in "Advances in CognitiveSystems", IET Publishers, 2009.

A Computational Model of Object Affordances

Luis Montesano Manuel Lopes Francisco Melo Alexandre Bernardino José Santos-Victor

Abstract

The concept of object affordances describes the possible ways whereby an agent (either biological or artificial) can act upon an object. By observing the effects of actions on objects with certain properties, the agent can acquire an internal representation of the world way of functioning with respect to its own motor and perceptual skills. Thus, affordances encode knowledge about the relationships between action and effects, lying at the core of high level cognitive skills such as planning, recognition, prediction and imitation. Humans learn and exploit object affordances through their entire lifespan, either by autonomous exploration of the world or by social interaction.

Building on a biological motivation and aiming at the development of adaptive robotic systems, we propose a computational model capable of encoding object affordances during exploratory learning trials. We represent this knowledge as a Bayesian network and rely on statistical learning and inference methods to generate and explore the network, efficiently dealing with uncertainty, redundancy, and irrelevant information. The affordance model serves as base for an imitation learning framework that exploits the recognition and planning capabilities to learn new tasks from demonstrations. We show the application of our model in a real world task in which a humanoid robot interacts with objects and uses the acquired knowledge and learns from demonstrations. Results illustrate the success of our approach in learning object affordances and generating complex cognitive behavior.

Keywords: Affordances, Humanoid robots, Cognitive robotics, Bio-inspired learning.

L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor are with the Instituto de Sistemas e Robótica, Instituto Superior Técnico, Lisboa, Portugal. {lmontesano,macl,alex,jasv}@isr.ist.utl.pt. F. Melo is with the Computer Science Department, Carnegie Mellon University, USA.

This work was partially supported by the Information and Communications Technologies Institute, the Portuguese Fundação para a Ciência e a Tecnologia, under the Carnegie Mellon-Portugal Program, the Programa Operacional Sociedade de Conhecimento (POS_C) in the frame of QCA III and the PTDC/EEA-ACR/70174/2006 project, and in part by the EU Projects (IST-004370) RobotCub and (EU-FP6-NEST-5010) Contact.

Parts of this chapter were previously published in the IEEE Transactions on Robotics [1] and IEEE/RSF International Conference on Intelligent Robotic Systems [2].

I. Introduction

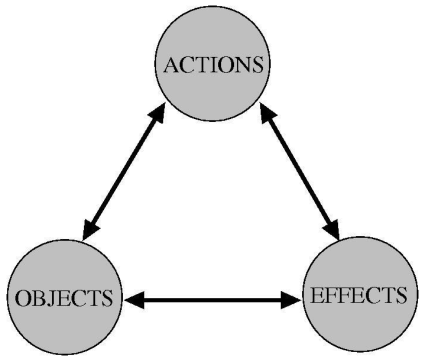
Humans routinely solve tasks that require manipulation and interaction with different types of objects. From simple everyday actions like pulling up a chair and sitting down, to complex skilled operations like driving or repairing a car, humans purposely exploit the objects to achieve their goals. Even when an object is not designed for a particular application, humans are creative enough to give it new usages. For example, it is not uncommon to use a chair to hang a jacket, even though a chair is made for sitting. In fact, what matters in obtaining a certain desired effect are the properties of the objects more than the objects themselves.

Objects have different usages depending on the agents' motor and perceptual skills. For example, a chair is only "sitable" by an individual of a certain height; a tree is only "climbable" by animals with specific capabilities. James J. Gibson denominated these "agent-dependent object usages" as *affordances* [3]. In this seminal work, object affordances were defined as all "action possibilities" with reference to the actor's motor and sensing capabilities. In a nutshell, affordances represent the link between an agent and the environment and depend on the agent's specific motor and perceptual skills.

Our dependence on objects to achieve even the simplest goals in daily tasks emphasizes the importance of the concept of affordances in human cognition. The knowledge of object affordances is exploited in most of our decisions: to choose the most appropriate way of acting upon an object for a certain purpose; to search and select objects that best suit the execution of a task; to predict the effects of actions on objects; to recognize ambiguous objects or actions; etc. Affordances are at the core of high-level cognitive skills such as planning, recognition, prediction and imitation.

A key question that this chapter addresses, from a computational point of view, is how affordance knowledge can be acquired during an agent's lifespan. The importance of experience in mastering object related skills suggests that affordance knowledge is acquired as the agent autonomously interacts with objects. In fact, infants interact with objects since early childhood and gradually learn how to use them in order to solve complex tasks. For instance, in the task of inserting shaped blocks into apertures [4], 14-months-old children demonstrate manipulation skills that are more exploratory than functional; 18-months-old children seem to understand the rules for fitting the blocks but cannot implement them. It is only at 22 months that children are able to have some success on the task and at 26 months that they solve the problem systematically. The capability to manipulate and exploit object properties is the result of a sophisticated ontogenetic development. Skills are acquired incrementally according to a genetic program conditioned by the surrounding environment, *i.e.,* through the interaction with the world and other people.

In this chapter, we discuss object affordances within the context of the long-term goal of building (humanoid) robots capable of acting in a complex world and interacting with humans and objects in a flexible way. In particular, we address the following key questions:



Input	Output	Function
(O, A)	E	Predict effect
(O, E)	A	Action recognition & planning
(A, E)	O	Object recognition & selection

Fig. 1: Affordances as relations between (A)ctions, (O)bjects and (E)ffects, that can be used for different purposes: predict the outcome of an action, plan actions to achieve a goal, or recognize objects/actions.

- What knowledge representations and cognitive architecture should such a system require to be able to act in an complex and generally unpredictable environment?
- How can the system acquire task and domain-specific knowledge to be used in novel situations?

From a robotics perspective, affordances are an extremely powerful concept that captures the essential world and object properties in terms of the actions the robot is able to perform. They can be used to predict the effects of an action, to plan actions to achieve a specific goal or to select the best object to produce a given effect if acted upon in a certain way (see Fig. 1).

Extending the concept a bit further, affordances also play an important role when interacting with other agents. An artificial system can gain a tremendous amount of information by observing another human or robotic agent, its actions and corresponding effects [5]. Affordance knowledge allows for action recognition in terms of the robot's motor capabilities and can be used, for example, in imitation [2]. Learning by imitation is one of the motivations behind our approach: we are interested in using generic affordances to learn a demonstrated task. We show that affordance knowledge can successfully be used to obtain imitation-like behaviors, *e.g.,* allowing a robot to learn from a teacher a sequence of actions leading to a particular outcome.

Finally, it is important to emphasize that having a robot learn, from scratch, a model of its interaction with the environment, is a tremendous task. It involves complex perceptual and motor skills, such as identifying objects and acting upon them. One possible way to deal with such complexity is by considering a bottom-up developmental perspective: basic sensory-motor skills are learned in initial stages upon which more complex cognitive capabilities can then be built. From the start, the robot should be able to individuate objects in the environment and execute directed exploratory motor actions upon them. In the same way, it is only after having learned a reasonable model of the world that imitation mechanisms will be operative and enable the robot to learn from other agents.

A. Related Work

Gibson's affordances [3] represent what the elements present in the environment afford to the agent. This very general concept was originally applied to entities such as surfaces (ground, air, water) or their frontiers. In psychology, there has been a lot of discussion to establish a definition or model of affordances (see [6] for a brief review). Several authors have shown the presence of some type of affordance knowledge by comparing percepts among different people [7], measuring response times to tasks elicited by specific object orientations [8] or perceiving heaviness [9]. Unfortunately, there is little evidence on how humans learn affordances.

From the robotics standpoint, affordances have been mainly used to relate actions to objects. Several works use affordances as prior information. A computational cognitive model for the learning of grasping actions in infants was proposed in [10]. The affordance layer in this model provides information that helps the agent to perform the action. Affordances have also been used as prior distributions for action recognition in a Bayesian framework [11] or to perform selective attention in obstacle-avoidance tasks [12].

Several works have investigated the problem of learning affordances and their subsequent application to different tasks. In [13], a robot learned the direction of motion of different objects when poked and used this information at a later stage to recognize actions performed by others. The robot used the learned maps to push objects so as to reproduce the observed motion. A similar approach was proposed in [14], where the imitation is also driven by observed effects. However, this work focuses on the interaction aspects and do not consider a general model for learning and using affordances. The biologically-inspired behavior selection mechanism of [15] uses clustering and self-organizing feature maps to relate object invariants to the success or failure of an action.

All previous approaches learn specific types of affordances using the relevant information extracted from sensor inputs. A more complete solution has been recently proposed in [16], where the learning procedure also selects the appropriate features from a set of visual SIFT descriptors. The work in [17] focuses on the importance of sequences of actions and invariant perceptions to discover affordances in a behavioral framework. Finally, based on the formalism of [18], a goal-oriented affordance-based control for mobile robots has been presented in [19]. Previously learned behaviors such as *traverse* or *approach* are combined to achieve goal-oriented navigation.

Regarding imitation, one must consider two fundamental problems: description of the observed motion in terms of the imitator's own motor capabilities (body correspondence) and selection of the goal of imitation (imitation metric) . The former has been addressed in different ways in the literature. Possible approaches include the hand-coding of the correspondence between the teacher and the imitator actions [20] or describing world-state transitions at the trajectory level [21].

Recent research in neuroscience has triggered some alternative ways of addressing the correspondence problem. In particular, area F5 in the primate's premotor cortex is dominated by action coding neurons, but several of them also respond to visual stimuli (*e.g.,* canonical neurons [22] and mirror neurons [23], [24]). Canonical neurons show object-related visual responses that are, in the majority of cases, selective for objects of certain size, shape and orientation. On the other hand, mirror neurons respond to the observation of actions upon objects. They respond to actions executed by the observer but also to similar actions executed by another individual. This suggests that the same neuronal circuitry may be involved in both the recognition and generation of object-directed actions.

The discovery of mirror neurons triggered a large interest in the study of action recognition and imitation behaviors. Mirror neurons constitute an observation/execution matching system that maps observed actions to the observer's internal motor representations. This facilitates the recognition of actions because, unlike visual data, internal motor representations are invariant to viewpoint and other visual distortions [11]. Mapping actions to internal representations allows the imitation of actions even when individuals are not morphologically identical. The imitator chooses from its own motor repertoire the action that best matches the observations. It may choose to match only the effect of the action (emulation) [14], [13] or also part of the action itself [25].

As described above, imitation can be interpreted at different levels from trajectory mimicking to effect emulation. There is no single solution for this problem and even humans change imitation strategies taking into account contextual information [26], [27], [28]. In artificial systems several authors have proposed different imitation metrics that result in different behaviors [25], [29], [20] that not all can be considered real imitation.

B. Our Approach

Learning affordances from scratch without assuming any previous knowledge can be overwhelming. It involves learning relations between motor and perceptual skills, resulting in an extremely high-dimensional search problem. On the other hand, affordances can be defined more appropriately once the robot has already learned a suitable set of elementary actions to explore the world.

We adopt a developmental approach [30], [31], in which the robot acquires skills of increasing difficulty on top of previous ones. Similarly to newborn children, the robot "starts" with a minimal subset of core (phylogenetic) capabilities [32] to bootstrap learning mechanisms that progressively lead to the acquisition of new skills by means of self-experimentation, interaction with the environment and with other agents.

We follow the developmental roadmap proposed in [33] and extend it to include the learning and usage of affordances in the world interaction phase. This framework considers three main stages in a possible developmental architecture for humanoid robots: (i) sensory-motor coordination; (ii) world interaction; and (iii) imitation (see Table I). In the sensory-motor coordination stage, the robot learns how to use its motor degrees of freedom and the

TABLE I: Learning stages of the developmental approach.

Sensory-Motor Coordination	1: Learn basic motor skills 2: Develop visual perception of objects
World Interaction	3: Perception of effects and categorization 4: Improve motor skills 5: Learn object affordances 6: Prediction and planning skills
Imitation	7: Perform imitation games Task inference from observation

coupling between motor actions and perception. In the world-interaction phase, the robot learns by exploring the effects of its own actions upon elements of the environment. In the imitation phase, the robot learns by observing and imitating other agents.

Affordances are central in the world-interaction stage. In this stage, the robot has already developed a set of perceptual and motor skills required to interact with the world. In this chapter, we propose a general model to represent knowledge about affordances, *i.e.,* relationships between the agent's actions, object properties and effects observed on these objects. The model consists of a Bayesian network (BN) [34], a probabilistic graphical model that represents dependencies between variables. In other words, a BN is a directed acyclic graph whose nodes represent (random) variables and whose connections express the correlations between them. The BN thus encodes the relation between different types of information. From the probabilistic model, the robot can use the information available from its sensory inputs to make different types of predictions about the world, infer situations from incomplete information and plan for actions depending on its goals.

In a second part of the chapter we use the world knowledge acquired in the form of affordances to be able to imitate others. Affordances were learned in a task-independent way and so they need to be written in a way that allows sequential decision making. Equipped with this knowledge the robot is able to infer the goal of an observed demonstration using Bayesian Inverse Reinforcement Learning [35]. Affordances provide another invaluable information that is the recognition of observed actions. The robot is thus able to extract information from the demonstration by recognition the observed motions in terms of its own motor repertoire.

We used the humanoid robot BALTAZAR (see Fig. 6 in Section IV) to validate the approach. We conducted several experiments to illustrate the capability of the system to discover affordances associated with manipulation actions (*e.g.,* grasp, tap and touch) when applied to objects with different properties (color, size, shape). The effects of these actions consist of changes perceived in the sensor measurements, *e.g.,* persistent tactile activation for grasp/touch actions and object motion for tap actions.

Summarizing, this chapter presents a model for learning and using affordances and its application to robot imitation. The main characteristics of the proposed model are: (i) it captures the relations between actions, object features and effects; (ii) it is learned through observation and interaction with the world; (iii) it identifies the object features that matter for each affordance; (iv) it provides a seamless framework for the learning and using of affordances; and (v) it allows social interaction by learning from others.

II. Affordance Modeling and Learning

In this section, we address the problem of learning object affordances. According to Table I, we assume the robot has already acquired a set of skills that allows it to reason in a more abstract level than joint positions or raw perceptions. More specifically, the robot has available a parametrized set of actions that allow it to interact with the world and is able to detect and extract categorical information from the objects around it (see Section IV for further details).

We pose the affordance learning problem at this level of abstraction, where the main entities are actions, objects properties and effects. A discrete random variable A taking values in the set $\mathcal{A} = \{a_i, i = 1, \ldots, n_a\}$ models the activation of the different motor actions. Each action a_i is parametrized by a corresponding set of parameters λ_i. For example, when approaching an object to perform a grasp action, the height of the hand with respect to the object or the closing angles of the hand are free parameters.[1] It is important to note that, from a sensory-motor point of view, different values for these free parameters result in the same action. Hence, at this stage of development, the robot cannot distinguish between them, since the differences will only be evident when interacting with those objects.

The object properties and effects are also modeled using discrete random variables as detected by the robot. We denote by $F_r = \{F_r(1), ..., F_r(n_r)\}$ and $F_o = \{F_o(1), ..., F_o(n_o)\}$ the sets of random variables corresponding to the descriptors (features) extracted by the perceptual modules and representing the agent itself and object o, respectively. Finally, we let $E = \{E(1), ..., E(n_e)\}$ denote the set of random variables corresponding to the possible effects detected by the robot after executing an action. The difference between object features and effects is that the former can be acquired by simple observation, whereas the latter require interaction with the objects. Thus, clustering the effects correspond to the first step of the world interaction stage in Table I and precedes the actual learning of the affordances.

We use a Bayesian network (BN) to encode the dependencies between object features, actions exerted upon such objects, and effects of those actions (see Fig. 2). Such a representation has several advantages. It allows us to take into account the inherent uncertainty in the world, encodes some notion of causality and provides a unified framework for both learning and using affordances. In the continuation, we briefly survey the fundamental concepts concerning representation, inference and learning using BNs and show how to apply them to our affordance problem.

[1]We refer to Section IV for further details on the actual action implementation.

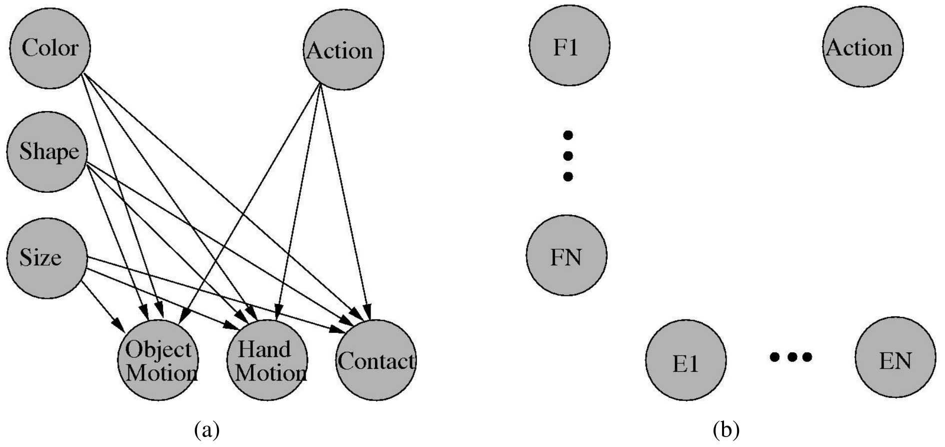


Fig. 2: BN model representing affordances. (a) An example of the proposed model using color, shape and size information for the object features; and motion and contact information as effects. (b) Generic model where the nodes represent the action, A, the object features available to the robot, $F(1)...F(n)$, and the effects obtained through the actions, $E(1)...E(m)$.

A BN is a probabilistic graphical model that represents dependencies between random variables as a directed acyclic graph. Each node in the network represents a random variables Y_i, $i = 1, \dots, n$ and the (lack of) arcs between two nodes Y_i and Y_j represents conditional independence of the corresponding variables. BNs are able to represent causal models since an arc $Y_i \rightarrow Y_j$ can be interpreted as Y_i *causes* Y_j [36]. The conditional probability distribution (CPD) of each variable Y_i in the network, denoted as $\mathbb{P}\left[Y_i \mid Y_{Pa(Y_i)}, \theta_i\right]$, depends on the parent nodes of Y_i, denoted collectively as $Y_{Pa(Y_i)}$, and on a set of parameters θ_i. The joint distribution of the BN thus decomposes in the following way:

$$\mathbb{P}\left[Y_1, ..., Y_n \mid \boldsymbol{\theta}\right] = \prod_{i=1}^{n} \mathbb{P}\left[Y_i \mid Y_{Pa(Y_i)}, \theta_i\right] \tag{1}$$

where $\boldsymbol{\theta}$ represents all the parameters in the different CPDs. If the conditional distributions and the priors on the parameters are conjugate, the conditional probability distributions and marginal likelihood can be computed in closed form, resulting in efficient learning and inference algorithms.

The set of nodes in the network, Y, comprises the discrete variable A and those in F_r, F_o and E, *i.e.,*

$$Y = \{A, F_r(1), \dots, F_r(n_r), F_o(1), \dots, F_o(n_o), E(1), \dots, E(n_e)\}.$$

Our ultimate goal is to uncover the relations between the random variables in Y, representing actions, features and effects (see Fig. 2). To this purpose, the robot performs an action on an object and observes the resulting effects. By repeating this procedure several times, the robot acquires a set of N samples of the variables in Y, $D = \{y^{1:N}\}$.[2]

Let us assume for the moment that the dependencies between the variables in Y are known, *i.e.,* the structure of the BN representing the affordances is known. Given the discrete representation of actions, features and effects,

[2]We represent a random variable by a capital letter Y and its realizations by y.

we can use a multinomial distribution and corresponding conjugate, the Dirichlet distribution, to model the CPDs $\mathbb{P}\left[Y_i \mid Y_{Pa(Y_i)}, \theta_i\right]$ and the corresponding parameter priors $\mathbb{P}\left[\theta_i\right]$. Let $\mathcal{Y}_i$ denote the range of values of the random variable Y_i and $\mathcal{Y}_{Pa(Y_i)}$ the range of values of the parents of Y_i. Assuming independence between the samples in D, and as seen in [37], the marginal likelihood for the random variable Y_i and its parents given D is:

$$\begin{aligned}\mathbb{P}\left[y_i^{1:N} \mid y_{Pa(y_i)}^{1:N}\right] &= \int \prod_{n=1}^{N} \mathbb{P}\left[y_i^n \mid y_{Pa(y_i)}^n, \theta_i\right] \mathbb{P}\left[\theta_i\right] d\theta_i \\ &= \prod_{j=1}^{|\mathcal{Y}_i|} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij}+N_{ij})} \prod_{k=1}^{|\mathcal{Y}_{Pa(Y_i)}|} \frac{\Gamma(\alpha_{ijk}+N_{ijk})}{\Gamma(\alpha_{ijk})}\end{aligned}$$

where N_{ijk} counts the number of samples in which $Y_i = j$ and $Y_{Pa(Y_i)} = k$, $N_{ij} = \sum_k N_{ijk}$ and Γ represents the gamma function. The pseudo-counts α_{ijk} denote the Dirichlet hyper-parameters of the prior distribution of θ_i and $\alpha_{ij} = \sum_k \alpha_{ijk}$. The marginal likelihood of the data is simply the product of the marginal likelihood of each node,

$$\mathbb{P}\left[D \mid G\right] = \mathbb{P}\left[Y^{1:N} \mid G\right] = \prod_i \mathbb{P}\left[y_i^{1:N} \mid y_{Pa(y_i)}^{1:N}\right] \tag{2}$$

where we have made explicit the dependency on the graph structure, G.

A. Learning the Structure of the Network

We are now interested in learning the structure of the network, G, which is actually an instance of a model selection problem. In a Bayesian framework, this can be formalized as estimating the distribution over all possible network structures $G \in \mathcal{G}$ given the data. Using the Bayes rule, we can express this distribution as the product of the marginal likelihood and the prior over graph structures,

$$\mathbb{P}\left[G \mid D\right] = \eta \mathbb{P}\left[D \mid G\right] \mathbb{P}\left[G\right] \tag{3}$$

where $\eta = 1/\mathbb{P}\left[D\right]$ is a normalization constant. The term $\mathbb{P}\left[G\right]$ (the prior) allows to incorporate prior knowledge on possible structures. Unfortunately, the number of possible BN structures is super-exponential in the number of nodes [38]. Thus, it is infeasible to explore all the possible graph structures and one has to rely on some form of approximation of the full distribution. Markov Chain Monte Carlo (MCMC) methods have been proposed to approximate the distribution $\mathbb{P}\left[G \mid D\right]$ [39]. In our case, this can be important during the first iterations of the learning process, as it allows the robot to keep a set of alternative hypotheses on the possible affordance model.

As the robot performs the actions itself, it is usually able to obtain information on all the variables Y_i in the BN. The model can also be applied to learning by observation, that is, by observing other people performing the actions. However, in this situation there may be some missing information. For example, the action is not directly available and has to be inferred from visual measurements. In this case, the learning task is much harder and several algorithms have been proposed such as augmented MCMC or structural EM [40].

Finally, it is important to consider causality. The previous learning techniques are able to distinguish among equivalence classes of graph structures.[3] An equivalence classes contain different causal interpretations between the nodes in the network. So as to be able to infer the correct causal dependency, it is necessary to use *interventional data*, where some of the variables are held fixed to a specific value to disambiguate between graph structures in the same equivalence class.

In the case of a robot interacting with its environment, there are several variables that are actively chosen by the robot: the action and the object. These variables are actually interventional, since they are set by the robot to their specific values at each experience. Interventional data is currently an important research topic within BN learning algorithms [41]. Under the assumption of a perfect intervention of node Y_i, the value of Y_i is set to the desired value, y_i^*, and its CPD is just an indicator function with all the probability mass assigned to this value, *i.e.*, $\mathbb{P}\left[Y_i \mid Y_{Pa(Y_i)}, \theta_i\right] = \mathbf{I}(Y_i = y_i^*)$. As a result, the variable Y_i is effectively cut off from its parents $Y_{Pa(Y_i)}$.

B. Parameter Learning and Inference

Once the structure of the network has been determined, the parameters θ_i of each node are estimated using a Bayesian approach [42]. The estimated parameters can still be updated on-line, allowing the incorporation of the information provided by new trials.

Since the structure of the BN encodes the relations between actions, object features and effects, we can now compute the distribution of a (group of) variable(s) given the values of others. The most common way to do this is to convert the BN into a tree and then apply the junction tree algorithm [43] to compute the distribution of interests. It is important to note that it is not necessary to know the values of all the variables to perform inference.

Based on these probabilistic queries, we are now able to use the affordance knowledge to answer the questions outlined in Fig. 1 simply by computing the appropriate distributions. For instance, predicting the effects of an observed action a_i given the observed object features f_j can easily be performed from the distribution $\mathbb{P}\left[E \mid A = a_i, F = f_j\right]$. The query can combine features, actions and effects both as observed information and as the desired output.

III. Imitation Learning

After interacting with the objects, the robot has acquired important information to support the social stage of its development. The robot can learn how to perform tasks by observing others and imitate their goals and actions. A general model for imitation learning is presented in the following. We adopt the formalist of [25] to learn complex task descriptions from human demonstrations. Afterwards, and based on this formalism, we explain how affordances knowledge can be used to provide the required inputs for imitation learning.

[3]Two directed acyclic graphs G and G' are equivalent if for every BN $B = (G, \Theta)$ there exist another network $B' = (G', \Theta')$ such that both define the same probability distribution.

A. *Formalism*

To imitate complex tasks, the robot must first understand the tasks executed by others. The robot will observe an action/sequence of actions by a human demonstrator and then choose its own actions accordingly. At each time instant, the robot must choose an action from its action repertoire $\mathcal{A}$, depending on the perceived state of the environment. We represent the state of the environment at time t by X_t and let $\mathcal{X}$ be the finite set of possible environment states.

At this level of abstraction, action selection can be seen as a *decision process*. This state of the world evolves according to some transition probabilities

$$\mathbb{P}\left[X_{t+1} = z \mid X_t = x, A_t = a\right] = \mathsf{P}_a(x, z), \tag{4}$$

where A_t denotes the robot's action at time t. The action-dependent transition matrix P thus describes the dynamic behavior of the process $\{X_t\}$.

At this stage we assume that the robot is able to recognize the actions performed during the demonstration. Latter, we will show how the affordances knowledge can be used to achieve this goal. Baring this assumption in mind, we consider that the demonstration consists of a sequence $\mathcal{H}$ of state-action pairs:

$$\mathcal{H} = \{(x_1, a_1), (x_2, a_2), \ldots, (x_n, a_n)\}.$$

Each pair (x_i, a_i) exemplifies to the robot the expected action (a_i) in each of the states visited during the demonstration (x_i). From this demonstration, the robot is then expected to perceive what the demonstrated task is and learn how to perform it optimally, possibly relying on some experimentation of its own. A *policy* is a map $\pi : \mathcal{X} \longrightarrow \mathcal{A}$, a decision-rule that determines the action of the robot as a function of the state of the environment. The robot must then *infer the task* from the demonstration and *learn the corresponding optimal policy*.

In our formalism, the task can be defined using a function $r : \mathcal{X} \longrightarrow \mathbb{R}$ describing the "desirability" of each particular state $x \in \mathcal{X}$. This function r works as a *reward* for the robot and, once r is known, the robot should choose its actions to maximize the total reward collected during its life-span, represented as the functional

$$J(x, \{A_t\}) = \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t r(X_t) \mid X_0 = x\right],$$

where γ is a discount factor between 0 and 1 that assigns greater importance to those rewards received in the immediate future than to those in the distant future. We remark that, once r is known, the problem falls back to the standard formulation of dynamic programming [44].

The relation between the function r describing the task and the optimal behavior rule can be evidenced by means of a function V_r, defined by the following recursive relation

$$V_r(x) = \max_{a \in \mathcal{A}} \left[r(x) + \gamma \sum_{z \in \mathcal{X}} \mathsf{P}_a(x, z) V_r(z) \right] \tag{5}$$

The value $V_r(x)$ represents the expected (discounted) reward accumulated along a path of the process $\{X_t\}$ starting at state x when the optimal behavior rule is followed. The optimal policy associated with the reward function r is thus given by

$$\pi_r(x) = \arg\max_{a \in \mathcal{A}} \left[r(x) + \gamma \sum_{z \in \mathcal{X}} \mathsf{P}_a(x, z) V_r(z) \right]$$

The computation of π_r (or, equivalently, V_r) given P and r is a standard problem and can be solved using any of several standard methods available in the literature [44].

B. Methodology

In the formalism just described, the fundamental imitation problem lies in the estimation of the function r from the observed demonstration $\mathcal{H}$. Notice that this is closely related to the problem of *inverse reinforcement learning* as described in [45]. We adopt the method described in [25], which is a basic variation of the *Bayesian inverse reinforcement learning* (BIRL) algorithm in [35].

For a given function r, we define the *likelihood of a pair* $(x, a) \in \mathcal{X} \times \mathcal{A}$ as

$$L_r(x, a) = \mathbb{P}\left[(x, a) \mid r\right] = \frac{e^{\eta Q_r(x,a)}}{\sum_{b \in \mathcal{A}} e^{\eta Q_r(x,b)}},$$

where $Q_r(x, a)$ is defined as

$$Q_r(x, a) = r(x) + \gamma \sum_{z \in \mathcal{X}} \mathsf{P}_a(x, z) V_r(z)$$

and V_r is as in (5). The parameter η is a user-defined *confidence parameter* that we describe further ahead. Notice that $L_r(x, a)$ is simply a *softmax* distribution over the possible actions, and translates the plausibility of the choice of action a in state x when the underlying task is described by r. Given a demonstration sequence

$$\mathcal{H} = \{(x_1, a_1), (x_2, a_2), \ldots, (x_n, a_n)\}.$$

the corresponding likelihood is

$$L_r(\mathcal{H}) = \prod_{i=1}^{n} L_r(x_i, a_i).$$

We use MCMC to estimate the posterior distribution over the space of possible r-functions (usually a compact subset of $\mathbb{R}^p, p > 0$) given the demonstration [35]. We then choose the r-function corresponding to the maximum of this distribution. Since we consider a uniform prior for the distribution over r-functions, the selected reward is the one whose corresponding optimal policy “best matches” the demonstration. The confidence parameter η determines the “trustworthiness” of the demonstration: it is a user-defined parameter that indicates how “close” the demonstrated policy is to the optimal policy [35].

Some remarks are in order. First of all, to determine the likelihood of the demonstration for each function r, the algorithm requires the transition model in P. If such transition model is not available, then the robot will only be able to *replicate particular aspects of the demonstration*. However, as argued in [25], the imitative behavior obtained in these situations may not correspond to actual imitation.

Secondly, it may happen that the transition model available is *inaccurate*. In this situation (and unless the model is significantly inaccurate) the robot should still be able to perceive the demonstrated task. Then, given the estimated r-function, the robot may only be able to determine a *sub-optimal policy* and will need to resort to *experimentation* to improve this policy. We discuss these aspects in greater detail in the continuation.

C. Combining affordances with imitation learning

In this section, we discuss, in greater detail, how the information provided by the affordances described in Section II can be combined with the imitation learning approach described above. We discuss the advantages of this approach, as well, as several interesting issues that arise from this combination.

In the methodology described above, we assumed the robot to be able to recognize the actions performed by the demonstrator. This action recognition needs not to be explicit, *i.e.,* the agent needs not to determine the action taken by the demonstrator. Instead, it needs only to *interpret* the observed action in terms of its own action repertoire. This interpretation may rely on the observed state transition or in the corresponding effects. It is important to emphasize that transitions and effects are different concepts: the same transition may occur from different actions/effects and the same effect can be observed in different transitions. To clarify this distinction, consider moving or jumping from one place to the other: the effects are different but the transition is the same. For another example, consider moving between different places with the same speed: the effect is the same (motion at a given speed) but the transitions are different.

If no action recognition/interpretation takes place, the robot will generally be able to learn only how to replicate particular elements of the observed demonstration. In our approach we want the robot to *learn the task* more than to replicate aspects of the demonstration. As seen in Section II, affordances provide a functional description of the robot’s interaction with its surroundings as well as the action-recognition capabilities necessary to implement imitation.

Affordance-based action recognition/interpretation works as follows. For each demonstrated action, the robot observes the corresponding effect. The affordance network is then used to estimate the probability of each action

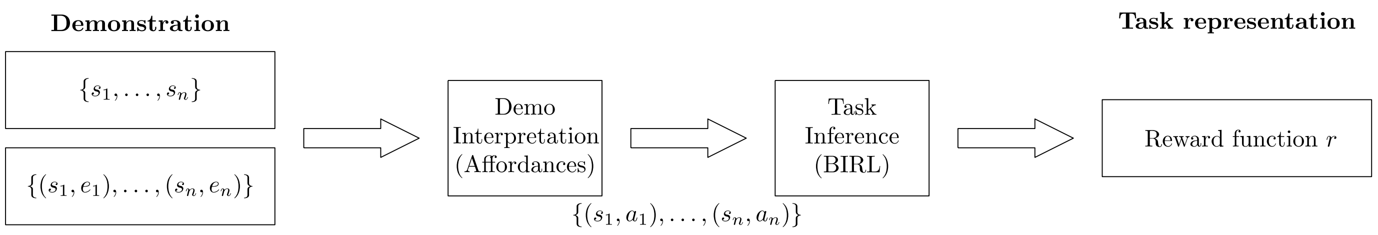

Demonstration
$\{s_1, \ldots, s_n\}$
$\{(s_1, e_1), \ldots, (s_n, e_n)\}$
Demo Interpretation (Affordances)
$\{(s_1, a_1), \ldots, (s_n, a_n)\}$
Task Inference (BIRL)
Task representation
Reward function r

Fig. 3: Representation of the fundamental elements of an imitation learner.

in the robot's action repertoire given the observed effects, and the action with greatest probability is picked as the observed action. Clearly, there will be some uncertainty in the identification of the demonstrated action, but as will be seen in the experimental section, this does not significantly affect the performance of the learning algorithm.

On the other hand, given the demonstration – consisting on a sequence of state-action pairs – the robot should be able to *infer* the task to be learned. This means in particular that once the robot realizes the task to be learned, it should be able to learn how to perform it *even in situations that were never demonstrated*.

Choosing between two policies generally requires the robot to have a model of the world. Only with a model of the world will the robot have the necessary information to realize *what task is more suitably accomplished by the demonstrated policy*. If no model of the world is available, then the robot will generally only *repeat the observed action pattern*, with no knowledge on what the underlying task may be. Also, the absence of a model will generally prevent the robot from *generalizing* the observed action pattern to situations never demonstrated.

As argued in Section II, affordances empower the robot with the ability to predict the effect of its actions in the surrounding environment. Once the adequate state-space for a particular task is settled, the information embedded in the affordance network can be used to extract the dynamic model describing the state evolution for the particular task at hand. This action-dependent dynamic model consists of the transition matrix P described in the previous subsection.

Figure 3 depicts the fundamental elements in the imitation learning architecture described.

It is important to notice that the affordance network is *task independent* and can be used to provide the required transition information for different tasks. Of course, the interaction model described in the affordance network could be enriched with further information concerning the state of the system for a specific task. This would make the extraction of the transition model automatic, but would render the affordance network *task-dependent*. This and the very definition of affordances justifies the use of a more general affordance model, even if requiring the transition model to be separately extracted for each particular task. This means that imitation can be successfully implemented in different tasks, provided that a single sufficiently general and task-independent model of interaction is available (such as the one provided by the affordances).

Another important observation is concerned with the fact that the affordance network is learned from interaction with the world. The combination of both learning blocks (affordance learning and imitation learning) gives rise to a complete architecture that allows the acquisition of skills ranging from simple action-recognition to complex sequential tasks.

In the remainder of the paper, we describe the implementation of this combined architecture in a humanoid robot. We illustrate the learning of a sequential task that relies on the interaction model described in the affordance network. We discuss the sensitivity of the imitation learning to action recognition errors.

IV. Experimental setup

In this section we present the robot used in the experiments, the experimental playground and the basic skills required to start interacting with the world and to learn the affordances. These skills include the basic motor actions and the visual perception of objects and effects.

A. Robot platform and playground

The experiments were done using BALTAZAR, a 14-degrees-of-freedom humanoid torso composed by a binocular head and an arm (see Fig. 6). The robot has implemented a set of parametrized actions based on a generic controller

$$\dot{\Theta} = m_i(\Theta^*, y, \lambda, \psi) \tag{6}$$

where Θ represents the controlled variables, Θ^* is the final objective and y the available proprioceptive measurements of the robot. Parameters ψ describe the kinematics/dynamics of the robot. Parameters λ can be used to shape the controller, *i.e.,* change desired velocities, energy criteria or postures. They can be tuned during affordance learning (refer to Fig. 8 in Section V), but are frozen by the system during the initial learning stage.

In this work we focus on object manipulation actions like grasping, tapping and touching (see Fig. 5). Each of these actions consist on three distinct steps: (i) bring the hand to the field of view in open-loop; (ii) approach the object using visual servoing and; (iii) actually grasp, tap or touch the object. The two former steps are learned by self-experience (see [33] for further details), while the latter is pre-programmed due to practical limitations of our current robotic platform. Using this approach, BALTAZAR is able to perform three different actions. In terms of our model, we have $\mathcal{A} = \{a_1 = grasp(\lambda), a_2 = tap(\lambda), a_3 = touch(\lambda)\}$, where λ represents the height of the hand in the 3D workspace when reaching the object in the image.

The robot executed its actions upon several different objects, each having one of two possible shapes ("box" and "ball"), four possible colors and three possible sizes (see Fig. 6). We recorded a set of 300 trials following the protocol summarized in Fig. 4. In each trial, the robot was presented with a random object. BALTAZAR randomly selected an action from its repertoire and approximated its hand to the object. When the reaching phase was

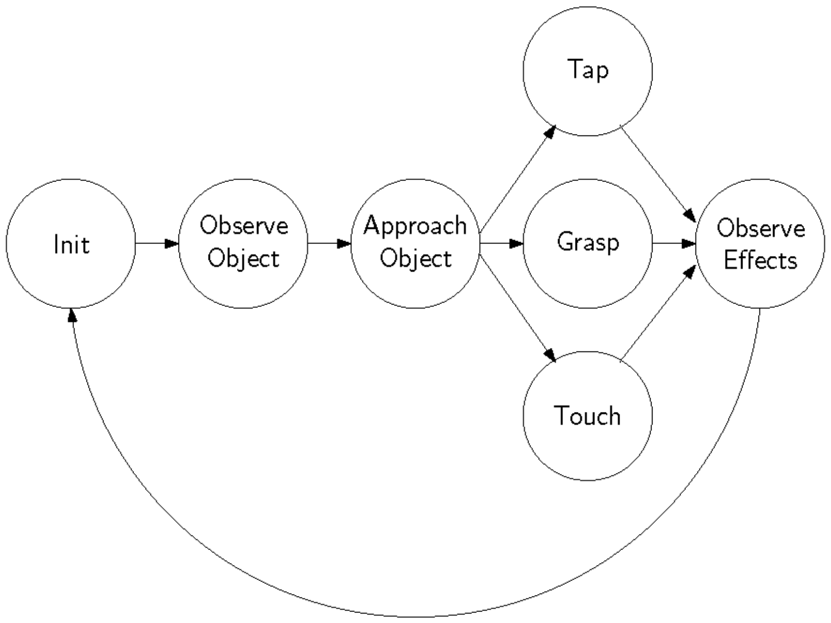
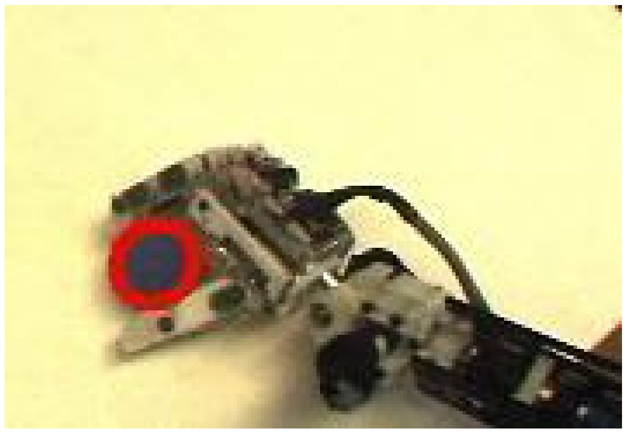
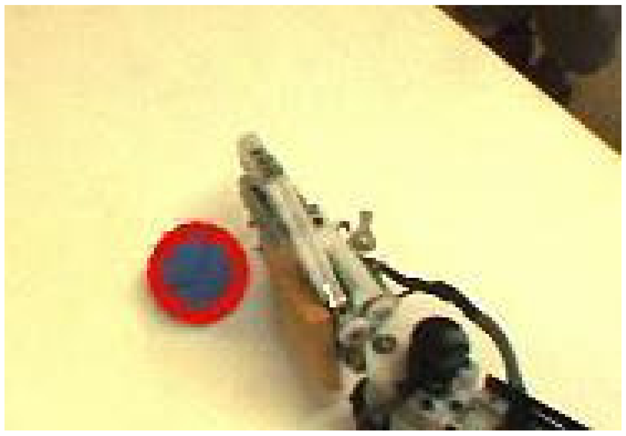


Fig. 4: Experiments protocol. The object to interact with is selected manually and the action is randomly selected. Object properties are recorded in the Init to Approach transition, when the hand is not occluding the object. The effects are recorded in the final Observe state. Init moves the hand to a predefined position in open-loop.



(a) Grasping.



(b) Tapping.

Fig. 5: Examples of actions as seen by the robot.

completed, it performed the selected action ($grasp(\lambda)$, $tap(\lambda)$ or $touch(\lambda)$) and returned the hand to the initial position. During action execution, the object features and effects were recorded. Note that the robot does not receive any feedback concerning the success or failure of the actions. The goal of this learning stage is to understand the causal relations between actions and effects in an unsupervised manner.

B. Visual Perception of Objects

Regarding object perception and feature extraction, we assume the system has simple segmentation and category formation capabilities already built-in. For the sake of experimental simplicity, we have constructed the "playground" environment shown in Fig. 6. In this environment, the robot plays with simple colorful objects over a white table

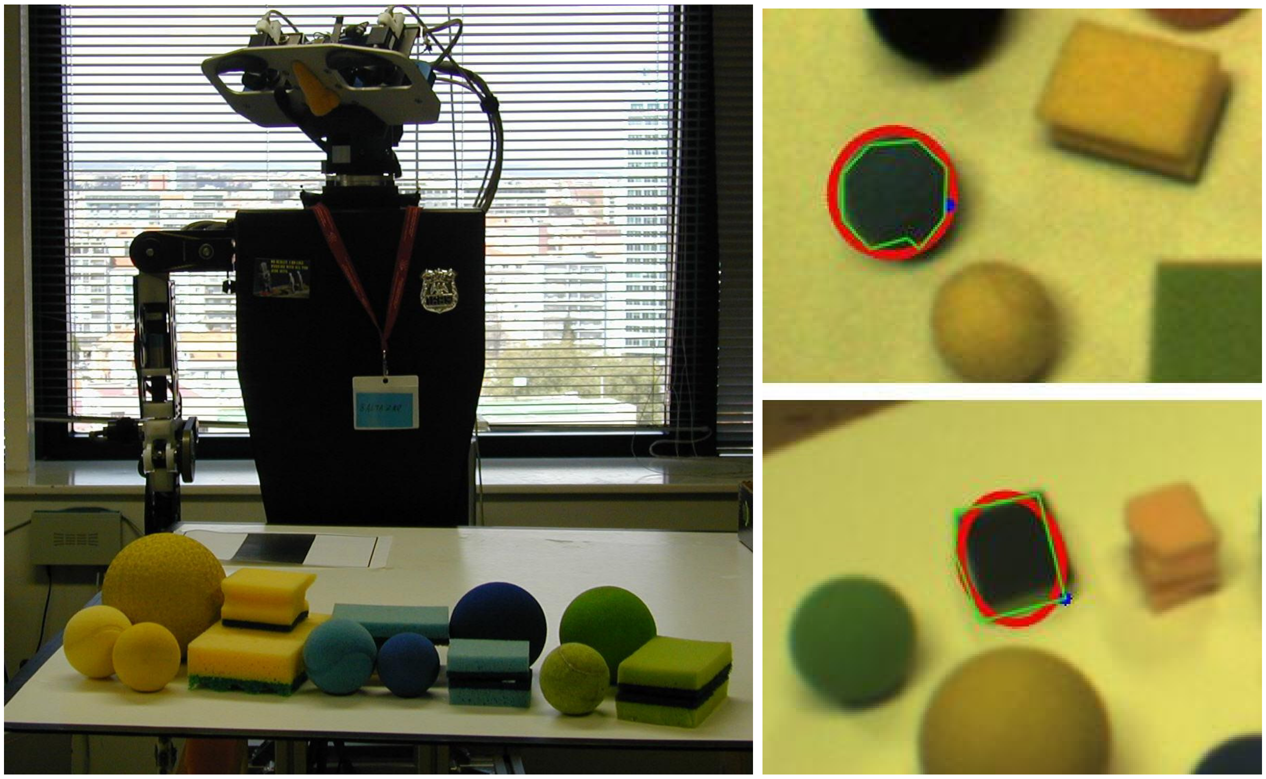


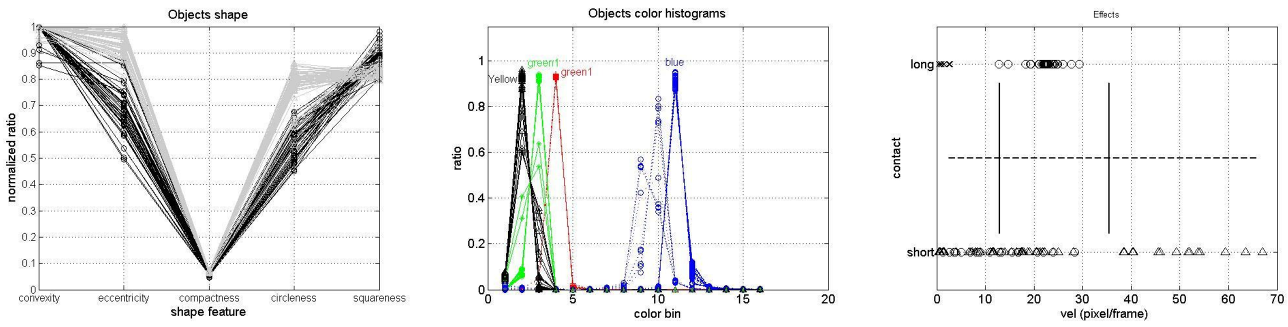
Fig. 6: Experimental setup. (Left) The robot's workspace consists of a white table and some colored objects with different shapes, sizes and colors. (Right) Object on the table are represented and categorized according to their size, shape and color, *e.g.,* 'ball' and 'square'.

and observes people playing with the same objects. At this stage, we rely on fast techniques like background and color segmentation to allow the robot to individuate and track objects in real-time. Along time, the robot collects information regarding simple visual object properties, like color, shape, size, etc. Figure 6 illustrates the robot's view of several objects, together with their color segmentation and extracted contour. After some time interacting with the objects, the robot is able to group their properties into meaningful categories. The set of visual features used consist of color descriptors, shape descriptors and size (in terms of the image). The color descriptor is given by the hue histogram of pixels inside the segmented region (16 bins). The shape descriptor is a vector containing measurements of convexity, eccentricity, compactness, roundness and squareness.

C. Perception of Effects

In our framework, effects are defined as salient changes in the perceptual state of the agent that can be correlated to actions. For example, upon interacting with an object, the robot may observe sudden changes in the object position or tactile information.

All effects are processed in the same way: when the action starts, the agent observes its sensory inputs during a certain time window that depends on the action execution time and the duration of the effects. It then records the corresponding information flow. We fit a linear model to the recorded temporal information associated with



(a) (b) (c)

Fig. 7: Clustering of object features and effects. (a) Shape description of the objects. Five features: convexity, eccentricity, compactness, roundness and squareness describe the objects. For the objects considered in the experiments, box and balls, they can be clustered automatically. Different clusters are represented by circles or plus signs. (b) Color histograms with the corresponding clusters. Each bin relates to a given Hue value. The clusters correspond to: $yellow$, $green_1$, $green_2$ and $blue$. (c) Clustering of object velocity and contact. For each observation grasp is represented by $\mathtt{x}$, tap by $\triangle$ and touch by $\circ$.The vertical lines show the clusters boundaries for velocity and the horizontal line for contact.

each observed effect and use the corresponding inclination and bias parameters as a compact representation of this effect. The regression parameters for the velocity of either an object, the hand or the “object-hand pair”, are determined from a sequence of image velocity norms. In this case, only the inclination parameter is used, since the bias parameter only reflects the absolute position in the image. Concerning contact effects, we consider only the bias parameter (offset), which gives a rough estimation of the duration of contact.

D. Discretization of Perceptual Information

This is an important step in the overall learning architecture, since it provides the basis for the discretization and categorization used in the affordance learning algorithm. In our example, we used the three features described below: color, shape and size. Each feature takes values in some n-dimensional vector space. We applied the *X-means* algorithm [46] to detect clusters in the space of each object feature and in the effects. We also discretized the space of the free actuator parameters, λ, using a predefined resolution and the same clustering algorithm. It is important to note that our ultimate goal is to learn the affordances given a set of available motor and perceptual skills and not to make a perfect object classification. Indeed, the clustering introduces some errors due, among other things, to different illumination conditions during the trials. As such, the features of some objects were misclassified and the affordance learning had to cope with this noise.

Figure 7(a) shows the results of the *X-means* algorithm for the object shape feature. The two resulting clusters are able to easily separate “balls” from “boxes” based mostly on roundness and eccentricity descriptors. Figure 7(b) gives the equivalent result for colors, where the feature vector is an histogram of hue values. As the objects have uniform color, each histogram has only one salient peak. Finally, for the unidimensional size, three clusters were

TABLE II: Summary of variables and values.

Symbol	**Description**	**Values**
A	Action	*Grasp, tap, touch*
H	Height	Discretized in 10 values
C	Color	*Green*$_1$*, green*$_2$*, yellow, blue*
Sh	Shape	*Ball, box*
S	Size	*Small, medium, big*
V	Object velocity	*Small, medium, big*
HV	Hand velocity	*Small, medium, big*
Di	Object-hand velocity	*small, medium, big*
Ct	Contact duration	*None, short, long*

enough to represent five different sizes of the objects presented to the robot.

Figure 7(c) shows the classes of object velocities and contact patterns detected by the robot, following the procedure described in Subsection IV-C. Roughly speaking, a grasp action resulted in “medium” velocity (except in one case where the ball fell down the table), a tap action produced different velocity patterns depending on the shape and size of the object, and a touch action induces small velocities. Also, contact information was more pronounced for grasp and touch actions than for tap ones. The combination of the different features produced patterns in the feature space that were used to infer statistical dependencies and causality. Table II summarizes the clustering results for the different variables and provides the notation used in the remainder of this section.

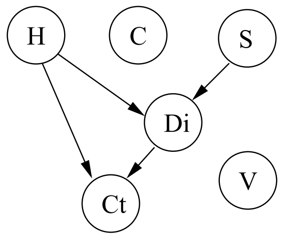
This categorization was conducted after having the robot interact with different objects for several trials, during which it collected information about the effects of its actions on the objects. The obtained clustering resulted in groups that are close in the sensory space. We thus have to assume that the motor and perceptual capabilities of the robot are such that the same action applied to the same object will in average yield similar effects – for example, all successful grasps will have the pressure sensors persistently activated. This clustering is not restricted to observed objects, because new objects will be categorized accordingly to their similarity to known ones.

V. Experimental Results

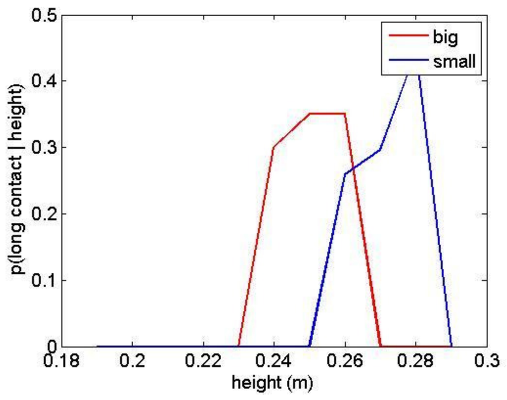
In this section we present the experimental results obtained with our approach. We start by illustrating the affordance learning stage, carefully evaluating its capabilities. We then proceed by present the application of the learned affordances in interaction games and in learning by demonstration of a complex task.

A. *Affordances*

We now describe the different experiments conducted to illustrate the ability of the proposed model to capture object affordances.



(a)



(b)

Fig. 8: Tuning the height for grasping a ball. (a) Dependencies discovered by the learning algorithm. The action and shape for this example are fixed and the color does not have an impact on the effects. Node labels can be found in Table II. (b) CPD of the height value given that the robot obtained a long contact (successful grasp).

1) Controller Optimization: The objective of the first experiment is to find the influence of a free actuator parameter on an action. The robot tries the action for different values of the free parameters. For a grasp, these parameters are closure of the fingers and approaching height of the hand. The former is used after reaching the object in the closing of the hand, whereas the latter is a free parameter of the sensory-motor map used to approximate the hand to the object. We computed the maximum likelihood graph with a random starting point and BDeu priors [37] to give uniform priors to different equivalence classes.[4]

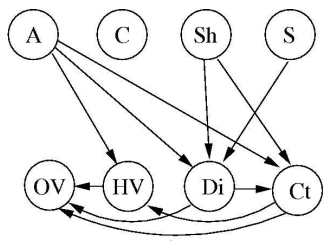
Figure 8(a) shows how the resulting network captures the dependency of the effects on these parameters. Interestingly, the CPDs provide the probability of producing different effects according to the values of the free parameters. Figure 8(b) shows the estimated probability of different height values conditioned on the observation of a long contact – indicating a successful grasp – for medium and small objects. Since big objects cannot be grasped by the robot's hand, all heights have zero probability for this class.

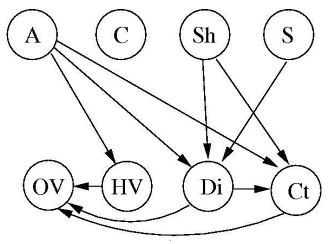
Note that the distribution of Fig. 8(b) can be used directly to adjust the height of the action for different object sizes and, as such, perform an optimization of the controller parameter based on the target object.

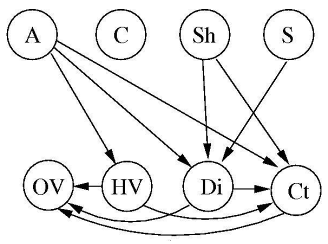
2) Affordance Network Learning: In the second experiment, we illustrates the robot's ability to distinguish the effects of different actions and simultaneously identify the object features that are relevant to this purpose. As in the previous experiment, we use BDeu priors and random initialization.[5] For the MCMC algorithm, we used $5,000$

[4]The implementation of the algorithms is based on the BNT toolbox for Matlab, http://bnt.sourceforge.net/.

[5]Although it is possible to use conditional independence tests to provide a rough initialization, in our case we got similar results using randomly generated networks.


(a)


(b)


(c)

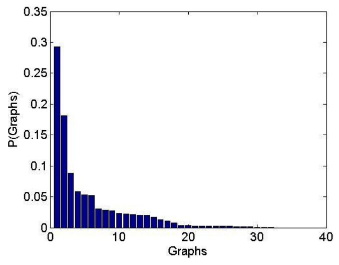

(d)

Fig. 9: Affordance model estimated by the MCMC algorithm. Node labels can be found in Table II. (a-c) Three most likely network structures obtained by the MCMC algorithm for three different datasets of common length. (d) Posterior probability over graphs, as computed by MCMC.

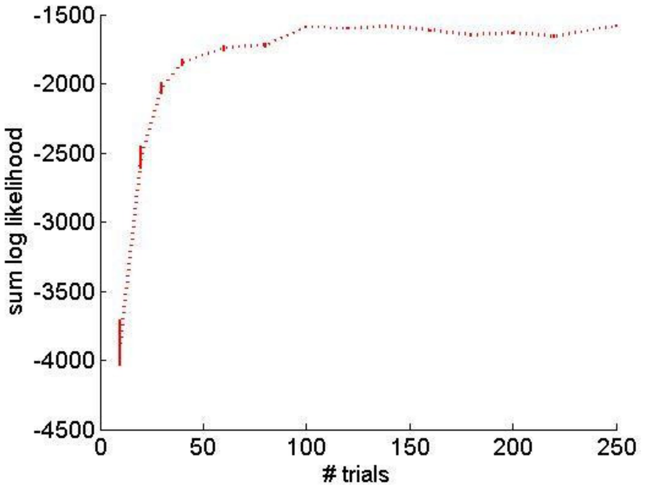
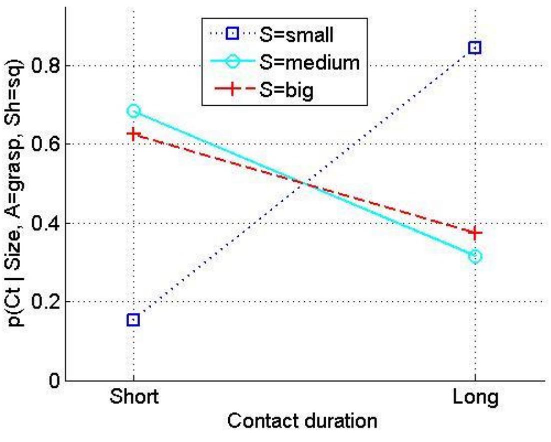


Fig. 10: Marginal likelihood of the data given the learned network structure as the number of trials included in the dataset increases. The vertical bars show the variance of the likelihood.

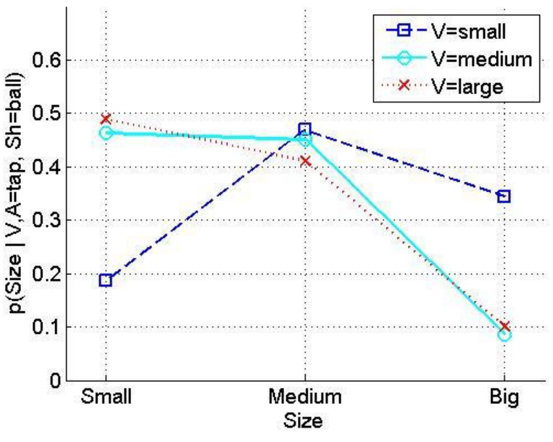
samples with a burn-in period of 500 steps.

Figures 9(a) through 9(c) show the three most likely networks computed by MCMC and Fig. 9(d) shows the posterior probability distribution over all sampled models. In order to show the convergence of the network toward a plausible model, we have estimated a network structure using datasets of different lengths. For each length, we have randomly created 100 datasets from the complete dataset, estimated the posterior over graph structures using MCMC and computed the likelihood of the whole data for the most likely model. Figure 10 shows how the marginal likelihood of the data converges as the length of the dataset increases. The figure also indicates that, after 100 trials, the improvement of the likelihood of the data given more experiments is very small, since the model was already able to capture the correct relations.

3) Affordances Conditional Probability Distributions: To ensure that the network actually captures the correct dependencies, we computed some illustrative distributions. The actual dependencies are encoded in the multinomial CPDs of each node and, as such, we can not rely on the typical mean squared error to validate the fitting on the



(a)



(b)

Fig. 11: Examples of CPDs for the learned network. (a) CPD of the contact duration when grasping a box, for different values of the size parameter – $\mathbb{P}\left[Ct \mid S, A = grasp, Sh = box\right]$. (b) CPD of the size of a tapped ball, for different values of the observed velocity – $\mathbb{P}\left[S \mid V, A = tap, Sh = ball\right]$.

training set. Although there is no ground truth to compare the estimated network structure, we see that color has been detected as irrelevant when performing any action. Shape and size are important for grasp, tap and touch since they have an impact on the observed velocities and contact.

Figure 11(a) depicts the predicted contact duration for a grasp action on a ball of different sizes. It basically states that successful grasps (longer contact between the hand and the object) occur more often with smaller balls than with bigger ones. Figure 11(b) shows the distribution over ball sizes after a tap action for different observed velocities. According to these results, smaller balls move faster than bigger ones and medium ball velocities are somewhat unpredictable (all velocities have similar likelihood). This actually reflects the behavior of the objects during the trials. For example, the mean and variance of the ball velocity (μ [pixel/frame] and σ^2 [pixel^2/frame^2]) were $(33.4, 172.3)$, $(34.3, 524.9)$ and $(17.5, 195.5)$ for a small, a medium and a big ball, respectively.

We can also verify if the robot is able recognize the actions in the training set. To this purpose, we performed leave-one-out cross-validation. For each trial, we computed the network structure and parameters using the data from the remaining trials and the MCMC algorithm. We then estimated the probability of each action given the object features and the object velocity, hand velocity and object-hand velocity. Since contact is a proprioceptive measurement, it is not usually available when observing others actions. The most likely action was correctly identified in more than 85% of the tests. The errors were due mainly to the absence of contact information, which makes touching and tapping of boxes very similar from the point of view of observed effects. After including contact information, the ratio of correct recognition raised to 98%.

4) Summary: We have shown how the robot can tune its motor controllers through experimentation by including the effects of its actions. Once this information is available, the robot can start to establish relationships between the features of the objects and the effects observed as resulting from its actions. The learning of the affordance model depends on the motor and perceptual skills of the robot and was conducted in a completely unsupervised manner. There is no notion of success or failure and the network may not be able to distinguish between non-separable objects, given the used descriptors. However, it is still able to construct a plausible model of the behavior of the different objects under different actions that can readily be used for prediction and planning.

B. Interaction games

Before delving in the problem of imitation learning, we present the results obtained in several simple interaction games using the learned affordance network. These results further validate the use of our proposed affordance model and illustrate the basic prediction, recognition and planning capabilities at the core of imitation learning.

We implemented a one-step emulation behavior. The robot observes a demonstrator perform an action on a given object. Then, using one of the functions described in Fig. 1, it selects an action/object that is more likely to achieve the observed effect.

Figure 12 depicts the demonstration, the objects presented to the robot and the selected action/object for different situations. We used two different demonstrations, a tap on a small ball, resulting in high velocity and medium hand-object distance, and a grasp on a small square, resulting in small velocity and small hand-object distance. Notice that contact information is not available when observing others. The goal of the robot is to replicate the observed effects.

TABLE III: Probability of achieving the desired effects for each possible action and each object in Fig. 12(b).

Object $\backslash$ Action	**Grasp**	**Tap**	**Touch**
Blue, big, ball	0.00	0.20	0.00
Yellow, small, box	0.00	0.06	0.00

The first situation (Fig. 12(a)) is trivial, as only tap has a non zero probability of producing high velocity. Hence, the imitation function selected a tap on the only object available. In Fig. 12(b) the demonstrator performed the same action, but the robot had to decide between two different objects. Table III summarizes the probability of observing the desired effects given the six possible combinations of actions and objects. The robot selected the one with highest probability and thus performed a tap on the ball.

Figures 12(c) and (d) illustrate how the inclusion of the object features in the goal criteria may lead to different behaviors. After observing the grasp demonstration, the robot had to select among three objects: a big yellow ball, a small yellow ball and a small blue box. In the first case (Fig. 12(c)), the objective was to replicate the same effects. The probability for each of the objects is 0.88, 0.92 and 0.52, respectively, and the robot grasped the small yellow ball even if the object used in the demonstration was different and was also on the table. Notice that this

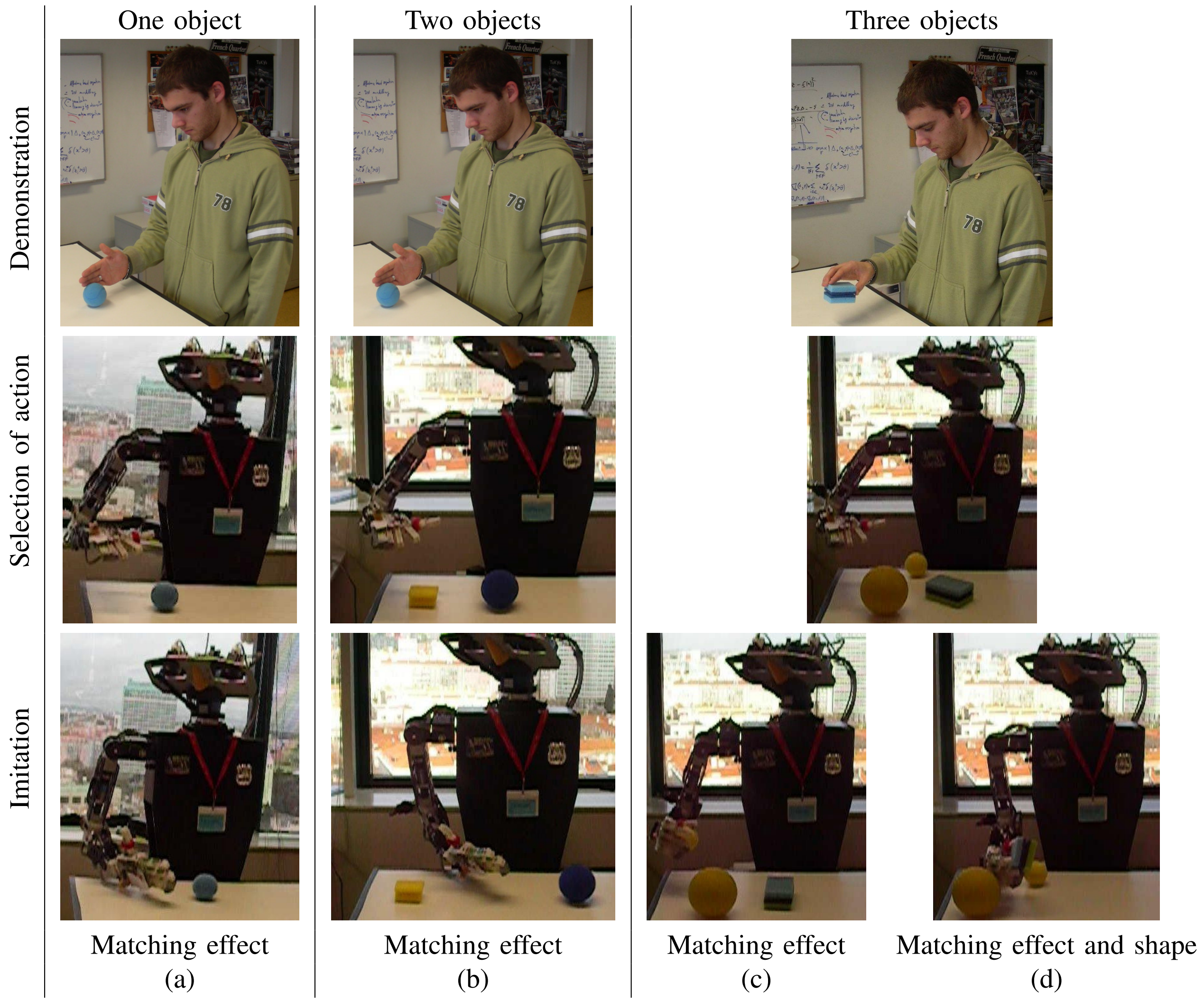


Fig. 12: Different imitation behaviors. (Top) Demonstration. (Middle) Set of possible objects. (Bottom) Imitation. (a-c) Imitation by matching the observed effect. (d) Imitation by matching the observed effect and shape.

is not a failure, since it maximizes the probability of a successful grasp. This was the only requirement specified by the task goal. When the goal is modified to also include a similar shape, the robot successfully selects the blue box (Fig. 12(d)).

In the continuation, we address the more complex scenario in which the robot must learn by imitation a full sequence of actions.

C. Imitation learning

To implement the imitation learning algorithm described in Section III, we considered a simple recycling game in which the robot must separate different objects according to their shape (Fig. 13). In front of the robot are two slots (Left and Right), where three types of objects can be placed: Large Balls, Small Balls and Boxes. The boxes should be dropped in a corresponding container and the small balls should be tapped out of the table. The large balls should be touched upon, since the robot is not able to efficiently manipulate them. Every time a large ball is touched, it is removed from the table by an external agent. Therefore, the robot has available a total of 6 possible actions: Touch Left (TcL), Touch Right (TcR), Tap Left (TpL), Tap Right (TpR), Grasp Left (GrL) and Grasp Right (GrR).

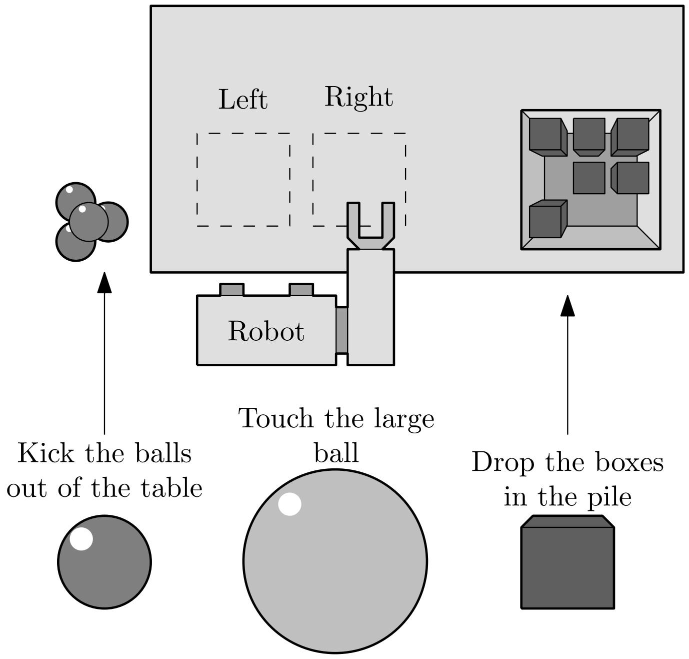


Fig. 13: A simple recycling game.

To describe the process $\{X_t\}$ for the task at hand, we considered a state-space consisting of 17 possible states. Of these, 16 correspond to the possible combinations of objects in the two slots (including empty slots). The 17th state is an invalid state that accounts for the situations where the robot's actions do not succeed. As described in Section III, determining the dynamic model consists of determining the transition matrix P by considering the possible effects of each action in each possible object. From the affordances in Figure 9 the transition model for the actions on each object are shown in Figure 14. Notice that, if the robot taps a ball on the right while an object is lying on the left, the ball will most likely remain in the same spot. However, since this behavior arises from the presence of two objects, *it is not captured in the transition model* obtained from the affordances. This means that the transition model extracted from the affordances necessarily includes some inaccuracies.

To test the imitation, we provided the robot with an error-free demonstration of the desired behavior rule. As expected, the robot was successfully able to reconstruct the optimal policy. We also observed the learned behavior when the robot was provided with *two* different demonstrations, both optimal, as described in Table IV. Each state is represented as a pair (S_1, S_2) where each S_i can take one of the values "Ball" (Big Ball), "ball" (Small Ball), "Box" (Box) or $\emptyset$ (empty). The second column of the table lists the observed actions for each state, and the third column lists the learned policy. Notice that, once again, the robot was able to reconstruct an optimal policy, by choosing one of the demonstrated actions in those states where different actions were possible.

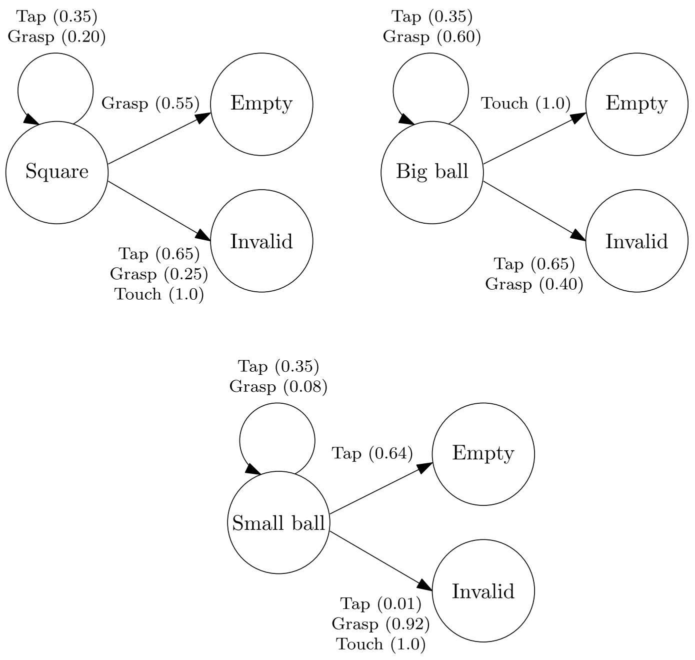
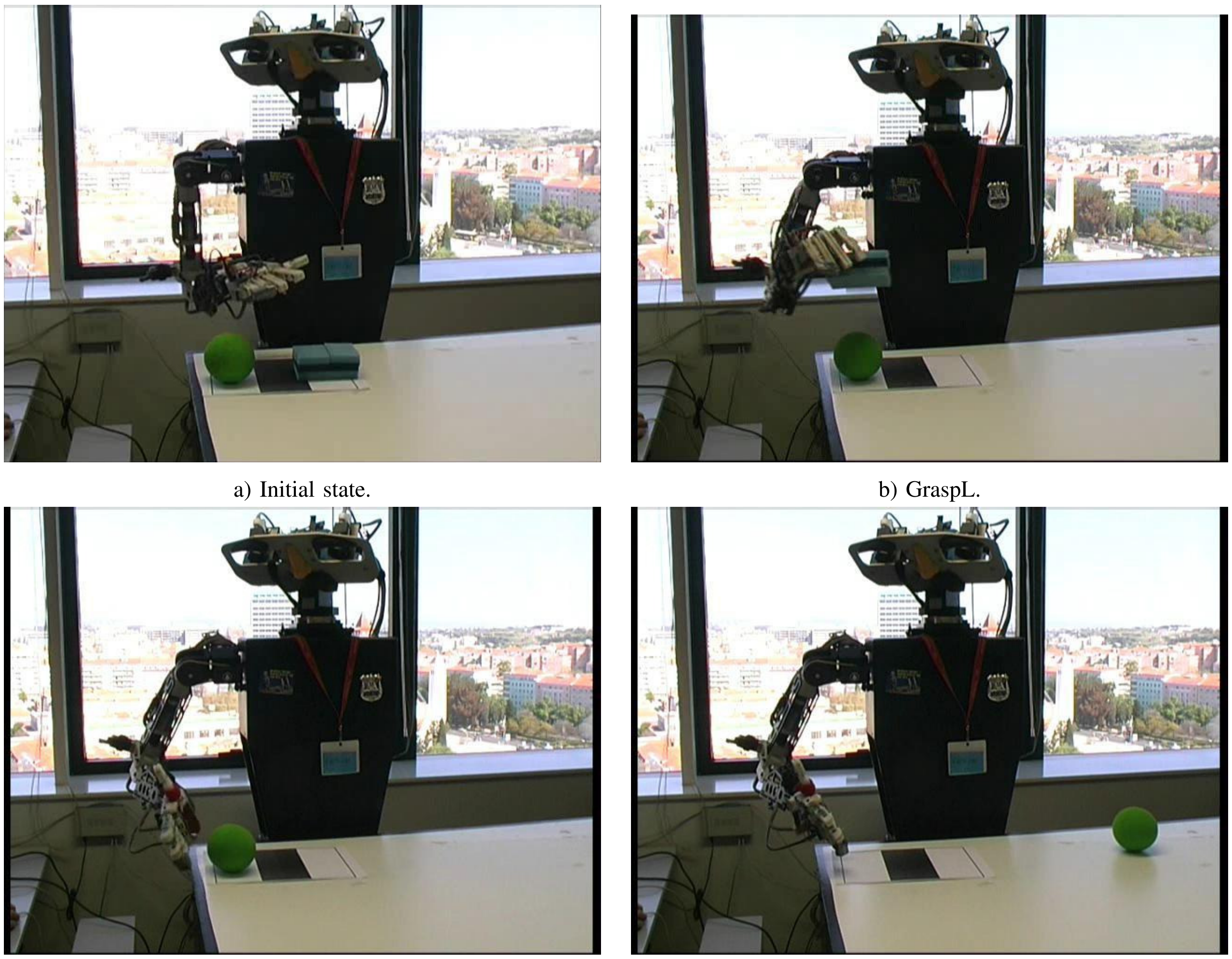


Fig. 14: Transition diagrams describing the transitions for each slot/object.

TABLE IV: Experiment 1: Error free demonstration (demonstrated and learned policies). Experiment 2: Inaccurate, incomplete demonstration (demonstrated and learned policies). The boxed lines in Demonstration 2 correspond to the incomplete and inaccurate actions.

State	**Demo1**	**Learned**	**Demo2**	**Learned**
$(\emptyset, \text{Ball})$	TcR	TcR	$\boxed{-}$	TcR
$(\emptyset, \text{Box})$	GrR	GrR	GrR	GrR
$(\emptyset, \text{ball})$	TpR	TpR	TpR	TpR
$(\text{Ball}, \emptyset)$	TcL	TcL	TcL	TcL
(Ball, Ball)	TcL,TcR	TcL,TcR	$\boxed{\text{GrR}}$	TcL
(Ball, Box)	TcL,GrR	GrR	TcL	TcL
(Ball, ball)	TcL	TcL	TcL	TcL
$(\text{Box}, \emptyset)$	GrL	GrL	GrL	GrL
(Box, Ball)	GrL,TcR	GrL	GrL	GrL
(Box, Box)	GrL,GrR	GrR	GrL	GrL
(Box, ball)	GrL	GrL	GrL	GrL
$(\text{ball}, \emptyset)$	TpL	TpL	TpL	TpL
(ball, ball)	TpL,TcR	TpL	TpL	TpL
(ball, Box)	TpL,GrR	GrR	TpL	TpL
(ball, ball)	TpL	TpL	TpL	TpL

In another experiment, we provided the robot with an *incomplete and inaccurate* demonstration. In particular, the action at state ($\emptyset$, Ball) was never demonstrated and the action at state (Ball, Ball) was *wrong*. Table IV shows the demonstrated and learned policies. Notice that in this particular case the robot was able to recover the *correct* *policy*, even with an incomplete and inaccurate demonstration. Figure 15 illustrates the execution of the optimal learned policy for the initial state (Box, SBall).[6]



a) Initial state.

b) GraspL.

c) TapR.

d) Final state.

Fig. 15: Execution of the learned policy in state (Box, SBall).

To assess the sensitivity of the imitation learning module to the action recognition errors, we tested the learning algorithm for different error recognition rates. For each error rate, we ran 100 trials. Each trial consists of 45 state-action pairs, corresponding to three optimal policies. The obtained results are depicted in Figure 16.

As expected, the error in the learned policy increases as the number of wrongly interpreted actions increases. Notice, however, that for small error rates ($\leq 15\%$) the robot is still able to recover the demonstrated policy with an error of only around 1%. In particular, if we consider the error rates of the implemented action recognition method (between 10% and 15%), we observe that the optimal policy is accurately recovered. This allows us to conclude that action recognition using the affordances is sufficiently precise to ensure the recovery of the demonstrated policy.

The accuracy of the recognition varied, depending on the performed action, on the demonstrator and on the speed

[6]For videos showing additional experiences see http://vislab.isr.ist.utl.pt/baltazar/demos/

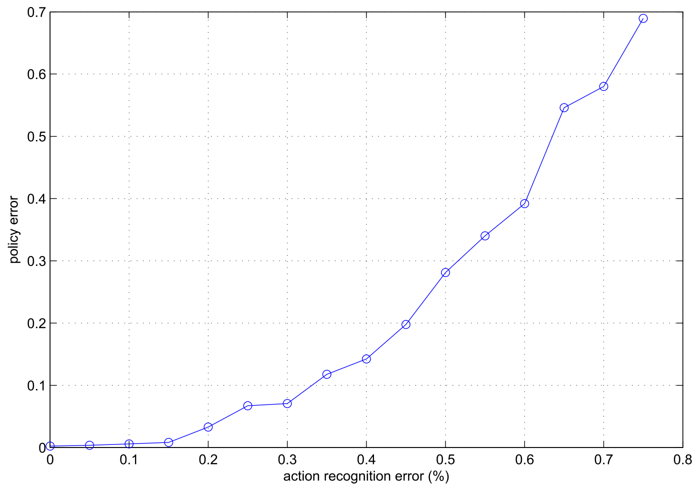


Fig. 16: Percentage of wrong actions in the learned policy as the action recognition errors increase.

of execution, but for all actions the recognition was successful with an error rate between 10% and 15%. The errors in action recognition are justified by the different view-points during the learning of the affordances and during the demonstration. In other words, the robots learns the affordances by looking at its own body motion, but the action recognition is conducted from an external point-of-view. In terms of the image, this difference in viewpoints translates in differences on the observed trajectories and velocities, leading to some occasional mis-recognitions. We refer to [29] for a more detailed discussion of this topic.

VI. Conclusions

We have presented a computational model of affordances relying on Bayesian networks and, based on it, an imitation learning framework. On one hand, the model captures the relations between actions, object properties and the expected action outcomes. Using well-established learning and inference algorithms, a robot can learn these relations in a complete unsupervised way simply by interacting with the environment. Our results show how the learned network captures the structural dependencies between actions, object features and effects even in the presence of the perceptual and motor uncertainty inherent to real-world scenarios.

On the other hand, affordances provide the basic skills required for social interaction. The proposed imitation framework learns reward functions from task demonstrations. It uses the inference capabilities of the learned affordance network to recognize the demonstrated actions, predict the potential effects and plan accordingly. In this sense, imitation is not limited to mimicking the detailed human actions. Instead, it is used in a goal-directed manner (emulation), since the robot may choose a very different action when compared to that of the demonstrator, as long as its experience indicates that the desired effect can be met.

The model has a number of interesting properties that are worth pointing out: not only does it bridge the gap between sensory-motor loops and higher cognitive levels in humanoid robots but also allows us to possibly gain some insight concerning human cognition. One interesting remark is that, by using this approach, objects are not represented only in terms of their visual attributes (*e.g.,* shape) but also taking into account their "behavior" when subject to actions. In other words, "objecthood" is defined as a consequence of the robot's own actions (and embodiment) and the corresponding action outcomes. This concept of action-based object categorization is fundamental for planning but it is also relevant from the point of view of human perception and behavior.

A second point to notice in our affordance model is that the same basic knowledge and inference methods are used both for action selection and action recognition. In a sense, our model displays a "mirror" structure as suggested by the existence of the so-called mirror neurons in the pre-motor cortex of macaque monkeys. The same same neuronal structures simultaneously support action generation and recognition.

Finally, we have illustrated how the ability of the learned model to predict the effects of actions and recognize actions can be used to play simple interaction games and to implement learning by imitation in a robot.

We believe that modeling the interplay between actions, objects and actions outcomes is a powerful approach not only to develop goal-oriented behavior, action recognition, planning and execution in humanoid robots but also to shed some light into some of the fundamental mechanisms associated with human learning and cognition.

References

[1] L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor, "Learning object affordances: From sensory–motor coordination to imitation," *IEEE Transactions on Robotics*, vol. 28, no. 1, February 2008.

[2] M. Lopes, F. S. Melo, and L. Montesano, "Affordance-based imitation learning in robots," in *IEEE/RSJ Intelligent Robotic Systems (IROS'07)*, USA, 2007.

[3] J. J. Gibson, *The Ecological Approach to Visual Perception*. Boston: Houghton Mifflin, 1979.

[4] H. Ornkloo and C. von Hofsten, "Fitting objects into holes: on the development of spatial cognition skills," *Developmental Psychology*, vol. 43, no. 2, pp. 404–416, 2007.

[5] S. Schaal, "Is imitation learning the route to humanoid robots," *Trends in Cognitive Sciences*, vol. 3(6), 1999.

[6] A. Chemero, "An outline of a theory of affordances," *Ecological Psychology*, vol. 15, no. 2, pp. 181–195, 2003.

[7] J. Konczak, H. Meeuwsen, and M. Cress, "Changing affordances in stair climbing: The perception of maximum climbability in young and older adults," *Journal of Experimental Psychology: Human Perception & Performance*, vol. 19, pp. 691–7, 1992.

[8] E. Symes, R. Ellis, and M. Tucker, "Visual object affordances: Object orientation," *Acta Psychologica*, vol. 124, no. 2, pp. 238–255, 2007.

[9] M. Turvey, K. Shockley, and C. Carello, "Affordance, proper function, and the physical basis of perceived heaviness," *Cognition*, vol. 73, 1999.

[10] E. Oztop, N. Bradley, and M. Arlib, "Infant grasp learning: a computational model," *Experimental Brain Research*, vol. 158, pp. 480–503, 2004.

[11] M. Lopes and J. Santos-Victor, "Visual learning by imitation with motor representations," *IEEE - Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, vol. 35, no. 3, June 2005.

[12] A. Slocum, D. Downey, and R. Beer, "Further experiments in the evolution of minimally cognitive behavior: from perceiving affordances to selective attention," in *Conference on simulation of adaptive behavior*, Paris, France, 2000.

[13] P. Fitzpatrick, G. Metta, L. Natale, S. Rao, and G. Sandini., "Learning about objects through action: Initial steps towards artificial cognition," in *IEEE International Conference on Robotics and Automation*, Taipei, Taiwan, 2003.

[14] H. Kozima, C. Nakagawa, and H. Yano, "Emergence of imitation mediated by objects," in *Second International Workshop on Epigenetic Robotics*, Edinburgh, Scotland, 2002.

[15] I. Cos-Aguilera, L. Cañamero, and G. Hayes, "Using a SOFM to learn object affordances," in *Workshop of Physical Agents (WAF)*, Girona, Spain, 2004.

[16] G. Fritz, L. Paletta, R. Breithaupt, E. Rome, and G. Dorffner, "Learning predictive features in affordance based robotic perception systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, 2006.

[17] A. Stoytchev, "Behavior-grounded representation of tool affordances." in *International Conference on Robotics and Automation*, Barcelona, Spain, 2005.

[18] E. Sahin, M. Cakmak, M. Dogar, E. Ugur, and G. Ucoluk, "To afford or not to afford: A new formalization of affordances towards affordance-based robot control," *Adaptive Behavior. In press*, vol. 124, no. 2, pp. 238–255, 2007.

[19] M. Dogar, M. Cakmak, E. Ugur, and E. Sahin, "From primitive behaviors to goal-directed behavior using affordances," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.

[20] A. Alissandrakis, C. L. Nehaniv, and K. Dautenhahn, "Action, state and effect metrics for robot imitation," in *15th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN 06)*, Hatfield, United Kingdom, 2006, pp. 232–237.

[21] A. Billard, Y. Epars, S. Calinon, G. Cheng, and S. Schaal, "Discovering optimal imitation strategies," *Robotics and Autonomous Systems*, vol. 47:2-3, 2004.

[22] A. Murata, L. Fadiga, L. Fogassi, V. Gallese, V. Raos, and G.Rizzolatti, "Object representation in the ventral premotor cortex (area f5) of the monkey," *J Neurophysiol*, vol. 78, pp. 2226–30, 1997.

[23] L. Fadiga, L. Fogassi, G. Pavesi, and G. Rizzolatti, "Motor facilitation during action observation: A magnetic stimulation study," *Journal of Neurophysiology*, vol. 73, no. 73, pp. 2608–2611, 1995.

[24] V. Gallese, L. Fadiga, L. Fogassi, and G. Rizolaatti, "Action recognition in the premotor cortex," *Brain*, vol. 119, pp. 593–609, 1996.

[25] F. Melo, M. Lopes, J. Santos-Victor, and M. I. Ribeiro, "A unified framework for imitation-like behaviors," in *4th International Symposium in Imitation in Animals and Artifacts*, Newcastle, UK, April 2007.

[26] A. Whiten, V. Horner, C. A. Litchfield, and S. Marshall-Pescini, "How do apes ape?" *Learning & Behavior*, vol. 32, no. 1, pp. 36–52, 2004.

[27] G. Gergely, H. Bekkering, and I. Király, "Rational imitation in preverbal infants," *Nature*, vol. 415, p. 755, Feb 2002.

[28] H. Bekkering, A. Wohlschläger, and M. Gattis, "Imitation of gestures in children is goal-directed." *Quarterly Journal of Experimental Psychology*, vol. 53A, pp. 153–164, 2000.

[29] M. Lopes and J. Santos-Victor, "Visual transformations in gesture imitation: What you see is what you do," in *IEEE - International Conference on Robotics and Automation*, Taipei, Taiwan, 2003.

[30] J. Weng, "The developmental approach to intelligent robots," in *AAAI Spring Symposium Series, Integrating Robotic Research: Taking The Next Leap*, Stanford, USA, Mar 1998.

[31] M. Lungarella, G. Metta, R. Pfeifer, and G. Sandini, "Developmental robotics: a survey," *Connection Science*, vol. 15, no. 40, pp. 151–190, December 2003.

[32] E. Spelke, "Core knowledge," *American Psychologist*, vol. 55, pp. 1233–1243, 2000.

[33] M. Lopes and J. Santos-Victor, "A developmental roadmap for learning by imitation in robots," *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, vol. 37, no. 2, April 2007.

[34] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[35] D. Ramachandran and E. Amir, "Bayesian inverse reinforcement learning," in *20th Int. Joint Conf. Artificial Intelligence*, India, 2007.

[36] J. Pearl, *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2000.

[37] D. Heckerman, D. Geiger, and M. Chickering, “Learning bayesian networks: the combination of knowledge and statistical data,” *Machine Learning*, 1995.

[38] G. Cooper and E. Herskovits, “A bayesian method for the induction of probabilistic networks from data,” *Machine Learning*, vol. 9, no. 4, pp. 309–347, 1992.

[39] D. Madigan and J. York, “Bayesian graphical models for discrete data,” *Intl. Statistical Review*, vol. 63, pp. 215–232, 1995.

[40] N. Firedman, “The bayesian structural em algorithm,” in *Uncertainty in Artificial Intelligence, UAI*, 1998.

[41] D. Eaton and K. Murphy, “Exact bayesian structure learning from uncertain interventions,” *AI & Statistics*, In Press.

[42] D. Heckerman, “A tutorial on learning with bayesian networks,” in *In M. Jordan, editor, Learning in graphical models.* MIT Press, 1998.

[43] C. Huang and A. Darwiche, “Inference in belief networks: A procedural guide,” *International Journal of Approximate Reasoning*, vol. 15, no. 3, pp. 225–263, 1996.

[44] M. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1994.

[45] A. Y. Ng and S. J. Russel, “Algorithms for inverse reinforcement learning,” in *Proc. 17th Int. Conf. Machine Learning*, USA, 2000.

[46] D. Pelleg and A. W. Moore, “X-means: Extending k-means with efficient estimation of the number of clusters,” in *International Conference on Machine Learning*, San Francisco, CA, USA, 2000.