Learning Behavior Fusion Estimation from Demonstration

Monica Nicolescu and Odest Chadwicke Jenkins and Adam Olenderski

Abstract

A critical challenge in robot learning from demonstration is the ability to map the behavior of the trainer onto the robot's existing repertoire of basic/primitive capabilities. Following a behavior-based approach, we aim to express a teacher's demonstration as a linear combination (or fusion) of the robot's primitives. We treat this problem as a state estimation problem over the space of possible linear fusion weights. We consider this fusion state to be a model of the teacher's control policy expressed with respect to the robot's capabilities. Once estimated under various sensory preconditions, fusion state estimates are used as a coordination policy for online robot control to imitate the teacher's decision making. A particle filter is used to infer fusion state from control commands demonstrated by the teacher and predicted by each primitive. The particle filter allows for inference under the ambiguity over a large space of likely fusion combinations and dynamic changes to the teacher's policy over time. We present results of our approach in a simulated and real world environments with a Pioneer 3DX mobile robot.

I. INTRODUCTION

The problem of estimating the state of the user lies at the core of human-robot interaction. Such state information can vary over several modalities, such as affective state and shared state. Human state in these problems is estimated and used to guide a robot's actions or a system design process to improve a user's experience or performance. In the case of learning from demonstration (LFD) [1], the objective is to estimate the human's decision state. That is, the single control policy out of all possible control policies that is utilized by the teacher during a demonstration. Once estimated, the learned policy can be used as the robot's control policy, assuming an appropriate transform between embodiments. LFD strives to provide open the programming of robot control to broader populations in society by learning implicitly from human guidance, as opposed to explicit programming of a computer language. In addition to LFD, knowledge of a user's decision state can be used to inform online human-robot interaction, such as for adjustable autonomy [2].

At a high level, estimating a user's decision state is finding the most likely control policy given sensory-motor observations from demonstration. In the context of a Markov Decision Process, a control policy $\mu(S) \rightarrow A$ is defined by a mapping of observed world state S to control outputs A. This control policy governed by a value function, $Q(S, A)$, that specifies the benefit for taking action A in world state S. This value function defines the decision state over the space of all state action (S, A) pairs. Atkeson and Schaal [3] define optimality (as a reward function) for this problem as minimizing the divergence in performance of the learned policy and observations from a demonstration. Specifically, given a world state occurring during demonstration, the motor output predicted by the learned policy should result changes to world state to the demonstration. By phrasing optimality based on demonstration, policies can be learned for arbitrary tasks without (or with) bias towards specific tasks.

However, learning such policies are subject to issues in partial observability in world state and generalization to new situations. For instance, the approach of Atkeson and Schaal, and subsequent work [4], are geared for top-down planning over the space of control policies given fully observable and relevant aspects of world state. However, such methods are suited for limited or one-time generalization that varies or refines existing behavior (e.g., correcting a demonstrated locomotion gait or variations of a tennis swing).

To address these issues, we propose a behavior-based approach to learning from demonstration that uses behavior fusion to provide bottom-up generalization to new situations. Assuming a set of preexisting robot behaviors

M. Nicolescu and A. Olenderski are with the University of Nevada, Reno, 1664 N. Virginia St. MS 171, Reno, NV, 89523 monica@cse.unr.edu, olenders@cse.unr.edu

O. C. Jenkins is with Brown University, 115 Waterman St., 4th Floor, Providence, RI 02912-1910 cjenkins@cs.brown.edu

This work has been supported by the National Science Foundation under contract number IIS-0546876 and by a UNR Junior Faculty Award to Monica Nicolescu.

expressed as schemas [5] or potential fields, our aim is to learn a coordination policy that linearly fuses their combined output in a manner that matches the teacher's demonstration. We phrase the learning of this coordination as a fusion estimation problem, i.e., state estimation in the space of linear combinations of primitive behaviors. For domains such as mobile robotics, fusion estimation is often subject to ambiguous changes in world state that are attributable to a large space of solutions.

In this paper, we present *behavior fusion estimation* as a method to learn from demonstration fusion weights for coordinating concurrently executing primitive behaviors. To account for this ambiguity and dynamic changes to the user's fusion policy, a particle filter is used to infer fusion estimates from robot sensory observations and motor commands. We focus on the limited case in which fusion is assumed to be unimodal for each discrete combination of behavior preconditions. Results are presented demonstrating fusion policies learned from data collected during simulated and real-world navigation demonstrations of a teleoperated Pioneer 3DX robot.

II. RELATED WORK

Our work falls into the category of *learning by experienced demonstrations*. This approach implies the robot actively participates in the demonstration provided by the teacher, and experiences the task through its own sensors.

Successful first person approaches have demonstrated learning of reactive policies [6], trajectories [7], or high-level representations of sequential tasks [8]. These approaches employ a *teacher following* strategy, where the robot learner follows a human or a robot teacher. Such sequential (or arbitrated) representations for coordination can be considered a subset of our approach. In contrast, our aim is to avoid hard binary decisions in coordination of and fusing control commands from different behaviors. This results in a general-purpose policy, which would allow the robot to perform the demonstrated task in any new environments, from any initial positions. We do not attempt to reproduce exact trajectories, but rather learn the underlying policy for executing the task. In making binary decisions, sequential methods represent all possible coordinations as discrete set of possibilities along each axis of fusion space. Platt et al. [9] proposed null-space composition as a fusion coordination mechanism limited to control states where behaviors do not affect each other. This coordination allows for off-axis (but discrete) combinations in fusion space. Although the work of Platt et al. is applied to dexterous manipulation, we focus only on their coordination mechanism and not platform specifics.

The choice of the particle filter [10] is only one of several methods available to infer behavior fusion. The most straightforward choice is a linear least squares optimization. While least squares worked well when little ambiguity was present in the fusion likelihood, our prior testing showed that it did not sufficiently find functional fusion policies as the number of primitives increased and introduced greater ambiguity. Nonlinear methods, such as Levenberg-Marquardt, could yield better results. However, we chose the particle filter to account for ambiguity explicitly. LFD in the form reinforcement learning methods, such as Q-Learning [11], are a viable option for fully observable states, but are nontrivial to extend for partial observability.

A significant challenge for all robotic systems that learn from a teacher's demonstration is the ability to map the perceived behavior of the trainer to their own behavior repertoire. We focus on the specific problem of learning *behavior fusion from demonstration* that could be cast into more holistic approaches to human-robot interaction, such as work by Breazeal et al. [12]. One successful approach to this problem has been to match observations to robot behaviors based on *forward models* [13], [14], in which multiple behavior models compete for prediction of the teacher's behavior [15], [16], and the behavior with the most accurate prediction is the one said to match the observed action.

III. BEHAVIOR REPRESENTATION

Behavior-Based Control (BBC) has become one of the most popular approaches to embedded and robotic system control both in research and in practical applications. We utilize a schema-based representation in the context of BBC, similar to approaches in [5]. This choice is essential for the purpose of our work because schemas with BBC provide a continuous encoding of behavioral responses and a uniform output in the form of vectors generated using a potential fields approach.

In our system, a controller consists of a set of concurrently running behaviors. Thus, for a given task, each behavior brings its own contribution to the overall motor command. These contributions are weighted such that, for example, an *obstacle avoidance* behavior could have a higher impact than *reaching a target*, if the obstacles in the field are significantly dangerous to the robot. Alternatively, in a time constrained task, the robot could give a higher contribution to getting to the destination than to obstacles along the way. These weights affect the magnitude of the individual vectors coming from each behavior, thus generating different modalities of execution for the task.

IV. BEHAVIOR FUSION ESTIMATION

The primary function in behavior fusion estimation is to infer, from a teacher provided demonstration, the contribu-

tion (or weight) of each primitive in the robot's repertoire such that their combination matches the observed outcome. These weights modulate the magnitude of control vector output by the individual primitives, thus influencing the resulting command from fusion and consequently the way the robot interacts with the world. However, choosing these weights is a non-trivial problem. To save time and resources (such as robot power), we automatically estimate appropriate weights for fusing behaviors according to the desired navigation style as demonstrated.

For a set of N primitives, behavior fusion estimation is accomplished by estimating the joint probability distribution of the fusion space (i.e., across weighting combinations) over the demonstration duration. For this work, demonstrations consisted of guiding the robot through a navigation task, using a joystick, while the robot's behaviors continuously provide predictions on what their outputs would be (in the form of a 2D speed and heading vector in the robot's coordinate system) for the current sensory readings. However, instead of being translated into motor commands, these predictions are recorded along with the turning rate of the robot, at that moment of time. Thus, for each timestep t, we are provided with a set of prediction vectors $V_p^t = v_1^t ... v_N^t$ from each primitive and a demonstration vector V_r^t expressing the realized control output of the robot. It is known that the resulting vector V_r^t is a linear combination of the prediction vectors $[v_1^t \cdots v_N^t]$ according to some unknown superposition weights $S^t = [s_1^t \cdots s_N^t]$:

$$V_r^t = \sum_{i=1}^{N} s_i^t v_i^t \tag{1}$$

We consider heading to be the most important consideration for behavior fusion in 2D navigation. Consequently, we normalize command vectors to unit length.

The goal of the algorithm is to infer the weights s over time or, more precisely the relative proportions among the weights that could produce the demonstration vector V_r.

A. Incorporating Behavior Preconditions

At any given time during the demonstration, multiple behaviors could be active, depending on whether their preconditions are met or not. We segment the demonstration into intervals based on the binary decisions set by the preconditions of each behavior. A segmentation of the demonstration trace is performed at the moments of time when the status of any of the behaviors' preconditions changes between met and not-met. The resulting segments represent different environmental situations, since different behaviors become "applicable" at the transition points. The weights of behaviors within each segment encode the mode of performing the current task given the situation and, thus within each segment, the weights of the applicable behaviors are constant. For example, for a *target reaching* task, the robot could behave under the influence of *corridor-follow, target-follow* and *avoid-obstacle* behaviors if in the presence of obstacle, but would behave only under the influence of *target-follow* if in an open space.

B. Fusion Parameter Estimation

Similar to Monte Carlo robot localization, a particle filter is used to recursively estimate the joint density in the parameter space of fusion weights S^t over time $t = 1 \cdots T$. Particle filters [17] have been used for state and parameter estimation in several different domains (such as robot localization [10], pose estimation [18], and insect tracking [19]). Restating these methods, mostly following [19], we use the standard form of the Bayes filter to estimate the *posterior* probability density $p(S^t|V_r^{1:t}, V_p^{1:t})$ in the space of fusion parameters given prediction and result vectors:

$$p(S^t|V_r^{1:t}, V_p^{1:t}) = \tag{2}$$
$$kp(V_r^t, V_p^t|S^t) \int p(S^t|S^{t-1}) p(S^{t-1}|V_r^{1:t-1}, V_p^{1:t-1})$$

where $p(V_r^{1:t}, V_p^{1:t}|S^t)$ is the *likelihood* of observing prediction and result vector given a vector of fusion parameters, $p(S^t|S^{t-1})$ is the *motion model* describing the expected displacement of parameter weights over a timestep, $p(S^{t-1}|V_r^{1:t-1}, V_p^{1:t-1})$ is the *prior* probability distribution from the previous timestep, and k is a normalization constant to enforce that the distribution sums to one. We simplify the likelihood using the chain rule of probability and domain knowledge (Eq. 1) that prediction vectors are not dependent on the fusion weights:

$$p(V_r^t, V_p^t|S^t) = p(V_r^t|V_p^t, S^t) p(V_p^{1:t}|S^t) = p(V_r^t|V_p^t, S^t) \tag{3}$$

The resulting Bayes filter:

$$p(S^t|V_r^{1:t}, V_p^{1:t}) = \tag{4}$$
$$kp(V_r^t|V_p^t, S^t) \int p(S^t|S^{t-1}) p(S^{t-1}|V_r^{1:t-1}, V_p^{1:t-1})$$

has a Monte Carlo approximation that represents the posterior as particle distribution of M weighted samples $\{S_{(j)}^t, \pi_{(j)}^t\}_{j=1}^M$, where $S_{(j)}^t$ is a particle representing a specific hypothesis for fusion weights and $\pi_{(j)}^t$ is the weight of the particle proportional to its posterior probability:

$$p(S^t|V_r^{1:t}, V_p^{1:t}) \propto kp(V_r^t|V_p^t, S^t) \sum_j \pi_{(j)}^t p(S^t|S^{t-1}) \tag{5}$$

The estimation of the posterior at time t is performed by 1) importance sampling to draw new particle hypotheses $S_{(j)}^t$ from the posterior at time $t-1$ and 2) computing

weights $\pi^t_{(j)}$ for each particle from the likelihood. Importance sampling is performed by randomly assigning particle $S^t_{(i)}$ to particles $S^{t-1}_{(j)}$ based on weights π^{t-1} and adding Gaussian noise. This process effectively samples the following proposal distribution:

$$S^t_{(i)} \sim q(S^t_{(i)}) \triangleq \sum_j \pi^t_{(j)} p(S^t | S^{t-1}_{(j)}) \quad (6)$$

and weights by the following likelihood as the distance between actual and predicted displacement direction:

$$\pi^t_{(i)} = p(V^t_r | V^t_p, S^t) = 2 - D(V^t_r, \hat{V}^t_{(i)})/2 \quad (7)$$

where $D(a, b)$ is the Euclidean distance between a and b and:

$$\hat{V}^t_{(i)} = \frac{\sum_{k=1}^N S^t_{(i),k} v^t_k}{|\sum_{k=1}^N S^t_{(i),k} v^t_k|} \quad (8)$$

The process described above is performed on each of the recorded instances of time, resulting in a posterior distribution of fusion weights for every time step during demonstration. Current particle filtering methods have three approaches to selecting appropriate parameters from this distribution as the mean of the distribution, the maximum of the distribution, or the robust mean (the mean of particles in some neighborhood around the maximum). All three estimators have advantages and disadvantages. For simplicity, we chose the mean of the distribution. While the computed fusion weights will vary throughout the task, a single fusion set will emerge as being the most consistent during the performance of each subtask. We find these fusion weights by removing obvious outliers and taking the average of the derived weights over all the records.

V. EXPERIMENTAL RESULTS

To validate the proposed learning algorithm we performed experiments with a Pioneer 3DX mobile robot equipped with a SICK LMS-200 laser rangefinder, two rings of sonars, and a pan-tilt-zoom (PTZ) camera, both with the real robot and with the Player/Stage simulation environment [20]. We performed three different sets of experiments (scenarios), in each of which the robot was equipped with a different underlying set of primitives. In each scenario the robot was given a set of different demonstrations, from each of which it learned different behavior weights for controlling the robot. Thus, for each specific scenario and demonstration, the robot had learned the task from a single trial. All demonstrations ended by taking the robot toward a goal represented by a colored target.

The complete set of behaviors consists of: laser obstacle avoidance (*avoid*), attraction to a goal object (*attract*), attraction to unoccupied space (*wander*, *wander-left*, *wander-right*), attraction to walls (*wallAttract*, *wallAttract-left*, *wallAttract-right*), rear sonar obstacle avoidance (*sonarAvoid*).

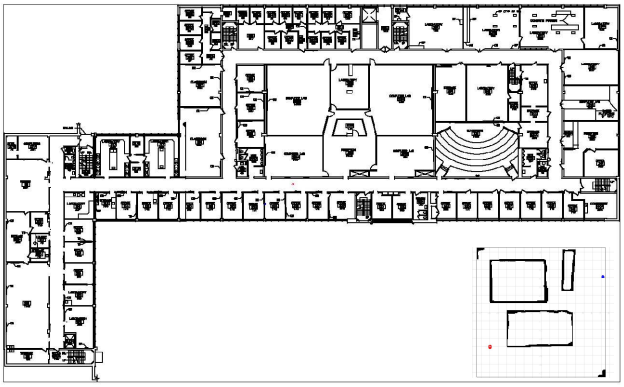

Fig. 1. Experimental setup: Top: SEM building map; Bottom right: small simulated environment

A. Scenario 1

During the first set of experiments, the robot was equipped with the following behaviors: *avoid*, *attract*, *wander*, *wallAttract*, *sonarAvoid*. We performed four different demonstrations, with the Pioneer 3DX robot, each consisting of making a tour of the floor in our SEM building (Figure 1). Every demonstration used a different navigation strategy as follows: 1) keep to the right, 2) keep to the center, 3) keep to the left and 4) slalom.

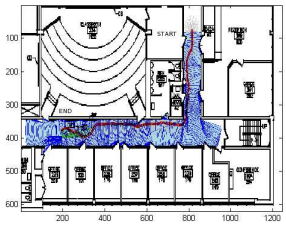

Fig. 2. Results from Scenario 1. The robot uses the weights from the center demonstration.

Results. From these experiments the robot had learned to safely navigate in the environment and to go toward the goal when seeing it. Since the robot was always close to walls on both sides during all demonstrations, and due to the fact that the only *wallAttract* behavior does not have a preference for left or right walls to robot learns a strong preference for staying close to the walls, irrespective of which weights are used (from the right/left/center/slalom navigation). Figure 2 shows a typical path that the robot would take using the weights from these demonstrations. The image shows the robot pulling away from the wall when the goal is detected, demonstrating that the robot

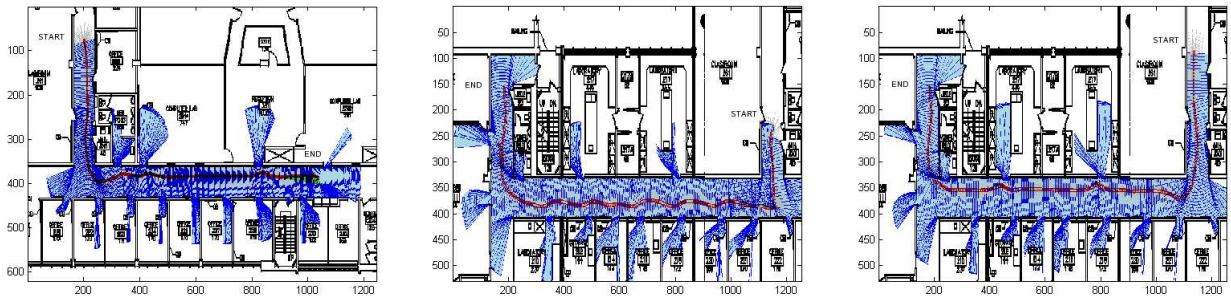

Fig. 3. Results from Scenario 3. The robot used learned fusion weights from the left, center and respectively right side demonstrations.

learns the proper weights for dealing with this situation. In this image, as for the following plots, the robot path is in red,the laser range readings are in blue and the detection of the goal is in green.

B. Scenario 2

During the second set of experiments, the robot was equipped with the same set of behaviors as in Scenario 1, with the only difference that the we replaced the general *wallAttract* behavior with *wallAttract-left* and *wallAttract-right*. The left and right wall attract behaviors respond only to the walls on their corresponding side, as opposed to all the combined approach of regular *wallAttract*. The experiments were performed in a simple simulated environment, shown in Figure 1, bottom right inset. The four demonstrations consisted of taking the following paths through the environment: one through the upper corridor, one through the bottom (wide) corridor, in both cases keeping on the left, then on the right. With the weights learned in this environment we tested the behavior of the robot in a simulated SEM building map.

Results. As opposed to the first set of experiments, due to the split of the wall-Attract behavior, the robot now learned a difference on which side to favor when navigating a corridor. However, a more significant difference in behavior is apparent when the robot approaches a T-junction. When using the weights learned from the right-following demonstrations (for both corridor demonstrations), the robot turns to the right when it approaches an intersection. That is, when it is given the choice of either “right or left” or “right or straight,” it will go right, because of how strongly it is attracted to the right wall. When it is given the choice “straight or left,” it will go straight. With the weights learned from the left-following demonstrations, the robot exhibits a similar behavior for the left side. These results demonstrate the robustness of our approach, in that controllers learned in one environment translate to different environments as well.

C. Scenario 3

In the third set of experiments, the set of behaviors was the same as in Scenario 2, with the only difference that instead of the *wander* behavior, we used its split versions, *wander-left* and *wander-right*. These two behaviors seek left and respectively right open spaces, as opposed to the regular *wander*, which looks for open space in any direction. We performed three demonstrations, in the SEM simulated environment, each consisting of taking a tour of the building, as follows: 1) keep to the right, 2) keep to the center, and 3) keep to the left.

Results. As expected from these experiments, the robot has learned similar preferences as those in the second scenario. Figure 3 (left) shows the trajectory of the robot, using the controller learned from the left follow demonstration. The robot starts at the top of the left corridor, chooses a left turn at the T-junction, then stops at the goal. During the entire run the robot keeps closer to the wall on the left. Figure 3 (right) shows a similar preference, this time for following walls on the right and choosing a right turn at T-junctions. In this experiment, the robot starts at the top of the right corridor and moves down and left. While for the left and right preferences the robot makes a clear turn in a particular direction when reaching the T-junction, for the center experiment the robot shows that it does not have a preferred direction, as shown by how far it goes into the T-junction before deciding to turn, due to its wander behavior (Figure 3 (center)). The robot navigates closer to the left due to the wandering behavior, which attracts the robot to the open spaces through the doors. We only show goal reaching capability for the left experiment: we stopped the center and right experiments before the robot made the full turn in the environment to reach the goal, located in a different area in the environment. If allowed to run longer, the robot was able to reach the goal in all situations.

In addition to learning the left and right preference, our results demonstrate that the additional refinement of the underlying behavior set in a *wander-left* and *wander-right* behavior, allowed the robot to capture additional aspects of the demonstration. In particular, whenever the robot found itself in the middle of a T-junction, with open space on both sides, the robot would choose to go in the direction given by the preference expressed during the demonstration: right

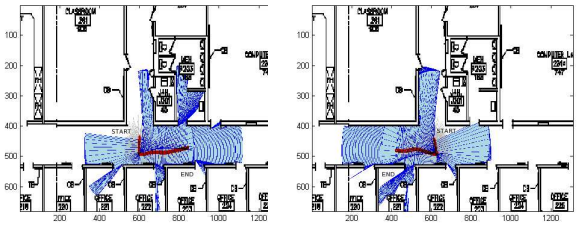

Fig. 4. The robot learns preferences of wandering in directions specified during the demonstration (left and right respectively).

for the right weights and left for the left weights. This preference was demonstrated even in the case in which the robot had more open space in the opposite direction. Under equal weighting of left and right wandering, the robot would normally follow the larger open space. Figure 4 shows this preference through the robot's trajectory. In the left image, the robot is using the weights learned from the left-follow demonstration. While the robot starts oriented slightly toward the right in the middle of the T-junction, as shown by its laser profile, the higher weight of the *wander-left* behavior pulls the robot in the left direction. Similarly, in the right image, the robot uses the weights from the right-follow demonstration. Even if oriented slightly to the left, where there is more open space, the robot chooses to go right due to the higher weight of *wander-right*.

The approach we presented demonstrates the importance of considering concurrently running behaviors as underlying mechanisms for achieving a task. Our method allows for learning of both the goals involved in the task (e.g., reaching a target) and also of the particular ways in which the same task can be performed. In addition, our results demonstrate the importance of choosing the primitive behavior set, an important and still open issue for behavior-based research. Our learned controllers are not restricted to a a particular path or execution sequence and thus are general enough to exhibit meaningful behavior even in environments different from the one in which the demonstration took place.

VI. SUMMARY

We presented a method for robot task learning from demonstration that addresses the problem of mapping observations to robot behaviors from a novel perspective. Our claim is that motor behavior is typically expressed in terms of concurrent control of multiple different activities. To this end, we developed a learning by demonstration approach that allows a robot to map the demonstrator's actions onto multiple behavior primitives from its repertoire. This method has been shown to capture not only the overall goals of the task, but also the specifics of the user's demonstration, thus enabling additional capabilities through learning by demonstration.

References

[1] S. Schaal, "Is imitation learning the route to humanoid robots," *Trends in Cognitive Sciences*, vol. 3, no. 6, pp. 233–242, 1999.

[2] M. Goodrich, D. Olsen, J. Crandall, and T.Palmer, "Experiments in adjustable autonomy." 2001.

[3] C. G. Atkeson and S. Schaal, "Robot learning from demonstration," in *ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997, pp. 12–20.

[4] S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert, "Learning movement primitives," in *International Symposium on Robotics Research*, 2004.

[5] R. C. Arkin, "Motor schema based navigation for a mobile robot: An approach to programming by behavior," in *IEEE Conference on Robotics and Automation, 1987*, 1987, pp. 264–271.

[6] G. Hayes and J. Demiris, "A robot controller using learning by imitation," in *Proc. of the Intl. Symp. on Intelligent Robotic Systems*, Grenoble, France, 1994, pp. 198–204.

[7] P. Gaussier, S. Moga, J. Banquet, and M. Quoy, "From perception-action loops to imitation processes: A bottom-up approach of learning by imitation," *Applied Artifi cial Intelligence Journal*, vol. 12(78), pp. 701–729, 1998.

[8] M. N. Nicolescu and M. J. Matarić, "Natural methods for robot task learning: Instructive demonstration, generalization and practice," in *Proc., Second Intl. Joint Conf. on Autonomous Agents and Multi-Agent Systems*, Melbourne, Australia, July 2003.

[9] R. Platt, A. H. Fagg, and R. R. Grupen, "Manipulation gaits: Sequences of grasp control tasks," in *IEEE Conference on Robotics and Automation*, New Orleans, LA, USA, 2004, pp. 801–806.

[10] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.

[11] W. D. Smart and L. P. Kaelbling, "Effective reinforcement learning for mobile robots," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA 2002)*, vol. 4, May 2002, pp. 3404–3410.

[12] C. Breazeal, A. Brooks, J. Gray, G. Hoffman, C. Kidd, H. Lee, J. Lieberman, A. Lockerd, , and D. Mulanda, "Tutelage and collaboration for humanoid robots," *International Journal of Humanoid Robotics*, vol. 1, no. 2, 2004.

[13] S. Schaal, "Learning from demonstration," *Advances in Neural Information Processing Systems*, vol. 9, pp. 1040–1046, 1997.

[14] O. C. Jenkins and M. J. Matarić, "Performance-derived behavior vocabularies: Data-driven acqusition of skills from motion," *International Journal of Humanoid Robotics*, vol. 1, no. 2, pp. 237–288, Jun 2004.

[15] D. Wolpert and M. Kawato, "Multiple paired forward and inverse models for motor control," *Neural Networks*, vol. 11, pp. 1317–1329, 1998.

[16] O. C. Jenkins, "Data-driven derivation of skills for autonomous humanoid agents," Ph.D. dissertation, The University of Southern California, 2003.

[17] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle fi lters for on-line non-linear/non-gaussian bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, Feb. 2002.

[18] M. Isard and A. Blake, "Condensation – conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[19] Z. Khan, T. R. Balch, and F. Dellaert, "A rao-blackwellized particle fi lter for eigentracking," in *IEEE Computer Vision and Pattern Recognition*, vol. 2, 2004, pp. 980–986.

[20] B. Gerkey, R. T. Vaughan, and A. Howard., "The player/stage project: Tools for multi-robot and distributed sensor systems," in *Proc., the 11th International Conference on Advanced Robotics*, 2003, pp. 317–323.