Planning with Affordances

Abstract

1 INTRODUCTION

2 BACKGROUND

2.1 AFFORDANCES

An Affordance is defined as: $\mathcal{A} = <a, p, \mathcal{G}>$, where:

$$a \in A$$

$$p : s \rightarrow \{0, 1\}$$

$$\mathcal{G} : s \rightarrow \{0, 1\}$$

Where a is a *primitive action* in the agent's set of available actions, p is a *precondition* which is a predicate on a given state s, and $\mathcal{G}$ is a desired *postcondition*, which is also a predicate on a particular state s.

The constituents that make up an Affordance parallel those of the other planning approaches discussed in the background section.

2.2 SUBGOALS

Subgoal planning leverages the intuition that certain goals in planning domains may only be brought about if certain preconditions are first satisfied. For instance, in the Minecraft domain, one must be in possession of grain in order to bake bread. In Branavan et. al, they explore learning subgoals and applying them in order to plan through a variety of problems in Minecraft.

In subgoal planning, the agent is given a pair of predicates, $<x_k, x_l>$, where x_l is the effect of some action sequence performed on a state in which x_k is true. Thus, subgoal planning entails high-level planning in

subgoal space, and low-level planning to get from subgoal to subgoal.

However, subgoal planning falls short in that sub-goals only go so far in direction the low-level planner. The planner will still waste many cycles searching through meaningless portions of the state-space.

2.3 OPTIONS

The options framework proposes incorporating high-level policies to accomplish specific sub tasks. For instance, when an agent is near a door, the agent can engage the 'door-opening-option-policy', which switches from the standard high-level planner, to running a policy that is hand crafted to open doors. An option o is defined as follows:

$$o = <\pi_0, I_0, \beta_0>, \text{ where:}$$

$$\pi_0 : (s, a) \rightarrow [0, 1]$$

$$I_0 : s \rightarrow \{0, 1\}$$

$$\beta_0 : s \rightarrow [0, 1]$$

Here, π_0 represents the *option policy*, I_0 represents a precondition, under which the option policy may initiate, and β_0 represent the post condition, which determines which states terminate the execution of the option policy.

As Konidaris and Barto point out, the classic options framework is not generalizable, as do not enable an agent to transfer knowledge from one state space to another. In Konidaris and Barto's recent paper, they expand on the classic options framework and allow for a more portable implementation of options. However, planning with options requires either that we plan in a mixed space of actions *and* options, or requires that we plan entirely in the space of options. Additionally, options requires that we provide the agent with option

policies, something which is not easily crafted by a human designer (especially if we want an optimal policy, which we do).

The Affordance formalism introduced above and expanded on in this paper, resolves the weaknesses of these other frameworks by limiting the complexity of the seed knowledge required of the designer, and providing enough knowledge of the right kind to limit the search space, and still maintain scalability. (assuming experiments back this up)

3   EXPERIMENTS

4   RESULTS

5   CONCLUSION