Imitation learning of motor primitives and language bootstrapping in robots

Thomas Cederborg and Pierre-Yves Oudeyer
INRIA, France
http://flowers.inria.fr

Abstract— **Imitation learning in robots, also called programing by demonstration, has made important advances in recent years, allowing humans to teach context dependant motor skills/tasks to robots. We propose to extend the usual contexts investigated to also include acoustic linguistic expressions that might denote a given motor skill, and thus we target joint learning of the motor skills and their potential acoustic linguistic name. In addition to this, a modification of a class of existing algorithms within the imitation learning framework is made so that they can handle the unlabeled demonstration of several tasks/motor primitives without having to inform the imitator of what task is being demonstrated or what the number of tasks are, which is a necessity for language learning, i.e; if one wants to teach naturally an open number of new motor skills together with their acoustic names. Finally, a mechanism for detecting whether or not linguistic input is relevant to the task is also proposed, and our architecture also allows the robot to find the right framing for a given identified motor primitive. With these additions it becomes possible to build an imitator that bridges the gap between imitation learning and language learning by being able to learn linguistic expressions using methods from the imitation learning community. In this sense the imitator can learn a word by guessing whether a certain speech pattern present in the context means that a specific task is to be executed. The imitator is however not assumed to know that speech is relevant and has to figure this out on its own by looking at the demonstrations: indeed, the architecture allows the robot to transparently also learn tasks which should not be triggered by an acoustic word, but for example by the color or position of an object or a gesture made by someone in the environment. To demonstrate this ability to find the relevance of speech, we show experiments where non linguistic tasks are learnt along with linguistic tasks and the imitator has to figure out when speech is relevant (in some tasks speech should be completely ignored and in other tasks the entire policy is determined by speech). This simulated experiment also demonstrates that the imitator can indeed find the number of tasks that has been demonstrated to it, discover what demonstrations are of what task, which framing is associated to which tasks, and for which of the tasks speech is relevant and finally successfully reproduce those tasks when the corresponding context is detected.**

I. INTRODUCTION

We introduce an architecture and algorithms that allow a robotic imitator to observe a set of unlabeled/uncategorized demonstrations and from these learn a set of tasks or motor primitives, each of them defined in its particular representational framing and associated with a context which might or might not include an acoustic linguistic name. Practically, this means that we aim at building a system which allows a human to teach how (which policy dynamics in which representational framing) and when a robot should achieve various complex motor tasks/motor primitives (what aspect of the context should trigger the motor primitive). For example, tasks may include "draw a circle around the object in front of you when you hear the "encircle" word", "stack objects in front of you if there are many and they are spread around", "eat the chocolate if somenone hands one to you", "grasp the object in front of you if you hear the "grasp it" word".

This research is grounded in a field of research known as imitation learning or programming by demonstration, which goal is to learn about the imitation behavior of humans and animals by building models of these abilities or, as in this paper, build useful artifacts whose learning algorithms are inspired by imitation learning abilities in biological organisms. In the approach to imitation learning investigated in this paper a human demonstrator provides a robotic imitator with a set of unlabeled demonstrations describing how to perform an unknown number of tasks/motor primitives. From the demonstrations the imitator infers how many different tasks it has observed, which task is being demonstrated for each of the demonstration, in what parts of the full state space S (i.e. in which context) the task should be executed and, for each task, what parts of the state space are relevant for representing and infering the task policy. The task space S_{t_i} of task nr i is such that the task policy π_i can be fully described as a function from S_{t_i} to motor outputs (we can say that $\pi_i = \pi_i(s_{t_i} \in S_{t_i})$) and is obtained using a framing $s_{t_i} = f_i(s \in S)$, for $s_{t_i} \in S_{t_i}$ (finding the task space thus amounts to finding the task framing f_i). A framing f_i for task i is thus here defined as a mapping which projects the representation of the current full state in the full available representational space S into a typically lower-dimensional space which corresponds to the intrinsic dimensions in which task i is defined (i.e. all left out dimensions are not relevant for defining the policy π_i).

This paper builds on the work presented in [1], which introduced incremental local online gaussian mixture regression (ILO-GMR) as a possible way for achieving incremental learning of new tasks/motor primitives, by considering the set of all tasks/policies as a single large dynamical system which could be updated locally, i.e. being a local alternative

We would like to thank Adrien Baranes for help with the graphical presentation. This research was partially funded by ERC Grant EXPLORERS 240007

of the imitation learning approach presented in [12]. Here, we extend this work in two main ways. One extension is that the imitator builds an explicit representation of the tasks and their properties using a sophisticated grouping algorithm (how many tasks/motor primitives there are and which demonstrations are associated to which task and to which framing and to which trigerring contexts). The second extension is to include language, through the acoustic medium and pronounced by an interactant, as a possible part of the context. This allows the demonstrator to teach the robot to perform tasks in a specific linguistic context, for example making the learning of commands possible and seamless in the architecture. Lets take the example of a set of demonstrations Dem provided to the imitator that contains a subset of demonstrations Dem_i such that for each $dem_k \in Dem_i$ an object is encircled and the speech input "circle" is present. If there are no $dem \notin Dem_i$ such that the speech input "circle" is present and the object is not circulated, the imitator has enough information to infer that when it hears the word "circle" it is to encircle the object.

Related work

Imitation learning in robots, also called programing by demonstration, has made important advances in recent years [3]. Techniques have been developped that allows a human to teach a robot motor primitives/tasks whose trigerring and action policies typically depends on aspects of the context like the position/speed of objects or of the robot itself [16].

One approach is to build a system that recognizes a set of pre defined motor primitives or postures and then uses the demonstration as a guide to learn how to sequence these lower level parts into a higher level task (see for example [4], [5], [6] and [7]). Another approach is to encode a policy at the trajectory level (see for example [8], [9] or the more recent [10] where a dynamical system is built using recurrent neural networks). Within the trajectory approach the use of Gaussian Mixture Regression [11] stands out as an easy to use method that does not require extensive parameter tuning by the programer and which has been successfully used in numerous robotics experiments (see [12], [13], [14], [15]).

While many of these techniques have focused on learning separately context-dependant tasks, [16] presents a combination of the above mentioned approaches and tackles the problem of an unlabeled set of demonstrations using a Hidden Markov Model (HMM) and global Gaussian Mixture Regression (GMR) based algorithm (HMM and GMR following the "sequence of lower level parts" approach and the "trajectory" approach respectively). Two different tasks (corresponding to a topspin and a drive stroke task in table tennis) are demonstrated by 4 demonstrations each to an imitator consisting of a robotic arm. The demonstrations are unlabeled and the imitator is not given the number of tasks. What task to be performed is determined by the starting position (during reproduction the imitator will perform a topspin stroke when starting in one region and a drive stroke when starting in another region). The imitator uses a two step process to build a global model of the tasks: first the number of states in the HMM are autonomously estimated using the bayesian information criterion (meaning that manual parameter tuning is not necessary). Then the global model is built and it is possible to determine the number of tasks by examining the transition probabilities of the states in the HMM (where each group of states that has no transitions to states outside the group, above some threshold, can be considered a separate task). The combination of autonomously estimating the number of states, the unlabeled demonstrations and the batch building of a global model means that when a demonstration of a new task is added the whole global model has to be re built (and since the demonstrations are not labeled the number of states has to be re checked every time there is a new demonstration of an already demonstrated task). The number of states increases linearly with the number of tasks and the computational complexity increases exponentially in the number of states. In settings where it is unlikely that a large number of different tasks will be needed, such as the table tennis setting in [16], this is not a problem. However, in language learning where one would like to teach the robot a possibly larger number of new motor skills and their potential associated linguistic triggering context, it seems impractical to have to rebuild a global model (re-learning all the words previously learnt) every time a new word is learnt, especially if the complexity of building such a global model increases exponentially with the number of words learnt. This is why we propose the grouping algorithm followed by the incremental online building of local models aided by the groups found.

Furthermore, by considering acoustic linguistic names as a potential, but not necessary, part of the context in an integrated architecture where speech is not treated differently than any other part of the context (e.g. position or speed of objects), we do not only explore a new kind of context in relation to the imitation learning litterature, but also propose a novel approach (to our knowledge) to computational modelling of language bootstrapping and acquisition. Indeed, there is a flourishing landscape of computational modelling of language acquisition research projects ([18], [19]), exploring in particular how language acquisition can be grounded into action [17]. In most of these models, language and action are assumed to develop in interaction but are still considered as separate processes which correspondances is learnt by the organism. As a consequence, this makes the understanding of how the function of language, i.e. to trigger cognitive or motor behaviours, is discovered quite complicated. In the approach presented in this paper, the imitator not only learns that some acoustic waves are associated with certain motor behaviours, but also learns the very fact that certain speech waves in some contexts, as well as potentially objects colors or gestures, shall or shall not trigger behaviours. Consequently, the imitator can learn seamlessly non-linguistic tasks, or linguistic tasks where the "symbolic/denoting" modality may be speech or gestures or any other aspect of the environment, without the need for specifying which particular modality will be used as linguistic before the start of learning.

II. Algorithms

A. Language as part of the context: The interactant, DTW and k-means

The paper explores context dependent sensorimotor learning by imitation where one part of the context is a linguistic input. This is integrated in a demonstration set-up where in front of the learning robot which is kinesthetically demonstrated the task (see [12]), we add an *interactant* human which may or may not pronounce speech sounds or words during the demonstration (see figure 1). When the demonstrator, helped by the interactant, wants the robot to learn a motor skill that should be triggered when a certain speech command is pronounced, the interactant pronounces the word during the motor demonstration achieved by the demonstrator.

Our technical approach aims at representing speech uniformly with other elements of the contexts S, e.g. position and speeds of objects and the robot. To do so, we use an algorithm which projects dynamic acoustic trajectories, initially encoded using Mel-Frequency Cepstral Coefficients (MFCCs), into a low-dimensional static $k-$dimensional space. To do so, the algorithm first samples a variety of speech sounds in the environment (here 50). Then, it runs a k-means algorithm where the similarity measure between two speech sounds is here Dynamic Time Warping, which allows identical words pronounced with various lengths and rhythms to be re-aligned and have a high similarity. Once k prototypes have been found, they are used to define a new representational space for new perceived sounds: each new perceived sound is first encoded in terms of MFCC trajectory, then the similarity between this trajectory and the trajectory of the k prototypes is computed using DTW, and the ordered vector s_{speech} of these similarity measures is used as the static representation of speech sounds, and is as such part of the global context ($s_{speech} \in S_{speech} \subset S$). In the current paper, the sound is recorded starting from the beginning of the demonstration and until the end of the demonstration (the demonstrator stops moving the arms of the robot). Silence is a potential sound to be recorded by the robot. In the following experiments, $k = 3$. The following algorithms will then deal with tasks where the linguistic input is important for what to do as well as tasks where the linguistic input is completely irrelevant (including for its triggering).

B. One demonstration $\Longrightarrow$ one motor primitive/task $\Longrightarrow$ one framing f_i

In this paper, we make an assumption which we think is reasonable and at the same time allows to considerably improve the the system if leveraged appropriately: we assume that a single given demonstration corresponds to a single motor primitive/task which itself is embedded in a single framing. Of course, several demonstrations might correspond to the same task and the same framing, and different tasks might have the same framing and vice-versa, but we exclude here demonstrations which are for example unsegmented sequences of motor primitives/tasks. Knowing that for each task there is a framing such that all the demonstrations of that task have a consistent policy when the data is viewed in this framing allows the imitator to group the different demonstrations with the below computationally tractable algorithms. This explicit grouping will allow the imitator to build an explicit representation of the number of tasks, what the relevant training data is for each task, what region of state space the task should be executed in and what the relevant framing is for this task[1]. During reproduction the current state lets the imitator infer the task (using the task regions), which allows it to find the task data and framing (giving the imitator access to all the relevant data represented in a relevant space and allowing it to ignore irrelevant data). A second assumption that we make in the following (but this is less crucial and might be possible to remove by some modifications of the algorithm not presented here) is that there is a finite number of pre-given available framings.

C. Grouping demonstrations and finding triggering regions and framings

Since the imitator does not know the number of tasks any grouping is possible. For $N = 20$ demonstrations (as in the experiment presented below) the number of possible combinations is very large ($>> 10^{10}$, and increasing very rapidly in the number of demonstrations) and instead of updating every possible combination, 20 possible tasks are assigned membership probabilities for each demonstration. This is a structure that can express every possible grouping but that for 20 demonstrations only has 400 values to update, referred to as m_{it} (the probability that demonstration number i is a member of task number t). If the number of tasks is smaller than the number of demonstrations this would be represented as some tasks having no members (or very low probability for each demonstration to be part of that task). Wether or not a demonstration should be part of a grouping is highly dependent on what other demonstrations are currently in this group. As soon as all the values m_{it} are changed the probabilities for a single demonstration is re normalized to sum to 1 (this will not be pointed out every time an update equation is presented). At the beginning of the grouping algorithm, membership probabilities are distributed randomly (of coarse summing the total probabilities of a demonstration to 1). There are three update procedures; first demonstration-task probabilities are concentrated in tasks whose members has similar policies (in some framing), then demonstration-task probabilities are concentrated in tasks whose members prefer the same framing and finally the demonstration-task probabilities are concentrated in tasks that have many probable members (this prevents the different demonstrations of a single task to form separate groups). At step s the old memberships M_s are stored before they are

[1] Some readers might understand the grouping algorithm better when viewing it as a dynamical system where each part of the algorithm is meant to make attractors around correct solutions more stable and/or to avoid the formation of attractors at incorrect solutions. The readers that does not like to think in terms of attractor dynamics is warned that some parts of the algorithm might be difficult to make sense of if seen as approximate inference or something similar

modified by the 3 updates to M_{mod} (as explained in detail below). The memberships M_{s+1} at step $s+1$ becomes $M_{s+1} = 0.5 * M_s + 0.5 * M_{mod}$ (preventing dramatic fluctuations in groups that are already fairly well established). The grouping algorithm is run for 50 steps (so that each type of update is done 50 times). The values of m_{it} is also not allowed to go below 0.0001[2]. For D demonstrations of T tasks there are $D*(D-1)*(D-2)*...*(D-(T-1))*T*(T-1)*(T-2)*...*2$ correct and identical solutions for this algorithm to find[3] (for the 20 demonstration of 5 tasks as in this paper this is $20*19*18*17*16*5*4*3*2 \approx 2.2*10^8$ correct identical solutions) and the total number of answers is D^D, which is $20^{20} \approx 10^{26}$ in the experiment investigated here (counting only binary values of the m_{mt} variables and constraining the total values of m_{mt} to 1 for each demonstration). When finding the of values M_{mod}, through the three update steps, the values of the previous iteration M_s is often used and will be referred to as $m_{mt_{old}}$ while the values of M_{mod} are simply referred to as m_{mt}. After M_{mod} is calculated the new membership values becomes $M_{s+1} = 0.5 * M_s + 0.5 * M_{mod}$ (and in the next iteration M_{s+1} is referred to as $m_{mt_{old}}$, etc).

Policy difference

$(N-1)^2$ demonstration policy differences $diff_{mn}$ are calculated for demonstrations m and n. For each framing the difference is calculated and then $diff_{mn}$ is set to the lowest difference score. First 10 points are selected from demonstration m. Then the closest points (with distance measured in the task space of the current framing) of demonstration n is found. The output values are compared and the difference $diff_{mn}$ is the mean square difference of the 10 pairs. The weighted mean error WME_{mt} is calculated for the other members of task t: $WME_{mt} = \sum_{i \neq m} diff_{mi} * m_{it}$. Then we have $w_{mn} = ((N-1)*(e^{-diff_{mn}/WME_{mt}}))/(\sum_{i \neq m} e^{-diff_{mi}/WME_{mt}})$. Finally, for each demonstration m, each demonstration $n \neq m$ and for each task t the update $m_{mt} \leftarrow (1 - m_{nt_{old}}) * m_{mt} + m_{nt_{old}} * m_{mt} * w_{mn}$ is performed. The new values of m_{mt} are used in the next update (but the same $m_{nt_{old}}$ values are used). When m_{mt} have been fully updated for all values of m and t the memberships are re-normalized so that the memberships of a single demonstration sum to one.

The policy difference is also used to favor tasks where tension $T_t = \sum_{i=1}^{N} WME_{it} * m_{it_{old}}$ is low. $m_{it} \leftarrow m_{it}/T_t$

Framing similarity

The preference of the different framings of a demonstration is calculated based on how similar the demonstration are to other demonstrations, when compared in the different framings. Preference by demonstration m for framing f: W_{mf} is calculated as $W_{mf} = (\sum_{i \neq m}(1/diff_{mif}))/(F * \sum_{f=1}^{F}(\sum_{i \neq m}(1/diff_{mif})))$ (so that the W_{mf} of a demonstration average 1), where $diff_{mif}$ (the difference between demonstrations m and i in framing f) is calculated as mentioned above. For every value of m, t and $n \neq m$ an update is performed. If $W_{mf} > 1$ and $W_{nf} > 1$ (meaning that the two demonstrations m and n both prefer the framing f), they will "pull" each other into the tasks they belong to: $m_{mt} \leftarrow (1 - m_{nt_{old}}) * m_{mt} + m_{nt_{old}} * m_{mt} * W_{mf} * W_{nf}$. Similarly if $W_{mf} > 1$ and $W_{nf} < 1$ (they prefer different framings) they will push each other out of the tasks they belong to: $m_{mt} \leftarrow (1 - m_{nt_{old}}) * m_{mt} + m_{nt_{old}} * m_{mt} * W_{mf}/W_{nf}$. If $W_{mf} < 1$ and $W_{nf} > 1$ (again preferring different framings) then the update $m_{mt} \leftarrow (1 - m_{nt_{old}}) * m_{mt} + m_{nt_{old}} * m_{mt} * W_{nf}/W_{mf}$ is performed. If $W_{mf} < 1$ and $W_{nf} < 1$ (neither prefer the framing) no update is performed.

Preferring few tasks

The membership probabilities for a task are multiplied with the square of the sum of the demonstration membership probabilities of that task ($m_{it} \leftarrow m_{it} * (\sum_{k=1}^{N} m_{kt})^2, \forall 1 < i < N, 1 < t < N$). This is meant to make attractors where demonstrations belonging to a single task are split into subgroups less stable. The membership values that are used in this update equation are the values after modification by the framing similarity. The values M_{mod} that results from this update is then used to get the M_{s+1} values that will be used as input to the policy difference update in the next step using the equation: $M_{s+1} = 0.5 * M_s + 0.5 * M_{mod}$. After 50 steps the algorithm has created stable groups (for more difficult experiments some more advanced convergence criteria could easily be used, terminating when the grouping algorithm is almost certain about where all the demonstrations belong).

Finding the framings of a task

Once the grouping algorithm is done the framing in which the members look the most similar is selected. The selected task framing for task t is the f for which the sum $\sum_{m=1}^{N} \sum_{n \neq 1} diff_{mnf} * m_{mt} * m_{nt}$ is minimized, where $diff_{mnf}$ is calculated as above (the policy difference when the nearest points are found using the distance measure of framing f).

Incremental grouping of demonstrations

The algorithm presented here does not build a global model of the actual global policy and there is therefore no need for costly recomputations when demonstrations of new tasks are added. The grouping algorithm presented as above is however a batch computation and, if not modified, would have to be re run for every new demonstration observed. It is however well suited for an incremental version (with current data it takes only a few seconds on a modern laptop but with larger number of tasks, demonstrations and number of possible framings, time could become a problem). When the algorithm has grouped all the seen demonstrations and found the corresponding data/framings and regions it can easily use this information when new demonstrations are added. For D new demonstrations and T tasks found in the previous data the number of possible tasks becomes $T + D$. The correlation's already found can also be re used by keeping the already obtained membership values m_{nt} so that what is left to determine is the m_{nt} of the new demonstrations. If the new demonstrations fit into an already established group this may go fast. It would also be possible to go one step further and check if the new demonstrations fit in the established groups

[2]Periodically changing the size of this lowest value as well as the ratio between modified and old values would result in something analogous to "temperature" fluctuations in simulated annealing. This was not implemented as the algorithm converged without it but for more noisy demonstrations it might be worth exploring

[3]the number of ways to draw T groups from D places times the number of ways to internally order the T groups

for some threshold of the (policy-similarity)/(average-group-similarity) ratio (in the already found task framing). If some demonstrations do not fit they can be grouped separately (this introduces some arbitrary cutoff constants that might be good to avoid but seems promising in cases with large numbers of tasks). This idea of including new data in existing local models if they meet some similarity cutoff criteria and otherwise create a new local model is similar to the algorithm presented in [21] (in [21], individual data points instead of entire demonstrations are checked for compatibility with a local model but the basic idea is the same).

D. Finding the task during reproduction

The mean μ_{dt} and variance σ_{dt}^2 of the data in dimension d of task t is calculated for each non empty task group of demonstrations found by the grouping algorithm. To determine what task is to be executed in the current state S each task grouping gets a relevance score $R_t = p_{1t} * p_{2t} * \ldots * p_{dt}$, where p_{it} is the probability density of a gaussian distribution with mean μ_{it} and variance σ_{it}^2 in the current state S_i. The task with the highest relevance score R_t is selected and the data of that group (seen in the framing of that group) is used to build local models during the entire reproduction. The relevance score of a task is designed to be higher if the current state is similar to the data of that task. If all the data of some task is within some tiny segment in some dimension and the current state is far from that segment the relevance score of that task will be very small (decreasing exponentially with the number of standard deviations from the mean).

E. Reproduction with Incremental Local Online Gaussian Mixture Regression (ILO-GMR)

Once the robot has identified which task/motor primitive should be achieved in the current context, it retrieves the associated demonstrations in their associated framing, and from them builds online a sensorimotor policy using incremental online gaussian mixture regression (ILO-GMR) as in [1] (but the important difference is that here we only consider data corresponding to the given identified task and framing, rather than considering all demonstrations of all tasks at each time step of the computation). We summarize this technique in the following.

Global Gaussian Mixture Regression The GMR approach [12] first builds a model using a Gaussian Mixture Model encoding the covariance relations between different variables. If the correlations vary significantly between regions then each local region of state space visited during the demonstrations will need a few gaussians to encode this local dynamics. Given data and the number of gaussians, the use of an Expectation Maximization (EM) algorithm finds the parameters of the model.

A Gaussian probability density function consists of a mean μ and a covariance matrix Σ. The probability density ρ of observing the output v from a gaussian with parameters μ and Σ is:

$$\rho(v) = \frac{1}{2\pi\sqrt{|\Sigma|}} \exp\{-\frac{1}{2}(v-\mu)^T \Sigma^{-1}(v-\mu)\} \qquad (1)$$

To get the best guess of the desired output (e.g. speed in cartesian space of the hand in the robot experiments below) $\hat{v}$ given only the current state x_q (e.g. position and speed of the hand in various referentials and position of an object construing the context, as in the experiments below) we have:

$$\hat{v}(x_q) = E[v|x = x_q] = \mu^v + \Sigma^{vx}(\Sigma^{xx})^{-1}(x_q - \mu^x) \qquad (2)$$

Where Σ^{vx} is the covariance matrix describing the covariance relations between x and v.

A single such density function can not encode non linear correlations between the different variables. To do this we need to use more than one gaussian to form a Gaussian Mixture Model defined by a parameter list $\lambda = \{\lambda_1, \lambda_2, \cdots, \lambda_M\}$, where $\lambda_i = (\mu_i, \Sigma_i, \alpha_i)$ and α_i is the weight of gaussian i. To get the best guess $\hat{v}$ conditioned on an observed value x_q we first need to know the probability $h_i(x_q)$ that gaussian i produced x_q. This is simply the density of the gaussian i at x_q divided by the sum of the other densities at x_q, $h_i(x_q) = \frac{\rho_i(x_q)}{\sum_{j=1}^{M} \rho_j(x_q)}$ (where each density $\rho_i(v)$ is calculated just as in (1), with Σ replaced by Σ_i^{xx}, v with x_q, etc). Writing out the whole computation we have:

$$h_i(x_q) = \frac{\frac{\alpha_i}{\sqrt{|\Sigma_i^{xx}|}} \exp\{-\frac{1}{2}(x_q - \mu_i^x)^T (\Sigma_i^{xx})^{-1}(x_q - \mu_i^x)\}}{\sum_{j=1}^{M} \frac{\alpha_j}{\sqrt{|\Sigma_j^{xx}|}} \exp\{-\frac{1}{2}(x_q - \mu_j^x)^T (\Sigma_j^{xx})^{-1}(x_q - \mu_j^x)\}}. \qquad (3)$$

Given the best guesses $\hat{v}_i(x_q)$ from (2), and the probabilities $h_i(x_q)$ that gaussian i generated the output, the best guess $\hat{v}(x_q)$ is given by:

$$\hat{v}(x_q) = \sum_{i=1}^{M} h_i(x_q)\hat{v}_i(x_q) \qquad (4)$$

The parameter list is found using an Expectation Maximization algorithm (EM) [20] that takes as input the number of gaussians and a database.

ILO-GMR The datapoints of all demonstrations are stored in D. Then, during each iteration of the reproduction of a task the imitator looks at its current state x_q and extracts a local database $D(x_q)$ consisting of the N points closest to x_q (measuring distance in the task space). These points are now used as input to GMR as described above. N is the first parameter of ILO-GMR and is typically slighlty superior to the second parameter M multiplied by the dimensionality of the sensorimotor space. The EM algorithm builds a GMM and then we get the best guess of the current desired speed $\hat{v}(x_q, D(x_q), N, M)$ as described above. So at each iteration new local data is extracted and a new local model is built and used to find the desired direction. The number of gaussians M and the number of local points N are not dependent on the number of tasks which removes the need for manual tuning ($M = 2$ and $N = 30$ are used for all tasks).

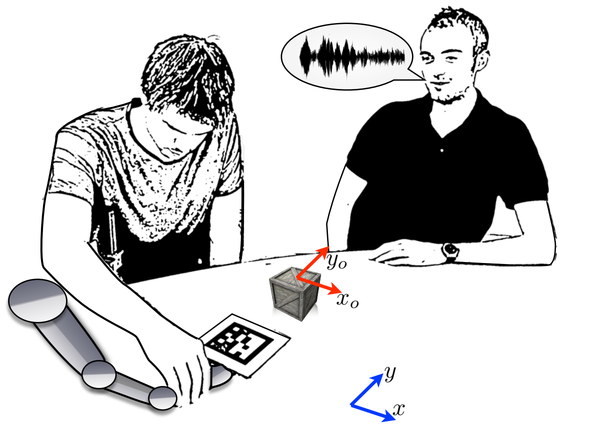

Fig. 1. The experimental setup simulated in this paper. The person to the right speaks and the person to the left moves the arm of the robot. The person guiding the arm of the robot can teach it when speech is relevant by consistently making certain movements in response to certain words and making other movements in response to certain object positions (completely ignoring what is said). The position of the object is different in different demonstrations and the person guiding the robot hand can pay attention to the hands position either in the absolute referential frame (framing 1, blue) or relative to the object (framing 2, red). The data from a demonstration consists of the processed speech and the hand movements. The actual trajectory data are gathered using a mouse capture function implemented in Matlab meant to simulate the above situation.

III. Experiment

A. setup

We use a simulated robot environment modelling the situation depicted in figure 1. A demonstrator provides kinesthetic demonstrations to the robot by directly moving its hand, which is done in 2D using a mouse here, and we assume that we have a low-level inverse controller for finding the corresponding joint trajectories. The robot “sits” in front of a table, on which an object is positioned at the beginning of the demonstration. In front of the robot, there is also an interactant human which pronounces a speech sound during certain demonstrations (speech sounds are not simulated, but real speech sounds pronounced by a human are used, never using the same speech wave for a given word). Then, for reproductions, the robot is not touched by the demonstrator, and has to achieve the appropriate inferred motor primitive/task depending sometimes on the position of the object, sometimes on the speech uttered by the interactant, sometimes on the initial position of its own hand, or sometimes on a combination of these features. It can encode sensorimotor policies using one of three framings. Framing 1 encodes the position and speed of its hand in an absolute fixed reference frame (in addition to the absolute position of the object and the speech sound). Framing 2 encodes the position and speed of its hand in the object centered referential (all other dimensions being equal). Framing 3 includes both the absolute and relative position and speeds of the hand. Five tasks are demonstrated to the robot (see figure 2), each task being demonstrated four times to the robot (in a random order: two successive demonstrations are not necessarily from the same task):

a) Encircle the object counter clockwise when the word ”flower” is spoken (framing 2);

b) Draw a triangle clockwise in the lower left corner when the word ”triangle” is spoken (framing 1);

c) Draw a big square clockwise when the word ”point” (to dispel any intuition in the reader that the word itself holds information that the imitator can use to find out what to do) is spoken (framing 1);

d) Draw a small square counter clockwise with the bottom right corner at the object no matter what the speech input is (framing 2);

e) Encircle counter clockwise the point (0,0) in the fixed reference frame no matter what the speech input is (framing 1). The policy in this task is identical to the one in task a) in that it is to encircle the point (0,0), with the only difference that the reference frame is different (besides different starting positions the demonstrations of task a in framing 2 looks just like the demonstrations of task e in framing 1).

The desired direction angle of the hand is encoded using two dimensions here. The directional angle is measured in its rescaled x and y components (under the constraint that $x^2+y^2=1$) which resolves difficulties with linear regression over a periodic variable (the fact that the angles $\theta=-\pi$ and $\theta=\pi$ are identical outputs makes the raw θ unsuitable for the linear regression methods used). Then the amplitude of local displacement are computed as averaged over the 7 nearest (in time) data points since the raw captured data is not of very good quality (sometimes smooth mouse movements will results in strange angles of the type; $p_{t=1}=\pi/2, p_{t=2}=0, p_{t=3}=\pi/2, p_{t=4}=\pi/2, p_{t=5}=0, \ldots$ and this type of data causes problems, for example in the policy similarity comparisons of the grouping algorithm). The speech was recorded in an ordinary office environment (without anyone talking in the background). For the non-linguistic tasks, a random sound different from the words used for the linguistic tasks was used as input. For the 3 linguistic tasks (tasks a, b and c) the same object position distribution was used (uniformly distributed over the intervals: $-1<x<1, 1<y<2$) and for the 2 non linguistic tasks the object y positions were drawn from the uniform distribution $-1.25<y<-0.5$ and the x positions were drawn from $-1<x<-0.25$ for task d and $.25<x<2$ for task e. The starting hand position (demonstration and reproduction) is always drawn from $-0.25<x<0.25, -1.5<y<-1.25$

B. Results

1) Results of the grouping algorithm: We can see in figure 2 that the demonstrations are grouped correctly; the demonstrations form 5 groups with 4 demonstrations each and no group contains demonstrations from different tasks.

2) Reproductions: In figure 3 we see 4 reproductions of each of the 5 tasks. In figure 4 we see the data selected during reproductions and the reproductions themselves (both presented in the framing found by the imitator).

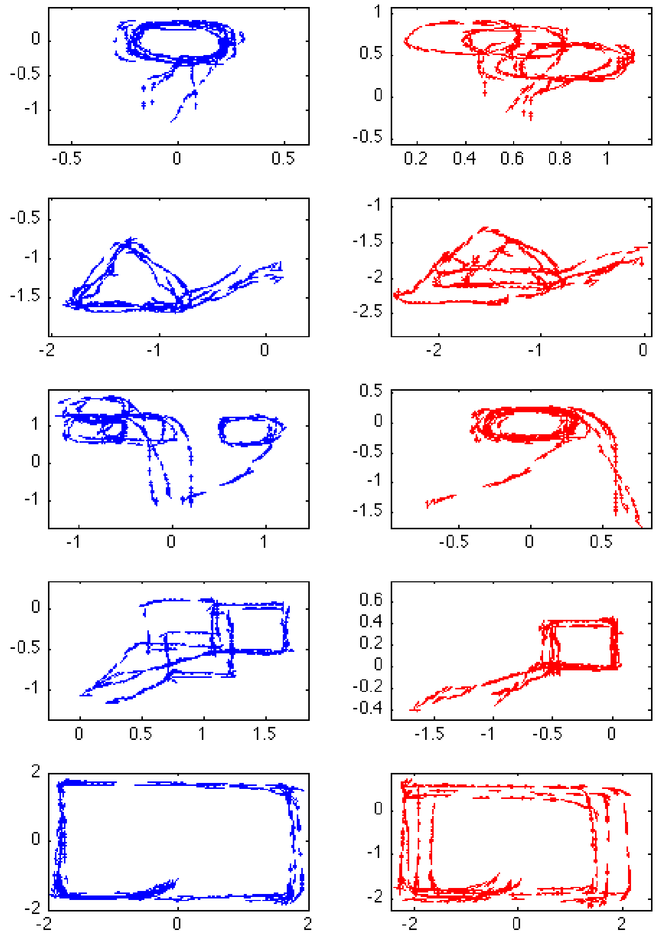

Fig. 2. This shows the five reconstructed tasks represented in framing 1 to the left and in framing 2 to the right. Tasks without any $m_{it} > 0.5$ are not shown (constituting 15 out of the 20 potential groupings). The ordering of the tasks will be different each time (this time it is e,b,a,d,c) but each time the same set of region-framing-data tuples are found. Within each group only demonstrations such that $m_{it} > 0.5$ are shown (with this cutoff all demonstrations are correctly grouped). Typical values are $m_{it} > 0.95$ or $m_{it} < 0.005$ (meaning that the grouping algorithm is nearly certain of the membership of each demonstration)

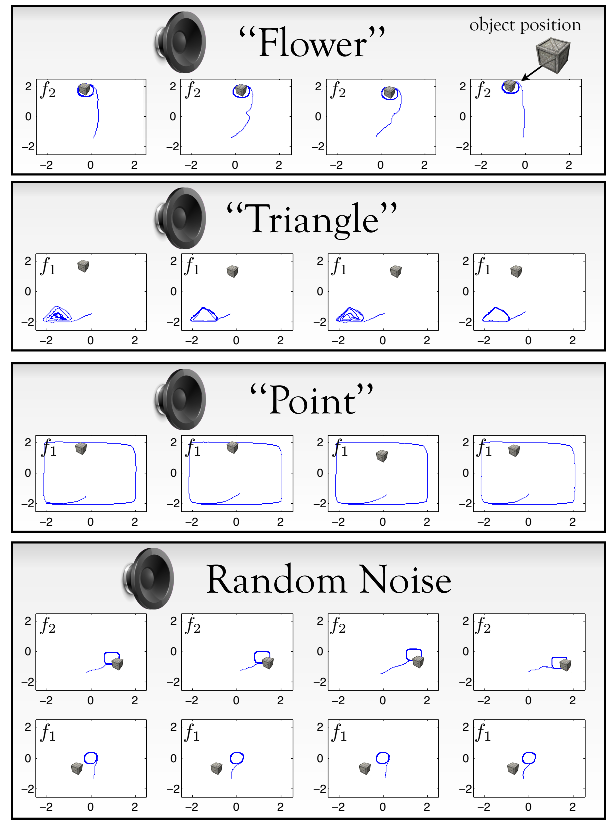

Fig. 3. This shows 4 reproductions each for the five reproduced tasks in the absolute reference frame (the f_1 or f_2 indicates what framing the imitator estimated during the reproductions; each time the framing found is correct). We see that the triangle task (task b) has a tendency to go into the middle of the triangle and circulate in a deformed trajectory after having completed a few correct laps. This problem is not due to the grouping algorithm and is a mistake made by the ILO-GMR algorithm that is presented with only relevant data in the correct framing (and it is presented with all the relevant data). Task a is also a little bit "twitchy" in reproductions 2 and 3 on the way to the object. The reason for this can be seen at the top of figure 4 where we see that there are no demonstration data points in the region of the "twitchy" demonstrations (when, as in that figure, seen in the framing found by the imitator)

IV. Conclusions and Future Work

We have demonstrated that it is possible for a robotic imitator to autonomously group unlabelled demonstrations into separate tasks and to find the correct framing for these task even if the number of tasks are not provided. We have also shown that language can be included as the context in a task and that the imitator can determine for what tasks the linguistic input is relevant.

An obvious continuation will be to test the architecture on real robots. It would also be possible to substitute the speech part of the input with hand gestures or perhaps include some tasks where speech is relevant and some tasks where hand gestures are important.

References

[1] Cederborg, T., Ming, L., Baranes, A., Oudeyer, P-Y: Incremental Local Online Gaussian Mixture Regression for Imitation Learning of Multiple Tasks, *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems* 2010, accepted.

[2] Nehaniv, C., and Dautenhahn, K., Of hummingbirds and helicopters: An algebraic framework for interdisciplinary studies of imitation and its applications, *Interdisciplinary approaches to robot learning.*,World Scientific Press, vol. 24, 2000, pp 136-161.

[3] Aude Billard, Sylvain Calinon, Ruediger Dillmann, Stefan Schaal, Robot Programming by Demonstration, *Springer Handbook of Robotics.*, 2008, pp 1371-1394.

[4] M.N. Nicolescu and M.J. Mataric, Natural methods for robot task learning: Instructive demonstrations, generalization and practice, *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS).*, 2003, pp 241-248.

[5] M. Pardowitz, R. Zoellner, S. Knoop, and R. Dillmann, Incremental learning of tasks from user demonstrations, past experiences and vocal comments, *IEEE Transactions on Systems, Man and Cybernetics, Part B. Special issue on robot learning by observation, demonstration and imitation*, 2007, pp 322-332.

[6] S. Ekvall and D. Kragic. Learning task models from multiple human demonstrations, Title, *Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN).*, 2006, pp 358-363.

[7] A. Alissandrakis, C.L. Nehaniv, and K. Dautenhahn, Correspondence mapping induced state and action metrics for robotic imitation, *IEEE Transactions on Systems, Man and Cybernetics, Part B. Special issue on robot learning by observation, demonstration and imitation.*, vol.

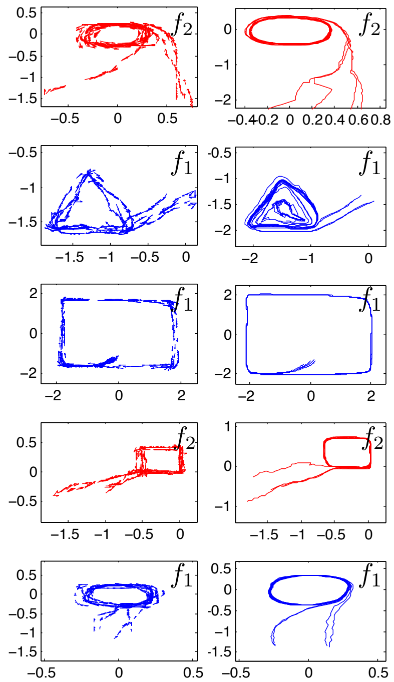

Fig. 4. To the left we see the data retrieved during reproductions of the 5 different tasks in the framing estimated. Each time the correct data is retrieved and the correct framing found. Since the imitator now only has to consider data relevant to the current task and knows what part of the input is relevant to policy the problem has become significantly more simple. To the right we see 4 reproductions each for the five reproduced tasks in the framing found during reproduction by comparing current state with the regions found by the grouping algorithm (blue for framing 1 with hand positions in the absolute coordinate system and red for framing 2 relative to the object). The framings found are correct in all 20 cases. It seems like the imitator falls into the same stable cyclic attractor during all the reproduction (except for in task b where there seems to be at least 2 other stable attractors inside the triangle).

4, 2007, pp 299-307.

[8] N. Delson and H. West., Robot programming by human demonstration: Adaptation and inconsistency in constrained motion., *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA).*, 1996, pp 30-36.

[9] Ude, A, Trajectory generation from noisy positions of object features for teaching robot paths, *Robotics and Autonomous Systems.*, 1993, pp 113-127.

[10] Ito, M., Noda, K., Hoshino, Y., and Tani, J, Dynamic and interactive generation of object handling behaviors by a small humanoid robot using a dynamic neural network model, *Neural Networks.* , 19 (3), 2006, pp 323-337.

[11] Zoubin Ghahramani and Michael I. Jordan, Supervised learning from incomplete data via an EM approach, *Advances in Neural Information Processing Systems.* vol. 6, 1994, pp 120-127.

[12] S. Calinon and F. Guenter and A. Billard, On Learning, Representing and Generalizing a Task in a Humanoid Robot, *IEEE Transactions on Systems, Man and Cybernetics, Part B* , vol. 37, 2007, pp 286-298.

[13] Calinon, S. and Billard, A, Incremental Learning of Gestures by Imitation in a Humanoid Robot, *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction (HRI),.*, 2007, pp255-262.

[14] Aude Billard, Sylvain Calinon, Ruediger Dillmann, Stefan Schaal, Robot Programming by Demonstration, *Springer Handbook of Robotics.*, 2008, pp 1371-1394.

[15] Calinon, S *Robot Programming by Demonstration: A Probabilistic Approach.*,EPFL/CRC Press , 2009.

[16] Calinon, S. and D'halluin, F. and Sauser, E. L. and Caldwell, D. G. and Billard, A. G., Learning and reproduction of gestures by imitation: An approach based on Hidden Markov Model and Gaussian Mixture Regression, Robotics and Automation Magazine, vol. 17, 2010, pp 44-54.

[17] Cangelosi A., Metta G., Sagerer G., Nolfi S., Nehaniv C.L., Fischer K., Tani J., Belpaeme B., Sandini G., Fadiga L., Wrede B., Rohlfing K., Tuci E., Dautenhahn K., Saunders J., Zeschel A. Integration of action and language knowledge: A roadmap for developmental robotics. *IEEE Transactions on Autonomous Mental Development*, 2010, In press.

[18] Steels L., Experiments on the emergence of human communication, *Trends in Cognitive Sciences*, 10(8), pp. 347-349, 2006.

[19] Kaplan, F., Oudeyer, P-Y., Bergen B. (2008) Computational Models in the Debate over Language Learnability *Infant and Child Development*, 17(1), pp. 55–80.

[20] A. Dempster and N. Laird and D. Rubin, Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society.* B, 39(1):, 1977, pp 1-38.

[21] Duy Nguyen-Tuong and Matthias W. Seeger and Jan Peters, Real-Time Local GP Model Learning, *in From Motor Learning to Interaction Learning in Robots*, 2010, pp 193-207.