Emergent structuring of interdependent affordance learning tasks using intrinsic motivation and empirical feature selection

Emre Ugur, *Member, IEEE,* and Justus Piater, *Member, IEEE*

***Abstract*—This paper studies mechanisms that produce hierarchical structuring of affordance learning tasks of different levels of complexity. Guided by intrinsic motivation, our system detects easy tasks first, and learns them in selected environments which are maximally different from the previously encountered ones. Easy tasks are learned from observed low-level attributes of the environment, and provide abstractions over these attributes. As learning progresses, the system shifts its focus and starts learning harder tasks not only from low-level attributes but also from previously-learned abstract concepts. Therefore, hard tasks are autonomously placed higher in the hierarchy if the easy task concepts are identified as distinctive input attributes of hard tasks. Use of abstract concepts allows hard tasks to be learned faster than learning them from scratch, i.e., from low-level perception only. We tested our system with the tasks of learning effect predictions for poke and stack actions using a dataset that includes 83 real-world objects. On the basis of a large number of runs of the method, our analysis shows that the hierarchical task structure emerged as expected, along with a consistent learning order. Furthermore, a significant bootstrapping effect in learning speed of the stack action was observed with the discovered hierarchy, albeit only when fully-learned poke actions were used from the beginning.**

***Index Terms*—intrinsic motivation, affordance, bootstrapping, discriminative learning, structure learning, transfer learning, re-use of concepts, prediction hierarchies, feature selection, diversity maximization**

I. Introduction

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. Consider an example where the robot learns stackability affordances, i.e. learns to detect if two given objects would stably stack on top of each other. In this case, the robot needs to explore a high-dimensional search space if it learns from low-level shape features of these objects such as curvatures of different sides. On the other hand, if the robot uses previously learned high-level abstractions, such as rollability, it would learn which objects can be stacked on top of each other by associating rollability and stackability from fewer examples, This is achieved because the robot already learns and encodes part of object-robot-environment dynamics in the higher-level attribute of rollability, and can re-use this attribute to bootstrap other related learning problems that share similar characteristics.

In this paper, we formulate the skill learning problem as learning inter-dependent affordances where a robot is expected to detect affordances of different complexities by learning the relations between objects, actions and effects. In our system the affordance detection is achieved by predicting what kind of effect would be created on an object given the visual properties of the object and the action applied. Following the re-use idea, an expert can design a system which learns simple affordances first, and gradually shifts its focus to more complex affordances taking advantage of the learned simple affordances. However in a life-long learning scenario where the objects in the environment are unknown and changing, and any arbitrary action can be added to the robot repertoire any time, the difficulty of affordances cannot be known in advance by the experts. A truly developmental agent should discover the best means to organize its learning strategy so that the skill re-use is most effective without such prior knowledge. In other words, the agent should simultaneously discover the learning order and the re-use structure of affordances.

Discovering the learning order of affordances means realizing an exploration strategy that shifts the focus of learning from simple to complex affordances. Since the robot learns from observations of the consequences of its actions on different objects, this exploration strategy practically corresponds to selecting the 'best' actions and objects in an active learning setting in a potentially open-ended system. For action selection, we used Intrinsic Motivation (IM) approach, which guides the robot with intelligent exploration strategies. IM is regarded as a set of active learning mechanisms for developmental robots, which enables efficient and effective learning in high-dimensional search spaces [1]. IM approach [2] in developmental robots [3] was inspired from curiosity based motivation mechanisms in human development, and has recently been effectively applied to cognitive robots where object knowledge is developed through self-exploration [4] and social guidance [5]. This approach adaptively partitions agent's sensorimotor space into regions of exploration and guides the agent to select the regions that are in intermediate level of difficulty. This is achieved by maximizing reduction in prediction error, in other words by maximizing the learning progress. In this paper, we propose to use this approach to guide the robot to explore different affordances by adaptively selecting the actions to execute, and updating the models of the affordance predictions based on the results of these actions.

E. Ugur is with Computer Engineering Department, Bogazici University, Istanbul, Turkey. J. Piater is with University of Innsbruck, Institute of Computer Science, Innsbruck, Austria.

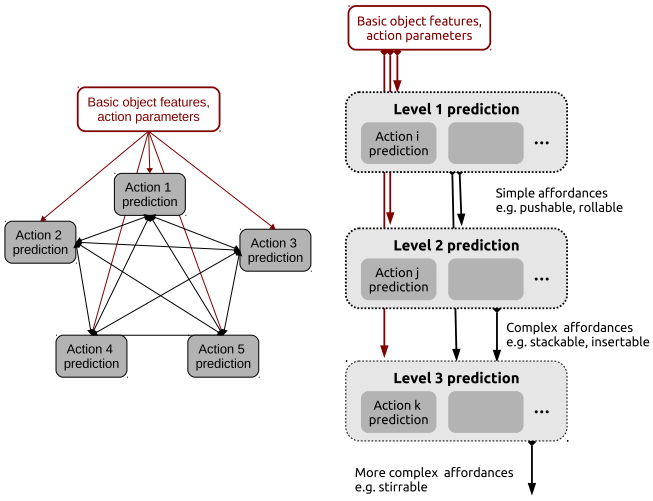
Through IM approach, we aim to achieve a developmental progression similar to those of infants in learning simple-to-complex skills and affordances.

In order to actively select which objects to interact with, on the other hand, we used a heuristic which maximizes the diversity of the objects used in training. In each learning step, the object, which is maximally different from the objects that are already used in previous learning steps, is selected with the aim to cover the variety in the object space.

Discovering the re-use structures of affordance corresponds to autonomously selecting the set of affordances that are useful in learning and predicting other affordances. Practically, this corresponds to deciding input/output links among affordance predictors. Our solution to this problem is inspired from Eleanor J. Gibson who was a developmental psychologist studying on mechanisms of affordance learning in biological systems. She argued that learning affordances is neither the construction of representations from smaller pieces, nor the association of a response to a stimulus. Instead, she claimed that learning is "discovering distinctive features and invariant properties of things and events" [6]. Learning is not "enriching the input" but discovering the critical perceptual information in that input. Following this idea, we identify distinctive inputs for each affordance predictor, and establish connection between output of an affordance predictor and input of another predictor only if the former affordance is discriminative in predicting the latter one. We formalized this through a relevant feature selection mechanism that selects the most relevant, non-redundant, and minimal set of inputs, which in turn can achieve maximal prediction accuracy.

In summary, we study the mechanisms that enable autonomous structuring of affordance learning tasks. Our system starts in a flat form as shown in Fig. 1(a) where output of each affordance predictor can be potentially used as an input of any other predictor. As learning progresses the robot actively updates these connections by selecting the most distinctive inputs of the corresponding action predictors. In detail, in each learning step, the robot selects the most 'interesting' action to perform based on Intrinsic Motivation, the most different object to explore based on diversity maximization, and updates the predictor of the corresponding action along with its distinctive inputs based on the observed effect on the object. We demonstrate these mechanisms, namely (i) the IM based selection of actions, and (ii) diversity maximization based selection of objects, and (iii) the use of the most distinctive inputs in affordance predictions, enable emergence of a hierarchical structure, similar to the one shown in Fig. 1(b). With this, the system autonomously discovers which tasks should be learned first, and which ones become simpler if expressed in terms of other tasks without any prior knowledge on the relative complexity of these tasks.

The rest of this paper is organized as follows. First, a summary of related work on affordances, re-use of learned concepts, and intrinsic motivation based robotics work is given in the next section. Next, representation of affordances, and mechanisms of active action, object and connection selection are detailed in Section III. In Section IV, the discovered learning order of predictors and evolution of the input/output connections are given along with the results on speed-up in learning complex affordances with the emerged structures. Finally, in Sections V and VI, we discuss how a number of assumptions can be addressed and how our system can be extended to handle more complex learning spaces.



(a) Flat prediction (b) Hierarchical prediction

Fig. 1. (a) shows a flat affordance learning structure, where the affordances are predicted based on low level object features, action parameters, and all other perceived affordances. (b) shows a simple hierarchical structure where simple affordance predictions can be used to detect complex affordances. This paper aims automatic discovery of such a hierarchical structure where the lower level predictors are learned first, and connected to the higher level predictors autonomously.

II. Related Work

Affordances, which were introduced by James Gibson in his ecological approach to visual perception [7], provide the agents a 'direct' means to perceive the action possibilities provided by the environment, and act on them. As detecting action possibilities and reasoning on them are crucial in robotics, affordances concept has been very influential in the last decade. Affordance research in robotics can be coarsely divided into two categories: learning affordances from passive observations or through robots own interaction with the world. The research in the first category does not involve the robot in active learning. For example Koppula et al. [8] studied learning of object affordances along with human activities using Markov Random Fields where the nodes encode the sub-activities and affordances; and the edges correspond to the learned relations between these components. Sun et al. [9] recently studied object-object interaction affordances, and learned motion models that represent interaction possibilities between pairs of objects from manually labeled video sequences. Myers et al. [10] labelled tool parts with multiple affordances and learns generalizable affordance detectors. Schoeler and Worgotter recently studied generalizing object affordances through effective transfer of part functionalities in real world objects [11]. The research in the second category, which involves learning affordances in the form of object-action-effect relations has been widely studied in robotics in

recent years [12, 13, 14, 15, 16]. While all different kinds of clustering and classification methods are used in literature, the standard learning setup includes a robot that actively explores the object in the environment. It interacts with objects using generally pre-defined set of behaviors, observes the effects generated by these behaviors, and learns the relations between the behaviors, behavior parameters, object properties and generated effects. Then, given a goal, the robot can chain effect predictions to find the sequence of behaviors, effectively generating a plan to reach that goal. Our approach in this paper belongs to the second category as we also study learning effects of actions of the robot on different objects using self-discovered effect categories that were transferred from previous robot interactions [13, 17].

Re-use of learned affordances and relational affordances have been addressed by only a few studies. Transfer of learned single-object affordances to bootstrap learning of paired-object affordances was studied by Moldovan et al. [18, 19]. The authors first used Bayesian Networks (BN) to learn (object, action, effect) relations from single and paired-object interactions, where relative position and orientation were encoded explicitly. Single-object action rules were transferred in learning paired-object action models when objects do not interact with each other. A relational knowledge representation model was derived from the BN affordance models, and later generalized to arbitrary number of objects. The system, which learned probabilistic rules for push and grasp actions in single and paired-object settings, were able to generalize its predictions to multi-object settings. Different from our model, Moldovan et al. focused on generalizing the rules that encode the position and orientation relations between objects; whereas our focus is to learn complex affordances that encode non-linear relations between arbitrary features of objects. In [19], the authors extended their system to continuous settings, however high-level shape primitives such as cubes and cylinders were pre-defined, whereas our system can discover such primitives from low-level shape features. Fichtl et al. also used predictions of action effects as inputs in predicting effects of other actions [20]. In a setting with 9 simulated actions, they used effect predictions of one or two actions in predicting effects of other actions. They showed that bootstrapping in learning speed can be achieved in the initial phases of learning especially if the learning problem is hard. This study is similar to ours as we also use of outputs of predictors as inputs of other predictors. However, our system can discover the input/output structure itself, and co-develop affordance predictors in an active learning setting that is guided by Intrinsic Motivation.

Intrinsic Motivation (IM) was first used as a 'manipulation drive' that explains why monkeys spend time on mechanical puzzles for long durations without any extrinsic reward [21], Activities that do not directly serve the goals of survival or material advantage such as play, curiosity, interest in novel stimuli and surprising events are said to be driven by intrinsic motivations [22, 23]. In order to achieve an open-ended development like infants, this mechanism has been heavily utilized in robotics in the last decade. Law et al. [24] realized a staged development with iCub that models an infant from birth to 6 months. iCub, through motor babbling driven with a novelty metric, started from uncontrolled motor movements, passed through several distinct behavioral stages, and achieved reaching and basic manipulation of objects, similar to the human infants. Ivaldi et al. [5] proposed a system where the iCub humanoid robot learned object properties by actively choosing among objects to explore, actions to execute and caregivers to interact. This socially guided intrinsic motivation framework [25] that combined robot's manipulatory actions with social guidance significantly increased object recognition performance, and could be directly used to increase speed of affordance learning. Hart and Grupen [26, 27] realized a longitudinal development, where a robot self-organized its sensorimotor space by assembling basic actions into hierarchical programs in a bottom-up way, and by learning to apply these programs in novel contexts in a top-down fashion. The staged learning of behaviors was guided by an intrinsic rewards mechanism that maximizes detection of and acting on affordances with the corresponding behaviors. Hart and Grupen's work was focused on behavior formation in a staged progression with mechanisms similar to accommodation and assimilation [28] through the so-called affordance discovery motivator with emphasis on closed-loop control programs as coupled dynamical systems. Other researchers used intrinsic motivation for autonomous acquisition of motor skills [29], and for autonomous selection of tasks to explore based on a measure of competence progress [30]. Please see Baldassarre et al. [22] for a recent Electronic Book (eBook) that compiled large number of interdisciplinary articles on Intrinsic Motivation within Neuroscience, attention, robotics, and social interactions research.

We previously studied intrinsic motivation and relevance based affordance learning in [31], and bootstrapping complex affordance learning with learned simple affordances in [32]. The method, discussions, and analysis of the system have been significantly extended in this paper as follows. First of all, the main components of our system, namely IM-based action selection, diversity-based object selection, and relevance based distinctive input selection are first time combined and analyzed in an integrated framework. Second the current work rigorously analyzes the proposed approach by running the algorithm large number of times with different configurations, and by providing the statistics obtained from these several runs. Finally, all material presented in the results section is new, and supported by more thorough discussions.

III. Active Learning of Affordances

A. Affordance representation

In this paper, affordances of an object (aff^o) correspond to the list of effects generated by the set of available actions:

$$aff^o = (\varepsilon^o_{a_1}, \varepsilon^o_{a_2}, ...)$$

where $\varepsilon^o_{a_1}$ is the discrete effect created on object o by action a_1. For example, a lying cylinder affords rollability when pushed from one side, pushability when pushed from another side, and liftability when grasped. The affordances of this cylinder are represented by the list of effects created with push and grasp actions, i.e. {rolled, pushed, lifted}.

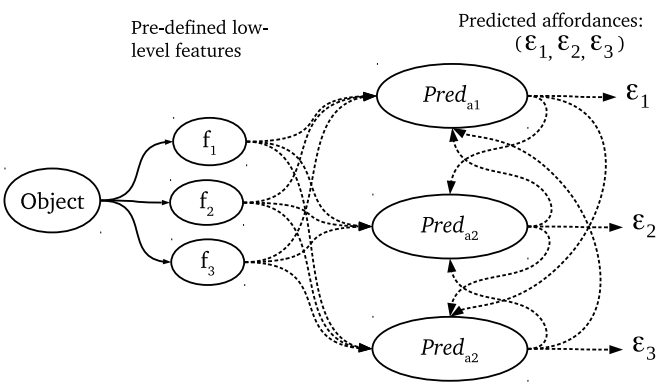


Fig. 2. Potential input/output links of the affordance predictors in a setup with three low-level features and three single-object actions.

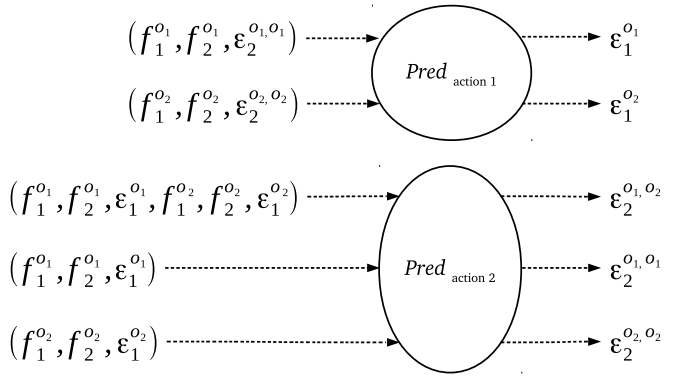


Fig. 3. Input/output of the affordance predictors in a setup with two low-level features, one single-object actions, and one paired-object action. In order to give a clear picture, we used a two action setting and showed only the important information.

A robot can predict the effect of an action on an object based on the visual properties of the object and/or through reasoning over how the object is affected by other actions. In this paper, we will call the first source of information as 'low-level features' of objects, and the second source as 'high-level features'. While the low-level features such as size and shape are previously defined by the programmer, the high-level features will be learned by the system. These features are used as inputs to $Pred$ operator that predicts the effect of action a_i on object o as follows:

$$\varepsilon^o_{a_i} = \textit{Pred}_{a_i}(o) = \textit{Pred}_{a_i}(\textit{feat}^o, \textit{aff}^o \backslash \varepsilon^o_{a_i})$$

Above, $\textit{feat}^o = (f^o_1, f^o_2..)$ corresponds to the low-level features. High-level features, on the other hand, are represented by $\textit{aff}^o$, which encodes 'other affordances', i.e. the effect predictions for other actions:

$$\textit{aff}^o \backslash \varepsilon^o_{a_i} = \{\textit{Pred}_{a_i}(o) | a_i \neq a_j\}$$

Actions that involve interaction with single objects and with pairs of objects are called single-object actions and paired-object actions, respectively. Likewise, affordances that are offered by single objects and pairs of objects are called single-object affordances and paired-object affordances, respectively.

Fig. 2 gives the general input/output structure for an affordance prediction system in a setup that includes three different types of pre-defined low-level features, and three single-object actions. Each predictor takes low-level features and the effect predictions of other actions as inputs. The links illustrated by the dashed lines correspond to only potential connections, which are selectively established by the learning system only if the corresponding links are necessary for predictions. While establishing the links, cyclic dependencies are not allowed because a predictor can produce an output only if all its inputs are available. In our previous work [31] we did not need to disallow cyclic dependencies because predicted features were attached to object identities, obliterating the need to re-predict. Note that outputs of predictors before initialization are fixed to non-existing effect categories (-1).

For simplicity, we described the effect prediction mechanism using actions that involve interactions with single objects. This prediction mechanism also supports actions that involve more than one object. In multi-object case, features and affordances of all the involved objects are used as input attributes of the corresponding action predictor. Fig. 3 illustrates the potential links in a setup with two low-level features, one single-object action, and one paired-object action. In paired-object actions, the predictor takes the following form:

$$\varepsilon^{(o_1,o_2)}_{a_i} = \textit{Pred}_{a_i}(\textit{feat}^{o_1}, \textit{feat}^{o_2}, \textit{aff}^{(o_1,o_2)} \backslash \varepsilon^{(o_1,o_2)}_{a_i})$$

Paired-object affordances, therefore, correspond to the collection of effects obtained from single-object actions for each object, and paired-object actions for each pair:

$$\textit{aff}^{(o_1,o_2)} = (\varepsilon^{o_1}_{a_1}, \varepsilon^{o_1}_{a_2}, ...\varepsilon^{o_2}_{a_1}, \varepsilon^{o_2}_{a_2}, ..., \varepsilon^{(o_1,o_2)}_{a_i}, \varepsilon^{(o_1,o_2)}_{a_j})$$

Multi-class Support Vector Machines (SVMs) with Radial Basis Function kernels and optimized parameters are used to learn these predictors [33]. Although we used a batch version of SVM as implemented in LibSVM library [34] for practicality, incremental versions, which are more suitable in active learning setups, were also shown to provide similar performance [35]. Our focus in this paper is how to discover the connection structure and learning order of these predictors. Therefore the details of prediction mechanisms have minor importance, and one can replace SVMs with their favorite classification method as long as the following active learning principles are preserved.

B. General learning approach

We followed an active learning approach to train the complete prediction system. This learning is achieved in episodes, whose major steps are summarized in Fig. 4. In the first step, the action that has the highest learning progress is selected for exploration. Next, a number of objects that are maximally different from the previously selected objects are selected. Based on the observed effects on these objects, the effect predictor of the corresponding action is updated by updating the SVM classifiers described in previous paragraph. During this update, the input links for this predictor are also found and these established connections are registered to the system.

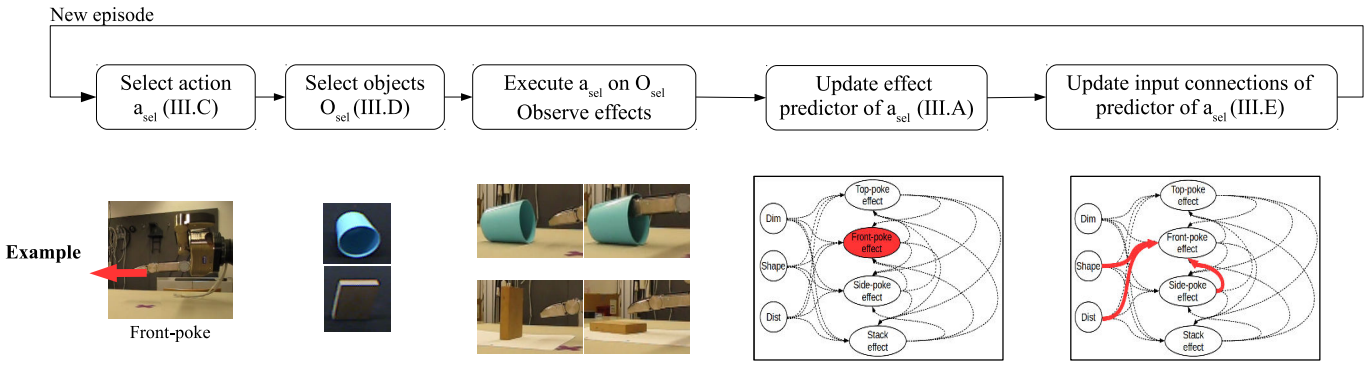


Fig. 4. Summary of one learning episode.

C. Active selection of actions

The first step of each episode is to select which action to explore next. As mentioned in Section I, we used Intrinsic Motivation (IM) criteria in active selection of the action. In its original formulation [3], IM can adaptively partition exploration space into distinct regions and guide the agent towards the regions that are neither too complicated nor too simple. The medium difficulty is formalized by guiding the agent to regions that provide maximal learning progress. In a similar manner, our system actively selects the actions with maximal learning progress. Learning progress (LP) of an action is defined based on the actual increase in the mean prediction accuracy of the predictor of the corresponding action($Pred_{a_i}$):

$$LP_{a_i}(t+1) = \overline{\gamma}_{a_i}(t+1) - \overline{\gamma}_{a_i}(t+1-\tau)$$

where $\overline{\gamma}_{a_i}(t+1)$ and $\overline{\gamma}_{a_i}(t+1-\tau)$ are defined as the current and previous mean prediction accuracies of the effect predictor, and τ is a time window, set to 2.

Here we define mean prediction accuracy by empirically setting a smoothing parameter θ to 5:

$$\overline{\gamma}_{a_i}(t+1) = \frac{\sum_{j=0}^{\theta} \gamma_{a_i}(t+1-j)}{\theta+1}$$

where predictor accuracy of the action ($\gamma_{a_i}(t)$) is equal to the ratio of the correct predictions on objects explored by the action at step t. This is only a local measure that approximates the real accuracy. This local value fluctuates substantially especially in the beginning of the learning as the immature predictors display highly variable performance for small number of samples. Therefore, smaller θ values would lead to highly fluctuating learning progress, whereas larger θ values would over-average accuracies losing the resolution to compute the increase. We used this local accuracy measure in our online incremental learning setup as the robot cannot access to ground truth, i.e. it cannot know the effect categories of the objects without actually executing its actions on all of them in a real setting.

Finally, the next action to be explored is selected based on the above learning progress criteria using ϵ-greedy strategy [36].

$$a_{\text{sel}}^t = \begin{cases} a_r & \text{if } t < t_{\text{init}} \\ a_r & \text{if } \zeta < \epsilon \\ \arg\max_{a_i} LP_{a_i}(t) & \text{otherwise} \end{cases}$$

where a_{sel}^t denotes the selected action at learning step t, $0 \geq \zeta \geq 1$ is a uniform random number, and ϵ is set to 0.10. This IM based action selection strategy requires an initial random exploration phase where the learning progress (LP) of each predictor is initialized. t_{init} is empirically set to 48 interactions.

D. Active selection of objects

After selecting the action, the system needs to decide which objects to interact with this action. For this, we propose a heuristic that maximizes the diversity of the objects used in training of the corresponding predictor. The system chooses an object from the available set of objects with maximal distance to the set of already interacted objects (O_{used}). The distance between pairs of objects can be computed by calculating the distance between feature vectors that represent the corresponding objects. However, each object feature channel is represented in a different metric, therefore the distance would depend on the relative weighting of the features in different channels. For example, while feature channel corresponding to object shape is encoded by histograms, dimension feature channel is encoded in meters. As it is not straightforward to combine these values directly, we propose a heuristic that computes the Euclidean distance in a feature channel space that is randomly selected in each time it is called:

$$o_{\text{sel}} = \underset{o_1 \in O \setminus O_{\text{used}}}{\arg\max} \sum_{o_2 \in O_{\text{used}}} dist_{\mathbf{c}}(o_1, o_2)$$

where O, O_{used}, and $O \setminus O_{\text{used}}$ correspond to the set of all objects, the set of objects used for training the corresponding predictor, and the set of the objects not used yet. $\mathbf{c}$ corresponds to the feature channel where distance is calculated, and is uniformly sampled from the set of low-level features {size, shape, distance}. If the predictor of the selected action, $Pred_{a_{\text{sel}}}$, takes input from other effect predictions, a random effect prediction from the corresponding set ($\{\varepsilon_{a_i} | a_i \neq a_{\text{sel}}\}$) is used instead:

$$o_{\text{sel}} = \underset{o_1 \in O \setminus O_{\text{used}}}{\arg\max} \sum_{o_2 \in O_{\text{used}}} \| \varepsilon_{a_i}^{o_1} - \varepsilon_{a_i}^{o_2} \|$$

In each episode, we chose four objects if a_{sel} is a single-object action, and four pairs of objects if a_{sel} is a paired-object action. Empirically, we observed that this active object selection strategy is effective only if it is applied in the initial training steps for each predictor. If it is applied for complete training, the choice of objects degenerates probably because the whole range of diversity cannot be represented by the available features, thus the overall performance of the predictors decrease. Therefore, we applied this active object selection strategy in the first 12 steps, which is a value empirically found. After 12 steps, o_{sel} is always selected randomly from $O \backslash O_{\text{used}}$.

E. Active selection of input connections

The input connections of the $Pred_{a_{sel}}$ (therefore input/output connections of the whole system) are updated in this step. For this, the minimal set of input connections that can achieve maximum available prediction accuracy for $Pred_{a_{sel}}$ is selected. One of the feature selection methods, which generates near-optimal feature sets [37], namely Sequentialfs, is used for this purpose. Each input connection in our system correspond to either a low-level feature channel or prediction of another action. The algorithm starts with an empty input set. At each iteration, a new input is selected and added to the input set of previous iteration. In order to select this new input, all candidate inputs are separately added to the previous input set and separate candidate input sets are formed. Then, the candidate input sets are evaluated through 10-fold cross-validation on trained predictors. The best performing candidate set is then transferred to the next iteration. In the experiments, we eliminated the ones that have no effect in accuracy increase, finalizing the most distinctive inputs for each trained predictor $Pred$. Therefore, we expect minimal redundancy in the input connections. Finally, in order to avoid cyclic predictions, the output of $Pred_{a_i}$ is not considered as a potential/candidate input for $Pred_{a_j}$, if the output of $Pred_{a_j}$ is previously selected as an input for $Pred_{a_i}$.

IV. Experiment Setup

1) Low-level object features: The system can visually detect the objects in the environment using the depth information of Kinect sensor, and compute a number of features from the point cloud data. $feat$ represents the continuous low-level feature vector that combines various visual properties of the object. It is composed of three different channels, which are feature vectors themselves:

$$feat^o = (dim^o, shape^o, dist^o)$$

dim represents the dimensions of the object in three different axes. $shape$ corresponds to the distribution of normal vectors obtained from the local surfaces on the point cloud. Normal vectors are obtained using Point Cloud Library [38], normal estimation package. This distribution is encoded by histograms of normal vectors, where each component in different axis is regarded separately. 8 bins are used to discretize the distribution in each axis, making the total size of $shape$ feature vector is $3 \times 8 = 24$. Finally, in order to capture the local discontinuities and distribution of distances in the neighborhood of each voxel, we use $dist$ feature channel. For each voxel, the neighboring voxels are identified in the 2-d depth image, and distances to the neighbors are computed along each four direction. For each direction, we created a histogram of 20 bins with bin size of $0.5cm$, obtaining a $4 \times 20 = 80$ sized vector for the $dist$.

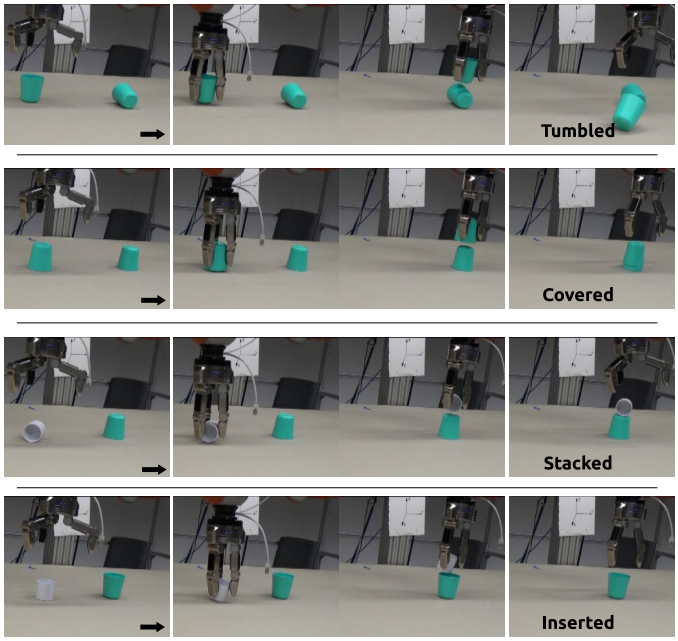


Fig. 6. Sample interactions observed during *stack* action execution on different object pairs. See http://www.cmpe.boun.edu.tr/~emre/material/TCDS2016/ for the robot videos and the source code used for learning.

2) Actions: The robot [1] is equipped with a number of manually coded actions that enable single and multi object manipulation. The robot can poke a single object from different sides using *front-poke*, *side-poke*, and *top-poke* actions. It can also stack one object on the other using *stack* action, where it grasps the first object, moves it on top of the other one and releases it.

3) Interaction Dataset: An interaction dataset, which is composed of (object, action, effect) tuples, is generated to run and analyze our method. 83 objects with different sizes, shapes, and affordances were used for this purpose (Fig. 5). The low-level visual features of these objects were computed by placing them on the table and by extracting the point cloud using the Kinect sensor. Note that object detection procedure and computed features were verified in real world robot execution experience in [17]. A number of sample robot interactions that involve some of these objects are given in Fig. 6 and Fig. 4 where different effects can be observed. With three single-object actions, and one paired-object action,

[1]The robot system is composed of a 7 DOF Kuka Light Weight Robot (LWR) arm placed on a vertical bar similar to human arm, a A 7 DOF 3 fingered Schunk gripper mounted to the robot arm, and a Kinect sensor placed over the 'torso' with a view of the table in front of the robot. Because the main focus of this paper is not on physical execution of the robot, we will omit the details concerning the robot setup.

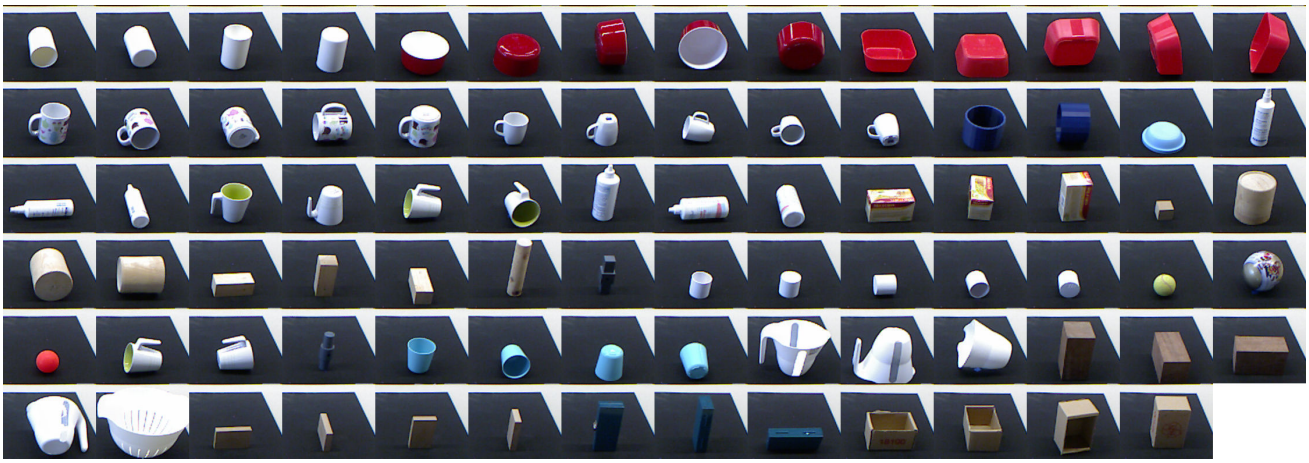


Fig. 5. Objects used in the experiments.

the total number of possible interactions with these objects is $3 \times 83 + 1 \times (83 \times 83) = 7138$. As we aim for a systematic analysis of the system behavior with large number of learning runs, we would like to collect and store the dataset in the beginning, and use the interaction samples whenever necessary. On the other hand, making over 7000 interactions is not feasible in the real world. It is also not critical to validate our method as our method is designed as a general approach for inter-dependent affordance learning rather than solving specific affordance problems in real-world settings. Therefore, we used a human expert, who observed the actions of the robot on different objects, and generalized his observations to other objects in order to manually fill-up the effect fields of the interactions [2].

4) Action effects: The effect of an action on a single object depends on various properties of the object being interacted. The same poke action generates different effects on different objects or even on the same objects in different orientations. For example, when poked from side, a lying cylinder will roll away, an upright thick cylinder will be pushed, an upright thin cylinder will topple down, and a lying hollow cylinder will stay still if the robot finger would go through the hole. When paired-object actions are considered, the effect depends on the properties of both objects and the relations between these properties. For example, while an 'inserted-in' effect is generated when a small cylinder is stacked on an upright bigger hollow cylinder, a 'tumbled' effect is observed if the big cylinder is not upright. Based on our previous work where effect categories were self-discovered in single-object actions [13] and in paired-object actions [17], and based on our observations obtained from the sample real-robot executions mentioned in the previous paragraph, the following effect categories are defined:

- Poke-effects: {pushed, rolled, toppled, resisted, nothing}
- Stack-effects: {stacked, inserted, covered, tumbled}

[2]Predicting the effects of actions on different objects without any reference to the real world performance of the robot has the risk of creating a human-biased interaction dataset. In order to reduce this risk, the human physically 'simulated' the actions on objects when it was difficult to assess the effect. For example, it is difficult to predict whether containers of similar sizes would be inserted, stacked or tumbled, when one is released on top of another one. To disambiguate these situations, the human expert dropped the object physically from some height with a small offset, and stored the generated effects. While this approach should be enough to collect the interaction dataset for our purpose, if one aims to verify a real-robot solution, the uncertainty in action level should be better addressed by the actual execution of the actions.

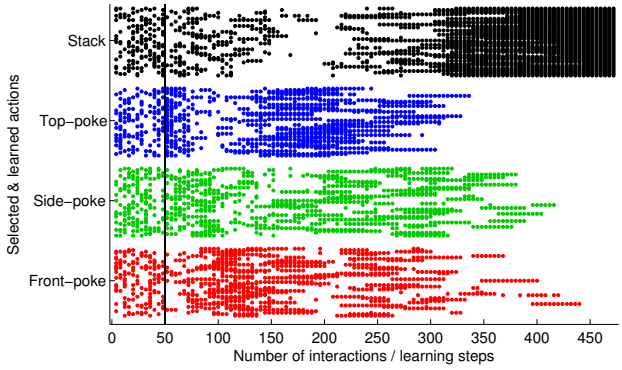


Fig. 7. The action selected for exploration and learning in each iteration of online learning of affordances. If an action is selected for learning in learning step t of a run r, the r^{th} row of the corresponding action is marked with a dot (•) at learning step t. As shown, more complex actions, e.g. *stack*, are learned later based on Intrinsic Motivation selection mechanism. This might be due to the fact that learning *stack* affordance is more difficult compared to others, and *stack* prediction of the effect of stack action requires learning of high-level attributes, i.e. single-object affordances, of both objects. Note that before the time-point shown with vertical line, the system is in its random exploration phase, i.e. randomly selects the actions to learn.

V. Experiment Results

Using the database of 83 objects, 4 actions, and their corresponding effects, we applied active learning of affordances method to discover the learning order (Section V-A) and prediction structure (Section V-B) of the affordance learning system. Furthermore, we verified that with the structure emerged, the learning of complex affordances significantly speeds up especially in the beginning of the learning trials (Section V-C).

In order to statistically analyze the system, we generated 50 separate learning runs that start with random seeds. In other words, our analysis provides the results from 50 different

affordance prediction structures that are trained with the same method but with different random seeds. These seeds affect action and object selection in different ways. First of all, in each run, the objects and actions are shuffled to avoid the effect of position of the elements in the lists. Next, ϵ-greedy action selection strategy, random channel selection in object distance calculation, and finally random object selection all use random selection from uniform distributions.

A. Discovered exploration and learning order

This section provides the obtained learning order of the affordance predictors. The learning order of predictors can be analyzed by examining the order and frequency of the corresponding actions that are selected during each iteration of the online learning of the complete system. The selected action for exploration in each learning step is shown in Fig. 7. The vertical line in the figure corresponds to the point where the system goes from random exploration to intrinsically motivated exploration phase. As the effect of paired-object actions depend on the relations between the properties of two objects, stack is a more complicated action, difficult to learn. As shown, the less complex poke actions are learned first, and more complex stack action is learned later. Prediction of the stack action can also benefit from simple-affordances (as we will show in the next section). Thus, stack action is explored and learned automatically after all other simpler actions are explored. In the figure, the stack action is observed to be explored also in the beginning of the learning in a number of steps either because of momentarily increases in local accuracy or due to the ϵ-greedy strategy. Fig. 8 provides a more detailed statistical analysis of the action selection strategy of the system, where the number of times each action is selected in each learning step window is provided in mean and standard deviation. In detail, each point on a line at learning step t in Fig. 8(a) corresponds to the number of times the corresponding action is selected within a window of ± 40 steps, averaged along the 50 runs. The bars in Fig. 8(b) corresponds to the number of action selections averaged over 50 runs and 80 learning steps. Calculating mean and variance from samples obtained from different runs enables us to statistically show the selection preference that is qualitatively visible in Fig. 7. During the initial exploration phase, which is marked by the vertical line in Fig. 8(a), the mean action selection frequencies are similar and the variances are high as the actions are selected completely randomly. After this phase, as Intrinsic Motivation selection mechanism steps in, the system starts selecting actions more systematically. First, it quickly loses its focus on stack action. There are two visible peaks in exploration: first, front-poke action and then top-poke action are explored and learned with visible peaks in frequency. Side-poke is also explored more compared to stack action in the beginning. Only towards end (275 interactions) exploration of the stack action takes lead while others are not learned anymore. This figure clearly shows that the system shifts its focus to stack action after learning all other simpler actions, however there was no such significant difference in learning order in between side-poke and stack actions. This requires further analysis of the behavior of the system.

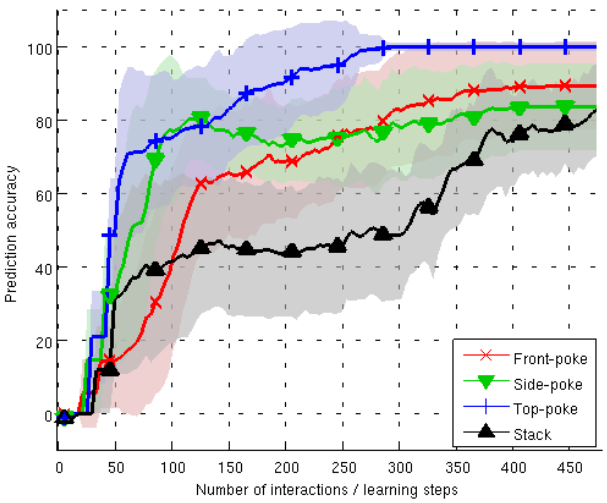


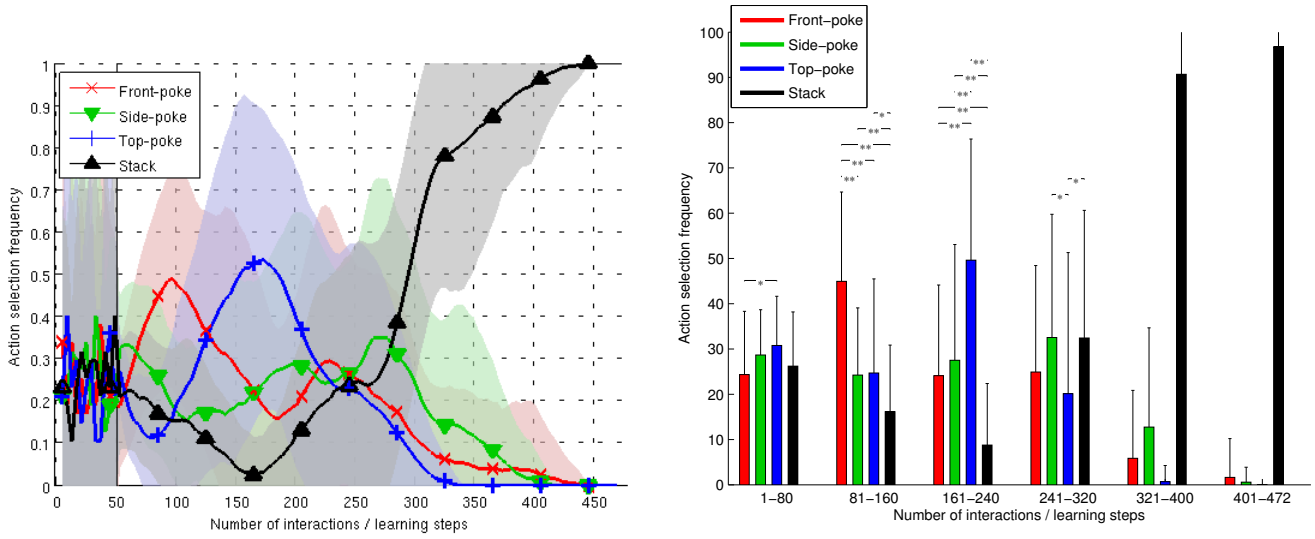
Fig. 9. The evolution of prediction accuracy of each predictor during online learning. The lines and shades correspond to mean and standard deviation of 50 different runs, respectively.

We plotted the local prediction accuracy γ evolution of each action in Fig. 9. As shown, the learning progresses of poke actions are high initially. After around 100 interactions, the learning progress of side-poke and front-poke slows down, however the system continues learning from top-poke action. The learning progress for side-poke and front-poke starts increasing again after 200 interactions, while the learning progress of stack action is lower in general. After all actions are learned sufficiently and there is no further progress, the learning progress of stack action increases.

B. Discovered affordance prediction structure

This section gives the results of the structure evolution of the affordance prediction system. Recall that the prediction structure is defined over the most distinctive inputs that are discovered to be most effective in predicting affordances. The ratio of the types of distinctive features used in prediction in different phases of the active learning are shown in Fig.10. Each plot in this figure corresponds to the evolution of the used features and affordances for a different action. One can notice the following emergent structuring properties and tendencies:

- Shape and dim low-level features are used by all the predictors in order to predict action effects. Shape features, i.e. distribution of the normal vectors, are probably important in distinguishing rollable objects from pushable objects. Dimension related properties are also important in order to differentiate objects that are toppled by front- or side-poke actions. It is peculiar that dim feature is also found to be important and used for prediction of top-poke action effect because size of the objects should have no influence on how they are affected by top-poke action. With a closer inspection, we found out that walls of some hollow objects were not detected by the perception system, and these hollow objects were perceived as very thin objects.
- Dist low-level feature was originally designed to capture the characteristics related to hollowness of the objects.



(a) (b)

Fig. 8. The number of times each action is selected in each learning step. (a) The lines and shades correspond to mean and standard deviation of 50 runs, respectively. (b) The learning episodes are divided into 6 portions, and statistical analysis on action selection is performed across 80 samples $\times$ 50 runs. Means and standard deviations are provided by the bars and lines over bars. $*$ and $**$ correspond to significance levels of $p < 0.05$ and $p < 0.01$, respectively.

However, probably shape features capture convexity and concavity good enough, therefore poke action predictors did not heavily use dist features.

- Most of the front-poke, side-poke and stack action predictors used predictions of top-poke action predictor. Top-poke predictor would predict if the object is hollow or not, therefore we were expecting that it would be used for predicting stack effects. Top-poke predictor also encodes rollability to different directions, probably that was why it was used by other poke action predictions.
- Around 25% of the front-poke predictors used side-poke and stack predictions, and around 25% of the side-poke predictors used front-poke and stack predictions. This result shows that no strict hierarchy appeared between front-poke and side-poke effect predictors. On the other hand, around 50% of stack action predictors used predictions from front-poke and side-poke predictors. We can suggest that differences in learning order of different predictors caused differences in different structuring strategies.

For illustrative purposes, we weighted directed links in between predictors based on the frequencies provided in Fig. 10 at iteration number 400, and obtained the connectivity graph provided in Fig. 11(a). As shown in the plots, each low-level feature affects affordance predictions in different levels, and shape features are observed to be the first discovered distinctive features, i.e. selected input connections, especially in the initial phases of development for all actions. However, more important in the context of this paper, affordances are observed not to be used in the initial phases, and all of them are only found to be used in predicting effect of stack action, i.e. predicting stackability affordances. Note that stack predictor starts using single-object affordances after around 30 samples, probably because the single-object affordance prediction was not good enough before that time-point.

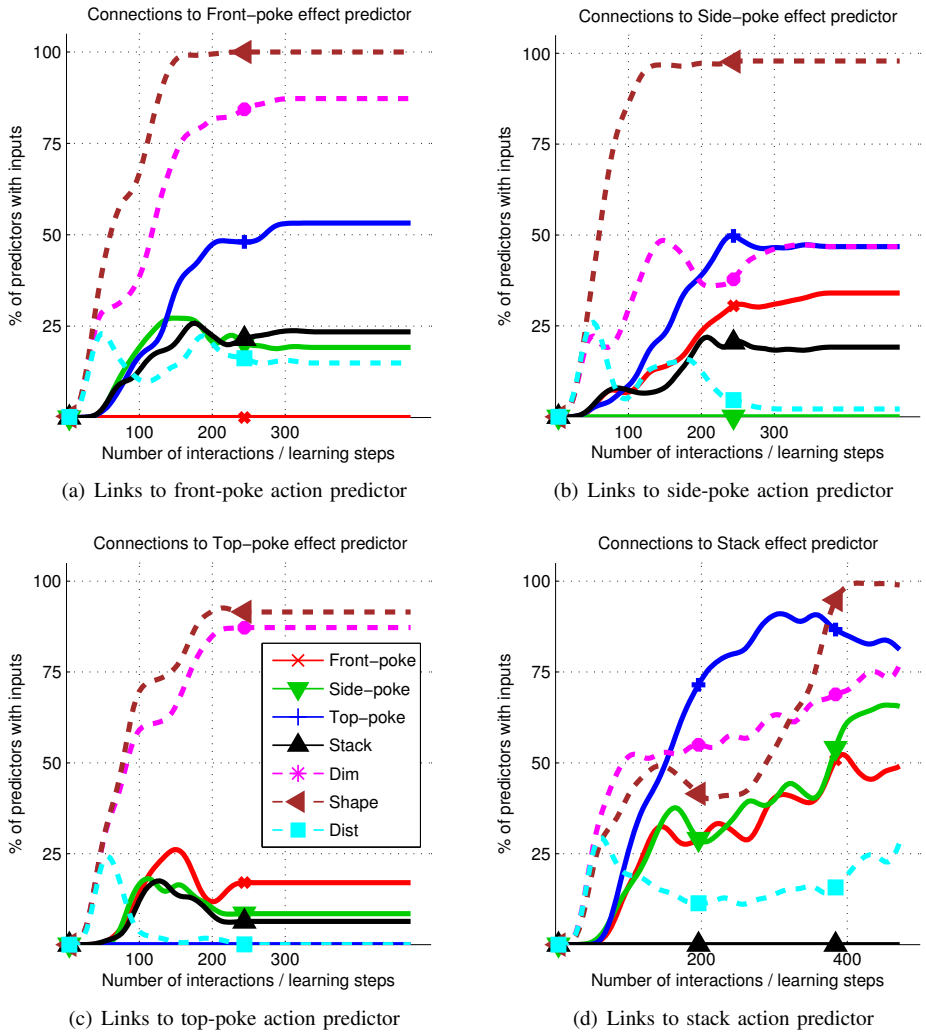
For illustration purpose, we removed the links that have weights below 25%. In other words, we removed the links if those links exist in less than 25% of the learning runs. When the graph is re-arranged, a loose but clear hierarchical prediction structure emerges as shown in Fig. 11(b).

C. Bootstrapping effect in stack learning

In this section, we analyzed the bootstrapping effect that is achieved by discovering use of single-object affordances for predicting paired-object affordances. In order to analyze this effect, we compared the learning speed with and without use of inputs from other effect predictions in predicting stack effects. The system, where effect predictions of other actions along with low-level features are used, is called 'with-affordance-input'. The system where only low-level features are used in learning and predicting effects of stack actions will be called 'without-affordance-input'.

When the learning speed of stack affordances is compared with- and without-affordance-inputs in an active learning setting where all affordances are learned as above, we noticed no significant difference between learning speeds. We argue that the bootstrapping is generally achieved in the beginning of learning the bootstrapped systems are shown to make better predictions with even small number of learned interactions. As soon as the number of samples get higher, the bootstrapping effect is reduced. In our system, stack predictor is trained along with other predictors, and in the initial phases of its training, it does not receive correct inputs from other predictors; therefore we could not achieve such bootstrapping effect.

Next we analyzed the case, where stack predictor learning starts after learning all poke predictors. Here, we assume that the system learned the learning order and structure of affordance predictors, and transfer the learned knowledge for new objects. Therefore, in this setting, the predicting poke



(a) Links to front-poke action predictor

(b) Links to side-poke action predictor

(c) Links to top-poke action predictor

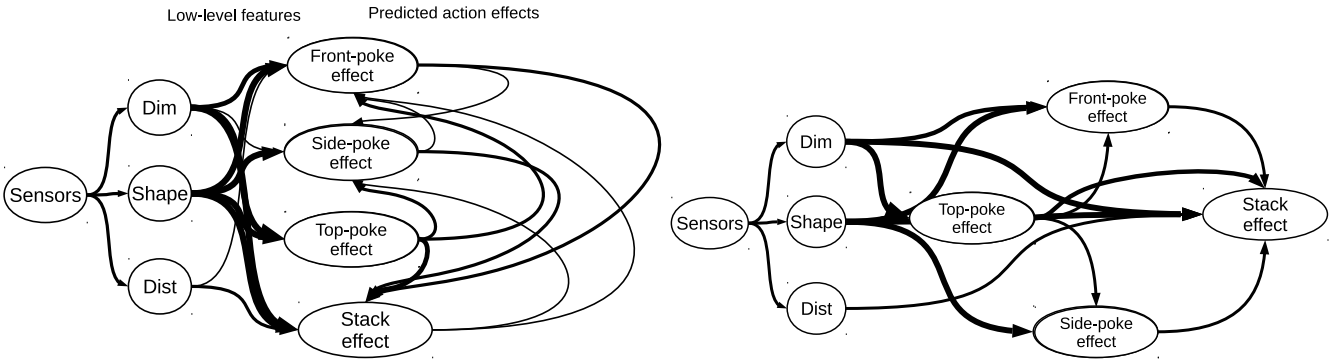
(d) Links to stack action predictor

Fig. 10. Evolution of the distinctive features for prediction, where dashed lines correspond distance, shape and dimension features, and solid lines correspond to the predicted affordances. The percentages are computed across 50 runs.

actions is learned first, and the outputs of the learned predictors are used as potential inputs. The predictions from poke actions are also used in active object selection in order to compute the distance between objects and to maximize the diversity in training set. We again performed 50 learning runs both with- and without-affordance-inputs in learning stack effect predictions. In this setting, we can see a significant bootstrapping effect in learning as shown in Figs. 12(a) and 12(b). Mean and standard deviation of the group of trained predictors with- or without-affordance-inputs are shown with the bold line and the filled areas. As shown in the figures, initial and final mean accuracies and variances of two different cases are same. Still, the expected bootstrapping effect in the beginning of the learning steps are clearly visible in both figures. At the bottom of each figure, the difference in accuracies between with- and without-affordance-inputs is given, which is confirmed with double-side t-test with $p < 0.01$. The boxes at the bottom in Fig. 12(a) show that the accuracy of with-affordance-inputs case becomes 10% higher with 40 training samples, compared to without-affordance-inputs case with significance level of $p < 0.01$. As shown, the bootstrapping effect quickly increases in the beginning and remains same for some time. The bootstrapping effect is more significant when the accuracy is computed with novel pairs. Novel pairs of objects $\{(o_1, o_2)\}$ for stack action corresponds to objects, where $\{o_1\}$ and $\{o_2\}$ have never been experienced in bottom and top roles during stacking, respectively. The effect becomes visible after 18 pairs instead of 22 pairs; and increases to a maximum value of 13% compared to 10%. This also shows that use of affordances as inputs in learning and predicting other affordances provides significant generalization capabilities.

VI. Discussion

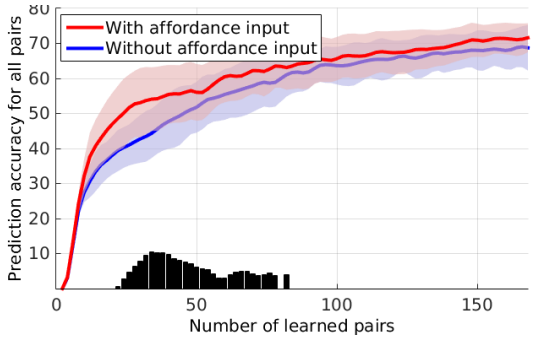
In this paper, the action and effect categories are known by the system. From a developmental point of view, these categories should be learned by the robot as it is not possible to predict and design the categorization that is suitable for the embodiment of the robot especially for unbounded set of complex actions. Therefore, we believe that it is necessary to discuss how such categorization can be achieved by interacting systems. In this section, we will summarize our previous



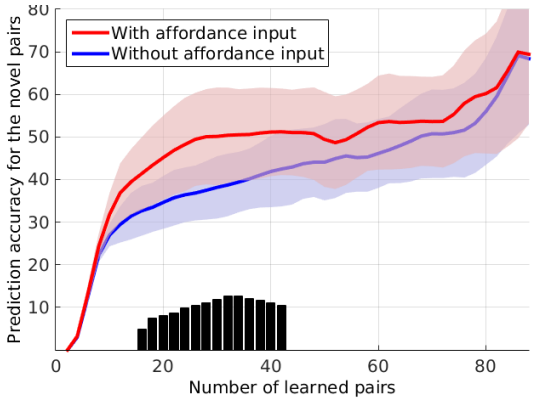
(a) Connectivity graph with weighted links

(b) Connectivity graph when loose links are removed

Fig. 11. The structure, i.e. input and output connections, obtained at the end of learning.



(a) Test with all pairs.



(b) Test with novel pairs.

Fig. 12. The evolution in prediction accuracy of stack effect predictor. The accuracy is computed by using all object pairs in (a) and by using only the novel object pairs in (b). The black bars show the amount of difference in prediction accuracies between learning systems with and without affordance inputs, in significance level of $p < 0.01$.

studies where categorization in action and effect spaces were self-discovered by a manipulator robot.

a) Learning action categories: We previously showed that a robot that was initialized with a basic reach- and enclose-on-contact movement capability, can discover a set of action primitives by exploring its movement parameter space [39, 12]. In that work, the robot was assumed to have only two basic movement mechanisms: a basic finger enclose behavior akin to infants palmar grasp reflex, and one basic arm action, that generates simple arm movements to transport the hand to the vicinity of an object. The contact information, sensed during the approach and possibly after the finger enclosure, is used to cluster the executed movements, yielding a set of action primitives such as push, 'release', and grasp. In order to learn more complex actions, such as 'move-object-over-another', we realized an imitation learning strategy where the robot observed and encoded complex demonstrations into a series of previously learned primitives with the help of a cooperative tutor [12]. In that work, inspired from infant development and motionese framework [40], we equipped the robot with mechanisms that can detect motion related cues such as pauses in order to more easily segment the demonstrated behavior into chunks that can be replicated by previously discovered action primitives. We argue that simple action categories can be discovered through exploration, and more complex actions that are composed of simple categories can be learned through imitation learning.

b) Learning effect categories: Effect categories are generally obtained in an unsupervised way by applying clustering methods to continuous effect space. We previously discussed that such clustering is sensitive to relative weighting of the effect features that are encoded in different units and channels. Furthermore, discovered effect categories should be instrumental for the latter predictions and reasoning; and completely unsupervised clustering does not provide any guarantee for this. Therefore, we proposed a 2-level clustering algorithm that takes into account the representational differences between different perceptual channels and utilizes a verification step that makes sure that the discovered effect categories can be predicted by the robot [39, 13]. In detail, in the lower level, channel-specific effect categories are found by clustering in the space of each channel, discovering separate categories for channels such as touch, position and shape. In order to ensure the predictability of the channel-specific effect categories, classifiers are trained. If a channel-specific effect category is found not to be predictable, the clustering is re-done with less number of desired clusters. After finding the final channel-specific effect categories, in the upper level these categories are combined to obtain all-channel effect categories using the Cartesian product operation. The final categories are only accepted if they are predictable. The discovered effect categories for push action were 'disappeared=rolled',

'grasped&disappeared', 'grasped', and 'pushed'. Similar ideas were also used by others, where categorization was also based on the ability to predict the outcome of action execution in Mugan and Kuipers [41], and categories were utilized if they appear in the learned rules in Pasula et al. [42]

With increasing complexity of actions, finding effect categories becomes more difficult. In finding effect categories of stack action for example, we observed that a naive clustering in continuous effect space was not effective as the interacting objects can generate various effects difficult to separate due to the complex interactions between them [17]. In that case, applying further exploratory actions on the objects after stacking helped the system to handle this complexity. For example, after a stack action, if the robot poked both of the objects one by one, effect categories were found to be distinguishable when observations obtained from the following poke actions were also taken into account. Some discovered effect categories for stack action were 'stacked', 'inserted-in', and 'tumbled' at the end.

VII. Conclusion

In this paper, we studied how interdependent affordance learning tasks can be autonomously structured along with the learning order of its components. In an online learning framework, we showed that intrinsic motivation mechanism, which select the next action to explore based on learning progress of the model of that action, can discover such a learning order where paired-object affordance learning follows single-object affordances learning. Next, we showed that by using the most discriminative features for affordance prediction, the expected hierarchical structure emerged autonomously where the learning system discovered that predictions of the single-object affordances are connected to the paired-object affordances. We validated our approach in a real dataset composed of 83 objects and pairs of these objects along with the effects of three poke actions and one stack action. The results show that hierarchical structure and learning order emerged from the learning dynamics that is guided by Intrinsic Motivation mechanisms and feature selection approach.

The results further show that a bootstrapping effect in learning speed of affordances could be achieved with such a hierarchical structure. This was probably achieved as simple affordances, which were used as inputs to complex affordances, provide more abstract and generalizable knowledge compared to low-level visual features. Bootstrapping effect was visible when generalization was required from learning of small number of samples, and disappeared in the latter steps as the system could learn equally well from low-level features with large number of training samples. Bootstrapping effect was also more significant in more difficult tasks such as when the predictors were tested in novel situations. In our system, the learning order of affordances was not strict, and determined based on the IM criteria with an approximate measure of future learning progress. Therefore, the hierarchical structure only appeared after initialization of learning progress through some exploration of all actions, and the initial bootstrapping effect was not observed in this emergent structuring setting. Other factors such as motor complexity of actions are also important in deciding which action to explore first, and can be used to achieve a more strict learning order, and therefore more significant bootstrapping effect.

In our framework, the system forms forward models [43] that enable it to predict the changes in the environment in terms of discrete effect categories. Generative models have been shown to be effective in learning (object, action, effect) relations and in making inferences on any element of the these relations. For example, they can infer the required action given the desired (object, effect) pair or they can predict the effect given (action, objects) pair [14]. 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 [44] possibly using complex non-linear functions without any initial categorization of object properties as in [14, 18]. 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 [13].

Our proposed framework, which exploits the idea of active selection of actions, objects, and connections, is very general and can be directly applied for a wide-variety of robot learning tasks and to different robotic platforms. It can also be extended to continuous actions where action parameters such as poke direction are used as inputs of the predictors, and to continuous action effects where powerful regressors are used for prediction.

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 agreements no. 270273, Xperience, and no. 610532, Squirrel.

References

[1] M. Lopes, P.-Y. Oudeyer *et al.*, "Guest editorial active learning and intrinsically motivated exploration in robots: Advances and challenges," *IEEE Transactions on Autonomous Mental Development*, vol. 2, no. 2, pp. 65–69, 2010.

[2] P.-Y. Oudeyer and F. Kaplan, "What is intrinsic motivation? a typology of computational approaches," *Frontiers in neurorobotics*, vol. 1, pp. 1–14, 2007.

[3] P.-Y. Oudeyer, F. Kaplan, and V. V. Hafner, "Intrinsic motivation systems for autonomous mental development," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 2, pp. 265–286, 2007.

[4] A. Baranes and P.-Y. Oudeyer, "Robust intrinsically motivated exploration and active learning," in *Development and Learning, 2009. ICDL 2009. IEEE 8th International Conference on*. IEEE, 2009, pp. 1–6.

[5] S. Ivaldi, S. Nguyen, N. Lyubova, A. Droniou, V. Padois, D. Filliat, P.-Y. Oudeyer, and O. Sigaud, "Object learning through active exploration," *IEEE Transactions on Autonomous Mental Development*, vol. 6, no. 1, pp. 56–72, 2013.

[6] E. J. Gibson, "Perceptual learning in development: Some basic concepts," *Ecological Psychology*, vol. 12, no. 4, pp. 295–302, 2000.

[7] J. J. Gibson, *The Ecological Approach to Visual Perception*. Lawrence Erlbaum Associates, 1986.

[8] H. S. Koppula, R. Gupta, and A. Saxena, "Learning human activities and object affordances from RGB-D videos," *International Journal of Robotics Research (IJRR)*, vol. 32, no. 8, pp. 951–970, 2013.

[9] Y. Sun, S. Ren, and Y. Lin, "Object–object interaction affordance learning," *Robotics and Autonomous Systems*, vol. 62, no. 4, pp. 487–496, 2014.
[10] A. Myers, C. L. Teo, C. Fermuller, and Y. Aloimonos, "Affordance detection of tool parts from geometric features," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 1374–1381.
[11] M. Schoeler and F. Worgotter, "Bootstrapping the semantics of tools: Affordance analysis of real world objects on a per-part basis," *IEEE Transactions on Autonomous Mental Development*, 2015, accepted, 10.1109/TAMD.2015.2488284.
[12] E. Ugur, Y. Nagai, E. Sahin, and E. Oztop, "Staged development of robot skills: Behavior formation, affordance learning and imitation with motionese," *IEEE Transactions on Autonomous Mental Development (TAMD)*, vol. 7, no. 2, pp. 119–139, 2015.
[13] 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.
[14] 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.
[15] J. Sinapov, C. Schenck, K. Staley, and A. Stoytchev, "Grounding semantic categories in behavioral interactions: Experiments with 100 objects," *Robotics and Autonomous Systems*, vol. 62, no. 5, pp. 632–645, 2014.
[16] S. Griffith, J. Sinapov, V. Sukhoy, and A. Stoytchev, "A behavior-grounded approach to forming object categories: Separating containers from noncontainers," *IEEE Transactions on Autonomous Mental Development*, vol. 4, pp. 54–69, 2012.
[17] E. Ugur and J. Piater, "Bottom-up learning of object categories, action effects and logical rules: From continuous manipulative exploration to symbolic planning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 2627–2633.
[18] 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 *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 4373–4378.
[19] B. Moldovan and L. De Raedt, "Learning relational affordance models for two-arm robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Chicago, IL, USA: IEEE, 2014, pp. 2916–2922.
[20] S. Fichtl, D. Kraft, N. Kruger, and F. Guerin, "Bootstrapping the learning of means-end action preconditions," *IEEE Transactions on Autonomous Mental Development (TAMD)*, 2015, submitted.
[21] H. F. Harlow, M. K. Harlow, and D. R. Meyer, "Learning motivated by a manipulation drive." *Journal of Experimental Psychology*, vol. 40, no. 2, pp. 228–234, 1950.
[22] G. Baldassarre, T. Stafford, M. Mirolli, P. Redgrave, R. M. Ryan, and A. Barto, "Intrinsic motivations and open-ended development in animals, humans, and robots: an overview," *Frontiers in psychology*, vol. 5, pp. 1–5, 2014.
[23] G. Baldassarre and M. Mirolli, *Intrinsically motivated learning systems: an overview*. Springer, 2013.
[24] J. Law, P. Shaw, K. Earland, M. Sheldon, and M. Lee, "A psychology based approach for longitudinal development in cognitive robotics," *Frontiers in neurorobotics*, vol. 8, pp. 1–23, 2014.
[25] S. M. Nguyen and P.-Y. Oudeyer, "Active choice of teachers, learning strategies and goals for a socially guided intrinsic motivation," *Paladyn Journal of Behavioural Robotics*, vol. 3, no. 3, pp. 136–146, 2012.
[26] S. Hart and R. Grupen, "Intrinsically motivated affordance learning," in *Workshop on Approaches to Sensorimotor Learning on Humanoids, IEEE Conference on Robotics and Automation (ICRA)*, 2009.
[27] ——, "Learning generalizable control programs," *IEEE Transactions on Autonomous Mental Development (TAMD)*, vol. 3, no. 3, pp. 216–231, 2011.
[28] J. Piaget, *The Origins of Intelligence in Children*. New York, USA: International University Press, 1952.
[29] A. G. Barto, S. Singh, and N. Chentanez, "Intrinsically motivated learning of hierarchical collections of skills," in *Proceedings of the 3rd International Conference on Development and Learning (ICDL 2004)*, 2004, pp. 112–119.
[30] A. Baranes and P.-Y. Oudeyer, "Active learning of inverse models with intrinsically motivated goal exploration in robots," *Robotics and Autonomous Systems*, vol. 61, no. 1, pp. 49–73, 2013.
[31] E. Ugur and J. Piater, "Emergent structuring of interdependent affordance learning tasks," in *IEEE Intl. Conf. on Development and Learning and on Epigenetic Robotics (ICDL-Epirob)*, 2014, pp. 489–494.
[32] E. Ugur, S. Szedmak, and J. Piater, "Bootstrapping paired-object affordance learning with learned single-affordance features," in *IEEE Intl. Conf. on Development and Learning and on Epigenetic Robotics (ICDL-Epirob)*, 2014, pp. 476–481.
[33] V. N. Vapnik, *Statistical Learning Theory*. Wiley-Interscience, September 1998.
[34] C.-C. Chang and C.-J. Lin, "Libsvm : a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 27, pp. 1–27, 2011.
[35] A. Bordes, S. Ertekin, J. Weston, and L. Bottou, "Fast kernel classifiers with online and active learning," *The Journal of Machine Learning Research*, vol. 6, pp. 1579–1619, 2005.
[36] R. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 1998.
[37] A. W. Moore and M. S. Lee, "Efficient algorithms for minimizing cross validation error," in *Proceedings of the 11th International Conference on Machine Learning*, R. Greiner and D. Schuurmans, Eds. Morgan Kaufmann, 1994, pp. 190–198.
[38] A. Aldoma, Z.-C. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R. B. Rusu, S. Gedikli, and M. Vincze, "Point cloud library," *IEEE Robotics & Automation Magazine*, vol. 1070, no. 9932/12, pp. 80–91, 2012.
[39] 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.
[40] R. J. Brand, W. L. Shallcross, M. G. Sabatos, and K. P. Massie, "Fine-Grained Analysis of Motionese: Eye Gaze, Object Exchanges, and Action Units in Infant-Versus Adult-Directed Action," *Infancy*, vol. 11, no. 2, pp. 203–214, 2007.
[41] J. Mugan and B. Kuipers, "Autonomous learning of high-level states and actions in continuous environments," *Autonomous Mental Development, IEEE Transactions on*, vol. 4, no. 1, pp. 70–86, 2012.
[42] H. M. Pasula, L. S. Zettlemoyer, and L. P. Kaelbling, "Learning symbolic models of stochastic domains," *Journal of Artificial Intelligence Research*, vol. 29, pp. 309–352, 2007.
[43] M. Haruno, D. Wolpert, and M. Kawato, "Hierarchical mosaic for movement generation," in *Excepta Medica International Coungress Series*. Amsterdam, The Netherlands: Elsevier Science B.V., 2003, pp. 575–590.
[44] 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*, 2011, pp. 4768–4773.

Emre Ugur is an Assistant Professor in Bogazici University, Istanbul Turkey. He received his Ph.D degree in Computer Engineering from Middle East Technical University in 2010, worked as a research scientist in ATR, Japan from 2009 to 2013; visited Osaka University, as a specially appointed Assistant Professor in 2015-2016, and worked as a senior researcher in University of Innsbruck between 2013-2016. He is interested in developmental and cognitive robotics and brain-robot interface methods.

Justus Piater After eight years as a professor at the University of Liege, Belgium, including one year as a visiting scientist at the Max Planck Institute for Biological Cybernetics in Tubingen, Germany, Prof. Piater moved to Innsbruck in 2010. There he founded the IIS group at the Institute of Computer Science, which currently counts 6 postdoctoral researchers and 8 doctoral students. He has written about 150 publications in international, peer-reviewed journals, conferences and workshops, and has been a principal investigator in 1 EU-FP6 and 6 EU-FP7 projects, of which 4 are currently active.