SAPHARI

SAFE AND AUTONOMOUS PHYSICAL HUMAN-AWARE ROBOT INTERACTION

Project funded by the European Community's 7th Framework Programme (FP7-ICT-2011-7)
Grant Agreement ICT–287513

Deliverable D5.2.1

Intuitive motion learning and iterative motion refinement

Deliverable due date: 30 April 2014	**Actual submission date: 26 June 2014**
Start date of project: 1 November 2011	**Duration: 48 months**
Lead beneficiary for this deliverable: TUM	**Revision: Final**

Nature: R	**Dissemination Level: CO**

R = Report
P = Prototype
D = Demonstrator
O = Other
	PU = Public
	PP = Restricted to other programme participants (including the Commission Services)
	RE = Restricted to a group specified by the consortium (including the Commission Services)
	CO = Confidential, only for members of the consortium (including the Commission Services)

www.saphari.eu

Executive Summary

This deliverable of work package WP5 deals with two problems. The first problem consists in how it is possible to learn and represent new skills in a simple, natural, intuitive and compact manner. The second problem concerns the possibility of incrementally refine the learned motions to compensate changes in the environment or errors in the training phase.

Learning by Demonstration (LbD), and in particular kinesthetic teaching, is a popular topic in the robot learning literature. In the approaches, an expert provides some demonstrations, used to learn a compact and easy to generalizable representation of the task. More recently, incremental learning algorithms and their application in robotics scenarios started to be investigated with the growing need to introduce robots in dynamically changing environments.

The solution developed within SAPHARI consists in combining kinesthetic teaching, batch learning, incremental learning and a customized kinematic multi-priority control framework. The kinesthetic teaching approach provides a simple, natural and intuitive to learn and update motion primitives. The customized interaction controller makes possible the physical interaction with the robot during the motion execution. The possibility to interrupt or modify the robot motion during the execution gives the user the possibility to update the motion primitive partially, leaving the rest unchanged. Learning of movements in the null-space, which do not affect the end-effector motion, is also possible within our framework.

The class of considered problems includes also specific SAPHARI requirements such as smooth transition between free-space motion and human-robot physical interaction situations, motion primitive generation, refinement and execution, human-robot competence sharing, and so on. The effectiveness of our approach is validated on two case studies, namely a point-to-point motion learning and null-space motion learning. The learned motion primitives can be adopted, for example, in the DLR user case, where the robot is required to pick and place different tools.

The present work interfaces with the reactive action generation algorithm in WP6. In particular, it provides the symbolic stochastic models that represent the base for the cognitive multimodal reactive motion generation of T6.2 in WP6.

Table of contents

1 Introduction

In real applications, the robot is required to execute complex tasks and to adapt his behavior to a dynamic environment. Skills acquisition and their compact representation as *motion primitives*, as well as on-line adaptation of those skills to new scenarios, are of importance both in industrial and service contexts.

The skills acquisition procedure has to be simple, natural and intuitive. In this way new skills can be easily learned and also non-expert users can teach the robot how to accomplish a certain task. The possibility of modify the learned motion primitives is also required.

A natural and intuitive way that humans use to teach new skills is kinesthetic teaching, i.e. manually guide the partner during the task execution. In this document, we propose a novel approach that combines kinesthetic teaching (physical interaction), batch and incremental learning techniques and a customized multi-priority kinematic control, the so-called *Task Transition Control (TTC)*. The learning algorithms are responsible for motion primitive creation and update. The TTC makes possible the physical human-robot interaction, guaranteeing a proper and safe response to the applied external forces.

Experimental results on a KUKA LWR manipulator show the effectiveness of the proposed approach to physically teach and refine motion primitives.

2 Motion primitives learning by demonstration

Learning by Demonstration (LbD) is a powerful tool widely used in robotics for acquiring new skills for robots. LbD has the advantage of learning new skills directly from user demonstrations in a simple and intuitive way. This makes possible to avoid tedious hand programming of new tasks. Moreover, by the means of learning algorithms, the skill can be represented in a compact form reducing the amount of data to store.

LbD works in two steps. Firstly, an expert provides some demonstrations of a task to execute. There are two main ways to collect these demonstrations. The user can directly drive the robot from an initial configuration to the desired one (kinesthetic teaching). Or, the user can execute the task himself several times while some sensors track its motion and collect data. For the experiments presented in this deliverable, we used kinesthetic teaching to collect tasks demonstrations. Secondly, the demonstrations are encoded using machine learning algorithms.

Hidden Markov Models (HMM) have been widely used to encode robot's skills from demonstrations and to retrieve the desired trajectory. An HMM is described by the set of hidden states S, the set of observables output symbols O, the initial state probability π, the state transition probability matrix A and the observation symbol probability distribution B. The set of learnable parameters is usually indicated with $\lambda = (\pi, A, B)$. When, as in this case, the observations o are continuous, a common choice is to represent the observation probability distribution as a mixture of M Gaussians:

$$B_j(o) = \sum_{k=1}^{M} c_{jk} N(o, \mu_{jk}, \Sigma_{jk}) \quad .$$

The HMM parameters $\lambda = (\pi, A, c, \mu, \Sigma)$ are estimated from training data using a variation of the EM algorithm, the so-called Baum-Welch algorithm [1]. Once the motion primitive is learned, a smooth trajectory is generated using Gaussian Regression [2], as detailed in [3]. The results of the learning and generation procedures are shown in Figure 1.

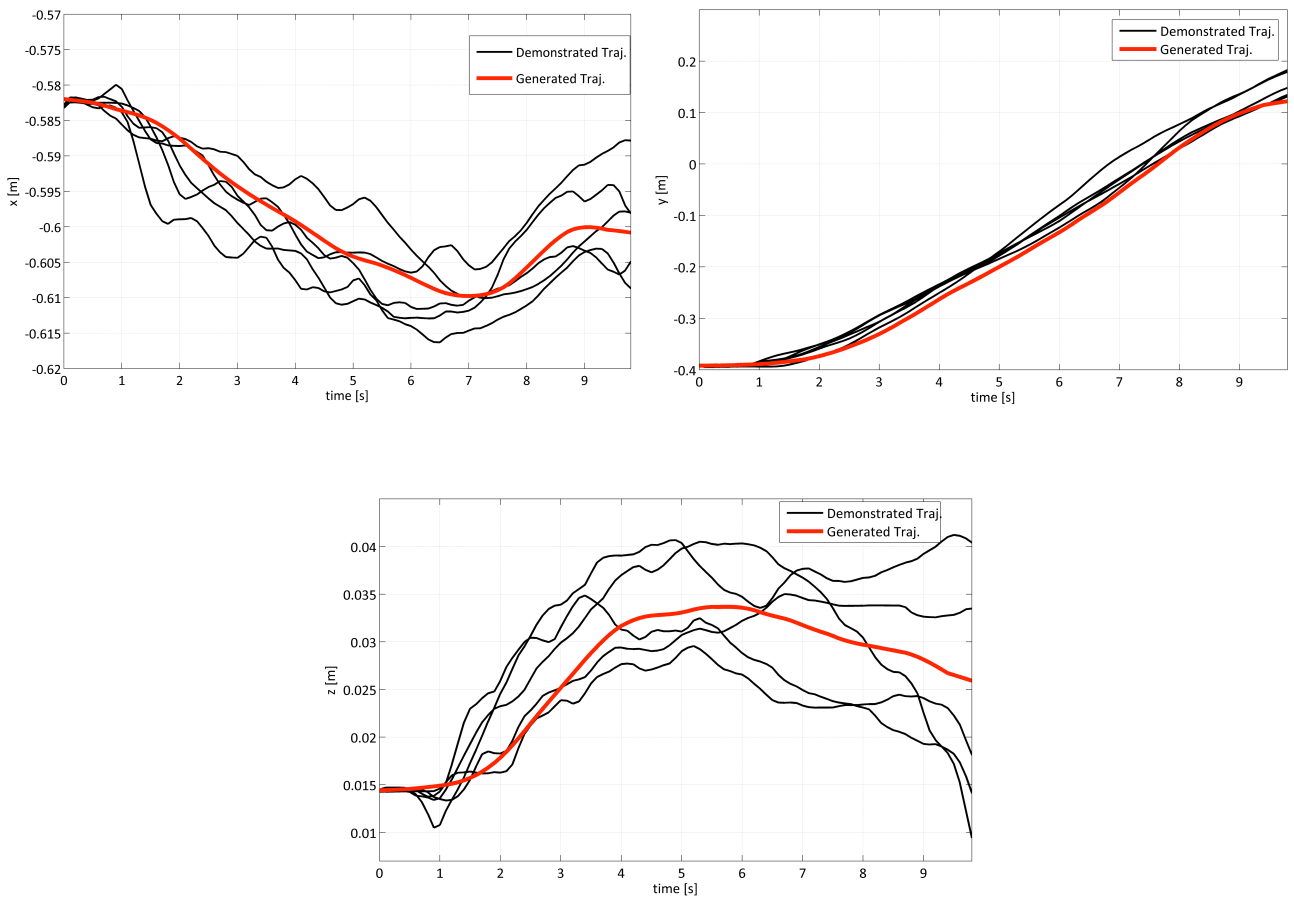

Figure 1. Motion learning from demonstrations (black lines) and smooth trajectory generation (red line) using HMM. Top-left: x direction, top-right: y direction, bottom: z direction.

3 Incremental motion refinement through kinesthetic teaching

Real scenarios are, usually, highly dynamic and the requirements for a correct task execution can change between different executions. Simple examples are changes in the desired final position or the presence of unforeseen obstacles in the robot work space. Hence, the robot is required to adapt to new scenarios during the motion execution.

Incremental learning of motion primitives consists in updating the previous knowledge of motion primitives as new demonstrations are provided, without keeping all the training data in the dataset. We decide to provide new demonstrations during the task execution by kinesthetic teaching. This has several advantages:

- The motion refinement procedure is natural and intuitive.
- The user can decide which part of the motion primitive to update, leaving the rest unchanged.
- Null-space motions can be learned without modifying the end-effector task execution thanks to the proposed customized multi-priority kinematic control (see Section 4).

One of the big limitations of many incremental learning approaches is that they become insensitive to new data when the data set becomes large. To avoid this behavior, a forgetting factor is used in the learning algorithm, as suggested in [3].

The idea is to use just two demonstrations: one is the new demonstration provided by the user, the other is the smooth trajectory generated from the current motion primitive (see Section 2). A weighting term w^d for each demonstration is given. For the new demonstration $w^d = \eta$, where η is the forgetting factor. For the generated motion trajectory is $w^d = 1 - \eta$. The new HMM parameters $\lambda = (\hat{\pi}, \hat{A}, \hat{c}, \hat{\mu}, \hat{\Sigma})$ are updated using the old ones $\lambda = (\pi, A, c, \mu, \Sigma)$ and the demonstrations as follows:

$$\hat{\pi}_i = \sum_{d=1}^{2} w^d \gamma_i^d(1)$$

$$\hat{A}_{ij} = \frac{\sum_{d=1}^{2} w^d \sum_{t=1}^{T_d-1} \xi_{ij}^d(t)}{\sum_{d=1}^{2} w^d \sum_{t=1}^{T_d-1} \gamma_i^d(t)}$$

$$\hat{c}_{ik} = \frac{\sum_{d=1}^{2} w^d \sum_{t=1}^{T_d} \gamma_{ik}^d(t)}{\sum_{d=1}^{2} w^d \sum_{t=1}^{T_d} \gamma_i^d(t)}$$

$$\hat{\mu}_{ik} = \frac{\sum_{d=1}^{2} w^d \sum_{t=1}^{T_d} \gamma_{ik}^d(t) o^d(t)}{\sum_{d=1}^{2} w^d \sum_{t=1}^{T_d} \gamma_{ik}^d(t)}$$

$$\hat{\Sigma}_{ik} = \frac{\sum_{d=1}^{2} w^d \sum_{t=1}^{T_d} \gamma_{ik}^d(t)(o^d(t) - \mu_{ik})(o^d(t) - \mu_{ik})^T}{\sum_{d=1}^{2} w^d \sum_{t=1}^{T_d} \gamma_{ik}^d(t)}$$

where $\gamma_i^d(t)$ is the probability of being in state *i* at time *t* for observation sequence $o^d(t)$, $\gamma_{ik}^d(t)$ is the probability of being in state *i* at time *t* with the *k*-th mixture component accounting for the observation $o^d(t)$, T^d is the time duration of $o^d(t)$ and $\xi_{ij}^d(t)$ is the probability of being in state *i* at time *t* and being in state *j* at time *t+1* for the sequence $o^d(t)$.

4 Task transition control for physical motion refinement

Human intervention is an essential part of the kinesthetic teaching of robots because the dexterity of the learned skills is limited by the way how human teaches robots. Also, training motions from the human demonstration eases the painful and time-consuming manual programming works. To realize the incremental motion refinement by physical contact requires a control framework that allows the robot to switch between heterogeneous tasks smoothly.

The proposed kinematic control extends the *An's method* in [4] to realize smooth transitions between prioritized tasks. We define a basic task as an unprioritized task that is given before the initial operating time. The basic task is a set of basic subtasks, which are reference motions for the robot. Stacking and assigning priorities to an arbitrary number of basic subtasks generates another set, the so-called induced task. For each induced task, there exists a prioritized inverse solution. We define a prioritized inverse solution set as a set that contains inverse solutions of all tasks. For the mathematical completion, we may define a null task that has a zero inverse solution. For example, in this work, the basic subtasks are: the desired end-effector linear and angular velocities $\dot{x}_{ee}$ and the interaction control law (an admittance control that transform the external forces into a desired velocity $\dot{x}_{ic}$). Usually, the mapping from the induced task set to the inverse solution set is surjective because there could be tasks that are equivalent to each other.

When the task definitions need to be changed during operations, smooth transitions between tasks are necessary to prevent undesirable discontinuous jumps in the joint trajectories. The existence of kinematic and algorithmic singularities and consecutive task transitions increase the complexity of the problem. Many researches have tried to find a control framework that interpolates the task trajectories for the smooth task transitions. Our idea is to interpolate the joint trajectories directly by using the barycentric coordinate, which is a popular way in the computer graphics that parameterize points inside of a convex polytope given by a set of vertices. There are two main differences in the way we use barycentric coordinates. At first, we are designing smooth transitions of the barycentric coordinates instead of calculating the coordinates of points from the vertices. Also, the inverse solutions of tasks will not always form a convex polytope. It could be a concave polytope or just a set of points in the joint space.

The non-negativity and the linearity properties of the barycentric coordinates give us the boundedness of inverse solutions, while the smoothness property is for the smooth transitions between tasks and there could be infinite number of ways to traverse between inverse solutions of tasks. This is a design problem of coordinates that produces smooth task transitions. Task transitions are triggered by the discrete input values that correspond to each induced task. For example, in this work, we use a threshold f_m to decide to insert the interaction control task $\dot{x}_{ic}$ as the lower priority task, generating extra null-space motions, and another threshold f_M to switch the tasks priorities. In other word, the robot firstly tries to project $\dot{x}_{ic}$ in its null-space. If the human perceives that the task is not correctly executed, he simply applies a bigger force until $\dot{x}_{ic}$ becomes the first priority task.

Smooth task transitions are guaranteed by a linear dynamical system that transforms the discrete trigger (system reference) into a continuous signal (system output). The convergence behavior can be tuned by the stabilizing control gains that do not generate overshoot of the linear dynamic system.

5 Case study

The proposed approach is tested on a KUKA LWR (7 degree-of-freedom) manipulator. The goal is to learn and incrementally refine a point-to-point motion. In particular, in the experiment the final end-effector position is gradually updated to match the desired one. In the second experiment, a null-space motion is learned to avoid an obstacle, without affecting the end-effector motion.

5.1 External forces estimation

The task transition control in Section 4 transforms the external Cartesian force in a desired velocity. Hence, an estimation of the external Cartesian force applied to the robot is needed. The estimation of the external force requires to steps. Firstly, the external torque τ_e, applied in each joint, must be estimated. Secondly, given the contact point *C*, the external force f_e is computed inverting the well-known equation $\tau_e = J_C^T f_e$, where J_C is the contact point Jacobian.

Approaches have been already developed to estimate the external torque and the contact link. For example in [5] a momentum based disturbance observer is used for collision detection and reaction. A similar approach is used within the SAPHARI project to balance a humanoid robot during kinesthetic teaching [6]. For the KUKA LWR manipulator, an estimation of the external joint torque is provided at 1 KHz through the *Fast Research Interface* [7].

Contact point estimation is still an open problem, also investigated in the SAPHARI project. The solution provided in WP3 makes use of exteroceptive sensors (RGB-D cameras). Since the estimation of the contact point is behind the scopes of this document, we simply assume that the contact always occur at the end of the contact link (identified as in [5]).

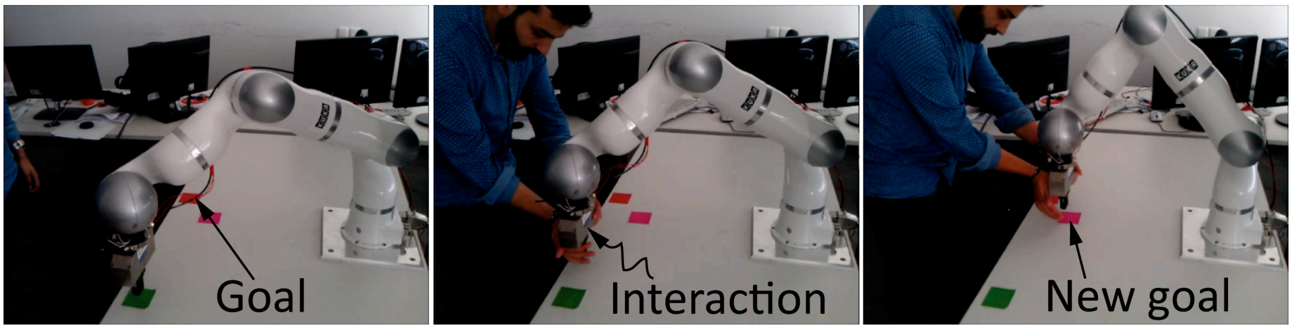

Figure 2. The original goal position is changed on-line using kinesthetic teaching.

5.2 Results

Learn a new goal position – In this experiment we show how the goal position the robot reaches can be changed online. The robot has to execute the point-to-point motion depicted in Figure 1. During the task execution, user provides new demonstrations by physically guiding the robot in a new goal position. The overall procedure is depicted in Figure 2, where the red and pink square indicated the original and new goal positions respectively. After four iterations, the refined trajectory converges to the new goal, as shown in Figure 3.

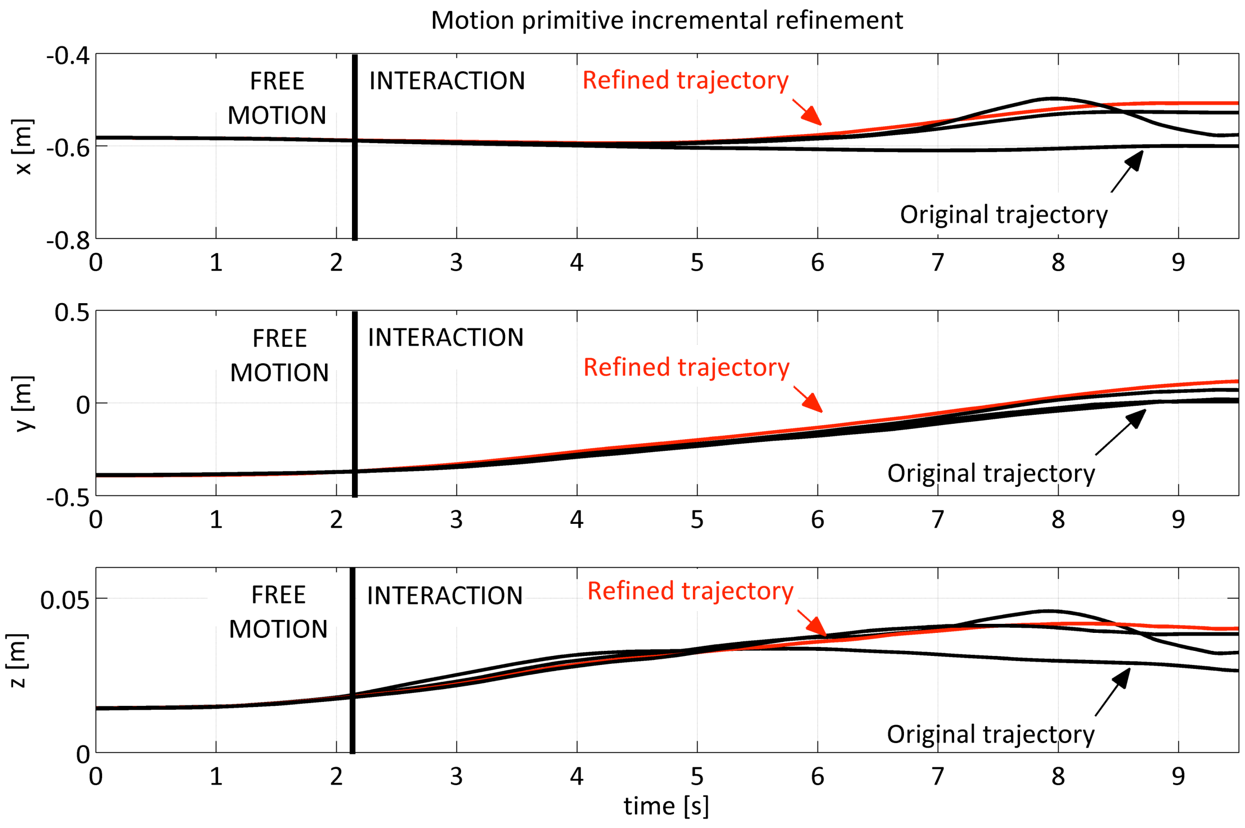

Figure 3. The motion primitive is incrementally refined providing new demonstrations. After four iterations the refined trajectory reaches the new goal position.

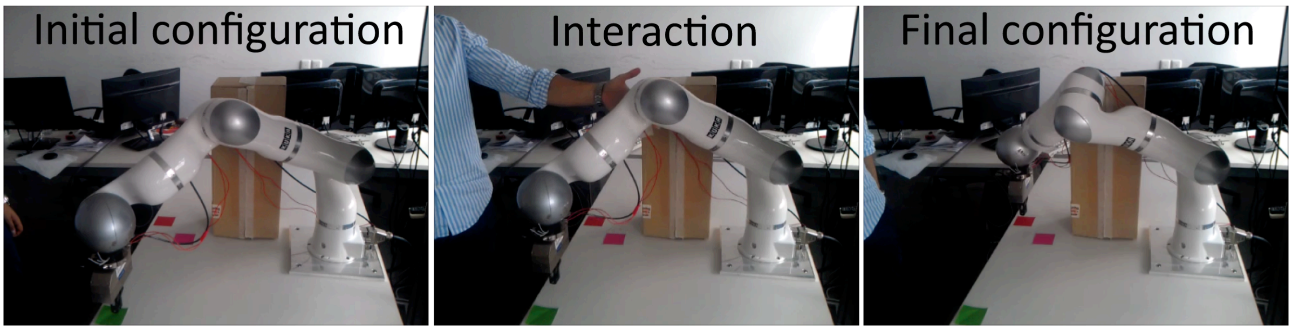

Figure 4. The elbow trajectory is learned on-line to avoid an unforeseen obstacle. The resulting motion is projected in the Jacobian null-space to not affect the end-effector task.

Learn null-space motions – In this experiment we show how to learn null-space motions using the proposed approach. As in the previous experiment, the robot has to execute, as a first priority task, the point-to-point motion in Figure 1. During the execution, user physically moves away the elbow from an unforeseen obstacle. Having sufficient remaining degree-of-freedom, the elbow motion is correctly executed in the Jacobian null-space, without affecting the end-effector motion. In this way it is possible to collect some demonstrations of the null-space task and learn another motion primitive (in this case for the elbow motion), as shown in Figure 5. Finally, the learned motion primitive can be inserted in the stack of tasks priorities and executed together with the point-to-point motion.

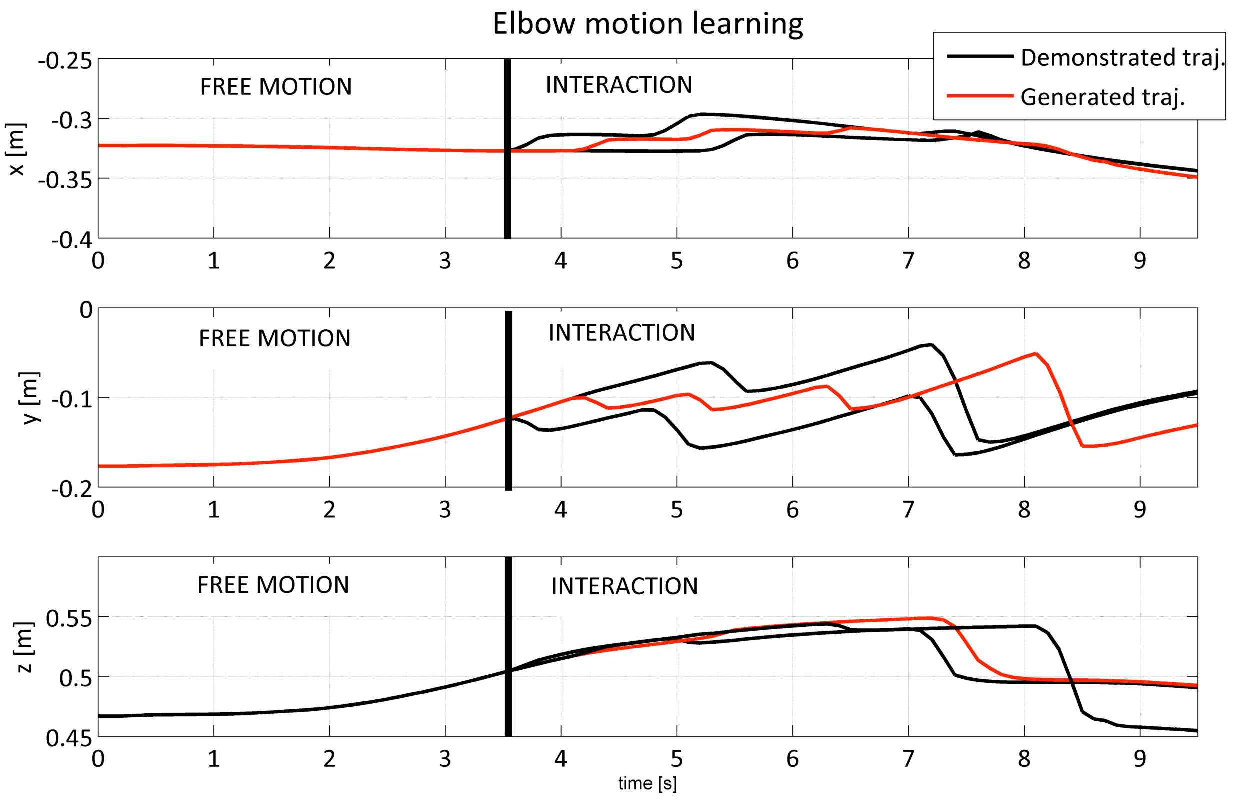

Figure 5. Elbow motion learning from demonstrations (black lines) and smooth trajectory generation (red line) using HMM. The demonstrations are collected physically interacting with the robot during the first priority task execution.

References

[1] L. R. Rabiner, *A tutorial on hidden Markov models and selected applications in speech recognition*, Proceedings of the IEEE, vol. 77, no. 2, pp. 257-286, 1989.

[2] D. A. Cohn, Z. Ghahramani and M. I. Jordan, *Active Learning with Statistical Models, Journal of Artificial Intelligence Research, vol. 4, no. 1, pp.* 129-145, 1996

[3] D. Lee and C. Ott, *Incremental Kinesthetic Teaching of Motion Primitives Using the Motion Refinement Tube*, Autonomous Robots, vol. 31 , no. 2, pp. 115-131, 2011.

[4] S. An and D. Lee, *Prioritized Inverse Kinematics using QR and Cholesky Decompositions*. IEEE International Conference on Robotics and Automation (ICRA), 2014.

[5] A. De Luca, A. Albu-Schäffer, S. Haddadin, and G. Hirzinger, “Collision detection and safe reaction with the dlr-iii lightweight manipulator arm,” in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1623-1630, 2006.

[6] C. Ott, B. Henze and D. Lee, *Kinesthetic teaching of humanoid motion based on whole-body compliance control with interaction-aware balancing*, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4615-4621, 2013.

[7] G. Schreiber, A. Stemmer, and R. Bischoff, *The fast research interface for the Kuka lightweight robot*, IEEE ICRA Workshop on Innovative Robot Control Architectures for Demanding (Research) Applications, pp. 15–21, 2010.