, \qquad (12)$$

– What is the probability that two segments were generated by the same skill?

– What is the probability that two segments were generated by two different skills?

Evaluations

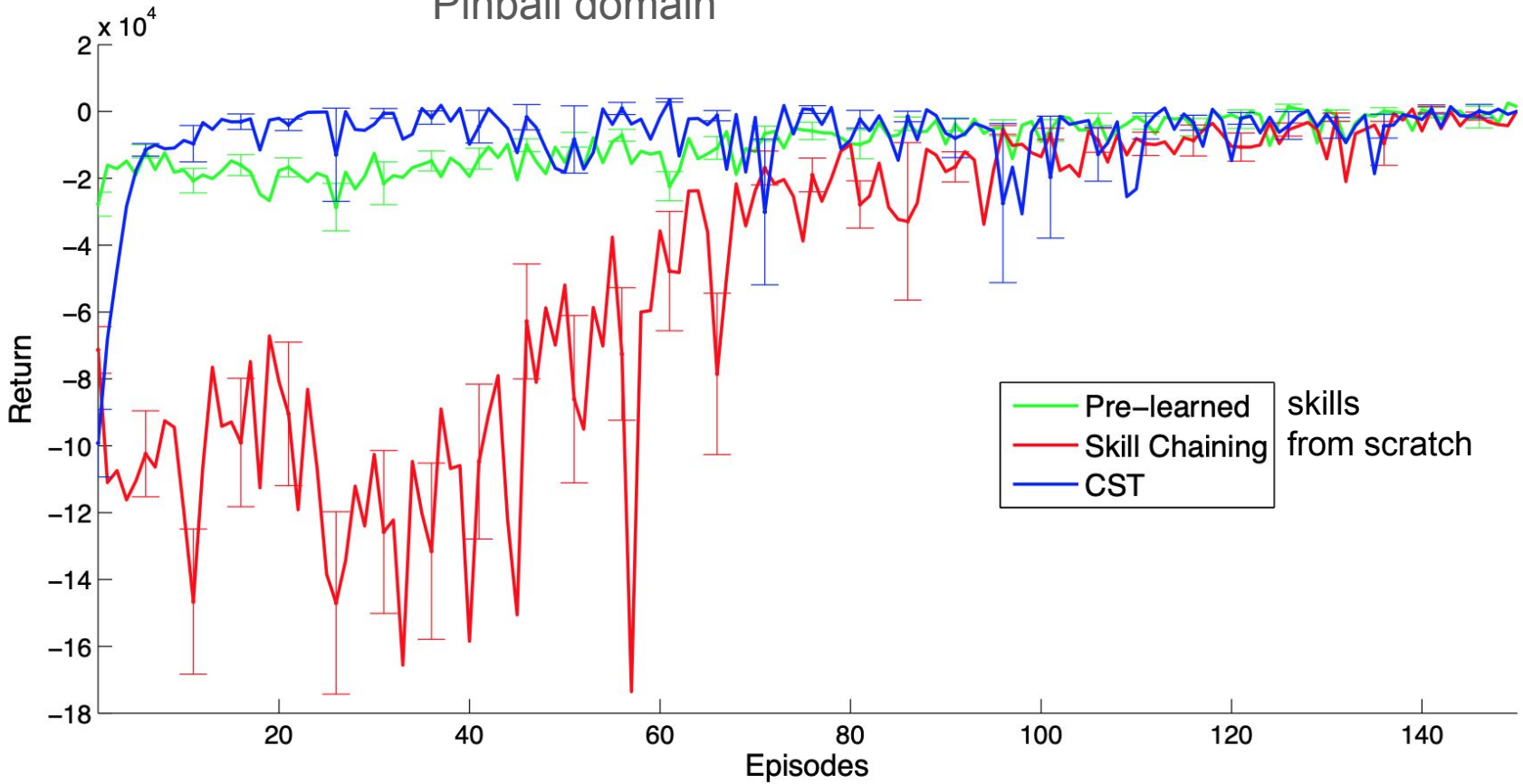
Pinball domain

Demonstration 1 segmentation

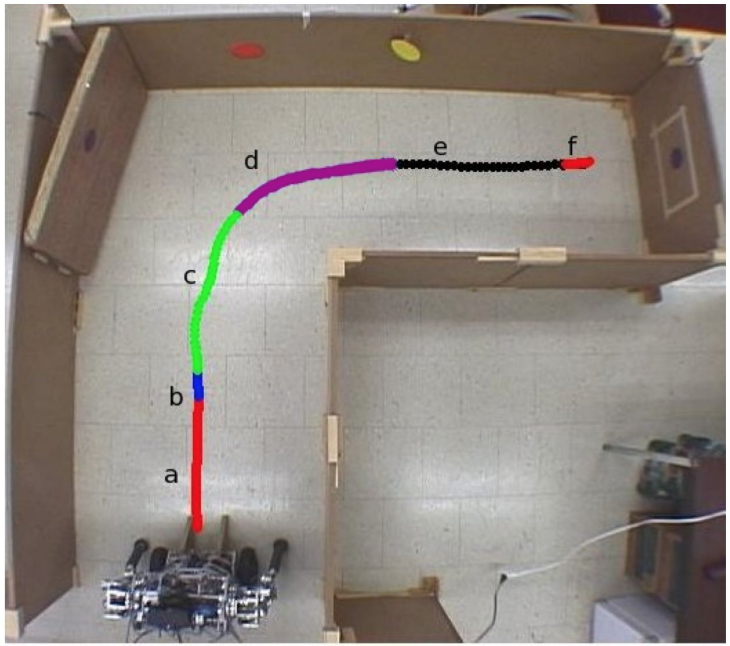
Demonstration 2 segmentation

Merged Skill Tree

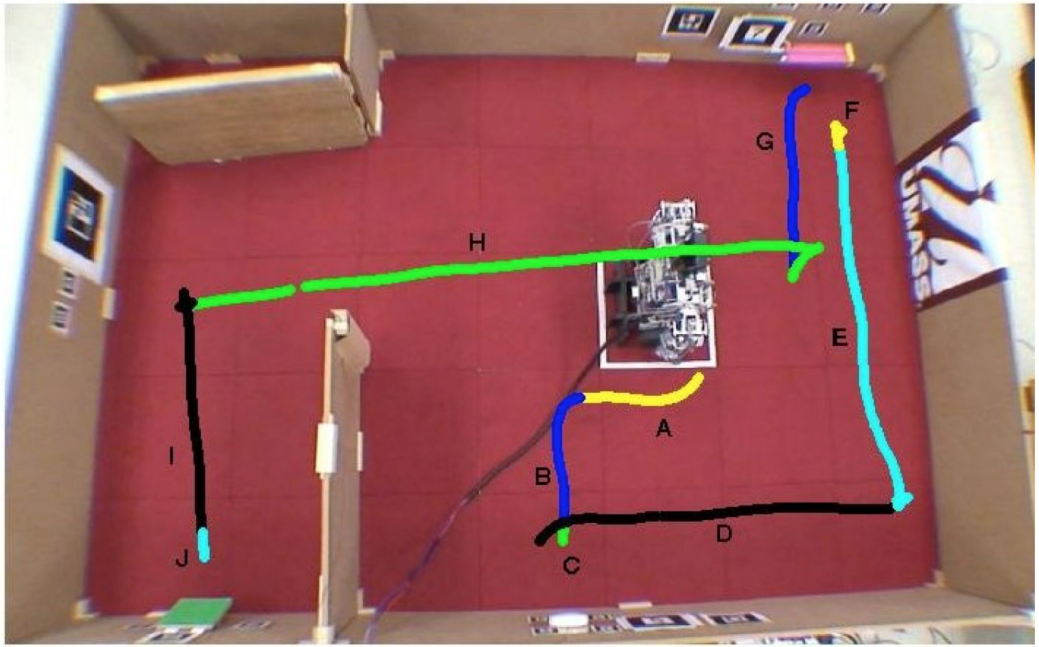
Evaluations



Evaluations - Mobile Robot Manipulation



Skills from human demonstration



Skills from robot controllers

Context / Related Work / Limitations of Prior Work

1 or more slides

Which other papers have tried to tackle this problem or a related problem?

- The paper's related work is a good start, but there may be others
- What is the key limitations of prior work(s)?

Proposed Approach / Algorithm / Method

1-5 slides

Describe algorithm or framework (pseudocode and flowcharts can help)

- What is the optimization objective?
- What are the core technical innovations of the algorithm/framework?

Implementation details should be left out here, but may be discussed later if its relevant for limitations / experiments

Theory (if relevant)

What are the assumptions made for the theory? Are these reasonable? Realistic?

If the theory build strongly on other prior theory / results, reference those and state them here.

Theory (if relevant, continued)

State main results formally

Give proof sketches

Refer students to the full proofs in paper

Experimental Setup

1-3 slides

Description of the experimental evaluation setting

- What is the domain(s), e.g., datasets, tasks, robot hardware setups?
- What are the baseline(s)?
- What scientific hypotheses are tested?

How did the authors evaluate the success of their approach?

- Clear description of the metrics that will be used

Experimental Results

>1 slide

Present the quantitative and qualitative results

Show figures / tables / plots / robot demos

Pinpoint the most interesting / significant results

Discussion of Results

Demonstrated their CST algorithm can be applied to increasingly complex domains and environments.

- Skill tree from human demonstration in pinball domain
- Skill tree from human demonstration
- Skill tree from robot demonstration (skills from closed-loop controllers)
    - Starting with no knowledge: 13 minutes per episode of interaction time

- CST “consistently extracted skills” – however, they did not present a metric to show this. What was the error rate? What was the success rate?

Critique / Limitations / Open Issues

Limitations: Assumptions:

- Assumes that all demonstrations have the same goal
- Assumes reward is available
- Assumes an abstraction library is available (not learned)

- Their CST algorithm is robust to parameters, except for the variance in the noise prior.
- Possible a skill graph may be formed, rather than a tree

Future Work

- **Utilize skill graphs – goal conditioned tasks**
- **Learn abstractions – are the abstractions Markovian?**
- **Reducing the simplifying assumptions**

Extended Readings

Stefan Schaal. *Learning from Demonstration*. NeurIPS 1996.

B. Argall, S. Chernova, M. Veloso, and B. Browning. *A survey of robot learning from demonstration*. Robotics and Autonomous Systems, 57:469–483, 2009.

George Konidaris and Andrew Barto. *Skill Discovery in Continuous Reinforcement Learning Domains using Skill Chaining*. NeurIPS 2009.

A. Bagaria, J. K. Senthil, G. Konidaris. *Skill Discovery for Exploration and Planning using Deep Skill Graphs*. ICML 2021.

George Konidaris and Andrew Barto. *Efficient skill learning using abstraction selection*. IJCAI 2009.

C. Allen, N. Parik, O. Gottesman, and G. Konidaris. *Learning Markov State Abstractions for Deep Reinforcement Learning*. NeurIPS 2021.

David Abel. *A Theory of Abstraction in Reinforcement Learning*. PhD Thesis 2020. Brown University.

G. Konidaris, L.P. Kaelbling, and T. Lozano-Perez. *From Skills to Symbols: Learning Symbolic Representations for Abstract High-Level Planning*. JAIR 2018.

Summary

- **Problem**: How to construct skill trees from demonstrations?
- **Importance**: Reusable skills. **Difficulty**: Lots of assumptions are made.
- **Limitations of Prior Work:** Policies from LfD lack composability.
- **Key Insights:** HMM of approximated value function predicts actual returns; used for changepoint detection.
- Demonstrated to work on increasingly complex domains, including mobile manipulation