INTEGRATING SIMULATION WITH ROBOTIC LEARNING FROM DEMONSTRATION

Anat Hershkovitz Cohen and Sigal Berman
Department of Industrial Engineering and Management
Ben-Gurion University of the Negev
POB 653, Beer-Sheva, Israel
E-mail: anhe@post.bgu.ac.il

KEYWORDS

Dynamic Motion Primitives, Reinforcement Learning, Simulation, Robotics.

ABSTRACT

Robots that co-habitat an environment with humans, e.g., in a domestic or an agricultural environment, must be capable of learning task related information from people who are not skilled in robotics. Learning from demonstration (LfD) offers a natural way for such communication. Learning motion primitives based on the demonstrated trajectories facilitate robustness to dynamic changes in the environment and task. Yet since the robot and human operator typically differ, a phase of autonomous learning is needed for optimizing the robotic motion. Autonomous learning using the physical hardware is costly and time consuming. Thus finding ways to minimize this learning time is of importance. In the current paper we investigate the contribution of integrating an intermediate stage of learning using simulation, after LfD and before learning using robotic hardware. We use dynamic motion primitives for motion planning, and optimize their learned parameters using the PI^2 algorithm which is based on reinforcement learning. We implemented the method for reach-to-grasp motion for harvesting an artificial apple. Our results show learning using simulation drastically improves the robotic paths and that for reach-to-grasp motion such a stage may eliminate the need for learning using physical hardware. Future research will test the method for motion that requires interaction with the environment.

INTRODUCTION

Robots and humans are starting to share common workspaces, where the robot operators are not necessarily skilled in robotics. Such shared workplaces are appearing in industry, agriculture, the medical establishment, and domestic environments. Methods for natural interaction between the human and the robot are a crucial component in such scenarios. It is additionally advantageous for such robotic co-workers to have motion characteristics that bear similarities to human motion as this can aid robot acceptance, cooperation, and safety.

In the agricultural environment, robot penetration is expected to increase considerably in the current decade (Forge and Blackman 2011). Selective harvesting of high-value crops, e.g., apples and peppers, is an appealing task for automation using robotic systems (Foglia and Reina 2006). Selective harvesting is seasonal, tedious, only ripe fruit should be picked, and fruit position and orientation on the branch are random. Selective harvesting requires dexterous manipulation capabilities so plant, fruit, and future fruit growth are not damaged (Allota et al. 1990).

In Learning from Demonstration (LfD), also termed programming by demonstration (PbD) or imitation learning, the robot learns from demonstrations of a teacher (Mataric 2007; Koenig et al. 2010). For teaching the robot what paths to follow and how, LfD can replace traditional robot programming methods which are tedious and require specialized knowledge (Argall et al. 2009). LfD can ease the deployment of robots for selective harvesting by alleviating the effort required from the farmer while enhancing the similarity of the robotic and human motion.

Motion primitives are simple motion elements that can be concatenated serially or in parallel. Many studies show that voluntary motion in both vertebrates and invertebrates is composed of such elements (Flash and Hochner 2005). Demonstrated trajectories can be used for learning motion primitives rather than just the direct path representation, to facilitate learning generalization and thus robustness to dynamic changes in the environment and task during execution. Dynamic Movement Primitives (DMPs) encode motion control parameters using nonlinear differential equations, where equation parameters can be learned from demonstration (Ijspeert et al. 2002; Tamosiunaite et al. 2011; Kulvicius et al. 2012). The control equations of a DMP are:

$$\frac{1}{\tau}\dot{v} = \alpha_v(\beta_v(g-p)-v)+f(s) \quad (1)$$

$$\frac{1}{\tau}\dot{p} = v \quad (2)$$

Proceedings 28th European Conference on Modelling and Simulation ©ECMS Flaminio Squazzoni, Fabio Baronio, Claudia Archetti, Marco Castellani (Editors)
ISBN: 978-0-9564944-8-1 / ISBN: 978-0-9564944-9-8 (CD)

where p is the position and v the velocity of the system and g is the desired goal of the trajectory. The parameters α_v and β_v are constants and τ is a time scaling constant. Equation (1), termed the controller equation, contains the nonlinear component f which is defined based on the demonstrated movement as a weighted sum of Gaussian functions:

$$f(s) = \frac{\sum_i \psi_i(s) w_i}{\sum_i \psi_i(s)} s \quad (3)$$

$$\frac{1}{\tau}\dot{s} = -\alpha s \quad (4)$$

Where ψ_i are the Gaussian functions and w_i are their weights, f depends explicitly on the phase variable, s, and α is a predefined constant.

The phase variable eliminates the time dependency and facilitates changing the movement duration using the scaling parameter, τ, without changing the trajectory. Equation (4) is termed the 'canonical system'. As s approaches one at the beginning of the trajectory and zero at the end of the movement causing f to vanish from the controller equation (equation 1) which then linearly converges to the goal, g (Nakanishi et al. 2004; Ijspeert et al. 2003; Pastor et al. 2009). DMPs facilitate learning dynamic trajectory characteristics while allowing adaptation to change during run-time (Hoffman et al. 2009). These characteristics are both crucial for robots in dynamic environments, such as the agricultural environment.

When there is a large difference between the demonstrating agent, e.g., the human, and the learning agent, e.g., the robot, several researchers have suggested a phase of autonomous training following the LfD phase. The purpose of the autonomous learning phase is to adapt the learned capabilities to the actual characteristics of the performing agent. Several Reinforcement learning (RL) methods have been suggested for the autonomous learning phase (Kormushev et al. 2010). RL is a commonly used method for learning where task dynamics are difficult to model (Sutton and Barto, 1998). Yet RL does not scale well to high dimensional search spaces in which it suffers from convergence problems. Using the parameters learned in the initial LfD phase as the starting point for RL-based autonomous learning can expedite the convergence, and help avoid local optima.

Policy Improvement with Path Integrals (PI^2) is a variant of RL that has been found to outperform other RL-based learning algorithms for DMP parameter optimization (Kappen 2007; Broek et al. 2008). PI^2 is an RL method based on stochastic optimal control where cost is minimized rather than reward maximized. PI^2 can scale to high dimensional problems rendering it applicable for physical robotic systems with many degrees of freedom (DOF) (Buchli et al. 2011; Theodorou et al. 2010). The learning procedure of PI^2 is organized in epochs, i.e., several trials (roll-out) are run in which exploration noise is added to the weights of the Gaussian functions of the DMP. The cost of each roll-out is evaluated according to the cost value function. The weights are then updated based on the cost of all the roll-outs in the epoch. An additional advantage of PI^2 is that it does not require parameter tuning apart from the exploration noise (Tamosiunaite et al. 2011). Recent papers present practical implementations of the PI^2 algorithm including a robot dog that jumps across a gap (Pastor et al. 2013; Theodorou et al. 2010), a robotic arm which pours liquid (Tamosiunaite et al. 2011), a robotic arm that opens a door using its handle and a robotic arm equipped with a three-fingered hand and a force-torque sensor at the wrist, that picks a pen from table (Kalakrishnan et al. 2011).

The cost of learning using a physical system is generally high, thus it is important to minimize the time required for such a learning stage. In the current study we explore the integration of an intermediate stage of learning using a simulation of the physical system and its contribution to improvement in performance. The study aims to answer whether integration of an autonomous learning stage using a simulation model of the system can reduce the need for learning using the physical system. The rest of this paper is organized as follows: The method section presents the conducted experiments along with a description of the implementation of the different learning phases. It is followed by a Results section which presents the results of the experiment along with a discussion of their implications. The results section is followed by a Conclusions and Future Research section.

METHOD

Environment and task

A reach-to-grasp task was chosen for examination of the contribution of the different learning phases. Apples are harvested by a wrist motion that applies shear force against their peduncle. It is important that the stem remains connected to the apple after harvesting otherwise the apple cannot be marketed as fresh produce. Reaching the apple in a pose (position and orientation), that will enable the motion required for detaching it, is crucial for successful harvesting.

The task was conducted by both the human demonstrators and the robotic manipulator in the telerobotics laboratory, at the Ben-Gurion University of the Negev (Figure 1). An artificial apple tree was located in the center of the laboratory. Apples are connected to the artificial tree such that pulling them from the branch detaches them from it.

Learning

The learning process was divided into three sequential phases (Figure 2). First, a set of movements for each

task was recorded from human demonstrators (demonstration phase). A DMP was created for each axis based on the demonstrated movements using LfD (LfD phase) and then optimized first in simulation and then using physical hardware in the autonomous training phase.

Demonstration phase

Three demonstrators participated in the demonstration phase. Each demonstrator stood with both arms alongside the body in front of the tree, at an arms-length from it. A six DOF magnetic motion sensor (FASTRAK™, Pulhamus) was attached to the demonstrator's wrist. Each demonstrator executed two harvesting movements, where the harvesting hand pose was once to the side and once in front of the apple. Each harvesting movement was divided into two sub-movements: a reach-to-grasp movement that ended when the demonstrator grasped the apple and a detachment movement. The demonstrators were requested to pause shortly between the two sub-movements. After each harvesting movement the apple was re-inserted into its place on the tree.

Figure 1: Reach-to-grasp of an artificial apple in the telerobotics laboratory. Top: The human demonstrator, Bottom: The Motoman UP6 robot.

LfD phase

The demonstrated trajectories were transformed to robot-base coordinates. The recorded movement was very noisy thus it was smoothed with a 7th degree polynomial and resampled at constant intervals such that each trajectory was represented by N=24 points. Three DMPs were created, one for each main axis in robot-base coordinates. The nonlinear movement component was extracted from the demonstrated movement and approximated using non-linear regression by a weighted sum of 10 Gaussian functions. The start and goal points of each axis were defined based on robot position and a trajectory was created for each axis using the computed DMP.

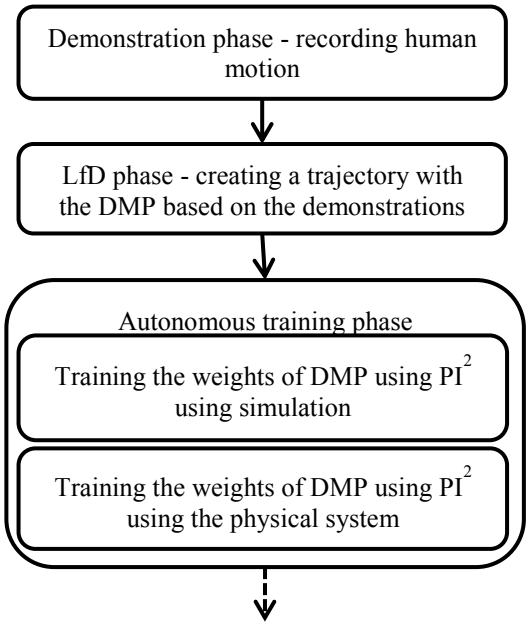

Figure 2: The learning process

Autonomous training using simulation

The PI^2 algorithm was implemented using MatLab (MathWorks, USA). The algorithm was run on an Intel® Core™ i7-2600 3.40 Ghz processor, with 4 GB RAM, and windows 7 Enterprise, 32 bit operation system.

Two cost functions were defined for creating a smooth robotic motion: minimum squared acceleration (MSA) and minimum acceleration change (MCA),

$$MSA_j = \frac{\sum_{i=1}^{N} a_j^2(i)}{N * a_{\max}^2} + \frac{\left(p_j(N) - g_j\right)^2}{\Delta p g_{\max}^2} + \frac{\sum_{i=1}^{N} \left(p_j(i) - d_j(i)\right)^2}{N * \Delta p d_{\max}^2} \quad (5)$$

Where j indicates the DMP axis, For the trajectory created by the DMP, $a(i)$ is the acceleration and $p(i)$ is the position at sample i, $d(i)$ the position of the demonstrated movement. The values of a_{max}, Δpg_{max}, and Δpd_{max} are the overall maxima of acceleration, difference between final position and goal over all the roll outs, and difference between position and demonstrated position respectively.

$$MCA_j = \frac{\sum_{i=2}^{N} \left(\frac{a_j(i) - a_j(i-1)}{T/(N-1)}\right)^2}{N * \Delta a'^2_{\max}} + \frac{\left(p_j(N) - g_j\right)^2}{\Delta pg^2_{\max}} + \frac{\sum_{i=1}^{N} \left(p_j(i) - d_j(i)\right)^2}{N * \Delta pd^2_{\max}} \quad (6)$$

Where Δa_{max} is the overall maxima of the change in accelerations, and T is the movement duration.

Both cost functions include three parameters, two of which are similar: the distance to the goal and the similarity to the demonstrated movement. The distance to the goal is measured by the distance between the actual final configuration and the planned goal. The similarity to the demonstrated movement is measured by the sum of the squared errors between the performed trajectory and the demonstrated trajectory. MSA additionally includes the acceleration amplitude measured by the sum of squared accelerations and MCA includes the change in accelerations measured by the sum of squared changes in acceleration. The parameters were chosen such that the final trajectory will be similar to the demonstrated path, converge to the goal, and will be smooth. All the parameters are normalized and equally weighted. The robotic motion is simulated based on classical motion equations.

The initial inputs to the simulation-learning phase were the DMPs computed based on the demonstrations. Ten roll-outs were executed in each iteration of the PI^2 algorithm. For each roll-out a vector of random numbers was sampled from the standard normal distribution. The random values were multiplied by a predefined constant. The constant was set by trial and error to be 20. The rate of change when using the MSA cost function converged to zero after approximately 5000 iterations and thus the number of iterations was set to 5000 for all trials.

Autonomous training using the physical system

The robot control software was programed using Microsoft C# and MotoCom robotic communication library for C++ (Motoman, Japan). The communication between the robot controller and the computer was established through a serial RS232 link. The robot program was written using Infrom (Motoman, Japan) for an XRC Motoman controller.

The inputs to the training phase using the physical system were the DMPs after the simulation phase. Iterations of ten roll-outs were executed using the robot. The roll-outs were produced using MatLab. The noise added to the weights was computed as in the simulation runs. A trajectory based on each roll-out was sent to the robot for execution. The trajectory of the robot contained 24 via positions and took about 46 sec to complete. During the execution of the trajectory, the robot's current position was sampled at constant time intervals of 2 sec. After consecutively executing the ten roll-outs, these positions were used to compute the cost function and to accordingly update the DMP weights.

Analysis

For all recorded trajectories three DMPs were computed based on LfD and then optimized with 5000 iterations of the PI^2 algorithm using simulation. Average improvement in all the parameters of each cost function (MSA and MCA) was computed based on the values of the simulated trajectory.

For one of the trajectories the DMPs after the simulation-training were used as input for training using the physical system. Using the physical system the PI^2 algorithm was run for two iterations using the MSA cost function. For this trajectory, improvement was measured based on the actual trajectory performed by the robot. A trajectory was formed based on the three final DMPs computed after each phase (initial DMP computed based on LfD, training using simulation, training using the physical system) and sent to the robot for execution. The values of the cost function parameters were computed based on the trajectories executed by the robot.

RESULTS

Training using simulation

A typical learning curve is presented in Figure 3. This curve depicts the value of the cost function as a function of the executed iterations. From the graph it is apparent that the value of the cost function converges after 5000 iterations. The average run-time of 5000 iterations was 17 minutes.

The improvement in all parameters based on simulated trajectories after 5000 iterations of the PI^2 algorithm using simulation is presented in Table 1. The parameters should be minimized, thus, negative percentages present a decrease in the parameter (and thus improvement). Overall both cost functions (MSA and MCA) improved (reduced) by more than 50%. The final distance to the goal considerably decreased after the simulation-training phase using both MSA and MCA cost functions. The acceleration values (MSA) and the change in accelerations also considerably decreased. On the other-hand the distance to the demonstrated

trajectory increased as there is a tradeoff between the parameters.

Typical trajectories are depicted in Figure 4. The trajectories were produced for the human demonstration and based on the robot's movement after the different phases: the initial trajectory based on the DMPs after the LfD process and the trajectory produced by DMPs after 5000 iterations of the PI^2 algorithm using simulation. From the figure we can see that the trajectory produced by the initial DMPs is very similar to the demonstrated trajectory. The trajectory after the simulation has indeed changed yet, as required, it is still reminiscent of the original trajectory.

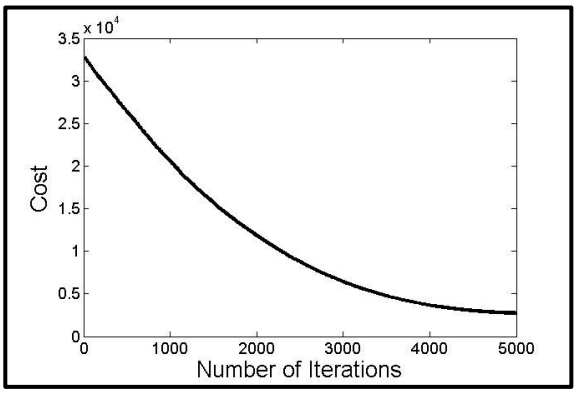

Figure 3: The cost as a function of the number of iterations using the MSA cost function

Table 1: Improvement based on simulated trajectories after training using simulation

	Distance to goal % (S.E.)	Distance to Demo % (S.E.)	Acceleration (change/value) % (S.E.)	Total % (S.E.)
MSA	-98 (0.05)	47.9 (0.6)	-21.9 (0.1)	-52.3 (0.2)
MCA	-85.5 (0.2)	166.6 (2.1)	-51.9 (0.3)	-53.1 (0.3)

S.E. – standard error

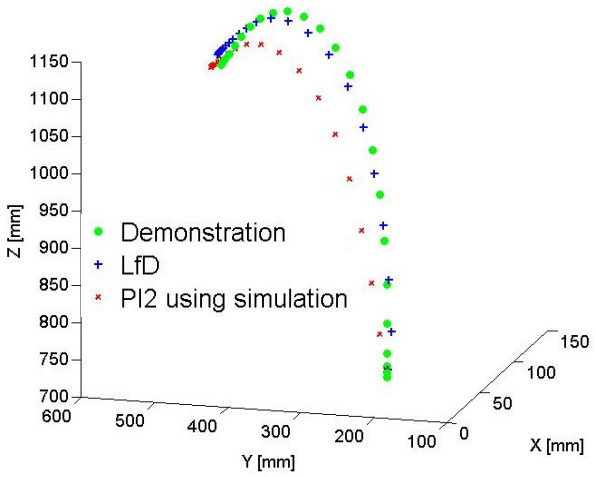

Figure 4: Reach-to-grasp trajectories: demonstrated, executed by the robot based on the initial DMPs learned from demonstration and after learning using simulation

Training using the physical system

For the selected trajectory and both cost functions the improvement based on actual robot trajectories (a single trajectory was tested for each cost function) in all parameters after 5000 iterations of the PI^2 algorithm in simulation is presented in Table 2.

Total cost is reduced using both cost functions. The final distance to the goal was reduced for both cost functions and the acceleration values (MSA) and the change in accelerations also considerably decreased. For MCA even the distance to the demonstrated trajectory decreased while for MSA this distance increased.

The run-time of two iterations of the PI^2 algorithm using the robot was 25 min. These iterations did not change the values of the cost function or its parameters.

Table 2: Improvement based on a single robot trajectory after training using simulation

	Distance to goal %	Distance to Demo %	Acceleration (change/value) %	Total %
MSA	-73.0	-14.2	-4	-69.5
MCA	-61.4	3.4	-20.6	-27.4

CONCLUSIONS AND FUTURE RESEARCH

For the reach-to-grasp motion in an apple harvesting task, the parameters of DMPs were learned based on human demonstration, and then adapted with the PI^2 algorithm using simulation and using hardware. The simulation-based training, which greatly improved the motion parameters, was considerably faster than hardware-based training. Training using hardware following the training using simulation did not additionally improve motion parameters.

Future work will examine integration of training using simulation in additional tasks and scenarios that require interaction with the environment, e.g., the apple detachment motion. We expect that training using hardware to be of importance in such tasks in which interaction dynamics are of importance.

ACKNOWLEDGMENT

The research is supported by the European Commission in the 7th Framework Programme (CROPS GA no 246252). The authors thank Noam Peles, Gil Baron, Nissim Abuhatzira and Josef Zahavi for their assistance in the hardware implementation.

REFERENCES

Allota, B., Buttazzo, G., Dario, P., and Ouaqlia, F. 1990. "A Force/Torque Sensor-Based Technique for Robot Harvesting of Fruits and Vegetables". In *Intelligent Robots and Systems '90 'Towards a New Frontier of Applications'* (Ibaraki, July 3-6). IEEE, 231-235.

Argall, B. D., Chernova, S., Veloso, M., and Browning, B. 2009. "A Survey of Robot Learning from Demonstration."

Robotics and Autonomous Systems 57, No. 5 (May), 469-483.

Bekey, G.A. 2005. *Autonomous Robots: From Biological Inspiration to Implementation and Control*. Cambridge, Mass. MIT Press.

Broek, B. v. d., Wiegerinck, W., and Kappen, B. 2008. "Graphical Model Inference in Optimal Control of Stochastic Multi-Agent Systems." *Journal of Artificial Intelligence Research* 32, No.1 (Oct), 95–122.

Buchli, J., Theodorou, E., Stulp, F., and Schaal, S. 2011. "Learning Variable Impedance Control." *The International Journal of Robotics Research* 30, No. 7 (June), 820-833.

Flash, T., Hochner, B. 2005. "Motor Primitives in Vertebrates and Invertebrates," *Current Opinion in Neurobiology* 15, No. 6 (Dec), 660‑666.

Foglia, M.M. and Reina, G. 2006. "Agricultural Robot for Radicchio Harvesting." *Journal of Field Robotics* 23, No. 6 (July), 363-377.

Forge, S. and Blackman, C. 2010. "A Helping Hand for Europe: The competitive outlook for EU robotics industry." Bogdanowicz, M. and Desruelle, P. (Eds.), Scientific and Technical Report. 24600 EN, Joint Research Center, Institute for Prospective Technological Studies (IPTS), European Commission.

Hoffman, H., Pastor, P., Park, D.H., and Schaal, S. 2009. "Biologically-Inspired Dynamical Systems for Movement Generation: Automatic Real-Time Goal Adaptation and Obstacle Avoidance." In *ICRA '09 Conference on Robotics and Automation* (Kobe, May 12-17). IEEE, 2587- 2592.

Ijspeert, A.J., Nakanishi, J., and Schaal, S. 2002. "Movement Imitation with Nonlinear Dynamical Systems in Humanoid Robots." In *ICRA '09 Conference on Robotics and Automation* (Washington DC, May 12-15). IEEE, 1398-1403.

Ijspeert, A.J., Nakanishi, J., and Schaal, S. 2003." Learning Attractor Landscapes for Learning Motor Primitives." In *Advances in Neural Information Processing Systems 15 2003*, Becker, S., Thrun, S., and Obermayer, K (Eds.). Cambridge, MAS: MIT Press, 1547-1554.

Kalakrishnan, M., Righetti , L., Pastor, P., Schaal, S. 2011. "Learning Force Control Policies for Compliant Manipulation." In *2011 RSJ International Conference on Intelligent Robots and Systems* (San Francisco CA, Sep. 25-30). IEEE, 4639-4644.

Kappen, H. J. 2007. "An Introduction to Stochastic Control Theory, Path Integrals and Reinforcement Learning." In *Cooperative Behavior in Neural Systems 2007*, Marro, J., Garrido, P. L., and Torres, J. J. (Eds.). *American Institute of Physics Conference Series 887*, 149–181.

Koenig, N., Mataric, M.M., and Takayama, L. 2010. "Communication and Knowledge Sharing in Human-Robot Interaction and Learning from Demonstration." *Neural Networks 23*, No. 8-9 (Oct-Nov), 1104-1112.

Kormushev, P., Calinon, S., Caldwell, D.G. 2010. "Robot motor skill coordination with EM-based Reinforcement Learning". In *Intelligent Robots and Systems 2010*, (Taipei, Oct. 18-22). IEEE, 3232-3237.

Matarić, M.J. 2007. *The Robotics Primer*. Cambridge, Mass: MIT Press.

Kulvicius, T., Ning, K., Tamosiunaite, M., and Wörgötter, F. 2012. "Joining Movement Sequences: Modified Dynamic Movement Primitives for Robotics Applications Exemplified on Handwriting." *IEEE Transactions on R*obotics 28, No. 1 (Feb), 145-157.

Nakanishi, J., Ijspeert, A. J., Schaal, S., and Cheng, G. 2004. "Learning Movement Primitives for Imitation Learning in Humanoid Robots." *Journal- Robotics Society of Japan 22*, No. 2, 17-22.

Pastor, P., Hoffmann, H., Asfour, T., and Schaal, S. 2009. "Learning and Generalization of Motor Skills by Learning from Demonstration." In *International Conference on Robotics and Automation 2009* (Ney York, Jan.1). IEEE, 763-768.

Pastor, P., Kalarishnan, M., Meier, F., Stulp, F., Buchli, J., Theodorou, E., Schaal, S. 2013. "From Dynamic Movement Primitives to Associative Skill Memories." Robotics and Autonomous Systems 6, No. 4 (April), 351-361.

Sutton, R.S., Barto, A.G. 1998. *Reinforcement Learning: An Introduction*. Cambridge, Mass: MIT Press.

Tamosiumaite, M., Nemec, B., Ude, A., and Wörgötter, F. 2011. "Learning to Pour with a Robot Arm Combining Goal and Shape Learning for Dynamic Movement Primitives." *Robotics and Autonomous Systems* 59, No. 11 (Nov), 910-922.

Theodorou, E., Buchli, J., and Schaal, S. 2010. "Reinforcement Learning of Motor Skills in High Dimensions: a Path Integral Approach." In *ICRA '10 Conference of Robotics and Automation* (Anchorage AK, May 3-7). IEEE, 2397–2403.

AUTHOR BIOGRAPHIES

ANAT HERSHKOVITZ COHEN is a MSc Student in the Department of Industrial Engineering and Management, Ben-Gurion University of the Negev, Beer-Sheva. She received a BSc in Industrial Engineering and Management, also from Ben-Gurion University of the Negev (2013). Her research focuses on robots algorithms that allow robots to learn from humans motions. Her e-mail address: anhe@bgu.ac.il.

SIGAL BERMAN is a senior lecturer in the Department of Industrial Engineering and Management, Ben-Gurion University of the Negev, Beer-Sheva. She received a Ph.D. in Industrial Engineering and Management from the Ben-Gurion University (2002) and a B.Sc. in Electrical and Computer Engineering, The Technion, Haifa. Her research interests include: Human motor control, robotics and telerobotics. Her e-mail address is: sigalbe@bgu.ac.il and her Web-page can be found at http://www.bgu.ac.il/~sigalbe/.