Noname manuscript No.
(will be inserted by the editor)

Incremental Learning of Skills in a Task-Parameterized Gaussian Mixture Model

Jose Hoyos · Flavio Prieto · Guillem Alenyà · Carme Torras

Received: date / Accepted: date

Abstract Programming by demonstration techniques facilitate the programming of robots. Some of them allow the generalization of tasks through parameters, although they require new training when trajectories different from the ones used to estimate the model need to be added. One of the ways to re-train a robot is by incremental learning, which supplies additional information of the task and does not require teaching the whole task again. The present study proposes three techniques to add trajectories to a previously estimated task-parameterized Gaussian mixture model. The first technique estimates a new model by accumulating the new trajectory and the set of trajectories generated using the previous model. The second technique permits adding to the parameters of the existent model those obtained for the new trajectories. The third one updates the model parameters by running a modified version of the Expectation-Maximization algorithm, with the information of the new trajectories. The techniques were evaluated in a simulated task and a real one, and they showed better performance than that of the existent model.

Keywords Programming by demonstration · Robot learning · Incremental learning

J. Hoyos
Universidad del Quindío
Av Bolivar cl 12N, Armenia-Colombia
and Universidad Nacional de Colombia PhD student.
E-mail: josegabrielh@uniquindio.edu.co

F. Prieto
Universidad Nacional de Colombia
Carrera 30 No 45-03, Bogota, Colombia
E-mail: faprieto@unal.edu.co

G. Alenyà · C. Torras
Institut de Robòtica i Informàtica Industrial CSIC-UPC
Parc Tecnològic de Barcelona. C/ Llorens i Artigas 4-6.
G. Alenyà
E-mail: galenya@iri.upc.edu
C. Torras
E-mail: torras@iri.upc.edu

1 Introduction

Programming by Demonstration (PbD) is a technique that allows a robot to learn how to perform a task from the demonstration of the same task [3] by a human being or another robot. This approach is used in low structured environments as it enables users with no knowledge in programming to teach new tasks to robots. Robots programmed using PbD should acquire the following abilities [2,4]: *i)* to make generalizations in new situations, *ii)* to recover from failures at execution, *iii)* to incrementally learn. This study is concerned with the last of this abilities.

One of the techniques used in PbD is the Task-Parameterized Gaussian Mixture Model (abbreviated in this work as TPGMM), which was presented by Calinon *et al* in 2012 [7]. When robots handle objects, their movements highly depend on the targets given and the object positions which could be defined by using reference frameworks. Specifically, the movement of the robot is conditioned by those frameworks which are called task parameters. Instead of representing each trajectory as a different model, the technique is based on a single model that encodes the different trajectories as a function of the parameters of a task. The model is based on the properties of a product of Gaussians.

This document proposes techniques that allow to add new trajectories to an existing TPGMM. Obviously, added trajectories are different from the ones used when estimating the existing TPGMM. The mentioned techniques are: *i)* Generative, *ii)* Model addition and, *iii)* Direct update. Strategies hereby presented operate under the assumption that trajectories with which the existing TPGMM was trained are not available, that the number of Gaussians is fixed for both the existing model and the incremented one, and that this number is suitable to sufficiently model the trajectories. The generative and direct update techniques are based on Calinon & Billard's work [6], where the authors proposed techniques that allow the increment of Gaussian mixture models (GMM), and the model addition of GMM technique is based on Hall & Hicks's [10] work. In this work we deal with task-parameterized GMM, in contrast to the two previously mentioned methods that deal only with GMM. Another difference is that, in the TPGMM model addition, the RMS error of the trajectories is employed as optimization function, contrary to the model's dependent function employed by Hall and Hicks.

The first proposed technique estimates a new model by using the previous model and the previous task parameters to generate trajectories. Using the generated trajectories and the ones that desire to be added, a set of trajectories is obtained and with them, the new TPGMM is estimated. The second technique is based on the addition of Gaussian Mixture Models (GMM) proposed by [10], which allows direct "adding" of the parameters of two models; in this case, the first term of the addition is the existent model parameters, the second term is the model parameters estimated with the new trajectories, and the result is the new model parameters. The third one is based on a modified version of the equations of the Expectation-Maximization (EM) algorithm, which estimates the model parameters. The modified equations separate the components related to the existent model parameters from the components that require new trajectories; therefore the model is directly updated with its parameters and the new trajectories.

The novelty in these techniques is that they produce new TPGMM models which facilitate the re-training of robots by demonstration, as it is not necessary to teach the robot all the trajectories again or to have these trajectories saved for

its estimation. However for retraining, the generative and model addition techniques require the storage of the task parameters from the previous trajectories. Contrarily, the direct update technique does not require so.

The techniques were validated by using simulated and real tasks. In the simulated task, a set of trajectories obtained manually was used; in the testing of a real task, trajectories were obtained with a WAM robot [19] performing the task of putting a sleeve on a mannequin arm for different positions forward or backwards. The Root Mean Square (RMS) error calculated between demonstrations and reproductions shows the performance of the techniques and how these improve the model response compared to that produced if only the existing model was used.

This paper has the following structure. In Section 2, some related studies are described. A brief of task-parameterized Gaussian mixture models is provided in Section 3. Proposed strategies are presented in Section 4. Test, results and a discussion of advantages and disadvantages of each technique are presented in Section 5. Finally, conclusions are drawn at the end of the paper.

2 Related Work

The TPGMM technique is described by Calinon *et al* [7] as a technique that allows the creation of new trajectories parameterized by points related to the task. The authors apply this technique to tasks such as following movements of objects with one or two arms and the task of sweeping by using two arms. The same model has also been employed to learn collaborative impedance-based robot behaviors as in the work of Rozo *et al* [18]. The work by Calinon *et al* [5] compares TPGMM techniques with other strategies that permit generalizing trajectories from demonstrated ones and task-related parameters. The comparison contrasts three groups of techniques: *i)* Strategies using N models for M demonstrations, *ii)* Multiple reference frames, *iii)* Coding in a single model. The task was rolling a pizza dough; the results graphically show, with four metrics, that the TPGMM technique gives a better performance over the others. On the other hand, a modification to the previous studies which allows generating new trajectories, even when one or several parameters are missing in the task, was presented by Alizadeh *et al* [1]. The modification is that only the parameters of the existing task were considered when calculating the equations. Incremental learning is not mentioned in these studies.

In PbD of robots, incremental learning can be divided into tasks incremental learning [15, 16] and skills incremental learning. A skill describes a basic action, for example, transport and manipulation; a task is a skill sequence [16]. The present work focuses on skills incremental learning.

Some works that allow to increase the number of trajectories in PbD have been presented as in the work of Calinon & Billard [6], where a technique that allows adding trajectories to an existing GMM is shown. The authors propose two types of strategies: direct and generative. The direct strategy is based on the assumption that new paths are very similar to the previous ones used to estimate the model; therefore, the EM algorithm to adjust the model is executed a number of additional times, but using the new trajectories. In the generative technique the existent model is used to generate random points that are gathered together with new trajectories. With this set, a model is estimated and there is a learning ratio that provides a little forgetting of the past trajectories. For the generative

technique, one of the differences of this proposal is that trajectories are added to a task-parameterized model and that there is no learning factor, since it is necessary to remember all trajectories with the smallest possible error.

Grollman *et al* [9] proposed a technique in PdD, in which a mobile robot incrementally learns input-output maps. The technique is called dogged learning, and it combines Locally Weighted Projection Regression (LWPR) and Mixed Initiative Control (MIC). Regression allows incremental learning; control is adjusted through demonstrations and then the robot carries out the actions; in addition, learning becomes interactive and the robot asks the human when facing new situations. Whereas the technique allows a robot to learn how to make generalizations in front of a variation of an input vector, its design does not permit that the incremental learning that taking place directly with trajectories related to task parameters.

Kulic *et al* [11] present a technique that allows to increment movement primitives by using a factorial hidden Markov model and a decision tree. The starting point is a database of initial primitive movements; when a new movement is found, the decision tree classifies it as the most similar one in the database; then, by using clustering techniques, the primitive movement is added to the group. Even though the technique works for several movements, it does not deal with the issue of generalizing new trajectories. A similar study was presented by Lee *et al* [14], but this time by using PCA and GMM before the classification.

In 2010, Cederborg *et al* [8] proposed a technique based on Gaussian Mixture Regression (GMR); the technique allows to increase the trajectories through a database with a KD-tree and GMR. It generates a GMR model in real time when needed. The data to estimate the model is recovered from the database through a quick search. Whereas the technique allows the reproduction of trajectories when the initial position changes, it does not allow variations in other parameters as it does in our case.

Lee and Ott [12,13] proposed a technique that allows to refine existent trajectories by using a compliance control, kinesthetic learning and a modification of the Hidden Markov Models technique (HMM). During the execution of the trajectory, the compliance controller allows the human to make corrections in a kinesthetic way, and such corrections are incrementally added to the model for subsequent reproductions. In the incremental technique, the authors generate a single trajectory by using the past model; with this trajectory and the new one, the HMM model is recalculated. In the current study, trajectories are added to a task-parameterized model so that the trajectories may vary in shape and location when the parameters are changed.

Due to the above-mentioned aspects, and to the authors knowledge, there are few studies in trajectory generalization that permit incremental task learning and that are a contribution to the development of generalization techniques in programming by demonstration.

3 Brief of Task-Parameterized Gaussian Mixture Models

Calinon *et al* [7] proposed the Task-PGMM technique, here a brief of this is presented. In order to estimate the model, M demonstrations are used as starting point, each with T_m points of data which build a set of data $\{\boldsymbol{\xi}_n\}_{n=1}^{N}$ with

$N = \sum_{m=1}^{M} T_m$, and $\boldsymbol{\xi}_n = [t_n, y_n]^T \in \mathbb{R}^{D+1}$, where D is the dimension of each data point and $\{t_n,\, y_n\}$ the time and Cartesian space. Each demonstration is associated with the task parameters $\{\boldsymbol{A}_{n,j}, \boldsymbol{b}_{n,j}\}_{j=1}^{N_P}$, representing N_P reference frames, related by $\boldsymbol{A}_{n,j}$ transformation matrices and $\boldsymbol{b}_{n,j}$ displacement vectors. Indexes n and j represent the time sample and the $j-th$ reference frame.

The model parameters are $\{\pi_i, \boldsymbol{Z}_{i,j}^{\mu}, \boldsymbol{Z}_{i,j}^{\Sigma}\}$, , which represent the mixture coefficients, the centers and the matrices of covariance for each frame j and the i mixture component; with these parameters, the $\boldsymbol{\mu}_{n,i}$ resulting center of the model and the $\boldsymbol{\Sigma}_{n,i}$ matrix of covariance of each i component, are calculated as the product of the Gaussians linearly transformed as in the work of Rozo *et al* [17]:

$$\mathcal{N}(\boldsymbol{\mu}_{n,i}, \boldsymbol{\Sigma}_{n,i}) = \prod_{j=1}^{N_P} \mathcal{N}(\boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\mu} + \boldsymbol{b}_{n,j}, \boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\Sigma}\boldsymbol{A}_{n,j}^{T}). \tag{1}$$

By using the product property of the normal distributions, the centroid and the matrix of the covariance in the previous equation are given by:

$$\begin{aligned} \boldsymbol{\mu}_{n,i} &= \boldsymbol{\Sigma}_{n,i} \sum_{j=1}^{N_P} (\boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\Sigma}\boldsymbol{A}_{n,j}^{T})^{-1}(\boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\mu} + \boldsymbol{b}_{n,j}), \\ \boldsymbol{\Sigma}_{n,i} &= \left(\sum_{j=1}^{N_P} (\boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\Sigma}\boldsymbol{A}_{n,j}^{T})^{-1} \right)^{-1}. \end{aligned} \tag{2}$$

The model parameters are iteratively estimated by using a modification of the Expectation-Maximization procedure Calinon *et al* [7]. Each iteration has two steps: calculating the posterior probability of the model and calculating the model parameters based on that probability. The difference with the traditional EM is that the algorithm takes into account the task parameters. The TPGMM reproduction is performed with Gaussian Mixture Regression (GMR), as in the work of Rozo *et al* [17], or Alizadeh *et al* [1].

4 Proposed Incremental Task-Parameterized GMM Techniques

In this section the incremental task-parameterized GMM techniques are presented, the generative and direct update techniques are based on the work of Calinon *et al* [6], and the technique of model parameters adding in that of Hall & Hicks [10]. The incremental estimation techniques require the knowledge of the previous TPGMM as well as the new trajectories to be added and his respective task parameters. The number of Gaussians is fixed in this proposal for both the existing and the incremented models. Note that such number has to be adjusted to correctly model all trajectories.

4.1 The Generative Technique

From the generative GMM technique proposed by Calinon *et al* [6], the equations were adapted to deal with task-parameterized Gaussian mixtures models. The trajectories related to the task parameters used to estimate the existent model are

generated by using the existent model. The new TPGMM is estimated with the set of these trajectories and the new ones that are to be added. The technique has the following steps:

1. The same number of trajectories is generated by using the existing model. The assumption is that the task parameters of the previous trajectories are known. The task parameters storage only requires a small amount of memory, compared to store of the data of the existent trajectories.
2. The new trajectories are added to the generated ones, creating a new dataset.
3. With the dataset obtained in the previous step, the initial parameters of the model are calculated.
4. A new model is estimated by using the modified EM algorithm, which is the same as that of the TPGMM.

Given the existent task-parameterized Gaussian mixture model and the task parameters $(\boldsymbol{A}, \boldsymbol{b})$, GMR is used to reproduce the M trajectories. Then, a mobile average filter is applied to each one, obtaining the generated trajectories $\boldsymbol{\xi_g}$. A set is formed with these and the new trajectories $\boldsymbol{\xi_n}$:

$$\begin{aligned} \boldsymbol{\xi}_c &= [\boldsymbol{\xi}_g, \boldsymbol{\xi}_n], \\ \boldsymbol{A}_c &= [\boldsymbol{A}, \boldsymbol{A}_n], \\ \boldsymbol{b}_c &= [\boldsymbol{b}, \boldsymbol{b}_{new}]. \end{aligned} \tag{3}$$

A new initial model is estimated with the previous information using the data time-division technique. The definitive values are estimated with the obtained model, by using the modified EM algorithm.

4.2 The Model Addition Technique

This technique permits adding an existing model with that estimated from the new trajectories that are to be added. It is based on the addition of GMM models proposed in 2004 by Hall & Hicks [10]. The modification presented here was to convert GMM equations into task-parameterized GMM ones. The technique has the following steps:

1. The model of the new trajectory or trajectories is estimated.
2. The model obtained in the previous step is concatenated with the existing one.
3. The model resulting from the concatenation is simplified by using an optimization process.

Concatenation: Given two task-parameterized GMM models: $\{\alpha_i, \boldsymbol{Z}_{i,j}^{\nu}, \boldsymbol{Z}_{i,j}^{C}\}$, $\{\grave{\alpha}_i, \grave{\boldsymbol{Z}}_{i,j}^{\nu}, \grave{\boldsymbol{Z}}_{i,j}^{C}\}$ with distribution probability:

$$p(\boldsymbol{\xi}) = \sum_{i=1}^{K_1} \alpha_i \mathcal{N}(\boldsymbol{\xi} | \boldsymbol{Z}_{i,j}^{\nu}, \boldsymbol{Z}_{i,j}^{C}), \tag{4}$$

$$q(\boldsymbol{\xi}) = \sum_{i=1}^{K_2} \grave{\alpha}_i \mathcal{N}(\boldsymbol{\xi} | \grave{\boldsymbol{Z}}_{i,j}^{\nu}, \grave{\boldsymbol{Z}}_{i,j}^{C}). \tag{5}$$

For $j = 1, ..., N_P$ reference frames, the added probability is:

$$\begin{aligned} r(\boldsymbol{\xi}) &= f_1 p(\boldsymbol{\xi}) + f_2 q(\boldsymbol{\xi}), \\ r(\boldsymbol{\xi}) &= f_1 \sum_{i=1}^{K_1} \alpha_i \mathcal{N}(\boldsymbol{\xi} | \boldsymbol{Z}_{i,j}^{\nu}, \boldsymbol{Z}_{i,j}^{C}) + f_2 \sum_{i=1}^{K_2} \grave{\alpha}_i \mathcal{N}(\boldsymbol{\xi} | \grave{\boldsymbol{Z}}_{i,j}^{\nu}, \grave{\boldsymbol{Z}}_{i,j}^{C}), \\ r(\boldsymbol{\xi}) &= \sum_{i=1}^{K_1+K_2} \beta_i \mathcal{N}(\boldsymbol{\xi} | \boldsymbol{Z}_{i,j}^{\vartheta}, \boldsymbol{Z}_{i,j}^{\Xi}). \end{aligned} \tag{6}$$

equation (6) is equal to concatenated model $\{\beta_i, \boldsymbol{Z}_{i,j}^{\vartheta}, \boldsymbol{Z}_{i,j}^{\Xi}\}$, that has $(K_1 + K_2)$ components, where $f_2 = 1 - f_1$.

Simplification: Given the elements of the concatenated model, it can be simplified into another model with K components where $K < K_1 + K_2$. Simplification is carried out based on a mixture matrix $\boldsymbol{w}$, between the concatenated model and the simplified:

$$\pi_l = \sum_{i=1}^{K_1+K_2} w_{i,l} \beta_i \tag{7}$$

$$\boldsymbol{Z}_{l,j}^{\mu} = \frac{1}{\pi_l} \sum_{i=1}^{K_1+K_2} w_{i,l} \beta_i \boldsymbol{Z}_{i,j}^{\vartheta}, \tag{8}$$

$$\boldsymbol{Z}_{l,j}^{\Sigma} = \frac{1}{\pi_l} \left(\sum_{i=1}^{K_1+K_2} w_{i,l} \beta_i \left(\boldsymbol{Z}_{i,j}^{\Xi} + \boldsymbol{Z}_{i,j}^{\vartheta} (\boldsymbol{Z}_{i,j}^{\vartheta})^T \right) \right) - \boldsymbol{Z}_{l,j}^{\mu} (\boldsymbol{Z}_{l,j}^{\mu})^T, \tag{9}$$

with the restrictions:

$$\sum_{i=1}^{K_1+K_2} w_{i,j} = 1 \tag{10}$$

$$\sum_{i=1}^{K_1+K_2} w_{i,j} \alpha_i < 1 \tag{11}$$

Different to the optimization function described by [10], in order to find the mixture matrix $\boldsymbol{w}$, which presents slow convergence, an optimization function was used based on the error between the generated and incremented trajectories with the simplified model result ones. The weights that permit obtaining the simplified model are found by using an initial matrix and the function that needs to be minimized. The initial values of $\boldsymbol{w}$ was heuristically selected, as follows:

$$\boldsymbol{w} = 0.45 \begin{bmatrix} I_{K_1 \times K} \\ I_{K_2 \times K} \end{bmatrix}, \tag{12}$$

where $\boldsymbol{I}$ is the identity matrix. The function to be minimized that was used is based on the RMS error:

$$f = \frac{1}{T_m} \sum_{m=1}^{M} \sum_{k=1}^{T_m} \| \boldsymbol{y}_m(k) - \boldsymbol{y}_m^s(k) \|^2, \tag{13}$$

where $\boldsymbol{y}_m$ is the set made by the trajectories reproduced using the previous model with existent task parameters and the added desire trajectories, and $\boldsymbol{y}_m^s$ are the equivalent trajectories reproduced by the simplified model using the task parameters (all trajectories) and are recalculated at each iteration during the minimization process. The Matlab function used for optimization is "fminsearchcon", which allows to include restrictions as those described in Eq. (10) and (11). Although for $\boldsymbol{y}_m$, the existent task parameters are needed, their storage only requires a small amount of memory, compared to store of the data of the existent trajectories.

4.3 Direct Update Technique

Similar to generative technique, the equations of direct update GMM described by [6] were modified into task-parameterized GMM ones. In this technique, enclosed in the equations of the Expectation-Maximization algorithm EM, the components related to existent model parameters are separated from the components related to those of the new trajectories. The technique assumes that the posterior probabilities set is similar for both the existent trajectories and the new ones.

By defining the existent model parameters as $\{\boldsymbol{\pi}, \boldsymbol{Z}^{\mu}, \boldsymbol{Z}^{\Sigma}\}$ with K Gaussians, and the new trajectories $\tilde{\boldsymbol{\xi}}$ to be increased, composed of T_m samples and $\tilde{M}$ demonstration trajectories, $\tilde{N} = \tilde{M}T_m$ data points of $D+1$ dimension are obtained. From the existent TPGMM model parameters and the new trajectories, the model is updated with the next steps:

1. Computation of the initial expectation vector from initial mixed coefficients and N existent data points:

$$E_i = N\pi_i \tag{14}$$

2. With the new trajectories that want to increase, an initial TPGMM $\{\tilde{\boldsymbol{\pi}}, \tilde{\boldsymbol{Z}}^{\mu}, \tilde{\boldsymbol{Z}}^{\Sigma}\}$, is estimated by using the data division time technique.
3. The modified EM algorithm is applied an additional quantity of times:

   E-step:

$$\tilde{h}_{n,i} = \frac{\tilde{\pi}_i \mathcal{N}(\tilde{\boldsymbol{\xi}}_n | \tilde{\boldsymbol{\mu}}_{n,i}, \tilde{\boldsymbol{\Sigma}}_{n,i})}{\sum_k^{N_K} \tilde{\pi}_k \mathcal{N}(\tilde{\boldsymbol{\xi}}_n | \tilde{\boldsymbol{\mu}}_{n,k}, \tilde{\boldsymbol{\Sigma}}_{n,k})}, \tag{15}$$

   where $\tilde{\boldsymbol{\mu}}_{n,i}$ and $\tilde{\boldsymbol{\Sigma}}_{n,i}$ is calculated like in Eq. (2).

M-step:

$$\begin{aligned}
\tilde{E}_i &= \sum_{n=1}^{\tilde{N}} \tilde{h}_{n,i}, \\
\tilde{\pi}_i &= \frac{E_i + \tilde{E}_i}{N + \tilde{N}}, \\
\tilde{\boldsymbol{Z}}^{\mu}_{i,j} &= \frac{E_i \boldsymbol{Z}^{\mu}_{i,j} + \sum_{n=1}^{\tilde{N}} \tilde{h}_{n,i} \tilde{\boldsymbol{A}}^{-1}_{n,j} [\tilde{\boldsymbol{\xi}_n} - \tilde{\boldsymbol{b}}_{n,j}]}{E_i + \tilde{E}_i}, \\
\tilde{\boldsymbol{Z}}^{\Sigma}_{i,j} &= \frac{\boldsymbol{s}_1 + \boldsymbol{s}_2}{E_i + \tilde{E}_i},
\end{aligned} \tag{16}$$

with:

$$\begin{aligned}
\boldsymbol{s}_1 &= E_i[\boldsymbol{Z}^{\Sigma}_{i,j} + (\boldsymbol{Z}^{\mu}_{i,j} - \tilde{\boldsymbol{Z}}^{\mu}_{i,j})(\boldsymbol{Z}^{\mu}_{i,j} - \tilde{\boldsymbol{Z}}^{\mu}_{i,j})^T], \\
\boldsymbol{s}_2 &= \sum_{n=1}^{\tilde{N}} \tilde{h}_{n,i} \tilde{\boldsymbol{A}}^{-1}_{n,j} [\tilde{\boldsymbol{\xi}}_n - \hat{\boldsymbol{\mu}}_{n,i,j}][\tilde{\boldsymbol{\xi}_n} - \hat{\boldsymbol{\mu}}_{n,i,j}]^T \tilde{\boldsymbol{A}}^{-T}_{n,j}, \\
\hat{\boldsymbol{\mu}}_{n,i,j} &= \tilde{\boldsymbol{A}}_{n,j} \tilde{\boldsymbol{Z}}^{\mu}_{i,j} + \tilde{\boldsymbol{b}}_{n,j}.
\end{aligned}$$

5 Tests and Results

The generative and model addition techniques were validated by using two tests: the simulation of a sweeping task, and the task of putting a sleeve on a mannequin arm using a WAM robot. The TPGMM reproduction for the existent and proposed techniques are performed with GMR [17,1], and the result trajectories filtered with two samples mobile filter. For the comparison of the trajectories that occur in each task, the RMS error between the demonstrated trajectory $\boldsymbol{\xi}_d$ and the reproduced one $\boldsymbol{y}$, the following equation is used:

$$e_{RMS} = \frac{1}{T_m} \sum_{k=1}^{T_m} \|\boldsymbol{y}(k) - \boldsymbol{\xi}_d(k)\|, \tag{17}$$

where T_m is the total of samples in the demonstrated trajectory. With this equation, the following comparisons can be established:

- The reproduction using the existing estimated model with the initial trajectories, compared to the ones obtained with the proposed techniques that added given trajectory.
- The reproduction using the TPGMM calculated with the set of initial trajectories and the added ones, compared to the reproduction of the models obtained by using the proposed techniques.

The previous tests intend to observe and analyze pros and cons of the proposed techniques. For the addition model method, the f_1 value was set in $0,5$ for both the sweeping simulation and the task of putting a sleeve on a mannequin.

For simplicity and easy understanding of the figures, and considering that the performance of the direct update technique is not high, the results of this technique

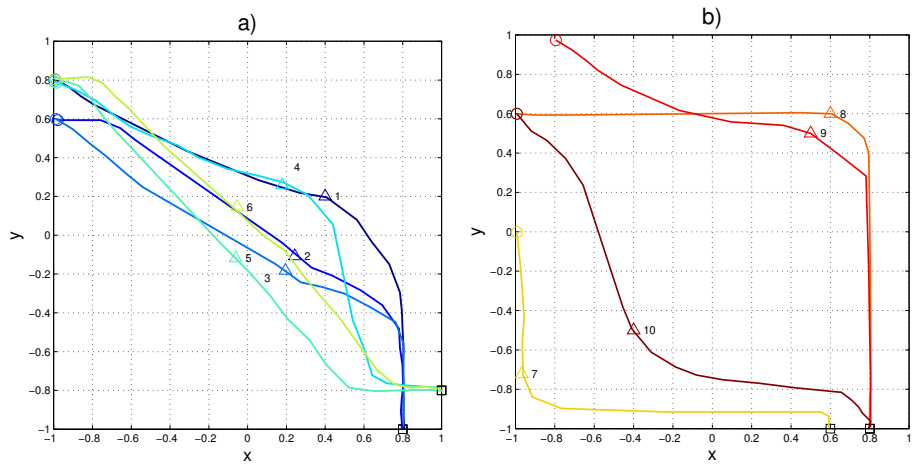


Fig. 1: Trajectories of the simulation test. a) Used to estimate the existing model (trajectories 1 to 6). b) Used to increment (trajectories 7 to 10). The circle indicates the starting point of the trajectory, the triangle represents the position of the object to be swept away and the square the position of the collector-dustpan.

are shown in the multiple addition trajectories test and Fig. 17 from the Discussion Section.

5.1 The Simulation of Sweeping

The simulation of sweeping an object starts with the end effector at an initial point; then the end effector moves to the object to be swept and takes it to the collection point [1]. The trajectories (see Fig. 1) were obtained manually by using a computer mouse. The points that made up the trajectory were re-sampled using the spline function of Matlab. The trajectories in Fig. 1a were used to estimate the existing model, and the trajectories in Fig. 1b were the ones added for testing purposes. The task parameters were: the starting point, the object location and the collector (i.e. dustpan) location. The matrix $\boldsymbol{A}$ and the vector $\boldsymbol{b}$, as shown in Appendix A.1, are calculated by using the previous mentioned positions. It was heuristically established in four the model states. That value was used for calculating the existent model as well as for calculating generative and model addition techniques.

An example of reproduction using the task parameters of trajectory number 7 (yellow line) is shown in Fig. 2. The existent model responses (dashed line in black); generative technique (dashed line in green), and dashed red line depicts the response of the model addition technique. When using the proposed techniques, an improvement can be seen in the model response compared to the one produced by the model calculated with only the existing trajectories.

Figure 3 shows a comparison of RMS error generated when using the model calculated with the initial trajectories, against the proposed techniques added with each of the trajectories (7 to 10). When using the proposed techniques, a reduction in the error can be noticed in all of the added trajectories. The addition model

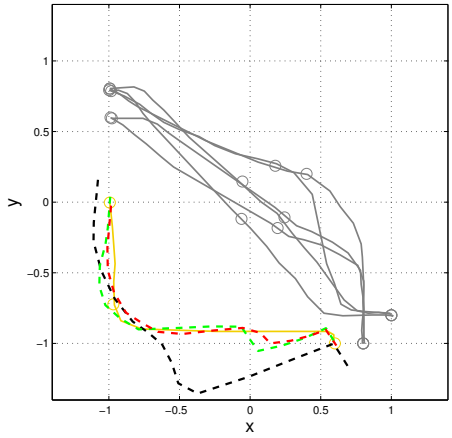


Fig. 2: Example of the reproduction using the task parameters of the trajectory number 7 (in yellow). The dashed line in black: Existing model. Dashed line in green: Generative technique. Dashed line in red: Model addition.

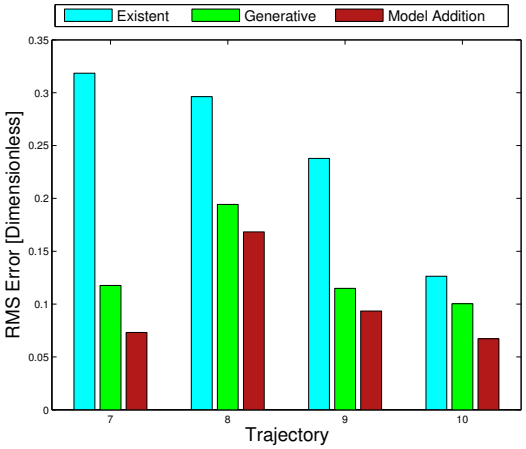


Fig. 3: Root mean square error for the simulation task and comparison between the existing model and the incremented models. Cyan: the existing TPGMM model. Green: Generative technique. Red: Model addition.

technique is the one showing the smallest error in the incremented trajectory, which is probably due to: *i)* the reconstruction, in the case of the generative technique, involves error generation; and *ii)* the optimization present in the addition technique reduces the RMS error.

The error variance is not show in Fig. 3 because the proposed techniques do not depend on the initialization or estimation of random values, and the error is always the same for each time.

Figure 4 compares the RMS error of the incremented TPGMM with a given trajectory against the proposed techniques. This figure also shows that, except for a few cases, the error is similar in all the techniques and that the addition model

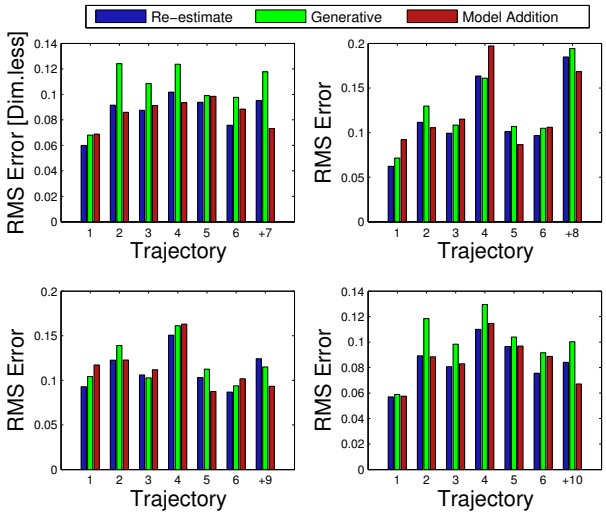


Fig. 4: Root mean square error for the simulation task and comparison between the TPGMM incremented (batch re-estimate) against the proposed incremented models. Blue: Incremented TPGMM. Green: the generative technique. Red: Model addition technique.

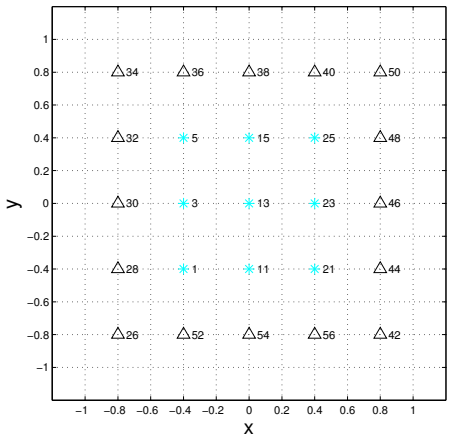


Fig. 5: Different positions of the object to be swept. The positions marked with an asterisk are related to trajectories used to estimate the existent TPGMM; and the ones marked with triangles correspond to the trajectories that are incremented.

is the technique that generates the lowest error value between the two incremental techniques for the initial trajectories (1 to 6). It should be noticed that the error obtained by the TPGMM technique is calculated by using the information of the demonstrated trajectories, both existing and added ones, which results in a smaller error value.

For the example presented in Fig. 4, only one trajectory is added for each test. The example below presents the addition of multiple trajectories.

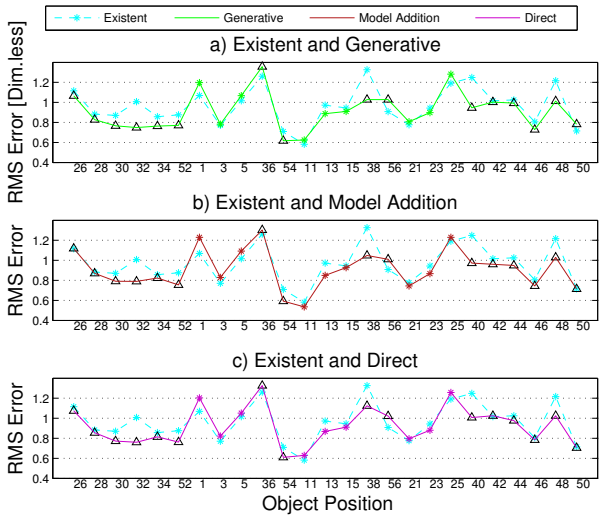


Fig. 6: Root mean square error for the different positions of the object to be swept. In cyan, the error of the existent model. a) In green, the generative technique; b) In red, the addition model technique; c) In magenta, direct update technique; the positions marked with asterisks are the trajectories used to estimate the existent TPGMM; the ones marked with triangles, are the added trajectories.

Testing the addition of multiple trajectories

Figure 5 shows the points that form a square resulting from varying the position object to be swept and leaving starting and end position fixed. A trajectory was carried out manually for each position of the square. In this test, the results of generative, model addition and direct update techniques are presented.

The existent model was estimated by using the central points (numbered asterisks with values smaller or equal to 25) and the new models were estimated by using all trajectories in the external square (triangles numbered with values greater than 25). Figure 6 shows the root mean square error for each demonstrate trajectory and produced by using the existent TPGMM (cyan). It also shows, in green, the root mean square error using the reproduction generated by the generative technique, in Fig. 6b in red, using the model addition technique and, in Fig. 6c in magenta, the direct update technique. The added trajectories are shown with a black triangle and in almost all the added trajectories the error is smaller. In this case, the error of proposed techniques does not decrease much as in the previous test, due to the fact that the existent model error is lower, because it is estimated with 9 trajectories (depicted as a central square marked with asterisks) and the external trajectories are similar to the central ones.

Figure 7 shows the reproductions by using the resulting models from the three techniques and the corresponding task parameters of the number one initial trajectory. The result trajectory obtained from the existent model is shown in black dashed line; the one from the generative technique in green dashed line; the model addition technique in red; and, the direct update technique in magenta. This last-named is the farthest one from the original trajectory (continuous gray line).

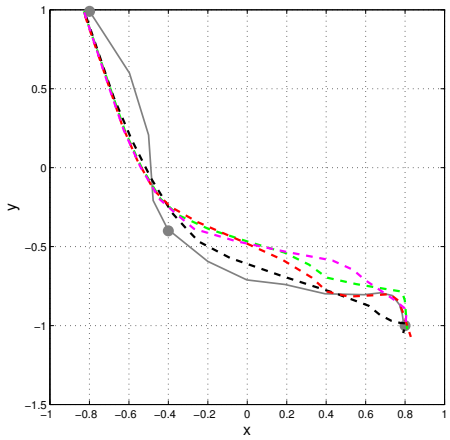


Fig. 7: Example of the reproduction using the task parameters of the initial trajectory 1. The dashed line in black: Existing model. Dashed line in green: Generative technique. Dashed line in red: Model addition. Dashed line in magenta: Direct update.

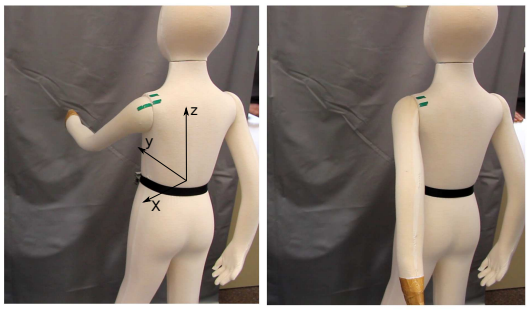


Fig. 8: Two examples of the arm position. Left: Arm towards the front. Right: Arm towards the back.

5.2 Performing a Real Task

The task of putting a shirt sleeve on a mannequin arm was implemented. It involves diverse trajectories when changing the position and curvature of the mannequin arm for both cases, that is, when the arm is towards the front and the back. Figure 8 shows two examples of the mannequin arm positions and Fig. 9 shows the trajectories obtained by kinesthetic demonstration.

A TPGMM with the trajectories of the arm towards the front was estimated obtaining the existing model. Then, with the generative and model addition techniques, the existing model was increased with each of the trajectories of the hand towards the back.

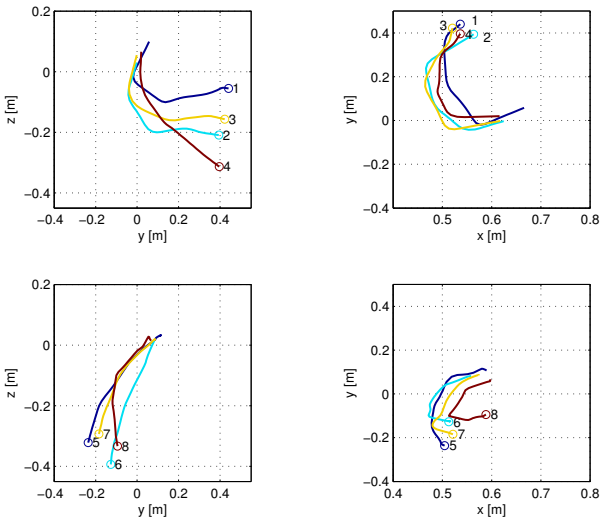


Fig. 9: Demonstrations used in the real task. Upper: Trajectories arm to the front. Lower: Trajectories arm to the back.

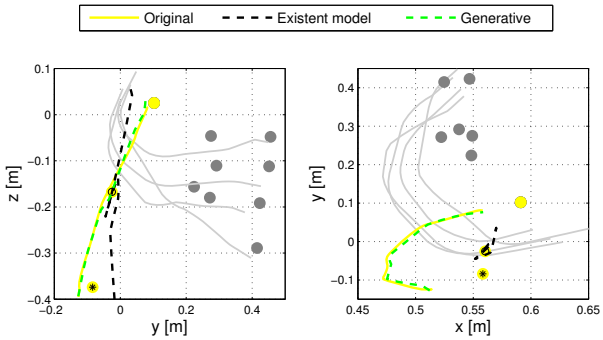
a) Hand. b) Elbow. c) Shoulder.

Fig. 10: Obtaining the task parameters by recording the robot's pose.

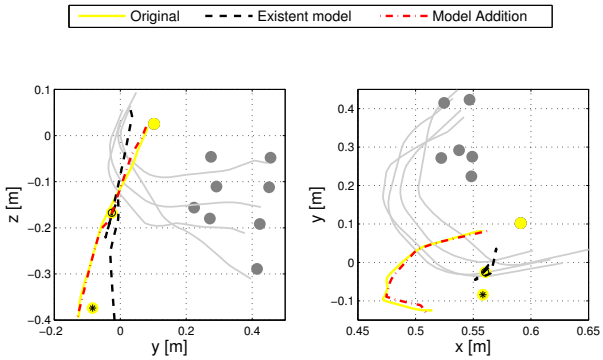
The following parameters were selected for the task: point on the hand, point on the elbow, and point on the shoulder. Figure 10 shows the way of the readings of the parameters for the task were acquired. The end-effector was oriented perpendicular to the ground in all measurements in order to make it simpler to obtain new parameters. As shown in Appendix $A.2$, the position and orientation of the end-effector were used to calculate the matrix $\boldsymbol{A}$ (direction) and the displacement vector $\boldsymbol{b}$ (position).

Four states were used in order to reproduce the trajectories with the TPGMM technique as well as the proposed techniques. Figure 11 shows an example of reproductions using the parameters of trajectory 6 (line in yellow). Figure 11a shows the response trajectories of the model estimated with the arm towards the front (dashed line in black) and with the generative technique model (in green). Figure 11b shows the addition model technique response (in red). Similar to the test of the simulation previously presented, when the proposed techniques were used, an improvement was noticed over the reproduction obtained using the calculated model with the existing trajectories.

Figure 12 shows the comparison of root mean square errors for the real task between the model estimated with the initial trajectories against the one obtained



a) Generative technique.



b) Addition model technique.

Fig. 11: Trajectories reproduced using the model estimated with the trajectories of the arm towards the front and the models obtained with the proposed techniques. Demonstrated trajectory number 6 in yellow; reproduction using the model estimated with the trajectories of the arm towards the front, in black dashed line; generative technique line in green; addition model technique line in red. Lines and points in gray represent existent trajectories and task parameters.

when adding each of the trajectories (5 a 8). There is a reduction in the error using the proposed techniques for all the added trajectories; in this case, the proposed trajectories have similar errors, when incrementing each of the trajectories with the hand moved towards the back.

Figure 13 shows a comparison of the root mean square error for the real task, between the TPGMM technique estimated with the set of initial trajectories and a given added trajectory against the error obtained using the proposed techniques. The proposed techniques RMS errors shows similar or slightly greater values for the initial trajectories than the obtained with the incremented TPGMM, which is calculated with initial and incremented demonstrations. Figure 14 shows frames of two sequences of putting-on the sleeve task by the robot; Sup.) mannequin hand towards the back; Below) mannequin hand towards the front.

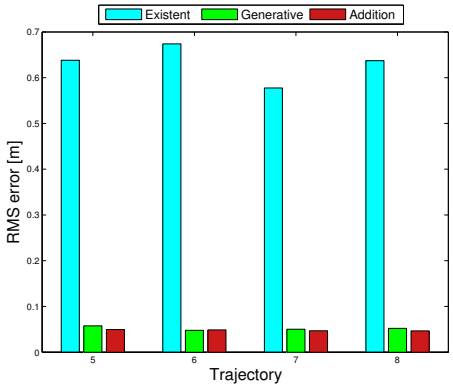


Fig. 12: Root mean square error for the real task. Comparing the existent model against the incremented models with each trajectory (5 to 8). Estimated with the trajectories of the arm to the front, in cyan. Generative technique in green. Model Addition in red.

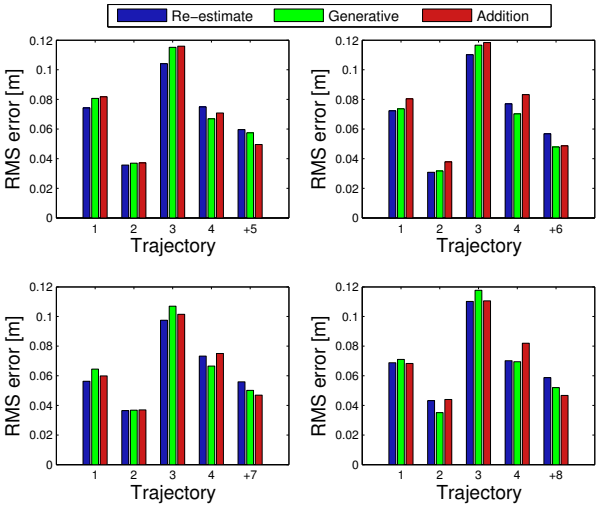


Fig. 13: RMS error for the real task and comparison of the incremented TPGMM (batch re-estimate) against the incremented proposed techniques result models. Estimated with the trajectories of the arm to the front and one to the back, in blue (batch re-estimate). Generative technique, in green. Model addition technique, in red.

Trajectories with distant task parameters

The three trajectories shown in Fig. 15 for the putting-on the sleeve task were used. Note that the parameters are far from the ones used for the hand towards the front and the hand towards the back.

In this case, the existent TPGMM is estimated with the trajectories of the hand towards the front and the generative and model addition techniques with the incrementing trajectories shown in Fig. 15. Figure 16 shows the root mean square error obtained by the TPGMM calculated with the set of trajectories having the mannequin arm to the front and one of the three trajectories shown in Fig. 15,

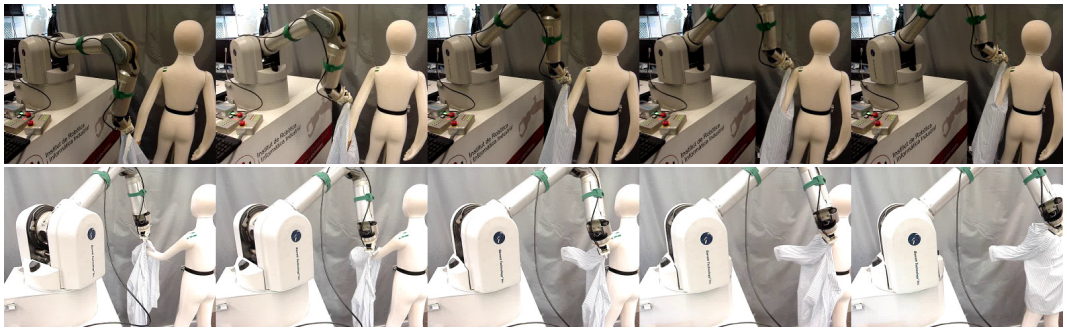


Fig. 14: Frames of two sequences on the task of putting the sleeve by the robot. Sup.) mannequin hand towards the back. Below) mannequin hand towards the front.

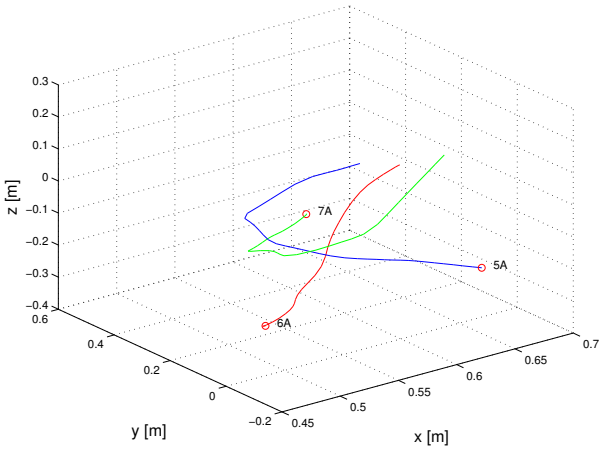


Fig. 15: Trajectories with distant task parameters.

compared to the error obtained using the proposed techniques incremented by each of the mentioned trajectories. Figure 16 shows that the generative technique is the one yielding the lowest error of the ones proposed, for the existent trajectories.

6 Discussion

Regarding the execution time spent to estimate the new model by each of the algorithms, the direct update technique takes 10 seconds in average for all tests, the generative technique takes 20 seconds in average for all tests, and the model addition technique takes in average 80 seconds for the simulated trajectories and 120 seconds for the trajectories of putting-on the sleeve. Even though the execution time of the latter is considerably high, it is shorter than the one a kinesthetic training of a robot would take including all the demonstrated trajectories if those were not known. In contrast of the results of an existent model estimated with few demonstrations, the proposed techniques achieve an average of 40% error reduction, as shown in Figures 3,12 and 16a.

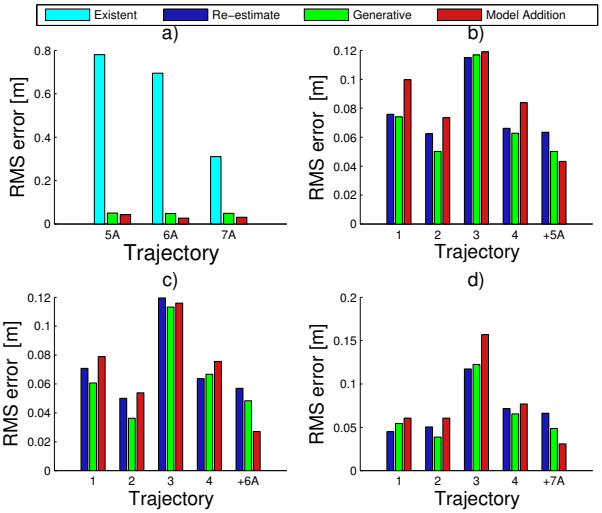


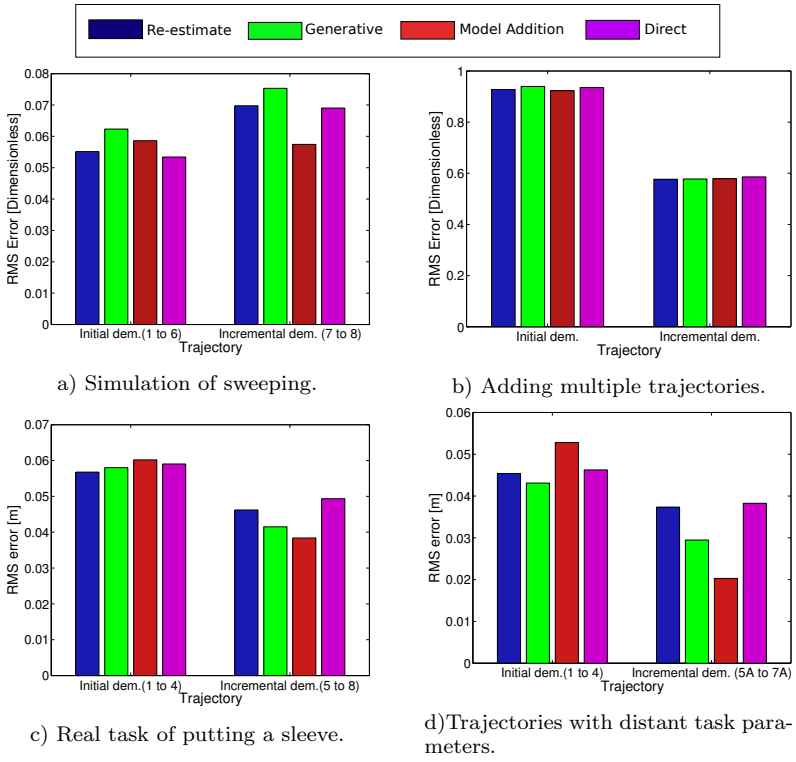
Fig. 16: RMS error for the real task using th three trajectories with distant parameters. Estimated with the arm to the front trajectories, in cyan. Estimated with the trajectories of the arm to the front and one with far parameter, in blue. Generative technique, in green. Model addition technique, in red.

Figure 17 shows the RMS bar diagrams for all tests. The results are grouped by trajectory type: initial and incremented. In the simulation sweeping test, the model addition has the best results. In the multiple trajectories and sleeve putting tests, the generative technique matches the results of the model addition technique. In the distant task parameters test, the generative technique obtained the best results. Errors resulting from the proposed techniques are generally similar or slightly greater than those of the incremented TPGMM (Fig. 18). Nevertheless, it should be taken into account that the incremented TPGMM is directly estimated with initial and incremented demonstrations. In the case of the initial demonstrations, the largest increase of RMS error is observed in the model addition and is about 9%.

The generative technique has the advantage of being faster than the addition model one, and the error is similar to that obtained with the incremented TPGMM directly estimated with the information of the trajectories. The disadvantage is that, if the generative technique is applied again on a model resulting from the addition using such technique, the error would accumulate.

The resulting trajectories of applying the the direct update technique tend to be more distant from the demonstrated ones, as was shown in Fig. 7. It is possible that at least for the TPGMM case, this effect may be due to the fact that added trajectories have a subsequent probability different from the existent trajectories, which is the mathematical basis of the direct update technique.

Using the proposed techniques with multiple trajectories yields better results than only one trajectory at a time.



a) Simulation of sweeping.

b) Adding multiple trajectories.

c) Real task of putting a sleeve.

d)Trajectories with distant task parameters.

Fig. 17: Root mean square error for all tests and the three proposed techniques. Results are grouped by trajectory type: initial and incremented. In blue with the batch re-estimate TPGMM; in green, generative technique; in red, the model addition technique; and, in magenta, the direct update technique.

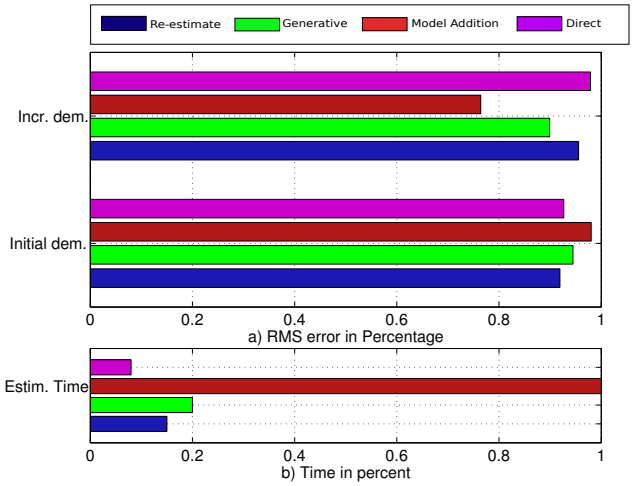


Fig. 18: General results from the proposed techniques. The RMS errors and time are normalized. In dark blue: Existing model. In green: Generative technique. In red: Model addition. In magenta: Direct update.

7 Conclusions

Three techniques, generative, model addition and direct update, were proposed to add trajectories to a task-parameterized GMM model previously estimated. Using simulated and real tasks, the performance of these techniques was tested. The best response was obtained using the proposed techniques and such response is better than the one that an existent model would produce.

During the tests, the trajectories added to estimate the incremented model may be different, which allows the new model to behave differently at the expense of a little increase in the error when reproducing past trajectories.

The proposed techniques for incremental learning of TPGMM models produced satisfactory results: RMS error reduction by 40% on average with respect to the results of using the existing model; and similar or only 9% greater errors when compared with those obtained by a batch re-estimation of the TPGMM model. Even though it takes the longest estimation time, the technique of model addition renders the best results as it allows continuous addition of trajectories. In second place, it is the generative technique that presents an acceptable compromise between estimation time and RMS errors of initial and incremented trajectories. The execution time of the three proposed techniques is similar, as they produce TPGMM models.

As a future work we plan to find techniques that reduce the time that the optimization takes in the model addition technique. Also we would like to find alternative techniques that allow to incrementally build models, using EM algorithm. In addition, we will try to improve the results of the direct update technique.

Acknowledgements This work has been partially funded by Universidad del Quindio (Colombia) and by the Spanish Ministry of Economy and Competitiveness under project Robinstruct TIN2014-58178-R, and by the CSIC project TextilRob 201550E028. The first two authors want to thank the Institut de Robòtica i Informàtica Industrial CSIC-UPC for their collaboration.

Appendix

Task parameter calculation

A.1 Simulated Trajectories case:
The calculation of $\boldsymbol{A}$ and $\boldsymbol{b}$ is carried out using two points ($\boldsymbol{p}_1$ and $\boldsymbol{p}_2$) on the Cartesian space; $\boldsymbol{p}_1$ point is related to some of the parameters of the task: starting point, object location or ending point; $\boldsymbol{p}_2$ point results from 3 samples away from point $\boldsymbol{p}_1$.

The transformation matrix $\boldsymbol{A}$ is made up of three vectors $\boldsymbol{v}_1$,$\boldsymbol{v}_2$, and $\boldsymbol{v}_3$:

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & v_{1x} & v_{2x} & v_{3x} \\ 0 & v_{1y} & v_{2y} & v_{3y} \\ 0 & v_{1z} & v_{2z} & v_{3z} \end{bmatrix}. \tag{18}$$

Vector $\boldsymbol{v}_2$ is the normal one to the difference $\boldsymbol{p}_2 - \boldsymbol{p}_1$, and vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_3$ are perpendicular to $\boldsymbol{v}_2$; $\boldsymbol{A}$ is a transformation matrix that contains information of the orientation of point $\boldsymbol{p}_1$ regarding point $\boldsymbol{p}_2$. In the case of the object location parameter, the identity matrix is used.

Table 1: $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ values, for each task parameter.

Task parameter	$\boldsymbol{p}_1$	$\boldsymbol{p}_2$
hand	hand position	elbow position
elbow	elbow position	-
shoulder	shoulder position	elbow position

The displacement vector $\boldsymbol{b}$ is formed using $\boldsymbol{p}_1$ point, as follows:

$$\boldsymbol{b} = [0, p_{1x}, p_{1y}, p_{1z}]^T \tag{19}$$

A.2 The real task case:
The calculation of $\boldsymbol{A}$ and $\boldsymbol{b}$ is carried out by using two points ($\boldsymbol{p}_1$ and $\boldsymbol{p}_2$) on the Cartesian space. The point $\boldsymbol{p}_1$ is related to any of the task parameters; Table 1 shows $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ for each case. The variable $\boldsymbol{b}$ of the task parameters is described by 8 values:

$$\boldsymbol{b} = \begin{bmatrix} t \\ x \\ y \\ z \\ qx \\ qy \\ qz \\ qw \end{bmatrix}, \tag{20}$$

where t is the time, $[x, y, z]$ is the Cartesian position and $[qx, qy, qz, qw]$ is the orientation in quaternion.

The position values $[x, y, z]$ are used to calculate $\boldsymbol{A}$:

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0\,0\,0\,0 \\ 0 & v_{1x} & v_{2x} & v_{3x} & 0\,0\,0\,0 \\ 0 & v_{1y} & v_{2y} & v_{3y} & 0\,0\,0\,0 \\ 0 & v_{1z} & v_{2z} & v_{3z} & 0\,0\,0\,0 \\ 0 & 0 & 0 & 0 & 1\,0\,0\,0 \\ 0 & 0 & 0 & 0 & 0\,1\,0\,0 \\ 0 & 0 & 0 & 0 & 0\,0\,1\,0 \\ 0 & 0 & 0 & 0 & 0\,0\,0\,1 \end{bmatrix}. \tag{21}$$

Vector $\boldsymbol{v}_2$ is the normal vector to the difference $\boldsymbol{p}_2 - \boldsymbol{p}_1$, and the vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_3$ are perpendicular to $\boldsymbol{v}_2$. $\boldsymbol{A}$ is a transformation matrix that contains information of the orientation of point $\boldsymbol{p}_1$ regarding point $\boldsymbol{p}_2$. In the case of the elbow location parameter, the identity matrix is used.

References

1. Alizadeh, T., Calinon, S., Caldwell, D.G.: Learning from demonstrations with partially observable task parameters. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA), pp. 3309 – 3314 (2014)
2. Argall, B.D., Chernova, S., Veloso, M., Browning, B.: A survey of robot learning from demonstration. Robotics and Autonomous Systems **57**, 469 – 483 (2009)
3. Billard, A., Calinon, S., Dillmann, R., Schaal, S.: Robot programming by demonstration. In: O.K. Bruno Siciliano (ed.) Handbook of Robotics, chap. 59, pp. 1371–1394. Springer, Berlin (2008)
4. Breazeal, C., Scassellati, B.: Challenges in building robots that imitate people. In: K. Dautenhahn, C.L. Nehaniv (eds.) Imitation in Animals and Artifacts, pp. 363 – 390. MIT Press (2002)
5. Calinon, S., Alizadeh, T., Caldwell, D.: On improving the extrapolation capability of task-parameterized movement models. In: IEEE International Conference on Intelligent Robots and Systems, pp. 610 – 616 (2013)
6. Calinon, S., Billard, A.: Incremental learning of gestures by imitation in a humanoid robot. In: IEEE Human Robot Interation Conference, pp. 255 – 262 (2007)
7. Calinon, S., Li, Z., Alizadeh, T., Tsagarakis, N.G., Caldwell, D.: Statistical dynamical systems for skills acquisition in humanoids. In: 8th IEEE-RAS International Conference on Humanoid Robots, pp. 323 – 329 (2012)
8. Cederborg, T., Li, M., Baranes, A., Oudeyer, P.Y.: Incremental local online gaussian mixture regression for imitation learning of multiple tasks. In: IEEE International Conference on Intelligent Robots and Systems, pp. 267 – 274 (2010)
9. Grollman, D., Jenkins, O.: Dogged learning for robots. In: IEEE International Conference on Robotics and Automation, pp. 2483–2488 (2007)
10. Hall, P., Hicks, Y.: A method to add gaussian mixture models. In: Technical Report,University of Bath, p. 18 (2004)
11. Kulic, D., Takano, W., Nakamura, Y.: Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden markov chains. The International Journal of Robotics Research **27**(7), 761 – 784 (2008)
12. Lee, D., Ott, C.: Incremental motion primitive learning by physical coaching using impedance control. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4133–4140 (2010)
13. Lee, D., Ott, C.: Incremental kinesthetic teaching of motion primitives using the motion refinement tube. In: Autonomous Robot, vol. 31, pp. 115 – 131. Springer (2011)
14. Lee, S.H., Kim, H.K., Suh, I.H.: Incremental learning of primitive skills from demonstration of a task. In: 6th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pp. 185–186 (2011)
15. Niekum, S., Chitta, S., Marthi, B., Osentoski, S., Barto, A.G.: Incremental semantically grounded learning from demonstration. In: Robotics: Science and Systems (2013)
16. Pardowitz, M., Knoop, S., Dillmann, R., Zollner, R.: Incremental learning of tasks from user demonstrations, past experiences, and vocal comments. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on **37**(2), 322–332 (2007)
17. Rozo, L., Calinon, S., Caldwell, D.: Learning force and position constraints in human-robot cooperative transportation. In: Robot and Human Interactive Communication, 2014 RO-MAN: The 23rd IEEE International Symposium on, pp. 619–624 (2014)
18. Rozo, L., Calinon, S., Caldwell, D.G., Jimenez, P., Torras, C.: Learning collaborative impedance-based robot behaviors. In: Conference on Artificial Intelligence (AAAI), pp. 1422 – 1428 (2013)
19. Townsend, W.T., Guertin, J.A.: Teleoperator slave - WAM design methodology. Industrial Robot: An International Journal **26**(3), 167–177 (1999)