Self-discovery of motor primitives and learning grasp affordances

Emre Ugur[1,2,3], Erol Şahin[3], and Erhan Oztop[4,1,2]

[1] ATR, NIA, Dept. of Dynamic Brain Imaging, Kyoto, Japan
[2] NICT, Advanced ICT, Kyoto, Japan
[3] KOVAN Research Lab., Computer Engineering, Middle East Technical University, Ankara, Turkey
[4] Ozyegin University, Istanbul, Turkey
Emails: emre@atr.jp, erol@ceng.metu.edu.tr, erhan.oztop@ozyegin.edu.tr

***Abstract*— Human infants practice their initial, seemingly random arm movements for transforming them into voluntary reaching and grasping actions. With the developing perceptual abilities, infants further explore their environment using the behavior repertoire they have developed, and learn causality relations in the form of affordances, which they use for goal satisfaction and motor planning.**

This study proposes and implements a developmental progression on a robotic system mimicking the aforementioned infant development stages: An anthropomorphic robot hand with one basic action of swing-hand and the palmar reflex (i.e. the enclosure of the fingers upon contact) at its disposal, executes swing-hand action targeted to a salient object with different hand speeds. During the executions, it monitors the changes in its sensors, automatically forming behavior primitives such as 'grasp', 'hit', 'carry-object' and 'drop' by segmenting and differentiating the initial swing-hand action. The study then focuses on one of these behaviors, namely grasping, and shows how further practice allows the robot to learn affordances of more complex objects, which can be further used to make plans to achieve desired goals using the discovered behavior repertoire.

I. INTRODUCTION

Newborns have several innate reflexes such as pupil reflex to light, sucking reflex or palmar-grasp reflex that help the development of motor and cognitive skills. Palmar reflex in particular is "integrated into later intentional grasping" [1, p. 7] after repeated activation of the reflex and execution of grasp action. This reflex is not always stable and by 6 months of age, it disappears [2, p. 199]. It takes 9 months for infants to reach for objects with correct hand-orientation and adjust their grip size based on objects' size before contact [2]. Hand orientation and grip size appear to develop later than hand-speed parameter since "babies younger than 9-months lack a fully-developed map between visually perceived orientations and corresponding hand orientations" [2, p. 200]. Between 7-9 months, babies explore the environment and objects using various behaviors including grasp, drop, and hit [3]. This indicates that, by this time, the infant has already transformed their initial seemingly uncontrolled 'move hand' actions into a set of behavior primitives from its most basic movement primitive, 'move arm'. Between 7-9 months, they learn the causality relations and object dynamics in response to their actions [3]. It is plausible to think that while interacting with the environment, babies monitor these consequences of their actions and relate the consequences to the visual properties of the objects they interact with. In other words, they learn object affordances, the action possibilities offered by their environment [4], in this phase.

Infants between 7-10 months have also acquired a set of behaviors that are qualitatively different and that can be used for different purposes such as grasping, dropping, reaching, shaking, etc. These actions can be considered as behavior primitives that are utilized to develop more advanced skills through practice. There is evidence that complex behaviors are represented in a modular fashion by the central nervous system. For example, the 'transport' and 'grasp' components of grasping action appear to be controlled by different regions of the human brain [2, p. 217]. Furthermore, there is a developmental order in maturation order of these areas. Thus, it's plausible that the infant starts from a small number of reflex like behaviors, and then progressively discovers and distinguishes new behaviors through use of old ones. Such developmental progression must be complemented in infant's perceptual system. First, a crude perception system should be employed to discover the basic behaviors, and a more advanced perception should be utilized to differentiate more complex behaviors and to discover more abstract concepts such as affordance relations.

During the recent years, robotic studies inspired by ideas in developmental psychology have increased considerably ([3], [5]). These studies argue that a robotic developmental pathway similar to infants is the right way for obtaining intelligent robots. They typically use exploration, learning and embodiment to enable robots learn about their environment via interaction with minimal expert knowledge.

This paper focuses on design of such a developmental robotic system which starts with only one basic reflex-like behavior (*swing-hand*[1]) and one basic reflex (*palmar-grasp reflex*). By exercising this behavior with different hand speeds (which corresponds to maturation of 5-month-old infants), and by using its crude tactile perception, the robot is able to

[1] "Even in the newborn infant, a basic neuro-muscular infrastructure for reaching and grasping is present. When an object is placed in the palm of a newborn infant, the tactile stimulation triggers a grasp reaction in which all digits are flexed around the object. Similarly, in newborns, reaching movements aimed towards objects within the center of the visual field are present. "[6, p. 235]

form a useful set of behavior primitives. In developmental phase II, the robot focuses on one of these primitives, namely grasp behavior. In this phase, which corresponds to 7-10 months in infants, the robot learns the affordances of various objects by executing the *grasp* behavior, and learning the relations between object features and the effects created. Here the robot uses a more advanced visual perception, which is inspired from *surface orientation selective* neurons of CIP area[2], in order to learn these complex relations.

The outline of this paper is as follows. In the next section, we will give the details of the robot, it's behavior and perceptual representation. In Section III, the two phased developmental approach that is developed for (1) discovering behavior primitives and (2) learning grasp affordances will be detailed. In Section IV, after providing the details of the learning experiments conducted in the simulator, the discovered behavior primitives and affordance prediction performance will be discussed. Next, the robot's performance in the real world after learned skills are transfered to the real robot will be demonstrated. We will conclude by summarizing the results and our contribution, and by discussing the relation of our study to similar ones.

II. EXPERIMENTAL FRAMEWORK

An anthropomorphic robotic system equipped with a range camera is used as the experimental platform. This system uses a 7 DOF Motoman robot arm, that is placed on a vertical bar similar to human arm as shown in Figure 1. A five fingered 16 DOF Gifu robot hand is mounted on the arms to enable manipulation. The maximum length of Motoman arm and Gifu hand is 123 cm. and 23 cm., respectively. There are tactile sensors distributed on the surface of the fingers and palm. For environment perception, an infrared range camera (SR-4000), with 176x144 pixel array, 0.23° angular resolution and 1 cm distance accuracy is used.

The simulator (Figure 2), developed using Open Dynamics Engine (ODE) library, is used during the exploration phase. The range camera is simulated by sending a 176×144 ray array from camera center with 0.23° angular intervals.

A. Behavior representation

The robot is assumed to have the ability to reach and bring the objects to it. *Swing-hand* behavior is used for this purpose where the robot reaches to the object and pulls back its hand. This behavior is implemented to generate minimum jerk trajectory in joint space using the inverse kinematic solution [8] from initial and final positions as in Figure 1. The hand that is either clenched or wide open prior to the behavior execution can reach to the object in different velocities. Due to an built-in *grasp-reflex*, if the robot feels anything in its palm using touch sensors, the hand is closed. Furthermore, at any moment, this reflex can be disabled randomly and in this case the robot hand is loosened even if there is an object inside.

[2]CIP area on the dorsal path of monkeys extracts visual data and forwards it to AIP that is thought to be responsible from perception of graspability affordance [7]

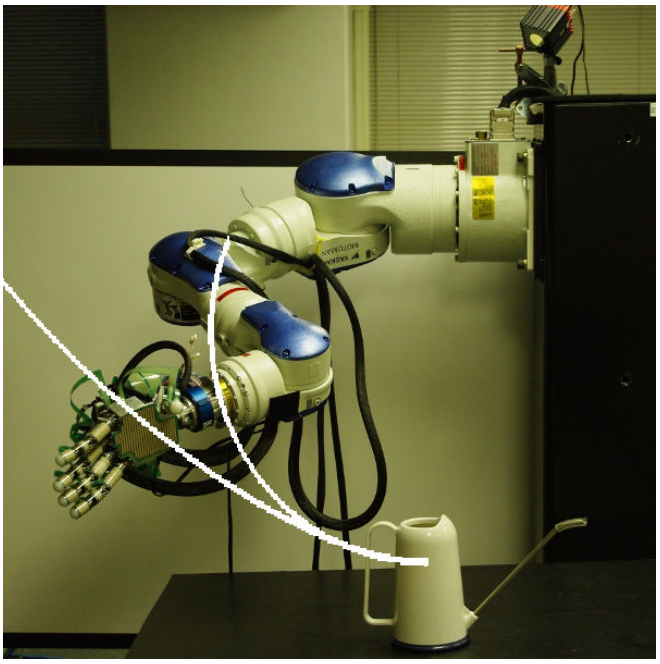
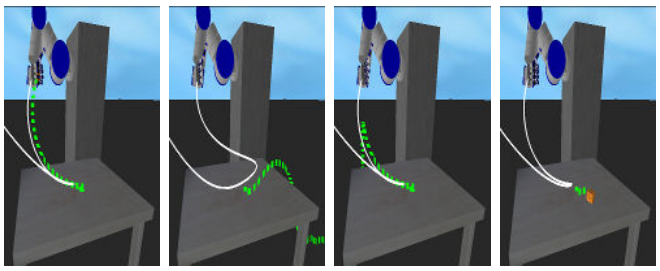


Fig. 1. The hand, arm, range camera (top-right), and the robot environment. The trajectory of the basic *swing-hand* behavior is also show is also shown.

The execution of the same *swing-hand* behavior over the same object with different parameters (velocity, disable-reflex, and initial hand state) produces different effects. Figure 2 shows hand and object trajectories during execution of the same behavior with four different parameter sets. In (a), the hand hits the object with a velocity of $V = 0.24$ cm/s, and due to the *grasp-reflex* the object was grasped and brought back. In (b), the high-velocity ($V = 0.42$ cm/s) collision between the hand and the object did not allow the object to be grasped on time. In (c), the hitting/reaching velocity of the hand was same with (a), so the object was grasped. However, while pulling back the hand, the *grasp-reflex* was disabled (randomly) so the object was released. Finally, the hand was initially clenched in (d), so the object was tapped only. Although the trajectory of the object being interacted can differ through experiments, these four different situations qualitatively cover all different possibilities if perfect reaching and tactile sensing is assumed.

How the objects are affected from the execution of the same behavior, depends on the free parameters of these behaviors. While *swing-hand* behavior has one parameter (hand speed), the discovered behavior primitives may have more than one parameter. Each behavior primitive is represented as $b_i(\boldsymbol{\alpha})$ where i corresponds to the index of the primitive and $\boldsymbol{\alpha}$ corresponds to parameter list. Since the behaviors are discovered from the *swing-hand* behavior by using the same method, they have a common encoding. In other words, each behavior primitive is encoded with a common set of descriptors that are automatically instantiated during behavior discovery. These descriptors are:

- initial and final touch states,
- initial and final hand velocities,



(a) Grasp (b) Hit (c) Drop (d) Tap

Fig. 2. Robot-hand and object trajectories during *swing-hand* behavior with different velocities and hand states. The labels that explain situations are only given to ease the understanding of the figures and are not used in any phase of development. Corresponding sensor trajectories and behavior primitive segments are provided in Figure 3.

- whether *grasp-reflex* is disabled, and
- hand movement direction (towards object or robot).

B. Perceptual Representation

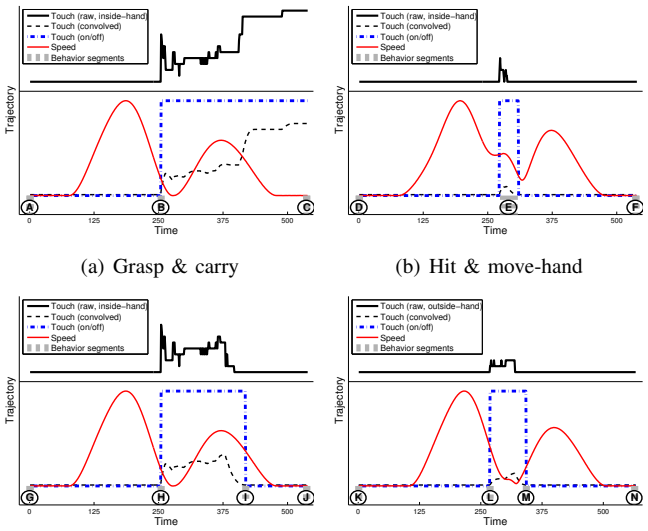
1) Touch perception: The sensory readings obtained from the distributed tactile sensors that cover the palm and fingers are processed as follows.

First, touch for each finger link and for the palm is detected and encoded as $5 \times 3 + 1 = 16$ binary touch signals. Then, their sum is normalized between $[0 - 1]$ to represent a general touch sensation for the hand, and called as *raw* touch signal. Since the readings are noisy, the *raw* touch trajectory is convolved with the first half of Gaussian window, $G(N = 50, \sigma = 0.5)$: $y_i = x_i \times e^{\frac{(i-M)^2}{2\sigma N}}$ where $M = (N-1)/2$. *Convolved* readings are used in perception during *affordance-learning phase*, Phase II.

On the other hand, in Phase I, the robot has a crude sense of touch as mentioned before. This limited sense is represented by three touch states, which are computed from binary touch signal, to ease the representation of this change. In order to compute the binary signal, the *convolved* trajectory is discretized to *on/off* sensor using a threshold $t = 0.02$. Figure 3 shows a number of sample trajectories where the *raw* values are *convolved* and *binarized*.

Touch states are defined based on the binary touch signal trajectory as follows:

- **on:** The touch sensor that is active during behavior execution.
- **off:** The touch sensor that is not active during behavior execution.
- **onf:** In some cases, the touch sensor is active for a short duration. Such cases are denoted by **on/off** or **onf** in short. There is such a case in Figure 3(b), change (E), where the robot hand hits the object with high velocity. A similar case in Figure 3(d) occurs, however since the approach velocity is lower, the object is dragged on the table rather then been hit. So the duration of touch is long in this case and hand state is represented by the consecutive **on** and **off** states instead of the **onf** state.



(a) Grasp & carry (b) Hit & move-hand

(c) Grasp & release & move-hand (d) Punch & drag & move-hand

Fig. 3. Velocity, touch sensor trajectories and segmentation of *swing-hand* behavior with different parameters.

2) Visual perception: The range image obtained from the infrared range camera is used to compute a number of features from the objects in the environment as follows:

Object detection: The first step of pre-processing is to filter out the pixels whose confidence values are below an empirically selected threshold value. Then the pixels outside the region of interest (table) are filtered out. As a result, the remaining pixels of the range image would belong to one or more objects that are segmented by the Connected Component Labeling algorithm [9]. In order to reduce the effect of camera noise, the pixels at the boundary of the object are removed, and the Median and Gaussian filters with 5×5 window sizes are applied. Finally, a feature vector for each object is computed using the positions of the corresponding object pixels as detailed in the next paragraph.

Object features: The object feature vector includes a binary feature for *object-visibility*, and a number of features related to size, position and shape of the object. The 3D position of the object center is used to represent the position feature. As for the shape features, inspired from CIP neurons in monkey brain, first, object surfaces are identified, then size, orientation, and curvature of each surface is computed. Object surfaces are found by grouping object pixels with similar local orientations. For this purpose, for each detected object pixel a local normal vector is computed ([10]) and a clustering algorithm is used to find 'normal vector clusters' that correspond to object surfaces. See Algo. 1 for the details of this algorithm and Figure 7 for sample surfaces. After finding surfaces, for each cluster (surface), standard deviation of the normal vectors is computed in each coordinate axis to represent curvature. Additionally, for each surface, height and width are computed. At the end, the following feature vector represents the object shape perception:

$$\boldsymbol{S} = (a_1, w_1, h_1, \boldsymbol{\mu}_1, \boldsymbol{\sigma}_1, ...a_s, w_s, h_s, \boldsymbol{\mu}_s, \boldsymbol{\sigma}_s)$$

where a_1, w_1 and h_1 correspond to number of pixels, width and height of the first detected surface, $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ refer to mean and standard deviation vectors (of size 3), respectively. s is the maximum number of surfaces, which is set to 3. Therefore, the object feature vector includes *object-visibility*, position and shape features, and has size of $1+3+(1+1+1+3+3)\times 3 = 30$. Although, this feature representation is limited with the number of surfaces and their order, we expect to obtain generalization in learning by including extra object feature vectors that are obtained from the original computed feature vector with shuffled surface orders.

Algorithm 1 Surface identification through normal vector clustering

s_{id} = Index of the surface (cluster).
s_{normal} = Normal of the corresponding surface (the mean of the cluster).
$\{\boldsymbol{N}\}$: The set of normal vectors computed for each pixel.
$cluster(A,k)$: Cluster sample set A into k clusters. Return the list of cluster index and means.
$dist(s_i, s_j)$: Distance between normal vectors of surfaces s_i and s_j.
$threshold_{dist}$: A threshold to decide the similarity of surfaces.
$nSameSurfaceNeighbors(p, s_id)$: Number of pixels with same s_{id} in 8-neighborhood of p

```
1: ({s_id}, {s_normal}) = cluster(N, 3)
2: for each surface pair < s_i, s_j > in surface list {s_id} do
3:    if dist(s_i, s_j) < threshold_dist then
4:       Combine s_i and s_j in {s_id}
5:    end if
6: end for
7: for each surface s_i in surface list {s_id} do
8:    for each pixel p in surface s_i do
9:       Delete p from s_i if nSameSurfaceNeighbors(p, s_i) < 3
10:   end for
11: end for
```

3) Entity Feature Vector Computation: Entity feature vector includes robot's visual and tactile percept in progressively increasing complexities in subsequent developmental phases:

In Phase I, entity feature vector is represented only with one feature, *touch state* (T):

$$\boldsymbol{f} = (T)$$

In Phase II, the *convolved* touch signal is used as tactile feature instead of *touch-state*. In this phase, the object perception is more complex and includes position and shape related information: $\boldsymbol{f} = (C, V, \boldsymbol{P}, \boldsymbol{S})$ where C, V, $\boldsymbol{P}$, and $\boldsymbol{S}$ represent *convolved* touch signal, *object visibility*, 3D object position, and object shape feature vector, respectively.

4) Effect Feature Vector Computation: Effect corresponds to the difference between final and initial perception of the robot and is defined as the vectorial difference between final and initial features: $\boldsymbol{f}^{b_i}_{\text{effect}} = \boldsymbol{f}^{(b_i)} - \boldsymbol{f}^{()}$, where $\boldsymbol{f}^{(b_i)}$ represents the feature vector of the entity perceived after b_i behavior is executed. Here 'after' refers to the timepoint where there is no change in perception anymore.

C. Affordance Representation

The affordances are represented as triples that consist of the initial percept of the agent (*entity*), the *behavior* applied and the produced *effect* [11]. Here, the *entity* corresponds to the feature vector which includes object features and robot's tactile sensor readings. The *entity* feature vector is represented as $\boldsymbol{f}$. The *effect* feature vector ($\boldsymbol{f}^{b_i}_{\text{effect}}$) represents the change in perception of the entities during behavior execution. Thus the affordance relation instance, which represents a sample interaction with the environment, will be represented as follows:

$$\{< \boldsymbol{f}^{b_i}_{\text{effect}}, \boldsymbol{f}, b_i(\alpha) >\}$$

III. TWO PHASE LEARNING

A. Phase I: Discovering Behavior Primitives

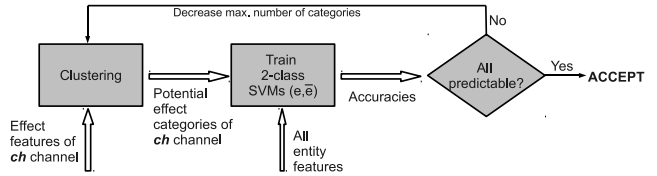
In this phase, the robot executes the *swing-hand* behavior on the same small object placed in a fixed reachable position. Different hand speeds and grasp-reflex disabling timings generate different hand movement trajectories. These different movement trajectories result in various hand-object interactions that can be measured with robot's touch perception. Thus, by observing the changes in touch state and segmenting the movement trajectories based on these changes, the robot is able to distinguish the segmented trajectory instances that transform one hand-object state to another one. In order to obtain generic behaviors, the robot finds the segments with the same initial and final touch states, i.e. the segments which creates same transformation between touch states. Finally, it groups these common segments to form behavior primitives.

Figure 3 shows a number of obtained segment instances from behavior executions demonstrated in Figure 2. As shown, from different executions and different segmentations, segment instances with common characteristics are observed. For example, the first segments obtained in (a) and (c) are similar since they correspond to the touch stage change of **off** $\rightarrow$ **on**. As another example, the last segments obtained in (b), (c), and (d) correspond to the common change (or no change) of **off** $\rightarrow$ **off**. These similar segments will be grouped together in order to create new behavior primitives. The representative behavior primitive of each group, which is composed of many individual examples, is computed by taking the average of initial and final velocities.

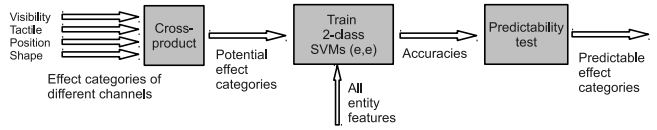
B. Phase II: Learning Affordances

In this phase, the robot executes the discovered behavior primitives on different objects of different positions, orientations and size in order to learn which affordances they provide. This learning takes place in 2 steps, where first discrete effect categories are found, and then given objects and behaviors, the means to predict effect categories is learned. The details of these steps are as follows:

Effect category discovery: In the first step, for each behavior b_i, a number of discrete effect categories $E^{b_i}_{id}$ are found using the set of effect feature vectors ($\{\boldsymbol{f}^{b_i}_{\text{effect}}\}$) observed during interactions. A novel 2-level clustering algorithm that takes into account the representational differences between different perceptual channels is used for this purpose since one-level clustering method is sensitive to relative weighting of the effect features that are encoded in different units. This algorithm first finds *channel-specific*



(a) *Channel-specific effect category* discovery for each channel *ch* in the lower-level



(b) *All-channel effect category discovery in the upper-level*

Fig. 4. 2-level effect category discovery method where obtained effect categories are tested based on their predictability

effect categories within each perceptual channel and then combines them to obtain *all-channel effect categories*. In detail, in the lower level *channel-specific effect categories* are found by clustering in the space of each channel, discovering separate categories for touch, visibility, 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 to be not predictable, the clustering is re-done with less number of desired clusters (Algorithm 2, Figure 4(a)). 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. If an *all-channel effect category* is not predictable, it is discarded and the corresponding affordance relation instance will not be used in the next step.

Learning effect category prediction: In the second step, classifiers are trained to predict the effect category for a given entity feature vector, a behavior id and behavior's parameter by learning the $(\boldsymbol{f}^{()}, \alpha) \rightarrow E_{b_i,id}$ mapping. Specifically, we used a Support Vector Machine (SVM) classifier with Radial Basis Function (RBF) kernel to learn this mapping for each behavior b_i, where $(\boldsymbol{f}^{()}, \alpha)$ is given as the input, and the corresponding $E_{b_i,id}$ as the target category.

IV. EXPERIMENTS

A. Phase I: Discovered Behavior Primitives

When only open-hand *swing-hand* behavior execution is considered, the approach velocity is randomly selected from the range $[0 - 0.70]$, six different behavior primitives were discovered by the robot. As mentioned before, the behavior types are automatically discovered by the robot based on differences in initial and final touch states. Figure 5 gives these primitives with their touch state change characteristic. For each primitive, the initial and final velocity distributions are given on the left and right plots of the figure, respectively. The meaningful labels for these primitives are also provided.

- **Hit** primitive corresponds to high velocity reach and hit to the object. The touch sensor was activated for short time and the object was not grasped at the end.

Algorithm 2 Discovery of *channel-specific effect categories*

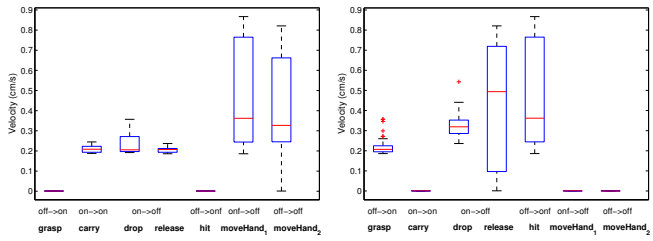
k_{max}: Maximum number of categories.
Reset(k_{max}): Reset k_{max} for new channel.
$f_{\text{effect}}(ch)$: Portion of feature vector limited to channel ch.
$Clusters(\{\boldsymbol{f}\}, k_{\text{max}})$: Find between 1-$k_{\text{max}}$ clusters with feature set $\{\boldsymbol{f}\}$.
$\{E_{id}\}^{i}_{ch}$: The set of effect categories found in channel ch.

for each channel ch in $[visibility, tactile, position, shape]$ **do**
  Reset(k_{max})
  while $k_{\text{max}} \neq 1$ **do**
    {STEP 1: Find optimal categorization by clustering n_{opt} times}
    for $i = 1 : n_{\text{opt}}$ **do**
      Find effect category set $(\{E_{id}\}^{i}_{ch})$ by $Cluster(\{f_{\text{effect}}(ch)\}, k_{\text{max}})$.
      Train classifier $(Predictor()^i)$ to learn mapping $(f, \alpha) \rightarrow E_{id}{}^{i}_{ch}$.
    end for
    Select effect category set $(\{E_{id}\}^{best}_{ch})$ with most accurate $Predictor()$
    {STEP 2: Verify predictability of the optimal categories in $\{E_{id}\}^{best}_{ch}$}
    for each effect category e in $\{E_{id}\}^{best}$ **do**
      Assign same category $(\overline{a})$ to all effects except e
      Train classifier to learn mapping $(f, \alpha) \rightarrow e, \overline{e}$
      if accuracy $<$ threshold **then**
        $k_{\text{max}} = k_{\text{max}} - 1$
        jump (3) {Categorization does not allow prediction}
      else
        continue {Check predictability of next category}
      end if
    end for
  end while
end for

- **MoveHand**$_1$ primitive corresponds to hand movement towards robot body without object after short duration touch to the object.
- **Grasp** primitive corresponds to slow velocity reach to the object and results in long period touch sensor activation, i.e. stable grasp.
- **Carry** primitive starts with slow velocity and stable grasp. When the primitive execution finished, the object is still at robot's hand indicated by **on** touch state.
- **MoveHand**$_2$ primitive corresponds to hand movement towards robot body without any object in the beginning and at the end.
- **Drop/release** primitive corresponds to hand movement towards robot, object in the hand in the beginning and no object at the end. The object falls as the result of unstable grasp in **drop** behavior in some situations, and as the result of disabled *grasp-reflex* in **release** behavior in other cases. We assumed that the robot can notice and and learn from the disabled grasp-reflexes.

Closed-hand *swing-hand* behavior was segmented to five different primitives, namely one tap, two drags, and 2 move-hands. Different from open-hand segments where the object drops due to unstable grasps, in closed-hand segments, the object may be dragged by robot's fist for certain time. The behavior segments are similar to previously found ones, so we will focus on open-hand *swing-hand* behavior only.

Figure 5 gives the velocity distributions for all experienced segments. Consider the final velocity (reach velocity)



(a) Initial velocity distribution (b) Final velocity distribution

Fig. 5. The distribution of hand velocities in the beginning and at the end of behavior primitive executions.

TABLE I

THE BEHAVIOR PRIMITIVES AND THEIR ENCODING. GR REFERS TO ACTIVATION OF GRASP REFLEX.

Name	Init Touch	Final Touch	Init Vel.	Final Vel.	GR	Dir.
Grasp	off	on	[0-0]	[0.19,0.22]	on	obj.
Carry	on	on	[0.19-0.21]	[0,0]	on	self.
Drop	on	off	[0.22-0.27]	[0.28,0.35]	on	self
Release	on	off	[0.19-0.21]	[0.10,0.70]	off	self
Hit	off	onf	[0-0]	[0.24-0.76]	on	obj.
MoveHand$_1$	onf	off	[0.25-0.46]	[0,0]	on	self
MoveHand$_2$	off	off	[0.25-0.65]	[0,0]	on	self

distribution of **hit** and **grasp** behavior primitives. Ideally, the velocity of the hand for grasping should be always smaller than the velocity for hitting. However, as shown, the velocity ranges of these primitives are not clearly separable and contain overlapping parts because of the complex interaction dynamics and noise in the system. While the robot hand could **grasp** the object with an approach velocity of $0.36cm/s$, it could not grasp with $0.19cm/s$ in some situations. Thus, in order to increase the confidence we include the portion between first and third velocity quartiles into behavior primitive descriptors.

Table I gives the set of descriptors that are automatically set for the corresponding discovered behavior primitive. As shown, the robot hand which approaches to the object with $[0.19-0.21]cm/s$ velocity would **grasp** the object, however if the approach velocity is $[0.24-0.76]cm/s$, the object cannot be grasped and the robot hand would **hit** it. As another example, even if the touch state is **on** initially, if the hand velocity is high ($[0.22, 0.27]cm/s$), it corresponds to high-speed grasp attempt, i.e. unstable grasp, thus the object **drops** from the hand.

B. Phase II: Learned Affordances

In order to learn grasp affordances, a table with 100×70 cm^2 surface area was placed with a distance of 40 cm in front of the robot. At the beginning of each exploration trial, one object of random size $[8cm-40cm]$ was placed on a fixed reachable position at random orientation. Since the grasp affordances of boxes of different sizes and orientations is more difficult to learn and predict, large number of boxes are included into interactions. The robot simulated 2000, 400, and 400 **grasp** interactions with boxes, cylinders, and spheres, respectively. Through experiments, the *approach-direction* parameter (α) of grasp is kept random. The set of relation instances were used in learning. The X-means algorithm was used to find channel-specific effect categories, and Support Vector Machine (SVM) classifiers were employed to learn effect category prediction.

TABLE II

THE EFFECT CATEGORIES (CAT.) DISCOVERED BY 2-LEVEL CHANNEL-BASED CLUSTERING.

Channel	Cat.	Prototype vector	2-category accuracy	Predic-table?	Accep-ted?
Visibility	2 cat.	-1 0	90.6 % 90.6 %	√ √	√
Tactile	3 cat.	0.40 0.11 0.00	79.73 % 64.42 % 68.82 %	√ X X	X
	2 cat.	0.38 0.04	80.15 % 80.15 %	√ √	√
Position	3 cat.	[9, 2, 1] [0, 0, 0] [12,-4,12]	67.00 % 81.14 % 79.20 %	X √ √	X
	2 cat.	[2, 1, 0] [12,-3,12]	78.2 % 78.2 %	√ √	√
Shape	3 cat.	Large Large Small	73.47 % 69.77 % 71.27 %	X X X	X
	2 cat.	Large Small	70.5 % 70.5 %	X X	X
	1 cat.	NA.	NA.	√	√

1) Discovered Effect Categories: The 2-level channel based clustering algorithm, detailed in Algorithm 2 is used to find the effect categories. The maximum number of categories, $k_{\max}$, is set to 5 for each channel. The accuracy threshold is set to 75%. The results are given in Table II.

- For visibility channel, X-means algorithm naturally finds 2 categories. The first and second categories correspond to *disappear* and *stay-in-view*, respectively. When an SVM predictor is trained to differentiate the first category from the other categories (from the second one), the prediction accuracy is found to be high. The same is valid for second category as well. Thus, these categories are valid and transformed to upper-level for cross-product operation.
- For tactile channel, X-means algorithm first finds 3 categories. The value shown in prototype vector is the ratio of activated touch sensors on hand. Thus, the first category corresponds to high-activation (grasp) and second and third categories correspond to low-activation (finger touch or no touch). The performance of the SVM classifier that is trained to differentiate the first category from the others (graspable from others) is high (79.73%). However, the SVM classifiers cannot distinguish second and third categories well since the accuracies are around 65%. As a result, maximum number of categories are decreased to 2, and X-means algorithm is executed again. The new categories correspond to high-touch-activation and low-touch-activation, and they are predictable (with 80% accuracy), so they are transformed to upper-level.
- For position channel, X-means algorithm finds 3 poten-

TABLE III

EFFECT CATEGORIES DISCOVERED FOR GRASP BEHAVIOR.

Effect id	Visibility	Tactile	Position	Shape	Comment
Effect 1	-1	0.04	no change	N.A.	Disappeared
Effect 2	-1	0.38	no change	N.A.	Grasped& disappeared
Effect 3	no change	0.38	no change	N.A.	Grasped
Effect 4	no change	0.04	[12, -4, 12]	N.A.	Pushed

tial categories, where two of them cannot be predicted. Thus, in the next iteration 2 categories are found and transformed to upper-level.

- For shape channel, neither 3 category nor 2 category set is found to be predictable. Thus, there is no categorization in shape channel. The failure in discovering any meaningful shape change category is mainly due to robot's inability to track object's surfaces.

After effect categories are found for each channel, they are combined in the upper level by cross-product operation. Some of the new categories are discarded since they do not satisfy *predictability* criteria (see Figure 4(b)). At the end, 4 meaningful effect categories are shown to remain (Table III). Disappeared effect was probably created by large spheres, and grasped & disappeared effect was created by very small object that becomes invisible inside hand.

2) Effect Prediction Performance: After the discovery of effect categories, the mapping from the initial object features to these categories is learned for grasp behavior by multi-class Support Vector Machines (SVMs). The LibSVM software package was used with optimized parameters of the RBF kernel through cross-validated grid-search in parameter space. 2200 simulated interactions were used in training and a separate set of simulated 600 interactions were used for testing. During training and testing, the same number of samples from each category are used to avoid any effect of dominant category. Figure 6 gives the change in accuracy based on number of samples used in training. Considering the chance level is 25% in predicting correct category out of 4 categories, high prediction accuracy is obtained with enough number of samples. Furthermore, since use of perfect object information (such as type of the object, dimensions, etc) does increase the performance only around 5%, the utilized visual features are proved to capture objects' physical affordances.

Note that although shape features did not contribute to effect categorization, they are utilized in this stage in predicting effects and perceiving affordances. For example, in order to detect rollability affordance of a ball placed in the middle of the table, the robot should use surface curvature which is represented by $\boldsymbol{\sigma}_s$; or to detect the graspability of a small object, the width feature should have been used.

C. Transfer of Learned Skills to the Real Robot

The behavior discovery and learning results were transfered to the real robot and the affordance perception is tested in the real world. For this purpose, first a number of objects were placed on the table and the robot predicted the effects to be created by the grasp behavior with fixed approach

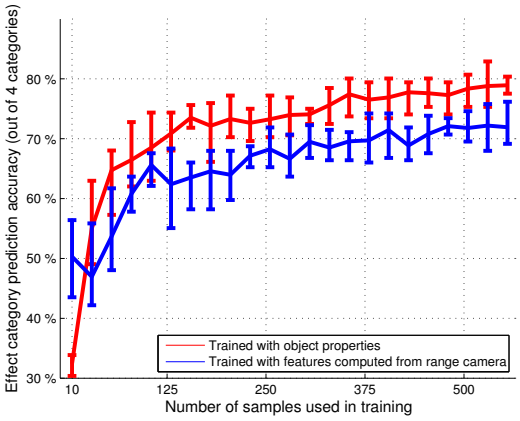


Fig. 6. The change in prediction accuracy of classifiers trained with real object information (such as object type, diameter, etc.) and computed features from range camera.

direction. The detected objects and identified surfaces can be seen on the range image in Figure 7. For those objects, the robot was able to make correct predictions in spite of the facts that (i) the surface identification is not sharp in many cases (e.g. ball) due to the noise in sensing, (ii) some objects were occluded, and (iii) some of the objects (4&5) were novel, i.e. were not used during learning. In this study, the robot learned single object affordances and executed the grasp behavior to one object at a time.

The grasp behavior was also executed on a number of objects with different positions and orientations in order to compare the predicted and actual effect categories. The resulting interactions and actual observed effects are shown in Figure 8 for a number of cases. The effect predictions in (a-c) were correct where a mug was Grasped in (a), a larger cylindrical object was Pushed in (b) and the same cylinder with a different orientation rolled off from the table and Disappeared in (c). In (d), the robot predicted the object to Disappear but the object stuck in between robot fingers and table edge at the end of the grasp action without being grasped (and disappeared). Finally in (e), the robot Pushed the object initially (as predicted), but later the object rotated while being pushed and was grasped from the other side with index and thumb fingers. In the real world with uncertainties and noise, the robot cannot perfectly predict the result of its actions; however it can observe its own action execution, compare the actual effect with the predicted one and perform corrective actions if necessary, and possibly update its prediction mechanisms. We leave such corrective actions and update for future work as they are not at the core of this current report (but see [12]).

V. CONCLUSION

In this paper, a developmental framework was provided for unsupervised discovery of a behavior repertoire and learning of object affordances for a manipulator robot. Initially, the robot possessed one basic action (*swing-hand*) and one basic reflex (*palmar-grasp*). First, by executing this action with dif-

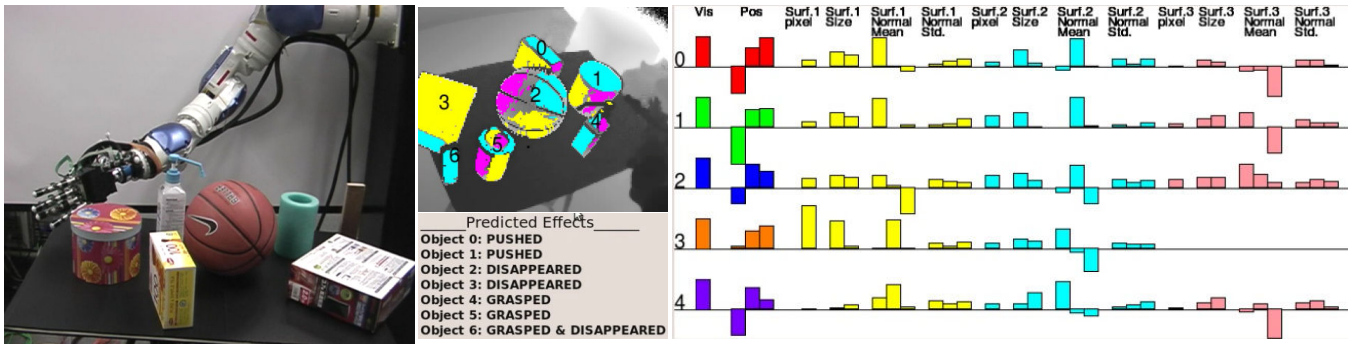


Fig. 7. Left: The test environment. Middle-top: The range image, detected objects and the identified surfaces of these objects. Right: The features computed from the objects and their surfaces. The robot predicts that different effects will be created if grasp behavior is executed, as shown in the middle-bottom figure. Note that the features of objects 5 and 6 are not shown in the figure due to the space constraints in the user interface.

(a) (b) (c) (d) (e)

Fig. 8. The actual effects of grasp behavior executions on different objects with different orientation and positions. The complete movie which also displays robot's perception, object features, and effect predictions with larger number of interactions can be downloaded from the url below. Note that the grasp behavior execution only includes reaching to the object and closing the hand. In other words, the execution is finished as soon as the hand is closed. The hand that is automatically opened and taken to initial position is not part of the grasp execution and experiment and dropping object after execution does not refer to grasp failure. Thus, for example out of 4 grasp predictions, only one of them were unsuccessful due to prediction error.
http://www.emreugur.net/videos/iros2012/

ferent hand speeds, and monitoring its crude tactile sensors, the robot was able to discover meaningful behavior primitives such as *carry*, *grasp*, *release*, and *push*. Second, by focusing on the *grasp* behavior, the robot learned affordances of more complex objects by first finding discrete effect categories and learning the relation between these categories and interacted object features. In summary, the robot discovers behavior primitive, effect categories and affordances, by progressively dividing behavior parameter space, effect feature space, and entity feature space. The robot uses the structures found in the previous steps in order to discover in skills. After learning, the skills are transferred to the real robot and shown to be effective in real world.

In many other studies, the robots learned affordances in terms of the relations between object features, behaviors and created effects. Furthermore, in [12], the robot was able to make simple plans using the learned affordances. However, in all these studies, a fixed set of hand-coded behaviors were assumed to exist. The findings of this paper (unsupervised emergence of the behavior repertoire from one hand-coded action) can be incorporated into those approaches to obtain a developmental progression that parallels infant development.

REFERENCES

[1] J. Piaget and B. Inhelder, *The Psychology of the Child*. New York, USA: Basic Books, 1966.

[2] D. A. Rosenbaum, *Human Motor Control*. London, UK: Academic Press, 1991.

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

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

[5] A. Stoytchev, "Some basic principles of developmental robotics," *IEEE Transactions on Autonomous Mental Development*, pp. 122–130, 2009.

[6] B. Vollmer and H. Forssberg, "Development of grasping and object manipulation," in *Sensorimotor Control of Grasping: Physiology and Pathophysiology*. Cambridge University Press, 2009.

[7] E. Oztop, H. Imamizu, G. Cheng, and M. Kawato, "A computational model of anterior intraparietal (AIP) neurons," *Neurocomputing*, vol. 69, pp. 1354–1361, 2006.

[8] B. Moore and E. Oztop, "Redundancy parametrization for flexible motion control," in *ASME IDETC*, 2010.

[9] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision, Volume I*. Addison-Wesley, 1992.

[10] E. Ugur and E. Şahin, "Traversability: A case study for learning and perceiving affordances in robots," *Adaptive Behavior*, vol. 18, no. 3-4, 2010.

[11] E. Şahin, M. Çakmak, M. R. Doğar, E. Ugur, and G. Üçoluk, "To afford or not to afford: A new formalization of affordances toward affordance-based robot control," *Adaptive Behavior*, vol. 15, no. 4, pp. 447–472, 2007.

[12] 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.