Technical Report 439

Motor Control and Learning by the State Space Model

Marc H. Raibert

MIT Artificial Intelligence Laboratory

This blank page was inserted to preserve pagination.

AI-TR-439

MOTOR CONTROL AND LEARNING BY THE STATE SPACE MODEL

Marc H. Raibert

September 1977

ARTIFICIAL INTELLIGENCE LABORATORY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Motor Control and Learning by the State Space Model

by

Marc H. Raibert

Massachusetts Institute of Technology

September 1977

Revised version of a dissertation submitted to the Department of Psychology on September 8, 1977 in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Abstract

A model is presented that deals with problems of motor control, motor learning, and sensorimotor integration. The equations of motion for a limb are parameterized and used in conjunction with a quantized, multi-dimensional memory organized by state variables. Descriptions of desired trajectories are translated into motor commands which will replicate the specified motions. The initial specification of a movement is free of information regarding the mechanics of the effector system. Learning occurs without the use of error correction when practice data are collected and analyzed.

The model was implemented using a small computer and the MIT-Scheinman manipulator. Experiments were conducted which demonstrate the controller's ability to learn new movements, adapt to mechanical changes caused by inertial and elastic loading, generalize its behavior among similar movements, and use a variety of coordinate systems for learning. A second generation model, based on improvements suggested by these experiments, is discussed.

The following are features of the implemented State Space Model:

1) Complexity: The computations performed by the model are quite simple, and especially suited to a parallel processing device.
2) Mechanical Interactions: Systematic compensation is provided for interactions between joints.
3) Constraints: Only weak constraints need be placed on the geometry of the limb under control, its actuators, and its sensors.
4) Remapping: The sensory information used by the model may be related to the joints of the limb, to visual space, or to any of a large class of coordinate systems.
5) Learning: The system demonstrates gradual improvement in performance as it gains experience from self-produced practice movements. The performance of selected movements can be more rapidly improved through intensive practice.
6) Generalization: Practice of one movement can improve performance of other similar movements, thereby showing a kind of transfer of training.
7) Adaptation: Adaptations to mechanical and certain sensory changes take place without an explicit error correction procedure.

Acknowledgments

This work could not have been done without the patient help of Fred Drenckhahn. He kept the Scheinman manipulator in good health and working order despite experimentation which often lead to its abuse.

My thanks go to Tomaso Poggio for introducing me to the generalized inverse. Thanks also to Nancy Cornelius, John Hollerbach, Berthold Horn, David Marr, Matt Mason, Ron Pankiewicz, Andres Polit, and Whitman Richards for their critical readings. Nancy Cornelius also helped prepare the figures.

Special thanks are due Whitman Richards whose encouragement helped me over the hump, and Nancy Cornelius for putting up with it all.

Table of Contents

Abstract 3

Acknowledgements 4

Table of Contents 5

List of Illustrations 9

Chapter 1 Introduction 11
1.1 Goals 12
1.2 Organization of Report 13
1.3 A Model for Motor Learning and Sensorimotor Integration:
An Overview 13
1.3.1 Introduction 13
1.3.2 Model Description 15
1.3.3 Properties of the Model 20
1.3.4 Methods 22
1.3.5 Results 25
1.3.6 Discussion 33
1.4 Analytical Equations vs. Table Look-up:
A Unifying Concept 34
1.4.1 Introduction 34
1.4.2 Parameterization 36
1.4.3 Parameterizing Equations of Motion 37
1.4.4 P = 0, N, 2N, 3N 38
1.4.5 State Space Model (SSM) 41
1.4.6 Configuration Space Method (CSM) 42

Chapter 2 The Problem 45
2.1 Constraining the Issues 45
2.1.1 Many Factors at Work 45
2.1.2 Emphasis on Pre-planning 46
2.1.3 High- and Low-Level Specialization 49
2.2 Mechanical Problems 52
2.2.1 Equations of Motion for a Limb 52
2.3 One Control Problem or Two? 57

Chapter 3 The State Space Model (SSM) 59
3.1 The Forward Computation -- *Translation* 59
3.2 The Inverse Computation -- *Learning* 61
3.2.1 Historical Perspective 61
3.2.2 The Inversion Equation 63
3.2.3 The Command-Torque Relationship 65
3.3 The State Space Memory and the Temporary Buffer 66
3.4 Combined Operation of the Components 68
3.5 Properties of Model 70
3.6 Discussion of Model 73

Chapter 4 Implementation and Test 73
4.1 Facility 79
4.2 Information Processing 85
4.2.1 Computation of the Inverse 85
4.2.1.1 Invertability Index 85
4.2.1.2 Use of the Generalized Inverse 86
4.2.2 The State Space Memory 88
4.2.2.1 Initialization 91
4.2.2.2 Time Constants 91
4.2.3 Translation 93
4.2.3.1 The Neighborhood Function 93
4.3 Tools for Testing 94
4.3.1 Prototypes 94
4.3.2 Two Types of Movements 96
4.3.3 Performance Indices 98

4.3.3.1 Competence Index 99
4.3.4 Practice Algorithm 101
4.4 Test Procedures 107

Chapter 5 Results and Discussion 109
5.1 Control and Learning 109
5.2 Generalization 116
5.2.1 The First Generalization Test 116
5.2.2 The Second Generalization Test 123
5.2.3 Type II Generalization 126
5.3 Adaptation 133
5.3.1 Inertial and Elastic Loads 146
5.3.2 Reorientation of Gravity 142
5.4 Flexibility of Coordinate System 144
5.5 General Discussion 146
5.5.1 Distributed vs Massed Trials 150
5.5.2 A Use for Error Data 150
5.5.3 The SSM and Optimal Control 151
5.5.4 A Fair Test of the Model? 152
5.6 Improvements to the Model 153
5.6.1 Insuring the Command-Force Relationship 153
5.6.2 Practice Improves Practice 155
5.6.3 Decaying Measurement Vectors 157

Chapter 6 Concluding Remarks 160
6.1 Derivative and Alternative Models 160
6.1.1 The Measurement Space Model 160
6.1.2 Configuration Space Method 161
6.1.3 Multiple Spaces Model 162
6.1.3 Visually Locate and Move 163
6.2 Problems for Further Research 165
6.2.1 Measurement + Error Correction 165
6.2.2 Plan + Servo 166
6.2.3 Practice 166
6.2.4 High Level Processes 167
6.3 Summary 168

Bibliography 170

Appendix: Experimental Conditions 177

Glossary of Terms 178

Glossary of Variables 181

List of Illustrations

Chapter 1

Fig. 1.1 Block diagram of model 21
1.2 The manipulator used for testing 24
1.3 Loads are applied to the manipulator 26
1.4 Learning and adaptation curves 27
1.5 Variations of the memory's time-constant 29
1.6 Learning curves showing generalization 30
1.7 Generalization gradients 32

Chapter 2

Fig. 2.1 Writing with various limbs 50
2.2 Schematic of arm showing position dependencies . . 54

Chapter 3

Fig. 3.1 Block diagram of the model 69

Chapter 4

Fig. 4.1 The MIT-Scheinman manipulator 80
4.2 Interactions between joints 82
4.3a Repeatability of acceleration estimates 83
4.3b Information content of acceleration estimates . . . 84
4.4 Overconstrained use of the generalized inverse. . . 87
4.5 Usable regions of motor space 89
4.6 Memory time-constant 92
4.7 A typical prototype 97
4.8 Distribution of practice data 100
4.9 A series of practice movements 102
4.10 Varying practice parameters 105
4.11 Practice by parts 106

Chapter 5

Fig. 5.1 Test movement series, PR-11 110
5.2 Learning curve, PR-11 112
5.3 Test movement series, PR-12 113
5.4 Learning curve, PR-12 114
5.5 Learning gravity compensation 117
5.6 Prototypes PR-10,11,12,13 118
5.7 Generalization learning curves, PR-10,11,12,13 . . 119
5.8 Generalization gradients, PR-10,11,12,13 121
5.9 Normalized competence learning curves, PR-10,11,12,13 124
5.10 Prototypes PR-20,21,22,23,24 125
5.11 Generalization learning curves, PR-20,21,22,23,24 . 127
5.12 Generalization gradients, PR-20,21,22,23,24 . . . 129
5.13 Learning with previous experience 131
5.14 Comparison of learning with and without experience . 132
5.15 Application of inertial and elastic loads 134
5.16 Adaptation curves, PR-11,12 136
5.17 Transient behavior when changing conditions. . . . 137
5.18 Re-initializing the temporary buffer 138
5.19 Varying the memory's time constant, ... 140
5.20 Re-initializing the state space memory 141
5.21 Arm mounted on wall 143
5.22 Simulation of Cartesian coordinate sensors 145
5.23 Cartesian prototype, PR-11XYZ 147
5.24 Learning curve for cartesian coordinates 148
5.25 Memory decay with increasing time-constants. . . . 159

1 Introduction

The human motor system is characterized by properties which are not exhibited by traditional man-made machines. Most basic of these properties is the ability to learn. Initially the human infant exhibits discoordinated movements which have no apparent purpose and are skillessly executed (Twitchell, 1965; 1970). But as the child develops, his movements take on a different character. They become directed and effective, smooth and graceful. The improved dexterity is attributable in part to the experience the developing organism receives from his own attempts to move (Bilodeau, 1966; Conolly, 1970; White, 1970; Held & Bauer, 1974). The adult, moreover, is able to select particular movements and center his attention upon them through practice until a high level of performance has been reached. The human is not limited to making only those movements which have been the subject of previous practice -- it is often the case that a movement which has never before been attempted can be executed with a fair degree of precision (Mednick, 1964; Welford, 1968).

Not only are we able to gain motor control of our bodies through ontogeny but we are able to maintain this control. Under normal circumstances we make the adjustments needed to control our limbs even though the masses and sizes of the various parts of the body undergo large changes throughout ontogeny. In the laboratory we are able to compensate for experimentally induced distortions made to our sensory inputs or the environment (Hein & Held, 1962; Held, 1961; Young, 1969; Kornheiser, 1976).

Finally, our limbs are useful tools only if they will do our bidding, but our wishes are phrased in a language which motoneurons and muscle do not understand. If we start with the simple, perhaps schematic instruction, "Close your eyes and move your hand so that the tip of your finger travels a path which is a straight line. " we are able to comply. We are able to comply even though this specification of the movement of the finger gives no explicit information about the requisite joint movements or muscle forces. This means that our motor system is able to convert a description of a movement given in one coordinate frame into a set of commands which are suited to act in an entirely different frame -- that of bone, joint, and muscle (Marr, 1969; Gelfand et al., 1971; Arbib, 1972).

1.1 Goals

The purpose of this thesis is to attack two related questions: First, how is the human nervous system able to achieve such exquisite motor control? Second, how can we make machines that perform with similar elegance? More specifically, by presenting a model and working implementation that exhibit properties remeniscient of human performance, I will examine a number of issues of fundamental importance to motor control and motor learning. I take the point of view that there is only one control problem which governs man-made mechanical arms and biological limbs (which are also largely mechanical in nature). We know that solutions to the limb control problem are possible because the human provides a superb existence proof. Demonstrating that the same solutions apply to both domains is made difficult at times, however, because no proofs of uniqueness are available and, indeed, may not in principle be possible.

1.2 Organization of Thesis

In the next section of this chapter, (Section 1.3), I highlight the important features of this work by presenting the problem, the State Space Model, and some of the experimental findings, all in a nutshell. By showing the relationship between the proposed model and other methods of arm control, Section 1.4 helps the reader to view this research in the proper perspective.

The details of the work are presented in the remaining chapters: Chapter 2 focusses on the problem of controlling a multi-linked arm as it moves through 3-dimensional space. Chapter 3 presents details of the State Space Model, and a description of its properties. In Chapter 4 I discuss the practical problems associated with developing an implementation of the model that is used to control a physical arm. Experimental data that illucidate the model's behavior are presented and discussed in Chapter 5. Chapter 6 concludes this report by proposing lines along which the research can continue.

1.3 A Model for Motor Learning and Sensorimotor Integration: An Overview

1.3.1 Introduction

After two decades of intensive study, control theorists, interested in controlling more complicated non-linear devices (Bryson & Ho, 1969), and physiologists, guided by experimental findings (Hammond, 1956; Melvill Jones & Watt, 1971a), have begun to look beyond the servo control feedback mechanism in order to examine the merits of *pre-planning* and the *central program* (Evarts et al., 1970; Melvill Jones & Watt, 1971b). For a limb comprised of interacting degrees of freedom, the transformation

from a desired trajectory into such a motor plan, a set of actuator control signals, is a computationally expensive operation. Yet the nervous system's ability to use motor plans interchangebly with a number of effector systems argues that the problem has been efficiently solved (Raibert, 1976). Moreover, the biological solution allows the organism to learn through practice, to generalize training between similar movements, and to adapt to mechanical and sensory changes (Held & Hein, 1963; White, 1970; Miles & Fuller, 1974; Gonshor & Melvill Jones, 1976).

Experiments by Held (1961) and Hein and Held (1963), and a model proposed by Marr (1969) have combined to motivate a new model for motor control, motor learning, and sensorimotor integration. The idea that an internal signal, Helmholtz's *efference copy* (1867), distinguishes an organism's self-produced movements from externally induced movements lead to Held and Hein's now classical experiments. Their results, showing that active movement is essential to motor learning and sensorimotor adaptation, suggest the nervous system assesses the response characteristics of the limbs using an input-output analysis. The problem remains to formulate the extremely complicated equations of motion characterizing a limb's mechanical behavior in a way which permits such an input-output analysis. Marr supplies the clue in his cerebellar model by stating that the *context* in which an elemental movement is made influences the movement's execution. Extensions of this idea show that using state variables as parameters produces dramatic simplifications in the equations of motion, a result which lays the groundwork for the present model (Raibert, 1977b).

Two interacting processes plus auxiliary memory functions explain learning of

new movements, transfer of training between similar movements, and adaptation to mechanical and sensory changes. Parameterization, the process of restating an equation with a subset of the independent variables held constant, recasts the equations of motion into a very simple form that allows learning based on practice. The parameterized equations, however, must be used in conjunction with a multi-dimensional memory in which *constants of mechanical description*, also parameterized by state variables, are stored. Learning, the process which supplies data to this tabular memory, takes place when torque vectors applied to the limb, T_m, are correlated with resulting accelerations vectors, $\ddot{\theta}$. Properties of the memory, its time-constant and accessing function, contribute to adaptation and generalization.

The power and simplicity of the model derive from the combined use of parameterization and learning. Without learning, the constants that make the parameterized equations usable can only be found by evaluating extremely complicated equations. Learning without parameterization, on the other hand, requires inversion of non-linear trigonometric differential equations comprised of thousands of terms. Parameterization makes learning possible, and learning makes parameterization usable.

1.3.2 Model Description

When each of the terms contributing to the torque acting on the joints of a limb are included, Newton's equation for rotary motion may be expressed schematically as:

$$T_m - \bar{G}(\theta) - B(\dot{\theta}) - C(\theta,\dot{\theta}) = J(\theta)\ddot{\theta} \qquad \text{(eq. 1.1)}$$

where:
T_m is the actuator torque vector
G is a vector-function for gravitational torque
B is a vector-function for frictional torque
C is a vector-function for Coriolis torque
J is a matrix-function for moment of inertia
θ, $\dot{\theta}$, and $\ddot{\theta}$ are the position, velocity, and acceleration vectors.

The full set of time-varying, non-linear equations with explicit expression of θ- and $\dot{\theta}$-dependencies has been worked out by Kahn (1969). His equations involve about 1600 terms and 13,000 multiplications for a general 3 degree of freedom limb.

The Translation Equations

By treating the state variables θ and $\dot{\theta}$ as parameters, (ie. letting them assume a number of fixed values), a simplified parametric form of Kahn's equations can be found:

$$T_m - G|_{(\theta=\alpha)} - B|_{(\dot{\theta}=\beta)} - C|_{(\theta=\alpha,\dot{\theta}=\beta)} = J|_{(\theta=\alpha)}\cdot\ddot{\theta} \qquad \text{(eq. 1.2a)}$$

where:
α parametric position vector
β parametric velocity vector

Or, more compactly:

$$T_m - G_\alpha - B_\beta - C_{\alpha\beta} = J_\alpha\cdot\ddot{\theta} \qquad \text{(eq. 1.2b)}$$

Here, each of the vector-function relationships $G(\theta)$, $B(\dot{\theta})$, $C(\theta,\dot{\theta})$, and $J(\theta)$ has become a parameterized set of constants. By grouping terms and making the equation explicit in muscle torque one further simplification can be made:

$$T_m = J_\alpha\cdot\ddot{\theta} + K_{\alpha\beta} \qquad \text{(eq. 1.3)}$$

where:
$K_{\alpha\beta} = G_\alpha + B_\beta + C_{\alpha\beta}$

Eq. 1.3 is the *translation equation.* It is linear in $\ddot{\theta}$ and without hidden dependencies on θ or $\dot{\theta}$.

We now define a state space having 2N dimensions, and associate one state variable, $\{\theta_1, \theta_2 \ldots \theta_N, \dot{\theta}_1, \dot{\theta}_2 \ldots \dot{\theta}_N\}$ with each dimension. For any point in this space, (α,β), there exists a set of values for J_α and $K_{\alpha\beta}$, such that Eq. 1.3 describes behavior of a limb when its state passes through that point. Furthermore, since the values of the components of J and K vary smoothly throughout the space, ie:

$$\nabla j_{ij,\alpha} < \infty \qquad (i=1,2 \ldots N;\ j=1,2 \ldots N)$$
$$\nabla k_{i,\alpha\beta} < \infty \qquad (i=1,2 \ldots N)$$

where:
∇ is the gradient operator

the space can be divided into a large number of hyper-regions throughout each of which the behavior of the limb, and the corresponding values of J and K, are reasonably uniform. This approach becomes useful when values of J and K corresponding to particular $(\theta, \dot{\theta})$ are available from a well organized tabular memory: Desired trajectories, $\ddot{\theta}_D(t)$, are processed by Eq. 1.3 after division into intervals of duration Δt, where different values for J and K for each interval are obtained from the *state space memory.*

Since data will not always be available for every hyper-region, (assuming the system begins tabula rasa), performance will be more robust if a memory accessing function is used that takes into account the gradual variations of mechanical behavior through state space -- if data from a particular hyper-region are not available, data from neighboring regions may be used instead. In addition to robustness, transfer of

training between similar practiced and unpracticed movements is an expected consequence of such an accessing function.

The Inversion Equations

Von Holst and Mittelstaedt used the efference copy, an internal copy of the motor command, to account for the fly's ability to distinguish between internally and externally produced changes in sensory stimulation (von Holst, 1954; Mittelstaedt, 1958). Their notion was that the relationship between an externally generated signal describing changes in sensory stimulation and an internally generated signal describing impending changes in the position of the sensory surface would always give unambiguous information about movement in the external world. In Held's model, (1961; Hein & Held, 1962) the Holstian view was augmented to allow attainment of perceptual accuracy even after changes were made to the meaning of the sensory signals. The efference copy was used to elicit the trace of previous reafference, which in turn was compared to the current afference. In 1965 Young and Stark modelled the ability of humans performing a tracking task to change control strategies when there were changes in the dynamics of the controlled element (Young & Stark, 1965). In that model the efference copy was used to drive an *internal dynamic model* of the controlled element, the output of which was compared with afference from the control task.

In the present model the relationship between efference copy, T_m and reafference, $\ddot{\theta}$, is used to compute descriptions of the mechanical properties of the limb, represented in Eq. 1.3 by J and K. The approach is somewhat similar to MacKay's idea of *evaluation* as opposed to *elimination* (MacKay, 1972). Here reafference is

obtained from measurements of the acceleration vector made from the limb's sensors, or other sensors which can monitor the limb's activity. Efference copy is a record of the actuator torque vector, internally available from the source of motor commands or possibly from actuator sensors. The use made of the efference copy in this model is somewhat unique in that there is no comparator, no error signal is calculated, and no error correction procedure is used. Rather, the limb's properties are found by examining the relationship between input and output, command and response. As a result the local minima problems associated with search procedures are avoided (Tsypkin, 1971).

Since the simplified equations of motion are linear, values of J and K can be found in a straightforward manner, provided that sets of measurements, $\{(T_{m,1},\ddot{\theta}_1), (T_{m,2},\ddot{\theta}_2) \ldots\}$ are available:

$$J = \Upsilon \cdot \ddot{\Theta}^{-1} \qquad \text{(eq. 1.4a)}$$

$$K = T_{av} - [\Upsilon \cdot \ddot{\Theta}^{-1}] \cdot \ddot{\theta}_{av} \qquad \text{(eq. 1.4b)}$$

where:

$\Upsilon = [T_1 \mid T_2 \mid \ldots \mid T_N] - [T_{N+1} \mid T_{N+1} \mid \ldots \mid T_{N+1}]$

$\ddot{\Theta} = [\ddot{\theta}_1 \mid \ddot{\theta}_2 \mid \ldots \mid \ddot{\theta}_N] - [\ddot{\theta}_{N+1} \mid \ddot{\theta}_{N+1} \mid \ldots \mid \ddot{\theta}_{N+1}]$

T_i and $\ddot{\theta}_i$ are the i'th measurements of T and $\ddot{\theta}$.

X_{av} denotes the average: $(X_1+X_2+ \ldots +X_{N+1})/(N+1)$.

(note: all torques are motor torques -- the m subscript has been dropped.)

Eq. 1.4 is the *inversion equation*. These calculations can be performed if N+1 input-output pairs, $(T_i,\ddot{\theta}_i)$, also called measurement vectors, are available. All measurements contributing to such a calculation must have been made while the limb was near a single hyper-region of interest. A temporary buffer is postulated to store such measurements until appropriate sets are available for inversion. The resulting

values of J and K are then stored in the state space memory in combination with previously stored data. The dynamic updating of the state space memory, adding data as they are available and combining them with old data, allows the system to adapt to changes in the limb's kinematic and dynamic properties, in addition to improving immunity to the effects of inverting noisy measurements, typically a problem when inverting physical data.

The block diagram shown in Fig. 1.1 summarizes the model's operation: High level processes produce descriptions of desired movements, $\ddot{\theta}_D(t)$, which are presented to the translator. The desired movement is sectioned into time intervals, each of duration Δt. For each time slice Eq. 1.3, the translation equations, used in conjunction with the constants of mechanical description, J and K values from the state space memory, generate a motor plan that will replicate the desired trajectory. The calculated force commands are issued to the limb and, during the movement, a copy of the command, T_m, and a copy of the sensory signals that indicate progress of the movement, $\ddot{\theta}$, are stored in a temporary buffer. Subsequently, the contents of the buffer and the inversion equations, Eq. 1.4, are used to calculate values of J and K, which are stored in the state space memory in combination with data that might have been stored there previously.

1.3.3 Properties of the Model

Initial performance will be quite poor since every attempt to use information about the mechanical character of the limb will be frustrated -- the state space memory will be tabula rasa -- empty. As movements of the limb are made, data

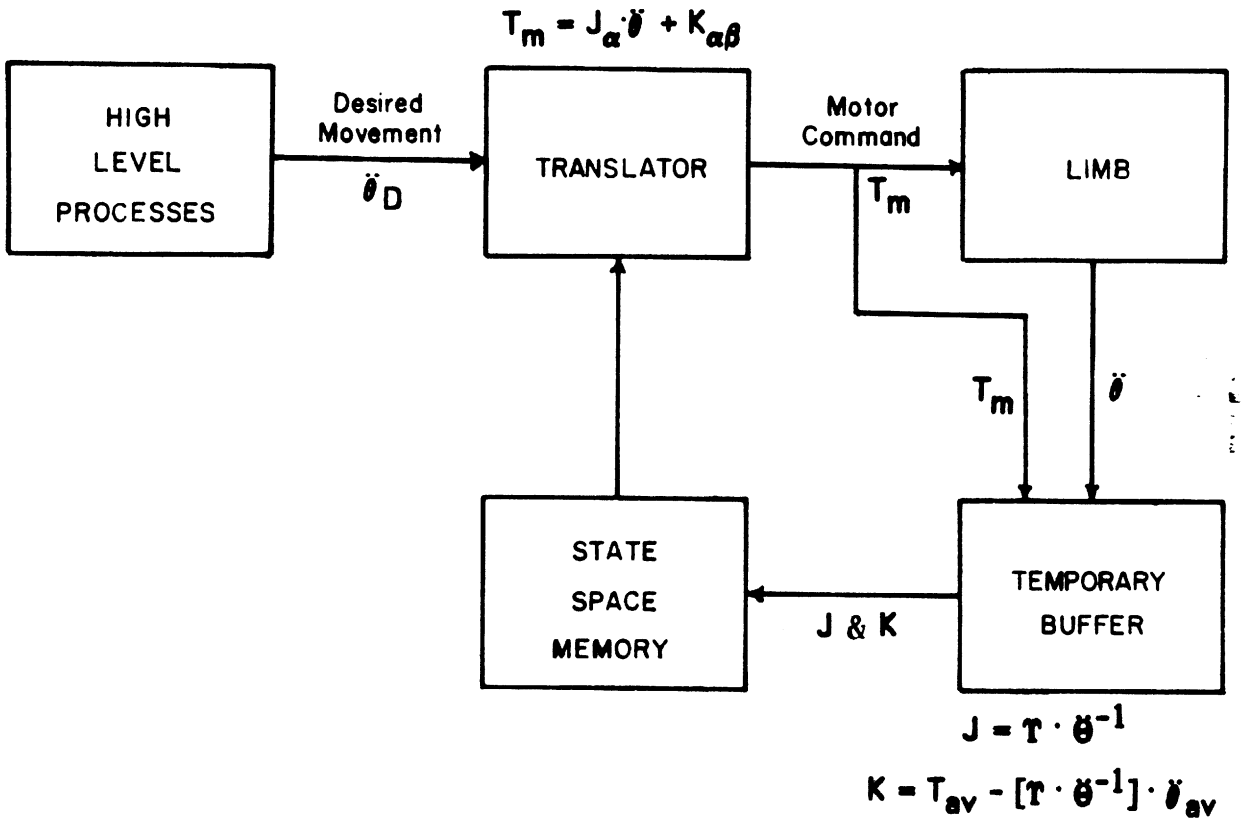


Fig. 1.1 Major components of the model. The translator converts descriptions of desired trajectories into motor commands suited to the kinematic and dynamic properties of a particular limb. The operation employs the tabular equations of motion in conjunction with the state space memory. Each movement of the limb generates data which, when processed by the inversion equations, contribute to the state space memory, and consequently, to future translations.

describing the mechanics of its operation become available. During this period of data acquisition the quality of translations will gradually improve. Practice will facilitate mastery of a practiced movement, while similar, (but not identical), movements will be improved more slowly. If the mechanical properties of the limb or sensors should change then the model adapts, since new constants of mechanical description are continuously being computed and stored.

The State Space Model can control limbs having a wide variety of dynamic and kinematic properties. A single translator can learn to control almost any limb or body part. This is a direct result of the tabular nature of the equations which describe the mechanical system. Though the development given above deals with torques applied to the joint, the actuator terms given in Eqs. 1.3 and 1.4 can be force applied to a tendon. In fact, actuators and sensors need not be affiliated with any one joint or subset of joints. Reafference can take the form of visual feedback just as readily as joint oriented proprioceptive feedback, provided the choice is made before learning commences and desired trajectories are described in the chosen coordinate system.

In order to evaluate and verify the power of the model, a set of computer programs embodying the various elements are used to control a mechanical arm. Tests of this implementation reveal the model's weaknesses and illustrate its strengths.

1.3.4 Methods

A PDP-11/45 computer is used to perform all computations, to issue commands to the manipulator, and to make measurements. The three joints of the MIT-Vicarm manipulator that allow the wrist to be positioned arbitrarily within the arm's work

space, are used, (N=3) (Horn & Inoue, 1974). See Fig. 1.2. Each joint is powered by a DC torque motor and provided with a potentiometer and tachometer. When a movement is made the computer specifies the torque delivered by each motor and measures angular position and velocity every 10 msec. In addition, velocity samples taken every 500 μsec allow the limb's average accelerations to be estimated over 60 msec intervals using least-mean-square error techniques, (Δt=60msec).

State Space Memory

Though only N+1 measurement vectors are theoretically required for each inversion, (here N+1=4), improved noise immunity is obtained by using the generalized inverse (Rust et al., 1966; Albert, 1972) to invert sets of 8 vectors. The resulting data are stored in a hash coded disk memory in weighted combination with data previously stored for the same hyper-region. Each new entry receives a weight of $1/\tau$, and previous data a weight of $(\tau-1)/\tau$, where τ is the memory's time constant.

The memory is 6 dimensional, (one dimension for each state variable), and quantized. Each dimension is partitioned into 10 ranges producing 10^6 possible hyper-regions. A single hyper-region measures $(15 \text{ deg})^3$ by $(13 \frac{\text{deg}}{\text{sec}})^3$. These regions are quite small and the mechanical properties of the arm are fairly constant throughout. Each access of the memory yields a weighted average of data from the desired hyper-region and all closest neighbors. Data from the desired hyper-region are given a weight equal to the number of times data were stored in that region. Neighbors are given a weight of 1 if any data are present, otherwise zero.

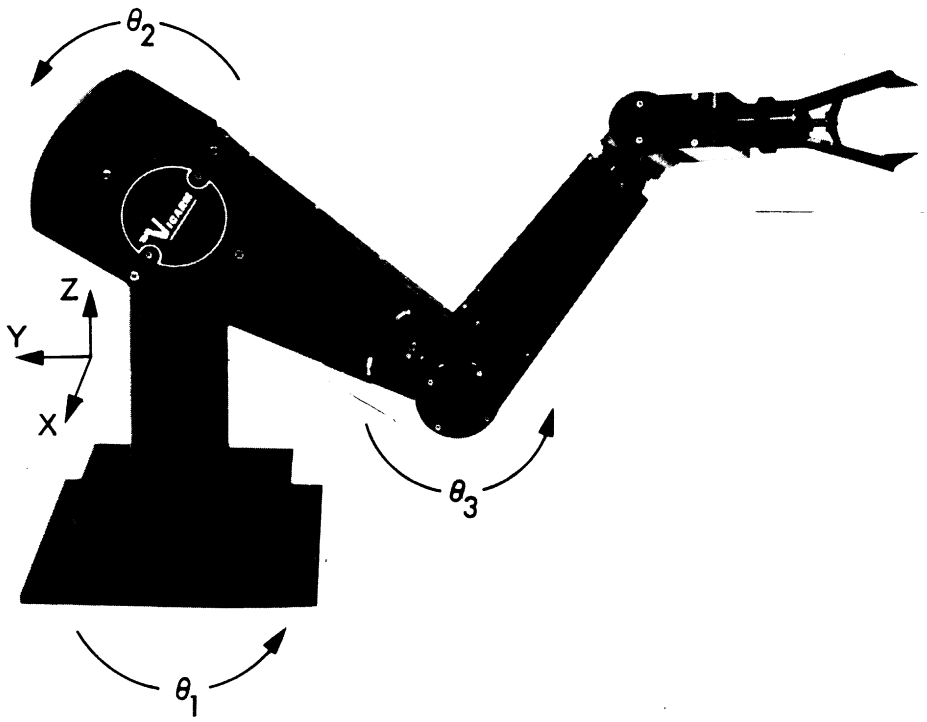


Fig. 1.2 Layout of the first three joints of the MIT-Vicarm manipulator. The manipulator is about the size of a human arm; base-to-shoulder = .273m, shoulder-to-elbow = elbow-to-wrist = .203m. Each joint is provided with a DC torque motor, a potentiometer, a tachometer, and a clutch-type brake.

Practice and Test

Input-output data are generated by exercising the arm under control of a practice program. This program generates a sequence of approximations to a pre-specified desired movement, called a ***prototype***. Periods of practice, analysis of practice data (Eq. 1.4), and execution of test movements, generated by Eq. 1.3 to test performance, are alternated during a learning session. Each test movement is evaluated by finding the root-mean-square position-error (RMS PE) or final-position-error, (RMS FPE) for the three joints.

After a baseline of practice is established, adaptation is measured by a manipulation of the mechanical state of the arm, (see Fig. 1.3), followed by continued training. Generalization of training is measured by testing performance of a set of prototypes, after practice of only one. The members of this set vary systematically in similarity to the practiced prototype. A learning index, LI, facilitates presentation of the generalization data:

$$LI = \frac{\Sigma(e_0 - e_i)}{\Sigma e_0}$$

where:

e_i is the RMS FPE for the i'th test movement
e_0 is the pre-training performance value.
Σ is the sum from $i=1$ to $n-1$
n is the number of test movements

1.3.5 Results

The left half of Fig. 1.4 is a learning curve for 3000 practice trials. As predicted, performance improves as more practice data are generated and analyzed. Rapid *jumps*

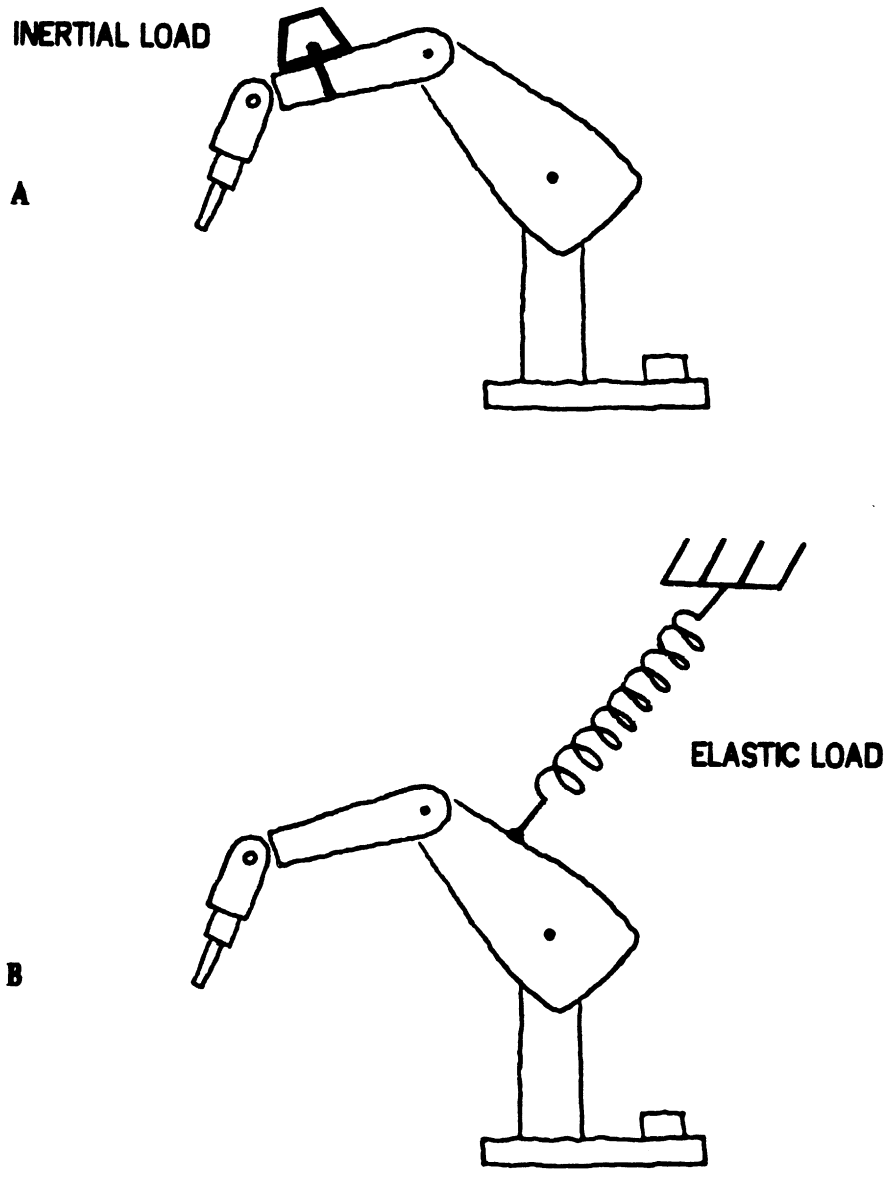


Fig. 1.3 Two methods of applying loads in order to disturb the manipulator's behavior are shown. A) A .19 kg. weight is attached to the third link of the manipulator. B) A 1.85 kg/m spring is attached from the second link to 'ground'. When movements start the spring is stretched .83m between coordinates (.17m,.0m,.25m) and (.02m,.70m,1.20m); see Fig. 1.2

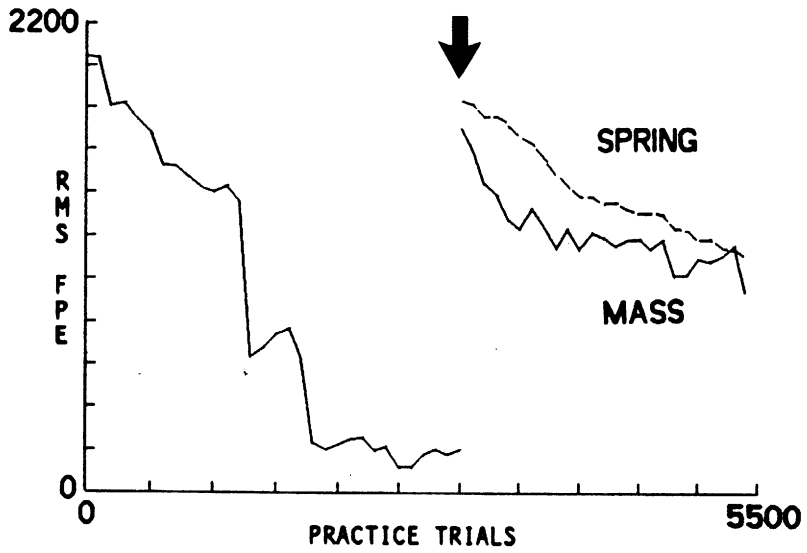


Fig. 1.4 Left) Acquisition of prototype PR-12 is shown as 3000 practice trials are executed and analyzed. Arrow) One of the two loads shown in Fig. 3 are applied. Right) The time course of adaptation to the two types of load is recorded. (τ=10)

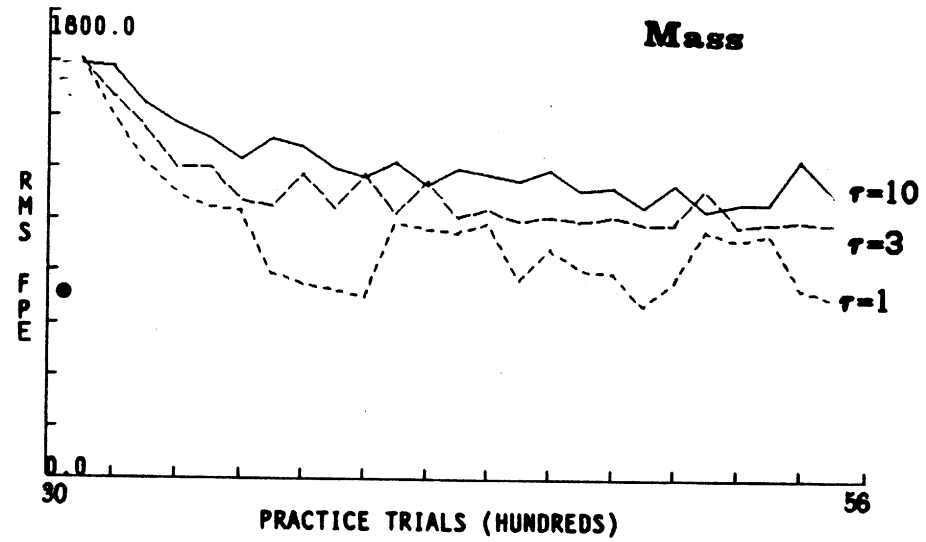
in performance arise when new hyper-regions of the memory are first provided with data, (asterisk in Fig. 1.4). When a load is applied to the arm, (arrow in Fig. 1.4), adaptation slowly takes place during the next 2500 practice trials under the new mechanical regime, as shown on the right side of Fig. 1.4. Modification of τ, the memory's time constant, results in improved rates of adaptation, though very small values of τ also introduce some instability. (See Fig. 1.5.)

Verification of the model's ability to generalize data derived from the practice of one movement to other similar movements is illustrated in Figs. 1.6 and 1.7. Throughout a 2400 trial learning session performance of the practiced prototype, PR-20, improves the most (Fig. 1.6a). Each of the other prototypes exhibit various degrees of improvement depending on their similarity to PR-20. (See caption to Fig. 1.6.) These generalization data are summarized quantitatively in Fig. 1.7 (diamonds), where the learning index, LI, is plotted for each prototype. To control for the possibility of gradients due to the particular choice of prototypes, a different member of the prototype set, PR-23, was practiced. The results, shown in Figs. 1.6b and 1.7 (triangles), reveal a similar pattern: the practice prototype shows the most improvement, with other movements improving according to their similarity to the practice prototype.

Figs. 1.4 through 1.7 verify the model's basic attributes:

1) Motor commands are generated suited to the kinematic and dynamic properties of the effector mechanism.

2) The quality of the motor commands improves with practice, though no error correction is used.



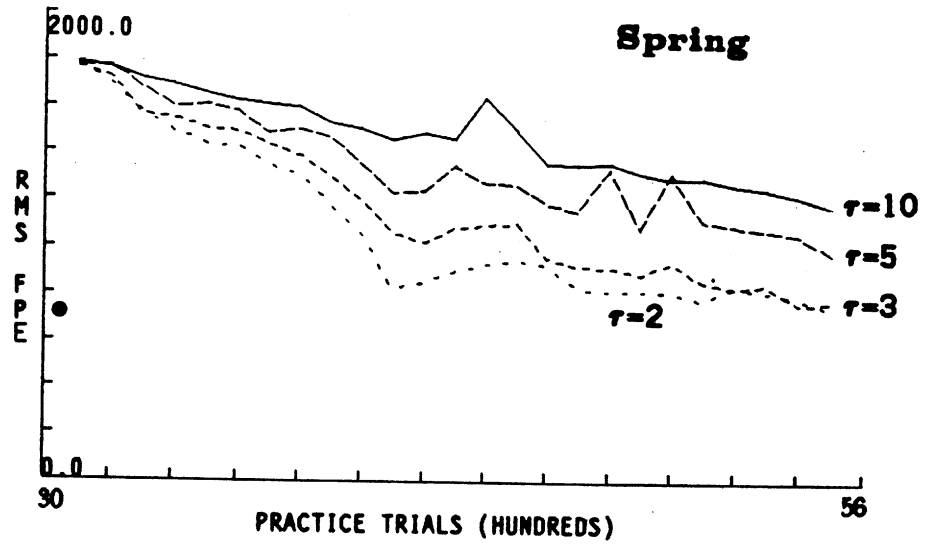


Fig. 1.5 The memory's time-constant is systematically varied. Smaller values of τ yield more rapid, but *noisier* adaptations. A) inertial load; B) spring load; (prototype PR-11). Closed circles indicate pre-adaptation levels.

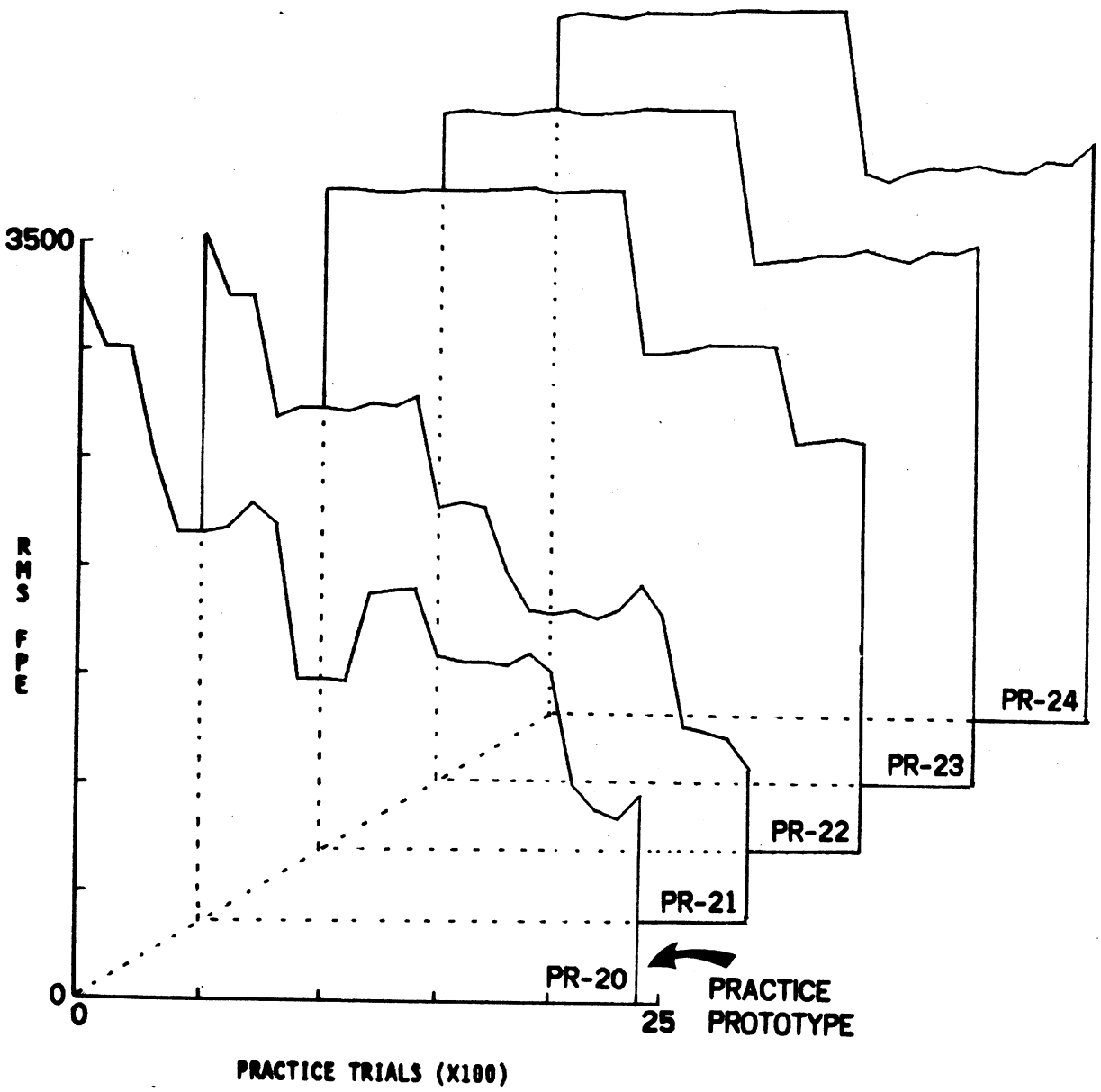


Fig. 1.6a Five learning curves that show generalization when prototype PR-20 was practiced and prototypes PR-20, PR-21, PR-22, PR-23, and PR-24 were tested. These prototypes share a common ending position and duration, but vary systematically in starting position; ((.285m,-.145m,.12m), (.265,-.145,.1), (.245,-.145,.3), (.245,-.165,.6), (.245,-.185,.4), respectively).

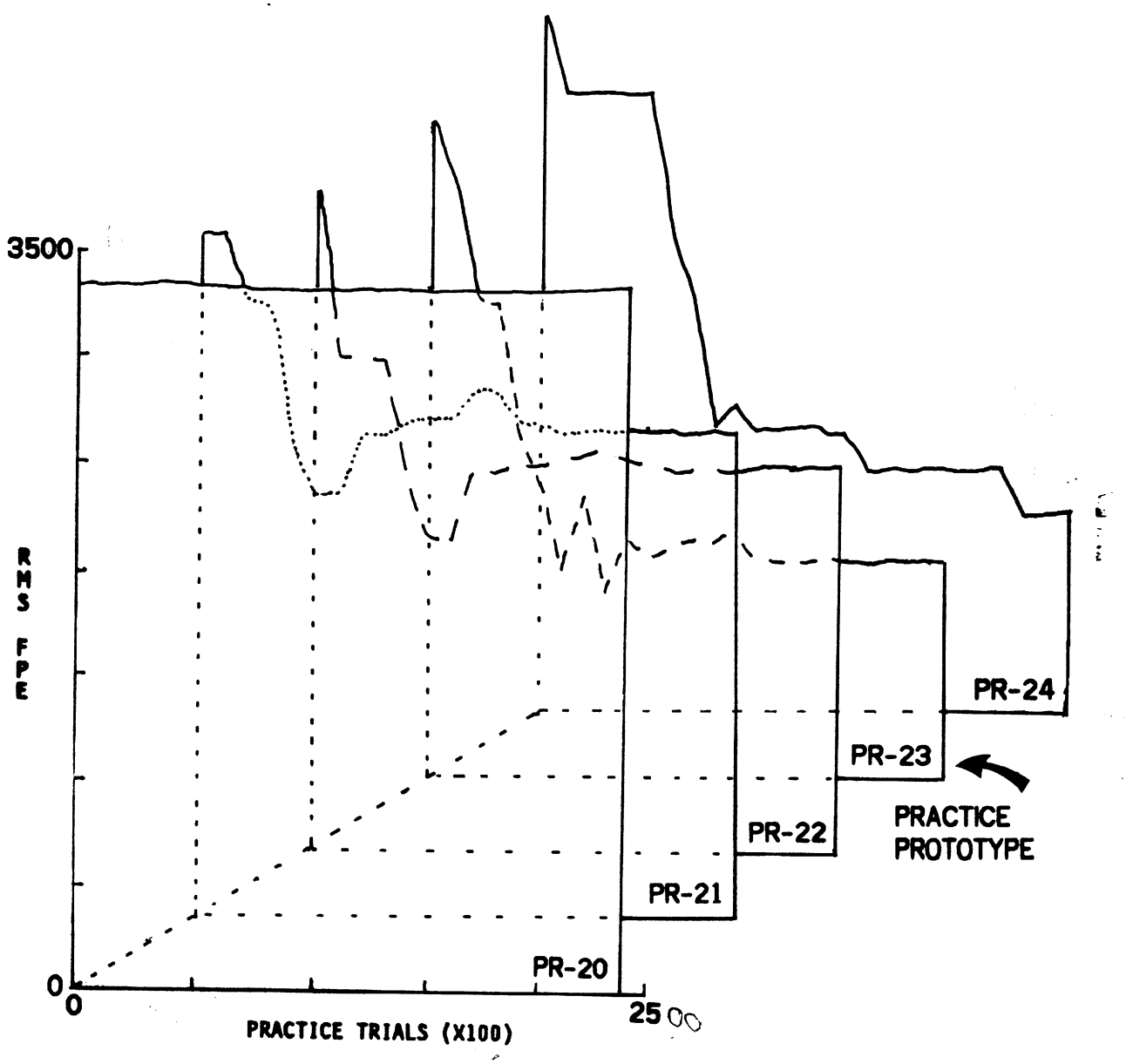


Fig. 1.6b Learning curves generated when prototype PR-23 was practiced and the entire set, PR-20, PR-21, PR-22, PR-23, and PR-24 was tested.

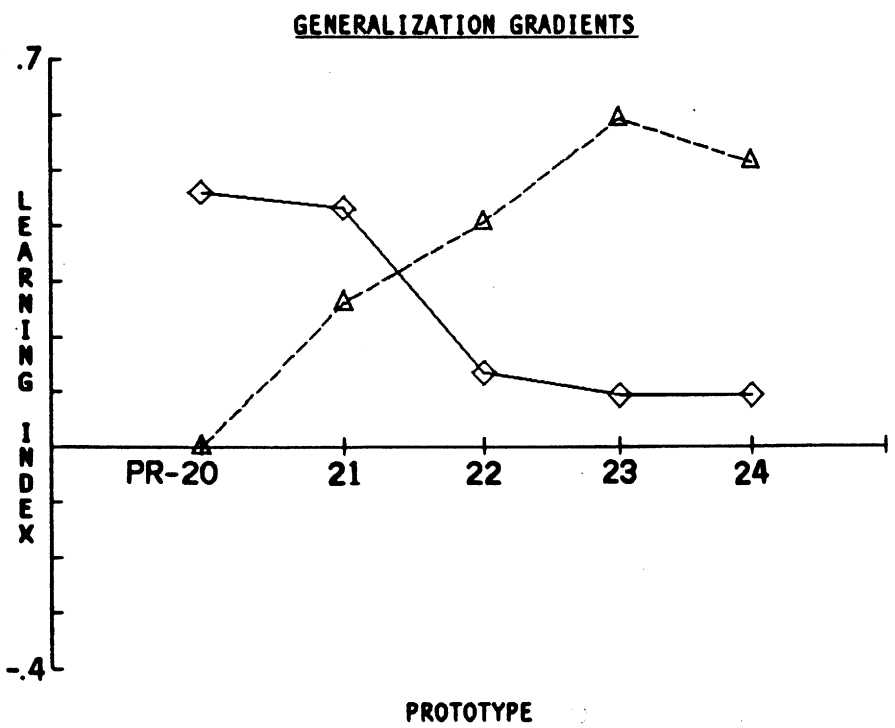


Fig. 1.7 Generalization curves summarizing the data of Fig. 5 are shown. Diamonds) Prototype PR-20 was practiced. Triangles) Prototype PR-23 was practiced.

3) Practice of one movement improves performance of others, provided they are similar to the practice prototype.

4) Control of the arm is maintained or reattained, despite changes in its mechanical properties.

1.3.6 Discussion

Examination of Eq. 1.4 reveals that $\ddot{\theta}_D$ is absent from the computation of J and K, the constants of mechanical description. Without knowledge of the desired response an error signal cannot be computed. The present system is able to learn without error information and is, therefore, somewhat unique among models for control. Systems that do use error correction rely on the signed magnitude and sometimes the derivatives of error in selecting the next, and hopefully, *better* command (Fu, 1971; Tsypkin, 1971). Unfortunately, local error data are not always useful in finding global maxima that correspond to *best* commands, and *hill-climbing* problems may result. The parametric equations are so simple, however, that a search procedure is not required for solution. Application of Eq. 1.4 only requires that N+1 independent measurements be available for the same hyper-region.

A mechanisms has been described that pre-computes a set of motor commands which are executed in the absence of feedback. Few practical applications, (biological applications included), can tolerate the imprecision of such open-loop operation, yet the problems of motor planning can probably be best developed in this type of isolation. Ultimately it will be necessary to find a compromise between pre-planning and servo control, and the compromise will yield dividends: The same data that are so useful in planning will facilitate on-line error correction, both processes benefitting from

experience. For the sake of clarity of results and presentation, however, consideration of the *plan+servo* approach has been postponed.

Young and Stark (1965) and others have proposed the use of an *internal dynamic model* to allow learning and adaptation. Their idea is that information describing the response of the plant to commands can be used to adjust an internal dynamic model that will be used in future selection of commands. Although this idea can be made to work, another concept which represents a different point of view is stressed here -- the *internal inverse dynamic model* (Paul, 1972; Waters, 1974). The idea of the inverse is that the a motor learning system should have a transfer function which converts responses into commands -- the inverse of the operation performed by the mechanical device. When the inverse and the device are operated in cascade the transfer function is the identity matrix -- the desired result. The internal dynamic model allows simulation of the inverse function with an approach similar to *analysis by synthesis* (Eden, 1962). Because it uses sets of extremely simple equations to describe the plant's behavior, however, the present model calculates the required inverse functions directly, for each region of space.

1.4 Analytical Equations vs. Table Look-up for Manipulation: A Unifying Concept

1.4.1 Introduction

Solution of the complete equations of motion for a serial link manipulator is computationally quite expensive and is usually not possible in real-time. Even off-line calculations frequently require simplifying approximations. Stanford's hand-eye project uses such simplifications in their approach to arm control (Paul, 1972). They begin with

the complete set of dynamic equations, and ignore inertial coupling and velocity interactions between joints. This allows them to pre-compute torque trajectories in a reasonable amount of time.

An alternative approach is to trade large amounts of computation for large amounts of storage. Albus (1975) proposed a controller, CMAC, which reduces the computational burden at the expense of a large hash-coded memory. CMAC in its simplest and most useful form may be characterized by: 1) elimination of complicated real-time computations, 2) a very large memory containing data descriptive of the arm's mechanical nature, and 3) a simple procedure for acquiring the memory's data from practice, without having an analytical model of the manipulator.

Though Albus' pure table look-up and Stanford's approximate analytical approaches are quite different, through introduction of some ideas on the nature of *parameterized equations* these apparently divergent approaches to the control problem can be brought together under one conceptual roof. Furthermore, a number of computationally intermediate formulations that had not been identified previously, have been isolated for study.

The main point of this paper is that the equations of motion for a manipulator can be dramatically simplified if a subset of the independent variables are treated as parameters. In order for the parameterized equations to be useful, however, simplified forms must be available for a large set of parametric values. Therefore, one complicated equation is traded for a set of simpler ones. (Of course, only one of the simplified equations need be evaluated for a given situation.) The particular choice of

variables submitted to parameterization determines the balance between computational complexity and storage. As a corollary effect of simplification, parametric forms of the equations of motion can be used to acquire information about the mechanics of the controlled device.

1.4.2 Parameterization

Consider the equation:

$$y = f(X) \qquad \text{(eq. 1.5)}$$

where:
X is a vector of the function's independent variables.
(Capital letters denote vectors.)

The value of y can be found by specifying values for the independent variables of the equation. As the independent variables change, the value of y will vary as dictated by the functional relationship, f.

In many cases, making one or more of the variables in an equation a parameter, that is, holding it equal to a known constant, will greatly simplify the functional relationship. For example:

$$y = f(x_1,x_2) = x_1 + x_2 + x_1x_2 + x_1^2 \qquad \text{(eq. 1.6a)}$$

If: $x_1=1$

then: $y|_{(x_1=1)} = f_{(x_1=1)}(x_2) = 2 + 2x_2 \qquad \text{(eq. 1.6b)}$

Of course, the original relationship is not expressed here unless the function $f_{x_1}(x_2)$ is available for every x_1 of interest. In general a function is simplified by taking some subset X_p, of X as parameters and recasting the functional relationship in terms of the remaining independent variables, X_v:

$$y|_{X_p} = f_{X_p}(X_v) \qquad \text{(eq. 1.7)}$$

where:

$X = [X_p; X_v]'$

X_p are the variables held constant, the parametric variables.

X_v are the remaining independent variables.

' indicates transpose.

1.4.3 Parameterizing Equations of Motion

For a mechanical arm the problem at hand is to find the functional dependence of motor torque on position, velocity and acceleration. Generally, if the mechanical system has N degrees of freedom it is described by a non-linear vector function, Γ, with independent variables θ, $\dot{\theta}$, and $\ddot{\theta}$, each an N-vector: The generic equations of motion for the general manipulator can be written as:

$$T = \Gamma(\theta, \dot{\theta}, \ddot{\theta}) \qquad \text{(eq. 1.8)}$$

where:

T is the motor torque vector.

θ, $\dot{\theta}$, $\ddot{\theta}$ are the position, velocity and acceleration vectors.

Applying the parameterization procedures introduced above, a subset of the independent variables are assigned values and become parameters of the equations. When the equations are evaluated, substituting the parametric values for the parametric variables, simplified expressions result. For example, if $\theta = K$, then θ is a parametric variable, and the expression for torque is denoted: $T|_{(\theta=K)} = \Gamma_{\theta}(\dot{\theta}, \ddot{\theta})$.

Let P be an indicator equal to the number of variables in X_p. By varying P, $0<P<3N$, a continuum is defined along which the character of the equations of motion gradually changes from the completely analytical form (P=0), to a form where all variables are parameters and no analytical expressions exist (P=3N):

$$T = \Gamma(\theta, \dot{\theta}, \ddot{\theta}) \rightarrow \Gamma_{X_p}(X_v) \rightarrow \Gamma_{X_p} \qquad \text{(eq. 1.9)}$$

For P=0 the equations are analytical. Only one equation need be evaluated for every possible combination of independent variables. For P=3N a *different equation* must be evaluated for each combination of independent variables. (For P=3N the equations are all constant: $\Gamma_X = K$.)

1.4.4 P = 0, N, 2N, 3N

$\Gamma(\theta, \dot{\theta}, \ddot{\theta})$ is simplified by defining X_p as a subset of the independent variables, and X_v as the remaining independent variables. In principle, the selection of the independent variables for the two subsets, X_p and X_v, can be made with complete freedom. P may assume any integer value between 0 and 3N. While many partitionings of the independent variables may lead to simplified expressions, the present discussion is limited to the important class of subdivisions that result when all components of an independent N-vectors, θ, $\dot{\theta}$, or $\ddot{\theta}$ are assigned to one subset or the other. For example:

If: $\theta_i \in \{X_p\}$ holds for any i (eq. 1.10)

then: $\theta_i \in \{X_p\}$ holds for all i, $1 \le i \le N$

Therefore, the points on the P-continuum of interest are P=0, N, 2N, 3N. The functional forms of the parameterized equations of motion for these values of P are given in columns 1 and 2 of Table 1.1.

Table 1.1

P	Expression	No. of Eq.	$\lambda=\mu=\nu=M$
0	$T = \Gamma(\theta,\dot{\theta},\ddot{\theta})$	1	1
N	$T = \Gamma_{\theta}(\dot{\theta},\ddot{\theta})$	λ^N	M^N
	$T = \Gamma_{\dot{\theta}}(\theta,\ddot{\theta})$	μ^N	M^N
	$T = \Gamma_{\ddot{\theta}}(\theta,\dot{\theta})$	ν^N	M^N
2N	$T = \Gamma_{\theta,\dot{\theta}}(\ddot{\theta})$	$\lambda^N\mu^N$	M^{2N}
	$T = \Gamma_{\theta,\ddot{\theta}}(\dot{\theta})$	$\lambda^N\nu^N$	M^{2N}
	$T = \Gamma_{\dot{\theta},\ddot{\theta}}(\theta)$	$\mu^N\nu^N$	M^{2N}
3N	$T = \Gamma_{\theta,\dot{\theta},\ddot{\theta}}$	$\lambda^N\mu^N\nu^N$	M^{3N}

Selection of P still does not always completely constrain the form of the equations of motion, since the selection of variables to serve as parameters is not specified. The equations which represent each extreme of the P-continuum, P=0 and P=3N, are of unique form. (This would be so even if the restriction described by Eq. 1.10 were not in force.) The intermediate cases, P=N and P=2N, however, are each characterized by three alternate forms. (See Table 1.1, columns 1 and 2.)

The number of equations required to represent any one of the forms listed in Table 1.1 can be determined by assuming that each independent variable takes on a discrete set of values. Since representation of an equation by a finite set of parameterized equations is only an approximation, the number of values required for each variable must be determined in context of the application. The number of parametric values can be chosen to produce an approximation of any desired accuracy. Let λ be the number of values assumable by each component of θ, μ by each component of $\dot{\theta}$, and ν by each component of $\ddot{\theta}$. The number of equations needed to

represent $\Gamma_{X_P}(X_V)$ is given in Table 1.1, column 3. If all parameters are limited to the same number of values, M, ($\lambda=\mu=\nu=M$), approximately M^P equations are required; Table 1.1, column 4.

While six forms of $\Gamma_{X_P}(X_V)$ are possible for P=N, 2N, the goal of reducing complexity of the equations at the expense of storage, (ie. more equations), is not served equally by each possibility. In order to estimate the cost of each form, a count is taken of the number of adds, multiplies, and trigonometric function evaluations required for each form. The amount of storage required per cell, and the number of cells (assuming $\lambda=\mu=\nu=M$) are also counted. Table 1.2 shows very rough estimates for these costs.

Table 1.2

P	Form	Mults	Trigs.	Cell size	Cells
0	$T = \Gamma(\theta,\dot{\theta},\ddot{\theta})$	$>N^{4*}$	2N	$>N^{4*}$	1
N	$T = \Gamma_{\theta}(\theta,\ddot{\theta})$	$2N^2+N^3$	0	$(N+N^2+N^3)$	M^N
	$T = \Gamma_{\dot{\theta}}(\dot{\theta},\ddot{\theta})$	$>N^{4*}$	2N	$>N^{4*}$	M^N
	$T = \Gamma_{\ddot{\theta}}(\theta,\dot{\theta})$	$>N^{4*}$	2N	$>N^{4*}$	M^N
2N	$T = \Gamma_{\theta,\dot{\theta}}(\ddot{\theta})$	N^2	0	$N+N^2$	M^{2N}
	$T = \Gamma_{\theta,\ddot{\theta}}(\dot{\theta})$	N^2+N^3	0	$N+N^3$	M^{2N}
	$T = \Gamma_{\dot{\theta},\ddot{\theta}}(\theta)$	$>N^{4*}$	2N	$>N^{4*}$	M^{2N}
3N	$T = \Gamma_{\theta,\dot{\theta},\ddot{\theta}}$	0	0	N	M^{3N}

*Very difficult to estimate for the general manipulator.

It is clear from Table 1.2 that θ must be a parametric variable to yield computational efficiency. One can understand this result by examining the mechanical nature of manipulators comprised of coupled, serial degrees of freedom. In general,

elements of the moment of inertia tensor, (both off diagonal and on diagonal terms), gravitational terms, and Coriolis terms are sums of products of trigonometric functions of angles between links, and sums of these angles (Kahn, 1969). Only when θ is made a parameter do these computations become unnecessary.

1.4.5 State Space Model (SSM)

The SSM, described more fully by Raibert (1976; 1977), results when P=2N and θ and $\dot{\theta}$ are parametric variables ($X_P=[\theta : \dot{\theta}]'$):

$$T = \Gamma_{\theta,\dot{\theta}}(\ddot{\theta}) \qquad \text{(eq. 1.11)}$$

When each of the terms contributing to the torque acting on the joints of an arm are included, Newton's equation for rotary motion may be expressed schematically as:

$$T - G(\theta) - B(\dot{\theta}) - C(\theta,\dot{\theta}) = J(\theta)\ddot{\theta} \qquad \text{(eq. 1.12)}$$

where:
- T is the actuator torque vector
- G is a vector-function for gravitational torque
- B is a vector-function for frictional torque
- C is a vector-function for Coriolis torque
- J is a matrix-function for moment of inertia

(Remember, the full set of time-varying, non-linear equations with explicit expression of θ- and $\dot{\theta}$-dependencies are not shown here.) Parameterizing the state variables, θ and $\dot{\theta}$:

$$T - G|_{(\theta=\alpha)} - B|_{(\dot{\theta}=\beta)} - C|_{(\theta=\alpha,\dot{\theta}=\beta)} = J|_{(\theta=\alpha)}\ddot{\theta} \qquad \text{(eq. 1.13a)}$$

where:
- α parametric position vector
- β parametric velocity vector

Or, more compactly:

$$T - G_\alpha - B_\beta - C_{\alpha\beta} = J_\alpha \ddot{\theta} \qquad \text{(eq. 1.13b)}$$

Here, each of the vector-function relationships $G(\theta)$, $B(\dot{\theta})$, $C(\theta,\dot{\theta})$, and $J(\theta)$ has become a parametric set of constants. By grouping terms and making the equation explicit in motor torque one further simplification can be made:

$$T = J_\alpha \ddot{\theta} + K_{\alpha\beta} \qquad \text{(eq. 1.14)}$$

where:
$K_{\alpha\beta} = G_\alpha + B_\beta + C_{\alpha\beta}$

An important property of this formulation is that values for J_α and $K_{\alpha\beta}$, the variables that characterize the equations, can easily be computed from input-output data obtained during motions of the manipulator:

$$J = \Upsilon \cdot \Theta^{-1} \qquad \text{(eq. 1.15a)}$$
$$K = T_{av} - [\Upsilon \cdot \Theta^{-1}] \cdot \ddot{\theta}_{av} \qquad \text{(eq. 1.15b)}$$

where:
$\Upsilon = [T_1 \vdots T_2 \vdots \ldots \vdots T_N] - [T_{N+1} \vdots T_{N+1} \vdots \ldots \vdots T_{N+1}]$
$\Theta = [\ddot{\theta}_1 \vdots \ddot{\theta}_2 \vdots \ldots \vdots \ddot{\theta}_N] - [\ddot{\theta}_{N+1} \vdots \ddot{\theta}_{N+1} \vdots \ldots \vdots \ddot{\theta}_{N+1}]$.
T_i and $\ddot{\theta}_i$ are the i'th measurements of T and $\ddot{\theta}$.
X_{av} denotes the average: $(X_1+X_2+ \ldots +X_{N+1})/(N+1)$.

In general, as a corollary effect of simplification, forms of equations near the P=0 end of the P-continuum are easily invertable and learning is facilitated (Raibert, 1976).

1.4.6 Configuration Space Method (CSM)

CSM is obtained when P=N and θ is the parametric variable ($X_p=\theta$):

$$T = \Gamma_\theta(\dot{\theta},\ddot{\theta}) \qquad \text{(eq. 1.16)}$$

Since velocities are not parameterized they still appear as arguments in the functional expression for motor torque. The equations of motion become:

$$T_m - G_\alpha - B(\dot{\theta}) - C_\alpha(\dot{\theta}) = J_\alpha(\ddot{\theta}) \qquad \text{(eq. 1.17)}$$

Neglecting friction and making the dependency on velocity explicit:

$$T_m - G_\alpha - [\dot{\theta}'C_{1,\alpha}\dot{\theta} \vdots \dot{\theta}'C_{2,\alpha}\dot{\theta} \vdots \ldots \vdots \dot{\theta}'C_{N,\alpha}\dot{\theta}]' = J_\alpha(\ddot{\theta}) \qquad \text{(eq. 1.18)}$$

where:
$C_{i,\alpha}$ is the Coriolis matrix for joint i evaluated at state α.
Note: The Coriolis force is determined by a vector of terms each quadratic in $\dot{\theta}$ (Bejczy, 1974).

Many control schemes used in practice ignore inertial interactions between joints and Coriolis forces, yet these terms can be important during high velocity motions. This compromise has important implications for industrial applications where the throughput of a manipulation process depends on the arm's speed. The importance of CSM is that these terms can be included at low computational cost with *reasonable* amounts of memory. Real-time trajectory calculations may also be possible, especially in the context of a distributed computation employing multiple microprocessors (Raibert & Horn, 1977).

CSM is primarily of interest to those interested in robotics, though the approach is not out of the question for explanation of nervous function. The disadvantage for biological systems is that the inversion operations required for learning are much more complicated than those characterizing P=2N and P=3N systems. For robotic applications, however, configuration space data can be calculated in advance, based on an analytic model of the mechanical device. This will be a one-time calculation for each manipulator. Compared to SSM and CMAC, (which theoretically also do not ignore inertial coupling and Coriolis forces), this approach offers the advantage of reduced storage requirements at the expense of increased run-time computation. While the

SSM requires about $2N^2$ operations for evaluation and $(M^{2N})(N+N^2)$ memory locations, CSM requires about $3N^2+2N^3$ operations for evaluation and $(M^N)(N+N^2+N^3)$ memory locations.

The relationships among the parameterized systems discussed are summarized in Table 1.3, where computational and storage costs are ranked.

Table 1.3

P	Form	Rank Storage Cost	Rank Comp. Cost	Approach
0	$T = \Gamma(\theta,\dot{\theta},\ddot{\theta})$	1	4	Analytic Eq. [4]
N	$T = \Gamma_{\theta}(\dot{\theta},\ddot{\theta})$	2	3	CSM [7]
2N	$T = \Gamma_{\theta,\dot{\theta}}(\ddot{\theta})$	3	2	SSM [5,6]
	$T = \Gamma_{\theta,\ddot{\theta}}(\dot{\theta})$			Not yet studied.
3N	$T = \Gamma_{\theta,\dot{\theta},\ddot{\theta}}$	4	1	CMAC [1]

2 The Problem

2.1 Constraining the Issues

2.1.1 Many Factors at Work

An often neglected first step in the study of the motor system is the selection of a class of behavior upon which attention may focus. The human body is a versatile mechanism capable of a staggering variety of movement. Its nervous system is also extremely versatile and employs a number of control strategies. Sometimes a limb is moved with great deliberation and precision using simultaneous activation of agonist and antagonist muscles. Other times more free-flowing motions are made in which agonist muscles accelerate the masses of a limb to high velocity, after which it coasts until slowed and stopped by the force of antagonist muscles (Kelley, 1968). Contrast, for example, the motion of a delicate paint stroke to that of a baseball pitch. Sometimes interaction with the environment is quite predictable and adjustments are virtually unnecessary while at other times the movement is nothing but a set of constant adjustments to external disturbances. When walking down a flight of stairs the position of each step and moment of foot contact is quite predictable. Conversely, standing still in a moving trolley car requires major adjustments at each lurch on the track. The aim of a movement may be to achieve a certain position, to move at a certain velocity, or to contact an object with a certain force. When one presses a button the position of the finger is important, while the velocity at which the violinist draws his bow influences the sounds which result. Imagine the effects of a masseur who cannot

regulate the force of his ministrations.

For each of these types of movement the problems of control are different and one reasonably homogeneous solution would not be expected to apply to every case. In order to develop a model that can help us to better understand movement we begin by acknowledging this diversity, and restrict study to a class of movement which is produced by one, unified control scheme or strategy. Of course, our readiness to make this choice indicates our belief in a certain discreteness of control function.

2.1.2 Emphasis on Pre-planning

After two decades of near fanatical interest in and devotedness to servo control and feedback mechanisms, physiologists and control theorists, motivated both by experimental findings (Evarts et al., 1970) and a desire to deal with non-linear, time varying devices (Schultz & Melsa, 1967; Bryson & Ho, 1969) have begun to look at the merits of *pre-planning*.

Servo theorists correctly state that no controller can predict the disturbances a mechanical device is likely to encounter in the real world. Even the parameters of the mechanical device cannot be known exactly. Therefore one must, if one wants accurate control, use feedback to assess and correct errors produced during operation. Unfortunately, servo advocates often stop there. However, there are a number of basic problems with servo-mechanisms for controlling arms. Briefly:

1) Simple servos are not designed to work with non-linear mechanical devices. They can be made to work, but the degree of success usually depends on the degree to which the system can be modelled as linear (Townsend, 1970).

2) When multiple degrees of freedom are involved, the simple servo usually cannot guarantee that all joints will pass through particular points at the same time. The usual methods of avoiding this problem result in jerky movements or relaxation of the accuracy with which the trajectory can be followed.

3) When errors are expressed in a coordinate system other than joint coordinates corrections cannot be generated without a transformation. For example a 1mm error in any one direction requires correction at a number joints, but the degree of involvement at each joint is not constant. For this reason, a servo controller without the ability to transform coordinates cannot correct for visually ascertained errors. (This objection may be somewhat unfair, since pre-planning does nothing to correct the problem.)

4) Servo controllers often require errors for the continued production of control signals.

Feedback is very important, but unable by itself to achieve the kind of flexible, accurate trajectory control we are after. More recent work has stressed the use of feedback with an open-loop plan. The approach is to pre-compute a set of commands which will drive the non-linear plant along the desired path in the absence of disturbances. Since this computation is done off-line there is time to use complicated analytical models of the plant, usually in the form of equations of motion. Furthermore, it only has to be done once for each trajectory since the planned commands are stored. During execution the commands are strobed from the memory and issued to the plant. A simple servo controller may be used in addition, to correct for residual errors (Bryson & Ho, 1969). The advantage of this scheme is that the servo is only responsible for producing control signals which will compensate for small deviations from the desired trajectory -- deviations for which behavior of the system is usually nearly linear.

Physiologists are also thinking along these lines. Now that the notion of the simple follow-up length servo for muscle control, (Merton, 1972; Marsden et al., 1972; 1976), has been substantially discredited (Severin et al., 1967; Murphy, 1975), ideas about pre-planned movement are gaining ground. (Ideas about preplanning have been around for quite a while (Lashley, 1951), but the servo story has obscured their impact.) Especially important are papers by Hammond (1956) and Melvill Jones & Watt (1971). They show that substantial corrective responses to mechanical disturbances are produced with a latency of about 120 msec. This does not imply that feedback is not used, but substantial computation is possible between the time such a disturbance is sensed, and the corrective action is taken.

One idea is that a new plan, one suited to the disturbance and designed to return the arm to the desired trajectory is formulated between disturbance and response. Rather than using a pre-plan or error correcting servo, this scheme advocates *reprogramming*. A variation of the open-loop movement is used that is composed of a number of short, open-loop segments executed in sequence. The sensory information produced during one segment only influences production of subsequent segments. This variation, dealt with in passing in this thesis, is mentioned in order to suggest the ultimate usefulness of solutions to the open-loop control and learning problems.

This thesis is aimed at the problems of producing a usable, pre-computed plan. Though *plan-plus-servo* is probably the only practical approach to these control problems (practical for biological as well as man-made systems), for the sake of clarity of results and crispness of presentation the work done here excluded the use of

servos and feedback. Once a movement is initiated no sensory information is used to alter that movement during its execution. Such movements are open-loop, but it should be realized that, ultimately, the loop is closed -- though the sensory information obtained during a movement is not used to alter the progress of that movement, it may be used to alter the motor system in such a way that subsequent movements are affected.

2.1.3 High- and Low-Level Specialization

The model under consideration here was not designed to account for all motor function. In addition to restricting the class of movements under study, we have limited the type of processing to be described. This means that other motor processors work along with the sub-system described here and the learning or execution of even a single movement probably relies on a number of processing elements. Figure 2.1 is an example of a familiar demonstration. Each of the orthographic strings shown in this figure are very similar, but the mechanical systems used to produce them, and therefore the motor commands, were quite different. Unless the subject learned to produce each form of output separately, (this was not the case in the example shown) we may draw two conclusions:

1) Motor plans exist in the nervous system which are expressed in a language which is independent of muscular and kinematic considerations. One such plan can be used to produce movements in any of a number of limbs or body parts.

2) Mechanisms exist in the nervous system which can translate general motor programs (as described in 1) into explicit instructions suitable for the muscles, mechanics, and sensors of a particular limb.

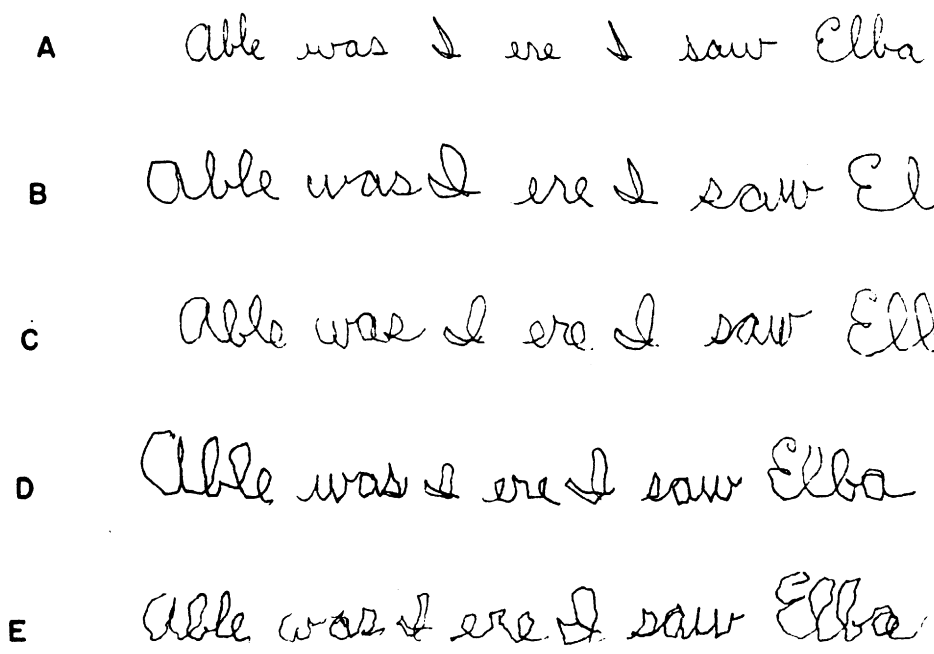


Fig. 2.1 Each of these orthographic strings are quite similar though different muscle and skeletal systems were used to produce each. The pen was moved by A) right hand, B) right arm, C) left hand, D) mouth (gripped in teeth), and E) right foot (taped to foot). The subject had essentially no previous experience writing with any body part other than A. A division of function into *high*- and *low*-level processes is suggested.

This type of architectural arrangement and the translation process have been discussed in the literature (Marr, 1969; Gelfand et al., 1971; Arbib, 1972; Waters, 1974). The power of such an arrangement is quite attractive. High level processors may formulate new movements or modify and combine old ones without having to take the mechanical properties of the effectors into consideration. It is supposed that these processors may perform symbolic operations through which planning and strategy decisions may also be made. They specify to the translator what the output of the limb should be.

The translating mechanism, on the other hand, is not organized around motor programs, but around the muscular, mechanical, and sensory systems with which it communicates. It is free from the responsibilities of strategy and planning, and need not be capable of performing symbolic operations. Its only duty is to accept detailed descriptions of movements and translate them into appropriate muscular commands. But to perform this function information about the kinematic, dynamic, sensory, and muscular properties of the limb must be available in a usable form. This information may not be present in the infant, and certainly must change as the organism grows. Effective translation therefore requires maintenance of an up-to-date source of limb-specific mechanical information.

The translating mechanism which converts descriptions of desired output into motor commands plus the support mechanism which acquires mechanical information and stores it in a usable form are the topics of interest in this paper and will be referred to collectively as the translator. The terms controller and translator are used

interchangeably throughout this paper.

2.2 Mechanical Problems

What mechanical problems must the low level system face? The human motor system deals with the mechanical nature of the skeletal and muscular systems, and the laws of physics which they must obey (Meriam, 1966). The forces and torques created by a muscle often influence a number of joints, even when the muscle is of the simple, single joint variety. Each joint is influenced by a number of muscles, not only because there are many muscles *across* the joint, but because reaction torques are produced when muscles accelerate other joints of the body. The degree and pattern of interaction is not constant, but depends on the limb's position. Yet our nervous control system effectively compensates for these mechanical interactions when precise movements are called for. Here we examine the nature of each of these factors in a somewhat conceptual way by developing a general form of the equations of motion.

2.2.1 Equations of Motion for a Limb

Let us begin by laying out the computations required by the translation process. Descriptions of movements must be converted into motor commands. The acceleration of an object, taken with its initial conditions, gives a complete description of its movement and the force on an object is that which commands its every motion. For this simple, unconstrained system we can specify the desired acceleration and use Newton's equation, $F = Ma$, as a translator to find the necessary force. Of course, this also applies to rotary motion, $T = J\cdot\ddot{\theta}$, where T is the torque, J is the moment of

inertia, and $\ddot{\theta}$ is the angular acceleration. If only a limb were as simple as that.

What is the acceleration of a limb? If we take the simple case where the coordinate system of interest is that of the limb's joints we can describe the acceleration by an N-vector, $\ddot{\theta}$, N being the number of joints or degrees of freedom. Newton's equation still applies, but the torque is now an N-vector and the moment of inertia must be expressed as a square matrix of rank N. This matrix, J, specifies the relationship between the torques and the resulting accelerations at each joint in the limb; j_{ik} = (torque applied to joint k) / (acceleration at joint i). Unfortunately, the elements of J, though dependent on the masses of the links, are not constant. They vary during each movement. (Note: Here we are not talking about the gradual changes in limb mass or geometry caused by growth.)

The off-diagonal elements of J, j_{ik} for $i \neq k$, represent inertial interactions between joints. The amount of interaction between two joints also depends on the position vector, θ. For some configurations of the limb interactions are pronounced, while for others they are small. Figure 2.2 illustrates this point. In Fig. 2.2a the geometry of the joints are arranged to allow large reactions at joint 2 for torques applied to joint 1. In Fig. 2.2b the magnitude of interaction is greatly reduced. On-diagonal elements of J, j_{ik} for $i=k$, represent the moments of inertia for each joint. They too vary with configuration. The effective moment of inertia of a joint is determined, not only by the masses of the links which are moved, (a link is that part of a limb between two joints) but also by the distances between the masses and the center of rotation. In Fig. 2.2c the moment arm, and therefore moment of inertia, are large, while in Fig. 2.2d they are

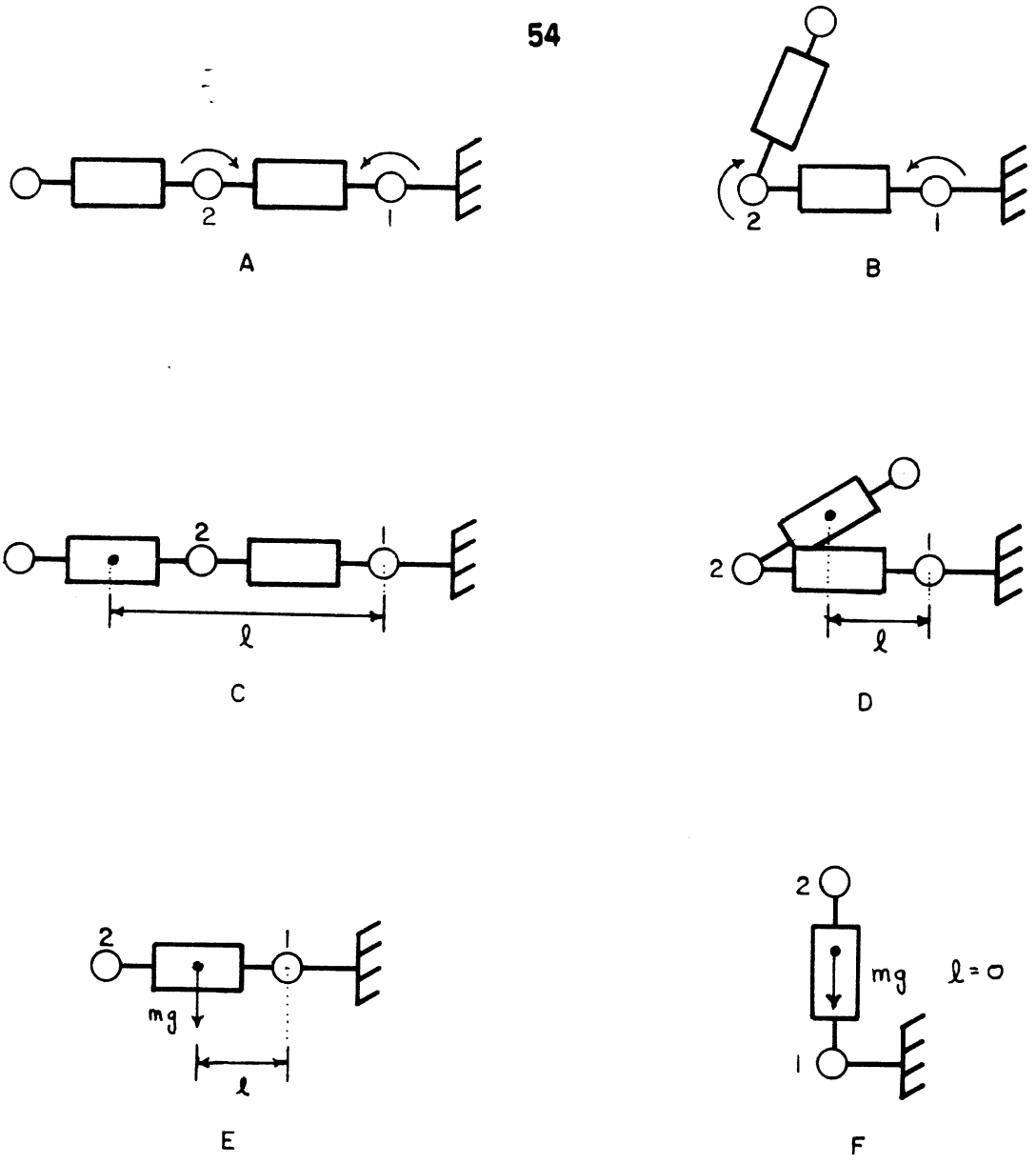


Fig. 2.2 Schematic diagrams of a limb that illustrate configuration dependent properties. Acceleration of joint 1 will cause a larger reaction torque about joint 2 in A than B due to the difference in position of joint 2. In C) the moment of inertia of joint 1 is maximum because the center of mass of link 2 is far from the center of rotation. D) Here the moment of inertia is almost minimized. The gravitational torque depends on the moment arm through which gravity acts. In E it is maximum, but no torque is produced in F.

small.

The torque shown in Newton's equation is only the *net* torque acting on the joints. In addition to the torques applied by the muscle, (or motor), we must consider the acceleration of gravity and the damping forces due to friction. The acceleration of gravity must be represented by an N-vector because each mass in the limb will be accelerated individually. As is true of the inertial terms, the gravity factors also depend on configuration. Since the moments through which gravity acts varies with configuration, so do the gravitational torques. (See Figs. 2.2e and 2.2f.)

Frictional torque, also an N-dimensional vector, is independent of θ, but depends on velocity of the moving joint, $\dot{\theta}$. The friction function can be expanded into a number of simple terms, none of which depend on variables related to other joints:

$$b_i(\dot{\theta}_i) = b_{v,i}\dot{\theta}_i^2 - b_{c,i}\mathrm{sgn}(\dot{\theta}_i) + b_{s,i}\delta'_{-1}(\dot{\theta}_i) \qquad \text{(eq. 2.1)}$$

where:
- $b_{v,i}$ is the i'th viscous friction term.
- $b_{c,i}$ is the i'th coulomb friction term.
- $b_{s,i}$ is the i'th stiction term. (δ'_{-1} is similar to a doublet.)

Unlike the gravitational and moment of inertia terms, which may be calculated from a blueprint of the limb, the frictional terms depend on factors which usually cannot be predicted by analysis, but must be measured. These factors are summarized by a single net friction term, $B(\dot{\theta})$, in the equations below.

A final factor relevant to the equations of motion that only introduces appreciable torques at high velocities, is the Coriolis term. This torque is produced by simultaneous rotation of an object about two orthogonal axes; the direction of its action is about a third axis orthogonal to the plane of the first two. The magnitudes of

Coriolis forces depend on the velocity and position vectors.

Standard formulations of equations of motion for limb-like devices do not include terms for elasticity. This is because most man-made arms have no elastic forces acting on the joints, or because such forces can be associated with the actuator's properties. Such terms are similar to gravity in that they depend only on configuration. For our purposes, therefore, the gravity function is augmented to include torques due to elastic elements.

Rewriting Newton's equation to include each term introduced above:

$$T_m - G(\theta) - B(\dot{\theta}) - C(\theta,\dot{\theta}) = J(\theta)\ddot{\theta} \qquad \text{(eq. 2.2)}$$

where:
T_m is the actuator torque vector
G is the augmented gravitational torque vector
B is the frictional torque vector
C is the Coriolis torque vector
J is the moment of inertia matrix
θ, $\dot{\theta}$, and $\ddot{\theta}$ are the position, velocity, and acceleration vectors.

This equation may still look manageable, but there is one more fly in the ointment that aggravates and accentuates the other problems. We have shown in a schematic way that some of the terms in Eq. 2.2 depend on positions of joint but have not worked out the exact relationships. While the schematic argument was simple and easy to understand intuitively, the evaluation of these factors in practice is extremely complicated (Peiper, 1968; Kahn, 1969; Bejczy, 1974). The problem is especially acute for a serial-link mechanical device because each set of interactions must take into account the geometry associated with interceding links. The system of coordinates determined by the joints are not orthogonal so there is a proliferation of residual

terms. Kahn (1969, see his appendix A) has worked out the explicit dependencies using a computer program which performs algebraic manipulations. His results for a general limb having three links and three joints (N = 3) but no friction are almost intractable, (the equations involve about 1600 terms and 13,000 multiplications) and virtually useless for a theory of motor function.

2.3 One Control Problem or Two?

At this point it is necessary to take a short digression so that an important question can be raised: "Is there one control problem that applies to biological and man-made limbs, or must there be two separate problems?" This thesis is based on the premise that for an important level of understanding there is only one.

There are differences. Human arms grow and are made of flesh and bone. Man-made arms are constructed from metals and plastics and must be bolted, welded or glued. But beyond these obvious, superficial differences lie important questions for which the similarities are more significant.

Are we being too simplistic when we characterize the inervation of a muscle in terms of a commanded force or position? It is easy to speak of motor commands when considering a manipulator. There is usually only one current or voltage for each actuator, and only one actuator working on a joint. Each muscle of a human arm, however, is enervated by thousands of motor neurons (Henneman, 1968), and several muscles simultaneously move a joint. Are we treating the elegance of the nervous system unfairly when we reduce the messages of thousands of receptors to the bare essentials and say they are signaling a joint's position, velocity, and acceleration? Just

because a typical manipulator may be equiped with a single position and velocity sensor for each joint are we to assume that only a single signal can be important? These questions cannot yet be answered. We can only promise to proceed with caution.

Despite these important differences, I feel that there is a level of approach to the arm control problem at which the biological and man-made arms are very similar indeed. Both are mechanical devices. The equations of motion developed in the last section apply equally well to the links, joints, and masses of either type of arm. Biological limbs are mechanical manipulators, and man-made manipulators are limbs. Though the particular configuration of parts, and materials used are different, to the extent we can talk about actuator commands and position, velocity and acceleration measurements, the control problems are identical.

One more point. There is an interesting parallel between the physiologists question, "What are the control variables?" and the engineer's question "Should we use torque motors, hydraulic positioning devices, or velocity servos?". Both groups, motivated by different goals are searching for answers to the same question. The final answer to this question is not yet known by either group. Evarts finds cells in motor cortex that seem to encode force (Evarts, 1973), and Bizzi argues that muscles understand final position commands (Bizzi et al., 1976; Polit & Bizzi, 1977). Meanwhile, Victor Scheinman designs arms that use DC torque motors and the Unimate is driven by hydraulic actuators under control of a velocity servo.

3 The State Space Model

In this section the elements of the State Space Model are presented along with a description of the system's operation and expected behavior.

3.1 The Forward Computation -- *Translation*

Before we totally discard the equations developed in Section 2.2.1, let us examine a special set of circumstances under which simplifications can be made. During a very short interval of time we observe that Eq. 2.2 still describes the behavior of the limb, but each term can be simplified. During a short interval, call it a time slice, or just a slice, we see that the position and velocity for each joint only change by small amounts. We may neglect these small changes or reduce the duration of the slice to the point where they may be neglected. Once this is done each element of a vector or matrix in Eq. 2.2 which had been dependent on the state of the system becomes a constant. (The state of the limb is uniquely determined by the positions and velocities of all the joints.) The simplified equations of motion can be represented as:

$$T_m - G|_{(\theta=\alpha)} - B|_{(\dot\theta=\beta)} - C|_{(\theta=\alpha,\dot\theta=\beta)} = J|_{(\theta=\alpha)}\ddot\theta \quad \text{(eq. 3.1a)}$$

where:
α is the position vector during the slice
β is the velocity vector during the slice

Or, more compactly:

$$T_m - G_\alpha - B_\beta - C_{\alpha\beta} = J_\alpha\ddot\theta \quad \text{(eq. 3.1b)}$$

By grouping terms and making the equation explicit in torque exerted by the muscle one further simplification can be made:

$$T_m = \bar{J}_\alpha \ddot{\theta} + K_{\alpha\beta} \qquad \text{(eq. 3.2)}$$

where:

$$K_{\alpha\beta} = G_\alpha + B_\beta + C_{\alpha\beta}$$

This is the *translation equation.* It must be remembered that this equation only applies to the motion of the limb during one time slice. Nothing prevents application of Eq. 3.2 to other time slices provided new values of J and K can be found (or are available) for the state of the system prevailing during those slices. Eq. 3.2 may be described as the piece-wise constant version of Eq. 2.2; the state for which the constants are chosen is the operating point. Although the development so far indicates that this equation calculates the torque needed at each joint of the limb, the value calculated can be the net force exerted by a muscle on the tendon, or a special version of the command to the muscle. (See Section 3.2.3)

Supposing the required constants are available, one can take the description of an entire movement, slice it up into enough time intervals so that the change in position and velocity for each joint is negligible, and determine the muscular torque needed to produce the desired acceleration for each interval. If the appropriate initial conditions are satisfied and each torque is applied for the duration of the interval for which it is computed, the resulting movement will closely resemble the originally specified movement. The error can be made arbitrarily small by reducing the duration of the time slices, provided the constants needed are available for each of these new, shorter slices.

This scheme will only work if the accelerations present in the description of the desired movement are limited in magnitude to those produceable by the limb's

actuators. Violation of this restriction will result in specification of a torque vector which is not achievable and the resulting motion, assuming that some attainable torque is used instead, will not conform to the desired response. It should be realized that this problem must be faced by any solution to the translating problem and is not unique to the solution given here.

3.2 The Inverse Computation -- *Learning*

The solution given so far is only a partial description of the computations performed by the translator since we have not yet indicated how the constants that describe the mechanical nature of the system are found, nor how they are affected by motor experience and changes in the mechanical system.

3.2.1 Historical Perspective

von Holst and Mittelstaed (von Holst, 1954; Mittelstaedt, 1958) developed a model designed to account for the fly's ability to descriminate between the sensory consequences of the fly's own self-produced movements, and externally produced movements. Their model used the relationship between an externally generated signal describing changes in sensory stimulation, *reafference*, and an internally generated signal describing impending changes in the position of the sensory surface, Helmholtz's *efference copy* (Helmholz, 1867), to provide unambiguous information about movement in the external world.

Held (1961), and Held & Hein (1962) extended the Holstian view to allow attainment of perceptual accuracy even after changes were made to the relationship

between activity in the external world, and the sensory stimulation produced by that activity. In this model the efference copy was used to "elicit the trace of previous reafference", which in turn was compared to the current afference.

Young and Stark (1965) proposed an elaborate model in order to account for human performance in a tracking task. They were not directly interested in how we control our limbs, but in how we use our limbs to control the external world. In their model the efference copy was used to drive an *internal dynamic* model of the controlled element, the output of which was compared with the afference from the control task.

In the State Space Model all learning centers around determination of the constants of mechanical description. The relationships between the efference copy and the reafference are used to determine these constants that describe the mechanical behavior of the limb to the translating mechanism. Reafference corresponds to measurements of the acceleration vector made from the limb's sensors, or other sensors that monitor the limb's activity. Efference copy is a record of the actuator torque vector, available from the source of motor commands or from actuator sensors. (See 3.2.3.)

This model makes rather unique use of efference copy in that no error signal is calculated, there is no comparator, and no error correction procedure is used for learning. Instead, the simplified form of the translation equation allows information about the mechanics of the limb to be found by examining the limb's input-output relations. Mechanical properties are derived directly from the results of the organism's

attempts to move. Furthermore, the State Space Model represents a rather direct example of Paul's and Waters' idea of an *internal inverse dynamic model* (Paul, 1972; Waters, 1974).

3.2.2 The Inversion Equation

We return, once more, to the simplified equations of motion which govern the system's behavior during a time slice, with the understanding that what must be found are the N^2+N constants which comprise each J and K. (Note: It is no longer necessary to distinguish between net torque, T, and actuator torque, T_m, so the subscript has been dropped: $T=T_m$. The α and β subscripts have also been dropped and should be assumed.) For the scalar equation, (N=1):

$$t = j\ddot{\theta} + k \qquad \text{(eq. 3.3)}$$

It is known that j and k can be found solving two simultaneous equations in two unknowns. Once measurements are made of the torque and acceleration for two movements j and k are calculated:

$$k = t_1 - \frac{t_1 - t_2}{\ddot{\theta}_1 - \ddot{\theta}_2} \cdot \ddot{\theta}_1 \qquad \text{(eq. 3.4a)}$$

$$j = \frac{t_1 - t_2}{\ddot{\theta}_1 - \ddot{\theta}_2} \qquad \text{(eq. 3.4b)}$$

where:
k, j, and t are scalar versions of K, J, and T.

For the case where N=1, (a limb having a number of joints), N^2+N measurements of torque and acceleration must be made in order to solve N^2+N equations. By analogy to eq. 3.4:

$$J = \Upsilon \cdot \ddot{\Theta}^{-1} \qquad \text{(eq. 3.5a)}$$

$$K = T_{av} - [\Upsilon \cdot \ddot{\Theta}^{-1}] \cdot \ddot{\theta}_{av} \qquad \text{(eq. 3.5b)}$$

where:

$\Upsilon = [T_1 \vdots T_2 \vdots \ldots \vdots T_N] - [T_{N+1} \vdots T_{N+1} \vdots \ldots \vdots T_{N+1}]$

$\ddot{\Theta} = [\ddot{\theta}_1 \vdots \ddot{\theta}_2 \vdots \ldots \vdots \ddot{\theta}_N] - [\ddot{\theta}_{N+1} \vdots \ddot{\theta}_{N+1} \vdots \ldots \vdots \ddot{\theta}_{N+1}]$.

T_i and $\ddot{\theta}_i$ are the i'th measurements of T and $\ddot{\theta}$.

X_{av} denotes the average: $(X_1+X_2+ \ldots +X_{N+1})/(N+1)$.

This is the *inversion equation*. These calculations can be performed if N+1 sets of $\ddot{\theta}_i$ and T_i are available, where the acceleration vector is the response produced by issuing the torque vector to the limb as a command. These computations derive information about the mechanical system from the relationship between the efference copy, T, and the reafference, $\ddot{\theta}$. Once again, the values of the N^2+N constants appropriate to a particular time slice can only be found when each measurement contributing to the computation, (Eq. 3.5), was made while the limb was near the state prevailing during the slice.

The procedure for finding the values of the mechanical constants J and K for one time slice are given above, but the goal is to process movements which are composed of many slices, each of which may correspond to different mechanical states of the limb. To insure the achievement of this goal the operations of collecting data and calculating constants must be organized. The necessary organization arises by considering a discretized state space and the use of two types of memory; the temporary buffer and the state space memory.

3.2.3 The Command-Torque Relationship

Until now discussion has centered around the torque applied to the joint rather than the command issued to the actuator. Unfortunately, the model constrains the relationship between the motor command and the actuator force. This relationship must be linear in the following sense:

$$t = a(\theta,\dot{\theta})\cdot u + b(\theta,\dot{\theta}) \qquad \text{(eq. 3.6)}$$

where:
t is the force applied to the tendon
u is the motor command
$a(\theta,\dot{\theta})$ and $b(\theta,\dot{\theta})$ are state dependent constants.

This restriction says that for any given state of the limb, incremental changes in the motor command must produce proportional changes in the torque delivered to the joint. This is a weak restriction. The actuator torque need not be proportional to the motor command, nor must it be constant if the command does not change. (Indeed, if a human arm is moving and the command to the muscle does not change, the force at the tendon will increase if the muscle is stretched and decrease if unloaded.) The only requirement is that given the same mechanical conditions (ie. the state does not change) all increases in command produce changes in force which are related by a constant multiplier.

Assuming the nature of the actuator does not conform to this restriction, the requirement can be satisfied by introducing local torque or force feedback. The problem solved by this local process, making sure the motor-command/actuator-output relationship is linear in the sense of Eq. 3.6, is only one dimensional: It can be solved easily because there are no dependencies on variables related to other joints.

3.3 The State Space Memory and the Temporary Buffer

The system cannot have stored, nor can it calculate the constants needed for every attainable state of the limb, for the number of such constants is infinite. The best it can do is let each state be *near* a state for which data are stored or can be stored. Let us divide the range of each dimension of the state space, (one dimension for each joint's position and velocity) into M intervals. The 2N dimensional state space is then partitioned into M^{2N} hyper-regions. If M is chosen to make the size of each hyper-region reasonably small, and the values of J and K are available for one state in the hyper-region, then all the states in that hyper-region can be said to be near a state for which data are stored. If all the measurements contributing to the calculation of a set of constants were generated while the state of the limb was in one hyper-region, the assumption can be made that the constants correspond to a state in that hyper-region. This statement will surely be true for large M.

If one keeps in mind this notion of a discretized state space, the operation of the translator with respect to the acquisition of constants of mechanical description can be made clear. During self-produced movements data are generated which must subsequently be used to calculate the constants of mechanical description. The data for these computations are pairs of simultaneously generated acceleration and torque vectors. These pairs of vectors cannot always be used immediately because each application of Eq. 3.5 requires N+1 sets of vectors from the same region of state space. Since the state of the limb is constantly changing, only a limited amount of data from each movement is pertinent to a given region of the state space at a time, and

the data that are available must be saved. Hence the temporary buffer. Although its use is quite different, the type of data stored in this buffer is similar to that of Held's correlation store (Held, 1961).

When N+1 pairs of vectors from the same region of state space accumulate in the temporary buffer, J and K are calculated by the translator, and they are saved. The state space memory is organized so that it can store N^2+N constants for each hyper-region of the space -- $(N^2+N) \cdot (M^{2N})$ constants in all. In certain cases values for J and K will be calculated for regions of the space for which previous results exist. In order to reduce noise and provide the ability to adapt to changes in the mechanical properties of the system, new and old values of J and K are averaged with some sort of weighting which favors recent data.

Two ideas regarding access to the state space memory are important if the translator is to realize its full power. A full treatment of these ideas must take into account details of the implementation, (type of storage, dimensionality of the space, slice duration, speed of computation, etc.), and are therefore only introduced here.

A direct interpretation of the arguments advanced in the previous section would indicate that translation of one slice of a desired trajectory requires only one memory access. Only data for the state of interest should be used. But suppose no data are available for the desired state. Since the mechanical behavior of the limb varies smoothly throughout state space, data from nearby regions could be used as substitutes. Such a procedure has a distinct advantage during the early stages of learning, and during generalization testing when there is a shortage of data. At these

times it will be desirable to make maximum use of available data, even if the values are somewhat deviant. At least the signs will usually be correct and movement will be possible. Responsibility for this procedure rests with the *neighborhood function*, an example of which is given in Section 4.2.3.1.

The tabularized equations embodied in the state space memory are only piecewise constant. Each time data for a state are desired, data for a nearby state are returned. One can imagine substantial improvements in accuracy of the resulting data if interpolation is used. Many possible interpolation schemes exist, some involving increased storage demands, others increased memory access and computation. A version of the model that implicitly combines a neighborhood function with an interpolation function is given in Section 6.2.1.

3.4 Combined Operation of the Components

Figure 3.1 is a diagram of the system under discussion. Its operation can be summarized as follows. High level processors produce descriptions of desired movements, $\ddot{\theta}_D(t)$, which are presented to the translator. These descriptions explicitly state the time course of the movement so that position, velocity, and acceleration information are available for each dimension of the coordinate system in use. The desired movement is sectioned into time intervals or slices, each of duration Δt. For each time slice Eq. 3.2, the translation equation, used in conjunction with the mechanical information in the state space memory, generates a force plan that will replicate the desired trajectory. The calculated force commands are issued to the limb and, during the movement, a copy of the command, the efference copy, and a copy of the sensory

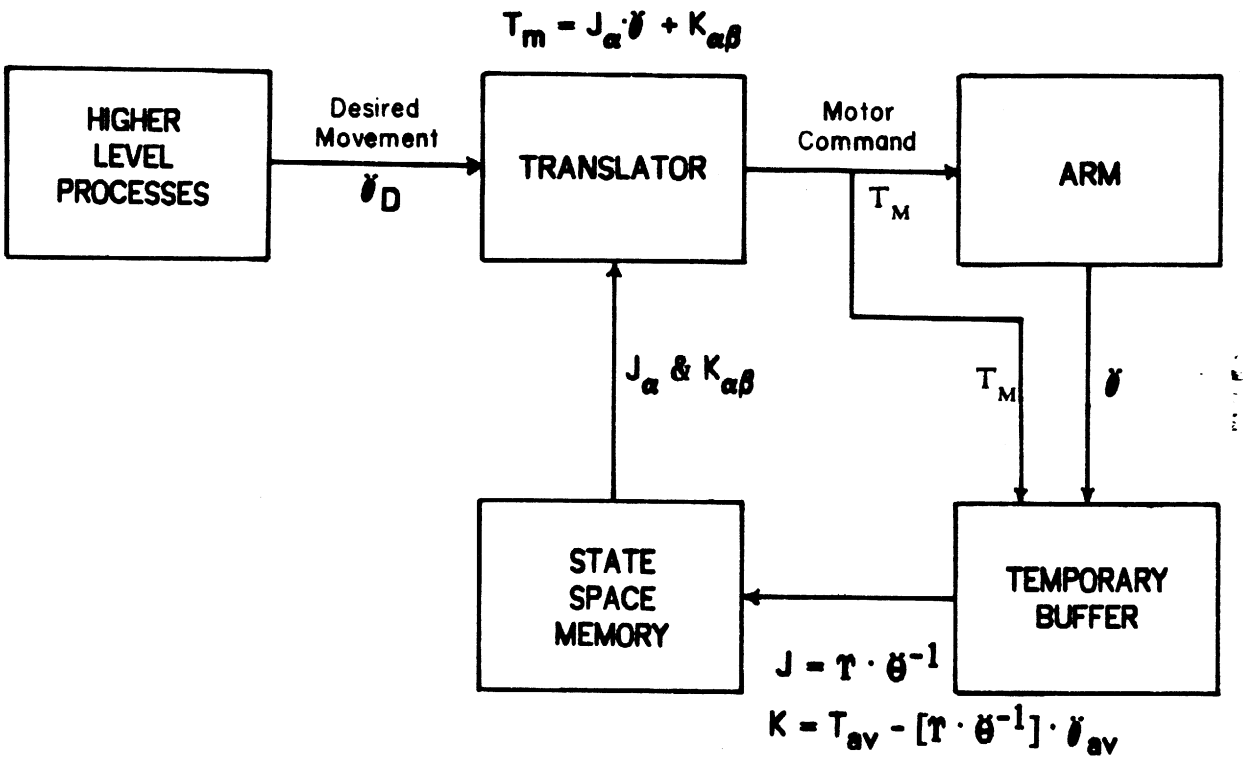


Fig. 3.1 Major components of the model. The translator converts descriptions of desired trajectories into motor commands suited to the kinematic and dynamic properties of a particular limb. The operation employs the tabular equations of motion in conjunction with the state space memory. Each movement of the limb generates data which, when processed by the inversion equations, contribute to the state space memory, and consequently, to future translations.

signals that indicate progress of the movement, the reafference, are stored in the temporary buffer with labels indicating the region of the state space to which they apply. Subsequently, the contents of the temporary buffer and the inversion equation, Eq. 3.4, are used to find values of J and K. These results are stored in the state space memory in combination with data that might have been stored there previously.

3.5 Properties of the Model

Initial performance of the translating mechanism will not be good. Every attempt to use information about the mechanical character of the limb will be frustrated because the state space memory will be empty. Data describing the mechanics of the limb are only available after movements have been processed. If no data from the state space memory are available two things can happen. The translator could use some preset or genetically encoded constants and proceed to generate a set of commands even though the resulting movement may be quite different from the one desired. Alternately, some other control system can take over when the translator finds that it has no usable information. Under this circumstance the controller will not take part in the production of the movement. In either case it is important that the remainder of the translator's functions, (ie. the analysis of the efference copy and the reafference by application of the inversion equation), be performed when the movement is executed, even though the resulting movement may bear little resemblance to that specified by the high level processor. If this were not the case the system would never have the opportunity to build up its memory and improve. (This would be something like the fellow who cannot get a job because he has no

experience, and cannot get any experience without a job.)

As more and more movements of the limb are made, more and more data describing the mechanics of its operation become available to the translator. During this period of data acquisition the quality of movement produced by the translator will gradually improve. Intensive practice, repeated approximations to the same movement, will facilitate mastery of a specific movement because a higher percentage of the incoming data are relevant to the regions of the state space memory accessed during replication of the movement of interest. It is also true that more movement data of any kind are available during intensive practice.

While heavy practice of one or a group of movements should improve the ability to execute the practiced movements, other movements will also be facilitated if they are similar to those practiced. The type of transfer described here, from a highly practiced movement to a similar, but less practiced one also contributes to the appearance of a general improvement in motor performance. In fact, the characteristic which prompts one to call the improvement general is that movements are performed with only modest amounts of error, though never before explicitly practiced. It must be understood that the effectiveness of intensive practice upon the practiced and similar movements may be influenced, to a large degree, by the details of the practice strategy -- details considered in Section 4.3.4. Since neighboring regions are only defined in terms of the neighborhood function, it is also important in this regard.

The State Space Model places very few restrictions on the dynamic and kinematic properties of the limb being controlled, or the geometries of the limb's sensors and

actuators. A single translator can learn to control almost any limb. This is a direct result of the tabular nature of the equations that describe the mechanical system. Each member of the chain of transformations between response and command:

1) sensor signal → sensor acceleration (eg. $x_{5,\alpha\beta}$ (mm/sec^2)/volt)
2) sensor acceleration → joint acceleration (eg. $x_{4,\alpha\beta}$ (rad/sec^2)/(mm/sec^2))
3) joint acceleration → joint torque (eg. $x_{3,\alpha\beta}$ (newton-meter)/(rad/sec^2))
4) joint torque → actuator force (eg. $x_{2,\alpha\beta}$ newton/newton-meter)
5) actuator force → command signal (eg. $x_{1,\alpha\beta}$ volt/newton)

can be represented in terms of a set of constants for a particular state. None of these transformations need be known in advance, nor are they ever known individually. The translator uses the inversion equation to compute a net transformation representing the total of these operations. Note that no particular units need be used; the controller 'thinks' in terms of actuator control signals and raw sensor readings.

From a practical point of view, this geometric freedom means that the joints can be revolute or sliding. The forces applied by the muscles can undergo non-linear transformations due to the joint-tendon geometry, without consequence. In fact, actuators and sensors need not be affiliated with any one joint or subset of joints. Reafference can take the form of visual feedback just as readily as joint oriented proprioceptive feedback, provided the choice be made before learning commences and desired trajectories are described in the chosen coordinate system.

To recapitulate, the controller described here will exhibit the following desirable properties. It will use practice to learn to translate descriptions of desired trajectories into motor commands. Training will transfer among similar movements and the system will adapt to mechanical changes. The geometry of sensory, linkages, and actuators is

quite flexible.

At this point I must reiterate that all properties of the motor system are not being attributed to the translating mechanism. Just because the translator learns and adapts does not mean that other processes do not also learn and adapt. It is assumed that they do.

3.6 Discussion of the Model

It should be stressed that the reason there is learning is not that errors in previous movements are explicitly corrected, nor that errors in the constants which specify the mechanical properties are explicitly corrected. Movement errors can only be detected if desired and produced trajectories are compared, but this is never done by the system presented here. Motor performance is gradually improved with experience for two reasons:

1) Each movement submitted for translation requires data from a number of regions in the state space memory. More of these data are available when the system is more experienced, because these data are generated directly from the movements which comprise experience.

2) If there is any noise in the system (there always is noise in physical systems) the data available from the state space memory become more accurately specified when they are calculated a number of times because noise is reduced through averaging.

When the constants for a region of the state space memory are calculated a number of times, the average of those calculations will converge upon the true value of the mechanical properties they represent provided the mechanical properties of the limb are constant and there is zero mean noise in the system. In the event the mechanical properties are not constant -- a situation which can occur when the

organism grows, the muscles get stronger, or the sensory elements change -- repeated calculations of the mechanical constants will reflect the changing properties and ultimately converge some time after the limb stabilizes. The exact nature of this adaptation process depends on the rules of combination that apply to the storage of new data into the state space memory. The only statement on this score to be made here is that a weighted average which favors recent data performs in an adaptive way. Improved noise rejection is demonstrated, however, if the time-constant of the memory is as long as possible, while still being short with respect to the time-constant of changes in the mechanical properties of the limb.

There are two reasons for generalization between similar movements:

1) Two similar movements will use data from the same hyper-regions of the state space, or from neighboring regions. (This type of transfer might correspond to Thorndyke's *identical elements* theory, though he probably had a higher level process in mind (McGeoch, 1952; Hilgard & Bower, 1975).)

2) Due to variations occuring during practice, training of one movement may generate data appropriate to other similar movements, even if no hyper-regions are in common.

These two processes interact, and are not distinguishable in all cases. By similar I mean that the same or nearby regions of the state space memory are used to generate the movement. This definition is quite limited. All the problems of pattern recognition and pattern description bear on this question of similar movements and more will need to be known about task analysis and motor function before adequate definitions are possible.

The ability to plan trajectories in a range of coordinate systems could contribute

a high degree of added versatility to any controller. Such a device could be trained to plan trajectories on the basis of measurements and targets in a Cartesian coordinate system, a polar system, a joint system, or any of a large class of other coordinate systems. This idea is appealing if one thinks of visual information as being specified in a cartesian or polar-like system while proprioceptive sensors work in joint coordinates. This idea is especially important in view of the fact that it could allow an arm to be controlled on the basis of on-line visual space errors, provided the translator is used in the reprogramming mode. A servo mechanism would require an additional coordinate transformation process in order to perform this function. Due to the tabular nature of the state space memory the transformation of coordinates takes place in the SSM at no extra cost .

Young and Stark (1965) and others have proposed use of the *internal dynamic model* as a mechanism for learning and adaptation. (These systems are also called *Model Reference Adaptation Systems (MRAS)*, (Landau, 1972).) They argue that commands can be tested on the internal model and adjusted until they produce the desired response. This idea can be made to work, but I would like to stress a concept which represents a different point of view -- the *internal inverse dynamic model* (Paul, 1972; Waters, 1974). The idea here is that the a motor learning system should have a transfer function that converts responses into commands -- the inverse of the operation performed by the plant -- not commands into responses. The overall transfer function is the identity matrix, I, the desired result.

The internal dynamic model allows one to simulate the inverse function with an

analysis by synthesis type approach (Eden, 1962). The State Space Model, however, because its uses sets of extremely simple equations to describe the plant's behavior, can directly calculate the required inverse functions for each region of space. It is only fair to stress that I am arguing point of view rather than computational approach. Iterative techniques for solving an inversion computation, (Young and Stark's approach is such an iterative procedure), are quite common and legitimate. But it is important to conceptualize the operation in clear terms.

The areas of motor physiology that deal with details of motor control problems have not advanced sufficiently to have developed a vocabulary suitable for discussion of problems related to learning. I therefore take the liberty of using general psychological terms which convey the rough intent of my meaning. The terms *transfer* and *generalization* are cases in point. They are used throughout this thesis to describe various properties, yet most of the biological learning situations in which transfer and generalization are defined and studied involve much higher-level tasks than those studied here. In this regard, my use of these terms is perhaps metaphorical.

Generalization usually refers to a lack of discrimination between or among stimuli and transfer refers to the effect a procedure has on a number of similar responses. The distinction between sets of stimuli and sets of responses cannot be drawn so sharply here. Each request for data from the state space memory specifies a state. This state acts as a stimulus, (Marr (1969) calls it the context), which is generalized by the neighborhood function. On the other hand, practice of one movement often produces data for remote states which are only appropriate to other movements. This

effect is more like transfer. The two affects combine to produce one behavioral result. The two terms are used interchangebly here.

A set of experiments by Held and Freedman (1963) and Held and Hein (1963) showed that self produced movements are required for motor learning and movements produced by an external agent are not adequate for learning. The state space model also behaves in this manner. Measurement vectors can be produced and learning can take place, only if torques are generated in a way that makes the actuator commands known to the system. If sensors are used to measure the torques applied at the joints (Section 5.6.1) rather than storing an efference copy, the forces that accelerate the limb must be applied in such a way that the sensors are stimulated. Biologically speaking, this means that tendon receptors adequately measure the forces delivered to the joints only if forces are applied *through* the tendons -- movement of the arm by a cradle does not produce such stimulation. By the way, any mechanism that relies on a form of the equations of motion for control and learning will probably have this constraint. The system proposed in Section 6.2.3, however, does not.

4 Implementation and Test

The predictions of the last chapter are based on reason and intuition; this chapter and the next examines these predictions experimentally. I would like answers to the following questions:

1) How well does the translator perform vis-a-vis its expected desirable properties?

2) How does the behavior of each component of the translator influence overall performance and contribute to successes and failures?

3) What relationships can be drawn between the behavior of the translator and that of the human?

4) How might the processes described here coexist with other models for control?

Answers to these questions depend on data obtained using a variety of tests applied to an implementation of the model which, it is hoped, adequately reflects its power and its weaknesses. These data include measures of overall performance during learning and adaptation, as well as information about the behavior of internal variables.

In order to evaluate and verify the power of the model, a set of computer programs were developed to embody the various computational elements. These programs are used to control a mechanical arm in order to study the detailed nature of the resulting movement. A number of notions have been introduced -- the translation equations, the temporary buffer, the inverse computations, the discretized state space memory, desired trajectories, and a translation process -- which now have to be made concrete.

4.1 Facility

A PDP-11/45 computer is used to perform all computations, to issue commands to the manipulator, and to make measurements. The manipulator is the MIT-Vicarm, manufactured by Victor Scheinman. It has six degrees of freedom; the three joints used in this study, (N=3), allow the wrist to be positioned arbitrarily within the arm's work space. See Fig. 4.1. Each joint is powered by a DC torque motor and provided with a clutch-type brake which can be used to hold the arm stationary when no movement is in progress. The PDP-11 may, through suitable circuitry, specify the current delivered to each motor. DC torque motors have the characteristic that the torque they deliver is proportional to the winding current, independent of armature velocity. Since the currents for each motor may be specified independently and simultaneously the PDP-11 computes a vector which determines the torques applied to the joints of the arm.

Signals proportional to angular position and velocity are available from potentiometers and tachometers provided for each joint. When a movement is made the computer makes position and velocity measurements every 10 msec. In addition the velocities, sampled every .5 msec., allow the accelerations to be estimated using least-mean-square error techniques. A record of each movement can be saved for future use where each record represents up to 1.2 seconds of movement and contains position, velocity, acceleration, and motor current information for each of the three joints.

The mechanics of the joints are completely backdrivable -- the torque produced

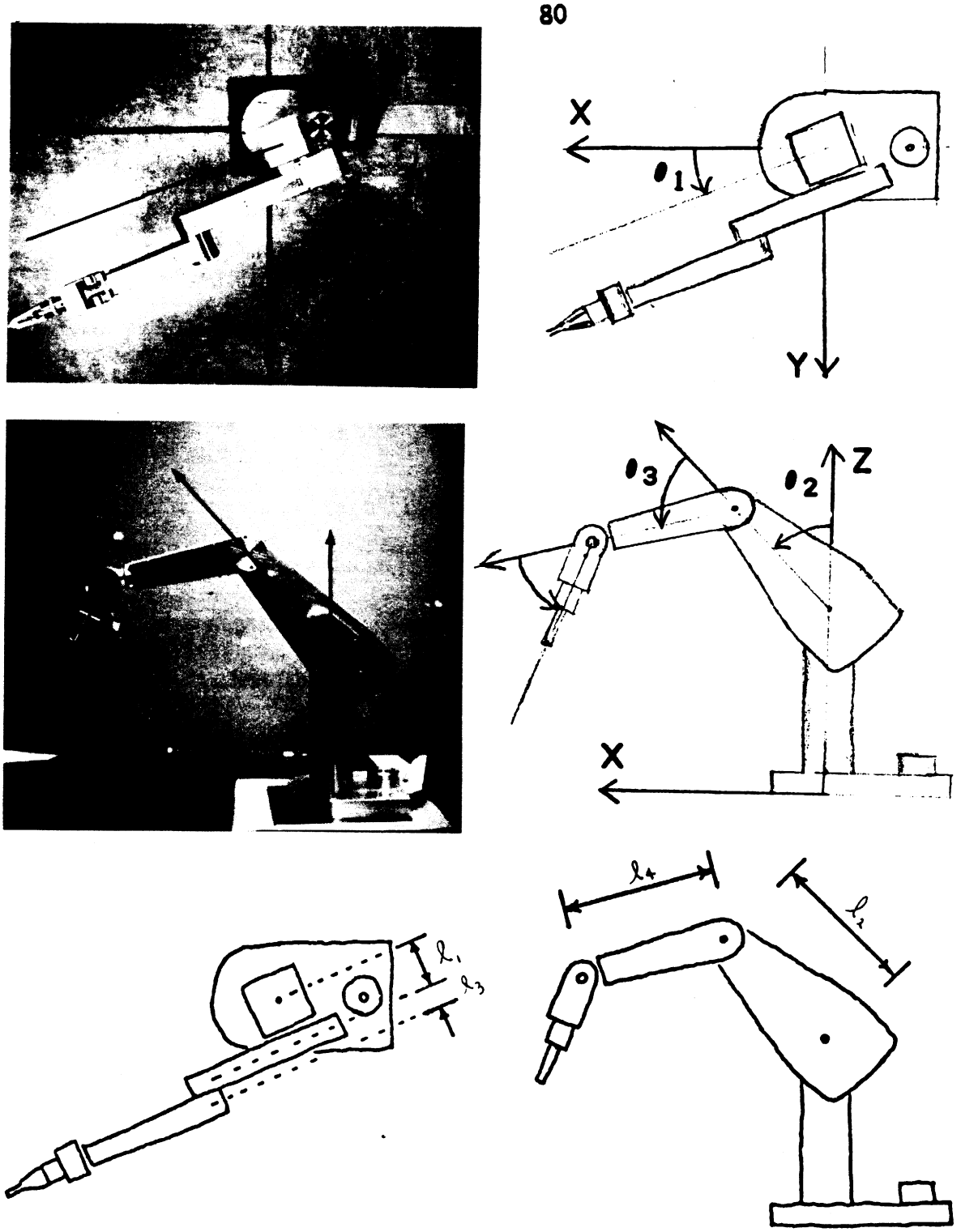


Fig. 4.1 Layout of the first three joints of the MIT-Vicarm manipulator. θ_1 acts about the vertical axis. The manipulator is about the size of a human arm; l_0=.273m, l_1=l_3=.059m, l_2=l_4=.203m. Each joint is provided with a DC torque motor, a potentiometer, a tachometer, and a clutch-type brake. The diagram is from (Horn and Inoue, 1974) with modifications.

by the motor plus externally induced torques sum to determine the motion of the joint. Consequently, the motions of each joint are a function of the torques applied to all the joints. This fact, which is generally true of biological limbs, is illustrated in Fig. 4.2. A step of current applied to the motor which drives joint 2 caused changes in the trajectories of joints 1 and 3. For some manipulators, these interactions may be ignored (Paul, 1972).

Accurate acceleration information is essential to the use of the inversion equations. Since the Vicarm manipulator has no acceleration sensors, accelerations are estimated by fitting straight line segments to the recorded velocity trajectory for each time slice. The duration of the slice, therefore, must be selected to optimize two conflicting factors affecting the quality of these estimates:

1) Constant acceleration estimates are only appropriate if the actual time-varying acceleration is nearly constant. This is most nearly true during very short time intervals.

2) When more velocity samples contribute to each estimate they are more precise and noise-free. Therefore, longer intervals are called for.

I examined sections of 40, 50, 60, 80, and 120 msec. (with 2 khz sampling rate per joint) in order to find an acceptable compromise. Figs. 4.3a and 4.3b show that 60 msec., the time slice duration used for all data reported here, was acceptable. The variation in acceleration estimates for 10 repetitions of the same movement are shown in Fig. 4.3a. The repeatability of most values is very good, indicating an adequately long estimation interval. Fig. 4.3b, showing a rather faithful reconstruction of an original trajectory from its estimated accelerations, demonstrates that most of the information present in the original time-varying acceleration trajectory is captured by the

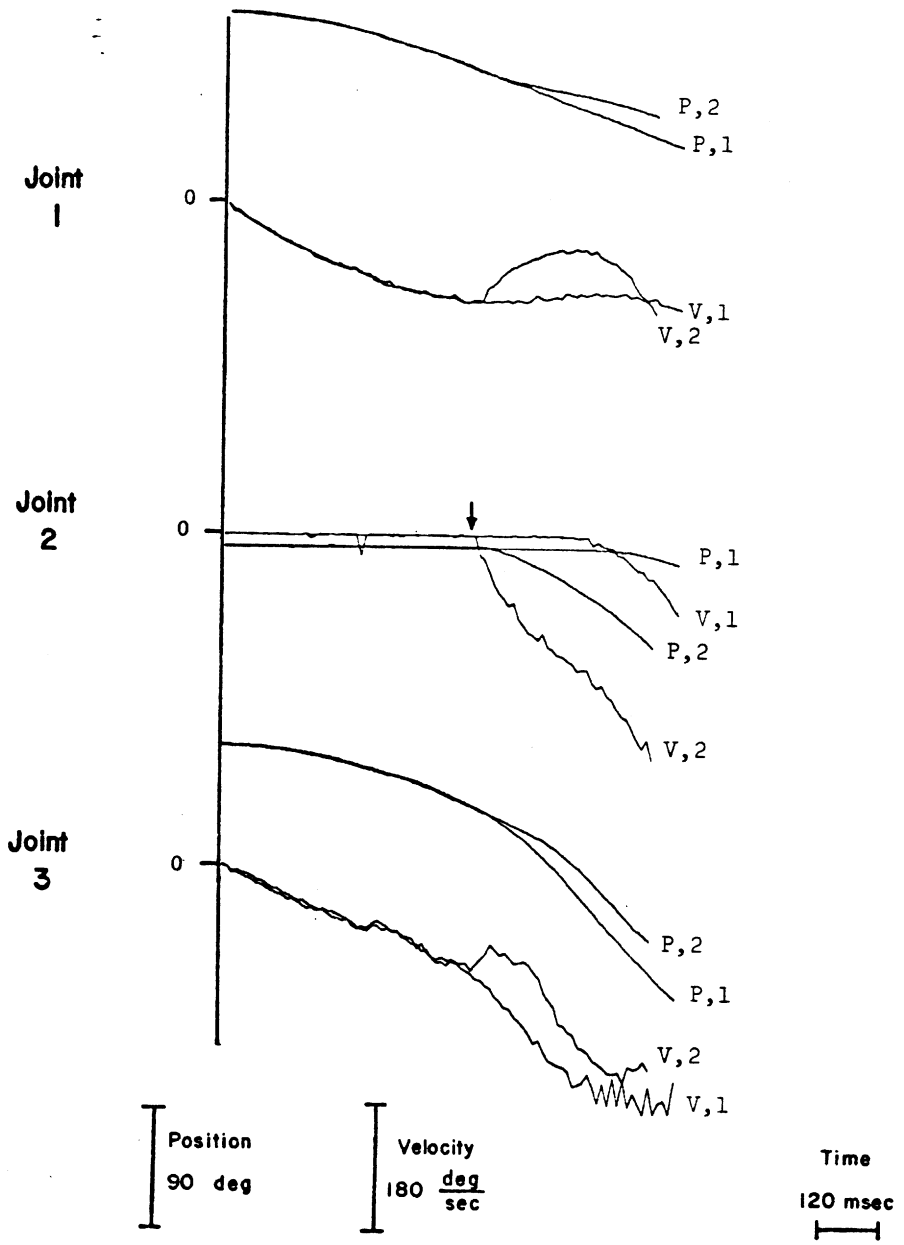


Fig. 4.2 These curves demonstrate the potential for mechanical interactions among the joints of the Scheinman arm -- a property characteristic of biological limbs. Each movement labelled *1* was made by applying constant torque to each joint. In movement *2* the torques at joints 1 and 3 were unchanged, but a step was applied to joint 2 after 500 msec. (at arrow). Note that the position and velocity trajectories of all three joints were affected. P-position; V-velocity.

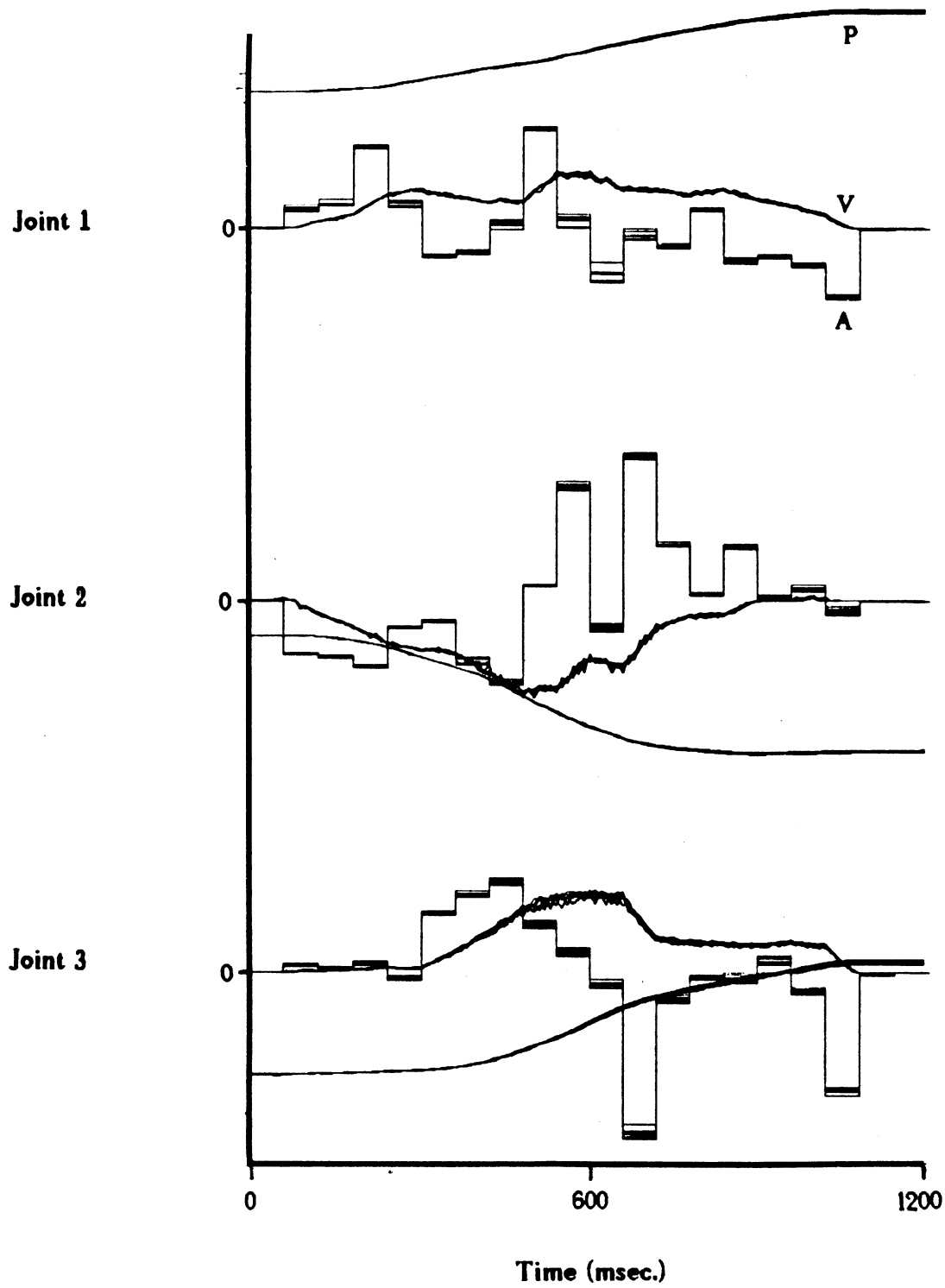


Fig. 4.3a The variability of acceleration estimates was determined by executing ten repetitions of the same movement. During most time slices there were only small variations in estimated acceleration, indicating an adequately long sampling interval.

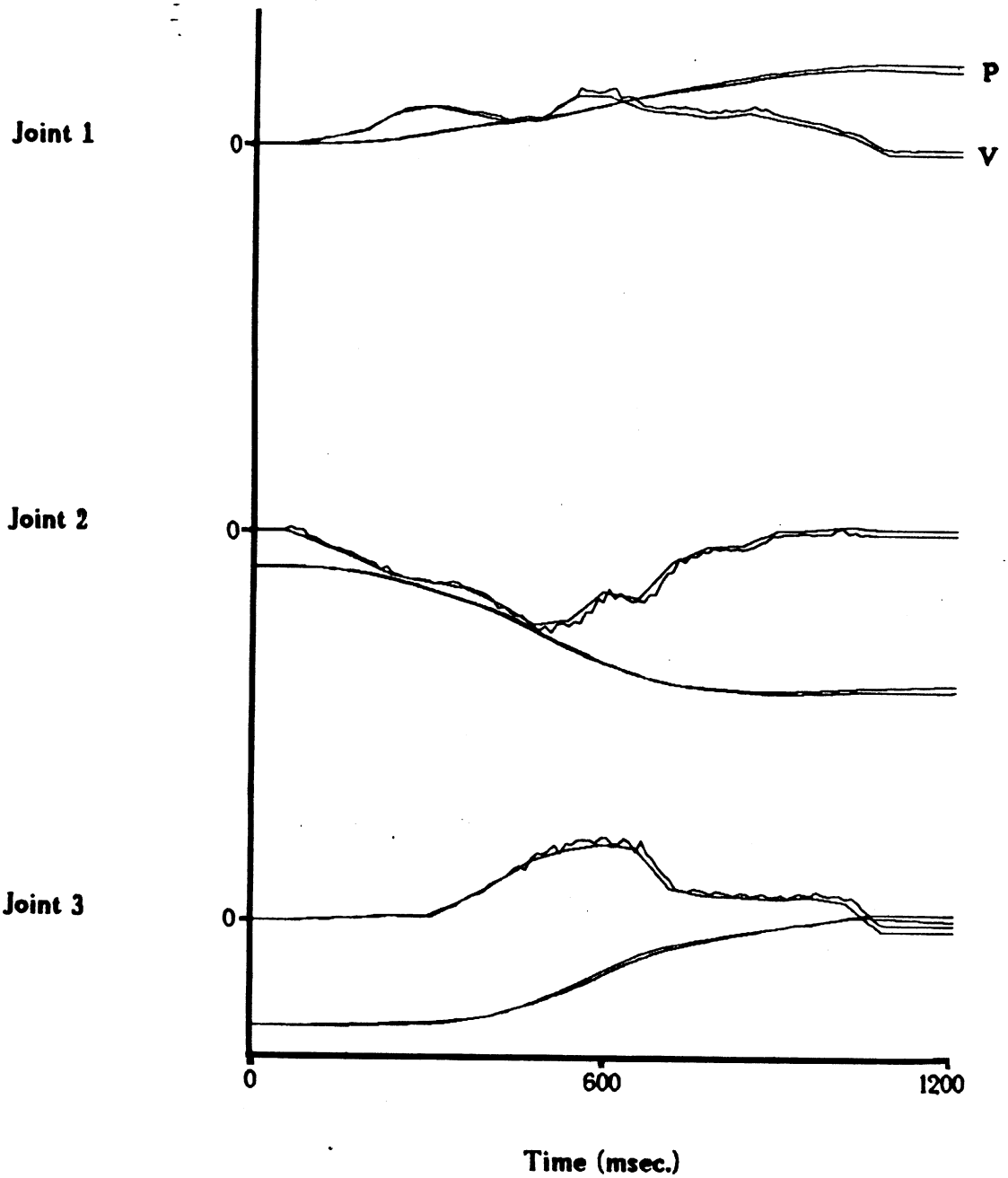


Fig. 4.3b These curves show that 60 msec. piecewise-constant estimates of acceleration are informationally adequate for description of a typical movement. During execution of the movement, position, velocity, and estimated acceleration were recorded. The reconstructed position and velocity trajectories were computed by integrating the estimates of acceleration. The correspondence between the recorded and reconstructed trajectories is quite good.

piecewise constant estimates.

4.2 Information Processing

4.2.1 Computation of the Inverse

4.2.1.1 *The Invertibility Index*

In order to calculate data for storage in the state space memory, the constants of mechanical description, it is necessary to invert an N dimensional matrix of acceleration measurements. (Actually these are acceleration differences taken from N+1 sets of measurements. See Eq. 3.4.) One can not invert a matrix if it is singular, but N sets of N measurements taken from a physical system are unlikely to be strictly linearly dependent. Care must be taken that the matrix of acceleration estimates is well conditioned, for inversion of an ill-conditioned matrix amplifies noise (Noble, 1969). In order to avoid the potentially disastrous effects of inverting an ill-conditioned set of noisy measurements, two precautions are taken.

Each group of vectors are screened before inversion by a conditioning index, x, which determines the degree to which a set of measurement vectors are independent:

$$x = \frac{|A'_1 : A'_2 : A'_3|}{\|A'_1\|+\|A'_2\|+\|A'_3\|} \qquad \text{(eq. 4.1)}$$

where:
$A'_i = A_i - A_4$
A_i is a column vector of dimension N=3
$\|A\|$ is the norm of A.

The numerator of this index will be small if the matrix, A, is nearly singular. The denominator insures that very large vectors do not make a nearly singular matrix appear to be non-singular. Only sets of measurements that meet a criterion value of

the index contribute to the state space memory. When a set of vectors fail the conditioning test, the two least contributory vectors (smallest cross-product) are averaged together and replaced in the temporary buffer. The criterion used in these experiments was chosen after a crude examination of results with various values. The value used in each experiment is listed in the Appendix.

4.2.1.2 *Use of the Generalized Inverse*

In theory, the calculations that produce data for the state space memory can be performed when only N+1 measurement vectors have been collected. When the computations are performed in this perfectly constrained manner, the effects of noise can be quite large. The resulting inverse rigidly applies to the measurement data, analogous to the way a straight line fits only two data points. More than N+1 measurements can be used to reduce the effects of noise, much the way a straight line fit to more than two data points minimizes the influence of noise present at each point. To perform the computation on more than N+1 measurements we have to use the generalized inverse, since a matrix must be square to be inverted in the usual sense. Using the generalized inverse any number of measurements can be regressed in an analogous manner to the line fit mentioned above. This operation does, in fact, minimize the error of the inverse in the mean square sense.

The value of using more than N+1 measurements is demonstrated in Fig. 4.4. These histograms were made by generating a set of measurement vectors relevant to one region in state space and inverting them N' at a time, where N' was 4, 6, and 8. The value distributed is one element of the resulting matrix. The figure shows that

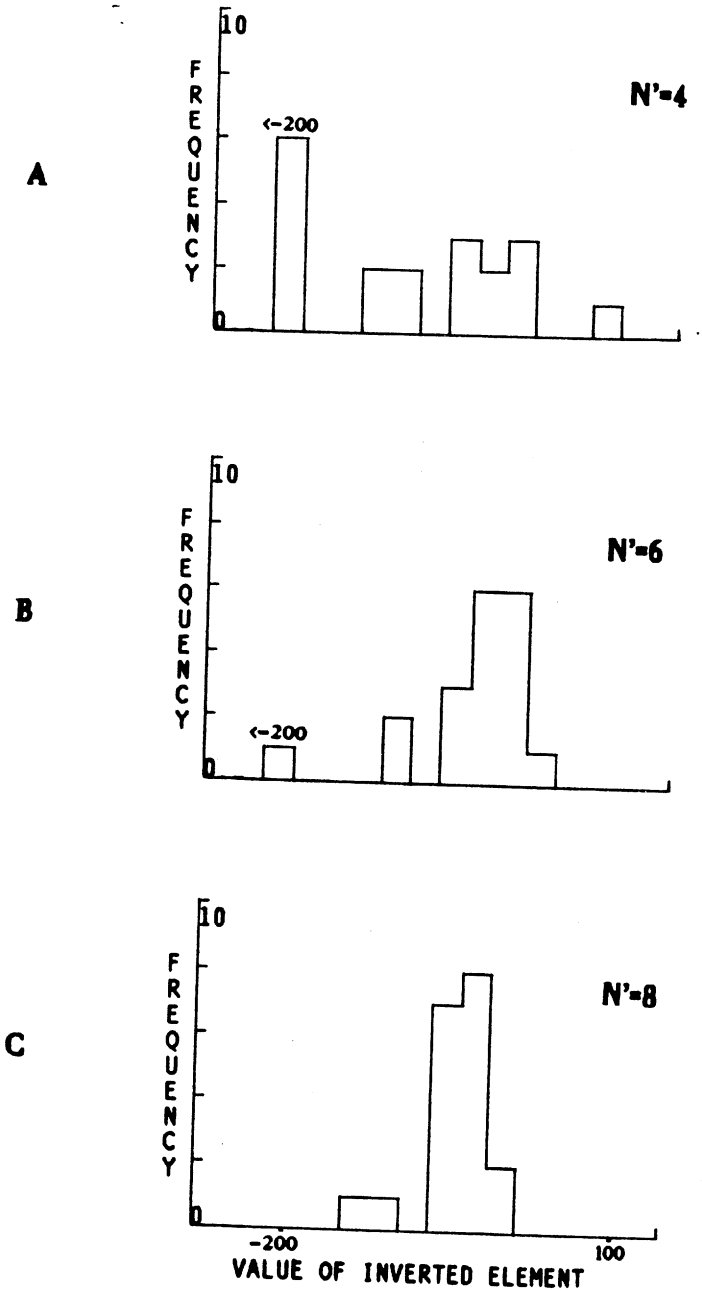


Fig. 4.4 Use of the generalized inverse with more than N+1 vectors reduces variability in the resulting computations. The histograms show the variation in computed values for one element of the J matrix with N' as a parameter. A) N'=4, (ordinary inverse); B) N'=6; C) N'=8, (used throughout thesis).

when more than the minimum number of measurements is used, N'>N+1, the results are more consistent and less subject to extreme variations. Values of N'=12 and 16 were also tested, but the additional computational burden was not justified by the resulting improvements in noise rejection. For this report N'=8. A discussion of the generalized or pseudo inverse is given by Albert (1972). The particular algorithm used here, an extension of an orthogonalization method, is given by Rust et al. (1966).

4.2.2 The State Space Memory

The state space memory is organized into a large number of small regions, each corresponding to a different state of the manipulator. Two factors determine the effective *size* of these hyper-regions; the parameter M, the number of divisions along each dimension of the state space, and the range of values each state variable is permitted to assume. The details of dividing up the state space memory are given in Table 4.1. For the implementation using joint variables as coordinate system variables and with M=10, (M^{2N} or 10^6 defined states (for N=3)), each hyper-region measures $(15 \text{ deg})^3$ by $(13 \frac{\text{deg}}{\text{sec}})^3$. These regions are actually quite small, and the mechanical properties of the arm are fairly constant throughout. Fig. 4.5a gives an idea of the region of joint space represented in the memory. Naturally, all six dimensions of the space can not be displayed so only the positional dimensions are represented.

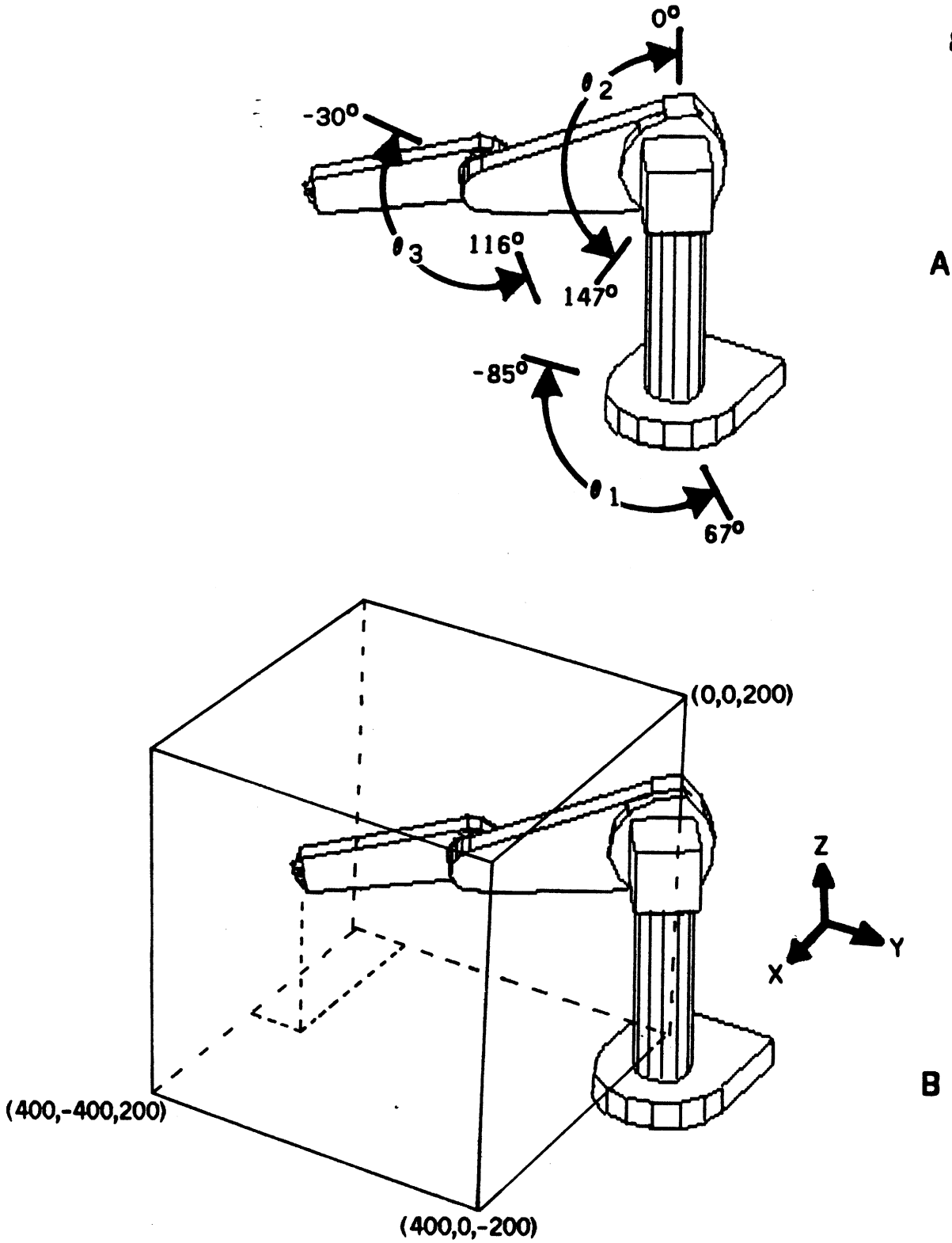


Fig. 4.5 Only a portion of motor space is represented in the state space memory. The range of the positional dimensions of the state space memory are shown. A) Joint coordinates are used for measurement data. B) Cartesian coordinates are used. Coordinates shown are given in mm.

Table 4.1

Joint Coordinate Measurement

Dimension	Min Value	Max Value	Cell Size
Position: joint 1	-85 deg	67 deg	15 deg
joint 2	0	147	15
joint 3	-30	116	15
Velocity: joint 1	-12 deg/sec	112 deg/sec	12 deg/sec
joint 2	-112	12	12
joint 3	-12	112	12

Cartesian Coordinate Measurement

Dimension	Min Value	Max Value	Cell Size
Position: X	0 mm	400 mm	40 mm
Y	-400	0	40
Z	-200	200	40
Velocity: X	-613 mm/sec	68 mm/sec	68 mm/sec
Y	-613	68	68
Z	-613	68	68

In some tests Cartesian coordinates were used for the state space dimensions. For those cases each hyper-region measures $(40\ \text{mm})^3$ by $(12\ \frac{\text{mm}}{\text{sec}})^3$. The region of Cartesian space represented in the state space memory is shown in Fig. 4.5b. Again, only positional dimensions are shown. Although Table 4.1, Figs. 4.5, and the previous discussion deal in physically dimensioned variables, the actual implementation works entirely in sensor units. Only the range of the sensor variables need be specified in advance since the relationship between sensor units and physical units is implicitly determined during learning.

In order to make efficient use of storage resources, the state space memory is hash coded. Collisions are avoided through a re-hash procedure. Though another investigator has incorporated hash coded memory as an explicit component of a system for control (Albus, 1975), in this work the hash coded memory is merely a concession

to a practical problem -- a shortage of storage. The hashing procedure is transparent to the translator, and has no deeper implications for the system's operation.

4.2.2.1 *Initialization*

An assumption of the state space model is that memory is initially tabula rasa. But what does that mean? For a neuronal mechanism it might be connections of zero strength, connections of random strengths, no connections, or nothing to do with connections. Here I have distinguished between no data and zero values. The two situations where this question arises, entering new data into the memory and applying data from the memory, are treated separately and explained below.

4.2.2.2 *Time Constants*

When new data are computed for regions of the memory, they must be stored in combination with data that are already present. Many procedures which combine new and old data will produce adaptive behavior. Without a great deal of theoretical justification, having designated τ the memory time-constant, the following data combination procedure was chosen:

> If $k<\tau$ where k is the number of times a hyper-region has been updated, and $k<\tau$, each new datum is weighted by $1/(k+1)$ and the old value is weighted by $k/(k+1)$ -- the first τ values are weighted uniformly. When $k>\tau$ the new value is weighted by $1/\tau$ and the old value by $(\tau-1)/\tau$.

τ should be chosen to give good immunity to noise while providing rapid adjustments to changes in the dynamics of the manipulator. There is a direct tradeoff between these two goals. Fig. 4.6 demonstrates the time course of the weighting factor for each piece of data stored in the memory for various values of τ. Small values of τ give a

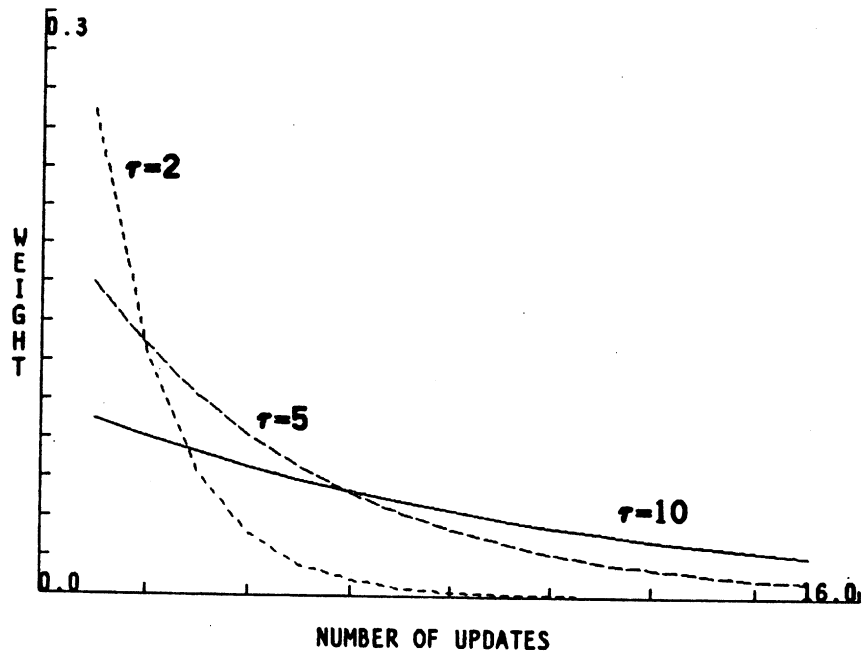


Fig. 4.6 τ is a parameter which determines the initial weight and decay-rate given to a piece of data when stored in the state space memory. There is a tradeoff between initially large weights and long lasting weights. Note that time is not a direct factor, but rate of decay depends on rate of subsequent memory updates.

large weight to new data, and the effectiveness of the data are rather transitory. Larger values of τ result in small weights, but longer lasting effects. Unless otherwise noted, τ=10 in this report. The effects of varying τ are described later. (It should be understood that the reference to time in this context is indirect: more data enter the memory as time passes. Therefore, τ has units of $1/updates$. Procedures for treating time explicitly are discussed in Sections 5.6.3 and 6.1.1)

4.2.3 Translation

When presented with the description of a desired trajectory, the translator uses the tabular equations of motion in conjunction with data which describe the mechanics of the limb to produce a set of motor commands. After dividing the desired trajectory into 60 msec. slices, (as is done to practice movements) and accessing the appropriate regions of the state space memory, the computation defined by Eq. 3.2, the translation equation, is performed in order to determine a set of motor currents. What are the *appropriate* regions of the memory?

4.2.3.1 *The Neighborhood Function*

Surely, data from the desired hyper-region are appropriate, but data from nearby states can also be useful. Use of data from neighboring states is justified since the mechanical behavior of our manipulator varies smoothly throughout the state space. Data from these neighbors can be used to advantage whenever the desired hyper-region has never been updated with information about the prevailing mechanical properties of the limb. This situation arises when data generated in the learning of one part of a movement are used to replicate other parts of the same movement, or when

a new movement makes use of data originally derived from a separate but similar movement. The distinction between these two cases can not always be drawn very sharply.

Experimentation with a number of data combination algorithms lead to the following simple and effective choice:

> Each of the desired hyper-regions' first order neighbors is accessed. (A first order neighbor differs from the desired region by only one unit on one dimension.) The contents of each are given a weight of one. The contents of the desired hyper-region are given a weight equal to the number of times that region has been updated with new information. The weighted average is used.

The range of the *neighborhood function* can have important effects on the generalization behavior of the system which are discussed more fully later.

Aside from considerations regarding generalization, the constants of mechanical description might provide better approximations of the ideal values if neighbors were used in an interpolation, rather than merely an average. Such procedures have many implications and possibilities, but were not employed here.

4.3 Tools for Testing

In this section a number of constructs are introduced that allow the implementation to be tested and evaluated.

4.3.1 Prototypes

Prototypes are internal representations of ideal movements. They are used to specify desired trajectories to the translator in the production of test movements. They are also used as target movement during practice sessions. Each prototype, is

produced in one of three ways:

1) The arm is moved manually by the experimenter while position and velocities are recorded from each joint, and accelerations are estimated.

2) A set of currents are selected by the experimenter and the arm is driven by these currents during which time the positions and velocities are measured, and accelerations are estimated.

3) A set of acceleration trajectories are selected by the experimenter and they are integrated to obtain position and velocity trajectories.

Each of these three methods produces position, velocity, and acceleration trajectories for each of the joints. Method (1) has the advantage that it facilitates the generation of complicated spatial patterns, which are difficult to decompose into the sensory system's coordinates. It is also important because programming industrial robots often makes use of this method. Method (2) has the advantage that the experimenter knows, a priori, what set of currents will produce the movement. This can be useful in conducting tests of competence rather than performance. Method (3) has the advantage that a set of prototype movements can be generated which vary in carefully controlled ways (eg. starting position, final position, maximum velocity, duration, etc.) Although all three techniques were used at some point in this study, most of the prototypes used to generate the data included in this report were produced using method (3).

Using method (3) sets of prototypes were generated and used to test the controller's performance. In order to assess the system's ability to generalize from a practiced movement to other similar movements, prototypes were generated in sets whose members systematically vary in similarity. The trajectories of another series

share final positions and durations, but differ in starting positions. These sets are used to assess the model's ability to generalize. A typical prototype, PR-11 is shown in Fig. 4.7. Other prototypes will be described with presentation of the data.

4.3.2 Two Types of Movements

Each manipulator movement may be classified according to the way the translator processes it. In theory, any process can generate commands used to produce practice movements. The important characteristic of the practice movement is that it generates data for the state space memory via the inversion, Eq. 3.4. Test movements are those produced by translation of desired trajectories in order to assess performance. These trajectories are converted into sets of motor commands using data from the state space memory and Eq. 3.2. When these commands are issued to the arm a test movement results. In principle a movement can be both test and practice, but for the sake of clarity no such overlap was permitted here.

In order to make use of a practice movement it must be divided into short duration pieces slices, just as is a desired trajectory. The duration of the slice was chosen to be the same as that used for translation and estimating accelerations, 60 msec. Once a practice movement has been divided into sections, vectors are produced and stored in the temporary buffer. These vectors contain a record of the motor currents, one for each joint, a set of acceleration estimates, and information regarding the state of the limb prevailing during the time slice. These measurement vectors are collected in the temporary buffer until enough (N') are present for a single state to perform an inverse computation.

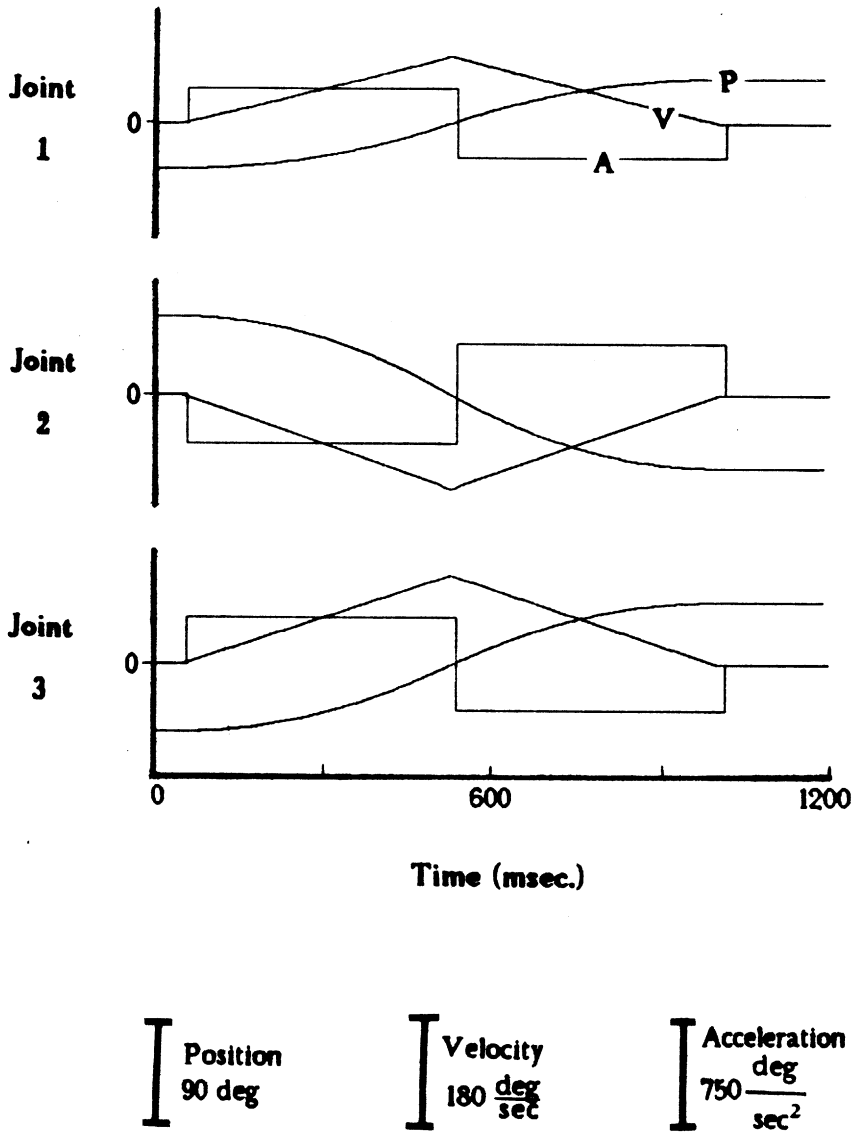


Fig. 4.7 These curves describe a typical prototype used to evaluate the model; PR-11. It was generated using method (3). P-position, V-velocity, A-acceleration.

4.3.3 Performance Indices

The behavior of the model is measured by applying performance indices to the test movements. Learning curves are created by plotting the values of one or another of these indices against the number of practice movements made by the system at the time of the test. Each index is applied to an error curve found by comparing the movement to the test prototype. These error curves are only used for analysis and do not effect performance of the system. The indices in use are:

1) Root-mean-square position error - The mean square position error for each joint and for all three joints is cumulated for the entire trajectory. The square roots of these values are reported. (RMS PE)

2) Root-mean-square final-position error - The position error at the end of a specified time interval is found for each joint. The total position error is found by taking the square root of the sum of squares of the errors for each joint. Since the joint coordinates are not orthogonal, this total measure is not equivalent to a resultant error in cartesian space. (RMS FPE)

3) Root-mean-square acceleration error - Same as RMS PE, but acceleration error is found. (RMS AE)

4) Root-mean-square velocity error - Same as RMS PE but the velocity error is found. (RMS VE)

Prototypes and movement plans have 1.2 sec. duration, but each performance indices used in this paper are usually only applied to the first 500 msec. of the test movement. This was done for practical and theoretical reasons:

1) Many movements made early in learning must be stopped before completion to avoid damage to the manipulator. Therefore, they are shorter than 1.2 sec. This argument does not apply to competence indices.

2) Most of the data produced by the practice algorithm are only useful for planning the first half the test movements. Fig. 4.8 shows that this was

true for one typical set of 3000 practice trials for one practice prototype. The extra investment of time needed to generate practice data for all sections of a prototype seemed unjustified.

3) All available evidence indicates that open loop segments longer than 300 msec. are not necessary for good control and are not found in nature (Hammond, 1956; Melvill Jones & Watt, 1971; Pew, 1974).

4.3.3.1 *Competence Index*

It is useful to distinguish between performance of the system and performance of the manipulator under control of the system. The latter is measured by the performance indices given above, while I feel the former should be assessed by a competence index. Drawing this distinction between competence and performance allows us to ignore *extraneous* factors related to the production of movement not under the influence of the controller. Furthermore, we can evaluate the controller's behavior in terms of variables more closely related to its internal workings. Of course, the only good controller is one which causes production of quality limb movements, but our success in finding such models and modifying existing ones is improved by measuring these internal variables in addition to terminal behavior. After all, do we want to casually reject a controller which produces very nearly the *right* control just because the manipulator behaves poorly under that control? (This can occur, for instance, when the mechanics of the manipulator include discontinuous non-linearities like stiction.)

One added feature of a competence measure is that it can be used to evaluate the entire 1.2 sec. duration of a movement while performance indices often must be restricted. The following index conveys information about the competence of the

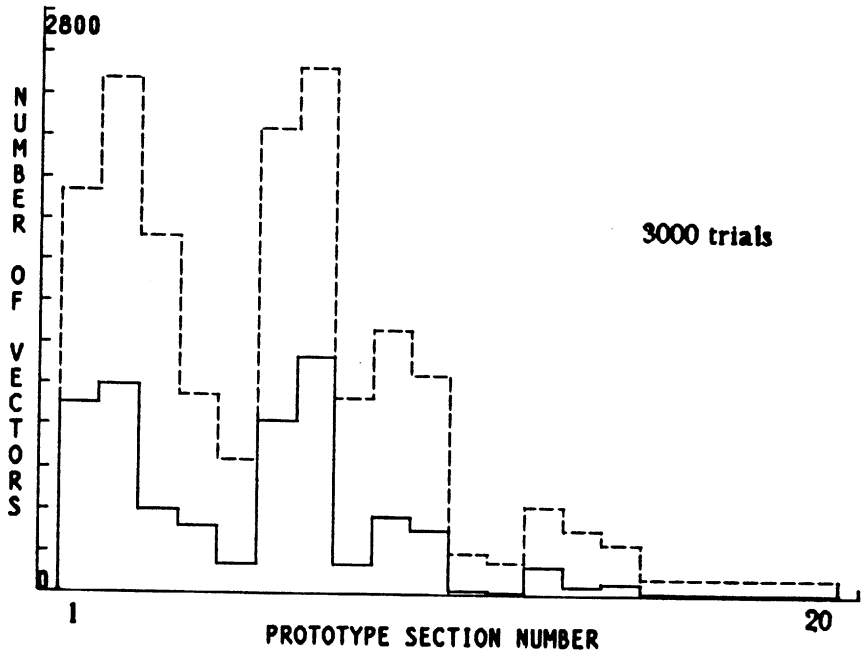


Fig. 4.8 The measurement vectors produced by 3000 practice trials are shown distributed on the time slices of practice prototype PR-11. Note that most of the data apply to the first 10 slices. (10 slices are 600 msec.)

system to generate motor plans while de-emphasizing the problems of production:

> Root-mean-square motor-current error - Same as RMS PE but the motor current error is found. (RMS MCE)

This index gives a measure of competence, but can only be used when the currents which will reproduce the prototype trajectory are known. This is normally the case only for prototypes generated from motor-current plans (see section on prototypes), but it was possible to estimate the currents for the prototypes used here. Since stiction plays an important role in the behavior of the Vicarm manipulator, there is not a unique motor current that will hold the arm stationary. For this reason this index often includes an artifactual constant term that does not improve with practice whenever a test movement incorporates a period during which the arm is stationary.

4.3.4 The Practice Algorithm

The program which generates practice movements is not actually a part of the controller. Since the behavior of the translator during testing is so intimately affected by the details of the practice algorithm, its operation is described here along with the implementation's other components. Once a movement is designated as the desired movement the practice routine takes the following steps:

> On each trial, for each section and joint, the Newton-Raphson method is used to choose a motor current predicted to achieve the desired acceleration. Only the previous two trials are used in making this prediction. Whenever the acceleration errors on the previous trial are within a set of limits for the section for all three joints, the motor currents for that section are not changed.

An example of seven consecutive practice trials are shown in Fig. 4.9 where the nature of progressive improvements is demonstrated. Since the duration of each practice

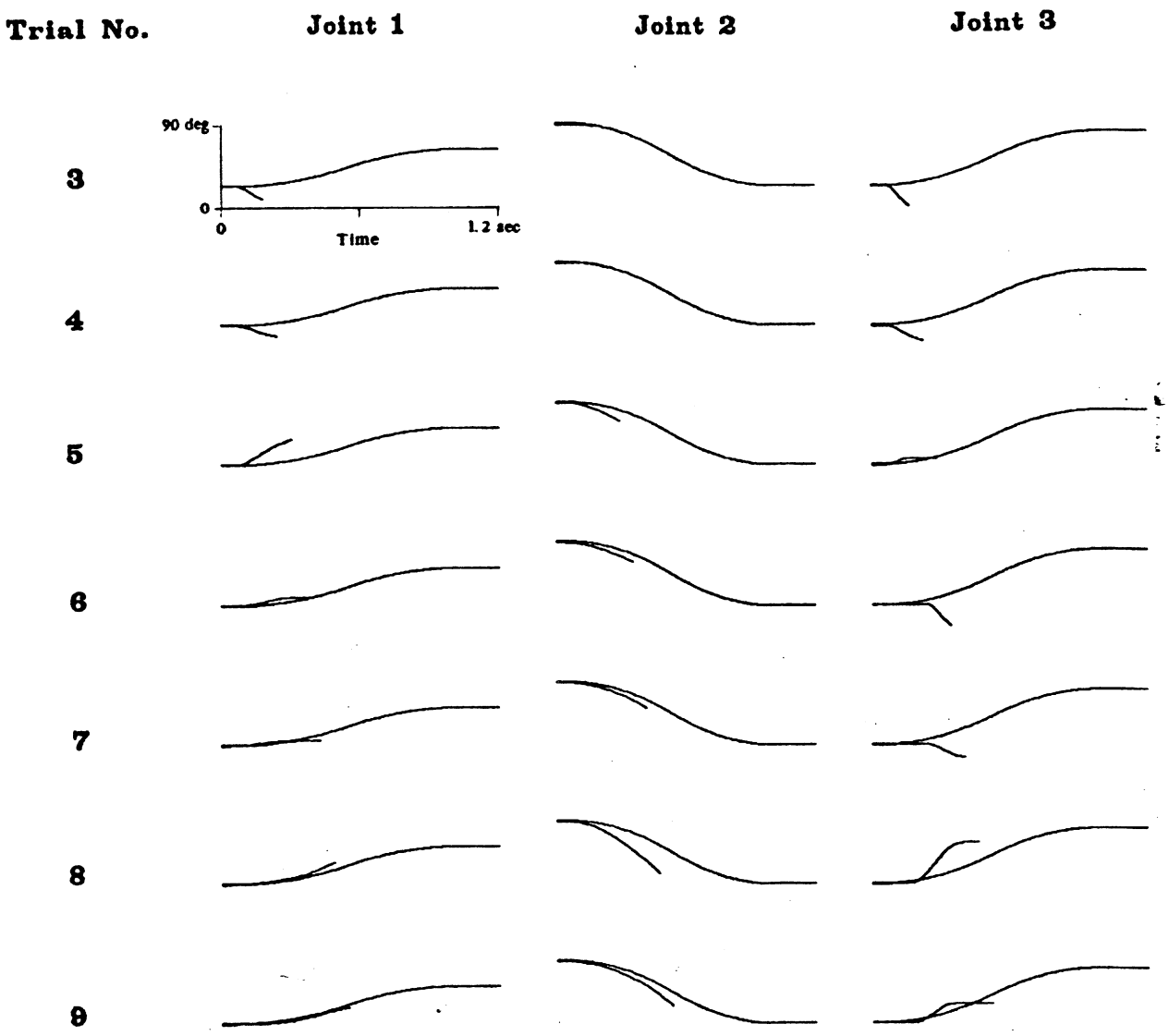


Fig. 4.9 Seven consecutive trials from a typical practice session are shown. Each curve includes the practice prototype, PR-11, and the attempted movement. At the beginning of each trial the manipulator is servoed to the correct starting position for the prototype. In order to avoid damage to the arm, most practice movements had to be terminated before completion.

movement varies, the number of trials of practice does not precisely indicate the amount of data generated for subsequent analysis.

It must be stressed that although the practice program relies on error correction procedures to ensure convergence, the learning displayed by the controller does not rely on error data in any way. The selection of this particular practice algorithm was made to simulate, in a simple way, the short term behavior of humans when practicing. Levine has preliminary data which suggest that a similar iteration method may be used by the cat when learning to make an optimal jump (Levine, 1975).

Originally it was assumed that the details of the practice algorithm would have little effect on the performance of the system, but this turned out to be quite false. The behavior of the translator depends on how many data are generated, how variable they are, and to which regions of the memory they apply. The rules which govern the former two of these factors are in direct conflict, at least for the simple algorithm used in this project. In order to produce useful data the practice algorithm has to produce movements which vary the output currents independently and by significant amounts. Once the practice algorithm converges upon an acceptable set of accelerations, (*acceptable* means within the acceleration limits; AL), the output currents are not changed for that section. Therefore, the same set of output currents are produced repeatedly -- not good for calculating mechanical constants.

These acceleration limits (AL) do insure, however, that once the correct values for the output currents are found, they are maintained so that subsequent sections can be practiced and processed. When the allowable error for a section is reduced, a

larger variety of movements is produced, but those sections late in the movement rarely receive enough attention to produce adequate measurement vectors. The effects of using AL = 50, 75, 90, and 115 are shown in Fig. 4.10. Limits of AL=75 provide a good tradeoff between variety of data and number of sections practiced.

This limiting effect can be overcome to some degree by practicing the movement in parts. A simple servo program moves the arm to the correct initial conditions for a point in the middle of the movement, after which the practice program continues with a normal practice movement. People are known to use such a strategy when they break a complicated movement into parts during learning (Welford, 1968; Cratty, 1970). Fig. 4.11 shows how this procedure can redistribute the effects of practice which would normally generate data primarily for the initial sections of the practice movement (Fig. 4.11a). When the servo is used to start the movement, sections in the middle of the movement also receive data (Figs. 4.11b, c, and d).

The choice of parameters has been described. In summary: Three manipulator joints are used for testing (N=3). Each dimension of the state space memory is divided into ten intervals, resulting in 10^6 hyper-regions (M=10). Movements are processed in terms of 60 msec. sections and eight measurements are inverted at a time (N'=8). When new data are stored into the memory they receive a weight of one tenth, and old data receive a weight of nine tenths (τ=10). When the memory is accessed, data from the desired hyper-region and from the first order neighbors are used in combination. The practice algorithm has been adjusted to generate moderately variable data while remaining near the prototype trajectory (AL=75).

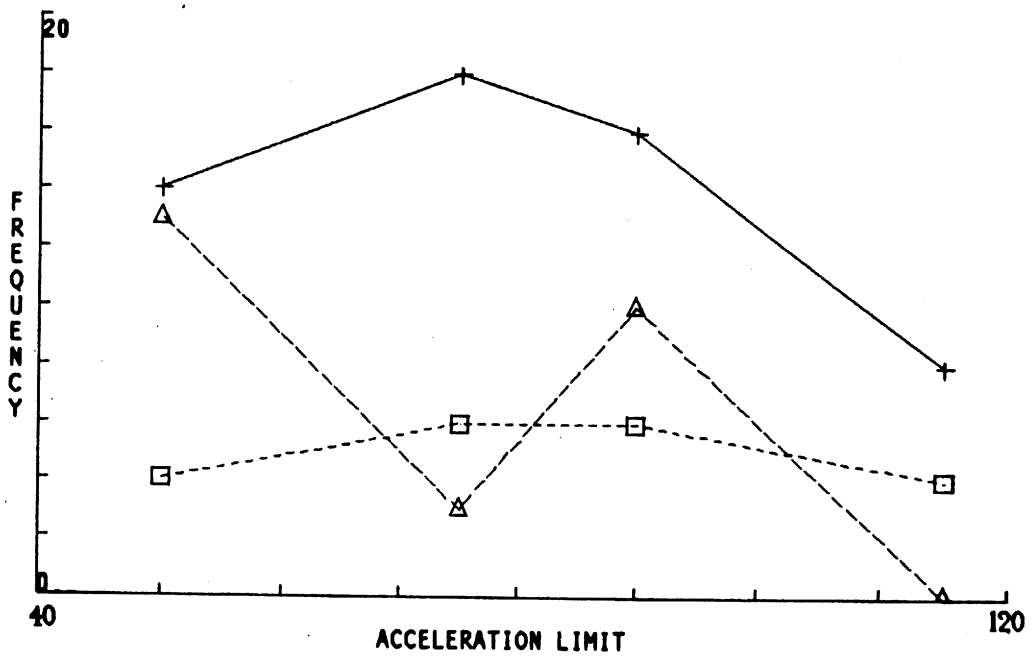


Fig. 4.10 Manipulating the practice algorithm's acceleration limit affects the algorithm's effectiveness. Three measures are plotted vs. the value of this limit: 1) Number of inverses computed for regions accessed directly by practiced prototype. (+) 2) Number of inverses computed for neighbors of practice prototype. (triangles) 3) Number of prototype sections for which data are provided. (squares) Acceleration limit, AL, is 75 for data in this report.

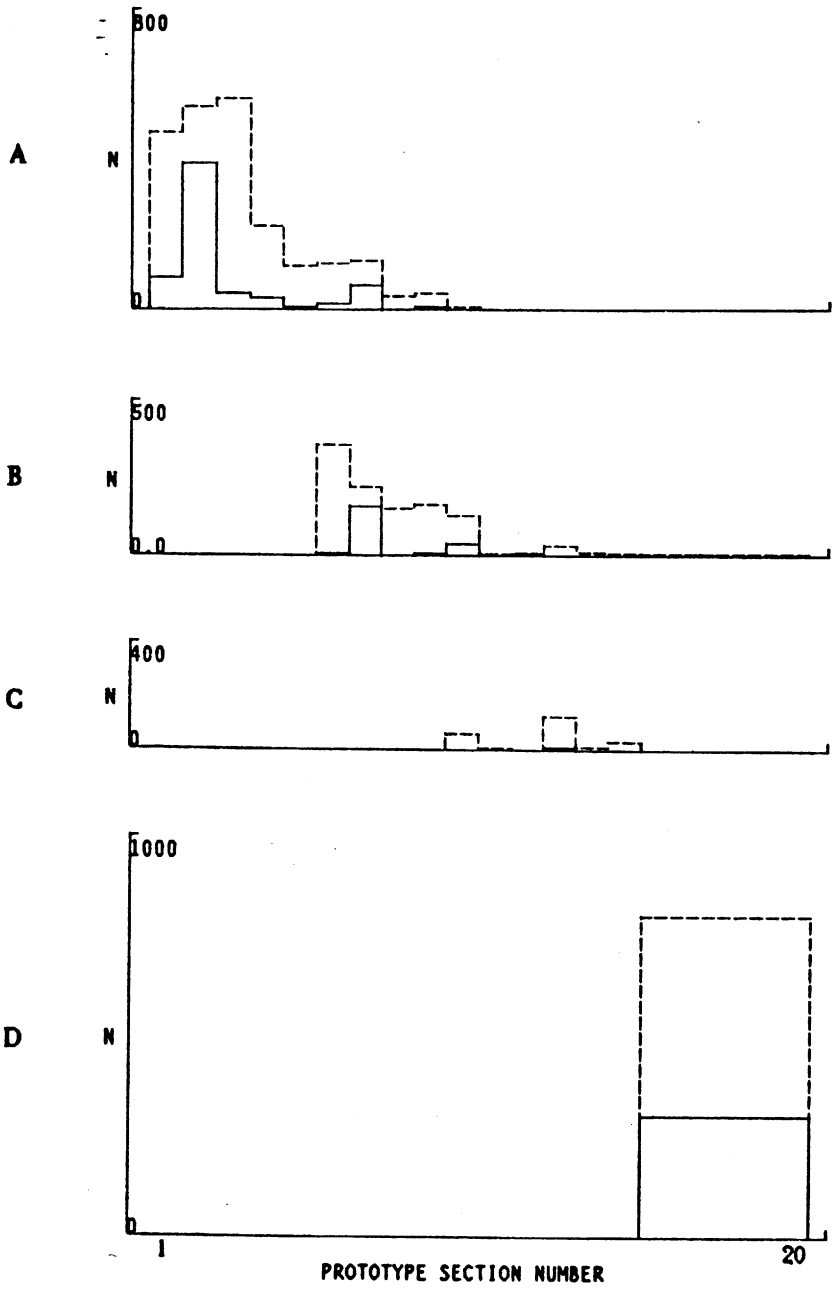


Fig. 4.11 Changing the starting section for a practice session redistributes the session's effect. For each histogram 250 practice trials were executed. At the start of each trial a servo routine was used so that the movements could start with time slice: a) 1, b) 6, c) 11, d) 16. The histograms show the number of usable measurement vectors generated for each section of the practice prototype, PR-1. The solid bars indicate vectors that apply to sections in the prototype, while the open bars indicate vectors that apply to first order neighbors (see text).

4.4 Test Procedures

During each practice session, the practice program, using one of the prototypes as a goal movement, generates 100 practice movements. At the end of each of these 100-trial blocks, the movements are processed by the translator, creating data which describe the mechanics of the manipulator. The prototypes are then used to plan test movements using the tabular equations of motion and the data from the state space memory. After the test movements are executed and recorded the performance and competence of the translator is measured. The results are used to construct learning curves which plot the values of a performance index as a function of experience. This procedure lies at the heart of all data reported in this thesis. Testing for generalization and adaptation, however, requires additional procedures.

To show generalization, sets of prototypes are generated which vary systematically in one or more properties. One of the set is chosen as the practice prototype and is used as the target of subsequent practice sessions. All members of the series are tested after each practice session. In addition to presenting sets of learning curves a learning index has been defined which allows learning curves to be compared:

$$LI = \frac{\Sigma(e_0 - e_i)}{\Sigma e_0}$$

where:
- e_i is the RMS FPE for the i'th test movement
- e_0 is the pre-training performance value.
- Σ is the sum from $i=1$ to $n-1$
- n is the number of test movements

LI=0 if no learning takes place and LI→1 with total, immediate learning. This index is

most useful when the learning curve is roughly monotonic. Generalization curves are composed by plotting LI for each test prototype of a series.

Adaptation is tested by establishing a baseline of performance with one prototype, after which the disturbing manipulation is made. Then additional practice is processed and the post-manipulation adaptation curve is plotted along with an indication of the pre-adaptation baseline performance.

5 Results and Discussion

5.1 Control and Learning

Position trajectories for a series of test movements made during a learning session are shown in Fig. 5.1a, each separated by 500 practice trials. The prototype which described the desired movement to the translator, PR-11, is also plotted. Each succeeding test movement is a better replica of the desired movement than the previous one, and the last movement shown is very similar to the prototype. Most of the residual error after 3000 practice trials is caused by deviations from the desired trajectory of joint three. Stiction forces in this joint are especially large, and are probably responsible for the observed deviations. Fig. 5.1b shows velocity trajectories for the same set of movements. The velocity deviations for joint 3 shown here support the stiction explanation for these errors.

The gradual improvement in performance indicated by Fig. 5.1 is expressed in quantitative form in Fig. 5.2. The performance index, root-mean-square final-position error (RMS FPE), was evaluated for each test movement and plotted against the number of practice trials processed by the system. The learning curve shows a rapid initial improvement with subsequent apparently asymptotic behavior.

Acquisition data for another test movement, PR-12, are shown in Figs. 5.3 and 5.4. (Same practice data as above.) Stiction was not a problem here because the motor-currents planned to produce this higher-velocity movement were adequate to overcome the static force on joint 3. (See last row of Fig. 5.3.) The learning curve

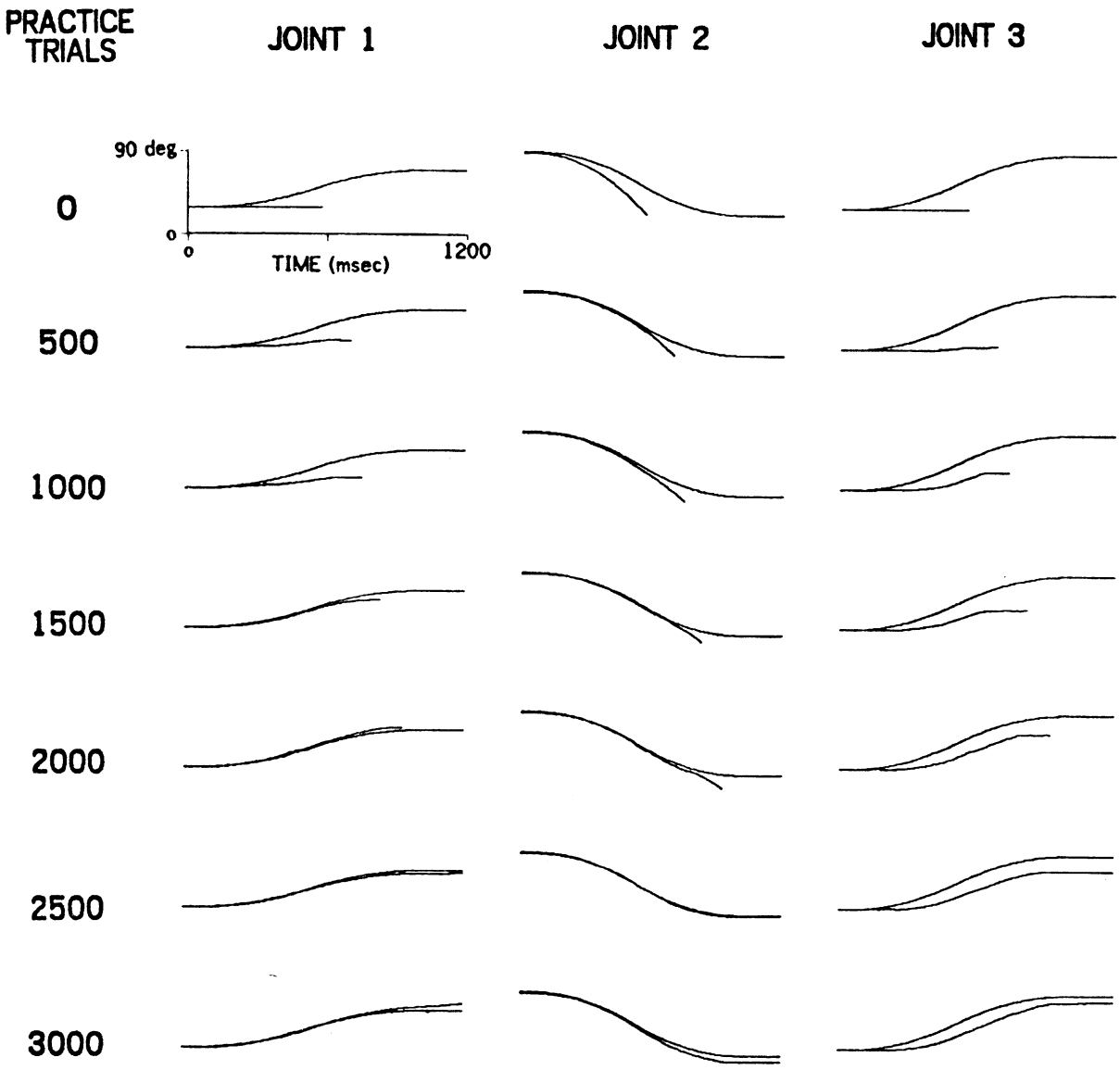


Fig. 5.1a Attempts to replicate prototype PR-11 are shown. Each is separated by 500 practice trials and is plotted along with the desired trajectory. (Position trajectories.)

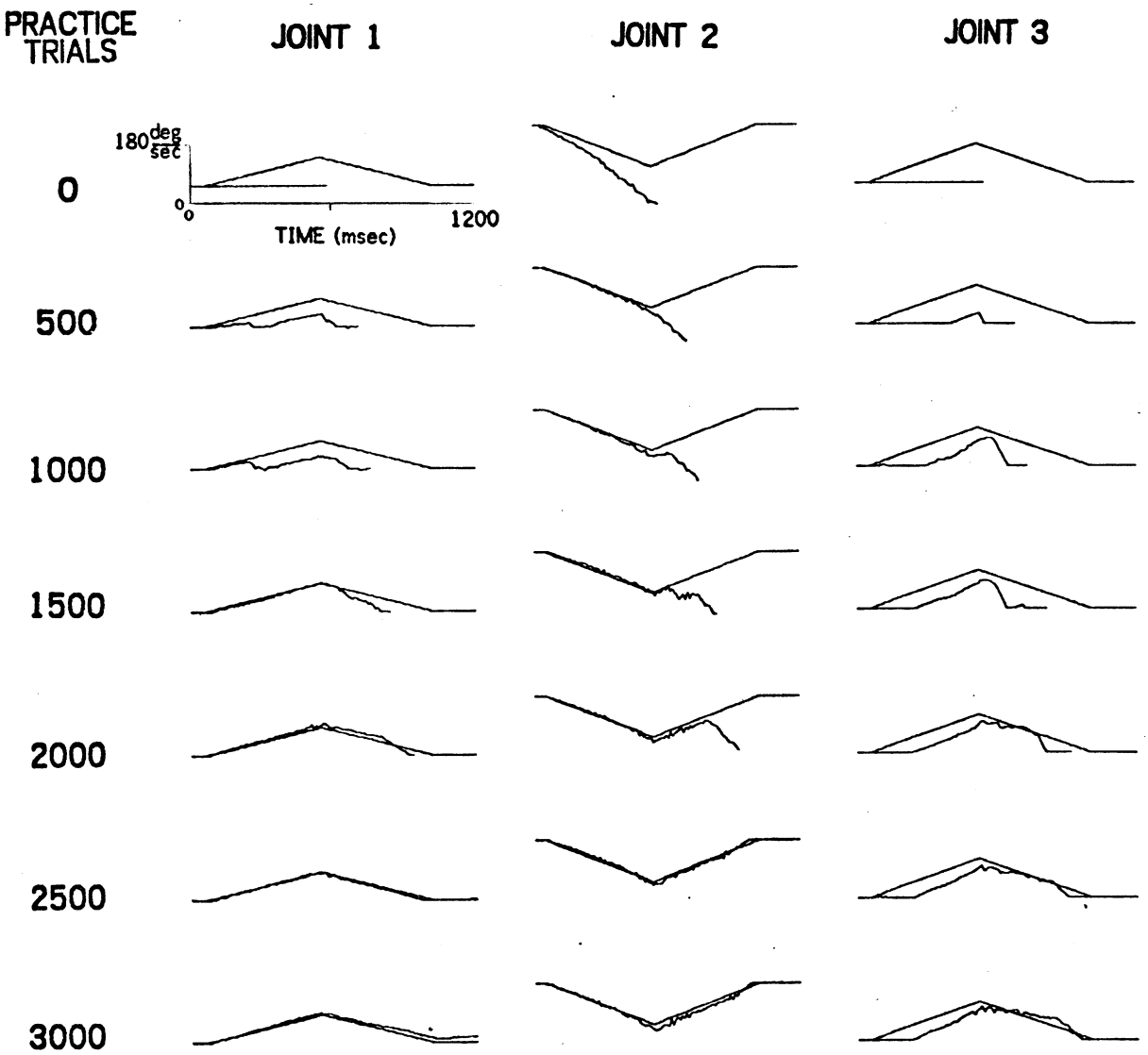


Fig. 5.1b Same movements as in Fig. 5.1a, but velocities are plotted. The effects of stiction in joint 3 are apparent.

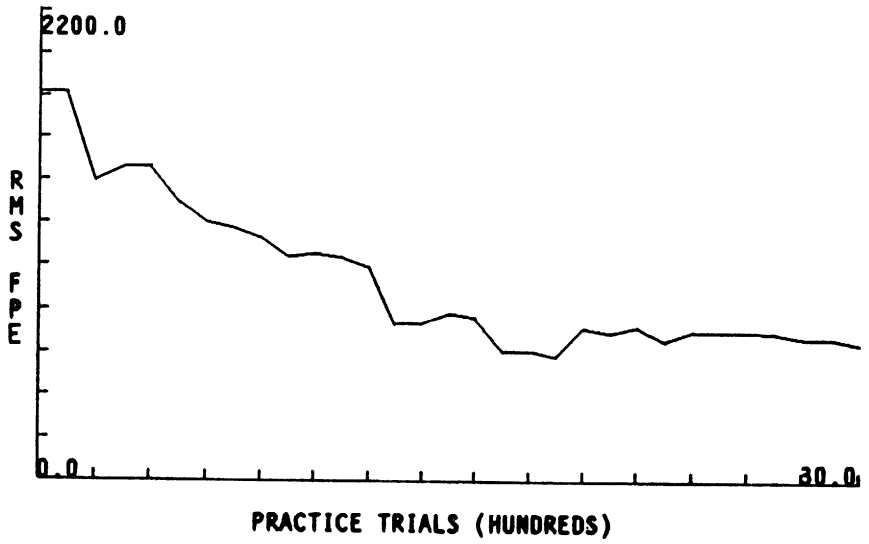


Fig. 5.2 After every 100 practice trials a set of test movements are produced and the performance indices are applied. This learning curve shows the performance index, RMS FPE plotted against number of practice trials. These data summarize Fig. 5.1.

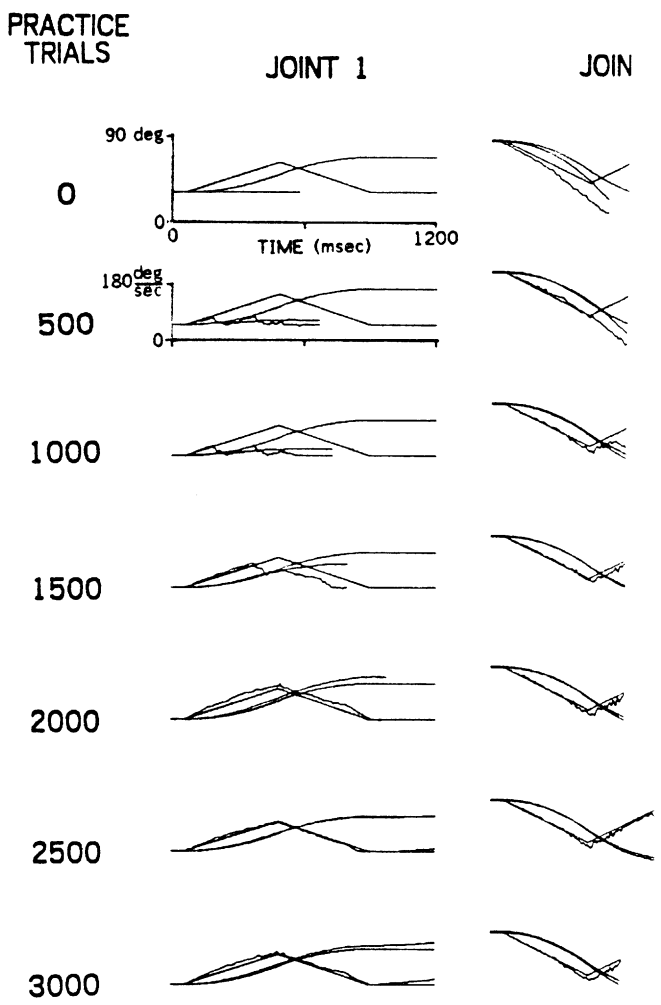


Fig. 5.3 A set of test movements showing (Note: PR-11 was the practice movement.) the same axes.

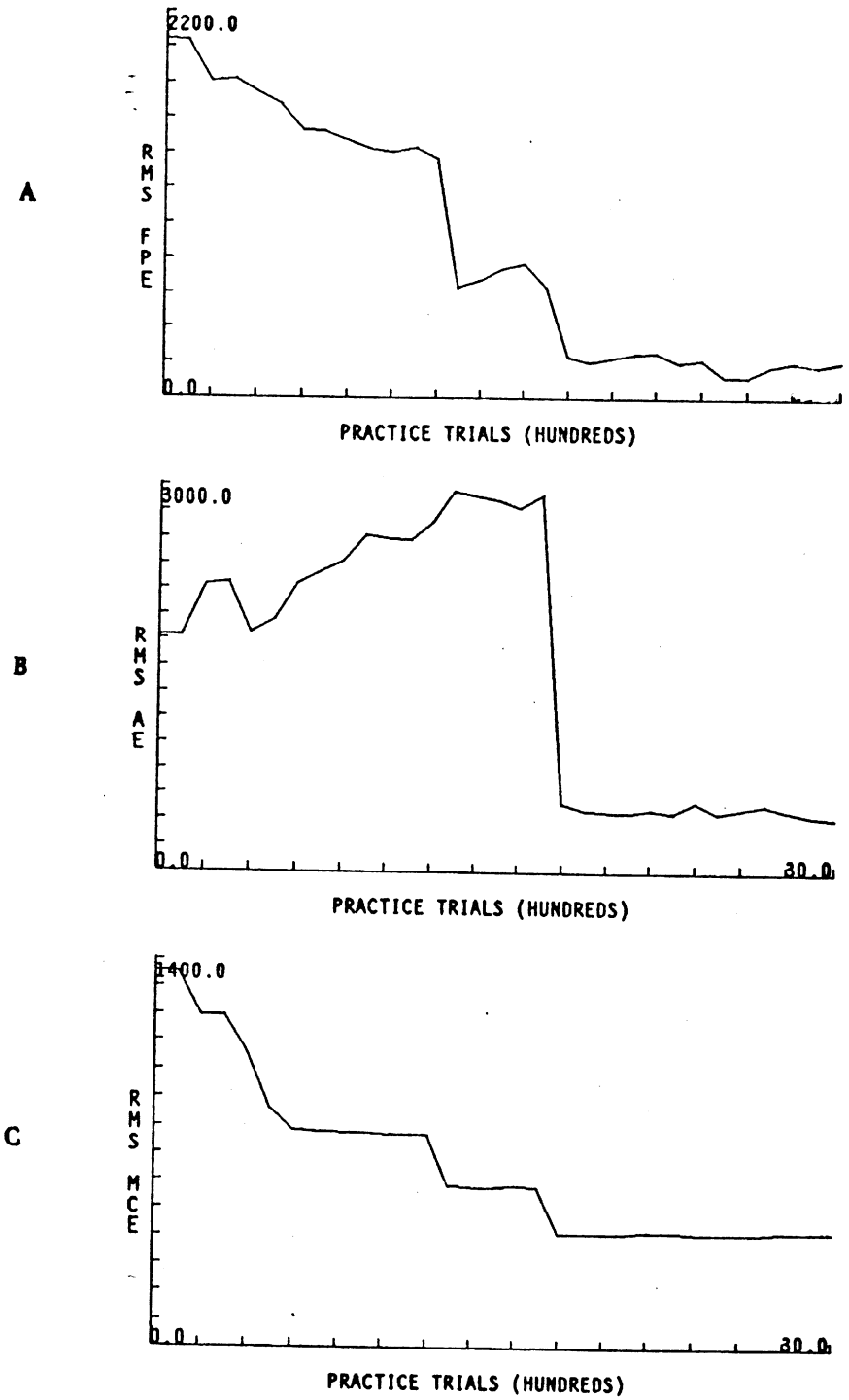


Fig. 5.4 These learning curves summarize the data of Fig. 5.3. Two performance indices and the competence index were used. A) RMS FPE, performance. B) RMS AE; Note the irregular, non-monotonic behavior. C) RMS MCE, competence.

shown in Fig. 5.4a presents a more dramatic example of acquisition. The abrupt improvement occuring after 1300 practice trials is characteristic of the SSM's learning behavior. These sudden improvements occur when new inverses are found for regions of the memory for which data did not previously exist. They are also important because they introduce ambiguity into the asymptotic nature of a learning curves. Though the learning curve for PR-11 looks asymptotic after 3000 trials, (see Fig. 5.2), a sudden improvement might occur at any time with additional practice.

One apparently peculiar result is that a set of test movements may show that the RMS FPE and RMS PE are converging nicely to small values while the RMS AE behaves somewhat disorderly. (See Fig. 5.4b.) Although one might suppose that these two measures are closely linked, some thought shows that small error in acceleration near the beginning of a movement may contribute to very large position errors, while acceleration errors near the end of a movement may not influence position error at all. On the other hand, there may be no change in acceleration error, or even a net decrease, while position errors have become quite large.

Fig. 5.4c shows that during the practice session the motor currents planned by the translator approach those which are known to produce the desired movement. The competence index, (RMS MCE), was used to produce this learning curve. As noted earlier, this curve may be affected by a constant error because the friction compensation term is not unique when the joint is stationary.

Gravity compensation could be tested separately from dynamic terms by using a movement which has no accelerations. Translation of this degenerate trajectory results

in an output determined by the non-inertial terms stored in the memory, the K terms. Since the velocities for such a movement are all zero, Coriolis and frictional forces are zero and gravity compensation is the only factor. A learning curve for this *null* or *hold-still* movement, PR-17, is shown in Fig. 5.5. The speed and completeness of learning is unrivaled by any other curve presented in this thesis.

Figures 5.1 through 5.5 provide substantiation for our basic claim; the state space model can acquire control of a limb-like mechanical device by processing data collected during movements of that device, without the use of error information.

5.2 Generalization

In addition to learning to perform new movements when they are practiced, the SSM should transfer training, or generalize from practiced movements to movements which have never been practiced, provided they are sufficiently similar. Initial attempts to verify this claim were only partially successful, though subsequent data provided more complete support for these ideas. Both sets of results illustrate properties of the state space model vis-a-vis generalization.

5.2.1 The First Generalization Test

Fig. 5.6 plots the set of prototype trajectories first used to test for generalization. The movements in this set consisted of a graded series, each having the same initial and final position, but differing in duration, and therefore velocity. Prototype PR-11 was practiced and the normalized learning curves of Fig. 5.7 plot acquisition for each prototype of the set throughout a session of 3000 practice trials.

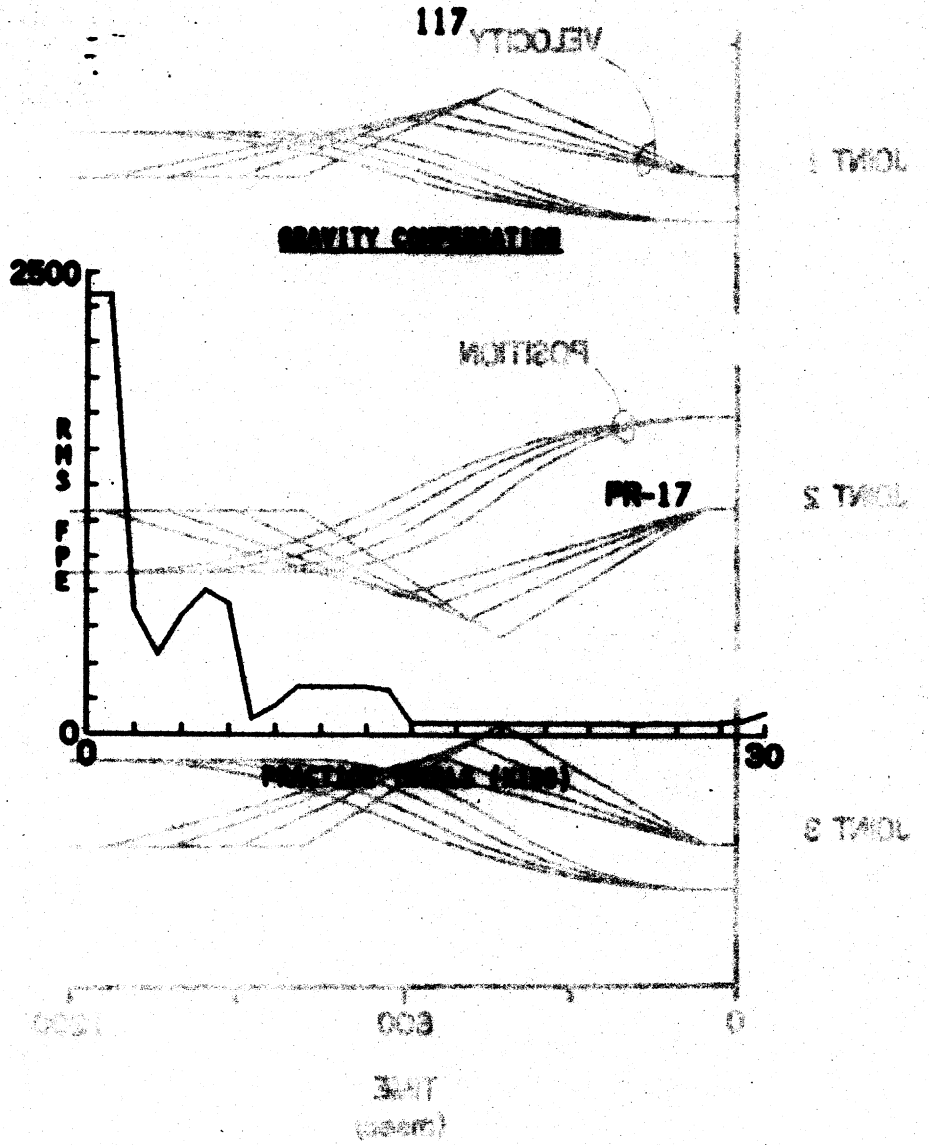


Fig. 5.6 A hold-out prototype, PR-17, has used to test acquisition of gravity compensation.

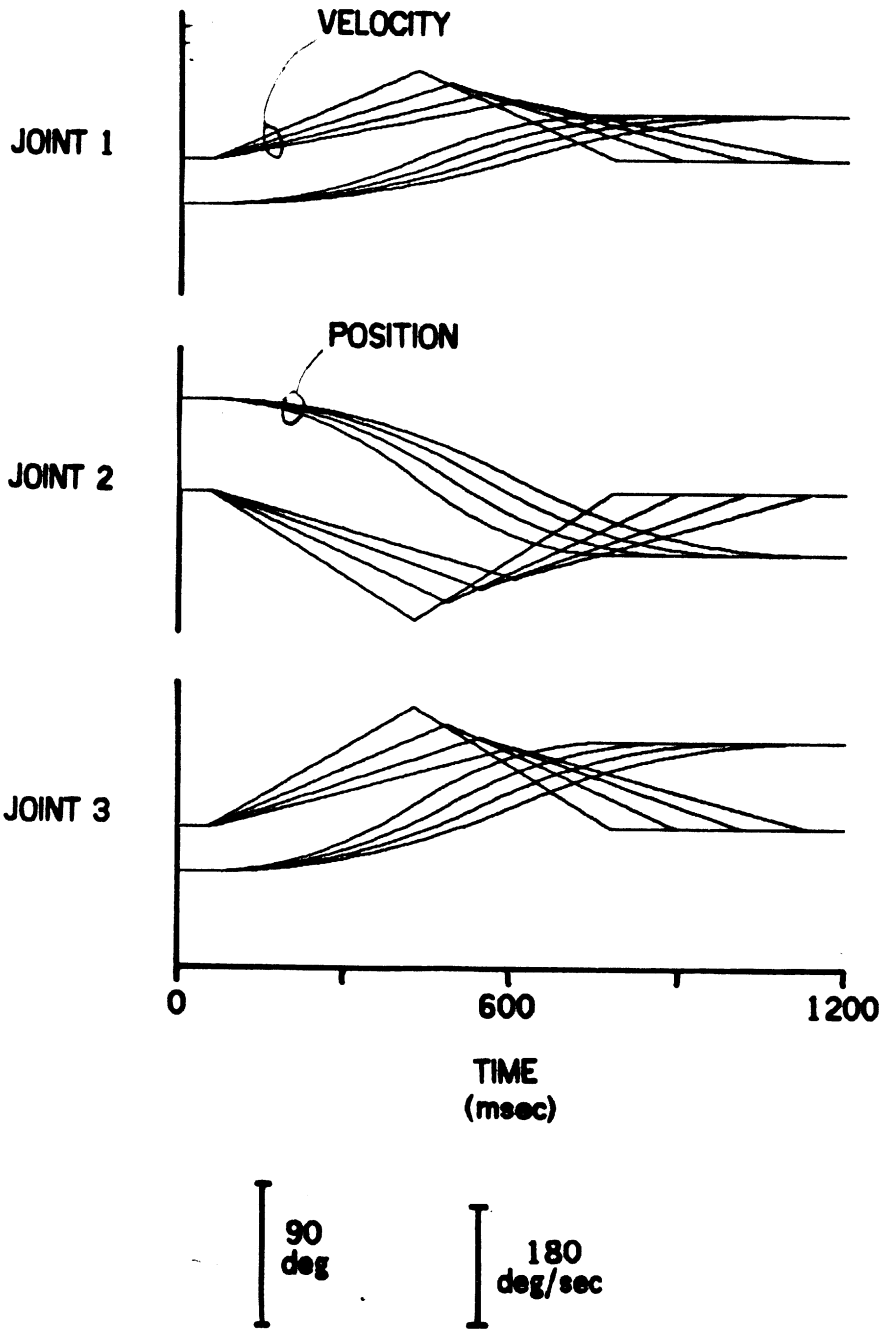


Fig. 5.6 These prototypes, PR-10, PR-11, PR-12, and PR-13, have the same starting and ending positions, but vary systematically in duration, (1080, 960, 840, and 720 msec. respectively). They were used in initial assessments of generalization.

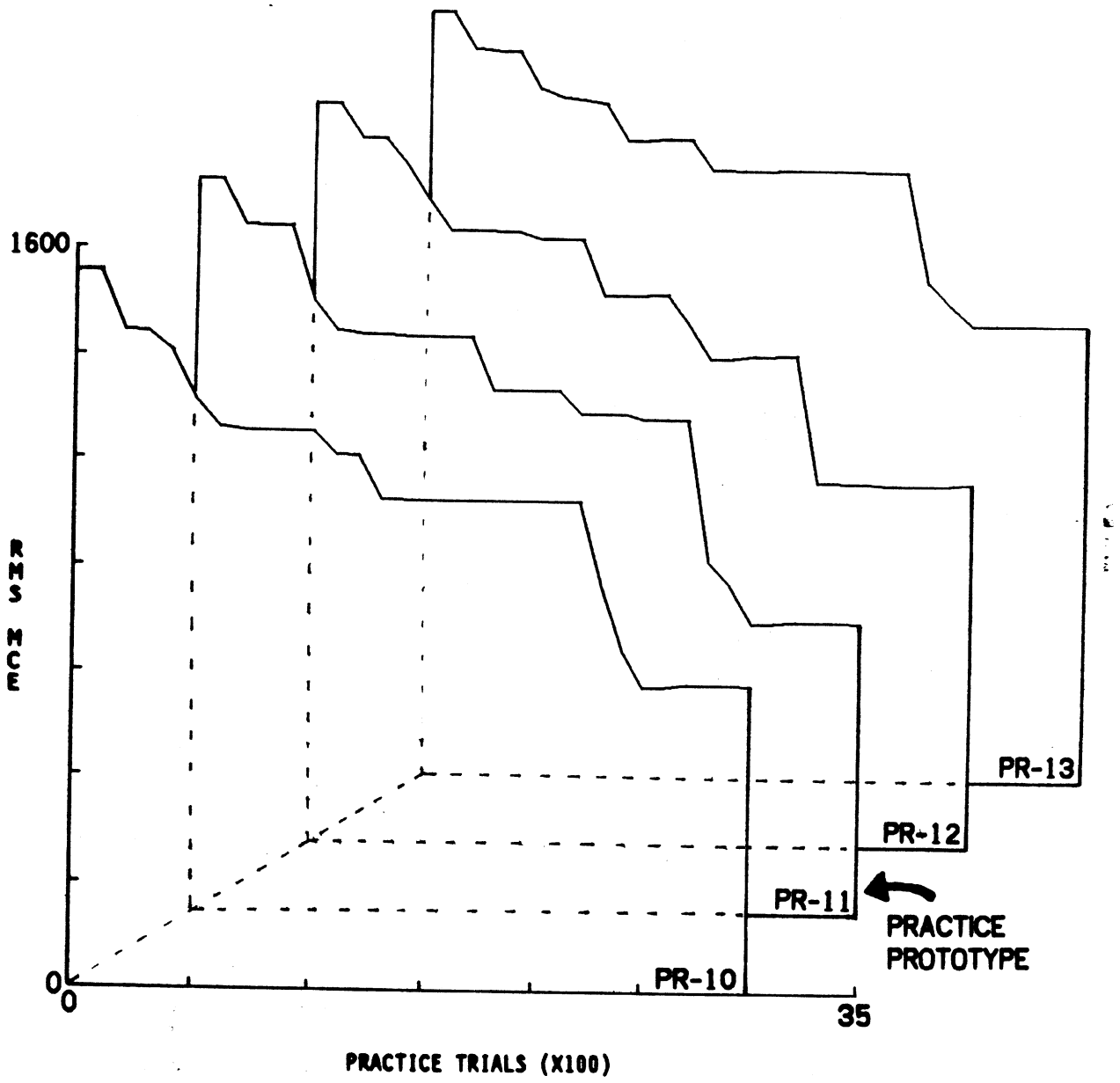


Fig. 5.7 Prototype PR-11 was practiced and each member of a graded set of prototypes (shown in Fig. 5.6) was tested. The data for each curve are normalized so relative improvement is shown.

Though transfer was extensive, (every prototype showed substantial improvement), a systematic deterioration of performance for dissimilar prototypes did not occur. In fact, Prototype PR-12 showed substantially better learning than the practice prototype, PR-11. In order to quantify these data the learning index, LI, was applied to the set of learning curves and the resulting generalization are plotted in Fig. 5.8 (triangles). The data show an unsystematic variation in generalization, despite the use of test movements that vary systematically in their relationship to the practice prototype.

These data satisfy the strict definition of generalization, but they are peculiar in certain respects. Normally, one expects a gradual deterioration in performance as the test movements become more and more different from the practiced movement. Such a systematic variation was not found among these performance curves. On the other hand, practice of one movement is clearly shown to improve performance of the others.

Perhaps I should clarify why I am unhappy with results which indicate that the controller does not generalize poorly among dissimilar movements. After all, the goal is to find efficient solutions for learning, and a controller which performs well on a number of movements with little practice is desirable. There are two very general principles which govern such solutions.

Principles of Generalization

1) A controller should maximize the use of available data by providing access to them whenever possible.

2) A controller should minimize the misuse of data by restricting access to them whenever necessary.

Since the dynamic behavior of the manipulator varies smoothly throughout state space,

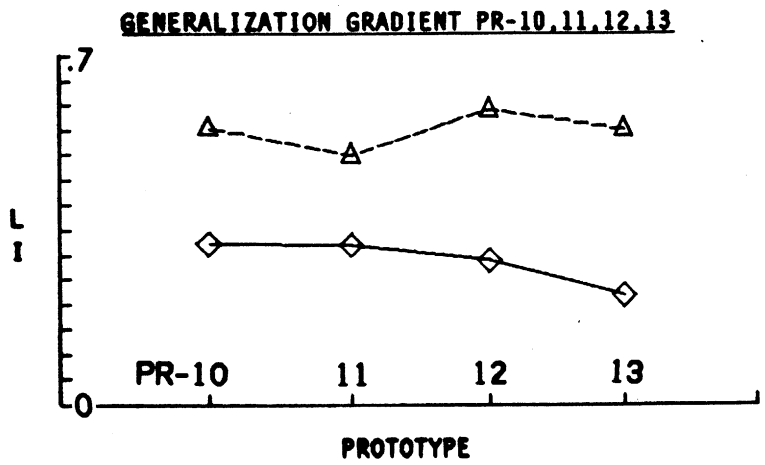


Fig. 5.8 These generalization curves were produced by applying the learning index, LI, to sets of learning curves. (Triangles) The data from the normalized learning curves, Fig. 5.7, are shown. There is no systematic generalization. (Diamonds) The competence learning curves of Fig. 5.9 were analyzed. The generalization curve is more systematic, but differences between prototypes are small (see text).

data generated for one region of the state space memory can be made available when planning movements through other, nearby parts of the space. But this *sharing* of data cannot go too far or the state dependent variations in mechanical properties will lead to the generation of very poor trajectories.

Each of these principles should induce a generalization gradient in the controller's behavior. The first because inappropriate use of available data will produce bad trajectories. The second because unavailability of data will produce bad trajectories. For these reasons our data must show a gradient in performance in order to verify our analysis of generalization.

One factor, closely related to these principles may be held responsible for the lack of observed regularity. Each trajectory was replicated with roughly equal precision -- replication of prototype PR-13 was as good as PR-11. Since the range of movements chosen for the tests was relatively small, none were sufficiently different from the practice movement to involve substantially changed mechanical behavior. Nor were much data required from unfilled regions of the state space memory. This problem may be solved by selecting test movements which span a larger range of movement space.

Another factor, related to the individual characteristics of certain trajectories, caused some to be replicated much more faithfully than the practice movement -- replication of prototype PR-12 was better than PR-11. A movement may be *easier* or *harder* to learn and replicate for a number of reasons. It may have fewer low velocity components, require more data from the memory, bear a particular relationship to the

practice algorithm, etc. Some of these factors, those related to production rather than learning, can be eliminated by examining competence rather than performance. (See Section 4.3.3.1.)

Fig. 5.9 plots the competence learning curves for the set of tests previously shown in Fig. 5.7, and Fig. 5.8 (diamonds) provides a quantitative summary. The results are somewhat more orderly for these data: The practice prototype is near the peak of a unimodal generalization curve, however, differences between prototypes are fairly small. The differences in form between the competence and performance generalization curves shown in Fig. 5.8 verifies the existence of prototype-specific *easiness factors.*

The lack of substantial differences between pairs of prototypes for performance and competence measures, indicates a poor choice of test prototypes. Indeed, practicing one movement facilitated performance of others, but the expected gradual deterioration of performance as the test trajectory varies was not observed.

5.2.2 The Second Generalization Test

This time each prototype trajectory in the series of five was chosen so that successive members shared fewer regions of the state space memory with the practice prototype than the previous ones. All members of the series had the same ending position and duration, but they varied in starting position. (They also varied in velocity.) The members of the set are plotted in Fig. 5.10. In order to control for gradients which might result from prototype-specific properties, two learning sessions were run.

In the first of these sessions prototype PR-20 was practiced while all five

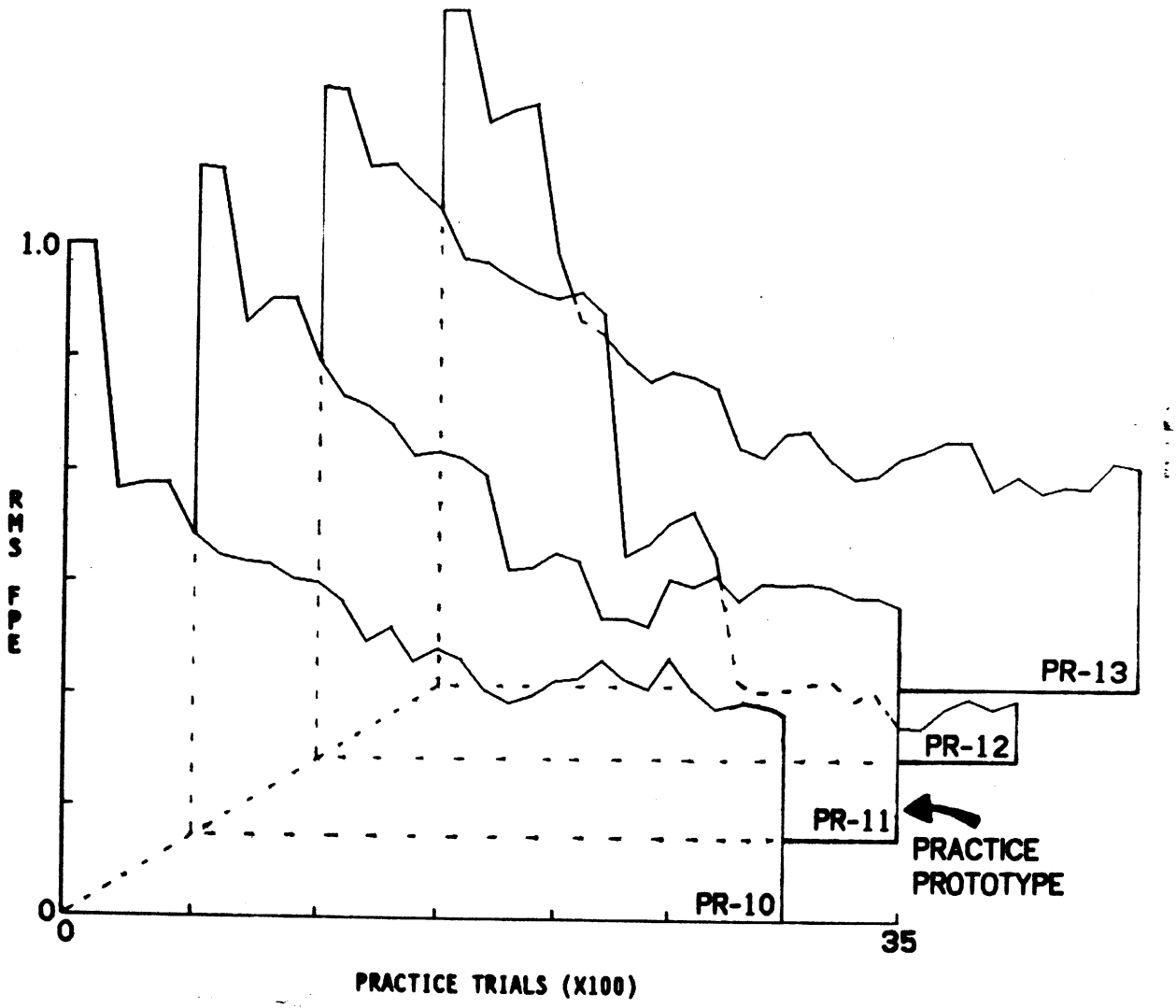


Fig. 5.9 The competence index, RMS MCE was applied to the learning data used to generate Fig. 5.7. The resulting competence learning curves are plotted here. Performance difficulties are eliminated here resulting in somewhat more systematic generalization data. Also see Fig. 5.8 (diamonds).

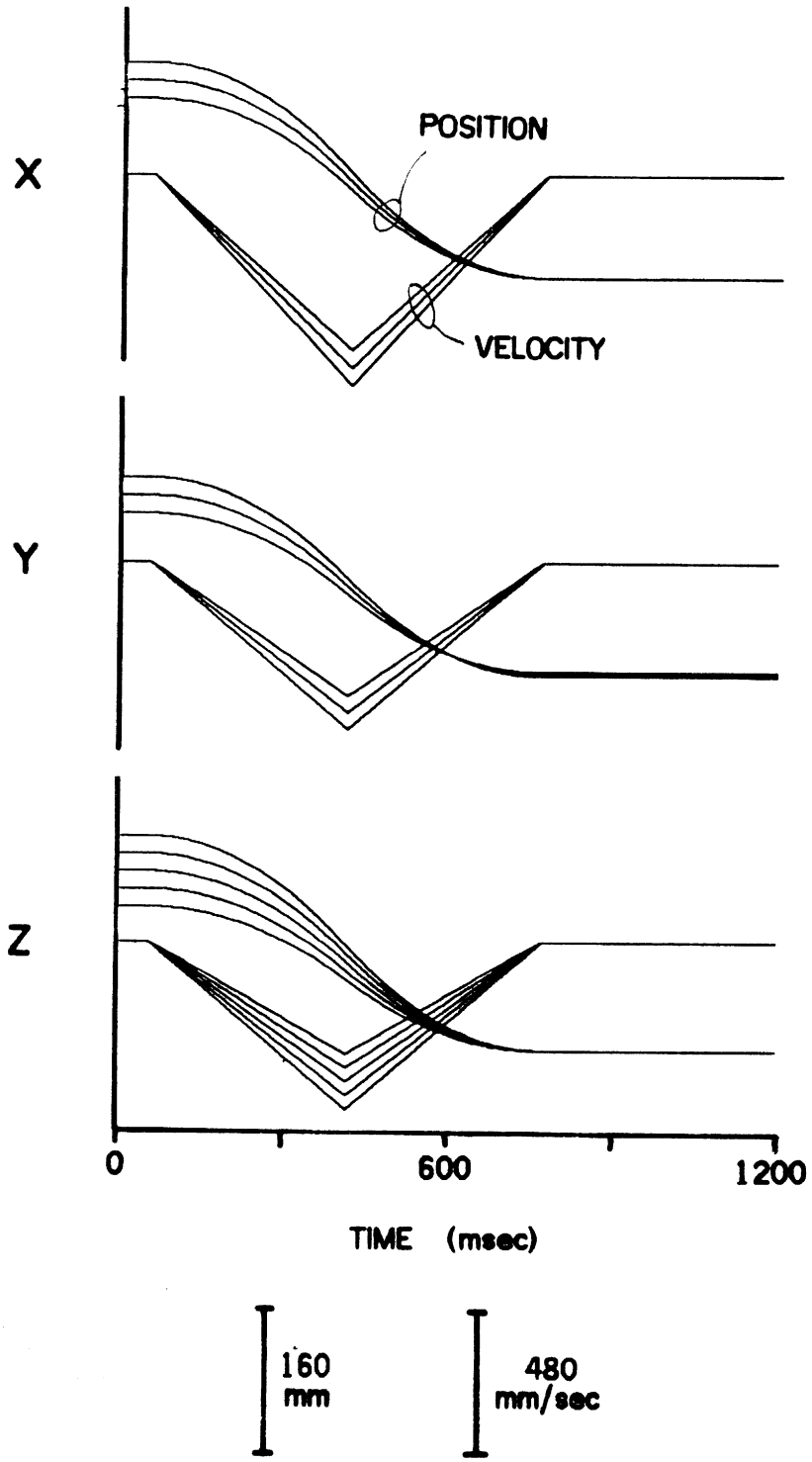


Fig. 5.10 These prototypes, PR-20, PR-21, PR-22, PR-23, and PR-24, have the same ending positions and durations, but vary systematically in starting position,((.285m,-.145m,.12m), (.265,-.145,.1), (.245,-.145,.3), (.245,-.165,.6), (.245,-.185,.4), respectively). They were used in additional tests for generalization.

members of the series were tested (PR-20 through PR-24). The learning curves are shown in Fig. 5.11a. In the second session the same test prototypes were used, but PR-23 was practiced. (See Fig. 5.11b.) The data from these two sessions are summarized by the generalization curves given in Fig. 5.12. The gradients show just the type of behavior we have come to think of as typical of human performance (Mednick, 1964): Practiced movements are improved most and similar movements less. The shift of peak learning with change of practice prototype, (see Fig. 5.12), rules out the possibility that extraneous effects or the choice of test movements were responsible for the observed gradients.

5.2.3 Type II Generalization

So far only one type of generalization has been discussed -- that which is typified by improved performance of one task after practice of a different, but similar task. Let us call this Type I generalization. We now consider two other forms of generalization, called here Type IIa and IIb. It may be an overstatement to call theme *types* rather than *measures*, but the underlying mechanisms are somewhat different.

When a movement is practiced after a baseline of performance has been established for a similar movement, one can expect rapid learning. (An example might be the rapidity with which the tennis player learns the correct stroke for squash.) There are two factors which contribute to such an effect. First, the initial level of performance will often be better than exhibited by the naive system. Any given level of proficiency will then take less time to achieve. I will refer to this as Type IIa generalization though it is really only the result of previous Type I generalization.

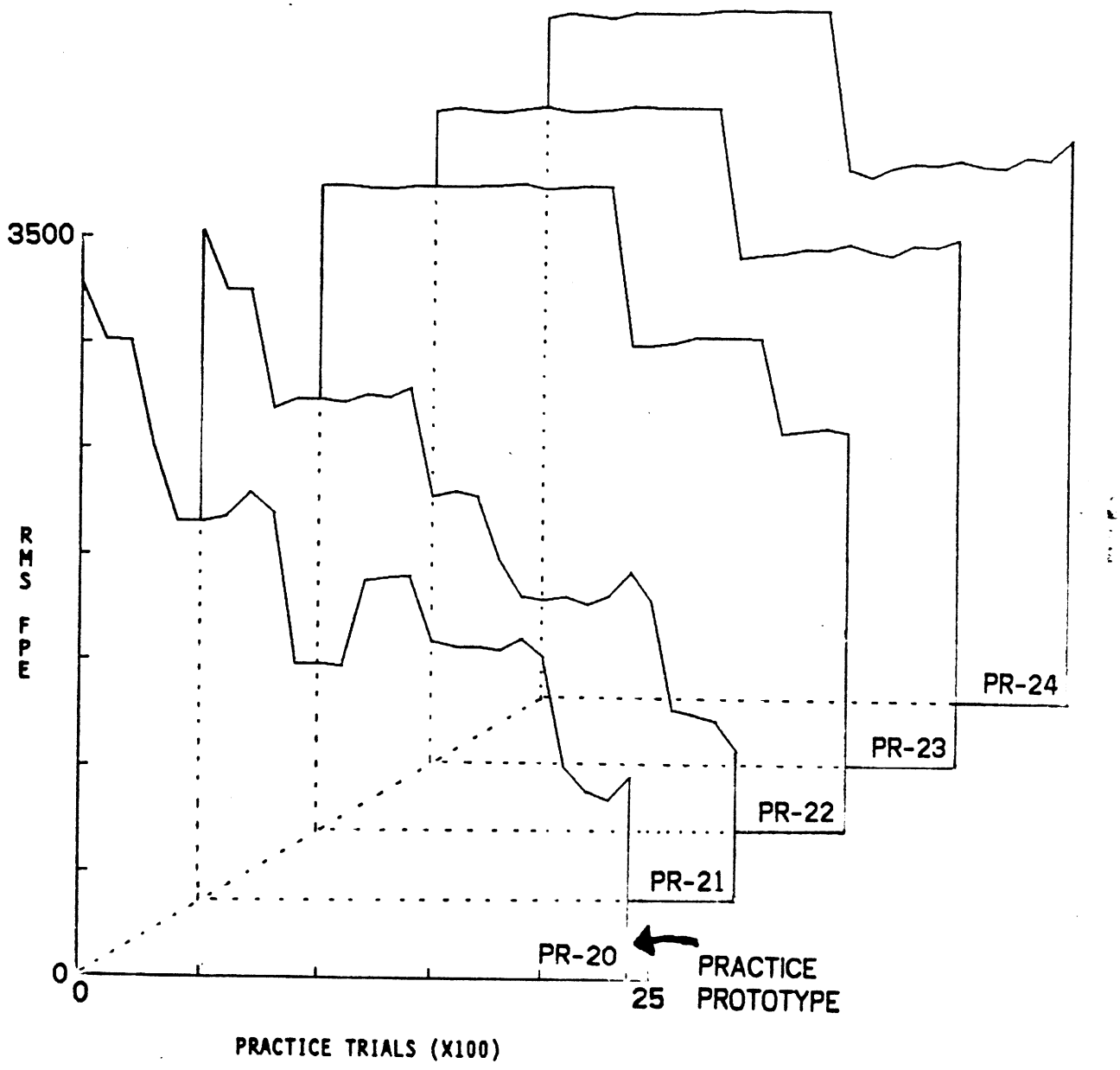


Fig. 5.11a Prototype PR-20 was practiced and prototypes PR-20 through PR-24 were tested. The resulting learning curves are shown. The practice prototype shows the most improvement.

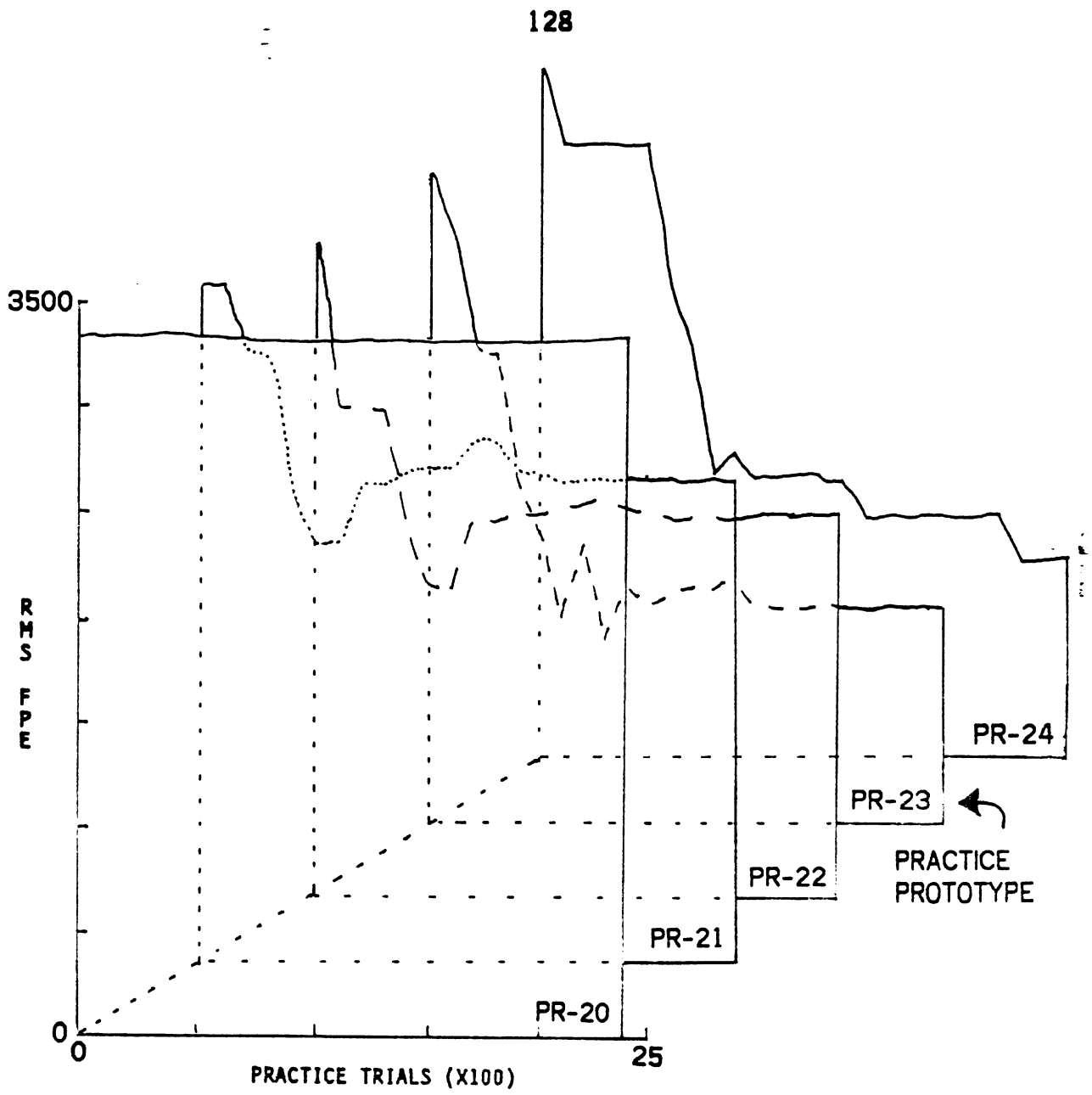


Fig. 5.11b Prototype PR-23 was practiced and, once again, prototypes PR-20 through PR-24 were tested. The resulting learning curves are shown. The practice prototype shows the most improvement.

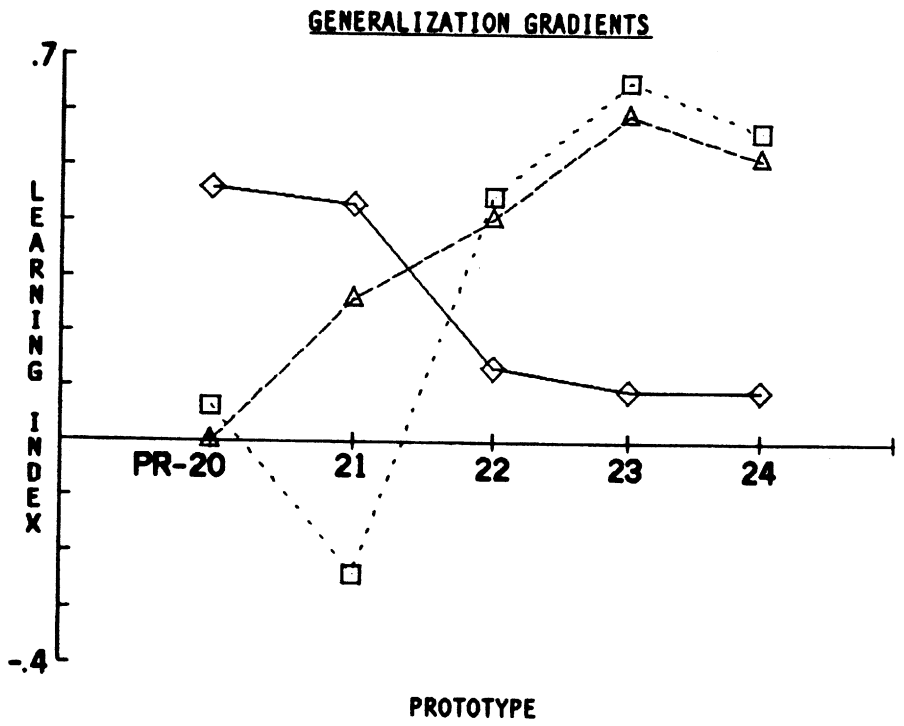


Fig. 5.12 Generalization curves for three conditions are shown. Diamonds) Prototype PR-20 was practiced (Fig. 5.11a). Triangles) Prototype PR-23 was practiced (Fig. 5.11b). Squares) After a baseline of 2400 trials of practice of prototype PR-20, 2400 additional practice trials of prototype PR-23 were executed. No Type IIb generalization is revealed. Prototype PR-21 shows retroactive inhabition.

Secondly, improvements may actually be more rapid: Since inversions only take place when the required number of measurement vectors are available from the temporary buffer, the presence of un-inverted, but accurate measurement vectors should facilitate learning. This will be called Type IIb generalization.

The practice data generated for the previous Section, 5.2.2, were re-used to test for Type II generalization. After establishing a baseline of 2400 trials with practice prototype PR-20, 2400 additional trials were generated using PR-23 as the target. Fig. 5.13 shows the set of learning curves generated under these conditions. Fig. 5.14, an explicit comparison of acquisition of prototype PR-23, with and without previous practice, clearly demonstrates Type IIa generalization. Performance of PR-23 is initially and subsequently better when the system has previous experience than when it does not. The figure shows, however, that the rate and time-course of learning for PR-23 are almost identical to the case when no previous data were present; no Type IIb generalization. The generalization curve depicted in Fig. 5.12 (squares) summarizes the data of Fig. 5.13. It further argues against Type IIb generalization since the index used, LI, which only measures learning relative to the starting value, yields about the same value for both cases. Type IIb generalization is not demonstrated.

Fig. 5.13 reveals another interesting process. Practice of PR-23 causes a slight deterioration of performance for PR-21. This result is an indication that the second principle of generalization is at work, (see Section 5.2.1), and can lead to *retroactive inhibition*. Data generated for PR-23 were made available to PR-21 through the

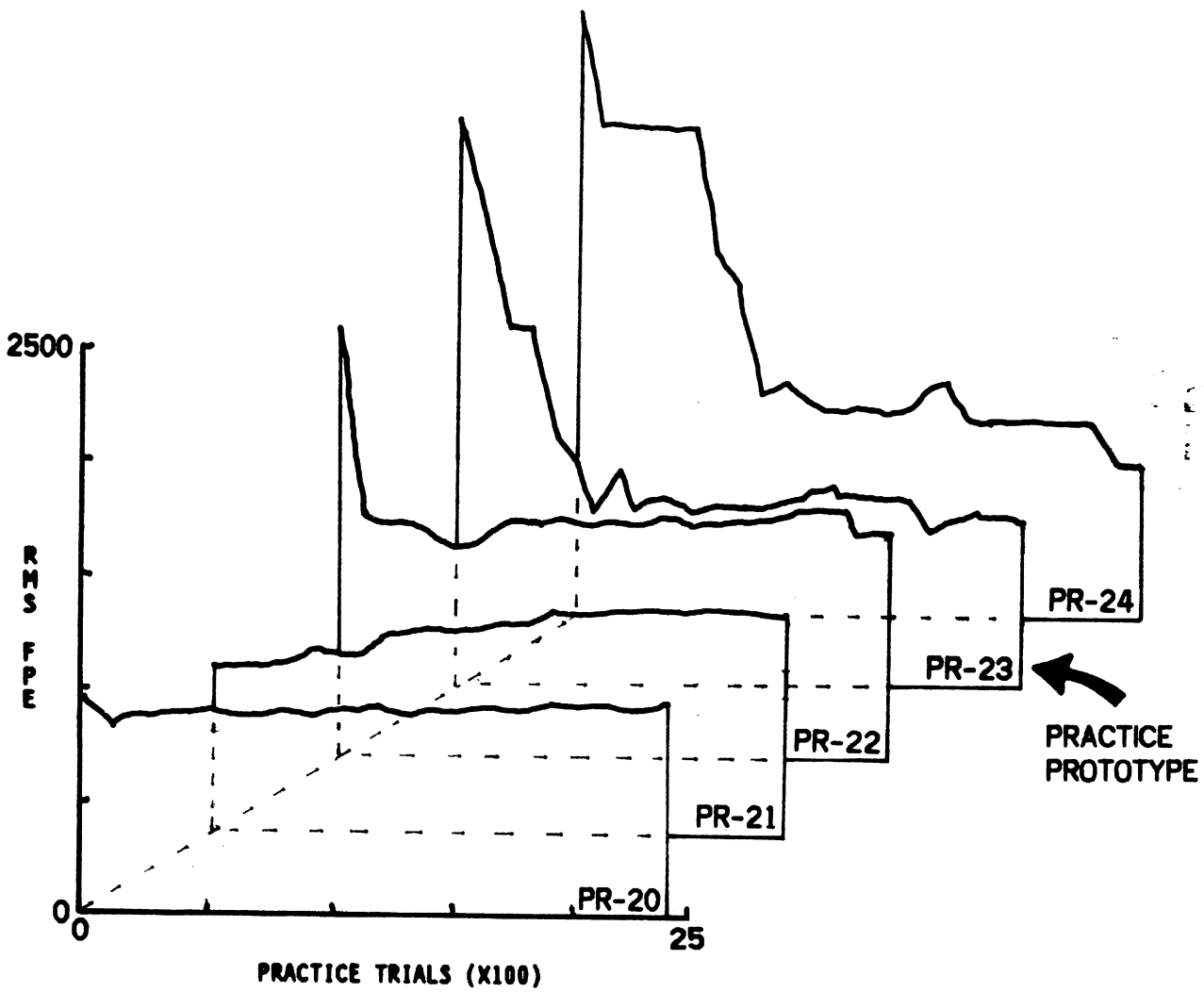


Fig. 5.13 A set of learning curves resulting when practice of prototype PR-23 is preceeded by 2400 trials using PR-20 as the practice prototype. Though Type IIb generalization is not shown, some retroactive inhibition of prototype PR-21 was observed. Also see Fig. 5.12 (squares).

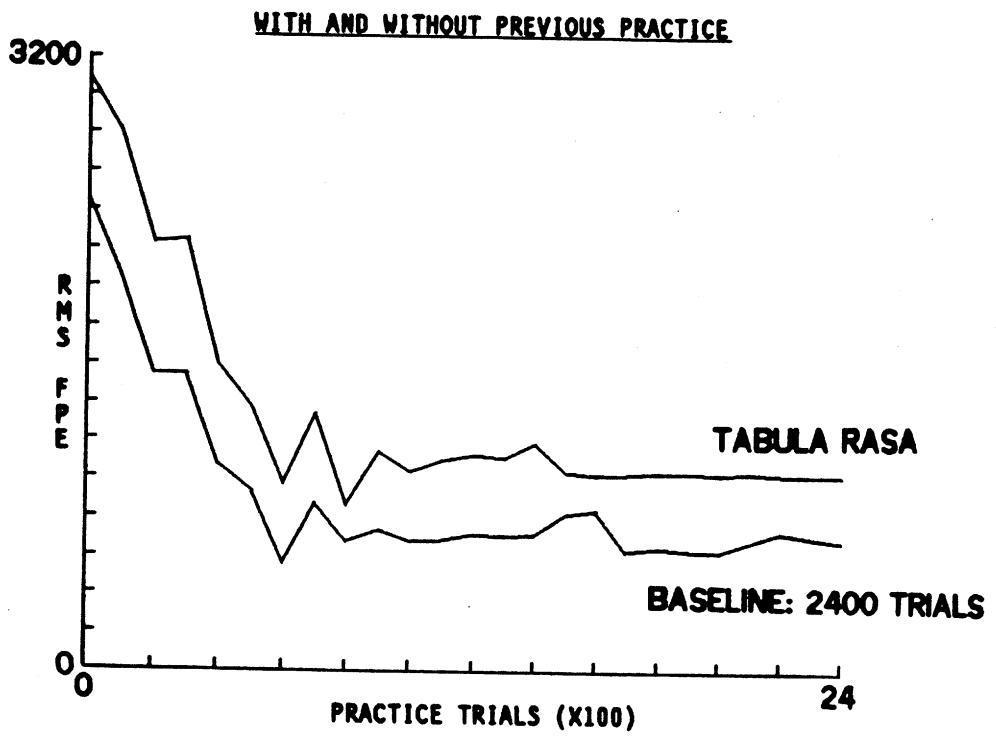


Fig. 5.14 An explicit comparison of learning of prototype PR-23 with and without previous experience reveals little difference. Note, however, that the improvements accrued in the previous situation are not lost.

neighborhood function, even though they were not quite suitable. This points up the care with which the neighborhood function must be chosen.

This test, learning with a baseline, is important because, in addition to testing for another kind of generalization, it most nearly resembles normal steady-state operation of the system. Only in the rare case of a brand new controller would the state space memory and the temporary buffer be tabula rasa, yet almost all of the experiments described in this report begin that way.

Figs. 5.6 through 5.14 show that the system can generalize between similar movements, that intensive practice of a particular movement improves its execution more than other movements, and that the system's general level of performance improves during practice.

5.3 Adaptation

5.3.1 Inertial and Elastic Loads

It is claimed that the model will adapt its motor commands to compensate for changes in the mechanics of the arm. Fig. 5.15 illustrates how the arm is modified to test this property. In one case (Fig. 5.15a) a .75 kg weight is attached to link 3 of the arm. The moments of inertia of all links and the effect of gravity on links 2 and 3 are increased. In the other case a spring, having a constant of 1.85 kg/m, is attached between link 2 and ground. (Fig. 5.14b.) Only static properties of the limb are changed by this manipulation.

The general finding is that application of a mechanical load causes a temporary disruption of motor control, but control is restored after practice with the new

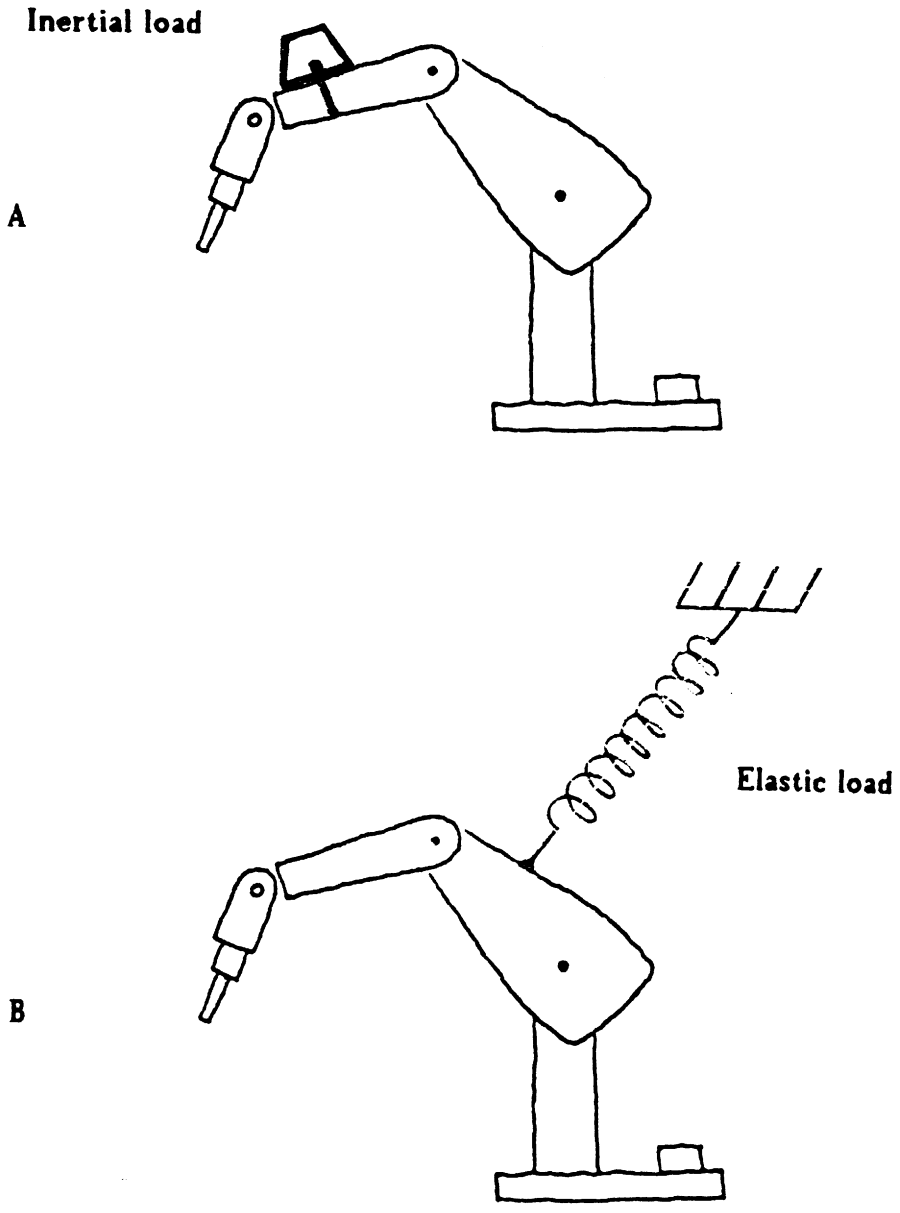


Fig. 5.15 Two methods of applying loads in order to disturb the manipulator's behavior are shown. A) A .19 kg. weight is attached to the third link of the manipulator. B) A 1.85 kg/m spring is attached from the second link to 'ground'. When movements start the spring is stretched .83m and runs from coordinates (.17m,.0m,.25m) to (.02m,.70m,1.20m); see Fig. 4.1

mechanical situation. This result is demonstrated by the data shown in Fig. 5.16. These curves were produced by establishing a 3000 practice trial baseline upon which the effects of disturbances were assessed. The figure shows that both types of load cause a large increase in error which is subsequently reduced. Although these results satisfy minimal expectations for adaptation, manipulations designed to improve the rate of adaptation were performed. Two factors might be responsible for retarding adaptation:

One factor arises because measurement vectors remain in the temporary buffer until they are used in an inversion. Therefore, data generated during the period following application of a mechanical disturbance are likely to result from computations based on combinations of measurements taken before and after application of the load. The constants obtained from these *interim* calculations may attain values which are quite different from either pre- or post-adaptation values -- they do not necessarily attain intermediate values. Fig. 5.17 demonstrates this counter-intuitive effect.

Eight measurement vectors were recorded in each of two different states. For each state a different set of mechanical conditions prevailed. As the ratio of number-of-vectors-from-state-A to number-from-B changes monotonically, the data produced by inversion vary non-monotonically. If one were averaging data from two groups, however, the transition would be monotonic. To assess the effects of these interim calculations, an adaptation test was conducted in which all data from the temporary buffer were removed when the load was applied. The heavier dotted line in Fig. 5.18 shows that this procedure produces no clear improvement in rate of

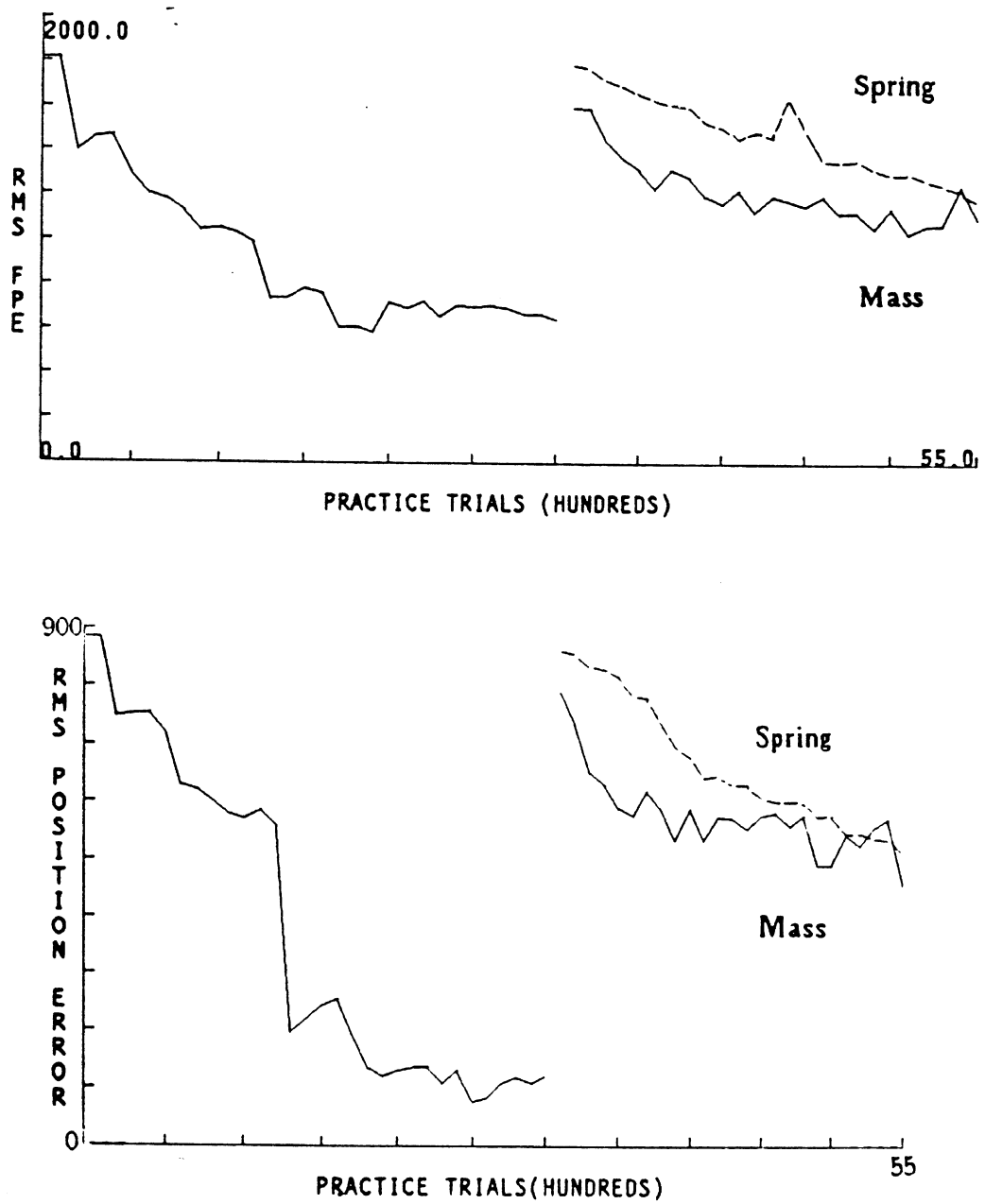


Fig. 5.16 A 3000 trial baseline having been established, adaptation to two types of load are shown. The load is applied at practice trial 3000 and the time course of adaptation is recorded. A) Prototype PR-11, B) Prototype PR-12. (τ=10)

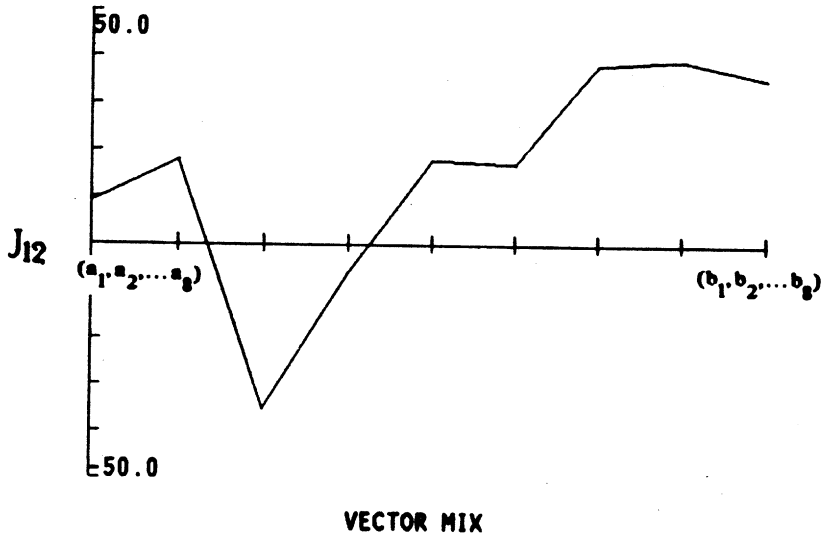


Fig. 5.17 A demonstration of the deleterious effects of 'mixing' measurement vectors when inverting. Two sets of vectors were used, (sets A and B), each consisting of 8 vectors generated for a mechanical situation. Nine inversions were performed where the vectors contributing to each inversion were: $(a_1,a_2,a_3,a_4,a_5,a_6,a_7,a_8)$, $(a_1,a_2,a_3,a_4,a_5,a_6,a_7,b_8)$, $(a_1,a_2,a_3,a_4,a_5,a_6,b_7,b_8)$, . . . $(b_1,b_2,b_3,b_4,b_5,b_6,b_7,b_8)$. The value plotted on the ordinate is one element of the resulting J matrix.

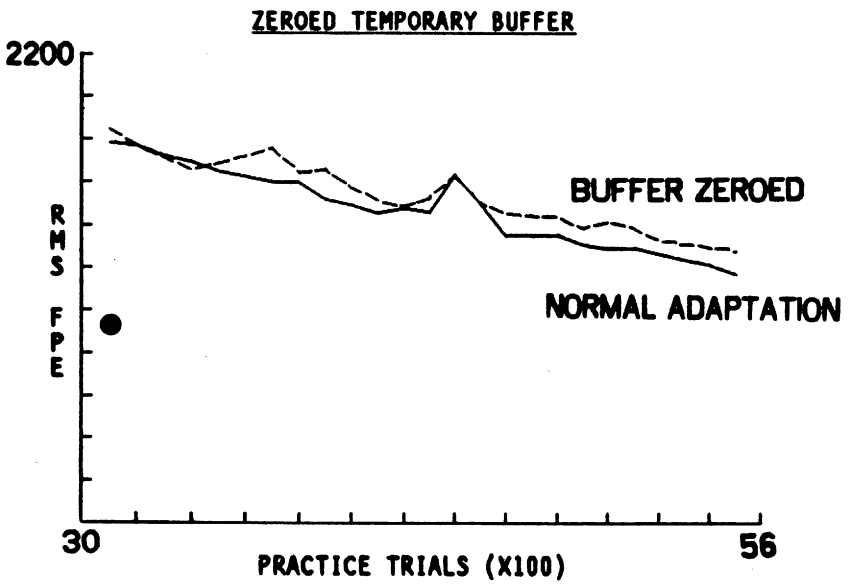


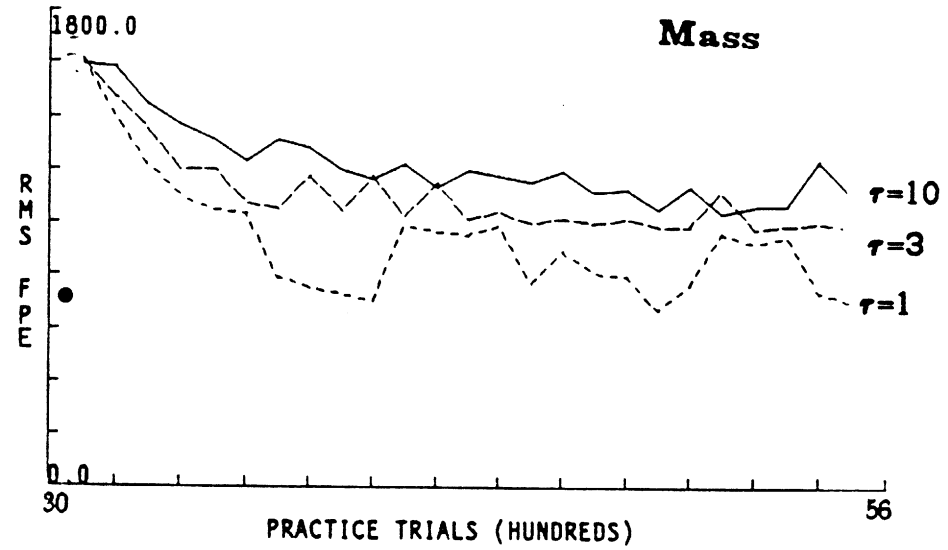
Fig. 5.18 The heavy dotted curve was produced by removing all measurement vectors from the temporary buffer when the spring load is applied. The solid curve is reproduced from Fig. 5.16a for comparison (τ=10). (Prototype PR-11) Closed circle indicates pre-adaptation level.

adaptation.

The rate of adaptation is also retarded when old and new data are combined in the state space memory by averaging. This factor can be adjusted by reducing the value of τ, the averaging parameter discussed earlier. (See Fig. 4.6.) The results of such a manipulation are shown by the curves in Figs. 5.19. Each successively smaller value of τ results in more rapid adaptation to the mechanical disturbance imposed by the spring. While reduction of τ improves adaptation rate, it may reduce the system's resistance to the effects of noisy measurements. (Data presently available do not substantiate this point, but it is strongly expected to be true based on our understanding of the model's operation.)

While reductions in τ decrease the effects of old state space memory data, they do not eliminate them. An experiment was done in which all previous state space data were eliminated at the time the load was applied. The temporary buffer was also zeroed. The dotted curve in Fig. 5.20a reveals an extremely rapid and rather complete adaptation. This is a dramatic result if compared to the adaptation rate shown in Fig. 5.19. Unfortunately, the improvement in adaptation rate is accompanied by an initial loss of control. The level of performance following application of the load is temporarily worse than that initially achieved when the memories are left intact (solid curve in Fig. 5.19a). A more severe loss of control resulting from initialization of the memories is shown in Fig. 5.19b.

Held and Hein (1963) conducted a series of experiments designed to show that initial learning and adaptations were merely two manifestations of the same process.



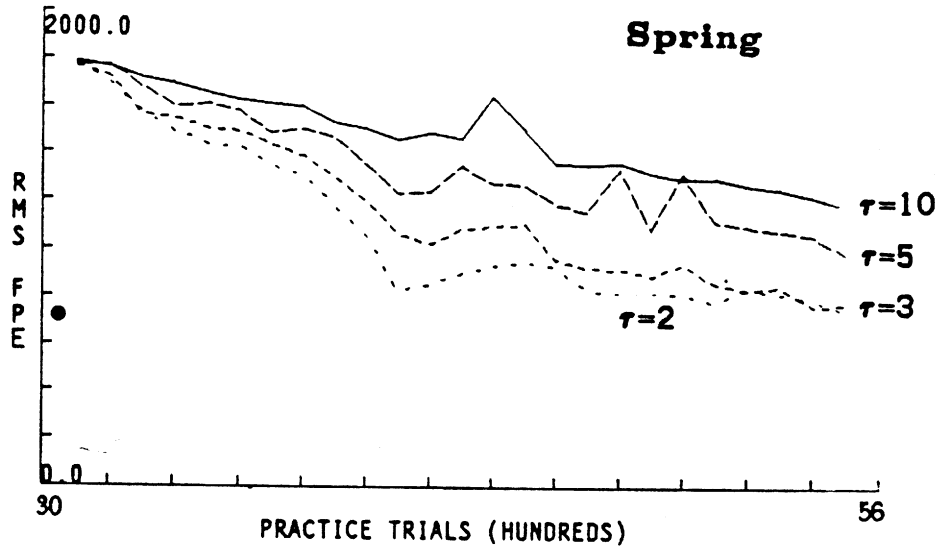


Fig. 5.19 The memory's time-constant is systematically varied. Smaller values of τ yield more rapid, but noisier adaptations. A) inertial load; B) spring load; (prototype PR-11). Closed circles indicate pre-adaptation levels.

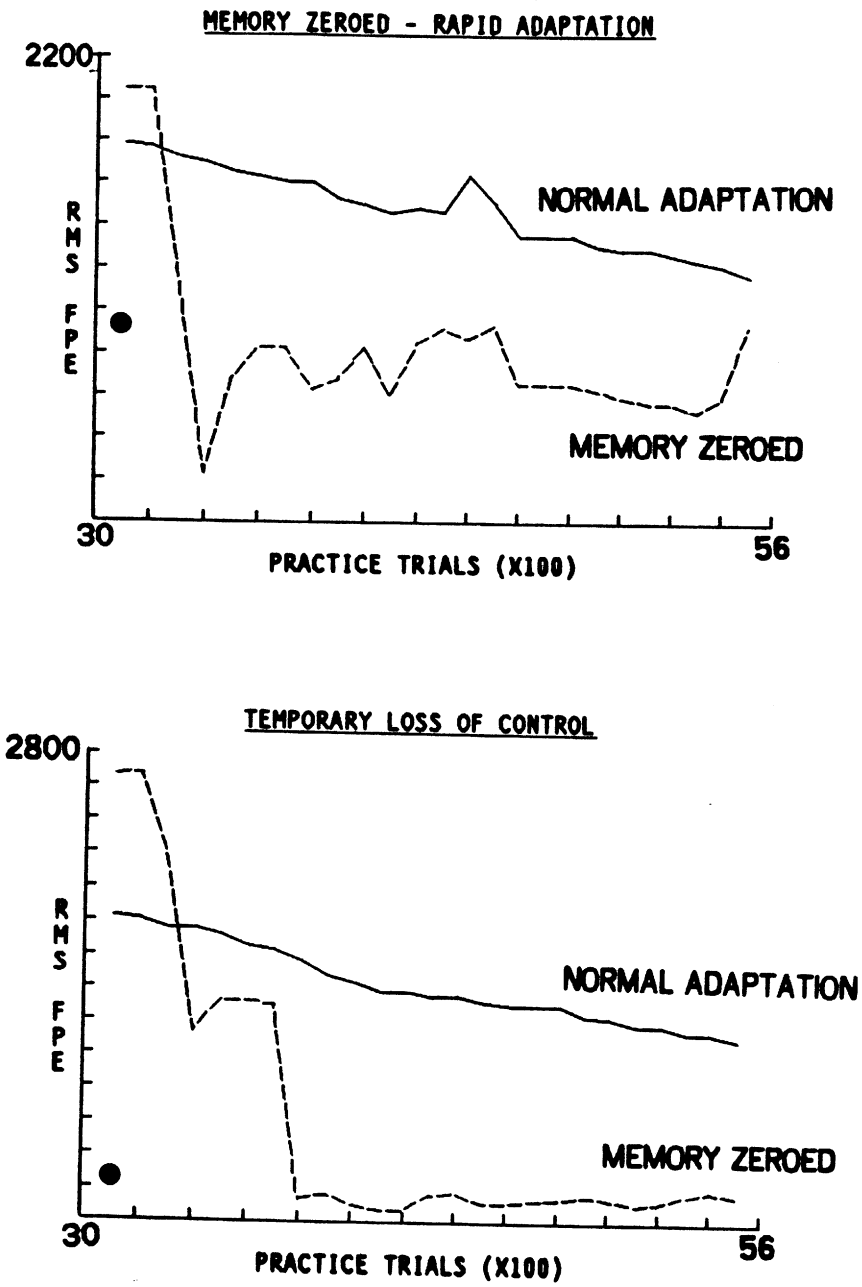


Fig. 5.20 The dotted adaptation curves were produced by removing all data from both the temporary and the state space memories. In A extremely rapid and complete adaptations resulted. In B rapid learning is accompanied by an initial period during which performance is quite poor. Closed circles indicate pre-adaptation levels.

Their results, though not conclusive, were rather provocative. The essential design of the SSM, (there are no provisions for zeroing memories), takes their notion as a premise; no special mechanisms are used for adaptation that aren't used during initial learning. The dramatic results presented in Fig. 5.20, however, argue strongly in favor of re-examining this position.

5.3.3 Reorientation of Gravity Vector

A test for the system's ability to acquire control of the manipulator after reorienting the gravity vector was combined with some of the generalization tests reported above. Though this was not actually an adaptation test -- a manipulation was not made after a baseline of performance had been established -- the results of such an adaptation test can be inferred from the resulting data.

The arm was mounted with its base on the wall so that the gravity vector influenced all three joints. (See Fig. 5.21.) This is an interesting manipulation since joint 1 is normally unaffected by gravity and the gravitational torques exerted at joints 2 and 3 are normally not influenced by the position of joint 1. (By normal, I mean when the arm is mounted as in Fig. 4.1.) A practice session was conducted and the normal procedures were used to assess learning. All of the learning curves reported in Sections 5.2.2 and 5.2.3 were generated with the arm in this position. It is clear from the data presented there, Figs. 5.11 through 5.14, that learning takes place under these circumstances.

In light of the results for the previous Section, 5.3.2, and the learning data from Sections 5.2.2 and 5.2.3, there is every reason to believe that, had there been an

ARM MOUNTED ON WALL

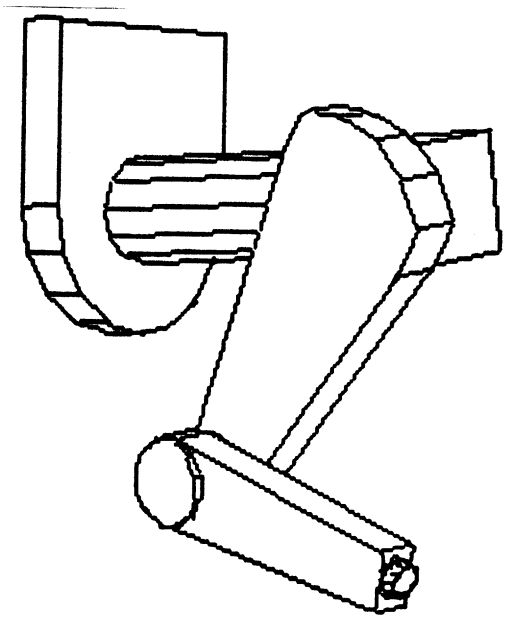


Fig. 5.21 The manipulator was mounted *sideways* to examine the controller's ability to deal with a reorientation of gravity.

established baseline of behavior with the normal orientation, the translator would have adjusted its plan to accommodate the reorientation. I am prepared to argue this point because the model's operation clearly represents the view that adaptation is just the form of learning which takes place when something else was learned first.

5.4 Flexibility of Coordinate System

Here the State Space Model's ability to learn with measurement data from a coordinate system other than those of the joints is demonstrated. The purpose of this test is to show that sensors which operate in coordinates other than those natural to the manipulator can be used as a source of measurement data. Since the arm's potentiometers and tachometers are the only available sources of position and velocity data, Cartesian coordinate data were generated by interpolating a computer program between the generation of measurement data and the use of those data. (See Fig. 5.22.) This program was merely a testing device and has no other function than to allow simulation of the desired sensors. Naturally, the state space map had to be modified to accommodate the new variable ranges. (See Fig. 4.5b.)

Cartesian coordinates were chosen for this test because they represent a large class of coordinate systems. The most important characteristic is that the unit vectors of the system do not coincide with those of joint coordinates. In fact, the relationship between the unit vectors of the two systems is a function of position in space. Cartesian coordinates are also attractive because they are a likely candidate for use by a visual system. (Actually, polar coordinates may be a better choice for vision, but joint 1 of the Scheinman arm coincides with the θ coordinate of such a system. The

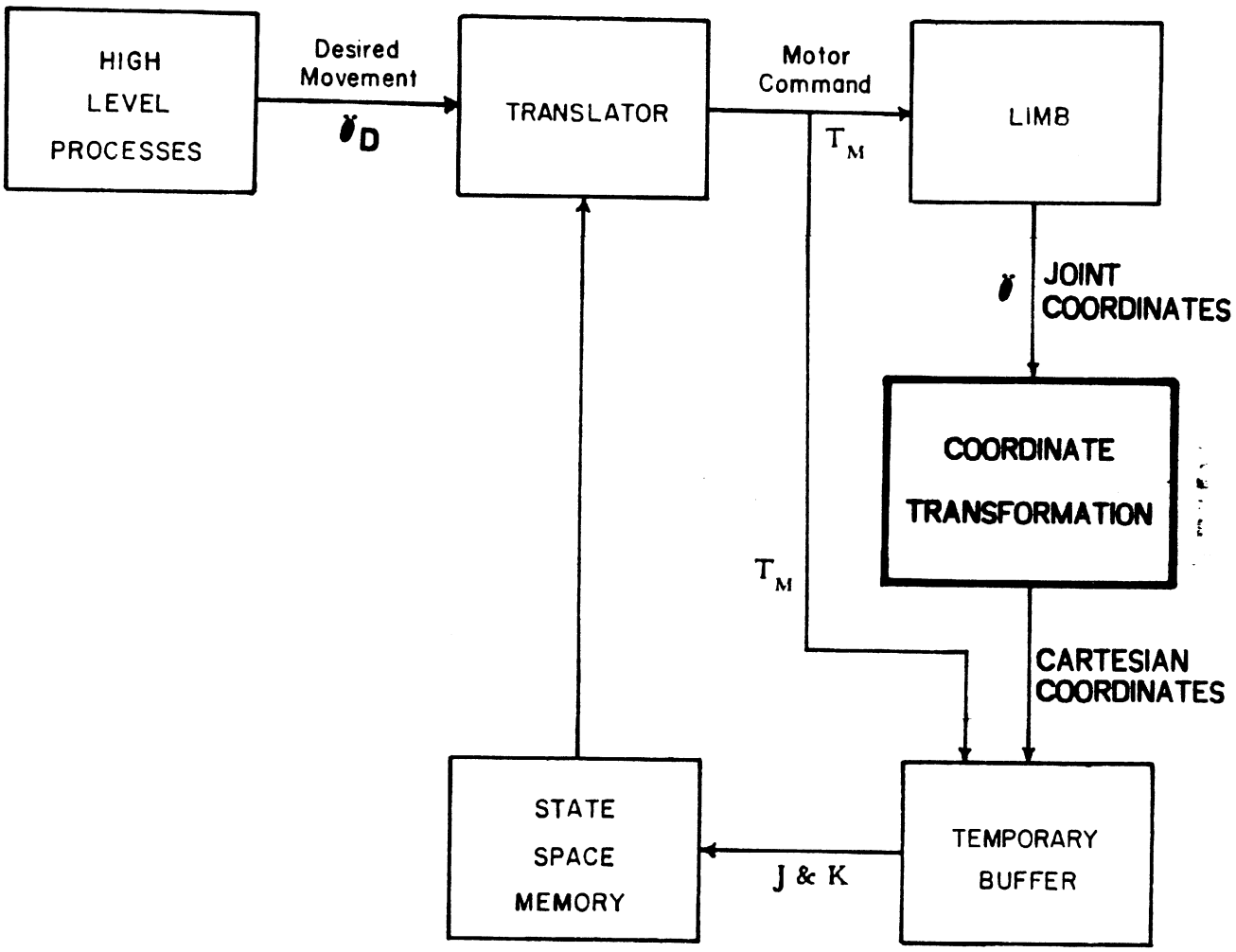


Fig. 5.22 A coordinate transformation is interpolated between the measurement and use of sensor data. Position and velocity data are transformed from joint to Cartesian coordinates.

Cartesian system was chosen to provide a more convincing test.)

The first experiment was conducted in a way that allowed learning in XYZ coordinates to be compared to learning in joint coordinates. Prototype PR-11 and all the practice data used to test acquisition of this movement (Section 5.1) were transformed into Cartesian coordinates. The resulting prototype, designated PR-11XYZ, is plotted in Fig. 5.23. The practice data were processed in the normal way and the resulting learning curve is plotted in Fig. 5.24. The learning shown here is quite good, and verifies the point that the coordinate system used for specification of desired movements is flexible.

A second test was conducted to emphasize the idea that high-level processes can plan movements in sensory space without regard to the joint movements which will be required. In this test a set of straight-line movements, (the tip of the manipulator traverses a straight line), were used as the prototype. The prototypes and resulting data, also used for the generalization tests of Section 5.2.2 and 5.2.3, are shown in Figs. 5.10 and 5.11.

5.5 General Discussion

The amount of generalization exhibited by the controller is largely determined by the range of the neighborhood function and the map that determines the quantization of the state space memory. But these parameters should be chosen with some knowledge of the behavior of the manipulator. That does not mean that a naive controller must have a priori knowledge of correct values for these parameters, but *optimum* choices must be postponed until some experience with the plant variables is gained. This issue

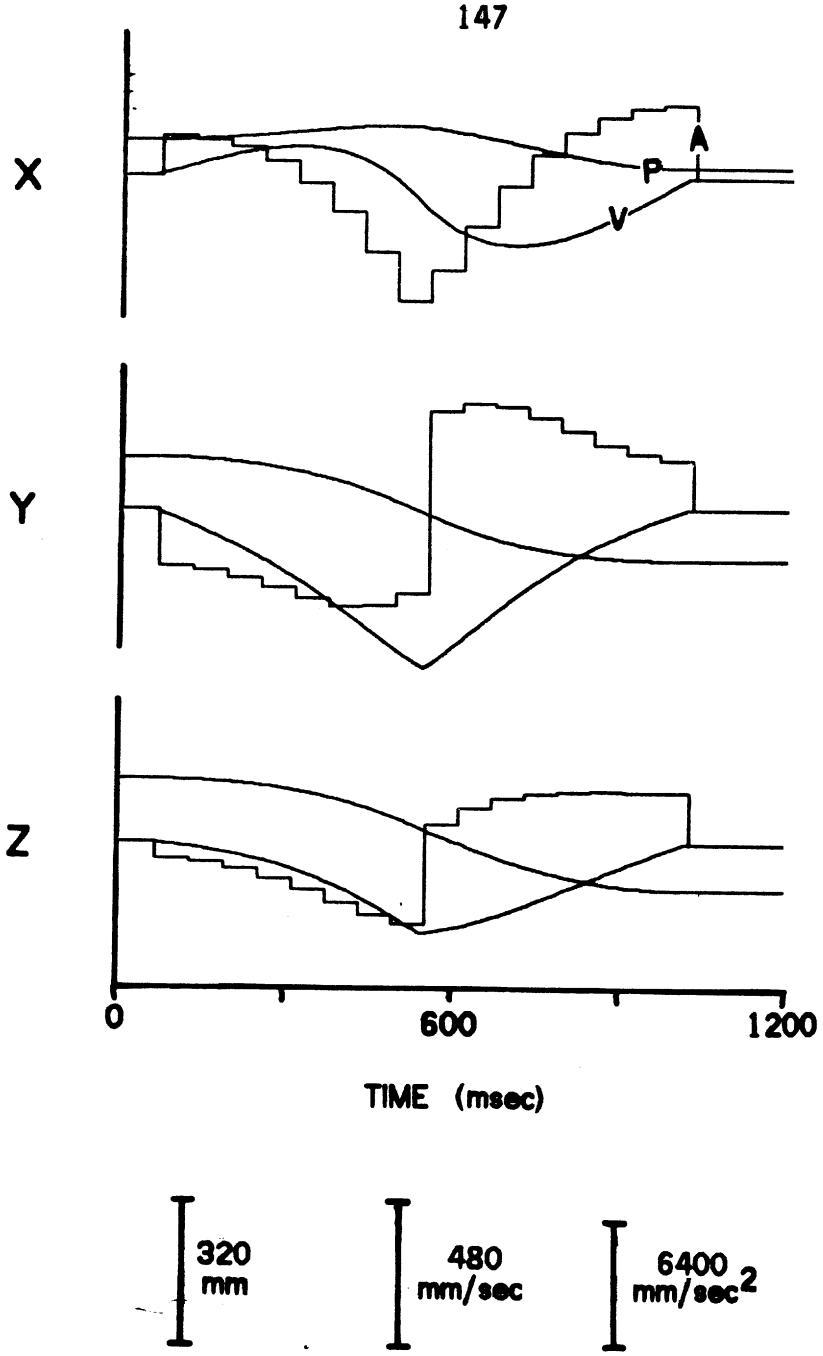


Fig. 5.23 Prototype PR-11XYZ, the Cartesian version of prototype PR-11, is shown. P-position, V-velocity, A-acceleration.

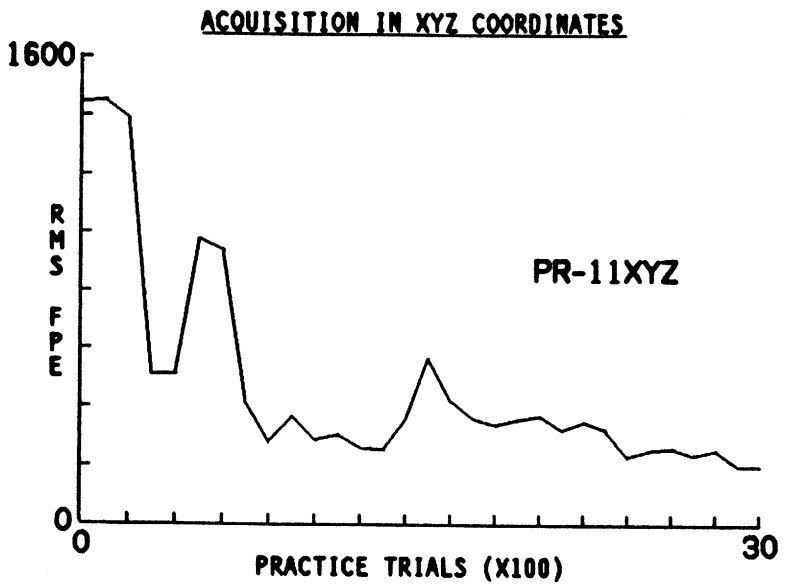


Fig. 5.24 The prototype, PR-11 and practice used to produce Figs. 5.1 and 5.2 were transformed into Cartesian coordinates. Prototype PR-11XYZ was tested to yield the plotted learning curve.

has not received systematic attention here. The neighborhood function and state space memory map were chosen and adjusted to give good performance. It is my opinion, however, that simple mechanisms can be found that will perform these selections automatically. Furthermore, such automatic selection could design memory maps and neighborhood functions which compensate for the rate at which the limb's properties vary with state.

Another difficulty in studying generalization is the lack of a good, general classification scheme for movement. As a consequence, the concept of *similar movements*, necessary for a precise study of generalization, is not well developed. Each pair of movements can be easily classified if we are willing to limit our consideration to a particular model or theory, but the results may be quite unappealing.

It has been shown that the SSM can control an arm mounted on the table or mounted on the wall. Joint coordinates can be used for specification of desired trajectories or Cartesian coordinates may be used. And the masses of the links and elasticities of the joints may vary without a permanent loss of control. Taken alone these results indicate a high degree of flexibility. Furthermore, these data lead to inferences about the overall potential power of the model.

While the data derived using Cartesian coordinates directly show that the coordinates used for measurement can be different from joint coordinates, these data indirectly suggest that the system of coordinates natural to the actuators can also be different from those of the joints: An actuator may apply forces to two or more joints. Indeed, the argument presented in Section 3.2.3 is based on a similar inference. The

Cartesian coordinate data, taken with the results on adaptation to mechanical changes, also imply that the model can effectively compensate for sensory distortions such as magnifications, inversions, and left-right reversals. Finally, though not explicitly demonstrated, I think the data suggest that the SSM can learn to control a number of kinematically and dynamically distinct limbs.

5.5.1 Distributed vs. Massed Trials

In a normal practice session, measurements vectors from temporally adjacent practice trials are often similar. The beginning of a practice block, however, is unlikely to include vectors similar to those at the end of the previous block. Since the invertibility index screens and combines sets of linearly dependent vectors, one would expect more inverses to be found, (more learning), near the beginning of a practice block than elsewhere. Taking this factor into account, one might expect more learning when 100 practice trials are broken down into 5 blocks of 20, than when they are practiced in one large block. This is reminiscent of behavior observed in humans and other animals: Learning is more efficient when trials are distributed in time than when long sessions of practice are employed (Cratty, 1964; Welford, 1968; Taub & Goldberg, 1973; Choe & Welch, 1974).

5.5.2 A Use for Error Data

It was shown that the rate of adaptation to mechanical disturbance was increased when outdated data were removed from the memory. (See Fig. 5.10.) Unlike reducing the value of τ, however, this manipulation requires information indicative of the data's

obsolescence. That information was provided by the experimenter for the tests described above, but a control system could provide those data for itself in a number of ways. For instance someday, high-level processes might visually ascertain that a coil shape object was now connected between arm and ceiling. Using its data base it could determine that such a device was probably a spring and would probably change the mechanical properties of the limb. . . Alternatively, a simple mechanism which merely examines error data could quickly determine a loss of control.

It is interesting to postulate a system that uses error data to determine that something went wrong, and measurement data to find out what went wrong. The expected behavior of such a system, rapid adaptation when error information is provided and moderately rapid adaptation when it is absent, is in agreement with experiments from the psychological literature (Pew, 1974). This combination of feedback and feedforward may prove to be a powerful concept for future models of adaptation.

5.5.3 The SSM and Optimal Control

The theory of optimal control describes how trajectories may be chosen to satisfy a set of movement constraints, while minimizing a cost function for a particular mechanical system (Bryson & Ho, 1969). The constraints might specify, for example, initial and final positions and execution time, while the cost function provides penalties for, say, errors in position, time of arrival, and expenditure of energy (the last of which is minimized by humans during at least one motor activity (Ralston, 1976)).

We have supposed that the functions of motor control are divided into:

High-level processes which plan trajectories without considering the mechanics of the motor apparatus, and low-level mechanisms which translate these trajectories into commands understood by the limb. Since the optimization process generates trajectories, one is inclined to include it with the high-level mechanisms mentioned. When variables related to the manipulator enter the cost function, however, as they do when energy is conserved, the optimization process threatens the presumed dichotomy between high- and low-level functions. In order to optimize energy consumption, the process which generates trajectories must know which motor commands will be required for production, and that is the business of the low-level translator.

This apparent merging of high- and low-level functions is avoided if the optimization process gets its information about energy costs, not from the translator, but from another source which remembers the measured costs associated with previous movements.

5.5.4 A Fair Test of the Model?

The ranges of certain variables have to be limited to satisfy technical considerations. For instance, all velocities have to be below a maximum. When very large velocities are allowed the current/force relationship for the DC motors is no longer valid. This restriction is especially annoying, because some of the more important properties of the control system are most clearly exhibited at high velocities. For obvious reasons, the value of M, the state space quantizing factor, also has to be restricted, thereby reducing the attainable precision of control.

Many combinations of the model's parameters are possible. Limited time

resources forced us to choose relatively few combinations for experimental investigation. In most cases the experimenter was guided by his intuition derived from previous experience, and the results were satisfactory. We have no way of knowing, however, what 'pockets' of unusual or revealing behavior may have gone undetected.

In spite of these difficulties, I feel that the tests presented here are representative of the model's abilities and power. From a research point of view, these drawbacks are compensated in a rather direct way by the degree to which each of the model's variables and parameters are available for examination and manipulation.

5.6 Improvements to the Model

In the course of developing and testing this model, a number of ideas emerged which were not included in the implementation presented above. Some were available at the outset but were not used in order to keep things simple. Others presented themselves after the experimenter became more familiar with the system's operation. Since they might be valuable for future work in this field, this section presents a brief list of these ideas with some discussion of their motivation. Most of them are not well developed and no plans exist for their implementation or test.

5.6.1 Insuring the Command-Force Relationship

Earlier it was pointed out that there are restrictions on the allowable relationships between the command issued by the controller and the net force or torque applied to the joint. The Scheinman manipulator used in these tests is powered by DC torque motors. They have the characteristic that, neglecting friction, the torque

produced is proportional to the current through the motor at all speeds. These motors are driven by servo amplifiers which insure, for a certain range of inputs and velocities, that the current through the motor is proportional to the voltage applied to the amplifier. Since the amplifiers only have a finite voltage swing (28v) and the motors produce a back emf when in motion, the amplifiers are not always able to drive the desired current. In order to check for this condition the actual voltages across the motors was monitored at all times. Whenever these values approached 28 volts during a measurement, that measurement was ignored because the amplifier might have been saturated and applied an unknown force.

A more systematic treatment of this problem could be developed if sensors were used to measure the actual force delivered by the actuator. Then the force measurement, rather than the command, could be stored with the resulting acceleration measurement. Measurements of actual force delivered would automatically adjust for any saturation effects in the actuator. On the other hand, the translator would only determine the force to apply to a joint, rather than command -- another piece of hardware would have to convert the desired force into a command which produced that force. But that one dimensional problem, involving data for only one joint, is easily solved. This type of arrangement would also have the advantage that changes in properties of the actuators which occur quickly, such as fatigue or warm-up, need not effect performance of the translator.

5.6.2 Practice Improves Practice

In the present implementation, each practice session is totally independent from every other practice session. Each session starts about the same, and often includes a number of wild trajectories that are very different from the desired movement. Therefore, although much of the data generated might be useful at some future time, or for replication of some other movement, they are useless for the task at hand -- learning to replicate the desired trajectory.

In man, on the other hand, experience influences practice. The sophisticated mover does not flail his limbs around each time he wishes to learn a new movement. On the contrary, he may begin by executing a reasonably good approximation to his goal on the very first try. After some practice he will be doing something very close to the desired response, and each attempt at that level may be rich in measurements usable by the learning mechanism.

This type of regenerative effect:

practice → learning → better practice → more learning → etc.

could be quite important for future studies. In order to make use of this approach, the practice algorithm must use the translator's expertise when planning movements.

In addition to accelerating learning, the practice-improves-practice approach might be used to explain another important human ability. It is common when learning a new movement to begin by practicing *slowed-down* approximations. The practice movements are gradually increased in rate until the desired movement is performed at required speed. If the translator uses the state space memory during practice, it too

could perform in this way:

The basic idea is that it is easier to practice slower movements because dynamic interactions are minimized. Once information is gained regarding slowed-down versions of a movement, slightly faster versions could be efficiently practiced since the system generalizes to neighboring states. The initial slow practice movement may not be within *generalization-distance* of the goal, but an intermediate trajectory can be improved. (Fig. 5.7 illustrates generalization from one movement to 3 slightly faster versions.) Eventually the system can learn to practice and execute movements of any speed.

Of course, not all skills are developed using the start-slow/speed-up paradigm. In many situations it is important to establish the correct form or rhythm of movement paying little attention to the details, which can be *fine-tuned* later. Here again, a SSM with a practice-improves-practice approach may prove enlightening, though some hard questions about the *form* of a movements may have to be answered first.

This scheme, the use of learned data during practice, is a large improvement over the implemented system, which relies on the unchanging practice algorithm to generate data for all appropriate states. We might note, however, that the methods of learning fast movements by starting with slow ones and of fine-tuning movements of roughly the correct gestalt will require help from relatively high-level processes.

5.6.3 Decaying Measurement Vectors

During the normal course of an organism's development, the mechanical properties of a limb will change in a number of ways. As new measurement data arrive describing these new properties, the translator's equations of motion will change. But there is a potential problem which impairs the translator's ability to adapt, and even allows for wildly deviant performance during the adaptation period.

At any given time there are usually a number of vectors stored in the temporary buffer awaiting the arrival of others, at which time inversion takes place. Each of these vectors may have been generated during different mechanical conditions, if the mechanical properties of the system are changing rapidly or measurement data are being generated slowly. If a single inverse computation includes vectors generated during changing mechanical conditions, the resulting state space data are not likely to be reliable. (See Fig. 5.17.) Even when the transition from old to new data is smooth, this effect tends to prolong the amount of time and practice needed to completely change the controls.

The translator described here does not know when its vectors are no longer applicable, though performance measures could be used to help obtain such information. Another solution is based on the notion of a decaying memory. If old measurement data are removed from the temporary buffer, or given reduced weights, there will be a reduction in the range of measurement dates found contributing to any one inversion. Of course, the same sort of temporal decay could also be applied to data in the state space memory. Observe that I am advocating a decay of weight, not a decay of value.

An interesting result comes from considering the consequences of applying a particular decay function to the state space memory. Suppose, first of all, that all data decay exponentially. During any time interval new, old, and intermediate data are all reduced by the same fraction. When data do not enter the memory, all weights are reduced, but the relationships among weights are the same and behavior remains unchanged. If an exponential with a growing time-constant is used, (see Fig. 5.25), new data will decay faster than old, and the system will tend to return to previously used values. Of course, the time-constant need not be continuously growing. An exponential decay function with a piecewise constant time-constants, (eg. short plus long term memory; see Fig. 5.25b), would also give a temporary large weight to recent data.

Now consider the following case: A large amount of data have been collected and stored. The mechanical properties of a limb are artificially manipulated and a practice period is permitted. If memory weights decay exponentially, the level of performance at any time after the adaptation period, but before new data are generated, should be the same as that found immediately after the adaptation period. A decay function that uses a growing time constant, however, should result in an initial improvement in performance, followed by a gradual return to pre-adaptation levels.

Hamilton and Bossom (1964), and Choe and Welch (1974) conducted prism adaptation experiments under these circumstances. Hamilton and Bossom suggest that their findings argue for a distinction between the mechanisms responsible for initial acquisition and those for adaptation. Their results, however, are consistent with the alternative notion of a variable time-constant memory.

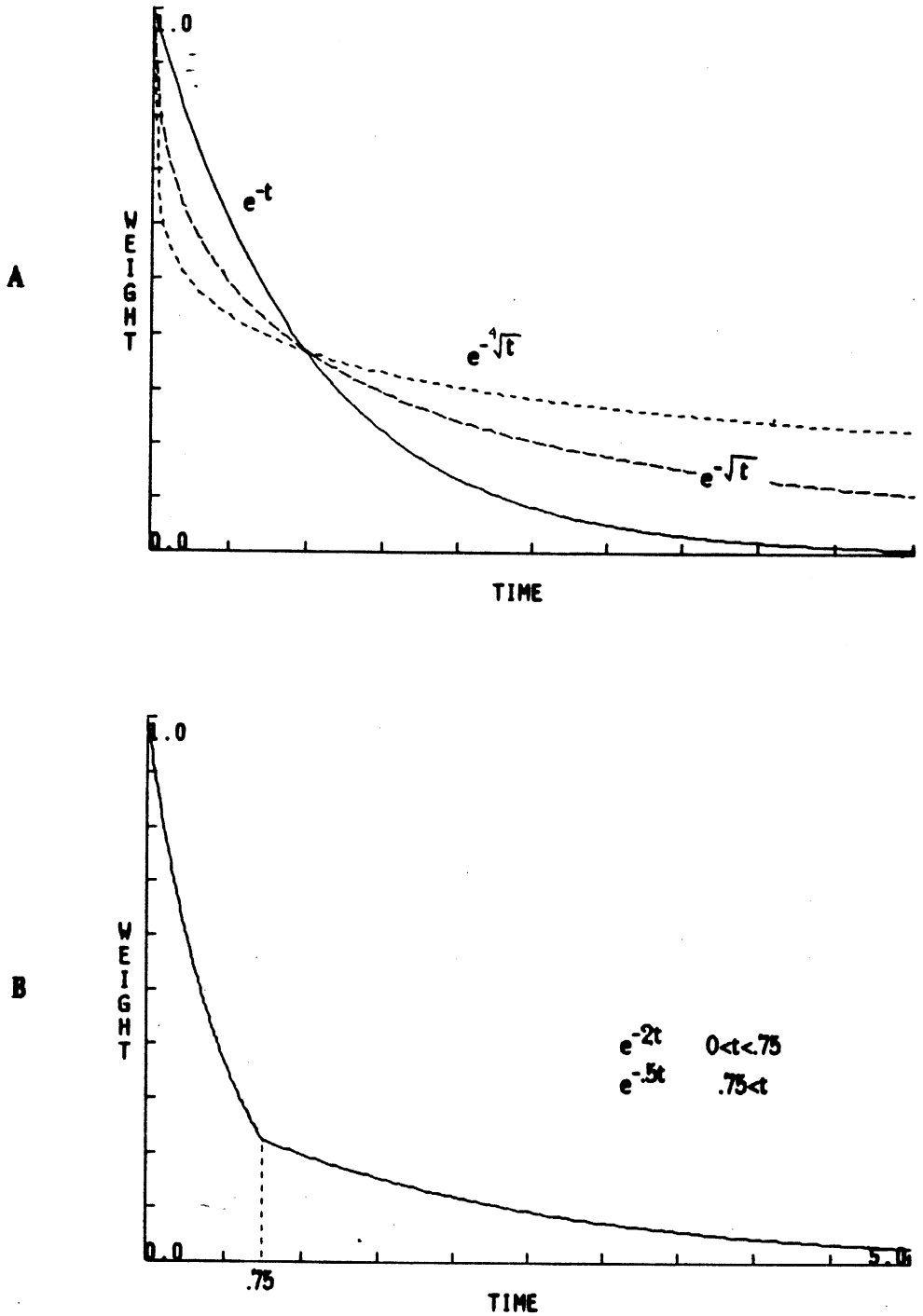


Fig. 5.25 Examples of decay functions with variable time-constants. Examples of two classes of such functions are shown a) Continuously varying time-constants: $\tau=-1.$, $\tau=-t^{-.5}$, and $\tau=-t^{-.75}$; b) Piecewise constant time-constant: $\tau=\{-2., 0<t<.75; -.5, .75<t\}$. A combination of short and long term memory falls into this second class.

6 Concluding Remarks

6.1 Derivative and Alternative Models

The State Space Model has been presented, implemented, tested, discussed, and put into perspective. We now examine cousins and decendents.

6.1.1 The Measurement Space Model

The following formulation incorporates a number of improvements into the design of the State Space Model, while maintaining its desirable properties. The major differences in the new formulation are:

1) The state space memory and temporary buffer are combined. Measurement vectors are stored in a state space memory. Inversions are only performed when data are used during translation.

2) Hyper-regions are no longer discrete entities.

3) Each measurement vector is labelled with its time of generation. The age of a vector contributes to its weight during inversion.

4) The neighborhood function, applied before inversion rather than after, gives each measurement vector a weight determined by its distance from the desired state.

5) During the planning of a movement the generalized inverse is used to invert all measurement vectors found in the neighborhood of the desired state.

This arrangement allows for a simpler, less ad hoc specification of the neighborhood function. When data are taken from the memory they are weighted as a function of their distance from the desired state. The neighborhood function can be continuous and the broader it is more the data in the memory are shared in the

production of different movements. A sharper function enables better replication of movements which travel through non-linear portions of the movement space. Since each measurement vector carries information about its age, the advantages of a decaying memory can be realized.

An invertibility index would no longer be needed because each use of the generalized inverse yields an optimal estimate of the constants of mechanical description in the mean square error sense. The procedure described in Section 4.2.2.2, averaging optimal estimates, is only sub-optimal. The proposed new method is truly optimal in that each computation of the mechanical constants relies on the maximum possible number of measurements -- all those present. Though large inversions of this type are expensive in terms of time and memory for serial computers, they present no special problems for computers characterized by large numbers of parallel interconnections.

6.1.2 The Configuration Space Method CSM

Many control schemes used in practice ignore inertial interactions between joints and Coriolis forces, yet these terms can be important during high velocity motions. This compromise has important implications for industrial applications where the throughput of a manipulation process depends on the arm's speed. The importance of CSM is that these terms can be included at low computational cost with *reasonable* amounts of memory. Real-time trajectory calculations may also be possible, especially in the context of a distributed computation employing multiple microprocessors (Raibert & Horn, 1977). Some of the details of this approach are given in Section 1.3.6.

6.1.3 Multiple Spaces Model

The worst drawback of the SSM is the amount of storage required. It increases as the power of the number of state variables; two for each degree of freedom, M^{2N}. For a few degrees of freedom this number is managably large, but it soon reaches unreasonable proportions, even for the renowned capacity of the central nervous system. (By some analyses the human arm and hand have a total of 35 degrees of freedom.) There are, however, a number of ways in which the memories required for each limb or manipulator can be kept to practical sizes.

A review of the terms entering the equations of motion for a manipulator reveals that, except for the Coriolis force, each is a function of the position state vector or the velocity state vector, but not both. (See Eq. 2.2) Gravitational forces and moments of inertia are dependent only on the position vector, while friction is primarily a function of velocity. Since Coriolis forces are typically small, one might propose the use of two memories; one with N positional dimensions and one with N velocity dimensions. Development of this approach could lead to practical control applications, especially if used together with hashing techniques.

Under certain circumstances a controller that uses one memory with dimensionality 2N can be replaced by two controllers, each of which employs a memory with fewer than 2N dimensions. This can be done whenever the mechanical properties of the plant may be decomposed into non-interacting subsections, or when the interactions are constrained to a few degrees of freedom. Then each controller will have state variables which represent the net influence of coupling with the other

mechanical component. One could imagine the use of this type of arrangement in controlling the interactions between the trunk of the body and each limb. Suppose that each of two limbs had 4 degrees of freedom, and the trunk had another 3. Further suppose that all the coupling at each shoulder could be represented in terms of 3 degrees of freedom. If the entire system were controlled from one memory it would require 2(4+4+3)=22 dimensions. If three separate controllers were used, however, 3 memories would be required, of dimension 14, 14, and 12. Though these results are still out of the question, we are encouraged to pursue this line of attack.

6.1.4 Visually Locate and Move

Man is able to guide his hands to visual targets. This is essentially a problem of transformation between coordinates systems. He learns to do so in infancy and early childhood (Held & Hein, 1963; White, 1970; Held & Bauer, 1974). Once established, the relationship between these coordinate systems is maintained despite natural or experimental changes to the visual and motor systems. The human mover can maintain or re-attain correspondence between hand and eye when the relationship between visual and motor worlds are disturbed by shifts, distortion, invertion, lateral reversal, magnification, and minification (Held, 1961; Held & Freedman, 1963; Miles & Fuller, 1974; Gonshor & Melvill Jones, 1976).

What mechanism could be responsible for such flexibility? Is it possible that the motor learning we see in each of us is not primarily an expression of improved control, but only an improvement in coordination? Glen Speckert and I designed and implemented a controller that employed two processes: one that learns to transform

trajectories from 2-d visual coordinates to 2-d arm coordinates, and another that moves the arm along desired arm-coordinate trajectories (Speckert, 1976). Results from this system (using a TV camera and the MIT-Scheinman arm) suggest that certain forms of learning, adaptation, and generalization are possible even if the process responsible for moving has no learning abilities. Rather, all improvements in performance correspond to a a more effective transformation process. In particular, the system can learn to pick up visually targetted objects when the visual data provided to the system are disturbed by any of the distortions mentioned above. (Not all of these have been demonstrated, but there is strong reason to believe that each will be correctly learned or adapted.) This is all done using no trigonometric operations and no complicated mathematics.

We assume that the transformation between a point expressed in visual coordinates, $\underline{V}_i$, and the same point expressed in arm coordinates, $\underline{A}_i$, only changes gradually as the point moves throughout space. When the coordinates for points are available expressed in both arm and visual coordinates, $\{(\underline{V}_1,\underline{A}_1), (\underline{V}_2,\underline{A}_2), \ldots$ etc.$\}$ an interpolation will yield the transformed coordinates of any new *nearby* point, P_i. Since topology is not changed by transformation of coordinates, nearby points are nearby in both systems.

The system is most useful when organized by a multi-dimensional table quite similar to the state space memory, but only having dimensions of visual position coordinates. (This makes sense since the dynamics of the problem are not included here.) Pairs of coordinates are stored in this table when they become available during

movement -- the position of the hand in arm coordinates is recorded along with the simultaneously observed position of the hand in visual coordinates. When a transformation is necessary the memory is accessed through the visual coordinates of the target point, $\underline{V}_i$. The pairs of nearby coordinates stored in the region of the space allow the new $\underline{A}_i$ to be approximated.

Of course, this is only the beginning of a theory. No mention has been made of using two moving eyes on a moving head. Furthermore, there is no reason to prohibit learning by the movement process. More complete work on this and related models will lead, I hope, to better understanding of *motor maps*, *sensory maps*, *sensorimotor maps*, and *mappings*.

6.2 Problems for Further Research

Many new ideas developed during the pursuit of this thesis. Those that were well organized were presented in the last section. They consist of new approaches to the manipulator control problem. A number of problems which are less well understood, but which bear more work are suggested here.

6.2.1 Measurement + Error Correction

Throughout this thesis I have emphasized the fact that learning by the SSM cannot be attributed to error correction procedures. Many investigators have devised quite effective adaptive controllers which do rely on error correction. (Virtually the entire subfield of control theory called *identification*, serves as an example.) I feel the most advanced learning controllers will probably result from the appropriate

combination of these two ideas: measurement and error correction. But it is not clear how this can be done.

In Section 5.5.2 I discuss a hybrid approach that uses errors only to detect changes in the plant, with correction made by measurement techniques. An alternate approach might make measurement adjustments until behavior ceases to improve, then use error correction to make final, high precision adjustments. This might be especially useful in overcoming one of the most important pitfalls of error correction learning algorithms -- their susceptability to local minima (Tsypkin 1971, 1973). The measurement data can be used to get the solution onto the correct hill, while error data are used to climb that hill.

6.2.2 Plan + Servo

Just as the best learning will probably result when measurement is combined with error correction, the best control will probably result from judicious combinations of preplanning and feedback control. A number of possibilities in this area remain to be studied:

1) Can state space data be used on-line in a reprogramming mode?

2) If the translator works in visual coordinates, can visual errors be used to reprogram?

3) Perhaps a simple servo-assistance mechanism is adequate in certain circumstances. What are they?

6.2.3 Practice

The initial conception of this project described the practice algorithm as an unimportant black box which 'practices movements'. An important finding of this work

has been, however, that the amount and type of learning, adaptation, and especially generalization is intimately tied to what a system does when it practices. A few parameters were adjusted (Section 4.3.4) but no systematic, comprehensive study was made. I would like to know:

1) Are their special movements which provide the most learning for the least practice (*eigenmovements*)?

2) Can state space data be effectively used during practice (Section 5.6.2)?

3) How much high-level planning would be needed to permit practice which started slowly and increased in speed (Section 5.6.2)?

4) Can special probe movements, or probe components be incorporated in a practice routine to accelerate learning? (Nashner 1976) advances this hypothesis.)

5) What do psychologists mean when they talk about practice? (Few realize that learning n repetitions of a task doesn't mean n identical sequences of stimulation and response (Mednick 1964 p.92).)

6.2.4 High-Level Processes

One of the important advantages of using a translator is that high-level processes can plan in one language and use the resulting plans to move a number of limbs which may *speak* different *languages*. The SSM was designed with this facility in mind, yet we have not substantiated the utility of such an arrangement. A number of workers in artificial intelligence may now be in a position to obtain such verification (Lozano Perez 1976, Mason 1977, Will 1975).

6.3 Summary

The human motor system acquires control of each limb in the body, adapts to mechanical and sensory changes, transfers training between practice movements, and performs coordinate transformations from sensory space to motor space. This thesis presents the theory and implementation for a model which exhibits these properties.

A controller is proposed which translates descriptions of desired trajectories into motor plans. The processes which provide input to this translator do not have to deal with the mechanical properties of the manipulator, and the specified trajectories may be expressed in a coordinate system appropriate to the available sensors. The translator's outputs are motor commands suited to the kinematic and dynamic properties of a particular manipulator and its actuators.

The model employs parameterized equations of motion in conjunction with a quantized, multi-dimensional memory organized by state variables. The memory is supplied with data derived from the analysis of practice movements. The analysis performed is quite simple and does not employ error correction or search techniques, as do many learning schemes currently in use. Since iterative methods are avoided, problems involving local minima are not encountered.

A small computer and three joints of the MIT-Scheinman manipulator were used to implement the controller and assess its properties. Tests have verified the controller's ability to:

1) acquire usable mechanical descriptions of the manipulator, and to use those descriptions to pre-plan effective trajectories.

2) adapt to mechanical disturbances caused by inertial and elastic loads,

and acquire control after the gravity vector is modified.

3) generalize information derived from the practice of one movement to the execution of other similar movements.

4) use a Cartesian coordinate system for specification of desired trajectories when measurement data are provided in that system, even though motor commands are expressed in joint coordinates.

Since these tests were conducted on a physical manipulator the possibility of undiscovered vaguaries are greatly reduced.

The nature of parameterized equations was investigated in order to bring the State Space Model and a number of other models into one conceptual framework. In addition to describing a number of alternate control schemes, improvements to the present model were proposed and discussed. These included:

1) Employment of a subsidiary controller that maintains the command-force relationship for each actuator.

2) The use of a more powerful practice algorithm

3) Inclusion of a decay factor in operation of the controller's memory.

4) Methods for reducing the size of the state space memory.

5) Elimination of the discrete nature of the memory.

Bibliography

Albert, A., *Regression and the Moore-Penrose Pseudoinverse*, (Academic Press, N.Y., 1972)

Albus, J.S., "Theoretical and experimental aspects of a cerebellar model", Ph.D Thesis, Biomedical Engineering, University of Maryland, 1972.

Albus, J.S., "A new approach to manipulator control: the Cerebellar Model Articulation Controller (CMAC)", *J. Dynamic Systems, Measurement, and Control, trans. of ASME*, 1975, Series G, 97 No. 3 pp. 220-227.

Albus, J.S., "Data storage in the Cerebellar Model Articulation Controller (CMAC)", *J. Dynamic Systems, Measurement, and Control, trans. of ASME*, 1975, Series G 97 No. 3 pp. 228-233.

Albus, J.S., and Evans, J.M., Jr., "Robot systems", *Scientific American*, February, 1976 pp 76-86b.

Arbib, M.A., *The Metaphorical Brain*, (Eiley & Sons, Inc., N.Y., 1972).

Bejczy, A.K., "Robot arm dynamics and control", *NASA Technical Memorandum 33-669*, February, 1974.

Bilodeau, E.A., (ed.), *Acquisition of Skill*, (Academic Press, N.Y., 1966).

Bizzi, E., Polit, A., and Morasso, P., "Mechanisms underlying achievement of final head position", *J. Neurophysiol.*, 1976 39 No. 2 pp. 435-444.

Bower, G.H., and Spence, J.T., *The Psychology of Learning and Motivation*, (Academic Press, New York, 1969).

Bryson, A.E., and Ho, Y.C., *Appied Optimal Control*, (Ginn and Company, Waltham, 1969).

Choe, C.S., and Welch, R.B., "Variables affecting the intermanual transfer and decay of prism adaptation", *J. of Experimental Psychology*, 1974 102 pp. 1076-1084.

Conolly, K., (ed.) "Skill development: problems and plans", *Mechanisms of Motor Skill Development*, (Academic Press, London, 1970), pp. 3-16.

Cratty, B.J., *Movement Behavior and Motor Learning*, (Lea & Febiger, Philadelphia, 1964).

Eden, M., "Handwriting and pattern recognition", *IRE trans. on Information Theory, IT-8*, 1962 pp. 160-166.

Evarts, E.V., "Brain mechanisms in movement", *Scientific American*, 1973 229 No. 1 pp. 96-103.

Evarts, E.V., Bizzi, E., Burke, R.E., Delong, M., and Thach, W.T., Jr., "Central control of movement", *Neuroscience Research Program Bulletin*, 1970 9 No. 1.

Freedy, A., Hull, F.C., Lucaccini, L.F., and Lyman, J., "A computer based learning system for remote manipulator control", *IEEE trans. on Systems, Man, and Cybernetics*, Vol. SCM-1, No. 4, 1971, pp. 356-363.

Fu, K.S., "Learning control systems and intelligent control systems: an intersection of artificial intelligence and automatic control", *IEEE trans. on Automatic Control*, Feb., 1971 pp. 70-72.

Gelfand, I.M., Gurfinkel, V.S., Tsetlin, M.L., and Shik, M.L., "Some problems in the analysis of movement", In: *Models of the Structural-Functional Organization of Certain Biological Systems*, Gelfand, I.M., (ed.) (M.I.T. Press, Cambridge, Mass., 1971).

Gill, A., "Visual feedback and related problems in computer controlled hand eye coordination", *Stanford Artificial Intelligence Memo No. 178*, October, 1972.

Gonshor, A., and Melvill Jones, G., "Short-term adaptive changes in the human vestibulo-ocular reflex arc", *J. Physiol.*, 1976, 256 pp. 381-414.

Gonshor, A., and Melvill Jones, G., "Extreme vestibulo-ocular adaptation induced by prolonged optical reversal of vision", *J. Physiol.*, 1976 256 pp. 361-379.

Greene, P.H., "Problems of organization of motor systems", *Institute for Computer Research Quarterly Report, Section II C*, University of Chicago, Spring, 1971.

Hamilton, C.R., and Bossom, J., "Decay of prism aftereffects", *J. of Experimental Psychology*, 1964 67 pp. 148-150.

Hammond, P.H., "The influence of prior instruction to the subject on an apparently involuntary neuro-muscular response", *J. Physiol.*, 1956 127 pp. 17-18.

Hein, A., and Held, R., "A neural model for labile sensorimotor coordinations", In: *Biological Prototypes and Synthetic Systems*, Bernard, E., and Kare, M. (eds.) Vol. 1, (Plenum Press, New York, 1962), pp. 71-74.

Held, R., "Exposure-history as a factor in maintaining stability of perception and coordination", *J. Nervous & Mental Diseases*, 1961, 132, pp. 26-32.

Held, R., "Plasticity in sensory-motor systems", *Scientific American*, 1965 213 No. 5 pp. 84-94.

Held, R., and Bauer, J.A., Jr., "Development of sensorially-guided reaching in infant monkeys", *Brain Reasearch*, 1974 71 pp. 265-271.

Held, R., and Bossom, J., "Neonatal deprivation and adult rearrangement: complementary techniques for analyzing plastic sensory-motor coordinations", *J. Comparative and Physiol. Psych.*, 1961 54 No. 1 pp. 33-37.

Held, R., and Freedman, S.J., "Plasticity in human sensorimotor control", *Science*, 1963 142 No. 3591 pp. 455-462.

Held, R., and Hein, A., "Movement-produced stimulation in the development of visually guided behavior", *J. Comparative and Physiol. Psych.*, 1963 56 No. 5 pp. 872-876.

Helmholtz, H.v., *Handbook of Physiological Optics, Vol. 3*, (Leipzig), 1867.

Henneman, E., "Peripheral Mechanisms involved in the control of muscle", In: *Medical Physiology, Vol. 2, Ch. 73*, Mountcastle, V.B., (ed)(C.V. Mosby, St. Louis, 1968), pp. 1697-1716.

Hilgard, E.R. and Bower, G.H., *Theories of Learning*, (Prentice-Hall, Inc., New Jersey, 1975).

Holst, E.v., "Relations between the central nervous system and the peripheral organs", *Brit. J. Animal Behavior*, 1954 2 pp. 89-94.

Horn, B.K.P. and Inoue, H., "Kinematics of the MIT-AI-Vicarm manipulator", *MIT Working Paper 69*, May 1974.

Kahn, M.E., "The near-minimum time control of open-loop articulated kinematic chains", *Stanford Aritficial Intelligence Memo No. 106*, December, 1969.

Kelley, D., *Manual and Automatic Control*, (J. Wiley & Sons, Inc., New York, 1968).

Kornheiser, A.S., "Adaptation to laterally displaced vision: a review", *Psychological Bulletin*, 1976 83 No. 5 pp. 783-816.

Landau, I.D., "Model reference adaptive systems -- a survey (MRAS) -- What is possible and why?", *J. Dynamic Systems, Measurement, and Control, trans. of ASME*, June, 1972 pp. 119-132.

Lashley, K.S., "The problem of serial order in behavior", In: *Cerebral Mechanisms in Behavior*, Jeffres, L.A., (ed.)(Wiley, 1951), pp. 112-136.

Levine, W.S., of University of Maryland, Department of Electrical Engineering, Personal Communication of 9/30/75.

Lozano Perez, T., "The design of a mechanical assembly system", *MIT Artificial Intelligence Technical Report No. 397*, December, 1976

MacKay, D.M., "Visual stability", *Investigative Opthalmology*, 1972 11 No. 6 pp. 518-524.

Marr, D., "A Theory of cerebellar cortex", *J. Physiol.*, 1969 202 pp. 437-470.

Marr, D., "How the cerebellum may be used", *Nature*, 1970 227 pp. 1224-1228.

Marsden, C.D., Merton, P.A., and Morton, H.B., "Servo action in human voluntary movement", *Nature*, 1972 238 pp. 140-143.

Marsden, C.D., Merton, P.A., and Morton, H.B., "Stretch reflexes and servo actions in a variety of human muscles", *J. Physiol., London*, 1976 259 pp. 531-560.

Mason, M.T., "Force feedback for manipulation", Masters thesis proposal, August, 1977.

McGeoch, J.A., *The Psychology of Human Learning*, 2nd Ed. Revised by Irion, A.L. (David McKay Co., Inc., New York, 1952).

Mednick, S.A., *Learning*, (Prentice-Hall International, Inc., London, 1964).

Melvill Jones, G. and Watt, D.O.D., "Observations on the control of stepping and hopping movements in man", *J. Physiol.*., 1971 219 pp. 709-727.

Melvill Jones, G. and Watt, D.O.D., "Muscular control of landing from unexpected falls in man", *J. Physiol.*., 1971 219 pp. 729-737.

Meriam, F.L., *Dynamics* (John Wiley & Sons, Inc., New York, 1966).

Merton, P.A., "How we control the contraction of our muscles", *Scientific American*, 1972 226 No. 5 pp. 30-37.

Miles, F., and Fuller, J.H. "Adaptive plasticity in the vestibulo-ocular responses of the rhesus monkey", *Brain Research*, 1974 80 No. 3 pp. 512-516.

Mittelstaedt, H., "The analysis of behavior in terms of control systems", In: *Group Processes*, Transactions of the Fifth Conference, Princeton, New Jersey, Schaffner, B., (ed.), 1958 pp. 45-84.

Mountcastle, V.B., Lynch, J.C., Georgopoulos, A., Sakata, H., and Acuna, C., "Posterior parietal association cortex of the monkey: command functions for operations within extrapersonal space", *J. Neurophysiol.*, 1975 pp. 871-908.

Murphy, J.T., Wong, Y.C., and Kwan, H.C., "Afferent-efferent linkages in motor cortex for single forelimb muscles", *J. Neurophysiol.*, 1975 38 pp. 990-1014.

Nashner, L.M., "Adapting reflexes controlling the human posture", *Exp. Brain Research*, 1976 26 pp. 59-72.

Noble, B., *Applied Linear Algebra*, (Prentice-Hall, Inc., New Jersey, 1969 pp. 229-273.

Paul, R., "Modelling trajectory calculation and servoing of a computer controlled arm", *Stanford Artificial Intelligence Memo No. 177*, November, 1972.

Peiper, D.L., "The kinematics of manipulators under computer control", *Stanford Artificial Intelligence Memo No. 72*, October, 1968.

Pew R.W., "Human perceptual-motor performance", In: *Human Information Processing: Tutorials in Performance and Cognition*, Kantowitz, B.H., (ed.), (John Wiley & Sons, N.Y., 1974), pp. 1-39.

Polit, A., "Development of reaching", unpublished report, 1976.

Polit, A., and Bizzi, E., "Properties of the motor programs underlying visually triggered arm movements in monkeys", To be presented at *Society for Neurosciences*, 1977.

Raibert, M.H., "A state space model for sensorimotor control and learning", *MIT Artificial Intelligence Memo No. 351*, January, 1976.

Raibert, M.H., "Control and learning by the State Space Model: experimental findings", *MIT Artificial Intelligence Memo No. 412*, March, 1977.

Raibert, M.H., and Horn, B.K.P., Work in progress. 1977.

Ralston, H.J., "Energetics of human walking", In: *Neural Control of Locomotion*, Herman, R.M., Grillner, S., Stein, P.S.G., and Stuart, D.G., (eds.), (Plenum Press, New York, 1976), pp. 77-98.

Roderick, M.D., "Discrete control of a robot arm", *Stanford Artificial Intelligence Memo No. 287*, August, 1976

Rust, B., Burrus, W.R., and Schneeberger, C., "A Simple algorithm for computing the generalized inverse of a matrix", *Communications of the ACM*, 1966 9 No. 5, pp. 381-387.

Schultz, D.G., and Melsa, J.L., *State Functions and Linear Control Systems*, (McGraw-Hill Book Co., New York, 1967)

Severin, F.V., Orlovsky, G.N., and Shik, M.L., "Work of the muscle receptors during controlled locomotion", *Biophysics, USSR*, 1967 12 No. 3 pp. 575-586.

Sobel, I., "On calibrating computer controlled cameras for perceiving 3-D scenes", *Artifiical Intelligence*, 1974 5 No. 2 pp. 185-198.

Speckert, G., "Hand eye coordination", *MIT Artificial Intelligence Working Paper No. 127*, July, 1976.

Szekely, Gy., Czeh, G., and Voros, Gy., "The activity pattern of limb muscles in freely moving normal and deafferented newts", *Exp. Brain Res.*, 1969 9 PP. 53-62.

Taub, E. and Goldberg, I., "Prism adaptation: control of intermanual transfer by distribution of practice", *Science*, 1973 180 pp. 755-757.

Townsend, A.L. Jr., "Linear control theory applied to a mechanical manipualtor", Masters Thesis, MIT Department of Mechanical Engineering, January, 1972.

Tsypkin, Y.Z., *Adaptation and Learning in Automatic Systems*, Translated by Nikoloic, Z.J., (Academic Press, New York, 1971).

Tsypkin, Y.A., *Foundations of the Theory of Learning Systems*, Translated by Nikolic, Z.J., (Academic Press, New York, 1973).

Twitchell, T.E., "The automatic grasping responses of infants", *Neuropsychologia*, 1965 3 pp. 247-259.

Twitchell, T.E., "Reflex mechanisms and the development of prehension", In: *Mechanisms of Motor Skill Development*, Conolly, K., (ed.)(Academic Press, London, 1970), pp. 25-37.

Waters, R.C., "A mechanical arm control system", *MIT Artificial Intelligence Memo No. 301*, January, 1974.

Welch, R.B., "Research on adaptation to rearranged vision", *Perception*, 1974 3 p. 367-392.

Welford, A.T., *Fundamentals of Skill*, (Methuen & Company, LTD, London, 1968).

White, B.L., "Experience and the development of motor mechanisms in infancy" In: *Mechanisms of Motor Skill Development*, Conolly, K., (ed.)(Academic Press, London, 1970), pp. 95-132.

Whitney, D.E., "State space models of remote manipulation tasks", Ph. D. Thesis, MIT Department of Mechanical Engineering, January, 1968.

Whitney, D.E., "Optimum step size control for Newton-Raphson solution of non-linear vector equations", *IEEE trans. of Automatic Control*, October, 1969 pp. 572-574.

Whitney, D.E., "The mathematics of coordinated control of prosthetic arms and manipulators", *J. of Dynamic Systems, Measurement, and Control*, December, 1972 pp. 303-309.

Wilkinson, A.D., "Visual-motor control loop: a linear system?", *J. of Experimental Psychology*, 1971 89 pp. 250-257.

Will, P.M., "Computer controlled mechanical assembly", *IBM Research Report RC-5428*, May, 1975.

Young, L.R., "On adaptive manual control", *Ergonomics*, 1969 12 No. 4 p. 661.

Young, L.R. and Stark, L., "Biological control systems -- a critical review and evaluation: developments in manual control", NASA, Cr-190, 1965.

Appendix A: Experimental Conditions

Fig.	Practice Prototype	τ	Coordinates	Arm Orientation	Inversion Criterion
1.4	PR-12	10	joint	upright	550
1.5	PR-11	-	joint	upright	550
1.6a	PR-20→24	10	XYZ	horizontal	200
1.6b	PR-20→24	10	XYZ	horizontal	200
1.7	PR-20→24	10	XYZ	horizontal	200
5.1	PR-11	10	joint	upright	550
5.2	PR-11	10	joint	upright	550
5.3	PR-12	10	joint	upright	550
5.4	PR-12	10	joint	upright	550
5.5	PR-17	10	joint	upright	550
5.7	PR-10→13	10	joint	upright	550
5.8	PR-10→13	10	joint	upright	550
5.9	PR-10→13	10	joint	upright	550
5.11	PR-20→24	10	XYZ	horizontal	200
5.12	PR-20→24	10	XYZ	horizontal	200
5.13	PR-20→24	10	XYZ	horizontal	200
5.14	PR-20→24	10	XYZ	horizontal	200
5.16a	PR-11	10	joint	upright	550
5.16b	PR-12	10	joint	upright	550
5.18	PR-11	10	joint	upright	550
5.19	PR-11	-	joint	upright	550
5.20a	PR-11	10	joint	upright	550
5.20b	PR-12	10	joint	upright	550
5.24	PR-11XYZ	10	XYZ	upright	250

Glossary of Terms

actuators	Devices which cause motion by exerting forces. Usually muscles, motors, hydraulics, etc.
adaptation	The adjustment of a control law appropriate to changes in the mechanical or sensory properties of the plant.
arm	A mechanical device, usually with serial degrees of freedom in the form of links separated by joints. Used to move objects and apply forces. Synonyms: *limb, manipulator.*
biological arm	A type of arm grown from biological materials.
CMAC	Cerebellar Model Articulation Controller: a controller proposed by Albus. (Recently renamed: Cerebellar Model Arithmetic Computer.)
controller	An information processing device which produces signals designed to produce a desired response in another system.
Coriolis term	A torque exerted about one axis of an orthogonal set caused by simultaneous movement about the other two axes. Sometimes used loosely in this report to include centrifugal terms.)
CSM	The Configuration Space Method of control. A method of control currently under investigation by Berthold K. P. Horn and myself.
efference copy	A record of the commands issued to a set of actuators.
error correction	A learning mechanism that compares the desired response with the actual response.
exafference	Afference caused by movement in the environment not produced by the organism.

Hyper-region A portion of 2N-dimensional space for which one set of constants, J and K, are stored.

internal dynamic model A component of a controller that mimics behavior of the plant under control.

internal inverse dynamic model A component of a controller which, when provided with a plant's response, describes the corresponding input.

generalization The process whereby the general level of motor performance is improved as the result of a particular set of practice data. Exhibited as *transfer*.

limb See *arm*.

man-made arm An arm manufactured by man, usually made from metals and plastics.

manipulator See *arm*.

movement A change in the positions of an arm's joints. Also refers to the path the arm makes during the change.

parametric variable An independent variable held constant during parameterization.

parameterization The process of simplifying a functional relation by substituting a set of constant values for a subset of the independent variables.

plant Any dynamic system.

prototype An internal representation of a desired trajectory

re-afference The sensory signals produced by an internally generated movement.

state The position and velocity of an arm.

state space memory A quantized, multi-dimensional memory organized by state variables.

SSM The State Space Model.

tabular equations A set of equations for which a table of coefficients are required for evaluation. The equations are usually simplified. Synonym: *parameterized equations.*

trajectory The path an arm takes during a movement.

transfer Behavior characterized by improvement of one task after practice of another. Closely related to *generalization.*

translator A device which converts descriptions of desired trajectories into appropriate motor commands. Also used in this thesis to include the mechanisms which allow acquisition of this function.

Glossary of Variables

AL	A constant internal to the practice algorithm which influences the degree of variability found in the practice movements.
B	A term in the equations of motion representing the force of frictional.
C	A term in the equations of motion representing the Coriolis force.
G	A term in the equations of motion representing the gravitational force.
J	A term in the simplified equations of motion which represents equivalent moment of inertia matrix.
K	A term in the simplified equations of motion which represents the net effect of gravity, friction, and Coriolis forces.
τ	The state space memory's time constant.
LI	The learning index.
M	The number of divisions on each dimension of the state space memory.
N	The number of joints of a limb.
N'	The number of measurement vectors contributing to one inversion.
P	The parameterization indicator.
RMS AE	Root-mean-square acceleration error.
RMS FPE	Root-mean-square final-position error.
RMS MCE	Root-mean-square motor-current error.
RMS PE	Root-mean-square position error.
T_m or T	Actuator torque vector.
X	Index of invertability
X_p	The independent variables of a function that have been parameterized.
X_v	The indep. variables of a function that have not been parameterized.
α	A parametric value of θ.
β	A parametric value of $\dot{\theta}$.
Δt	One time slice. Usually 60 msec.
θ	The position vector.
$\dot{\theta}$	The velocity vector.
$\ddot{\theta}$	The acceleration vector.
$\ddot{\theta}_D$	The desired acceleration vector.

This empty page was substituted for a blank page in the original document.

CS-TR Scanning Project
Document Control Form

Date : 3/21/96

Report # AI-TR-439

Each of the following should be identified by a checkmark:
Originating Department:

☒ Artificial Intellegence Laboratory (AI)
☐ Laboratory for Computer Science (LCS)

Document Type:

☒ Technical Report (TR) ☐ Technical Memo (TM)
☐ Other:_____

Document Information

Number of pages: 182 (187-IMAGES)
Not to include DOD forms, printer intstructions, etc... original pages only.

Originals are:
☐ Single-sided or
☒ Double-sided

Intended to be printed as :
☐ Single-sided or
☒ Double-sided

Print type:
☐ Typewriter ☐ Offset Press ☐ Laser Print
☐ InkJet Printer ☐ Unknown ☒ Other:_____

Check each if included with document:

☐ DOD Form ☐ Funding Agent Form ☒ Cover Page
☐ Spine ☐ Printers Notes ☐ Photo negatives
☐ Other: _____

Page Data:

Blank Pages(by page number): FOLLOWS PAGE 181

Photographs/Tonal Material (by page number): 24, 80

Other (note description/page number):

Description : Page Number:

(A) IMAGE MAP: (1-187) UNN'ED TITLE PAGE, UNN'ED REVISION PAGE 3-181, UNN BLANK, SCANCONTROL, COVER, TRGT'S (3)

(B) MANY PAGES HAVE XEROXING MARKS.

(C) PAGES 167-170 OF THE DOCUMENT WERE OUT OF ORDER. THEY WERE SCANNED IN THE CORRECT ORDER BY THE SCANNING OPERATOR.

Scanning Agent Signoff:

Date Received: 3/21/96 Date Scanned: 3/28/96 Date Returned: 4/4/96

Scanning Agent Signature: Michael W. Cobb

Scanning Agent Identification Target

Scanning of this document was supported in part by the **Corporation for National Research Initiatives,** using funds from the **Advanced Research Projects Agency** of the **United states Government** under Grant: **MDA972-92-J1029.**

The scanning agent for this project was the **Document Services** department of the **M.I.T Libraries.** Technical support for this project was also provided by the **M.I.T. Laboratory for Computer Sciences.**

Scanned

Date: 3/29/1996

M.I.T. Libraries
Document Services