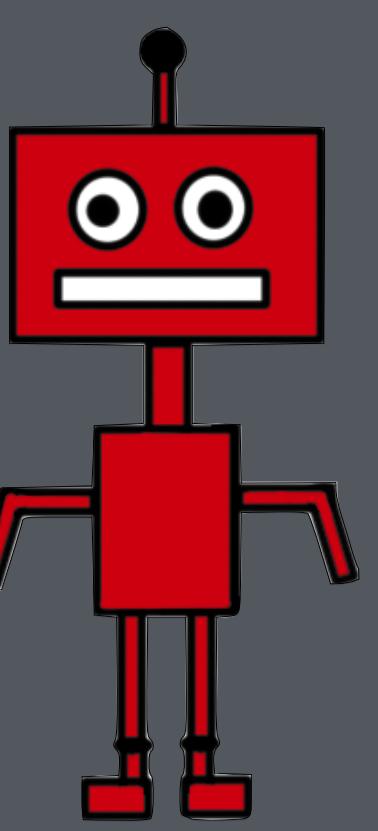


# Learning Dirichlet Priors for Affordance Aware Planning



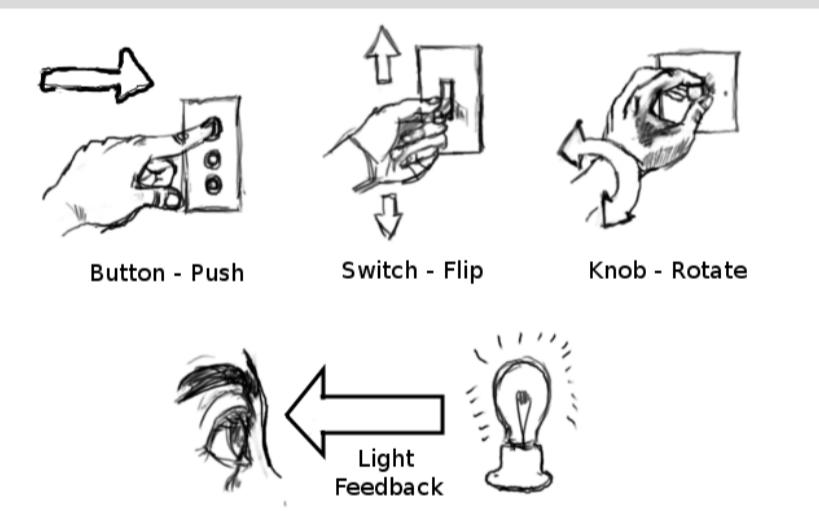
David Abel & Gabriel Barth-Maron, James MacGlashan, Stefanie Tellex  
Dept. of Computer Science, Brown University

## Goal

Enable autonomous agents to learn how to plan efficiently in massive stochastic state spaces.

## Background

**Affordances:** Direct agent toward relevant action possibilities.



"What [the environment] offers [an] animal, what [the environment] provides or furnishes, either for good or ill"  
- J.J. Gibson, 1977

Formalism:

$$\Delta = \langle p, g \rangle \mapsto \mathcal{A}'$$

$p$  = predicate on states  
 $g$  = lifted goal description  
 $\mathcal{A}'$  = subset of MDP Actions

**Domain:** Minecraft



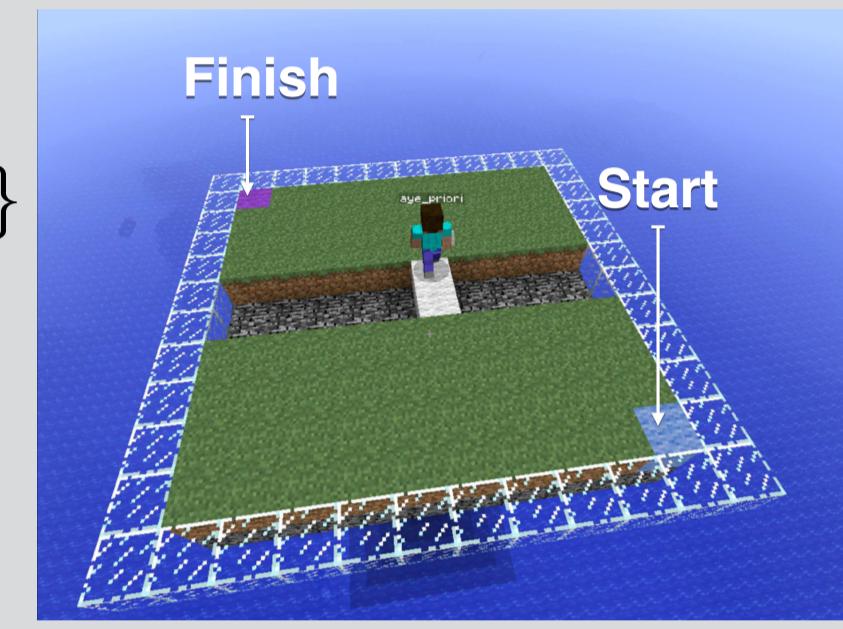
$\approx$  Turing Complete Legos

## Affordance Example

$$\Delta_1 = \langle \text{nearPlane}, \text{atLoc} \rangle \mapsto \{\text{move}\}$$

$$\Delta_2 = \langle \text{nearTrench}, \text{atLoc} \rangle \mapsto \{\text{place}\}$$

If  $\Delta$ 's predicate is true and  $\Delta$ 's goal type matches the current goal, use  $\Delta$ 's actions.

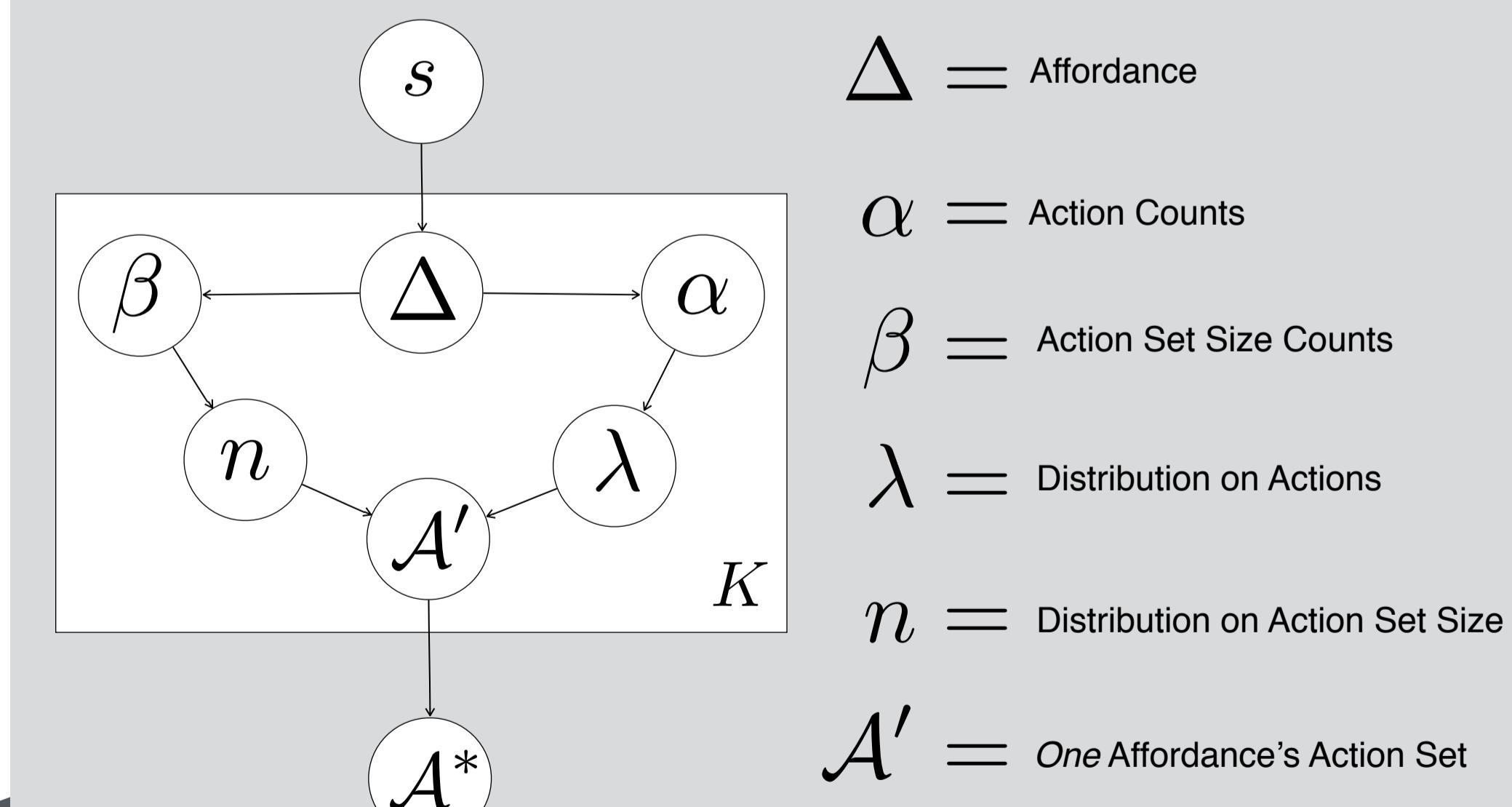


## Learning

**Goal:** For a given state, for each affordance, learn which actions are most relevant:

$$\Pr(\mathcal{A}^* \mid s, \Delta_1 \dots \Delta_K)$$

**Graphical Model:**



Where:

$$\Pr(\lambda \mid \alpha) = \text{DirMult}(\alpha)$$

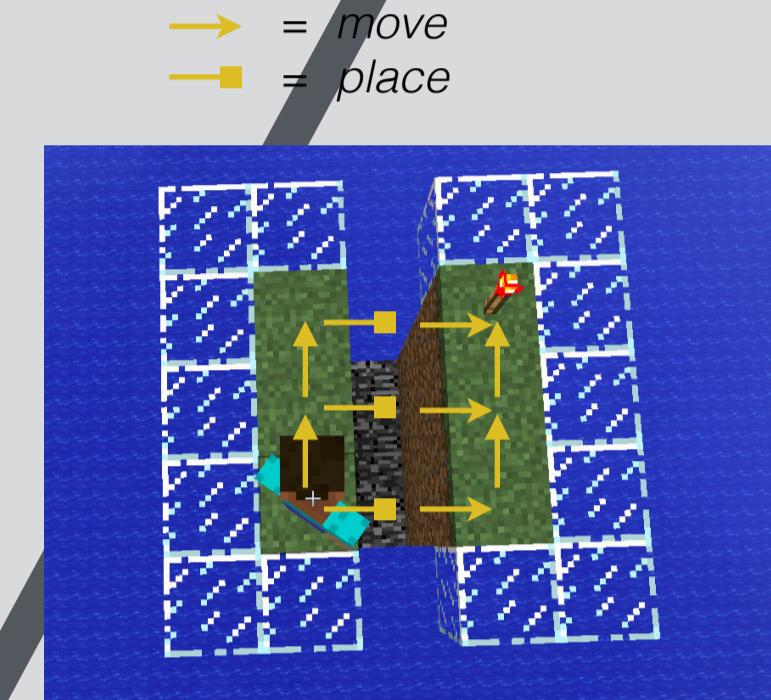
$$\Pr(n \mid \beta) = \text{Dir}(\beta)$$

## Learning Example

1) For each activated affordance, count:

$\alpha$  = number of worlds in which each action was used

$\beta$  = number of unique actions used in each world



$$\Delta_1 = \langle \checkmark \text{nearTrench}, \checkmark \text{atGoal} \rangle$$

$$\Delta_1.\alpha.\text{moveRight}++, \Delta_1.\alpha.\text{moveForward}++, \Delta_1.\alpha.\text{placeRight}++$$

$$\Delta_1.\beta.3++$$

2) When solving the MDP on a new state space, in each state  $s$ :

$$\mathcal{A}^* = \bigcup_{i=1}^K (\Delta_i.\text{getActions}(s))$$

3) Where

$$\Delta_i.\text{getActions}(s):$$

```

 $\lambda \leftarrow \text{DirMult}(\Delta_i.\alpha)$ 
 $n \leftarrow \text{Dir}(\Delta_i.\beta)$ 
 $\mathcal{A}' \leftarrow_n \lambda$ 
return:  $\mathcal{A}'$ 
  
```

## Results

Avg. # Bellman Updates Per Converged Policy

