Overcoming the Pitfalls of Prediction Error in Operator Learning for Bilevel Planning

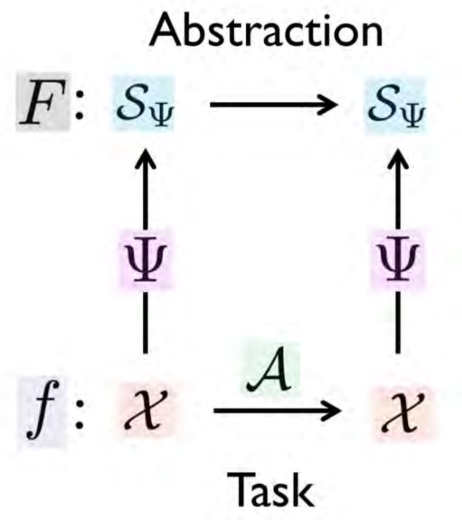
Nishanth Kumar*, Willie McClinton*, Tomás Lozano-Pérez, Leslie Pack Kaelbling
MIT Computer Science and Artificial Intelligence Laboratory
{njk, wbm3, tlp, lpk}@mit.edu

Learning Operators For Planning

Planning in robotics domains is hard
1. Continuous state spaces
2. Continuous action spaces
3. Long horizons

Abstractions can help
1. State abstractions
2. Action abstractions
3. Transition model abstractions

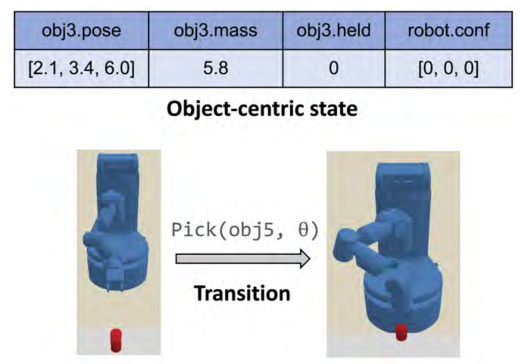
We focus on given 1. how do we learn 2. and 3.



Our objective: *Learn* operators that can be used for *effective and efficient planning* in robotics domains where optimizing prediction error fails.

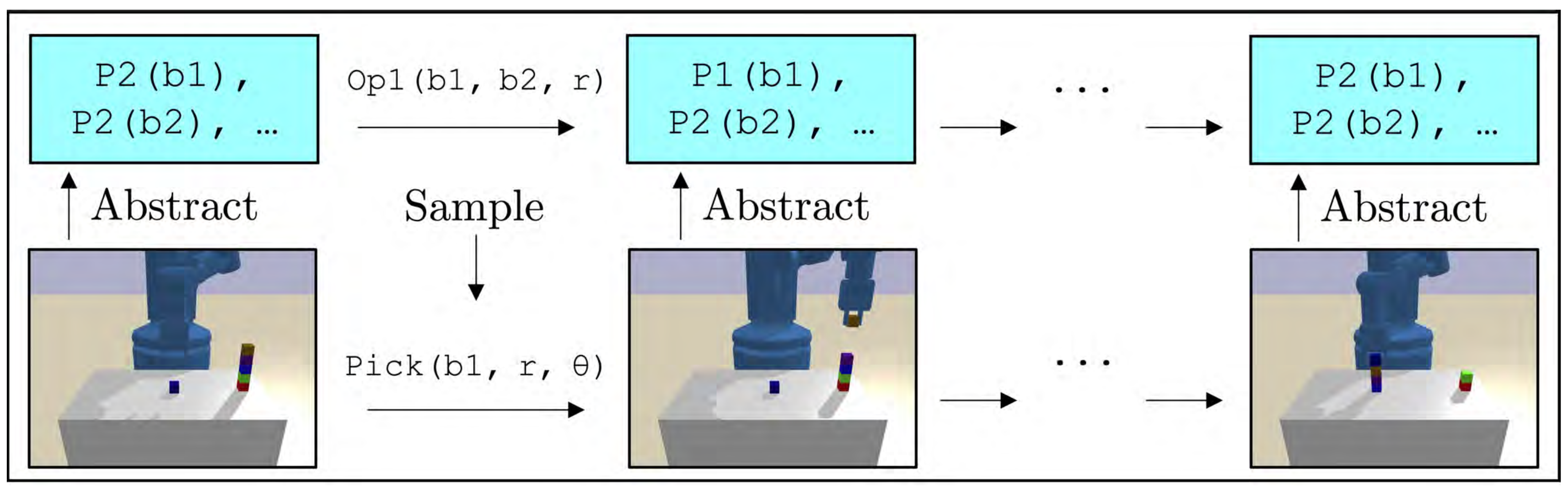
Problem Setting

- Deterministic & fully observed
- States are **object-centric**
  - Continuous feature vector per object (right)
- Actions are **parameterized controllers**
  - E.g., Pick(obj5, [2.3, 6.7, 5.8])
- Transition model is known
- Training tasks & demos, evaluation tasks
  - Evaluation tasks unknown during training
- Task goals use given **goal predicates**



Problem: The *"Prediction Error"* view in an attempt to predict the entire next state leads to complex operators and not well-aligned with our real objective. See panel below.

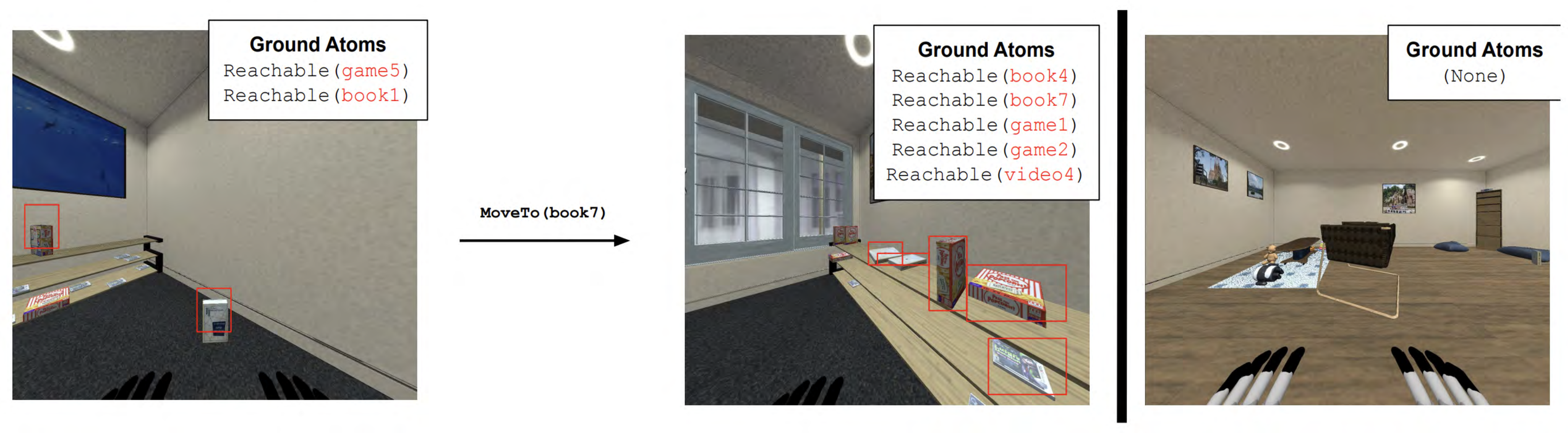
Bilevel Planning



- **Outer loop:** AI planning with learned symbolic operators
- **Inner loop:** backtracking search with learned neural samplers needed to handle *Non-Downward Refinability*

Key Terms:
- *Non-Downward Refinability* is a property of abstractions for planning where finding a high-level plan does not guarantee success on the first execution of low-level actions
- *Necessary Atoms* are a subset of the abstracted state's atoms which are the conditions that must hold true for an abstract plan suffix to legally achieve the goal.

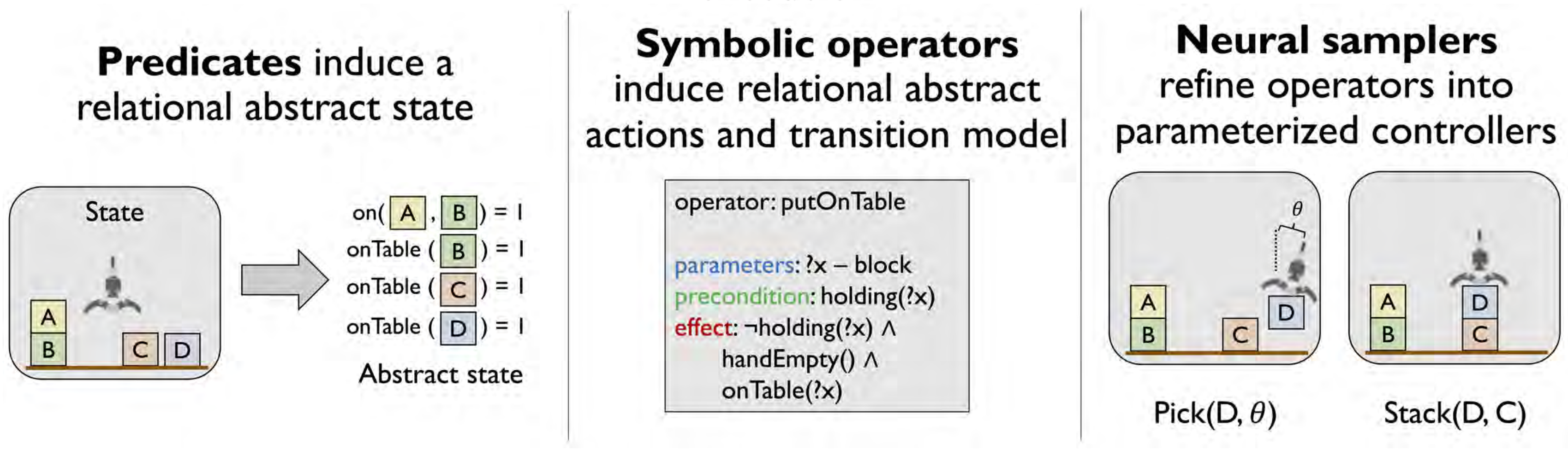
Learned Neuro-Symbolic Operators



Op–MoveToBook–Prediction–Error:
Args: ?objA ?objB ?objC ?objD ?objE ?objF ?objG
Preconditions: (**and** (Reachable ?objA) (Reachable ?objB))
Add Effects: (**and** (Reachable ?objC) (Reachable ?objD)(Reachable ?objE) (Reachable ?objF)(Reachable ?objG))
Delete Effects: (**and** (Reachable ?objA) (Reachable ?objB))

Op–MoveToBook–Necessary–Changes:
Args: ?objA
Preconditions: ()
Add Effects: (Reachable ?objA)
Delete Effects: $(\forall ?x.\ ?objA \neq ?x \Rightarrow$ (Reachable ?x))

Neuro-Symbolic Operators contain an operator (above) defining high-level logic used for planning, as well as a parametrized controller and a sampler for low-level execution



Operator Learning Objective

We learn operators directly optimized for efficient planning
1. Estimate the set of *necessary atoms* from our data given our current candidate operators
2. Learn new operators that 'cover' those *necessary atoms*
3. Iterate between 1-2 until a fix point is reached within our *hill-climbing search*

Specifically, we minimize the following objective:

$$J(\Omega) \triangleq (1 - \texttt{coverage}(\mathcal{D}, \Omega)) + \lambda \texttt{complexity}(\Omega) \quad (1)$$

Our objective is designed to prevent the overfitting that optimizing prediction error leads to. We want to measure coverage in terms of necessary atoms matching trajectories and not in terms of prediction error, as is typically done.

Now, we define a fine-grained notion of coverage for optimization via hill-climbing:

$$\texttt{coverage}(\Omega, \mathcal{D}) = \Sigma_{(\overline{x}, \overline{u}, g, \mathcal{O}) \in \mathcal{D}} \frac{\eta(\omega, (\overline{x}, \overline{u}, g, \mathcal{O}))}{|\overline{u}|} \quad (2)$$

Learning Operators via Hill-Climbing

PreimageBackchaining($(\Omega, (\overline{x}, \overline{u}), \mathcal{O}, g)$)
- $n \leftarrow$ length($\overline{x}$) ; $\alpha_n \leftarrow g$
- **for** $i \leftarrow n-1, n-2, \ldots, 0$ **do**
  - $s_i, s_{i+1} \leftarrow$ Abstract(x_i), Abstract(x_{i+1})
  - $\underline{\omega_{\text{best}}} \leftarrow$ FindBestConsistentOp(Ω, $s_i, s_{i+1}, \alpha_{i+1}, \mathcal{O}$)
  - **if** $\underline{\omega_{best}} = Null$ **then**
    - **break**
  - $\underline{\omega_{i+1}} \leftarrow \underline{\omega_{\text{best}}}$
  - $\alpha_i \leftarrow \underline{P_{i+1}} \cup (\alpha_{i+1} \setminus \underline{E^+_{i+1}})$
- **return** $(\underline{\omega_{i+1}}, \ldots, \underline{\omega_n}), (\alpha_i, \ldots, \alpha_n)$

Algorithm 1: Preimage backchaining procedure (details in (§C)).

HillClimbingSearch($\mathcal{D}$)
- $J_{\text{last}} \leftarrow \infty$; $J_{\text{curr}} \leftarrow$ J($\Omega, \mathcal{D}$)
- $\Omega \leftarrow \emptyset$
- **while** $J_{curr} < J_{last}$ **do**
  - $\Omega' \leftarrow$ ImproveCoverage($\Omega, \mathcal{D}$)
  - **if** J($\Omega', \mathcal{D}$) $< J_{curr}$ **then**
    - $\Omega \leftarrow \Omega'$; $J_{\text{curr}} \leftarrow$ J($\Omega, \mathcal{D}$)
  - $\Omega' \leftarrow$ ReduceComplexity($\Omega, \mathcal{D}$)
  - **if** J($\Omega', \mathcal{D}$) $< J_{curr}$ **then**
    - $\Omega \leftarrow \Omega'$; $J_{\text{curr}} \leftarrow$ J($\Omega, \mathcal{D}$)
  - $J_{\text{last}} \leftarrow J_{\text{curr}}$; $J_{\text{curr}} \leftarrow$ J($\Omega, \mathcal{D}$)
- **return** Ω

Algorithm 2: HillClimbingSearch learns operators that optimize objective J.

- *PreImageBackchaining (Algorithm 1)* employs a backchaining procedure to compute necessary atoms and plan suffixes, starting from the known atoms of the final timestep, and using a heuristic to select the best operator from multiple possibilities
- *Hill-Climbing Search (Algorithm 2)* optimizes our objective by using preimage backchaining to compute necessary atoms leverages

Experimental Results

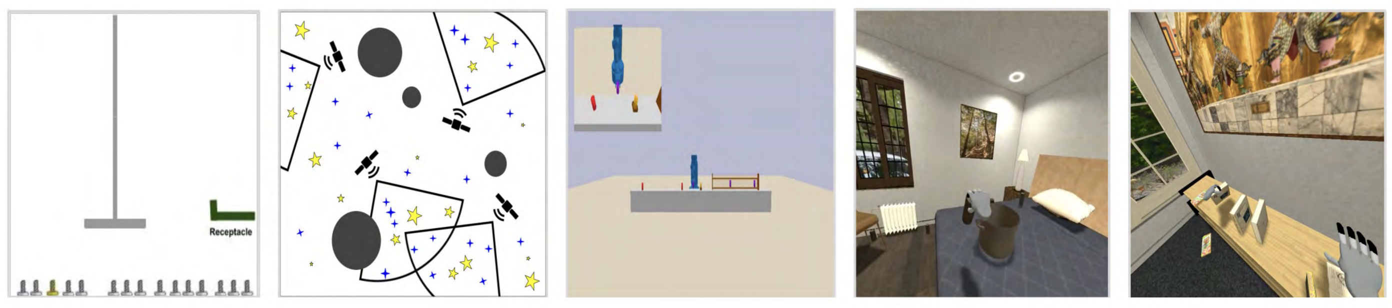


Figure 2: **Environments**. Visualizations for *Screws*, *Satellites*, *Painting*, *Collecting Cans*, and *Sorting Books*.

Key findings:
- Performs very well compared to several baselines in challenging robotics domains
- Learned abstractions that are robust to non-Downward Refinablity
- Little data is required to learn good abstractions

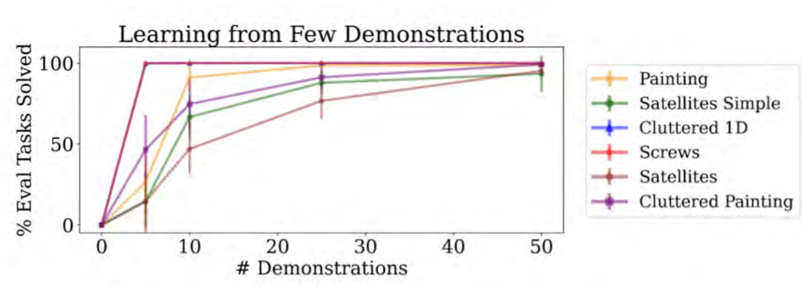


Figure 3: Data-efficiency of main approach.

Environment	Ours	LOFT	LOFT+Replay	CI	CI + QE	GNN Shoot	GNN MF
Painting	**98.80 (0.42)**	0.00 (0.00)	**98.20 (0.91)**	**99.00 (0.31)**	93.40 (1.47)	36.00 (3.39)	0.60 (0.29)
Satellites	**93.40 (3.52)**	0.00 (0.00)	34 (5.28)	**91.60 (2.68)**	**95.20 (1.30)**	40.40 (3.04)	11.00 (1.44)
Cluttered 1D	**100.00 (0.00)**	17.20 (5.46)	0.00 (0.00)	17.40 (5.52)	92.80 (0.90)	98.60 (0.63)	98.60 (0.63)
Screws	**100.00 (0.00)**	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	50.00 (15.81)	95.60 (3.05)	95.80 (3.07)
Cluttered Satellites	**95.20 (0.75)**	0.00 (0.00)	0.00 (0.00)	1.60 (0.61)	6.00 (1.57)	4.80 (1.27)	0.00 (0.00)
Cluttered Painting	**99.20 (0.42)**	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	4.60 (1.16)	0.00 (0.00)
Opening Presents	**100.00 (0.00)**	0.00 (0.00)	-	83.00 (10.77)	83.00 (10.77)	28.00 (5.96)	0.00 (0.00)
Locking Windows	**100.00 (0.00)**	0.00 (0.00)	-	90.00 (4.47)	88.00 (4.42)	0.00 (0.00)	0.00 (0.00)
Collecting Cans	**77.00 (11.75)**	0.00 (0.00)	-	0.00 (0.00)	1.00 (0.94)	0.00 (0.00)	0.00 (0.00)
Sorting Books	**69.00 (11.61)**	0.00 (0.00)	-	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)

Link to paper!

Acknowledgements: We gratefully acknowledge support from grants by NSF, AFOSR, ONR MURI, ARO, MIT-IBM Watson Lab, and the MIT Quest for Intelligence. Nishanth and Willie are supported by NSF Graduate Research Fellowships.