# Relational affordance learning for task-dependent robot grasping

Laura Antanas[1], Anton Dries[1], Plinio Moreno[2], Luc De Raedt[1]

[1] Department of Computer Science, Katholieke Universiteit Leuven, Belgium
[2] Institute for Systems and Robotics, IST, University of Lisboa, Portugal

**Abstract.** Robot grasping depends on the specific manipulation scenario: the object, its properties, task and grasp constraints. Object-task affordances facilitate semantic reasoning about pre-grasp configurations with respect to the intended tasks, favoring good grasps. We employ probabilistic rule learning to recover such object-task affordances for task-dependent grasping from realistic video data.

## 1 Introduction

Robot grasping skills are essential for acting in dynamic environments. Objects can be grasped in different ways depending on the specific manipulation scenario: the object, its properties, task and grasp constraints. Inspired by the definition of object affordances – which refers to the properties of an object to allow actions to be performed on it by a human or other entity, we investigate the benefits of object-task affordances for task-dependent grasping in a kitchen environment. Our earlier work on task-dependent grasping [2] shows that, when combined with probabilistic reasoning and object/task ontologies, they facilitate compact grasping models which generalize over object/task categories in a natural way, while showing robustness to uncertainty and missing information. Here we propose, as key contribution, a statistical relational learning approach to learn object affordances for task-dependent grasping.

Let us consider the scenario in Fig. 1. A mobile robot with grasping capabilities must grasp a bottle from the shelf and place it on the table. The environment constraints (e.g. narrow spaces) and task constraints (e.g. the most stable pre-grasp gripper pose for grasping the bottle) present a difficult problem which can be solved using semantic reasoning. If we consider the top, middle and bottom as semantic parts of the bottle, the best part to grasp it is from the middle, given that it needs to be placed on the table upright and the top is partially obstructed by the shelf above. Given such semantic object parts (or pre-grasps), object properties, and the intended task, we can learn probabilistic grasp-related rules for our kitchen scenario, e.g., that a bottle affords pick and placing on a surface by grasping it from the middle. The resulting task-dependent affordances give the robot the capability to semantically reason about the best pre-grasp and thus, help the grasp planner. Our experiments show that we can learn reliable relational affordances from realistic and uncertain video data.

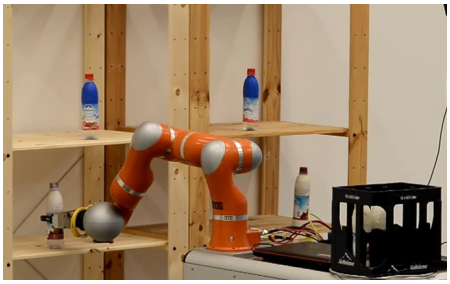

Fig. 1: Manipulation scenario: grasp the bottle from the shelf and place it upright on the table.

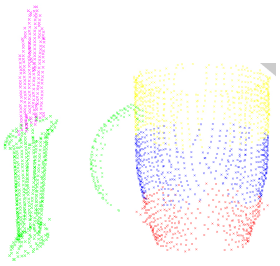

Fig. 2: Semantic parts for knife and cup: yellow-top, blue-middle, red-bottom, green-handle, and magenta-usable area.

## 2 Related work

Much recent work focuses on incorporating task constraints in robot grasping by learning a direct mapping function between good grasps and various constraints (on actions and geometry), action features and object attributes [16,22,17,9,10,8]. We extend this work by considering either object categorical information as an additional feature to predict suitable task-dependent grasping constraints or a task-dependent setting that uses probabilistic logic and world knowledge to reason about best pre-grasps.

Affordances have been considered before in robot manipulation. While in [23] the authors employ estimated visual-based latent affordances, the work in [4] reasons about grasp selection by modeling affordance relations between objects, actions and effects using either a fully probabilistic setting or a rule-based ontology. In contrast, we employ a SRL approach to learn object affordances which generalize over similar object parts and object/task categories. Closely related is the semantic grasping pipeline in [5]. It employs a semantic affordance map which relates gripper approach directions to particular tasks. We exploit additional world knowledge in form of ontologies. This allows us to experiment with a wide range of object categories. Further, to compute plans comprising sequences of actions and to solve complex manipulation tasks, [1] combines symbolic reasoning and learning from demonstrations. In [13] meaningful symbolic relational representations are used to solve sequential manipulation tasks in a goal-directed manner via active relational reinforcement learning. Relational Markov networks have been extended to build relational object maps for mobile robots in order to enable reasoning about hierarchies of objects and spatial relationships amongst them [15]. Related work for generalizing over doors and handles using SRL has been proposed in [18].

However, none of these frameworks solves the problem of learning affordances for semantic task-dependent grasping. Relational affordance models for robots have been learned in a multi-object manipulation task context [19]. Differently, we propose learning pre-grasp configurations using task-category affordances. Our approach features semantic generalization and can tackle unknown objects.

This research topic has great importance in robotics as robots aimed at working in daily environments should be able to manipulate many never-seen-before objects and to deal with increasingly complex scenarios.

The paper is structured as follows. The next section introduces the relational problem of affordance learning. Subsequently, our SRL approach is described. After the experiments section, follow the concluding notes.

## 3 Problem description and representation

Each scene contains one object and the task to be executed. Its semantic visual description consists of the task, object parts, category, pose, and containment together with their probabilities. In a kitchen scenario, the perception algorithm proposed in [6] can segment objects, distinguish between upright and sideways poses and label each part with one of the labels: top, middle, bottom, handle or usable area. This reduces the search space for robot grasp generation, prediction and planning. The object category can be obtained using any object classifier. However, due to good results for grasping point prediction, we employ the manifold-based graph kernel approach proposed in [20]. It ensures a good appearance-based predictor for the object category. The prediction has the form of a probability distribution on object categories. Our kitchen setup considers 11 object categories: {*pan, pot, cup, glass, bowl, bottle, can, hammer, knife, screwdriver, cooking_tool*}. We pick the category with the highest probability to characterize the object in the grasping scenario.

Further, our kitchen setup includes a set of 7 tasks: {*pass, pourOut, pourIn, pickPlaceInUpright, pickPlaceInUpsidedown, pickPlaceInSideways, pickPlaceOn*}. The task *pass* refers to grasping and passing the object to a human in the exact same pose, the tasks *pourOut* and *pourIn* to the actions of pouring liquid out of and inside the object, respectively, after grasping it. Tasks *pickPlaceInUpright*, *pickPlaceInUpsidedown* and *pickPlaceInSideways* refer to picking the object from the current pose and placing it inside a shelf in the upright, upsidedown and sideways poses, respectively. Finally, the task *pickPlaceOn* is defined as picking and placing the object on a surface in the same initial pose.

The scene is represented as a set of relational visual observations. For the scenario in Fig. 1 they are encoded using probabilistic facts, such as 1.0 :: `object(o)`, stating that an object `o` is observed with probability 1.0. The observation `object(o)` is a logical atom, while `object/1` is a predicate symbol of arity 1. The object identifier `o` is a constant and represents a ground term. Terms can also be variables when denoted in uppercase. Ground atoms or facts, such as `object(o)` and `part(o,p1,top)`, do not contain variables and represent particular relations. They possess truth-values. Relational visual observations for our scenario are illustrated in Example 1. We consider that the task is given and not observed, thus it has probability 1.0. We represent it as a probabilistic ground term, e.g., 1.0 :: `task(o,t1,pickPlaceOn)`.

*Example 1.* Relational representation for our scenario in Fig. 1:

```
1.0::object(o).
0.8::category(o,bottle).
0.5::pose(o,upright).
0.9::contains(o,full).
0.5::part(o,p1,top).
0.9::part(o,p2,middle).
0.5::part(o,p3,bottom).

1.0::task(o,t1,pourOut).
1.0::task(o,t2,pass).
1.0::task(o,t3,pourIn).
1.0::task(o,t4,pickPlaceInUpsidedown).
1.0::task(o,t5,pickPlaceInUpright).
1.0::task(o,t6,pickPlaceInSideways).
1.0::task(o,t7,pickPlaceOn).

1.0::affords(o,t1).
1.0::affords(o,t2).
0.0::affords(o,t3).
...
1.0::affords(o,t7).
```

```
0.0::impossible(o,t1).
0.0::impossible(o,t2).
1.0::impossible(o,t3).
1.0::impossible(o,t4).
...
0.0::impossible(o,t7).

0.1::grasp(o,t1,p1).
1.0::grasp(o,t1,p2).
0.01::grasp(o,t1,p3).
0.5::grasp(o,t2,p1).
1.0::grasp(o,t2,p2).
0.01::grasp(o,t2,p3).
0.01::grasp(o,t3,p1).
0.01::grasp(o,t3,p2).
0.01::grasp(o,t3,p3).
...
0.5::grasp(o,t7,p1).
1.0::grasp(o,t7,p2).
0.01::grasp(o,t7,p3).
```

### 3.1 Object category-task (CT) affordances and constraints

We define an object-task affordance as the task afforded by an object category considered in our robot grasping setup. We keep in mind the manipulation capabilities of the gripper mounted on a robotic arm, in our case a KUKA LightWeight Robot (LWR) with two fingers [21]. Fig. 3 illustrates a set of 46 common sense affordances marked with ✓ in the form of a table. They allow us to relate object-task concepts based on human experience and inspired by *AfNet: The Affordance Network* (www.theaffordances.net). By looking at the table, we can extract possible object-task affordance pairs which can be encoded as logical rules. For example the rule $\texttt{affords}(\texttt{X},\texttt{T}) \leftarrow \texttt{bottle}(\texttt{X}),\texttt{task}(\texttt{T},\texttt{pass})$ states that a bottle indicated by variable `X` affords the passing task indicated by variable `T`. The set of affordances can be extended to include new object or task categories.

We can further make abstraction of fine-grained object categories by plugging in an object category ontology as in Fig. 4 (top). The super-categories in the ontology are defined based on the object functionality, and are represented by: {*kitchenContainer, dish, openContainer, canister, container, tool, object*}. For example, the super-category *dish* subsumes the categories *bowl*, *glass* and *cup*. Similarly, tasks can be grouped in super-tasks such as: {*pickPlaceIn, pickPlace, pour, task*} (Fig. 4 bottom). The super-task *pour* refers to the action of pouring the liquid in or out, while the super-task *pickPlaceIn* subsumes the tasks *pickPlaceInUpright*, *pickPlaceInUpsidedown* and *pickPlaceInSideways*. The benefit

| affordances task/object | | | container | | | | | | | tool | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | open container | | | | | canister | | | | | |
| | | | dish | | | kitchen | | | | | | | |
| | | | cup | glass | bowl | pan | pot | bottle | can | hammer | knife | screwdr | cooking |
| pass | | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| pour | | in | ✓ | ✓ | ✓ | ✓ | ✓ | - | - | - | - | - | - |
| | | out | ✓ | ✓ | ✓ | - | - | ✓ | ✓ | - | - | - | - |
| p&p | in | upright | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | - | - | - | - |
| | | upsidedown | ✓ | ✓ | ✓ | - | - | - | - | - | - | - | - |
| | | sideways | - | - | - | - | - | - | - | ✓ | ✓ | ✓ | ✓ |
| | on | | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

Fig. 3: Object-task affordances are marked by ✓, constraints by −.

of exploiting ontological structure is that we can make abstraction of the fine-grained object categories and tasks. Ontologies are symbolic high-level knowledge which allows a compact representation of the affordance model by generalizing over similar object and task categories in a natural and straightforward way. As a result, they allows us to experiment with a wide range of objects and better deal with missing, uncertain or inaccurate information.

Ontologies can be translated into deterministic logical rules and directly used by our learner. For example, $\texttt{supercategory}(\texttt{X}, \texttt{container}) \leftarrow \texttt{category}(\texttt{X}, \texttt{bottle})$ states that "any bottle is a container", $\texttt{pour}(\texttt{T}) \leftarrow \texttt{pourIn}(\texttt{T})$ specifies that "any task of pouring liquid in to fill some object is a pouring task". The arguments `X` and `T` are variables and indicate the object identifier and task, respectively. We can then generally state that any container affords the task of pouring, i.e., $\texttt{affords}(\texttt{X}, \texttt{T}) \leftarrow \texttt{container}(\texttt{X}), \texttt{pour}(\texttt{T})$. However, this is not always true, as pouring liquid in a canister is an almost impossible task, even for a human. We encode such constraints via the *impossible*/2 predicate. The rule $\texttt{impossible}(\texttt{X}, \texttt{T}) \leftarrow \texttt{canister}(\texttt{X}), \texttt{pourIn}(\texttt{T})$ states that a canister does not afford the task of pouring in. Constraints are marked in Table 3 by −.

A first goal of this work is to improve robot grasping by learning relational object-task affordances and constrains from data. This is done by specifying two separate learning problems. The CT affordance learning problem is indicated by keeping as learning target the $\texttt{affords}(\texttt{X}, \texttt{T})$ predicate, while the CT constraint problem via the target predicate $\texttt{impossible}(\texttt{X}, \texttt{T})$.

### 3.2 Object part-category-task (PCT) affordances

Further, depending on the object properties, its parts and task, the object should be grasped in different ways. To reason about good pre-grasp configurations given

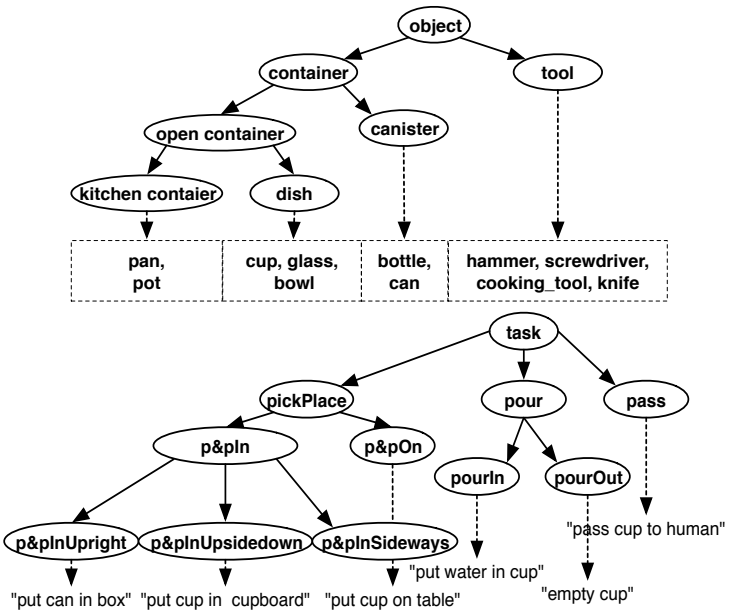


Fig. 4: Object category ontology (top) and task ontology (bottom).

the intended task, we use semantic object parts. Similar to object categories, pre-grasps can be associated to specific tasks. Each task activates grasping affordances according to associations between object categories, object parts and gripper poses. Besides object category-task associations, the second goal of our work is to learn object part-category-task relations. While the first are general pairwise affordances, the second are grasp-related triplets that may rely on the first. For example, the rule $\texttt{grasp}(\texttt{X},\texttt{T},\texttt{P}) \leftarrow \texttt{affords}(\texttt{X},\texttt{T}),\texttt{pickPlaceInUpsidedown}(\texttt{T}),\texttt{glass}(\texttt{X}),\texttt{pose}(\texttt{X},\texttt{upsidedown}),\texttt{part}(\texttt{X},\texttt{PId},\texttt{bottom})$ states that a glass X in the upside-down pre-grasp pose affords the task T of picking and placing inside the cupboard in an upside-down post-grasp pose by grasping it from the bottom. The PCT affordance learning problem is specified via the target predicate $\texttt{grasp}(\texttt{X},\texttt{T},\texttt{P})$. We can make abstraction of fine-grained object categories and tasks by plugging in the object category and task ontologies here as well. Most of the times we can state that any dish in an initial upside-down pose can be picked and placed inside a cupboard in any pose by grasping it from the bottom, i.e., $\texttt{grasp}(\texttt{X},\texttt{T},\texttt{P}) \leftarrow \texttt{affords}(\texttt{X},\texttt{T}),\texttt{pickPlaceIn}(\texttt{T}),\texttt{dish}(\texttt{X}),\texttt{pose}(\texttt{X},\texttt{upsidedown}),\texttt{part}(\texttt{X},\texttt{PId},\texttt{bottom})$. Thus, the introduction of super-categories and super-tasks reduces considerably the number of rules rendering much more compact model which are easier to interpret.

### 3.3 The affordance learning problem

Using the visual observations introduced in Example 1 one can learn several affordances, e.g., that the bottle affords pick and placing on a surface in the initial upright pose and, in this case, it should be grasped from the middle or from the top (from the bottom the gripper might hit the shelf below, from the top with a smaller probability since the bottle is full and a bit difficult to grasp), that it affords pouring out, and, in this case the bottle should be grasped from the middle (from the bottom the gripper might hit the shelf below, from the top it is rather difficult even for a human to pour out). Further, constraints can be also inferred, e.g., the bottle cannot be poured in liquid or placed upside-down. In order to do so, we assume to have labeled examples: object category-task relations specified via the target predicate `affords`/2, object category-task constraints via `impossible`/2, and object part-category-task affordances indicated by the target predicate `grasp`/3.

Ground target predicates or learning examples for each problem are illustrated in Example 1. Each learning problem is tackled in turn. Every learning example is a fact labeled with a target probability. In our scenario, target atom 1.0 :: `affords(o,t1)` states that bottle `o` allows pouring out with maximum probability and is a learning example for the CT learning problem. Target label 1.0 :: `grasp(o,t1,p2)` is a learning example for the PCT problem and asserts that the bottle can be grasped by the middle part `p2` with probability 1.0. The resulting set of probabilistic ground facts from all the scenarios corresponding to one learning problem represent input data for our probabilistic rule learner. For example, for the CT affordance problem, the learner takes as input features all grounded object, category, pose, containment and task predicates and target `affords`/2 predicates, while for the PCT affordance problem it takes, in addition as input features all grounded parts and as targets the `grasp`/3 predicates. In our learning from entailment setting, probabilistic ground targets are positive learning examples. ProbFOIL+ derives negative examples automatically by taking combinations of possible values for the target arguments. A sampling step is performed such that the number of negatives balances the number of positives.

## 4 Approach: probabilistic rule learning

Given the representation of our input and output we employ probabilistic rule learning for affordance learning. The learned probabilistic rules would have the form `x :: target` ← `body`, where the target is represented, for example, by the predicate `affords(X,T)` in the CT affordance learning problem and the body is represented by the set of pre-grasp configurations w.r.t the object category, pose, containment and task. The set of rules obtained are used to predict grounded target predicates. We proceed with learning from entailment since our examples are facts that are probabilistically entailed by the theory. This setting is incorporated in the probabilistic rule learner ProbFOIL+. It combines the principles of the rule learner FOIL with the probabilistic Prolog called ProbLog [11] and is capable of learning probabilistic rules from probabilistic ground input facts.

Because ProbFOIL+ is a natural probabilistic extension of ILP and rule learning with respect to probabilistic data, we employ it to learn relational affordances. ProbFOIL+ generalizes FOIL, nFOIL [14], mFOIL [12] and ProbFoil [7]. Its output is a probabilistic classifier in the form of a set of generalized rules that returns a probabilistic target atom.

The ProbFOIL+ algorithm directly generalizes the mFOIL rule learner. It follows a typical sequential covering approach where the outer loop of the algorithm starts from an empty set of clauses and repeatedly adds clauses to the hypothesis until no more improvements are observed with respect to some global scoring function (e.g. accuracy, recall or F1-measure). The clause to be added is obtained in a greedy manner by performing beam search using $m$-estimate such that it maximizes a local scoring function using a refinement operator. Each clause is learned in a greedy manner by performing beam search using $m$-estimate as a local scoring function. What sets ProbFOIL+ apart from mFOIL is its support for probabilistic data by generalizing the concepts of true/false positive/negative to a probabilistic context. In addition, it performs an additional step of parameter learning that allows it to learn rules that express probabilistic relationships.

While ProbLog and Prolog assume that the rules are definite clauses, in ProbFOIL+ we use probabilistic rules. We note that all facts for such rules are independent of one another, and the probability is determined by the rule learning algorithm. ProbFOIL+ uses versions of standard scoring functions for rule learning. As the global scoring function, which determines the stopping criterion of the outer loop, we use F1 measure. The local scoring function is based on the m-estimate, a variant of precision that is more robust against noise in the training data. Both metrics are based on the number of examples correctly classified as positive (true positives) and the number of examples incorrectly classified as positive (false positives) which are upgraded for use in a probabilistic setting. While in a deterministic setting, each example $e_i$ has a 1/0 target classification, in ProbFOIL+ it has a probability value $p_i$. This means that every example contributes $p_i$ to the positive part of the dataset and (1  $p_i$) to the negative part of the dataset, which generalizes the deterministic setting with $p_i = 1$ for positive and $p_i = 0$ for negative examples. ProbFOIL+ defines we define the positive part of a dataset of size $M$ as $P = \sum_{i=0}^{M} p_i$ and the negative part as $N = \sum_{i=0}^{M} 1 - p_i$. The same approach generalizes the predictions of a model to the probabilistic setting where a hypothesis $H$ will predict a value $p_{H,i}$ for example $e_i$ instead of 0 or 1. In this way the rule learner uses a probabilistic version of the true positive and false positive rates of the predictive model. If $H$ overestimates the target value of $e_i$, that is, $p_{H,i} > p_i$ then the true positive part will be maximal, that is, equal to $p_i$. The remaining part $p_{H,i} - p_i$, is part of the false positives. If $H$ underestimates the target value of $e_i$ then the true positive part is only $p_{H,i}$ and the remaining part $p_i - p_{H,i}$ contributes to the false negative part of the prediction.

In order to avoid learning large hypotheses with many clauses that only have limited contributions, ProbFOIL+ uses a significance test, used also by mFOIL.

It is a variant of the likelihood ratio statistics. As a local stopping criteria for finding a viable next candidate, the clause must have a refinement that has a higher local score than the current best rule, has a significance that is high enough (according to a preset threshold), and has a better global score than the current rule set without the additional clause.

As input we provide ProbFOIL+ with the target predicate to be learned (e.g. `affords`/2) and a description of the refinement operator in terms of mode declarations. For example, `task`$(+,+,$`c`$)$ indicates that the first two arguments should be variables that already exist in the clause, and the third argument is to be replaced by a constant. ProbFOIL+ then proceeds by iteratively extending the clause with one literal, pruning the least promising candidates at each step, until no more improvement can be made.We refer to the ProbFOIL+ paper for more details.

## 5 Experiments

We experiment on task-dependent robotic grasping datasets for kitchen-related scenarios introduced in [2]. We consider two datasets to quantitatively investigate the robustness and power of generalization of ProbFOIL+ for learning grasping affordances. The synthetic dataset denoted $S_{SYN}$ considers flawless detection of objects from 3D meshes. The object points are distributed uniformly on the object surface according to their size by applying the midpoint surface subdivision technique. The object pose, its parts and object containment are manually labeled, while the object category is estimated using the global similarity classifier in [2]. The dataset is synthetic and actual grasps are not executed. It contains 41 objects belonging to all categories in our ontology and 102 grasping scenarios. This synthetic dataset serves as an upper-bound comparison scenario to the other realistic scenarios and allows an extensive evaluation of the generalization capabilities of the affordance learner.

The other dataset is obtained with the ORCA simulator [3] which provides sensors (laser range camera Asus Xtion PRO and the Universal Gripper WSG 50 force sensor), the robotic arm (KUKA LightWeight Robot (LWR)), objects and interface to a physics engine (Newton Game Dynamics library) for robot grasping simulation. The other modules that we use on top of ORCA, i.e., object completion, part and pose detection, category recognition and the tree-based motion planner (available in the Open Motion Planning Library), are external to ORCA and interfaced with the simulated robot. The datasets contain 25 objects belonging to all categories, except pot and cooking tool, and 134 grasping scenarios. We assume all containers empty. Each object is placed on top of a table. We obtain the dataset $S_{REAL}$ by estimating object pose, category and its parts safter the point cloud completion. It may have missing parts, when they are occluded or not detected, or extra parts according to the limitations of the detection algorithm. The pose and the parts have associated a probability according to the limitations of the detection algorithms.

| | CT affordances | | CT constraints | | PCT affordances | |
|---|---|---|---|---|---|---|
| Dataset | $S_{SYN}$ | $S_{REAL}$ | $S_{SYN}$ | $S_{REAL}$ | $S_{SYN}$ | $S_{REAL}$ |
| No rules | 40 | 15 | 42 | 41 | 33 | 22 |
| Accuracy % | 98 | 98 | 97 | 85 | 95 | 93 |
| F1 | 0.86 | 0.86 | 0.82 | 0.83 | 0.52 | 0.54 |
| Recall | 0.87 | 0.77 | 0.77 | 0.76 | 0.37 | 0.58 |

Table 1: Number of learned rules, accuracy, F1 and recall for fine-grained categories and tasks. Input features used are object parts, category and executed task (without pose and containment).

| | CT affordances | | CT constraints | | PCT affordances | |
|---|---|---|---|---|---|---|
| Dataset | $S_{SYN}$ | $S_{REAL}$ | $S_{SYN}$ | $S_{REAL}$ | $S_{SYN}$ | $S_{REAL}$ |
| No rules | 51 | 16 | 39 | 52 | 81 | 21 |
| Accuracy % | 98 | 98 | 98 | 98 | 97 | 96 |
| F1 | 0.89 | 0.89 | 0.86 | 0.84 | 0.83 | 0.62 |
| Recall | 0.79 | 0.80 | 0.77 | 0.89 | 0.76 | 0.52 |

Table 2: Input information, besides object parts, category and executed task, includes object pose and containment. Number of learned rules and accuracy are reported for fine-grained categories and tasks.

Our goal is to investigate if we can recover affordances from labeled data and to evaluate is these rules are good. In order to do so, we manually inspect the learned rules for the 3 datasets and compare them against the affordance table. Besides the number of correct rules recovered, we report recall, F1 measure, accuracy which are calculated based on classified probabilistic examples. The goal of this work is to focus on a more qualitative evaluation, and thus, we use all available data for training. Reported evaluation results are obtained by optimizing F1 measure on this data.

### 5.1 Results for object category-task (CT) affordances

We obtain CT affordances by specifying `affords`/2 as the learning target. This gives a dataset of 714 examples for $S_{SYN}$, and 882 examples for $S_{REAL}$. As input information we consider two settings. In a first setting we use only object category, parts and task in order to asses the importance of the object category and pose for affordances. In the second setting we include the later as well. Fig. 5 shows part of learned rules on $S_{SYN}$ for the first input setting by employing object fine-grained categories. A learned rule in the discovered set is $0.78 :: \texttt{affords(A,B)} \leftarrow \texttt{category(A,cup)}, \texttt{task(A,B,pourIn)}$. We obtain 40 rules out of which 38 are fine-grained affordance rules (from 44 possible cf. Table 3) and an accuracy of 98% as Table 1 shows. We note that our dataset did not contain positive targets for pot-pourIn and pan-pourIn. The other affordances were not recovered because they depend also on the object initial pose

affords(A,B)← task(A,B,pickPlaceOn), category(A,cup). 0.73
affords(A,B)← task(A,B,pickPlaceOn), category(A,glass). 0.54
affords(A,B)← task(A,B,pickPlaceOn), category(A,bowl). 0.67
affords(A,B)← task(A,B,pickPlaceOn), category(A,pot). 0.73
affords(A,B)← task(A,B,pickPlaceOn), category(A,bottle). 0.58
affords(A,B)← task(A,B,pickPlaceOn), category(A,can). 0.63
affords(A,B)← task(A,B,pickPlaceOn), category(A,knife). 0.69
...
affords(A,B)← category(A,cup), task(A,B,pickPlaceInUpsidedown). 0.78
affords(A,B)← category(A,bowl), task(A,B,pickPlaceInUpsidedown). 0.75
affords(A,B)← category(A,glass), task(A,B,pickPlaceInUpsidedown). 0.82
affords(A,B)← category(A,cup), task(A,B,pourIn). 0.78
affords(A,B)← category(A,bowl), task(A,B,pourIn). 0.75
affords(A,B)← category(A,glass), task(A,B,pourIn). 0.81

affords(A,B)← supercategory(A,dish), task(A,B,pickPlaceOn). 0.73
affords(A,B)← supercategory(A,canister), task(A,B,pickPlaceOn). 0.54
affords(A,B)← supercategory(A,tool), task(A,B,pickPlaceOn). 0.62
...
affords(A,B)← supercategory(A,dish),task(A,B,pickPlaceInUpsidedown).0.82
affords(A,B)← supercategory(A,dish),task(A,B,pourIn). 0.88

Fig. 5: Examples of CT affordances learned for $S_{SYN}$ using object fine-grained categories (top) and super-categories (bottom).

affords(A,B)← task(A,B,pickPlaceOn), supercategory(A,object). 0.63
affords(A,B)← task(A,B,pass), supercategory(A,object). 0.77
affords(A,B)← task(A,B,pickPlaceInsideSideways), supercategory(A,tool). 0.81
affords(A,B)← task(A,B,pickPlaceInsideUpright), supercategory(A,dish). 0.84
affords(A,B)← task(A,B,pickPlaceInsideUpright), supercategory(A,canister). 0.86
affords(A,B)← task(A,B,pickPlaceInsideUpright), supercategory(A,openContainer).0.87
affords(A,B)← task(A,B,pourOut), supercategory(A,canister). 0.87

Fig. 6: Examples of learned CT affordances for $S_{REAL}$ using super-categories.

and its containment. When we include them, the number of meaningful affordances learned does not increase. The learner discovers more category-task-pose (specialized CT affordances with the pose refinement), task-pose dependencies, but not new category-task affordances. This indicates that the pose is not so relevant for the CT pairs. The extra 2 input features do not notably impact the evaluation measures, which also proves, that the dataset, even synthetic, is not perfect. Next, using super-categories, we can summarize the set of 38 fine-grained affordances with 15 rules, while keeping the same accuracy, recall and F1-measure. Learned supercategory-task affordances are more general, specifically cup, glass, bowl are replaced by dish, bottle and can by canister, and hammer, screwdriver, cooking tool and knife by tool. Examples of more general rules learned are illustrated in Fig. 5.

For the realistic dataset we can recover 35 fine-grained affordances out of 39 possible (cf. Table 3 without pot and cooking tool which are not included in the datasets) for $S_{REAL}$. We obtain 2 affordance rules for category pot which is not in the dataset (according to Table 3 one is correct, the other not), but none for *pickPlaceInUpsidedown*. This is due to object misclassification. We note that 22 of these affordances are summarized by 2 rules, i.e., 0.44 :: $\texttt{affords}(\texttt{X},\texttt{T}) \leftarrow \texttt{task}(\texttt{X},\texttt{T},\texttt{pickPlaceOn})$ and 0.72 :: $\texttt{affords}(\texttt{X},\texttt{T}) \leftarrow \texttt{task}(\texttt{X},\texttt{T},\texttt{pass})$. In the body appears only the task because the rule applies to all categories (11). This is due to lack of negative examples.

By using super-categories, we can replace the set of fine-grained rule with 7 generalized rules keeping similar evaluation values. The obtained rules are more general and can apply to new object categories. ProbFOIL+ does not learn again any rules for task pickPlaceInUpsidedown and pourIn. Examples of general rules using super-categories for $S_{REAL}$ are illustrated in Fig. 6.

### 5.2 Results for object category-task (CT) constraints

To obtain CT constraints we give as target predicate $\texttt{impossible}/2$. It represents the inverse of $\texttt{affords}/2$ probabilistically in the sense that what is affordable with a very small probability it is impossible with a high probability. We note that we obtain 42 constraint rules for $S_{SYN}$ and 41 for $S_{REAL}$ using fine-grained categories, without pose and containment. When we include the later 2 features, the model slightly improves in terms of rules, but also accuracy. A mistake that the learner returns is the constraint 0.8 :: $\texttt{impossible}(\texttt{X},\texttt{T}) \leftarrow \texttt{task}(\texttt{X},\texttt{T},\texttt{pickPlaceInUpsidedown}), \texttt{category}(\texttt{X},\texttt{cup})$. This constraint holds only when the cup is full. If we include pose and containment, this constraint is removed. Looking at the evaluation results as well, we note that for CT constraints the initial pose of the object and its containment play a fairly important role. By using super-categories the learned affordance model reduces from 42 rules to 13 rules for $S_{SYN}$ while keeping similar accuracy, F1 and recall. For $S_{REAL}$ we obtain 18 rules instead of 41 with better evaluation values.

### 5.3 Results for object part-category-task (PCT) affordances

The target to be learned is $\texttt{grasp}/3$. This gives us a dataset of 2093 examples for $S_{SYN}$ and 2674 for $S_{REAL}$. We focus on the setting that considers as input information the task, object category, parts and pose, since the part from which to grasp an object for a given task highly depends on the pose as well, as tables 1 and 2 show. Experiments using ProbFOIL+ give us a grasping model of 81 affordance rules for $S_{SYN}$ and 21 rules for $S_{REAL}$ (when pose is considered). By introducing super-categories the grasp-based models are generalized from 81 rules to 31 and from 21 to 17, respectively, while keeping a close accuracy. A part-category-task affordance learned from $S_{SYN}$ is for example 0.8 :: $\texttt{grasp}(\texttt{A},\texttt{B},\texttt{C}) \leftarrow \texttt{part}(\texttt{A},\texttt{C},\texttt{usable\_area}), \texttt{supercategory}(\texttt{A},\texttt{tool}), \texttt{task}(\texttt{A},\texttt{B},\texttt{pass})$. More learned rules are illustrated in Fig. 7.

```
grasp(A,B,C)← stask(A,B,pickPlaceOn), scategory(A,canister), part(A,C,middle). 0.78
grasp(A,B,C)← stask(A,B,pickPlaceOn), scategory(A,canister), part(A,C,bottom), pose(A,upsidedown). 0.86
grasp(A,B,C)← stask(A,B,pickPlaceOn), scategory(A,canister), part(A,C,top), pose(A,upright). 0.88
grasp(A,B,C)← stask(A,B,pickPlaceOn), scategory(A,canister), part(A,C,top), pose(A,sideways). 0.91
grasp(A,B,C)← stask(A,B,pickPlaceOn), scategory(A,dish), part(A,C,middle). 0.85
grasp(A,B,C)← stask(A,B,pickPlaceOn), scategory(A,dish), pose(A,upsidedown), part(A,C,bottom). 0.85
grasp(A,B,C)← stask(A,B,pickPlaceOn), scategory(A,dish), pose(A,upsidedown), part(A,C,handle). 0.86
grasp(A,B,C)← stask(A,B,pickPlaceOn), scategory(A,dish), pose(A,upright), part(A,C,top). 0.88
grasp(A,B,C)← stask(A,B,pickPlaceOn), scategory(A,dish), pose(A,upright), part(A,C,middle). 0.81
grasp(A,B,C)← stask(A,B,pickPlaceOn), scategory(A,dish), pose(A,sideways), part(A,C,middle). 0.83
grasp(A,B,C)← stask(A,B,pickPlaceOn), scategory(A,kitchenContainer), pose(A,sideways), part(A,C,handle). 0.84
grasp(A,B,C)← stask(A,B,pickPlaceOn), scategory(A,kitchenContainer), pose(A,upright), part(A,C,handle). 0.85
grasp(A,B,C)← stask(A,B,pickPlaceOn), scategory(A,tool), part(A,C,handle). 0.83

grasp(A,B,C)← stask(A,B,pickPlaceInsideUpsidedown), scategory(A,dish), pose(A,sideways), part(A,C,middle). 0.95
grasp(A,B,C)← stask(A,B,pickPlaceInsideUpsidedown), scategory(A,dish), pose(A,upsidedown), part(A,C,middle).0.94

grasp(A,B,C)← stask(A,B,pourIn), scategory(A,dish), pose(A,upsidedown), part(A,C,middle). 0.92
grasp(A,B,C)← stask(A,B,pourIn), scategory(A,dish), pose(A,upsidedown), part(A,C,bottom). 0.94
grasp(A,B,C)← stask(A,B,pourIn), scategory(A,dish), pose(A,sideways), part(A,C,middle). 0.94
```

Fig. 7: Examples of PCT affordances learned for $S_{SYN}$ using super-categories and super-tasks.

## 6 Conclusions

Our previous experiments on robot grasping with respect to the intended high-level task confirm the importance of high-level reasoning and world knowledge as opposed to using solely local shape information for robot grasping. The use of affordances and object/task ontologies plays a key role in obtaining better robot grasping. In this paper we propose a probabilistic rule learning approach to learn rule-based affordances that generalize over similar object parts and object/task categories and can be used to semantically reason in task-dependent robot grasping. Our experiments show that we can learn different reliable relational affordances from realistic data.

## References

1. N. Abdo, H. Kretzschmar, and C. Stachniss. From Low-Level Trajectory Demonstrations to Symbolic Actions for Planning. In *ICAPS Workshop on Combining Task and Motion Planning for Real-World Applications*, pages 1–8, 2012.
2. L. Antanas, P. Moreno, M. Neumann, R. Pimentel de Figueiredo, K. Kersting, J. Santos-Victor, and L. De Raedt. High-level reasoning and low-level learning for grasping: A probabilistic logic pipeline. *CoRR*, abs/1411.1108, 2014.
3. H. Baltzakis. Orca simulator.
4. C. Barck-Holst, M. Ralph, F. Holmar, and D. Kragic. Learning grasping affordance using probabilistic and ontological approaches. In *ICRA*, pages 1–6, 2009.

5. H. Dang and P. K. Allen. Semantic grasping: Planning robotic grasps functionally suitable for an object manipulation task. In *IEEE/RSJ ICIRS*, pages 1311–1317, 2012.
6. Rui Figueiredo de Figueiredo, Plinio Moreno, and Alexandre Bernardino. Automatic object shape completion from 3d point clouds for object manipulation. In *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications VISIGRAPP*, pages 565–570, 2017.
7. L. De Raedt and I. Thon. *Probabilistic Rule Learning*, pages 47–58. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.
8. Renaud Detry, Carl Henrik Ek, Marianna Madry, and Danica Kragic. Compressing grasping experience into a dictionary of prototypical grasp-predicting parts. In *International Workshop on Human-Friendly Robotics*, pages 1–1, 2012.
9. Renaud Detry, Carl Henrik Ek, Marianna Madry, and Danica Kragic. Learning a dictionary of prototypical grasp-predicting parts from grasping experience. In *ICRA*, pages 601–608, 2013.
10. Renaud Detry, Carl Henrik Ek, Marianna Madry, Justus H. Piater, and Danica Kragic. Generalizing grasps across partly similar objects. In *ICRA*, pages 3791–3797, 2012.
11. Anton Dries, Angelika Kimmig, Wannes Meert, Joris Renkens, Guy Van den Broeck, Jonas Vlasselaer, and Luc De Raedt. *ProbLog2: Probabilistic Logic Programming*, pages 312–315. Springer International Publishing, Cham, 2015.
12. S. Džeroski. Handling imperfect data in inductive logic programming. In *SCAI*, pages 111–125, 1993.
13. J. Kulick, M. Toussaint, T. Lang, and M. Lopes. Active learning for teaching a robot grounded relational symbols. In *IJCAI*, pages 1451–1457. AAAI Press, 2013.
14. Niels Landwehr, Kristian Kersting, and Luc De Raedt. nfoil: Integrating naïve bayes and foil. In *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 2*, AAAI'05, pages 795–800. AAAI Press, 2005.
15. B. Limketkai, L. Liao, and D. Fox. Relational Object Maps for Mobile Robots. In *IJCAI*, pages 1471–1476, 2005.
16. Marianna Madry, Dan Song, Carl Henrik Ek, and Danica Kragic. "Robot bring me something to drink from": object representation for transferring task specific grasps. In *ICRA Workshop on Semantic Perception, Mapping and Exploration*, pages 1–6, 2012.
17. Marianna Madry, Dan Song, and Danica Kragic. From object categories to grasp transfer using probabilistic reasoning. In *ICRA*, pages 1716–1723, 2012.
18. B. Moldovan, L. Antanas, and M. Hoffmann. Opening doors: An initial SRL approach. In *LNCS*, volume 7842, pages 178–192, 2013.
19. Bogdan Moldovan, Plinio Moreno, Davide Nitti, José Santos-Victor, and Luc De Raedt. Relational affordances for multiple-object manipulation. *Autonomous Robots*, May 2017.
20. M. Neumann, P. Moreno, L. Antanas, R. Garnett, and K. Kersting. Graph kernels for object category prediction in task–dependent robot grasping. In *MLG*, pages 1–6, 2013.
21. KUKA robotics. Kuka lightweight robot (LWR).
22. D. Song, K. Huebner, V. Kyrki, and D. Kragic. Learning task constraints for robot grasping using graphical models. In *IEEE/RSJ ICIRS*, pages 1579 –1585, 2010.
23. J. Sweeney and R. A. Grupen. A model of shared grasp affordances from demonstration. In *Humanoids*, pages 27–35, 2007.