Learning Relational Affordance Models for Robots in Multi-Object Manipulation Tasks

Bogdan Moldovan Plinio Moreno Martijn van Otterlo José Santos-Victor Luc De Raedt

***Abstract*— Affordances define the action possibilities on an object in the environment and in robotics they play a role in basic cognitive capabilities. Previous works have focused on affordance models for just one object even though in many scenarios they are defined by configurations of multiple objects that interact with each other. We employ recent advances in statistical relational learning to learn affordance models in such cases. Our models generalize over objects and can deal effectively with uncertainty. Two-object interaction models are learned from robotic interaction with the objects in the world and employed in situations with arbitrary numbers of objects. We illustrate these ideas with experimental results of an action recognition task where a robot manipulates objects on a shelf.**

I. Introduction

Robotics aims to develop mobile, *physical* agents capable of reasoning, learning and manipulating their environment. Early approaches such as the well-known Shakey robot used the logical STRIPS representation [6], and several other such symbolic representations have been used [9], [20]. Approaches based on logic are effective at dealing with higher knowledge needed for planning and reasoning, but the physical aspect of robots requires dealing with various kinds of *uncertainty*, typically not handled by such formalisms. These aspects include interpreting noisy sensors, processing image streams from cameras, controlling noisy physical actuators for manipulation, and in general, solving many *grounding* and *anchoring* problems [2]. The use of *probabilistic reasoning and learning* techniques is now widespread in robotics [21], yet mostly without employing structured, logical representations. For this we need *statistical relational learning* (SRL) [4], [3] which combines logical representations, probabilistic reasoning and machine learning. Recent works have explored the use of SRL and shown how to effectively combine probabilistic and logical methods in robotics domains (e.g. a kitchen scenario [10]).

A promising approach for the development of humanoid robots' skills is the learning of *object affordances*. Affordances are modeled as relations between three variables, *objects* (O), *actions* (A) and *effects* (E), such that given two, one can predict the third. Thus, affordances allow to perform three tasks: i) predict the *outcome* (and plan) an action (infer E, given O and A), ii) recognize a performed action (A, given E and O), or iii) select objects according to a task requirement (determine O, based on observed A and E) [17]. The introduction of affordances to robotics follows the developmental framework, which proposes to acquire new skills on top of previous ones by experimentation and interaction with the environment [15]. This allows the robot to perform more complex tasks and learn by imitation in a goal oriented manner [17]. The typical scenario of affordance learning considers isolated objects in the environment, which is a reasonable assumption for many robotic tasks. However, there are several cases where it is necessary to consider the (spatial) relations between several objects such as: shelf sorting, placing groceries in a shopping cart and packing items in a bag. Here we address such settings, which we refer to as *affordance learning* in multiple-object settings.

A. Approach

We study the extension of the affordances model to include more than one object, where the robot's actions may affect several objects in the environment. Previous approaches adopt Bayesian networks (BNs) in order to encode the relations between an object's properties, actions and effects in a probabilistic manner [14], [17], [11]. A straightforward extension to that model to handle multiple objects is to add more variables to the BN and perform structure learning and inference as in previous works. However, two issues must be considered: (i) a separate BN is learned and instantiated for every distinct number of objects in the environment and (ii) the number of samples needed to learn a BN with more than three objects is very large. We address these issues using SRL (and probabilistic programming languages) [5], which allows to build just one model (probabilistic program) that supports inference for any number of objects, so the structure learning of several BNs is not needed and performing inference does not need to switch between BNs. The SRL generalization also enables inference on scenarios with (spatial) relations between more than two objects, using only samples of scenarios with one and two objects.

We use the simulator of the iCub robot [16] to perform experiments, where the humanoid robot is capable of discovering the affordances associated with manipulation actions (*grasp*, *push* and *tap*), applied to objects with different properties (*color* and *shape*). The *effects* include *displacement* and *orientation changes* for each object and *contact* between objects. These effects are closely related to the multiple object scenarios above mentioned, where the position and contact information between objects guide the robot to select

Bogdan Moldovan and Luc De Raedt are with the Department of Computer Science, Katholieke Universiteit Leuven, Belgium

Plinio Moreno and José Santos-Victor are with the Electrical & Computer Engineering Department, Instituto Superior Técnico, Portugal

Martijn van Otterlo is with the AI Department, Radboud University Nijmegen, The Netherlands

Bogdan Moldovan is supported by IWT (agentschap voor Innovatie door Wetenschap en Technologie). This work is supported by the European Community's 7th Framework Programme, grant agreement First-MM-248258.

the appropriate object and action for the subsequent step.

An example of the type of applications for multi-object affordance models is the *shopping shelf scenario*, depicted in Figure 1. The robot's task is to place the magenta object onto a shelf at the given target location. This involves first pushing the other objects to the left to make space (1), followed by pushing the magenta object in its target location (2).

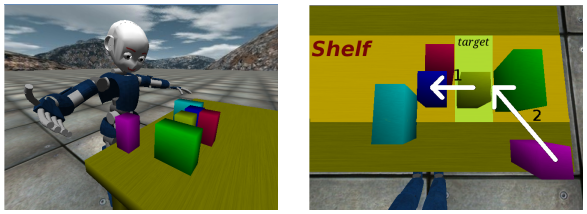


Fig. 1. Shelf scenario with action sequence for object placement.

B. Contributions and Outline

Affordance models using BNs need to be tailored to the specific number of objects in a scene, and their size increases with this number. The contribution of this paper is the use of SRL methods for extending affordance models to multi-object scenes by robustly going from two-object interactions to a variable number of objects by generalization over objects and modeling of probabilistic aspects. SRL provides ease of modeling and increased comprehensibility by allowing the transfer of the model structure to other domain sizes. The experiments using an iCub robot show in multi-object scenarios that this approach gives comparable results to a tailored BN, while maintaining the flexibility of not needing retraining and the robustness for large number of objects.

The rest of this paper is organized as follows: Section II presents related work, Section III the technical setting and basic skills of the robot and Section IV introduces affordance-based models and illustrates them for a single object. Section V includes the main contribution of this paper: it explains the extension toward relational, multi-object affordance models and describes examples of two-object interactions. Section VI presents experimental results in a multiple-object setting and we conclude in Section VII.

II. Related Work

This paper builds upon previous work in affordances, a concept introduced by J. J. Gibson [7] and used to model world-robot interaction. More specifically, we extend previous work on affordances in the context of imitation learning in [14], [17] where an affordance model for a single object is learned with a BN in the context of a robot interaction game. Related work is also the use of the affordance models in the context of word-to-meaning association in [11]. In order to extend the affordance model to the relational domain, this paper builds upon work in SRL, including [3], [4]. The ProbLog language which is used for modeling in this paper is described extensively in [5], [8]. Finally, this work falls in the same area as that of probabilistic robotics [21] and of providing robots with logic and probabilistic reasoning capabilities [10], [9], as well as in the context of planning [12] and object-action complexes [23].

III. Basic Skills of the Robot

We employ, both in a real setting and in simulation, the *iCub humanoid robot*, which has a head with two cameras, two arms and two legs. We use only one arm, the cameras and the software modules that provide: (i) motion control to reach a target position [18], (ii) image segmentation [1] and (iii) stereo triangulation. We build on these elements the basic skills of the robot: motor skills to perform the actions and perceptual skills to measure object features and effects.

We assume the robot is provided with a set of core motor actions with parameters adjusted after self-experience [17]. We consider three actions: $A = \{push, tap, grasp\}$. Each moves the iCub hand over a preprogrammed orientation and distance, but can also be parameterized in the future. The motion of the hand to a target position is provided by visual skills. The hand is moved to the target position by a minimum-jerk Cartesian controller which reaches a position as close as possible to a given rest position while coping with the kinematic constraints of the robot, such as joint limits, damage avoidance and hand orientation [18]. The action is performed after the Cartesian controller has terminated. The action execution is preprogrammed due to the limitations of the simulator and the complexity of the iCub's hand.

The perceptual skills of the robot include color segmentation and 3D object localization. The color segmentation algorithm relies on a synergistic approach that combines a confidence-based edge detector and mean shift segmentation [1]. We apply the image segmentation algorithm on both cameras in order to find the enclosing region of objects on each image. Then, the centroid of the segmented region is extracted on both cameras in order to perform stereo triangulation that provides the 3D position of the object's centroid, which is the target position to execute the action.

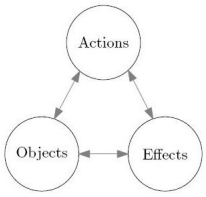
The effects are measured as differences in object attributes before and after the action is performed. We measure: (i) the magnitude of the object displacement relative to its initial position, (ii) the orientation angle of the displacement, and (iii) the contact between objects, computed from the intersection between the segmented regions in both images.

As the robot itself is not mobile and the arm has a specific action range, each $a_i \in A$ can be performed when the object is located in a specific action space. This is learned by exploratory manipulation of objects in various locations, and measuring and k-means clustering [13] (using Matlab) of the effects (e.g. displacement) in three clusters. The lowest displacement values correspond to the object not being reached, and the medium ones to where the action cannot be performed fully. The cluster with the highest values (i.e. effects are consistent) determines the action space, which will be enforced by the introduction of (logical) rules.

The real setting, with the iCub manipulating objects in over 100 scenarios, was used as a proof-of-concept demonstrating that our approach can be used as well in a physical setting with just minor adjustments. However, because of practical constraints regarding the availability of the iCub, most of the results we report on were obtained in simulation.

IV. Affordance-based Models

Affordance models, based on a concept introduced by J. J. Gibson [7], are typically used for robotic *imitation learning* settings [14], [17]. Affordances capture *action opportunities* to structure the environment. They define the relationships between the robot and the environment through the robot's available sensing and motor capabilities [17], e.g. a cup is handled in a different way than a ball. To achieve this task, affordances model the correlations between the set of *objects* and their *properties* as being detected by the robot sensors: $O = \{o_1, o_2..., o_n\}$, the repertoire of *actions* available to the robot, $A = \{a_1, a_2..., a_n\}$, and the *effects* of performing those actions $E = \{e_1, e_2..., e_n\}$ as detected by the sensors as changes in object features. A generic affordance model is depicted in Figure 2. We will describe learning of, and using the, affordance models in a one-object setting using a BN.



Inputs	Outputs	Function
(O, A)	E	Effect prediction
(O, E)	A	Action recognition/planning
(A, E)	O	Object recognition/selection

Fig. 2. Affordances: relations between objects, actions, effects [14], [17].

A. Learning Affordances for One Object

When the robot learns the action space for each $a_i \in A$, it also gathers data for O and E. The subset of the data in the action space is used to learn the affordance model to represent the environment. For this we use a BN [19], a probabilistic graphical model, to encode the dependencies in the affordance model, using a two-step approach similar to [17], [14]. In a first phase we induce the connection structure of the BN, using the K2 algorithm (using Matlab). This is computationally faster than the alternative MCMC approach [17]. K2's greedy nature may generate an incomplete structure which can easily be corrected using domain knowledge. The resulting BN structure of the affordance model is depicted in Figure 3. In a second step, similar to [17], [14], the parameters (i.e. the probabilities) of this BN are learned. In Section V we will elaborate on this step; at this point the robot has obtained the BN representing a single object affordance, which it can now use to reason about its interactions with the environment.

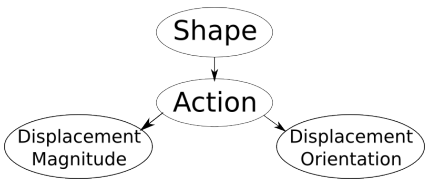


Fig. 3. BN representing affordances for the one-object setting.

B. Using the Affordance Model

Figure 2 shows the three possible uses of the affordance model, in which one of the three feature sets (O, A, E) can be computed by providing the other two. We will focus on the task of action prediction or recognition, where the robot observes O and E and has to *infer* A, the action that was performed. This is useful in imitation learning: the robot observes a human manipulating an object (and properties), observes the effects of the demonstration and tries to infer the action to imitate the effects in terms of its own action repertoire (obviously different from that of the human). Action prediction is also a basic step in *planning*, as the robot knows the effects it wants to achieve and tries to find the sequence of actions needed. The experimental section will address this task—we will show how our approach performs action prediction in the context of single-action planning. In order to perform *action recognition*, we use the previously learned BN representing the affordance model and compute the *maximum a posteriori probability* (MAP) estimate: $\arg\max_A P(A|O, E) = \arg\max_A \frac{P(A,O,E)}{P(O,E)}$, with the observed values for O and E.

V. Relational Models for Affordances

In this section we describe our main contribution: the extension of the previously introduced one-object affordance models employing BNs to more general settings using a probabilistic relational model. In these, manipulation skills can involve multiple objects, object interactions occur while manipulating and different behaviors are required depending on the spatial relations between objects. These relations include e.g. *relative distance* between objects and their *angle of orientation* and *contact*. Our multi-object setting requires first-order logic to capture—and generalize over —the (spatial) relations in the domain, probabilistic information to deal with uncertainty in perception and action and learning to induce the affordance models from interaction with the environment. Through the use of *variables*, i.e. place-holders for individual objects, in the relational representation, the models are able to generalize to arbitrary numbers of objects in the scene. That is, we derive a relational knowledge representation model only by taking into account the single and the two-object affordance model (which includes relational features between two objects). This model can then be applied in any multi-object scene, as shown in the experiments section (e.g. to be evaluated with 6 objects).

A. Probabilistic Programming Languages

A *probabilistic programming language* (PPL) is a programming language specifically designed to efficiently describe and reason with probabilistic relational models. For representation of, and inference using, the multi-object affordance model, we utilize the state-of-the-art PPL ProbLog [5], which is a probabilistic extension of the well-known Prolog logic programming language. To describe a probabilistic relational model in ProbLog, one writes a program which consists of a set of probabilistic facts and a set of logical rules (which express *domain knowledge* and *constraints*).

Let us illustrate this briefly with an example with causal-probabilistic logic (CP-logic)[22] style syntax in ProbLog, where predicates and atoms start with a small letter while variables start with a capital letter. To model the shape of an object being randomly chosen from a set of two predefined shapes (i.e. the shape can take exclusively one of the two values with a probability of $\frac{1}{2}$) one would write the clause:

$\frac{1}{2}$ :: shape(cube); $\frac{1}{2}$ :: shape(cylinder) ← true.

Formally, a CP-logic clause is a statement $\forall x(A_1 : p_1) \vee ... \vee (A_n : p_n) \leftarrow \phi$, where ϕ is a universally quantified first-order formula of some tuple of variables x, and A_i are atoms containing variables in x, such that for each x, ϕ causes at most one of the A_i to become true; A_i becomes true with probability p_i. [22] If $x = \emptyset$ the clause is ground, and assigning values to the variables in x is called grounding. [22] Now we can generalize our example over objects:

$\frac{1}{2}$ :: shape(O, cube); $\frac{1}{2}$ :: shape(O, cylinder) ← obj(O).

variable O is universally quantified over the set of all objects.

We can augment our model with logical rules to represent background knowledge, for example:

dispMag(O, 2) ← obj(O), action(O, push).

which models that the displacement magnitude of any object takes the value "2" when that object is pushed. Formally, a logical rule is a deterministic clause, where ϕ causes some atom A with probability one, $(A : 1) \leftarrow \phi$ [22].

Now that we have seen the building blocks, we want to have a general ProbLog model derived from the one (and later two) object affordances, which are modeled with BNs. As explained in more detail in [22], any BN can be modeled by constructing A_n for every node n of the BN, with parents $q_1, ..., q_m$, domain $\{v_1, ..., v_k\}$ and conditional probability table giving probabilities $p_1, ..., p_k$ for $q_1 = w_1, ..., q_m = w_m$, rules of the form: $(A_n(v_1) : p_1 \vee ... \vee A_n(v_k) : p_k) \leftarrow A_{q_1}(w_1) \wedge ... \wedge A_{q_m}(w_m)$ for all parent values in their domain.

Using the BN model in Figure 3 with previously learned parameters, part of the relationship between the action and the object displacement magnitude is modeled as:

0.03 :: dispMag(Obj, 1); 0.22 :: dispMag(Obj, 2);
0.25 :: dispMag(Obj, 3); 0.5 :: dispMag(Obj, 4)
← action(Obj, push).

When an object is pushed, its displacement magnitude takes one of the 4 possible values with the specified probability.

We can now model the setting as a ProbLog program with clauses c_i that can be probabilistic facts with probability p_i or logical rules (i.e. $p_i = 1$). Formally, a ProbLog program $T = \{p_1 : c_1, ..., p_n : c_n\}$ defines a probability distribution over logic programs $L \subseteq L_T = \{c_1, ..., c_n\}$ as $P(L|T) = \prod_{c_i \in L} p_i \prod_{c_i \in L_T \setminus L}(1 - p_i)$. Once the program describing the probabilistic relational model is defined, several inference methods are available for computing the probabilities of a user's query. This means asking for the success probability $P(q|T)$ of a query q, which is the probability that q has a proof given the distribution over logic programs [5].

Section IV mentioned that we need to learn the parameters of the affordance model BN. For a single or small predefined number of objects this can be done as in [14], [17] but this will not generalize over any number of objects or deal with partial observations (i.e. not all of O or E observed, e.g. faulty sensors). ProbLog supports learning in this more general setting through *learning from interpretations* (LFI) [8]. LFI uses the previously learned structure encoded as a ProbLog program $T(p)$ for the parameters $p = \langle p_1, ..., p_n \rangle$, and our gathered training data $D = \{I_1, ..., I_M\}$, I_i the data from instance i, to compute the maximum likelihood parameter estimation: $\hat{p} = \arg\max_P P(D|T(p)) = \arg\max_P \prod_{m=1}^{M} P_w(I_m|T(p))$, thus obtaining the probability parameters of the (relationally encoded) BN.

B. Learning Two Object Models

For multi-object scenes we introduce, in addition to the O and E features detected for the single object case, two spatial relational object properties: the *relative distance* between two objects and the *relative orientation* of one with respect to the other, and one relative effect: whether there is contact between them. The robot first explores the action space for two objects as seen in Figure 4 (we ran about 600 scenarios) to learn an affordance model. In the two-object setting, we define the *main* object to be the one the robot acts upon, and the *secondary* object the other object in the scene, which may interact with the main one through the robot's actions. Both are arguments of the robot's hand motion, and the action is defined over any secondary object:

action(OMain, A) ← handMotion(OMain, _, A).

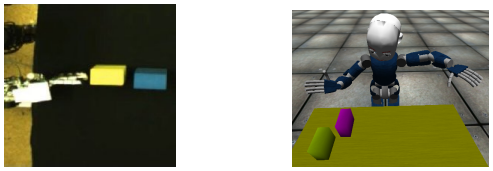


Fig. 4. Real(l) and simulation(r) screenshot of the two object setting.

From the data a grounded BN (structure and parameters) was learned in a similar manner as for the single object case in Section IV. We use this BN in ProbLog while generalizing over the number of objects by introducing variables for objects, resulting in the non-grounded BN in Figure 5.

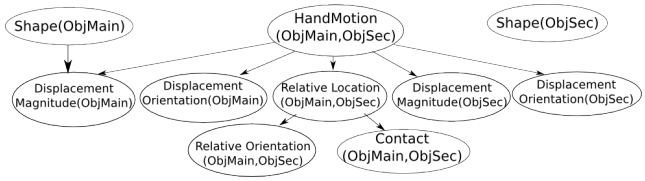


Fig. 5. Non-grounded BN for two-object interaction.

The single object BN (Section IV) can be reused by generalizing if there is no object interaction. In the displacement magnitude example, this is the case if the two objects are the furthest away (initial relative distance "4"). The effects on the main object are the same as in the one-object case:

0.03 :: dispMag(OMain, 1); 0.22 :: dispMag(OMain, 2);
0.25 :: dispMag(OMain, 3); 0.5 :: dispMag(OMain, 4)
← handMotion(OMain, OSec, push),
initRelDist(OMain, OSec, 4).

Logical rules are added for defining background knowledge,

which the robot would take longer to learn by itself otherwise. In the case above, the secondary object would not move (lowest displacement: "1") if the objects are far away:

$$\texttt{dispMag}(\texttt{OSec}, 1) \leftarrow \texttt{handMotion}(\texttt{OMain}, \texttt{OSec}, \texttt{push}), \\ \texttt{initRelDist}(\texttt{OMain}, \texttt{OSec}, 4).$$

Logical rules can also be used to enforce constraints in the setting, e.g. if two objects are close together a grasp should not be performed (the secondary object would interfere with the hand). Rules can also be added for enforcing the action space or encoding relations not caught by structure learning.

The new model generalizes over specific objects and can be applied in arbitrary scenes. In contrast to a standard BN, we do not have to encode all relations between all objects and generic rules are applied to arbitrary objects, so the number of parameters to be learned can be exponentially lower.

C. ProbLog Inference for Action Recognition

The ProbLog model can now be used by the robot for probabilistic inference. Here we are interested in action prediction (finding A given O and E). This means calculating a MAP estimate: $\arg\max_{A,ObjMain} P(A_{ObjMain}|O,E)$. In practice one just needs to do inference using the ProbLog program to obtain the required probability. Another advantage is that if a sensor fails and a feature value is missing, ProbLog is able to marginalize over the missing variables and find the required probabilities and predict A nonetheless.

The example below illustrates one such case of inference:

Detected O: o1=*rect*, o2=*square*, Relative Dist.=*1*

Desired E: Displacement o1=*4*, Displacement o2=*2*
Displ.Orientation o1=*NNE*, Displ.Orientation o2=*NNW*, Contact=*1*

Predicted A: push on o1: 7.7%, tap on o1: 0%, grasp on o1: 2.0%
push on o2: 90.1%, tap on o2: 0.2%, grasp on o2: 0.0%

In this case, the action predicted is a `push` on $o2$.

VI. Evaluation and Results

We want to show that the SRL model obtained from two-object interactions can be used in a general multi-object setting with comparable results to a BN model trained specifically for that number of objects. This will also show that interactions between three or more objects need not be explicitly considered because the relevant dependencies can be captured by pairwise interactions, and any additional influence of these interactions is negligible. We do this in the context of single-action planning where the robot needs to pick the object to act on and the best action to match some required effects. We want to find out: (i) Can the robot pick the right object? (ii) Can it pick the right action? (iii) Can it pick the right action on the right object? (iv) How does this compare with the BN approach?

We use a six-object setting to investigate if our SRL model is able to generalize from two-object interactions. The objects are placed in front of the robot on the table. Three objects are always in the field of action of the robot, though the robot might not be able to perform all the actions on every object as this might violate some rules (e.g. action space for that action, interference with the hand from nearby objects). The other three objects are placed behind these and might interact with them when performing an action. All the objects are randomly placed within certain margins and have a random shape. One such placement is shown in Figure 6(1). This is similar to the shelf setup, where the objects at the back are "in the shelf" while the ones in the front need to be arranged. In this setting, we execute all possible actions with the iCub to get real-world matching effects. Given these effects and the object properties, we predict the action and compare it against the ground truth action we performed.

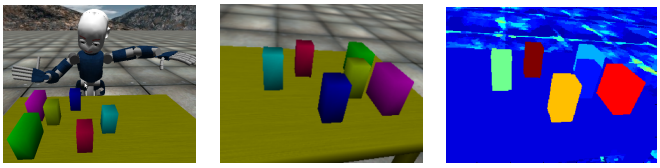


Fig. 6. Six objects(1), left eye image(2) and its segmented objects(3).

Our SRL model consists of the *generalization of the two-object interactions* described by the BN in Figure 5 and the following rules (one of each type): (i) *generalization from one object*: if the main and secondary object are far away, use the displacement orientation from the one-object case for the main object, (ii) *background knowledge*: if the main and secondary object are far away, displacement of the secondary object is lowest, (iii) *constraint enforcement*: grasp is only allowed if object distance is not lowest. Experiments will show that just this can give good results. Adding more rules increases the prediction rate, but inference will take longer.

To investigate single-action planning, consisting of action and main object prediction, we ran 200 experiments against which we tested our SRL model. For the BN model we split the same data into two sets and used one for training and one for testing. We did this six times and averaged the results, since the results vary because the number of experiments is relatively small given the high number of nodes.

We do color segmentation, illustrated in Figure 6 (2, 3) for the left eye, and then 3D object localization using stereo vision on the scene to find out the position of each object.

The object properties are: shape, relative distance and orientation between each pair of objects. After an initial domain analysis, they were discretized as follows: the relative distance in 4 clusters separated at $6cm$, $10cm$ and $14cm$; the orientation angle in 8 clusters of $45\,^{\circ}$. Every possible $a \in A$ is executed and recorded. For calculating the effects, the scene is segmented again, and we compute the displacement and its angle orientation, which are similarly clustered, with the displacement clusters separated at $1cm$, $3cm$ and $5cm$. On a given image, the contact feature relies on the area of intersection between the convex hull of the two segmented areas. The intersection is normalized in two ways: (i) with respect to the minimum object area in order to remove image size dependencies and (ii) with respect to the distance of the farthest object in order to remove depth dependencies. The normalized contact feature is averaged over both cameras.

We use the O and E data values and the SRL model to predict the action and main object by calculating $\arg\max_{A,ObjMain} P(A_{ObjMain}|O,E)$ and compare them to

the real action-object pair. Similarly, we obtain the best predicted object to act on by summing over all possible actions in the above formula, and the best predicted action by summing over all possible main objects. To have a baseline to compare our SRL approach to, we learn the six-object BN model. In this model, the action node has 9 possible values (3 actions for each of the 3 reachable objects), and we compute the MAP estimate to find out the predicted object-action duo. The comparison of the two approaches is shown in Table I.

TABLE I

ACTION PREDICTION IN SIX-OBJECT SCENARIOS.

	Prediction Task	Total exp.	Success	Pct.
ProbLog Model	$ObjMain$	200	149	74.5%
	A	200	137	68.5%
	$ObjMain$ and A	200	116	58%
BN Model (avg over 6 train/test sets)	$ObjMain$	100	67	67%
	A	100	69.2	69.2%
	$ObjMain$ and A	100	67	67%

Because each object-action combination is modeled by a different value in the action node of the BN and the relatively low number of training examples, the BN predicts the object-action duo and its individual components with almost the same rate. The approaches have comparable results; the SRL is slightly better at predicting the object to act on, while the BN is slightly better for the object-action duo. However, the SRL model can be used without being changed for any number of objects, while the BN needs to be learned again and can get very big (a six-object setting BN already has 65 nodes). In addition, the SRL enables transferring structural parts of the model (e.g. abstract action-effect rules) to similar domains with more, less or other types of objects. If we want improved accuracy tailored to a setting, the SRL model can also be trained with data from that setting using LFI.

We also looked at some learning statistics for our SRL model (summarized in Table II): i) the confidence of the predictions (i.e. the value of $P(A|O, E)$) and (ii) the number of correct effects produced by an incorrect predicted action.

TABLE II

LEARNING STATISTICS OF THE SRL AFFORDANCE MODEL.

Prediction Result	Statistic	Pct.
Success	$ObjMain$ confidence	73.1%
	A confidence	90.1%
	$ObjMain$ and A confidence	62.6%
Failure	Correct effects	67.2%

We see that the confidence of predicting an action is very high, while (as expected) that of the object-action duo is lower. When our prediction is wrong, executing the action still manages to produce about $\frac{2}{3}$ of the 27 effects correctly, sometimes a good compromise in complex scenarios.

VII. CONCLUSION AND FUTURE WORK

We have presented an approach for robotic affordance model learning in a probabilistic relational setting. Moving to multi-object scenes requires expressive representation schemes to generalize over specific spatial configurations of objects and dealing with uncertainty and partial knowledge about the environment. We showed that a relational extension of the affordance models of two object interactions can be used for modeling a multi-object scene with success.

Future work will study imitation learning, involving recognizing low-level "atomic actions" of one or two objects in a multi-object scene using the learned models. We also want to use affordance models for *planning* of manipulation strategies, where a task consists of sequences of actions and the robot learns a high-level manipulation strategy. The long term goal is to go towards an autonomous shelf sorting capability of the humanoid robot, as in Section I.

REFERENCES

[1] C. M. Christoudias, B. Georgescu, P. Meer, and C. M. Georgescu. Synergism in low level vision. In *In International Conference on Pattern Recognition*, pages 150–155, 2002.

[2] S. Coradeschi and A. Saffiotti. An introduction to the anchoring problem. *Robotics and Autonomous Systems*, 43(2-3):85–96, 2003.

[3] L. De Raedt. *Logical and Relational Learning*. Springer, 2008.

[4] L. De Raedt and K. Kersting. Probabilistic inductive logic programming. In *Prob. Ind. Log. Progr.*, pages 1–27, 2008.

[5] L. De Raedt, A. Kimmig, and H. Toivonen. Problog: A probabilistic prolog and its application in link discovery. In *IJCAI*, pages 2462–2467, 2007.

[6] R. E. Fikes and N. Nilsson. STRIPS: A new approach to the application theorem proving to problem solving. *Artificial Intelligence*, 5(2):189–208, 1971.

[7] J. J. Gibson. *The Ecologial Approach to visual perception*. Boston: Houghton Mifflin, 1979.

[8] B. Gutmann, I. Thon, and L. De Raedt. Learning the parameters of probabilistic logic programs from interpretations. In *ECML*, 2011.

[9] J. Hertzberg and R. Chatila. AI reasoning methods for robotics. In B. Siciliano and O. Khatib, editors, *Handbook of Robotics*, pages 207–223. Springer, 2008.

[10] D. Jain, L. Mösenlechner, and M. Beetz. Equipping robot control programs with first-order probabilistic reasoning capabilities. In *ICRA*, pages 3626–3631, 2009.

[11] V. Krunic, G. Salvi, A. Bernardino, L. Montesano, and J. Santos-Victor. Affordance based word-to-meaning association. In *ICRA*, 2009.

[12] T. Lang and M. Toussaint. Approximate inference for planning in stochastic relational worlds. In *ICML*, pages 585–592, 2009.

[13] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[14] M. Lopes, F. S. Melo, and L. Montesano. Affordance-based imitation learning in robots. In *IROS*, pages 1015–1021, 2007.

[15] M. Lungarella, G. Metta, R. Pfeifer, and G. Sandini. Developmental robotics: A survey. *Connection Science*, 15:151–190, 2003.

[16] G. Metta, G. Sandini, D. Vernon, L. Natale, and F. Nori. The iCub humanoid robot: an open platform for research in embodied cognition. In *Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems*, 2008.

[17] L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor. Learning object affordances: From sensory-motor coordination to imitation. *IEEE Transactions on Robotics*, 24:15–26, 2008.

[18] U. Pattacini, F. Nori, L. Natale, G. Metta, and G. Sandini. An experimental evaluation of a novel minimum-jerk cartesian controller for humanoid robots. In *IROS*, pages 1668–1674, 2010.

[19] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[20] F. Stulp and M. Beetz. Combining declarative, procedural, and predictive knowledge to generate, execute, optimize robot plans. *Robotics and Autonomous Systems*, 56:967–979, 2008.

[21] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[22] J. Vennekens, M. Denecker, and M. Bruynooghe. CP-logic: A language of causal probabilistic events and its relation to logic programming. *Journal of Theory and Practice of Logic Programming*, 9(3):245–308, 2009.

[23] F. Wörgötter, A. Agostini, N. Krüger, N. Shylo, and B. Porr. Cognitive agents – a procedural perspective relying on the predictability of object-action complexes (OACs). *Robotics and Autonomous Systems*, 57:420–432, 2009.