The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

An Adaptive Probabilistic Approach to Goal-Level Imitation Learning

Haris Dindo and Guido Schillaci

Abstract— **Imitation learning has been recognized as a promising technique to teach robots advanced skills. It is based on the idea that robots could learn new behaviors by observing and imitating the behaviors of other skilled actors. We propose an adaptive probabilistic graphical model which copes with three core issues of any imitative behavior: observation, representation and reproduction of skills. Our model, *Growing Hierarchical Dynamic Bayesian Network (GHDBN)*, is *hierarchical* (i.e. able to characterize structured behaviors at different levels of abstraction), and *growing* (i.e. skills are learned or updated incrementally - and at each level of abstraction - every time a new observation sequence is available). A GHDBN, once trained, is able to recognize skills being observed and to reproduce them by exploiting the generative power of the model. The system has been successfully tested in simulation, and initial tests have been conducted on a NAO humanoid robot platform.**

I. Introduction

The state-of-the art robotic systems, while exhibiting impressive performances in hardware design and control algorithms, posses restricted capabilities of perception, reasoning and action in novel and unstructured environments. In order to overcome these limitations, robots should understand the actions performed by others, infer their intentions, and formulate appropriate responses based on the current context and their own internal motivations and drives. Furthermore, robots should be able to acquire new skills, behaviors and knowledge through an active interaction with other agents.

Imitation learning has been recognized as a promising technique to teach robots advanced skills. It is based on the idea that robots could learn new behaviors by observing and imitating the behaviors of other skilled actors. Each imitative process, no matter how complex it is, is essentially composed of three fundamental steps: *observation*, *representation* and *reproduction*. Classical methods of Programming by Demonstration have usually been focused on reproducing the surface of the demonstrated behavior [6]. However, in order to achieve the highest level of imitation, we must go beyond the surface of the perceived sensorial patterns. Our aim is to have robots capable of advanced social interactions and imitation, where imitation is seen not only as the process of copying an observed motor act without any understanding, but rather as a process of inference intention where the goal hidden in the observation of the act is imitated. This *goal-level imitation*, in our view, is a major prerequisite for having artificial agents capable of advanced interaction with the world, and able to adapt and learn from it.

The authors are with the Department of Computer Science, University of Palermo, Italy (dindo,schillaci@dinfo.unipa.it)

Recent hypotheses, corroborated by several empirical findings, postulate that the understanding of the external world (in terms of objects, actions performed on objects, and intentions) is achieved by employing one's own internal structures used to actively interact with the external world, i.e. one's own motor apparatus. These ideas originated in non-robotics scientific communities, such as cognitive science and neuroscience, by trying to explain advanced social cognition capabilities in humans [7]. This has inspired the use of internal models (which simulate aspects of planning, control and learning) for action recognition and reproduction in robotics [5]. However, recent studies have discovered that novel behaviors are recognized by an inferential interpretive system rather than by the Mirror Neuron System (MNS) [8]. Indeed, the MNS does not infer the high level intention of a complex action (see [9] for a review). An observer, to understand the intention of an action, must be able to describe it either at the goal level (i.e. representing, as done by the MNS, the short term goals necessary to realize the intention) or at the more abstract intention level (not encoded by the MNS, as claimed in [8]) by having only access to the visual information.

Recognizing the goals and intentions of an acting agent could be computationally interpreted as a problem of model matching. Demiris describes two approaches for classifying an observation: descriptive or generative [4]. In the former, a set of low-level features are extracted from the observed pattern and then matched against pre-existing representations (which are labeled with the goals and intentions that underlie their execution) to generate the actions corresponding to these representations. Within a generative approach, a set of latent variables encode the causes capable of producing the observed data. Billard et al., in [6], defines two typologies of skill representation: trajectory-level encoding (processes are represented as a non-linear mapping between sensory and motor data) and symbolic-level encoding (tasks are described symbolically using classical machine learning techniques). By using the symbolic approach, one can abstractly represent hierarchies of behaviors and sequences of states making the intention recognition easier.

We propose an adaptive probabilistic graphical model which copes with the core issues of any imitative behavior: observation, representation and reproduction of skills. Our model, *Growing Hierarchical Dynamic Bayesian Network (GHDBN)*, is *hierarchical* (i.e. able to characterize structured behaviors at different levels of abstraction), and *growing* (i.e. skills are learned or updated *incrementally* - and at each level of abstraction - every time a new observation sequence is available) [1]. A GHBDN, once trained, is able both to

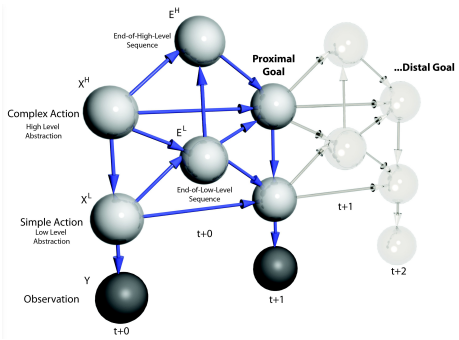

Fig. 1. Growing Hierarchical Dynamic Bayesian Network

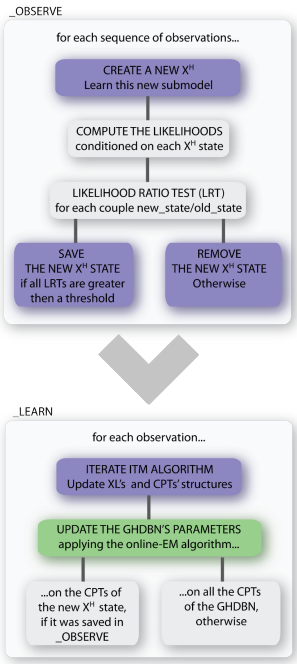

Fig. 2. Learning the GHDBN model

recognize skills being observed, and to reproduce them by exploiting the generative power of the model. In addition, we will show how the model encodes the *proximal goal* of an action together with the global intention of a motor act, e.g. its *distal goal*.

The rest of the paper is structured as follows: section II shows how complex actions can be represented and learned by using our probabilistic framework; section III shows how to perform imitation by using GHDBNs; in section IV we show some preliminary experiments on representing and reproducing tasks in simulation; finally, section V outlines the conclusions and future works.

II. Representing complex actions

An efficient representation of actions is a typical problem arising in imitation learning settings. This section provides an overview of our approach to action representation, which is the result of merging two existing probabilistic graphical models: Growing Hidden Markov Models[1] [10] and Hierarchical Dynamic Bayesian Network[2] [3], resulting in a unique model: *Growing Hierarchical Dynamic Bayesian Network (GHDBN)* [1]. Learning in the model, at different layers of abstraction, is performed incrementally without the need of a priori labeled data.

A. GHDBN model

Dynamic Bayesian Network is a probabilistic graphical model used in representing time series which depend on certain hidden variables. In general, a probabilistic graphical model is identified by its structure and its parameters. The former characterizes the number of variables of the networks, their domain, and their conditional dependencies, while the latter quantifies their conditional probability distributions.

The GHDBN (Figure 1) is a two-level HDBN where each level is described by: two discrete stochastic variables, namely X^H (high level representation of the skill/action) and X^L (low-level behavior); two binary variables, E^H and E^L, representing the end-of-states-sequence markers for each level; and Y, which models a multivariate Gaussian distribution from which are drawn the observations. The peculiarity of this model is that the number of states of X^H and X^L and their state transition structures are not constant: complex actions and elementary behaviors are learned incrementally without any prior information.

B. Learning the model

Representing a process with a probabilistic graphical model means learning its parameters and/or its structure. In the GHDBN, both structure and parameters are learned online in an unsupervised fashion, by processing the continuous flow of information.

Structure learning includes several problems related to graph construction and to definition of the number of states and connections between them. Within the GHDBN, the graph is fixed in terms of number of stochastic variables, their domains and interconnections. Our goal is to find an unsupervised method to learn the number of states (i.e. behaviors at both levels of abstraction) and their connections. The problem is challenging since we need to learn the structure of both X^H's and X^L's states, that is, to discover/modify/remove complex behaviors (states of X^H) and possible transitions between them, and discover/modify/remove low-level events (elementary behaviors, or states of X^L) and possible transitions between them.

[1]A GHMM is a HMM whose hidden discrete variable depends on a topological representation of the feature space. Its parameters are updated by using an Incremental version of the Expectation-Maximization algorithm where inference is performed by a modified Forward-Backward operator. Unfortunately, using this approach involves processing all the past informations when observing a new example which may result in severe memory issues with the growing number of observation. Moreover, it is extremely difficult to model complex multi-level stochastic processes with a unique hidden variable of the GHMM.

[2]A HDBN is a DBN whose variables are grouped into levels. Each level is composed of two types of variables: a variable X_t, representing the state of the system (at that level of abstraction) at time t, and a binary valued variable E_t, representing whether or not the sequence of states of its level has run until its end.

Starting with the structure of the low-level variable X^L, we claim that it should reflect the spatial structure of the feature space discretization. In other words, transitions between basic behaviors are only allowed if the corresponding nodes of a topological map over the feature space are connected. Actually, the observation space is clustered incrementally by using the Instantaneous Topological Map algorithm [2], [10], which provides a discrete representation of the continuous feature space [3] in the form of a graph where feature space regions are represented as nodes and adjacent regions are connected by edges. The topological map is updated at every new observation O_t in order to minimize the number of nodes and try to keep the same average distance between neighbors. The overall complexity is linear in time and memory with respect to the number of nodes.

Each variable is associated with a probability distribution conditioned by the variable's parents. As the hidden variables are discrete, these distributions are represented as N-dimensional tables (named *conditional probability tables, CPT*), where N is the number of states of a variable. In a dynamic model, the parent of a variable could be another variable in the same time slice, or the same variable in the previous time slice (*transition probability table*). In the first time step, this CPT represents the *prior probability table* (e.g. the probability that an action starts with a particular behavior). Finally, the *end-of-sequence probability* distributions represent the probabilities of terminating a high-level or low-level sequence of actions.

Each topological update, made by the ITM algorithm, corresponds to an update of the structures of the conditional probability tables. Adding (removing) a node of the topological map causes increasing (decreasing) the number of X_L states, while adding (removing) an edge causes setting a not null (null) value in the corresponding element of the conditional probability table associated with the variable X^L (where a null value corresponds to an impossible transition between the considered states). In general, constraining the possibility of state transition on the neighborhood between regions of the feature space is not always a correct assumption. In action recognition, however, we are often measuring features (which somehow depend on some physical laws) in a continuous domain. Passing from a state to another apart in the map should be impossible, as changes in the observation are usually gradual (if analyzed in a short time window) and correspond to a series of state transitions.

The structural learning of X^H is more complicated as this variable is not directly observed. The GHDBN could be imagined as many GHMM (as the number of X^H states) unified in a single structure. Having a sequence of observations, it's possible to calculate a likelihood score for each of the submodels to determine the most probable high level state which has generated the observations. The solution used here is based on measuring the *Likelihood Ratio Test*, between the likelihood of the sequence conditioned on a new X^H state (representing the current sequence of observations) and the likelihoods conditioned on the others states. If this score is below a threshold, we are probably observing a complex action which has still not been modeled, so a new X^H state will be added and its relative parameters will be learned[4]. Moreover, the scores for all X^H states are compared to each other. If the scores of two states differ by a quantity smaller than a specific threshold, probably these states represent a similar complex action so they could be merged.

Once the structure of the model has been learned, the second problem is how to learn the entries of the various CPTs which will be used as the transition model. The classical approach for parameters learning in probabilistic graphical models is the Expectation-Maximization (EM) algorithm. Several variants of the EM exist. Here we classify an EM algorithm as *batch* or *online*, emphasizing how the whole set of observations are computed to learn the model. EM algorithms, basically, iterate for maximizing a function. Batch-EM considers the whole set of observations in each iteration, so it requires the initial knowledge of the whole set of observations (we should collect the data of all the actions we have to model before starting the learning phase). Although Batch-EM ensures convergence at least to a local maximum of the observed data likelihood function, it results in slow learning; moreover, the learned model does not adapt, hence making it inapplicable in real-world applications.

We need a purely online EM algorithm which updates the parameters by processing only the current observation. Several purely online EM algorithms exist, but the majority encompasses both the stochastic gradient algorithm and the EM algorithm. An example, used for parameters estimation in Bayesian Networks, is the EM(η) algorithm [11], later improved by the Voting-EM [13]. A similar version has been implemented for DBNs (see [12]).

Let X_i be a node of a generic DBN, Pa_i the set of the parents of X_i. An entry in the CPT of the variable X_i is

$$\theta_{ijk} = P(X_i = x_i^k | Pa_i = pa_i^j) \tag{1}$$

So, the updating rule is:

$$\theta_{ijk}^T = (1-\eta)\theta_{ijk}^{T-1} + \eta[\frac{P(x_i^k, pa_i^j | y_t, \theta_{ijk}^{T-1})}{P(pa_i^j | y_t, \theta_{ijk}^{T-1})}] \tag{2}$$

Such online updating rule referred to stochastic learning, with $\frac{P(x_i^k, pa_i^j | y_t, \theta^{T-1})}{P(pa_i^j | y_t, \theta^{T-1})}$ be the instantaneous gradient estimate of the optimization problem constrained by $\sum_k \theta_{ijk} = 1$.

Since our target is learning complex processes in real time, a recursive filter approach is needed as inferential algorithm so that received data can be processed sequentially. The technique used here is the well known Particle Filter (PF)

[3] As an example, in a monodimensional feature space representing the velocity, the basic behaviors (nodes) could be: *still* (null value), *walking* (medium value), *running* (high value).

[4] Other two methods have been tested: the first comparing the Bayesian Information Criterion score conditioned on each X^H states, the second comparing the LRT with a specific percentile of the χ^2 distribution. Let λ be the Likelihood Ratio between two models; $\hat{\chi}^2 = -2log(\lambda)$ approximates the χ^2 distribution (function of k degrees of freedom, equals to the difference between the number of parameters of the two models). If $\hat{\chi}^2$ is greater than χ^2, so we are probably observing a new complex process.

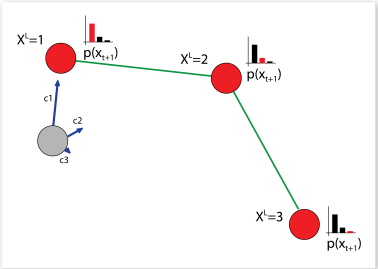

Fig. 3. Imitation in GHDBNs. The current X^L probability distribution is used to generate the motor command to be applied to the robot. Red circles represent ITM nodes where edges specify their connections. Gray circle is the observation extracted from the current scene configuration. Probability distributions depicted next to the nodes represent the current X^L probability distribution. In this example, node #1 is the most probable one; the observation will first reach node #1, then will move towards the node #2 (following the transition CPT), ...

algorithm [14]. PFs are sequential Monte Carlo algorithms used to represent probability densities with point masses (or *particles*) in state-space models. Online EM algorithms exhibit asymptotic behavior towards the real parameter below a certain variance. This variance is proportional to η, thus the estimate will oscillate around the true parameter. η can be viewed as a *forgetting bias* of the learning algorithm. The system forgets the past at an exponential rate proportional to η. However, the oscillation of the parameter permits to get out from local maxima when the environment changes and the model needs to adapt again its parameters.

The learning algorithm for GHDBN is a mix of the previous procedures; figure 2 shows a schematic view of the whole learning algorithm. Note that ITM starts with 2 random nodes, X^H is initialized to have one state (the first action known is the initial observation sequence) and the X^L states reflect the topological map. We suppose that an observation sequence (in the learning process) is complete (i.e. it represents a complex event from the beginning to the end).

III. Imitation

In this section we will show how actions can be reproduced by using the GHDBN framework presented before. Our approach reflects the goal-level imitation paradigm: reproducing a complex action means either reproducing the sequence of sub-goals needed to achieve the high-level goal, or the sequence of high-level goals implicit in the complex action. At each discrete time step, the learner has a set of hierarchically distributed goals: a high-level goal (part of a sequence of complex actions) and a low-level goal (part of a sequence of simpler action conditioned to the current most probable high-level goal).

For imitation to take place, learned processes must be mapped onto the appropriate motor system (i.e. this is the well-known correspondence problem). Imitation process is initialized with an initial position of the robot, with features extracted and projected into the learned Instantaneous Topological Map. The current state distribution is sampled from the prior distributions X^H and X^L by using the particle filter algorithm (weights are computed by using the current observation). The first complex action to be performed is selected as the most probable state in the current X^H states distribution.

The current X^L probability distribution is used to generate the motor command to be applied to the robot. For each state of X^L, a control (with the intensity proportional to the probability of that X^L state, see figure 3) is generated to move the current observation towards the node of the ITM which corresponds to that X^L state. The overall control is given by the linear combination of the controls relatives to all X^L states. We remark that the observation space is usually different from the configuration space of the robot, meaning that the desired movement in the ITM space has to be translated into an appropriate motor command (see section IV-B for experimental results).

The whole imitation algorithm is shown in Figure 4. Imitation is performed by iterating three steps: sampling (after the initial time step, the proposal distribution is given by the transition probability tables conditioned to the most probable X^H state), control generation and observation update (extracting features from the new position in the robot's configuration space). The process ends where the end-of-X^H-sequence distribution is triggered by a termination state. This ensures that the state of the system starts from the prior distribution of the GHDBN and follows its transition probability, actually reproducing the whole modeled sequence of low-level states conditioned on the most probable X^H state.

The proposed framework is able to learn rapidly, and can reproduce novel behavior without any reprogramming and without manually labeling a desired motor control for a given behavior. GHDBN is updated when novel actions are discovered, and it can be used in directly linking the low-level state space with the feature space, which corresponds to a particular configuration in the motor commands of the robot. Moving into the low-level state space means changing controls accordingly. As stated before, high-level behaviors can be reproduced as a sequence of low-level behaviors. GHDBN provides robots with a method to produce motor controls following a high level plan, that is, translating a high-level state sequence into a sequence of low-level states, where each state corresponds to a change in the motor control.

IV. System at work

In this section we will show the performance of our framework on both learning and representing the behaviors, and in an imitation task.

A. Learning and representing actions

In this experiment, goal directed actions are performed by a human, where the experimental setup (observed from an external camera) is a table with several objects that can be manipulated. Several state-of-the art machine vision algorithms have been adopted in order to extract a 3-dimensional

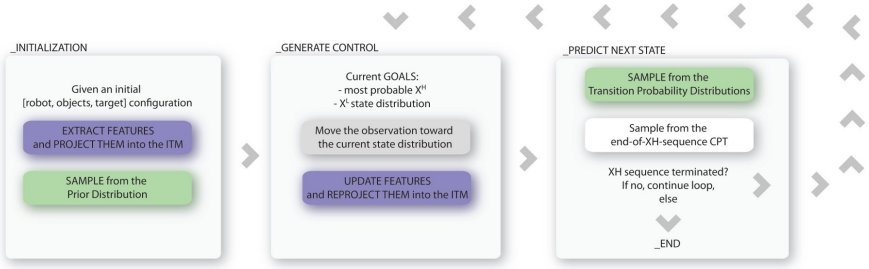

Fig. 4. Imitation algorithm

feature vector represented by the following signals: *hand velocity*, *hand-to-object distance variation*, and *hand-to-object angle variation*. The latter represents the angle between the direction of the hand motion and the hand-to-object direction [5].

As an example, figure 5 shows the temporal evolution of the probability distribution related to the X^H variable (the model has previously been trained on three simple actions, `approach-object`, `grasp-and-dislocate-object`, and `hit-object`). The whole sequence of actions represents a complex activity.

The performances of the GHDBN model have been tested with observations taken from 24 videos containing various sequences of complex actions, for a total of 1655 frames (observations) analyzed. Each frame is classified as the observation of a particular action, if the corresponding state in the current probability distribution of X^H is the most probable. Table I represents a confusion matrix, which shows, in percentages, the guessing right of the model. In particular, 67.58% of the observations concerning the action `dislocate` have been classified, correctly, as part of a `dislocate` action; instead, the model has classified, erroneously, the 5.80% of actions as `approach` and the 26,62% of actions as `hit`. This GHDBN has been trained sequentially with observations classified as (chronologically) a `dislocate` action, an `approach` action and a `hit` action. Note that the `hit` action has a grater recognition rate since it has been the last observed sequence. In order to justify this, recall that the learning rate η of the online EM algorithm introduces a forgetting bias. For major details and the ROC analysis of the model the reader should consult [1].

TABLE I
CONFUSION MATRIX PERCENTAGE

		Actual		
		Dislocate	Approach	Hit
Predicted	Dislocate	67.58%	22.26%	12.07%
	Approach	5.80%	76.26%	6.90%
	Hit	26.62%	1.48%	81.03%

[5]Experiments are implemented on a Intel Core2Duo 2.53GHz with 4Gb RAM under Ubuntu 8.10. Image processing is done in real-time with a framerate of 25fps.

B. Imitation

We have tested our probabilistic framework in reproducing learned behaviors by exploiting its generative abilities. In order to apply the model in a goal-level imitation setting, we have expanded our feature space with information containing a possible "target position" of a `displacement` action. Our modified feature space includes the following signals: *normalized hand-target distance*, *angle between hand-motion and hand-target direction* and the binary *hand-object contact* trigger.

Imitation starts with a randomly generated configuration of robot's hand, object and target position. In this experiment, neither predefined attractors, nor potential fields have been used to move the hand or the object to their target positions. In addition, no information is provided about which element of the scene (object or target goal) is the target of the current action. The target of the current high-level action is automatically recognized by performing inferences in the GHDBN: for example, in a scene with an object, a target position and an initial hand position, we can infer which is the most probable X^H action for each couple hand-object or hand-target by predicting the future states and selecting those with the highest likelihood. For example, when the imitation starts with an empty hand, one can guess that the next high-level action to be performed is *approach* (because it is the only action which starts with the null "hand-object contact" signal), and that the destination of the action should be the object (because this will trigger the signal "hand-object contact" to 1) instead of the target position (because no hand-object contact could be observed in order to terminate the action).

Once the most probable X^H action and its target are selected, low-level controls are generated by using the algorithm described in the section III. Here, for simplicity, controls are reduced to applying a velocity and a direction to the robot's hand in order to reproduce the desired movement in the feature space. Figure 6 (left) shows a sequence of frames and the probability distribution of the X^H variable in reproducing an approach-and-dislocate action. The intention (not known a priori, but inferred) to move the object toward the target position has been reproduced correctly.

V. CONCLUSIONS

We have presented GHDBN, an adaptive probabilistic framework for learning and reproducing complex behav-

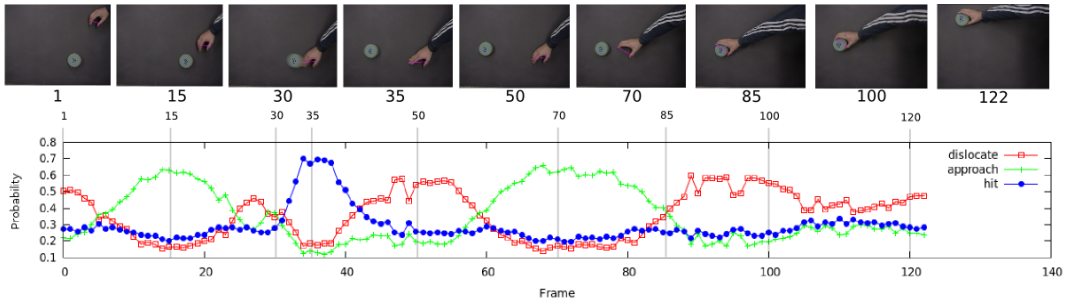

Fig. 5. An example of observing a sequence of complex actions. The second row depicts the probability distribution of X^H for each discrete time step. For instance, between frames ♯85 and ♯120, the most probable action is *dislocate* since the signal *hand-to-object distance variation* is constant (which is the main characteristic of the action *dislocate*). However, *dislocate* is erroneously recognized as the most probable action also between frames ♯20-♯30 and ♯40-♯60 (since the hand is in an idle state, and the *hand-to-object distance variation* is constant). To prevent this it would suffice to train a *dummy* state corresponding to the no-motion condition in the world.

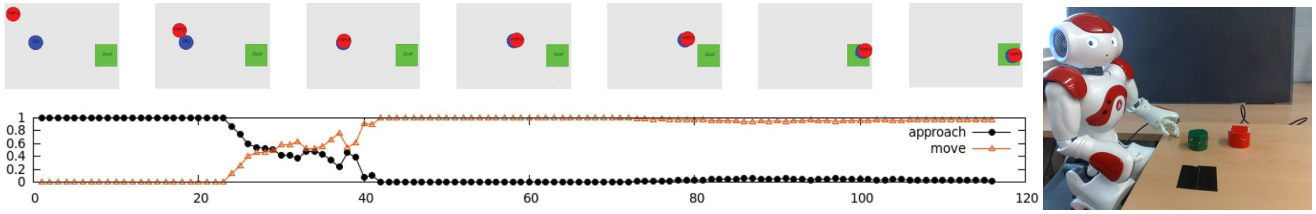

Fig. 6. (Left) Imitating a sequence of high-level actions. The upper row represents the evolution of the robot's "mental states" in simulating the action reproduction. Red circle represents the robot's hand, blue circle the object, and green square is the target location where the object should be placed. The signals in the bottom row represent, for each time step, the probability of reproducing an *approach* or *dislocate* action. (Right) Our current experimental setting involves the humanoid robot NAO.

iors. Algorithms for performing inferences, and learning the structure and the parameters from the data are presented, together with the experimental results in using GHDBN for modeling and recognizing complex actions from data, and in reproducing intentions of actions.

Experimental results demonstrate the ability of the GHDBN model to adapt itself to novel observations, its flexibility over similar models, and provide a method to reproduce complex behavior by recognizing the target of an action and predicting its proximal and distal goals.

Our current work is focused on porting the system from the simulated environment to the NAO humanoid robotic platform[6] (see Figure 6 (right)). Future works will include the adoption of Rao-Blackwellised Particle Filter (see [15]), inclusion of an adaptive learning rate η, and testing the model in other real-world sequential learning tasks.

ACKNOWLEDGMENTS

This work has been partially supported by the EU funded project HUMANOBS: Humanoids That Learn Socio-Communicative Skills Through Observation, contract no. FP7-STREP-231453 (www.humanobs.org).

REFERENCES

[1] H. Dindo, G. Schillaci, *An Adaptive Probabilistic Graphical Model for Representing Skills in PbD Settings*, in 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI 2010), Osaka, Japan, March 2-5, 2010

[2] J. Jockusch and H. Ritter. *An instantaneous topological mapping model for correlated stimuli*, In Proc. of the International Joint Conference on Neural Networks, Washington (US), vol. 1, pp. 529-534, 1999.

[3] K. P. Murphy. *Dynamic bayesian networks: representation, inference and learning*, University of California, Berkeley, 2002.

[4] Yiannis Demiris. *Prediction of intent in robotics and multi-agent systems*, Cognitive Processing, Vol. 8, No. 3., pp. 151-158, Cognitive Processing, September 2007.

[5] D. M. Wolpert, K. Doya, M. Kawato. *A unifying computational framework for motor control and social interaction*. Philos Trans R Soc Lond B Biol Sci 358:593602, 2003.

[6] A. Billard, S. Calinon, R. Dillmann, S. Schaal. *Robot Programming By Demonstration*. Handbook of Robotics, 2008.

[7] G. Rizzolatti, L. Craighero *The Mirror-Neuron System* Annu. Rev. Neurosci. 27, 169-192, 2004.

[8] M. Brass, R. Schmitt, S. Spengler, G. Gergely, *Investigating Action Understanding: Inferential Processes versus Action Simulation*. Current Biology, n.17, 2117-2121, 2007.

[9] J. M. Kilner, C. D. Frith, *Action Observation: Inferring Intentions without Mirror Neurons*. Current Biology, n.18, 2008.

[10] Vasquez, Fraichard, Laugier, *Incremental learning of statistical motion patterns with growing hidden markov models*, Transactions on intelligent transportation systems, 2007.

[11] E. Bauer, D. Koller, and Y. Singer, *Update rules for parameter estimation in bayesian networks*, in Uncertainty in Artificial Intelligence (UAI), pages 313, 1997.

[12] H. C. Cho and S. M. Fadali, *Online estimation of dynamic bayesian network parameter*, in International Joint Conference on Neural Networks, Vancouver, Canada, 2006.

[13] I. Cohen, A. Bronstein, and F. G. Cozman, *Adaptive online learning of bayesian network parameters*, University of Sao Paulo, Brasil, 2001.

[14] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, *A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking*, IEEE Transactions on signal processing, Vol. 50, N. 2, 2002.

[15] A. Doucet, N. de Freitas, K. P. Murphy, S. J. Russell, *Rao-blackwellised particle filtering for dynamic bayesian networks*, in Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence, pages 176-183, San Francisco, USA, 2000

[6] http://www.aldebaran-robotics.com