Action*

Michael I. Jordan
Massachusetts Institute of Technology

David A. Rosenbaum
University of Massachusetts, Amherst

March 1988
COINS Technical Report 88-26

---

*To appear in M. I. Posner (Ed.), *Handbook of Cognitive Science*, Cambridge, MA: MIT Press. Preparation of this chaper was supported in part by the Air Force Office of Scientific Research, Bolling AFB, through grant AFOSR-87-0030, National Science Foundation grants BNS-8408634 and BNS-8710933, and a Research Career Development Award K04-NS00942 from the National Institutes of Neurological and Communicative Diseases and Stroke (to DAR). We thank Heather Jane Barnes, Andrew Barto, Emilio Bizzi, Vijaykumar Gullapalli, Robert Jacobs, Larry Parsons, Brian Pinette, Michael Posner, Elliot Saltzman, Michael Seymour, and an anonymous review for helpful comments.

Contents

Abstract

We discuss three fundamental problems for theories of the organization of action—the degrees-of-freedom problem, the serial order problem, and the problem of sensorimotor learning. Our emphasis is on the interrelated nature of these problems. Several recent models and algorithms which address various aspects of these problems are described and evaluated.

Introduction

For cognitions to be communicated, they must be physically enacted. It follows from this observation that a complete account of the cognitive system must explain how it transmits information to the environment as well as how it takes information in, retains, and elaborates it. This fact alone justifies a chapter on action in a handbook of cognitive science, but there are other reasons as well.

One of the effects of the motor system is to move the sensory organs, thereby allowing the organism to acquire information actively. The sensory system in turn allows movements to be carried out efficiently, providing the motor system with the information it needs to correct for and, in some cases, anticipate errors. Noting the intimate connections between perception and action, a number of investigators have suggested that perceptual processing depends on the activation of stored motor routines; this is the so-called *motor theory* of perception (Coren, 1986; Liberman, Cooper, Shankweiler, & Studdert-Kennedy, 1967). Others have proposed that the planning of actions incorporates representations of anticipated perceptual consequences (Adams, 1971; Greenwald, 1970; James, 1890; Ladefoged, DeClerk, Lindau, & Papçun, 1972; Meyer & Gordon, 1983). On either view, perception and action are viewed as highly interrelated. Thus, cognitive science, insofar as it regards perception as one of its core problems, cannot afford to ignore action.

The trend in the study of action, as in perception, is a deepening awareness of the computational complexity of the problems involved. Action is characterized at all levels by a "degrees-of-freedom" problem, a surplus of options, choices, and potential variables. There are choice points at the highest levels of planning pertaining to the method for carrying out a task, and there are choices at lower levels about the patterning of muscular innervation. Though in many cases, some of the choices are clearly preferable to others, the criteria for making choices are often obscure and may depend on subtle situational variables. This, of course, is what gives behavior its enormous flexibility. For the theorist, the problem is to understand how the system discovers successful patterns of action among the thicket of possibilities.

Our focus in this chapter is on some of the ways the computational burden of producing actions may be reduced. We organize the discussion around three topics. The first is the degrees-of-freedom problem *per se* – an orientation to motor control originating with Bernstein (1967) which emphasizes the kinematic and dynamical variables of the biomechanical system. Next, we discuss the *serial order* problem. From one point of view, the ability to perform sequences of actions cuts down on degrees of freedom, both because the same set of system elements are being used repeatedly and because of the possibility for "divide and conquer" strategies. However, sequential performance introduces its own complexities, as will be seen here. Finally, we discuss *learning*. Our point of view on learning is that it too contributes to decreasing the computational load. A system that learns can solve difficult problems in a series of small steps.

The Degrees-of-Freedom Problem

The degrees of freedom of a system are "the least number of independent coordinates required to specify the position of the system elements without violating any geometrical constraints" (Saltzman, 1979). Generally speaking, the larger the number of degrees of freedom of a system, the more difficult it is to make the system behave as desired. However, simply counting degrees of freedom oversimplifies the issue. It is the manner in which degrees of freedom *interact* that determines the difficulty of controlling a system. For example, if the n degrees of freedom of a system are independent of one another, then the controlling system need only possess an algorithm which is adequate for the control of a single degree of freedom; the algorithm can be replicated n times in order to control the ensemble. On the other hand, if the degrees of freedom are not independent (i.e., if the effects of specifications of values for a particular degree of freedom depend on the values of other degrees of freedom), then a team of independent controllers is no longer adequate, and more complex control algorithms must be considered. Complexity can also vary depending on the form of interactions, according to what the controller is able to represent. For example, certain nonlinear interactions pose particular difficulties for certain classes of controller. Thus, to be somewhat more precise, it is the number, form, and magnitude of the interactions between degrees of freedom, in the context of a particular control architecture, that determine the difficulty of controlling a system.

Another aspect of the degrees-of-freedom problem that is often emphasized is the issue of indeterminacy (Bernstein, 1967; Saltzman, 1979). Indeterminacies arise when 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. In such cases, there can be multiple solutions to the control problem (i.e., the problem of finding specifications of values for the system's degrees of freedom so that it performs the task as desired). There is also an increase in the size of the space of incorrect controls and there are problems with singularities. Algorithms designed for determinate problems often either break down or undergo a qualitative change in complexity when extended to a corresponding indeterminate problem. Thus, an important step in the development of theories of motor control is to characterize the indeterminacies that arise. We know that the motor control system can solve indeterminate problems; otherwise, humans would not have the capacity for *motor equivalence*–the ability to achieve the same physical objective in more than one way.

To make these issues more concrete, consider the anatomy of the motor system. The human body has scores of joints, hundreds of muscle groups, and thousands of muscle fibers. At the level of the muscle fibers, the motor control problem is highly indeterminate with respect to most real-world tasks. The large degree of indeterminacy makes the motor control problem intractable at this level, even though the interactions between degrees of freedom are of small magnitude. At the level of the joints, the problem would seem to be more tractable. For example, the number of degrees of freedom of the arm is seven––three for the shoulder, and two each for the elbow and wrist (ignoring the finger joints, which add many more degrees of freedom). Positioning tasks require the specification of only six values of position and orientation. Thus, for such tasks, there is only one excess degree of freedom. Motor control at this level is simplified with respect to the issue of indeterminacy. However, the interactions between degrees of freedom become accordingly much more salient. For example, in the case of an idealized arm, the dynamical equations relating torques to joint angle variables contain interaction terms whose magnitude can be large enough to exert significant effects on behavior (Bejczy & Lee, 1983; Hollerbach & Flash, 1982), and whose coefficients are complex nonlinear functions of the arm's configuration (Kahn, 1969).

It should be kept in mind that the problem of indeterminacy is not solely anatomical. Since actions have temporal and spatial extent, there are many possible trajectories that can be followed in moving between one position/orientation and another. These trajectories arise from different patterns of efferent signals. To specify such signals, values must be specified for each signal at each moment in time. Thus, if this is the level at which we analyze the motor control problem, the number of degrees of freedom the controller must specify depends on the product of the number of anatomical degrees of freedom and the duration of the movement. This value may be greatly in excess of the number of degrees of freedom in the task, unless the task itself specifies an entire trajectory.

Approaching the Degrees-of-Freedom Problem

Let us begin by emphasizing a point which has been implicit in our discussion. The motor control problem can be analyzed at many levels, and it is important to match the level to the specification of the task. This point is particularly important in tasks in which complex movement trajectories are involved, because such tasks may often have more compact descriptions than the time series of effector positions, and these descriptions can lead to simplified control algorithms. To take an example from Raibert (1986), the task of running can be described as a requirement that the net acceleration of the body over a stride cycle be zero, so that forward movement at a fixed speed, with a stable posture, is maintained. Raibert's insight was that this requirement is achieved by a simple algorithm which ensures that the patterns of body and leg movements have particular even and odd symmetries during the stance phase.

Even if a simple control algorithm which matches the particular task requirements is in operation at a given level, the manner in which other levels participate in action must still be specified, and new aspects of the degrees-of-freedom problem may arise in each transformation between levels. One candidate for a theoretical structure in which both high-level and low-level control can be specified is a hierarchy. In a hierarchy, high-level units control lower-level units, which control still lower-level units.[1] Although the low-level units can send feedback to higher-level units, the critical features are that each level exerts descending control only on the units directly beneath it, and that a superordinate unit need not know the details of how a subordinate unit organizes its computation (Greene, 1972; Simon, 1969). The hierarchy helps deal with the problem of indeterminacy by imposing implicit constraints on the kinds of solutions that can be found. Furthermore, if the structure of the hierarchy matches the dynamical structure of the system being controlled, then the hierarchy also helps deal with the problem of interactions between degrees of freedom.

This approach essentially advocates a partitioning of degrees of freedom. Bernstein (1967), a Russian physiologist, proposed the idea that connections, physical or physiological, between muscle groups can serve this function. He proposed that there are *synergies* among muscle groups that help reduce the degrees of freedom to be managed. Researchers in the United States have called these hypothesized elements *coordinative structures.* Discussions of synergies and coordinative structures

[1] Note that the structure need not be fixed anatomically, but may be reconfigurable and sensitive to task requirements.

can be found in Easton (1972), Fowler, Rubin, Remez, & Turvey (1981), Greene (1972), Kugler, Kelso, & Turvey (1980), Saltzman (1979), and Turvey (1977). Some of the evidence for them will be reviewed in the next section.

A technique for reducing the degrees-of-freedom problem which is well ensconced in engineering design is the use of feedback (Melsa & Shultz, 1969). Among their other virtues, feedback controllers can make do with less than perfect compensation for the interactions between the degrees of freedom of a system, as long as the tendency is to move the system in the desired direction, so that errors are corrected over time. Putting a feedback loop around a system defines a new system, which is typically easier to control than the original system. For example, in the case of a linear system, there are design techniques for choosing fixed feedback gains that result in a closed-loop system having any desired impulse response. For human motor control, the delay in the proprioceptive loop is arguably too great for feedback to be useful in the case of rapid, discrete movements (cf. Schmidt, 1982). However, in slower movements, and throughout the execution of extended tasks, there are several levels of control at which feedback may have an important role to play (Albus, 1982; Hollerbach, 1982).

Another way to reduce or partition degrees of freedom is to introduce cost functions which provide criteria for choosing among actions. The idea is that though there may be a huge number of ways to carry out an action, there is often a rational basis for selecting one method over another, usually on the basis of efficiency. For example, though there are many possible ways to lift a glass, some are less time-consuming, energy-consuming, or awkward than others. Evaluating the efficiency of possible glass-lifts provides a way of determining which one should be used, which correspondingly reduces the degrees of freedom for glass-lifting behavior. Theoretical tools for making such choices have been developed in optimal control theory (Kirk, 1970), and the approach has begun to be used in the context of motor control (cf. Hogan, 1984; Nelson, 1983).

There is a further approach to the degrees-of-freedom problem which will be discussed more at length in a later section. The idea is to avoid computations by allowing the inherent dynamical characteristics of the limb to determine the trajectory (cf. Bizzi & Mussa-Ivaldi, this volume). This approach can be used if certain parameters of the limb (e.g., muscle stiffnesses) can be tuned, so that the resulting system has dynamical behavior characterized by basins of attraction around points that correspond to desired movement endpoints (or, more generally, attractor trajectories corresponding to desired movement trajectories).[2]

Synergies and coordinative structures

Let us return to the first approach to reducing degrees of freedom: the introduction (or discovery) of connections between or among potentially independent elements. Saltzman (1979) illustrated the basic idea as follows. Consider two points in the plane. Each point has two degrees of freedom, corresponding to its Cartesian coordinates, and so the system of two points has four degrees of freedom. If the two points are connected by a line of fixed length, L, the number of degrees of freedom

[2] A basin of attraction is a region in state space such that trajectories that start in the region approach a limit set contained in the region over time. For example, a mass-spring system with damping is a system with a single attractor point and a basin of attraction that includes the entire state space (in this example, the state space is two-dimensional: one dimension for position and one for velocity).

is reduced to three, since the line provides an equation of constraint,

$$(x_1 - x_2)^2 + (y_1 - y_2)^2 = L^2,$$

where specification of any three variables constrains the fourth. Saltzman noted that, in general, the number of degrees of freedom of a system is $(Dn - c)$ where D is the number of dimensions, n is the number of elements, and c is the number of equations of constraint. Such equations need not be limited to algebraic relationships; constraints can also be defined among derivatives or integrals of the variables.

How can an equation of constraint be implemented? One method relies on fixed linkages between muscles, where the activation of one muscle results in the activation or inhibition of other muscles. A number of such connections have been noted (see Fowler et al., 1981; Partridge, 1986). However, when muscles are linked in a hard-wired fashion, their interactions are relatively inflexible. "Softer" dependencies may also exist, in the form of synergies or coordinative structures. Consider, for example, an experiment on speech production by Kelso, Tuller, Vatikiotis-Bateson, and Fowler (1984) (see also Abbs & Gracco, 1983). Their subjects repeated the phrase "a baez again" or "a baeb again." Occasionally and unexpectedly, a downward force was applied to the jaw and the instantaneous responses of the upper lip and tongue were recorded. Kelso et al. found that during production of "baeb" the upper lip rapidly descended to the lower lip, completing the necessary bilabial stop. During production of "baez," however, the upper lip did not descend to the lower jaw; instead, electromyographic activity of the tongue was greater than usual, allowing for completion of the fricative. Kelso et al. argued that such data provided evidence for soft linkages between muscle groups because 1) the response was different depending on the phonetic goal, and therefore could not be based on a fixed linkage between muscles, 2) the adjustments could not have been based on simple error-correcting feedback, because the major effect was not on the perturbed articulator, and 3) the speed of the adjustments was too great to be based on the recomputation of a high-level movement plan.

The Mass-Spring and Equilibrium Trajectory Models

Whereas synergies rely on neural links between muscle groups, the physical characteristics of muscles themselves also provide a way of reducing degrees of freedom. In particular, the visco-elastic properties of muscle provide a way of generating entire movement trajectories through a few control settings. By analogy with a simple mass-spring system, in which movement results from the relationship $F = k(x - x_0)$, which holds between the spring tension F, the length x, the equilibrium length x_0, and the stiffness k, it has been proposed that limb movement occurs through the selection of length-tension curves for the muscles. According to one proposal, the thresholds of reflexes are adjusted to select particular length-tension curves (Berkenblitt, Fel'dman, & Fucson, 1986). According to another proposal, the stiffnesses of muscle antagonists are specified, which selects pairs of opposing length-tension curves (Polit & Bizzi, 1979; Sakitt, 1980). The point at which the opposing tensions balance out defines an equilibrium point for the system and thus an endpoint for the limb movement. In either case, the limb is brought to the same final position regardless of the limb's starting position, regardless of transient perturbations, and without explicit error-correcting computations based on feedback. This capability is called *equifinality*, and demonstrates that entire trajectories can be generated even though the controller is limited to the number of degrees of freedom needed to specify a final postural configuration. If detailed control over the shape of the trajectory is required, however, such a scheme may be insufficient (Delatizky, 1982). The general idea can be extended by assuming that the controller specifies a series of intermediate postural equilibrium points during the movement (Bizzi & Mussa-Ivaldi, this volume; Flash & Hogan, 1985; Hinton, 1984; Hogan, 1985).[3] Note, however, that this approach reintroduces one of the problems that the mass-spring model was able to circumvent—the need to specify entire trajectories. Indeed, in the work of Hogan (1985) and Flash and Hogan (1985), this problem is solved in a different manner: The path the hand is to follow is specified through an optimality criterion (that the mean square jerk integrated over the path be minimal for all possible choices of paths). This reference path defines a trajectory of equilibrium points, which the limbs are able to follow due to the modulable elastic properties of the muscles. The remaining problem is to specify how reference paths, such as the minimum jerk path, are actually generated. One idea is to reapply the philosophy of the mass-spring model, but at a higher level, assuming that the *controller* has inherent dynamics which generate entire trajectories given a simple specification of the desired movement (cf. Bullock & Grossberg, 1988). We will pursue this idea further in the next section.

Attractor dynamics

Another approach, which combines aspects of the mass-spring model and the coordinative structure idea, has been proposed by Saltzman and Kelso (1987), in

[3]For evidence that such intermediate points are actually generated during arm movements, see Bizzi, Accornero, Chapple, & Hogan, 1984.

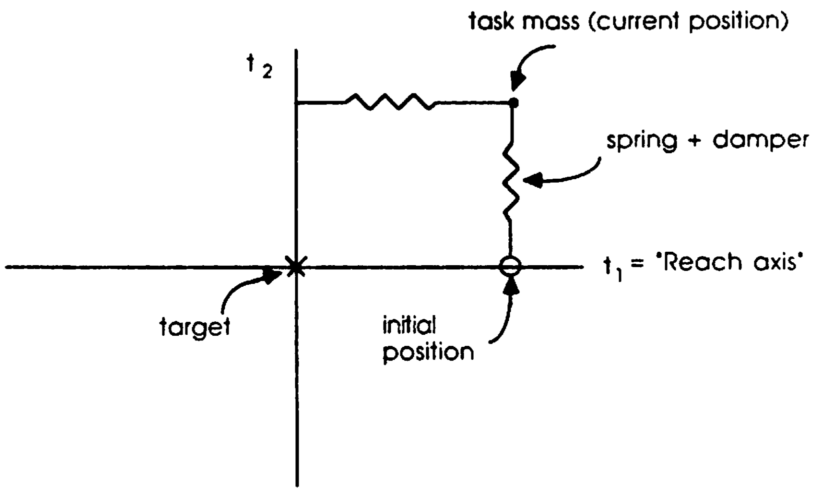

Figure 1: *The task space for a discrete reaching task. Movement of the task mass toward the target results from the effects of an orthogonal pair of damped mass-spring systems.* [Adapted from Saltzman & Kelso, 1987].

the form of their *task dynamics* framework. The basic idea of task dynamics is to formulate movement problems in an abstract, task-based coordinate system in which the equations governing movement are simple (e.g., mass-spring dynamics), uncoupled, and have attractor trajectories. By a series of coordinate transformations, these equations are converted into equations appropriate for the underlying physical system. The control problem, then, is to ensure that the physical system behaves as specified by the latter equations.

Consider, for example, the problem of moving a manipulator endpoint to a target position. As shown in Figure 1, Saltzman and Kelso consider a task space with the origin at the target position and the abscissa connecting the initial position of the endpoint and the target position. They then assume that the task space dynamics is of the form

$$M_T\ddot{\mathbf{t}} + B_T\dot{\mathbf{t}} + K_T\mathbf{t} = 0, \tag{1}$$

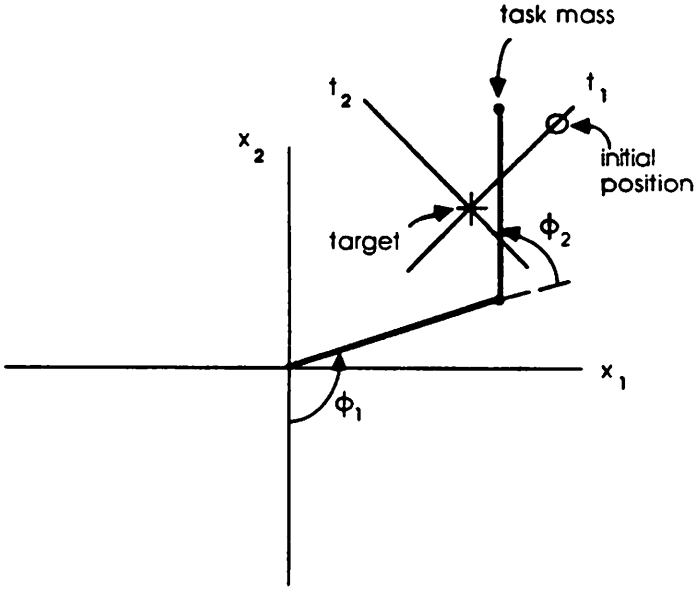

Figure 2: *A joint angle description of the reaching task. The task space of Figure 1 has been rotated and translated within the shoulder-centered* (x_1, x_2) *coordinate system.* [Adapted from Saltzman & Kelso, 1987].

where $\mathbf{t}$ is the vector of task space coordinates, M_T is a mass matrix, B_T is a damping matrix, and K_T is a stiffness matrix. All of these matrices are assumed to be diagonal matrices, so that the equation reduces to a pair of independent damped mass-spring systems. A mass at position $\mathbf{t}$ would move toward the origin as if connected to each axis by a damped spring. The origin is therefore an attractor point in the four-dimensional state space of positions and velocities.

The dynamical system described by Equation 1 can be described in other coordinate systems, in particular, in the system of joint angles of an arm, as shown in Figure 2. Performing the change of coordinates transforms the equation as follows:

$$M_T R J \ddot{\phi} + B_T R J \dot{\phi} + K_T R(\mathbf{x}(\phi) - \mathbf{x_0}) = -M_T R \dot{J} \dot{\phi}, \qquad (2)$$

where ϕ is the vector of joint coordinates, R is a rotation matrix which rotates task-space coordinates to the shoulder-centered frame shown in Figure 2, $\mathbf{x_0}$ is the origin of task space relative to the shoulder, and J is the Jacobian matrix of the manipulator.[4] In this equation, the coefficient matrices are no longer diagonal, and, furthermore, due to the fact that the Jacobian matrix is a function of the configuration ϕ, the equation is time-varying. Thus, the equations are not simple in joint coordinates, although they reflect an underlying simplicity in task space.

Finally, Saltzman and Kelso propose two approaches to controlling an actual arm so that it behaves like Equation 2. One approach is to pair terms in Equation 2 with terms in the dynamical equation for the real arm and thereby define a state-feedback control law for the arm. The second approach is to actually run Equation 2 as an "internal simulation"[5] with coupling terms to the real arm so that the simulation can entrain the arm. In either case, the endpoint of the real arm should move on an approximately straight line toward the origin of task space as if it were a simple mass attached to a spring. Furthermore, if the endpoint of the arm is perturbed from the straight line trajectory en route to the origin, then it should move back to the x-axis as though it were attached to the axis by a second spring.

A major virtue of task dynamics is that many of the details of trajectory planning and responding to perturbation arise directly from the structure of the dynamical equations and need not be dealt with explicitly by higher levels. Higher levels simply instantiate an invariant (e.g., constant coefficient) dynamical organization in the task space. Note that other dynamical forms, such as limit cycles, can also be

[4]The Jacobian matrix is a matrix which relates small changes in endpoint position to small changes in joint angle configuration. That is, it is the matrix whose i,j^{th} entry is the partial derivative $\frac{\partial x_i}{\partial \theta_j}$, where x_i is the i^{th} endpoint position coordinate and θ_j is the j^{th} joint angle.

[5]This is our language; Saltzman and Kelso refer to this approach as a *network coupling* approach.

defined in task space, through choice of an appropriate differential equation. It is also possible to add forces in task space through devices such as artificial potential functions (cf. Khatib, 1987), leading to a variety of sophisticated behavior, such as obstacle avoidance, with minimal supervisory control. Finally, it is important to note that the framework of task dynamics is applicable in the case of excess degrees of freedom (the inverse of the Jacobian matrix in Equation 2 becomes a pseudoinverse). Indeed, Saltzman (1986) has demonstrated in simulations of the speech articulators that a step perturbation imposed on one of the articulators tends to be compensated by changes in the other articulators so that a target position is still achieved.

These virtues are not obtained without cost, and the task dynamic approach requires a large amount of computation at lower levels. Due to the direction in which coordinate transformations are performed, inverses of matrices must be computed, in particular, the inverse of the time-varying Jacobian. When there are excess degrees of freedom, this inverse becomes a pseudoinverse, algorithms for which are computationally expensive for even medium-sized matrices. An alternative to computation, for which the brain seems well suited, is to store pseudoinverse entries in a large memory. However, the number of entries required can be large. For example, with ten degrees of freedom quantized to five levels each, over one billion entries are required to store the pseudoinverse. Such an analysis should not be seen as ruling out task dynamics, as there may well be simplifications which can be made, but is nevertheless a serious consideration.

In task dynamics, the attractor dynamics of the system is ensured by choosing differential equations for task space that are known to have particular attractor basins and limit sets. There are other techniques available for obtaining systems with such behavior. For example, many connectionist networks are nonlinear dynamical systems with attractor dynamics. In the case of networks with symmetric connections, it is possible to prove that the state space for the network is partitioned into regions that are attractor basins with limit points (Cohen & Grossberg, 1983; Golden, 1986; Hopfield, 1982). The positions of the limit points are determined through a learning algorithm. The sequential network to be discussed in the next section (cf. Figure 4) can be demonstrated empirically to have attractors (Jordan, 1986b); if the network learns cyclical trajectories, then those trajectories are limit cycles that partition the state space into attractor basins.

Serial order

Most skilled action involves not just a single movement, but rather a *sequence* of movements. This fact would have no particular relevance to our discussion if interactions across time were functionally equivalent to interactions across space. That is, in a sequence of n actions, each of which is composed of m components, we could assume that the controller is forced to deal with a system having mn interacting degrees of freedom. However, as is well known in linguistics, interactions across time tend to be restricted in a variety of ways. Consider, for example, the relationships between the phonemes of a sentence. Sentences can be segmented into component words, and there are strong interactions between values for the phonological degrees of freedom within a word. Between words, these interactions are of a different and more restricted nature, essentially arising indirectly from higher-order syntactic and semantic interactions. Thus, in speech production, the problem of specifying the phonological structure of a sentence is effectively decoupled into a sequence of independent subproblems, with relationships between the subproblems defining a degrees-of-freedom problem on a higher level.

A difficulty with this classical argument is that it breaks down at the lower levels of skilled behavior. When phenomena nearer the periphery are analyzed, actions do not appear to be performed as separate units, rather, the form that they take can depend strongly on their temporal context. In speech, for example, the articulatory configuration associated with the production of a given phoneme changes substantially depending on the identity of phonemes nearby in the sequence (Kent & Moll, 1975). Moreover, these effects appear across word boundaries. For example, Moll and Daniloff (1972) observed that in phrases such as "*Free Ontario*," the opening of the velum[6] for the nasal /*n*/ began during the /*i*/ in "*Free*." Such phenomena are referred to as *coarticulation*, and are essential in allowing speech to proceed as rapidly as it does. Coarticulation is ubiquitous in speech production and has analogs in other domains as well (cf. Gentner, Grudin, & Conway, 1981; Olsen & Murray, 1976).

In this section, we begin by discussing theoretical approaches to serial order with particular emphasis on the problem of context sensitivity. We discuss parallel activation models, which have had some degree of success with this problem. We then broaden the discussion and turn to other aspects of the serial order problem where symbolic, hierarchical models have had more success.

Context-sensitive allophones

One approach to the problem of context sensitivity is to reintroduce degrees of

[6]The velum is a muscular tissue that opens to allow air to pass between the pharynx and the nasal cavities.

freedom with higher-order units that explicitly encode for context. This is essentially the idea behind the context-sensitive allophone (Wickelgren, 1969)—a phoneme-sized unit that takes into account the left and right context. For example, the context-sensitive allophones for the string "*pin*" are $\{_{\#}p_i,\ _pi_n,\ _in_{\#}\}$, where # is a boundary marker. Serial order can be achieved with associative links in a network of such units. Note that the unordered set of context-sensitive allophones implicitly encodes the serial order. Thus there is no problem of ambiguity about which links to follow in the network, as there is in a network of non-context-sensitive units (cf. Lashley, 1951).

The proliferation of degrees of freedom in this theory solves the context sensitivity problem but introduces other problems. One difficulty is that the approach deals poorly with generalization and rule-governed behavior (Halwes & Jenkins, 1971). For example, the rule in English that voiceless stops become voiced when they follow a sibilant would have to be encoded separately in every relevant context-sensitive allophone. Another problem is that coarticulation effects can depend on non-contiguous phonemes (Benguerel & Cowan, 1974; Öhman, 1966). The number of context-sensitive allophones needed to account for such phenomena would be unreasonably large. These issues are dealt with more satisfactorily in parallel models, to which we now turn.

Parallel, distributed processing models

One approach to accounting for context sensitivity without increasing the number of degrees of freedom is to allow some degree of parallelism (Fowler, 1980). An approach which incorporates such parallelism is embodied in the model of typing proposed by Rumelhart and Norman (1982). As shown in Figure 3, their model is

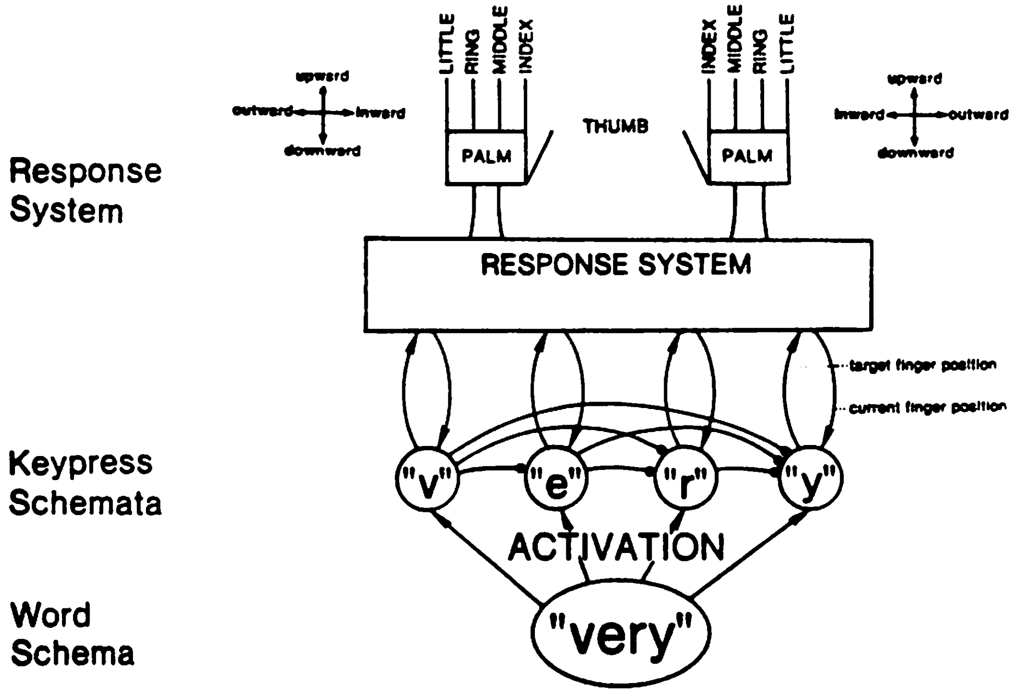

Figure 3: *A network configured to type the word "very"*. [Adapted from Rumelhart & Norman, 1982].

a network, with a processing unit for each action in the sequence to be produced. There are inhibitory links from units early in the sequence to units later in the sequence. The inhibition enforces a graded pattern of activation across the units, with the units appearing early in the sequence having the most activation. Each unit pulls the appropriate finger toward a key on the keyboard, in proportion to its activation, so that the overall movement of the hand is the vector sum of parallel influences from several future actions. When the activation of a unit reaches a threshold, it is turned off, which disinhibits the remaining units. Rumelhart and Norman have demonstrated that this model can account for a variety of phenomena in typing, including the context sensitive nature of the movements toward the keys.

This general approach to introducing parallelism into a sequential process is not entirely without problems. One difficulty is that the model provides no general mechanism for dealing with repeated actions. If a repeated action were represented by separate nodes, one for each token of the repeated type, the the total activation for that action would increase, thereby moving its execution forward in time, whether or not that was desirable. The Rumelhart and Norman model has only one node for each type, so that a sequence such as ABCA must be performed as two separate pieces. Interestingly, the special-purpose mechanism that the model uses for contiguous repeated actions leads to errors similar to the kinds of errors that typists actually make.[7] However, data from speech show that coarticulation can occur across sequences such as ABCA (Benguerel & Cowan, 1974), which is not possible in the model. A related difficulty is that there is no way to restrict unwanted parallel interactions. In speech, for example, it is not necessarily the case that the summation of parallel effects on the articulators leads to a correct acoustic output, due to nonlinearities in the mapping from articulatory configuration to acoustic output (Stevens, 1972).

Jordan (1986a) has proposed another parallel approach to the serial order problem. In the network shown in Figure 4, an action is a pattern of activation across the output units (the output units can be thought of as representing action features). Sequences of actions are learned as the weights in the network are changed. The external input to the network is a constant *plan* vector, which designates the sequence to be performed. A further input to the network is a *state* vector, which varies in time due to the recurrent connections impinging on the state units. The recurrent connections need only produce discriminable state patterns at each time step, so that, with the added discriminability provided by the plan, the remainder of

[7]Typists make errors such as AAB — ABB, which Rumelhart and Norman argue results from the transposition of A and a "doubling operator."

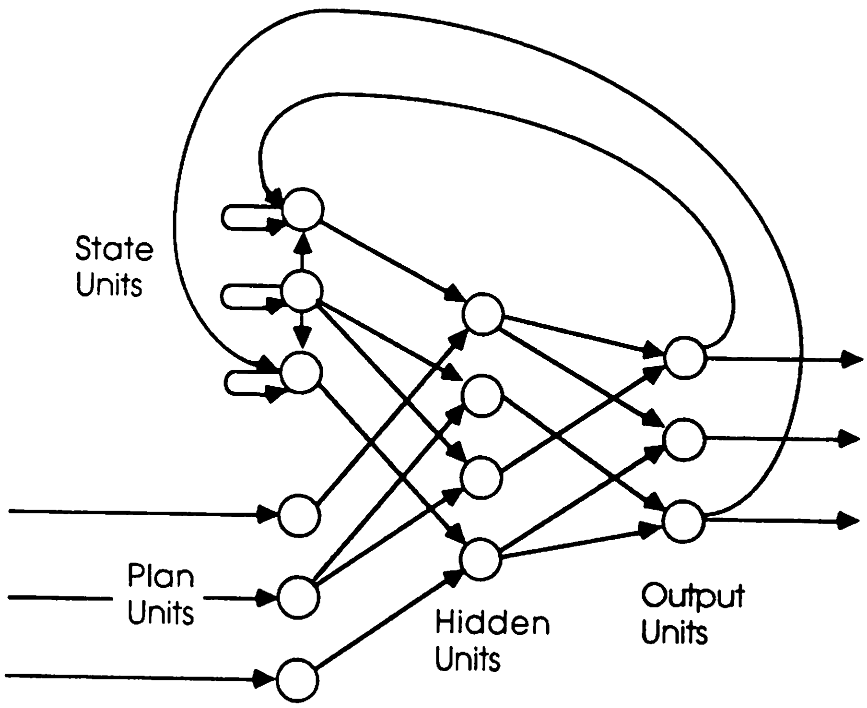

Figure 4: *A connectionist sequential machine. Sequences are learned as the weights in the network are changed. Different sequences are produced in response to different plan vectors.* [From Jordan, 1986a].

the network can learn to map these patterns to target output patterns at each time step. The idea is to decompose the serial order problem into two subproblems how to vary the state vector in time, and how to map states to actions. With this approach, it is possible to construct networks that can learn arbitrary sequences (although particular sequences are more difficult to learn than others).

In the network in Figure 4, there is only a single output vector (with three components), so that all actions are a function of the same set of tunable weights. Therefore, changes made to the weights due to the learning of a particular action have an automatic effect on other actions in the sequence. This means that context sensitivity arises as a form of generalization that depends on: 1) the similarity structure of the states as they change over time, and 2) the way in which target patterns are specified to the network (a point to which we will return in the section on learning). Jordan (1986a) has shown how this approach can be used to model coarticulation in speech production.

The parallel models we have discussed are essentially dynamical approaches to the serial order problem.[8] They are able to deal with some of the criticisms leveled at associationist approaches (cf. Lashley, 1951; Halwes & Jenkins, 1971). They deal particularly well with context sensitivity, and some of the models treat learning. However, these models remain simple non-hierarchical models, treating interactions on only one level. As we discussed earlier, sequential phenomena often seem to have interactions at many levels, which are of different kind and to which different rules apply. These phenomena are suggestive of hierarchies, and furthermore, suggest symbol structures in which entities can stand for other entities. Whether or not the parallel, distributed processing framework is generally inconsistent with a symbolic

[8]See also Grossberg (1980) and Grudin (1981).

view, or whether or not structural phenomena can be accounted for without symbol structures, is currently controversial (cf. Fodor & Pylyshyn, 1987; McClelland & Kawamoto, 1986; Smolensky, 1988). In the current context, layers or tiers of networks may provide a fruitful point of departure. Higher level networks could produce outputs to be treated as input plans for lower level networks, and state vectors at the higher level could depend on the state at lower levels. Furthermore, there need be no explicit flow of control--all levels could be active simultaneously, allowing for contextual effects both between and within levels.

One area which is particularly interesting with regard to these issues is the study of speech errors. On the one hand, certain aspects of speech errors seem generally consistent with a parallel, distributed processing approach—there are similarity effects between units that interact in errors (Fay & Cutler, 1977; Fromkin, 1971; MacKay, 1970), as well as output bias effects, such that when units slip at one level, they tend to form legal strings at the next higher level (Baars, Motley, & MacKay, 1975). Other aspects of speech errors, most notably the fact that words that interact are nearly always from the same syntactic category (Garrett, 1975), are suggestive of a slot-filling process with restrictions on fillers (Reich, 1977; Shattuck-Hufnagel, 1979). To account for these phenomena, Dell (1986) has proposed a hybrid model of speech production. The model has a system of frames with categorized slots which acts as a generative component and an activation network which is responsible for lexical retrieval. Speech errors are proposed to arise at the interface between these two subsystems.

Hierarchies

A number of authors have proposed that serial ordering is achieved with hierarchically organized plans or programs, a view largely supported by error patterns in spontaneous behavior (Lashley, 1951; Gordon & Meyer, 1987; MacKay, 1982; Rosenbaum, 1987). Even if behavioral plans are hierarchically organized, however, the question remains of how such plans actually control production of behavioral sequences. One possibility is that the hierarchy defines the identity and order of commands to the musculature (represented as terminal nodes in the hierarchy) but the commands are issued one after the other without reference to higher levels (Figure 5*a*). Thus, the structure of the plan is hierarchical though its execution is not. Another possibility is that execution as well as structure are hierarchical (Figure 5*b*).

The available evidence favors the latter method. Collard and Povel (1982) and Rosenbaum, Kenny and Derr (1983) found that latencies of successive keystrokes in the speeded production of memorized sequences increased with the number of superordinate nodes to be traversed to get from one terminal node to the next; the number and types of error also conformed to a tree-traversal model. Rosenbaum (1985) also showed that an influential set of results reported by Sternberg, Monsell, Knoll, and Wright (1978) could be accounted for with the tree-traversal model. Sternberg et al. showed that the time to produce a brief burst of highly prepared responses increased with the length of the sequence. This was true both for the latency of the first response and the latencies of subsequent responses (interresponse times). From the perspective of a tree-traversal model, one expects longer sequences to be more deeply nested than shorter sequences, so provided that traversing extra nodes takes extra time, the effect of sequence length on initiation time can be accounted for by assuming that initiation of the sequence awaits activation of the root

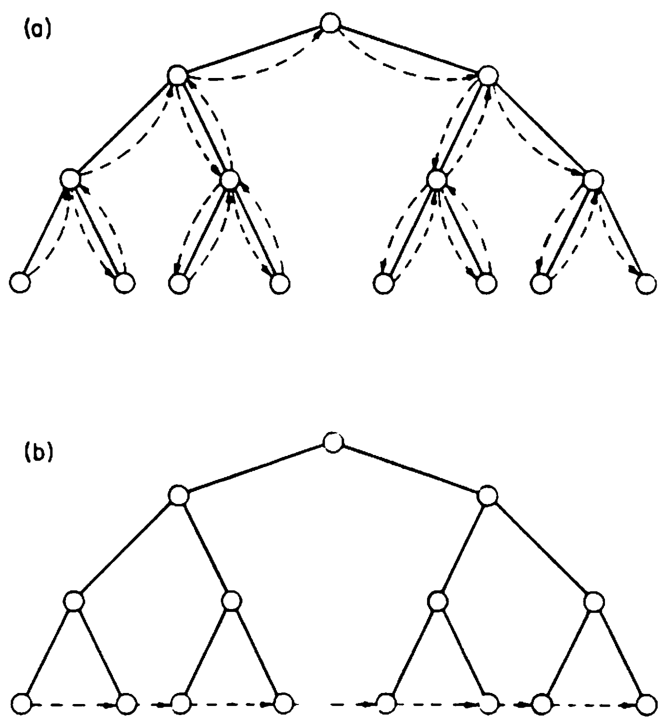

Figure 5: *Two methods for relying on a hierarchy to control the execution of an action sequence. (a) Readout of terminal as well as non-terminal nodes via a tree-traversal process. (b) Linear readout of terminal nodes only.* [Adapted from Rosenbaum, Kenny, & Derr (1983)].

of the tree and the distance from the root of the tree to the leftmost terminal node (corresponding to the first response) increases with sequence length.

The success of the tree-traversal model suggests that execution of rapid response sequences is achieved through hierarchical decoding processes. This conclusion is appealing because similar processes are known to characterize performance in other domains of cognition, most notably verbal recall (e.g. Reitman & Rueter, 1980). As in these other domains, relying on hierarchies minimizes the load on peripheral memory structures, which have limited storage capacities. Another advantage of hierarchical control is that hierarchies are convenient structures for the application of transformational rules, such as repetition, mirroring, and transposition (Greeno & Simon, 1974; Restle, 1970). By applying such rules at various levels, selected spans of behavior can be transformed, allowing for a large number of structures that must be stored in long-term memory.

Parameters and Extended Action Sequences

Just as transformational rules may be applied to all or part of an action program, it may also be possible to apply abstract parameters that define the specific form the program will take. This possibility was noted as long ago as 1932, when Bartlett advocated the view that plans for motor activities are represented schematically. The view has since been endorsed by a number of authors (Arbib, 1981; Pew, 1974; Schmidt, 1975).

Recent experiments have provided evidence about the way abstract parameters may be applied to action programs. The experiments have shown that when a parameter can apply to an entire program, it can be specified in a single processing operation. In one experiment (Rosenbaum, Gordon, Stillings & Feinstein, 1987), subjects were asked to produce one of two possible utterances in response to one of two possible signals. In each condition, the two possible utterances contained 1, 2, or 3 syllables, and differed with respect to a single vowel. In the 3-syllable condition, the choice was /gibifi/ versus /gubudu/, in the 2-syllable condition the choice was /gibi/ versus /gubu/, and in the 1-syllable condition the choice was /gi/ versus /gu/. The reaction signals were a high-pitched tone and a low- pitched tone. In the compatible mapping conditions, the high-pitched tone was mapped to the /i/ sequences and the low-pitched tone was mapped to the /u/ sequences whereas in the incompatible mapping conditions, the mapping was reversed. As seen in Figure 6, it took longer for subjects to start speaking in the incompatible condition than in the compatible condition. There was also an effect of number of syllables, but the interaction between compatibility and number of syllables was not statistically significant. This result supports the hypothesis that the choice of vowel occurred only once. After the choice was made (the time to do so being affected by the compatibility manipulation), it applied to all the syllables. Note that the increase of choice reaction time with number of syllables corroborates the tree-traversal model. The model predicts that the time to initiate a sequence should increase with its complexity. Since the choice of vowel may have occurred before the first descent from the top of the tree; that is, the vowel became part of the information represented in the top node of the tree and so applied *distributively* to the entire sequence. Other experiments have confirmed this interpretation and demonstrated the generality of the distributive assignment method (Inhoff, Rosenbaum, Gordon & Campbell, 1984). Distributive assignment is appealing because of its efficiency: If a number of subprograms take the same parameter value, one would like all the subprograms to receive that value at once.

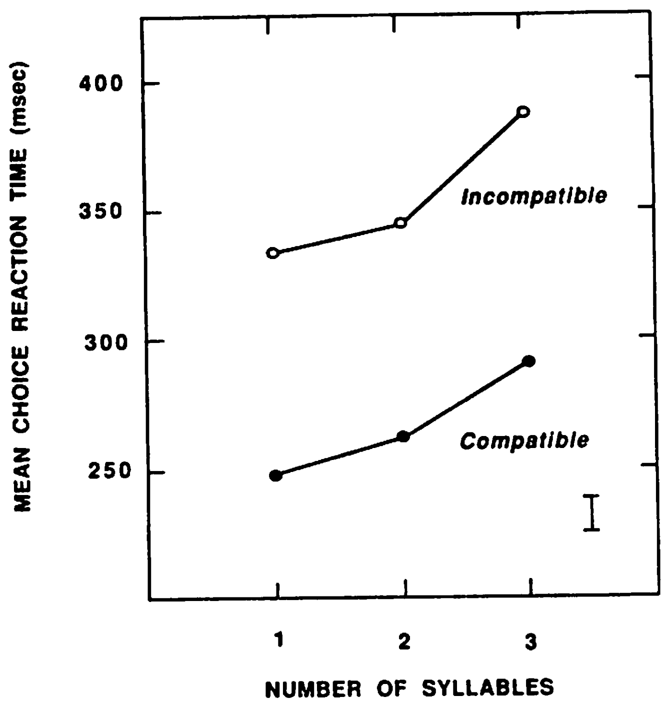

Figure 6: *Results of choice reaction time experiment in which subjects responded to one of two possible signals with utterances of 1, 2, or 3 syllables to which the signals were compatibly or incompatibly mapped. See text for details.* [From Rosenbaum, Gordon, Stillings, & Feinstein (1987)].

For the distributive assignment method to work with particular parameters, those parameters must exist as autonomous functional entities. Such autonomy allows for another kind of efficiency: It allows programs to be *edited* for successive productions. To see why this is desirable, consider the possible fates that could await action programs. One possibility is that programs are simply lost from memory, in which case subsequent actions that require the same program or a similar program would have to be programmed "from scratch." Another possibility is that programs are saved, in which case they would be usable for later actions. Pursuing the latter possibility, if the program needed at a given time were similar but not identical to a recently saved program, it would be desirable to able to edit just those aspects of the saved program that distinguish it from the new one needed. Not only would this minimize the number of changes to be made; it would also allow extended sequences of actions to be represented in memory as a single program with instructions about its needed changes.

If editing were used, programs that differ from recently used programs should take longer to assemble than programs that are identical to recently used programs. This prediction was recently confirmed by Rosenbaum, Weber, Hazelett, and Hindorff, (1986), who showed that when response sequences are performed from memory and aspects of the sequence vary in successive production cycles, the rate of performance cycles, the rate of performance is slower than when the identical sequence is performed over and over again. The latter result is obtained even when subtle aspects of the responses, such as stress levels of individual syllables, vary in successive production cycles. Qualitatively similar results have been reported by Barmack (1970) for eye movements and by Bock (1984) for syntactic organization in sentence production. Relatedly, Miller (1986) has noted that movement perseveration is a major sign of the motor disorder *dyspraxia*, a deficit in the high-level control of movement. Collectively, these results support the hypothesis that programs are normally retained after being executed and are edited to generate similar, subsequent response sequences. It is likely that this is a general property of systems that engage procedural knowledge, for even when people solve series of similarly structured problems, their speed of performance increases (Luchins, 1942).

Learning

The difficulty of the task of executing actions is largely ameliorated by the fact that the system can learn from experience. The system can find solutions to difficult control problems by constructing successive approximations to the solutions over repeated trials, rather than having to produce the solution outright. This cuts down on the complexity of the computations required for any given performance at the expense of more sophisticated processes of memory and integration over trials. Another virtue of learning, typically emphasized in the literature on the subject, is that it allows the system to deal with changing and/or underspecified environments.

The literature on learning is vast and we are accordingly forced to severely restrict our vision in this section. We will discuss some recent computational approaches to the problem of sensorimotor learning, and will focus in particular on the kinds of information that a learning process might reasonably be expected to need, how this information could be acquired, and how it could be used to improve performance. Readers desiring a broader and more historical approach to the subject should consult Adams (1984) and Schmidt (1982).

Consider the case of a child learning to make speech sounds. We can assume that the environment provides targets $\mathbf{y}_i^*$, which are acoustic tokens from the language being learned. These targets could be patterns of energy in frequency bands, for example. The problem for the child is that there is no one to provide the patterns of motor outflow $\mathbf{x}_i^*$ which map into the acoustic targets. These motor patterns must be discovered through experience. Note that the problem reduces to one of inverting a function: If we denote by f the function which maps articulatory to acoustic to auditory patterns, then the problem is to find a solution $\mathbf{x}_i^*$ such that $f(\mathbf{x}_i^*) = \mathbf{y}_i^*$. The difficulty is that we cannot assume that the function f or its inverse (if it exists) are known a priori. This point is more clear in a large variety of other sensorimotor tasks, such as shooting baskets. There the targets $\mathbf{y}_i^*$ can be taken to be the visual patterns of balls passing through hoops, and the problem is to discover motor patterns $\mathbf{x}_i^*$ which achieve these desired visual results. The function f, which includes the physics of basketballs, is clearly not known a priori. Note finally that even if there is a priori knowledge, it must be malleable, as there is simply too much maturational change for a fixed model to be useful.

In control systems terminology, the problem we are discussing involves the adaptation of the *feedforward* component of a system. As shown in Figure 7, a control system may include both a feedforward component, which transforms the reference signal $\mathbf{y}^*$ to a signal $\mathbf{x}$, and a feedback component, which compares the reference signal $\mathbf{y}^*$ to the output $\mathbf{y}$ of the controlled system or *plant*.[9] The signals from the feedforward controller and the feedback controller are summed to form a signal ξ, which is the control input to the plant. The feedforward adaptation problem we are considering can be seen in Figure 7 by ignoring the feedback term, that is, by letting $\xi = \mathbf{x}$.

[9] "Plant" is the traditional term used in control theory for any system being controlled.

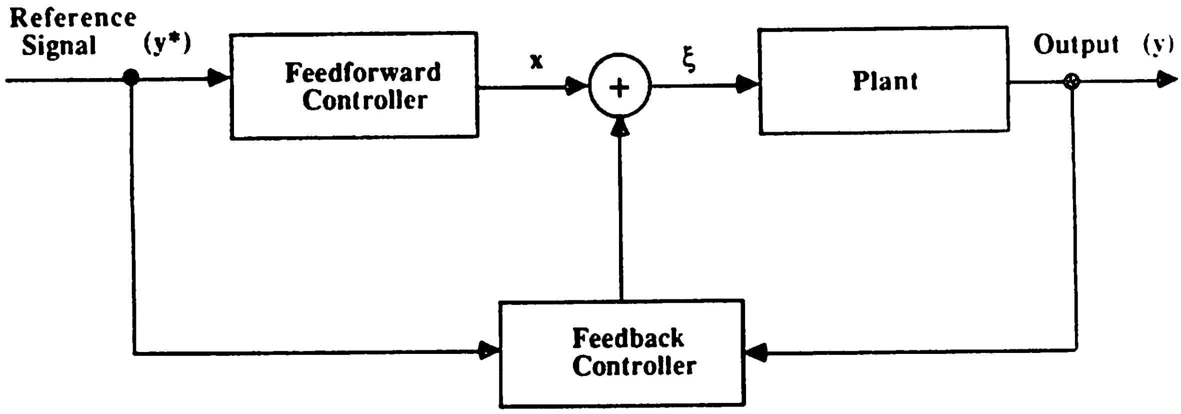

Figure 7: *A control system with feedback and feedforward controllers. The feedforward controller is described by the function g and the plant is described by the function f.*

Note that it is also possible to consider algorithms which adapt the feedback controller. Indeed, the major thrust of adaptive control theory (cf. Gupta, 1986) is to do just that. Our neglect of this theory is based on the fact that, in humans, response to feedback is quite slow relative to the high precision, rapid movements that can be made, and there seems to be more to gain by pursuing feedforward approaches, considering human abilities to plan and predict. This orientation is consistent with current trends in human motor control (cf. Schmidt, 1982; Hollerbach, 1982).

It should be emphasized that all of the various approaches to learning that we will discuss are of recent origin, and it is difficult to make judgments about their relative strengths and weaknesses. In principle, all of the approaches solve the sensorimotor learning problem that we posed, and all make reasonable requirements with respect to a priori knowledge and the amount of computation needed on a learning trial. However, few of the approaches have been tested on a wide range of sufficiently difficult problems. Also, it should be kept in mind that learning is probably not a single entity, and the possibility should not be overlooked that several of the mechanisms we discuss may have some role to play in a complete theory.

Finally, a word on notation. In systems theory, a distinction is often made between systems with or without memory (Vidyasagar, 1973). A system is memoryless if the system function depends only on the instantaneous value of its input. If the function depends as well on past time values of its input, then the system is said to have memory. In this section, a symbol such as $\mathbf{y}^*$ will denote the instantaneous value of a vector variable, when an algorithm is applied to a memoryless problem. For systems with memory, certain of the algorithms we will discuss apply only when the input signal contains not only the current value of some vector, but a finite number of past values of the vector as well. In such a case, the symbol $\mathbf{y}^*$ is taken to represent an mn-dimensional vector constructed by stacking subvectors $\mathbf{v}(1), \mathbf{v}(2), \ldots, \mathbf{v}(n)$, where $\mathbf{v}$ is an m-dimensional vector representing the instantaneous configuration of the underlying system. For other algorithms, past values of a signal $\mathbf{v}$ are not used, rather, the velocity and acceleration vectors $\dot{\mathbf{v}}$ and $\ddot{\mathbf{v}}$ are assumed to be available. In this case, $\mathbf{y}^*$ is formed by stacking $\mathbf{v}$, $\dot{\mathbf{v}}$, and $\ddot{\mathbf{v}}$. This ambiguity in notation allows us to ignore certain differences between algorithms and present the basic ideas in a uniform mathematical framework.

Reinforcement learning

One approach to learning with a long history in psychology is that of *reinforcement learning* (Bower & Hilgard; Bush & Estes, 1959; Mowrer, 1960; Thorndike, 1911). Recently, there has been renewed interest in reinforcement learning algorithms (Barto & Anandan, 1985; Fu, 1970; Sutton, 1984; Widrow, Gupta, & Maitra, 1973; Williams, 1987), and there has been some discussion of the application of such algorithms to control problems (Barto & Epstein, 1983; Mendel & McLaren, 1970; Widrow, Gupta, & Maitra, 1973).

In the reinforcement learning paradigm, there is a set of possible responses, which we will denote as $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$. Associated with the i^{th} response is a probability p_i of selecting that response as an output $\mathbf{x}$ on a given trial. Once a response is selected and transmitted to the environment, a scalar evaluation or reinforcement signal is computed as a function of the response and the state of environment. The reinforcement signal is then used in changing the selection probabilities for future trials: If reinforcement is high, the probability of selecting a response is increased, otherwise, the probability is decreased. Typically, the probabilities associated with the remaining (unselected) responses are also adjusted in some manner, so that the total probability sums to one. An example of such an algorithm is given by the following pair of equations (Fu, 1970), in which the variables are indexed by the time step k. For the selected action $\mathbf{x}_i$, the update rule is:

$$p_i(k+1) = (1-\alpha)p_i(k) + \alpha r(k),$$

where α is a learning rate between zero and one, and $r(k)$ is the reinforcement, also a value between zero and one. For the unselected actions $\mathbf{x}_j$, where $j \neq i$, $j = 1, 2, ..., n$, the rule is:

$$p_j(k+1) = (1-\alpha)p_j(k) + \frac{\alpha}{n-1}(1 - r(k)).$$

Note that if the probabilities sum to one on the k^{th} time step, then they continue to do so on the $k+1^{st}$ time step.

In a feedforward controller, it is necessary to be able to produce different outputs based on the input; thus, the reinforcement learning paradigm needs to be augmented to allow the selection probabilities to depend on an input vector $\mathbf{y}^*$. Mendel and McLaren's (1973) proposal is to partition the input space into a mutually exhaustive set of non-overlapping regions, and maintain a different set of probabilities for each region. Within each region, a reinforcement learning algorithm is used to adjust the probabilities.

Another approach, discussed by Barto and Epstein (1983), is to consider *associative* reinforcement learning algorithms, in which the response probabilities are a parameterized function of the input vector. The reinforcement signal is used to alter the parameters, thereby indirectly changing the response probabilities. Although the algorithms they discuss are restricted to two responses, and probability distribution functions which depend on a linear function of the input, Barto and colleagues have shown elsewhere that *networks* of computational elements using such algorithms are able to produce vector outputs and learn nonlinear mappings (Barto & Anderson, 1985).

Reinforcement learning algorithms are able to learn in situations in which very little instructional information is available from the environment. In particular, such algorithms need make no comparison between the input goal (the vector $\mathbf{y}^*$) and the result obtained (the vector $\mathbf{y}$) in order to find a control signal that achieves the goal. When such a comparison can be made, however, reinforcement learning is still applicable[10] but will be slower than other algorithms (such as the ones that we will discuss in the next two sections) that make use of such information. Although we have suggested that in cases such as speech acquisition, where there is a strong

[10]The reinforcement signal can be defined as a function of the error $\mathbf{y}^* - \mathbf{y}$, with larger reinforcement corresponding to a smaller error vector.

imitative component, a comparison with a goal vector is feasible,[11] the question is empirical and as yet unresolved (cf. Adams, 1971): Does feedback during learning serve 1) to strengthen or weaken the action just emitted, or 2) to provide structural information about how to change the action just emitted into a more suitable action? The discussion in the next two sections may provide some insight into this classical issue, as it describes several attempts to provide algorithms by which the second alternative could be realized.

Inverse modeling

The goal of sensorimotor learning, as we have formulated the problem, is to find control signals so that the actual outcome $\mathbf{y}$ matches the desired outcome $\mathbf{y}^*$. Thus, referring back to Figure 7, the system must configure itself so that the composition of the feedforward controller and the plant is the identity transformation. The feedforward controller should therefore be the inverse of the plant. One approach to learning this inverse is to build an inverse model directly by observing the input-output behavior of the plant. This approach is represented by the recent work of An, Atkeson, & Hollerbach (1985), Grossberg & Kuperstein (1985), Kawato, Furukawa, & Suzuki (1987), Kuperstein (1987), Raibert (1978), and Widrow & Stearns (1985).

Suppose that the feedforward controller is an associative device that depends on a vector of adaptable parameters $\mathbf{w}$. Thus we have $\mathbf{x} = g(\mathbf{w}, \mathbf{y}^*)$, where g represents the input-output behavior of the feedforward controller. On successive learning trials, the system chooses a random vector $\mathbf{x}'$ as input to the plant, observes the output $\mathbf{y}$, assigns $\mathbf{y}^* = \mathbf{y}$, and then associates $\mathbf{y}^*$ to $\mathbf{x}'$ in the controller by changing the parameters $\mathbf{w}$. Depending on the form of the function g, there are

[11] Note that we are *not* suggesting that the control solution $\mathbf{x}^*$ is available for comparison.

many methods available for changing the parameters. One popular method, which is commonly used in associative memories, is gradient descent on an error surface which is a function of the difference between the controller's actual output $\mathbf{x}$ and the given test vector $\mathbf{x}'$. Consider the error measure E defined by:

$$E = \frac{1}{2}(\mathbf{x}' - \mathbf{x})^T(\mathbf{x}' - \mathbf{x}),$$

where the raised T denotes the transpose operation.[12] This function is a quadratic surface around the test vector $\mathbf{x}'$, and goes to zero when $\mathbf{x} = \mathbf{x}'$, that is, when the controller is producing the desired output. The gradient descent[13] learning rule is then obtained by using the chain rule:

$$\begin{aligned} \Delta\mathbf{w} &= -\alpha\nabla_{\mathbf{w}}E \\ &= -\alpha\nabla_{\mathbf{w}}\frac{1}{2}(\mathbf{x}' - \mathbf{x})^T(\mathbf{x}' - \mathbf{x}), \\ &= \alpha\frac{\partial\mathbf{x}}{\partial\mathbf{w}}^T(\mathbf{x}' - \mathbf{x}) \end{aligned} \tag{3}$$

where $\frac{\partial\mathbf{x}}{\partial\mathbf{w}}^T$ is the transpose of a matrix of partial derivatives, and α is a learning rate. Note that the matrix $\frac{\partial\mathbf{x}}{\partial\mathbf{w}}^T$ depends only on the structure of the controller input-output function g, and it is therefore available to the system. As the error

[12]The vector notation used here should not obscure the fact that this error measure is simply the sum of squares: $E = \frac{1}{2}\sum_{i=1}^{n}(x_i' - x_i)^2$, where the subscript i picks out the i^{th} component of the vector.

[13]The gradient, denoted by $\nabla_{\mathbf{w}}E$, is simply the vector of partial derivatives of E with respect to the components of $\mathbf{w}$. It points in the direction of steepest ascent of E. The negation sign in the learning rule changes the search to gradient *descent*.

measure goes to zero, the controller is able to take an input $\mathbf{y}^*$ into an output $\mathbf{x}$ which, when applied to the plant, will produce $\mathbf{y} = \mathbf{y}^*$. For the full inverse mapping to be approximated, experience with a variety of vectors is necessary. The process can be aided if g has continuity properties to allow generalization or interpolation.

The associative approach to building an inverse model has been proposed by Kuperstein (1987), and Widrow & Stearns (1985), both of whom use the least mean squares algorithm of Widrow & Hoff (1960), which is a gradient descent learning rule. Kuperstein uses this approach to model the learning of the mapping from retinal coordinates to hand positions. After a phase during which the mapping is learned, it is possible to command the hand to move to the position where the eyes are focused.

There are other approaches to learning an inverse model. An, Atkeson, & Hollerbach (1985) have proposed a non-associative approach in which knowledge of the structure of the differential equations of the plant (a robot arm, in their case) is used. The problem then reduces to estimating the inertial terms in these equations, and An, et al. have shown that a linear estimation scheme that can do this in a small number of trials. This approach has the advantage that if these terms are estimated correctly, even on a restricted set of test trajectories, then the inverse model will be correct for all other trajectories. There is no need to randomly explore the space of trajectories. A disadvantage is that the structure of the differential equations may not always be known, particularly when the plant includes a dynamical system external to the body.

Finally, it is also possible to simply store the vectors $\mathbf{x}'$ in a table indexed by $\mathbf{y}^*$, so as to limit the amount of computation involved in using the feedforward controller. A problem with such tables is that they can be very large, given reasonable assumptions about the quantization and dimensionality of the indexing variables. One way to lessen this problem is to use hash coding (Albus, 1982). Another approach is to combine the use of tables with knowledge of the structure of the differential equations of the plant. In work with a robot arm, Raibert (1978) proposed such a hybrid approach in which certain of the variables (the positions and velocities) were used as indices into a table of parameters, and other variables (the accelerations) were combined with the parameters in the resulting linear computation of the torques.

One drawback with the various inverse model approaches as we have presented them so far is that there is no mechanism for goal-directed learning. That is, there is no way to improve on a particular target trajectory $\mathbf{y}^*$. On a given trial, if an incorrect output $\mathbf{y}$ is produced, then that output is treated as an input to the controller and associated to the corresponding test vector $\mathbf{x}'$ that produced it. This will be useful on some future trial if that $\mathbf{y}$ is the target, but it is not necessarily of help with respect to the current target $\mathbf{y}^*$.

One approach to goal-directed learning is to have the error correction inherent in the *feedback* controller aid the learning process. This is the approach taken by Kawato, et al. Their feedforward controller is an associative memory and learning proceeds by gradient descent, as above. However, rather than associating the output from the plant to a randomly chosen input to the plant, they associate the target trajectory $\mathbf{y}^*$ to an input to the plant which is the sum of the feedforward contribution and the feedback contribution (ξ in Figure 7). To see how this approach works, consider first the initial situation where the feedforward signal is zero. On the first trial, the input to the plant is given by the feedback signal alone.[14] In general, this feedback signal will be sufficient for the output of the plant to come close to the goal

[14] In the simplest case, this would be given by $K(\mathbf{y}^* - \mathbf{y})$, where K is a gain matrix.

trajectory, but it will not constitute a solution $\mathbf{x}^*$. However, this input ξ can still be stored away in the associative memory and thereby used as the feedforward contribution on the next trial. The feedback will then bring the resulting output even closer to the target. Thus the process will move successively closer to a solution.

Another approach to goal-directed learning is to make do with an incorrect model and instead change particular feedforward commands corresponding to particular desired output trajectories. Atkeson and McIntyre (1986) propose updating the particular feedforward command after each trial by incorporating the feedback command (as in Kawato, et al.) and a further term which is an estimate of the difference between the current command and the correct command $\mathbf{x}^*$. This estimate is made by applying the (inaccurate) inverse model to the current output $\mathbf{y}$ and the desired output $\mathbf{y}^*$. If the model is not too inaccurate, then this further term should speed learning.

Forward modeling

The final approach to sensorimotor learning that we will discuss involves using a *forward* model of the plant to construct the feedforward controller (Rumelhart & Jordan, 1988). The basic idea is as follows. As in the previous section, let us assume that the feedforward controller has the form $\mathbf{x} = g(\mathbf{w}, \mathbf{y}^*)$, where $\mathbf{w}$ is a vector of adjustable parameters. Let us now consider a quadratic error surface based not on the output of the controller, but rather on the output of the plant. That is, let

$$E = \frac{1}{2}(\mathbf{y}^* - \mathbf{y})^T(\mathbf{y}^* - \mathbf{y}),$$

which will be minimal when $\mathbf{y} = \mathbf{y}^*$, that is, when the plant is producing the desired response. Given that $\mathbf{y}$ is a function of $\mathbf{w}$, that is, $\mathbf{y} = f(g(\mathbf{w}, \mathbf{y}^*))$ we can base a learning rule on the gradient of this error surface by using the chain rule as follows:

$$
\begin{aligned}
\Delta \mathbf{w} &= -\alpha \nabla_{\mathbf{w}} E \\
&= -\alpha \nabla_{\mathbf{w}} \frac{1}{2}(\mathbf{y}^* - \mathbf{y})^T(\mathbf{y}^* - \mathbf{y}) \\
&= \alpha \frac{\partial \mathbf{y}}{\partial \mathbf{w}}^T (\mathbf{y}^* - \mathbf{y}) \\
&= \alpha \left(\frac{\partial \mathbf{y}}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial \mathbf{w}}\right)^T (\mathbf{y}^* - \mathbf{y}) \\
&= \alpha \frac{\partial \mathbf{x}}{\partial \mathbf{w}}^T \frac{\partial \mathbf{y}}{\partial \mathbf{x}}^T (\mathbf{y}^* - \mathbf{y}) \qquad (4)
\end{aligned}
$$

As before, the first matrix depends only on the structure of the function g, and is therefore available to the learner, as is the vector $\mathbf{y}^* - \mathbf{y}$, which is obtained by comparing the desired output of the plant with the actual output. On the other hand, the middle matrix $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$), is not available a priori. It is a matrix which relates small changes in the components of $\mathbf{x}$ to small changes in the components of $\mathbf{y}$, and depends on the nature of the plant. Note, however, that these are all *forward* derivatives (as compared to the matrix $\frac{\partial \mathbf{x}}{\partial \mathbf{y}}$), and are readily obtained by differentiation if a forward model of the plant can be found.[15]

The proposal, then, is to first learn a forward model of the plant. The model is of the form $h(\mathbf{v}, \mathbf{x})$, where $\mathbf{v}$ is a vector of adjustable parameters, and gradient descent is used so that the model comes to approximate the input-output behavior of the plant. In Rumelhart and Jordan, the model is implemented as a multilayer connectionist network, and gradient descent is achieved by using the backpropaga-

[15]The matrix $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$ is referred to as the Jacobian matrix of the plant function f. (Compare with footnote 2, in which f is the forward kinematic function of a manipulator).

tion rule of Rumelhart, Hinton, and Williams (1986).[16] After the forward model is learned, it is possible to learn the controller. Again, backpropagation is used, with the gradient descent now being performed with respect to the underlying parameters $\mathbf{w}$. Given a target $\mathbf{y}^*$, the controller produces a vector $\mathbf{x}$, which leads to an output $\mathbf{y}$. The error vector $\mathbf{y}^* - \mathbf{y}$ is then propagated back through the (now fixed) weights of the model, which multiplies by the Jacobian matrix of the plant, computing the vector $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}^T(\mathbf{y}^* - \mathbf{y})$, and, furthermore, this vector is then propagated down into the controller so as to multiply by the remaining matrix $\frac{\partial \mathbf{x}}{\partial \mathbf{w}}^T$. Note that this procedure, as given by Equation 4, is analogous to the learning rule in Equation 3, with the error $\mathbf{x}' - \mathbf{x}$ replaced by the term $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}^T(\mathbf{y}^* - \mathbf{y})$.

To summarize, once the forward model of the plant has been learned, all of the terms in Equation 4 are available to the system and can be computed in a single pass using the backpropagation algorithm. The system essentially uses the model to find out how to change the weights of the controller so as to perform gradient descent in the "distal" error measure E. Rumelhart and Jordan have demonstrated this algorithm in the inverse kinematic problem of positioning a six degree-of-freedom manipulator in a planar world.

The forward model approach is the most complex approach we have considered, requiring a model of the plant which must be learned separately from the feedforward controller, as well as a differentiation procedure (e.g., backpropagation) so that the forward model can be used in the learning of control signals. There are, however, advantages to offset the complexity. First, once the forward model is learned, the

[16] Backpropagation is an algorithm for computing partial derivatives in a layered connectionist network with nonlinear units.

algorithm is inherently goal-directed, given that it works directly to minimize the error between the goal $\mathbf{y}^*$ and the actual output $\mathbf{y}$. Second, it may be useful for other reasons to have a forward model of the plant. Such a model allows the system to predict the result it expects, given the output it has produced. This ability allows the system to notice its own errors, as well as to learn through "mental practice" (cf. MacKay, 1982; Singer, 1980). Third, the forward model approach deals nicely with excess degrees of freedom, as we will discuss in a later section. Finally, it should be pointed out that the forward model need not be exact for learning to occur. Jordan (1988) has shown that even given a large amount of noise in the forward model, perfect learning of the controller can still be obtained. This occurs if the model tends to lead the system downhill in error, even if not directly down the gradient.

Discussion

The currently dominant psychological theory of motor learning is schema theory (Schmidt, 1975, 1982). It is interesting to consider this theory in light of the work that we have discussed above. According to schema theory, four kinds of information are available after a movement—the movement outcome, the response specification (control signal), the initial conditions, and a proprioceptive signal. Learning a motor program involves changing a "recall schema," in such a way that the movement outcome and initial conditions can be used on future trials to retrieve the corresponding response specifications. The schema concept is invoked to deal with two issues that were problematic for earlier theories—how the system can produce novel control signals for novel situations, and how the system avoids a massive storage problem. These problems are lessened in schema theory, because the schema is an abstraction of the observed data points, and it is able to interpolate as needed.

These properties of a schema are embodied in the feedforward controllers that we have discussed. In the associative approach, for example, learning proceeds by changing the parameters of a continuous function g. The function g is an abstraction of the data points[17], and the continuity provides interpolation. In fact, if we let $\mathbf{y}$ represent the movement outcome and $\mathbf{x}'$ be the corresponding response specification, then the learning of a recall schema is essentially the same as the inverse model approach. Indeed, like the unadorned version of that approach, schema theory provides no account of goal-directed learning. The recall schema simply learns the relationship between actual outcomes and response specifications and no mechanism is provided by which the schema can be changed so as to provide a more correct control signal with respect to the *currently* desired movement outcome. This problem was addressed to some degree in the theory which preceded schema theory—Adam's (1971) closed loop theory. In that theory, the movement outcome ("knowledge of results") was treated as the input to a problem-solving process that led to a varied response on the next trial. However, this process was never made explicit in the theory. The issue has perhaps been overlooked because both of these theories were designed to address data from discrete, single degree-of-freedom movements. In such movements, the problem of converting from knowledge of results to a change in the response specification is not particularly difficult. However, in tasks with many degrees of freedom, this conversion is arguably the most difficult part of the learning process.

Schema theory also proposes a "recognition schema," which learns associations from movement outcomes and initial conditions to proprioceptive feedback. This

[17]The gradient descent rules we have discussed are an iterative approach to nonlinear least-squares regression.

schema is used for error recognition after fast movements and provides a reference signal for a servomechanism during slow positioning tasks. The recognition schema and an "error labeling" schema, which maps to predicted movement outcomes, act somewhat like a forward model. They are able to serve as a substitute for external information about the movement outcome, and thereby provide an input for the learning of the recall schema. However, these schemata are not used directly in making changes to the recall schema, as in the forward model approach that we discussed above.

In terms of relating the approaches that we have discussed to experimental data, there is clearly much to be gained from making comparisons with schema theory. One example of such data, which has been taken as strong support for schema theory, is the relationship between the amount of learning in a task and the amount of variability during practice. In some tasks, it is more beneficial to experience repeated trials in the neighborhood of a target than at the target itself (Shapiro & Schmidt, 1981). This is clearly a result that would be predicted by the forward model approach, because variable experience yields a forward model which is correct over a larger domain, which in turn provides better derivative information for learning the controller. The estimation procedures of An, et al. would also benefit from varied experience. It is less clear how the other inverse model approaches would fare with this datum; for example, Kawato et al.'s model would seem to learn most from experience with the target. Finally, reinforcement learning would not appear to predict the result, because the overall probability of reward for the correct output would be lower in the variable practice condition.

Toward an integration

In preceding sections, we have organized our discussion of computational issues in the study of action around the three topic areas of degrees of freedom, serial order, and learning. For the most part, our discussion has treated these problems separately, reflecting the separate histories of research in these areas. In this section, we illustrate why an integrated treatment of these problems would be desirable, by sketching some recent work that takes a limited step in this direction (Jordan, 1988).

Let us reconsider the problem of context sensitivity. The notion that the same action can have different forms depending on the context implies that there are excess degrees of freedom somewhere in the system. We can model the situation by considering a *target space*, in which actions appear as points, and an *articulatory space*, which maps into target space through a many-to-one function f. (Target space corresponds to vectors $\mathbf{y}^*$ in the previous section, and articulatory space to vectors $\mathbf{x}$). In articulatory space, actions are the inverse images of points in target space, and are regions rather than points, due to the assumption that f is many-to-one. We also assume that a task is specified to the system as a sequence of points $\mathbf{y}_1^*, \mathbf{y}_2^*, ..., \mathbf{y}_n^*$ to be followed in target space. As shown in Figure 8, the learning problem facing the system is to find a trajectory in articulatory space passing through the regions in the specified order (where the regions are defined as inverse images of the target points). Clearly, this requirement is fulfilled by many possible trajectories. However, if we add the constraint that values of articulatory degrees of freedom should change little over time, then certain trajectories are to be preferred to others. This constraint represents an interaction between the serial nature of the task and the existence of excess degrees of freedom—the fact that an action appears in a temporal context of other actions imposes implicit constraints

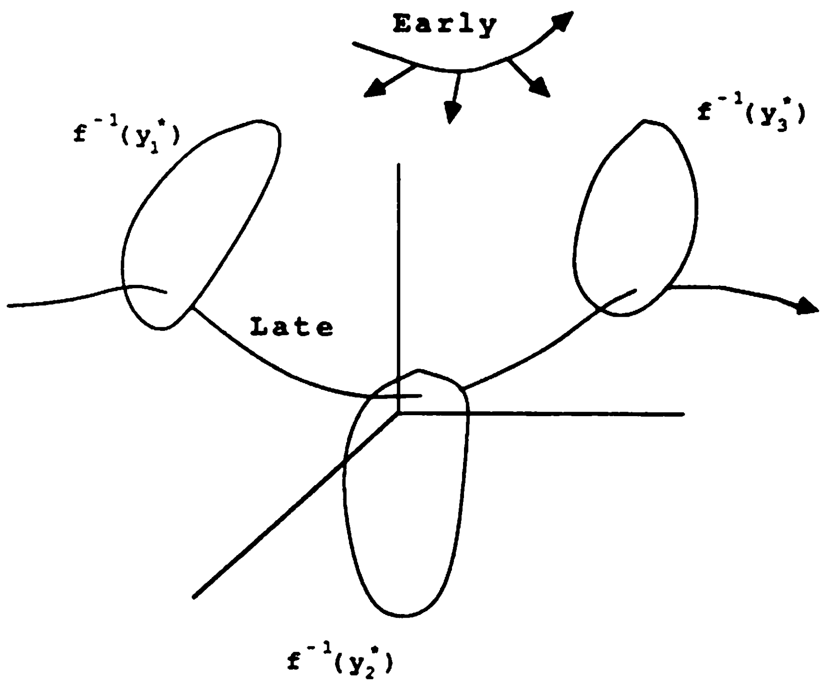

Figure 8: *A representation of articulatory space. The three regions are the inverse images of points in target space (which is not shown). Also shown are a trajectory early in learning, with the transformed error vectors computed during the steps of a learning trial, and a trajectory late in learning.* [From Jordan, 1988].

on possible configurations of degrees of freedom to realize the action.

The problem we have described can be formulated as a problem in optimal control theory, but this approach is complicated by the nature of the constraints on the trajectories (they appear as inverse images of an unknown function f). It is accordingly difficult to see how the system could generate preferred trajectories on the fly, particularly given the need to anticipate future actions. However, if we include a role for learning, then the problem becomes somewhat easier. Referring to Figure 8, let us suppose that the short trajectory represents an attempt made by the system on an early learning trial. At each time step, we assume that learning proceeds by: 1) a comparison in target space between the target action and the actual action, thereby generating an error vector, 2) a transformation of the error vector into articulatory space, and 3) an updating of the trajectory based on the transformed error vector. The overall effect of these steps is to pull the trajectory toward the region corresponding to the currently specified target point. For the system to move toward preferred trajectories, it is possible to bias the adjustments made at a particular point on the trajectory by the adjustments made at nearby points. Metaphorically, we can imagine attaching springs along the trajectory to bias the learning process toward solutions in which the articulatory variables change little over time.

The computations we have described arise naturally in the parallel, distributed processing network shown in Figure 9. This network consists of the sequential network shown previously in Figure 4 composed with an additional layered network. This additional network connects the output units (which encode articulatory space), to a set of units which encode target space and serves as a forward model of the function f. The model is learned during an exploration phase during which

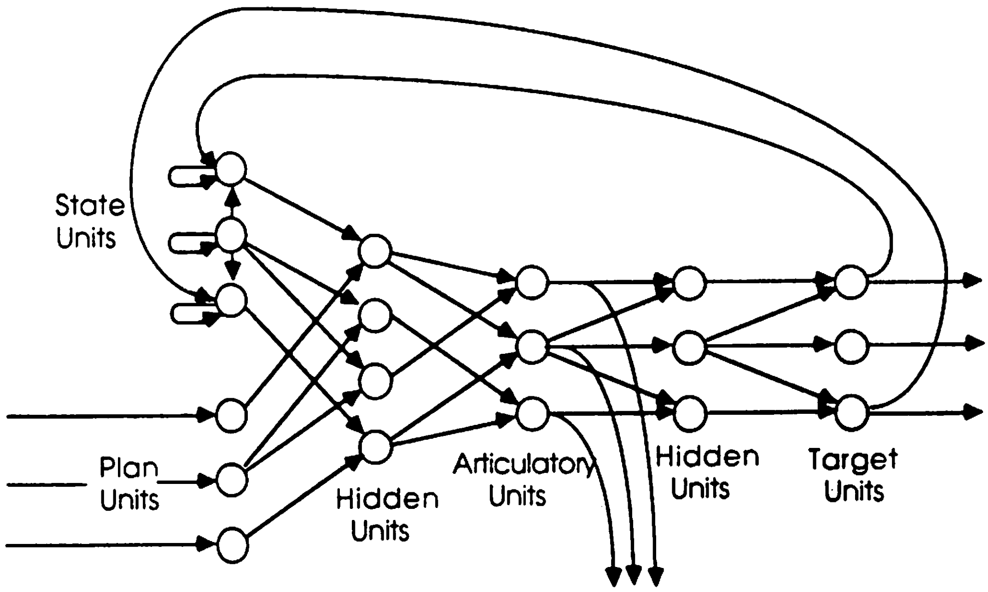

Figure 9: *The connectionist sequential machine of Figure 4 augmented with a layered network which serves as a forward model. The articulatory units are the output units of the network. There can also be feedback from the articulatory units to the state units.* [From Jordan, 1988].

random vectors are placed on the articulatory units and the observed result (e.g., auditory pattern) is used as a target for learning the weights between the articulatory units and the target units. Once these weights are learned, the learning of sequences can proceed. To learn to imitate a target sequence, errors are generated at the target units between actual and desired outputs and are propagated back through the fixed connections of the model and down into the sequential network, where the weights are changed. This process implements the three steps described in the previous paragraph and occurs for each action in the sequence. Furthermore, the fact that adjustments for each action are made to the same set of underlying weights means that adjustments automatically tend to bias one another, the form of the bias depending on the similarities between the patterns on the input units of the sequential network at nearby points in time.[18] Thus, the sequential network is biased to find articulatory representations that are sensitive to their temporal context.

The results of a simulation of such a network are shown in Figure 10. The network had six output units for the six articulatory degrees of freedom of the "hand" being controlled, and four target units for the four Cartesian coordinates of the endpoints. The task for the system was to learn to touch the indicated positions in a specified order, using either finger, with the choice of finger based on a simple distance comparison. As shown in the figure, the network found a solution which both met the endpoint constraints and made use of the excess degrees of freedom to anticipate future actions.

While preliminary, these results suggest that it may be useful to consider inte-

[18] Jordan (1988) also discusses a method which generates additional errors directly at the articulatory units based on the time-rate of change of their activations.

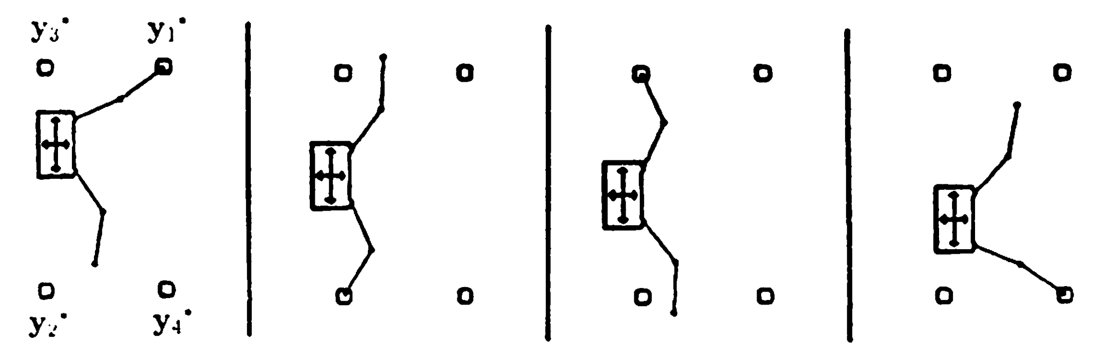

Figure 10: *A six degree-of-freedom (two translational and four joint angles) system which has learned a sequential positioning problem, moving an end effector through the sequence* y_1^*, y_2^*, y_3^*, y_4^*. [From Jordan, 1988].

grated treatments of degrees-of-freedom problems. In particular, learning appears to be a powerful way to introduce constraints which reduce degrees of freedom. Finally, it is worth pointing out that excess degrees of freedom are not always best thought of as a problem in the pejorative sense of the term. Rather, excess degrees of freedom can allow flexibility and responsiveness to contextual variables. Computations which reduce degrees of freedom may only make sense when we consider the larger context in which a problem is posed (cf. Fowler & Turvey, 1978).

Conclusions

The study of action has a long history, yet a computational approach to the area is still in its infancy. There have been promising developments, and current research in the area is an interesting confluence of ideas from disciplines such as psychology, linguistics, control theory, robotics, and physics. In this chapter, we have chosen to emphasize the unified nature of the study of action, by treating various of the problems that arise as instances of a general degrees-of-freedom problem. In taking this approach, we have been influenced by several authors, including Albus (1982), Greene (1972), Norman & Shallice (1986), Saltzman (1979), and Turvey (1977). Our perspective has been that the major problem for an action system is to control relationships between interacting systems of degrees of freedom. These degrees of freedom may appear at a variety of levels of analysis and may refer to both spatial and temporal aspects of behavior. We feel that this perspective provides a useful framework for discussion and will allow meaningful comparisons to be made as computational theories continue to be developed.

References

Abbs, J. H. & Gracco, V. L. (1983). Sensorimotor actions in the control of multimovement speech gestures. *Trends in Neuroscience*, *6*, 393-395.

Adams, J. A. (1971). A closed-loop theory of motor learning. *Journal of Motor Behavior*, *3*, 111-149.

Adams, J. A. (1984). Learning of movement sequences. *Psychological Bulletin*, *96*, 3-28.

Albus, J. S. (1981). *Brains, behavior, and robotics.* Peterborough, NH: BYTE Books.

An, C. H., Atkeson, C. G., & Hollerbach, J. M. (1985). Estimation of inertial prarmeters of rigid body links of manipulators. *Proceedings of the 24th Conference on Decision and Control*, Fort Lauderdale, FL.

Arbib, M. A. (1981). Perceptual structures and distributed motor control. In V. B. Brooks (Ed.), *Handbook of physiology -- The nervous system II.* Bethesda, MD: American Physiological Society.

Atkeson, C. G. & McIntyre, J. (1986). Robot trajectory learning through practice. *IEEE Conference on Robotics and Automation.*

Attneave, F. & Benson, B. (1969). Spatial coding of tactual stimulation. *Journal of Experimental Psychology* , *81*, 216-222.

Baars, B. J., Motley, M. T., & MacKay, D. G. (1975). Output editing for lexical status from artificially elicited slips of the tongue. *Journal of Verbal Learning and Verbal Behavior*, *14*, 382-391.

Barmack, D. (1970). Dynamic visual acuity as an index of eye movement control. *Vision Research, 10*, 1377-1391.

Bartlett, F. C. (1932). *Remembering.* London: Cambridge University Press.

Barto, A. G. & Anandan, P. (1985). Pattern-recognizing stochastic learning automata. *IEEE Transactions on Systems, Man, and Cybernetics, 15*, 360-375.

Barto, A. G. & Anderson, C. W. (1985). Structural learning in connectionist systems. *Proceedings of the Seventh Annual Conference of the Cognitive Science Society.* Hillsdale, NJ: Erlbaum.

Barto, A. G. & Epstein, S. (1983). Adaptive networks and sensorimotor control. *Proceedings of the Second Workshop on Visuomotor Coordination in Frog and Toad: Models and Experiments*, 105-158.

Bejczy, A. K. & Lee, S. (1983). Robot arm dynamic model reduction for control. *IEEE Conference on Decision and Control*, San Antonio, TX.

Benguerel, A. P. & Cowan, H. A. (1974). Coarticulation of upper lip protusion in French. *Phonetica, 30*, 41-55.

Berkenblit, M. B., Fel'dman, A. G. & Fucson, O. I. (1986). Adaptability of innate motor patterns and motor control. *Behavioral and Brain Sciences, 9*, 585-638.

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

Bertelson, P. (1965). Serial choice reaction-time as a function of response versus signal-and-response repetition. *Nature, 205*, 217-218.

Bizzi, E., Accornero, N., Chapple, W., & Hogan, N. (1984). Posture control and trajectory formation during arm movement. *Journal of Neuroscience, 4*, 2738-2745.

Bock, J. K. (1982). Theory of serial pattern production: Tree traversals. *Psychological Review, 89*, 1-47.

Bower, G. H. & Hilgard, E. R. (1966). *Theories of learning.* New York: Appleton-Century Crofts.

Bullock, D. & Grossberg, S. (1988). Neural dynamics of planned arm movements: Emergent invariants and speed-accuracy properties during trajectory formation. *Psychological Review*, in press.

Bush, R. R. & Estes, W. K. (Eds.), *Studies in mathematical learning theory.* Stanford, CA: Stanford University Press.

Cohen, M. A. & Grossberg, S. (1983). Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. *IEEE Transactions on Systems, Man, and Cybernetics, 13*, 815-825.

Collard, R., & Povel, D-J. (1982). Theory of serial pattern production: Tree traversals. *Psychological Review, 85*, 693-707.

Coren, S. (1986). An efferent component in the visual perception of direction and extent. *Psychological Review, 93*, 391-410.

Dell, G. S. (1986). A spreading-activation theory of retrieval in sentence production. *Psychological Review, 93*, 283-321.

Delatizky, J. (1982). Final position control in simulated planar arm movement. *Neuroscience Abstracts, 8*, 283-292.

Easton, T. A. (1972). On the normal use of reflexes. *American Scientist, 60*, 591-599.

Fay, D. & Cutler, A. (1977). Malapropisms and the structure of the mental lexicon. *Linguistic Inquiry, 8*, 505-520.

Fel'dman, A. G. (1986). Once more on the equilibrium-point hypothesis (γ model) for motor control. *Journal of Motor Behavior, 18*, 17-54.

Flash, T. & Hogan, N. (1985). The coordination of arm movements: An experimentally confirmed mathematical model. *The Journal of Neuroscience, 5*, 1688-1703.

Fodor, J. A. & Pylyshyn, Z. W. (1988). Connectionism and cognitive architecture: A critical analysis. *Cognition, 28*, 3-71.

Fowler, C. A. (1980). Coarticulation and theories of extrinsic timing. *Journal of Phonetics, 8*, 113-133.

Fowler, C. A., Rubin, P., Remez, R. E., & Turvey, M. T. (1981). Implications for speech production of a general theory of action. In B. Butterworth (Ed.), *Language production. Vol. 1: Speech and talk* (pp. 373-420). London: Academic Press.

Fowler, C. A. & Turvey, M. T. (1978). Skill acquistion: An event approach with special reference to searching for the optimum of a function of several variables. In G. Stelmach (Ed.), *Information processing in motor control and learning.* New York: Academic Press.

Fromkin, V. A. (1971). The nonanomalous nature of anomalous utterances. *Language*, *47*, 27-52.

Fu, K.S. (1970). Stochastic automata as models of learning systems. In J. M. Mendel & K. S. Fu (Eds.), *Adaptive, Learning, and pattern recognition systems.* New York: Academic Press.

Garrett, M. F. (1975). The analysis of sentence production. In G. H. Bower (Ed.), *The psychology of learning and motivation.* New York: Academic Press.

Gentner, D. R., Grudin, J., & Conway, E. (1981). *Finger movements in transcription typing* (Tech. Rep. 8001). San Diego: University of California,Center for Human Information Processing.

Golden, R. M. (1986). The "brain-state-in-a-box" neural model is a gradient descent algorithm. *Journal of Mathematical Psychology, 30*, 73-80.

Gordon, P. C. & Meyer, D. E. (1987). Control of serial order in rapidly spoken syllable sequences. *Journal of Memory and Language*, *26*, 300-321.

Greene, P. H. (1972). Problems of organization of motor systems. *Progress in Theoretical Biology*, *2*, 303-338.

Greeno, J. G. & Simon, H. A. (1974). Processes for sequence production. *Psychological Review, 81*, 187-197.

Greenwald, A. G. (1970). Sensory feedback mechanisms in performance control: With special reference to the ideo-motor mechanism. *Psychological Review*, 7, 73-99.

Grossberg, S. (1978). A theory of visual coding, memory, and development. In R. Rosen & F. Snell (Eds.), *Progress in Theoretical Biology*, (Vol. 5). New York: Academic Press.

Grossberg, S. & Kuperstein, M. (1986). *Neural dynamics of adaptive sensory-motor control: Ballistic eye movements.* Amsterdam: North-Holland.

Grudin, J. T. (1981). *The organization of serial order in typing.* Unpublished doctoral dissertation, University of California, San Diego.

Gupta, M. (1986). *Adaptive methods for control system design.* New York: IEEE Press.

Halwes, T. & Jenkins, J. J. (1971). Problem of serial order in behavior is not resolved by context-sensitive associative memory models. *Psychological Review, 78*, 122-129.

Hinton, G. E. (1984). Parallel computations for controlling an arm. *Journal of Motor Behavior, 16*, 171-194.

Hogan, N. (1984). An organising principle for a class of voluntary movements. *Journal of Neuroscience, 4*, 2745-2754.

Hogan, N. (1985). The mechanics of multi-joint posture and movement control. *Biological Cybernetics, 53*, 1-17.

Hollerbach, J. M. (1982). Computers, brains, and the control of movement. *Trends in Neuroscience, 5*, 189-193.

Hollerbach, J. M., & Flash, T. (1982). Dynamic interactions between limb segments during planar arm movement. *Biological Cybernetics, 44*, 67-77.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences, 79*, 2554-2558.

Inhoff, A. W., Rosenbaum, D. A. Gordon, A. M. & Campbell, J. A. (1984). Stimulus-response compatibility and motor programming of manual response sequences. *Journal of Experimental Psychology, 10*, 724-733.

Jordan, M. I. (1986a). *Serial order: A parallel, distributed processing approach* (ICS-8604). La Jolla, CA: University of California, Institute for Cognitive Science.

Jordan, M. I. (1986b). Attractor dynamics and parallelism in a connectionist sequential machine. *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum.

Jordan, M. I. (1988). *Sequential dependencies and systems with excess degress of freedom*. (COINS Tech. Rep. 88-27). Amherst, MA: University of Massachusetts, Computer and Information Sciences.

Kahn, M. E. (1969). *The near-minimum time control of open-loop articulated kinematic chains.* (Stanford Artificial Intelligence Memo No. 106). Stanford, CA.

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

Kelso, J. A. S., Holt, K. G., Kugler, P. N., & Turvey, M. T. (1980). On the concept of coordinative structures as dissipative structures: II. Empirical lines of convergency. In G. E. Stelmach & J. Requin (Eds.), *Tutorials in motor behavior*, (pp. 49-70). Amsterdam: North-Holland.

Kelso, J. A. S., Tuller, B., Vatikiotis-Bateson, E., & Fowler, C. A. (1984). Functionally specific articulatory cooperation following jaw perturbations during speech: Evidence for coordinative structures. *Journal of Experimental Psychology: Human Perception and Performance, 10*, 812-832.

Kent, R. D. & Minifie, F. D. (1977). Coarticulation in recent speech production models. *Journal of Phonetics, 5*, 115-117.

Khatib, O. (1987). A unified approach for motion and force control: The operational space formulation. *IEEE Journal of Robotics and Automation, 3*, 43-54.

Kirk, D. E. (1970). *Optimal control theory.* Englewood Cliffs, NJ: Prentice-Hall.

Kugler, P. N., Kelso, J. A. S., & Turvey, M. T. (1980). On the concept of coordinative structures as dissipative structures: I. Theoretical lines of convergence. In G. E. Stelmach & J. Requin (Eds.), *Tutorials in motor behavior*, (pp. 49-70). Amsterdam: North-Holland.

Kuperstein, M. (1987). Adaptive visual-motor coordination in multijoint robots using parallel architecture. *IEEE Conference on Robotics and Automation*, pp. 1595-1602.

Ladefoged, P., DeClerk, J., Lindau, M. & Papçun, G. (1972). An auditory-motor theory of speech production. *UCLA Working Papers in Linguistics*, *22*, 48-75.

Lashley, K. S. (1951). The problem of serial order in behavior. In L. A. Jeffress (Ed.), *Cerebral mechanisms in behavior.* New York: Wiley.

Liberman, A. M., Cooper, F. S. Shankweiler, D. P., & Studdert-Kennedy, M. G. (1967). Perception of the speech code. *Psychological Review*, *74*, 431-461.

Luchins, A. S. (1942). Mechanization in problem solving. *Psychological Monographs*, 54 (Whole No. 248).

MacKay, D. G. (1970). Spoonerisms: The structure of errors in the serial order of speech. *Neuropsychologia*, *8*, 323-350.

MacKay, D. G. (1982). The problem of flexibility, fluency, and speed-accuracy trade-off in skilled behavior. *Psychological Review*, *89*, 483-506.

McClelland, J. L., & Kawamoto, A. H. (1986). Mechanisms of sentence processing: Assigning roles to constituents. In J. L. McClelland & D. E. Rumelhart (Eds.), *Parallel distributed processing: Volume 2.* Cambridge, MA: MIT Press.

Melsa, J. L. & Schultz, D. G. (1969). *Linear control systems.* New York: McGraw Hill.

Mendel, J. M. & McLaren, R. W. (1970). Reinforcement-learning control and pattern recognition systems. In J. M. Mendel & K. S. Fu (Eds.), *Adaptive learning and pattern recognition systems.* New York: Academic Press.

Meyer, D. E. & Gordon, P. C. (1983). Speech production: Motor programming of phonetic features. *Journal of Memory and Language, 24*, 3-26.

Miller, N. (1986). *Dyspraxia and its management.* London: Croom Helm.

Moll, K. L. & Daniloff, R. G. (1971). Investigation of the timing of velar movements during speech. *Journal of the Acoustical Society of America, 50*, 678-684.

Mowrer, O. H. (1960). *Learning theory and behavior.* New York: Wiley.

Nelson, W. (1983). Physical principles for economies of skilled movements. *Biological Cybernetics, 46*, 135-147.

Norman, D. A. & Shallice, T. (1986). Attention to action: Willed and automatic control of behavior. In R. J. Davidson, G. E. Schwartz, & D. Shapiro (Eds.), *Consciousness and self-regulation: Advances in research, Vol. IV.* New York: Plenum Press.

Öhman, S. E. G. (1966). Coarticulation in VCV utterances: spectrographic measurements. *Journal of the Acoustical Society of America, 39*, 151-168.

Olsen, R. A. & Murray, R. A., III. (1976). Finger motion in typing texts of varying complexity. *Proceedings 6th Congress of the International Ergonomics*

Association, 446-450.

Partridge, L. (1986). Frogs solve Bernstein's problem. *The Behavioral and Brain Sciences*, *9*, 619-620.

Pew, R. W. (1974). Human perceptual-motor performance. In B. H. Kantowitz (Ed.), *Human information processing: Tutorials in performance and cognition* (pp. 1-39). Hillsdale, NJ: Lawrence Erlbaum.

Polit, A. & Bizzi, E. (1978). Processes controlling arm movements in monkeys. *Science*, *201*, 1235-1237.

Rabbitt, P. M. A., Vyas, S. M., & Fearnley, S. (1975). Programming sequences of complex responses. In P. M. A. Rabbitt & S. Dornic (Eds.), *Attention and performance V*. London: Academic Press.

Raibert, M. H. (1978). A model for sensorimotor control and learning. *Biological Cybernetics*, *29*, 29-36.

Reich, P. A. (1977). Evidence for a stratal boundary from slips of the tongue. *Forum Linguisticum*, *2*, 211-217.

Reitman, J. S. & Rueter, H. H. (1980). Organization revealed by recall orders and confirmed by pauses. *Cognitive Psychology*, *12*, 554-581.

Restle, F. (1970). Theory of serial pattern learning: Structural trees. *Psychological Review*, *77*, 481-495.

Rosenbaum, D. A., Kenny, S., & Derr, M. A. (1983). Hierarchical control of rapid movement sequences. *Journal of Experimental Psychology: Human Perception and Performance*, *9*, 86-102.

Rosenbaum, D. A. (1985). Motor programming: A review and scheduling theory. In H. Heur, U. Kleinbeck, & K-M. Schmidt (Eds.), *Motor behavior: Programming, control, and acquisition* (pp. 1-33). Berlin: Springer-Verlag.

Rosenbaum, D. A., Gordon, A. M., Stillings, N. A., & Feinstein, M. H. (1987). Stimulus-response compatibility in the programming of speech. *Memory and Cognition*, *15*, 217-224.

Rosenbaum, D. A., Weber, R. J., Hazelett, W. M., & Hinforff, V. (1986). The parameter remapping effect in human performance: Evidence from tongue twisters and finger fumblers. *Journal of Memory and Language*, *25*,

Raibert, M. H. (1986). Symmetry in running. *Science*, *231*, 1292-1293.

Rumelhart, D. E., Hinton, G. E., Williams, R. J. (1986). Learning internal representations by error propagation. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel distributed processing: Volume 1.* Cambridge, MA: MIT Press.

Rumelhart, D. E. & Jordan, M. I. (1988). *Supervised learning with a distal teacher.* Paper in preparation.

Rumelhart, D. E. & Norman, D. A. (1982). Simulating a skilled typist: A study of skilled cognitive-motor performance. *Cognitive Science*, *6*, 1-36.

Sakitt, B. (1980). A spring model and equivalent neural network for arm posture control. *Biological Cybernetics*, *37*, 227-234.

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

Saltzman, E. L. (1986). Task-dynamic coordination of the speech articulators: A preliminary model. In H. Heuer & C. Fromm (Eds.), *Generation and modulation of action patterns* (Experimental Brain Research Series 15, pp. 129-144). New York: Springer-Verlag.

Saltzman, E. L. & Kelso, J. A. S. (1987). Skilled actions: A task-dynamic approach. *Psychological Review, 94*, 84-106.

Schmidt, R. A. (1975). A schema theory of discrete motor skill learning. *Psychological Review, 82*, 225-260.

Schmidt, R. A. (1982). *Motor control and learning.* Champaign, IL: Human Kinetics Publishers.

Shapiro, D. C. & Schmidt, R. A. (1981). The schema theory: Recent evidence and developmental implications. In J. A. S. Kelso & J. E. Clark (Eds.), *The development of movement control and coordination.* New York: Wiley.

Shattuck-Hufnagel, S. (1979). Speech errors as evidence for a serial- ordering mechanism in sentence production. In W. E. Cooper & E. C. T. Walker (Eds.), *Sentence processing: Psycholinguistic studies presented to Merrill Garrett.* Hillsdale, NJ: Erlbaum.

Simon, H. A. (1969). *The sciences of the artificial.* Cambridge, MA: MIT Press.

Singer, R. N. (1980).*Motor learning and human performance* (3rd ed.). New York: Macmillan.

Smolensky, P. (1988). On the proper treatment of correctionism. *Behavioral and Brain Sciences, 11*, forthcoming.

Sternberg, S., Monsell, S., Knoll, R. L., & Wright, C. E. (1978). The latency and duration of rapid movement sequences: Comparisons of speech and typewriting. In G. E. Stelmach (Ed.), *Information processing in motor control and learning* (pp. 117-152). New York: Academic Press.

Stevens, K. N. (1972). The quantal nature of speech: Evidence from articulatory-acoustic data. In E. E. David, Jr. & P. B. Denes (Eds.), *Human Communication: A unified view.* New York: McGraw-Hill.

Sutton, R. S. (1984). *Temporal aspects of credit assignment in reinforcement learning.* Unpublished doctoral dissertation University of Massachusetts, Amherst.

Thorndike, E. L. (1911). *Animal intelligence.* Darien, CT: Hafner.

Turvey, M. T. (1977). Preliminaries to a theory of action with reference to vision. In R. Shaw & J. Bransford (Eds.), *Perceiving, acting, and knowing: Toward an ecological psychology.* Hillsdale, NJ: Erlbaum.

Vidyasagar, M. (1978). *Nonlinear systems analysis.* Englewood Cliffs, NJ: Prentice-Hall.

Wickelgren, W. A. (1969). Context-sensitive coding, associative memory, and serial order in (speech) behavior. *Psychological Review, 76*, 1-15.

Widrow, B., Guptra, N. K. & Maitra, S. (1973). Punish/reward: Learning with a critic in adaptive threshold systems. *IEEE Transactions on Systems,* *Man and Cybernetics, 5*, 455-465.

Widrow, B. & Hoff, M. E. (1960). Adaptive switching circuits. *Institute of Radio Engineers, Western Electronic Show and Convention, Convention Record, Part 4*, 96-104.

Widrow, B. & Stearns, S. D. (1985). *Adaptive signal processing.* Englewood Cliffs, NJ: Prentice-Hall.

Williams, R. J. (1987). *Reinforcement-learning connectionist systems.* (Tech. Rep. NU-CCS-87-3). Northeastern University, College of Computer Science.