Cañamero, L., Oudeyer, P.-Y., and Balkenius, C. (Eds) Proceedings of the Ninth International Conference on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems. Lund University Cognitive Studies, 146.

Affordance learning from range data for multi-step planning

Emre Ugur[1,2,3] **Erol Sahin**[3] **Erhan Oztop**[1,2]
[1] NICT, Biological ICT Group Kyoto, Japan
[2] ATR, Computational Neuroscience Labs. Kyoto, Japan
[3] METU, CENG, Kovan Lab. Ankara, Turkey

Abstract

In this paper we present the realization of the formalism we have proposed for affordance learning and its use for planning (Şahin et al., 2007) on an anthropomorphic robotic hand. In this realization, the robot interacts with the objects in its environment using the programmed push and grasp-and-lift behaviors, and records its interactions in triples that consists of the initial percept of the object, the behavior applied and the observed effect, defined as the difference between the initial and the final percept. The interaction with the environment allows the robot to learn object affordance relations to predict the change in the percept of the object when a certain behavior is applied. These relations can then be used to develop multi-step plans using forward chaining. Our experiments have shown that the robot is able to learn the physical affordances of objects from 3D range images and use them to build symbols and relations that are used for making multi-step plans to achieve a given goal.

1. Introduction

There exists a representational gap between discrete symbols used in AI planning and the continuous sensory-motor experiences of a robot and the means to bridge this gap remains a long-standing problem in autonomous robotics. Learning of the mapping between the sensory-motor readings and these symbols is one approach that is a part of the so called symbol grounding problem (Harnad, 1990) and has been studied since the days of STRIPS. The learning studies in this context typically assume that the planning symbols are pre-coded, and only the relation of continuous sensory-motor reading to these symbols are learned(Klingspor et al., 1996).

On the other hand, (Sun, 2000) argued that symbols "are not formed in isolation" and that "they are formed in relation to the experience of agents, through their perceptual/motor apparatuses, in their world and linked to their goals and actions". In fact, these types of views are becoming common place in robotics as indicated by the increasing number studies with similar views. For example, symbol formation in a robot interacting with its world was studied in (Pisokas and Nehmzow, 2002), where self-organizing maps were used to cluster low-level sensory data and to form perceptual states. The planning is performed by successively predicting the next perceptual states. As will be described later on, prediction is also central to our approach although we believe that rather than learning the state-to-state transitions, learning the "change" in the current state could be more beneficial. (Geib et al., 2006) focused on planning and grounded the pre-defined high-level domain structures in the form of preconditions and effects. In our approach, we too address planning; but in addition, importantly we require that the affordance relations be learned bottom-up through interaction with the environment. The affordance notion we adopt has been adopted by other researchers as well. For instance, (Fitzpatrick et al., 2003) implemented a system where pushability affordances and the roll directions of the objects after the application of the push were learned. (Montesano et al., 2008) proposed a general probabilistic model based on Bayesian networks to learn the relationship between actions, objects, and effects through interaction with the environment. Given objects and actions (or any pair of components), the system had the ability to predict the effect (or the third component). In (Sinapov and Stoytchev, 2008) the affordances of the tools attached to the robot arm are learned by building a hierarchical models for behaviors and their observed outcomes. In (Griffith et al., 2009), the object affordances were learned through interaction for a task that requires categorization of container and non-container objects. Although these studies focused on affordance learning and prediction mech-

anisms through interaction with the environment, the use of learned/acquired knowledge has not been demonstrated for making multi-step plans.

In this paper we attempt to fill this gap by presenting a robotic system that interacts with its environment for learning the effects of its actions and representing affordance relations. After the learning phase, we show that the robot can make non-trivial multi-step plans involving push and grasp-and-lift behaviors based on the learned affordance relations. From a developmental point of view, this learning phase can be related to development of infants between 7-11 months, who explore the environment and learn the dynamics of the objects by hitting, grasping and dropping them and observing the results of their actions (Asada et al., 2009).

1.1 Affordances and Robot Control

According to Ecological Psychologist J.J. Gibson (Gibson, 1986) the organisms do not need to recognize the action-free meanings of the objects and make complex inferences over these meanings in order to act on them. For example we do not need to identify the objects when we need to interact with them. Instead, we look for a specific combination of the object properties taken with reference to us and our actions in order to detect their affordances. This introspection is also supported by neuroscientific findings. It is known that primate brain process visual information at in least two pathways: dorsal and ventral pathways. The ventral pathway appears to be responsible for object identification; whereas the dorsal pathway is more involved in perception for action (Culham and Valyear, 2006, Goodale, 2008, Goodale and Milner, 1992, Ungerleider and Mishkin, 1982). In particular, the anterior intraparietal area (AIP) appears to be the neural basis of manipulation related affordances as it is involved in computation of object features relevant for grasping (Sakata et al., 2005, Oztop et al., 2006).

Recently, we proposed a formalism (Şahin et al., 2007) for using affordances as a framework at different levels of robot control ranging from perceptual learning to planning. The formalism defines affordances as general relations that pertain to the robot-environment interaction, and represented them as triples of (1) the initial percept of the object, (2) the behavior applied and (3) the effect produced. For instance, the lift-ability affordance is represented as a relation between the (properties of an) object, the behavioral capabilities of the robot and the effects produced by the lift behavior.

In this paper we present the realization of this formalism on an anthropomorphic robotic hand and show that the robot interacts with the objects in its environment and records its interactions as affordance relations and later use them to make multi-step plans for achieving given goals.

2. Experimental framework

An anthropomorphic robotic system, equipped with a range sensor, and its physics-based simulator is used as the experimental platform (Fig. 1). The robot platform is composed of a five fingered 16 DOF robot hand (Gifu Hand III, Dainichi Co. Ltd., Japan) and a 7 DOF robot arm (PA-10, Mitsubishi Heavy Industries). As for the range sensor, SwissRanger SR-4000 infrared range finder, with 176x144 pixel array, 0.23° angular resolution and 1 cm distance accuracy was used. The simulator on the other hand is developed using Open Dynamics Engine (ODE) and mainly utilized in training and interaction phase because it is not feasible to make large number of exploratory interactions in the real robot.

The robot is equipped with three *push* behaviors and one *lift* behavior. For all behaviors, the hand is placed to a 'reset' position out of the view of the camera before and after behavior execution except for *lift* . The object position computed from the range finder is used as parameter by the behaviors to enable the robot interact with objects placed in different positions. The hand is wide-open initially for all behaviors, is clenched into a fist during *push-forward* execution, and remains open for other *push* behaviors. *push-forward* , *push-left* , and *push-right* behaviors first place the robot hand at the back, right and left of the object, respectively. Then, the hand moves towards object center and pushes the object in the appropriate direction. In *lift* behavior, the robot hand is placed at the back-right diagonal of the object first, then moved towards the object and while this move the fingers are closed to grasp the object. Afterwards, the closed hand is lifted vertically.

The robot interacts with three types of objects: boxes, cylinders and spheres, with different size and orientations. During the execution of *push* behaviors, the robot observes different consequences of its actions. For instance, when the robot pushes a box ( ) or an upright cylinder ( ), the object is dragged during the execution of the behavior and stand still at the end of the action. However, when the robot pushes a sphere ( ), the object would roll away and fall down from the table, so at the end of the action the object disappears. The effect of a push behavior over lying cylinders ( ) on the other hand depends on the relative orientation of the cylinder and direction of the *push* . The *lift* behavior would succeed in lifting an object, if the object is within the arm length of the robot and small enough to fit into the robot hand. However the consequences of the *lift* behavior execution is not limited to lifting the objects and can be complex. For exam-

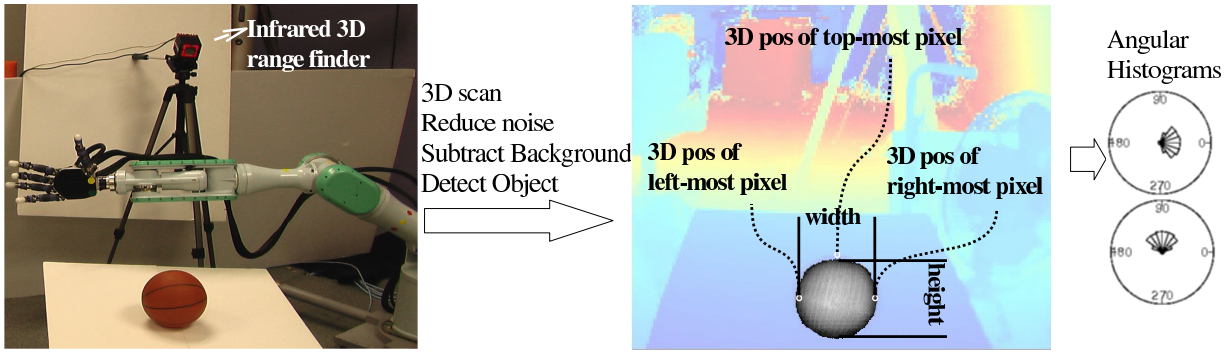


Figure 1: On the left, the 23 DOF hand-arm robotic platform, infrared range camera and a spherical object placed on the table are shown. On the right, the range image obtained from the 3D scan and a number of features computed from this image are given. Note that the subtracted background is blurred.

ple, some spheres can roll-away out of the view after an attempt to grasp and lift, and the large boxes are pushed away but remains in the view after *lift* behavior execution.

2.1 Perception

Pre-processing: The robot perceives the world through its 3D infrared range finder which provides the depth values in a range image and the 3D positions of the corresponding pixels. First, the range image is subtracted from the *background image* that was obtained from an object-free environment. The resulting image is segmented and the remained region is assumed to belong to an object. In the experiments reported in this paper only one object is presented to the robot. In order to reduce the effect of noise, the pixels at the boundary of the object are removed and then median and Gaussian filters with 5x5 window sizes are applied. Finally, the object features are computed using the depth values and 3D positions corresponding to the object pixels.

Object feature vector computation: The perception of the robot at time t is denoted as $\boldsymbol{f}_o^t$, where o is the object label and $\boldsymbol{f}$ is a feature vector of size 53. Height, width and depth are used as dimension related features of the object. Closest, furthest, left-most and right-most points of the object are extracted and their 3D positions are included into the feature vector. The average distance of the pixels are used as the distance feature of the object. As shape related features, distribution of the local surface normal vectors of the object pixels are used. Specifically frequency histograms of normal vector angles in latitude and longitude are computed and used as follows.

The normal vectors of the local surfaces for all pixels are computed using any two neighbors of the corresponding pixel:

$$\mathbf{N}_r = (\mathbf{p}_{r_1} - \mathbf{p}_r) \times (\mathbf{p}_{r_2} - \mathbf{p}_r)$$

where r, r_1 and r_2 represent indexes of the pixel, and two neighbor pixels, and $\boldsymbol{p}$ corresponds to 3D position. The direction of each normal vector is recorded in two base-dimensions, latitude and longitude. Two angular histograms are computed for each of these dimensions and the histograms are sliced into 18 intervals of 20° each. Frequency values of angular histograms obtained from normal vectors of the surface points in the region are used as 36 shape-related features. This representation encodes the distribution of the local surface normal vectors of the object.

In some situations (after execution of some behaviors) the object can move out of view. So we included a boolean *object visibility* feature in the feature vector to represent this qualitatively different situation.

Effect feature vector computation: For each object, the effect created by a behavior is computed as the difference between the final and initial features:

$$\boldsymbol{\xi}_o^{b_i} = \boldsymbol{f'}_o - \boldsymbol{f}_o$$

where $\boldsymbol{\xi}_o^{b_i}$, $\boldsymbol{f'}_o$ and $\boldsymbol{f}_o$ represents the effect, final and initial feature vectors, and b_i represents the behavior executed.

3. Learning of affordance relations

During the interaction phase, the robot interacts with the environment and in each interaction a *relation instance* of the form $(\boldsymbol{\xi}_o^{b_i}, \boldsymbol{f}_o, b_i)$ is created. After interaction phase is completed, by using effect instances $(\{\boldsymbol{\xi}_o^{b_i}\})$ that are obtained from all different objects, similar effects are grouped together to get a more general description of the effects that each behavior can create. This grouping is done using X-means clustering algorithm and for each cluster an *effect-id* is assigned. The associated *effect prototype* for the cluster is defined as the mean of the cluster, denoted as $\bar{\boldsymbol{\xi}}^{b_i}$. Hence, for a given $\boldsymbol{\xi}$, the correspond-

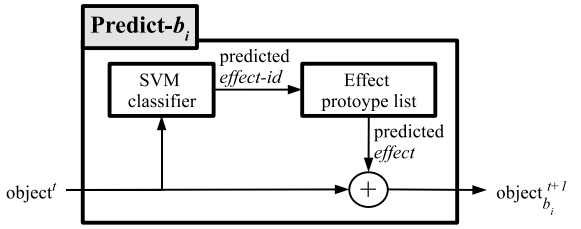


Figure 2: The `Predict-` operator that is trained to predict the next state of an object based on the predicted effect of applying behavior b_i.

ing *effect-id* (c) can be found as:

$$c_{\boldsymbol{\xi}} = \underset{1 \le i \le k}{\operatorname{argmin}} \|\boldsymbol{\xi} - \bar{\boldsymbol{\xi}}_i\| \quad (1)$$

where $1 \le i \le k$ is the cluster index.

Formally, the mapping between the object features and the effects created by a particular behavior b_i is learned by a classifier (χ^{b_i}) using the data set:

$$\mathbf{T}^{\mathbf{b_i}} = \{(\boldsymbol{f}_o, c_{\boldsymbol{\xi}_o^{b_i}})\}$$

where $\boldsymbol{f}_o$ is given as the input feature vector to the classifier χ^{b_i}, and c is the corresponding target effect category. Specifically, we used a Support Vector Machine (SVM) classifier with linear kernel to learn this mapping for each behavior b_i using this training set[1]. After training, predicted effect category can be found without applying behavior b_i to an object with perceptual features $\boldsymbol{f}_o$ by:

$$c^{\text{predicted}}(b_i, o) = \chi^{b_i}(\boldsymbol{f}_o)$$

The predicted percept of the object after the application of the behavior can then be computed as (see Fig. 2):

$$\boldsymbol{f'}_o(\{b_i\}) = \boldsymbol{f}_o + \bar{\boldsymbol{\xi}}^{b_i}_{c^{\text{predicted}}(b_i,o)}$$

4. Planning

The learned affordance relations can be used as operators for planning.

States A state is represented with the object feature vector that is perceived or expected to be perceived after execution a number of behaviors in t steps:

$$S^t_{\{{b_i}^1 \ldots {b_i}^{t-1}\}} = \boldsymbol{f}^t_{o,\{{b_i}^1 \ldots {b_i}^{t-1}\}}$$

where o corresponds to the perceived object, and $\boldsymbol{f}^t_{o,\{{b_i}^1 \ldots {b_j}^{t-1}\}}$ is the expected percept after execution of the behavior sequence $\{{b_i}^1 \ldots {b_j}^{t-1}\}$.

[1]For this study, the LibSVM software is used. In (Uğur et al., 2007) we showed that the method is not constrained to batch learning and the training can be done in an online manner. The number of samples were minimized in online version by selecting the most interesting situations for interaction instead of random exploration.

Actions The pre-coded behaviors; namely the three *push* behaviors and the *lift* behavior, constitute the actions. Different from standard techniques, the actions do not have any pre-conditions and their description does not include pre-defined state transition rules. All actions are applicable in all states, where the next state depends on the learned effect prediction operators summarized in Fig. 2.

$$S^t_{\{{b_i}^1 \ldots {b_j}^{t-1}\}} \xrightarrow{{b_k}^t} S^{t+1}_{\{{b_i}^1 \ldots {b_k}^t\}}$$

Goals A goal is specified as a partial state, in terms of values of some object features within states. The user can define a goal based on feature values of the object. For example, the state that includes an object feature vector with $d_{mean} = 0.8m$ will satisfy the goal of *move object to* $0.8m$ *distance*. As another example, the goal of *pick-up a particular object* is satisfied in a state, where the closest-point z feature value of the corresponding object is large ($z_c > 0.3m$) in the range image.

Plan generation Forward chaining is used to generate totally ordered plans starting from the initial state. This process can be viewed as the breadth-first construction of a plan tree where the branching factor is the number of behaviors. The next states are computed using the prediction operator in Fig. 2. If the state in any time step satisfies the goal, the sequence of the behaviors which lead the initial state to the goal is accepted as a potential plan.

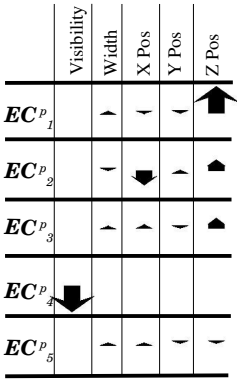
5. Experiments

The learning experiments are conducted in the physics based simulator and the results are tested in the real robot. As shown in Fig. 1, a table is placed in front of the robot both in simulation and real world. In the beginning of interactions, one random object o (among) is placed on the table, in a random orientation and size $[20cm - 40cm]$. The robot makes 3D scans before and after executing one of its behaviors (b_i) to compute the object ($\boldsymbol{f}_o$) and effect ($\boldsymbol{\xi}^{b_i}_o$) feature vectors. After the behavior is applied, if the object is still visible and if there is change in the object features, another random behavior is applied. Otherwise, the object should have been fallen down the table or it is out of reach of the arm, so the object is removed and a new random object is placed. For all behaviors, approximately 1000 interactions are simulated. The resulting set of relation instances are then used in training.

5.1 Discovered effect categories for push

After robot-environment interactions are completed and affordance relations instances are collected, X-means algorithm found 5 effect categories for each

Table 1: EC_i^p represents ith effect category of *push-forward* behavior. In (a), selected feature values of the corresponding effect prototypes are given. The magnitude of the arrow corresponds to the size of the change in feature value, whereas whether the feature is increased or decreased can be figured out by arrow's direction. In (b), in which situations such effect categories are formed is explained. The number of the object types appear during interactions are given in the first four column, the average real width and distance of the objects in those interactions are given in the last two columns.



(a) Effect features

					Width	Dist
EC_1^p	0	45	40	65	17.0	86.3
EC_2^p	0	0	0	55	23.1	90.8
EC_3^p	0	85	15	20	17.3	94.1
EC_4^p	125	10	175	15	17.5	88.8
EC_5^p	50	90	170	40	18.1	124.4

(b) Info. on objects

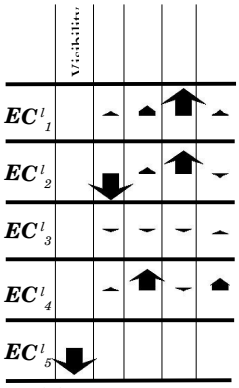
push behavior. In this section, the effect categories are interpreted by inspecting particular feature values of the corresponding effect prototypes and by identifying the situations in which these categories are generated. In all three *push* behaviors, similar categories are formed so here we present only *push-forward* behavior. Table. 1(a) gives selected feature values from effect prototypes formed for *push-forward* , $\bar{\boldsymbol{\xi}}_i^{push-forward}$ where $1 \leq i \leq 5$. In other words, the amount of change in different features is provided. Table 1(b) summarizes the situations in which effect categories are generated. The interpretation of effect categories is as follows:

- As seen in the Table. 1(a), no effect in any feature is monitored in EC_5^p . When initial average distance of the objects is examined for EC_5^p from Table 1(b), it is found to be around 124.4 cm which corresponds to out of the range points. So the robot discovers interaction range of its *push* behaviors and represent it in EC_5^p .

- Visibility of objects drop from 1 to 0 in EC_4^p , which means the objects fall off the table in these situations. This can happen when the objects are pushed and rolled away out of the table. Indeed when the types of the object are inspected in Table 1(b), it is seen that significant number of spheres and lying cylinders create this effect category. Small number of boxes (10) and upright cylinders (15) are also included in the EC_4^p because either they fall down from the edge of the table when pushed or they were at the robot hand as the result of previous *lift* behavior. When *push-forward* is activated, all robot angles including fingers are set to their initial positions, the hand will be opened and the object will drop from the hand and fall down to the ground. Note that the later effect of object drop is an emergent one, ie. the release behavior is not deliberatively intended by the behavior designer. This emergent effect category will play a major role during planning in Section 5.3.

- In EC_1^p , EC_2^p and EC_3^p , the Z position of the objects is increased as the result of *push-forward* behavior and at the end of interaction the objects still remain visible as seen in the Table 1(a). When the object types are inspected, it is seen that these effect categories are produced mostly by boxes and upright cylinders. Such effects are not generated when the object is a sphere because spheres always roll-away when pushed, but some lying cylinders can lead to these effects because different orientations of lying cylinders afford either rollability or pushability.

Table 2: EC_i^p represents ith effect category of *lift* behavior. For further explanation please see Table 1.



(a) Effect features

					Width	Dist
EC_1^l	60	10	85	0	16.7	90.0
EC_2^l	0	30	5	45	11.5	88.7
EC_3^l	55	75	160	105	17.8	122.4
EC_4^l	0	90	5	40	20.6	94.3
EC_5^l	65	25	95	50	20.8	88.7

(b) Info. on objects

5.2 Discovered effects categories for lift

Lift behavior, although not represented differently in robot's behavioral repertoire, is conceptually different from *push* behaviors, so the effect categories for *lift* behavior are interpreted separately. The feature values of effect prototypes and the situations in which effect categories are generated are given in Table 2 and the interpretation is as follows:

- EC_3^l describes effects that do not change significantly at all and corresponds to not-reachable

objects. This effect category is similar to EC_5^p for *push-forward* behavior.

- In EC_5^l , the objects become invisible during execution of *lift* behavior. Disappearing from the view was not expected and against our intention in *lift* behavior. There are two different type of situations for existence of such an effect. First, the object may be already in robot hand before execution of *lift* behavior so the initialization step of *lift* can result in falling the object off to the ground. Second, the rollable large objects cannot be grasped by robot hand, but they will be dragged and rolled down the table. The second reason explains the existence of large number of spheres and lying cylinders in EC_5^l .

- In EC_4^l , the height of the object (Y pos) does not change, but its position with respect to the robot changes. In other words, the object is not lifted, but dragged over the table. This effect is created by large ungraspable objects. Different from EC_5^l (previous item), they are not rollable. This claim is supported by large number of boxes (90) and lying cylinders (40) in EC_4^l , and relatively large average width of the corresponding objects (20.6 cm).

- In EC_1^l and EC_2^l , the height of the objects (Y Pos) are increased, so these effect categories correspond to situations where objects were actually lifted. One significant difference between two categories is on the change of perceived width of the objects. The perceived width of the object is decreased more in EC_2^l probably because it is better covered inside hand. Based on the actual average widths of the objects for these effect categories, smaller objects are seen to create EC_2^l . This is consistent with the explanation in decrease of perceived width because small objects are better grasped and tend to disappear inside hand.

5.3 Learning affordances and planning

After effect categories are discovered, the mapping from initial features to these categories are learned by training SVM classifiers χ^{b_i} for each behavior. 800 interactions are used in training and a separate set of 200 interactions are used in testing the classifiers. At the end, in predicting correct effect categories around 90% accuracy is obtained for different behaviors.

The planning capability, which is based on the learned affordance prediction system, is tested and demonstrated in the real robot platform. The robot, infrared range camera, and table are placed similar to the simulated interaction environment. A system with three main modules, namely*Perception*, *Planner* and *Execution Control*, are used for online verification of the approach. The *Perception Module* is connected to the infrared range camera and is responsible for computing the object features and sending them to the *Planner Module*. Moreover, the *Perception Module* informs the *Execution Control Module* about the position of the objects for behavior parameterization. The *Execution Control Module* on the other hand receives a sequence of behaviors (a multi-step plan) from the *Planner Module* and executes the behaviors one by one using the positions received from the *Perception Module*. The *Execution Control Module* also informs the *Planner* when the plan is completed. When the plan completed signal is received, the *Planner Module* is responsible for sending a new multi-step plan to the *Execution Control* using the object features from the *Perception*. In fact, the *Planner Module* generates plans continuously but sends the plans only when plan completed signal is received. Continuous plan generation enables the *Planner* to monitor whether the execution of the original plan proceeds as planned or not. This system is tested with different objects and object placements for two different goals.

Keep the table clean The motivation of the first case study is to keep the table clean. In order to satisfy this goal, the desired value for *object-visibility* feature is set to be 0 (or false). So the planner needs to find a sequence of behaviors which leads to an object state with that particular feature value 0. The snapshots taken from this experiment are provided in Fig. 3. When a ball is placed in the middle of the table, the *Planner Module* selects *push-right* behavior and after the behavior is executed, the ball rolls away and falls off the table. When an upright cylinder is placed almost in the same place, the *Planner Module* generates a two-step plan (*lift* and *push-forward*). First *lift* behavior is executed and the object is lifted. Later, *push-forward* is activated, so the arm and hand joints need to move to their original (initial) position. During the initialization of the behavior, the hand opens and the object falls down. As we discussed earlier, this is rather an emergent behavior that was not planned by behavior designer but discovered by the learning system. In other situations when the object is placed at the edge of the table, *push-right* or *push-left* behaviors are selected and the object is pushed off the table. This experiment verifies that affordances related to physical characteristics of the objects (ball and upright cylinder). Moreover, the characteristics of the environment are also learned through interaction and system makes different plans based on different positions of the cylinder.

Bring the object to a target position The task in this case study is to bring the object to a desired position. The goal is defined over three feature values, that correspond to the 3D position of the ob-

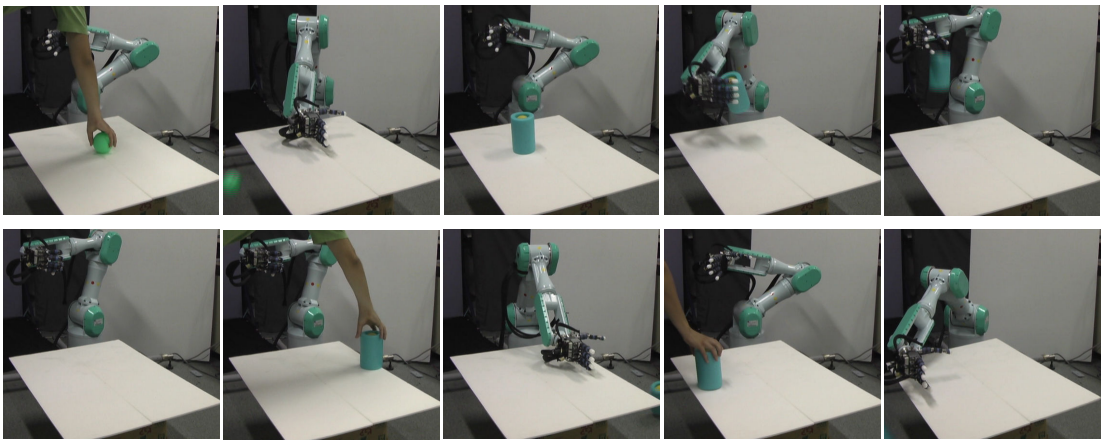


Figure 3: The table is kept clear by setting a desired state where *object-visibility* feature is 0. The corresponding movies can be downloaded from `http://www.kovan.ceng.metu.edu.tr/~emre/epirob09`.

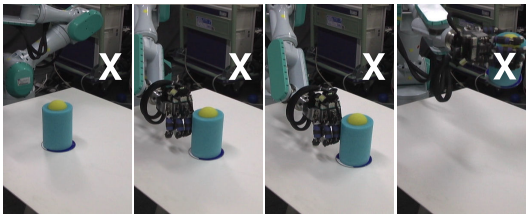


Figure 5: A plan is executed for the task of bringing the object to the target position represented by X.

ject's closest perceived pixel. In the first experiment, the target is set as a point on the table and is shown with a cross (X) in Fig. 4. The object is placed at the back and on the right side of the target from robot's view. In this case, the *Planner Module* generates a 4-step plan which is composed of *push-left* , *push-left* , *push-forward* , *push-forward* . After the plan is executed successfully, the same object is placed on the left side of the target from robot's view, closer to the robot when compared to previous case. The 4-step plan that is composed of one *push-right* and three *push-forward* behaviors is also executed successfully. In both cases, the exact order of behaviors is not important and in fact many different 4-step plans are generated with same behaviors arranged in different orders.

A more complex task description is given in Figure 5 where also desired height of the object is provided in the goal state. In this case, the *Planner* generates a 3-step plan composed of two *push-forward* and one *lift* behaviors. Different from previous case where target position is on table, only one plan with this particular order is generated because a *push-forward* behavior executed after *lift* behavior has the emergent effect of dropping the object from hand.

6. Conclusion

In this paper, we have shown that an anthropomorphic robotic hand can learn the physical affordances of objects from range images and use them to build symbols and relations that can be used in making multi-step predictions about the affordances of objects and achieve complex goals.

First, the robot is shown to discover different effect categories that represent qualitatively different set of situations in a completely unsupervised manner. Furthermore, some effects were not intended by the behavior designer but emerged during interactions. For example, although no 'release' behavior is implemented explicitly, the robot is shown to drop the object from hand during initial stage of the *push-forward* behavior if the the robot was holding the object in its hand.

The mapping between object features and effect categories are later learned by training classifiers which are used to form basic prediction operators. In two case studies, the knowledge that is acquired through learning in the simulator is directly transferred to the real robot. The robot generated multi-step plans for both table cleaning and object moving tasks and executed successfully. Because the robot formed the prediction operator based on its sensory-motor experience, it was able to make grounded and sometimes unexpected emergent plans for the same goal in different situations. These experiments verified that physical properties of the objects and characteristics of the environment are reflected in the learned affordances and generated plans. In future, the method will extended to multi-object environments and more complex non-discrete behaviors.

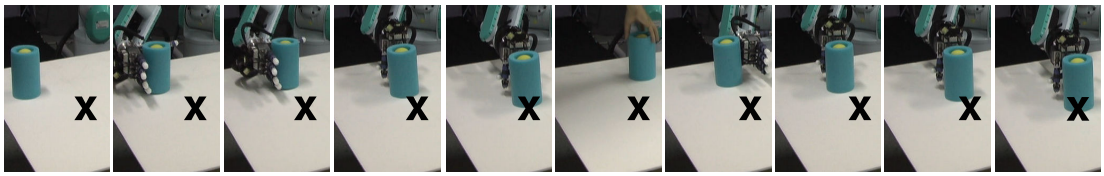


Figure 4: Different plans are executed in different situations for the task of bringing the object to the target position represented by X.

Acknowledgments

This work was partially funded by the European Commission under the ROSSI project (FP7-216125). We would like to acknowledge that the ideas presented in this paper partially shaped through discussions with Maya Çakmak and Mehmet R. Doğar.

References

Asada, M., Hosoda, K., Kuniyoshi, Y., Ishiguro, H., Inui, T., Yoshikawa, Y., Ogino, M., and Yoshida, C. (2009). Cognitive developmental robotics: a survey. *IEEE Transactions on Autonomous Mental Development*, 1(1):12–34.

Şahin, E., Çakmak, M., Doğar, M. R., Uğur, E., and Üçoluk, G. (2007). To afford or not to afford: A new formalization of affordances toward affordance-based robot control. *Adaptive Behavior*, 15(4):447–472.

Culham, J. C. and Valyear, K. F. (2006). Human parietal cortex in action. *Current Opinion in Neurobiology*, 16:205–212.

Fitzpatrick, P., Metta, G., Natale, L., Rao, A., and Sandini, G. (2003). Learning about objects through action -initial steps towards artificial cognition. In *Proc. of ICRA 03*, pages 3140–3145.

Geib, C., Mourăo, K., Petrick, R., Pugeault, N., Steedman, M., Krueger, N., and Wörgötter, F. (2006). Object action complexes as an interface for planning and robot control. In *Workshop: Towards Cognitive Humanoid Robots at IEEE RAS Int Conf. Humanoid Robots*. IEEE RAS.

Gibson, J. (1986). *The Ecological Approach to Visual Perception*. Lawrence Erlbaum Associates.

Goodale, M. A. (2008). Action without perception in human vision. *Cog Neuropsycho*, 25:891–919.

Goodale, M. A. and Milner, A. D. (1992). Separate visual pathways for perception and action. *Trends Neurosci*, 15:20–25.

Griffith, S., Sinapov, J., Miller, M., and Stoytchev, A. (2009). Toward interactive learning of object categories by a robot. In *Proc. of 8th ICDL*, Shanghai, China.

Harnad, S. (1990). The symbol grounding problem. *Physica D*, 42(1-2):335–346.

Klingspor, V., Morik, K., and Rieger, A. D. (1996). Learning concepts from sensor data of a mobile robot. *Machine Learning*, 23(2-3):305–332.

Montesano, L., Lopes, M., Bernardino, A., and Santos-Victor, J. (2008). Learning object affordances: From sensory–motor maps to imitation. *IEEE Transactions on Robotics*, 24(1):15–26.

Oztop, E., Imamizu, H., Cheng, G., and Kawato, M. (2006). A computational model of anterior intraparietal (aip) neurons. *Neurocomputing*, 69:1354–1361.

Pisokas, J. and Nehmzow, U. (2002). Experiments in subsymbolic action planning with mobile robots. Technical report.

Sakata, H., Tsutsui, K. I., and Taira, M. (2005). Toward an understanding of the neural processing for 3d shape perception. *Neuropsychologia*, 43:151–161.

Sinapov, J. and Stoytchev, A. (2008). Detecting the functional similarities between tools using a hierarchical representation of outcomes. In *Proc. of 7th ICDL*.

Sun, R. (2000). Symbol grounding: A new look at an old idea. *Philosophical Psychology*, 13(149–172).

Ungerleider, L. G. and Mishkin, M. (1982). *Two cortical visual systems*, pages 549–586. Cambridge MA: MIT Press.

Uğur, E., Doğar, M. R., Çakmak, M., and Şahin, E. (2007). Curiosity-driven learning of traversability affordance on a mobile robot. In *Proc. of ICDL'07*.