A Model for Command Generation in Motor Cortex

by

Ola Ayaso
B.E. Electrical Engineering 1999
American University of Beirut

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2001

Author ....
Department of Electrical Engineering and Computer Science
August 2001

Certified by ....
Munther Dahleh
Professor
~~Thes~~is Supervisor

Accepted by ....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

A Model for Command Generation in Motor Cortex

by

Ola Ayaso

B.E. Electrical Engineering 1999

American University of Beirut

Submitted to the Department of Electrical Engineering and Computer Science
on August 2001, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

In this thesis, I suggest a model for motor command generation in the sensorimotor cortex during unobstructed and unloaded horizontal point-to-point movements of the arm through air. The first step towards modeling the motor cortex, the "chief executive" of the brain, was to understand, by modeling the arm behavior during the task under consideration, the nature of the commands that need to be generated by the sensorimotor cortex and issued to the arm. The two versions of the equilibrium point models of the arm, lambda and alpha, are explored; but eventually, a second order model is adopted in the simulations with the proportional integrator command generator (PICG) model of the sensorimotor cortex. The PICG model suggested here is based on as parsimonious assumptions as possible, that remain true to the anatomy of the brain and are neuro-biologically reasonable; still, simulations of the PICG, together with the second order arm model, match the kinematic and the neurophysiological data of humans and monkeys reasonably well. Furthermore, the PICG model allows for adaption or learning via tunable gains.

Thesis Supervisor: Munther Dahleh
Title: Professor

Acknowledgments

I would like to extend my sincere appreciation to all those who, either directly or indirectly, made this work possible.

Thank you to my advisors, Professor Dahleh and Professor Massaquoi, for their guidance and patience.

Thank you to my friends and family for their support and encouragement.

Contents

1 Introduction 13

1.1 Literature Review . . . 13

1.1.1 Anatomy of the Brain . . . 13

1.1.2 Motor Cortex: Encoding Muscle and/or Movement Parameters 17

1.1.3 Movement Strategies and Motor Control Hypotheses . . . 19

1.2 Problem Motivation and Formulation . . . 29

2 Arm Models 33

2.1 Motor Spaces . . . 33

2.1.1 Muscle space . . . 34

2.1.2 Joint space . . . 36

2.1.3 Task space . . . 37

2.2 Skeletal Dynamics . . . 38

2.3 Muscle Models . . . 39

2.3.1 Alpha-type model . . . 40

2.3.2 Lambda-type model . . . 48

2.4 Single Joint Arm . . . 50

2.4.1 Equilibrium point trajectories . . . 52

2.4.2 Comparing the α and λ single joint arm models . . . 55

2.5 Double Joint Arm . . . 58

2.6 Arm Model Activation Signals . . . 61

2.7 Summary . . . 62

3 Command Generation in the Sensorimotor Cortex 65
3.1 Neuronal Network Model 66
3.1.1 Remote Inputs of Neurons 69
3.1.2 Local Interactions of Neurons 72
3.1.3 Inhibition, Excitation, and Interneurons 75
3.1.4 Network Output 77
3.2 The PI Command Generator 78
3.2.1 Simulated Features of Arm Kinematics 82
3.2.2 Simulated Features of Neuronal Activity 87
3.3 Summary 92
4 Future Work 95
4.1 Evaluation of the Work Done 96
4.2 Improvements to the Model 98
A Simulink Simulations 101
Bibliography 107

List of Figures

1-1 Summarized connection diagram of various brain structures. 14

1-2 Block diagram showing the motor control loops 16

1-3 Lambda model invariant characteristic 24

1-4 Simple block diagram of the system 31

2-1 Definition of muscle-space variables. 35

2-2 Definition of inertial parameters. 36

2-3 Force versus length curves for the Alpha and Lambda models. 40

2-4 The musculotendon actuator. 41

2-5 Force-length and force-velocity curves for the contractile element. . . 43

2-6 Block diagram of the muscle fiber model. 45

3-1 Firing rate profiles of sensorimotor cortical neurons 67

3-2 Abstract model of the neuron 68

3-3 The population coding as a result of the layout of remote connections 71

3-4 Local interactions improve directivity of the population code 74

3-5 Firing rate profiles of sensorimotor cortical neurons: Inhibition 76

3-6 Block diagram of the PICG-arm model 79

3-7 Sample trajectories simulated using the PICG-arm model 83

3-8 Tangential hand velocity simulated using the PICG-arm model 84

3-9 Varying the g parameters in the PICG-arm model 85

3-10 Simulated hand trajectories using optimal g values 86

3-11 Movement simulated at different speeds using the PICG-arm model . 88

3-12 Simulated hand trajectories for $d = 0.1m$ and $d = 0.2m$ 89

3-13 Firing rate profiles of PICG neurons 90

3-14 Directional tuning of neurons in the PICG model 91

3-15 Population coding of neurons in the PICG model 92

A-1 Simulink block diagram of the PICG-arm model 104

List of Tables

2.1 The terms of the linearized α and λ muscle models 57

2.2 Alpha model parameters 58

2.3 Lambda model parameters 58

2.4 Skeletal parameters 60

2.5 Muscle-related model parameter values for the six muscles 61

Chapter 1

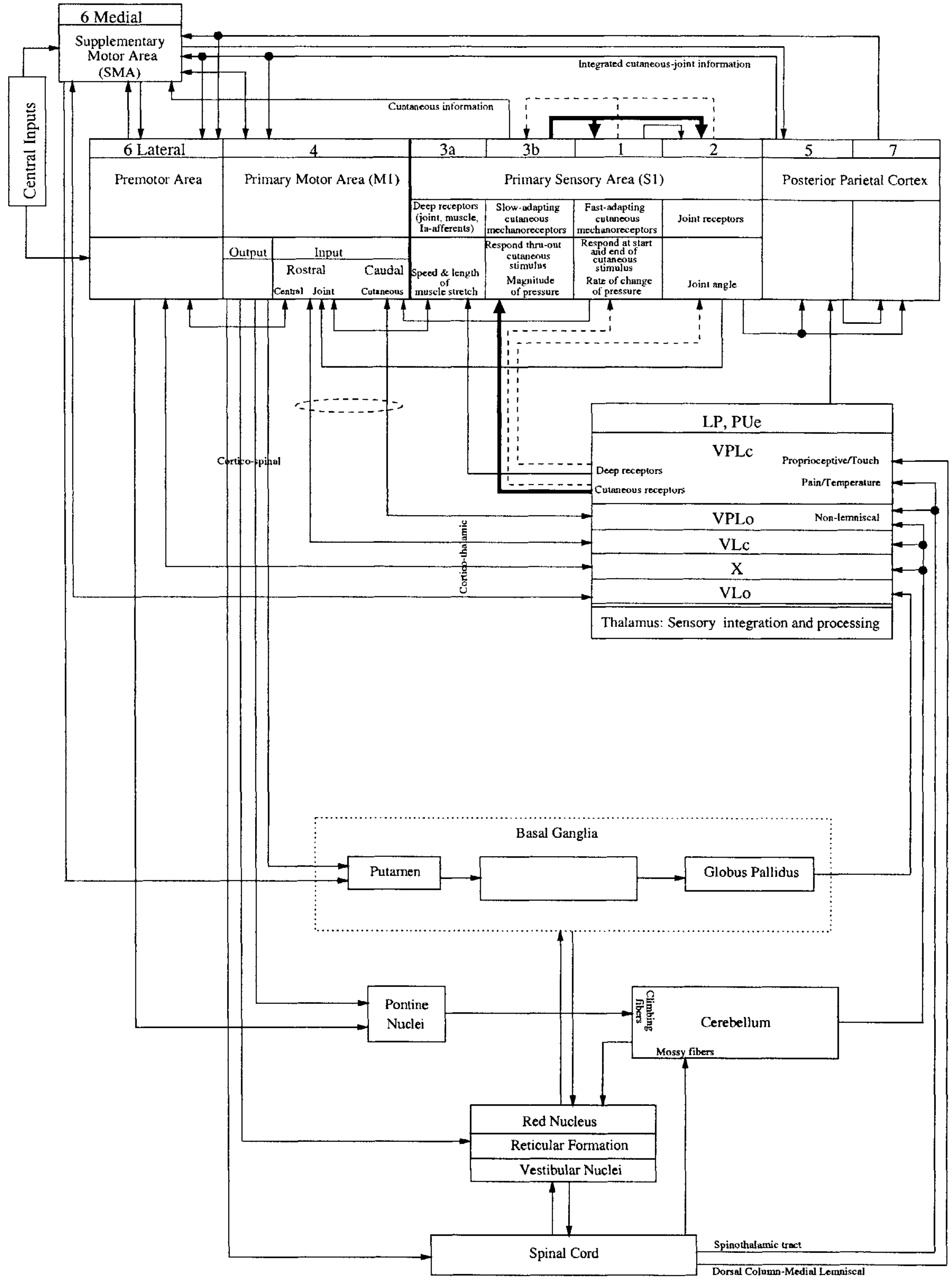
Introduction

1.1 Literature Review

Though many have measured signals from the motor cortex, what variables these signals encode is still a debatable issue. The topology of the brain, however, is well known. The neural projections to and from the various areas have been studied in some detail (and are summarized in figure 1-1) for knowing the connections between areas helps understand the functional properties of the neurons. The understanding of the anatomic connections of the sensorimotor cortex is the first step in attempting to understand how the brain controls movements and postures and, in particular, how the motor cortex generates and issues commands.

1.1.1 Anatomy of the Brain

Even the simplest of motor tasks, such as arm reaching that is unobstructed, and not subject to any external loads, seems to involve a complex sequence of activations. At the highest level of the functional motor hierarchy proposed by Brooks in [7], the association cortex, which includes parietal cortex (areas 5 and 7), could be responsible for formation of motor plans, strategies and intentions. In fact, neurophysiological studies indicate that the parietal cortex may be involved in planning movement in extrapersonal space and in the integration of sensory information with central com-



Legend

▬	Dense projections	——	Density of projections not specified
- - - - -	Weak projections		
⬭	Interaction: Lesions of dorsal column nuclei (dcn) eliminate M1 responses; dcn signals necessary for short latency discharges in M1		

References

1. Porter, R. Handbook of Physiology- Nervous system II. Chapter 22.
2. Evarts, E. Handbook of Physiology- Nervous system II. Chapter 23.
3. Brooks, V. The Neural Basis of Motor Control. pp. 200, 208, 211, 216, 218.
4. Kandel et al. Principles of Neural Science. 2000. Ch. 18, 24, 25, 36.

Figure 1-1: Summarized connection diagram of various brain structures.

mands to follow up on on-going movements as area 5 receives inputs from both the thalamus and area 2 of the primary sensory cortex (see below) and projects its outputs to the premotor area (lateral area 6), supplementary motor area (medial area 6), and motor cortex (area 4) [9].

At the middle level, the sensorimotor cortex, cerebellum, putamen loop of the basal ganglia, and brainstem are possibly involved in transforming the strategies of the higher level to the needed movement parameters, like movement direction or trajectory. Brooks suggests that this level may also be responsible for transforming parameters expressed in external world coordinates to joint coordinates. The output signals are then issued out from the motor cortex to the muscles via corticospinal neurons that synapse on the motoneurons in the spinal cord, the lowest level of the motor hierarchy. At the spinal cord, spinal circuitry maps the input corticospinal signal to the appropriate muscle activations hence muscle stiffness (or invariant force-length characteristic). The motoneurons innervate muscle units which send feedback information back to the motor cortex through the thalamus which processes the afferent information then sends one copy of the information to the motor cortex and another to the sensory cortex for further processing and then projection to the motor cortex. The afferent information is possibly used by lower motor loops, such as the cerebellar loop, to regulate motion and by the motor cortex to update motor plans. Some of the loops involved in the motor control are shown in the block diagram of figure 1-2.

The sensorimotor areas include the primary motor cortex (area 4), the primary sensory cortex (areas 3a, 3b, 1, and 2), and the premotor and supplementary motor areas (area 6). Area 6 receives projections from areas 5 and 4 and makes projections to the same areas. It is believed to take part in programming motor subroutines and planning and executing well learned movements using sensory information that it receives from parietal cortex. The parietal cortex receives processed sensory information from both the thalamus and area 2 of the primary sensory area. The primary sensory area consists of 4 areas that receive projections from the thalamus: area 3a, which receives deep receptor inputs (that is from joints and Ia-afferents that con-

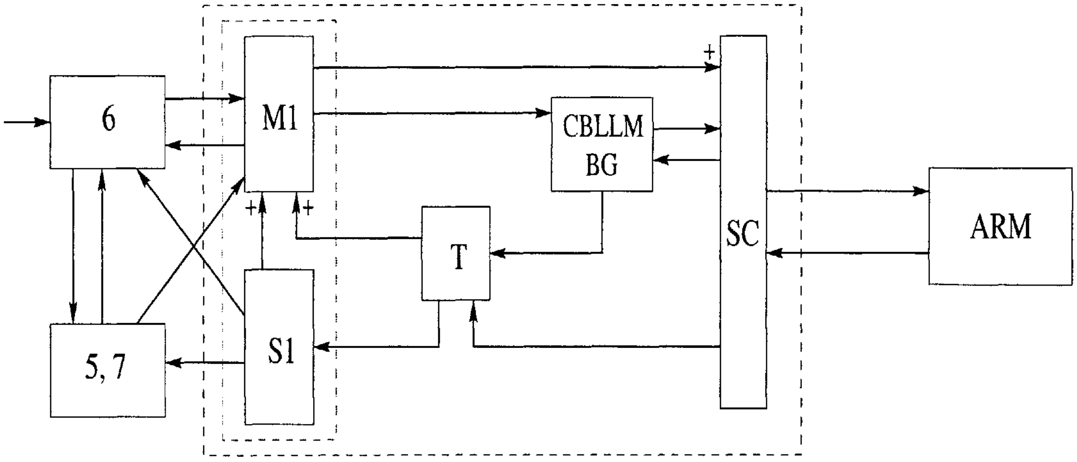


Figure 1-2: Some of the feedback loops that are involved in motor control are shown in this diagram. 6: premotor area, 5, 7: posterior parietal cortex, M1: primary motor area, S1: primary sensory area, T: thalamus, CBLLM: cerebellum, BG: basal ganglia, SC: spinal cord. The M1-SC-Arm-S1 loop is the subject of investigation in this work.

vey information about the speed and magnitude of muscle stretch); area 3b, receiving inputs from slow adapting superficial (cutaneous) mechanoreceptors that convey magnitude of pressure information; area 1, receiving inputs from fast adapting cutaneous mechanoreceptors that convey rate of change of pressure information; and finally area 2, receiving joint receptor information about joint angles. Only area 2 makes output connections from the primary sensory cortex to parietal areas 5 and 7, though each of the primary sensory areas do send projections to the motor cortex [7] [33] [43].

In fact, the motor cortex neurons receive this somatosensory feedback from the body regions that are closely affected by the muscles that the same motor cortex neurons innervate. This correspondence between motor fields and afferent input zones is known as tight input-output coupling [43]. Interestingly, the motor cortex also receives its own copy of afferent information from the thalamus. The motor cortex also receives transcortical projections from various parts of the cerebral cortex. The output connections of the motor cortex include nonpyramidal targets (such as the basal ganglia, thalamus, red nucleus, and reticular formation) and pyramidal neurons, or corticospinal neurons that innervate motoneurons and spinal interneurons in the spinal cord. These corticospinal neurons may either diverge from a single cortical focus to different spinal motoneurons such that the same point in motor cortex can activate several muscles, or converge from distinct cortical foci onto the same spinal motoneurons such that the same muscles can be activated from cortical points that may be located several millimeters apart [7].

1.1.2 Motor Cortex: Encoding Muscle and/or Movement Parameters

The variables that the sensorimotor cortex encodes are not known. The input signals to sensory cortex and output signals from the motor cortex and their correlations with task variables such the position of a limb in space, speed of motion of the limb, or force exerted on that limb have been measured by many (like [20], [19], [22], [44], and [9]). But these correlations do not necessarily imply causal or controlling relationships. There is debate about what variables the signals actually represent and how the message is conveyed by motor cortex neurons to motor systems such as the muscles in the arm performing the simplest of all tasks, reaching from point A to point B in a straight line as quickly and as accurately as possible.

In general, it has been observed that the firing of cells, motor cells in studies of vertebrate movements and sensory cells in studies of reflexes of invertebrates, is:

1. coarsely tuned with respect to the parameters of resulting movement, like force, amplitude, or direction in the case of motor neurons, and with respect to the features of the stimuli they encode as in the case of sensory neurons, and

2. widely distributed in that many sensory or motor cells that are located in widely distributed areas of the nervous system are active simultaneously before and during movement.

Yet this cell activity leads to accurate movements like arm reaching or saccadic eye movement in monkeys and leech local bending or cockroach orientation escape. So this rules out the possibility that individual highly specialized cells are responsible for behaviors and suggests that aggregate groups of neurons, called populations, are involved [47].

The vector averaging hypothesis was first suggested by Sparks and colleagues (1974) when they observed that in the monkey superior colliculus, SC, which is responsible for orienting eye movements (saccades), each neuron fires before a broad range of saccades and a large number of these neurons is active before each saccade. Yet, the direction and amplitude of the saccades are precisely controlled. Their claim was that the direction and amplitude of the resulting saccade would be determined by the direction and magnitude of the sum of the vectors associated with populations of active neurons; each vector has direction of the population's preferred direction (described below) and amplitude proportional to the firing rate of that population. Predictions of the hypothesis were tested by activating and deactivating specific groups of neurons with known functional properties and seeing whether the observed behavior was consistent with the predicted behavior. The outcomes supported the vector averaging hypothesis for the SC. Population coding evidence was also found in various other motor systems studied like turtle scratching and monkey arm reaching [47].

Georgopoulos found that neuron populations in the motor cortex encode direction of arm reaching movement [22] [21]. Each population has a preferred direction of firing; neurons are said to be "directionally tuned". When the arm moves in the preferred direction of some population of neurons, the firing rate of that population is stronger than firing would be if the arm was moving in any other direction; in fact, firing rate is broadly tuned around that preferred direction, θ_p. One possible tuning function is a cosine, $f(\theta) = a\cos(\theta - \theta_p) + b$, where f is the firing rate, a and b are constants, and θ is the direction of motion of the arm in the horizontal plane relative to the sagittal plane. Thus, for any given direction, the population's firing rate is modulated by a cosine centered on the preferred direction.

Now, for any given reaching movement, a number of populations with different preferred directions will fire with rates of varying intensity. If each population's firing is represented by a vector in the direction of that population's preferred direction and of magnitude proportional to the firing rate at the direction of current movement as determined by the tuning curve, then, Georgopoulos found, the sum of all the population vectors, $\vec{\nu}(\theta) = \sum_{\text{all populations}} f_i(\theta) \angle\theta_{p_i}$, will be a vector in the same direction of the actual movement, $\angle\vec{\nu} = \theta$. Furthermore, if the arm is perturbed on its route to the destination, the sum of population vectors will be in the direction the arm needs to move to overcome the perturbation and get back on track. Interestingly, Prud'Homme and Kalaska have found that sensory cortex neurons exhibit roughly the same type of population vector encoding in unloaded reaching movements [44].

Thus, Georgopoulos' results suggest that the motor cortex encodes direction of arm movement for unloaded arm reaching movements. Yet, there is debate about what variables the motor cortex signals actually represent; the variable encoded by the brain in the vectoral manner does not necessarily need to be direction. Mussa-Ivaldi shows that the brain could be encoding other muscle state variables, like desired rate of muscle shortening for example [41]. Whatever the variable itself may be, question arises as to how the brain performs such vectoral encoding when the only variable of its outgoing signals that it really physically controls is the rate of the firing of its output neurons. More importantly, question arises as to how the motor cortical signal is used to control and execute movements and what its significance is in the context of a certain model of the arm.

1.1.3 Movement Strategies and Motor Control Hypotheses

Since the task of having to calculate, possibly compensating for external and joint interaction forces, muscle torques needed for the desired movement is computationally complex, it is generally believed that the central nervous system simplifies the task of controlling movement by using certain motor control strategies, set relationships between different movement parameters. These strategies result in the experimentally observed movement stereotypes such as the relationship between movement duration, amplitude and accuracy, or the relationship between movement curvature and velocity.

It is believed that the brain uses relatively simple strategies that translate into the required signals to control muscles involved in arm movement. For example, according to Gottlieb et al., reaching movements are either speed sensitive, for which reaching the target within a certain time limit is paramount, or speed insensitive, for which accuracy is paramount and speed is not an issue [27] [28]. For speed sensitive movements the time, $t_{0.9} - t_{0.1}$, it takes for the joint angle (which is assumed to follow a smooth sigmoid, as experimentally observed) to rise from 10% to 90% of its final value, θ_f, is held constant regardless of the movement amplitude. To achieve this, the speed of the movement has to be increased as the movement amplitude is increased. On the other hand, under the speed insensitive strategy, the duration of the movement does not matter so the speed of the movement is chosen such that desired accuracy is achieved. Thus, as the movement amplitude is varied, the speed is held constant; in this case, the farther the final target (larger movement amplitude) the longer it will take to reach that target.

These different strategies require different signals from the brain. Based on experimental evidence they collect, Gottlieb et al. hypothesize that when the movement is speed sensitive, the brain adjusts amplitude of the signal to the alpha motoneuron pool (or more precisely rate of firing) to achieve different reaching speeds and amplitudes under the strategy; and when the movement is speed insensitive, the brain adjusts the duration of the signal to the muscle motoneurons [27] [28]. Although Gottlieb's strategies work out nicely for single joint movements, similar conclusions are not as readily made for multiple joint movements [1].

A more widely accepted idea is the equilibrium point hypothesis, originally proposed by Feldman [15]. According to this hypothesis, the central nervous system (CNS) specifies to the muscles required equilibrium positions, usually assumed to be in (virtual) joint angle coordinates; to produce motion, these equilibrium positions are shifted. Two versions of the hypothesis have emerged. The alpha version, attributed to Bizzi, suggests that the CNS regulates and controls the level of activity to alpha motoneuron pools of muscles [35]; but this hypothesis has been criticized by the Feldman camp and its supposed weaknesses that are discussed in detail in [15] are discussed below. Feldman and colleagues suggest a lambda version of the equilibrium hypothesis instead. This hypothesis claims that the nervous system regulates and controls the threshold muscle length of the stretch reflex, that is, the muscle length at which the recruitment of alpha motoneurons begins. Movement occurs when the CNS shifts the thresholds for agonists and antagonists in opposite directions and co-contraction results when the thresholds are shifted in the same direction.

The Alpha Model

In [5], Bizzi describes his experiments on monkey head movements and arm reaching movements. In the first group of experiments the monkey was trained to move its head in response to a visual cue, which triggered the movement but did not remain to guide the movement, so a certain final head position was not explicitly required. Head trajectories and neck muscle EMGs were measured under the following conditions: when there was no load added to the head; when a constant unexpected load was applied after movement onset, then removed again before movement termination; and when an inertial load was unexpectedly applied such that the movement trajectory was perturbed, but the load was not a steady state disturbance. Similar experiments were carried out on deafferented monkeys. The second group of experiments on monkeys performing arm reaching was very similar. The monkeys were required to point to a target light and hold the arm at that position. Visual feedback of the arm was not allowed. Trials were carried out on both normal and deafferented monkeys for unloaded movements and when unexpected positional disturbances were applied at random times after movement onset.

For the head movement experiments, the added constant load caused an undershoot of the final head position that would have been reached in an unloaded movement, even though EMG activity did increase when the load was added; however, removing the load caused the head to reach the final head position. Application of an inertial load, on the other hand, did not change the final head position that would have been reached in an unloaded movement but did cause an overshoot of head position before reaching the final position. EMG signals increased as the load was added, but a silent period was observed during the overshoot period. Experiments on deafferented monkeys gave similar results, and final head position was reached despite the fact that there were no visual or vestibular feedback; but since the monkeys were deafferented, EMGs for unloaded and loaded movement were similar showing the absence of the stretch reflex. The results of the monkey arm reaching experiments were similar. The monkeys always reached the intended final position regardless of whether or not they were deafferented.

These observations led Bizzi to hypothesize that final head position is preprogrammed and not dependent on proprioceptive information the CNS receives during the movement. The CNS controls posture (movement) by setting (varying) the level of activity of the alpha motoneurons of agonists and antagonists, which determines the stiffness of these muscles. By setting the level of alpha activation, the CNS selects a set of tension-length curves of the agonist and antagonist, which consequently results, due to the interaction of the muscles, in the specification of the equilibrium point that will yield the desired end position. Thus, motor programming is a result of specification of equilibrium pints between agonists and antagonists rather than specification of amplitude or duration of movement; and the same mechanism of setting equilibrium points is used to control both posture and movement because "according to this hypothesis, movements are at the simplest level transitions in posture" [5].

The findings that final arm or head positions were reached regardless of applied disturbances even in deafferented monkeys seem to support the equilibrium point hypothesis but are not meant to undermine the importance of afferent feedback in the control of movement. Bizzi finds that the deafferented monkeys have significantly poorer pointing accuracy when the initial posture of the animal was changed. Thus Bizzi suggests that the afferent information may play a role in selecting programmed motor/neural patterns. Finally, a point is made that the suggested equilibrium point hypothesis does not explain the mechanism whereby the CNS determines movement velocity and thus there may coexist some other parallel mechanism.

The Lambda Model

The lambda hypothesis was born out of Feldman's unloading experiments in which the arm of the subject started at a certain position under a load, which was then gradually removed [11] [12] [13] [15]. The subject was given the instructions "not to intervene voluntarily to forearm deflections when the load is removed" and the subject's elbow angle was recorded under the different load conditions. Feldman's recordings yielded a family of curves that resembled static spring like relationships between the exerted elbow muscle torque and the joint angle, each curve's zero torque point located at a different point along the angle axis, shown in figure 1-3. In the rest of the discussion, the curves are discussed in terms of force-muscle length instead of torque-elbow angle as that transformation is definitely plausible.

Feldman termed each curve an "invariant characteristic" (IC) because for a given lambda, the point at which torque departs from zero, a unique curve is specified. When the subject made voluntary changes in the position of the elbow, a transfer from one IC to another was observed. Thus lambda was assumed to be the independent variable that was set by central control. It is generally agreed that static muscle force depends on both muscle length and activation by the CNS; since lambda seems to be a centrally controlled variable and neighboring ICs travel in parallel, Feldman hypothesizes that muscle activation is actually proportional to the difference between the muscle length and lambda, the muscle is active when the muscle length exceeds the threshold length. This fact is observed experimentally as the EMG activity becomes more intense for points on the IC that are farther away from the threshold, lambda.

The variable lambda has various interpretations. Since force starts to develop in the muscle at muscle lengths that exceed lambda, it is a threshold length beyond which motoneurons are recruited thereby activating more muscle fibers at faster rates and increasing muscle force; it is the threshold of the stretch reflex. But there is also a bioelectrical interpretation. When the muscle is relaxed, some motoneuron that innervates it will have a certain sub-threshold membrane potential. As the muscle is stretched, the membrane potential of the motoneuron increases due to the stretch reflex. As the muscle keeps stretching, at some point the membrane potential will reach its threshold value and the motoneuron will fire tonically and activate muscle fibers that start to actively produce force. Lambda is the value of muscle length at which the membrane potential of the motoneuron exceeds its threshold and starts to fire. Here, one must note that muscles are innervated by several motoneurons,

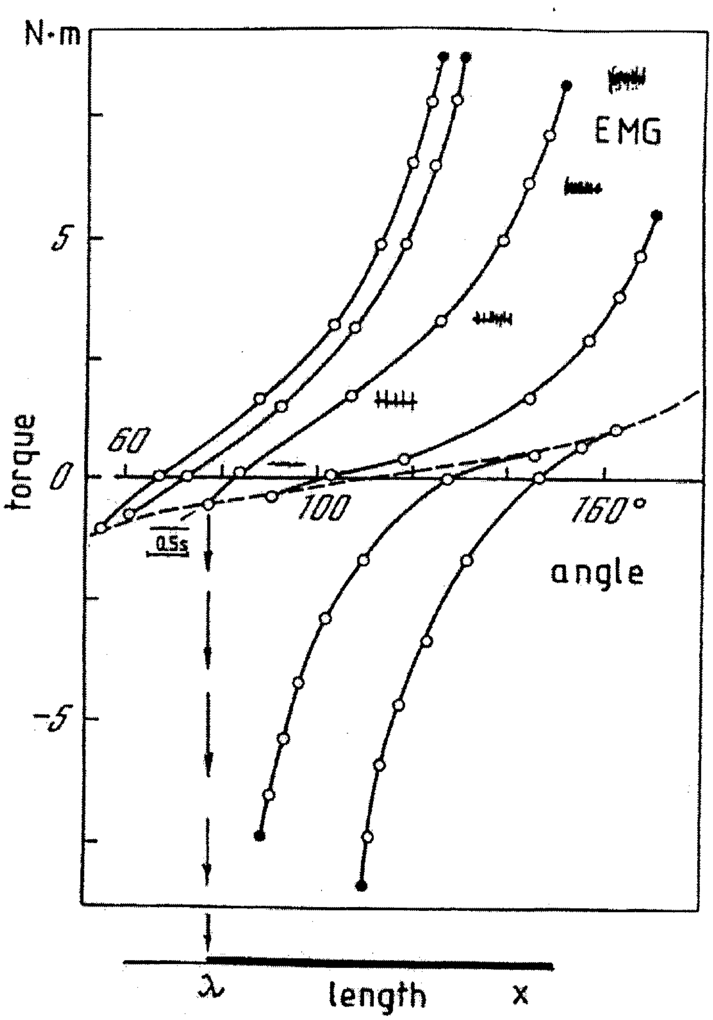


Figure 1-3: Feldman's recordings of arm position in his "do not intervene" unloading experiments yielded this family of invariant characteristics (ICs). The upper curves are the elbow flexor ICs and the lower curves correspond to elbow extensors. At separate trials, the arm was started at the solid circles and settled at the open circles as the load was gradually removed. The recorded EMG activity is shown for one IC: the activity decreases from the starting position of the arm as the biceps shortens until the IC merges with the passive elbow muscle characteristic (dashed line) [13].

each having a different threshold membrane potential. According to the hypothesis, lambda corresponds to the lowest threshold among the various motoneurons, but all thresholds of motoneurons are interrelated in an unambiguous manner.

The CNS controls posture by setting the values of lambda for various muscles. An equilibrium point, or stable muscle length, is determined by the interaction of the load and muscle; in particular, the intersection of the load-length and the muscle force-length curves. However, there are requirements for postural stability: first, the slope or stiffness of the muscle IC at the equilibrium point needs to be larger than the load gradient at the same point so that perturbations from the equilibrium point will cause the system to return to the equilibrium; and secondly, there should be sufficient damping or change in muscle force with change in velocity, which is indeed a characteristic of muscles, so that the system does not oscillate around the equilibrium point.

Under this hypothesis, posture and movement are controlled by a single mechanism. To produce voluntary movement, the CNS varies the setting of lambda thereby producing shifts in the equilibrium points. However, dynamic muscle force depends on length, activation, and velocity of stretch (tension developed drops with speed of shortening). Feldman introduces the dependence of muscle force on velocity by introducing a dynamic speed-dependent threshold of activation, λ^*, that is related to the static threshold, lambda.

Experimental data shows that a rapid extension of a muscle at rest produces a phasic excitation in the muscle, that is, the EMG increases briefly then decreases in a bell shaped manner. Thus, activation of the muscle is observed even though it was initially at rest (that is, muscle length is less than static lambda) which means that the dynamic threshold, λ^*, can be lower than the static lambda during muscle lengthening, or positive velocities. On the other hand, when the muscle is tonically active (meaning that its length is greater than the static threshold lambda hence producing some constant level of excitation), sudden muscle unloading causes a silent period in the EMG, which indicates that in dynamics, and during muscle shortening or negative velocities, the activation threshold can be greater than the static threshold.

Thus, based on the experimental data, the hypothesis put forward is that $\lambda^* = \lambda - \mu \cdot v$, where μ is a parameter that is set by the CNS to control the amount of proprioceptive velocity feedback information, and the assumption is that the relationship between λ^* and the muscle stretch velocity, v, is linear. Applying this hypothesis to predict the EMGs that would be observed during motor performance yields EMGs that are similar to those experimentally observed.

Comparing the Models

Both lambda and alpha versions of the equilibrium point hypothesis claim that the CNS varies the force-length (f-l) spring-like characteristic of muscle such that a desired stable resting point is reached when the muscle interacts with the load. The major point of contention between the models concerns the interpretation of the CNS activation of muscles, or how the CNS varies the f-l characteristics. While the lambda model asserts that the CNS varies the threshold of the stretch reflex, or the resting length of the muscle at which motoneurons will begin to be recruited, the alpha version claims that the CNS varies the alpha activation of motoneurons, which alters muscle stiffness. In terms of the effect of activation on the "invariant characteristics", in the lambda version, the f-l curves are shifted along the length axis almost in parallel due to different specifications of lambda, whereas in the alpha version, the slope of the f-l curve will change as alpha activation changes but the resting length of muscle does not change.

A striking consequence of this difference in the definitions of activation is the cause and effect relationships between the equilibrium position and the EMG or muscle activation. In the lambda model, the CNS elicits voluntary movements when it sets the desired threshold length by activating both alpha and gamma motoneurons. This causes a shift in the invariant characteristic, or f-l curve, and an equilibrium point is reached upon interaction with the load. Muscle activation, or EMG, is assumed to be proportional to the muscle length minus the threshold length. In fact, different points on the invariant characteristic are associated with different EMG levels. Thus, EMG patterns are consequences of the shift in equilibrium points; this is the absolute reverse of the alpha model's claim that the EMG settings determine the equilibrium points.

And it is precisely this claim that lends weakness to this model, that is, that the alpha model is unable to balance isoelectric loads or loads with increasing EMG gradient. For example, when the load is isoelectric the EP is either undefined or not uniquely determined; the load and muscle curves will, at best, coincide. On the other hand, for loads with positive EMG gradients, the system will be unstable at the points of intersection with muscle characteristics if no matter what activation the CNS provides, the load is such that the slope of the load curve is always greater than the slope of the muscle curve at the point of intersection. This would happen for example if the load curve intersects the length axis at a greater value of length than the resting length of the muscle.

Furthermore, since the CNS set EMG level, which essentially means setting stiffness of muscles, loads near zero "cannot be lifted at all despite intense variation of muscle stiffness" [13]; but clearly, the real system handles light loads well and has trouble with heavy loads. Finally, in the alpha model, the resting length of the muscle is assumed constant regardless of muscle activation, but in reality, muscle resting-lengths lie inside the range of lengths at which the muscle could be active. Since the resting length in Bizzi's model is constant, the muscles are unable to generate force at lengths below the resting length no matter what the CNS does.

But this is not to say that the lambda version of the hypothesis is unchallenged. Though Gottlieb agrees that the core of modern theories of movement is the idea of movable equilibrium positions, such that in the absence of external forces the system will approach and rest at its equilibrium position, he claims that the lambda version of the equilibrium hypothesis is inadequate [26]. Gottlieb points out three weaknesses of this hypothesis. First of all, Gottlieb claims that a major assumption necessary to translate the lambda commands of the various muscles into the R and C commands of the joints (that supposedly set the joint equilibrium position and the joint stiffness) as Feldman describes them is that agonist and antagonist muscles have identical force-length and force-velocity characteristics, and this is simply not true except for few joints. Thus, the relationships between the joint R and C commands and the muscle lambda commands need to be redefined.

Secondly, in order for movements to end smoothly at their targets, simulations based on this hypothesis require joint viscosities that are much higher than experimentally derived values. Finally, the equilibrium point hypothesis provides detailed rules on how the nervous system may set the reciprocal command, R, that supposedly sets the joint equilibrium position, but the other two variables, C, which sets the level of agonist/antagonist cocontraction or equivalently joint stiffness, nd μ, which controls the feedback gain, are not described, even though, according to the model, they are equally important in the control of movement but are neither obvious nor simple, if not impossible, to determine when the dynamics of the moving system are not to be explicitly considered (as a major claim of the equilibrium point hypothesis is that movement control is done by shifting the "frames of reference").

Thus, while there is agreement on muscle contractile mechanisms and characteristics such as the force-length and force-velocity relationships, there is much disagreement about how the central nervous affect these muscle characteristics. While it is clear that a muscle model should account for certain physiological aspects, such as motoneuron recruitment, reflex delays, calcium kinetics, and nonlinear muscle stiffness, it is not clear what rules the central nervous system uses to set muscle state variables such that the specific movement features such as distance, speed, and load are attained. But, the variable that is assumed to be the centrally controlled variable is a crucial part of any motor control model. For example, it has been shown that by controlling the threshold of stretch reflex, simple piecewise linear signals are sufficient to yield arm reaching movements [29]. Gribble claims that conclusions that the control signals need to be any more complex, like those drawn by Gomi and Kawato [24] [25] or which is obvious in the case of the alpha-model where the EMGs that are assumed to be the centrally controlled input usually have complex patterns, are faulty as they use an overly simplified model of the arm. In fact, using an accurate enough model that captures non-linearities has a significant effect on the shape of the required control signal [29]. Moreover, the second order model (such as that used by

Gomi and Kawato in [24] and [25]) is an approximation that merely captures some aspects of the behavior of the arm such as joint stiffness during movement, but is probably inadequate when it comes to modeling other variables [52]. Thus, one must take care in drawing conclusions from such models since they may accurately model some behaviors or quantities, but misrepresent others.

1.2 Problem Motivation and Formulation

The idea that population vectors encode task variables is a handy tool to predict the arm's state variables given observations of neuronal firing. In fact, it has been suggested in [16] and shown (in [41], [45], and [49]) that as long as there are enough neurons with a certain distribution of the "preferred directions" between them, the "direction" of the output population vector will always match the task variable under study. Furthermore, this match will occur regardless of whether the neurons actually have output effects on muscles or not; population vectors give no proof of neural coding as the observed neural activity need not contribute to movement, hence task parameters, in any way. So, unless the output connections of cells and their effects on muscles is known, the consequence of their activity during task performance will not be clear. Indeed, there is a need for "conceptual and computational work to understand how the activity of neurons within a motor map is used to generate temporal patterns of activity required by separate motoneuronal pools" [47]; and this could shed light on how the population "vector" translated into movement.

The population vector hypothesis is supported by experimental evidence, not underlying theory of how the CNS may generate command signals and represent state variables. Furthermore, claims that the motor cortex encodes one variable or another are based on observations that firing of neurons is correlated with some behavioral task; but correlations indicate neither causal nor controlling relationships. In fact, these claims are results of experiments that are subject to two different types of bias: "task-induced" bias and "selection" bias. "Task-induced" bias refers to the fact that the experimenters record neural activity as an animal performs a behavioral task that is designed to test a certain hypothesis and the activity of the cells is interpreted in relation to that task. "Selection" bias refers to the fact that experimenters tend to ignore cells that have firing patterns that are "paradoxical" or "un-interpretable" given the investigated task, and cells that do not show any modulation with the task. In fact, only cells that best illustrate the hypothesis are reported. Succinctly said, "a rarely acknowledged fact of life in the neurophysiology laboratory is that neurons in many regions, including the motor cortex, exhibit an enormous variety of responses, a fact that provides an opportunity to find cells related to any given functional hypothesis" [16].

Thus, question arises as to how the motor cortical signal is used to control and execute movements and what its significance is in the context of a certain model of the arm. What is needed is a physiologically accurate model (as opposed to artificial neural network models which do not necessarily reflect the way the biological system works) that will explain the command generating mechanism of the motor cortex in relation to the output signals that are needed by specific motoneuron pools.

To begin to address the questions mentioned above, for the purposes of this work, the motor control system for horizontal load-free arm reaching has been broken into two parts, as shown in figure 1-4: the plant, or the arm and spinal cord, and the command generator, the sensorimotor cortex. The problem was attacked backwards, starting at the lowest level of the hierarchy as it is probably the most well-understood part; thus, the more concrete platform that an arm model provides was chosen to be the point of departure of the investigation to gain insight into the problems and issues involved in the modeling of the motor control system, and in particular how the motor cortex generates the signals needed to execute movements and updates its commands as it receives proprioceptive feedback.

In chapter two, two variations of the equilibrium point hypothesis, the lambda and the alpha models, are explored. According to the equilibrium point hypothesis, there exists a map from the equilibrium point that might be specified by the CNS in terms of joint angles, to the actual input to the muscle. This map would have to be different for the two different models if they were to be driven from the same signal.

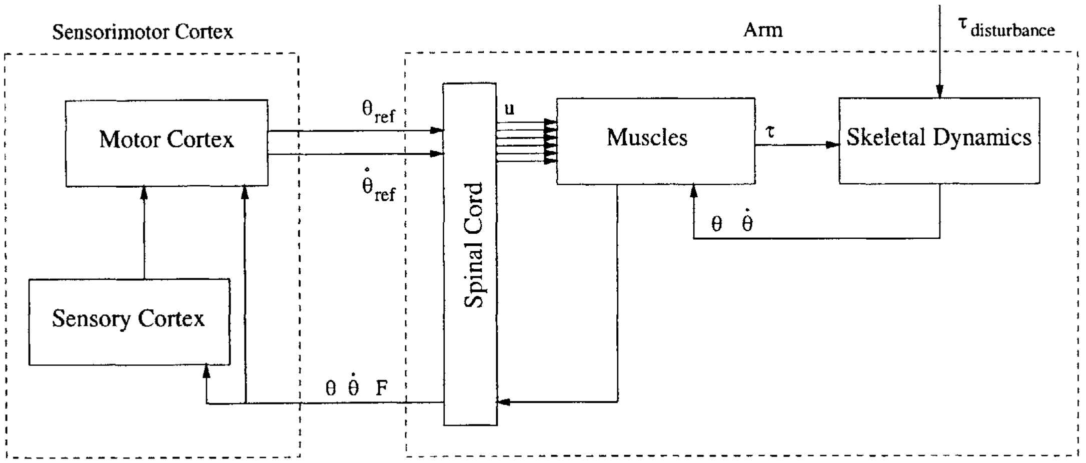


Figure 1-4: The system that is investigated is made up of a plant, the arm and spinal cord, and a command generator, the sensorimotor cortex. The various other components of the more complex multi-loop feedback system, such as cerebellum, thalamus and basal ganglia, are not shown here. The command generator issues the desired joint and position signals, θ_{ref} and $\dot{\theta}_{ref}$, to the spinal cord which scales and routes the signals to the muscles. The command generator receives joint position, θ, velocity, $\dot{\theta}$, and force, F, information from the plant.

So the map is explored in the context of the two specific models by determining the relationship between

- the equilibrium joint angle,
- the supposed muscle input signal, the alpha activation in the case of the alpha model and the threshold of the stretch reflex in the case of the lambda model,
- and the muscle driving signal.

The effect of musculoskeletal geometry and physiological properties of the muscles on these relationships is also investigated. Having done that, the reasonableness of assuming the existence of this map and its implementation by spinal circuitry is evaluated in the interest of arguing that using a simpler second-order arm model in simulations reduces modeling complexity, but, for the purposes of this work, does not forfeit behavioral or performance characteristics of the arm.

Assuming that an appropriate map exists from a centrally specified variable such as the equilibrium joint angle to the required muscle input, and that it is reasonable that the map might be implemented in spinal circuitry [6], then lumping the spinal cord together with a muscle-based arm model results in a second-order nonlinear "arm" model that receives as inputs equilibrium point trajectories, such as joint position θ_{ref} and velocity $\dot{\theta}_{ref}$. In this case, the task of the middle level of the CNS (sensorimotor and association cortices) would be to generate the needed equilibrium point trajectory. Where and how the equilibrium point is calculated by the brain is unknown. In chapter three, the possibility that the motor cortex issues the equilibrium point trajectory is explored. The known projections to and from the motor cortex and the observed cortical signals and their correlations with task variables underly the proposed command generating mechanism.

In chapter four, the work done is assessed for strengths and weaknesses. The issues that have arisen in the course of the investigation are discussed and future directions and plans are suggested.

Chapter 2

Arm Models

In order to make less complex and more well-defined the task of understanding and interpreting the observed signals in the sensorimotor cortex and then attempting to suggest how these signals are generated in the context of horizontal planar arm movements, the problem was first approached backwards. Given a certain model of the arm, what are the signals needed to drive the arm to produce the desired behavior, and how do these compare with the observed sensorimotor signals. In this chapter, the two variations of the equilibrium point hypothesis, the alpha muscle model and the lambda model are investigated.

2.1 Motor Spaces

The kinematics and dynamics of the arm can be described using different frameworks, or relative to different motor spaces: task, joint, and muscle. In building an overall model of the double joint (elbow and shoulder) arm, the conversions between the different spaces, though not always well-posed, are useful to move from the detailed experimentally derived descriptions of individual muscles, to the description of the joint angle kinematics, and finally the end-effector (hand) movement in space.

The inverse kinematic transformations from task space to joint space or from muscle space to joint space, are ill-posed. This is due to redundancy: the muscles, the "actuators" in the system that result in the positioning of the joints, are more numerous than the joints. Thus, there are several sets of allowable muscle lengths that will result in setting a desired joint angle; there are several sets of allowable joint angles that will result in setting the desired end-effector position. The inverse kinematics are usually solved using some optimization criteria and are sometimes subject to geometric constraints such as the allowed workspace and positions of the arm or the mechanical constraints of the limbs, such as the fact that the elbow cannot bend backwards.

The well-posed of the conversions are the forward kinematic transformations, from joint space to to task space or from joint space to muscle space. Below, the different spaces are described and the forward kinematic equations that will be useful later in deriving the overall arm model are presented.

2.1.1 Muscle space

The muscle space description consists of the length, q, and rate of change of length, $\dot{q}$, of each muscle. The muscle lengths can be related to the joint angles (joint space) under the simplifying assumption that when a muscle is stretched, it wraps around the joint like a string around a pulley. Thus, starting from some reference muscle length, q_{ref}, that corresponds to some reference joint angle, θ_{ref}, usually chosen to be the middle of the range of operation of the joint, the change in muscle length will be:

$$q - q_{ref} = m(\theta - \theta_{ref}). \tag{2.1}$$

Using vector notation, which useful in generalizing for different uses: for two muscles around a single joint in single joint flexion and extension motions or for the six muscle groups assumed to be involved in the two joint motion (see figure 2-1):

$$\mathbf{q} - \mathbf{q_{ref}} = M(\mathbf{\Theta} - \mathbf{\Theta_{ref}}) \tag{2.2}$$

where for the single joint, $\mathbf{q} = \begin{bmatrix} q_{agonist} & q_{antagonist} \end{bmatrix}'$ and $\mathbf{\Theta} = \theta_{joint}$; for the double joint, $\mathbf{q} = \begin{bmatrix} q_1 & q_2 & q_3 & q_4 & q_5 & q_6 \end{bmatrix}'$ and $\mathbf{\Theta} = \begin{bmatrix} \theta_{shoulder} & \theta_{elbow} \end{bmatrix}'$. m is the moment

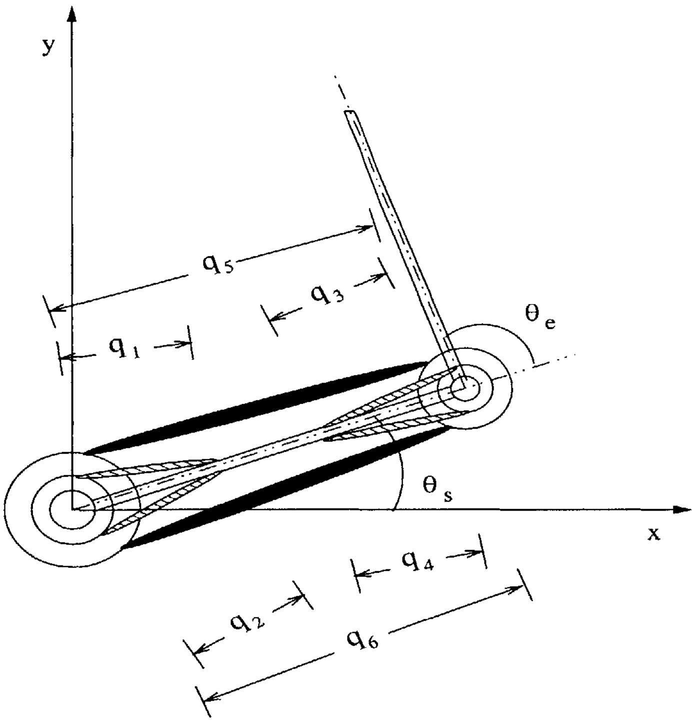


Figure 2-1: The muscle groups acting on the bones are shown; 1: shoulder flexors, 2: shoulder extensors, 3: elbow flexors, 4: elbow extensors, 5: biarticular flexors, 6: biarticular extensors.

arm, or "radius" of the joint, and M is a matrix containing the moment arms of the different muscles; assuming that the equivalent moment arms of the shoulder and elbow joints are constant (independent from the joint position), for the single joint,

$$M = \begin{bmatrix} -r_{agonist} & r_{antagonist} \end{bmatrix}' \tag{2.3}$$

and for the double joint,

$$M = \begin{bmatrix} -m_1 & m_2 & 0 & 0 & -m_{51} & m_{61} \\ 0 & 0 & -m_3 & m_4 & -m_{52} & m_{62} \end{bmatrix}'. \tag{2.4}$$

The numbering refers to the muscle groups shown in figure 2-1; and m_{ij} is the moment arm for biarticular muscle of group i at the shoulder ($j = 1$) or at the elbow ($j = 2$).

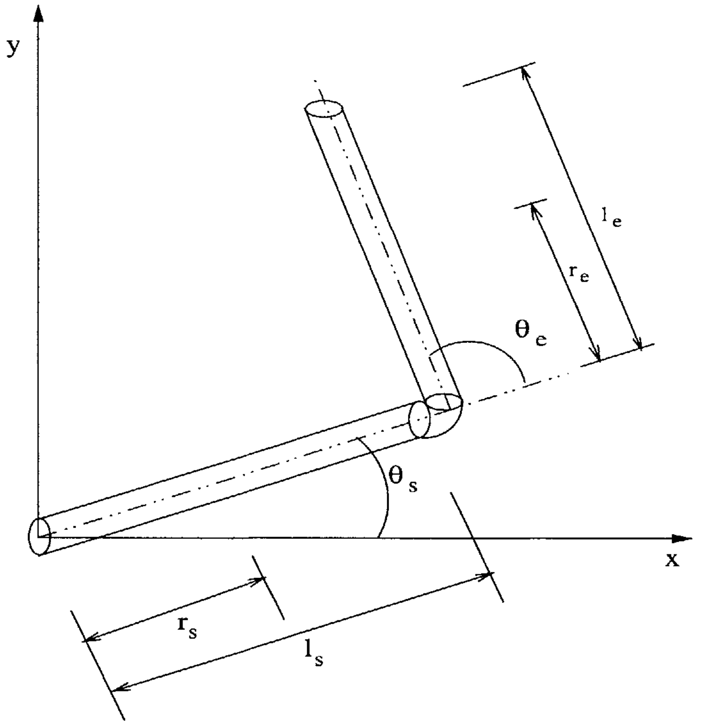


Figure 2-2: The arm skeleton is modeled as a two link rigid body.

2.1.2 Joint space

The joint space description consists of the joint angle magnitudes, θ, and angular velocities, $\dot{\theta}$. The forward kinematic relationships between the joint space variables and the muscle and task space variables are described in sections 2.1.1 and 2.1.3. But the *inverse* kinematic conversion is needed on several occasions, and a useful set of equations that relate the elbow and shoulder joint coordinates to the Cartesian and polar coordinates of the end-effector (task space) and work well near the center of the workspace are:

$$\begin{aligned} \theta_e &= \arccos\left(\frac{x^2+y^2-{l_s}^2-{l_e}^2}{2\cdot l_s\cdot l_e}\right) &&= \arccos\left(\frac{\rho_{ee}^2-{l_s}^2-{l_e}^2}{2\cdot l_s\cdot l_e}\right) \\ \theta_s &= \arctan\left(\frac{y}{x}\right) - \arctan\left(\frac{l_e\cdot\sin\theta_e}{l_e\cdot\cos\theta_e+l_s}\right) &&= \phi_{ee} - \arctan\left(\frac{l_e\cdot\sin\theta_e}{l_e\cdot\cos\theta_e+l_s}\right) \end{aligned} \tag{2.5}$$

where ρ_{ee} and ϕ_{ee} are the polar coordinates of the end-effector. Refer to figure 2-2 for the definitions of the rest of the variables.

2.1.3 Task space

Finally, the task space description consists of the Cartesian (x, y) or polar (ρ_{ee}, ϕ_{ee}) coordinates of the the end-effector, and the tangential velocity of the end-effector, v_t. The Cartesian x and y coordinates of the end-effector can be related to the shoulder and elbow angles, θ_s and θ_e of the joint space (refer to figure 2-2 for variable definitions):

$$\begin{aligned} x &= l_s \cos\theta_s + l_e \cos(\theta_s + \theta_e) && (2.6) \\ y &= l_s \sin\theta_s + l_e \sin(\theta_s + \theta_e). && (2.7) \end{aligned}$$

Taking differentials,

$$\begin{bmatrix} \mathrm{d}x \\ \mathrm{d}y \end{bmatrix} = \begin{bmatrix} -l_s \sin\theta_{s_o} - l_e \sin(\theta_{s_o} + \theta_{e_o}) & -l_e \sin(\theta_{s_o} + \theta_{e_o}) \\ l_s \cos\theta_{s_o} + l_e \cos(\theta_{s_o} + \theta_{e_o}) & l_e \cos(\theta_{s_o} + \theta_{e_o}) \end{bmatrix} \cdot \begin{bmatrix} \mathrm{d}\theta_{s_o} \\ \mathrm{d}\theta_{e_o} \end{bmatrix}.$$

In vector notation,

$$\mathrm{d}\mathbf{x} = J\,\mathrm{d}\mathbf{\Theta_o}. \tag{2.8}$$

where the Jacobian,

$$J = \begin{bmatrix} -l_s \sin\theta_{s_o} - l_e \sin(\theta_{s_o} + \theta_{e_o}) & -l_e \sin(\theta_{s_o} + \theta_{e_o}) \\ l_s \cos\theta_{s_o} + l_e \cos(\theta_{s_o} + \theta_{e_o}) & l_e \cos(\theta_{s_o} + \theta_{e_o}) \end{bmatrix} \tag{2.9}$$

is clearly dependent on the initial (unperturbed) shoulder and elbow positions, θ_{s_o} and θ_{e_o}. The joint angle differential, $\mathrm{d}\Theta_o = \Theta - \Theta_o$, represents small perturbations of θ_s and θ_e from their initial unperturbed positions.

Alternatively, taking the derivative with respect to time in equations 2.6 and 2.7 yields:

$$\dot{\mathbf{x}} = J\,\dot{\mathbf{\Theta}} \tag{2.10}$$

where $\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} & \dot{y} \end{bmatrix}'$ and $\dot{\mathbf{\Theta}} = \begin{bmatrix} \dot{\theta}_s & \dot{\theta}_e \end{bmatrix}'$. And the magnitude of the tangential velocity of the end-effector is $v_t = \sqrt{\dot{x}^2 + \dot{y}^2}$.

2.2 Skeletal Dynamics

The arm skeleton is usually modeled as a two-link rigid body (for example in [30], [17], [40] , [29], [24]), and the parameters, l_s, l_e, r_s, and r_e, and variables, θ_s and θ_e, that are used are shown in figure 2-2.

The torque, $T = \begin{bmatrix} \tau_s & \tau_e \end{bmatrix}'$, needed to displace the joints along a certain trajectory when the system is subject to some external disturbance torque is a function of the inertial dynamics of the links and the Coriolis forces that arise due to the interaction of the links. Using Newton-Euler method for the two-link body in a horizontal plane (as in [34]),

$$T = H(\Theta)\ddot{\Theta} + C(\Theta,\dot{\Theta})\dot{\Theta} + T_d \tag{2.11}$$

where $T_d = \begin{bmatrix} \tau_{ds} & \tau_{de} \end{bmatrix}'$, is a vector containing the disturbance torques in $N \cdot m$ at the shoulder and the elbow links respectively; Θ, $\dot{\Theta}$, and $\ddot{\Theta}$ are the vectors $\begin{bmatrix} \theta_s & \theta_e \end{bmatrix}'$, $\begin{bmatrix} \dot{\theta}_s & \dot{\theta}_e \end{bmatrix}'$, and $\begin{bmatrix} \ddot{\theta}_s & \ddot{\theta}_e \end{bmatrix}'$, containing the shoulder and elbow angular positions in rad, angular velocities in rad/s, and angular accelerations in rad/s^2.

$H(\Theta)$ and $C(\Theta, \dot{\Theta})$ are matrices that contain the inertial terms associated with centripetal and Coriolis forces:

$$H = \begin{bmatrix} I_1 + I_2 + m_s{r_s}^2 + m_e({l_s}^2 + {r_e}^2 + 2l_s r_e \cos\theta_e) & I_2 + m_e({r_e}^2 + l_s r_e \cos\theta_e) \\ I_2 + m_e({r_e}^2 + l_s r_e \cos\theta_e) & I_2 + m_e{r_e}^2 \end{bmatrix} \tag{2.12}$$

$$C = \begin{bmatrix} -(m_e l_s r_e \sin\theta_e)\dot{\theta}_e & -(m_e l_s r_e \sin\theta_e)(\dot{\theta}_e + \dot{\theta}_s) \\ (m_e l_s r_e \sin\theta_e)\dot{\theta}_s & 0 \end{bmatrix}. \tag{2.13}$$

Here, m_s and m_e, l_s and l_e, I_1 and I_2, and r_s and r_e are the masses in Kg, the lengths in m, the moments of inertia in $Kg \cdot m^2$, and the positions of the centers of mass in m of the shoulder and elbow links; refer to figure 2-2 for parameter definitions.

2.3 Muscle Models

The muscles are the "actuators" that provide the torque needed to move the passive arm skeleton. The force generated by each muscle is a function of the muscle length, q, rate of change in muscle length, $\dot{q}$, and input to the muscle by the central nervous system (CNS), u,

$$F_{muscle} \triangleq f(q, \dot{q}, u) \tag{2.14}$$

A first order approximation of this equation using the Taylor series expansion around some equilibrium trajectory, $(q_{eq}(t), \dot{q_{eq}}(t))$, gives:

$$F_{muscle} = f(q_{eq}(t), \dot{q_{eq}}(t), u) + K(q_{eq}(t), \dot{q_{eq}}(t), u)\,\mathrm{d}q + B(q_{eq}(t), \dot{q_{eq}}(t), u)\,\mathrm{d}\dot{q} \tag{2.15}$$

where

$$\begin{aligned} K &= \frac{\partial f}{\partial q} && (2.16) \\ B &= \frac{\partial f}{\partial \dot{q}} && (2.17) \end{aligned}$$

K and B are stiffness and viscosity terms that are not necessarily linear with respect to the CNS input, u.

The two muscle models that are investigated differ in their definitions of the CNS input, $u = N$, the level of activation of motoneurons for the alpha model, and $u = \lambda$, the threshold of the stretch reflex for the lambda model; refer to section 1.1.3 for detailed descriptions of N and λ. Proponents of the alpha model claim that the CNS activation modulates the stiffness of the muscle directly by setting the slope of the muscle force-length characteristic proportional to N, whereas the proponents of the lambda model claim that the CNS modulates the stiffness of the muscle indirectly by setting the threshold length of the stretch reflex, λ (see figure 2-3). In this section, the terms K and B in equations 2.16 and 2.17 for the two models and the input signals that drive the models, N and λ are compared.

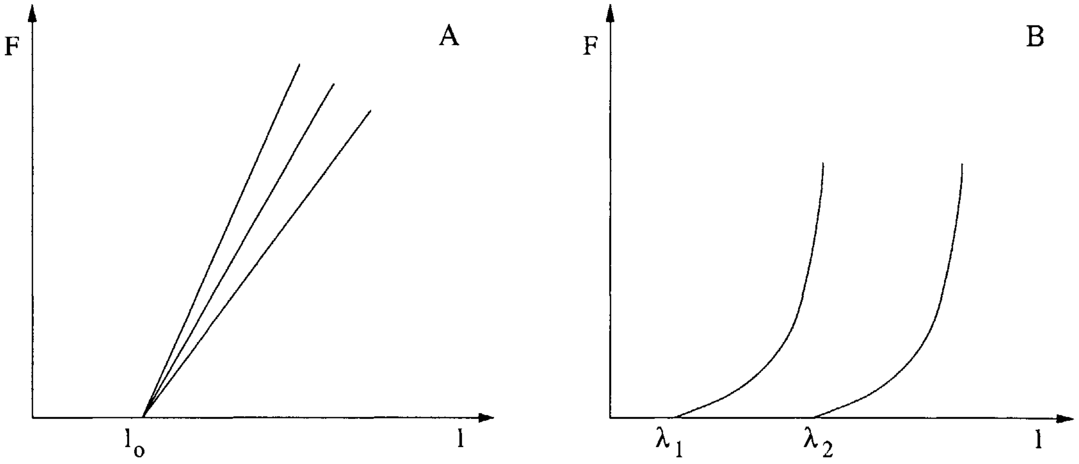


Figure 2-3: Plots for muscle force versus length in the static case, that is when the rate of change of length of the muscle is negligible, based on the alpha model (A) and the lambda model (B). A: According to the alpha model the CNS will set the slope of the F-l curve. B: According to the lambda model, the CNS sets the threshold muscle length for force generation,λ.

2.3.1 Alpha-type model

This model is based on the Hill model where the actuator, the muscle, is considered to be composed of two elements, the muscle fibers and the tendons, both functioning together to produce the force that is exerted on the skeleton, see figure 2-4. So, the muscle fiber and tendon properties that have been carefully studied experimentally and modeled in isolation need to be combined in order to accurately model the "musculotendon actuator" [53]. But adding the tendon to the model adds computational complexity that could be avoided for the purposes of this study by simply ignoring that element.

The tendon is modeled by a nonlinear spring, that is, the force produced is a function of the length of the tendon, [52] and [53]. The force produced in the muscle fibers is a function of both the muscle fiber length and the muscle fiber rate of change of length. Thus, because the two elements are connected in series, the total muscle force is:

$$\begin{aligned} F_{muscle} &= F_{tendon} &= F_{muscle\,fibers} \\ F_{muscle} &= g_t(L_t) &= g_m(u, L_m, \dot{L_m}) \end{aligned} \tag{2.18}$$

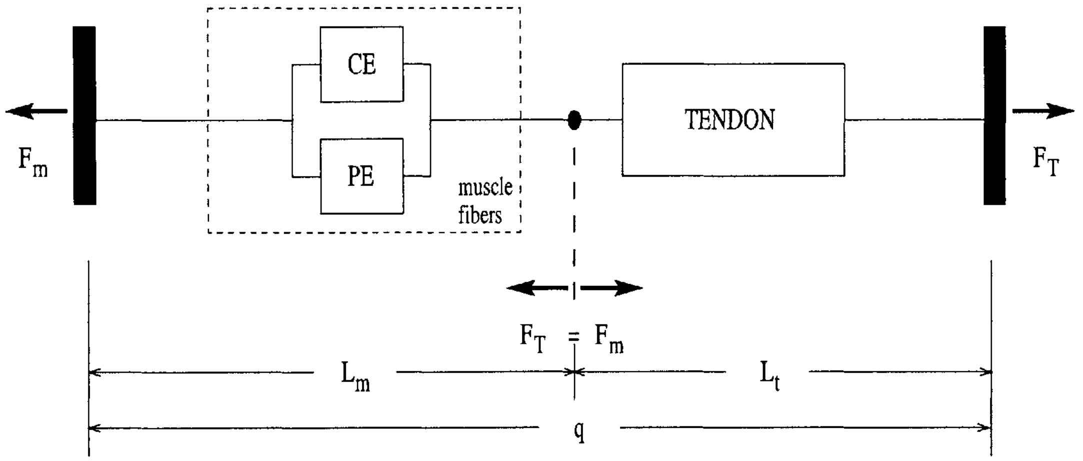


Figure 2-4: The musculotendon actuator is composed of two series elements, the tendon and the muscle fiber. The muscle fibers are modeled as two parallel elements, the contractile element (CE) and the passive element (PE). F_T is the force exerted by the tendon and F_m is the force exerted by the muscle. L_m is the length of the muscle fibers, L_t is the length of the tendon, and q is their combined length.

where $F_{musclefibers} = g_m(u, L_m, \dot{L_m})$ will be discussed in detail below, and the variables L_t and L_m are the lengths of the tendon and the muscle fiber respectively, see figure 2-4. But the total muscle length is $q = L_m + L_t$ and the rate of change of the muscle length is $\dot{q} = \dot{L_m} + \dot{L_t}$. So, in terms of the total muscle length,

$$F_{muscle} = g_t(L_t) = g_m(u, q - L_t, \dot{q} - \dot{L_t}). \tag{2.19}$$

When some given q and u are plugged into the functions g_t and g_m of equation 2.19, the result will be a differential equation in L_t which is usually nonlinear. Using numerical methods, this differential equation can be solved for L_t, which is then used in 2.18 to determine the muscle force F_{muscle}.

If the tendon is assumed to be much stiffer than the muscle fibers, then the muscle fibers will tend to respond to changes in length of the muscle, lengthening or shortening as needed, while the tendon length remains constant, $\dot{L_t} = 0$. This simplifies the calculations significantly in that for a given q and u, and since L_t is now a known parameter, F_{muscle} is easily found by direct substitution into equation 2.19.

In effect, the static force-length relationship of the muscle, $F_{muscle} = g_m(u, q - L_t, 0)$ vs. q, is now a shifted version of the muscle fiber force-length characteristic, $F_{muscle} = g_m(u, L_m, 0)$ vs. L_m, where the shift reflects the fact that the muscle length is equal to the fixed tendon length plus the varying muscle fiber length. Once this shift is taken into account, one can ignore the tendon.

In the Hill model, the muscle fibers are composed of two elements in parallel, the contractile element (CE), which models the sarcomeres that actively produce force by contracting in response to neural activation, and the passive element (PE), which models the passive properties of the muscle such as spring-like resistance to stretch.[1]The total muscle fiber force is equal to the sum of the forces produced by the two parallel elements.

$$F_{muscle\,fiber} = g_m(u, L_m, \dot{L_m}) = F_{PE}(L_m) + F_{CE}(u, L_m, \dot{L_m}) \tag{2.20}$$

The force produced by the contractile element is a function of muscle fiber length, change in length, and the neural activation as will be discussed below. The force produced by the passive element, which is frequently modeled as a non-linear spring as in [51] and [8], where:

$$F_{PE} = a \cdot (e^{c(L_m - L_o)} - 1). \tag{2.21}$$

For simplicity, here the passive force is assumed to be linear with respect to the muscle fiber length, as in [29]

$$F_{PE} = K_p \cdot (L_m - L_o) \tag{2.22}$$

where K_p is the passive stiffness of the muscle and L_o is the muscle resting length, see table 2.5 for the values used.

The dependence of the force produced by the contractile element on the muscle fiber length, L_m, and the rate of change of length, $\dot{L_m}$, is shown in figure 2-5. As in [51], [8], and [36], the contractile force can be broken down in to a static length-

[1] More detailed models also include a passive spring-like element in series with the contractile element (CE) to model the compliance of the cross bridges [52], [53], and [18]. This is neglected here as it adds significant but unneeded complexity to the model.

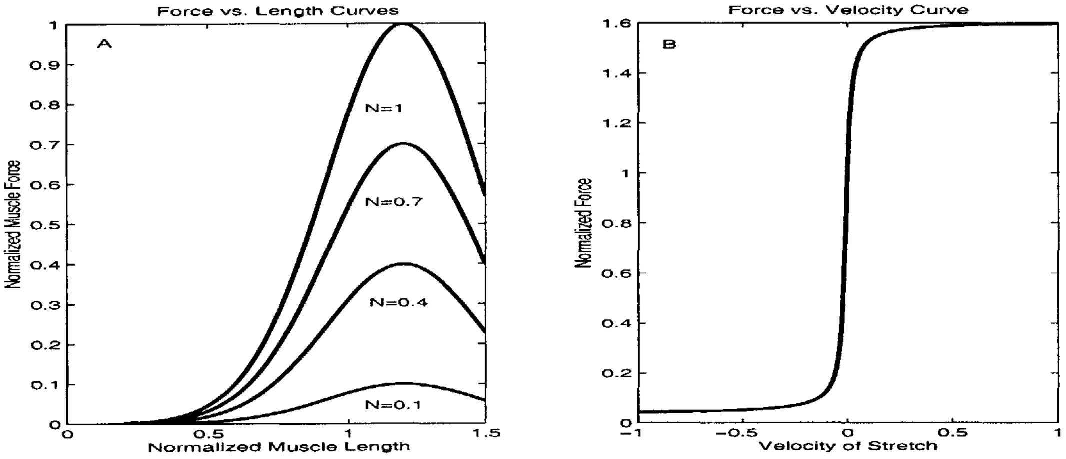


Figure 2-5: The force-length characteristics of the contractile element for different activation levels are shown in A. The force-velocity characteristics is shown in B.

and activation-dependent force that is then scaled by a velocity-of-stretch-dependent function:

$$F_{CE} = h_s(N, L_m) \cdot h_d(L_m, \dot{L}_m) \tag{2.23}$$

Here, the function used for the force-length relationship was a modified version of that used in [51]; the activation dependent shift of the curves in [51] was eliminated such that the curves resembled those of [8]. Still, the low activation curves, which are the ones that come into play for the no-load reaching movements, closely approximate the ones used in [51]:

$$h_s(N, L_m) = NF_{max}e^{-\left(\frac{\frac{L_m}{L_o}-1.2}{0.4}\right)^2} \tag{2.24}$$

where L_o is the muscle fiber length when the CNS input is zero (the *resting* length). It can be seen from this equation that the static force is directly proportional to the activation, N, of the contractile element. This can be also seen from figure 2-5, where the left-halves of the curves have slopes that are monotonically increasing in N. That is, the stiffness of the muscle is directly proportional to the activation, and this means that the model belongs to the Alpha-type category.

The velocity-of-stretch-dependent scaling function models the dependence of the contractile force on the velocity of stretch. Less force can be produced by the muscle fibers as they shorten. The function used here, from [29], is a good approximation of the various different functions used in the literature, for example in [51], [8] and [3]; but this is simpler because this single function models both shortening (by convention, $\dot{L_m} < 0$) and lengthening ($\dot{L_m} > 0$) as opposed to having two separate functions for the different cases:

$$h_d(\dot{L_m}) = c_1 + c_2 \arctan(c_3 + c_4 \cdot \dot{L_m}) \tag{2.25}$$

where, as in [29], c_1, c_2, c_3, and c_4 are 0.82, 0.5, 0.43, and 58 respectively.

Finally, the time dependence of force generation needs to be included in the model. This corresponds to the sluggishness or low-pass nature of the muscles, the gradual build-up of force in response to the input activation as a result of the excitation-coupling that is due to calcium activation dynamics. One can model this phenomenon in two ways, which are equivalent. Frequently, a low pass filter (inserted before block "a" in figure 2-6) is used to to shape the input activation signal, $u(t)$, as in [53], [3], and [36]. One could alternatively use a low pass filter to shape the force output from the force-length characteristic (that is, the filter, block "b" in figure 2-6, is inserted after block "a"), as in [29] and [18]. Whether block "b" comes before or after block "a" makes no difference because for any given muscle fiber length, L_m, block "a" is a simple gain (see equation 2.24). The filter used here is the one used in [29]:

$$H_{act}(s) = \frac{1/\tau^2}{(s + \frac{1}{\tau})^2} \tag{2.26}$$

where τ is equal to 15 ms.

The overall muscle model is shown in figure 2-6, where blocks "a", "b", "c", and "d" represent equations 2.24, 2.26, 2.25, and 2.22 respectively.

Activation inputs: The inverse muscle model

One of the reasons for which the functions of equations 2.24, 2.25, 2.26, and 2.22 were chosen instead of other functions that are frequently used in the literature is that they are relatively simple yet capture the desired muscle characteristics for the reaching

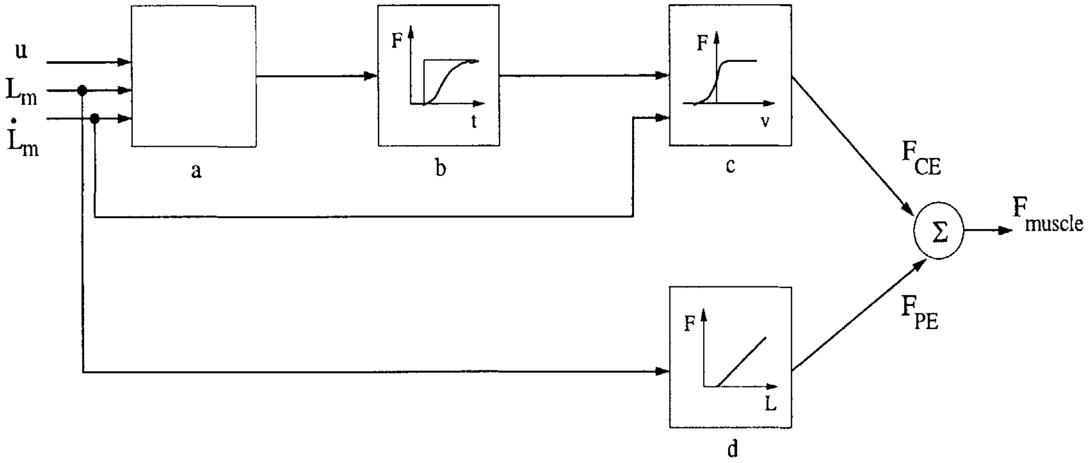


Figure 2-6: Block diagram, modified from [29], of the muscle fiber model; see text for details. Block "a" is intentionally left empty because the function that the block represents is different for alpha and lambda models.

movements. Because the functions are simple, it is easy to invert the overall muscle model, equation 2.20, and explicitly solve for the activation given a required level of force, F_d, for some muscle length L_d, and velocity of stretch, $\dot{L}_d$, provided that the low pass filtering is ignored:

$$N = g_m^{-1}(F_d, L_d, \dot{L}_d) \tag{2.27}$$

$$N = \frac{F_d - (K_p \cdot (L_d - L_o))}{F_{max} e^{-\left(\frac{\frac{L_d}{L_o} - 1.2}{0.4}\right)^2} (c_1 + c_2 \arctan(c_3 + c_4 \cdot \dot{L}_d))} \tag{2.28}$$

It turns out that the muscle activation that is calculated this way is not affected by the fact that the low pass filtering was ignored, and the actual muscle length and velocity of stretch track the desired ones almost perfectly.

The muscle force can be calculated from the skeletal dynamics equations; it is the force needed to move the end-effector through some distance in some desired direction, or equivalently the joint angles through some desired trajectory. The desired trajectory can be easily converted to the desired L_m and $\dot{L}_m$ using equations 2.2, refer to section 2.1.1. So, the desired activation is easily calculated from the skeletal dynamics and equation 2.28. Alternatively, the desired force maybe calculated without knowledge of the skeletal dynamics and based on the assumption of the equilibrium point control that the torque at the joint is equal to zero (see section 2.4.1).

Though this is useful of the purpose of analysis of the model and comparison with other models, there is no reason to believe that the central nervous system does indeed use either inverse dynamics to determine the required muscle force, nor an inverse muscle model in order to determine the feed-forward activation signals for the muscle.

The spinal reflex

The alpha muscle actuator model was derived without taking into account the effects of the stretch reflex; both the static and dynamic characteristics of the contractile element, equations 2.24 and 2.25, are experimentally determined for deafferented muscles [53] and [8]. Thus, the effect of the spinal reflex, the excitation of homonymous muscles via Ia afferent feedback and inhibition of the opposing muscles via the Ia afferents and inhibitory spinal neurons, needs to be added to the model. Ultimately the spinal reflex serves to increase the stiffnesses of the joint, refer to section 2.4.2.

As in [3], for each muscle, the excitatory activation that feeds back to the same muscle, N_e, and the inhibitory activation that feeds back into the opposing muscle, N_i, that arise due to the spinal reflex were found by solving two equations. The difference between the excitatory activation and the inhibitory activation was approximated by a second order function, and the two activations were related by a hyperbolic function:

$$\begin{aligned} N_e - N_i &= K_s(L_m(t-\delta) - x_s) + B_s(\dot{L_m}(t-\delta) - \dot{x_s}) && (2.29) \\ N_e &= \frac{{N_r}^2}{N_i}. && (2.30) \end{aligned}$$

Here, N_r is the inhibition constant; δ represents the delay in the spinal feedback loop, the difference between the time when L_m and $\dot{L_m}$ are available to the Ia afferents, and the time that the activation commands are issued to the muscles. x_s and $\dot{x_s}$ are "setpoints" that are issued by the brain; these signals were set to be equal to the desired L_m and $\dot{L}_m$, constant, or a combination of the aforementioned signals. In reality, these setpoints would correspond to the gamma activation of the muscles that tunes the sensitivity of the spindles, that produce the Ia afferents, to the stretch. Letting x_s and $\dot{x}_s$ follow the desired L_m and $\dot{L}_m$ is essentially equivalent to letting the spindles (that generate the Ia afferent signal) reflect the velocity of stretch of the muscle fibers because the resultant activations will be proportional to $\dot{L}_m$; when x_s and $\dot{x}_s$ are constant, the spindles reflect the magnitude of stretch of the muscle fibers; and the combination if signals allows the spindles to reflect both the velocity of stretch and magnitude of stretch of muscles.

Thus, the total muscle activation is the sum of the "feed-forward" CNS excitation, as in equations 2.23 and 2.27, the excitation due to feedback from homonymous (or the same muscle for simplicity) muscles, and inhibition from the opposing muscle, from equations 2.29 and 2.30. The activation for for muscle a whose action is opposed by b is therefore,

$$A_a = N_a + N_{e_a} + N_{i_b} \tag{2.31}$$

$$A_a = j_\alpha(F_{d_a}, L_{d_a}, \dot{L}_{d_a}) + \tilde{e} + \dot{\tilde{e}} \tag{2.32}$$

where

$$\begin{aligned} N_a &= j_\alpha(F_{d_a}, L_{d_a}, \dot{L}_{d_a}) &= g_{m_a}^{-1}(F_{d_a}, L_{d_a}, \dot{L}_{d_a}) \\ N_{e_a} &= \tilde{e} &= \zeta_a(L_{m_a}(t-\delta) - x_{s_a}, \dot{L}_{m_a}(t-\delta) - \dot{x}_{s_a}) \\ N_{i_b} &= \dot{\tilde{e}} &= \zeta_b(L_{m_b}(t-\delta) - x_{s_b}, \dot{L}_{m_b}(t-\delta) - \dot{x}_{s_b}), \end{aligned} \tag{2.33}$$

where the spinal "setpoints", x_{s_i} and $\dot{x}_{s_i}$, are linear combinations of L_{d_i} and $\dot{L}_{d_i}$; and the function ζ_i is a polynomial in its arguments because solving equations 2.29 and 2.30 simultaneously yields a quadratic in N_e (or N_i) . So, N_a is a function of the feed-forward signals and N_{e_a} and N_{i_a} are error-like signals.

2.3.2 Lambda-type model

The λ model used here differs slightly but significantly from the α-type model in that the difference between the two models is only in the function corresponding to block "a" in figure 2-6, but the interpretation of the CNS-issued driving signal, $u(t)$, for the different models is different. Here, $u = \lambda$, which essentially parameterizes the parallel force-length curves of 2-3B, represents not the activation of the muscle, but the threshold of its stretch reflex: the length that the muscle must be stretched to before it begins to generate an active force.

Thus, as in [48], [29] and [18], the muscle activation is a combination of the CNS-set threshold, λ, and the muscle length and velocity of stretch that are fed back by the Ia afferent fibers (and are therefore delayed by δ):

$$A = [L_m(t-\delta) + \mu \dot{L}_m(t-\delta) - \lambda(t) - \varrho]^+ \tag{2.34}$$

where

$$[x]^+ = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$$

that is, activation cannot be negative; μ represents the amount of velocity feedback available, which is probably set by dynamic gamma activation of the spindles; and ϱ is inhibition due to the opposing muscles. This activation will recruit motoneurons that will activate the muscle fibers to produce the contractile force:

$$F_{CE} = \rho(e^{cA} - 1) \tag{2.35}$$

where ρ and c are constants calculated such that the functions fit the experimentally determined muscle characteristics; see tables 2.5 and 2.3 for the values of these parameters.

Now, equation 2.35 is combined with 2.25, 2.26, and 2.22 to give the overall muscle model.

Activation inputs: The inverse muscle model

Again, as in section 2.3.1, for the sake of model comparisons and signal analysis, it was easy and useful to invert the muscle model and derive the necessary activation for desired muscle force, F_d, and muscle length and velocity trajectories, L_d and $\dot{L}_d$. Ignoring the filter (block "b" in figure 2-6):

$$\lambda = L_d + \mu \dot{L}_d - \frac{1}{c}\ln\left[\frac{F_d - (K_p \cdot (L_d - L_o))}{\rho(c_1 + c_2 \arctan(c_3 + c_4 \cdot \dot{L}_d))} + 1\right] \tag{2.36}$$

Now, substituting equation 2.36 into 2.34, one can see how the muscle activation signal is a combination of an error signal, e , its derivative, $\dot{e}$, and a feed forward signal:

$$A = e + \dot{e} + j_\lambda(L_d, \dot{L}_d, F_d) \tag{2.37}$$

where

$$e = L_m(t - \delta) - L_d \tag{2.38}$$

$$\dot{e} = \dot{L}_m(t - \delta) - \dot{L}_d \tag{2.39}$$

$$j_\lambda(L_d, \dot{L}_d, F_d) = \frac{1}{c}\ln\left[\frac{F_d - (K_p \cdot (L_d - L_o))}{\rho(c_1 + c_2 \arctan(c_3 + c_4 \cdot \dot{L}_d))} + 1\right] + \varrho. \tag{2.40}$$

The spinal reflex

Since the lambda model is based on the force-length characteristics of the intact human arm [13], [11], [12], and [14], there is no need to explicitly include spinal reflex in the model; it is automatically taken care of in that the stiffness of the muscle (tangent to the force-length characteristic) increases as the length of the muscle and the velocity of stretch increase, see equations 2.34 and 2.35 and figure 2-3B. On the one hand this reduces the complexity of the model in that there is one less component to model, but on the other hand, the driving signals of the model, λ, have a slightly more complex interpretation.

2.4 Single Joint Arm

The torque exerted on the elbow joint is a result of the forces exerted by the agonist and the antagonist muscles (the negative sign ensures that the torque is positive in the direction of increasing elbow angle, θ_e):

$$T = -M'F \tag{2.41}$$

where M is the moment arm vector of equation 2.3 and F is the vector

$$\begin{bmatrix} F_{agonist} & F_{antagonist} \end{bmatrix}'$$

and $F_{agonist}$ and $F_{antagonist}$ are forces generated by the muscles, as in equation 2.20, for either λ or α models. When this torque produced by the muscles is equated to the torque required by the passive skeleton, the complete description of the arm is obtained.

For the sake of analysis and comparison of the models, suppose that the expressions for F_{muscle}, where $muscle = agonist$ or $antagonist$, are linearized, as in equation 2.15, around a static equilibrium point, specifically, some muscle length, q_o, at which the muscle will be at rest for the CNS input u for which the net force produced by the muscles about the joint is zero (2.41 is set equal to zero); q_o was chosen to be the mean operating length of the muscle. The mean operating length is assumed correspond to approximately 0.7 of the muscle "resting" length; see figure 2-5. Since static equilibrium is assumed, the rate of change of muscle length is zero, $\dot{q} = 0$. Then, equation 2.15 becomes:

$$F_{muscle} = f_{muscle}(q_o, 0, u) + K_{muscle}(q_o, 0, u)(q - q_o) + B_{muscle}(q_o, 0, u)\dot{q} \tag{2.42}$$

or, to make the notation more concise, and since u is the only variable in the functions f, K, and B of the above equation:

$$F_{muscle} = f_{muscle}(u) + K_{muscle}(u)(q - q_o) + B_{muscle}(u)\dot{q} \tag{2.43}$$

which is valid for small and slow deviations of muscle length from its resting position. In vector notation,

$$\underline{\mathbf{F}} = \underline{\mathbf{f}}(\underline{\mathbf{u}}) + \underline{\mathbf{K}}(\underline{\mathbf{u}})\,(\underline{\mathbf{q}} - \underline{\mathbf{q}}_{\mathbf{o}}) + \underline{\mathbf{B}}(\underline{\mathbf{u}})\,\dot{\underline{\mathbf{q}}} \tag{2.44}$$

where the vectors and matrices are bold and underlined to emphasize the fact that they are matrices:

$$\begin{align}
\underline{\mathbf{F}} &= \begin{bmatrix} F_1 & F_2 & \cdots & F_i & \cdots & F_n \end{bmatrix}' \tag{2.45}\\
\underline{\mathbf{f}}(\underline{\mathbf{u}}) &= \begin{bmatrix} f_1(u_1) & f_2(u_2) & \cdots & f_i(u_i) & \cdots & f_n(u_n) \end{bmatrix}' \tag{2.46}\\
\underline{\mathbf{q}} &= \begin{bmatrix} q_1 & q_2 & \cdots & q_i & \cdots & q_n \end{bmatrix}' \tag{2.47}\\
\underline{\mathbf{q}}_{\mathbf{o}} &= \begin{bmatrix} q_{o_1} & q_{o_2} & \cdots & q_{o_i} & \cdots & q_{o_n} \end{bmatrix}' \tag{2.48}\\
\dot{\underline{\mathbf{q}}} &= \begin{bmatrix} \dot{q}_1 & \dot{q}_2 & \cdots & \dot{q}_i & \cdots & \dot{q}_n \end{bmatrix}' \tag{2.49}\\
\underline{\mathbf{K}}(\underline{\mathbf{u}}) &= diag\Big(K_1(u_1) \quad K_2(u_2) \quad \cdots \quad K_i(u_i) \quad \cdots \quad K_n(u_n) \Big) \tag{2.50}\\
\underline{\mathbf{B}}(\underline{\mathbf{u}}) &= diag\Big(B_1(u_1) \quad B_2(u_2) \quad \cdots \quad B_i(u_i) \quad \cdots \quad B_n(u_n) \Big), \tag{2.51}
\end{align}$$

n is the number of muscles acting on the joint, so for the single elbow joint, $n = 2$ and $i = agonist, \quad antagonist.$

Now, substituting 2.2 and 2.44 into 2.41,

$$T = -M'\underline{\mathbf{f}}(\underline{\mathbf{u}}) - M'\underline{\mathbf{K}}(\underline{\mathbf{u}})\cdot(\underline{\mathbf{q}}_{\mathbf{o}} - \underline{\mathbf{q}}_{\mathbf{ref}}) - M'\underline{\mathbf{K}}(\underline{\mathbf{u}})M\cdot(\Theta - \Theta_{ref}) - M'\underline{\mathbf{B}}(\underline{\mathbf{u}})M\cdot\dot{\Theta} \tag{2.52}$$

where for the single joint,

$$\begin{align}
\Theta &= \theta_e \notag\\
M'\underline{\mathbf{f}}(\underline{\mathbf{u}}) &= r_1 f_1(u_1) - r_2 f_2(u_2) \tag{2.53}\\
M'\underline{\mathbf{K}}(\underline{\mathbf{u}})M &= r_1^2 K_1(u_1) + r_2^2 K_2(u_2) \tag{2.54}\\
M'\underline{\mathbf{B}}(\underline{\mathbf{u}})M &= r_1^2 B_1(u_1) + r_2^2 B_2(u_2) \tag{2.55}
\end{align}$$

where the subscript 1 refers to the *agonist* and 2 refers to the *antagonist*.

2.4.1 Equilibrium point trajectories

An (static) "equilibrium point" of the system is the joint angle (or corresponding lengths of the muscles acting at the joint) at which the torque exerted on the joint is zero, $T = 0$, and the system is at rest, $\dot{\Theta} = 0$. That is, the torque produced by the agonists is equal and opposite to that produced by the antagonists, and any small disturbance to the system will cause it to go back to the equilibrium configuration. The definition used by different researchers for the equilibrium point is slightly different when there is an external load acting on the system; the lambda model proponents define the equilibrium point as that at which total joint torque, including torques applied by external loads, is equal to zero. The proponents of the alpha model, on the other hand, define the equilibrium point as that at which the total torque exerted at the joint is zero in the absence of any external torques. In the presence of external torque, the actual joint angle at which all forces are balanced is different from the equilibrium position that would be reached when there is no load. Thus, the no-load equilibrium point in this case was called a *virtual* equilibrium point [6]. When there is no external load applied to the system, which is the case studied here, the physical equilibrium point is the same as the so-called *virtual* equilibrium point.

According to the equilibrium point hypothesis, the CNS achieves the desired static postures by setting the desired equilibrium point; that is, the CNS activation appropriately selects the agonist and antagonist force-length curves such that the limb is positioned at the required joint angle for which:

$$T = -M'F = 0 \tag{2.56}$$

To produce slow movements, the equilibrium point is shifted such that at all times, 2.56 holds. The difference between the actual position of the limb and the equilibrium point specified by the CNS will generate a force in the spring-like muscles that will propel the limb towards the new equilibrium point. Thus, the equilibrium trajectory is a series of equilibrium points specifying equilibrium postures.

So, for the single joint arm, the desired agonist and antagonist forces should be such that:

$$T = r_1F_1 + r_2F_2 = 0 \tag{2.57}$$

where the the subscript 1 refers to the *agonist* and 2 refers to the *antagonist*. Equation 2.57 is solved by specifying a level of cocontraction, which is necessary for the production of the forces needed for faster movements, by the following equation:

$$F_c^2 = F_{agonist} \cdot F_{antagonist} \tag{2.58}$$

where F_c is the cocontraction force level. Now, as was mentioned in section 2.3.1, the desired force can be substituted into equation 2.28 or 2.36 to calculate the desired muscle activation level. Thus, the desired activation is determined without the explicit use of the skeletal dynamics as in [29], [18], and [3]. This is useful because the brain need not keep track of the limb inertias, which are required in the calculation of forces from desired trajectories and skeletal dynamics and are required to be known with accuracy or else the inaccurate values of inertia may lead to errors in the inverse dynamics that could result in unstable system behavior.

Alternatively, the equilibrium point condition, 2.56 can be used to relate the central inputs, $\underline{\mathbf{u}}$, to a centrally specified equilibrium point, such that the force required by the muscles need not be explicitly calculated [6]. For example, setting equation 2.52 to zero and solving for the static ($\dot{\Theta} = 0$) equilibrium joint position, Θ_{eq}:

$$T = \xi(\Theta_{eq}, \underline{\mathbf{u}}) = 0 \tag{2.59}$$

yields:

$$\Theta_{eq} = \Theta_{ref} - (M'K(\underline{\mathbf{u}})M)^{-1}(M'\underline{f}(\underline{\mathbf{u}}) + M'K(\underline{\mathbf{u}}) \cdot (\underline{\mathbf{q}}_{\mathbf{o}} - \underline{\mathbf{q}}_{\mathbf{ref}})) \tag{2.60}$$

where $M'K(\underline{\mathbf{u}})M$ is invertible when it is positive definite (note, $K(\underline{\mathbf{u}})$ is a diagonal matrix with positive terms), which is indeed the case, as experimentally verified in [40].

Thus, a map [2] is established between the inputs to the individual muscles and the equilibrium joint position (which is identical to the desired joint position in the absence of external loads). Now, the CNS may specify a specific equilibrium point, Θ_{eq}, by choosing the activations to the individual muscles, in vector $\underline{\mathbf{u}}$, such that equation 2.60 holds. Equivalently, by setting some equilibrium point, the CNS also chooses a certain subset in the space of allowable muscle activations, or vectors $\underline{\mathbf{u}}$, and those are the ones that satisfy the equality in 2.60. The joint position can be changed to a new equilibrium position by changing the vector $\underline{\mathbf{u}}$ such that equation 2.60 holds for the new equilibrium position. For a certain equilibrium point, one can find a unique solution for $\underline{\mathbf{u}}$ in equation 2.60 by adding additional constraints, like imposing a constant tonic activation level that relates the activations of muscles around a certain joint, for example:

$$N_t^2 = N_{agonist} \cdot N_{antagonist} \tag{2.61}$$

It is interesting to note that if $\underline{\mathbf{u}}$ is such that:

$$M'\underline{f}(\underline{\mathbf{u}}) + M'K(\underline{\mathbf{u}}) \cdot (\underline{\mathbf{q}}_{\mathbf{o}} - \underline{\mathbf{q}}_{\mathbf{ref}}) = M'\underline{\mathbf{K}}(\underline{\mathbf{u}})M \cdot (\Theta_d - \Theta_{ref}) + M'\underline{\mathbf{B}}(\underline{\mathbf{u}})M \cdot \dot{\Theta}_d \tag{2.62}$$

then, substituting back in 2.52:

$$T = \underline{\mathbf{R}}(\underline{\mathbf{u}})(\Theta_d - \Theta) + \underline{\mathbf{D}}(\underline{\mathbf{u}})(\dot{\Theta}_d - \dot{\Theta}) \tag{2.63}$$

where Θ_d and $\dot{\Theta}_d$ are the vectors of desired joint angles and angular velocities corresponding to the desired end-effector trajectory and $R = M'\underline{\mathbf{K}}(\underline{\mathbf{u}})M$ and $D = M'\underline{\mathbf{B}}(\underline{\mathbf{u}})M$ are the joint stiffness and joint viscosity matrices.

Now, when the joint torque produced by the muscles, 2.63, is equated to the joint torque required by the passive skeleton, like in equation 2.11 for the double joint case, a second order equation is obtained to describe the system; this will be

[2] This map is linear with respect to N of the alpha model but exponential with respect to λ of the lambda model.

discussed further in section 2.5.

2.4.2 Comparing the α and λ single joint arm models

Computer models were constructed for the single joint arm using both the alpha and lambda muscle models. The expressions for muscle force derived in section 2.3 for the α and λ models are repeated here for convenience (the low pass filter, block "b" of figure 2-6, is ignored in the equations but included in the computer simulations):

$$F_{muscle} = K_p(L_m - L_o) + F_a(c_1 + c_2 \arctan(c_3 + c_4 \cdot \dot{L_m})) \tag{2.64}$$

where

$$F_a = \begin{cases} NF_{max}e^{-\left(\frac{\frac{L_m}{L_o}-1.2}{0.4}\right)^2} & \text{for } \alpha \text{ model} \\ \rho(e^{c(L_m+\mu\dot{L_m}-\lambda)}-1) & \text{for } \lambda \text{ model} \end{cases}$$

Inverse dynamics was used to determine the required CNS activation levels. There were two sets of models, both required the use of appropriate (inverse) muscle models to determine the required muscle inputs, that is, either equation 2.28 or 2.36 for the first set or equation 2.64 for the second set. In the first set of models, the desired joint torque was determined using the skeletal dynamics and desired joint angles. Then, the desired muscle force was determined using the equilibrium point condition of equation 2.57 and the cocontraction criterion of 2.58. Now, the desired muscle force, along with desired joint position trajectory, were substituted in the inverse muscle model expressions of equation 2.28 or 2.36 to determine the required α or λ inputs. As discussed earlier, there are reasons to believe that this open loop method for determining the required activation of the muscle is probably not the one used by the CNS; for example, it requires accurate knowledge of the parameters of the skeletal and muscle model such as limb inertias, and uncertain or incorrect parameters, when used to calculate the CNS input, result in inputs that may destabilize the system. Still, these calculations were useful to compare the models and the activation or driving signals required by each.

In the second set of models, the skeletal dynamics were not used to determine the requisite joint torque; in fact, the desired joint torque was not calculated at all. Instead, for the alpha model, the expressions for the agonist and antagonist muscle forces, from equation 2.64, were substituted in the equilibrium point condition of equation 2.57 to get an equation of two unknowns, $N_{antagonist}$ and $N_{agoinist}$. A unique solution was determined by solving this simultaneously with equation 2.61. For the lambda model, the value of lambda was set to the required muscle length trajectory plus a corresponding cocontraction signal, as was done in [48]. This is merely an approximation of the exact value required by equation 2.36, where the cocontraction signal approximates the term $-\frac{1}{c}\left[\frac{F_d-(K_p)\cdot(L_d-L_o)}{\rho(c_1+c_2\arctan(c_3+c_4\cdot L_d))}\right]$, which is a constant when the cocontractive forces are assumed constant and much greater than the forces involved in the actual movement and the passive force, such that the change in $F_d-(K_p)\cdot(L_d-L_o)$ with time is basically negligible. Again, these calculations are open loop and work in the absence of external disturbances; the activations obviously need to be modified by the CNS in the presence of disturbances according to the feedback that the brain receives from the arm. This will be explored later.

For both sets of models, both alpha and lambda models tracked the desired joint angle and exhibited the bell-shaped velocity profile, even in the presence of the reflex delays and calcium activation dynamics (block "b"), and even though these were not accounted for in the inverse calculations for the desired activation level. However, the reflex delays did result in a small delay between the desired joint trajectory and the actual joint trajectory. Furthermore, the addition of an external torque step did not destabilize the system, but, as expected, did alter the final position reached by the arm as feedback to the brain, that may result in adjustments of the muscle activations such that the external torque is countered and the original final position is reached, has still not been added to the model; this outcome of the models is exactly like that observed by Bizzi in deafferented monkeys [5] and [4]. Thus, both models exhibit the observed kinematics and spring-like properties of the single joint arm. Moreover, both models involved some work to determine the required activation signals. The equilibrium point hypothesis mapping of equilibrium joint angle to the muscle inputs

	α	λ
$f(u)$	$N \cdot F_{max} C e^{-\frac{1}{4}}$	$\rho C(e^{c(L_o-\lambda)} - 1)$
$K = \frac{\partial f}{\partial L_m}$	$K_p + N \cdot F_{max} C \left(\frac{10}{4L_o} e^{-\frac{1}{4}} \right)$	$K_p + e^{c(L_o-\lambda)} c \rho C$
$B = \frac{\partial f}{\partial \dot{L}_m}$	$N \cdot F_{max} e^{-\frac{1}{4}} \frac{c_2 \cdot c_4}{1+c_3^2}$	$\rho \left(e^{c(L_o-\lambda)} \left(\frac{c_2 \cdot c_4}{1+c_3^2} + \mu C \right) - \frac{c_2 \cdot c_4}{1+c_3^2} \right)$

Table 2.1: When the α and λ models are linearized around some muscle resting length, L_o, and $\dot{L}_m = 0$ as in equation 2.43, the tabulated stiffness, K, and viscosity, B terms, in addition to a function, $f(u)$, of the muscle input, $u = N$ for the alpha model and $u = \lambda$ for the alpha model, are obtained. Here, $C = c_1 + c_2 \arctan c_3$.

which enables the CNS to simply set the desired joint trajectory was not explored for the single joint case, but will be used for the double joint arm in section 2.5.

To further highlight the differences and investigate similarities of the two models, F_{muscle} is linearized, as in equation 2.43, and the terms of the linear model are tabulated in table 2.1. These were then used in equations 2.54 and 2.55 to determine the joint stiffness and viscosities for the activation levels calculated when the models described above were simulated for small perturbations around the resting length, L_o. The parameter values used in the models are shown in tables 2.5, 2.2, 2.3 and 2.4.

Table 2.1 shows that the muscle stiffness and viscosity values vary with the central inputs (N and λ), either proportionally (with N) or exponentially (with λ). Thus, it follows that the joint stiffness and viscosities, of equations 2.54 and 2.55, also vary with the central inputs. In fact, as the muscle input increases, the joint stiffness predicted by the alpha model also increases; therefore, it is easy to see from here that, as mentioned in section 2.3.1, the additional excitation that is provided by the spinal reflex serves to increase the joint stiffness.

It was found that for the same small perturbations form the resting position, the lambda model muscle stiffness was larger than that calculated using the alpha model. The *joint* stiffness and viscosity values, however, were within the experimentally derived ranges in [40] and [17]. Furthermore, to produce the same small movements around the resting position, the λ inputs were around ten times smaller than the α inputs; this is not surprising as the λ inputs are related exponentially, not propor-

δ (s)	τ (s)	c_1	c_2	c_3	c_4
0.03	0.015	0.82	0.5	0.43	58

Table 2.2: The values of the parameters used in the alpha model are tabulated here and in table 2.5. These are values also used in the lambda model, from [29] and [18].

c	δ (s)	τ (s)	μ	c_1	c_2	c_3	c_4
112	0.03	0.015	0.15	0.82	0.5	0.43	58

Table 2.3: The values of the parameters used in the lambda model, from [29] and [18] are tabulated here and in table 2.5

tionally, to the desired muscle force. But, despite these differences, it was observed that, for both models, when the inputs, λ and N that were calculated using inverse dynamics, were used in equations 2.53 to 2.55, the equality of equation 2.62 was satisfied, and therefore equation 2.63 holds. This is useful in the modeling of the double joint arm, as is discussed in section 2.5.

2.5 Double Joint Arm

Assembling a model for a double joint arm is a more complex task than that of modeling the single joint arm; there are six muscle groups arranged in agonist-antagonist pairs around the two joints as shown in figure 2-1. Thus, a new problem of redundancy arises due to the "extra" muscle pair, and determining the activations to the muscles involves either optimization or making certain assumptions on how the desired forces at a joint are distributed among the muscle pairs that act at that joint. For example, when the skeletal dynamics equations are used to determine required joint torques that are substituted into equation 2.41 or when the equilibrium point condition of equation 2.56 is used, two equations with six unknowns, the muscle forces, are obtained. One way to solve for the muscle forces is to assume a cocontraction level for each of the muscle pairs and distribute the required torque at a joint equally (or according to some ratio) between the joint muscle pair and the biarticular pair, as was done in [3]. Alternatively, the muscle force vector, F, can be found by minimizing the norm of F such that net torque is zero, that is, equation 2.56 is satisfied, and then adding cocontraction forces to the muscle pairs proportionally such that the net torque around each joint remains zero, as in [29] and [18]. But, as discussed in section 2.4.1, there is no strong reason to believe that the CNS works this way; still, use of either of the two above methods to determine muscle activation will work, and the models will exhibit the required kinematic properties of multi-joint arm movement.

The mapping between the desired (or equilibrium) joint trajectories and the required muscle activations that is derived from the equilibrium point condition, like that of equation 2.62 in section 2.4.1, does not simplify the task of finding the individual muscle activations either. Again because of redundancy, the muscle activations are not unique, and some optimization criterion is needed to determine the muscle activations from the desired joint trajectory. But, since the mapping implies that the specification of equilibrium joint position is equivalent to specifying the muscle activation, if one assumes that the mapping is taken care of automatically one can simplify the muscle model by assuming that the CNS inputs are the desired trajectories, as in equation 2.63, repeated here for convenience:

$$T = \underline{\mathbf{R}}(\underline{\mathbf{u}})(\Theta_d - \Theta) + \underline{\mathbf{D}}(\underline{\mathbf{u}})(\dot{\Theta}_d - \dot{\Theta})$$

where Θ_d and $\dot{\Theta}_d$ are the vectors of desired joint angles and angular velocities corresponding to the desired end-effector trajectory and $R(\underline{\mathbf{u}}) = M'\underline{\mathbf{K}}(\underline{\mathbf{u}})M$ and $D(\underline{\mathbf{u}}) = M'\underline{\mathbf{B}}(\underline{\mathbf{u}})M$ are the 2x2 joint stiffness and joint viscosity matrices.

One may bypass the need to assign individual muscle activations by modeling the limb as a whole instead of modeling the individual muscles and putting them together to model the double joint arm. It has been experimentally shown that like the individual muscles, the multi-joint arm as a whole exhibits the elastic spring-like properties and the stiffness matrix of the arm has been derived for different postural conditions [40], and when the limb is in motion [24]. This approach of viewing the multi-joint arm as a damped spring leads to the same equation, 2.63, that was derived

		Shoulder Link	Elbow Link
		$i = s$	$i = e$
m_i	Kg	1	1
l_i	m	0.3	0.3
r_i	m	0.15	0.15
I_i	$Kg \cdot m^2$	0.075	0.075

Table 2.4: The values of the parameters related to the skeletal dynamics, defined in section 2.2 and shown in figure 2-2, are tabulated here. Here, the subscript i is s for the shoulder link and e for the elbow link.

by assuming the relationship of equation 2.62 between the muscle inputs and the desired trajectories. A version of this equation (that does not include $\dot{\Theta}_d$ and uses constant stiffness and viscosity matrices) was used by Flash along with straight end-effector virtual equilibrium trajectories to successfully model the kinematics of the multi-joint reaching movement [17]. But the model was successful for slow movements. By adding the dependence on desired angular velocity, $\dot{\Theta}_d$, as in equation 2.63, a model that works for faster movements is obtained [38].

Equating the torque produced by the muscle, equation 2.63, to the torque required by the passive skeleton, equation 2.11 leads to a second order equation of the multi-joint arm:

$$H(\Theta)\ddot{\Theta} = -C(\Theta, \dot{\Theta})\dot{\Theta} + R(\underline{u})(\Theta_d - \Theta) + D(\underline{u})(\dot{\Theta}_d - \dot{\Theta}) \tag{2.65}$$

$$\ddot{\Theta} = H^{-1}(\Theta)\left[-C(\Theta, \dot{\Theta})\dot{\Theta} + R(\underline{u})(\Theta_d - \Theta) + D(\underline{u})(\dot{\Theta}_d - \dot{\Theta})\right] \tag{2.66}$$

where $H^{-1}(\Theta)$ is an invertible matrix because the moments of inertia of the links are always positive (nonzero).

In this equation, the inputs are the desired joint position and joint angular velocity trajectories as well as the muscle inputs, u, that vary the joint stiffness and viscosity matrices. In [17], these matrices were assumed to be constant through out the movement. Gomi and Kawato's measurements for the stiffness of the moving limb, however, show that the stiffness matrix does change during the course of the movement.

	Muscle	Muscle Function	Passive Stiffness (N/m)	Maximum Force (N)	Resting Length (m)	Moment arm (m)	ρ (in eq. 2.35)
1	Pectoralis	Shoulder flexor	258.5	700	0.1153	0.03	6.8
2	Deltoid	Shoulder extensor	258.5	263	0.2195	0.04	11
3	Biceps long head	Elbow flexor	190.9	740	0.1055	0.04	3.6
4	Triceps lateral head	Elbow extensor	209.9	1000	0.2855	0.02	6
5	Biceps short head	Biarticular flexor	36.5	400	0.3002	0.05; 0.05*	2.1
6	Triceps long head	Biarticular extensor	116.3	587	0.3055	0.04; 0.02*	6.7

Table 2.5: The numbers in the first column correspond to the same numbers in figure 2-1. The values for the passive stiffness were taken from [29]; the moment arms and ρ were taken from [18]; the values for maximum or peak force were taken from [36]; and the muscle resting lengths were taken from [42]. * For the biarticular muscles that span two joints, the first number is the moment arm at the shoulder and the second is the moment arm at the elbow joint.

2.6 Arm Model Activation Signals

The purpose of looking at different models of the arm, the nonlinear models that were based on either the alpha or the lambda muscle models and the simpler second order models that are based on linearizing the muscle models and experimental evidence of the limb's damped-spring-like behavior, was to gain a better understanding of the feed-forward signals that need to be issued by the brain to drive the models. Though the driving signals of the different models, N, λ, or $\left[\theta_d, \dot{\theta}_d\right]$, are quantitatively different, they are similar in that they appear to be linear combinations of the desired joint trajectory, which is a smooth sigmoid, and its derivative, the bell-shaped velocity profile:

$$u = w_1\theta_d + w_2\dot{\theta}_d \tag{2.67}$$

where w_1 and w_2 are constants. This was not only observed from the simulations, but is also evident in equation 2.62.

Assuming that there is some map from the desired joint angle and joint velocity trajectories to the individual muscles, and it is possibly implemented in the spinal cord through the wiring of corticospinal neurons that synapse onto different motoneurons and from the motoneurons to different muscle units, then the driving signals that need to be issued by the CNS are simply $\left[\theta_d, \dot{\theta}_d\right]$. How these might be generated in the brain and how they might be modified by proprioceptive feedback are the subjects of the next chapter.

2.7 Summary

In this chapter two versions of equilibrium point arm models, α and λ, have been explored and compared. The two models essentially emphasize different behaviors of the muscle force-length characteristic in response to activation: a change in slope according to the α model and a shift of the characteristic according to the λ model. Using either muscle model to build a model of a single or double joint arm, however, results in arm models which have similar behaviors, especially when reflex loops, which are built into the lambda formulation but not the alpha formulation, are added to the alpha model. Even though the physical interpretation of the inputs to either model, $u = N$ or λ, is different, both models are based on the notion that the muscle properties propel the arm from its current equilibrium state to a centrally specified equilibrium state, such as desired arm position.

It has been shown that there is a mathematical mapping between the desired arm position and the muscle inputs; and reformulating either the α or λ arm model in terms of the desired arm position simplifies it from one with six inputs in the double joint case, one for each muscle around the shoulder and elbow joints, to a simpler model with desired joint position and velocity as inputs. This removes some of the complexity that is introduced to the model by having the six muscles, such as the need to deal with the redundancy of six muscles by activating the muscles according to some predetermined, possibly somewhat artificial, scheme such as one whereby the desired force changes are distributed "evenly" among muscles. Furthermore, it eliminates the need for calculating precise muscle inputs using inverse muscle models, which is not necessarily the way the CNS operates. Though the reduced complexity comes at the price of losing some biological accuracy, the overall performance and behavior of the arm model, which are for the purposes of this work more important than the detailed muscle mechanism underlying arm movement, are not forfeited.

So, given that there is a mapping that essentially exploits muscle properties such that, potentially, simpler central commands need to be issued to the "plant", the question is whether this map actually exists in reality, and if so, where it is implemented. Since it has been observed that the inputs to the muscles, $u = N$ or λ, are basically weighted combinations of joint position and velocity signals, it has been suggested here that the "map" in question, which basically involves scaling and routing joint position and velocity signals to the appropriate muscles, takes place in the spinal cord. This seems plausible for several reasons. First, the pyramidal neurons project both directly onto motoneurons and indirectly via interneuron pools; this provides both scaling and routing capabilities. Projection strengths are determined by synaptic connections, which can have different strengths, so signals can be weighted. Furthermore, anatomy reveals both convergence of cortical signals from different pyramidal cells onto the same motoneurons and divergence of cortical signals from the same pyramidal cell onto different motoneurons; that is, the spinal cord does route signals. Secondly, there is some experimental evidence of the existence of the map in the spinal cord [6].

Assuming that the map is taken care of in the spinal cord means that, effectively, the spinal cord has been lumped with the six-muscle arm model to obtain a simpler model of the plant with inputs θ_{ref} and $\dot{\theta}_{ref}$. For the purposes of this work, namely to propose and test a mechanism that is based on simple biologically reasonable assumptions by which sensorimotor cortex commands the "arm" as will be discussed in the next chapter, the second order "arm"[3] is adequate. However, since the dynamics of the arm model play an important role in the outcome of the simulations of the proposed command generator, using six muscle models to test the command generator is an important part of future work. This entails a more detailed study of the different maps for different muscle models, and whether the spinal circuitry that implements

[3] From here on, "arm" is to mean the spinal cord and six muscle arm in the context of horizontal point-to-point reaching.

them exists. This is to be addressed in future work, as is mentioned in chapter 4.

Chapter 3

Command Generation in the Sensorimotor Cortex

The previous chapter was concluded with the statement that the equilibrium point arm models, whether alpha, lambda, or some second order approximation of these, can be driven by appropriate combinations of signals that encode desired joint position and joint velocity. If these signals are generated somewhere in the brain, then it is possible that they are properly scaled and sent to the individual muscles via the spinal cord circuitry. Here, the existence of this circuitry is going to be assumed and is not the subject of investigation; rather, the question investigated is the required structure of the command (joint velocity and position) generating mechanism in the sensorimotor cortex given the known anatomy and physical structure of the cortex. A major concern is how and where each functional block of the command generator is actually implemented in the sensorimotor cortex. The operation of the proposed command generating model is then tested with a simple second order model of the arm for features that conform (or possibly don't) with the observed kinematics of human arm movements, as reported in [17], [31], [50], and [2]; and features that conform (or possibly don't) with the data from neurophysiological studies such as [22], [23], [32], [37], [39], [44], and [46]. First, however, the model of the basic building block of the generator, the neuron, and how it fits in a larger network of neurons will be discussed.

3.1 Neuronal Network Model

Before the model of the neuron is described, it should be noted that central to the whole work is the observation that the envelopes of firing signals (which corresponds to the profile of neuron firing rate over time) of individual cells in sensorimotor areas (motor, premotor, and sensory cortex) and association areas (parietal cortex) as shown in neurophysiological studies ([22], [23], [37], [39], and [44]) have sigmoidal or bell shaped profiles (or a combination of both) that are reminiscent of the profiles of the hand kinematics, bell-shaped tangential velocity, and consequently the sigmoidal joint position and bell-shaped joint velocity profiles. The sigmoidal and bell-shaped profiles as well as a combination of both are shown in figure 3-1. These are smoothed versions of the envelopes of the actual firing signals of the neurons, which, as can be seen in the aforementioned papers, appear more "bumpy". Still, to a first approximation, the envelopes resemble the kinematic profiles; thus, the main assumption is that the envelopes of the firing signals contain position and velocity information, and it is these envelopes or firing rates that the model of the neuron, described below, receives as input and will consequently output.

The model of the neuron used in this study is a simple abstraction that does not capture any neuronal dynamics; refer to figure 3-2. The output firing rate of some neuron, j, is function of a weighted sum of the firing rates of input neurons and the threshold of that neuron, j.

$$y_j = f(\sum_{\mathrm{i}} w_{ij} y_i + \sum_{\mathrm{k}} v_{kj} x_k - \vartheta_j) \tag{3.1}$$

where y_j is the output firing rate of the j^{th} neuron in the cortical area under consideration, w_{ij} and v_{kj} are synaptic weights that are described in sections 3.1.1 and 3.1.2 and x_k is the firing rate of the k^{th} remote neuron (located in some different cortical area). Finally, ϑ is the firing threshold of the j^{th} neuron and f is a nonlinear function that describes the input-output behavior of the neuron, the simplest of which is a saturation function.

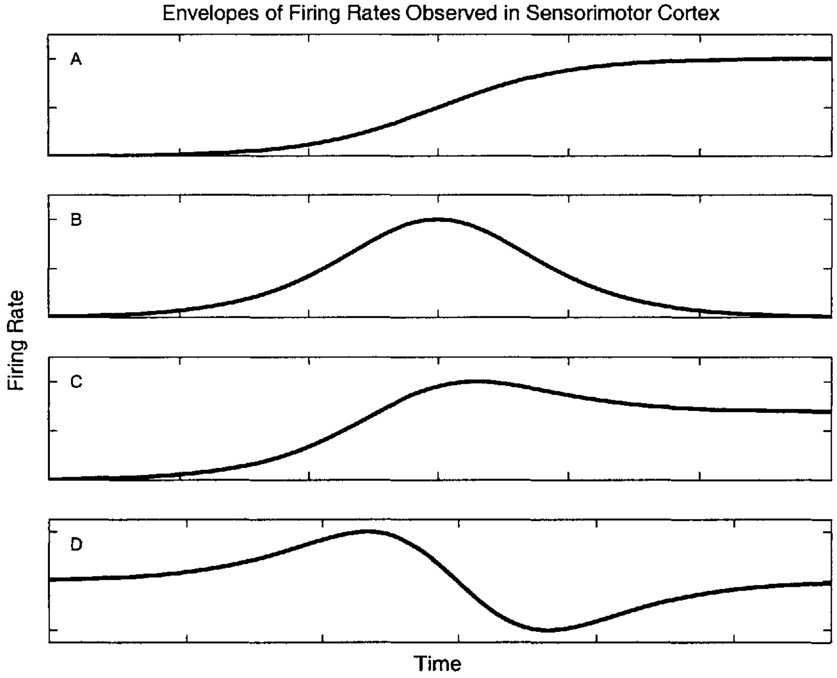


Figure 3-1: Neurons in the motor, premotor, sensory, and parietal cortices have firing rates that vary with time as reported in neurophysiological studies and shown above. The (smoothed) profiles are either sigmoidal (A), bell-shaped (B), some combination of the previous two (C), or bi-phasic (D).

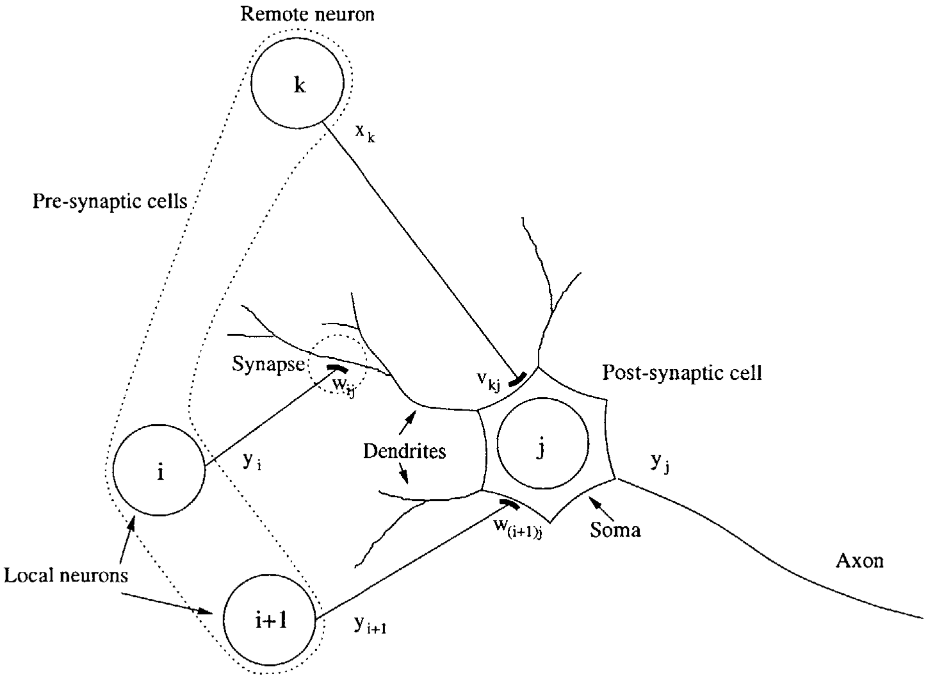


Figure 3-2: The abstract model of the neuron is described in detail in the text.

The output firing, y, is assumed to be instantaneous; there is no time delay between the moment the neuron receives its input and the moment it fires. The neuron's threshold for firing, ϑ, was ignored for simplicity, but this does not change the model. The nonlinearity in the input-output neuron model, f, the simplest of which is a saturation function, was not explicitly coded in the simulations, rather, f was simply a gain constant; but, the firing rates were monitored closely to make sure that no neuron fires excessively, that is, the neurons never reach saturation. This simplicity was intentionally seeked as a start and in order to see what one could or could not achieve by using the simplest possible models; thus, the complexity that the more detailed but more realistic models would add was forsaken in order to find out to what extent this detail is really necessary and a determining factor in the simulation results. As will be discussed in chapter 4, an interesting improvement to be implemented in future work is to incorporate the neuron dynamics, such as delays at synapses and attenuation of signals traversing the dendritic tree, within more detailed models of neurons and see if and how this changes the results.

3.1.1 Remote Inputs of Neurons

As was discussed in detail in section 1.1.1, the motor cortex receives afferent and re-efferent information via projections from the sensory cortex, the parietal cortex, the premotor areas, and the thalamus. The projections to and from the motor cortex are summarized in figure 1-1. Such projections from areas outside the area under consideration are here called "remote inputs" to distinguish them from the inputs a neuron may receive from neurons within the same area, which will be discussed in section 3.1.2 below.

Any modeled neuron can receive any number of remote inputs, each of which may influence the cell to a different extent. In the neuronal model described above, the remote inputs are designated by the letter x. The strength of the influence of the k^{th} remote neuron on the j^{th} neuron, the neuron under consideration, is determined by the weight v_{kj}. Thus, a weight, v_{kj}, of zero, for example, means that the j^{th} neuron does not receive input from the k^{th} remote neuron. A positive (negative) weight, $v_{kj} > 0$ $(v_{kj} < 0)$, means that the j^{th} neuron receives excitatory (inhibitory) input from the k^{th} remote neuron. Excitatory and inhibitory connections will be discussed further in section 3.1.3.

In the network model that is implemented in this work, the values for the weights, v_{kj}, from a network of N remote neurons onto some j^{th} neuron were chosen such that the values of $(v_{1j}, v_{2_j}, \ldots, v_{Nj})$ are inversely proportional [1] to the "distance" between remote neurons; that is, the strongest projections to neuron j are from remote cells whose distance from each other is smallest. First, "distance" does not necessarily mean physical distance between cells; rather, distance refers to the difference between the values of the variable encoded by the cells. For example, if the remote neurons encode some desired movement direction, then the "distance" between a pair of neurons is the difference between the directions encoded by these neurons. Second, the weight assignment was carried out by first making a connection from an arbitrary remote neuron to the the target neuron, j, and assigning it the highest weight, and

[1] Both here and in subsequent uses, an "inversely proportional" relationship between a and b is such that when a increases, b decreases in some sense, not necessarily reciprocally.

then relating the weights of the rest of the connections from the remaining remote neurons to that arbitrary neuron such that connections from neurons which are at a smaller distance from the first arbitrarily chosen neuron had stronger weights. More precisely, an equivalent way to achieve this was that the target neuron, j, was assigned a random number, ξ_j, that lies with in the needed range of the variable assumed to be encoded by the remote neurons, as will be discussed below. The weight from the k^{th} remote neuron was then calculated by:

$$v_{kj} = k_v \, e^{-\frac{(\xi_j - \phi_k)^2}{2\sigma}} \tag{3.2}$$

where k_v and σ are arbitrary constants, and ϕ_k is the value of the variable encoded by the k^{th} remote neuron.

For example, the remote neurons could represent an ensemble code for arm movement; that is, the neurons map the workspace of the hand such that each neuron or group of neurons represent a small region in the workspace to which the hand is required to move. Then depending on the position to which the arm is required to move, a certain group of remote neurons will fire; if one thinks of the workspace in polar coordinates, then the fact that a certain group of neurons fire could indicate the required direction of movement, and the distance of movement (and speed) could be encoded by both the magnitude of the firing rate and the the duration of the change in the firing rate. Thus, under this formulation, the variable ξ that the remote neurons encode is the direction of the required arm position relative to some fixed point in the workspace.

A network was built and simulated with 4 remote neurons, representing 4 directions in the workspace: $\frac{\pi}{2}$, π, $\frac{3\pi}{2}$, and 2π. The regions of the workspace with directions in between the above directions are easily specified by having several remote neurons representing different directions fire with the appropriate magnitudes. This is analogous to vector addition; in fact, each of the neurons could be considered to represent a vector in the workspace and as long as two neurons form a basis, then any point in the workspace could be theoretically represented. Having the four neurons instead

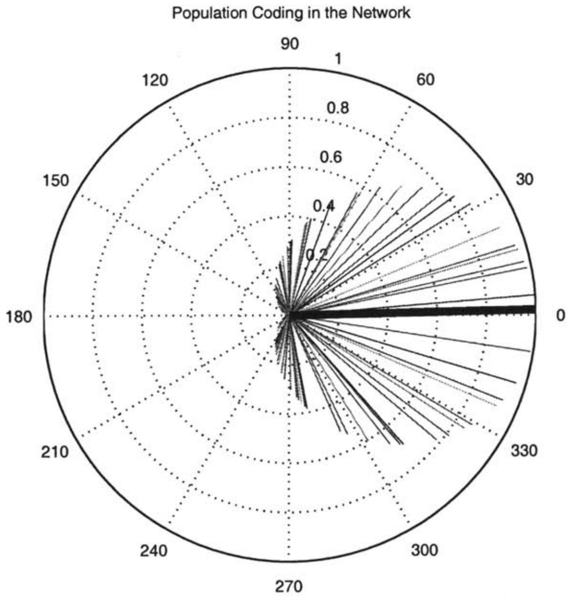


Figure 3-3: The simulated firing rate of 100 cortical neurons in response to input from a remote neuron that encodes required movement to the right is plotted versus the variable ξ assigned to each of the 100 neurons. The magnitude of each of the thin lines represents the firing rate and the direction represents the value of the corresponding variable ξ. The thick line is the population vector, which is obtained by vectorally adding the thin lines, and is in the same direction as the required movement direction signaled by the remote neuron.

of only two means that all positions in the workspace could be realized using only positive scaling of the vectors (that is, positive firing rates, though it will be seen in section 3.1.3 that negative firing rates are allowed in certain circumstances and have a special interpretation). It can be seen in figure 3-3 that the network simulated with 100 local neurons was observed to exhibit the population coding that is reported in the cortical areas in studies like [22], [23], [32], [37], [39], and [44], even though directional tuning is not "hard wired" into the neurons. That is, population coding (discussed in section 1.1.2) is a result of the way the inputs are wired to the neurons in the network, and not a result of the firing function of the neurons (which is in fact linear in these simulations; that is, f in equation 3.1 is simply a gain constant).

3.1.2 Local Interactions of Neurons

Neurons within a cortical area, such as the motor cortex or sensory cortex, receive as inputs the outputs of neurons from other areas, as discussed above, but also receive inputs from cells of the same area as well, often via inhibitory and excitatory interneurons [43]. These projections from neighboring neurons of the same area will be referred to as local projections.

The local interactions of neurons have been studied by several investigators as in [23] and [37]. In [23], Georgopoulos et al. report that the firing of pairs of neurons in the motor cortex is negatively correlated with the difference in their preferred directions; that is, if the difference in the preferred directions of a pair of neurons is small (close to 0°), the two neurons are likely to be interconnected via strong excitatory synapses, while neurons with nearly opposite preferred directions (that is, the difference between their preferred directions is close to 180°) are likely to be interconnected via strong inhibitory synapses. In fact, through their analysis, Georgopoulos et al. conclude that "mean synaptic strength was negatively correlated with the angle (0° and 180°) between the preferred directions of the two neurons ... throughout the range of connections from positive (excitation) to negative (inhibition)." In [37], on the other hand, while Maynard et al. also report on the interactions between the neurons in the motor cortex, they do not find a correlation between these interactions and the preferred direction of the cell. However they do report that neurons whose firing is correlated when the arm moves in one direction will exhibit a negative correlation in their firing for the opposite direction; through their analysis they conclude that the correlation between neuron firing actually carries extra information that improves the cortical population coding of movement direction.

While there is no consensus on the the correlation of the local interactions of neurons with the preferred direction of neurons, there seems to be agreement that neurons whose firing is strongly correlated in one direction of the hand movement in the workspace will have negatively correlated firing in the opposite direction. In this work, this result was the basis of setting up the local interactions of neurons such that the neurons receiving inputs from the same remote neurons had strongest interactions; the larger the "distance" between the strongest remote inputs that a pair of neurons receive, the weaker their interactions. Practically, this was implemented conveniently by choosing the weights w_{ij} that determine the strength of the connection between the output of the i^{th} and the j^{th} target neuron using the variable ξ that was discussed in the previous section. The weights, w_{ij}, were calculated using:

$$w_{ij} = k_E \, e^{-\frac{(\xi_j - \xi_i)^2}{2\sigma_E}} - k_I \, e^{-\frac{(\xi_j - \xi_i)^2}{2\sigma_I}} \tag{3.3}$$

where k_E, k_I, σ_E, and σ_I are constants chosen such that $k_E > k_I$ and $\sigma_I > \sigma_E$; the subscripts E and I stand for excitatory and inhibitory connections. The weight w_{ij} is the sum of all the excitatory connections that the j^{th} neuron receives from the i^{th} neuron, which are subsumed in the first exponential term on the right hand side of equation 3.3, plus all the inhibitory connections, subsumed in the second exponential term on the right hand side of equation 3.3. Obviously, this is a simplification that eliminates any dynamics or timing differences in the arrival, at the target neuron soma, of several signals from the same neighboring neuron due to more than one synapse of the neighboring neuron onto the target neuron. These timing differences could be a result of the different locations of the synapses of the different inputs from the i^{th} neighboring neuron on the dendritic tree of the target neuron. Here, however, it is assumed that any interaction between two neurons is a result of at most two synapses: one from the j^{th} to the i^{th}, and the second from the i^{th} to the j^{th}. Finally, w_{jj}, that weights the signal from the output of the j^{th} neuron back onto itself as an input, which represents recurrent excitation via a collateral of the axon (and is physically observed to happen in the motor cortex [43]) was allowed but for simplicity ignored; it resulted in no changes in the results when the proper weighting of projections onto the output neurons was chosen.

A network of four remote neurons and 100 local neurons that were fully interconnected as described above was simulated; a sample result is shown in figure 3-4. It can be seen in this figure that the local interactions have served to shape the firing

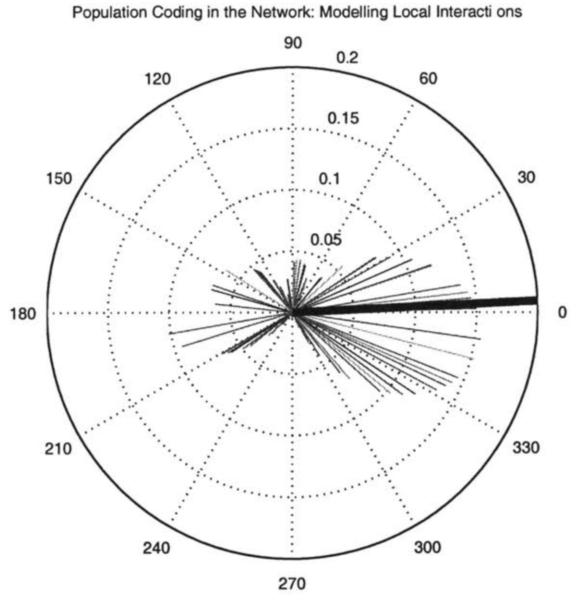


Figure 3-4: The simulated firing rate of 100 interconnected cortical neurons in response to input from a remote neuron that encodes required movement to the right is plotted versus the variable ξ assigned to each of the 100 neurons. The magnitude of each of the thin lines represents the firing rate and the direction represents the value of the variable ξ. The thick line is the population vector, which is obtained by vectorally adding the thin lines, and is in the same direction as the required movement direction signaled by the remote neuron. Local interactions between the neurons that were modeled in this network appear to improve the directivity of the population code.

distribution of the cells such that the "main lobe" which peaks around the desired movement direction is narrower than was the case when there were no local interactions (compare with figure 3-3).

3.1.3 Inhibition, Excitation, and Interneurons

The weights v_{kj} and w_{ij} could theoretically take on either positive or negative values. The sign of the weight is an abstraction that models the type of the synapse. If the sign of the weight is positive, then the synapse is excitatory; signals arriving from the pre-synaptic cell will result, if some threshold is exceeded due to the input, in the increased firing (higher firing rate) of the post-synaptic cell. If the sign of the weight is negative, then the synapse is inhibitory; signals arriving from the pre-synaptic cell will result, if some threshold is reached, in the decreased firing of the post-synaptic cell. In practice, the signs that the weights were allowed to take on were values that reasonably model the actual connections in the brain. For example, v_{kj} had positive values, to model excitatory connections between cortical regions. The local interactions, as modeled by w_{ij}, took on both positive and negative values as was described in the previous section.

Physically, the output firing rate of some neuron might increase from some initial level in response to its inputs, as shown in figure 3-1, or decrease in response to its inputs, as shown in figure 3-5 and can be seen in reports of electrophysiological studies on the motor, sensory and parietal cortices, such as [22], [23], [37], [39], and [44]. In this work, it was found simpler to ignore initial firing rate; and indeed, it is not modeled in the equation 3.1. This is another simplification that yielded abstraction in the sense that inhibited outputs are manifested by negative signals. In reality, there is no such thing as a negative firing rate; but as an abstraction, it is to be thought of as a drop from some initial firing rate. In future work, it might be interesting to incorporate the additional information provided by initial firing rates in the model of the neuron, and study the value of this information provided, how it affects the dynamics of the command generating model suggested below, and what brain structures determine, monitor and regulate such bias levels.

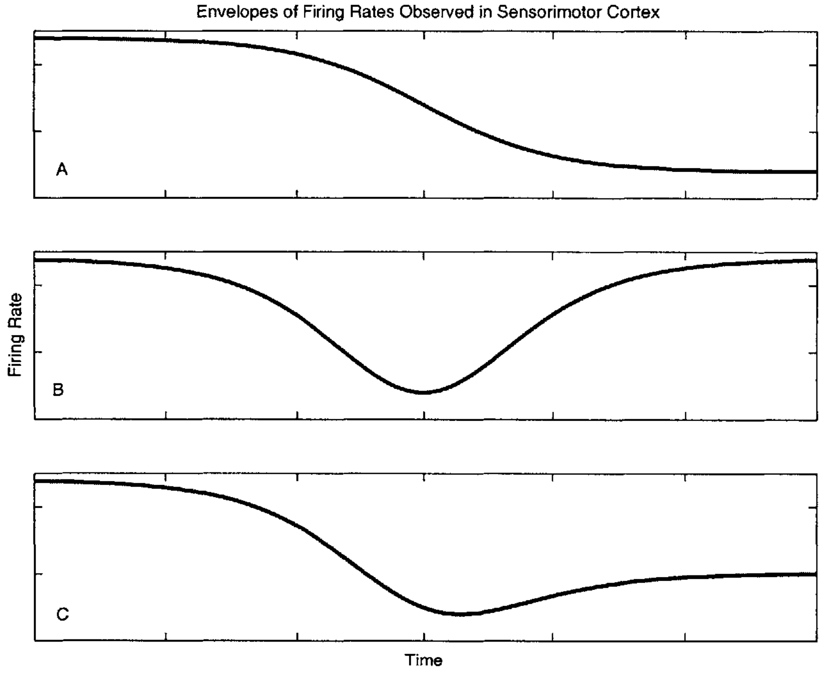


Figure 3-5: Neurons in the motor, premotor, sensory, and parietal cortices have firing rates that vary with time as reported in neurophysiological studies and shown above. The firing rate of a neuron may either increase in response to input signals (excitation), as was shown in figure 3-1, or decrease (inhibition) as shown here.

3.1.4 Network Output

In the motor cortex, pyramidal cells are the output neurons that integrate the information processed in the motor cortex and then project this information to the appropriate circuitry in the spinal cord. This concept of output neurons was useful in the network model of the proportional integrator command generator model (PICG) that is discussed below, where there was a need for output "neurons" that integrate the signal that comes out of the neuronal network and interface with the "plant" (the arm model). The arm model used requires four input signals: θ_s, θ_e, $\dot{\theta}_s$, and $\dot{\theta}_e$. So, the signals generated in the network of neurons as a result of remote inputs need to be channeled in an appropriate way to the arm model.

The output neurons are modeled again by equation 3.1. Again, the output of the neuron is a weighted combination of the inputs. But now, the inputs are from each of the local cortical neurons of the network (there are no "remote" inputs). In the PICG model, two output neurons were used, the reason for which will become apparent later. The weights of the connections of the the local cortical neurons onto the output neurons is probably the most crucial aspect of the whole model, and in this choice lies a lot of power and flexibility to make the network produce the required signals in response to some input.

In the PICG model, the weights of the connections of cortical neurons onto the output neurons were chosen such that the network as a whole behaved like a "kinematics converter"; that is, it converts the remote neuron inputs that specify a desired Cartesian position (or velocity) for the end-effector to the desired joint position (or velocity) by proper scaling. The output neurons were set up this way because under the circumstances explored in this work, unobstructed, unloaded, point-to-point movements in air, and because of the propositions in the literature that the motor cortex possibly processes information in joint coordinates while it receives inputs from areas, such as the parietal cortex, that possibly process information in Cartesian (extrapersonal) coordinates (these will be discussed in more detail below), then the function of the motor cortex would essentially be to convert the Cartesian commands it receives to joint coordinates that the spinal circuitry can process and route to the proper muscles. Mathematically speaking, the conversion between Cartesian and joint coordinates (inverse Jacobian) is a function of the position of the arm in the workspace, its (x, y) or (θ_s, θ_e) coordinates, as is apparent from equation 2.9. But, in the PICG model, it suffices to make the neuronal network approximate the inverse Jacobian of a central workspace position. Why and how this is so is the subject of the next section where the PI command generator will be discussed in detail.

3.2 The PI Command Generator

The proportional integrator command generator that is suggested in this work as a possible mechanism by which the motor cortex produces the desired signals to command an arm model is shown in figure 3-6. The blocks of remote neurons, local neurons, and output neurons have already been described in sections 3.1.1, 3.1.2, and 3.1.4. As has been described in section 3.1.1, four remote neurons signal the desired final location in Cartesian coordinates of the end-effector in the workspace. The time profile of the signals was a sigmoid, as in figure 3-1A and 3-5A. Physically, such a network of neurons could exist in the parietal cortex (areas 5 and 7) which monitors ongoing movement of the arm in the workspace, compares the movement to the intended movement commanded by central inputs, and modifies central commands in response to peripheral inputs [7] [32]. Since area 7 receives visual information and visual information is probably represented an extrapersonal frame of reference (the Cartesian representation of the workspace for example) [10], it has been suggested that information in the parietal cortex may be represented in extrapersonal coordinates [32].

Physically, the parietal cortex neurons project to the motor cortex; in the PICG, the remote neurons provide the input to the "local network," which is possibly realized in the motor cortex. The local network is set up as described in section 3.1.2. However, instead of using a hundred neurons or more with a random value assigned to the ξ variable corresponding to each neuron, only four neurons were used, having the

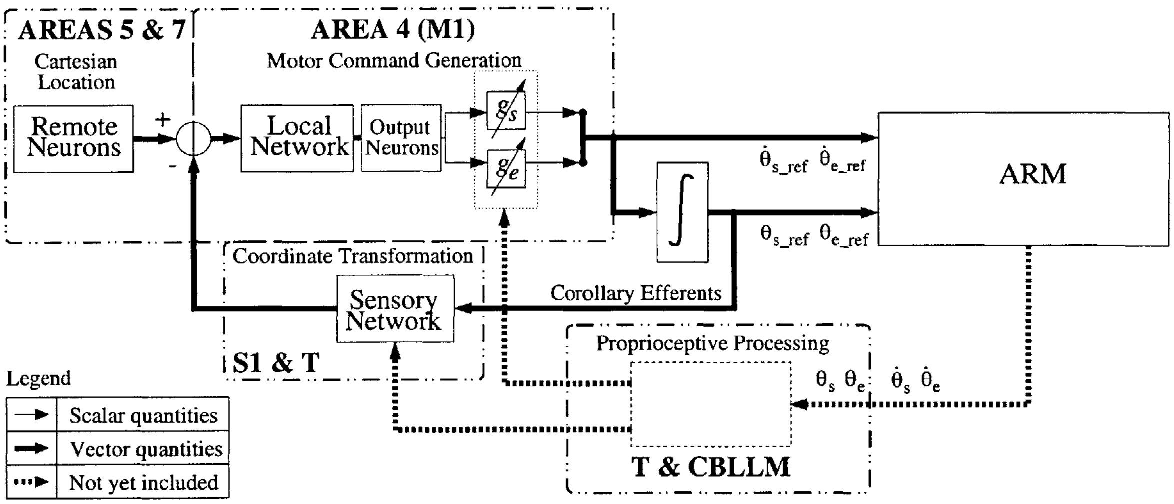


Figure 3-6: The PICG model provides the needed joint position and velocity signals to the arm model. The g_s and g_e blocks are tunable gains. The "sensory network" block in the smaller feedback path represents the processing of re-efferent signals, possibly by the thalamus (T) and sensory cortex (S1), and takes care of a needed coordinate transformation; it is discussed in the text. The dashed part of the figure represents parts that were not investigated in this work but are to be explored in future work. The dashed lines represent the flow of proprioceptive information that is fed back from the limb; the dashed blank box represents processing of that fed-back information, possibly by cerebellum (CBLLM), basal ganglia, and thalamus, before it reaches the cortical areas. It is proposed that the processed information is then used to adapt the gains g_s and g_e.

preselected ξ values: $\frac{\pi}{2}$, π, $\frac{3\pi}{2}$ and π. Essentially, the only effect that this has is to simplify the modeling by reducing the size of the needed local network; fixing the values of the ξ variable such that its values are uniformly spaced is analogous to having a large number of neurons with (uniformly) randomly assigned ξ values. The large number of the neurons is needed to obtain the population coding observed in the motor cortex [41] [49].

The output neuron block, again, set up as described previously in section 3.1.4, is such that the local network and the output network together effectively perform a kinematic conversion from Cartesian to joint coordinates. It should be noted that, in the PICG, the input to the local network is the difference between the desired Cartesian trajectory of the end-effector, as specified by the remote neurons, and the actual issued Cartesian trajectory that is fed-back via the sensory cortex (and discussed below). Physically, the comparison of the desired signal of the remote network with the fed-back signal could take place in the parietal cortex, since, as was mentioned above, it is believed to monitor and update command signals that it then sends to the motor cortex. It is also possible that the comparison occurs in the motor cortex itself, because it is known that the motor cortex receives re-efferent signals via the thalamus; or, the comparison could occur in both these cortical areas. In any case, the local neurons receive a bell-shaped velocity-like error signal like that shown in figure 3-1B and 3-5B, that, when scaled properly and goes through the inverse Jacobian transformation of the local and output networks, results in joint velocity signals.

The joint velocity signals project to both the arm model and an integrator; as such, the outputs of the integrator are the desired joint positions which are both projected to the arm and fed through a feedback module back for comparison with the desired signal. Thus, the PICG model is basically a servo that scales and tracks the desired input.

Physically, the integrator could be implemented in either the cerebellum, basal ganglia, or perhaps in the motor cortex itself. This is certainly possible via the dynamic responses of various types of neurons located in these structures and the interconnections between these neurons, but this is a thesis topic on its own and is an area of future work, as will be discussed in chapter 4. The fed-back joint position information goes through a block that performs another coordinate transformation which is needed for proper signal comparison. The block could be located in the thalamus and sensory motor cortex which are structures known to process joint information that they receive and project back to parietal and motor cortex; as discussed above, in the parietal or motor cortex the desired position might be then compared to the issued signal. This kind of a loop that consists of the signal to be issued to the "plant" being fed-back to the motor cortex is biologically plausible as it is known that the motor cortex receives such "re-efferent" information from cerebellum and basal ganglia via the thalamus. But the precise role of the re-efferent information in motor control is not known and has been subject of speculation in the literature; so, it is possible that the re-efferent information is actually used as proposed here.

Finally, the gain blocks labeled g_s and g_e in figure 3-6 are the tunable parameters of the model and model both adaption in the motor cortex and the need for several populations with different interconnections within the cortex. These gains provide the ability to fine tune the output of the command generator such that the desired behavior (performance) is achieved by the limb. Tuning these gains is equivalent to tuning the synaptic weights in the local network, which happens physically during learning. However, in this model, not only did it suffice to tune these two gains, as opposed to changing the weights in the local network, but it was simpler to see the effect of tuning by varying only these two parameters as opposed to varying many more parameters in the local network. Biologically, the "tuning" (learning) would occur as a result of peripheral information such as proprioceptive and visual information (as represented by the dotted line in figure 3-6 from the arm back to the gain blocks). The exact mechanism for such tuning is another area for further work, as will be discussed in chapter 4.

Thus, tuning g_s and g_e during a single movement (between the same two points) models adaption that happens in the motor cortex during learning. Alternatively, tuning g_s and g_e could take on a new meaning for different movements (that is, movements between different sets of points). As will be discussed below, the gain parameters may take on different values to achieve the same kinematic performance of the arm in different regions in the workspace. This is expected because the connections of the local network are kept constant and, when taken together with the output neurons they provide the inverse Jacobian of a fixed location in the workspace; but as has been previously pointed out, the inverse Jacobian is not the same all over the workspace. Thus, tuning the gains serves to improve the inverse Jacobian transformation that is implemented. Physically, the local network, output neurons and new gain parameters could be thought of as a new population of neurons, with a different set of interconnections, that comes into play when a different region in the workspace is traversed. These roles of the gain parameters will become clearer as these parameters are discussed below in the context of the arm kinematics that are simulated by the model.

3.2.1 Simulated Features of Arm Kinematics

The PICG was tested on a second order arm model (refer to equation 2.66) using MATLAB's Simulink, as described in appendix A. As was mentioned previously, it is assumed that the spinal cord circuitry takes care of scaling and routing the PICG signals to the proper muscles; since such circuitry was not explored in this work (but could be another area for future work), the six-muscle models were not used in the simulations.

The performance of the PICG was tested using the arm model by simulating movements in different directions and in different locations of the workspace. Figure 3-7 shows sample simulated hand trajectories for medium-speed movements (having a peak hand tangential velocity of $170mm/s$) starting at the center and ending at the eight different positions around the circle. The thick lines are the trajectories followed by the hand when it receives the joint velocity and position commands from the PICG; the thin lines are the trajectories followed by the hand when it receives the proper joint velocity and position commands directly. That is, the thin lines take on the shown curved trajectories due to the dynamics of the arm model; but the trajectories

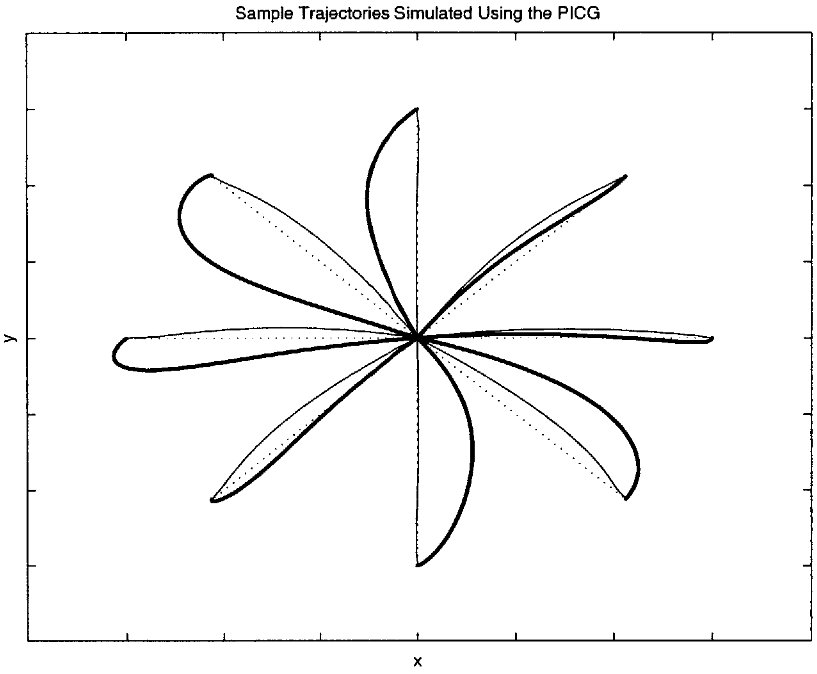


Figure 3-7: When g_s and g_e are fixed at 3, the commands issued by the PICG cause the end-effector of the second order arm model to follow the thick trajectories. The thin lines are the trajectories that would be followed by the hand had the desired joint position and velocity commands been issued directly to the arm, without a PICG module. The dotted lines are the straight line paths from the central starting position of the movement to each of the eight target positions located $0.15m$ from the starting position. The peak speed of hand movement is approximately $170mm/s$.

produced using the PICG are quite different and are not characteristic of the arm dynamics alone.

It is clear from figure 3-7 that performance varies for the different directions when the gain parameters are fixed at the same values for all the directions; more specifically, the trajectories followed in the 0°, 45°, and 225° directions are closer to the dotted straight line paths. Still, by tuning the gain parameters for each of the different movements in the workspace, the required performance can be obtained; this is illustrated in figure 3-9. It is significant to note, however, that the PICG trajectories are qualitatively similar to observed hand paths reported in [17], [31], [50], and [2]; furthermore, the tangential velocity of the hand is approximately bell-shaped as shown in figure 3-8 and also reported in the aforementioned studies.

For example, in figure 3-7, the shapes of the PICG model generated hand trajec-

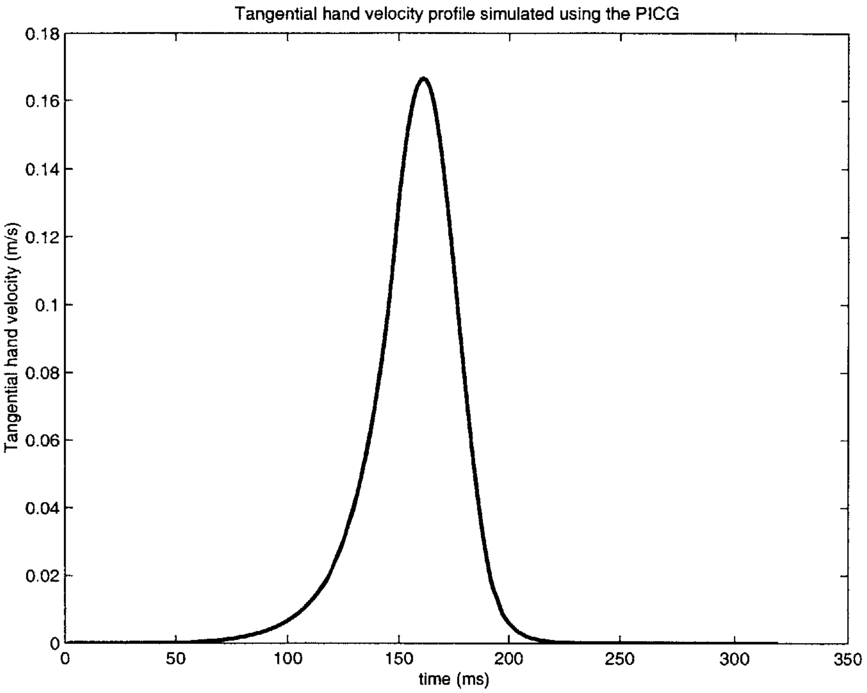


Figure 3-8: The tangential hand velocity has a bell-shaped time profile, as shown in this sample plot, where $g_s = g_e = 3$ and the movement direction is 45°.

tories for the movement in the 0°, 45°, and 225° directions look very much like the trajectories recorded by Flash and reported in [17]. Noteworthy are the small hooks that occur at the end of the movements, which are more visible in figure 3-9, and are not at all present when the arm model is simulated without the PICG. The trajectories in the other directions shown in figure 3-7 look like those recorded in null fields (especially the 180° trajectory) and in force fields [50] [2] [3], as well as those reported in [31] when visual feedback is modified (especially the 135° trajectory). Tuning the gain parameters, however, results in simulated trajectories that are similar to those observed under more normal conditions.

The effect of varying g is shown in figure 3-9 for fast movements (peak hand tangential velocity of $500mm/s$). The higher the values of the gain parameters, the more accurately the output of the PICG tracks the desired input, so the performance of the model improves; still, even for the relatively large gains, $g_s = g_e = 20$, the trajectory is curved in a way that is not characteristic of the dynamics of the arm alone (refer to figure 3-7). This is encouraging because physically, it has been observed

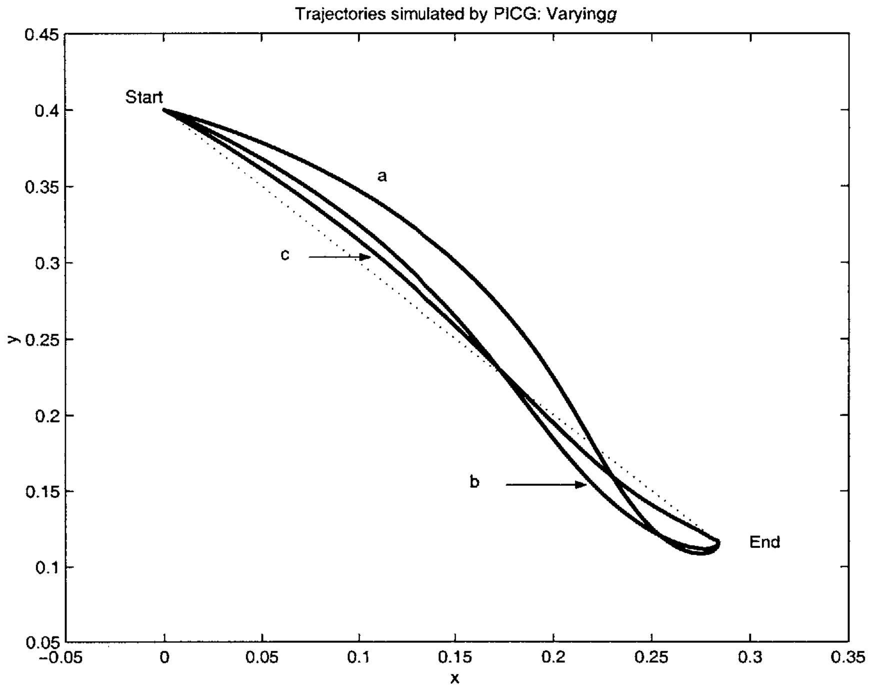


Figure 3-9: The trajectories generated by the hand due to the PICG model improve as the g parameter increases. Fast movements (peak tangential hand speed of $500mm/s$) in a direction 315° from the horizontal are simulated here when $g_s = g_e = 3$ (a), $g_s = 5$, $g_e = 7$ (b), and $g_s = g_e = 20$ (c).

that fast movements are rarely perfectly straight, and because the combined PICG-arm dynamics seem to match real movements nicely. As for "large" gain values, further work needs to be done to determine the limits of amplification of neuronal firing rates, and this entails a closer look at the dynamic responses of individual neurons and as the neruons interact within larger groups; this will be discussed in chapter 4. Still, it should be noted that these larger values may not even be needed in the presence of proprioceptive feedback and correction via the cerebellum, which is completely ignored in this model but will be added in future work.

In figure 3-10, the values of the gains, g_s and g_e, have been roughly optimized for each of the eight movement directions; the gains were manually tuned to achieve the best possible linear hand trajectory. In some cases, further tuning of the gains did not yield any notable improvements to the trajectories shown in figure 3-10, as for the 0°, 45°, 90° and 225° trajectories; but in others, it was not possible to achieve the best performance without increasing the gain parameters too much, as for the

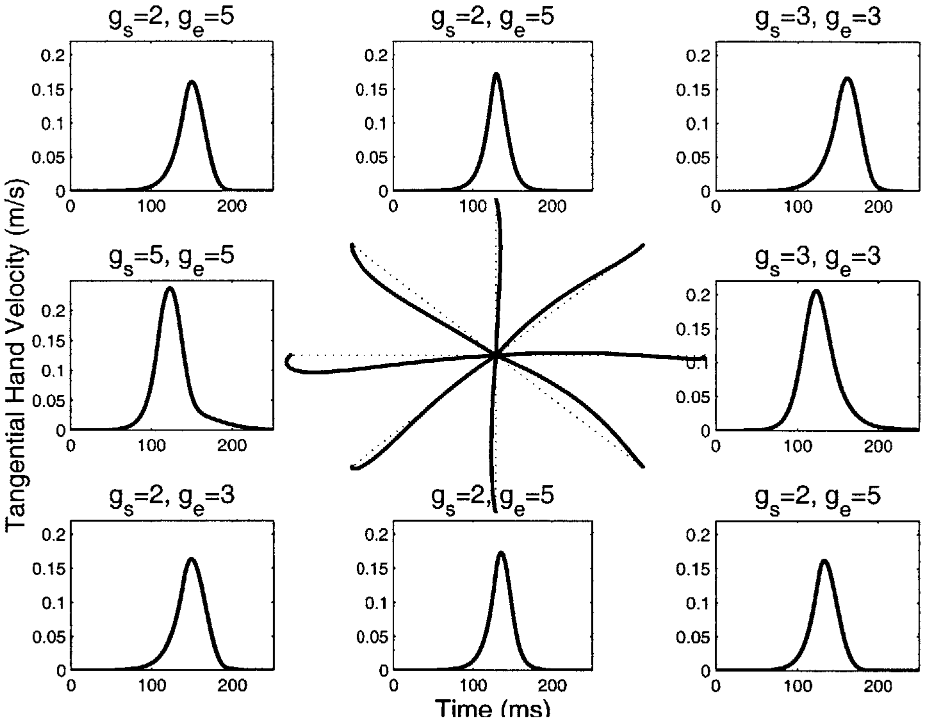


Figure 3-10: The central plot shows the trajectories followed by the hand moving a distance of $0.15m$ in eight directions when the gains, g_s and g_e, have been tuned for best performance, as described in the text. The surrounding plots show the bell-shaped tangential hand velocities for the movement trajectories shown in the center. The optimal gains, g_s and g_e, are indicated for each of the eight movements over the corresponding plot of tangential hand velocity.

135°, 180°, and 315° trajectories. This result resembles biological reality as human arm movements characteristically do not follow perfectly linear trajectories except for very slow movements. The movements illustrated in figure 3-10 cannot be considered slow; the peak tangential hand velocities lie between $160mm/s$ and $250mm/s$, as can be seen on the tangential hand velocity plots for each of the eight movements.

Finally, the PICG-arm model also exhibits qualities of real movements for fast versus slow movements and for movements of different distances. When the gain values, g_s and g_e, are fixed, slow movements simulated by the PICG-arm model are straighter than fast movements. This is illustrated in figure 3-11. Figure 3-12 shows that when the gain values are fixed for a given movement direction, movements through smaller distances tend to be straighter than movements through larger distances. The movements through larger distances are simulated by having the appropriate remote neurons provide a sigmoid of a larger step magnitude, that is, a sigmoid of larger final value; however, the duration of the step, or the time for the value of the signal to increase from its initial value to its final value, remains fixed. The result of this is that, for any movement direction, the peak tangential hand velocity, shown in figure 3-12 for the movements in each of the eight directions, doubles as the distance of the movement is doubled. This is a characteristic of "speed-sensitive" movement (refer to section 1.1.3) which was hypothesized to occur when the CNS issues a "pulse" of increasing magnitude but fixed duration [28]. The fact that PICG-arm model does indeed exhibit speed-sensitive behavior due to increasing the magnitude of the remote neuron signals and keeping the duration of the command fixed, points to the possibility that the CNS may in fact employ the aforementioned command strategy to achieve speed-sensitive movement.

3.2.2 Simulated Features of Neuronal Activity

In addition to causing the arm model to closely approximate the kinematics of arm movement when used to provide the command signals, the PICG model also exhibited the features of neuronal activity that are characteristic of the sensorimotor cortex, such as directional tuning and population coding; refer to section 1.1.2 for a discussion

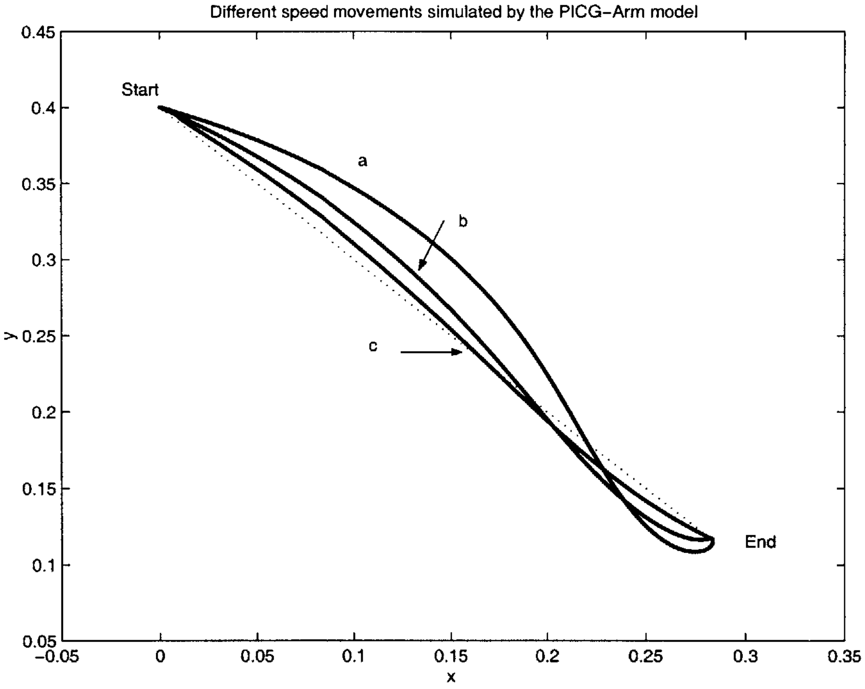


Figure 3-11: The trajectories generated by the hand due to the PICG model improve as the movement speed drops. The gain parameters are fixed at $g_s = g_e = 3$ as the movements in a direction 315° from the horizontal are simulated for different movement speeds, with peak tangential hand velocities: $500mm/s$ (a), $180mm/s$ (b), and $80mm/s$ (c).

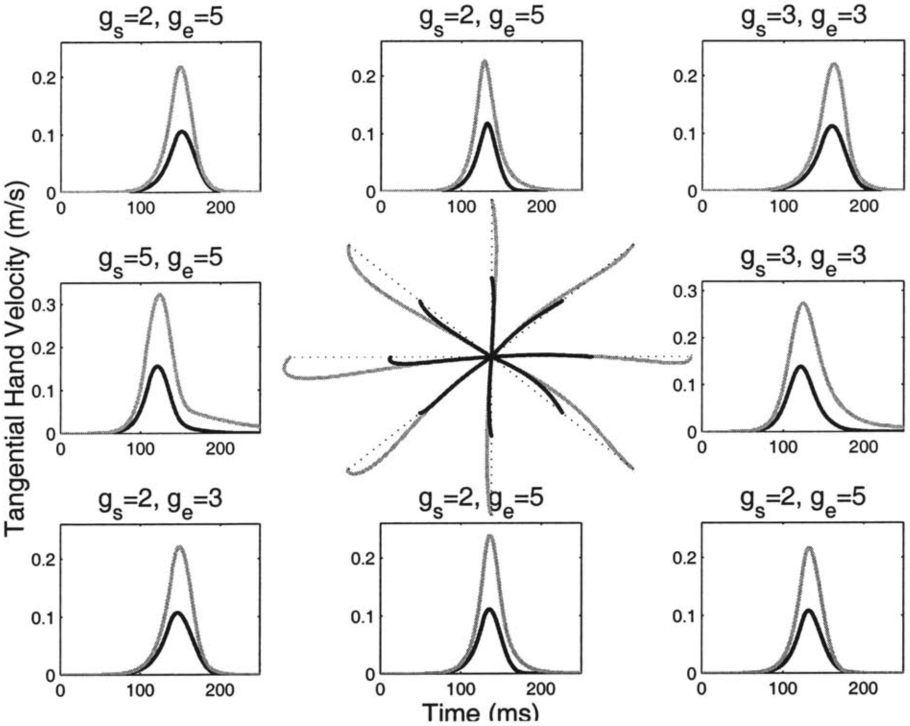


Figure 3-12: The central plot shows the hand trajectories simulated by the PICG-arm model in the eight directions for two movement distances: $0.1m$ (shown in black) and $0.2m$ (shown in grey). The surrounding plots of tangential hand velocities for both movement distances, $0.1m$ (black) and $0.2m$ (grey), in each of the eight movement directions show that as the movement distance doubles, the peak tangential hand velocity also doubles. The values of the gains, g_s and g_e, are indicated for each of the eight movements over the corresponding plot of tangential hand velocity.

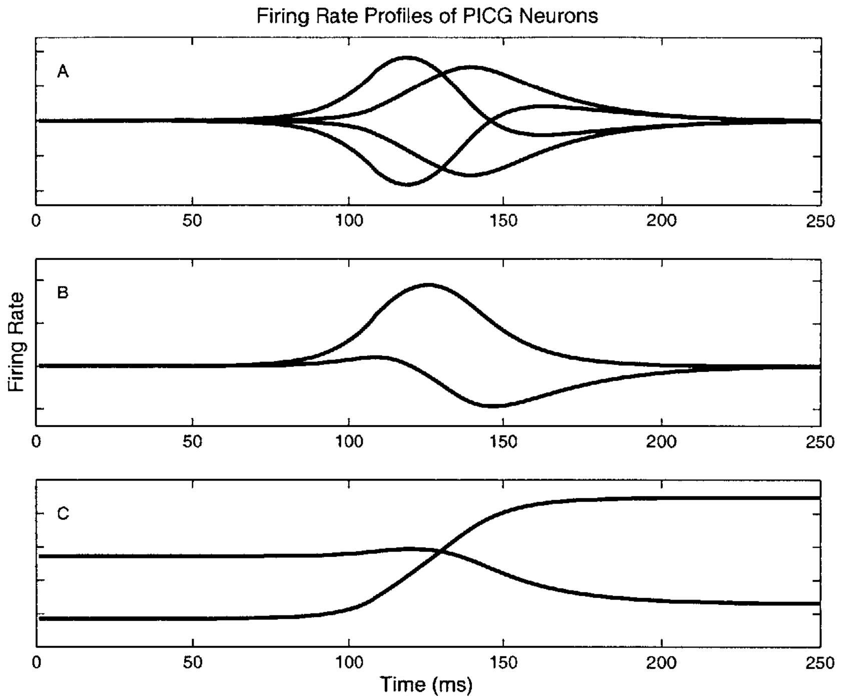


Figure 3-13: The input and output signals to the local network, (A) and (B) respectively, and the input signals to the re-efferent network (C) are shown for a hand movement of $0.15m$ in the 180° direction and peak tangential hand velocity of $230mm/s$. $g_s = g_e = 5$.

of directional tuning and population coding. But first of all, since the input signal generated by the remote network is a sigmoidal signal, the error signal input to the local network is a bell-shaped signal, and in fact, all the signals in the PICG are roughly either bell-shaped or sigmoids, as shown in figure 3-13; that is, they resemble the signals actually observed in the sensorimotor cortex (refer to figure 3-1).

Secondly, the local network "neurons" were indeed directionally tuned, though, as was explained in section 3.1.2, directional tuning is not a property of the firing function of the neuron, but is a result of the *way the network is wired.* The PICG-arm model was run for the arm starting at a central position and reaching towards each of the eight targets, at 0°, 45°, 90°, 135°, 180°, 225°, 270°, and 315°. Both the average and the maximum output firing rates of each of the local neurons were recorded for each of the trials and plotted against the direction of arm movement. Both plotting the maximum firing rate and the average firing rate, the latter being the one usually used in studies such as [46] and [39], revealed that the neurons were in

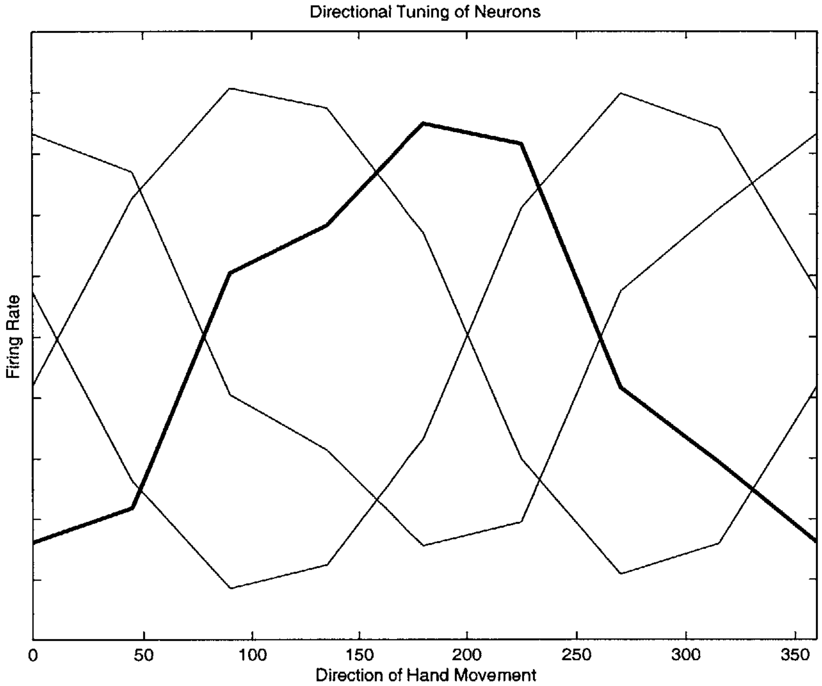


Figure 3-14: Each of the four curves represents the maximal firing rate of one of the four neurons of the local network during arm reaching toward a target, plotted versus the direction of movement of the arm. The local neurons exhibit maximal discharge rates in their "preferred" directions: 0°, 90°, 180°, and 270°.

fact directionally tuned. A sample plot is shown in figure 3-14, where it can be seen that discharge rate of each neuron was highest with movements in a certain direction (0°, 90°, 180°, and 270°).

Finally, using the above preferred direction information, the population code algorithm was implemented using readings of neuron firing rates that were gathered from simulations of arm movements in the eight different directions. The population vector that was calculated for each direction had a direction close to the real movement direction, but within a margin of error; figure 3-15 shows the population vectors calculated using the simulations. The difference between the direction of the population vector and the actual arm movement direction is probably due to the fact that the number of neurons used to derive the population vectors is too small (four only); when a larger population is used to calculate the population vector, the error will average out. So, it seems reasonable to conclude that the PICG neurons satisfactorily exhibit population coding.

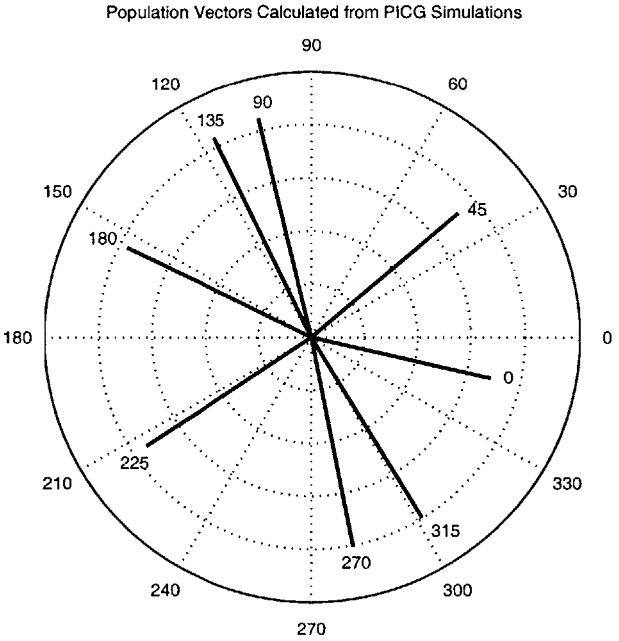


Figure 3-15: The thick lines are the population vectors that are calculated using the firing rates of the four local neurons of the PICG when arm movement was simulated in each of the eight target directions around the circle. The numbers at the tip of each population vector are the directions of movement of the arm for which the population vector was calculated.

3.3 Summary

In this chapter, a sensorimotor proportional-integrator command generator (PICG) has been proposed to model the mechanism by which the primary motor area (M1) interacts with surrounding areas, the primary sensory area (S1) and posterior parietal areas (areas 5 and 7), to produce the feedforward signals that are needed to command the arm, modeled by a standard second order nonlinear arm-spinal cord model, during unloaded and unobstructed point-to-point arm reaching. The goal was to investigate the simplest possible structure that would be needed for the PICG, but remains true to the biology of the sensorimotor cortex.

The PICG model that is proposed is a "skeleton" model that is based on several simplifying assumptions that are biologically reasonable. Still, when simulated with the arm module, the kinematic performance of the overall sensorimotor-arm system resembles performance of human and primate arms, especially in the curvature of point-to point movements and the properties of slow versus fast and short versus long movements. Furthermore, the PICG module exhibits some of the characteristics of cortical neuron activity such as directional tuning and population vector coding. These results are powerful in light of the simplicity of the proposed PICG model, but, there is much to be learned by gradually taking away the simplifying assumptions made in this work and building up the complexity of both the arm and PICG models to resemble the real biological system more accurately. Improvements that are to be made in future work are discussed in the next chapter, and the work reported in this thesis is evaluated.

Chapter 4

Future Work

When studying the work done to model the function of the brain, one discerns two approaches to modeling. The first approach attempts to construct the model using as building blocks detailed models of the building blocks of the system itself. For example, detailed models of neurons and the knowledge of their interconnections are used to put together models of a brain area with some known function and behavior. The second approach depends on the knowledge of the features and attributes (input-output behavior) of the system to reconstruct its behaviors using as parsimonious assumptions that observe the limits imposed by the known physical properties of the system as possible. To begin with, both approaches require different degrees of abstraction; the extent to which one abstracts the elements of a model depends on the amount of complexity one is willing to forgo, or, rather, needs to reduce. And this changes as a model is tested and refined, and depending on what is expected and required of the model.

In this thesis, the second approach was the one adopted; the sensorimotor cortex function was investigated in a very particular context, un-obstructed and un-loaded point-to-point arm reaching, the kinematics of which has been studied extensively, and for which there exists a wealth of studies and electrophysiological data from the sensorimotor cortex and other involved brain regions. But, these studies indicate, at best, correlations between the observed neuronal firing rates and the behavior of the arm and characteristics of movement, rather than causal relationships. So, what the variables are that the brain encodes in its signals, how the brain signals are translated to produce movement, and why the movements have certain characteristics or stereotypes is still unknown.

Hence, there is a need for models that aim to understand the nature of the signals of the brain in the context of its functions; models that might explain what the signals are that neurophysiologists are recording and how they are used. In fact, there are not many models in the literature that attempt to do this while observing the anatomical features of the brain; many of the models that have been proposed are neural network models, which are often times not biologically reasonable. For example, these models may require training via methods that are not necessarily biologically plausible, or require an unreasonably large number of iterations to learn a behavior, which is not necessarily true of human learning. So, it is the aim here to remain as true to the physiology of the brain as possible, while using reasonable abstractions. A successful model should reproduce the observed behavior (arm reaching) and its characteristic features (kinematics of the arm and dynamics of the brain) while allowing for adaptation. Finally, it should be able to predict the signals to be measured from the brain and thus lend insight to how the brain works, the ultimate goal of an endeavor, such as this modest one, in this field.

4.1 Evaluation of the Work Done

The work in this thesis was done in two parts. First, the "plant", the arm, was modeled; second, the needed command generator was suggested. The arm modeling provided insight and justification to the signals that are claimed to be needed of a command generator. Thus, modeling the arm provided a launch-pad for attacking the problem of brain modeling.

Part of arm modeling entailed the exploration of the equilibrium point hypothesis and its two main variations: the lambda and the alpha models. Some level of detail was observed in the modeling, in that each muscle was modeled using both active and passive elements and including calcium activation dynamics. But other detail was left out, for example, proprioceptive information carried by Golgi tendon organs was ignored and spindle dynamics were not modeled, so the Ia-afferents were assumed to carry joint position and velocity signals. This was of no consequence in this work because the proprioceptive feed-back from the arm was not used by the PICG, and in fact, this level of detail is beyond that adopted in this work; for this work, the assumption of the availability of proprioceptive joint and position signals *somewhere* in the nervous system and the fact that that information ultimately reaches the brain is sufficient. However, as the model becomes more detailed in future work, the issue of adding more biological detail to the plant model should be addressed.

Looking at the two versions of the arm model was instructive in that it was an exploration of the differences and similarities of two models that are in the literature and have divided opinion. It seems that each of the two models is an approximation that emphasizes a different feature of the muscle's force-length activation curves. While the alpha model emphasizes the change of the slope of the static force length curves as activation to the muscle changes, the lambda model emphasizes the shift in the curves. In reality, both things happen at the same time, and can be taken into account in the same model. This "combination" model need not be more complex than either of the alpha or lambda models because complexity can be reduced using abstraction that does not necessarily forfeit the important features of the muscle activation curves. Such a combination model should be the one used if more biological detail is to be added. Otherwise, the second order model was sufficient for the level of detail used in this work. Still, simulating the more complex nonlinear models was important to gain an understanding of the activation signals that are to be issued to the brain and of the extent, if at all, to which the added complexity could be significant from the perspective of the brain.

Given the knowledge of the activation signals that the arm needs to be driven, the second part of the thesis suggests the mechanism by which the sensorimotor cortex generates those signals. The model suggested was based on its required function, the generation of the joint signals using a kinematic conversion of variables, and the available building blocks to provide that function, the abstract model of the neuron.

Some pieces of the model were suggested as a matter of necessity, like the integrator blocks; but the existence of brain structures that perform suggested functions, such as integration, does not seem to be unreasonable. Still, where and how integration of a signal may occur in the brain are issues for future investigation.

Nevertheless, the suggested PICG model was expected to satisfactorily simulate reaching movements, and it indeed did. The simulated reaching movements were not characteristic of the arm model alone; but the dynamics of both the PICG and arm model combined resulted in kinematics that were similar to those of human and monkey movements. Furthermore, the tunable gains in the model allow for adaption or learning, though an algorithm for this needs to be worked out in the future. In fact, though the PICG model performs quite well considering its simplicity and the level of detail and abstraction employed, on the one hand, it bears room for improvement, and on the second hand, the effect of additional detail in the modeling on the performance of the model seems to be interesting to explore.

4.2 Improvements to the Model

Both the PICG model and the arm model used in this work could be refined to incorporate more biological detail. Improvements and small changes to the the second order arm model that was used in the PICG-arm simulations may affect the performance of the PICG and are thus important to carry out and test. For example, the stiffness and viscosity matrices used in the arm model were held constant, although in reality, experimental measurements have shown that these values change as a function of the arm position in the workspace. Secondly, reflex delays were neglected for simplicity, and these should be added to make the model more biologically accurate.

There are some minor additions that are to be made to the PICG model as well. For example, one could address the issue of interfacing the PICG to a six-model arm via a block whose function might be physically accomplished by the spinal cord. More interesting to explore, however, are adjustments that require more in-depth work. For example, closing the loop of proprioceptive information from the arm model to the brain entails more work to determine the adaption mechanism of the cortex, or an algorithm by which the gains g_s and g_e are tuned and how they vary in different locations in the workspace, and a more detailed model of the thalamus and sensory cortex, the solid blank box in figure 3-6.

Secondly, the issue of how the CNS deals with the time delays in the flow of afferent information from the arm to the brain and efferent information the other way round has been addressed by other work [36], and of interest in regard to this work is how well the suggested scheme of [36] could work with the PICG in place. Furthermore, one could study the need and applicability of the wave variable method of [36], or other predictive methods, to handle the delays in the re-efferent pathway, the solid feedback loop in figure 3-6, which are shorter than the afferent delays.

Finally, the neuronal dynamics that were ignored in this work in order to simplify the model in the first stage of modeling should be included, as the model is refined, to better represent the biology of the brain, which was one of the priorities from the outset of this work. At a third stage of refinement, one can also take into account the stochastic nature of the cell responses.

So, in conclusion, given that the PICG-arm model presented in this thesis is a first approximation model that was built on simplifying assumptions and abstractions, it seemed to model well the behavior of the arm during unobstructed and unloaded point-to-point arm reaching through air; the main characteristics of the kinematics of simulated movement matched the experimentally observed characteristics, and some features of cortical neurons such as directional tuning and population coding were also features of the PICG model, even though these features were not directly hard-wired into the model. The model still needs work, especially in the addition of the proprioceptive feedback loop. The addition of more biological detail and refinements to the model might turn out to be necessary as this change, and others, such as the aforementioned improvements to the arm model, are implemented.

Appendix A

Simulink Simulations

The PICG-Arm model was implemented and simulated using MATLAB's Simulink package. The block diagram of the Simulink implementation is shown in figure A-1. Indicated on the figure are the blocks corresponding to the remote neurons, the local network, output neurons and the re-efferent network that have been shown in figure 3-6 and discussed in detail in chapter 3. Although 4 remote neurons and 4 local neurons were used in the actual simulations discussed in chapter 3, for the sake of generality, it is assumed in the following discussion of the Simulink block diagram that there are N remote neurons and M local neurons.

As was discussed in section 3.1.1, the remote neurons encode Cartesian coordinates of target hand positions in the workspace. For a fixed desired hand movement distance, the k^{th} remote neuron fires maximally when some direction, ϕ_k, with which that remote neuron is associated, is required. If desired movement is thought of as a vector pointing in the desired direction and having some magnitude (possibly proportional to the desired velocity), the firing rate of the k^{th} neuron depends on the magnitude of the component of the desired movement vector in the direction ϕ_k. Thus, each remote neuron is associated with a firing rate (a magnitude), z_k, and a direction, ϕ_k. The desired Cartesian position will then be encoded by the population of remote neurons according to:

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = \begin{bmatrix} \cos\phi_1 & \cos\phi_2 & \dots & \cos\phi_N \\ \sin\phi_1 & \sin\phi_2 & \dots & \sin\phi_N \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ \dots \\ z_N \end{bmatrix} \tag{A.1}$$

where x_d and y_d are the Cartesian coordinates that the arm is required to track (and can vary with time) and,

$$T = \begin{bmatrix} \cos\phi_1 & \cos\phi_2 & \dots & \cos\phi_N \\ \sin\phi_1 & \sin\phi_2 & \dots & \sin\phi_N \end{bmatrix}. \tag{A.2}$$

The remote neuron block in figure A-1 translates the trajectory, specified in Cartesian coordinates, that the arm is required to track to the proper firing rate profiles of the appropriate remote neurons via the block labeled T^{-1} in figure A-1, which is the pseudo-inverse of T. The output of the block T^{-1} is the vector, $\begin{bmatrix} z_1 & z_2 & \dots & z_N \end{bmatrix}^{\mathrm{T}}$, that contains the firing rate magnitudes of the N remote neurons at any time instant. In order for the temporal profiles of these firing rates to take on the assumed sigmoidal shapes (refer to section 3.1), the Cartesian trajectories are issued as sigmoids by the block "p2cart", which also converts the desired movement distance and direction, which are more convenient to specify, to the Cartesian coordinates of the final position. The parameters A and C of p2cart are related to the shape of the sigmoids produced by the block; C is the time, t, at which the magnitude of the sigmoid is $0.5 * (x_f - x_o) + x_o$, where x_f and x_o are the final and initial values of the signal and A specifies the slope (hence the duration for a fixed step magnitude) of the rising portion of the sigmoid.

The signals generated by the remote neurons drive the local network neurons, as was explained in chapter 3. The output firing rate, y_f, of the j^{th} local neuron is, according to equation 3.1, and incorporating all the assumptions pertaining to the output firing rate as discussed in chapter 3:

$$y_{f_j}{}^{t+1} = \sum_{\mathrm{i}} w_{ij} y_{f_i}{}^{t} + \sum_{\mathrm{k}} v_{kj} z_k{}^{t} \tag{A.3}$$

where $y_{f_j}{}^{t+1}$ is the output firing rate of the j^{th} neuron at time interval $t+1$, w_{ij} and v_{kj} are synaptic weights that are described in sections 3.1.1 and 3.1.2 and $z_k{}^t$ (used instead of x_k for the sake of clarity and continuity from the above discussion) is the firing rate of the k^{th} remote neuron at the time interval t.

Rewriting equation A.3,

$$y_{f_j}{}^{t+1} = \begin{bmatrix} w_{1j} & w_{2j} & \dots & w_{Mj} \end{bmatrix} \begin{bmatrix} y_{f_1}{}^t \\ y_{f_2}{}^t \\ \dots \\ y_{f_M}{}^t \end{bmatrix} + \begin{bmatrix} v_{1j} & v_{2j} & \dots & v_{Nj} \end{bmatrix} \begin{bmatrix} z_1{}^t \\ z_2{}^t \\ \dots \\ z_N{}^t \end{bmatrix} . \tag{A.4}$$

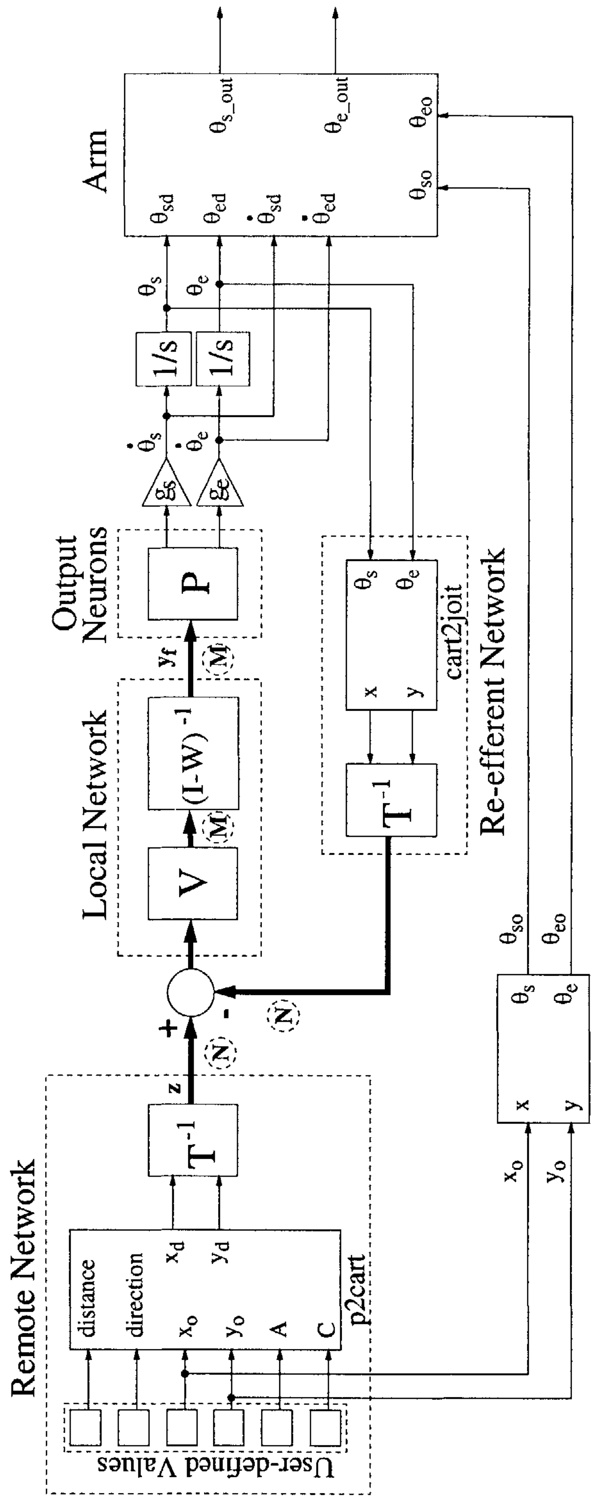


Figure A-1: The block diagram of the Simulink implementation of the PICG-arm model is shown here, and all variables and blocks are described in the text. The thin lines indicate pathways of scalars; the thick lines indicate pathways for vector quantities, the lengths of which are enclosed in the dashed circles.

And, for the M local neurons:

$$\begin{bmatrix} {y_{f_1}}^{t+1} \\ {y_{f_2}}^{t+1} \\ \dots \\ {y_{f_M}}^{t+1} \end{bmatrix} = \begin{bmatrix} w_{11} & w_{21} & \dots & w_{M1} \\ w_{12} & w_{22} & \dots & w_{M2} \\ \vdots & \vdots & \ddots & \vdots \\ w_{1M} & w_{2M} & \dots & w_{MM} \end{bmatrix} \begin{bmatrix} {y_{f_1}}^{t} \\ {y_{f_2}}^{t} \\ \dots \\ {y_{f_M}}^{t} \end{bmatrix} + \begin{bmatrix} v_{11} & v_{21} & \dots & v_{N1} \\ v_{12} & v_{22} & \dots & v_{N2} \\ \vdots & \vdots & \ddots & \vdots \\ v_{1N} & v_{2N} & \dots & v_{NN} \end{bmatrix} \begin{bmatrix} {z_1}^{t} \\ {z_2}^{t} \\ \dots \\ {z_N}^{t} \end{bmatrix}, \tag{A.5}$$

which is, in matrix notation,

$$\mathbf{y_f}^{\mathbf{t+1}} = \mathbf{W}\mathbf{y_f}^{\mathbf{t}} + \mathbf{V}\mathbf{z}^{\mathbf{t}} \tag{A.6}$$

where, as explained in chapter 3, the elements of the NxN matrix $\mathbf{W}$ and the MxN matrix $\mathbf{V}$ are calculated according to equations 3.2 and 3.3.

The superscripts indicating time intervals are included here to emphasize that, in general, the output firing rate of a neuron is not instantaneously affected by the inputs to the neuron. In fact, if one assumes fixed length time intervals, the output of a given neuron need not be updated by the next time interval; furthermore, updating for all neurons is not necessarily synchronous, that is, the outputs of different neurons are updated at different times. These are details that will be addressed in future work as they potentially raise important issues such as well-posedness and stability. For now, though, they are ignored for the sake of simplicity, and, as stated in chapter 3, the output is assumed to be instantaneously affected by the inputs. That is:

$$\mathbf{y_f} = \mathbf{W}\mathbf{y_f} + \mathbf{V}\mathbf{z}. \tag{A.7}$$

So,

$$\mathbf{y_f} = (\mathbf{I} - \mathbf{W})^{-1}\mathbf{V}\mathbf{z}. \tag{A.8}$$

The matrices $\mathbf{V}$ and $(\mathbf{I} - \mathbf{W})^{-1}$ of equation A.8 are implemented in the blocks of the same names that are shown in figure A-1. The block "P" represents a matrix $\mathbf{P}$ that corresponds to the strength of the connections from the local neurons to the output neurons. As explained in chapter 3, these connections are chosen such that the local and output neurons perform an inverse Jacobian transformation on the Cartesian commands issued by the remote neurons. That is, **P** is chosen such that:

$$\mathbf{J^{-1}} = \mathbf{P}(\mathbf{I} - \mathbf{W})^{-1}\mathbf{V}\mathbf{T^{-1}} \tag{A.9}$$

or equivalently,

$$\mathbf{J^{-1}T} = \mathbf{P}(\mathbf{I} - \mathbf{W})^{-1}\mathbf{V} \tag{A.10}$$

where $\mathbf{J^{-1}}$ is the inverse Jacobian for a "central workspace position", $x = 0.1m$ and $y = 0.2m$. **P** is calculated using the "pinv" function in MATLAB.

Finally, the re-efferent network shown in figure A-1 performs a transformation from joint coordinates to Cartesian coordinates. This is done by the block labeled "cart2joit" using equation 2.7. The T^{-1} block distributes and interfaces the output from the cart2joit block to N signals that correspond to each of the remote neurons.

Bibliography

[1] G. L. Almeida, D.-A. Hong, D. Corcos, and G. L. Gottlieb. Organizing principles for voluntary movement: Extending single joint rules. *Journal of Neurophysiology*, 74:1374–1381, 1995.

[2] N. Bhushan. A computational approach to human adaptive motor control. Master's thesis, Johns hopkins University, 1998.

[3] N. Bhushan and R. Shadmehr. Computational nature of human adaptive control during learning of reaching movements in force fields. *Biological Cybernetics*, 81:39–60, 1999.

[4] B. Bizzi, W. Chapple, and N. Hogan. Mechanical properties of muscles: Implications for motor control. *TINS*, 5(11):395–398, November 1982.

[5] E. Bizzi. Central and peripheral mechanisms in motor control. In G.E. Stelmach and J. Requin, editors, *Tutorials in motor behavior*, Advances in Psychology, pages 131–144. North Holland, Amsterdam, 1980.

[6] E. Bizzi, N. Hogan, F. A. Mussa-Ivaldi, and Giszter S. Does the nervous system use equilibrium-point control to guide single and multiple joint movements? In P. Cordo and S. R. Harnad, editors, *Movement Control*, pages 1–11. Cambridge University Press, Cambridge, England, 1994.

[7] V. Brooks. *The Neural Basis of Motor Control*. Oxford University Press, Oxford, 1986.

[8] I. E. Brown, Scott S. H., and Loeb G. E. Mechanics of feline soleus: II design and validation of a mathematical model. *Journal of Muscle Research and Cell Motility*, 17:221–233, 1996.

[9] P. Burbaud, C. Doegle, C. Gross, and B. Bioulac. A quantitative study of neuronal discharge in areas 5, 2, and 4 of the monkey during fast arm movements. *Journal of Neurophysiology*, 66(2):429–443, August 1991.

[10] R. Caminiti, P. B. Johnson, and A. Urbano. Making arm movements within different parts of space: Dynamic aspectsin the primate motor cortex. *Journal of Neuroscience*, 10(7):2039–2058, 1990.

[11] A. G. Fel'dman. Change in the length of the muscle as a consequence of a shift in equilibrium in the muscle-load system. *Biofizika*, 19(3):534–538, 1974.

[12] A. G. Fel'dman. Control of the length of the muscle. *Biofizika*, 19(4):749–753, 1974.

[13] A. G. Feldman. Once more on the equilibrium-point hypothesis (lambda model) for motor control. *Journal of Motor Behavior*, 18(1):17–54, 1986.

[14] A. G. Feldman, S. V. Adamovich, and D. J. Ostry. The origin of electromyograms: Explanations based on the equilibrium point hypothesis. In J. M. Winters and S. L-Y. Woo, editors, *Multiple Muscle Systems: Biomechanics and Movement Organization*. Springer-Verlag, New York, 1990.

[15] A. G. Feldman and M .F. Levin. The origin and use of positional frames of reference in motor contro. *Behavioral and Brain Sciences*, 19:723–806, 1995.

[16] E. E. Fetz. Are movement parameters recognizably coded in the activity of single neuron? *Behavioral and Brain Sciences*, 15:679–690, 1992.

[17] T. Flash. The control of hand equilibrium trajectories in multi-joint arm movements. *Biological Cybernetics*, 57:257–274, 1987.

[18] A. A. Frolov, M. Dufosse, S. Rizek, and A. Kaladjian. On the possibility of linear modeling the human arm neuromuscular apparatus. *Biological Cybernetics*, 82:499–515, 2000.

[19] Q.-G. Fu, D. Flament, J. D. Coltz, and T. J. Ebner. Temporal encoding of movement kinematics. *Journal of Neurophysiology*, 73:836–854, 1995.

[20] Q.-G. Fu, Suarez J. I., and T. J. Ebner. Neuronal specification of direction and distance during reaching movements in the superior precentral premotor area and primary motor cortex of monkeys. *Journal of Neurophysiology*, 70:2097–2116, 1993.

[21] A. P. Georgopoulos. Neural integration of movement: Role of motor cortex in reaching. *FASEB Journal*, 2:2849–2857, 1988.

[22] A .P. Georgopoulos, R. E. Kettner, and A. B. Schwartz. Primate motor cortex and free arm movements to visual targets in three-dimensional space II. coding of the directions of movements by a neuronal population. *Journal of Neuroscience*, 8:2928–2937, 1988.

[23] A .P. Georgopoulos, M. Taira, and A. V. Lukashin. Cognitive neurophysiology of the motor cortex. *Science*, 260:47–52, 1993.

[24] H. Gomi and M. Kawato. Equilibrium-point control hypothesis examined by measured arm stiffness during multi-joint movement. *Science*, 272:117–120, 1996.

[25] H. Gomi and M. Kawato. Human arm stiffness and equilibrium-point trajectory during multi-joint movement. *Biological Cybernetics*, 76:163–171, 1997.

[26] G. L. Gottlieb. Rejecting the equilibrium-point hypothesis. *Motor Control*, 2:10–12, 1998.

[27] G. L. Gottlieb, D. M. Corcos, and G. C. Agarwal. Organizing principles for single-joint movements. I. speed-insensitive strategy. *Journal of Neurophysiology*, 62:342–357, 1989.

[28] G. L. Gottlieb, D. M. Corcos, and G. C. Agarwal. Organizing principles for single-joint movements. II. speed-sensitive strategy. *Journal of Neurophysiology*, 62:358–368, 1989.

[29] P .L. Gribble, D. J. Ostry, V. Sanguineti, and R. Laboissiere. Are complex control signals required for human arm movement? *Journal of Neurophysiology*, 79:1409–1429, 1998.

[30] J. M. Hollerbach and T. Flash. Dynamic interactions between limb segments during planar arm movement. *Biological Cybernetics*, 44:67–77, 1982.

[31] H. Imamizu, Y. Uno, and M. Kawato. Internal representations of the motor apparatus: Implications from generalizations in visumotor learning. *Journal of Experimental Psychology: Human Perception and Performance*, 21(5):1174–1198, 1995.

[32] J. F. Kalaska, R. Caminiti, and A. P. Georgopoulos. Cortical mechanisms related to the direction of two-dimensional arm movements: Relations in parietal area 5 and comparison with motor cortex. *Experimental Brain Research*, 51:247–260, 1983.

[33] E. R. Kandel, J. H. Schwartz, and T. M. Jessell. *Principles of Neural Science.* McGraw-Hill, New York, fourth edition, 2000.

[34] A. J. Koivo. *Fundamentals for Control of Robotic Manipulators*, pages 160–163. John Wiley and Sons, New York, 1989.

[35] M. L. Latash. *Control of Human Movement.* Human Kinetics Publishers, Champaign, Illinois, 1993.

[36] S. G. Massaquoi. *Modelling the function of the cerebellum in scheduled linear servo control of simple horizontal planar arm movements.* PhD dissertation, Massachusetts Institute of Technology, Laboratory of Information and Decision Systems, June-August 1999.

[37] E. M. Maynard, N. G. Hatsopoulos, C. L. Ojakangas, B. D. Acuna, J. N. Sanes, Normann R. A., and J. P. Donoghue. Neuronal interactions improve cortical population coding of movement direction. *Journal of Neuroscience*, 19(18):8083–8093, 1999.

[38] J. McIntyre and E. Bizzi. Servo hypotheses for the biological control of movement. *Journal of Motor Behavior*, 25(3):193–202, 1993.

[39] D. W. Moran and A. B. Schwartz. Motor cortical representation of speed and direction during reaching. *Journal of Neurophysiology*, 82:2676–2692, 1999.

[40] F. A. Mussa-Ivaldi, N. Hogan, and E. Bizzi. Neural, mechanical, and geometric factors subserving arm posture in humans. *Journal of Neuroscience*, 5(10):2732–2743, October 1985.

[41] F.A. Mussa-Ivaldi. Do neurons in the motor cortex encode movement direction? an alternative hypothesis. *Neuroscience Letters*, 91:106–111, 1988.

[42] P. Pigeon, L. Yahia, and A. G. Feldman. Moment arms and lengths of human upper limb muscles as functions of joint angles. *Journal of Biomechanics*, 29(10):1365–1370, 1996.

[43] R. Porter. Internal organization of the motor cortex for input-output arrangements. In J. M. Brookhart, V. B. Mountcastle, and V. Brooks, editors, *Handbook of Physiology. Section 1: The Nervous System*, volume II: Motor Control, chapter 22, pages 1063–1081. American Physiological Society, Bethesda, Maryland, 1981.

[44] M. J .L. Prud'Homme and J. F. Kalaska. Proprioceptive activity in primate primary somatosensory cortex during active arm reaching movements. *Journal of Neurophysiology*, 72:2280–2301, 1994.

[45] T. D. Sanger. Probability density estimation for the interpretation of neural population codes. *Journal of Neurophysiology*, 76:2790–2793, 1996.

[46] A. B. Schwartz, R. E. Kettner, and A .P. Georgopoulos. Primate motor cortex and free arm movements to visual targets in three-dimensional space I. relations between single cell discharge and direction of movement. *Journal of Neuroscience*, 8(8):2913–2927, 1988.

[47] D. L. Sparks, W. B. Kristan Jr., and B. K. Shaw. The role of population coding in the control of movement. In P.S.G. Stein, S. Grillner, and D.G. Stuart, editors, *Neurons, Networks, and Motor Behavior*, pages 21–32. The MIT Press, Cambridge, Massachusetts, 1997.

[48] N. St-Onge, S. V. Adamovich, and A. G. Feldman. Control processes underlying elbow flexion movements may be independent of kinematic and electromyographic patterns: Experimental study and modeling. *Neuroscience*, 79(1):295–316, 1997.

[49] S. Tanaka. Numerical study of coding of the movement direction by a population in the motor cortex. *Biological Cybernetics*, 71:503–510, 1994.

[50] Y. Uno, M. Kawato, and R. Suzuki. Formation and control of optimal trajectory in human multijoint arm movement. *Biological Cybernetics*, 61:89–101, 1989.

[51] J. M. Winters. An improved muscle reflex actuator for use in large scale neuro-musculoskeletal models. *Annals of Biomedical Engineering*, 23(4):357–74, July-August 1995.

[52] J. M. Winters and L. Stark. Muscle models: What is gained and what is lost by varying muscle complexity. *Biological Cybernetics*, 55:403–420, 1987.

[53] F. E. Zajac. Muscle and tendon: Properties, models, scaling, and application to biomechanics and motor control. *CRC Critical Reviews in Biomedical Engineering*, 17:359–415, 1989.