2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Using Learned Affordances for Robotic Behavior Development

Mehmet R. Doğar
University of Genoa
and
Italian Institute of Technology
Genova, Italy
Email: dogar@ceng.metu.edu.tr

Emre Ugur, Erol Şahin
Kovan Research Laboratory
Dept. of Computer Engineering
Middle East Technical Unv.
Ankara, Turkey
Email: {emre,erol}@ceng.metu.edu.tr

Maya Çakmak
College of Computing
Georgia Institute of Technology
Atlanta, GA 30318, USA
Email: maya@cc.gatech.edu

***Abstract*— "Developmental robotics" proposes that, instead of trying to build a robot that shows intelligence once and for all, what one must do is to build robots that can develop. These robots should be equipped with behaviors that are simple but enough to bootstrap the system. Then, as the robot interacts with its environment, it should display increasingly complex behaviors. In this paper, we propose such a development scheme for a mobile robot. J.J. Gibson's concept of "affordances" and a formalization of this concept provides the basis of this development scheme. We show that an autonomous robot can start with pre-coded primitive behaviors, and as it executes its behaviors randomly in an environment, it can learn the *affordance relations* between the environment and its behaviors. We then present two ways of using these learned structures, in achieving more complex, intentional behaviors. In the first case, the *sequencing* of these primitive behaviors are such that new more complex behaviors emerge. In the second case, the robot makes a "blending" of its pre-coded primitive behaviors to create *new* behaviors.**

I. INTRODUCTION

The objective of this work is to propose a robotic development scheme which is based on the concept of affordances. Starting from a set of simple pre-coded behaviors/actions, through interaction and experience, we aim to realize a transition from these unintentional behaviors to intentional behaviors for the robot. This development should also result in demonstration of novel/enriched behaviors that are different from the pre-coded existing behaviors. On a more conceptual level, our objective is to contribute to the view which suggests that robots, just like human beings and other animals, should go through a developmental process, where they shape their "intelligence" through their own experience. In proposing the behavioral development of a robot, we placed the *affordance* concept, which provides us with a tool to deal with robotic problems in terms of agent-environment interactions, and a recent formalization of the concept [1], at the core of our study.

II. BEHAVIOR DEVELOPMENT AND AFFORDANCES

How behavior develops in humans and other animals have been the subject of many scientific studies. Theories of development have been proposed in the area of psychology. At the level of the nervous system, neuroscience has investigated behavior control and motor development. In robotics also, there have been efforts to make robots learn and develop behaviors. Some of these studies also refer to affordances, in the context of development.

A. Behavior development in psychology

According to Piaget's *theory of cognitive development*, existing structures called *schemata* are transformed by the processes of *assimilation* and *accommodation* through interaction with the external world. For Piaget, the development of behavior also occurs in this framework [2]. The newborn baby has existing structures in the form of innate reflexes. It executes and tries these reflexes and primitive behaviors, trying to accommodate them to the environment. As the baby experiments with these behaviors, they differentiate into more complex behavioral structures [3].

E.J. Gibson was the first one to investigate affordances in the context of development. She studied the mechanisms of the *learning of affordances* and used the ecological approach to study child development. E.J. Gibson believed that babies have innate exploratory activities, such as mouthing, reaching and shaking, and they use these to gain perceptual data. She suggested that these activities bring about "information about changes in the world that the action produces" [4]. As development proceeds, exploratory activities become performatory and controlled, executed with a goal.

B. Motor control in neuroscience

In executing a motor behavior, the central nervous system commands the muscles through the motor neurons. This is a difficult problem, since it constitutes a mapping from a small number of variables (motor goals) to a large number of variables (neuron signals) that drive multiple muscles [5].

An approach that tries to explain how complex patterns of motor behavior emerge says that, these complex patterns are actually the result of combining more simple primitive actions [5]. For example, in [6] Mussa-Ivaldi et al. found that when separate modules in the spinal cord of a frog are stimulated one-by-one, they correspond to a limited number of force patterns and motor actions. But when two modules are stimulated simultaneously, the resulting force pattern corresponds to the vector summation of the individual force patterns of each individual module. Mussa-Ivaldi et al. viewed this as a support to the view that "central nervous system may generate a wide repertoire of motor behaviors through the vectorial superposition of a few motor primitives stored within the neural circuits" [6].

Another study that supports this position is the work of "population coding" by Georgopoulos et al. [7]. Through experiments they conducted on rhesus monkeys Georgopoulos et al. found that the arm movements of the monkey can be predicted using the activation values of a population of neurons in the monkey brain. In this population of neurons, it was observed that each individual neuron has a preferred direction, and when it fires it makes the monkey arm move towards that direction. But when multiple of these neurons fire together, it was seen that the resulting direction of the monkey arm was a weighted sum of each individual neuron's preferred direction. Moreover, these weights were given by the activation values of each neuron.

C. Behavior learning and development in robotics

In robotics, there have been increasing interest in behavioral development and learning in recent years.

In [8] Likhachev et al. use case-based reasoning in learning parameters for their goal-directed navigation behaviors. The robot starts with cases which output random behavior parameters. Then, the performance of each case is computed, and a gradient-ascent search is made over the output behavior parameters of these cases to find the best parameters. In another study [9] reinforcement learning is used for "purposive behavior acquisition". In this study, the robot has a fixed set of behaviors, and using these navigational behaviors it aims to shoot a ball into a goal. At the beginning the robot does not know when to execute which behavior to score goals. But through a reinforcement learning process the robot learns using its behaviors purposively.

In [10], Oudeyer et al. made a robot show different phases of cognitive development. They placed a robot-dog in a playground that included various simple toys. In this environment, by executing some primitive behaviors randomly, the robot learned the relation between its behaviors and the events in the environment. When also provided with an external motivation the robot autonomously went through a developmental sequence, during which its complexity of activities increased at each stage.

D. Affordances as a framework for developmental robotics

J.J. Gibson [11] introduced the concept of *affordances* to refer to the action possibilities offered to the organism by its environment. For instance, a horizontal and rigid surface affords walk-ability, a small object below a certain weight affords throw-ability, for a human. The concept of affordances, with its implicit but central emphasis to the interactions between the organism and the environment, is highly relevant to *developmental/epigenetic robotics* as has already been noted [12]. Developmental robotics treats affordances as a higher level concept, which a developing cognitive agent learns by interacting with its environment.

In [1] we proposed a formalization for affordances such that it can provide a view of affordances from the perspective of the robot and lay a framework over which affordances can be utilized at different levels of robot control.

Based on this formalization, we investigated several problems in robotics. In [13] we presented the integrated view of our experimental studies, where we made a robot learn the "traversability" affordance, extend the system with an on-line learning process, and demonstrated our preliminary results in planning. In [14] we made the robot learn to use a set of primitive behaviors goal-directedly. In this current study, we extend this previous work, by formally defining two different ways of using learned affordance relations in developing behaviors, and showing how *new* behaviors can be created from the pre-coded primitive behaviors.

III. Experimental Framework

The Kurt3D robot platform: Kurt3D is a medium-sized ($45cm \times 33cm \times 47cm$), differential drive mobile robot, equipped with a 3D laser range finder[1]. The scanner is capable of taking full resolution (720×720) range image in approximately 45 seconds. Kurt3D is simulated in MACSim, a physics-based simulator, built using ODE (Open Dynamics Engine)[2], an open-source physics engine. The sensor and actuator models are calibrated against their real counterparts.

Primitive behaviors: We implemented three primitive behaviors on the robot for our experiments. These are *move-forward*, *turn-left*, and *turn-right* behaviors. The *move-forward* behavior drives the robot straight ahead that places it approximately $40cm$ away from its initial position, if the move is not obstructed by any obstacles. The *turn-left*, and *turn-right* behaviors turns the robot in place for approximately $55°$. The wheel speeds are set to either -0.25 m/s or $+0.25$ m/s for each behavior.

Interaction environment: In the learning phase each trial is performed with a single object in the environment. Objects with simple geometries such as rectangular boxes, spheres and cylinders are placed in random orientations and random locations within a proximity of $70cm$ to the robot, in the frontal area spanning $180°$. After learning is completed, developed behaviors are tested in an environment cluttered with randomly distributed objects.

Perception and representation of entities and effects: The robot perceives its environment mainly through its 3D scanner. It uses the range images from the scanner to extract a set of shape and distance related features which consists the robot's perception of the environment.

The feature set is obtained in three steps as shown in Fig. 1. The robot makes a full resolution scan of 720×720. First, the image is down-scaled to 360×360 pixels. Then, it is split into grids of size 12×12 pixels. There are 900 such grids (since $(360/12)^2 = 900$), in total. Then, for each grid, distance and shape related features are extracted. The distance related features are the distance of the closest point, distance of the furthest point, and the mean distance of all the points within a grid. The shape related features are computed

[1] URL: http://www.ais.fraunhofer.de/ARC/kurt3D/

[2] URL: http://ode.org/

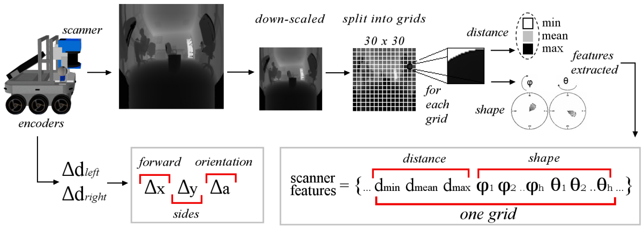


Fig. 1. Phases of perception. Distance and shape features are extracted from the scanner range image. Also three displacement values are extracted from the encoders.

from the normal vectors in the grid. A normal vector for each point in a grid is computed using the range values. Then the direction of each normal vector is recorded in two base-dimensions, φ and θ, in latitude and longitude. Two angular histograms are computed for each of these dimensions. The histograms are sliced into 18 intervals of 20° each, and the frequency values in each of these slices of the histograms are used as the shape related features. Since there are two channels of 18 values each, there are 36 shape related feature for each grid. Adding the three distance related features of a grid, there are 39 features to represent a single grid. We mentioned that there are 900 such grids. So the total number of features to describe the scene becomes $900 \times 39 = 35100$. In addition to the scanner features, values from the wheel-encoders are also recorded.

IV. Learning affordance relations

In this section, how affordance relations are learned from a number of affordance relation instances obtained during robot's interactions with the environment will be described. A more detailed explanation is provided in [14].

In our formalization, entities are defined as the perceived state of the environment before robot's behavior execution, and effects are described as the change in the perceived state. Suppose that B is the set of primitive behaviors and $b \in B$ is a behavior in this set. Let $\boldsymbol{g}_s^i$ be the scanner features of grid i in situation s, and $\boldsymbol{p}_s$ / $\boldsymbol{p'}_{s,b}$ corresponds to the entities (feature vectors) obtained before / after execution of the behavior, respectively. Then $\boldsymbol{g}_s^i$ and $\boldsymbol{p}_s$ are defined as:

$$\boldsymbol{g}_s^i = [d_{min}^i, d_{mean}^i, d_{max}^i, \varphi_1^i, \varphi_2^i, ...\varphi_{18}^i, \theta_1^i, \theta_2^i, ...\theta_{18}^i]$$

$$\boldsymbol{p}_s = [\boldsymbol{g}_s^1, \boldsymbol{g}_s^2, ...\boldsymbol{g}_s^{900}]^T$$

where $1 \leq i \leq 900$ is the grid index, and s denotes which sample situation is dealt with. Effects are represented as changes in robot's perception of the world including changes in proprioceptive sensors:

$$\boldsymbol{e}_{s,b} = [(\boldsymbol{p'}_{s,b} - \boldsymbol{p}_s)^T, \triangle x, \triangle y, \triangle a]^T$$

where $\boldsymbol{e}_{s,b}$ is the effect obtained during execution of b. $\triangle x$, $\triangle y$, and $\triangle a$ corresponds to forward and side displacements, and change in orientation of the robot, respectively.

In order to learn affordances and develop generic affordance relations, a number of (*effect*, (*entity*, *behavior*)) relation instances are acquired through interactions with the environment. In data collection step, for each behavior b, the robot makes 3000 interactions with the environment, and stores the entities ($\boldsymbol{p}_s$) and effects ($\boldsymbol{e}_{s,b}$) in a training set. For simplicity, from now on, the method will be described over one behavior and $\boldsymbol{e}_{s,b}$ will be replaced by $\boldsymbol{e}_s^b$. The training set (S_{train}) stores affordance relation instances, which are represented as nested triples of entities, effects and behaviors:

$$S_{train} = \{(\boldsymbol{e}_s^b, (\boldsymbol{p}_s, b))\}$$

where $\boldsymbol{e}_s^b$ is the effect observed when behavior b is executed over entity $\boldsymbol{p}_s$ in situation s.

Using effect instances in the training data ($\{\boldsymbol{e}_s^b\}$), effects that are similar to each other are grouped together to get a more general description of different kinds of effects that behavior can create. This is achieved by clustering the effect instances in an unsupervised way. K-means algorithm is used for this purpose where k is experimentally set to 10.

The prototype effect-id (*effect-id*$_s$) to which any effect $\boldsymbol{e}_s^b$ belongs to can then be found by:

$$\textit{effect-id}_s = \operatorname*{argmin}_{1 \leq i \leq 10}(\boldsymbol{e}_s^b - \boldsymbol{c}_i^b)$$

where $1 \leq i \leq 10$ is the cluster index, and $\boldsymbol{c}_i^b$ is the mean of i^{th} cluster and corresponds to the *effect prototype* of that cluster. An interpretation of the effect classes obtained for primitive behavior *move-forward* has been provided in [14].

After identifying a number of different *effect prototype*s, the robot learns the mapping from the *entities* to these prototypes (or *effect-id*s), for the execution of a behavior. This is achieved by training classifiers with the collected affordance *relation instances*. A separate Support Vector Machine (SVM) is trained for each behavior, using the set $\{(\boldsymbol{p}_s, \textit{effect-id}_s)\}_{(1 \leq s \leq 3000)}$, where $\boldsymbol{p}_s$ (which includes only the relevant features[3]) is given as the input, and the corresponding *effect-id* of each instance (s) as the target category. These SVMs are then used in the execution phase, to predict what kind of *effect* a behavior will generate, given a perceptual representation ($\boldsymbol{p}_{s'}$) of the current environment:

$$\textit{effect-id}_{s'}^{predicted} = svmPredict(\boldsymbol{p}_{s'}, b)$$

where s' denotes the current situation.

The effect predicted when robot encounters with a situation s' would be the prototype (mean) of the corresponding cluster that was generated for behavior b:

$$\boldsymbol{e}_{s'}^{b,predicted} = \boldsymbol{c}_{\textit{effect-id}_{s'}^{predicted}}^b \qquad (1)$$

V. Using learned affordances for behavior development

In creating new more intelligent behavior from the primitive behaviors, the primitive behaviors (that the robot has learned about, and has done its training with) can be used in

[3] Original size of *entity* vector ($\boldsymbol{p}_s$) is 35100 and most of the features in this vector are irrelevant for affordance of any behavior. Thus, a feature selection method, ReliefF is applied in order to select the 2000 features in the *entity* vector that are most relevant to the *effect*s created.

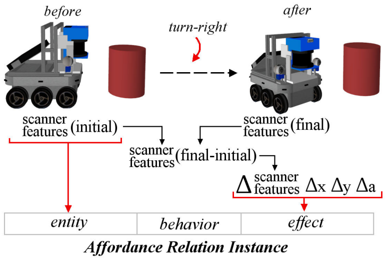

before
turn-right
after
scanner features (initial)
scanner features (final)
scanner features (final-initial)
Δ scanner features Δx Δy Δa
entity
behavior
effect
Affordance Relation Instance

Fig. 2. Representation of the entity and the effect. Distance and shape features extracted from the scanner image, taken before the execution of a primitive behavior, constitute the *entity*. The difference between the features extracted after the execution of the behavior and features extracted before the execution of the behavior constitute the representation of *effect*, together with the displacement values extracted from the encoders (see Fig. 1).

two ways as the result of development. In the first case, the primitive behaviors can be used as they are, therefore there will be one single primitive behavior active at an instant. But the cumulative effect of the execution of these will form a goal-directed behavior on a wider time-scale. In the second way of using the primitive behaviors in behavior development, the primitive behaviors can be blended, such that, at an instant it is not any of the primitive behaviors in action, but a new behavior that was not seen or demonstrated by the robot before, yet is used by it intelligently to create effects in the environment that are more in accordance with its goals than any of the primitive behaviors.

A. Developing behaviors through the sequential usage of primitive behaviors

In this first approach, the robot will use its primitive behaviors in a sequential manner to achieve goal-directed behavior. The robot uses the learned *affordance relations* to select the primitive behavior in achieving goal-directed behaviors. Given the perceptual representation of the current environment as an *entity*, the trained classifiers will predict an *effect-id* which indicates the effect class that the behavior, for which the classifier was trained, will produce in this environment. Then the robot can select the behavior which predicts the *effect prototype* that is most similar to the desired effect determined by its current goal. Therefore the selected behavior will produce the most useful effect in achieving its goal. Formally the behavior selection can be described as:

$$b_{selected} = \operatorname*{argmin}_{b \in B} \left(\boldsymbol{e}_{s'}^{b,predicted} - \boldsymbol{e}_{desired} \right)$$

where $\boldsymbol{e}_{s'}^{b,predicted}$ is the predicted effect of applying behavior b in situation s' (Eqn. 1) and $\boldsymbol{e}_{desired}$ is the desired effect.

In [14] we showed that using this strategy, our robot was able to develop different higher-level behaviors using its three primitive behaviors and the learned affordance relations. First our robot demonstrated the "traverse" behavior, using which it was able to wander around perceiving the "traversability" of the environment. Our robot was able to move over objects like spheres, or cylinders in an appropriate orientation that can be rolled away; but avoid non-rollable objects like boxes. As a second example, the robot demonstrated a classical obstacle-avoidance behavior. Here, it avoids contact with any object while wandering around. The third behavior was the "approach" behavior, where the robot approaches and drives towards the objects.

The robot demonstrated these three different behaviors ("traverse", "approach", "avoid") using the same learned affordance structures. We were able to make the robot demonstrate different behaviors through the specification of the *desired effect* for each behavior. For the *traverse* behavior, we set the "desired effect" so that the "forward displacement" features in the *effect-prototypes* has values greater than a certain threshold. This means the robot should not select the *effect-categories* corresponding to the cases where the robot got stuck to an obstacle, but executes *move_forward* when there is an empty space or a rollable object in front. For the *avoid* behavior we specified the desired effect as having a high increase in the mean distance features of the grids in the middle portion of the range image. This results in a behavior where the robot avoids contact with any object by turning away whenever something appears on its front. When the desired effect is changed to a high decrease in the mean distance, an *approach* behavior emerges. The robot moves forward towards an object on its front, and turns towards an object on its side, to obtain the desired decrease.

Note that, rather than aiming to make the robot learn a specific behavior, our work proposes a generic development scheme. This becomes obvious when one notices that the training our robot goes through is independent from the behaviors it is able to display at the end.

B. Behavior generalization through the blending of the primitive behaviors

In the previous section the robot was able to use a set of primitive behaviors such that when viewed on a wider time-scale the robot's behavior corresponded to goal-directed intelligent behaviors. But at any given moment the robot executed a single primitive behavior, and these were the same behaviors that it also used during the training interactions, and they were programmed into the robot.

In this section we will try to achieve *behavioral generalization* (or more correctly, a generalization over the motor control parameters of the behaviors), so that, after training, the robot will not be constrained with the fixed set of pre-programmed behaviors but will be able to demonstrate novel behaviors. While the robot will still have a limited set of behaviors during training, after training it will react to situations with new behaviors, that are more effective than the primitive behaviors in creating the desired effect in the environment. To be able to do that, the robot needs to make a generalization over the motor parameters that it uses for the behaviors, and relate these to the effects it can create with these parameters. Then, when it needs to create a specific effect in the environment, the robot can choose the correct set of parameters to create the effect.

The robot will use its primitive behaviors simultaneously to achieve goal-directed behavior. We achieved this generalization using a weighted sum of the motor control parameters of the primitive behaviors, where the weights are determined according to the similarity of their effects to the desired effect. Practically, this will again correspond to feeding in the current *entity* representation to the trained classifiers, of which there is one for each action. Then the predictions of each classifier, which are *effect-prototypes*, are compared with the goal representation to see how similar each *behavior*'s *effect* prediction is to the desired *effect*. The similarity values will then be used as weights for the behavioral parameters, in blending the primitive behaviors so that a new behavior emerges. The inspiration for this comes from the work on *population coding* (See Section II-B).

Method: Suppose that there are n primitive behaviors $B_1, B_2, ..., B_n$, and each behavior B_i has a set of motor parameter values $v_{i1}, v_{i2}, ..., v_{im}$ for each of the m motors $M_1, M_2, ..., M_m$. Further suppose that D is the desired effect prototype, and $p_1, p_2, ..., p_n$ are the predicted effect-category prototypes in the current environment for each of the n behaviors. Also, let's say that there is a similarity function S that takes two effect prototypes as arguments and returns a value indicating the similarity between these two prototypes. Then, in an arbitrary environment, we can find the new value v'_j to be passed to motor M_j as:

$$v'_j = \sum_{i=1}^{n} \frac{S(D, p_i)}{\sum_{k=1}^{n} S(D, p_k)} * v_{ij} \qquad (2)$$

That is, the resulting motor parameter value is the sum of each behavior's contribution for that parameter, and this contribution is proportional to the similarity of the predicted effect for that behavior to the desired effect. Note that, other than the learned affordance relations, we also need to define a similarity function that would indicate how similar a predicted effect is to the desired effect.

VI. Experimental results

In this section we will present the results of applying the two strategies of using the learned affordance relations for the "approach" behavior. These two strategies were analyzed in systematic experiments, where the object to be approached was placed in different positions in front of the robot. The object was placed at angles $\{-60°, -45°, -30°, -15°, 0°, 15°, 30°, 45°, 60°\}$ in front of the robot, and the distances of the objects changed between $20cm$ and $70cm$. The robot executed the behavior (or the blending of the behaviors) it selected for each object in front of it. After the execution of the behavior, the relative angle of the object with respect to the robot's heading direction was recorded as the error for that case. The results of the experiments conducted at 50 different distances for each angle can be seen in Fig. 3. Each boxplot in the figure shows the distribution of the errors (in radians) the robot made at that angle for different distances of the object. The two boxplots at each angle corresponds to the two different methods: behavior generalization (on the left) and using only primitive behaviors (on the right).

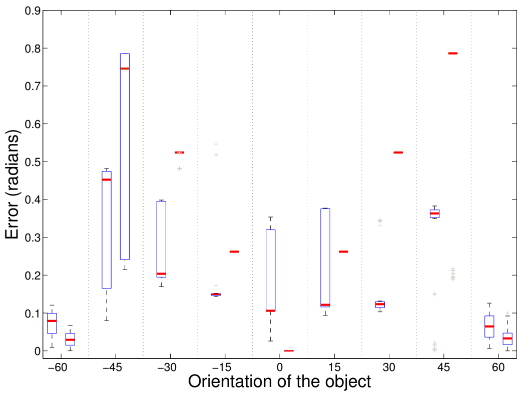
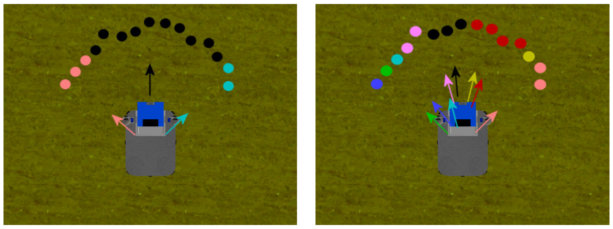


Fig. 3. Comparison of the two methods of *behavioral generalization* and *using only the primitive behaviors* for the "approach" behavior. Each boxplot in the figure shows the distribution of the errors (in radians) the robot made at that angle for different distances of the object. The red line refers to the median. The box is bounded by lower and upper quartile values, and the whiskers show the extends of the data. At each angle, the boxplot on the left and the boxplot on the right refers to the errors made by behavioral generalization, and using only primitive behaviors, respectively.

When the object is placed 60 degrees to the left/right ($-60°$/$60°$), or directly ahead ($0°$) of the robot, and when the robot approaches to the object using only its primitive behaviors, the error is very close to zero. This is an expected result, since these three angles are exactly the ones that the three primitive behaviors turns/drives the robot to. The fact that the errors are very close to zero at that angles also proves that the robot is really able to choose the correct behaviors for the approach behavior: *turn_left* when the object is on the left, *turn_right* when the object is on the right, and *move_forward* when the object is ahead. The behavioral generalization method gave relatively high error rates at these angles, because using this strategy the correct behavior's purity is tempered by some contribution from the other behaviors. When using the behavioral generalization method, the average errors made at the angles in between the extremes $(-45°, -30°, -15°, 15°, 30°, 45°)$ are smaller than the cases where the only the primitive behaviors are used. This shows that, using the behavioral generalization method, the robot is able to turn to the angles in between, that it can not approach using only the primitive behaviors.

This can be seen more clearly in Fig. 4. Here, it can be seen that, when compared with using only the primitive behaviors, the behavioral generalization approach spans the same angular range in turning towards the object, but it does so in a more finer manner, spanning whole of the angular range. The trade-off is some lose of precision in the directions of the original primitive behaviors.

The behavior development scheme and the trained classifiers were also transferred to the real robot and tested. We placed real world objects in front of the robot and tested to see if it was able to approach to the objects. We also compared the results for the behavioral generalization method, with using only primitive behaviors. In Fig. 5 a box shaped object is placed slightly to the left of the



(a) (b)

Fig. 4. Robot's reaction to different situations using the two different strategies of using only primitive behaviors, and using behavioral generalization. The arrows show the robot's position and heading direction after executing the behavior. The circles denote the object's position in each different case. If a circle and an arrow are of the same color, this means that when the object is in the location indicated by the circle, the robot's heading direction and position after executing the chosen behavior is indicated by the arrow of the same color. In (a) the robot uses only the primitive behaviors in approaching the object. Therefore, in the figure, there are only three arrows, representing the robot's position and heading direction after executing each of these three behaviors. It can be seen that the robot is able to approach the object and select the correct primitive behavior. But one can also notice that these primitive behaviors are very crude in turning towards the object. In (b) the robot uses the behavioral generalization strategy in turning towards the objects. Here again the robot is successful in turning towards the object, but this time it makes more detailed movements towards the objects showing an improvement over the case of using only the primitive behaviors.

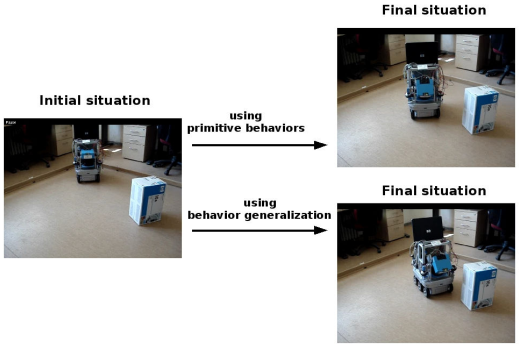


Fig. 5. Real robot's reaction to a situation where the object is placed slightly to the left of the robot.

robot, and the final situations after the execution of the behaviors are shown. It can be seen that the behavioral generalization approach is more successful than the case of using only primitive behaviors in which the robot executes the MOVE_FORWARD behavior.

VII. Conclusion

In this paper we proposed a behavior development scheme for a robot. The concept of "affordances" [11] provided the basis of our proposed development scheme. We used a formalization of affordances [1] to make the robot learn the dynamics of its interactions with its environment. Using this framework, in [14] we made the robot learn to use a set of primitive behaviors goal-directedly. In this current study, we extend this previous work, first, by formally defining two different ways of using learned affordance relations in developing behaviors. Second, we show how *new* behaviors can be created from the pre-coded primitive behaviors.

We have seen that both in the studies of *developmental psychology* and in the studies of *motor control and learning* in neuroscience, the idea of starting from pre-coded primitive behaviors, and through development, achieving more complex behaviors is accepted as a possibility. If we combine the approach of developmental psychologists (which says that a baby starts from innate primitive reflexes and enriches them through experience until they become voluntary action) with the approach of neuroscience (which says that complex patterns of motor behavior can be explained using combination of simple pre-coded behaviors), then we believe that this presents a very good research potential for robotic behavior development. In this kind of research, one should investigate how robots equipped with simple pre-coded(innate) behaviors can develop to achieve more complex behaviors through the usage of these simple behaviors. This actually constitutes the grounds where this work aims to make its contribution.

VIII. Acknowledgments

This work was partially funded by TUBITAK and the European Commission under the MACS project (FP6-004381). URL: `http://macs-eu.org`

References

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

[2] J. Piaget, *The Essential Piaget*. New York: Basic Books, 1977. edited by H. E. Gruber and J. J. Voneche.

[3] R. Gelman, "Cognitive development," in *The MIT Encyclopedia of Cognitive Sciences* (R. Wilson and F. Keil, eds.), Cambridge, MA: MIT Press/Bradford Press, 1998.

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

[5] E. Bizzi, "Motor control," in *The MIT Encyclopedia of Cognitive Sciences* (R. Wilson and F. Keil, eds.), Cambridge, MA: MIT Press/Bradford Press, 1998.

[6] F. A. Mussa-Ivaldi, S. F. Giszter, and E. Bizzi, "Linear combinations of primitives in vertebrate motor control.," *Proceedings of the National Academy of Sciences, USA*, vol. 91, pp. 7534–7538, August 1994.

[7] A. P. Georgopoulos, A. B. Schwartz, and R. E. Kettner, "Neuronal population coding of movement direction.," *Science*, vol. 233, pp. 1416–1419, September 1986.

[8] M. Likhachev, M. Kaess, and R. Arkin, "Learning behavioral parameterization using spatiotemporal case-based reasoning," in *Proc. of ICRA'02*, 2002.

[9] M. Asada, S. Noda, S. Tawaratsumida, and K. Hosoda, "Purposive behavior acquisition for a real robot by vision-based reinforcement learning," *Machine Learning*, vol. V23, pp. 279–303, May 1996.

[10] P.-Y. Oudeyer, F. Kaplan, V. V. Hafner, and A. Whyte, "The playground experiment: Task-independent development of a curious robot," in *Proceedings of the AAAI Spring Symposium on Developmental Robotics, 2005* (D. Bank and L. Meeden, eds.), (Stanford, California), pp. 42–47, 2005.

[11] J. Gibson, *The Ecological Approach to Visual Perception*. Lawrence Erlbaum Associates, 1986. Originally published in 1979.

[12] M. Lungarella, G. Metta, R. Pfeifer, and G. Sandini, "Developmental robotics: a survey," *Connection Science*, vol. 15, no. 4, pp. 151–190, 2003.

[13] M. Çakmak, M. Doğar, E. Uğur, and E. Şahin, "Affordances as a framework for robot control," in *Proceedings of The 7th International Conference on Epigenetic Robotics, EpiRob'07*, 2007.

[14] M. R. Doğar, M. Çakmak, E. Uğur, and E. Şahin, "From primitive behaviors to goal-directed behavior using affordances," in *Proc. of IROS'07*, 2007.