LETTER Communicated by R. Caruana

A Developmental Approach Aids Motor Learning

Volodymyr Ivanchenko
vivanchenko@bcs.rochester.edu
Robert A. Jacobs
robbie@bcs.rochester.edu
Department of Brain and Cognitive Sciences, University of Rochester, Rochester, NY 14627, U.S.A.

Bernstein (1967) suggested that people attempting to learn to perform a difficult motor task try to ameliorate the degrees-of-freedom problem through the use of a developmental progression. Early in training, people maintain a subset of their control parameters (e.g., joint positions) at constant settings and attempt to learn to perform the task by varying the values of the remaining parameters. With practice, people refine and improve this early-learned control strategy by also varying those parameters that were initially held constant. We evaluated Bernstein's proposed developmental progression using six neural network systems and found that a network whose training included developmental progressions of both its trajectory and its feedback gains outperformed all other systems. These progressions, however, yielded performance benefits only on motor tasks that were relatively difficult to learn. We conclude that development can indeed aid motor learning.

1 Introduction

Why is learning to control a motor system, either a human body or a robot, so difficult? At least part of the difficulty arises from the degrees-of-freedom problem. Motor systems often have many degrees of freedom, where the degrees of freedom in a system are the number of dimensions in which the system can independently vary (Rosenbaum, 1991). These degrees of freedom are often redundant, meaning that the number of degrees of freedom of the system carrying out some task exceeds the number of degrees of freedom needed to specify the task to be carried out (Jordan & Rosenbaum, 1989). Importantly, redundancy can occur at many different levels of sensorimotor representation (Saltzman, 1979). For example, consider the problem of touching the tip of your nose. The location of your nose has three degrees of freedom (its x, y, and z position in Cartesian coordinates), but the joints of your arm have seven degrees of freedom (the shoulder has three degrees of freedom, and the elbow and wrist each have two). Consequently, there are many different settings of your arm's joint positions that allow you to

touch your nose. As a second example, consider the problem of controlling your muscle forces in order to apply a specified torque at a joint. Because the joint torque has one degree of freedom and the muscle forces have multiple degrees of freedom, there are many different settings of your muscle forces that all result in the same joint torque. In general, the larger the number of degrees of freedom in a system and the more redundant these degrees of freedom, the more difficult it is to control the system so that it behaves in a desired manner (Jordan & Rosenbaum, 1989).

Bernstein (1967), who first identified the degrees-of-freedom problem, suggested that people attempting to learn to perform a difficult motor task try to ameliorate the degrees-of-freedom problem through the use of a developmental progression. Early in training, people maintain a subset of their control parameters (e.g., joint positions) at constant settings and attempt to learn to perform the task by varying the values of the remaining parameters. With practice, people refine and improve their early-learned control strategy by also varying those parameters that were initially held constant. In other words, novice performers tend to lock a subset of their joint in constant positions. As they gain more experience, they then let these joints move more freely. Experimental evidence in support of this hypothesis comes from several sources (though the hypothesis and the empirical data are topics of recent debate; see Newell and Vaillancourt, 2001) .

Arutyunyan, Gurfinkel, and Mirskii (1968) analyzed the motion and stability of people learning to shoot hand-held pistols. Novice shooters tended to hold their wrists and elbows rigid. As they became more practiced, they tended to unlock their wrists and elbows, and their firing accuracy improved. It was found that unlocking the wrist and elbow allowed the arm to compensate for motion in the hand, and vice versa. Counterintuitively, variability in the position of the gun barrel was reduced toward the end of training even though more motion occurred in the arm and hand than before. A similar developmental trend was found by McDonald, van Emmerik, and Newell (1989) in a study of people practicing a dart-throwing task. Vereijken, van Emmerik, Whiting, and Newell (1992) studied people practicing slalom-like ski movements on a ski apparatus. During the early stages of learning, the joint angles of these people's lower limbs and torso displayed little movement, as measured by the standard deviations and ranges of angular motions. In addition, joint couplings were high, as evidenced by the large cross-correlations between joint angles. With practice, however, angular movement significantly increased in all joint angles of the lower limbs and torso, and the cross-correlations decreased. Berthier, Clifton, McCall, and Robin (1999) studied reaching movements in infants at the age at which they first successfully moved their hands to targets presented in their work space (median age of 11 weeks when first tested). Kinematic analyses revealed that infants tended not to show elbow flexion or extension during reaches. Instead, their hands moved toward the target locations due to shoulder and torso rotations.

Taken as a whole, this collection of results suggests that infants and adults simplify the problem of learning to perform a difficult motor task through the use of a developmental progression. If so, early in training, people maintain a subset of their control parameters (e.g., joint positions) at constant settings and attempt to learn to perform the task by varying the values of the remaining parameters. With practice, they refine and improve this early-learned control strategy by also varying those parameters that were initially held constant. Presumably, this developmental progression is sensible because in early stages of training, it is significantly easier to learn to perform a difficult motor task, albeit in an approximate way, by using a smaller number of control parameters than it is to learn to perform the task using the full set of parameters. Once a person has learned to make an approximate but easily learned motor movement, this knowledge serves as a useful foundation for learning to make a more complex and accurate movement with additional practice.

This article reports the results of a number of simulations in which artificial neural networks were trained to move a robot arm along a reference trajectory by producing appropriate feedforward torques. Some networks were trained in a manner consistent with the developmental progression suggested by Bernstein (1967), whereas the training of other networks did not include a developmental progression. The results indicate that the use of the developmental progression results in significantly better motor performance at the end of training. This developmental progression, however, yielded performance benefits only on motor tasks that were relatively difficult to learn. We conclude that development can indeed aid motor learning.

From a computational perspective, the research that most closely resembles the work reported here is that of Sanger (1994), who also trained neural networks to control robot arms using developmental progressions. In one study, the gains of a feedback controller, which was used in conjunction with a neural network to control the robot arm, were initialized to large values at the start of training and then were gradually reduced during the course of training. In another study, a network was trained to move a robot arm along a reference trajectory at a relatively slow speed at the start of training. During the course of training, the speed at which the arm was required to traverse the trajectory was gradually increased. In both cases, Sanger found that networks trained with a developmental progression outperformed networks that were not. An important difference between our work and that of Sanger is that we are concerned with the specific developmental progression proposed by Bernstein (1967).

2 Data Sets and Training Procedures

We simulated a three-joint robot arm with highly nonlinear dynamics. This arm was identical to the arm simulated by Kawato, Furukawa, and

Table 1: Values of the Physical Parameters of the Robot Arm.

Parameter	First Link	Second Link	Third Link
L_k(m)	0.4	0.4	0.4
S_k(m)	—	0.15	0.15
M_k(kg)	15.0	7.0	3.0
I_{kx}(kg · m^2)	—	0.589	0.251
I_{ky}(kg · m^2)	—	0.584	0.253
I_{kz}(kg · m^2)	0.017	0.00673	0.0034
b_k(kg · m/s)	20.0	15.0	5.0

Note: L_k, S_k, M_k, I_{kx}, I_{ky}, I_{kz}, and b_k are the length, the position of the center of mass, the mass, the three inertial moments, and the viscosity coefficient for the kth link, respectively.

Suzuki (1987) with the exception that our arm did not include a payload. Our simulations used the Matlab Robotics Toolkit (Corke, 1996). The values of the physical parameters of the arm are shown in Table 1.

Results using two reference trajectories are reported in this article. Each reference trajectory was a complex motion in three-dimensional space whose duration was 4 seconds. Each trajectory was computed using the Matlab Robotics Toolbox. In short, there were five moments in time when all joint velocities and accelerations were zero. The joint positions at these moments, referred to as landmark positions, were specified, and the Toolbox computed trajectory values for the joint positions, velocities, and accelerations at all moments in time by fitting a smooth polynomial between these landmark positions. The reference trajectories are shown in Figure 1.

The use of a reference trajectory that specifies the desired joint positions, velocities, and accelerations at each moment in time does not eliminate the degrees-of-freedom problem because this problem typically occurs at many levels of sensorimotor representation (Saltzman, 1979). As indicated above, a person given a reference trajectory at the level of joint variables must still cope with the redundancy arising from the fact that the desired joint values can be achieved using many different settings of the person's muscle forces. Analogously, a neural network given a reference trajectory at the level of joint variables must still cope with the redundancy arising from the fact that the desired joint values can be achieved using many different settings of the network's hidden unit activations or connection weights. Although Bernstein (1967) was primarily concerned with redundancy at the level of joints, our focus is on evaluating the implications of his proposed developmental progression for aspects of motor learning. We do not consider the use of this progression for solving the degrees-of-freedom problem at the level of joints.

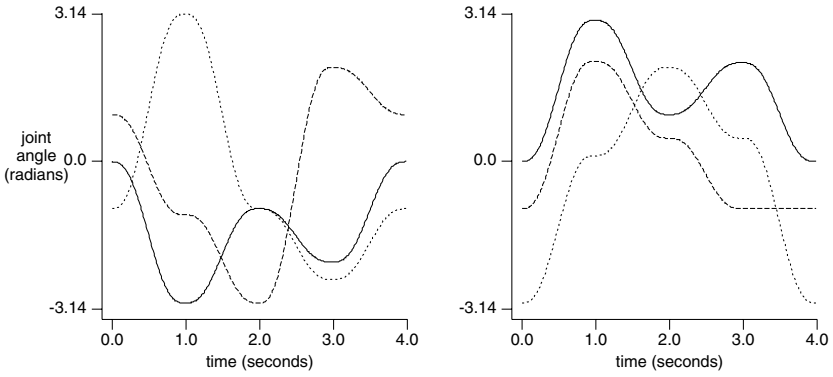

Figure 1: The two reference trajectories. The horizontal axis of each graph represents time; the vertical axis gives the reference joint positions for the shoulder (solid line), elbow (dashed line), and wrist (dotted line).

Neural networks were trained to serve as feedforward controllers for the robot arm. As illustrated in Figure 2, the training procedure involved training an adaptive feedforward controller, implemented as a neural network, to control the arm in conjunction with a fixed feedback controller. The feedback controller aided in generating training data that the feedforward controller used to learn a model of the arm's inverse dynamics. The feedback controller that we used was nearly identical to the one simulated by Kawato et al. (1987). It was a proportional-derivative (PD) controller whose proportional feedback gains were 517.2, 746.0, and 191.4 and whose derivative gains were 16.2, 37.2, and 8.4. The training method is known in the motor control literature as feedback error learning (Jordan & Wolpert, 2000).

Let the state of the arm at time t be represented in joint space by a position vector $\theta(t)$ and a velocity vector $\dot{\theta}(t)$. In order to achieve a desired acceleration $\ddot{\theta}(t)$, an appropriate torque $\tau(t)$ must be applied to the arm. The relationship between acceleration and torque is the inverse dynamics of the arm and is written

$$\tau(t) = f^{-1}(\theta(t), \dot{\theta}(t), \ddot{\theta}(t)). \tag{2.1}$$

The goal of the learning procedure was to train a feedforward controller to model this relationship.

The feedforward controller was trained on-line in the following manner. Torques were applied to the robot arm at a rate of 100 Hz, meaning that discrete time steps occurred at 10 msec intervals. At each time step, the control signal was obtained by summing the outputs of the feedforward and feedback controllers (see the top portion of Figure 2). The inputs to the feedforward controller were the desired joint positions, velocities, and accel-

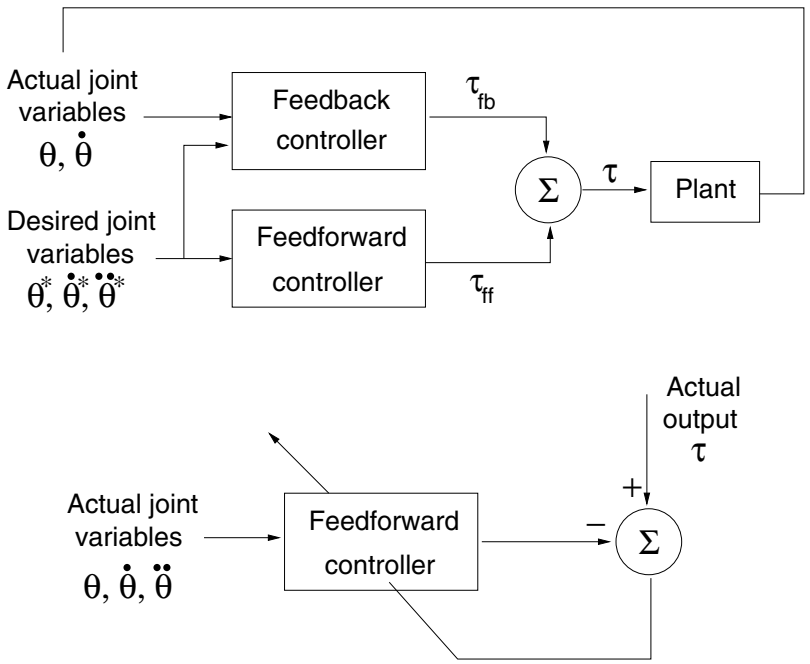

Figure 2: In feedback error learning, an adaptive feedforward controller and a fixed feedback controller are used to control the robot arm. To train the feedforward controller, the actual joint variables are provided as inputs. The target output for the controller is the torque applied to the arm.

erations for the current time step, as specified by the reference trajectory. The inputs to the feedback controller were the desired and actual joint positions and velocities. The sum of the feedforward and feedback control signals is a torque vector that was applied to the arm. The resulting joint accelerations were observed. The feedforward controller then received the actual joint positions, velocities, and accelerations as inputs, and it computed new outputs (see the bottom portion of Figure 2). An error was computed between this output and the actual torques applied to the arm, and the error was used to modify the weights in the controller using the backpropagation algorithm. In this manner, the feedforward controller received samples of the inverse dynamic mapping and learned a model of this mapping by minimizing its prediction error. Early in training, the feedback controller dominated, and the arm followed the desired trajectory imprecisely. As the feedforward controller improved at modeling the arm's inverse dynamics, it began to generate torques that allowed the arm to follow the desired trajectory more faithfully.

Six neural networks were trained to serve as feedforward controllers. Two networks did not use a developmental progression and thus are referred to as control networks. The remaining four networks are referred to as experimental networks because they used a developmental progression. Each network had nine input units corresponding to the positions, velocities, and accelerations of the three joints and three output units corresponding to the feedforward torques at each joint. Unless specified otherwise, each network also had a layer of 25 hidden units (this number was chosen because the performance of the first control network described below on the first reference trajectory did not significantly improve when more hidden units were added to this network). The initial weight values for a unit of a network were randomly sampled from a uniform distribution whose mean was zero and whose standard deviation was equal to the inverse of the square root of the fan-in of the unit (i.e., the number of connections received by the unit; see LeCun, Bottou, Orr, & Müller, 1998). For the purposes of simulation, the joint positions, velocities, and accelerations as well as the joint torques were linearly scaled so that the activations of the input and output units were nearly always between -1 and 1. The hidden units used a hyperbolic tangent activation function, whereas the output units used a linear activation function. The networks were trained to minimize their sum of squared errors. A learning rate of 0.00005 was used because this value resulted in the best performance for the first control network described below on the first reference trajectory. Networks were trained for 1000 epochs, where each epoch is one attempt at controlling the arm to traverse the reference trajectory. We found that the performances of all networks converged at 1000 epochs, but the networks tended not to overfit the training data as measured by their generalization performances on a novel reference trajectory (this trajectory was identical to the first reference trajectory except that the robot arm was required to perform this sequence at a 20% faster rate). The six neural networks were as follows:

1. Nondevelopmental network (*network ND*). This was a conventional neural network. Its training did not include a developmental progression.
2. Modular nondevelopmental network (*network ND-M*). This system was actually three networks—one for each feedforward joint torque. Each network had 9 input units, corresponding to the positions, velocities, and accelerations of the three joints; 25 hidden units; and 1 output unit. The activation of the output unit of network i was a feedforward torque for joint i. A motivation for this three-network system is as follows. A possible advantage of the developmental progression proposed by Bernstein (1967) is that it roughly uncouples the dynamic interactions among the joints by training different joints at different moments in time. We thought it would be interesting, therefore, to evaluate a modular system that did not use a

developmental progression but that did use uncoupled neural networks that independently learned to control the joints.

3. Network with development of its trajectory (*network D-T*). This network was trained in stages. At all times, the joint closest to the base of the robot arm (i.e., the arm's "shoulder") was trained to follow the position, velocity, and acceleration values specified by the reference trajectory. The other two joints (the arm's "elbow" and "wrist"), however, were trained to maintain constant values of their positions during early stages of training. During later stages of training, they were trained to follow the values specified by the reference trajectory more closely. In other words, the network was trained to freeze a subset of its joints (its elbow and wrist) toward the beginning of training and then to vary these joints more freely toward later training periods.

This was implemented as follows. Let $\theta_i^*(t)$ be the desired value of either the elbow's or the wrist's position at time t, where the subscript i indexes either the elbow or wrist joint. Also, let $\theta_{\mathrm{i,ref}}(t)$ be the value of the position of joint i at time t specified by the reference trajectory. For the elbow and wrist joints only, the desired value was a linearly scaled version of the reference value:

$$\theta_i^*(t) = \theta_{\mathrm{i,ref}}(0) + k \times [\theta_{\mathrm{i,ref}}(t) - \theta_{\mathrm{i,ref}}(0)], \tag{2.2}$$

where k is a gain parameter whose value varied from zero to one during the course of training. In other words, the desired value at time t was a linear function of how much the reference value at time t deviated from its initial value at time $t = 0$. The value of k at each epoch of training was given by a logistic function:

$$k(e) = \frac{1}{1 + \exp\{-(e - \mu)/\sigma\}}, \tag{2.3}$$

where e denotes the current epoch, the threshold parameter μ was set to 250 epochs, and the slope parameter σ was set to 12.3. The desired values for joint i's velocity and acceleration at each point in time were computed from the desired position values via differentiation.

4. Network with development of its feedback gains (*network D-G*). This system combines aspects of the ideas of Bernstein (1967) regarding motor learning with the idea of using large feedback gains during early stages of training and then gradually decreasing their values during the course of training. At all times, the feedback gains for the shoulder joint were set to the values given above. However, the feedback gains for the elbow and wrist joints were set to very large values (four times the values given above) during early training stages and then gradually decreased to the values given above.

This was implemented as follows. Let $\vec{\mathbf{g}}_i^*$ be a vector whose elements are the feedback gains corresponding to either the elbow or wrist. The subscript i

indexes either the elbow or wrist joint. Let $\vec{\mathbf{g}}_{\mathrm{i,norm}}$ be a vector whose elements are the normal feedback gains for joint i, meaning the values of the gains given above. For the elbow and wrist joints only, the feedback gains were scaled versions of their normal values,

$$\vec{\mathbf{g}}_i^* = k \times \vec{\mathbf{g}}_{\mathrm{i,norm}}, \tag{2.4}$$

where k is a gain parameter whose value varied from four to one during the course of training. The value of k at each epoch of training was given by the equation

$$k(e) = 4 - \frac{3}{1 + \exp\{-(e - \mu)/\sigma\}}, \tag{2.5}$$

where e denotes the current epoch, the threshold parameter μ was set to 250 epochs, and the slope parameter σ was set to 12.3.

5. Network with development of its trajectory and of its feedback gains (*network D-TG*). This system combined the trajectory development of network D-T with the feedback gain development of network D-G. We predicted that this system would show the best performance. Early in training, the desired values of the elbow's and wrist's positions were set to constant values. Indeed, these joint variables were nearly frozen at these constant values due to the high feedback gains on these joints. Importantly, this greatly simplified the learning problem for the network during early training stages because it was only the joint variables corresponding to the shoulder that showed significant variation during this time period. We reasoned that the knowledge gained during these early training stages would provide a useful foundation for further learning at later training stages when the desired joint values for all joints are closer to those specified by the reference trajectory and when the feedback gains are closer to their normal values.

Among all the networks we simulated, network D-TG is the one most broadly consistent with the ideas of Bernstein (1967). Note that this network uses two types of developmental progressions: identical to network D-T, it uses a progression of its trajectory, and identical to network D-G, it uses a progression of its feedback gains. By comparing the performance of this network with those of networks D-T and D-G, we can evaluate the relative benefits of each of these progressions.

6. Network with reverse development of its trajectory and of its feedback gains (*network RD-TG*). This network was identical to network D-TG except that the developmental progression proceeded in the opposite order. Early in training, the desired values of the shoulder's positions were set to constant values, and the feedback gains for the shoulder were set to large values. As training progressed, the desired values of the shoulder's positions were set closer to those of the reference trajectory, and the feedback gains for the shoulder were set closer to their normal values. In contrast,

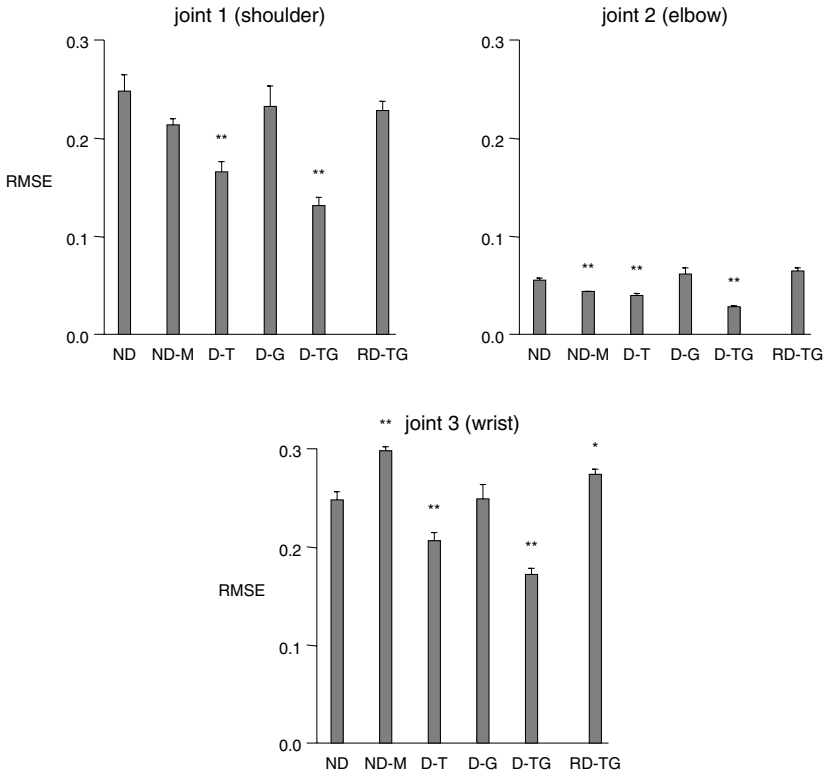

Figure 3: Simulation results using the first reference trajectory.

the elbow and wrist joints did not undergo a developmental progression. At all times, the desired values of their positions, velocities, and accelerations were determined by the reference trajectory, and the feedback gains for these joints were set to their normal values. A motivation for this system is to evaluate the extent to which the order of the developmental progression, from either shoulder to elbow and wrist or from elbow and wrist to shoulder, influences a system's performance.

3 Simulation Results

The simulation results using the first and second reference trajectories are shown in Figures 3 and 4, respectively. The horizontal axis of each graph gives the network. The vertical axis gives the root mean squared error (RMSE) at the end of training based on the difference at each point in time between a joint's actual position (in radians) and its desired position as spec-

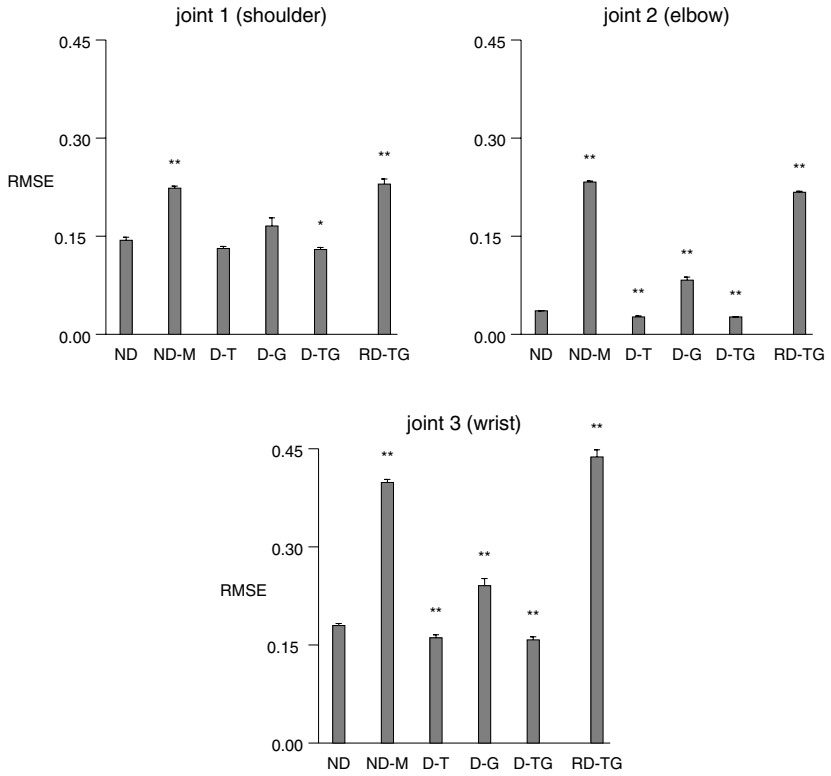

Figure 4: Simulation results using the second reference trajectory.

ified by the reference trajectory. The upper-left graph in each figure is for joint 1 (shoulder), the upper-right graph is for joint 2 (elbow), and the lower graph is for joint 3 (wrist). The error bars give the standard errors of the means based on 20 simulations of each network. Our primary interest is in whether the developmental networks outperformed the nondevelopmental network ND. Statistical tests (two-tailed t-tests with 38 degrees of freedom) were conducted to evaluate whether each network's performances were significantly different from network ND's performances. Bars in Figures 3 and 4 that have one star above them indicate that a network's performance is significantly different at the $p < 0.05$ level; two stars indicate that a network's performance is significantly different at the $p < 0.01$ level.

Consistent with our prediction, network D-TG showed the best performance in the sense that it had the smallest errors. On both the first and second reference trajectories, this network had significantly smaller errors than network ND for all joints. Network D-T also showed relatively good

performance. On the first trajectory, it had significantly smaller errors than network ND for all three joints; it had significantly smaller errors for two of three joints on the second trajectory. These results are important because they demonstrate that a system that uses a developmental progression can outperform one that does not.

In contrast, developmental networks D-G and RD-TG did not show superior performances. This result is important because it demonstrates that not all developmental progressions result in performance benefits. To the contrary, only progressions whose characteristics are well matched to the nature of the task should be expected to yield superior performance. It is interesting to note that even though a developmental progression of the feedback gains by itself did not lead to performance benefits, as evidenced by the poor performance of network D-G, it did lead to enhanced performance when used in combination with a developmental progression of the trajectory, as evidenced by the superior performance of network D-TG relative to network D-T. It is also interesting to note that a developmental progression that proceeds from the shoulder to the elbow and wrist yields performance benefits relative to a progression that proceeds in the opposite order, as evidenced by the superior performance of network D-TG relative to network RD-TG.

We have shown the performances of the six networks on two sample reference trajectories. Overall, the pattern of performances suggests the usefulness of the progression proposed by Bernstein (1967). It is important to consider the generality of these results by asking whether the same pattern of results is found using other reference trajectories. Although the results are not presented here, we have evaluated the networks' performances using a wide variety of trajectories. Our findings can be summarized as follows. If a reference trajectory was relatively easy to learn, meaning for our purposes that the nondevelopmental network ND was able to learn to control the robot arm to follow the trajectory with relatively small error, then the use of the developmental progressions considered here tended not to result in performance benefits. Fortunately, they also did not result in performance decrements (meaning they did not have an important impact on final performances). In contrast, if a reference trajectory was relatively difficult to learn, meaning for our purposes that network ND learned to control the arm to follow the trajectory with relatively large error, then the developmental progressions used by networks D-TG and D-T tended to yield performance benefits as reported above.

Because the robot arm had highly nonlinear dynamics, it is unreasonable to expect an adaptive feedforward controller trained with one reference trajectory to generalize well to novel trajectories. Nonetheless, we report the results of such a generalization test here. The novel trajectory used the same sequence of joint positions as the first reference trajectory, though the robot arm was required to perform this sequence at a 20% faster rate.

The results are shown in Figure 5. Network D-TG, the system whose developmental progression is closest to the progression proposed by Bern-

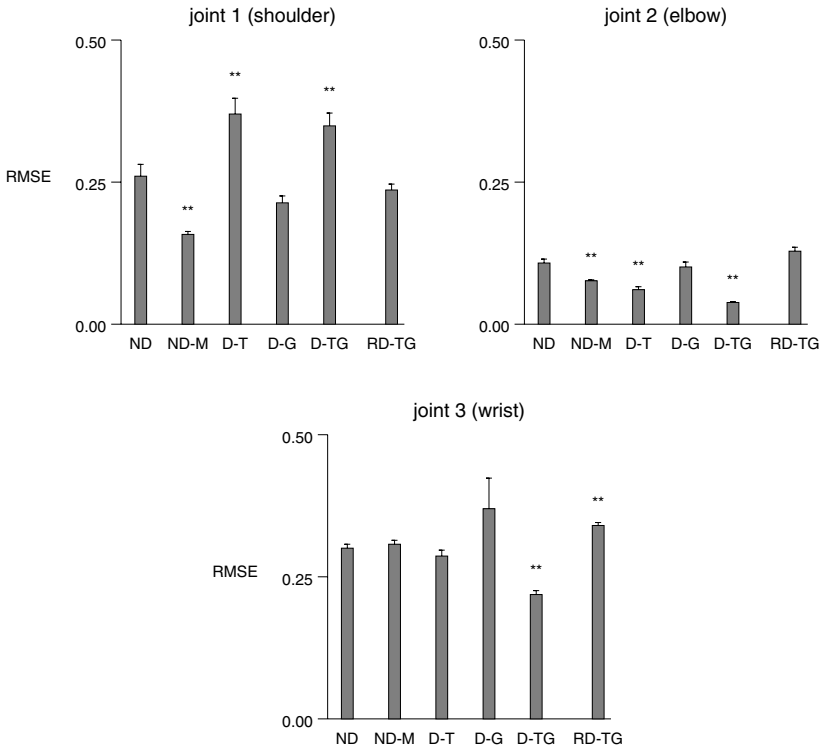

Figure 5: The joint position errors when tested on the generalization trajectory.

stein (1967), outperformed all other networks when we consider generalization errors for the elbow and wrist joints. However, it performed relatively poorly when we consider generalization errors for the shoulder joint. A possible explanation for this result is to note that the set of desired values for the elbow and wrist varies widely during the course of training due to the developmental progression, whereas the set of desired values for the shoulder remains fixed. Based on this result, we speculate that the developmental progression proposed by Bernstein may help leaners to perform their target motor task accurately, but it will not necessarily result in important performance improvements on other tasks.

4 Summary

In summary, Bernstein (1967) suggested that people attempting to learn to perform a difficult motor task try to ameliorate the degrees-of-freedom problem through the use of a developmental progression. Early in training, people maintain a subset of their control parameters (e.g., joint positions)

at constant settings and attempt to learn to perform the task by varying the values of the remaining parameters. With practice, they refine and improve this early-learned control strategy by also varying the parameters that were initially held constant. We evaluated Bernstein's proposed progression using six neural network systems: (1) network ND did not use a developmental progression in its training; (2) network ND-M did not use a developmental progression, though it used separate networks to learn to produce the appropriate feedforward torques for the different joints; (3) network D-T used a developmental progression of its trajectory; (4) network D-G used a developmental progression of its feedback gains; (5) network D-TG used developmental progressions of both its trajectory and its feedback gains; and (6) network RD-TG used developmental progressions of both its trajectory and its feedback gains, though these progressions proceeded from the elbow and wrist joints to the shoulder joint, which is the opposite order from the progressions used by other networks. We found that network D-TG outperformed all other systems. This developmental network, however, yielded performance benefits only on motor tasks that were relatively difficult to learn. We conclude that development can indeed aid motor learning.

Acknowledgments

We are grateful to N. Berthier for interesting discussions regarding motor learning and to the anonymous reviewers for their comments on an earlier version of this article. This work was supported by NIH research grant R01-EY13149.

References

Arutyunyan, G. H., Gurfinkel, V. S., and Mirskii, M. L. (1968). Investigation of aiming at a target. *Biophysics, 13*, 536–538.

Bernstein, N. (1967). *The coordination and regulation of movements*. London: Pergamon.

Berthier, N. E., Clifton, R. K., McCall, D. D., & Robin, D. J. (1999). Proximodistal structure of early reaching in human infants. *Experimental Brain Research, 127*, 259–269.

Corke, P. I. (1996). A robotics toolkit for MATLAB. *IEEE Robotics and Automation Magazine, 3*, 24–32.

Jordan, M. I., & Rosenbaum, D. A. (1989). Action. In M. I. Posner (Ed.), *Foundations of cognitive science*. Cambridge, MA: MIT Press.

Jordan, M. I., & Wolpert, D. M. (2000). Computational motor control. In M. S. Gazzaniga (Ed.), *The new cognitive neurosciences* (2nd ed.). Cambridge, MA: MIT Press.

Kawato, M., Furukawa, K., & Suzuki, R. (1987). A hierarchical neural-network model for control and learning of voluntary movement. *Biological Cybernetics, 57*, 169–185.

LeCun, Y., Bottou, L., Orr, G. B., & Müller, K.-R. (1998). Efficient backprop. In G. Orr & K.-R. Müller (Eds.), *Neural networks: Tricks of the trade*. Berlin: Springer-Verlag.

McDonald, P. V., van Emmerik, R. E., & Newell, K. M. (1989). The effects of practice on limb kinematics in a throwing task. *Journal of Motor Behavior, 21*, 245–264.

Newell, K. M., & Vaillancourt, D. E. (2001). Dimensional change in motor learning. *Human Movement Science, 20*, 695–715.

Rosenbaum, D. A. (1991) *Human motor control*. San Diego: Academic Press.

Saltzman, E. (1979). Levels of sensorimotor representation. *Journal of Mathematical Psychology, 20*, 91–163.

Sanger, T. D. (1994). Neural network learning control of robot manipulators using gradually increasing task difficulty. *IEEE Transactions on Robotics and Automation, 10*, 323–333.

Vereijken, B., van Emmerik, R. E., Whiting, H. T., & Newell, K. M. (1992). Free(z)ing degrees of freedom in skill acquisition. *Journal of Motor Behavior, 24*, 133–142.

Received July 12, 2002; accepted February 10, 2003.