Learning Manipulation by Sequencing Motor Primitives with a Two-Armed Robot

Rudolf Lioutikov, Oliver Kroemer, Guilherme Maeda and Jan Peters

Technische Universitaet Darmstadt, Intelligent Autonomous Systems
Hochschulstr. 10, 64289 Darmstadt, Germany
{lioutikov, kroemer, maeda, peters}@ias.tu-darmstadt.de

Abstract. Learning to perform complex tasks out of a sequence of simple small demonstrations is a key ability for more flexible robots. In this paper, we present a system that allows for the acquisition of such task executions based on dynamical movement primitives (DMPs). DMPs are a successful approach to encode and generalize robot movements. However, current applications involving DMPs mainly explore movements that, although challenging in terms of dexterity and dimensionality, usually comprise a single continuous movement. This article describes the implementation of a novel system that allows sequencing of simple demonstrations, each one encoded by its own DMP, to achieve a bimanual manipulation task that is too complex to be demonstrated with a single teaching action. As the experimental results show, the resulting system can successfully accomplish a sequenced task of grasping, placing and cutting a vegetable using a setup of a bimanual robot.

1 Introduction

With further advances in the field of autonomous robotics the role of imitation learning has become increasingly important. A state-of-the-art approach to skill learning from demonstrations is the usage of dynamical movement primitives (DMPs). While DMPs have been successfully applied to a multitude of both discrete (point-to-point) [1, 2] and rhythmic (cyclic) movements [3, 4], the possibility to sequence various DMPs to achieve a more complex task has not received as much attention. Example tasks for discrete DMPs include the ball-in-a-cup game [1] and hitting a ball [2]. Where rhythmic DMPs have been applied to play drum [3] and to walk [4]. While those examples are complex in nature, the generalization properties of DMPs allows them to encode these tasks by a single primitive.

However, these properties also enable the sequencing of multiple DMPs in order to achieve challenging multi-step tasks which can not be represented by a single primitive. Examples of such multi-step activities include difficult preparation and assembly tasks, which additionally take place in constantly changing environments. Therefore the ability to generalize and adapt those tasks becomes an important aspect of their execution. Such flexibility is not given by encoding

the entire task as a single primitive. This motivates the need for sequences of DMPs.

Breaking the task down into multiple steps allows for the generalization of each steps and therefore increases the adaptability of the entire task significantly. At the same time the teaching of the single steps is easier and more robust with respect to complications during the teaching than the attempt to teach the entire task at once. By using DMPs for each step it is possible to encode each step as a single primitive regardless of discontinuities between the demonstrations. This approach is illustrated in Fig. 1, where a sequence of three different demonstrations (in gray), each corresponding to one step, is used to achieve a multi-step task. Note that the primitives obtained from the demonstration do not create a continuous chain (in gray). However, continuity can be enforced by generalizing DMPs to different initial and final conditions such that the resulting movement is feasible (in red).

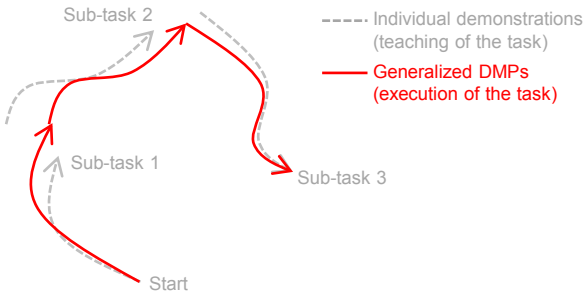


Fig. 1. Sequence of demonstrated steps where each step is encoded by a DMP.

Since bimanual manipulation is one of the key elements to have robots autonomously accomplishing complex tasks, we also motivate this work by a task that requires the use of two manipulators. As illustrated in Fig. 2 we propose teaching both arms with independent sequences of movement primitives, while still demaning a colaborations between the manipulators in order to successfully complete the task.

The contribution of this work is to exploit the inherent characteristics of DMPs for the sequencing of multi-step and bimanual tasks. Additionally a modular system architecture is presented which combines the widespread ROS framework with the real-time Simulation Laboratory (SL). The DMP-based bimanual manipulation is evaluated on an advanced robotic system in a vegetable cutting task. The experiments show that, DMPs become a natural choice for bimanual manipulation due to their inherent generalization and time scaling capabilities.

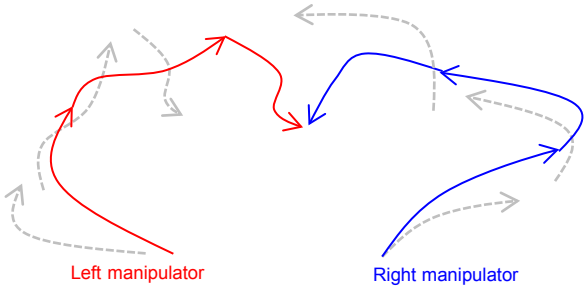


Fig. 2. Illustration of a bimanual task, by using a sequence of DMPs.

2 Related Work

In this work the sequencing of DMPs is proposed in order to achieve complex tasks. In particular we focus on bimanual manipulation, where each primitive finishes with zero velocity, which is the appropriate boundary condition for movement representation with DMPs.

Certainly, for applications outside the scope of manipulation, zero velocity at the transition of primitives can be restrictive and modifications have been proposed. A simple modification presented in [5] is to start the next primitive before the previous one finishes by taking advantage that it is possible to specify the initial velocity condition for the next DMP. In [6], the authors propose a modification of the original DMP for hitting and batting such that the final velocity of a DMP can be specified. Although the original in [6] did not aim at sequencing of primitives, it could be adopted as an extension of this paper to address transitions with non-zero velocity. An approach for sequencing DMPs based on overlapping the last and first kernels of subsequent primitives is presented in [7], which guarantees smooth transitions in position and velocity of the trajectory.

In general, the use of DMPs for bimanual manipulation has been much less explored when compared to single-arm manipulation tasks. In [8], a dual-arm system based on DMPs was used to learn pool stroke movements and movements to flip a box using a pair of chopsticks. The demonstration was made by kinesthetic teaching where the teacher taught both hands at the same time. The left and right arm swing movements for maintaining the robot balance during walking were also learned by DMP in [9] where the demonstration was obtained by motion capture from real human walking. DMPs were also applied to make a robot lift a box with both hands [10] with their motions being coupled by virtual springs. Regarding DMP use in dual-am tasks, this work differs from the previously cited works as the demonstrations are independent, such that each arm executes its own sequence of canonical systems.

3 Dynamical movement primitives

Dynamical movement primitives (DMPs) represent movements in the form of a dynamic system

$$\begin{aligned} \tau \dot{z} &= \alpha_z[\beta_z(g-y) - z] + f(x) \\ \tau \dot{y} &= z \end{aligned} \tag{1}$$

where τ is a temporal scaling variable that controls the duration of the movement, α_z and β_z are positive constants given as design parameters, and $f(x)$ is a forcing function. When $f(x)$ is zero, the movement described by (1) is that of a critically damped, linear spring-damper system with stiffness $\alpha_z\beta_z$ and damping α_z whose equilibrium point is the goal g.

The forcing function is usually represented as a linear combination of N pre-defined basis functions ψ_i as

$$f(x) = \frac{\sum_{i=1}^{N} \psi_i(x)}{\sum_{i=1}^{N} \psi_i(x)} w_i x, \tag{2}$$

where w_i are weights that can be regressed from a given demonstration. The function $f(x)$ represents a force that acts on the linear spring-damper system, which allows to represent arbitary trajectories.

The phase variable x acts as a replacement for time and synchronizes the movements. It is given by the canonical system

$$\tau \dot{x} = -\alpha_x x. \tag{3}$$

The canonical system indexes the activation of the basis functions and always starts with 1.

As usual, this work adopts normalized Gaussian kernels with centers c_i and widths determined by h_i

$$\psi_i(x) = \frac{\exp[-h_i(x-c_i)^2]}{\sum_{i=1}^{N} \exp[-h_i(x-c_i)^2]}. \tag{4}$$

Given an observed movement of duration T, for example as a demonstrated movement via kinesthetic teaching, the required forces f_{demo} can be computed from (1) as

$$f_{\text{demo}} = \tau^2 \ddot{y}_{\text{demo}} - \alpha_z[\beta_z(g - y_{\text{demo}}) - \tau \dot{y}_{\text{demo}}]. \tag{5}$$

The weights w_i in (2) can then be regressed using least square estimation

$$w_i = \frac{\mathbf{x}^T \Psi_i \mathbf{f}_{\text{target}}}{\mathbf{x}^T \Psi_i \mathbf{x}}, \tag{6}$$

where $\mathbf{x} = [x_0, ..., x_T]^T$, $\mathbf{f}_{\text{demo}} = [f_{\text{demo},0}, ..., f_{\text{demo},T}]^T$, and $\Psi_i = \text{diag}([\psi_0, ..., \psi_T])$. For a thorough and recent review on DMPs refer to [11].

4 System Architecture

In addition to the evaluation of DMP sequences, we briefly introduce a system which enables the fast demonstration, learning, and execution of DMPs. We want the possibility to easily integrate the work of others and access a variety of different features. For these reasons we chose to make use of an existing system. A framework which proved to be a reliable and widely accepted choice is ROS. ROS offers a large variety of tools and libraries and even supports multiple languages. These features make it an attractive choice for multiple areas of robotics. The goal of performing highly dynamic tasks, in real-time on robots demands performance critical, reliable, high-frequency control. A system which was successfully applied on a multitude of such tasks is the Simulation Laboratory (SL). SL is a simulation as well as a controller for the real robot. Additionally it offers various functionalities which facilitate the work on robots significantly.

In order to benefit from both systems we introduced a new system, entitled SL_robcom , which connects ROS and SL. Simplified SL_robcom can be summarized as a framework to send UDP based commands to SL, including a convenient API for clients and a corresponding server in SL. In this setup we distribute the software over two computers. The first computer, SL-PC, runs SL and functions as a real-time controller for the real robot. SL exchanges motor commands and sensor information with the real robot on a 1kHz basis. Therefore the task loop of SL is subject to strict real-time constraints. The second computer, ROS-PC, runs ROS and executes additional, potentially resource heavy applications. A simplified illustration of the architecture is shown in Figure 3.

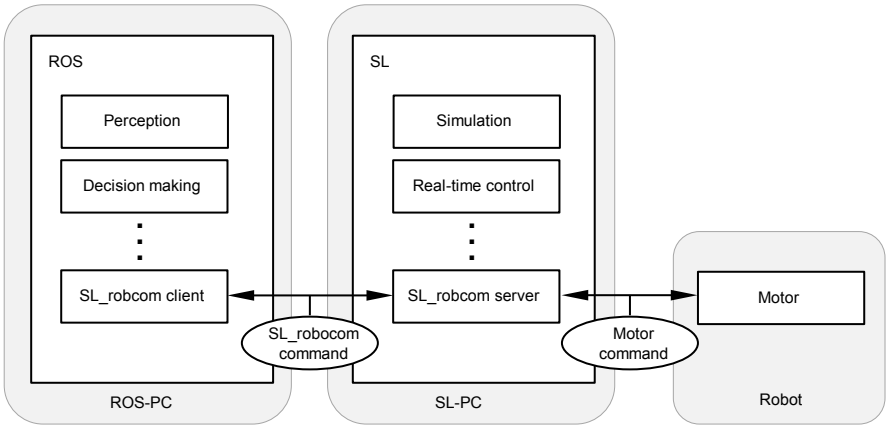


Fig. 3. Simplified illustration of the system architecture applied in this work.

In order to not violate the real time constraints, the server receives the messages in a separate thread. The message is passed to the SL task loop after all

packages were successfully received and the message was parsed. Meanwhile the robot remains in an idle state. After the command was executed, the robot returns to the idle state. Alternatively, if interrupted by another command, the robot starts executing the new command. The ROS-PC is running a node, which uses the client API of SL_robcom to communicate with SL while using various ROS packages. Given this setup we can now use the modularity and the various features of ROS without endangering the real-time constraints on the SL-PC.

SL_robcom is not limited to the communication between ROS and SL, but offers a client API which can be accessed without ROS, e.g. directly via C++ and Matlab.

5 Experimental Setup

We evaluate the sequencing of DMPs on a complex multi-step cutting task. The task is performed by a bimanual humanoid platform and was executed after a single demonstration of each sequence.

5.1 Hardware

For this task we used the Darias robot platform, which is shown in Figure 4. It consists of a dual-arm setup based on the third generation Kuka Light-Weight Robot arm with 7 DoF each. Each arm has a five-finger DLR-HIT Hand II with 15 DoF each, i.e. three per finger. During experiments, low-gain position-based impedance control running at 1 kHz was adopted. The position and orientation of the cutting board were tracked using am optitrack motion capture system.

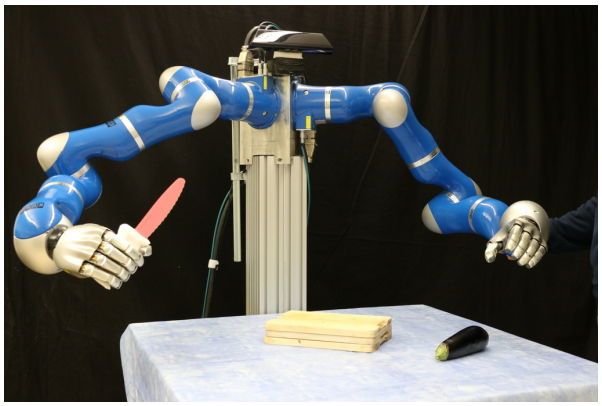
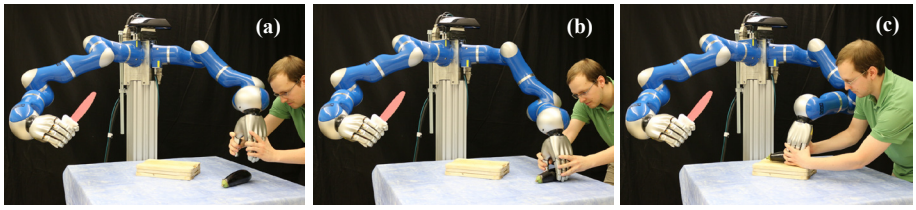
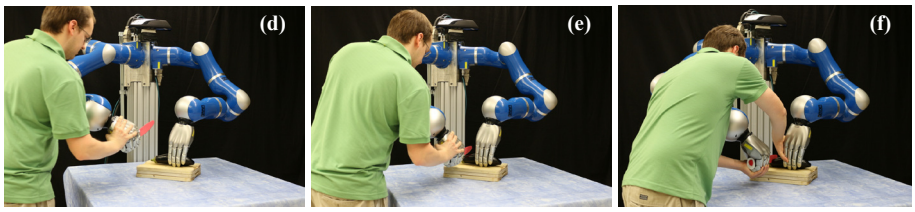


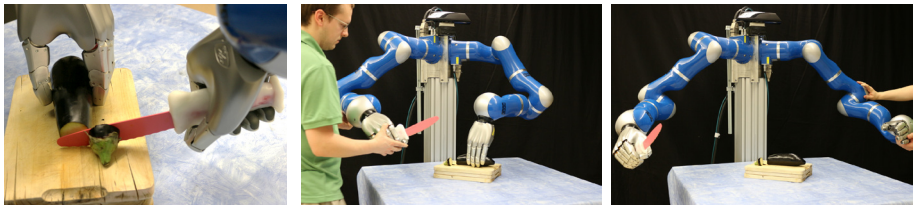
Fig. 4. The two-arm, two-hand robotic platform used for bimanual manipulation.



(i) Kinesthetic teaching of the left arm.



(ii) Kinesthetic teaching of the right arm.



(iii) the taught cut and the demonstration of the final movement

Fig. 5. The demonstration of the task. The top row (i) shows the demonstration of the reaching and the grasping of the eggplant, as well as the position on the cutting board. In the middle row (ii) the knife is lead to the eggplant and a subsequent cutting motion is taught. The bottom row (iii) shows the final cut as well as the demonstration of the final position.

5.2 Cutting Task

The robot is supposed to grasp an eggplant, place it and keep it on a tracked cutting board with his left hand, while his right hand holds the knife, positions it over the eggplant and cuts the eggplant. The task is divided into multiple steps, where each step is first demonstrated to the robot by kinesthetic teaching and afterwards executed autonomously as a sequence of learned DMPs. During the autonomous execution, both the cutting board and the eggplant were placed on different positions than during the teaching. In this manner we demonstrate the generalizing ability of DMPs and the resulting adaptability of the complete task. For each step a set of 22 DMPs are learned for each arm. The first seven encode the position and the orientation of the end effector, where the position is given as X,Y,Z coordinates and the orientation as an quaternion. The remaining 15 DMPs represent the joint position for the fingers. During the generalization

of the DMPs it is not granted that the quaternions stay normalized, therefore it is important to normalize the resulting quaternions before applying them.

For safety reasons, the task starts with the robot already holding the knife in his right hand.

5.3 Teaching

The demonstrations of the single steps are performed independently by kinesthetic teaching, shown in Figure 5. During the teaching, both arms remained in gravity compensation mode. Starting from the initial posture, the left arm approaches the eggplant and grasps it. Afterwards, the eggplant is picked up and positioned relative to the cutting board, Figure 5(i). Next the right arm positions the knife over the eggplant and, subsequently, performs the cut, as shown in Figure 5(ii). After the successful cut, the left hand releases the eggplant and both arms move back to their final position, Figure 5(iii).

The task is initialized by sending a command from the ROS-PC to the SL-PC which orders the robot to go into the initial posture. In addition a task space for the single steps can be defined, i.e. setting the goal relative to the object. For each subsequent demonstration, a corresponding command is sent from the ROS-PC to the SL-PC, which transfers the robot into gravity compensation mode and starts the recording of the joint positions and the end effectors positions and orientations in task space. Each step is demonstrated within a time frame from 10 to 18 seconds. After each demonstration the recorded data is returned from the SL-PC to the ROS-PC where the data is processed and used to learn the DMP weights, while at the same time the next step on the real robot can be demonstrated. Once all weights have been learned the user at the ROS-PC can send the weights as part of the corresponding command to the SL-PC in order to execute the learned DMP on the real robot. Again data can be recorded and sent back in order to compare the demonstrated and the executed trajectory.

5.4 Results

In order to show the generalizing ability of DMPs and the resulting adaptability of a sequence of DMPs, we do not simply reproduce the learned trajectories. Instead we increase the execution speed by a factor of about two, which leads to an 8 second execution per step. Furthermore, we change the position of the eggplant as well as the position of the cutting board. The grasps are defined relative to them, and are therefore adopted automatically. In Figure 6 both the demonstrated and the executed end effector trajectories are shown. Even though the positions of the eggplant and the cutting board changed, the robot was able to perform similar motions which resulted in the successful execution of the task. The annotations (a)...(f) and (a')...(f') correspond to different steps and positions during the demonstration and the execution of the task. Corresponding pictures of the actual robot platform can be found in Figure 5 and Figure 7. It is also notable that the executed trajectories are smoother than the demonstrations. This effect occurs due to the clean up effect of the DMPs.

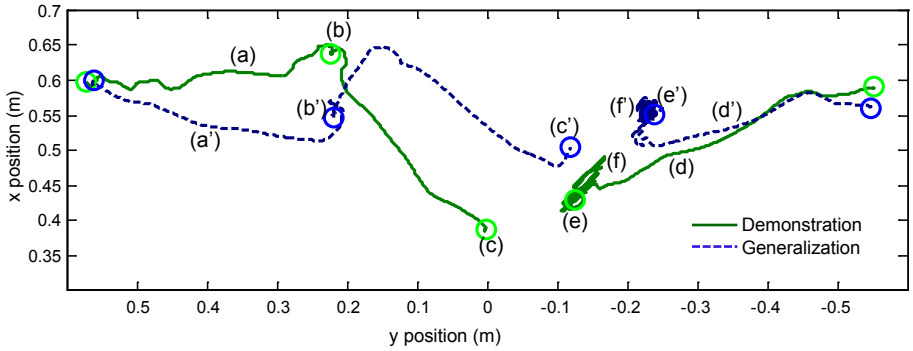
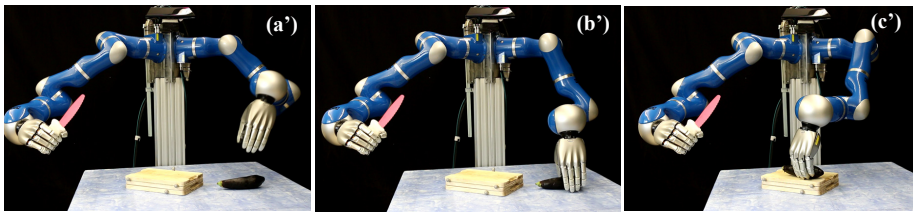


Fig. 6. Trajectories of the demonstrated (green) and generalized (blue) motions projected on the XY-plane. The x and y axes are arranged accordingly to the perspective of the robot. The annotations (a)...(f) correspond to the steps and position depicted in Figure5(a)...(f). The annotations (a')...(f') relate analogously to Figure7.

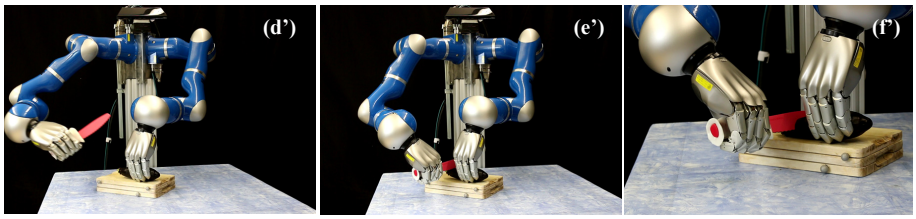
The magnitude of this effect depends on the number of Gaussian kernels as well as their bandwidth. A higher number as well as a smaller bandwidth lead to a less expressive effect.

During the execution of the task it was exposed that the taught cutting motion, if only executed once, was not sufficient to cut through the eggplant. Instead the cutting motion had to be executed at least three times. The forces applied by the human during the demonstrations are currently not learned and therefore the robot, cannot reach the desired position in one execution. In addition, the position of the knife in the hand varies between demonstration and execution. Since the cutting motion was trained with one specific posture of the knife a different posture makes the demonstration less effective. A solution to the later problem would be to track the knife and the learn the cutting motion with respect to the knifes posture instead of the hand posture. Thus, no matter the position of the knife inside the hand the relation between knife and eggplant would be maintained. Further improvement of the cutting could be achieved by representing the motion as a rhythmic DMP, which is executed until the cutting is completed.

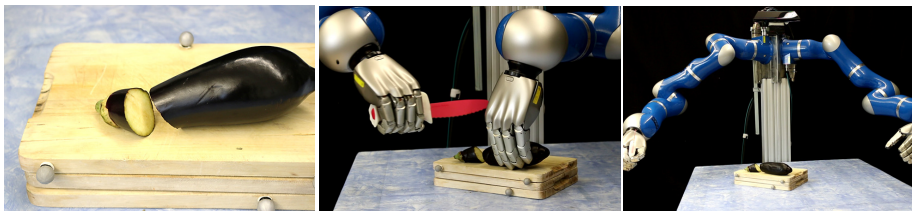
A limitation in the current robot-setup is, that it is impossible to teach forces to the robot. This means that once the robot reaches the desired configuration it will not exert any forces upon the object. For deformable objects like the eggplant, this problem can be circumvented by teaching a grasp that deforms the object slightly. For non-deformable objects, like the knife, the taught configuration needs to be modified manually, such that the attempt to reach this position will create enough force to keep the object in a stable position. Instead of relearning the weights or changing the desired goal state directly, we define those modifications as offsets in the goal state. These offsets can be passed to the command, which incorporates them into the target configuration, before the command is sent to the SL-PC.



(i) Execution of the generalized steps of the left arm.



(ii) Execution of the generalized steps of the right arm.



(iii) The executed cut and movement to the final position.

Fig. 7. The execution of the task. The top row (i) shows the autonomous execution of the learned DMPs, generalized with respect to the position of the eggplant and the cutting board. In the middle row (ii) the knife is positioned above the eggplant and the cutting motion is executed. The bottom row (iii) shows the executed cut as well as the movement to the final position.

6 Conclusion

This work presented the implementation of the DMP framework to learn and generalize tasks represented as a sequence of primitives on a dual-arm robotic system. The sequencing allowed the execution of a cutting task that can be naturally demonstrated as a concatenation of movements associated to the of sub-goals of the task. Also, our framework allows each arm to be taught and executed independently of each other. The system architecture comprised the integration of the widely used robotics framework ROS with the real-time simulation and control execution capabilities of SL. Our implementation allows the seamless demonstration of a sequence of sub-tasks while learning DMP weights for a subsequently generalized execution of the task. The SL_robcom system will be made publicly available in the near future.

Acknowledgment

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7-ICT-2013-10) under grant agreement 610878 (3rdHand).

References

1. Kober, J., Mohler, B., Peters, J.: Learning perceptual coupling for motor primitives. In: Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on, IEEE (2008) 834–839
2. Mülling, K., Kober, J., Kroemer, O., Peters, J.: Learning to select and generalize striking movements in robot table tennis. The International Journal of Robotics Research **32**(3) (2013) 263–279
3. Schaal, S.: Dynamic movement primitives-a framework for motor control in humans and humanoid robotics. In: Adaptive Motion of Animals and Machines. Springer (2006) 261–280
4. Nakanishi, J., Morimoto, J., Endo, G., Cheng, G., Schaal, S., Kawato, M.: Learning from demonstration and adaptation of biped locomotion. Robotics and Autonomous Systems **47**(2) (2004) 79–91
5. Pastor, P., Hoffmann, H., Asfour, T., Schaal, S.: Learning and generalization of motor skills by learning from demonstration. In: Proceedings of the 2009 IEEE International Conference on Robotics and Automation. (2009) 763–768
6. Kober, J., Mulling, K., Kromer, O., Lampert, C., Scholkopf, B., Peters, J.: Movement templates for learning of hitting and batting. In: Proceedings of the 2010 IEEE International Conference on Robotics and Automation, IEEE (2010) 853–858
7. Kulvicius, T., Ning, K., Tamosiunaite, M., Worgotter, F.: Joining movement sequences: Modified dynamic movement primitives for robotics applications exemplified on handwriting. Robotics, IEEE Transactions on **28**(1) (2012) 145–157
8. Pastor, P., Kalakrishnan, M., Chitta, S., Theodorou, E., Schaal, S.: Skill learning and task outcome prediction for manipulation. In: Robotics and Automation (ICRA), 2011 IEEE International Conference on, IEEE (2011) 3828–3834
9. Matsubara, T., Hyon, S.H., Morimoto, J.: Learning parametric dynamic movement primitives from multiple demonstrations. Neural Networks **24**(5) (2011) 493–500
10. Gams, A., Nemec, B., Zlajpah, L., Wachter, M., Ijspeert, A., Asfour, T., Ude, A.: Modulation of motor primitives using force feedback: Interaction with the environment and bimanual tasks. In: Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on, IEEE (2013) 5629–5635
11. Ijspeert, A.J., Nakanishi, J., Hoffmann, H., Pastor, P., Schaal, S.: Dynamical movement primitives: learning attractor models for motor behaviors. Neural computation **25**(2) (2013) 328–373