EVOLUTIONARY TRAINING OF A BIOLOGICALLY PLAUSIBLE SPINO-NEUROMUSCULAR SYSTEM MODEL

A Dissertation

Presented in Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

with a

Major in Computer Science

in the

College of Graduate Studies

University of Idaho

by

Stanley Phillips Gotshall

August 2007

Major Professor: Terence Soule, Ph.D.

AUTHORIZATION TO SUBMIT DISSERTATION

This Dissertation of Stanley Phillips Gotshall, submitted for the degree of Doctor of Philosophy with a major in Computer Science and titled "Evolutionary Training of a Biologically Plausible Spino-neuromuscular System Model," has been reviewed in final form. Permission, as indicated by the signatures and dates given below, is now granted to submit final copies to the College of Graduate Studies for approval.

Major Professor _____ Date _____
Dr. Terence Soule

Committee Member _____ Date _____
Dr. Richard Wells

Committee Member _____ Date _____
Dr. Robert Rinker

Committee Member _____ Date _____
Dr. Michael O'Rourke

Department Administrator _____ Date _____
Dr. Mark Manwaring

Discipline's College Dean _____ Date _____
Dr. Aicha Elshabini

Final Approval and Acceptance by the College of Graduate Studies

_____ Date _____
Dr. Margrit von Braun

ABSTRACT

There is an increasing need for researchers to develop a greater understanding of the neuromuscular system. The medical treatment of many diseases and disorders depends on physicians and practitioners having specific knowledge of how damage to certain spinal pathways can affect motor control. To that end, an important step in increasing our understanding of the spino-neuromuscular system (SNMS) is to develop a model in which researchers can conduct controlled virtual experiments within the spinal cord. This dissertation develops such a model while addressing limitations in current modeling methods of neuromuscular systems. This dissertation also shows that evolutionary algorithms train robust and stable SNMS models that yield key biological behaviors. This type of model is widely applicable in areas such as evolutionary robotics, neuroprosthetics, and modeling neuromuscular diseases since all these areas investigate the importance of specific components in biological or biologically related systems.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION AND MOTIVATION

Although the general functionality and major signal pathways of the SNMS (Spino-neuromuscular System) are well understood, the details of the control processes mediated by the SNMS are still largely a mystery due to the difficulties of performing carefully controlled experiments at the level of the individual neurons and neural pathways. Similarly, although the general short term compensatory and long term adaptive and rehabilitative responses of the SNMS to injury are understood, the role of specific pathways, neuron types, and pathway dynamics are not well understood. Such detailed knowledge of the Spino-neuromuscular System (SNMS) would have significant advantages in the diagnosis of diseases and treatment of injury.

Our long term goal is to develop a model that is sufficiently biological to test hypotheses regarding the function of the SNMS. In particular, we will address how to train the parameters of the model to identify emerging biologically plausible behaviors. A model that behaves biologically on multiple levels: gross anatomical movements, specific pathway behaviors, and individual neuron behaviors, will make it possible to perform carefully controlled, virtual experiments to test hypotheses regarding a spinal injuries and diseases.

To accomplish this goal this dissertation applies stochastic learning algorithms to train a biologically plausible model of the human arm. The primary goal of this dissertation is to show that constructing a biologically realistic model is possible by using current knowledge of the anatomy of the SNMS plus learning algorithms to fill in the unknown details, e.g. the strengths of individual synaptic connections, and the strengths of the individual muscle fibers. An equally important goal is to make the model sufficiently biological that biologically realistic behaviors (e.g. realistic arm

motion, muscle excitation patterns, and motor unit recruitment) emerge from trained instances of the model. To confirm our success we incrementally increase the model's complexity and compare the results to data generated from human subjects and show that key biological behaviors emerge. Finally we show that trained instances of the model are also stable when used with untrained inputs and unfamiliar tasks. Evolving stable systems and observing the emergence of biological behaviors are particularly important in this research because of the potential applications in nonlinear control system design and investigating properties of neuromuscular diseases and injuries.

The main contributions of this dissertation are threefold. First, this research shows that stochastic optimization techniques (GAs/PSOs) are effective methods for training controlled motion in a biologically based neuromuscular system model. The training techniques used in this dissertation are not tailored towards biologically based systems. Thus, it is reasonable to expect that these techniques will effectively train pulsed control systems with similar components. Second, this research demonstrates that training a sufficiently biological neuromuscular model with stochastic optimization techniques produces key biological behaviors. The emergence of these behaviors paves the way for further research in simulating the behavior of neuromuscular disorders and other biological phenomena. The third main contribution of this research is demonstrating that the training process produces stable and robust systems that can carry out untrained tasks.

Chapter 2

SCIENTIFIC BACKGROUND

The following subsections describe the fundamental components of the spino-neuromuscular system and training algorithms used in this dissertation. The biological components, muscle fibers, neurons, and other components of the spinal cord each directly corresponds to an element in model used in simulation in Chapters 4, 5, and 6. Similarly, the subsections describing genetic algorithms (GA)s, particle swarm optimizers (PSO)s, and breeding swarm optimizers (BSO)s summarize the three main training algorithms used in these chapters as well.

2.1 Muscle Fibers and Motor Units

Muscle fibers are the elemental units of skeletal muscles. These fibers contract upon stimulation from spinal motor neurons and are categorized according to their strength/fatigability. The weakest and slowest muscle fiber type, type I fibers, respond very slowly to action potentials but can maintain their tension for a relatively long time. Type II fibers on the other hand respond quickly to action potentials and are categorized into two subgroups, type IIA and type IIB. Type IIA fibers, also known as fast fatigue-resistant fibers, combine fast twitch behavior with contractile force which is sustainable for several minutes. Type IIB fibers generate approximately 100 times the force of the type I fibers, but fatigue quickly and take hours to recover fully [34].

Muscle fibers which generate contractile force for the muscle, also called extrafusal muscle fibers, are the types mentioned above; however another class of muscle fiber has a different purpose. Modified fibers, called intrafusal fibers, are embedded with the extrafusal fibers but do not generate contractile force. These fibers make up what is called the muscle spindle. The nervous system uses the muscle spindle as part of the feedback mechanism to monitor the state of muscles. Intrafusal fibers encode length and

contractile velocity information and relay it back to the spinal cord. This creates a closed loop system which enables the spinal cord to generate smooth precise movements in skeletal muscles.

2.2 Neurons

The entire nervous system is made up of cells which process chemically encoded information and maintain the structure and health of nervous tissue. The information processing cells are commonly known as neurons, or nerve cells. A typical neuron processes electrical signals via an input/output system composed of three parts: dendrites, the cell body, and the axon. Axons and dendrites are thin tubule processes which connect to the cell body. Dendrites respond to signals sent by axons of other neurons, so dendrites act as an input mechanism to the neuron. Over time, as the dendrites of a neuron receive repeated electrical signals, the electric potential across the cell body increases and the decreases depending on the input signal patterns. When the neuron reaches a threshold potential level, a chain reaction begins which forces the potential very high for a short time (depolarization) and then quickly returns to slightly below the resting potential (hyperpolarization). The axon then carries this pulse, called an action potential, to the dendrites of neighboring neurons. Hodgkin and Huxley [29] were the first scientists to directly observe this phenomena in the giant axon of the squid.

2.3 Spinal Cord Organization

The brain and the spinal cord work together via electrochemical signals to control all voluntary and involuntary muscles in the body. In particular, skeletal muscles are controlled via signals from the motor cortex which travel down white mater columns to a given cross section in the spinal cord [34,66]. Cross sections control and receive feedback signals from skeletal muscles via neurons in the gray matter regions. The ventral region of the gray matter, known as the ventral horn, contains motor neurons which directly innervate skeletal muscles. These neurons trigger electrical impulses in muscle fibers causing them to contract. The general purpose of muscle fiber contraction

is to shorten the overall length of muscles to move limbs. The spinal gray matter, however, consists of several specialized types of cell types categorized by function.

2.3.1 Muscle Fibers and Motor Neurons

The single most important nerve cell with respect to motor control in the spinal cord is the alpha-motoneuron (α-MN) [67]. A single α-MN innervates many extrafusal fibers so when an α-MN fires every innervated muscle fiber contracts, hence the definition of "motor unit" [34]. Alpha-MNs vary in size depending on the number of fibers they innervate. Larger motor neurons typically innervate a large number of fatigue resistant fibers whereas small motor neurons innervate small numbers of weak fatigue-resistant fibers.

Another class of motor neuron, the gamma-MN (γ-MN), innervates intrafusal muscles fibers. This activity adjusts the sensitivity of the muscle spindle and influences how the spinal neurons respond to changes in the muscle. Sustained γ-MN activity increases the sensitivity of the muscle spindle and causes it to respond significantly to small changes in length and contractile velocity. Low γ-MN activity causes the spindle to only sense large changes in length and contractile velocity. Gamma-motoneurons are generally activated by descending signals in parallel with α-MNs (α-γ coactivation) which generally provides the correct amount of activation for the spindle to sense changes in the muscle [34]. The information encoded by the spindle is then relayed back to the spinal cord via afferent neural pathways.

2.3.2 Afferent Pathways

Muscle spindles, and other mechanisms, encode length, velocity, and tension information in the muscle and this information is sent to the spinal cord via sensory afferent neurons. Each class of afferent encodes specific types of information. The three most discussed afferents are the group Ia (primary), group II (secondary), and Ib which responds to the Golgi tendon organ. The primary afferents sense changes in the muscle's length and velocity. The secondary afferents however respond only to length and ignore the rate of contraction.

2.3.3 Other Neurons

Motor neurons directly stimulate muscle fibers, but many other interneurons also influence the behavior of these cells. Interneurons are neurons in which signals do not terminate nor originate. Motor control signals originate in the upper motor neurons in motor cortex and brain stem and terminate in innervating motor neurons. Essentially, any other neuron that influences the propagation of an neural signal is an interneuron.

Like any other neuron, an interneuron either excites or inhibits its post synaptic neurons and is thus called an excitatory or inhibitory interneuron, respectively. Both types of interneurons facilitate communication among fibers of a single muscle as well as communication between muscles in agonist/antagonist muscle pairs. Interneurons often mediate signals between afferent structures, such as group II afferents, and motor neurons. Renshaw cells, although not directly associated with afferent pathways, also play an important role as inhibitory interneurons. Alpha-MNs synapse on these cells so they can reciprocally inhibit the synapsing α-MN in order to modulate their firing patterns to prevent over-contraction in muscles.

2.4 Training Algorithms

Genetic algorithms, particle swarm optimizers, and breeding swarm optimizers are general purpose problem solving algorithms. Each algorithm maintains a population of individuals where each individual encodes a potential solution to given problem. Populations are typically generated randomly and during a given timestep each individual receives a fitness value based on how well it solves the problem. Algorithm specific operators then manipulate the individuals in the current population to generate a new population in the hopes that more fit individuals emerge. The algorithm repeats this process for a fixed number of generations or until some criteria is satisfied, such as an individual achieving a given fitness value.

2.4.1 Genetic Algorithms

Genetic algorithms (GA)s are based on evolutionary concepts of natural selection and recombination [12]. During each iteration (generation), genetic operators corresponding to biological mutation and recombination take the information encoded in selected parent individuals to generate offspring. Recombination, also called crossover, is the primary mechanism for exploration as it enables the population to make large jumps through the search space. Mutation, however, exploits locally optimal solutions in order to make small changes for small increases in fitness. Typically, this process is repeated until the new population contains the same number of individuals as the previous population.

2.4.2 Particle Swarm Optimizers

Particle swarm optimizers (PSO)s also use a population based approach, but the population traverses the search space in a way similar to that of a swarm of bees searching for pollen [38]. At any given time, each individual (particle) in the population is traveling in a direction and speed given by its current n-dimensional velocity vector. The position of the particle at the next time step is simply the current position added to the current velocity. The velocity of each particle is influenced by the personal best position seen by that particle and the best position the entire population has seen. A given particle tends to travel back to these two points as well as explore new parts of the space. Random elements in the update equations keep the population from converging prematurely by maintaining diversity. The population eventually swarms tightly around a single point which can be taken as the final solution.

2.4.3 Breeding Swarm Optimizers

Both the genetic algorithm and particle swarm optimizer have desirable features in regard to how they search for solutions. Settles et al. developed the breeding swarm optimizer (BSO) which combines recombination and mutation of GAs with the PSO [58]. In the BSO algorithm, a small number of particles in the swarm are chosen for breeding

at a given generation, and their offspring are propelled away from where the parents are traveling. This mechanism appears to increase diversity and generate better solutions.

Chapter 3

INTRODUCTION TO PAPERS

This dissertation presents a spino-neuromuscular (SNMS) modeling project that began in the fall of 2003. The initial goal was to produce a simple proof of concept model and then to progressively make the model more biological. This project began a relatively simple system of 51 neurons and 3 muscle fibers controlling an arm actuated by single muscle. Over the last four years it has evolved into a system of 180 neurons with over 500 synaptic links to control an arm actuated by a biceps/triceps pair of 12 muscle fibers. Over time, these enhancements significantly changed fundamental properties of the model and produced increasingly realistic behaviors.

In the beginning our goal was to train the system with a GA for a simple controlled behavior. At this stage we aimed only to train the system to move the arm upward and downward at a constant rate. We were successfully able to train the system to raise the forearm completely and lower it to its resting angle. We then made our first biological enhancement by adding Renshaw cells to the model, which in biology plays an important role in modulating the behavior of α-MNs. This addition made the model significantly easier to train and produced more controlled motions. These results were published and presented at the International Joint Conference on Neural Networks 2005 in Montreal [23].

The next goal was to introduce more biological components, i.e. the triceps, afferent feedback, and more spinal neurons, into the model to train more biologically realistic behaviors. Preliminary results indicated that the training algorithms take advantage of the newly added afferent feedback to produce more dynamic motor unit behaviors described in Chapter A and published in Boston at the International Conference on Cognitive and Neural Systems. Further experimentation showed that key biological behaviors emerge that were not explicitly trained including, tonic tension during resting

phases of motion, biceps/triceps coactivation, and recruitment-like α-MN firing patterns, described in Chapter 4 [22]. These results led to the exploration of potential medical applications relating to muscle injury, also described in Chapter 4.

The next goal was to shift our perspective of the project by treating it like a robotics control systems problem. In particular, this goal focused on the application of pulsed neural networks as a signal processing tool for nonlinear control systems. We examined the inherent limitations of traditional connectionist neural networks from a biological perspective and emphasize the usefulness of stochastic algorithms in training control systems. The results showed that each of the several algorithms used for succeeded in training the target motion, with varying degrees of success, and that each algorithm generated the same types of behaviors. This is described in Chapter 5, and published and presented at the Genetic and Evolutionary Conference 2007 in London [24].

Our final goal continues to view the SNMS model as a nonlinear control system by examining its ability to respond appropriately to unfamiliar inputs and conditions. The human neuromuscular system is able to adapt and adjust to new situations, thus training the model with a sufficient number of tasks may result in a reasonably robust and stable system. Chapter 6 investigates the behavior of the SNMS with respect to varying the mass and velocity of the forearm. The results show that training a given SNMS for multiple tasks results in a system that responds in a smooth and controlled fashion when controlled with unfamiliar input frequencies and tasks.

Chapter 4

STOCHASTIC OPTIMIZATION OF A BIOLOGICALLY PLAUSIBLE SPINO-NEUROMUSCULAR SYSTEM MODEL: A COMPARISON WITH HUMAN SUBJECTS [1]

The first goal of this project was to determine if the SNMS model produces key biological behaviors relating to motion. This lays the groundwork for conducting controlled virtual experiments relating to medical applications because it demonstrates that the model behaves biologically. In order to identify key biological behaviors this chapter compares simulated neuromuscular in the model with the neuromuscular activity of human subjects.

4.1 Abstract

Simulations and modeling techniques are becoming increasingly important in understanding the behavior of biological systems. Detailed models help researchers answer questions in diverse areas such as the behavior of bacteria and viruses and aiding in the diagnosis and treatment of injuries and diseases. However, to yield meaningful biological behavior, biological simulations often include hundreds of parameters that correspond to biological components and characteristics.

This paper demonstrates the effectiveness of genetic algorithms (GA) and particle swarm optimizer (PSO) based techniques in training biologically plausible behavior in a neuromuscular simulation of a biceps/triceps pair. The results are compared to human subjects during flexion/extension movements to show that these algorithms are effective in training biologically plausible behaviors on both neural and gross anatomical levels.

---

[1] THIS PAPER HAS BEEN ACCEPTED FOR PUBLICATION IN GENETIC PROGRAMMING AND EVOLVABLE MACHINES. AUTHORS: STANLEY GOTSHALL, KATHY BROWDER, JESSICA SAMPSON, TERENCE SOULE, AND RICHARD WELLS

Specific behaviors of interest that emerge include tonic tensions in both muscles during resting periods, biceps/triceps coactivation patterns, and recruitment-like behaviors. These are all fundamental characteristics of biological motor control and emerge without direct selection for these behaviors. This is the first time that all of these characteristic behaviors in a model emerge without direct selective pressure.

4.2 Introduction and Motivation

Although the general functionality and major signal pathways of the spino-neuromuscular system (SNMS) are well understood, the details of the control processes mediated by the SNMS are still largely a mystery due to the difficulties of performing carefully controlled experiments at the level of the individual neurons or neuron types. Similarly, although the general short term compensatory and long term adaptive and rehabilitative responses of the SNMS to injury are understood, the role of specific pathways, neuron types, and pathway dynamics are not well understood. Such detailed knowledge of the SNMS would have significant advantages in the diagnosis of diseases and treatment of injury.

Our long term goal is to develop a model that is sufficiently accurate to test hypotheses regarding the function of the SNMS. In particular, we will target two medically significant areas: the role(s) of segmental sensory and motor components in resolving redundancy in controlling normal movement and the role of the SNMS in injury compensation and rehabilitation. A model that behaves biologically on multiple levels (e.g. gross anatomical movements, specific pathway behaviors, and individual neuron behaviors) will make it possible to perform carefully controlled, virtual experiments in these two areas.

In this paper we apply three stochastic learning algorithms to train a biologically plausible model of the human arm. Our goal is to show that constructing a biologically realistic model is possible by using current knowledge of the anatomy of the SNMS and with learning algorithms to fill in unknown details, e.g. the strengths of individual synaptic connections and the strengths of the individual muscle fibers. An equally important goal is to make the model sufficiently biological that biologically realistic

behaviors (e.g. realistic arm motion, muscle excitation patterns, and motor unit recruitment) emerge from trained instances of the model. To confirm our success we compare our results to data generated from human subjects and show that key behaviors emerge: tonic tensions in both muscles during resting periods, biceps/triceps coactivation patterns, and recruitment-like behaviors.

4.3 Biomedical Background

Electrochemical signals exchanged between the brain and the spinal cord work together to control all voluntary and involuntary muscle systems in the body. Signals that initiate volitional movements originate in the premotor and primary motor cortices in the brain and descend to and synapse on cells in a given cross sectional region of the spinal cord's ventral horn [34]. The spinal cord itself is divided into 31 lateral pairs of nerves which span nearly the entire body. At the spinal cord, the nerves are arranged laterally in a gradient fashion with respect to which region of the body they control and receive sensory information from. Roughly speaking, the upper spinal nerves and lower spinal nerves control muscles and gather sensory information from the upper and lower body, respectively.

Signals from the brain descend to a spinal region to control a given group of muscle fibers. Each cross section of the spinal cord consists of several neuron types that are classified by their function. The most commonly known neuron type is the alpha-motoneuron (α-MN) [34] (Figure 4.2). Alpha-MNs directly innervate muscle fibers causing muscle contractions. Muscle fibers generate contractile force during physical activity as well as involuntary motions. A single α-MN along with all the muscle fibers it innervates is called a motor unit. The less studied gamma-motoneuron (γ-MN) has a similar function as the α-MN. However these neurons innervate modified muscle fibers that do not directly contribute to contractile muscle force. The purpose of these fibers is purely sensory since they stretch and contract along with the normal muscle fibers. A third key neuron type is the Renshaw cell. Unlike the motor neurons, the Renshaw cell is an inhibitory interneuron; it directly inhibits the α-MNs. It is generally believed that the Renshaw cell is a mechanism to help modulate and control α-MN firing patterns to

produce smooth motions [7].

Spinal motor neurons (e.g. α-MNs and γ-MNs) innervate skeletal muscle fibers, but these fibers have different functions with respect to motor control. On one hand, the fibers innervated by the α-MNs are called extrafusal fibers and are directly responsible for muscle contraction. That is, when a given α-MN fires, all extrafusal muscle fibers in that motor unit contract which contributes to the overall shortening of the muscle. However, the γ-MNs innervate modified muscle fibers called intrafusal fibers [34]. The shortening and lengthening of these fibers does not directly contribute to the overall contraction of the muscle. Instead, these fibers act as sensors which are sensitive to changes in the muscle's length. Gamma-MN activity is responsible for normalizing the length of the intrafusal fibers with respect to the extrafusal fibers to reflect the current length of the muscle. This information is encoded in the intrafusal fibers and relayed back to the spinal cord via direct and indirect afferent pathways. Information is sent back to the spinal cord via sensory nerve fibers wrapped around the intrafusal muscle fibers. The intrafusal fibers and these sensory nerve fibers are contained within structures called muscle spindles. A typical muscle spindle consists of 3-12 intrafusal fibers along with the afferent nerve endings [54]. Fibers in the spindle encode information about the current length and contractile velocity and transmit this information back to the α-MNs and other spinal neurons via specific afferent pathways. The two most studied are the group Ia and group II afferent pathways, commonly called the primary and secondary afferents, respectively. The Ia afferent pathway also synapses on an interneuron that inhibits the antagonist, the Ia inhibitory interneuron (Figure 4.2).

4.4 Previous Work

A key aspect of modeling systems that simulate neural interactions with muscles is to use biologically plausible models of muscles. These models usually exhibit nonlinearities to reflect the stretching and complex behaviors of muscle fibers. Skeletal muscles in biological models are commonly modeled with single elastic elements. Although certain models include several distinct muscles in complex configurations [4,49], a given muscle consists of many muscle fibers innervated by many

motor neurons. A given motor axon typically innervates many muscle fibers in a given muscle. Therefore, biologically plausible neuromuscular simulations should allow for complex behavior with multiple motor units.

Neuromuscular models that describe the interactions between the spinal cord and skeletal muscles also include mathematical representations of either the individual neurons or of neuron groups. These models usually include mathematical representations of the individual neurons or neuron groups. Interactions among these neurons and the neural network inputs determine how the network's output affects the muscle. In modeling the dynamics of biological systems, practitioners and researchers often use connectionist-style neurons to account for synaptic strengths and connections [32, 40, 48, 49]. While these neuron types are suitable for Kohonen, pattern matching, and classification style networks, they lack the ability to model a time-dependant dimension present in biological neural networks. Specifically, these types of neurons can model the frequency of firing rates from one neuron to another, but cannot model *when* a given neuron fires. These neurons do not retain state information corresponding to the voltage of biological cell membranes. The static nature of this neuron type limits the ability of networks of these neurons to retain phase and timing information with respect to neuron firing. Thus, biologically plausible neuromuscular simulations can be improved by including time-dependant neuron models that retain phase and state information present in biological neurons.

After designing the neuromuscular model and implementing it in simulations, or building it mechanically, the parameters of the model must be tuned to achieve some desired or meaningful behavior. In practice, these parameters are sometimes chosen randomly or tuned by hand [9, 31]. Typically, this kind of non-automated training makes generalizing the model for different functions difficult because a single change to the model could require retuning of the entire network. Therefore, these types of models are commonly trained via supervised learning [32]. Non-automated training also incorporates user bias into the model, making it difficult to test hypotheses objectively.

Our goal is to make a generalizable SNMS model that addresses these limitations and that can answer questions regarding the importance of specific spinal pathways in motor

control. This requires a rapid and bias-free training mechanism and an appropriate neuromuscular model. Automated biologically plausible models are vital to improving our understanding of spinal pathways and how alterations in these pathways affect motor behavior. Thus, a primary goal of this research is to show that stochastic algorithms can train a flexible and biologically plausible SNMS model and that this training leads to biologically meaningful/plausible behavior. We confirm our success by observing biologically plausible behavior in trained systems and similarities between our simulations and data recorded from human test subjects.

4.5 Models

In this paper we use the same basic model as used in our previous research. Each of the following sections describe key components and training algorithms for the SNMS model. The arm consists of two groups of muscle fibers corresponding to the biceps and triceps. Each muscle fiber is controlled by semi-independent neural subnets. Information about the state of the muscle fibers is fed back into its muscle fiber group's network to adapt to changes in the system. The timestep for all model components is 0.01 seconds.

4.5.1 Joint Model

The joint model used in our experiments (Figure 4.1) has a single frictionless degree of freedom. The motion of the forearm is dependant on contractions of both muscles and the force of gravity. Furthermore, rotational springs are encountered at π-2.4 and π-0.4 radians to model resistance to over-contraction and hyperextension, respectively, in the human arm. The connections of the muscles are shown as I_1 and I_2 in Figure 4.1 and the movable section of the limb is modeled as a uniform cylinder. The parameters of the joint used for training are shown in Table 4.1.

4.5.2 Muscle Model

The Hill model [2,15,28] is commonly used to simulate skeletal muscle systems and their nonlinearities. We use the second canonical, functionally equivalent, form of the

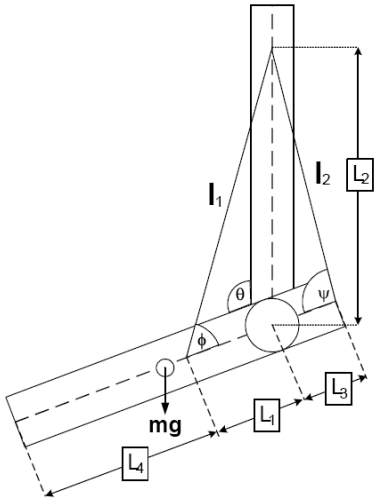


Figure 4.1: The joint has a movable forearm with center of mass indicated by mg. The parameters of the joint model are listed in Table 4.1.

Table 4.1: Joint model parameters

Forearm Radius	0.2 meters
Movable Limb Mass	0.5, 0.6, and 0.7 kg
Timestep	0.01 seconds
L1	0.08 meters
L2	0.8 meters
L3	0.08 meters
L4	0.72 meters
Spring Constant	20
Spring Damping Constant	10

Hill model because this form is easiest to apply to multiple muscle fibers acting in parallel with one another [43]. Our implementation of each muscle has six independently triggerable muscle fibers connected to a common tendon. Each muscle fiber receives a binary signal from its alpha-motoneuron (α-MN) and contracts according to the Hill equations. A muscle fiber contracts during a given timestep if its α-MN is active and relaxes otherwise. The tendon component of the model stretches and shortens depending on the activity of each muscle fiber. The muscles connect to the upper portion of the arm and to the forearm near the joint as indicated in Figure 4.1.

The Hill model is organized into four mechanical components: the 1) contractile element, 2) damping element, 3) serial elastic components (SEC)s, and the 4) parallel elastic component (PEC). This canonical form of the Hill model used here depicts the muscle fibers as the "serial" elastic component and the tendon as the "parallel" elastic component. This model also generalizes to represent many motor units where a motor unit is comprised of a single motoneuron and all the muscle fibers it innervates[2]. The SEC models the elastic properties of muscle fibers in a motor unit while the PEC models the elasticity of a single tendon. The contractile element models the contraction and relaxation of muscle fibers and the damping element models the coupling between the rate of muscle contraction and fiber tension.

4.5.3 Neuron Model: Integrate-and-Fire

Each neuron in the network is an integrate-and-fire (IF) spiking neuron with time constant τ_0=0.05 seconds [8,21,59]. Over time an IF neuron receives input signals from presynaptic neurons which increase the neuron's potential. In the absence of input, the neuron's potential decays exponentially. However, if a volley of input signals causes the neuron's potential to exceed its threshold, the neuron fires and sends a signal to all of its postsynaptic neurons. After a neuron fires it enters a refractory period of a single timestep (0.01 ms in these experiments) during which it cannot build up any potential. This is commonly referred to as the absolute refractory period [34]. With this refractory

[2]In this paper, each group consisting of one contractile element, one damping element, and one serial elastic component will be called a motor unit where an entire muscle consists of six motor units. Biologically, a motor unit also includes the innervating α-MN.

period, each of our simulated neurons can have a maximum frequency of 50 Hz, which is within the range of biological firing rates for regular spiking neurons [60]. After the refractory period, the neuron can build potential and fire again. The IF neuron is modeled with

$$s(t+\Delta t) = \begin{cases} (1-\frac{\Delta t}{\tau_0})s(t) + \frac{\Delta t}{\tau_0}\sum_n \omega_n x_n & , s(t) < 1 \\ 0 & , s(t) \geq 1 \end{cases} \tag{4.1}$$

and

$$y(t) = \begin{cases} 1 & , s(t) \geq 1 \\ 0 & , s(t) < 1 \end{cases} \tag{4.2}$$

where the functions $s(t)$ and $y(t)$ are the neuron's electric potential and output, respectively, at time t. The summation in Equation 4.1 represents the sum of synaptic inputs multiplied by their respective weights where n is the number of inputs,w_n is the n^{th} synaptic weight, and x_n is the n^{th} input value.

4.5.4 Neural Network

Our neural network model is designed to incorporate what are believed to be the key control pathways in the human SNMS, including the group Ia and group II afferent pathways, the Ia inhibitory interneurons, and Renshaw cell inhibition (Figure 4.2). Each individual subnet (Figure 4.2) consists of 15 IF neurons. The system's neural network consists of 180 IF neurons with over 500 synaptic connections and is composed of twelve subnets, one for each of the twelve muscle fibers in the biceps and triceps. The subnets have identical topologies and receive independent descending inputs (Figure 4.2). The individual subnets contain chains of neuron clusters (synfire nodes) which model individual descending pathways to the spinal neurons. The agonist muscle refers to the muscle fiber innervated by the α-MN and antagonist refers to fibers in the opposing muscle. The input to a given subnet is implemented as three synchronous unipolar inputs.

In biology, it is difficult for a simple chain of neurons to propagate a signal because a

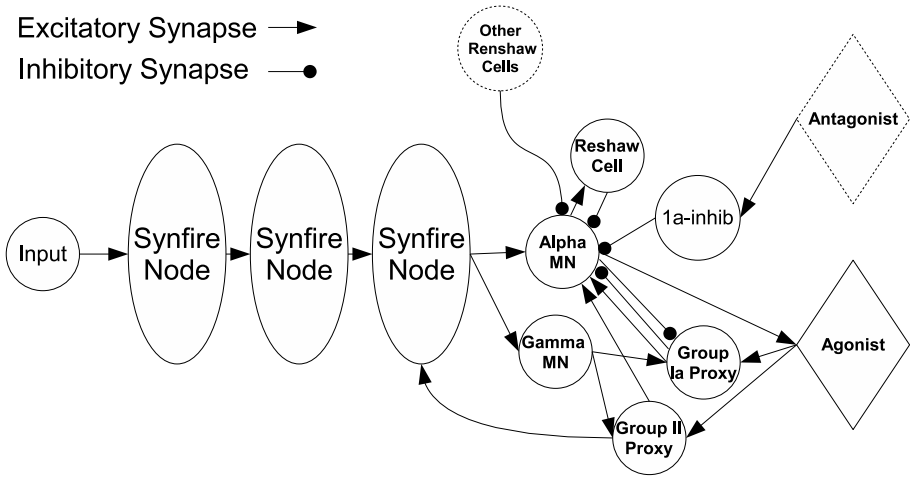


Figure 4.2: This figure shows a single neural subnet with descending neural pathways to the spinal neurons for one agonist motor unit. The group Ia and group II afferent structures are mathematical proxies for the overall effect of extrafusal and intrafusal fibers on the spindle nerve endings [34]. This subnet is repeated five times for the remaining agonist motor units and six times for the six antagonist motor units.

relatively high input frequency is needed for each neuron in order to fire. Even with a high input frequency to a simple chain, the firing frequency of each successive neuron decreases significantly unless the synaptic weights are arbitrarily large. It is more likely that these pathways are organized as groups of neurons where information is passed from one group to the next. This architecture is known as a synfire chain. Each descending node in Figure 4.2 is implemented as a synfire node [1,68,72], where each synfire node is a collection of three IF neurons that are fully connected to the preceding node and following node. The use of synfire nodes allows signals to propagate without frequency attenuation. It also allows for complex variations in descending signal patterns. Thus, the signal received by the α-MN may be very different from the input signal to its subnet.

In biology, after neuronal signals propagate from the brain and down the spinal cord, they synapse on the α-MN which activates skeletal muscle fibers. In the SNMS model, when activated, the α-MN sends a signal to muscle fibers causing them to contract and shorten which moves the associated joint. Furthermore, like many artificial nonlinear control systems, the spinal cord has multiple inhibitory feedback mechanisms, most prominently are the Renshaw cells. These cells are thought to modulate and control the α-MN's rate of discharge to produce controlled motions [7]. The connections of the α-MN and Renshaw cell are shown in Figure 4.2. Each Renshaw cell has inhibitory synaptic links to all six α-MNs in its muscle's network.

The spinal cord also has feedback systems to receive information from muscle fibers. The most prominent types of feedback in skeletal muscles are the primary (group Ia) and secondary (group II) afferents which relay the contractile velocity and length of muscle fibers, respectively, back to the spinal cord. In the model, the Ia proxy afferent neuron receives two real valued inputs which reflect the average change in length per timestep (i.e. velocity) of the six muscle fibers during the current timestep. One velocity input is active when the average muscle fiber length is decreasing and the other is active when it is increasing. These inputs, and both links to the alpha-MN, are separated to reflect the high and low frequencies of action potentials associated with muscle fiber contraction and relaxation. The model also includes a group II proxy afferent neuron in each subnet

that directly receives, as a real value, the average muscle fiber length at each timestep. This neuron sends its output only to the local α-MN and the last descending synfire node (Figure 4.2). Both afferent neurons' outgoing excitatory synaptic weights are allowed three times the magnitude of other neurons outgoing weights to have an effect comparable to the other excitatory inputs to the α-MN.

Alpha-MN activation causes direct contraction of normal muscle fibers, also called extrafusal fibers. The gamma-motoneuron (γ-MN), on the other hand, causes contraction of modified muscle fibers called intrafusal fibers. Contraction of these fibers does not contribute to the contractile force of the muscle but increases their sensitivity to changes in length as the entire muscle's length changes. In turn, the intrafusal fibers serve as the input to the afferent feedback neurons mentioned previously. The model also includes the γ-MN which receives descending signals in parallel with the α-MN [34] and excites the intrafusal fibers, making them more sensitive the changes in length. This models the phenomenon known as $\alpha - \gamma$ (alpha-gamma) coactivation. We simulate the sensitivity of excited intrafusal fibers by adding a synapse from the γ-MN to the proxy afferent neurons (Figure 4.2). Furthermore, the inhibitory connection from the α-MN to the Ia afferent neuron is a mathematical proxy for the muscle shortening that occurs just after the α-MN fires. After a contraction, the tension on the muscle fiber decreases momentarily lowering the afferent neuron's potential.

In addition to training hundreds of synaptic weights, training the system also requires adjusting 12 real values used to determine the strength of each contractile element(muscle fibers) and 12 binary values to determine which biceps motor units receive descending inputs during the upward (6) and downward (6) motion. We use a GA (Genetic Algorithm) to train the system because it worked well in previous experiments [23, 24] and it easily expands to accommodate future changes in the model. GAs are also well suited for training pulsed neural networks when the required output pattern is not known.

4.6 Training Algorithms

Each of the following sections describe key components and training algorithms for the SNMS model. The arm consists of two groups of muscle fibers corresponding to the biceps and triceps. Each muscle fiber is controlled by semi-independent neural subnets. Information about the state of the muscle fibers is fed back into its muscle fiber group's network to adapt to changes in the system. The timestep for all model components is 0.01 seconds.

In the following experiments we use GAs, PSOs, and a PSO variant, BSO, to train the SNMS model. Each of these techniques are population based algorithms in which each successive iteration of the algorithm yields a new population of individuals. This creation of the new population is driven by the fitness of each individual in the previous population (previous generation). Genetic algorithms are general purpose learning algorithms based on the evolutionary concepts of natural selection, recombination, and mutation whereas PSO and PSO-based algorithms are based on a metaphor of social interaction and swarm intelligence. Genetic algorithms traditionally have two components that contribute to their search strategy, recombination and mutation. Recombination is a mechanism for making relatively large jumps around the search space to find diverse solutions (exploration) and mutation is used to make small changes to individuals to achieve a small increase in fitness (exploitation).

With PSOs, on the other hand, the search process uses a physics-like simulation of particles moving in multidimensional space. Each particle has a position vector which represents a potential solution and also has a velocity vector. Each particle keeps a memory of the best location it has seen and also knows the best location the entire swarm has seen. These two "best" locations, at a given timestep, influence how the particle's velocity vector will change in the next timestep, and thus determine the next position of that particle. PSOs also usually have a mechanism to slow down the particles over time since initial velocities are usually set to relatively large random numbers. The BSO algorithm is largely based on PSO, but incorporates the evolutionary concepts of mutation and recombination to increase diversity in the population in the hopes of generating more fit solutions.

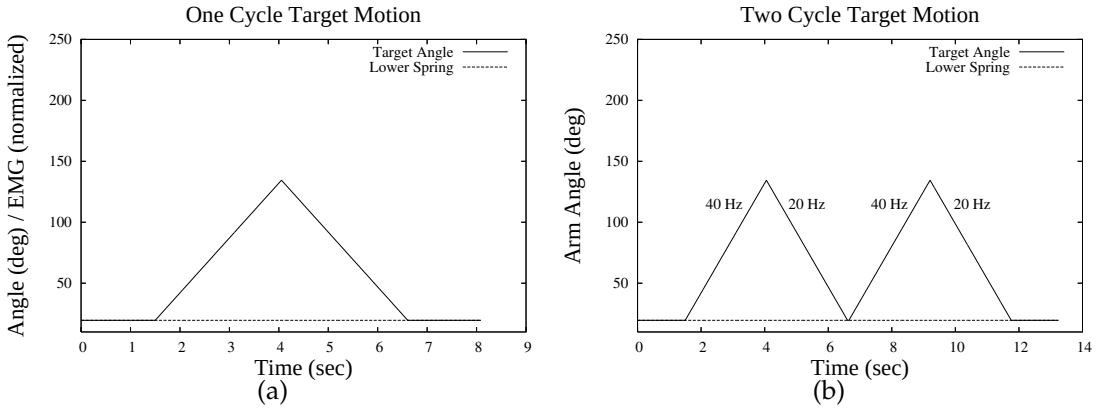


Figure 4.3: This shows the target motions for each of the training cases. Beginning at 0 seconds, no descending pathways receives an input pattern. At 1.5 seconds, an input pattern of 40 Hz is sent to a subset the biceps' subnets, as chosen by the training algorithm, and the signals across each input neuron is phase shifted evenly across a window of $\frac{1}{40Hz}$. At a given frequency, the inputs to the subnets are out of phase and do not overlap during a given timestep unless the frequency is sufficiently high. During the downward motion the biceps network receives a descending input of 20 Hz.

4.7 Training and Simulation Experiments

The following experiments are broken into two categories based on the target behavior, one-cycle and two-cycle. The one-cycle experiments are used for comparing algorithm performance and the two-cycle experiments are used for comparison with the human subjects. One-cycle and two-cycle simply means that the forearm is moved up and down once, or twice, respectively. All cycles have the same goal of training the system to move the forearm up and down at a constant rate of $45\frac{deg}{sec}$. The input frequency to signal the arm for upward and downward motion is 40 Hz and 20 Hz to the biceps, respectively. The triceps receives no descending inputs in any experiment, thus it responds purely to feedback pathways. The training cases are shown in Figure 4.3.

To model the readings of surface electrodes on human subjects we use a state variable in the Hill model, one for each motor unit, which approximates the electric potential across the neuromuscular end-plate. This variable changes when a given motor unit receives input from its α-MN. In our results, this value is multiplied by the corresponding force multiplier to reflect the relative impact of a motor unit on a surface

EMG reading. This is based on the assumption that the contraction of larger (stronger) fibers has a larger impact on EMG readings. All six scaled end-plate potential values are then averaged over a sliding average window of 300 ms.

All three algorithms train three parameter types: synaptic weights, the force multiplier for each motor unit, and two binary values for each motor unit which determine whether or not that motor unit's descending pathway will receive the input pattern for the upward or downward motion. In other words, the training algorithm selects the quantitative parameters of the model and chooses which motor units receive descending input during the upward and downward phases of movement.

Each algorithm evaluates fitness of potential solutions with the equation

$$F = -\sum_{all\ t}(\Theta(t) - target(t))^2 \tag{4.3}$$

where $\Theta(t)$ is the angle between the movable forearm and the fixed upper arm at time t, and $target(t)$ is the target angle at time t. Thus, it is more difficult for a two-cycle motion to achieve a given fitness than a one-cycle motion, because there is twice the window for error. It is important to note that the fitness equation does not attempt to minimize overall muscle activity or encourage specific neural behavior. All parameters and operators used in the GA, PSO, and BSO, are typical and not fine-tuned for this particular problem.

4.7.1 Genetic Algorithm Training

We use a standard generational GA with two elitist individuals. Each real value in each individual is initialized with uniform random numbers in their respective ranges shown in Table 4.2. Each individual is assigned its fitness during a given generation and a temporary population is populated with the offspring during selection and recombination. Each gene in each offspring undergoes mutation with probability p shown in Table 4.2. Each GA operator is closed over the respective initialization intervals.

Table 4.2: GA parameters

GA Type	Generational
Trials	30
Population Size	30
Generations	5000
Elitism	2 Most fit individuals
Parent Selection	Tournament size 1
Representation	String of doubles and bit-string
Synaptic Weight Range	[0,4] ([-4,0] Inhibitory Links)
Force Multiplier Range	[0,6]
Crossover Type	Arithmetic (reals) (α = 0.5)/Uniform (bits) [12]
Crossover Probability	1.0
Mutation Type	Gaussian(0,0.2) (reals) bit-flip (bits)
Mutation Probability	$\frac{1}{n}$ (n = Number of alleles)

4.7.2 Particle Swarm Training

The swarm behavior of the PSO is modeled by a population of particles where each particle knows its position and velocity in n-dimensional space. Each particle remembers the best location it has seen and also has access to the best location seen by the entire swarm. The maximum and minimum constraint values in the GA are used as constraints for the particles' velocities during training but not their positions. Particles' velocity and position vectors are initialized with uniform random numbers in their respective intervals used in the GA given in Table 4.2. Negative force multiplier values in a particle are treated as their absolute value because force multipliers must be positive. We implement an inertia-style PSO [58] in which the velocity vector of a given particle at timestep t is

$$v(t) = \Omega(t-1)v(t-1) + (c_1)(r_1)(x_b(t-1)) + (c_2)(r_2)(x_{gb}(t-1)) \tag{4.4}$$

where c_1 and c_2 are standard social constant values shown in Table 4.3, the vector x_b is the particle's personal best position, x_{gb} is the global best position, Ω is the population's current inertia value, and r_1 and r_2 are independent uniform random variables on the interval [0,1].

For real numbers, the position vector is then updated as

$$p(t) = p(t-1) + v(t-1) \tag{4.5}$$

and for the binary numbers it is updated as

$$p(t) = \begin{cases} 1 & , rand < s(v(t-1)) \\ 0 & , otherwise \end{cases} \tag{4.6}$$

where $rand$ is a uniform random variable on the interval [0,1] and $s(v)$ is the bipolar sigmoid function $s(v) = 2/(1+e^{-v}) - 1$ where v is the velocity vector.

Table 4.3: PSO parameters

PSO Type	Inertia
Trials	30
Inertia Interval	Linear decay [9⟶3]
Social Constants	$c_1 = c_2 = 2$
Population Size	30
Generations	5000

4.7.3 Breeding Swarm Training

The BSO algorithm is identical to the PSO with added steps of recombination and mutation. Crossover is implemented with a variant of Velocity Propelled Averaged Crossover (VPAC) [58] and mutation is the same as implemented in the GA. In the original implementation of VPAC, offspring 1 and 2 directly inherit velocity vectors from parents 1 and 2, respectively.

Standard VPAC is defined by the following equations:

$$x_{c1} = (x_{p1} + x_{p2})/2 - (\phi_1)(v_{p1}) \tag{4.7}$$

$$x_{c2} = (x_{p1} + x_{p2})/2 - (\phi_2)(v_{p2}) \tag{4.8}$$

where ϕ_1 and ϕ_2 are independent uniform random variables on the interval [0,1]. Each offspring's personal best vector is reset to the current position, $v_{c1} = v_{p1}$, and $v_{c2} = v_{p2}$. However, the modified VPAC used in the these experiments initializes the velocities of the offspring to $v_{c1} = -(v_{p1})(\delta_1)$ and $v_{c2} = -(v_{p2})(\delta_2)$, where δ_1 and δ_2 are independent uniform random variables on the interval [1,3]. This is an experimental enhancement to the BSO to increase the exploratory abilities of the offspring. Each allele is then mutated with a linearly decaying β according to Table 4.4.

To determine the number of particles to undergo crossover, the algorithm requires a breeding ratio r. Typically, a small fixed breeding ratio is chosen and $popsize \cdot r$ particles are generated via VPAC crossover and mutation. The offspring particles then replace the least fit $popsize \cdot r$ particles in the population. VPAC crossover generates offspring in pairs, so after using the breeding ratio to determine the number of offspring we round down to the nearest multiple of 2. In this implementation with a population size of 30 and r=0.1, 2 offspring are created and inserted at each generation. All other parameters are the same as to the PSO.

Table 4.4: BSO parameters

BSO Type	Inertia
Trials	30
Crossover Type	Modified VPAC [58]
Crossover Probability	1.0
Mutation Type	Gaussian(0,β) (reals) / bit-flip (bits)
Mutation β	Linear decay [1$\longrightarrow$0]
Mutation Probability	$\frac{1}{n}$ (n = Number of alleles)
Inertia interval	Linear decay [9$\longrightarrow$3]
Social Constants	$c_1 = c_2 = 2$
Population Size	30
Generations	5000

4.8 Human Subject Experiments

Twenty males, aged 18-30 years, volunteered for participation in these experiments. Subjects had to be involved in resistance training for at least 6 months, and individuals

with previous upper extremity injury were excluded from participation. Each subject completed an informed consent in congruence with university regulations.

4.8.1 Equipment

The TeleMyo 900 (Noraxon USA, Scottsdale, AZ) was used to obtain electromyographic (EMG) recordings for the biceps brachii and the lateral head of the triceps brachii of the dominant arm. All EMG data were collected at a sample rate of 1000 Hz. After cleaning and abrading electrode surface area, a Noraxon dual Ag-AgCl electrode with an inter-electrode distance of 2 cm (Product #272, Noraxon) was placed over each muscle on the dominant arm. A ground electrode was placed on the dominant elbow. To minimize differences in where the electrodes were placed on subsequent days, the electrode sites were outlined with a permanent marker. An uniaxial electrogoniometer was positioned across the elbow joint to allow for synchronization between joint position and EMG recordings. Signals were processed with the MyoResearch XP Master Version 1.04 software (Noraxon USA, Scottsdale, AZ, USA). A Cybex isokinetic dynamometer (Model 340, Cybex Division of Lumex, Inc., New York) was used to conduct MVC testing. All torque data were collected at a sample rate of 100 Hz.

4.8.2 Strength Tests

Two types of strength tests were performed for each subject. All angles θ in the following description correspond to $180\,\mathrm{deg} - \theta$ in Figure 4.1. Each type of test is described below.

Maximal Voluntary Contraction (MVC) Test

Four MVC tests were performed so that EMG collected during submaximal testing could be normalized across subjects: elbow flexion at 45deg, elbow flexion at 90deg, elbow extension at 45deg, and elbow extension at 90deg. The subject was asked to lie supine on a bench with his hips flexed approximately 45deg and his knees flexed approximately 60deg, feet resting on an adjustable platform located at the end of the bench. His forearm was fixed to the dynamometer arm. The shoulder was maintained in a position of $0\,\mathrm{deg}$ of flexion and 45deg of abduction. Once the elbow was positioned, subjects were asked

to push or pull as hard as possible against the immovable dynamometer arm for 6 sec. The subject was not allowed to raise his shoulder, trunk, or legs during the test.

One Repetition Maximum (1RM) Test

A 1RM test for elbow flexion was performed across two days to determine loads for submaximal testing. For this test, the subject stood erect, with their shoulder in anatomical position. On the first day, the subject was asked to perform an initial set of 10 repetitions at a weight that he thought he could lift 10 times or less. After this set, a three-minute rest period was allotted. The weight was then increased by approximately 5-10 pounds (or more, depending on the difficulty of the first set), and a second set was completed in which the subject performed as many repetitions as possible with good form. Good form was defined as having no shoulder movement and using a full range of motion for the elbow flexion movement. Another three-minute rest was given, and then the weight was increased again. The cycle was continued until a weight was found that could be lifted only one time with good form. The official 1-RM is the amount of weight for the last trial that can be successfully complete with good form. On the second day, the subject was started at the maximal weight that was identified on the first day. Then, the cycle was completed as described until the 1RM was found.

4.8.3 Submaximal Testing Protocol

Each subject completed submaximal elbow flexion and extension to produce data for comparison with the SNMS model. The subject was asked to stand with his upper arm held by his side and perform a biceps curl under 9 different conditions: 3 speeds (45 $\frac{deg}{sec}$, 90 $\frac{deg}{sec}$, 135$\frac{deg}{sec}$) and 3 loads (20%, 50%, and 80% of 1RM). Five repetitions were completed for the 20% and 50% conditions; three repetitions were completed for the 80% condition. Speed was controlled with a metronome. The order of speeds was randomized and the order of loads within a speed was randomized. Subjects were allowed to practice the speed of the movement in an unloaded condition prior to beginning testing for that speed condition. A two-minute rest was permitted between loads and a five-minute rest was permitted between speed conditions.

4.8.4 Testing Procedures

Testing was completed during four sessions. Prior to each session, subjects completed a warm-up routine in which they exercised for 10 minutes on either a dual action elliptical trainer or stationary bike. During Session One, subjects were familiarized with the tests that would be performed on subsequent test days: MVC test, 1RM test, and submaximal test. Subjects completed practice trials for each test. All tests were conducted on the dominant arm, or the arm with which the subject wrote. The practice 1RM test was performed first to determine a reasonable weight for the actual test to be performed during Session Three. In each test, particular attention was given to technique, and corrections were made to ensure consistent performance in subsequent testing. During Session Two, subjects were first prepped for EMG testing. MVC testing for elbow flexion and elbow extension was then performed as described previously. Torque and EMG data were recorded continuously for the 6 second time interval. The order of the tests was randomized for each muscle group and for each position (45deg and 90deg) within muscle group. Subjects were allowed 5 minutes of rest between each test. During Session Three, subjects were prepared for EMG testing and then completed 1RM testing as described above. Each EMG was recorded continuously. During Session Four, subjects were prepared for EMG and then completed submaximal testing as described above.

4.8.5 Data Reduction

For the MVC trials, the EMG signal was rectified and a mean EMG was determined between one and five seconds for each muscle group at each joint position. The mean value from the 90deg position was then used to normalize all subsequent trials. The EMG data for 1RM and submaximal trials were rectified and smoothed using a moving average of 300 ms. For the purposes of this paper we will use the submaximal $45\frac{deg}{sec}$ cases with 20% load test cases.

4.9 Results

4.9.1 One-cycle Training

Figure 4.4 shows the performance of each algorithm averaged across 30 trials. Both the simulations and the human subject data use a moving average window of 300ms for the EMG traces. Each algorithm achieves very similar fitness at the end of training. The PSO and BSO converge at about 2,500 generations, whereas the GA continues to make small improvements throughout the run. Figures 4.5, 4.6, and 4.7 show the best fit solution for each training algorithm and a solution with a fitness value equal to that of the average fitness at the end of training. Each figure shows the simulated EMG reading for the biceps brachii and triceps brachii as well as the arm motion with respect to the target motion. In each case the algorithms generate a variety of solutions with respect to overall muscle activity. Figures 4.5(a), 4.6(b), and 4.7(b) each have relatively high coactivation patterns (i.e. significant biceps/triceps activations) while the remaining figures do not. In each figure, the biceps are predominately active during the upward motion and in some cases, such as in Figures 4.5(b) and 4.7(b), the triceps is predominantly active during the downward motion. Note that the EMGs are not proportional to torque generated due to the nonlinearities in the Hill model. Thus, when the biceps and triceps EMGs are equal this does not imply equal and opposite torques.

4.9.2 Two-cycle Training vs. Human Subjects

Figures 4.8, 4.9, and 4.10 compare the arm motion and EMG readings of human test subjects 18, 20, and 1, respectively, with three random samples of the trained SNMS model. The scale of the trained SNMS samples is shifted to match the resting angle of the human test subjects. Furthermore, the actual resting angle of each human subject is the same even though the numeric angle in the figure may vary. Training two-cycle behaviors, similarly to one-cycle behaviors, results in a variety of activation patterns that satisfy the target motion. Figures 4.8(b) and 4.10(b) have fairly similar EMG readings to their respective human subject counterparts. Figure 4.9 is similar in this regard, however the relative difference between the biceps and triceps activities are higher. Figure 4.11(a)

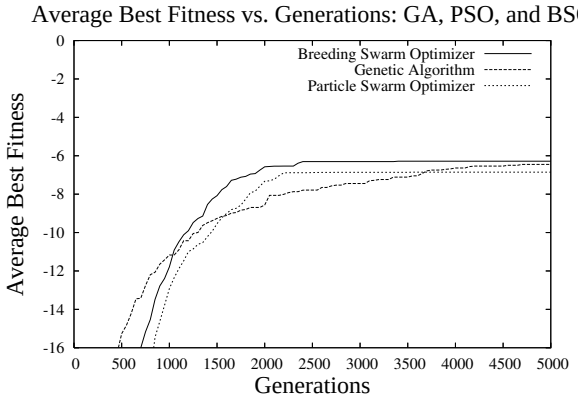


Figure 4.4: The performance of each algorithm with respect to fitness averaged across 30 trials. The modified BSO appears to perform slightly better over the training duration, but the GA and PSO achieves nearly the same level of fitness at the end of the run. The means and standard deviations of the BSO, GA, and PSO at the end of training are -6.44($\pm$ 3.57), -6.85($\pm$ 2.62) -6.28($\pm$ 4.53) respectively. The Student's t-test p values are 0.67, 0.84, and 0.59 for GA vs. PSO, GA vs. BSO, and PSO vs. BSO, respectively, at the end of training, thus there is no statistical difference between any of the training algorithms after 5,000 generations.

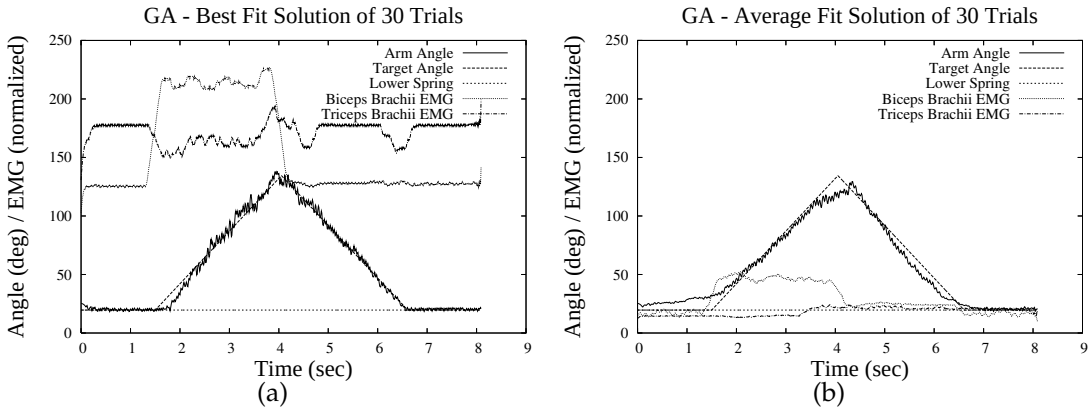


Figure 4.5: Subfigure (a) shows the best fit individual (fitness = -3.1) out of the 30 runs produced by the GA and (b) shows an individual with the fitness of approx. -6.5 which is representative of the average fitness of the GA at the end of training.

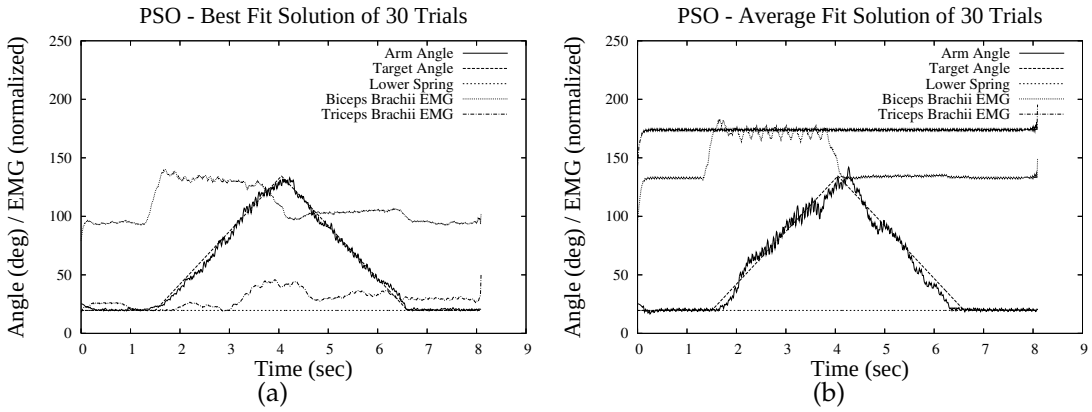


Figure 4.6: Subfigure (a) shows the best fit individual (fitness = -2.1) out of the 30 runs produced by the PSO and (b) shows an individual with the fitness of approx -6.8 which is representative of the average fitness of the PSO at the end of training.

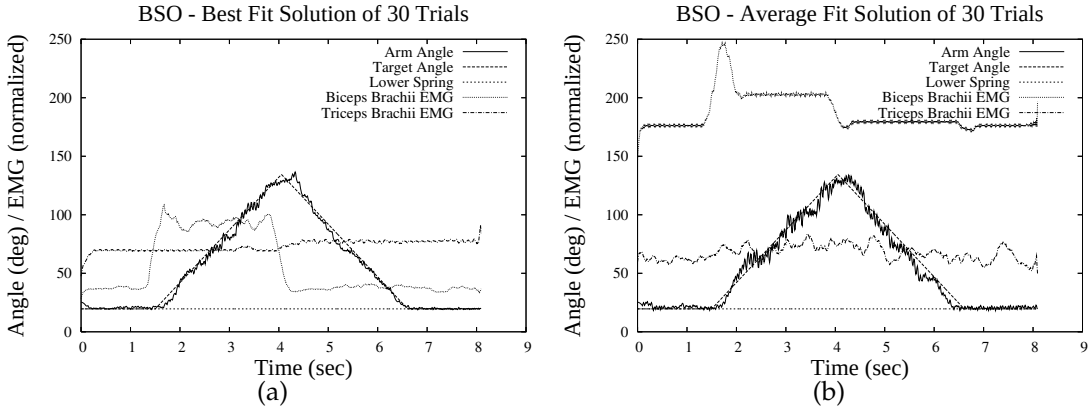


Figure 4.7: Subfigure (a) shows the best fit individual (fitness = -2.9) out of the 30 runs produced by the modified BSO and (b) shows an individual with the fitness of approx. -6.5 which is representative of the average fitness of the BSO at the end of training.

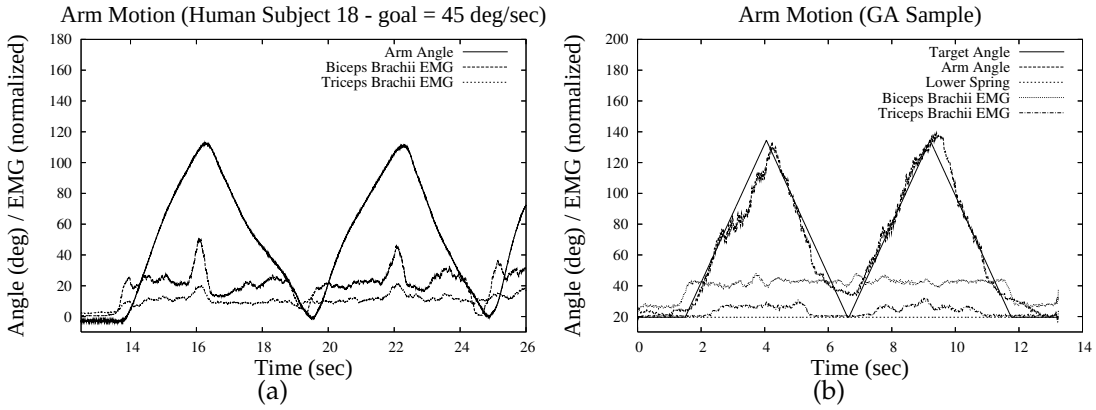


Figure 4.8: Human Subject 18 (a) compared with GA generated solution (b).

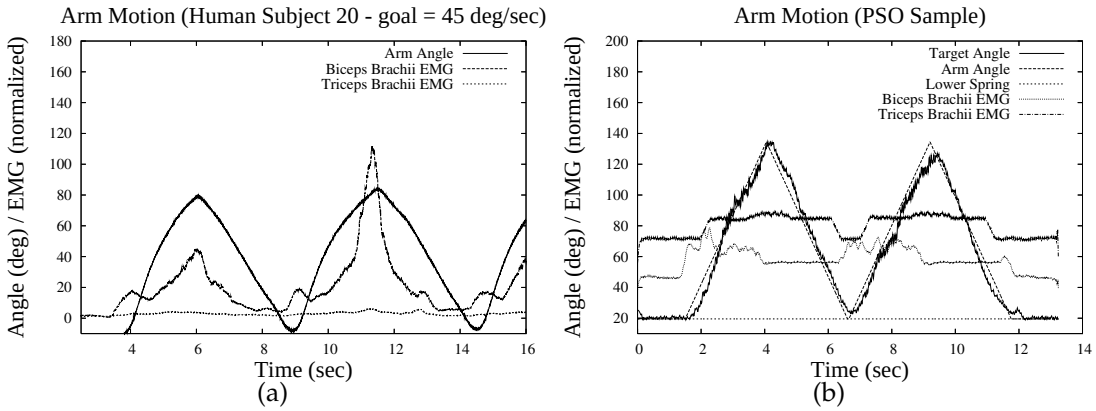


Figure 4.9: Human Subject 5 (a) compared with a PSO generated solution (b).

shows the α-MN firing patterns of the first cycle in Figure 4.10(b). This is to show the potential for emerging motor unit recruitment behaviors and is discussed in the next section.

4.10 Discussion

4.10.1 Algorithm Performance

The purpose of using GAs, PSOs, and BSOs in this paper is not to attempt to compare the absolute usefulness of these algorithms, but to show that they are each *viable candidates* for training biologically plausible behaviors in the SNMS model as well as

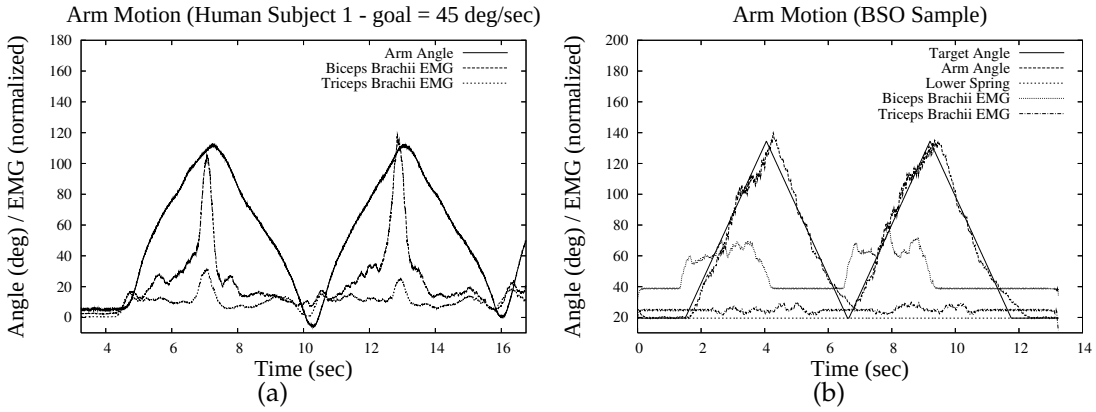


Figure 4.10: Human Subject 1 (a) compared with BSO generated solution (b).

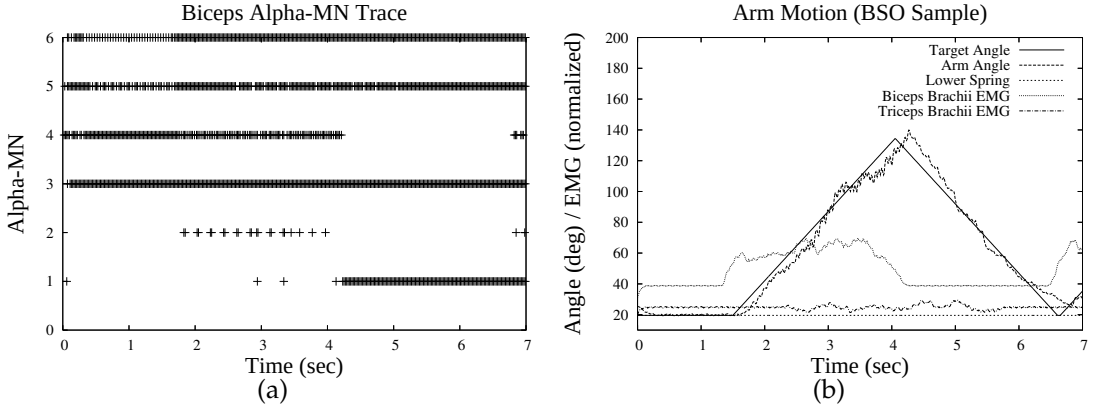


Figure 4.11: This shows the biceps α-MN firing patterns of Figure 4.10(a) and the corresponding simulated EMGs and arm motion(b)(also shown in Figure 4.10(b)). These figures only show the first cycle of motion. Each point in (a) represents an α-MN action potential. Note that the frequencies of α-MNs 5 and 6 increase when upward motion begins (vertical line). Note that α-MNs 1 and 2, which are the second and third strongest motor units, respectively, are "recruited". Alpha-MNs 1 and 2 also slightly increase their frequency when the arm approaches $90\mathrm{deg}$, where gravitational torque is maximized.

other similar models. Each training algorithm has a number of parameters such that where one algorithm outperforms another with one parameter set, the case may be reversed with another parameter set.

It is interesting to note that neither the PSO, BSO, nor the GA were originally designed to accommodate both real values and discrete values simultaneously. Given the complexity of the SNMS model, it is an interesting result by itself that each training method generated reasonable and plausible solutions despite the mixed types of parameters. This is an interesting question to investigate because the consequences of mixing parameter types in this manner are not known.

Also, the fitness of a trained system is determined by the deviation from the target motion and not by desired neural patterns. As a result, there is no explicit pressure on the algorithm to minimize biceps/triceps opposition to avoid trembling behaviors or to exploit certain neural pathways more than others. An advantage to our approach is that we could train specific EMG behaviors to match biological data by adding a "neural factor" to the fitness function. Specifically, the fitness function could reward solutions which minimize overall muscle contraction. In this paper, we did not include such a factor in the hopes that biologically plausible behaviors would emerge without bias, which in fact occurred.

4.10.2 Biological Behaviors Observed in Simulation

The results show that the training algorithms are in fact effective in achieving correct arm movements as well as yielding other biological behaviors: tonic tension at the resting angle, biceps/triceps coactivation patterns, and key aspects of motor unit recruitment. The algorithms do not directly train these behaviors that represent fundamental features of biological movement.

Tonic Tension

All neurons in the spinal cord are constantly bombarded with signals from other neurons which cause low levels of synaptic activity at the neuromuscular junction [34]. Even when muscle fiber groups are shortening they still receive small levels of neural

stimulation. This is a stretch/posture reflex mechanism which allows muscles to respond more rapidly when a descending signal causes the muscle fibers to contract [56].

In our model we observe similar tonic behavior. We observe low level α-MN activity during resting periods, seen in Figures 4.8(b), 4.9(b), and 4.11(a), which causes slight to moderate motor unit contractions. Since our simulations have only 6 motor units per muscle, the EMGs generated by the tonic behaviors are relatively large. This is an interesting result because it shows that the training algorithms take advantage of the afferent pathways to generate biological realistic tonic tension behaviors. In biology these afferent pathways play a key role in maintaining tonic behaviors and influence the responsiveness and strength of motor units.

Biceps/Triceps Coactivation

The second key biological behavior we observe is biceps/triceps coactivation. When engaging in physical activity both muscles in an agonist/antagonist pair are usually active (e.g. Figures 4.8, 4.9, and 4.10). For example, performing a biceps curl with weights, as in the human experiments, the biceps is dominantly active over the triceps. However, the antagonist muscle is usually somewhat active as well to stabilize motion.

Each of the three algorithms successfully trains biceps/triceps coactivation activity to help fit the target curve during upward and downward motions. In fact, the motion is significantly closer to the target of constant velocity than the human subjects. The biceps/triceps coactivation matches the general behavior of the human subjects (Figure 4.8(a), 4.9(a), and 4.10(a)). The fact that the training spontaneously adopted coactivation patterns is surprising, given that we know from previous experiments the triceps are not necessary to match the target motion [23] and presumably not activating them would be the simpler solution. Furthermore, all triceps activity is due to the afferent feedback pathways. Thus this proves that the training algorithms take advantage of these pathways and of the triceps to improve the systems ability to match the target motion.

An interesting aspect of the human test subject data is the variance of triceps EMG activity relative to the biceps. During upward motions, subjects 1 and 18 have small spikes in triceps activity as the subject approaches the highest angle. Neurologically, this

could be due to the length-sensitive afferent fibers in the triceps; however, it is more likely that this activity is a result of the human subject stabilizing the arm with increased triceps activity to lift the weight at a steady rate.

In contrast, human subject 5 yields very little triceps activity throughout the motion. This subject was a unique case in that he was an experienced weight lifter and had significant definition and control in his upper body. In light of this case it is interesting to note that even among human subjects there is a variety of EMG patterns which generate the same types of motion. Given the variability of human EMG readings for a given motion, with respect to a single subject (Figure 4.9) or multiple subjects, it is reasonable to expect a variety of simulated EMGs that generate the same motion. Thus, in modeling this type of system it reasonable to expect a variety of neuromuscular patterns that produce the same motion.

Recruitment Behaviors

The third key biological behavior we observe is that of recruitment-like activity. This phenomenon in biology is explained with the size principle which states that motor units are recruited during contraction in order from weak fatigue-resistant fibers to the strong fatigable fibers. Weak motor units remain active throughout the motion and stronger motor units activate if more contractile force is needed. Another characteristic of recruitment is an increase or decrease in firing rates when the muscle requires more or less contractile force, respectively.

Figure 4.11(a) shows recruitment-like firing patterns during upward motion. Alpha-MN 2 is recruited shortly after the motion is initiated, but the brief delay before it initially fires makes it clear that it is not only responding to the descending inputs, but was recruited by the afferent feedback pathways. Furthermore, α-MN 1 fires twice near $90\mathrm{deg}$ to compensate for gravitational torque, which is maximized at this angle. Both these neurons' behaviors are characteristic of strong fatigable muscle fibers. Alpha-MNs 5 and 6 remain active throughout the motion, which is characteristic of weak fatigue-resistant muscle fibers.

Miscellaneous Behaviors

It is important to note that there are minor oscillations in the simulations during the upward and downward motions. This behavior occurs primarily because the Hill model and the neural network is a pulse-coded system. During a given timestep a motor unit either contracts or relaxes. This contraction/relaxation scheme is more biological than a standard closed-loop controller. However, biological muscles consist of thousands of muscle fibers which act to smooth out these irregularities.

These oscillations are also due to biceps/triceps coactivation patterns. Low overall activity, shown in Figures 4.6(a) and 4.7(a), results in small oscillations during movement. On the other hand, high biceps and triceps EMG readings result in significantly larger oscillations as shown in Figures 4.6(b) and 4.7. These high coactivation patterns can presumably be trained out of the system with a fitness function that incorporates EMG patterns.

4.10.3 Long Term Applications

To show the potential applications of this method we present two (of many) specific research areas that we will investigate once the model is fully developed. First, practitioners have hypothesized that segmental feedback from muscle receptors, particularly from Golgi tendon organs (GTO), is responsible for the altered recruitment of motor neuron firing during eccentric contraction of the biceps brachii muscle. An eccentric contraction occurs when a muscle lengthens as tension increases, which is often the case when a muscle works to decelerate a limb movement or control limb movement caused by an external force [70]. The size principle of motor neuron recruitment [25–27] predicts that for a given movement at a joint, the order of activation among the motor units that are innervating the particular skeletal muscle responsible for the movement is as follows: 1) small alpha motor neurons (slow muscle fibers), 2) medium-sized alpha motor neurons (fast fatigue-resistant muscle fibers), 3) large alpha motor neurons (fast, fatigable muscle fibers) [64].

However, the size-principle for motor unit recruitment is often superseded in the case of eccentric contraction by a muscle. This exception to the rule has been

demonstrated for muscles acting at the metacarpo-phalangeal [30], elbow [45,55], and ankle [46,47] joints. The reason(s) for this exception is not clear [13,14]. However, eccentric muscle contractions provide an optimal stimulus for activating Golgi tendon organs (GTO)/Ib fibers as well as muscle spindles [14].

The size principle of motor unit recruitment is largely based on a consideration of only excitatory input to the motor neuron pool. But one can consider its result in terms of the summation of combined excitatory and inhibitory synaptic inputs. Consequently, if an inhibitory input was initiated to a motor neuron pool that had all or most of its motor neurons active to a similar degree, then the motor neurons that slow their discharge first should be the smaller ones as opposed to the larger ones. This occurs because the small motor neurons should have their membrane potential lowered by inhibitory post synaptic potentials (IPSP)s to a greater extent than the larger motor neurons. Therefore, one could expect to see relatively unabated activity for larger motor neurons even when the number of action potentials discharged by small motor neurons in the same neuron pool diminished. It is precisely this scenario that has been demonstrated in certain eccentric muscle contractions [13,14].

In our model, with uniform motor units, we have observed the shift from an isometric to an eccentric motion resulting in changing discharge rates of the α-MNs and the "derecruitment" of some, but not all, of the motor units [23]. By introducing or removing GTO group Ib pathways from our model it will be possible to examine their role in recruitment during eccentric motions. This type of wholesale inclusion or removal of a neuron type is clearly not possible in actual test subjects.

Our second major research area is the role of the SNMS in compensating for and adapting to injury. For example, consider anterior cruciate ligament (ACL) injuries. At the knee joint, quadriceps femoris (QF) dysfunction (in the form of muscle weakness, altered muscle activity patterns, and coordination deficits) has been consistently demonstrated in persons who sustain an anterior cruciate ligament (ACL) injury. However, the reasons for this dysfunction remain unclear. One hypothesis is that QF muscle receptor function (muscle spindle and GTO) is aberrant after ACL injury [69]. This aberrant function is a result of alterations in collagen and connective tissues due to

decreased physical activity which, in turn, causes decreased stiffness in the series elastic component of the connective tissue. This decreased stiffness increases the time for force transfer from the muscle to the bone and results in a demonstrable increased electromechanical delay [35]. Subjects, in turn, alter muscle recruitment (”keep the QF turned on”) to compensate for the increased delay. Aberrant function has been experimentally observed in the form of diminished proprioception in the ACL deficient knee [51].

A competing hypothesis states that increased afferent (feedback) stimulation of the QF muscle receptors occurs due to the increased anterior sheer load on the QF in the ACL deficient knee. This increase, coupled with the absence or alteration of afferent input from the injured ACL may explain the altered muscle activation patterns that have been observed [63] in an apparent attempt to reduce anterior tibial translation. With our model, we can simulate ACL-like injury in a previously trained model by changing the appropriate functional parameters. We can then retrain the model to reproduce normal behavior despite the modified functional parameters (representing the damaged tissue). The selection of compensatory neural pathways would provide insight into which mechanism is responsible for the altered muscle recruitment patterns. This increased understanding of neuromuscular adaptations will in turn aid in the development of optimal treatment strategies.

4.11 Conclusions

In this paper we show that GAs, PSOs, and BSOs can be used to fill many unknown details, e.g. the strengths of the individual muscle fibers and synaptic links, in a SNMS model. We find that these algorithms are effective in training the SNMS model with multiple parameter types to track target motions for several training cases.

We also show that these techniques spontaneously adopt biologically plausible behaviors on the neural and gross anatomical levels without direct selective pressure (i.e. Equation 4.3 only uses target motion). We confirm this by comparing simulation results to experiments involving human subjects. Behaviors such as biceps/triceps coactivation and noticeable increases in triceps activity during the motion show that this

model is sufficiently complex to generate plausible neuromuscular patterns. We also observe notable tonic behaviors during the resting phases of motion which resembles neuromuscular background activity when muscles are at rest. Finally, we find that trained instances of the model can yield recruitment-like behaviors during eccentric motions.

These patterns emerge only from specifying the target anatomical movement and not by training for any types of neuromuscular behavior. To generate even more realistic neural patterns researchers can adjust the fitness function, and the architecture of the model if needed, to reflect the desired types of neuromuscular behaviors as well as expand the training cases to different types of motion. The fitness function can include additional factors to minimize overall EMG activity, eliminate strong oscillations in movement, or penalize for any undesired behavior. Researchers can also expand the model architecture by adding neuron types and new types of feedback such as joint or pain receptors. The model can also be changed easily to simulate various motions in different positions such as moving the arm in the horizontal plane or in a supine position.

Acknowledgments

We thank Dr. Richard Wells of the University of Idaho's Department of Electrical and Computer Engineering and Dr. Mark DeSantis of the University of Idaho's Department of Biological Sciences for their helpful discussions and suggestions.

Chapter 5

STOCHASTIC TRAINING OF A BIOLOGICALLY PLAUSIBLE SPINO-NEUROMUSCULAR SYSTEM MODEL [1]

Stanley Gotshall, Terence Soule
Department of Computer Science, University of Idaho, Moscow, ID

Chapter 4 demonstrates that key biological behaviors emerge in trained instances of the SNMS model. However, the experiments in Chapter 4 require extensive CPU time to complete. Thus, the next goal was to focus on effective methods of training the SNMS model while retaining the biological behaviors describes in Chapter 4. Thus this chapter compares various stochastic optimization techniques to determine which algorithms are most effective in training the SNMS model. This chapter also expands on the application of the SNMS model as a robotic control system.

5.1 Abstract

A primary goal of evolutionary robotics is to create systems that are as robust and adaptive as the human body. Moving toward this goal often involves training control systems that process sensory information in a way similar to humans. Artificial neural networks have been an increasingly popular option for this because they consist of processing units that approximate the synaptic activity of biological signal processing units, i.e. neurons. In this paper we train a nonlinear recurrent spino-neuromuscular system (SNMS) model and compare the performance of genetic algorithms (GA)s, particle swarm optimizers (PSO)s, and GA/PSO hybrids. Several key features of the SNMS model have previously been modeled individually but have not been combined

[1] THIS PAPER HAS BEEN ACCEPTED FOR PUBLICATION IN THE GENETIC AND EVOLUTIONARY COMPUTATION CONFERENCE 2007. AUTHORS: STANLEY GOTSHALL AND TERENCE SOULE

into a single model as is done here. The results show that each algorithm produces fit solutions and generates fundamental biological behaviors, such as tonic tension behaviors and triceps activation patterns, that are not explicitly trained.

5.2 Introduction

In biology, neural networks involving the spinal cord provide the body with a remarkably precise and adaptive system for control. Motor neurons and muscle fibers exchange electrical signals to produce controlled anatomical movements throughout the body. Thus, designing a biologically plausible neural controller can potentially allow researchers to harness the properties of biological systems that are uniquely suited to control complex mechanical systems.

The nervous system has an unmatched ability to control highly nonlinear systems, i.e. skeletal muscles. Muscles consist of thousands of muscle fibers that respond to electrochemical signals from the spinal cord to move the body [34]. In turn, feedback signals from muscle fibers allow the spinal cord to respond and activate muscle groups in order to generate smooth and precise motions. By modeling control systems after the spinal cord, we increase our understanding of the role and importance of certain neural pathways in the nervous system. It also opens the door to developing prosthetic devices that interpret brain signals the same way the spinal cord does and to design robots with similar processing abilities.

An overarching goal of evolutionary robotics is to create autonomous agents that are as robust and precise as the human body. This typically involves training systems to process information received from visual, auditory, and tactile sensors. Research in evolutionary robotics focuses on goals such as training agents to generate controlled motions and find optimal paths through mazes. However, these methods typically do not use biologically realistic architectures to process sensory signals. Although these methods often use spiking neuron models that mimic the computational power of a single nerve cell, the network architecture does not propagate and process the information in a biological way. Thus, a key step in moving toward biologically realistic information processing is to use a biologically plausible network architecture to control

robotic systems.

This paper applies three stochastic/evolutionary algorithms: 1) genetic algorithms (GA)s, 2) particle swarm optimizers (PSO)s, and 3) breeding swarm optimizers (BSO)s, which is a GA/PSO hybrid, to train anatomical motion in a biologically based spino-neuromuscular system (SNMS) model. Neural networks in the spinal cord simultaneously integrate signals from the brain and muscle sensors to determine the amount of stimulation that the muscle fibers require to accomplish a motion. Our model mimics this integration by simulating descending brain signals and muscle feedback pathways which encode the contractile velocity and length of muscle fibers.

Our results show these algorithms, with varying degrees of success, effectively train the 600+ real-value and binary parameters in the model to generate controlled anatomical movements. Furthermore, comparisons with data from human subjects show that the training process produces key biological control mechanisms that are not explicitly trained.

5.3 Background and Previous Work

Artificial neural networks use individual processing units, neurons, to mimic the processing power of the nervous system. Traditional connectionist neural networks consist of neurons that compute static functions of their inputs [11,42,71] which, in the biological sense, represents an activity level (i.e. frequency) for that neuron. Networks consisting of these neurons are effective for many applications such as classification, prediction, and control systems. However, as our understanding of the nervous system increases, more biologically realistic neuron models have emerged, i.e. spiking neurons. Spiking neurons more accurately model the temporal dynamics of biological neurons. Thus, networks of spiking neurons can more accurately approximate the signal processing abilities of neural networks in the brain or spinal cord.

Although spiking neural networks are uniquely suited for biological applications, researchers also use them for various types of problems commonly solved with traditional neural networks [17,50]. Furthermore, spiking neurons are computationally more powerful than traditional neurons [41]. Thus, practitioners can construct small

networks of spiking neurons to accomplish the same task as a traditional network of greater size. This is done in the hope that the smaller number of synaptic weights will ease the training process.

However, training spiking neural networks requires innovative techniques. Traditional feed-forward neural networks are trainable via back-propagation or Hebbian learning. Unfortunately, these training techniques cannot accommodate the temporal characteristics of spiking neural networks. Fortunately, researchers are developing a new generation of neural network training techniques. Instead of training numeric input-output patterns, these methods use mathematical methods to train temporal spiking patterns [36, 52]. These techniques are useful in pattern recognition and control system problems, but only when the objective is to train a known input/output spiking pattern. However, in some problems the patterns needed to solve the problem are not known. This is often the case when the task of a neural network is to control a mechanical system because the goal is typically quantified in terms of a desired motion and not a particular network output pattern. These scenarios need a new training approach that does not depend on practitioners knowing the input/output patterns a priori.

Thus, our goal is to use stochastic algorithms to train a spiking neural network, and other parameters in the system, to control a biologically plausible neuromuscular model of an arm. This approach lets the user specify the quality of potential solutions in terms of error from a target motion. This way the practitioner does not need to know the specifics of the network's output patterns required to produce the desired motions.

5.4 Models

The models used in this chapter are omitted here and described in Chapter 4.

5.5 Experiments

The following algorithms train three distinct parameter types in the model: 1) synaptic weights, 2) strengths of muscle fibers, and 3) binary selectors for neural subnets. All

individuals in each algorithm are randomly initialized in the same manner where synaptic weights are randomly selected uniformly in the interval [0,4] (excitatory) and [-4,0] (inhibitory), fiber strengths in the interval [0,6], and subnet selectors in the uniform binary interval [0,1]. The velocity vectors in the PSO and BSO initialize with the intervals as their respective position vectors both in the positive and negative direction with the exception of the binary values which initialize their velocity vectors in the interval [-6,6].

Each algorithm evaluates fitness of potential solutions with the equation

$$F = -\sum_{all\ t}(\Theta(t) - target(t))^2 \tag{5.1}$$

where $\Theta(t)$ is the angle between the movable forearm and the fixed upper arm at time t, and $target(t)$ is the target angle at time t. It is important to note that the fitness equation does not attempt to minimize overall muscle activity or encourage specific neural behaviors. The fitness function measures the error from the target trace in Figure 5.1. The following subsections describe each algorithm in more detail.

Table 5.1: GA, PSO, and BSO parameters

Parameter	GA	PSO	BSO
Trials	35	35	35
Fitness Evaluations	150k	150k	150k
Population Size	30	30	30
Selection Type	Tournament	N/A	Tournament
Tournament Size	2	N/A	2
Crossover Probability	1.0	N/A	N/A
Crossover Type (reals)	Arithmetic + N(0,0.2) [12]	N/A	VPAC [58]
Crossover Type (bits)	Uniform	N/A	Uniform
Mutation Probability	1/dimensions	N/A	1/dimensions
Mutation Type (reals)	N(0,0.2)	N/A	N(0,1→0)
Mutation Type (bits)	Uniform	N/A	Uniform
Mutation Variance (reals)	N/A	N/A	1.0 → 0.0
Social (c_1, c_2)	N/A	2,2	2,2
Inertia	N/A	0.9→ 0.3	0.9→0.3

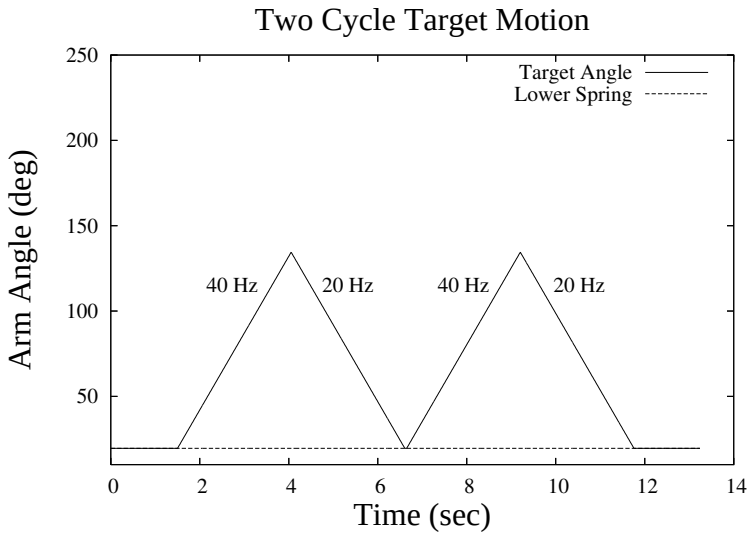


Figure 5.1: Two cycle target motion. Starting at 0 seconds, no descending pathways receive an input pattern. At 1.5 seconds, an input pattern of 40 Hz is sent to a subset the biceps' subnets, as chosen by the training algorithm, and the signals across each input neuron is phase shifted evenly across a window of $\frac{1}{40Hz}$. At a given frequency, the inputs to the subnets are out of phase and do not overlap during a given timestep unless the frequency is sufficiently high. During the downward motion the biceps network receives a descending input of 20 Hz. The upward and downward target angular velocities are 45 $\frac{deg}{sec}$ in both directions.

5.5.1 Genetic Algorithm Training

These experiments use the two standard models of GAs, steady-state and generational. The generational GA creates a new population every generation and automatically preserves the two most fit individuals from the previous generation (elitism). The steady-state version on the other hand does not generate a new population, but replaces the two least fit individuals with two offspring during each iteration. For the real-value segments of the individuals, recombination is performed via arithmetic crossover [12] added to a small normal random variable as shown in Table 5.1. Uniform crossover is used for the binary component. In both algorithms, after an individual is generated via crossover each allele in the offspring then undergoes mutation with probability p, shown in Table 5.1.

5.5.2 Particle Swarm Training

The PSO is a relatively new stochastic search and optimization technique based on swarm intelligence developed by Eberhart and Kennedy in 1995 [37]. The following experiments employ the two main PSO types, inertia and constriction. The PSO is modeled with a population of particles where each particle knows its position and velocity in n-dimensional space. Each particle remembers the best location it has seen and also has access to the best location seen by the entire swarm. The maximum and minimum values in the GA are used as constraints for the particles' velocities but not their positions. PSO-style algorithms usually do not need artificial constraints on position values unless the problem itself requires certain constraints. The inertia PSO [58] particle velocity at timestep t is

$$v(t) = \Omega(t-1)v(t-1) + (c_1)(r_1)(x_b(t-1)) + (c_2)(r_2)(x_{gb}(t-1)) \tag{5.2}$$

where c_1 and c_2 are social constants shown in Table 5.1, the vector x_b is the particle's personal best position, x_{gb} is the global best position, Ω is the population's current inertia value, and r_1 and r_2 are independent uniform random variables in the interval [0,1]. The constriction PSO [10] velocity update equation is

$$v(t) = \tau[v(t-1) + (c_1)(r_1)(x_b(t-1)) + (c_2)(r_2)(x_{gb}(t-1))] \tag{5.3}$$

where τ is the constriction coefficient derived from the equation

$$\tau = \frac{2}{|2 - \phi - \sqrt{\phi^2 - 4\phi}|} \tag{5.4}$$

where

$$\phi = \begin{cases} c_1 + c_2 & , c_1 + c_2 > 4 \\ 4.1 & , \text{otherwise}. \end{cases} \tag{5.5}$$

In these experiments $c_1 + c_2$=4, thus ϕ=4.1 and $\tau \approx 0.73$. For real numbers, the position vector for both PSO types is then updated as

$$p(t) = p(t-1) + v(t) \tag{5.6}$$

and for the binary numbers it is updated with

$$p(t) = \begin{cases} 1 & , rand < s(v(t-1)) \\ 0 & , otherwise \end{cases} \tag{5.7}$$

where $rand$ is a uniform random variable in the interval [0,1] and $s(v)$ is the sigmoid function $s(v) = 1/(1 + e^{-v})$ where v is the velocity vector [37].

5.5.3 Breeding Swarm Training

The BSO algorithm is identical to the PSO with added steps of crossover and mutation. Crossover is implemented with Settles' VPAC (Velocity Propelled Averaged Crossover) [58]. VPAC is defined as

$$x_{c1} = (x_{p1} + x_{p2})/2 - (\phi_1)(v_{p1}) \tag{5.8}$$

$$x_{c2} = (x_{p1} + x_{p2})/2 - (\phi_2)(v_{p2}) \tag{5.9}$$

where ϕ_1 and ϕ_2 are independent uniform random variables on the interval [0,1]. Both offsprings' x_b (personal best) vector is reset, $v_{c1} = v_{p1}$, and $v_{c2} = v_{p2}$ after crossover. Each allele is then mutated with a linearly decaying variance according to Table 5.1.

To determine the number of particles to undergo crossover, the algorithm uses a breeding ratio r. Typically, a small fixed breeding ratio is chosen and $popsize \cdot r$ particles are generated via VPAC crossover and mutation. The offspring particles then replace the least fit $popsize \cdot r$ particles in the population. VPAC crossover generates offspring in pairs, so after using the breeding ratio to determine the number of offspring we round down to the nearest multiple of 2. In this implementation with a population size of 30 and r=0.1, 2 offspring are created via crossover and mutation and then inserted at each generation.

5.5.4 Human Subjects

In our experiments, twenty human subjects completed submaximal elbow flexion and extension to produce data for comparison with our SNMS model. Each subject was asked to stand with his upper arm held by his side and perform a biceps curl under 9 different conditions: 3 speeds (45 $\frac{deg}{sec}$, 90 $\frac{deg}{sec}$, 135$\frac{deg}{sec}$) and 3 loads (20%, 50%, and 80% of the subject's maximum repetition weight (MRW)). Five repetitions were completed for the 20% and 50% conditions and three repetitions were completed for the 80% condition. Speed was controlled with a metronome. The order of speeds was randomized and the order of loads within a speed was randomized. Subjects were allowed to practice the speed of the movement in an unloaded condition prior to beginning testing for that speed condition. A two-minute and five-minute rest was permitted between the load and speed conditions, respectively. In this paper we present data corresponding to three human subjects at a speed of 45 $\frac{deg}{sec}$ at 20% and 50% of the subjects' MRW.

5.6 Results

Now we compare the relative performance of each algorithm and examine the distribution of best fitness values at the end of each algorithm run. We also examine the behavior of individually trained systems and compare this with electromyogram (EMG)

data recorded from the human subjects. EMGs reflect the amount of overall muscle activity by measuring the excitation of a muscle.

5.6.1 Fitness

Figure 5.2 shows the average best fitness of each algorithm averaged over 35 trials over 150,000 fitness evaluations. On average, the steady state GA yields the highest fit solutions at the end of training followed closely by the inertia BSO. Figure 5.3 and Table 5.2 show the distribution of solutions for each algorithm with respect to fitness at the end of each training run.

Table 5.2: Mean, Standard Deviation of Mean, Best, and Worst Fitness of Each Algorithm at the End of Training

Algorithm	Mean	Std. Dev.	Best	Worst
GA (Steady State)	-18.56	4.94	-10.95	-38.75
GA (Generational)	-30.48	12.91	-15.66	-75.05
PSO (Inertia)	-36.60	28.04	-9.12	-116.13
PSO (Constriction)	-64.40	53.17	-11.70	-210.80
BSO(Inertia)	-25.59	20.46	-5.72	-88.79
BSO (Constriction)	-46.73	43.10	-10.01	-177.16

5.6.2 Behaviors

These results show behaviors of individually trained SNMS models and the human subjects with EMG readings. In our experiments we observe that each algorithm generates indistinguishable solutions for a given fitness value. Figures 5.4 and 5.5 produce similar anatomical motions and are generated by the steady state GA and the inertia BSO, respectively. Each algorithm consistently yields solutions which activate the triceps during the upward motion[2]. Interestingly, the triceps EMG in two human subjects (Figures 5.6 and 5.7) and the PSO solution (Figure 5.5) steadily increases during

[2]EMG data from the simulation is taken from a variable in the Hill model which corresponds to the voltage across the neuromuscular endplate. Surface electrodes approximate this voltage on a larger scale.

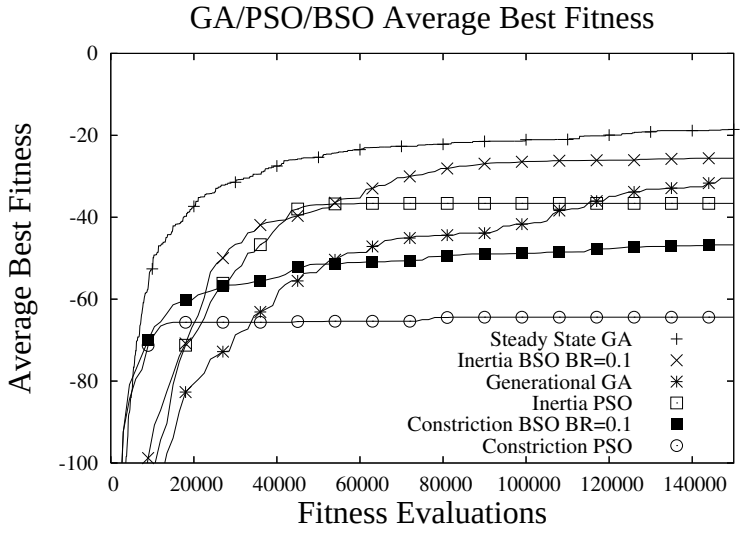


Figure 5.2: This shows the average best fitness of each algorithm over 35 runs. The steady state GA mean is statistically higher than all other algorithms with $p < 0.01$, however it is higher than the constriction PSO with $p < 0.10$ (Student's t-test). The means of the BSO with inertia and constriction are marginally higher than their PSO counterparts with $p = 0.065$ and $p = 0.013$, respectively (Student's t-test).

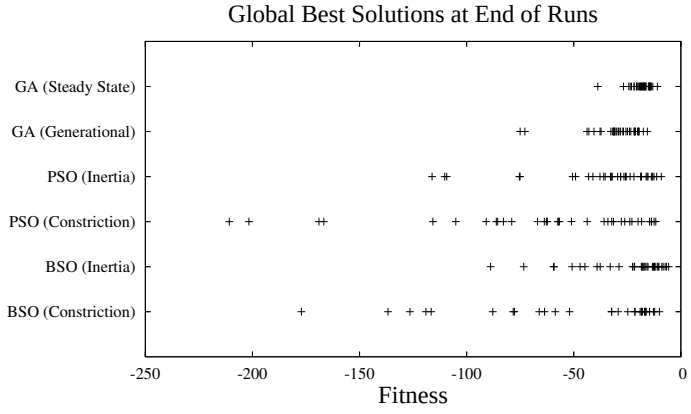


Figure 5.3: The distribution of the steady state and generational GAs tightly surround their respective means whereas the other algorithms have much wider distributions. The BSO algorithms appear to have slightly tighter distributions around their means than their PSO counterparts.

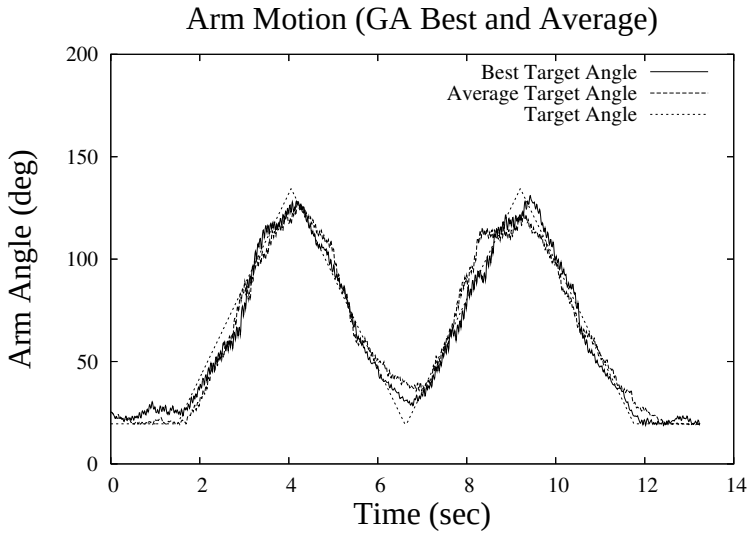


Figure 5.4: This shows the best fit solution generated by the steady state GA and an average fit solution. The GA's best solution binds more tightly to the target angle throughout the motion.

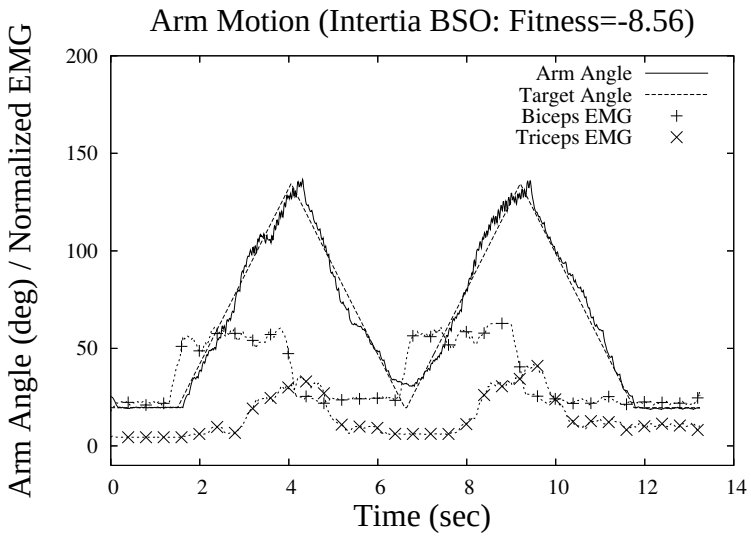


Figure 5.5: The BSO finds a solution which uses the afferent feedback pathways and activates the triceps even though it receives no descending input.

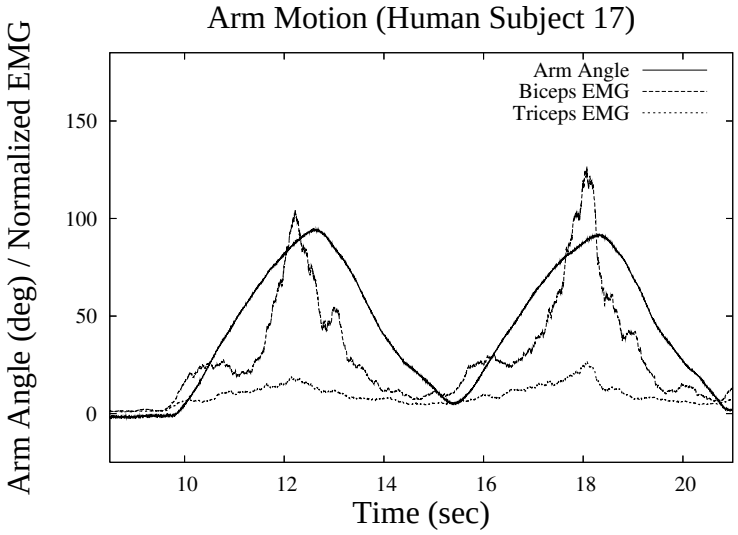


Figure 5.6: This shows the anatomical movements and surface electrode EMG readings during human subject 17's upward and downward motions at $45\frac{deg}{sec}$ at 50% of MRW.

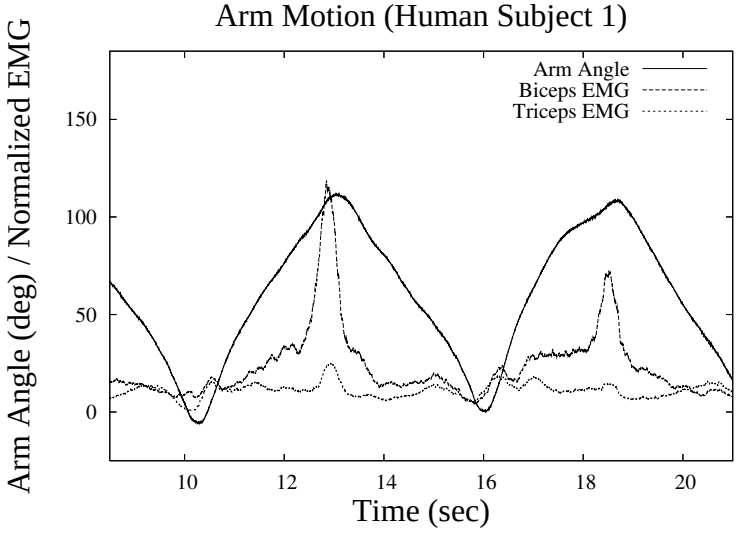


Figure 5.7: This shows the anatomical movements and surface electrode EMG readings during human subject 1's upward and downward motions at $45\frac{deg}{sec}$ at 20% of MRW.

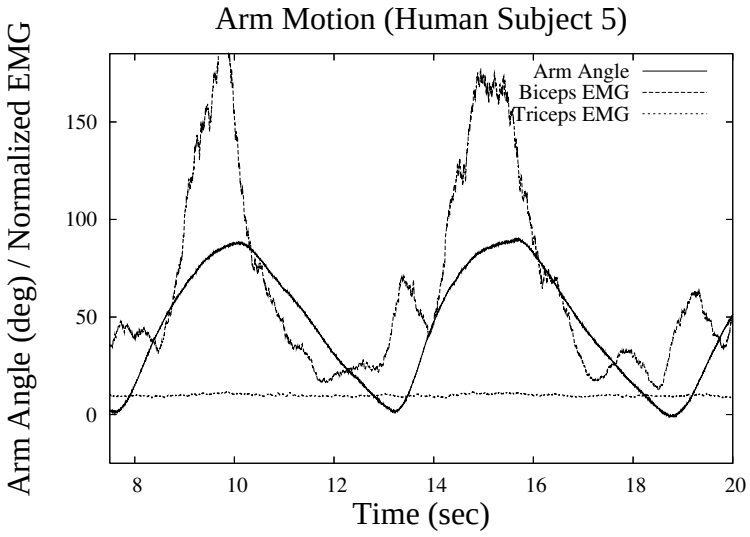


Figure 5.8: This shows the anatomical movements and surface electrode EMG readings during human subject 5's upward and downward motions at $45\frac{deg}{sec}$ at 50% of MRW. Note that for this subject increased triceps activity is not needed for stable motion.

the upward motion. However, we hypothesize that human subject 5 has a particularly strong and toned upper body because virtually no extra triceps activity is needed to stabilize the motion (Figure 5.8). Many solutions also generate small amounts of muscle fiber excitation, i.e. tonic tension, in the absence of descending input also shown in Figure 5.5 which is also seen in human subject 17 (Figure 5.6) just before motion begins.

5.7 Conclusions

In this paper we show that GAs, PSOs, and BSOs can be used to fill in the unknown details of the SNMS model, e.g. the strengths of the individual muscle fibers and synaptic links, to integrate them all in a single model. We find that these algorithms effectively train the SNMS model to follow the target motion. On average, the steady state GA outperforms all the other algorithms and generates the most consistent results with respect to fitness. We hypothesize that the PSO based algorithms yield less consistent results because the position update equation is a function only of the velocity and not position. The velocity values for the binary component direct the particle to a particular position in binary space, but as the velocity for a given dimension approaches

0, the position alternates between 0 and 1 with equal probability (Equation 5.7). Since the binary values in the particle represent which subnets receive descending inputs, changing these values during algorithm convergence will likely result in large changes in fitness which may make it difficult for the PSO based algorithms to generate more consistent solutions.

Nevertheless, these algorithms generate fundamental biological behaviors in the SNMS model that are not directly trained. In biology, during contraction of the biceps, the triceps usually becomes active to stabilize motion (Figure 5.6) which is observed in the model (Figure 5.5). It is important to note that the triceps motor units do not receiving descending signals in the simulation, and thus could remain inactive. However, the algorithms tend to utilize the triceps via the afferent pathways to generate solutions that satisfy the target motion. This suggests that the afferent feedback components are sufficient to generate the same behaviors they are responsible for in biology.

The algorithms also tend to train the feedback pathways in the model to produce small amounts of muscle fiber excitation, i.e. tonic tension, in the absence of descending input (Figure 5.5). In biology, the spinal cord's stretch reflex mediated by the afferent pathways maintains this tension which plays a key role in maintaining balance and posture. In the simulation, it facilitates rapid muscle fiber responses at the initiation of the upward motion which is consistent with the function of the stretch reflex and other related reflexes [34].

The successful training of our SNMS model opens the door for modeling increasingly complex neuromuscular system models for a range of anatomical motions such as raising and lowering the arm at varying speeds and by training the system to lift various weights. Researchers can also expand the model architecture by adding neuron types and new types of feedback such as joint or pain receptors. The model can also be changed easily to simulate various motions in different positions such as moving the arm in the horizontal plane or in a supine orientation. These types of improvements in neuromuscular system modeling will increase our understanding of the spinal cord as a nonlinear control system. This knowledge will also allow researchers enhance the

information processing abilities of robotic control systems.

5.8 Acknowledgements

We thank Dr. Kathy Browder and her assistant Jessica Sampson for their helpful discussions and for providing the human subject data.

Chapter 6

EVOLVING STABLE BEHAVIOR IN A BIOLOGICALLY PLAUSIBLE SPINO-NEUROMUSCULAR SYSTEM MODEL. AUTHORS: STANLEY GOTSHALL AND TERENCE SOULE

Chapter 5 shows that a variety of stochastic training techniques successfully single-cycle and double-cycle motion in the SNMS model. This chapter generalizes this goal by training the SNMS model to achieve multiple tasks and multiple speeds. The purpose of these experiments is to test these multi-goal models on unfamiliar input patterns and unfamiliar tasks to establish that the SNMS model is stable.

6.1 Abstract

Simulations for control systems are increasingly important in understanding the behavior of complex biomechanical systems such as a neuroprosthetic arm. Detailed models help researchers answer questions about the behavior of these systems, such as a neuroprosthetic arm, in typical scenarios and in situations for which the controller was not specifically designed or trained. To yield meaningful behavior, simulations of these systems often include hundreds of fine-tuned parameters that correspond to biological components and characteristics.

This paper demonstrates the effectiveness of genetic algorithms in training a neuromuscular model of the human arm. In particular, we test the stability of trained instances of the system with respect to unfamiliar control input patterns and untrained loads. The results show that small changes in the input frequency and arm load will result in small changes in velocity, demonstrating that the system can reasonably accommodate unfamiliar circumstances. This is a critical feature for biologically based nonlinear control systems.

6.2 Introduction and Motivation

Neural networks in the spinal cord provide the body with a remarkably precise system for control. Motor neurons and muscle fibers exchange electrical signals to produce controlled anatomical movements throughout the body. Thus, designing a biologically plausible neural controller based on the biological system would allow researchers to harness the properties of biological systems that are uniquely suited to control complex mechanical systems.

The nervous system has an unmatched ability to control highly nonlinear systems, i.e. skeletal muscles. Muscles consist of thousands of muscle fibers that respond to electrochemical signals from the spinal cord to move and stabilize the body [34]. In turn, feedback signals from muscle fibers allow the spinal cord to respond and activate muscle groups in order to generate smooth and precise motions. By modeling control systems after the spinal cord, we increase our understanding of the role and importance of certain neural pathways in the nervous system. It also opens the door to developing prosthetic devices that interpret brain signals the same way the spinal cord does and designing robots with movement abilities comparable to humans.

Designing this type of system also potentially makes it possible to hypotheses regarding motor control, such as how motor control commands are encoded in the brain. Researchers are far from reaching a consensus on how motion is encoded in the primary motor cortex (M1), however there is some agreement that it must directly correspond with some kinematic property of the limb, such as position ad direction of movement. Certain studies strongly support the hypothesis that M1 firing is primarily correlated with direction and speed of motion based on monkey reaching tasks [33, 44] However, other studies show that M1 firing is correlated with arm position [39], acceleration [16], target position [20], distance to target [18], and several other variables [62]. Typically, those who support the latter hypothesis assert that M1 codes for motion around single joints and that the apparent reach-direction code is an epiphenomenon due partly to the physical restrictions of the musculoskeletal system. A sufficiently detailed model could be used to address these questions

However, from a systems control perspective, the most important goal is to design a

system that can process input control patterns, including unfamiliar ones, to produce stable and predictable results. Our goal is to create a stable system by evolving a system capable of multiple tasks. In this paper we use a GA to train the SNMS model with two respective sets of tasks. The first set is trained to move at different speeds controlled at different input frequencies. The second set is trained at a fixed speed with three different loads. We hypothesize that small changes to SNMS systems, such as slightly different input frequencies and arm weights, will result in small changes in behavior. This would prove that a GA does not over-train the system to only respond to familiar inputs and tasks. Rather, it produces a stable system that generalizes its behavior for a variety of input frequencies and loads.

6.3 Background and Previous Work

Artificial neural networks use individual processing units, neurons, to mimic the processing power of the nervous system. Traditional connectionist neural networks consist of neurons that compute static functions of their inputs [11,42,71] which, in the biological sense, represents an activity level (i.e. frequency) for that neuron. Networks consisting of these neurons are effective for many applications such as classification, prediction, and control. However, as our understanding of the nervous system increases, more biologically realistic neuron models have emerged, most prominently spiking neurons. Spiking neurons more accurately model the temporal dynamics of biological neurons. Thus, networks of spiking neurons can more accurately approximate the signal processing abilities of neural networks in the brain or spinal cord.

Although spiking neural networks are well suited for biological applications, researchers also use them for various types of problems commonly solved with traditional neural networks [17,50]. Furthermore, spiking neurons are computationally more powerful than traditional neurons [41]. Thus, practitioners can construct small networks of spiking neurons to accomplish the same task as a traditional network of greater size. This is done in the hope that the smaller number of synaptic weights will ease the training process.

However, training spiking neural networks requires innovative techniques.

Traditional feed-forward neural networks are trainable via back-propagation or Hebbian learning. Unfortunately, these training techniques cannot accommodate the temporal characteristics of spiking neural networks. Fortunately, researchers are developing a new generation of neural network training techniques. Instead of training numeric input-output patterns, these methods use mathematical methods to train temporal spiking patterns [36,52]. These techniques are useful in pattern recognition and control system problems, but only when the objective is to train a known input/output spiking pattern. However, in many important problems the output patterns needed to solve the problem are not known. This is often the case in control systems because the goal is quantified in terms of a desired motion and not a particular network output pattern. These scenarios need a new training approach that does not depend on practitioners knowing the input/output patterns a priori. GAs are a promising technique for training these systems because GAs need only to assign a single number, in this case the error from the target, to globally search the solution space. In previous research [23,24] we have shown that GAs are effective in training increasingly complex SNMS models for various motion tasks. However, the issue of stability with respect to unfamiliar inputs and loads was not addressed. Thus in this paper we train the SNMS model with a GA to carry out multiple tasks and test the system's stability on unfamiliar inputs and tasks.

6.4 Models

The models used in this chapter are omitted here and described in Chapter 4. This chapter uses a modified version of the model in Chapter 4. Each α-MN's threshold now equals the evolved value of the corresponding force multiplier. This relationship more accurately reflects the relationship between α-MN size and muscle fiber strength. Also, because significantly more CPU time is required for the experiments in this chapter, the second descending synfire node is removed and one synapse from the last descending synfire node to the γ-MN is removed, which presumably reduces the search space during training and thus reduces the CPU time needed. In total, this modification removes 120 synapses and 36 neurons from the model. Note that the triceps does not receive descending input in these experiments. Thus, the removal of the synfire nodes in

the triceps networks does not affect model functionality.

6.5 Experiments

These experiments use a steady-state GA for training. The GA maintains a population of individuals and replaces the two least fit individuals with two offspring during each iteration. For the real-value segments of the individuals, recombination is performed via arithmetic crossover [12] added to a small normal random variable as shown in Table 6.1. Uniform crossover is used for the binary components. After an individual is generated via crossover, each allele in the offspring then undergoes mutation with probability p, shown in Table 6.1.

The GA trains three distinct parameter types in the model: The 1) synaptic weights, 2) strengths of muscle fibers, and 3) binary selectors for neural subnets. All individuals in the GA is randomly initialized in the same manner: synaptic weights are randomly selected uniformly in the interval [0,4] (excitatory) and [-4,0] (inhibitory), fiber strengths in the interval [0,5], and subnet selectors in the uniform binary interval [0,1]. The GA operators are closed over the respective initialization intervals. In biological systems, stronger muscle fibers are innervated by larger α-MNs which require more presynaptic activity to fire. Thus, as a first order approximation, each α-MN uses its evolved force multiplier as its threshold value. All other neurons have a threshold of 1 as indicated in Equation 4.2.

The GA evaluates fitness of potential solutions with the equation

$$F = -\sum_{all\ t}(\Theta(t) - target(t))^2 \tag{6.1}$$

where $\Theta(t)$ is the angle between the movable forearm and the fixed upper arm at time t, and $target(t)$ is the target angle at time t. The fitness function measures the error from the target trace shown in Figure 6.1 and similar traces with different slopes for different velocities. In these experiments, the total fitness is the sum of each individual test case, for example each case for each frequency.

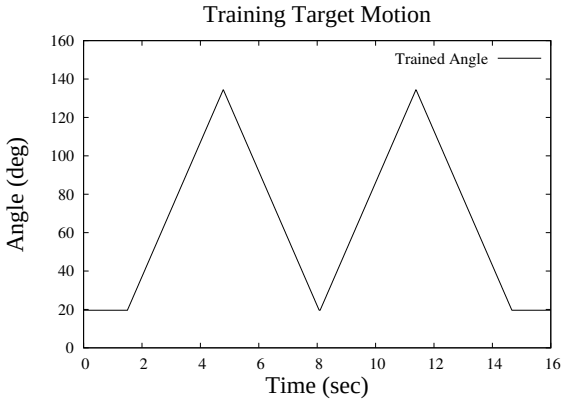


Figure 6.1: Starting at 0 seconds, no descending pathways receive an input pattern. At 1.5 seconds, the input pattern for upward motion is sent to a subset the biceps' subnets, as chosen by the training algorithm. The signals across each input neuron is phase shifted evenly across a window of $\frac{1}{freq}$. At a given frequency, the inputs to the subnets are out of phase and do not overlap during a given timestep unless the frequency is sufficiently high. At the peak of motion the biceps network receives the input for the downward motion. The target behavior shown is for the $45\frac{deg}{sec}$ case.

Table 6.1: GA parameters

Parameter	GA
Trials	50
Fitness Evaluations	60k
Population Size	30
Selection Type	Tournament (size 1)
Crossover Probability	1.0
Crossover Type (reals)	Arithmetic + N(0,0.2) [12]
Crossover Type (bits)	Uniform
Mutation Probability	1/dimensions
Mutation Type (reals)	N(0,0.2)
Mutation Type (bits)	bit-flip

6.5.1 *Varying Frequency With Speed*

The first experiment set involves training the SNMS model to raise and lower the arm at different speeds as determined by different input frequencies to the network. In this set, each system is trained on the criteria of raising and lowering the forearm at three different speeds; 35 $\frac{deg}{sec}$ upward and 35 $\frac{deg}{sec}$ downward, 45 $\frac{deg}{sec}$ upward and 45 $\frac{deg}{sec}$ downward, and 55 $\frac{deg}{sec}$ upward and 55 $\frac{deg}{sec}$ downward. These speeds are trailed with the following respective input frequencies; 25hz and 25hz, 30hz and 20hz, 35hz and 15hz. It is important to note that a given frequency for an upward motion has different meaning for the downward motion. The GA evolves which subnets receive input for the upward motion separately from the downward motion. Typically, the GA trains the system to use significantly more subnets for the upward motion because more muscle excitation is needed to move the arm upward as opposed to downward. Thus, for example, in the case of 35$\frac{deg}{sec}$ upward and downward the 25hz signal goes to one subset of the 6 input neurons and for the downward motion the 25hz signal goes to another, typically smaller, subset of the 6 input neurons.

After the GA trains a given system to move at the three speeds, the system is tested on a variety of unfamiliar inputs. The general assumption behind the paring of training frequencies with arm speeds is that higher input frequencies generate higher amounts of muscle excitation which roughly corresponds to the findings that frequencies in M1 are related to the direction [19] and speed of motion [44, 57]. Each system is trained to raise and lower the arm at the same absolute speed. Thus, the terms “faster” and “slower” will denote speeds during the upward and downward phases of motion. These terms are also used loosely under the assumption that higher input frequencies will result in higher levels of muscle contraction. Thus to generate a “faster” motion with respect to a given trained speed, the frequency is increased by 2Hz during the upward phase of motion and decreased by 2Hz during the downward phase. Likewise, to generate a “slower” motion with respect to a given trained speed, the frequency is lowered by 2Hz during the upward phase and increased by 2Hz during the downward phase.

6.5.2 Varying Arm Weight

The second experiment set involves training the model to move upward and downward at a single speed of 45 $\frac{deg}{sec}$ with 3 different arm weights. Each system is trained with a forearm mass of 0.5, 0.6, and 0.7 kg, and tested on every mass in the closed interval [0.4,0.8]kg in 0.05 kg increments.

6.6 Results

To determine overall performance we measure the average best fitness of the GA shown in Figure 6.2. To test the robustness we show best fit and average fit samples of both experiment sets on inputs they are not trained for. In general, the results show that relatively small changes to trained input frequencies and forearm loads results in small changes in behavior. All test case simulations are also extended 4 seconds in the case the arm takes unpredictably long to descend. In both experiment sets the trained motion is shown for both up/down cycles of motion, where the tests are only run for one cycle.

6.6.1 Fitness

The following data (Figure 6.2) shows the performance of the GA with respect to both experiment sets and shows the best and average fit solutions run with trained and untrained inputs and weights. As is typical for evolutionary algorithms, the GA improves its solutions most in the first half of the fun (before 15,000 iterations) then fine tunes the best solutions for the remainder of the evolution.

6.6.2 Effects of Altering Frequency

The following figures are samples of the best fit solution out of the 50 runs and a solution representative of the average best fitness at the end of evolution. The figures show the trained behavior with the behavior when tested with unfamiliar frequencies. For the solutions trained to move the arm at various speeds, the best solution has a fitness of -115.99 and the average solution has a fitness of -187.52. Both solutions

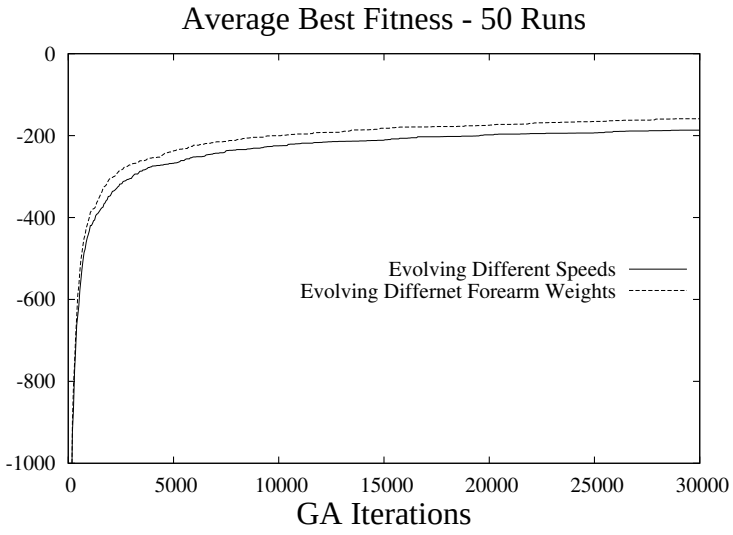


Figure 6.2: The GA evolves solutions to both training sets achieving fitnesses, on average, of -186.72($\pm$39.35) and -161.84($\pm$40.56) for the sets training varying speed/frequency and weight, respectively, at the end of training.

generate similar behavior, but overall the average solution deviates significantly more from the target over all three speeds.

In general, the GA successfully trains the system to match the target motion in both sample cases for each pair of input frequencies for the upward and downward motion. The best fit individual shows no unusual aberrations during the upward or downward motion nor irregularities during the change in direction (Figures 6.4(a) and 6.5(a)), except for the 35 $\frac{deg}{sec}$ case (Figure 6.3(a)) which overshoots the target at the peak and has slightly irregular motion. The average fit individual performs nearly as well as the best fit solution but has marginally larger deviations from the target motion (Figures 6.6(a), 6.7(a), and 6.8(a)). It is important to note that the total fitness value for a system is the sum of fitnesses over each of the three test cases.

Interestingly, each instance of using untrained frequencies results in small to moderate deviations from the corresponding trained behavior. When decreasing the speed of both phases of motion, upward and downward, the best fit solution, in all three cases, decreases its rate of descent (Figures 6.6(b), 6.7(b), and 6.8(b). However, the same solution seems to inhibit faster motion for the ”faster” input pattern (Figures 6.3(b),

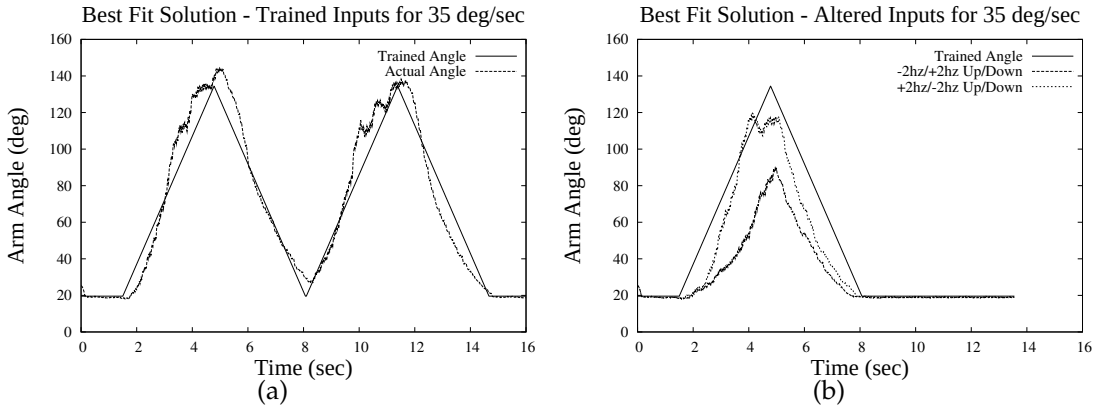


Figure 6.3: (a)Trained behavior with 25hz/25hz for up/down motion for 35 deg/sec. (b)Test cases using 27hz/23hz and 23hz/27hz for up/down motion.

6.4(b), and 6.5(b)). The most notable instance of this is Figure 6.3 where the higher input frequency during the upward motion is delayed and the arm's peak angle is notably lower than the trained behavior. The average fit solution on the other hand appears to be more susceptible to larger changes in velocity. Each pair of altered inputs results in faster and slower movements corresponding to the "faster" and "slower" frequencies. However, both altered inputs for the $55\frac{deg}{sec}$ case (Figure 6.6(b)) move slightly faster than the trained counterpart and overshoot the target. In general, each instance of using untrained frequencies results in relatively smooth motion during the entire motion suggesting that the system is robust and does not require training on all possible frequencies.

6.6.3 Effects of Changing Forearm Weight

The best individual trained for multiple forearm weights has a fitness of -82.48 where average (Figure 6.2) samples 1 and 2 have fitness values of -162.08 and -160.80, respectively. Similarly to the previous experiment set, the GA also successfully trains the system to move the arm at a fixed speed although the arm changes weight between trials. For each of the 3 the trained cases, with only minor exceptions, lighter arm weights approach the peak angle faster than heavy weights (Figures 6.9(a), 6.10(a), and 6.11(a)). In general, for the test cases the heavier weights cause the motion to lag and

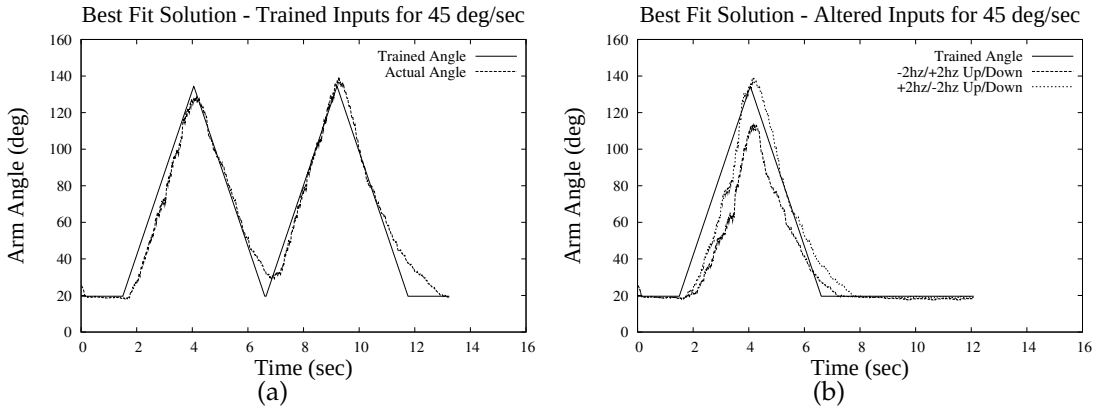


Figure 6.4: (a)Trained behavior with 30hz/20hz for up/down motion for 45 deg/sec. (b)Test cases using 32hz/18hz and 28hz/22hz for up/down motion.

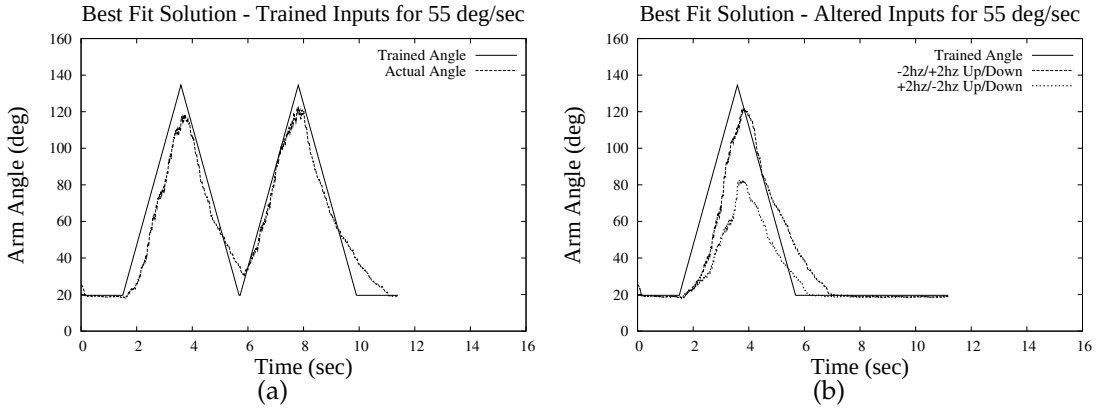


Figure 6.5: (a)Trained behavior with 35hz/15hz for up/down motion for 55 deg/sec. (b)Test cases using 37hz/13hz and 33hz/17hz for up/down motion.

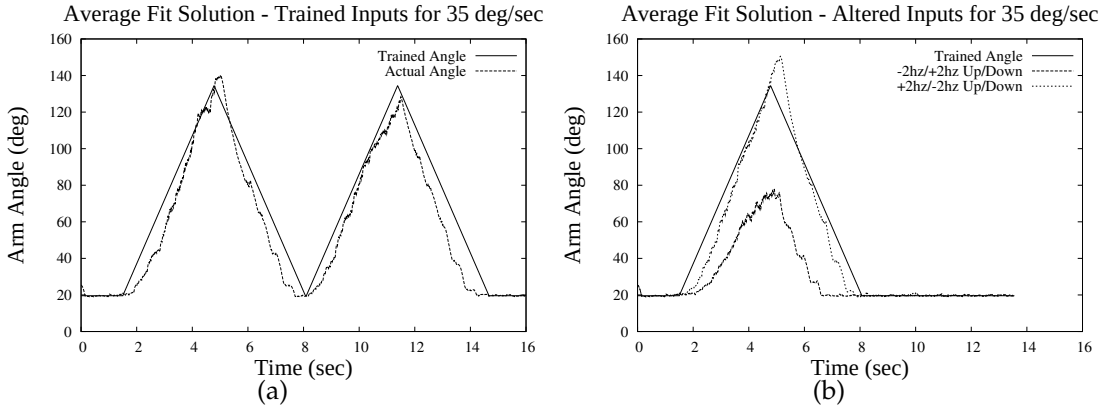


Figure 6.6: (a)Trained behavior with 25hz/25hz for up/down motion for 35 deg/sec. (b)Test cases using 27hz/23hz and 23hz/27hz for up/down motion.

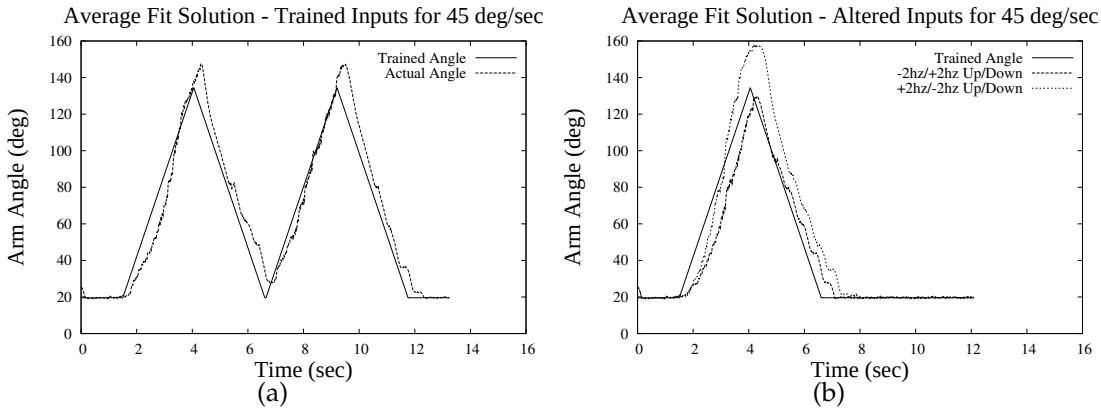


Figure 6.7: (a)Trained behavior with 30hz/20hz for up/down motion for 45 deg/sec. (b)Test cases using 32hz/18hz and 28hz/22hz for up/down motion.

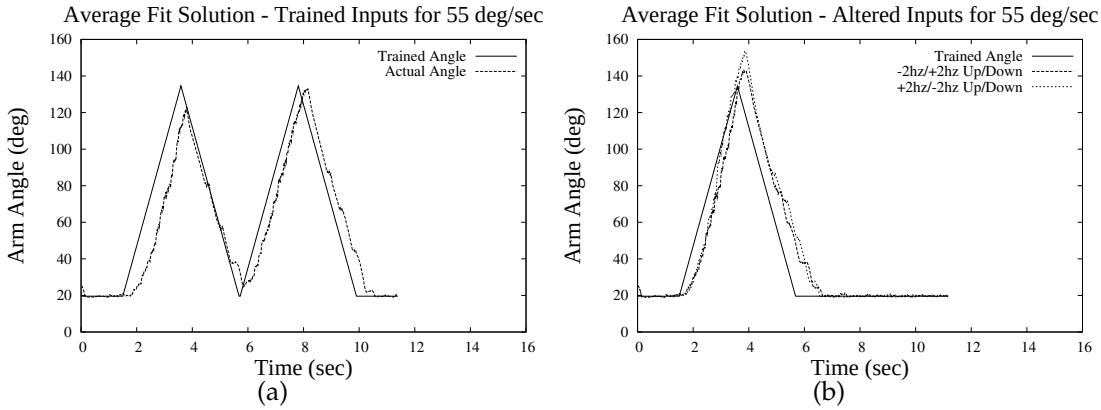


Figure 6.8: (a)Trained behavior with 35hz/15hz for up/down motion for 55 deg/sec. (b)Test cases using 37hz/13hz and 33hz/17hz for up/down motion.

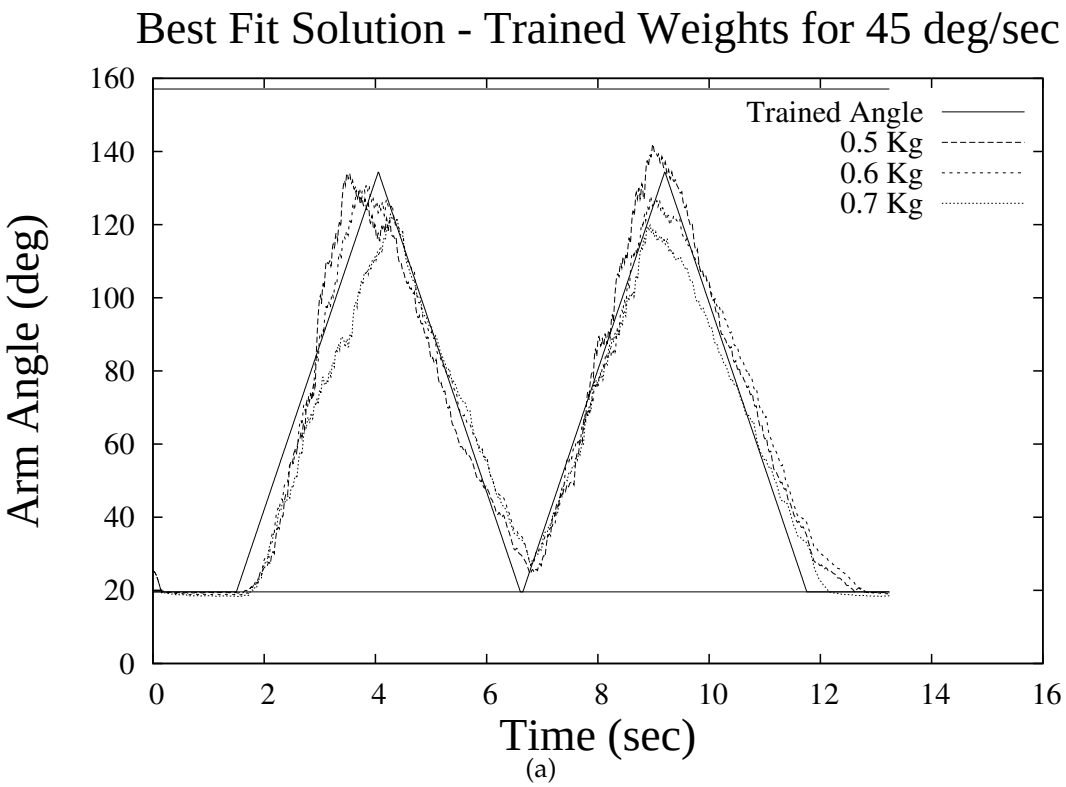
slow down and the light weights cause the motion to speedup and overshoot the peak of the target. However, just as higher input frequencies do not always result in higher levels of contractile force, lighter weights do not always result in faster upward motion. Interestingly, Figure 6.10(b) clearly shows that the test cases with 0.45kg approach the target's peak angle faster than the 0.4kg test case, although the 0.4 kg case does move faster than the 0.5kg. This clearly indicates a nonlinear relationship between weight and velocity. Some behaviors also encounter the maximum angle spring (Figures 6.10(b) and 6.11(b)). Similar nonlinear relationships also appear in Figure 6.9 where the 0.65kg test case approaches the peak angle faster than the 0.55kg test case. Figure 6.11(b) on the other hand yields monotonic results. That is, the upward motion consistently slows as the weight increases.

6.7 Discussion

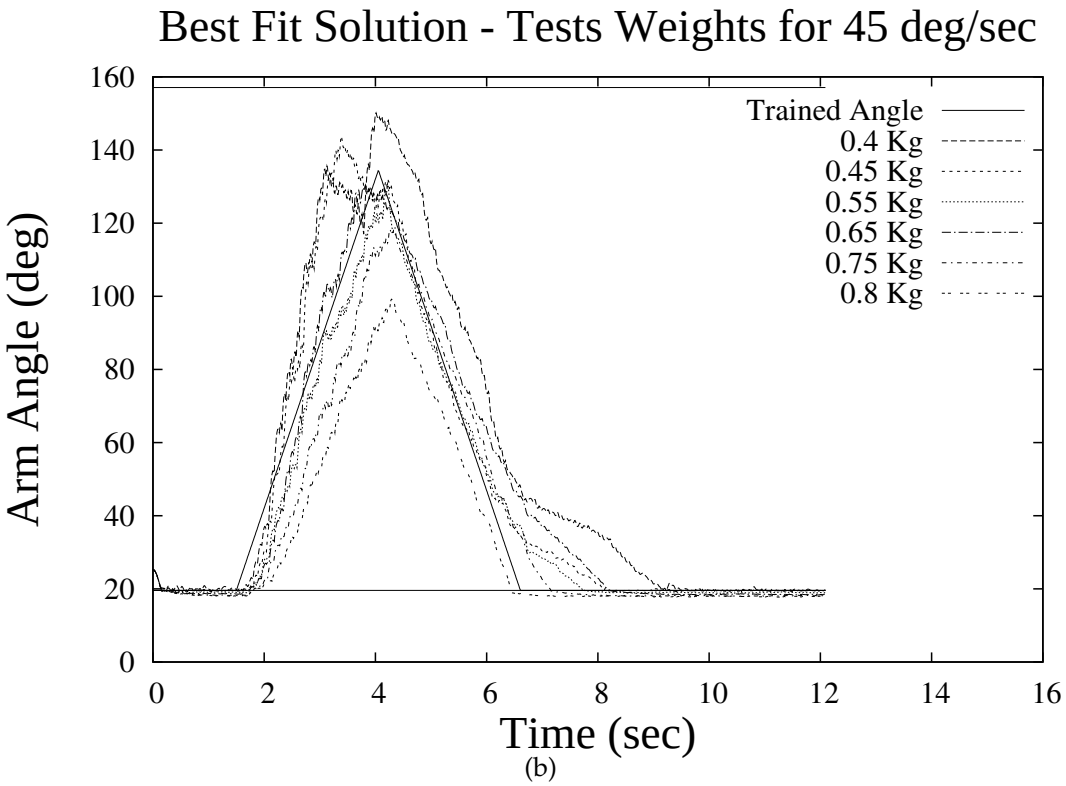
In practice, training any type of neural network may result in a system that only recognizes inputs patterns it is trained for. If trained for tasks such as classification or prediction, these networks, sometimes referred to as over-trained networks, do not perform well on unfamiliar inputs. One of several possible approaches to address this is to add a small amount of noise to the training cases to keep the network from classifying only the test cases correctly. This overtraining problem takes on new meaning in applications with pulsed neural networks. Instead of dealing with a system that generates static outputs from static inputs, we now have a highly asynchronous system capable of processing information in a different way. This means that it may be even more difficult to predict the system's behavior or untrained inputs making the problems caused by overtraining more significant and more difficult to solve.

6.7.1 Unexpected Frequencies

The SNMS's stable behavior when receiving unfamiliar input frequencies has interesting implications in traditional control theory as well as fault tolerant systems. A component failure in system using the SNMS model could result in unexpected or irregular patterns in muscle/actuator contraction. If a system is designed to lift an object at one of n speeds

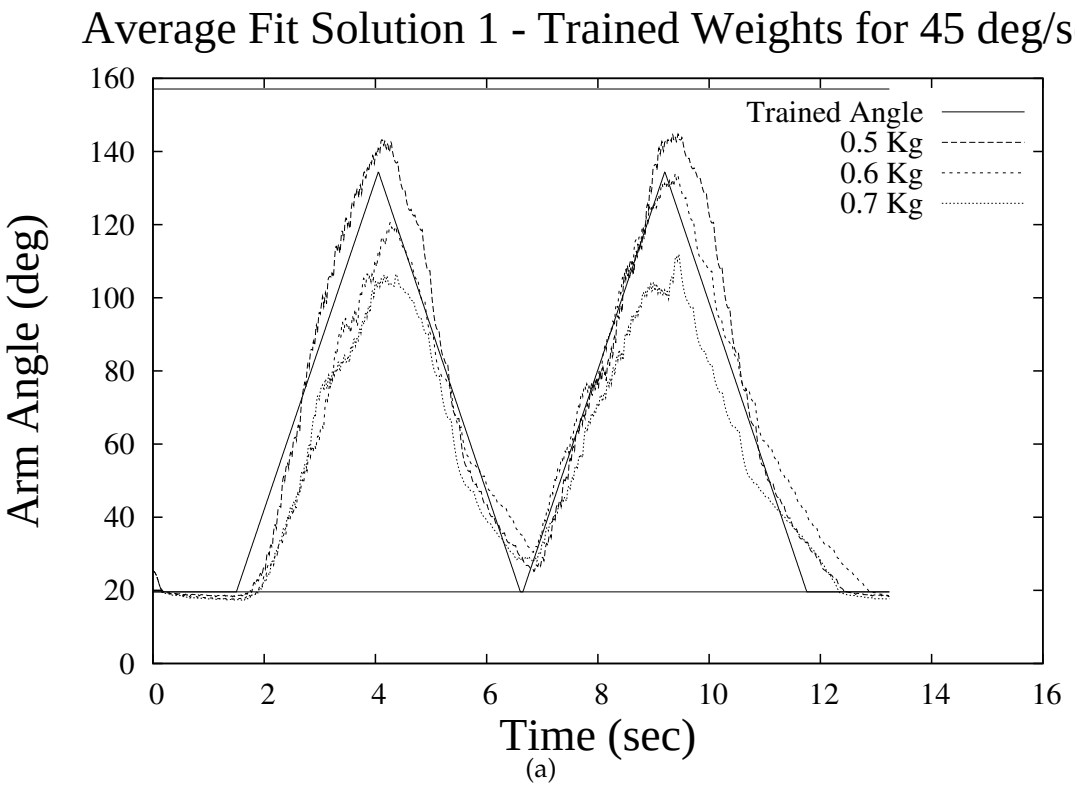

Best Fit Solution - Trained Weights for 45 deg/sec
Arm Angle (deg)
Time (sec)
Trained Angle
0.5 Kg
0.6 Kg
0.7 Kg
0
20
40
60
80
100
120
140
160
0
2
4
6
8
10
12
14
16

(a)

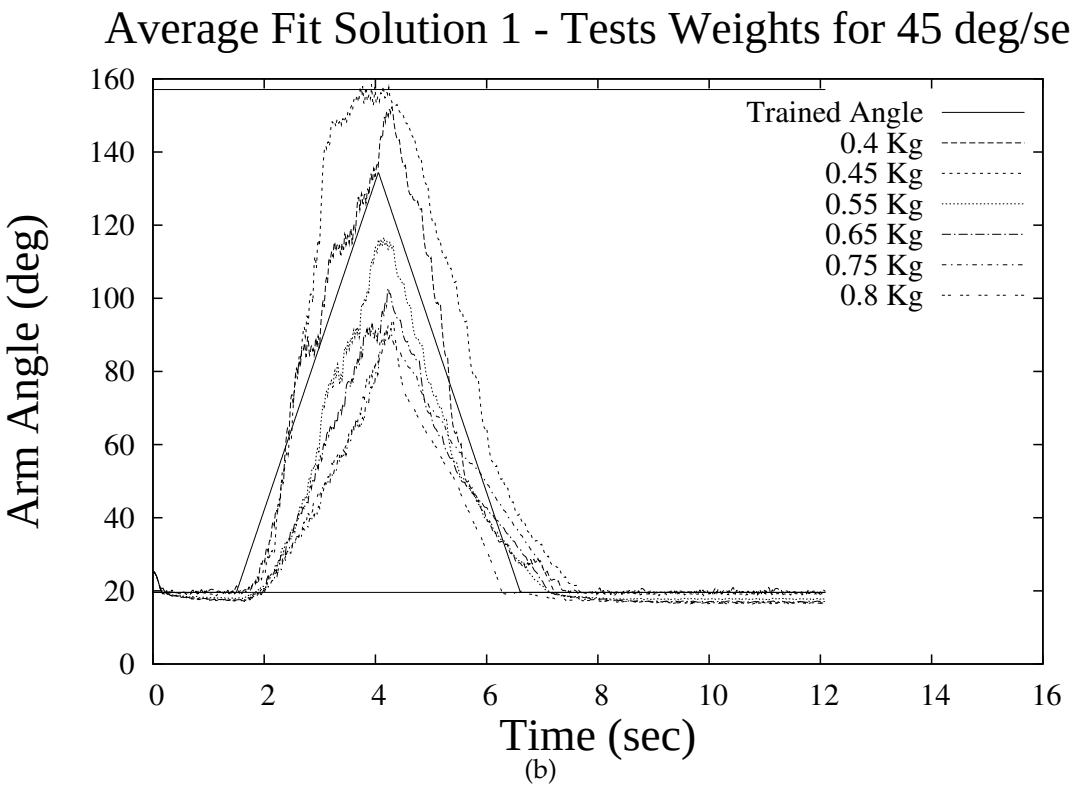

Best Fit Solution - Tests Weights for 45 deg/sec
Arm Angle (deg)
Time (sec)
Trained Angle
0.4 Kg
0.45 Kg
0.55 Kg
0.65 Kg
0.75 Kg
0.8 Kg
0
20
40
60
80
100
120
140
160
0
2
4
6
8
10
12
14
16

(b)

Figure 6.9:

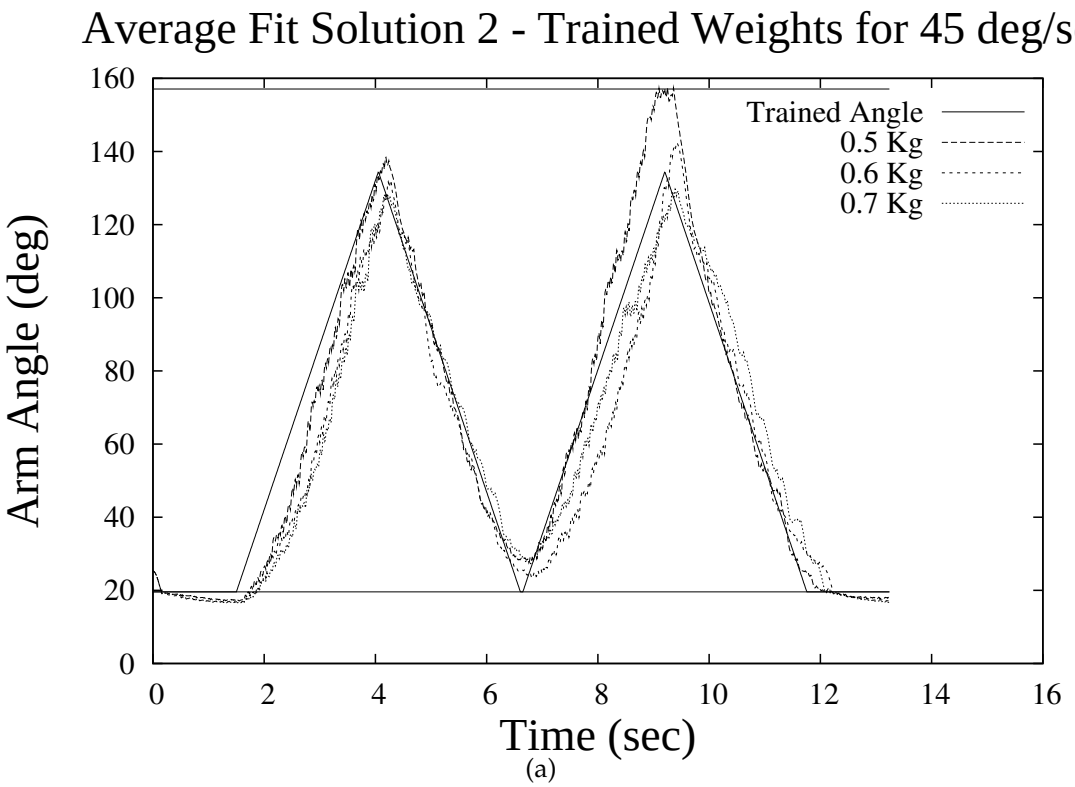


(a)

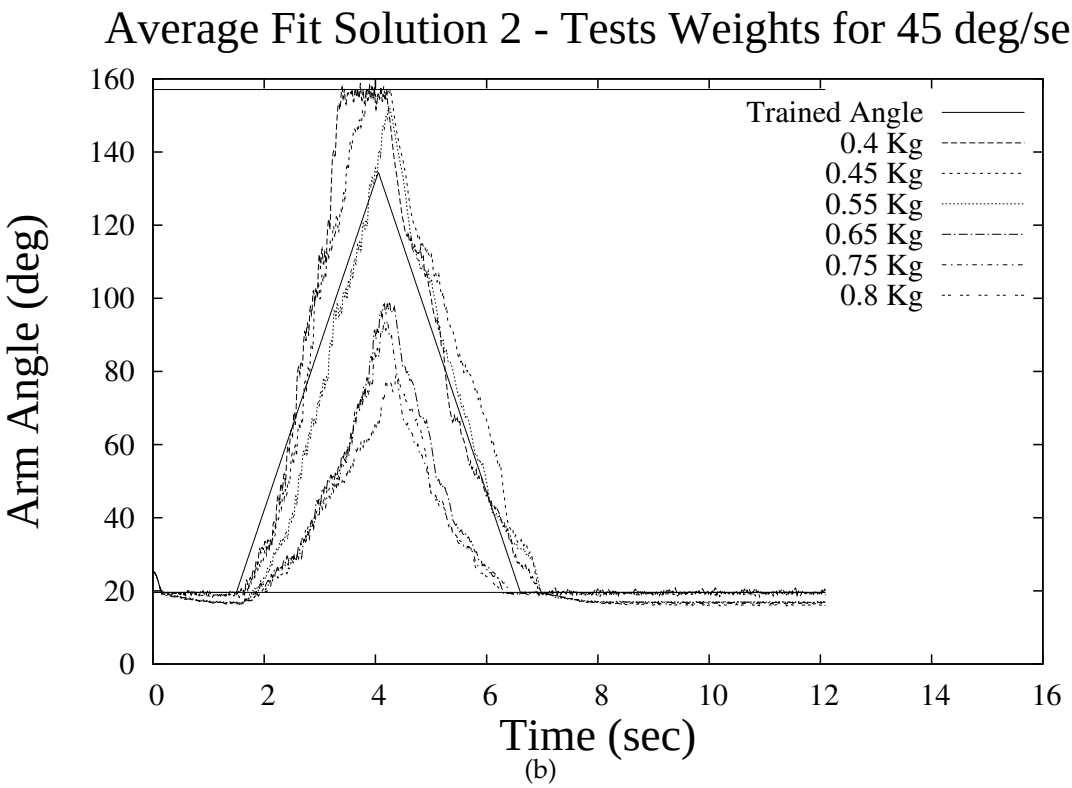


(b)

Figure 6.10:



(a)



(b)

Figure 6.11:

with $n\prime$ corresponding input signals, a malfunction would have to occur for the system to receive an unrecognized input. If the system does not recognize that the input is unfamiliar, it may process that signal and the system may become unstable. In using a biologically-based controller, it is useful to know that a slightly altered input signal, one that has not been trained or designed for, will most likely not result in unstable behavior or behavior far outside the predicted range.

Our system is relatively robust with respect to inputs. A notable example of this robustness is the behavior of the best fit solution at altered frequencies (Figures 6.3(b), 6.4(b), 6.5(b)). In each of the three cases the higher input frequency has little effect on the upward velocity of the forearm or slows its movement and undershoots the target as in Figure 6.3(b). From the interval [4,5] seconds the forearm appears to be competing with some evolved mechanism to keep it from exceeding 120 degrees. That is, the best fit individual appears to have evolved a mechanism to prevent excessive amounts of contraction. A likely candidate for this is mechanism the Renshaw cell, which is generally known to have protective/modulatory effect on α-MN firing patterns. This mechanism is particularly useful because variations in motion that are slower than expected are generally safer than those that are faster than expected.

6.7.2 Unexpected Weights

It is interesting to note that for a given input frequency, lighter weights do not always result in faster ascending movement. This shows that for a given input the system is doing more than generating constant quantities of force over time. The faster motions trigger nonlinearities in the motion that appear to inhibit excessive amounts of muscle contraction (Figures 6.3(b) and 6.5(b)). The SNMS model includes mechanisms that account for this type of behavior in biology including the Renshaw cell and the Ia inhibitory interneuron. These pathways, affected directly by descending input signals, muscle fiber's length, and contractile velocity, also allow for various compensation behaviors that incorporate the position and speed of movement. With this information the system may develop a "preferred" set of weights, for a given target motion, which do not correspond to a linear function of the forearm's weight. On the other hand having

a "preferred" weight set may be the result of testing on a less fit solution. The best fit solution does not show such a dramatic difference between 0.4kg and 0.45kg. In fact, the 0.4kg test case moves upward slightly faster than the 0.45kg case (Figure 6.9).

In applications such as prosthetic devices or a mechanical arm on an assembly line, it is important that unfamiliar tasks, such as changes in forearm weight, do not result in erratic behavior. Trained instances of the SNMS model yield smooth behavior over a range of familiar and unfamiliar inputs, suggesting that this type of control system is a good candidate for controlling various nonlinear systems. It is reasonable to hypothesize that as the model grows with more biological components, behaviors will become more controlled for both trained and untrained cases, although it may also require more extensive training.

6.7.3 Future Work

In a simulation there is less noise and fewer unpredictable nonlinearities than in a physical system, thus future work includes using the SNMS model to control a physical robotic arm as well as addressing key biological questions regarding spinal cord and how motor commands are encoded in M1. Using the SNMS will likely require training on a physical arm unless the arm can be modeled sufficiently well in the SNMS model. If the nonlinearities in a physical arm fit well with a set of equations, the SNMS model can use these equations in place of Hill's model and training can take place mostly, if not completely, in simulation. Ideally, little training would be needed on the physical arm to minimize training time.

Further development of the SNMS model also opens the door for answering questions about the neuromuscular system. One task suited to this model is investigating the importance and role of specific mechanisms and pathways in the spinal cord by modeling diseases and injuries. The SNMS can simulate these scenarios by altering parameters in individual neurons or parameters in the muscle fibers to affect their responsiveness to stimulation.

Another task of interest is testing hypotheses regarding how motor commands are coded in the primary motor cortex (M1) of the brain as described in the introduction.

With further development, the SNMS model could be used to train motion with input patterns corresponding to various theories as to M1 motor commands are coded, such as the “preferred direction” hypothesis. This hypothesis emphasizes the importance of the velocity vector of a limb in motion (e.g. hand) in M1 coding. Certain coding schemes may favor more biologically plausible behaviors than others increasing its effectiveness as a control system. Thus, additional research in this area can yield improved stability of biologically based robotic systems.

6.8 Conclusions

This paper measures the inherent stability, with respect to unfamiliar input control signals and forearm weights, of a SNMS system model trained with a genetic algorithm. The results show that input frequencies that are slightly different from those the model is trained on result in small to moderate variations in speed while still achieving relatively smooth motion. Similarly unfamiliar forearm weights results in stable and smooth motion at different speeds. In all test cases the motion remains relatively smooth and the arm returns to the stable resting point at the end of the simulation. This stable behavior on untrained cases demonstrates the robustness of the SNMS model and advances the possibility of successfully developing more complex models and tasks for biologically-based control systems.

Chapter 7

CONCLUSIONS

The results of this dissertation show that multiple types of stochastic training algorithms, i.e. genetic algorithms and particle swarm optimizers effectively train the SNMS model to move the arm as trained. Furthermore, the training process causes biologically plausible behaviors to emerge from model, i.e. tonic tension, biceps/triceps coactivation patterns, and recruitment-like behaviors. The results also demonstrate the inherent stability of the model given that small changes to familiar control signals or tasks results in small to moderate changes in behavior. These advances pave the way for more controlled virtual experiments in the nervous system to help answer questions as to the role and importance of certain components in motor control applicable to disorders of the nervous system and medical rehabilitation. The SNMS model has potential for a variety of applications in evolutionary robotics and neuroprosthetic devices which involve designing robust, stable, and fault tolerant control systems.

7.1 Future Work

The SNMS model in its current form allows for a great deal of flexibility to increase the scale of the model, its complexity, and simulate a variety of neuromuscular phenomena and diseases.

7.1.1 Neuron Models and Networks

In it's present form, the SNMS model requires that each muscle fiber receive pulsed inputs over time. This feature of the model reflects biological all-or-none behavior in muscle fibers when stimulated by α-MNs. At a given timestep, the α-MN triggers the contraction many muscle fibers and generates force in the muscle. The presented form of Hill's muscle model only requires that the neuron triggering a given muscle fiber be

all-or-none, thus the model can accommodate a variety of other pulsed neurons. In future versions of the model it may be beneficial to use neuron models designed to produce biological spiking patterns, such as Rulkov's [53] new neuron model, to allow for more biologically realistic muscle contraction patterns.

Biological neural networks are also inherently adaptive. As the body learns motor functions, synapses in the central nervous system change to alter signal pathways. Researchers have also presented evidence that cortical structures in neocortex, the newest part of the brain from an evolutionary perspective, dynamically relink themselves [61] over time. The most common analog of this in artificial neural networks is known as adaptive neural networks. In traditional neural networks, adaptation means that synaptic weights change their value during the learning process. A well known technique, Hebbian learning, accomplishes this by strengthening synaptic connections between two neurons proportionally to the stimulation levels of the presynaptic neuron on the postsynaptic neuron.

However, there is no known method for pulsed neural networks to learn adaptively. In theory, the learning process can change the synaptic strength between pulsed neurons, but traditional Hebbian learning was not made to accommodate the properties of pulsed neural networks. One potential way to allow pulsed networks to adapt is with some sort of actor-critic architecture in a reinforcement learning scheme. This is a potentially fruitful area of artificial neural networks.

7.1.2 Joint and Muscle Enhancements

The SNMS model also allows for a great deal of flexibility with respect to the number of muscle fibers, orientation during movement, and many other variables. Chapter 4 demonstrates how a SNMS model with biologically plausible components can yield many types of realistic behaviors. Thus it is reasonable to hypothesize that further additions such as other arm muscles, more types of afferent feedback, and a larger scale neural network could yield new types of emergent behavior. Behaviors of interest include the size principle of motor unit recruitment, muscle fatigability, and other phenomena directly observed in biology. Furthermore, modeling friction in the joint and

simulating the geometry human arm will assist in more detailed applications in prosthetics.

7.1.3 Disorders and Injuries

The ability to perform controlled virtual experiments in the SNMS also opens the door to modeling the neural behavior of motor control disorders. Each nervous system disorder cripples the nervous system in a unique way, inhibiting fine motor control, initiation of movement, or even sensory function. With certain neuron models, researchers can simulate the altered behavior of individual neurons by modifying behavioral parameters.

Multiple sclerosis is a degenerative disorder that causes the gradual destruction of the axon's myelin sheath. This causes a slowdown in the signal propagation from one neuron to the next which leads to many symptoms including muscle weakness, abnormal muscle spasms, and changes in sensation. These types of symptoms can be directly observed, quantitatively, in the SNMS model in the forms of altered contractile force, sporadic contractions, and altered feedback patterns, respectively. Observing some symptoms may require adding pathways, particularly sensory pathways if investigating changes in sensation. This technique can also generalize to other nervous system disorders such as Parkinson's disease, Huntington's disease, and muscular dystrophy.

Another potential application of the SNMS model is investigating which components of the spinal cord are most important during rehabilitation from injury. Observing this type of rehabilitation may require implementation of the previously mentioned adaptive network, but is nevertheless plausible by manipulating trained instances of the model to mimic the properties of a given disorder or injury. Nerve damage, that is dead cells, can be modeled by inhibiting synaptic input to a neuron and a fractured bone by altering the geometry of the joint model. This could apply to a variety of other situations including muscle or tendon damage and excessive muscle fatigue.

The most important result of this research may be that it proves that even large scale, extremely complex, highly nonlinear systems can be successfully trained using stochastic approaches. These techniques lead to both relatively stable solutions and

generate complex control strategies. In this research GAs, PSOs, and BSOs, generated complex control strategies that mimic those evolved over millions of years of biological evolution. Thus, this research suggests that a vast range of control problems that previously may have been considered intractable are actually solvable.

7.2 Acknowledgements

This work was supported in part by the National Science Foundation under award number 0243885. These experiments were performed on a Beowulf Cluster built with funds from NSF grant EPS-80935 and a generous hardware donation from Micron Technology.

Appendix A

EVOLUTION OF BIOLOGICALLY PLAUSIBLE BEHAVIOR IN A MODEL OF THE SPINO-NEUROMUSCULAR SYSTEM [1]

Stanley Gotshall
Department of Computer Science
University of Idaho, Moscow, ID

Despite considerable advances in knowledge regarding the structure and function of the spino-neuromuscular system (SNMS) [5] there is still a need to expand our understanding of the role of key neural mechanisms involved in initiating and controlling movement and compensating for injury and disease. If a movement is disrupted, it is not known which neural pathways are most significant in correcting for the injury. Many researchers have used neuromuscular models based on known neurophysiology to explore questions regarding the role of key neural mechanisms in motor control by assuming that selected movements are optimal with respect to some criterion [3, 6]. In contrast, research focused on modeling the neural circuitry of the SNMS has generally been used to demonstrate that a particular model, with a limited number of specific neural pathways, can generate biologically realistic neural activity or biologically realistic movements. Such research is extremely useful in showing that the particular neural pathway(s) encoded in the model can plausibly account for particular patterns of neural activity or particular movements. However, because of the diversity and redundancy of neural pathways, there are multiple neural pathways that can plausibly account for any particular observed behavior. Thus, showing that a particular neural pathway can generate a biologically realistic movement in a model is not the same as showing that that pathway does generate the movement in an organism.

---

[1]THIS ABSTRACT AS BEEN ACCEPTED FOR PUBLICATION IN THE INTERNATIONAL CONFERENCE ON COGNITIVE AND NEURAL SYSTEMS 2006.

We have created a biologically plausible quantitative model of the SNMS and show how specific neural activity results in specific motions. Our SNMS model consists of a recurrent network of spiking neurons controlling a model of a portion of a simulated skeletal muscle system. The neural network and muscle model is trained with a genetic algorithm (GA) based on specific target behaviors. The primary advantages of using GAs are that they can train complex nonlinear control systems and produce robust solutions that arent overly sensitive to the exact parameters of the model.

Our experimental results show that GAs find innovative and biologically plausible solutions that utilize the key biological components in the model. Our results also show that biologically realistic neural patterns emerge as key pathways are added to the model. These patterns include a form of motor unit recruitment accompanied by variable firing rates of motoneurons to compensate for changes in torque. The addition of certain feedback pathways also decreases the training time to produce a given motion [23]. From this we conclude that the neuromuscular model trained with a GA is a reasonable way to model behavior in the SNMS. Furthermore, we conclude that GAs give researchers a flexible technique to train biologically realistic neuromuscular behavior and makes it possible to train increasingly complex neuromuscular system models.

Acknowledgments: These experiments were performed on a Beowulf cluster built with funds from NSF grant EPS-80935 and a generous hardware donation from Micron Technology.

Appendix B

HILL MODEL EQUATIONS

The following equations describe the second canonical form of A.V. Hill's muscle model with multiple contractile elements [65].

Table B.1: Hill model constants

l_0	$\sqrt{L_1^2 + L_2^2}$	resting muscle fiber length
a	4	damping element constant
b	$0.1 * l_0$	Scaled l_0(used for damping element and fiber length)
Δt	0.01 seconds	time step
τ	0.04 seconds	Q_C endplate potential time constant
τ_B	0.04 seconds	Q_C x_1 time constant
β	$8.3 grams/cm^2$	muscle density
μ	$\beta(e^{-\alpha})$	used in spring constant K_{SEC}
α	2	indirect K_{SEC} constant and PEC constant
α_s	2	K_{SEC} constant
C_0	8	end-plate potential upper bound

B.1 System-Level Equations

The total contractile force of the entire muscle is calculated as

$$T = \sum_{all\ \gamma} (T_{sec}^{\gamma} F^{\gamma}) + T_{pec} \tag{B.1}$$

where γ is a given motor unit, T_{sec}^{γ} is the force due to γ, F^{γ} is the force multiplier for γ, and T_{pec} is the force due to the parallel elastic component (tendon). The force multiplier for a given muscle fiber determines how much contractile force that fiber contributes to overall muscle contraction. To match the joint simulation, the total contractile force is calculated every 0.01 seconds.

The total tension due to a given motor unit is calculated as

$$T_{sec}^{\gamma} = (T_C^{\gamma} + T_B^{\gamma})u(x_2^{\gamma}) \tag{B.2}$$

where x_2 is the length of the parallel elastic component, T_C is the force due to the contractile element, and T_C is the force due to the damper element. For simplicity subsequent equations will omit the gamma superscript.

B.2 Contractile Element

Each of the following equations calculate quantities for a particular motor unit. The contractile force due to a single motor unit is calculated as

$$T_C = Q_C \cdot A(x_1/l_0) \tag{B.3}$$

where

$$A(x_1/l_0) = \begin{cases} 4(x_1/l_0) - 2.40 & , x_1 < 0.8l_0 \\ \frac{4}{3} \cdot (x_1/l_0) - \frac{1}{3.175} & , 0.8l_0 \leq x_1 < 0.95l_0 \\ 1 & , 0.95l_0 \leq x_1 \leq 1.05l_0 \\ -\frac{10}{7} \cdot (x_1/l_0) + 2.50 & , x_1 > 1.05l_0 \end{cases} \tag{B.4}$$

The variable x_1 corresponds to the current length of a given contractile element. Equation B.4 shows that muscle fibers generate maximum tension within about 5% of the resting length l_0.

Each calculation directly computing the next tension due to a given motor unit incorporates Q_C which approximates the electric endplate potential at a given muscle fiber's neuromuscular junction. Each junction is the equivalent of a synapse or muscle cell. This function is calculated as

$$Q_C(t) = Q_C(t - \Delta t)e^{-\Delta t/\tau} + p(t)C_0(1 - e^{-\Delta t/\tau}) \tag{B.5}$$

.

For convenience and functional purposes, τ = 0.04 seconds and C_0 = 8. The constant C_0 effectively acts as the upper asymptotic bound of the function Q_C which limits the neural excitation at the neuromuscular junction. The function p(t) = 1 if this motor unit is active at timestep t and p(t) = 0 otherwise.

B.3 Damper Element

The damper element for a given muscle fiber is calculated as T_B, where

$$T_B = \begin{cases} \frac{T_C+a}{b-\dot{x_1}} \cdot x_1 & , -v \leq \dot{x_1} < 0.1 \cdot v \\ \frac{0.1 \cdot (1+k)}{k-0.1} \cdot T_C & , \dot{x_1} \geq 0.1 \cdot v \\ -T_C & , \dot{x_1} \leq -v \end{cases} \tag{B.6}$$

where $v = b/k$ and $k = a/T_C$ where a and b are constants (Table B.1). In the case where T_C is very small, k is set to a large number to avoid division by zero.

The first case in Equation B.6 is in the form of a nonlinear viscous damper with damping coefficient B depending on velocity and contractile force T_C ([65] Page 13). The second case calculates T_B as a sliding friction element, while the third case cancels the force generated by the contractile element when the muscle contracts excessively fast.

B.4 Serial Elastic Component (SEC)

We determine the SEC spring constant used to calculate $\dot{x_1}$ in Equation B.9 as

$$K_{SEC} = \begin{cases} \frac{K_{SEC0}}{\alpha_s x_2} \cdot (e^{\alpha_s x_2} - 1) & , x_2 \geq 0 \\ 0 & , x_2 < 0 \end{cases} \tag{B.7}$$

$$B = \frac{T_C + a}{b - \dot{x_1}} \tag{B.8}$$

The instantaneous velocity of x_1, $\dot{x_1}$, is calculated as

$$\dot{x_1} = -\frac{K_{SEC}}{B} x_1 + \frac{K_{SEC}}{B} \cdot l - \frac{Q_C \cdot A(l/l_0)}{B} \tag{B.9}$$

where

$$x_1 = \begin{cases} l(t-\Delta t) & , \dot{x_1} \leq v \\ l(t-\Delta t) - (1+\zeta)Q_C(t)A((t-\Delta t)/l_0)K_{SEC} & , \dot{x_1} > 0.1 \cdot v \\ x_1(t-\Delta t) \cdot e^{-(\Delta t/\tau_B)} + (l(t-\Delta t) & \\ -\frac{\tau_B}{B}Q_C(t)A(l(t-\Delta t)/l_0)(1-e^{-\Delta t/\tau_B}) & , -v \leq \dot{x_1} < 0.1 \cdot v \end{cases} . \tag{B.10}$$

where $\tau_b = B/K_{SEC}$ and $\zeta = 0.1(1+k)/(k-0.1)$ where $k = a/T_C$ and $B = (T_C + a)/(b - \dot{x_1}(t-\Delta t))$. Similar to Equation B.6, the equation for x_1 has three cases which correspond to rates of contractile velocity.

B.5 Parallel Elastic Component (PEC)

The parallel elastic component, which corresponds to the tendon connection to each muscle fiber, is calculated as

$$T_{PEC} = K_{PEC0}\frac{(e^{\alpha \cdot (\Delta\lambda)} - 1)}{(\alpha \cdot \Delta\lambda)} \cdot \Delta l \cdot u(l - l_0) \tag{B.11}$$

where

$$K_{PEC0} == \frac{\alpha\beta A}{l_0} == \lim \Delta l \to 0 \tag{B.12}$$

$$l = x_1^\gamma + x_2^\gamma u(x_2^\gamma) \tag{B.13}$$

and

$$u(x) = \begin{cases} 1 & , x > 0 \\ 0.5 & , x = 0 \\ 0 & , x < 0 \end{cases} \tag{B.14}$$

BIBLIOGRAPHY

[1] M. Abeles, G. Hayton, and D. Lehmann. Modeling compositionality by dynamic binding of synfire chains. *Journal of Computational Neuroscience*, 17:179–201, 2004.

[2] J. E. Baker and D. D. Thomas. A thermodynamic muscle model and a chemical basis for a.v. hill's muscle equation. *Journal of Muscle Research and Cell Motility*, 21:335–344, 2000.

[3] R. Balasubramaniam and A. Feldman. Guiding movements without redundancy problems. *Coordination Dynamics, V.K. Jirsa and J.A.S. Kelso (Eds.)*, 2004.

[4] D. Boothe and A. Cohen. A model of limbed locomotion for a four muscle system. *Neurocomputing*, 44-46:743–752, 2002.

[5] A. Brown. *Organization in the Spinal Cord*. Springer-Verlag, New York, 1981.

[6] T. Buchanan, D. Lloyd, and T. Besier. Neuromusculoskeletal modeling: Estimation of muscle forces and joint movements and moments from measurements of neural command. *Journal of Applied Biomechanics*, 20:367–395, 2004.

[7] T. Bui, S. Cushing, D. Dewey, R. Eyffe, and P. Rose. Comparison of the morphological and electronic properties of renshaw cells, ia inhibitory interneurons, and motoneurons in the cat. *Journal of Neurophysiology*, 90:2900–2918, 2003.

[8] B. Cartling. Control of computational dynamics of coupled integrate-and-fire neurons. *Biological Cybernetics*, 78:383–395, 1997.

[9] C.-P. Chou and B. Hannaford. Study of human forearm posture maintenance with a physiologically based robotic arm and spinal level neural controller. *Biological Cybernetics*, 76:285–298, 1997.

[10] M. Clerc. Towards a deterministic and adaptive particle swarm optimization. In *Proceedings of the Congress on Evolutionary Computation*, pages 601–610, 1999.

[11] N. Durand and J.-M. Alliot. Neural nets trained by genetic algorithms for collision avoidance. *Applied Intelligence*, 13:205–213, 2000.

[12] A. Eiben and J. Smith. *Introduction to Evolutionary Computing*. Natural Computing. Springer, 1998.

[13] R. Enoka. Eccentric contractions require unique activation strategies by the nervous system. *Applied Physiology*, pages 2339–2346, 1996.

[14] R. Enoka. *Neuromechanical Basis of Human Movement*. Human Kinetics, 3rd edition, 2002.

[15] G. J. Ettema and K. Meijer. Muscle contraction history: Modified hill versus an exponential decay model. *Biological Cybernetics*, 83:491–500, 2000.

[16] D. Flament and J. Hore. Relations of motor cortex neural discharge to kinematics of passive and active elbow movements in the monkey. *Journal of Neurophysiology*, 60:1268–1284, 1988.

[17] D. Floreano and C. Mattiussi. Evolution of spiking neural controllers for autonomous vision-based robots. In *Proceedings of the International Symposium on Evolutionary Robotics From Intelligent Robotics to Artificial Life*, pages 38–61. Springer-Verlag, 2001.

[18] Q.-G. Fu, D. Flament, J. D. Coltz, and T. J. Ebner. Temporal encoding of movement kinematics in the discharge of primate primary motor and premotor neurons. *Journal of Neurophysiology*, 73:836–854, 1995.

[19] A. P. Georgopoulos, J. Kalaska, R. Caminiti, and J. Massey. On the relations between the direction of two-dimensional arm movements and cell discharges in primate motor cortex. *Journal of Neuroscience*, 20:1527–1537, 1982.

[20] A. P. Georgopoulos, J. T. Lurito, M. Petrides, A. B. Schwartz, and J. T. Massey. Mental rotation of the neuronal population vector. *Science*, 243:234–236, 1989.

[21] M. Giuglaiano, M. Bove, and M. Grattarola. Activity driven computational strategies of a dynamically regulated integrate-and-fire model neuron. *Journal of Computational Neuroscience*, 7:247–254, 1999.

[22] S. Gotshall, K. Browder, T. Soule, and R. Wells. Stochastic optimization of a biologically plausible spino-neuromuscular system model: A comparison with human subjects. *Genetic Programming and Evolvable Machines*, In Press, 2007.

[23] S. Gotshall, C. Canine, C. Jennings, and T. Soule. Evolutionary training of a biologically realistic spino-neuromuscular system. In *Proceedings of the International Joint Conference on Neural Networks*, pages 280–285, 2005.

[24] S. Gotshall and T. Soule. Stochastic training of a biologically plausible spino-neuromuscular system model. In *Proceedings of the genetic and evolutionary computation conference*, 2007.

[25] E. Henneman and L. Mendell. *Handbook of Physiology - Functional Organization of the Motoneuron Pool and its Inputs.*, volume 1. American Physiological Society, 1981.

[26] E. Henneman, G. Somjen, and D. Carpenter. Excitability and inhibitibility of motoneurons. *Journal of Neurophysiology*, 28:599–620, 1965.

[27] E. Henneman, G. Somjen, and D. Carpenter. Functional significance of cell size in spinal motoneurons. *Journal of Neurophysiology*, 28:560–580, 1965.

[28] A. Hill. The heat of shortening and the dynamic constants of muscle. *Proc. Roy. Soc. London B*, 126(843):136–195, 1938.

[29] A. Hodgkin and A. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *Journal of Physiology*, 117:500–544, 1952.

[30] J. Howell, A. Fuglevand, M. Walsh, and B. Bigland-Ritchie. Motor unit activity during isometric and concentric-eccentric contractions of the human first dorsal interosseus muscle. *Journal of Neurophysiology*, 74:901–904, 1995.

[31] D. Ivashko, B. Prilutsky, J. Chapin, and I. Rybak. Modeling the spinal cord neural circuitry controlling cat hindlimb movement. *Neurocomputing*, 52-54:621–629, 2003.

[32] J. Izawa, T. Kondo, and K. Ito. Biological arm motion through reinforcement learning. *Biological Cybernetics*, 91:10–22, 2004.

[33] J. F. Kalaska, D. A. D. Cohen, M. L. Hyde, and M. Prud'homme. A comparison of movement direction-related versus load direction-related activity in primate motor corted, using a two-dimensional reaching task. *The Journal of Neuroscience*, 9(6):2080–2102, 1989.

[34] E. R. Kandel, J. H. Schwartz, and T. M. Jessell. *Priciples of Neural Science*. McGraw-Hill, New York, 4 edition, 2000.

[35] F. Kaneko, K. Onari, K. Kawaguchi, K. Tsukisaka, and S. Roy. Electromechanical delay after acl reconstruction: An innovative method for investigating central and peripheral contributions. *Journal of Orthopaedic and Sports Physical Therapy*, 32:158–165, 2002.

[36] A. Kasinski and F. Ponulak. *Experimantal Demonstration of Learning Properties of a New Supervised Learning Method for the Spiking Neural Networks*, pages 145–152. Lecture Note in Computer Science. Springer Berlin / Heidelberg, 2005.

[37] J. Kennedy and R. Eberhart. A discrete binary version of the particle swarm algorithm. In *Proceedings of the Conference on Systems, Man, and Cybernetics*, pages 4104–4109, 1997.

[38] J. Kennedy and R. Eberhart. *Swarm Intelligence*. Morgan Kaufmann Publishers Inc., San Fransisco, 2001.

[39] R. Kettner, A. Schwartz, and A. Georgopoulos. Positional gradients and population coding of movement direction from various movement origins. *Journal of Neuroscience*, 8:2938–2947, 1988.

[40] M. J. Kurz and N. Stergiou. An artificial neural network that utilizes hip joint actuations to control bifurcations and chaos in a passive dynamic bipedal walking model. *Biological Cybernetics*, 93:213–221, 2005.

[41] W. Maass and B. Ruf. The computational power of spiking neurons depends on the shape of the postsynaptic potentials. *Electronic Colloquium on Computational Complexity (ECCC)*, 3(25), 1996.

[42] M. Mandischer. Evolving recurrent neural networks with non-binary encoding. In *IEEE International Conference on Evolutionary Computation*, volume 2, pages 584–589, 1995.

[43] T. A. McMachon. *Muscles, Reflexes, and Locomotion*. Princeton University Press, 1984.

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

[45] K. Nakazawa, Y. Kawakami, T. Fukunaga, H. Yano, and M. Miyashita. Differences in activation patterns in elbo flexor muscles during isometric, concentric, and eccentric contractions. *European Journal of Applied Physiology and Occupational Physiology*, 66:214–220, 1993.

[46] A. Nardone, C. Romano, and M. Schieppati. Selective recruitment of high-threshold human motor units during voluntary isotonic lengthening of active muscles. *Journal of Physiology (London)*, 409:451–471, 1989.

[47] A. Nardone and M. Schieppati. Shift of activity from slow to fast muscle during voluntary lengthening contractions of the triceps surae in humans. *Journal of Neurophysiology*, 595:363–381, 1988.

[48] N. Ogihara and N. Yamazaki. Generation of human bipedal locomotion by a bio-mimetic neuro-musculo-skeletal model. *Biological Cybernetics*, 84:1–11, 2001.

[49] C. Paul, M. Bellotti, S. Jezernik, and A. Curt. Development of a human neuro-musculo-skeletal model of investigation of spinal cord injury. *Biological Cybernetics*, 93:153–170, 2005.

[50] N. Pavlidis, O. Tasoulis, V. Plagianakos, G. Nikiforidis, and M. Vrahatis. Spiking neural network training using evolutionary algorithms. In *Proceedings of the 2005 International Joint Conference on Neural Networks*, pages 2190–2194, 2005.

[51] B. Reider, M. Arcand, and L. Diehl. Proprioception of the knee before and after anterior cruciate ligament reconstruction. *Arthroscopy*, 19:2–12, 2003.

[52] B. Ruf and M. Schmitt. Learning temporally encoded patterns in networks of spiking neurons. *Neural Processing Letters*, 5:9–18, 1997.

[53] N. Rulkov, I. Timofeev, and M. Bazhenov. Oscillations in large-scale cortical networks: Map-based model. *Journal of Computational Neuroscience*, 17:2003–223, 2004.

[54] K. Saladin. *Human Anatomy*. The McGraw-Hill Companies, 1st edition, 2005.

[55] M. Schiappati, F. Valenza, and M. Rezzonico. Motor unit recruitment in human biceps and brachioradialis muscles during lengthening contractions. *European Journal of Neuroscience. Suppl.*, 4:303, 1991.

[56] R. A. Schmidt and T. D. Lee. *Motor Control and Learning*. Human Kinetics, 3rd edition, 1999.

[57] A. Schwartz. Direct cortical representation of drawing. *Science*, 265:540–542, 1994.

[58] M. Settles and T. Soule. Breeding swarms: a ga/pso hybrid. In *Proceedings of the 2005 conference on Genetic and evolutionary computation, GECCO 2005*, pages 161–168. ACM Press, 2005.

[59] M. J. Shelley and L. Tao. Efficient and accurate time-stepping schemes for integrate-and-fire neuronal networks. *Journal of Computational Neuroscience*, 11:111–119, 2001.

[60] G. Shepherd. *Neurobiology*. Oxford University Press, New York, 3 edition, 1994.

[61] J. P. Sutton and G. Strangman. *The behaving human neocortex as a dynamic network of networks*. Springer-Verlag, London, UK, 2003.

[62] E. Todorov. Direct cortical control of muscle activation in voluntary arm movements: a model. *Nature Neuroscience*, 3(4):391–398, 2000.

[63] M. Torry, M. Decker, H. Ellis, H. Shelburn, M. Sterett, and J. Steadman. Mechanisms of compensating for anterios cruciate ligament deficiency during gait. *Medicine and Science in Sports and Exercise*, 36:14–03–1412, 2004.

[64] B. Walmsley, J. Hodgson, and R. Burke. Forces produced by medial gestrocnemius and soleus muscles during locomotion in freely moving cats. *Journal of Neurophysiology*, 41:1203–1216, 1978.

[65] R. Wells. Kinetics and muscle modeling of a single degree of freedom joint part i: Mechanics. Technical report, University of Idaho, 2003.

[66] R. Wells. Spinal sensorimotor system part i: Overview. Technical report, University of Idaho, 2003.

[67] R. Wells. Spinal sensorimotor system part ii: Motoneurons and motoneuron pathways. Technical report, University of Idaho, 2003.

[68] T. Wennekers and G. Palm. Controlling the speed of synfire chains. In *International Conference on Artificial Neural Networks (ICANN)*, pages 451–456, Berlin, 1996. Springer.

[69] G. Williams, P. Barrance, L. Snyder-Mackler, and T. Buchanan. Altered quadriceps control in people with anterior cruciate ligament defiency. *Medicine and Science in Sports and Exercise*, 36:1089–1097, 2004.

[70] D. Willoughby and L. Taylor. Effects of concentric and eccentric muscle actions on serum myostatic and follistation-like related gene levels. *Journal of Sports and Medicine*, 3:226–233, 2004.

[71] X. Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, 1999.

[72] A. Yazdanbakhsh, B. Babadi, S. Rouhani, E. Arabzadeh, and A. Abbassian. New attractor states for synchronous activity in synfire chains with excitatory and inhibitory coupling. *Biological Cybernetics*, 86:367–378, 2002.