SHAPING AS A METHOD FOR ACCELERATING REINFORCEMENT LEARNING

Vijaykumar Gullapalli and Andrew G. Barto
Computer Science Department
University of Massachusetts
Amherst, MA 01003
U.S.A.

Abstract

Shaping is an ancient animal training procedure that has also been studied by experimental psychologists. The principle underlying shaping is that learning to solve complex problems can be facilitated by first learning to solve related simpler problems. Although defining related simpler problems might be difficult when cognitive behavior such as language or mathematics is to be learned, it is relatively easy to determine a sequence of approximations that will lead to mastery of a target *physical* behavior. Indeed, shaping has been most useful for teaching motor skills to animals, and, for the same reason, shaping can also prove useful for training artificial learning systems to perform as controllers. In this paper, we present experimental results illustrating the utility of shaping in training controllers via reinforcement learning methods.

1 Introduction

Shaping has been used for hundreds of years to train animals and has been studied by experimental psychologists [4] interested in animal learning. The principle underlying shaping is that learning to solve complex problems can be facilitated by first learning to solve related simpler problems. The term "shaping" itself has been attributed to the psychologist Skinner [7], who used the technique to train animals such as rats and pigeons to perform complicated sequences of actions for rewards. Skinner describes how the technique is used to train pigeons to peck at a specific spot:

> We first give the bird food when it turns slightly in the direction of the spot from any part of the cage. This increases the frequency of such behavior. We then withhold reinforcement until a slight movement is made toward the spot. ...We continue by reinforcing positions successively closer to the spot, then by reinforcing only when the head is moved slightly forward, and finally only when the beak actually makes contact with the spot...
>
> The original probability of the response in its final form is very low; in some cases it may even be zero. ...By reinforcing a series of successive approximations, we bring a rare response to a very high probability in a short time. ...The total act of turning toward the spot from any point in the box, walking toward it, raising the head, and striking the spot may seem to be a functionally coherent unit of behavior; but it is constructed by a continual process of differential reinforcement from undifferentiated behavior, just as the sculptor shapes his figure from a lump of clay. (Skinner [7] pp. 92-93)

The phrase "...reinforcing a series of successive approximations..." expresses the essence of shaping. Given the task of training an animal to produce complex behavior, the trainer has to be able to (1) judge what constitutes an approximation to, or a component of, the target behavior, and (2) determine how to differentially reinforce successive approximations so that the animal easily learns the target behavior. Unfortunately, neither of these two components of shaping have been formalized rigorously in the psychology literature, even though shaping is widely used both in psychological studies and to train pets and circus animals. Staddon [8], for example, observes that the trainer often has to rely on an intuitive understanding of the way the animal's behavior is generated when determining which behavioral variations are precursors to the target behavior and how to reinforce these precursors. Variations in the behavior of individual animals also must be accounted for when making these judgements.

This research was supported by funding provided to Andrew G. Barto by the Air Force Office of Scientific Research under Grant AFOSR-89-0526, and by the National Science Foundation under Grant ECS-8912623. Additional support was obtained from the National Science Foundation grants IRI-9116297 and CDA-8922572 to the Laboratory for Perceptual Robotics.

The limitations of relying on intuition when judging behavioral distances are especially obvious when the behavior under consideration is cognitive in nature (for example, learning language or mathematics). However, when the physical behavior of the animal is being shaped, behavioral distances can be treated as physical distances, and hence it is easier to determine a sequence of behavioral approximations that will lead to mastery of the target behavior. It is therefore not surprising that shaping has been used most often for teaching motor skills to animals. For the same reason, shaping can also prove useful for training artificial learning systems to perform as controllers.

Several connectionist researchers have noted that training a controller to perform one task can facilitate its learning a related second task (e.g., [6, 2, 10]). Selfridge, Sutton, and Barto [6] studied the effect of shaping over time when training a controller to balance a pole mounted on a cart. They observed that overall learning times were typically shorter when an existing controller was retrained whenever modifications were made to the cart-pole system than when a new controller was trained from scratch. This was demonstrated for several types of modifications including increasing the mass of the pole, shortening the pole, and shortening the track.

Wieland [10] illustrated the utility of shaping using a different version of the cart-pole task in which the controller had to simultaneously balance two poles mounted on a cart. Because it is easier to solve the two-pole balancing problem when the pole lengths are very different than when the pole lengths are almost equal, Wieland trained a controller to balance poles of lengths 1.0m and 0.9m by starting with poles of lengths 1.0m and 0.1m and gradually increasing the length of the shorter pole to 0.9m. Although it is very difficult to balance poles with lengths as close as 1.0m and 0.9m, the shaping process resulted in a controller that was able to do so. Wieland and Leighton [9] also studied the utility of shaping schedules for accelerating learning methods based on gradient descent.

Other applications of shaping in connectionist research have been in the area of training recurrent nets. Allen [1] trained recurrent nets to generate long sequences of outputs using a shaping procedure that involved initially training the nets with short target sequences and introducing longer sequences gradually over training. Another related form of shaping is described in Nowlan [5]. In this case, a robust attractor state for a recurrent network is developed by first training from initial states near the attractor, and then gradually increasing the distance of initial states from the attractor.

In this paper, we present experimental results illustrating the utility of shaping in training controllers via reinforcement feedback. In the reinforcement learning paradigm, training information to the controller is in the form of performance evaluations provided by a *critic*. Because appropriate control behavior has to be inferred from this training information, reinforcement learning involves search guided by the critic's evaluations. The paucity of information in the evaluation signal used in reinforcement learning tasks, when compared with the training information available in supervised learning tasks, makes the issue of scaling reinforcement learning methods to more complex tasks important.

Learning complex control behavior can be facilitated by building in some initial control knowledge into the learning controller. In addition, a natural way of introducing domain knowledge, especially in reinforcement learning systems, is through shaping. Domain knowledge can be used to determine (1) a series of approximations to the target behavior and (2) how to differentially reinforce successive approximations to the target behavior. Shaping a reinforcement learning controller's behavior over time by gradually increasing the complexity of the control task as the controller learns makes it possible to scale reinforcement learning methods to more complex tasks. This is illustrated in this paper with the help of an example.

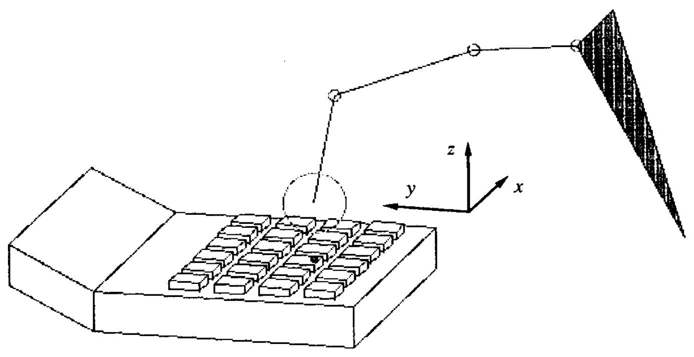


Figure 1: *The task setup for the key-pressing task. The simulated Stanford/JPL hand and the calculator are shown to scale. Only the index finger of the hand is shown because only that finger is used in the task. The triangle represents the palm of the hand; the large circle represents the fingertip. The small dark circle on the calculator face is the "footprint" of the center of the fingertip.*

2 A Test Task: Key-pressing using a robot hand

We demonstrate the utility of shaping using a control task that involves pressing keys on a simulated calculator keypad using the index finger of a simulated dynamic model of the Stanford/JPL hand. The finger has three degrees of freedom and the motion of the hand is restricted to a plane parallel to the x-y plane in which the calculator face lies. Thus there are five degrees of freedom in all to be controlled. The task setup is depicted in Figure 1. The axes of rotation of the finger joints of the Stanford/JPL hand are as follows: the first joint (linking the finger to the palm) permits rotation about an axis parallel to the z-axis, the other two joints have axes of rotation that are perpendicular to both the first link of the finger and the z-axis.

The control actions are positioning commands that locate the hand base in its plane of motion and position the three joints of the index finger. The key to press is specified by setting a single bit in a 24 bit command input vector supplied to the controller. Additional inputs to the controller include proprioceptive feedback of the positions and velocities of the finger joints and the hand, a fingertip force sensation, and a binary "key-pressed" sensation that is set whenever a key is pressed and reset whenever a new target key is specified. For successfully pressing a key, the fingertip must depress the key to the level of the face of the calculator ($z = 0$). Provision of the hand position and velocity feedback permits the controller to learn to press any key starting from any initial hand configuration.

Because pressing a key involves positioning the fingertip over the key, pressing it, and then releasing, the controller has to learn a sequence of actions for executing a single key-press operation. In the case when the reinforcement learning controller starts with no initial control knowledge, the probability of generating such a complex behavior through stochastic search is infinitesimal. A simple evaluation criterion that only signals successful key-presses is therefore not very useful for training such a controller. Fortunately, we can shape the control behavior by defining successive approximations to the key pressing operation and providing more informative differential evaluations that can facilitate learning.

2.1 Training methodology

The first problem in implementing shaping, which is to determine a series of approximations to the target behavior, is not very difficult for the key-pressing task. A fairly intuitive series of approximations to the key-press operation is the following:
(1) Raising the fingertip so that it is not in contact with the keypad surface (to prevent accidental key strikes).
(2) Moving the fingertip towards the target key keeping the fingertip raised.
(3) Positioning the fingertip over the target key keeping the fingertip raised.
(4) Positioning the raised fingertip over the target key and then pressing down with the fingertip.
(5) Positioning the raised fingertip over the target key and pressing down until the key is fully depressed.
(6) Positioning the raised fingertip over the target key, pressing down until the key is fully depressed, and then releasing the key by raising the fingertip.

The second problem in implementing shaping, i.e., deciding how to differentially reinforce control behavior, is more complicated. In order to differentially reinforce the controller as it learns a series of approximations, the critic providing the reinforcement to the controller has to maintain a behavioral history of the controller and infer from the history how well the controller has learned each approximation to the target behavior. Based on this inference, the critic has to determine if the controller requires further training on a particular approximation or if it is ready to be trained on the next, more sophisticated, approximation. While it is wasteful to continue training the controller on an approximation that it has already mastered, switching to a more sophisticated approximation too quickly can also be detrimental to rapid learning of the target behavior. To best realize the benefits of shaping, accurate judgement of the controller's ability at every stage in training is therefore necessary. Clearly, one factor affecting the accuracy of these judgements is the extent to which the controller's behavioral history is maintained.

Our approach to the problem of differentially reinforcing the controller's behavior is to (1) break up the controller's training into *training runs*, i.e., individual attempts at the target behavior that last for a fixed number of time steps, and (2) require that the controller's behavior satisfy the criteria for all the approximations to the target behavior, starting with the simplest, at each time step in a training run. This approach allows the controller's goal to be switched to increasingly sophisticated approximations quickly over training, while at the same time ensuring that the controller is trained on an approximation only if it has successfully met the criteria for all simpler approximations. Moreover, this approach sidesteps the question of how to infer the controller's ability at a stage in training from its behavioral history.

In the key-pressing task, for example, there is a set of evaluation functions, one for each approximation to the target behavior. Each evaluation function assigns a number between 0 and 1 to each state of the hand, with a 1 indicating the best possible evaluation assigned to states in which the corresponding approximate behavior is considered to be accomplished. At every time step in a training run, each evaluation function, starting with the one corresponding to the simplest approximate behavior, is applied to the current state of the hand. The first function returning a value that is not 1 is selected to provide the evaluation to the learning controller at that time step. Thus, the evaluation function selected at each time step corresponds to the simplest approximate behavior that has not been accomplished. For example, if the fingertip is raised but is not located over the target key, an evaluation function that rewards motion towards the target key keeping the fingertip raised is selected; if the fingertip has already been positioned over the target key by the controller, the function selected rewards downward motion of the fingertip while keeping it located over the target key; and so on.

Initially, the controller might spend the entire duration of a training run learning to satisfy the criterion for the simplest approximation. With time, the controller learns to consistently satisfy the criteria for the simpler approximations, and the frontier of learning shifts to approximations closer to the target complex behavior. Thus, most of a training run is spent in learning the approximation to the target behavior at the current frontier of learning.

In order to keep computer simulation time reasonable while retaining all the essential aspects of the key-pressing task, we restricted the choice of the target keys to three keys, which are highlighted in Figure 2. These keys were chosen so as to require a broad range of motion of the fingertip. The controller was trained in a series of training runs, which began with a new target key being picked randomly. The probability of picking the key used in the previous training run was 0.1, while that of picking either of the other two keys was 0.45. The initial hand configuration used in a training run was the configuration of the hand at the end of the previous training run. However, if the previous training run left the fingertip touching the keypad, the last two finger joints were repositioned so that the fingertip was no longer in contact with the keypad. In either case, the initial velocities were set to zero. Each training run lasted 15 time steps, during which the controller was trained using the appropriate evaluation criterion at each time step as described above. The sensory feedback to the controller was also updated at each time step. Details of the implementation of the controller and the simulation of the calculator/hand system are given in [3].

For the purpose of comparison, we also attempted to train a controller on the key-pressing task without resorting to shaping. An identical training procedure was followed in this case, with the sole modification being the use of a single evaluation criterion that only rewarded pressing of the target key.

2.2 Results

The performance of the controller on the key-press task after 25,000 training runs is shown in Figures 3–5. Each figure contains four panels that show the motion of the hand over twelve time-steps when pressing each of the three target keys. Panel (a) contains three strip charts that show the value of each of three quantities at each time step over the course of a key-press operation. These are the distance of the fingertip to the target key (marked D and ranging from 0 to 9 centimeters), the height of the fingertip above the calculator face (marked Z and ranging from 0 to 1 centimeter), and the payoff, or evaluation, (marked P and ranging from 0 to 1). The strip charts show that the fingertip is lowered as it approaches the target key until it makes contact with the key and begins to depress it, and, once the key is fully depressed, the fingertip is raised to release the key. The sharp drop in the evaluation on the time step when the key is fully depressed (i.e., to $z = 0$) is due to the switching of the evaluation criteria from one that rewards downward movement to one that rewards upward movement. Panels (b), (c), and (d) of each figure show the motion of the hand during the key-press operation from three different viewpoints.

As evidenced by these figures, the controller has learned to successfully execute the key-press operation for all three keys. Note that due to the dynamic nature of the hand model, the motion of the fingertip depends on the initial state of the hand. So the figures presented here are merely representative samples. However, we tested the controller's performance with the hand starting in 10,000 random initial states, and in all the test runs, the target key was pressed successfully. In comparison, without shaping, the controller could not learn the key-press task even after 500,000 training runs. Moreover, this was true even when the controller was trained to press just a single target key (the key marked "8"). These results support the observations in Section 2 regarding the difficulty of training a reinforcement learning controller to generate complex behavior without the benefit of shaping due to the improbability of occurrence of the target complex behavior.

3 Conclusions and Comments

The utility of shaping when reinforcement learning is used to solve complex problems is apparent from the empirical results presented in this paper. As discussed in Section 1, shaping can also be useful in other kinds of learning tasks. Moreover, shaping is a natural way to introduce domain knowledge into the learning process. However, rigorously formalized procedures for shaping are lacking in the literature. Systematic studies of shaping by the machine learning community are therefore necessary to fully realize its potential benefits. In particular, procedures for implementing shaping must be formalized and evaluated. An important contribution of this paper is the automated shaping procedure presented in Section 2, which addresses some of the issues in implementing shaping. This general procedure can be used for many different learning problems and can also be easily implemented in other machine learning paradigms.

References

[1] R. B. Allen. Adaptive training of connectionist state machines. In *ACM Computer Science Conference*, Louisville, February 1989.

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

[3] V. Gullapalli. *Reinforcement Learning and its application to control.* PhD thesis, University of Massachusetts, Amherst, MA 01003, 1992.

[4] W. K. Honig and J. E. R. Staddon. *Handbook of operant behavior.* Prentice Hall, Englewood Cliffs, NJ, 1977.

[5] S. J. Nowlan. Gain variation in recurrent error propagation networks. *Complex Systems*, 2:305–320, 1988.

[6] O. Selfridge, R. S. Sutton, and A. G. Barto. Training and tracking in robotics. In *Proceedings of the Ninth International Joint Conference of Artificial Intelligence*, Los Angeles, CA, Aug. 1985.

[7] B. F. Skinner. *Science and Human Behavior.* Macmillan, New York, 1953.

[8] J. E. R. Staddon. *Adaptive behavior and learning.* Cambridge University Press, 1983.

[9] A. P. Weiland and R. R. Leighton. Shaping schedules as a method for accelerating learning. In *International Neural Network Society Meeting*, 1988.

[10] A. P. Wieland. Evolving controls for unstable systems. In D. S. Touretsky, J. L. Elman, T. J. Sejnowski, and G. E. Hinton, editors, *Connectionist Models: Proceedings of the 1990 Summer School.* Morgan Kaufmann Publishers, 2929 Campus Drive, Suite 260, San Mateo, CA 94403, 1991.

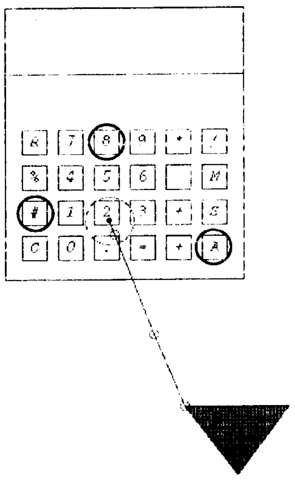


Figure 2. *The set of target keys used in the key-pressing task, shown encircled by bold circles.*

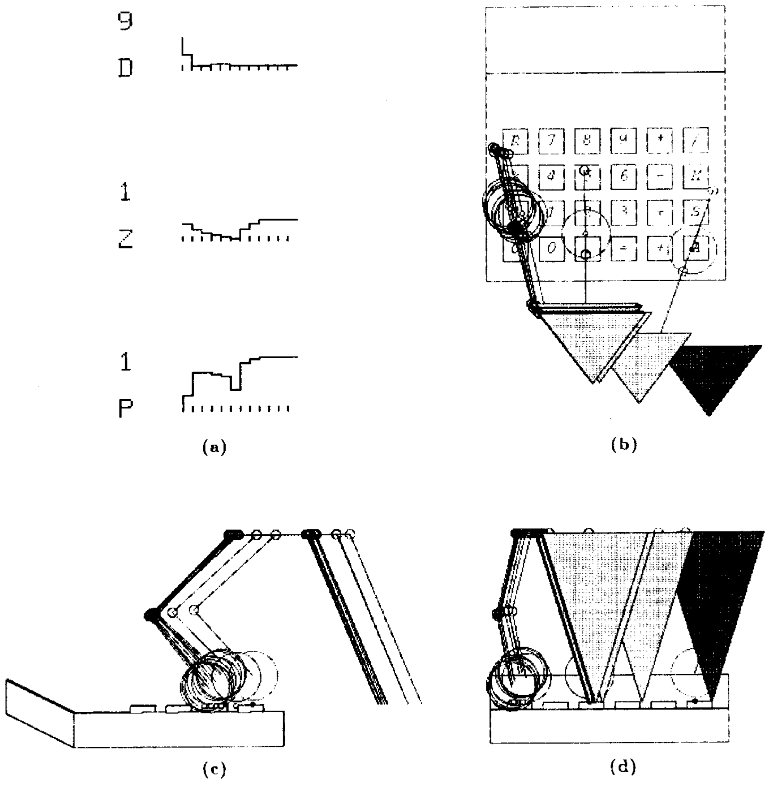


Figure 3: *Pressing the key marked "#" on the calculator keypad. The initial position of the fingertip was the final position after pressing the key marked "A". The dark triangle denotes the initial location of the hand. Panel (a) shows strip charts (top to bottom) of the fingertip's distance to the target key, its height above the keypad, and the payoff, or evaluation, during the course of the key-press operation. Panels (b), (c) and (d) show top, left and bottom views of the hand and the calculator during the course of the key-press operation.*

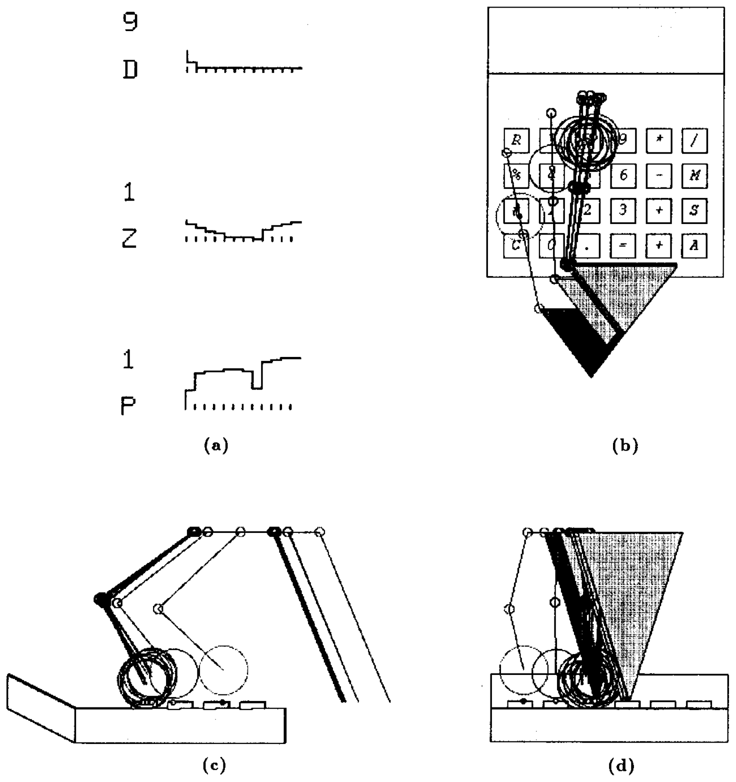
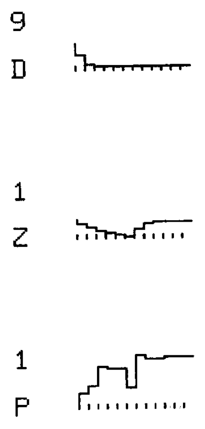
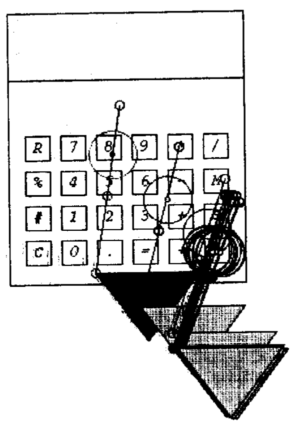


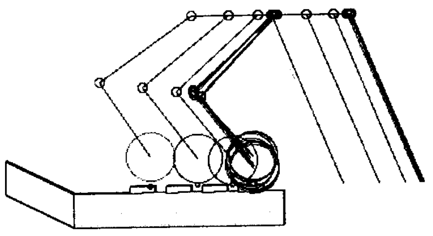
Figure 4: *Pressing the key marked "8" on the calculator keypad. The initial position of the fingertip was the final position after pressing the key marked "#". The dark triangle denotes the initial location of the hand. Panel (a) shows strip charts (top to bottom) of the fingertip's distance to the target key, its height above the keypad, and the payoff, or evaluation, during the course of the key-press operation. Panels (b), (c), and (d) show top, left, and bottom views of the hand and the calculator during the course of the key-press operation.*



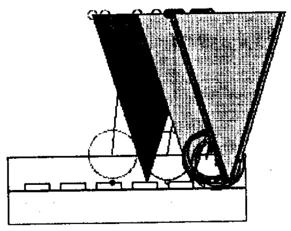
(a)



(b)



(c)



(d)

Figure 5: *Pressing the key marked "A" on the calculator keypad. The initial position of the fingertip was the final position after pressing the key marked "8". The dark triangle denotes the initial location of the hand. Panel (a) shows strip charts (top to bottom) of the fingertip's distance to the target key, its height above the keypad, and the payoff, or evaluation, during the course of the key-press operation. Panels (b), (c), and (d) show top, left, and bottom views of the hand and the calculator during the course of the key-press operation.*