In Proc. of the 1992 IEEE International Conference on Robotics and Automation, Nice, France. pp 1475-1480.

Learning Reactive Admittance Control

Vijaykumar Gullapalli, Roderic A. Grupen and Andrew G. Barto

Computer Science Department
University of Massachusetts
Amherst, MA 01003.

Abstract

In this paper, a peg-in-hole insertion task is used as an example to illustrate the utility of direct associative reinforcement learning methods for learning control under real-world conditions of uncertainty and noise. An associative reinforcement learning system has to learn appropriate actions in various situations through search guided by evaluative performance feedback. We used such a learning system, implemented as a connectionist network, to learn active compliant control for peg-in-hole insertion. Our results indicate that direct reinforcement learning can be used to learn a reactive control strategy that works well even in the presence of a high degree of noise and uncertainty.

1 Introduction

The peg-in-hole insertion task has been widely used by roboticists for testing various approaches to robot control. Peg-in-hole insertion is also studied as a canonical robot assembly operation by researchers in industrial robotics. Both two-dimensional [15] and three-dimensional [8] peg-in-hole tasks have been studied in detail, and a number of approaches to the general problem have been proposed (see [6] for an overview).

While the abstract peg-in-hole task can be solved quite easily, real-world conditions of uncertainty and noise can substantially degrade the performance of traditional control methods. Sources of uncertainty and noise include (1) errors and noise in sensations, (2) errors in execution of motion commands, and (3) uncertainty due to movement of the part grasped by the robot. The first of these is the most common source of uncertainty. Position sensors can be especially unreliable when the robot is interacting with external objects, resulting, for example, in inaccurate knowledge of relative locations of parts of the robot with respect to the external objects. Noisy force sensations further compound the problem of control when the robot is interacting with external objects. Moreover, hybrid position/force controllers often exhibit instabilities near the boundary between contact and non-contact. The second source of uncertainty is due to the dynamics of the robot itself. Because of friction, gear backlash, etc., giving the same command in the same starting configuration can result in different motions of the robot. The third source of uncertainty stems from the possibility of parts moving within the grasp of the gripper. Many grippers have no means of detecting such motion.

Approaches proposed for peg-in-hole insertion under conditions of uncertainty and noise can be grouped into two major classes: methods based on off-line planning, and methods based on reactive control.[1] Off-line planning methods combine geometric analysis of the peg-hole configuration with analysis of the task statics to determine motion strategies that will result in successful insertion [15, 8, 6]. In the presence of uncertainty in sensing and control, researchers have suggested incorporating the uncertainty into the geometric model of the task in configuration space. *Pre-images* [11] or *backprojections* [5] of the goal are used in conjunction with a generalized damper control model to compute motions that result in successful insertion. If the peg is located within a pre-image, a motion command exists which will result in successful insertion. Frequently, however, the peg does not lie in any pre-image of the hole. In such an event, a recursive procedure can be used [11], wherein pre-images, starting with that of the task goal (i.e., the hole), are successively treated as goals for which new pre-images are computed. If a pre-image containing the peg is found at any stage in the recursion, then a trajectory can be planned through the successive pre-images that will result in successful insertion. Other strategies for off-line planning have also been proposed (e.g., [3, 4]).

Off-line planning is based on the assumption that a realistic characterization of the margins of uncertainty is available. While planning, only plans that assure successful insertion even at the limits of the margins of uncertainty are considered. Although it is possible to inflate the margins of uncertainty, thereby ensuring successful physical execution of the generated plan, allowing excessive margins can actually hamper planning by eliminating physically realizable solutions.

Methods based on reactive control, on the other hand, try to counter the effects of uncertainty with on-line modification of the motion control based on sensory feedback. *Compliant* motion control, in which the motion trajectory is modified by contact forces

We thank Kamal Souccar for his help with the Zebra Zero. This material is based upon work supported by the Air Force Office of Scientific Research, Bolling AFB, under Grant AFOSR-89-0526 and by the National Science Foundation under Grants ECS-8912623, CDA-8921080, and CDA-8922572.

[1] Although these methods have been developed for general robotic tasks involving physical interactions between the robot and its environment, for convenience, we will discuss them from the point of view of peg-in-hole insertion.

or tactile stimuli occurring during the motion, is often used, with the compliant behavior either being actively generated or occurring passively due to the physical characteristics of the robot. For the peg-in-hole task, compliance-based strategies are usually keyed on the forces due to interactions between the peg and the hole. The remote center compliance (RCC) device [15], for example, is a passive mechanical device that embodies a linear compliance mapping from forces to corrective motions for small initial positioning errors. However, as Asada [1] points out, many tasks including the peg-in-hole task require complex nonlinear compliance or admittance behavior that is beyond the capability of a passive mechanism. Hence there is great interest in active generation of compliant behavior. But humans find it quite difficult to prespecify appropriate compliant behavior [11], especially in the presence of uncertainty and noise, and hence techniques for learning compliant behavior can be very useful.

In this paper, we describe an approach to learning a reactive control strategy for peg-in-hole insertion under uncertainty and noise. This approach is based on active generation of compliant behavior using a nonlinear admittance mapping from sensed positions and forces to velocity commands. The controller learns this mapping through repeated attempts at peg insertion. In the next section, we describe a learning paradigm that can be used to train such a controller. Following this, we present an implementation of a controller trained to perform peg-in-hole insertions using a Zebra Zero robot. Finally, the performance of this controller is presented and discussed.

2 Learning reactive control

Our approach to learning control is based on a learning paradigm called *associative reinforcement learning* [2]. In associative reinforcement learning tasks, the learning system interacts in a closed loop with its environment. At each time step, the environment provides input to the learning system based on which the learning system generates an action. Based on both the input to the learning system and the action it generated for that input, the environment computes and returns an evaluation or "reinforcement". Over time, the learning system has to learn to respond to each input with the action that has the highest expected evaluation. In order to iteratively improve the evaluation obtained for the action associated with each input, the learning system has to determine how modifying the action affects the ensuing evaluation, for example, by estimating the gradient of the evaluation with respect to its actions. Direct reinforcement learning methods rely on stochastic perturbation of the learning system's actions in order to estimate this gradient (e.g., [2, 16, 7]), whereas indirect methods construct and use a model of the evaluation function to estimate the gradient (e.g., [9, 14]).

In principle, both direct and indirect learning methods can be used to train controllers to perform peg-in-hole insertion. However, in the presence of uncertainty and noise, hand-crafting or learning an adequate model of the evaluation function—imperative if one is to use indirect methods for training the controller—can be very difficult. Therefore direct reinforcement learning methods are preferable in such situations. Peg-in-hole insertion under uncertainty and noise is also an example of a control task where it is difficult to obtain training information in the form of prespecified instructions on how to perform the task, while, at the same time, evaluating the performance of a controller on the task is fairly straightforward. Direct reinforcement learning methods can be very useful for learning control in such situations because these methods can discover the appropriate control action for a given input through search guided by the evaluation.

Our approach is somewhat similar to an earlier attempt to learn a compliant control strategy by Simons et al. [13], in which stochastic automata were used to learn corrections to the motion of the peg in a plane perpendicular to the direction of insertion, based on the forces sensed in that plane. More recently, Asada [1] used a multi-layer connectionist network to learn a non-linear compliance mapping from forces to motion corrections in a peg-in-hole insertion task. However, no performance results were given, and it is not clear how well the learned compliance behavior improves peg-in-hole insertion in the presence of uncertainty. Note that this Asada's approach relies on training information in the form of human-specified compliant motions, which, as mentioned earlier, are difficult to obtain in the presence of uncertainty and noise.

In another recent study, Lee and Kim [10] used an expert system based on a paradigm called EARSA (Expert Assisted Robot Skill Acquisition) to learn skilled peg-in-hole insertions starting with an initial expert-specified rule base. As in our approach to the peg-in-hole task, uncertainty and noise were handled by learning a compliant control behavior based on position and force feedback. Although implemented in a symbolic system, their approach—in which random search among the expert's rules is used to discover the best control action in the face of uncertainty and noise—is similar to ours, and we compare these two approaches in some detail in Section 5.

3 Implementation details

We demonstrate the direct reinforcement learning approach to learning control by training a controller to perform peg-in-hole insertions using a Zebra Zero robot. This robot is equipped with a wrist force sensor, and can also sense positions of the joints of the arm through position encoders. As an initial implementation, a two-dimensional version of the peg-in-hole task, depicted in Figure 1, was attempted. In the experiments reported here, the peg used was $50mm$ long and $22.225mm$ ($7/8in$) wide, while the hole was $23.8125mm$ ($15/16in$) wide. Thus the clearance between the peg and the hole was $0.79375mm$ ($1/32in$). Six inputs were provided to the controller at each time step: the position of the peg (X, Y, Θ), computed

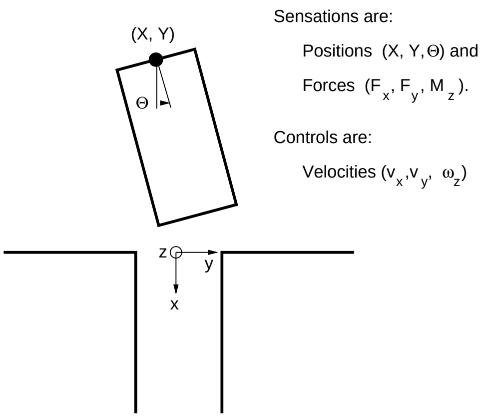


Figure 1: *The peg-in-hole task.*

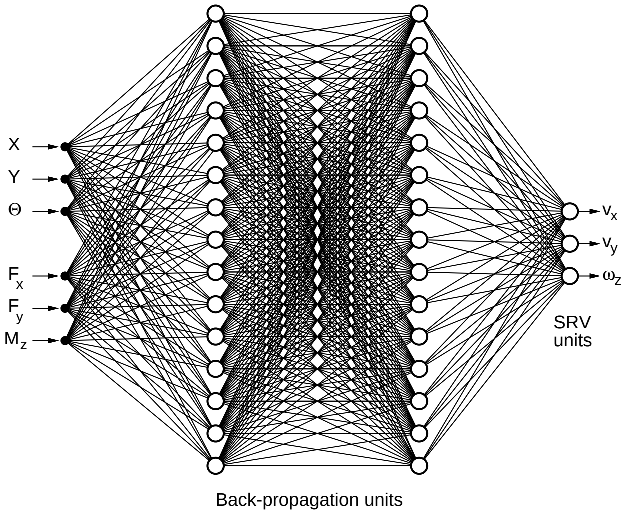


Figure 2: *The network used for the peg-in-hole task. The network has an input layer of 6 units, two hidden layers of 15 back-propagation units each and an output layer of 3 SRV units. All the units in a layer are connected to all the units in the succeeding layer.*

from the sensed joint positions of the robot arm, and the force and moment sensations (F_x, F_y, M_z). Using these inputs, the controller had to compute a velocity command for that time step. Because it was acting in a closed loop with the robot, the controller could learn a reactive, or closed-loop, control strategy for performing the insertion task.

3.1 The controller

Direct reinforcement learning methods have been developed mainly within the framework of connectionist networks. Moreover, several methods exist for training connectionist networks to compute non-linear functions of their inputs. We therefore implemented the controller using a connectionist network shown in Figure 2. A special feature of the network used in our implementation was that the output layer of the network consisted of three stochastic real-valued (SRV) units [7]. The SRV unit has been designed specifically for direct reinforcement learning; the SRV unit generates real-valued output stochastically and uses the ensuing evaluations to adjust its output so as to maximize the expected evaluation over time [7]. The unit does this by estimating the local gradient of the evaluation with respect to its output and using this estimate to perform gradient ascent (see [7] for details). Using the SRV units in the output layer enabled the network to conduct a search in the space of control actions in order to discover appropriate compliant behavior. Additionally, using a multilayer network with back-propagation units (e.g., [12]) permitted the controller to compute a nonlinear mapping from inputs to control actions.

3.2 Training methodology

The controller was trained in a sequence of training runs, each of which started with the peg at a random position and orientation with respect to the hole and ended either when the peg was successfully inserted to a depth of $35mm$ into the hole, or when 100 time steps had elapsed. The following interaction took place between the controller and the robot at each time step during training. To begin, the sensed peg position and forces were input to the controller network, which used them to compute a control action. This action was executed by the robot, resulting in some motion of the peg. The network's output was then evaluated based on the new peg position and the forces acting on the peg. The evaluation, r, which ranged from 0 to 1 with 1 denoting the best possible evaluation, was computed as

$$r = \begin{cases} r_{\text{base}} & \text{if all forces are} \leq 0.5N, \\ r_{\text{base}} - 0.1F_{\max} & \text{otherwise,} \end{cases}$$

where $r_{\text{base}} = \max(0, 1 - 0.01(|X| + |Y|) - |\Theta|)$ and $F_{\max}$ denotes the largest magnitude force component. As this equation indicates, the evaluation essentially depended on the discrepancy between the current sensed position (X, Y, Θ), of the peg and the desired position, $(0, 0, 0)$, with the peg inserted in the hole. However, a penalty term was subtracted from this evaluation when any of the sensed forces on the peg exceeded a preset maximum of 0.5 *Newtons*. Using this evaluation, the controller network adjusted its actions appropriately and the cycle was repeated.

4 Performance Results

A learning curve showing the final evaluation over 500 consecutive training runs is shown in Figure 3. The final evaluation levels off after about 150 training runs because after that amount of training, the controller is consistently able to perform successful insertions within 100 time steps. However, performance as measured by insertion time continues to improve, as is indicated by the learning curve in Figure 4, which shows the time to insertion decreasing continuously over the 500 training runs. These curves indicate that the controller becomes progressively more skillful at peg insertion with training. In order to further characterize the learning behavior, the performance of the controller was tested on several test cases after 100, 200, 300, 400, and 500 training runs. In these test cases, insertion was attempted with the peg at fixed initial locations and orientations. Figure 5 shows sample performance on one test case in terms of the evaluation received over the course of insertions starting from an initial location of ($-20mm$, $-25mm$, $-0.15rad$). In addition to the obvious speedup in insertion time, another characteristic of the learning behavior can be observed from this figure by contrasting the curves showing performance after 100 and after 500 training runs. The presence of jitter in the evaluation after 100 training runs indicates that the force threshold is being violated often, causing drops in the evaluation. This jitter is not present in the curve showing performance after 500 training runs, indicating that the controller has learned to perform insertions without causing excessive forces to act on the peg.

We should emphasize here that our learning approach resulted in good control performance despite the high degree of noise in the sensations returned by the Zebra Zero. Figure 6 shows graphs of the various sensations returned by the Zebra Zero during an insertion starting with the configuration shown in Figure 5. We consider this demonstration of our approach in the real-world, as opposed to in a simulated system, very significant, since simulations rarely capture all the essentials of real-world problems.

5 Discussion and conclusions

In this section, we compare and contrast our approach to the peg-in-hole insertion task with that of Lee and Kim [10]. Lee and Kim [10] reported experiments using a simulated two-dimensional peg and hole that demonstrated how their system could learn new rules that improved insertion speed and avoided jamming. Only three different initial peg locations were used. In their experiments, the peg used was $30.48mm$ long and $12.7mm$ ($0.5in$) wide, while the hole was 12.954 ($0.51in$) wide. Thus the clearance between the peg and the hole was $0.127mm$ ($0.005in$). They also corrupted the simulated sensory feedback of positions and forces with mean-zero gaussian noise which had a standard deviation of $0.00127mm$ and was bounded to be less than $0.0127mm$ in magnitude. Note that the maximum position error is therefore an order of magnitude smaller than the clearance between the peg and the hole.

While the performance results reported by Lee and Kim [10] are comparable to those obtained using our approach, several factors suggest that our approach might have greater advantages. To begin with, Lee and Kim had to use discretized representations of the positions, forces, and velocities in their symbolic system, whereas we could use continuous-valued variables to represent these quantities in our learning system. This is a major advantage because one does not have to deal with the problem of selecting appropriate discrete representations for quantities that are naturally continuous. Second, Lee and Kim's controller started with an initial expert rule base that enabled it to perform insertions even before any learning took place. It is not clear how good this initial knowledge has to be for their approach to work in general. In contrast, our controller could learn skilled peg insertion starting with no initial control knowledge.

Our results also indicate that our approach works well with a physical system, despite the much higher magnitudes of noise and consequently greater degree of uncertainty inherent in dealing with physical systems. For example, while the maximum error in the sensed position of the peg was a degree of magnitude smaller than the clearance between the peg and the hole in Lee and Kim's simulated system, for the Zebra Zero, the position error was frequently more than three times the clearance.

In conclusion, our results indicate that direct reinforcement learning can be used to learn a reactive control strategy that works well even in the presence of a high degree of noise and uncertainty. The reinforcement learning approach to training the controller can be considered an approach for automatically synthesizing robot control strategies that satisfy constraints encoded in the performance evaluations.

References

[1] H. Asada. Teaching and learning of compliance using neural nets: Representation and generation of nonlinear compliance. In *Proceedings of the 1990 IEEE International Conference on Robotics and Automation*, pages 1237–1244, 1990.

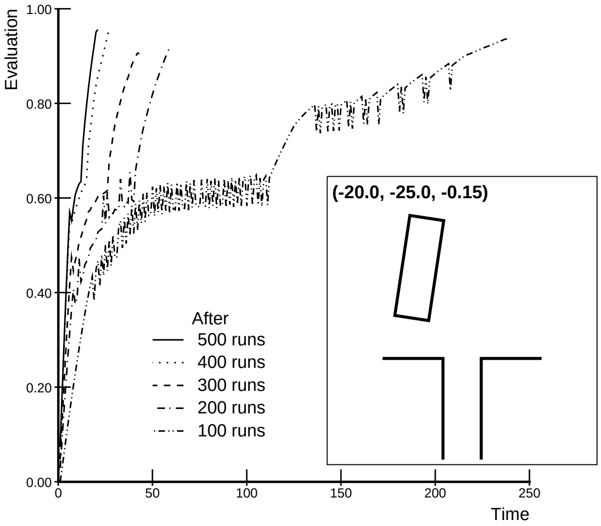
[2] A. G. Barto and P. Anandan. Pattern recognizing stochastic learning automata. *IEEE Transactions on Systems, Man, and Cybernetics*, 15:360–375, 1985.

[3] M. E. Caine, T. Lozano-Pérez, and W. P. Seering. Assembly strategies for chamferless parts. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 472–477, May 1989.

[4] B. R. Donald. Robot motion planning with uncertainty in the geometric models of the robot and environment: A formal framework for error detection and recovery. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1588–1593, 1986.

[5] M. Erdmann. Using backprojections for fine motion planning with uncertainty. *International Journal of Robotics Research*, 5(1):19–45, 1986.

[6] S. J. Gordon. *Automated assembly using feature localization.* PhD thesis, Massachusetts Institute



Performance on the peg insertion task

Figure 5: *Peg-in-hole task: Performance on a test case after various intervals of training. The inset shows the initial position and orientation of the peg for this test case.*

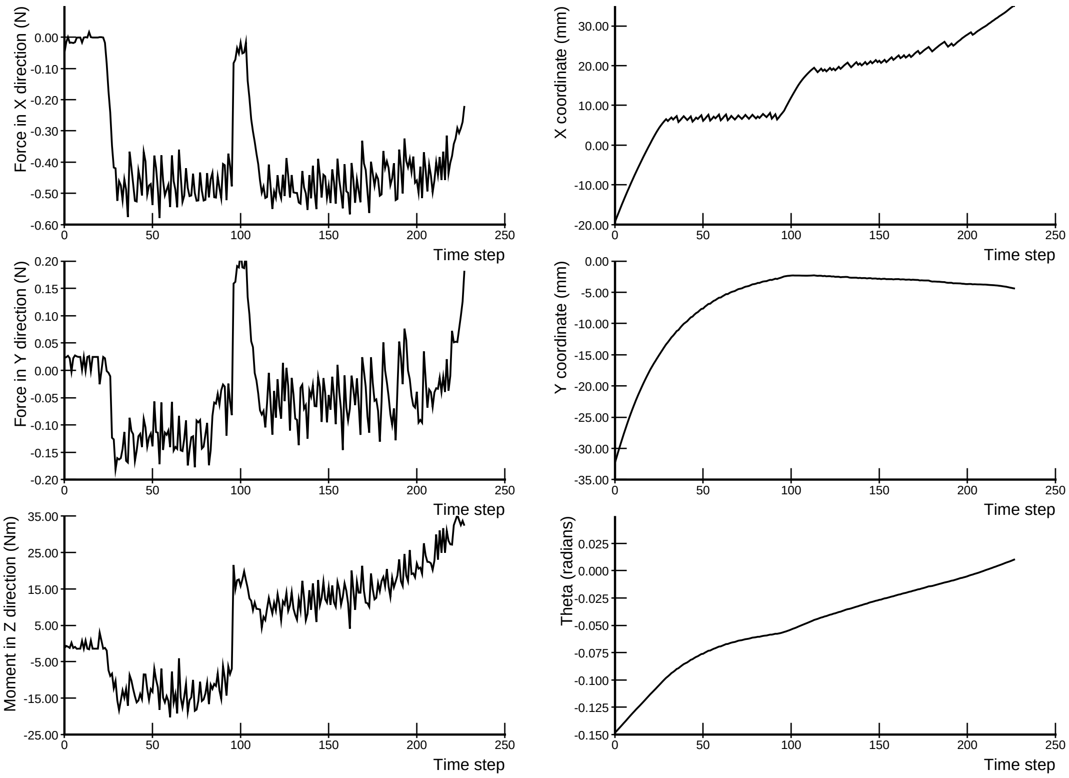


Figure 6: *Peg-in-hole task: Sensory feedback during the course of insertion for the test case in Figure 5. These data were obtained after the controller had completed 100 training runs.*

of Technology, MIT AI Laboratory, Cambridge, MA, 1986. Technical Report 932.

[7] V. Gullapalli. A stochastic reinforcement learning algorithm for learning real-valued functions. *Neural Networks*, 3:671–692, 1990.

[8] R. E. Gustavson. A theory for the three-dimensional mating of chamfered cylindrical parts. *Journal of Mechanisms, Transmissions, and Automated Design*, December 1984.

[9] M. I. Jordan and D. E. Rumelhart. Forward models: Supervised learning with a distal teacher. Center for Cognitive Science Occasional Paper #40, Massachusetts Institute of Technology, Cambridge, MA, 1990.

[10] S. Lee and M. H. Kim. Learning expert systems for robot fine motion control. In H. E. Stephanou, A. Meystal, and J. Y. S. Luh, editors, *Proceedings of the 1988 IEEE International Symposium on Intelligent Control*, pages 534–544, Arlington, Virginia, USA, 1989. IEEE Computer Society Press: Washington.

[11] T. Lozano-Pérez, M. T. Mason, and R. H. Taylor. Automatic synthesis of fine-motion strategies for robots. *The International Journal of Robotics Research*, 3(1):3–24, Spring 1984.

[12] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol.1: Foundations*. Bradford Books/MIT Press, Cambridge, MA, 1986.

[13] J. Simons, H. V. Brussel, J. D. Schutter, and J. Verhaert. A self-learning automaton with variable resolution for high precision assembly by industrial robots. *IEEE Transactions on Automatic Control*, 27(5):1109–1113, October 1982.

[14] P. J. Werbos. A menu of designs for reinforcement learning over time. In T. Miller, R. S. Sutton, and P. J. Werbos, editors, *Neural Networks for Control*, chapter 3. The MIT Press, Cambridge, MA, 1990.

[15] D. E. Whitney. Quasi-static assembly of compliantly supported rigid parts. *Journal of Dynamic Systems, Measurement, and Control*, 104, March 1982. Also in *Robot Motion: Planning and Control*, (Brady, M., et al. eds.), MIT Press, Cambridge, MA, 1982.

[16] R. J. Williams. On the use of backpropagation in associative reinforcement learning. In *Proceedings of the IEEE International Conference on Neural Networks*, San Diego, 1988.

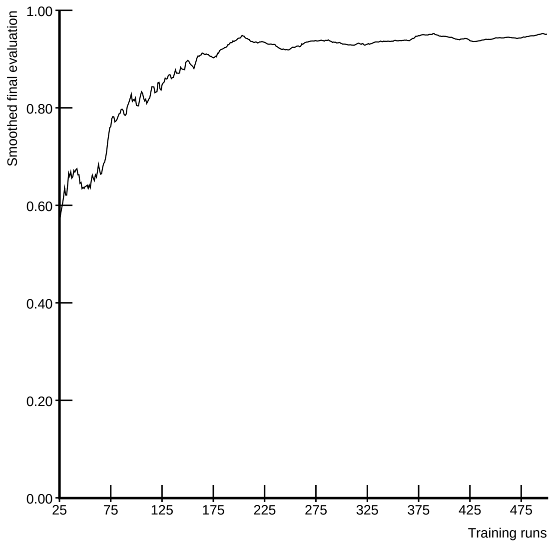


Figure 3: *Peg-in-hole task: Smoothed final evaluation received over 500 consecutive training runs. The smoothed curve was obtained by filtering the raw data using a moving-average window of 25 consecutive values.*

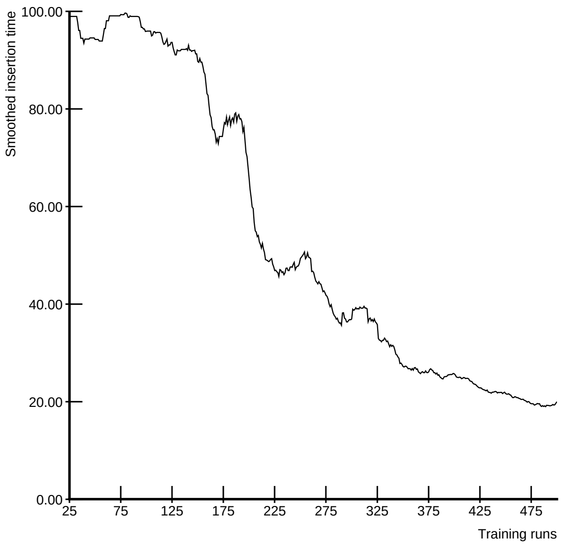


Figure 4: *Peg-in-hole task: Smoothed insertion time (in simulation time steps) taken on each of 500 consecutive training runs. The smoothed curve was obtained by filtering the raw data using a moving-average window of 25 consecutive values.*