4th International Conference on
Development and Learning and on Epigenetic Robotics
October 13-16, 2014. Palazzo Ducale, Genoa, Italy

Bootstrapping paired-object affordance learning with learned single-affordance features

Emre Ugur, Sandor Szedmak, and Justus Piater

Intelligent and Interactive Systems, Institute of Computer Science,
University of Innsbruck

***Abstract*— The aim of this paper is to propose a system where complex affordance learning is bootstrapped through using pre-learned basic-affordances as additional inputs of the complex affordance predictors or as cues in selecting the next objects to explore during learning. In the first stage, the robot learns affordances in the form of developing classifiers that predict effect categories given object features for different discrete actions applicable to single objects. These predictions are later added to robot's feature set as higher-level *affordance features*. In the second stage, the robot learns more complex multi-object affordances using object and affordance features. We first applied our idea in an artificial interaction database which includes discrete actions, several manually coded object categories, and actions effects. Finally, we validated our bootstrapping approach in a real robot with poke and stack actions. We expected to obtain higher performance with *affordance-features* especially in small training datasets as the object-robot-environment dynamics should have already been partially learned and encoded in affordances. The experiment results showed that complex affordance learning significantly speeds up with predictors that are bootstrapped with *affordance-features* compared to predictors that use low-level features such as shape descriptors. We also showed that by actively selecting the next objects and by increasing the diversity of the training set using a distance measure based on learned single-object affordances, the effect of bootstrapping can be further increased.**

I. Introduction

This study is part of a research effort where a robot system gradually develops skills and competencies in subsequent stages of development, similar to human infants. In our previous work, we showed that similar to human infants who learn a set of actions by the age of 7 months such as grasp, hit and drop [1], a robot could also self-discover a number of behavior primitives such as push, grasp and release by interacting with objects using its crude 'reach' action and grasp reflex, and observing the changes in its tactile perception [2]. Next, we showed that similar to infants who learn object dynamics after 7-9 month of age, our robot could learn affordances in an unsupervised way by first discovering the effect categories it could generate in the environment, and then by learning the mapping from the object features to the effect categories. After learning, the robot was shown make plans to achieve desired goals, emulate end states of demonstrated actions, monitor the plan execution and take corrective actions using the perceptual structures employed or discovered during learning. Finally, we showed that more complex actions that involve multiple objects (such as bring object 1 over object 2) can be taught to the robot through imitation using the structures developed in the previous stages with mechanisms inspired from parental scaffolding and motionese [**?**]. In the current study, we assume that a number of actions (such as push and stack) and effect categories (such as rolled and pushed), which were discovered in the previous stages as summarized above, are transferred to the next stage where complex affordances such as stackability are learned. We study how this complex affordance learning can be bootstrapped by use of learned simple affordances as (i) additional inputs in prediction, and (ii) in active selection of objects to explore next in an active learning setting.

One hallmark feature of bootstrapped learning is that learning problems stack in the sense that higher-level learners use as input attributes concepts produced by lower-level learners. These higher-level attributes should allow faster learning than if the higher-level concepts had to be learned from the lower-level attributes alone. The aim of this paper is to propose a learning system where a developmental robotic system benefits from bootstrapping where learned simpler structures (affordances) that encode robot's interaction dynamics with the world are used in learning of complex affordances. In detail, our robot learns the affordances of single objects and uses these affordances as additional features in the next stages of development where paired-object affordances are discovered. The use of learned similarities in the form of affordances are expected to bootstrap the learning in the next stages.

Our approach can be explained by the following intuitive example: Let us assume that the robot learned rollability affordances of the objects in the first development stage, and can now predict the rollability based on object shape properties. In the next stage, robot learns a more complex affordance such as stackability from two sample interactions where it observes that stacked two balls tumble over and stacked two boxes pile up. The robot, trained only with those two stacking interactions, can find a correspondence between stackability and rollability. Then, even if the robot does not have any stacking experience with cylindrical objects, it can make better predictions for stackability depending on the roll orientation (and affordance) of the cylinders.

In the context of robot affordance learning research, paired-object affordance learning has not been studied ex-

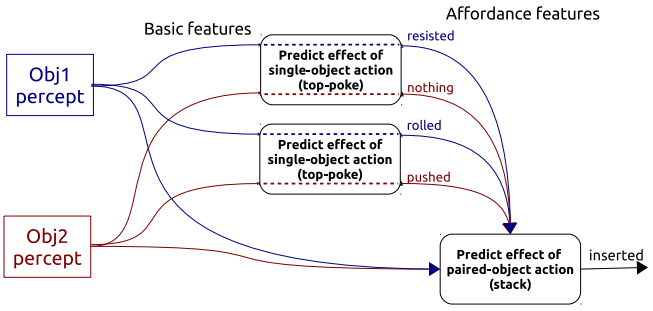

Fig. 1. Learning and prediction of action effects using *basic-features* (visual features) are shown with solid lines. The learned *affordance-features*, i.e. predicted effects, can be further used as input to classifiers which predict action effects of object pairs, i.e. in learning paired-object affordances.

tensively with exceptions of [3] where 'tool objects' are interacting with other objects, and [4] where two-object relational interaction models were directly learned. However none of these studies attempted to bootstrap their robot-object-object interaction dynamics with previously obtained skills and affordance knowledge.

Our method summarized in the next section is validated with an artificial interaction dataset that includes rich set of objects and interactions in Section III, in real world with robot experiments in Section IV.

II. Method

Learning of affordances corresponds to learning the relations between objects, actions and effects [5]. In this study, affordances are acquired through learning to predict what type of effects, i.e. discrete effect categories, can be generated given discrete robot actions and continuous object properties. To achieve this, we simply train a classifier for each action, which takes object features as input and predicts the effect category.

A. Object features

Here, we distinguish two different sets of object features. The first set includes hand-coded basic general-purpose features, computed from visual perception[1] with no explicit link to robot's actions. These may include standard features used in literature related to size, shape and local distance properties of the objects. The second set of features are acquired through interaction and they correspond to the higher-level learned ones that are computed from basic features. They encode the dynamics between robot actions (A) and object response (effect, ε). The first set of features is called *basic-features* whereas the second that is learned through interaction is called *affordance-features* as the latter includes the relations between objects, actions and effects.

[1]In this paper, we limit ourselves with the features that can be captured by vision only. However, object properties such as object friction or weight plays an important role on object-robot interaction dynamics. Thus, exploratory actions that can be used to perceive such properties should be implemented in a full-fledged scenario.

The straightforward approach to learn effect prediction is to train a classifier c for each action a that takes *basic-features* as input:

$$c^a_{\text{basic}}(\textit{basic-feat}) \rightarrow \text{effect}$$

whereas we propose to speed up learning of complex effect prediction using *affordance-features* that are computed using the learned basic effect prediction:

$$c^a_{\text{complex}}(\textit{basic-feat}, c_{\text{basic}}(.)) \rightarrow \text{effect}$$

which, in a *flat* form, corresponds to:

$$c^a_{\text{complex}}(\textit{basic-feat}, \textit{affordance-feat}) \rightarrow \text{effect}$$

Our approach is summarized in Fig. 1. The features shown with blue and red solid lines correspond to *basic-features* and action predictions based on these features give rise to *affordance-features*. The dashed lines correspond to *affordance-features*, that are learned in previous stages. The learning and prediction of complex affordances benefit from previously learned affordance features as shown in 'Predict effect of action k' predictor. Note that action k is considered to be a complex action as two objects are involved in execution.

In particular affordance features are represented as a vector of categorical variables, i.e. the list of the effect categories, predicted to be generated by single-object actions:

$$\textit{affordance-feat} = (\varepsilon^o_{a_1}, \varepsilon^o_{a_2}, ...)$$

where

$$\varepsilon^o_{a_i} = c^{a_i}_{\text{basic}}(\textit{basic-feat})$$

Complex affordance learning can be realized in different ways. In this paper, the action possibilities that are provided by two (or more) objects are considered to be complex. For instance, the effects created by a *stack* action (where the object is grasped and released over another one) is determined by the properties of both objects. We will use *affordance-features* (such as rollability, pushability, etc) and *basic-features* to learn and predict stackability affordances, and show that this learning significantly speeds up with predictors that are bootstrapped with *affordance-features*.

B. Active object selection based on affordances

We claim that the bootstrapping effect can be further increased if the objects to be explored (and learned next) are selected intelligently. A learner which is provided with a rich set of qualitatively different objects in its initial phases of development can perform better compared to the ones trained with complete random objects. Thus, an online learning system actively selects the next object to maximize the diversity of the training set, and the learned single-object affordances will be used as 'high-level' similarity measures between objects in computing this diversity.

The next object is selected from the set of possible objects (PossObjs) by maximizing the total distance from the next object to the already explored objects (ExpObjs) as follows:

$$nextObj = \underset{o_1 \in \text{PossObjs}}{\arg\max} \sum_{o_2 \in \text{ExpObjs}} dist_t(o_1, o_2)$$

where $dist_t(o_1, o_2)$ is the distance between two objects in the space of affordances.

$$dist_t(o_1, o_2) = \sum_a (1 - \delta_{\varepsilon_a^{o_1}, \varepsilon_a^{o_2}})$$

where $a \in A_{\text{Basic}}$, and $\delta_{i,j}$ is Kronecker delta function.

III. Bootstrapping in Artificial Data

In this section, we report our bootstrapping results obtained from a manually prepared artificial database of objects and interactions. The set of objects include cylinders, boxes, spheres and triangular prisms in different orientations and with/without holes as shown in Fig. 2. The set of manually encoded actions and their effects are as follows:

- Actions: {side-poke, top-poke, front-poke, stack}
- Poke-effects: {pushed, rolled, toppled, resisted, nothing}
- Stack-effects: {piled-up, inserted-in, covered, tumbled-over}

When poked from different directions, hypothetically, different effects can be generated with these objects. For example, when poked from side, lying cylinders would roll away, boxes would be pushed, objects with holes in poke direction would not be affected as finger would go through the hole without any interaction, and the tall objects would topple down. The effect of stacking objects on top of each other depends not only on their shape but also on their relative size as well. For example, while 'inserted-in' effect would generated when a small box is stacked on a hollow cylinder, 'piled-up' effect would be observed when the box is larger than the opening on top of the cylinder. Based on these assumptions, we manually created a hypothetical set of rules that give the effect based on object categories and their relative sizes.

A. Basic and affordance features

The classifier trained with *basic-features* uses the following features for training (and prediction later):

$$\text{TS}_{\text{basic}} = \{(shape^{o_1}, shape^{o_2}, dim^{o_1}, dim^{o_2})\}$$

where $shape$ includes mean and variance of the normals of the lateral surfaces, and the direction of the hole if it exists; and dim encodes the object size in different axes.

The classifier trained with *affordance-features* uses the following features:

$$\text{TS}_{\text{aff}} = \{(\varepsilon^{o_1}_{*\text{-}poke}, \varepsilon^{o_2}_{*\text{-}poke}, dim^{o_1}, dim^{o_2})\}$$

where ε^o refers to the effects of the corresponding poke action on the object o.

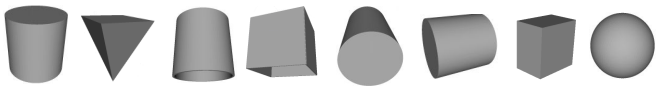

Fig. 2. The set of objects used in the artificial interaction database.

B. Bootstrapping Results

The performances of the classifiers trained with *basic-features* and *affordance-features* are provided in Fig. 3. We evaluated the classifiers by systematically changing the number of categories used in training set. For each number of categories, we trained 10 classifiers by selecting 5 objects of random size from each training category. To test these classifiers, we created test sets with random sized object from the remaining categories. Each bar corresponds to mean performance of these 10 classifiers. As shown, the prediction performance of both *basic-features* and *affordance-features* based classifiers improve by including more categories into the training set. We also included the performance of a category based predictor (which takes category index as input) to show the baseline. Because the categories used in training set are never included into test set, category-based predictors do fail independent of the training set size.

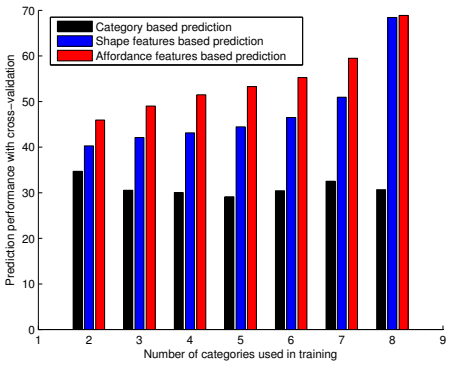

Fig. 3. The effect prediction performance of stack action obtained in artificial interaction database. The training of classifiers are done with the indicated number of categories with either shape features or affordance features. As shown, use of affordance features enable bootstrapping of the learning system.

These results show that because *affordance-features* already include properties related to object dynamics (pushability, rollability etc), classifiers that use these features have better performance especially for small training sets. With the increasing training set size, the effect of using high-level features is reduced as the *basic-features* classifier can also find the invariance related to stackability affordance with large dataset. Finding this invariance with small datasets is easier with *affordance-features* as they already include some properties of the agent-object-environment interactions.

IV. Bootstrapping in Real World

This section provides the details of the real world experiments where the effect of bootstrapping is analyzed. We first present the robot setup along with the details of robot's action

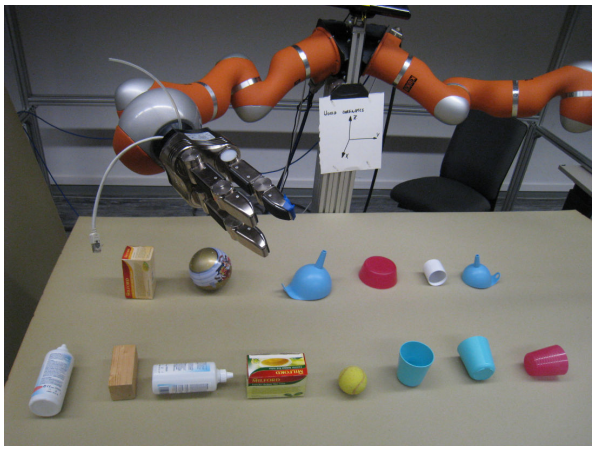

Fig. 4. The experiment setup. The environment includes one or two objects during experiments depending on the action type.

control and perception. Next, we showed that single-object affordances can be learned through interaction, and use of these acquired single-object affordances bootstraps paired-object affordance learning.

A. Robot system

The robot system employs a 7 DOF Kuka Light Weight Robot (LWR) arm, which is placed on a vertical bar similar to human arm in Fig. 4. A 3 fingered Schunk gripper is mounted on the arm to enable manipulation. For environment perception, Kinect sensor placed over the torso is used. The objects shown in Fig. 4 are used in learning single-object affordances as well as pairwise-affordances.

1) Object features: The robot's workspace consists of several objects and a table where the region of interest is defined as the volume over the table. First, the point cloud obtained from Kinect is transformed to robot's task space, then table is removed from the point cloud with a filtering along z-axis (see Fig 4), and finally objects are segmented based on depth information. Point Cloud Library normal estimation software is used next to compute a normal vector for each point of the object. The projection of each normal vector along each axis is separated, and histograms of normal vectors along each axis are computed. Using 8 bins for each histogram, $3 \times 18 = 54$ sized feature vector is obtained for shape related features. Note that in these experiments, an object is represented by a feature vector composed of only shape related features. Please see [5] for more details on histogram representation of normal vectors.

2) Robot Actions: The robot is equipped with a number of manually coded actions that enable single and paired object manipulation. The robot can 'poke' a single object from its side, front and top with *s-poke*, *f-poke*, and *t-poke* actions, respectively. It can also stack one object on the other using *stack* behavior, where it grasps the first object, move it on top of the other one and release it. The object position in world coordinate (shown in Fig. 4) is computed using the depth image of Kinect sensor. An inverse kinematic solver is used to compute the joint angles for initial and final

Fig. 5. The training set used for learning single-object affordances.

points defined in Cartesian space, and Reflexxes library [6] is utilized to generate smooth trajectories to achieve point-to-point movement. The action execution is as follows:

- Regarding to *poke* actions, the robot gripper is placed on one side of the object with $5cm$ distance with an orientation depending on the poke type. Two of the fingers are flexed to enable only the third finger to physically interact with the object (similar to index finger poking in humans). Next, the robot hand moves in the corresponding direction for $10cm$ towards the object and it is retracted after the poking is completed.
- Regarding to *stack* action, one object is grasped from above first by placing the gripper in a vertical orientation $10cm$ over of the object, then moving the wide-open gripper towards the object and finally enclosing it. Next, the gripper that carries the grasped object is repositioned over the second object in a vertical orientation again, and the object in the gripper is released over the first one by extending all the fingers.

3) Effect Categories: In the real world experiments, depending on the object(s) and the action executed, different effects were generated. When *poke* action was executed, the object was pushed, toppled over or rolled away depending on its shape. There was no effect in object state or robot's sensors if the robot finger went through the hole on the object. Finally, for *t-poke* action, all solid objects created resistance and obstructed gripper's movement that was detected using the force sensors. When *stack* action was executed, the objects in general piled up on top of each other if the object below provided a proper support (for example if it had a flat top surface). Depending on the existence of concave surfaces and holes, the released object was inserted in or hided the object below by encapsulating it. The released object also tumbled over due to the lack of stable support. Based on the above possibilities that we observed empirically, the sets of effect categories ($\mathcal{E}$) were set same as in previous section.

B. Experiment Results

1) Learning single-object affordances: The robot executed its poke actions on the objects (Fig. 4) placed in different orientations, and it collected 24 interaction instances for each poke action. The object shape features along with generated effect categories are stored for learning affordances. Support Vector Machine classifiers are used to learn the mapping between object features and effect categories. In order to analyze if the affordances for *poke* action are generalizable, we divide the interaction set into training and

Object from robot's view	Explanation	Front-poke prediction	Side-poke prediction	Top-poke prediction
	Lying cylinder	pushed	rolled	resisted
	Lying mug	rolled	pushed	resisted
	Cylinder wall	pushed	pushed	no-change
	Tennis ball	rolled	rolled	resisted
	Lying mug	pushed	rolled	resisted
	Wooden block	pushed	pushed	resisted
	Lying mug	pushed *	rolled	resisted
	Tea box	pushed	pushed	resisted
	Thin wooden block	pushed	toppled ***	resisted
	Thin wooden block	toppled **	pushed	resisted

Fig. 6. Robot's basic-affordance prediction on objects which are not included in the training set with the same orientations. Prediction fails in the examples with star (*), which are difficult cases to predict.

test sets with the deliberate purpose of distributing objects with same affordances into different sets. For each poke action, we trained a classifier using the objects given in Fig. 5. Then, we tested these classifiers by predicting the action effects on novel objects given in Fig. 6. As shown, the robot was able to detect the affordances of the object (in terms of effect prediction) correctly, except a small number of cases shown with stars (*), where the prediction also required perception and learning of material properties.

The trained three classifiers (for *s-poke*, *t-poke*, and *f-poke*) are transferred to the next stage and their predictions are used as high-level features to learn complex affordances.

2) Learning paired object affordances: In this section, the robot learns the paired-object affordances by exploring the two-object environments with its *stack* action. This learning is again achieved by training an SVM classifier that predicts the effect of the *stack* action given object features. Here we compare the prediction performance of the classifiers that are trained either with *basic-features* or *affordance-features*. Regarding to *basic-features*, normal vector histograms are used as we did in learning single-affordances in the previous subsection. Regarding to *affordance-features*, the list of effect predictions (provided by the classifiers transferred from the previous stage) for the poke actions are used.

The robot executed *stack* action with 18 pairs of random objects. A number of snapshots taken during these interactions are given in Fig. 7, where all the possible effects were observed with different object pairs. In each interaction, *basic-features* and *affordance-features* of both objects are computed and stored along with the observed effect category.

The classifier trained with *basic-features* uses the following features for training (and prediction later):

$$\text{TS}_{\text{basic}} = \{(shape^{o_1}, shape^{o_2})\}$$

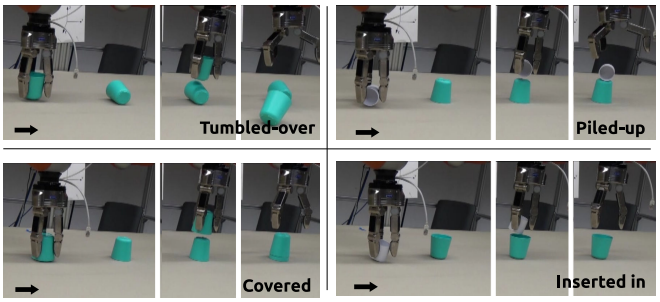

Fig. 7. Sample interactions observed during *stack* action execution.

and the classifier trained with *affordance-features* uses the following features:

$$\text{TS}_{\text{aff}} = \{(\varepsilon^{o_1}_{\textit{s-poke}}, \varepsilon^{o_1}_{\textit{f-poke}}, \varepsilon^{o_1}_{\textit{t-poke}}, \varepsilon^{o_2}_{\textit{s-poke}}, \varepsilon^{o_2}_{\textit{f-poke}}, \varepsilon^{o_2}_{\textit{t-poke}})\}$$

where $\{\}$ corresponds to the set operator.

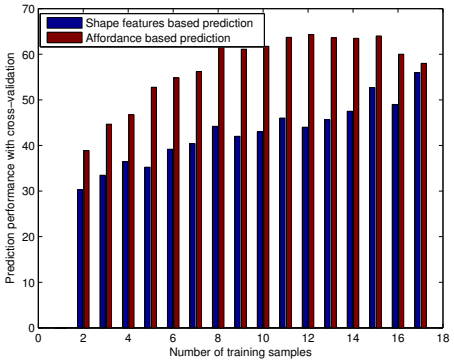

Fig. 8. The effect prediction performance of stack action that involves two objects. The training of classifiers are done with the indicated number of samples (interactions) with either shape features or affordance features. The initial high performance of *affordance-features* based classifiers demonstrates the advantage of using bootstrapping.

We evaluated the performance of these classifiers by systematically changing the size of the training set. For each training set size, we trained 10 classifiers using randomly selected samples. We tested each classifier using the remaining sample interactions. Fig. 8 gives these cross-validation results. Real-world experiment results are similar to the results obtained from the synthetic interaction dataset. As the *affordance-features* were obtained through interaction with the environment, they already encode the object-environment dynamics, which provides bootstrapping effect in learning multi-object affordances as shown. The *basic-features* are real valued larger sized vectors that encode object shape properties independent of robot-object dynamics. Thus they require more training data for learning. Additionally, *basic-features* are used in computation of *affordance-features*, so they contain the information to make predictions with the performance of affordance features. As shown in the figure, with increasing number of training samples, *basic-features* based classifier's performance indeed approached to the bootstrapped classifier's performance.

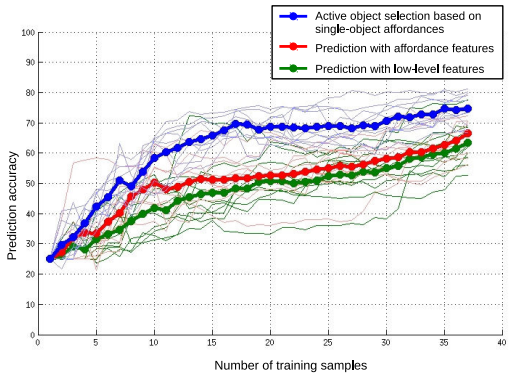

Fig. 9. The bootstrapping obtained by active selection of pairs of objects for learning of stack affordances.

V. Active selection of objects based on affordances

We used affordance distance measure defined in Section II.B, to maximize the diversity in training of paired-object affordance learning in an online learning setting. 83 objects and their poke and stack effects are used in this experiment (please see [7] for the complete list of objects). The object features are computed from Kinect depth image as in previous section, however, we used a human expert to label stack effect categories and it was not feasible to execute $83 \times 83 = 6889$ stack actions in the real robot. Fig. 9 shows the resulting performance of the basic-features and affordance-features based effect predictors trained with randomly selected objects, and of the effect predictors trained with the proposed active object selection strategy. Each type of predictor was trained 10 times, starting from a different random set of objects, and the thick lines correspond to the average accuracy for each predictor type. As shown, active selection of objects based on single-affordances provides a further bootstrapping effect in learning paired-object affordances. The best predictor was trained with affordance-features, but we observed that a similar performance was achieved even if it was trained with basic-features.

VI. Conclusion

Single-object affordances encode characteristics related to robot-object-environment dynamics as they are learned through robot's interaction with the objects. In this study, we showed that learned basic affordances can be used as additional features in order to bootstrap the next stage of development where complex paired-object affordances are learned. In our general model, affordances are used as 'additional features' for learning complex affordances, but in the experiments, in order to compare their independent performances, we used either only basic features or only affordance features. If an important basic action (such as top-poke) was unavailable, the affordance features, i.e. effect predictions for side-poke and front-poke actions, would have failed to predict insertability. Therefore, both channels should be used during learning and possibly a feature selection algorithm can filter out unnecessary channels.

While this work serves as one of the proof-of-concept application of the structural bootstrapping idea, we need to adapt advanced representations and learning methods (such as knowledge propagation framework of [8]) that can truly exhibit the real potential of this idea. We showed that this bootstrapping enabled the robot to speed up its learning particularly with small training data. Recently generative models have been proved to be effective in their ability in capturing object-action-effect dynamics, and in making predictions in different directions, for example in inferring the required actions to achieve desired effects given object properties [9], [4]. Particularly, hierarchical Bayesian networks directly encodes the desired structure and allows inference in several directions [10]. We discuss that our 'discriminative' model still provides powerful mechanisms as it can effectively map the continuous object feature and behavior parameter spaces to the corresponding effects [11] without any initial categorization of object properties as in [9], [4]. Furthermore, while bi-directional relations are not explicitly encoded in our system, we showed that our robot was able to make predictions in different directions, and made plans that involved sequence of actions on automatically selected objects[5].

Acknowledgements

This research was supported by European Community's Seventh Framework Programme FP7/2007-2013 (Specific Programme Cooperation, Theme 3, Information and Communication Technologies) under grant agreement no. 270273, Xperience.

References

[1] M. Asada, K. Hosoda, Y. Kuniyoshi, H. Ishiguro, T. Inui, Y. Yoshikawa, M. Ogino, and C. Yoshida, "Cognitive developmental robotics: a survey," *IEEE Tran. Auton. Mental Dev.*, vol. 1-1, 2009.

[2] E. Ugur, E. Sahin, and E. Oztop, "Self-discovery of motor primitives and learning grasp affordances," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 3260–3267.

[3] J. Sinapov and A. Stoytchev, "Detecting the functional similarities between tools using a hierarchical representation of outcomes," in *Proceedings of the 7th IEEE International Conference on Development and Learning*. IEEE, Aug. 2008, pp. 91–96.

[4] B. Moldovan, P. Moreno, M. van Otterlo, J. Santos-Victor, and L. De Raedt, "Learning relational affordance models for robots in multi-object manipulation tasks," in *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2012, pp. 4373–4378.

[5] E. Ugur, E. Oztop, and E. Sahin, "Goal emulation and planning in perceptual space using learned affordances," *Robotics and Autonomous Systems*, vol. 59, no. 7–8, pp. 580–595, 2011.

[6] T. Kroger, "Opening the door to new sensor-based robot applications – the reflexxes motion libraries," in *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2011, pp. 1–4.

[7] S. Szedmak, E. Ugur, and J. Piater, "Knowledge propagation and relation learning for predicting action effects," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014.

[8] S. Szedmak and J. Piater, "An active learning based sampling design for structural bootstrapping," Univ. of Innsbruck, Tech. Rep., 2014.

[9] L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor, "Learning object affordances: From sensory–motor maps to imitation," *IEEE Transactions on Robotics*, vol. 24, no. 1, pp. 15–26, 2008.

[10] E. Gyftodimos and P. A. Flach, "Hierarchical bayesian networks: A probabilistic reasoning model for structured domains," in *ICML WS on Development of Representations*, 2002, pp. 23–30.

[11] E. Ugur, E. Oztop, and E. Sahin, "Going beyond the perception of affordances: Learning how to actualize them through behavioral parameters," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.