Representing and Reasoning With Complex Affordances

Maija Filipovica

University of Birmingham

Abstract

The concept of affordances has been used in robotics research to endow artificial agents with the ability to reason about available actions and uses of objects. Previous research has demonstrated that a representation of affordances improves performance when planning and reasoning about available actions, and has introduced a distinction between primitive and complex affordances. Complex affordances refer to actions the execution of which depends on several different abilities, or aspects of object usage. Problem solving research in behavioural biology and psychology provides some evidence that flexible tool use and creative problem solving require a type of abductive reasoning that can be supported by a representation of complex affordances, following the definition described in previous works. The present study aimed to build on findings demonstrating the utility of primitive affordances, by developing and implementing a representation of complex affordances that capture the prerequisite conditions for complex actions which involve multiple intermediate steps. Complex affordances are encoded and tested in Answer Set Prolog (ASP), a declarative programming paradigm that supports non-monotonic logical reasoning for planning and diagnostics. The utility of complex affordances was evaluated by running paired trials, where planning was carried out with complete domain knowledge and missing domain axioms relating to complex affordances. Use of complex affordances led to the production of minimal plans, and resulted in a larger number of plans found for a given goal. Domain axioms involving complex affordances led to a higher overall success rate, while exclusion of these axioms led to the production of invalid plans. The present study therefore demonstrated a successful implementation of complex affordances, and their utility. Finally, areas of future work were explored, including the applicability of complex affordances in the creation of complex plans, problem solving, and plan recognition.

1. Introduction

1.1. Background and Motivation

In order to provide useful assistance and cooperate effectively with humans, artificial agents (e.g. robots) will need to be capable of learning the complex dynamics of real world domains. Furthermore, it will be crucial for them to reason efficiently about a given situation and effectively utilize the acquired knowledge in their respective domains. This is especially true in settings where limited information is available, and a decision needs to be made without additional observations due to time constraints, or the low likelihood of obtaining a particular observation. One area this applies to is reasoning about actions and action capabilities. For example, in an assistive setting, a robot may need to infer human action capacities without access to additional empirical observations of the person carrying out a

particular action (e.g. if they are not capable of this action due to an injury). Humans are capable of such inference, as exemplified by their ability to infer motor capabilities of other humans by observing their movement [1, 2], therefore missing such a capacity might significantly impair human-robot interaction (HRI) [3]. This implies the need for extrapolating existing knowledge about the action capabilities of agents to new situations.

A similar need for analytic reasoning is true for flexible tool use and complex problem solving. Research in psychology has long related the inability to perceive an action capability i.e. the concept of functional fixedness [4], to impaired problem solving. For example, learning the typical function of an object makes individuals less likely to perceive other actions afforded by an object, and subsequently utilize the object in a given task [4, 5].

One way of counteracting the problem of functional fixedness on a robot or an agent could be through utilizing the concept of complex affordances. Primitive affordances describe situations (e.g. presence of a particular object, domain or object properties) in which an action is available given an agent's characteristics (such as an agent's strength, or the type of their actuators) [6]. Complex affordances denote different aspects of object usage and functionality by combining primitive affordances, and provide a way to reason about action capabilities analytically, once the primitive affordances have been learned through empirical observation. Complex affordances also provide a way to relate an agent's capabilities and properties to different actions and contexts: for example, if an agent is able to grasp handles to pick up objects, they might also be able to grasp handles of doors and windows, and therefore open them.

1.2. Related Literature

Affordances are a useful concept in robotics and artificial intelligence (AI) as they provide a way to reason about agent and object properties conceptually. This enables flexible reasoning, and allows a semantic mapping from perceived or represented physical attributes to their use, that is not restricted to their appearance [3, 7]. Due to this, research on affordances in robotics and AI is abundant, and the concept has been implemented in a number of different architectures [3, 8].

In humans, direct perception of affordances has been linked to problem solving in numerous studies. For example, if an actor has learned the affordances supporting the typical function for which an object has been created, they are less likely to identify alternative uses for the object [4, 5]. This impacts whether a solution for a given problem is found, and the speed at which this solution is arrived at. Duncker[4] coined the term functional fixedness, which refers to a strong perceptual

focus on a particular functionality an object is associated with. This prevents the object from being used in situations where it would appropriately solve a problem. As a result, his, as well as later research in psychology have viewed functional fixedness as a hindrance for creative problem solving.

Affordances, originally defined by Gibson [9], but also in their other permutations described in later research [3, 10, 11], are a diametrically opposed concept, in that the uses of an object are directly perceived. Thus it stands to reason that use and representation of affordances in cognitive architectures is necessary for creative problem solving and tool use. It is important to note that human participants in the experiments mentioned above were capable of inferring that the objects could be used in other ways, based on these simple concepts, without any empirical trials. This is true despite their perceptive bias towards the functionality of the object due to previous experience [5]. This implies that extrapolating, and applying knowledge to a new setting requires abductive reasoning, whereas most of the previous research in robotics has mainly focused on simpler variations of affordances through experience [8]. Some research exists on using more sophisticated affordances in robotics systems, for example Jamone et al. [7] model affordances of intermediate objects that are involved in the execution of a composite action, however, in their implementation affordances are modeled in separate categories for different aspects of existing tools. While ability to reason about tool usage is valuable, their model of affordances does not explicitly incorporate, or attend to the properties which give tools a particular functionality. This may impede on flexible tool use, as some properties of objects may be used for different purposes, depending on the context (e.g. a broom may be used for cleaning, but the handle of a broom may also be used to poke or move objects that are out of reach). Other research in AI alludes to affordances as being necessary for creative problem solving [12]. In Olteteanu [12] the importance of affordances is emphasized in situations where agents are faced with ill-structured problems. They also postulate that problems requiring creative problem solving can be viewed at two resolutions, representing low level motor actions, and higher level concepts.

In comparison, the concept of affordances in psychology and cognitive biology has been applied in the context of more complex actions, at times describing highly abstract capabilities to solve life-tasks involving a number of social and practical factors [13]. Interestingly, in cognitive biology, affordances which would correspond conceptually to the primitive affordances in the definition described by Langley, Sridharan and Meadows [6] followed here, can be seen to be learned associatively, and the individuals may not attend to, or reason about the functional properties of objects enabling these affordances [14, 15, 16]. In contrast, human toddlers

attend specifically to the functional aspects of objects, and are able to quickly transfer their knowledge of object uses, and generalize to different tasks [17]. These two findings can be seen as conceptually parallel to the definition of complex and primitive affordances proposed in Langley, Sridharan and Meadows [6], where they postulate that primitive affordances can be learned empirically through trial and error, whereas complex affordances are inferred analytically, combining multiple known aspects of the domain and extending it to a particular situation.

Previous research has shown that discovery of primitive affordances improves plan quality and that these affordances provide a way to simplify knowledge inference and reasoning [18, 19]. Missing knowledge, i.e. missing axioms relating to affordances, leads to the failure of finding plans, and the selection of invalid actions [18, 20]. However, the implementation and utility of complex affordances has not yet been investigated. Following these previous findings, the aim of the present research is to first implement complex affordances, and evaluate their utility in plan generation.

Complex affordances should model how the ability to do one action determines the capacity for another action, e.g. if an agent has flexible legs, they may squat to lift a cumbersome object, but this also enables them to use stairs, and step over high ledges. However, complex affordances can also inform when an action is possible, if it consists of several intermediate actions, e.g. the ability to grasp a handle may be necessary to lift an object that is otherwise unwieldy to lift.

Both categories described above are modelled as complex affordances in the present study. The utility of such a representation was tested by comparing plan generation for random goals for a program which had full knowledge of such axioms, and a program that was missing this information. This enabled an investigation of how the representation of complex affordances influenced the length and quality of discovered plans.

2. Methods

The utility of complex affordances was evaluated by running paired trials simulating agents that reason with complete domain knowledge and partial domain knowledge, where complete domain knowledge (CDK) consists of the full domain description. Conversely, partial domain knowledge (PDK) consists of a domain description where axioms encoding complex affordance relations and their corresponding executability conditions are omitted. This was implemented in Answer Set Prolog (ASP), allowing for non-monotonic logical reasoning and search for minimal plans [21]. The domain in this implementation was a variation of the Blocks World (detailed description below). Observational uncertainty is abstracted away, enabling the isolation of the impact for each added complex affordance, or its corresponding executability conditions.

2.1. Domain Description

The domain in the present implementation is a variation on the blocks world domain, and contains elements of setups used in problem-solving research within cognitive biology [22] and ecological psychology [23]. The agent is situated in a domain consisting of several enclosed areas, such as rooms and corridors, and has access to different objects. The agent's objective may be to stack objects using its arms, or to change its own location. Both objectives may also require the agent to move itself or objects to a different area, such as a room, which may be accessible through an exit. Exits connecting adjacent areas may be located above, out of the agent's reach, requiring them to create a suitable configuration of objects in order to reach the opening, as well as step through the exit on the other side.

The agent's ability to pick up objects, and travel to different surfaces depends on several properties. These mostly concern the objects location along the vertical axis in the simulated space, and the characteristics of the agent's limbs. Agents legs and arms may possess different levels of strength and different capacities for movement, characterized by joint mobility. For example, agents with flexible arms and legs are able to pick up objects that are located low compared to the agent i.e. on, or just below, the surface the agent is standing on. These properties further interact and are influenced by the object's weight, and the agent's strength.

Agents with flexible legs can travel to surfaces within a specified range of the surface they are located on. This same capacity also determines the broader set of situations where it is possible for an agent to go through an opening to another area. For example, agents with better leg mobility and strength are also able to use an exit to change their location in situations where the exit is not aligned with

the agents legs or actuators, if the lower threshold of the exit is within their leg movement range, and if a surface on the other side of the exit is also within their leg movement range. To give an example of how this may translate to a real-world scenario, an agent or a person with good leg mobility may be able to go to another room via a connecting window at the height of their waist. Conversely, a wheeled robot may not be able to travel to an adjacent area through a door that has a very high threshold and a person with a leg injury may not be able to use the stairs. As such, complex affordances in this domain capture the following information:

- Heavy objects just above the agent cannot be picked up unless the agents arms are strong and flexible, (i.e. they can reach and remain stable when lifting arms above their head)
- Heavy objects just below the agent cannot be picked up unless the agent can squat, and remain stable while lowering their arms (the latter is captured by the previous affordance relation concerning heavy objects above the agent).

These examples each have two complex affordance relations for different agent ability and object weight combinations. In this case the complex affordance relation combines primitive relations that refer to the same action, i.e. the same characteristics that enable the agent to pick up heavy objects placed higher than the agent are also one of the required characteristics for picking up heavy objects placed at the level of the agent's base. Complex affordance relations can also be modelled by combining relations for several different actions. The following is an example in the present domain:

- Agents cannot travel to surfaces at a different height, unless the agent's legs are flexible, and the surfaces are within the specified movement range
- If the surface the agent is on, and the surface on the other side of the exit is not aligned with its lower threshold, the agent cannot use the exit, unless the agent also has sufficient leg mobility to travel to surfaces at a different height.

That is, an agent's capacity to move to surfaces at a different height is a prerequisite for the action of going through an opening to another area, if the lower threshold of this opening is not in line with the surface the agent is on. The 2d location, and intermediate actions in this domain are abstracted away - as an example, for the affordance relation described above, no intermediate 'squat' action is described in the domain dynamics.

2.2. Encoding of domain

The dynamics of the domain are first captured in a system description $\mathcal{SD}$, consisting of a set of action language $\mathcal{AL}_d$ statements [24]. The action language $\mathcal{AL}_d$ allows a formal description of the state action transitions of a domain using causal laws, state constraints and executability conditions [25] following the general form:

1. a **causes** l **if** p
2. l **if** $p_0 ... p_n$
3. **Impossible** a **if** $p_0 ... p_n$

Where a is an action, l is a domain literal (any domain property or its negation), and p is a domain property.

System descriptions in $\mathcal{AL}_d$ have a sorted signature consisting of *statics*, $fluents$ and *actions*. *Statics* are domain elements with unchanging truth values. Fluents represent domain elements with truth values that are dynamic: *inertial* fluents change their value as a direct consequence of actions, whereas the values of *defined* fluents are determined by some aspects of the current state. Different entities and domain attributes are described by *sorts*; the system description $\mathcal{SD}$ contains a sorted signature. Sorts of the domain include *limb*, *agent*, *object*, *area*, *weight*, *material*. The signature also contains the ground terms of these sorts, e.g. *room* of sort *area*, *arm* and *leg* of sort *limb*, *door* of sort *exit*. Sorts *vertsz*, and *step* are numeric, and are used to represent height and time step accordingly. Static properties such as agents and objects height and weight, and object material, as well as agents limb strength are expressed with predicates. Fluents of the domain include *z_loc*, *location*, and *in_hand*; *z_loc* specifies the location of an entity along the vertical/ Z axis, at a particular time-step, in integers. *Location* specifies the area an agent or an object is in at a particular time-step. Defined fluents include *in_range*, specifying the vertical distance from the base of two entities. Actions include *pick_up*, *put_down*, *go_to* and *go_through*. Fluents, actions, and relations are described by the sorts of their arguments.

The domain representation also includes a history $\mathcal{H}$, which retains the truth values of fluents observed at a particular time-step, and the occurrences of actions: this information is used for diagnostic inference.

2.3. Representation of Complex Affordances

Affordance relations specify actions that are available for a particular agent, given the agent and object (if applicable) properties. As multiple affordance relations can apply to the same action, affordance relations are also specified by an

identifying index ID. As some of the properties which shape affordances in the present domain vary over time (such as z_loc), affordance relations also specify the time at which they enable an action. Following previous research in this area [18], enabling affordance relations take the following general form:

$affordance\_permits(A, ID, I)$ **if**...

Impossible A **if**..., *not* $affordance\_permits(A, ID, I)$

where A is an action, ID is the affordance relation Identifier, and *not* is a default negation operator indicating atoms for which truth values are not known, i.e. the above rules state that action A is normally impossible, unless the complex affordance relation ID is known to be true at time I [18]. The first axiom specifies domain properties which need to apply at a given time-step in order for the affordance relation to be true, and the second rule is an executability condition stating the conditions in which an action is only possible if it is enabled by the affordance relation.

Enabling affordances specify situations where an action, which normally would be considered impossible, is possible in the context of a specific agent and an object, whereas forbidding affordances specify situations where an action is impossible given particular agent and object properties [18, 20]. Primitive affordances in the domain are represented using both enabling and forbidding affordances.

Complex affordances can denote different aspects of object and tool usability and model situations where the execution of a more complex action depends on the agents ability to perform the individual steps [6]. Another benefit of using complex affordances to describe action capabilites is that the individual affordance relations can be used to encode other actions depending on the same characteristics. Complex affordances in the present implementation are represented by combining several enabling affordances. Following the two examples from the previous section, a complex affordance combining several aspects of the same action can be modelled as follows:

$$affordance\_permits(pick\_up(A, O), 14) \textbf{ if } joint\_mobility(A, arm, good), \\ limb\_strength(A, arm, good).$$

$$affordance\_permits(pick\_up(A, O), 16) \textbf{ if } affordance\_permits(pick\_up(A, O), 14), \\ limb\_strength(A, leg, good), \\ joint\_mobility(A, leg, good).$$

This also allows abstracting some intermediate actions, the consequences of which are not necessary to model for a particular application. For example, the agent being able to bend its legs and squat could be encoded by a squat action through adding the corresponding causal laws to represent this action in the system description, however, this is not required for the purposes of the present investigation.

2.4. Implementation

The above $\mathcal{SD}$ in $\mathcal{AL}$, containing the description of actions and state transitions, affordance relations, and history $\mathcal{H}$, is translated into an ASP program Π, for planning and inference. ASP supports non-monotonic logical reasoning, and default reasoning using consistency restoring rules. ASP allows for minimal planning and diagnostics, through the use of consistency restoring (CR) rules [21]. More specifically, while a goal (i.e. a domain literal that is not known to be true) has not been achieved, it causes an inconsistency, and a CR rule permits the agent to select an action in order to resolve the inconsistency. In the case of diagnostics, observations which do not match the expected outcomes of actions lead to an inconsistency, and a CR rule permits the agent to consider the failure of actions [26, 27]. Plans and explanations correspondingly are found in the answer sets obtained by solving Π. Π also contains an inertia axiom which specifies that inertial fluents retain their truth values overtime, unless specifically changed by an action, and a closed world assumption (CWA) for actions, defined fluents, and affordance relations. By removing an axiom from the domain knowledge, a planning agent may consider actions that are impossible leading to the production of incorrect plans. Upon the execution of such plans, some actions may fail, resulting in a prediction error (i.e. inconsistency between the expected and received sensor observation). Alternatively, exclusion of knowledge may prevent the agent from considering a legal action, resulting in longer plans.

2.5. Experimental setup

Experiment 1: The objective of the first experiment was to test whether plans generated when reasoning with partial domain knowledge were minimal. Here, partial domain knowledge refers to the deletion of domain axioms. In this experiment, the axioms related to the implementation were either complex affordance relations or an executability condition containing a complex affordance in its body. The deletion of these two types of axioms has different effects on the outcome. Missing affordance relations prevent the agent from finding plans that are possible to execute, as it is not known that an enabling affordance applies in

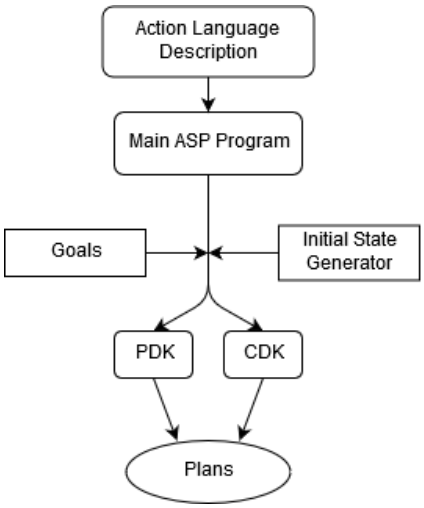

Figure 1: Overall structure of the implementation. Action Language description is used to produce the main ASP program. Initial states and goals are generated randomly, and added to the complete and partial knowledge programs, which produce the resulting plans.

a given state. This forces the agent to perform unnecessary actions in order to reach a particular state, resulting in longer plans, or a failure to find the solution. Deleting an executability condition containing an enabling affordance should result in the agent finding plans which are incorrect, as the agent is not informed that a particular transition is impossible. This may also lead to finding shorter plans.

Due to their different effects, experiment 1 compared performance when reasoning with omission of an increasing number of complex affordance relations, and the omission of executability conditions separately. The two experimental groups forming the results were therefore complete domain knowledge (CDK) and partial domain knowledge (PDK). The domain contained a total of six complex affordance relations, informing 11 executability conditions, and the omission of 1-6 axioms in each category was tested. 1000 paired trials were run at each level of deletion, where goals, initial conditions, and the omitted axioms were randomized for each trial. The plans were searched for using ASP inference with CR rules, therefore the plans returned were minimal. The number of returned plans, sequences of actions they contain, and their length were recorded.

Experiment 2: The second experiment tested whether the plans produced by reasoning with CDK and PDK in the first experiment were correct. Each plan was simulated by executing its constituent actions in an ASP program containing complete knowledge of the modelled domain dynamics. The action consequences were added to history $\mathcal{H}$ as observations. In cases where the action has failed, this produced an inconsistency and invokes the diagnostics module, which identified the failed action leading to this discrepancy. Information about failed actions was recorded, and the true success rate of an agent operating with partial domain

knowledge was compared to plan failure rates when reasoning with complete domain knowledge. In addition, for plans containing failed actions, the first time at which a failed action occurs was recorded.

3. Results

3.1. Experiment 1

Experiment 1 compared differences in plan length when reasoning with complete (CDK) and partial (PDK) domain knowledge. The average plan length increased per each omitted affordance relation. In order to assess whether the distributions of plan length were statistically significantly different, a paired two-tailed t-test between the plans produced in CDK and PDK conditions was performed at each level of axiom deletion. Although the average plan length was different, this difference was not found to be statistically significantly different when reasoning without 1-2 axioms. However, the plan length when reasoning with 3-6 missing complex affordance relations was found to be significantly higher, providing evidence that missing complex affordance relations prevented finding minimal plans. These results are summarized in Table 1 and can be seen visually in Figure 2.

N deleted axioms	**1**	**2**	**3**	**4**	**5**	**6**
t-value	0.69	1.03	1.97	2.82	3.74	10.61
mean difference	0.02	0.08	0.10*	0.13**	0.18***	0.16***

Table 1: Summary of t-values from paired t-tests for PDK and CDK differences in plan length with number of deleted affordance relations. Significance levels: $^{*} \leq 0.05, ^{**} \leq 0.01, ^{***} \leq 0.001$

Changes in plan length depending on the number of deleted affordance relations are displayed in Figure 2. The values are scaled relative to the maximum plan length returned in the experiments, therefore the values presented here are relative to 1. When collapsing across the number of deleted axioms, the overall average plan length was 0.30 for CDK and 0.41 for PDK. Average plan length for PDK increased by 15% with each additional deleted affordance relation.

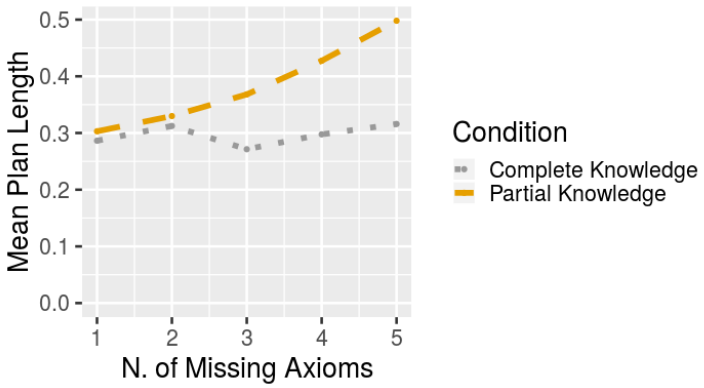

Figure 2: Changes in average plan length in conditions readoning with CDK, and PDK with the omission of complex affordance relations.

When executability conditions were omitted in PDK, the plans found with PDK were shorter. This result confirms the action constraints imposed by the

executability conditions of complex affordance relations.

In summary, experiment one has provided evidence for an increase in average plan length as the number of missing axioms increases in the partial domain knowledge. Furthermore, with more than two missing complex affordance relations, this increase became statistically significantly different from the complete domain knowledge condition.

3.2. Experiment 2

Experiment 2 further tested whether the plans returned when reasoning with PDK and CDK were correct. In the present analysis, this was quantified by two factors: the ratio of success and the ratio of correct plans. Ratio of success represents the fraction of trials in which a plan was found for a given goal, whereas ratio of correct plans shows the true fraction of successful plans, that do not contain an impossible action which would cause the plan to fail.

Overall, the results from experiment 2 show that inclusion of complex affordances enabled finding plans in a large fraction of trials, that would otherwise be falsely considered as impossible. The results of experiment 2 show that overall success rates when reasoning with CDK were 2.5 times higher than PDK, when averaging across different numbers of ommited affordance relations. The success rate for PDK decreased by 21 % on average with each deleted axiom. Figure 3 illustrates this change by showing the success rate for CDK and PDK with each deleted complex affordance relation.

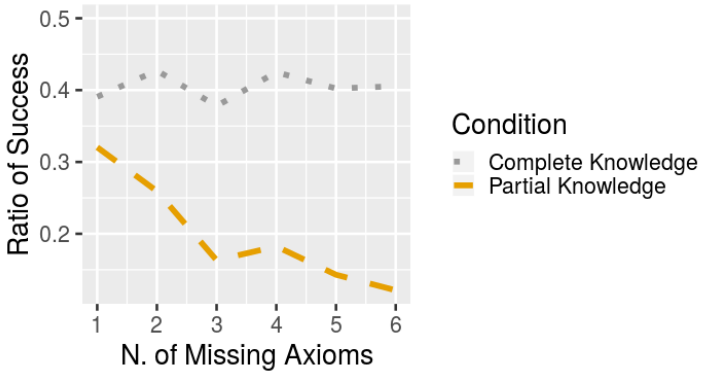

Figure 3: Fraction of successful trials when reasoning with CDK and PDK for different numbers of omitted affordance relations.

In addition to higher success rates, the addition of complex affordances enabled finding more plans to achieve a given goal. In trials where plans were found, the average number of plans found was greater when reasoning with CDK. These results are depicted in Figure 4. On average, CDK returned 5.91 plans, 2.93 times more than PDK condition, in which 2.16 plans were found per successful trial.

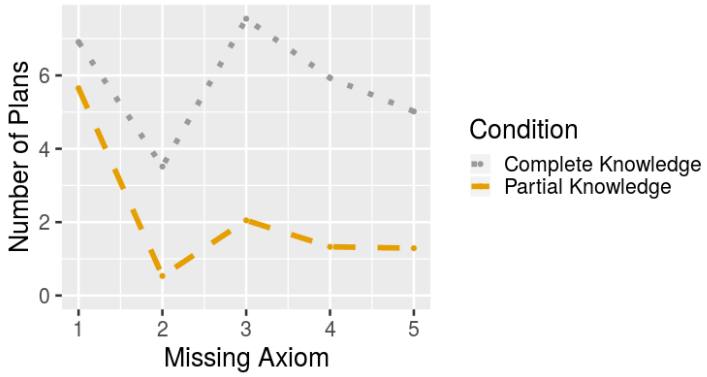

Figure 4: Changes in number of plans generated per trial with increasing numbers of missing complex affordance relations.

This implies that for any given goal, a larger set of alternative plans is returned when information about complex affordances in the given domain was available.

With the omission of complex affordance relations, all plans returned by reasoning with both PDK and CDK were correct. This in in agreement with the results stated above, as missing complex affordances prevents the agent form considering some actions to achieve a particular goal.

With missing executability conditions, all plans generated with CDK were correct, withstanding the simulation test. Conversely, each omitted executability condition resulted in a decrease in correct plans found with PDK: the fraction of correct plans decreased from 0.97 with 1 missing executability condition, to 0.30 with 6 missing executability conditions (Figure 5). On average across all six levels of deletion, 38% of the plans generated by PDK were incorrect. Each omitted executability condition resulted in a 20% increase of incorrect plans.

Figure 5: Ratio of correct clans out of all plans generated with increasing numbers of missing executability conditions 1-6.

Executability conditions referring to complex affordances also affected the speed at which the set of incorrect plans found in the PDK condition failed. When fewer executability conditions were excluded from the domain knowledge, impos-

sible actions which led to plan failure occured later in the plan. These results are summarised in Table 2. On average, the first time-step at which an action was detected to fail was 2.42, when reasoning with 1 missing executability condition. This decreased to the first action failure to be observed at 1.28 time-steps when reasoning with 6 missing executability conditions.

N deleted axioms	**1**	**2**	**3**	**4**	**5**	**6**
Time to First Failure	2.42	2.60	1.87	1.38	1.62	1.28

Table 2: Average Time to First Failure for Increasing Numbers of Deleted Executability Conditions

4. Discussion

Overall, the results in the experiments conducted here have shown that adding a representation of complex affordances improves plan quality, and can increase the proportion of goals for which an agent with given characteristics is able to find a plan. Depending on the type of missing axiom, a domain representation lacking complex affordances and their respective executability conditions led to the production of incorrect, non-minimal plans.

These findings are in agreement with previous research, building on findings that demonstrate the utility of affordances in both finding newly enabled plans and excluding unsuitable plans. This demonstrates that in a given domain, a large fraction of the possible state transitions can be captured with relatively few statements. Complex affordance relations inform the agent about what actions are possible based on a complex interplay of agent characteristics, object attributes and configurations that would be complicated to represent with separate executability conditions.

Based on previous research in robotics [18, 20, 12, 6], psychology [1, 13, 17], and cognitive biology [28], such a representation can be used to improve planning, plan recognition, HRI, and generalizing knowledge to novel tasks, allowing creative problem solving. In addition, this type of representation can allow to temporarily abstract intermediate actions without consequences on selecting valid actions, simplifying inference and planning to achieve goals involving intermediate steps. This type of zooming out could help constructing complex plans that extend over longer time periods, while retaining enough detail in the domain description to also determine the intermediate steps and actions. Previous research has coupled ASP inference with the execution of lower level motor commands, operating at different, but tightly coupled levels of refinement [29], where low level motor actions are reasoned with probabilistically, and higher level actions are reasoned about with non-monotonic logical inference. If applied in such architectures, complex affordances may enable agents to reason about more abstract actions at a coarse resolution. With sufficiently complex domain representations, this may enable a more general form of problem solving and long-term plan creation similar to the highly abstract affordance concepts in human situational problem solving [13]. In an assistive setting an agent may take into account the importance of a person's belongings, for example their house keys, a computer, and a cello, and reason about their role in achieving immediate and long-term goals through affording highly abstract actions occurring on different time-scales such as going on an errand, or earning livelihood. Such concepts may also enable a more natural HRI, where the robot is able to understand the concept of long-term goals, or is able to explain an

event or action at the appropriate level of detail.

In a similar fashion, complex affordances may be also be used to increase the autonomy of a robotic system through capturing knowledge about what long-term behavioural strategies or goals may be valuable for a particular agent. Another way such a representation can improve HRI is through plan recognition [6]. Implementations of plan recognition using Natural Language Processing and Combinatory Categorical grammars may lend itself to an extension which incorporates complex affordances, as these methods decompose goals into sub-goals, and individual actions, therefore representing plans at different resolutions. Complex affordances may disambiguate the composition of sub-tasks in the grammars used to generate plan libraries [30].

All of the above-mentioned applications may only be successful if complex affordances can be learned, or inferred. Immediate follow up research should address this by implementing or adapting an existing architecture for learning complex affordances.

A possible drawback of the present implementation, is that the intermediate actions were abstracted away and did not have a direct causal law informing the agent of their effects. For example, the actions could be subdivided into bending legs and squatting, which results in lowering the agent and changing the range of action available for its arms. However, as stated above, previous research exists where low level motor commands have been abstracted away, allowing agents to reason about the higher level sequences of actions they need to take. In principle, it is possible to represent the causal structure (and the corresponding affordances) of the intermediate actions, however the present implementation attempts to simplify action modeling and focuses on testing the implementation of complex affordances. Testing of complex affordances in the domain developed in the present study did not require the information about some of the intermediate actions represented in the domain description. Nonetheless, follow up research in this area should endeavor to develop a more sophisticated representation, if this enables the production of useful complex affordances.

5. Conclusion

The present study implemented a representation of complex affordances, and demonstrated their utility in plan generation. Future research should investigate the inference and learning of complex affordances, and explore their application in areas such as plan recognition, HRI, and others.

6. References

[1] V. R. Ramenzoni, T. J. Davis, M. A. Riley, K. Shockley, Perceiving action boundaries: Learning effects in perceiving maximum jumping-reach affordances, Attention, Perception & Psychophysics 72 (2010) 1110–1119.

[2] T. A. Stoffregen, C. M. Yang, M. R. Giveans, M. Flanagan, B. G. Bardy, Movement in the perception of an affordance for wheelchair locomotion, Ecological Psychology 21 (2009) 1–36.

[3] P. Zech, S. Haller, S. R. Lakani, B. Ridge, E. Ugur, J. Piater, Computational models of affordance in robotics: a taxonomy and systematic classification, Adaptive Behavior 25 (2017) 235–271.

[4] K. Duncker, On Problem-Solving, Psychological Monographs 58 (1945) 1–113.

[5] L. Ye, W. Cardwell, L. S. Mark, Perceiving multiple affordances for objects, Ecological Psychology 21 (2009) 185–217.

[6] P. Langley, M. Sridharan, B. Meadows, Represention, Use, and Acquisition of Affordances in Cognitive Systems, in: Proceedings of the AAAI Spring Symposium on Integrating Representation, Reasoning, Learning, and Execution for Goal Directed Autonomy.

[7] L. Jamone, G. Saponaro, A. Antunes, R. Ventura, A. Bernardino, J. Santos-Victor, Learning object affordances for tool use and problem solving in cognitive robots, CEUR Workshop Proceedings 1544 (2015) 68–82.

[8] L. Jamone, E. Ugur, A. Cangelosi, L. Fadiga, A. Bernardino, J. Piater, J. Santos-Victor, Affordances in Psychology, Neuroscience, and Robotics: A Survey, IEEE Transactions on Cognitive and Developmental Systems 10 (2018) 4–25.

[9] J. J. Gibson, The Theory of Affordances, Halsted Press Division, 1977.

[10] M. Steedman, Plans, affordances, and combinatory grammar, Linguistics and Philosophy 25 (2002) 723–753.

[11] J. G. Greeno, Gibson ' s Affordances, Psychological Review 101 (1994) 336–342.

[12] A.-M. Olteeanu, Two general classes in creative problem-solving? An account based on the cognitive processes involved in the problem structure - representation structure relationship., Proceedings of the Workshop "Computational Creativity, Concept Invention and General Intelligence" (2014).

[13] N. Cantor, Life Task Problem Solving: Situational Affordances and Personal Needs, Personality and Social Psychology Bulletin 20 (1994) 235–243.

[14] K. Davids, D. Araújo, Perception of Affordances in Multi-Scale Dynamics as an Alternative Explanation for Equivalence of Analogical and Inferential Reasoning in Animals and Humans, Theory & Psychology 20 (2010) 125–134.

[15] J. C. Holzhaider, G. R. Hunt, V. M. Campbell, R. D. Gray, Do wild New Caledonian crows (Corvus moneduloides) attend to the functional properties of their tools?, Animal Cognition 11 (2008) 243–254.

[16] T. Gruber, M. N. Muller, V. Reynolds, R. Wrangham, K. Zuberbühler, Community-specific evaluation of tool affordances in wild chimpanzees, Scientific Reports 1 (2011).

[17] D. G. Kemler Nelson, Attention to functional properties in toddlers' naming and problem-solving, Cognitive Development 14 (1999) 77–100.

[18] M. Sridharan, B. Meadows, Learning Affordances for Assistive Robots, in: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), volume 10652 LNAI, Trukuba, Japan, pp. 1–11.

[19] M. Sridharan, B. Meadows, Should I do that? using relational reinforcement learning and declarative programming to discover domain axioms, in: 2016 Joint IEEE International Conference on Development and Learning and Epigenetic Robotics, ICDL-EpiRob 2016, IEEE, 2017, pp. 252–259.

[20] M. Sridharan, B. Meadows, R. Gomez, What can I not do? Towards an architecture for reasoning about and learning affordances, in: Proceedings of the Twenty-Seventh International Conference on Automated Planning and Scheduling, AAAI, 2017, pp. 461–469.

[21] M. Balduccini, M. Gelfond, Logic programs with consistency-restoring rules, in: International Symposium on Logical Formalization of Commonsense Reasoning, AAAI 2003 Spring Symposium Series, volume 102, The AAAI Press.

[22] D. Premack, G. Woodruff, Chimpanzee problem-solving: A test for comprehension, Science 202 (1978) 532–535.

[23] S. Cornus, G. Montagne, M. Laurent, Perception of a Stepping-Across Affordance, Ecological Psychology 11 (1999) 249–267.

[24] M. Gelfond, V. Lifschitz, Action Languages, Linkoping Electronic Articles in Computer and Information Science 3 (1998).

[25] M. Gelfond, D. Inclezan, Some properties of system descriptions of Ad, Journal of Applied Non-Classical Logics 23 (2013) 105–120.

[26] M. Gelfond, Y. Kahl, Diagnostic agents, in: Knowledge representation, reasoning, and the design of intelligent agents : The Answer-Set Programming Approach, Cambridge University Press, New York, 2014, pp. 527–540.

[27] Z. Colaco, M. Sridharan, What happened and why? A mixed architecture for planning and explanation generation in robotics, Australasian Conference on Robotics and Automation, ACRA (2015).

[28] A. M. von Bayern, R. J. Heathcote, C. Rutz, A. Kacelnik, The Role of Experience in Problem Solving and Innovative Tool Use in Crows, Current Biology 19 (2009) 1965–1968.

[29] M. Sridharan, M. Gelfond, S. Zhang, J. Wyatt, REBA: A Refinement-Based Architecture for Knowledge Representation and Reasoning in Robotics, Journal of Artificial Intelligence Research 8755 (2015).

[30] C. W. Geib, C. W. Geib, M. Steedman, On Natural Language Processing and Plan Recognition, in: Proceedings of the 20th international joint conference on Artifical intelligence (IJCAI), pp. 1612–1617.