Learning the Combinatorial Structure of Demonstrated Behaviors with Inverse Feedback Control

Olivier Mangin[1,2] and Pierre-Yves Oudeyer[1]

[1] Flowers Team, INRIA, France
[2] Université Bordeaux 1, France

Abstract. In many applications, such as virtual agents or humanoid robots, it is difficult to represent complex human behaviors and the full range of skills necessary to achieve them. Real life human behaviors are often the combination of several parts and never reproduced in the exact same way. In this work we introduce a new algorithm that is able to learn behaviors by assuming that the observed complex motions can be represented in a smaller dictionary of concurrent tasks. We present an optimization formalism and show how we can learn simultaneously the dictionary and the mixture coefficients that represent each demonstration. We present results on a idealized model where a set of potential functions represents human objectives or preferences for achieving a task.

1 Introduction

Robots are expected to have promising applications in fields such as domestic assistance, health care or education. However bringing robots to our everyday environment and improving their interaction capabilities requires that they are capable of understanding natural human behaviors.

Human activities are numerous and highly diverse, and feature large variability between individuals, situations, and times. Making robots or intelligent systems capable to recognize or even understand or reproduce such behaviors, thus requires a high level of adaptivity which makes learning algorithms promising candidates for this task.

It is however still a difficult problem to design or adapt learning algorithms so that they can deal well with essential properties of natural human behaviors. In fact natural human behaviors are complex and one won't generally observe something as "fill a glass of water" but rather "grasp a glass, walk to the tap, open the tap while keeping the glass straight". Being able to cope with the combinatorial structure of behaviors is thus necessary for their understanding.

In both examples each primitive behavior must be separated from the other behaviors composing the general activity and the relevant features must be identified as the glass being filled, not as the exact trajectory of the elbow or the position of the glass. These two difficulties are actually related to wider topics of research from which efficient algorithms and representations can benefit

A.A. Salah et al. (Eds.): HBU 2012, LNCS 7559, pp. 134–147, 2012.

human behavior understanding by leveraging compositional structure of human activities and represent tasks or objectives that drive the activities.

First separating complex behaviors into simpler parts is very close to both the decomposition of complex motions into simpler motor primitives and dictionary learning techniques from machine learning.

Then, focusing on representations of behaviors in terms of the cost function they are optimizing rather than the specific way to solve it is closely related to inverse feedback control and inverse reinforcement learning approaches which can lead to better generalization properties, as for example when learning to imitate.

In this article we address aspects of the issues of representing, learning and reproducing human behaviors and their compositional structure. We introduce a dictionary learning approach for representing and reproducing the combinatorial structure of motor behaviors that are only observed through demonstrations of several concurrent motor behaviors. We focus on motor behavior representations that directly model the objective of the user underlying demonstrations. We illustrate the presented algorithm on a simple toy example.

2 Background and Related Work

2.1 Decomposition of Motor Skills: Motor Primitives

Motor primitives have been introduced as a form of re-usable motor skills that may be used as elementary building blocks for more complex motor control and skills. The concept of motor primitives that can be combined together has the appealing property to enable combinatorial growth of the skill repertoire. As detailed by Konczak [1], examples of motor primitives can be found both in biological and robotic systems, and can be either innate or acquired.

The notion of combination of motor primitives can take different forms. One could consider a behavior composed of a **sequence** of simple actions, like moving one's hand to a glass, grasping it, bringing it back to one's mouth, etc.

The structure of some behaviors however does not fit well in this sequential representation. Many behaviors or tasks are better described in terms of elementary movements executed **simultaneously** (e.g. on different parts of the body) or **concurrently**, like speaking while smiling and shaking someone's hand. Concurrent combinations of behaviors is particularly studied in this article.

2.2 Using HMMs to Learn Motor Primitives

Hidden Markov models (HMM), often coupled with clustering techniques or mixture models, have been largely used to learn sequences of primitives. For example, Kulic and Nakamura have proposed in [2] a method that first performs an unsupervised segmentation of the motion signal into small successive blocks (the segmentation technique itself is based on HMMs), and then performs clustering over HMM representations of each segmented block. Each group of similar motions is interpreted as a motor primitive.

In a different setting, Kruger et al. [3], have focused on a notion of motor primitive based on the effect of actions on objects from the environment. They have proposed to first discover primitives by clustering action effects on manipulated objects and then use the found clusters, composed of actions that have similar effects, to segment the stream of motions into coherent actions. Then parametrized hidden Markov models are trained to represent the actions and enable both their recognition and reproduction.

Finally Calinon et al. [4] and Butterfield et al. [5] use Gaussian mixture models to represent motion primitives and HMMs to discover and represent the transitions and sequential combinations of primitives. All the approaches presented in this paragraph are capable of recognizing and reproducing the learned motions.

2.3 Using Dictionary Learning to Learn Motor Primitives

Dictionary learning approaches by matrix factorization are machine learning techniques widely used to solve problems where an input signal has to be decomposed into a linear combination of atoms. They target the learning of both the dictionary of atoms and the coefficients used in their combinations.

The possibility to enforce structural constraints on the dictionary and coefficient matrices enables better modeling of many problems and participates in the versatility of dictionary learning techniques. Such constraints include for example non-negativity [6,7], sparsity or group sparsity [8], constraining atoms to be convex combinations of the demonstrations, which can be seen as a generalization of clustering [9], constraining atoms to be stochastic vectors, etc.

In the field of motion decomposition, Li et al. [10] have used orthogonal matching pursuit to decompose complex motions into simple motion patterns activated shortly along time. The decomposition is used to perform both compression, classification and reproduction of visualizations of the movement (but is not tested on real reproduction). The article uses constraints such as sparse activation coefficients and sparse motion patterns in Fourier domain.

Hellbach et al. [11] have also used non-negative matrix factorization to perform a decomposition of globally unstructured motions in low level components. They use time invariance and sparsity of dictionary atoms to guide the learning toward discovering short sequences of positions that can be concatenated into the observed trajectory. These capabilities are tested on a dataset of real movements for prediction but not to produce motion on a real robot.

Time sequences of motor primitives learnt by methods from Li et al. [10] and Hellbach et al. [11] may include overlap, and can therefore be considered as hybrid methods enabling the learning of motor primitives combined both in sequence and parallel. They are however mainly focused on representing trajectories by superposition of time shifted simple local patterns and do not explore how the structure of complex behaviors composed of simultaneous primitive gestures can be leveraged towards better understanding of the observed activity.

In our previous work [12] we demonstrated how non-negative matrix factorization can be used to decompose complex behaviors into simultaneous combinations of primitive gestures. We presented an experiment in which dance

choreographies are demonstrated by a human. Each choreography is composed of several simultaneous gestures. For example, one leg gesture and one gesture on each arm. A set of symbolic linguistic labels corresponding to the gestures occurring in the choreography are also provided with the demonstrations, which is a form of linguistic guidance. A learning system is trained by observing both demonstrations of the choreography and the associated labels. The system then observes new dances and has to reconstruct the associated set of labels, that is to say, tell which gestures were combined to form the choreography. It is shown in the article that the system performs well even if the demonstrated choreography is a combination of gestures that have never been demonstrated together during training. This setting emphasizes the ability of the system to capture the compositional structure of the choreographies.

[12] presents a technique that permits classification of complex behaviors, but it cannot reproduce them since the motion representation is only discriminative. This article presents a dictionary learning approach based on inverse feedback control, which as a generative representation enables motion reproduction.

2.4 Inverse Feedback Control

Approaches which consist in direct representation and reproduction of the policy (state to action mapping) observed through trajectories of the demonstrator's (or imitator's) body are often denoted as **policy learning**. Most techniques presented in Sections 2.2 and 2.3 belongs to this category. The policy can either be a direct representation of the trajectory [13] or a probabilistic model of the policy [4].

In opposition, **inverse optimal control** [14] and **inverse reinforcement learning** [15] are approaches based on the idea that, in some situations, it can lead to better generalization to model aspects of the task that the demonstrator is trying to solve instead of modeling the particular solution in the demonstrated context. The capacity of inverse optimal control to achieve better generalization has been demonstrated in the experiment performed by Abbeel et al. [16], in which an helicopter performs acrobatic motions after observing demonstrations from a human expert remotely controlling the helicopter. In that example the learned trajectories even overtake the skills of the demonstrating expert.

Jetchev and Toussaint [17] have adapted inverse optimal control techniques to a single grasping task on a real robot. Furthermore they have shown how the inverse optimal control approach, coupled with a sparsity constraint on the task representation can be used to discover relevant features in the task space.

Finally Brillinger [18] has developed an algorithm based on least square regression to learn potential functions modeling the motion of wild animals in natural parks.

In this article we extend Brillinger's technique to address a different problem: instead of learning a flat representation of a single task, the learner must infer several primitives cost functions/skills that can be composed to explain the mixing of concurrent tasks that are demonstrated. We use a very similar behavior representation, but introduce dictionary learning for solving the new problem.

3 Problem Definition and Algorithm

We introduce a simple synthetic imitation learning experiment in which an imitator learns to reproduce behaviors observed from a demonstrator.

More precisely we model the task underlying each behavior as a cost function on states of the agent (either the demonstrator or the imitator), which can be seen as representing the preferences of the demonstrator. For example the task of filling a glass of water will be represented by a cost function giving increasing values to increasing levels of water in the glass. In the case where the "filling the glass" behavior is mixed with the "smiling to someone" behavior, the mixed behavior will be represented by a mixed cost function valuing both full glass and smiling position of the lips.

Each demonstration consists in a trajectory in the demonstrator state space, from a specific initial position. The objective of the imitator is to produce a trajectory (either from the same initial position than the demonstration, or another) that fits the demonstrator preferences (i.e. minimize the cost function).

This setup introduces two important difficulties for the imitator. On the one hand each demonstration only presents aspects of the cost function locally, around the trajectory. Each demonstration is thus not sufficient to fully understand the underlying task. On the other hand, each demonstration presents a mixture of several tasks. Thus, while the primitive tasks are observed many times, they are never observed alone and each particular mixture is generally only observed once. It is thus necessary to leverage the compositional structure of the behaviors to be able to understand them, and reproduce them with new initial positions.

3.1 Agent and Demonstrator Models

We will assume that both the demonstrator and imitator are identical. This corresponds for example to the case where demonstrations are performed on the imitator body (kinesthetic demonstrations). Following Jetchev et al. [17], we consider a robotic agent which configurations q belong to a state space $\mathcal{Q} \in \mathbb{R}^S$. Each trajectory is denoted by a sequence $(q_t)_{t \in [|1,T|]}$.

We assume that there exists a cost function $f : \mathcal{Q} \longrightarrow \mathbb{R}$ such that each task is modeled as the demonstrating agent trying to minimize the cost $f(q)$ to which is added a penalization on the square norm of $\frac{\partial q}{\partial t}$, which can be seen as a penalization of the energy consumed while moving to optimize $f(q)$.

We will focus on very simple agents which actions are motions in the state space and are governed by the local optimization of $f(q) + \alpha \left\| \frac{\partial q}{\partial t} \right\|^2$ which means that each action, at each time step, is chosen such that:

$$q_{t+1} = \underset{q}{\operatorname{argmin}}\ f(q) + \alpha \left\| \frac{q - q_t}{\delta_t} \right\|^2,$$

where δ_t is the time elapsed between samples t and $t+1$.

The solution of this equation, without additional constraints, and assuming that the cost function f is differentiable, is well known to be proportional to the gradient of f, as $-\frac{1}{\alpha}\nabla f(q)$.

It can be noticed that since the agent we have defined only follows policies driven by local optimization it will only achieve local optimization of the cost function. While this is a simplification of the agent, it also features an important property of real demonstrators: real demonstrators are in general imperfect and do not always succeed in reaching the optimal solution of the task. It is thus important for a imitator to be able to also learn from imperfect demonstrations of behaviors.

In this article we focus on complex tasks: each demonstration corresponds to the minimization of a separate cost function f which is only observed through one demonstration. However f is composed of parts that also occur in other demonstrations and are thus observed several time mixed in various way and in various contexts.

Lets consider N demonstrations, observed as trajectories $(q_t^i)_t,\ i \in [|1, N|]$ in the agent state space. We assume that each demonstration corresponds to a given f^i. To model complex demonstrations we assume that there exists a dictionary of primitive tasks, composed of K cost functions $(g^k)_{k\in[|1,K|]}$, such that, for all demonstration i, there exist coefficients $(a_k^i)_{k\in[|1,K|]}$ such that, for all state q, $f^i(q) = \sum_{k=1}^{K} a_k^i g^k(q)$.

We present a learning algorithm which observes one demonstration associated with each function f^i and learns a dictionary of primitive cost functions g^k, and the coefficients of their combinations into demonstrated tasks f^i.

3.2 Inferring a Task from a Demonstration

The problem of inferring a single task from a demonstration is studied in Brillinger's article [18]. The cost function is represented by a linear parameter $\beta \in \mathbb{R}^F$ on a space of potentially non-linear features $\varphi : \mathcal{Q} \longrightarrow \mathbb{R}^F$. Its minimization is modeled by an agent policy such that:

$$\frac{\partial q}{\partial t} = -\lambda \mathbf{J}(q)^T \beta \tag{1}$$

where $\mathbf{J}$ is the Jacobian of φ (lines of $\mathbf{J}$ are gradients of coordinates of φ).

When discrete trajectories are considered, equation (1) approximates into $\frac{q_{t+1} - q_t}{\delta_t} = -\lambda \mathbf{J}(q_t)^T \beta$ for all $t \in [|1, T-1|]$. By denoting $y_{t+1} = \frac{q_{t+1}-q_t}{\delta_t}$, $Y \in \mathcal{R}^{S\times(T-1)}$ the vector obtained by vertically stacking all y_t for $t \in [|2, T|]$, and $\mathbf{\Phi}$ the $S \times (T-1)$ by F matrix obtained by vertically stacking all $-\lambda \mathbf{J}(q_t)^T$, we get:

$$Y = \mathbf{\Phi}\beta \tag{2}$$

Equation (2) transforms the problem of inferring one task from one demonstration into a linear regression problem, which constitutes an essential contribution of Brillinger's article.

In the case where the Euclidean distance between the vector Y, computed from observations, and its reconstruction through the task model $\mathbf{\Phi}\beta$ is considered, we get the classical least square regression problem. It is solved, assuming $\mathbf{\Phi}^T\mathbf{\Phi}$ is non-singular, by:

$$\beta = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T Y \tag{3}$$

More details on the associated derivations can be found in [18]. The algorithm presented above is capable, from one demonstration, to infer the cost function modeling a behavior of the demonstrator. Once the cost function is inferred, the imitator can in turn produce trajectories that minimize it. Such an agent that directly infers all the parameters of the cost function is denoted **flat imitator** in the following.

3.3 Learning a Dictionary of Primitive Tasks from Mixed Demonstrations

The algorithm presented in Section 3.2 only applies to a single demonstration generated from a single task model. In this section we introduce a matrix factorization algorithm to learn a dictionary of primitive tasks and associated coefficients from several demonstrations.

Each demonstration corresponds to a mixing of primitive tasks which is modeled by a β^i in the feature space. To model the concurrent mixing of primitive tasks, we introduce a dictionary represented by a F by K matrix $\mathbf{D}$ such that each column of $\mathbf{D}$ is the parameter representing the primitive tasks g^k in the feature space. The concurrency between the primitive tasks in a mixing is represented through a weighting coefficient. Coefficients of the i^{th} demonstrated task are given by a vector $a^i \in \mathbb{R}^K$, $\beta^i = \mathbf{D}a^i$.

For each demonstration we define the vector Y^i and the matrix $\mathbf{\Phi}^i$ associated with the observed trajectory, by following the method described in Section 3.2. It follows that for each demonstration:

$$Y^i = \mathbf{\Phi}^i\mathbf{D}a^i \tag{4}$$

Learning a factored model of the demonstrated tasks that minimize Euclidean distance to demonstration is equivalent to solving equation (5).

$$\underset{\mathbf{D},\mathbf{A}}{\operatorname{argmin}}\ \mathcal{L}(\mathbf{D},\mathbf{A}) \text{ with } \mathcal{L}(\mathbf{D},a) = \sum_{i=1}^{N} \left\|Y^i - \mathbf{\Phi}^i\mathbf{D}a^i\right\|_2^2 \tag{5}$$

We propose an algorithm based on alternate minimization with respect to $\mathbf{D}$ and $\mathbf{A}$ to solve this problem.

Minimization with respect to $\mathbf{A}$ This sub-problem assumes that the dictionary is known and thus consist, from a demonstration, in inferring the task decomposition on the dictionary. It is similar to the algorithm presented in previous section, but the K decomposition coefficients (the vector a) are inferred instead of all the F coefficients of the cost function.

This problem is separable in one sub-problem for each demonstration i which are all equivalent to the regression problem presented in Section 3.2 where the matrix $\mathbf{\Phi}$ is now replaced by the product $\mathbf{\Phi}^i\mathbf{D}$. Thus the solution of the optimization with respect to $\mathbf{A}$ is given, for Euclidean distance, by equation (6). Other norms or penalization could as well be used to solve the regression (e.g. methods enforcing non-negativity or sparseness of coefficients).

$$a^i = (\mathbf{D}^T{\mathbf{\Phi}^i}^T\mathbf{\Phi}^i\mathbf{D})^{-1}\mathbf{D}^T{\mathbf{\Phi}^i}^T Y^i \tag{6}$$

Minimization with respect to $\mathbf{D}$ The second sub-problem assumes that the decomposition coefficients of the demonstrated task are known but not the dictionary $\mathbf{D}$. We use a gradient descent approach to learn $\mathbf{D}$. The differential of the loss with respect to each of the coefficients of $\mathbf{D}$ is given by equation (7).

$$\nabla_{\mathbf{D}}\mathcal{L}(\mathbf{D},\mathbf{A}) = -2\sum_{i=1}^{N}{\mathbf{\Phi}^i}^T\left[Y^i - \mathbf{\Phi}^i\mathbf{D}a^i\right]{a^i}^T \tag{7}$$

Global algorithm The global algorithm simultaneously learns the dictionary $\mathbf{D}$ and the coefficients $\mathbf{A}$ by alternation of the two procedures from previous paragraphs. Matrices $\mathbf{D}$ and $\mathbf{A}$ are initiated randomly or according to any heuristic. Then $\mathbf{D}$ is learnt, assuming $\mathbf{A}$ contains the correct decomposition coefficients, after which $\mathbf{A}$ is inferred assuming $\mathbf{D}$ is the correct dictionary, and so on. This approach to matrix factorization problems has often proved to be efficient ([7,8]).

4 Experiments

To illustrate the algorithm introduced in Section 3 we consider a simple toy experiment. We define an agent which state q belongs to $\mathcal{Q} = [0,1]^2$. Cost functions are parametrized on a 5 by 5 grid of Gaussian radial basis functions , which means $\phi(q)^T = (\dots, \frac{1}{2\pi\sigma}\exp(-\frac{\|x-\mu_f\|^2}{2\sigma^2}), \dots)$ where μ_f are points from a regular 5 by 5 grid on $\mathcal{Q}$ and σ is fixed such that the task parameter space is of dimension $F = 25$.

We use in this experiment a dictionary of 6 primitive tasks that is represented in Figure 1 (first row). Combinations of 2 or 3 concurrent primitive tasks are generated randomly for training and testing. For a given mixed tasks, a starting point is randomly chosen inside $\mathcal{Q}$ and trajectories are generated by the demonstrator or imitator from the initial position, according to equation (1). In the remaining of this section we will describe two separate experiments where a dictionary is learnt by a agent observing mixed combinations of tasks.

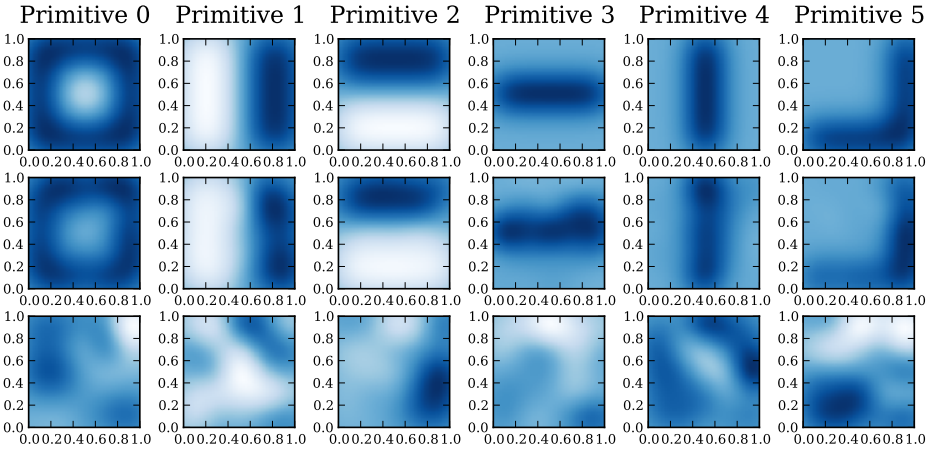

Fig. 1. Dictionary of primitive tasks represented as cost functions over $\mathcal{Q} = [0, 1]^2$. First row corresponds to original primitive tasks (as used by the demonstrator), second row to the one reconstructed by the learner described in Section 4.1 and third row to the learner described in Section 4.2. Dark areas correspond to high positive costs and light areas to negative costs. (Best viewed in color).

4.1 Recovering the Dictionary from Given Coefficients

In this section we consider an experiment in which during training the learner both observes demonstrations of mixed tasks and the associated mixing coefficients. This hypothesis models the situation where some labels associated with the task that are mixed together in the demonstration are given to the learner (e.g. inferred from spoken language). This experiment enables the evaluation of the second part of the algorithm we introduced.

Since the mixing coefficients are known by the learner during training, only the second part of the algorithm presented in Section 3.3 is used to learn the dictionary $\hat{\mathbf{D}}$. We train such a learner on 200 trajectories generated from a dictionary $\mathbf{D}$. Both the original dictionary of primitive tasks $\mathbf{D}$ and its reconstruction $\hat{\mathbf{D}}$ are represented in Figure 1.

Once the imitator has built a dictionary of tasks from observations, it is evaluated in the following way: for a set of coefficients, corresponding to mixed tasks, and a random starting position, the imitator and demonstrator yield trajectories. The demonstrator and imitator trajectories are then compared. Examples of trajectories from both the learner and the imitator are given in figure 2.

The relative L_2 error between the trajectories generated by the demonstrator and the imitator is used to evaluate the quality of the reconstruction. An average error of 0.001127 is obtained on the train set (tasks observed while learning the dictionary) and 0.002675 is obtained on the test set (unobserved tasks).

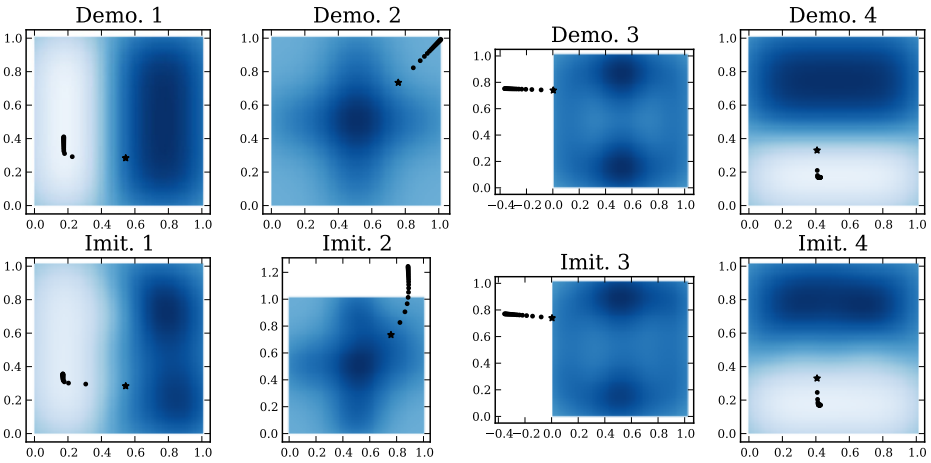

Fig. 2. Examples of demonstration trajectories generated from mixed concurrent primitives tasks (first row) and their reproduction by the learner from experiment one. Initial positions are marked by stars, others position are marked by circles. The associated cost functions (the one inferred in the case of the imitator) are also represented. Dark areas correspond to high positive costs and light areas to negative costs. (Best viewed in color).

4.2 Learning Both Primitive Tasks and Mixing Coefficients from Concurrent Demonstrations

We illustrate the full algorithm presented in Section 3.3 on an experiment where the learner only observes demonstrated trajectories without knowing the coefficients. The learner's reconstructed dictionary is given in Figure 1, bottom row.

Once the dictionary has been learnt, we use the following imitation protocol to test the imitator. A new unobserved combination of primitive tasks is chosen together with an initial position. Then the demonstrator provides a trajectory corresponding to the task. From the observation of the demonstrated trajectory and the learnt dictionary of primitive tasks, the learner infers the task's decomposition on the learnt dictionary (using the first part of the algorithm presented in Section 3.3). Finally the imitator is asked to produce trajectories corresponding to the same task, both from the demonstrator's initial position and randomly chosen initial positions. Changing the initial position from the demonstrated one is a way to evaluate how well the imitator's model of the task generalizes from the demonstration context to new ones.

In order to evaluate the impact of learning the dictionary, that is to say the combinatorial structure of the demonstrated data, we compare reproductions of the task by an agent that has learnt the dictionary denoted as *full dictionary learner*, to ones by an agent, denoted as *flat imitator*, that directly infers the parameters of the tasks without using a dictionary (algorithm presented in Section 3.2). We also compare the agent described in the previous section that

has learnt the dictionary from both demonstrated trajectories and mixed coefficients, denoted *dictionary from coefficients learner*. Examples of demonstrated and imitated trajectories are provided in Figure 3.

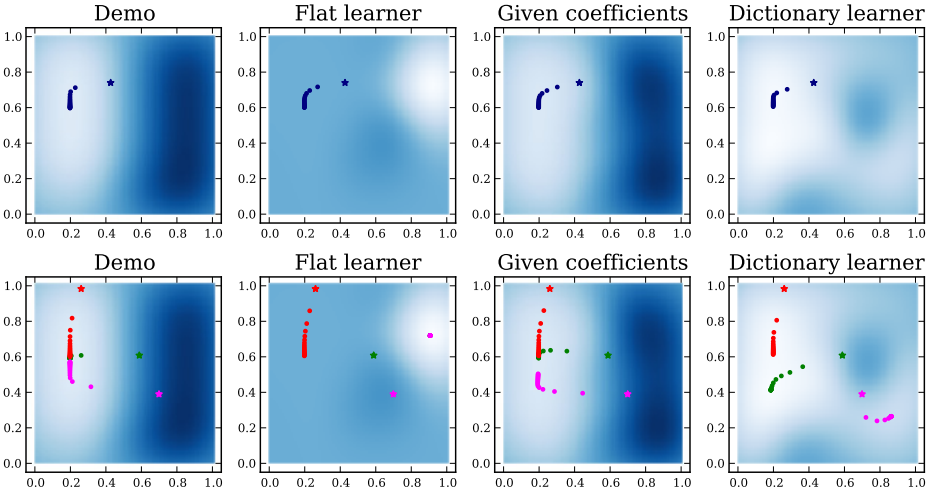

Fig. 3. Examples of imitated trajectories. First row presents the demonstrated trajectory (first column) and its imitation by the flat learner, the dictionary learner from first experiment (coefficients observed while learning the dictionary) and the full dictionary learner. Second row correspond to imitations of the same task from initial positions that were not observed (the demonstrator trajectories for those positions are given for comparison purpose). (Best viewed in color).

5 Discussion

The first agent presented in Section 4.1, is able, by observing motions solving composed tasks and the mixing coefficients, to learn the dictionary of primitive tasks. The acquired dictionary is evaluated in different ways: visually from the plots of the associated cost functions, from trajectories solving a mixed task whose mixing coefficients are given, and from imitation, in random contexts, of a mixed task that is inferred from a single demonstration (this last result is presented together with second experiment).

In our previous work [12], we present an algorithm that learns from mixed behaviors presented together with labels similar to the mixing coefficients. The learner is able to yield the labels from test demonstrations of the motions. Actually the experiment evaluates the algorithm directly on the quality of the estimation of the coefficients, since the system is not able to reproduce the demonstrated gestures. The first agent presented in this article learns in a similar setting than the algorithm from [12] but extends its capabilities to the reproduction of the demonstrated behaviors.

The second agent described in Section 4.2 is capable of learning a dictionary that enables the factorial representation of demonstrated tasks, without directly observing the dictionary or the mixing coefficients. The factorial representation enables imitation of tasks that are observed through a single demonstration. However the performance of the imitator is not evaluated due to the illustrative nature of the experimental setup. In particular the least square regression from [18] (described in Section 3.2) is not performing well on the particular form of mixing of cost functions we have chosen for the illustrative toy example. However our algorithm is compatible with any regression method. Thus, interesting further work could use the comparison of performances between various regression methods, on real human data, to get better insight on the combinatorial properties of human activities.

The dictionary learnt by the agent is very different from the one of the demonstrator. Actually the problem of representing a set of demonstrated mixed tasks as linear combinations of primitive tasks is ill posed and does not have a unique solution. For example one can scale the primitive cost function by some factor and associated coefficients by its inverse or change the order of the primitive and coefficients without changing the linear combination. Mathematically these difficulties could be solved by adding constraints to the form of the learnt dictionary (e.g. normalize primitive costs) or by adapting the way to compare dictionaries (e.g. to make it invariant to re-ordering).

To overcome this difficulty, several ways of making some possible decompositions more salient than others can guide the learning, in the same way humans easily identify salient behaviors even when mixed with others. First, saliency can come from one's history: if one already knows all but one primitive behavior present in the scene, it is possible to identify the unexplained parts of the behavior and learn it as a new primitive. Investigating this part would require to extend the learning model to an incremental learner. The algorithm we presented can be extend to become online following a similar method than [19] although this is not investigated in this article.

Then, a particular form of factorization could also be shaped by information coming from another modality or social interaction. This aspect is demonstrated both in our previous work [12] and in the first experiment (Section 4.1), where observing the mixing coefficients, that can be seen as linguistic labels, enables the learner to adapt its internal model (i.e. the dictionary) to a communication channel. Aspects of social learning have already been shown to improve motor learning by Massera et al. [20]. Solving the ambiguity in the decomposition of human activities thus constitutes a new application for social learning.

Finally intrinsic constraints can be applied to the learnt dictionary to prefer some solutions . Two examples of such constraints for which many machine learning algorithms have been developed are non-negativity and sparsity. Non-negativity of the coefficients will for example focus on representations that allow primitive behaviors to be added to but not subtracted from an activity in which they do not appear. Jetchev et al. [17] have shown how enforcing sparsity of a task representation can make this task focus only on a few salient features, thus

performing task space inference. Other examples are given by Li et al. [10] and Hellbach et al. [11].

Extending the algorithm we presented to include constraints or evaluating it on an online learning experiment would help investigating these questions and thus constitute very interesting future work. For the result to be relevant, the setup would however have to include more realistic aspects, such as non-trivial action to state change mapping or more sophisticated agent models (e.g. capable of planification).

6 Conclusion

In this article we studied aspects of the combinatorial structure of human behaviors and of their representation as tasks or objectives. We introduced an algorithm to learn a dictionary of primitive tasks from demonstrations of concurrently mixed behaviors. We demonstrated on an illustrative experiment how the dictionary can be used to represent and generalize new demonstrations. Finally we discussed how dealing with ambiguities in factorial representation of behaviors might involve social interactions, multimodality of the sensory experience or intrinsic saliency mechanisms.

References

1. Konczak, J.: On the notion of motor primitives in humans and robots. In: Fifth International Workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems, vol. 123, pp. 47–53. Lund University Cognitive Studies (2005)
2. Kulic, D., Nakamura, Y.: Incremental Learning of Human Behaviors using Hierarchical Hidden Markov Models. In: IEEE International Conference on Intelligent Robots and Systems, pp. 4649–4655. IEEE Computer Society Press (2010)
3. Kruger, V., Herzog, D., Baby, S., Ude, A., Kragic, D.: Learning actions from observations. Robotics and Automation Magazine 17(2), 30–43 (2010)
4. Calinon, S., D'Halluin, F., Sauser, E.L., Caldwell, D.G., Billard, A.G.: An approach based on Hidden Markov Model and Gaussian Mixture Regression. IEEE Robotics and Automation Magazine 17(2), 44–54 (2010)
5. Butterfield, J., Osentoski, S., Jay, G., Jenkins, O.C.: Learning from Demonstration using a Multi-valued Function Regressor for Time-series Data. In: International Conference on Humanoid Robots, vol. (10). IEEE Computer Society Press, Nashville (2010)
6. Paatero, P., Tapper, U.: Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. Environmetrics 5(2), 111–126 (1994)
7. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization.. Nature 401(6755), 788–791 (1999)
8. Jenatton, R., Mairal, J., Obozinski, G., Bach, F.: Proximal methods for sparse hierarchical dictionary learning. In: Proceedings of the International Conference on Machine Learning, ICML (2010)
9. Ding, C., Li, T., Jordan, M.I.: Convex and semi-nonnegative matrix factorizations. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(1), 45–55 (2010)

10. Li, Y., Fermuller, C., Aloimonos, Y., Ji, H.: Learning shift-invariant sparse representation of actions. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 2630–2637. IEEE, San-Francisco (2010)
11. Hellbach, S., Eggert, J.P., Körner, E., Gross, H.-M.: Basis Decomposition of Motion Trajectories Using Spatio-temporal NMF. In: Alippi, C., Polycarpou, M., Panayiotou, C., Ellinas, G. (eds.) ICANN 2009, Part II. LNCS, vol. 5769, pp. 804–814. Springer, Heidelberg (2009)
12. Mangin, O., Oudeyer, P.-Y.: Learning to recognize parallel combinations of human motion primitives with linguistic descriptions using non-negative matrix factorization. To Appear in International Conference on Intelligent Robots and Systems (IROS 2012), Vilamoura, Algarve (Portugal), IEEE/RSJ (2012)
13. Schaal, S., Ijspeert, A.J., Billard, A.G.: Computational approaches to motor learning by imitation. Phil. Transactions of the Royal Society of London B: Biological Sciences 358, 537–547 (2003)
14. Ratliff, N.D., Bagnell, J.A., Zinkevich, M.A.: Maximum margin planning. In: International conference on Machine learning, ICML 2006, vol. (23), pp. 729–736. ACM Press, New York (2006)
15. Lopes, M., Melo, F., Montesano, L.: Active Learning for Reward Estimation in Inverse Reinforcement Learning. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009, Part II. LNCS, vol. 5782, pp. 31–46. Springer, Heidelberg (2009)
16. Abbeel, P., Coates, A., Ng, A.Y.: Autonomous Helicopter Aerobatics through Apprenticeship Learning. The International Journal of Robotics Research 29(13), 1608–1639 (2010)
17. Jetchev, N., Toussaint, M.: Task Space Retrieval Using Inverse Feedback Control. In: Getoor, L., Scheffer, T. (eds.) International Conference on Machine Learning, ICML 2011, pp. 449–456. ACM Press, New York (2011)
18. Brillinger, D.R.: Learning a Potential Function From a Trajectory. IEEE Signal Processing Letters 14(11), 867–870 (2007)
19. Lefèvre, A., Bach, F.R., Févotte, C.: Online algorithms for Nonnegative Matrix Factorization with the Itakura-Saito divergence. In: IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), pp. 1–9. IEEE Computer Society Press (2011)
20. Massera, G., Tuci, E., Ferrauto, T., Nolfi, S.: The Facilitatory Role of Linguistic Instructions on Developing Manipulation Skills. IEEE Computational Intelligence Magazine 5(3), 33–42 (2010)