Evolutionary Training of a Biologically Realistic Spino-neuromuscular System

Stanley Gotshall
Department of Computer Science
University of Idaho
Moscow, ID
gots9018@uidaho.edu

Christopher Canine
Department of Electrical and Computer Engineering
University of Idaho
Moscow, ID
cani6203@uidaho.edu

Benjamin Jennings
Department of Physics
Whitworth College
Spokane, WA
bjennings07@whitworth.edu

Terence Soule
Department of Computer Science
University of Idaho
Moscow, ID
tsole@cs.uidaho.edu

Abstract—This paper presents a biologically realistic model of the spino-neuromuscular system (SNMS). The model uses a pulse-coded recurrent neural network to control a simulated human-like arm. We use a genetic algorithm to train the network based on a target behavior for the arm. Our goal is to create a useful model for studying the function and behavior of neural pathways in the SNMS. The genetic algorithm is able to train the network to actuate the arm to achieve controlled motion. Our experimental results demonstrate that certain types of feedback pathways are important for controlling muscle movements.

I. INTRODUCTION

Despite considerable advances in knowledge about the structure and function of the Spino-neuromuscular system (SNMS), there is still a need to expand the understanding of the role of neural pathway diversity in motor control and in compensating for injury and disease [3], [7], [14], [17]. If a movement is disrupted, it is not known which neural pathways are most significant in correcting the movement [26], [10], [19], [27]. If a limb is injured, it is generally unknown how the selection of neural pathways change to compensate for the injury. Advances in the field of artificial neural networks has made it plausible to simulate portions of the human nervous system at the level of individual neurons. The goal of this paper is to show that this is possible for the SNMS with with automated training and that this can be a useful tool for modelling complex neural pathways.

In this paper, we use a biologically realistic neural network to control a joint with a Hill muscle system [6], [11]. The Hill muscle is used as the bicep in a model of a human-like arm with a single degree of freedom. The control network is modelled after the neural connections in the spinal cord’s ventral horn. The synaptic connections in the network are trained with an evolutionary algorithm, specifically a genetic algorithm (GA), which is easily generalized to any topology. GAs automatically sample a wide range of possible solutions and select, by analogy to biological evolution, the most fit ones [5]. The SNMS model simulates the function of key spinal neurons and links to the Hill muscle simulation. Our experiments investigate the significance of Renshaw cells and their synaptic links in controlling movement [20], [9].

II. BACKGROUND

Numerous researchers have used models of the muscular and skeletal system to resolve redundancy questions by assuming that selected movements are optimal with respect to some unknown criterion [4], [12], [16], [18]. However, this research usually does not include neural circuitry connected to the muscle systems. In contrast, research focused on modelling 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 [4], [1], [2], [8], [13], [22]. Such research is useful in showing that the particular neural pathway(s) encoded in the model can account for any particular behavior. However, showing that a particular neural pathway can generate a biologically realistic movement in a model is not the same as showing that the pathways does generate the movement in an organism. Thus, our goal is to use a GA to find the most natural neural pathways for specific
motions.

Boothe and Cohen modelled a neural circuit after a central pattern generator (CPG) to study inhibitory interactions between agonist and antagonist muscle pairs [2]. Their experiments support the idea that motor control is strongly influenced by a neural CPG. However, their model does not include quantitative simulations of the muscles themselves. Thus, the next step is to include a biologically realistic muscle for the neural model to control.

Ivashko et al. also use a CPG based model and include Hill-type muscles [8]. They successfully use their neural model to control a simulated cat hind limb. However, in their experiments, they configure the synaptic links manually to achieve stable motion. This makes generalizing the neural model to different topologies difficult because one change to the network could require manual re-tuning of the entire network and incorporates user bias. Our goal is to make a generalizable SNMS model to answer questions about the importance of certain spinal neural pathways in motor control. This requires a rapid, bias-free training mechanism. Thus, a primary goal of this research is to show that a GA can train the SNMS network.

III. MODELS

A. Joint Model

The joint model use here (Fig. 1) has a single degree of freedom. The vertical segment of the arm is fixed and tension due to the bicep and force due to gravity moves the forearm. To better simulate a human arm simulated springs are placed at \( \pi \) -2.4 and \( \pi \) -0.4 radians to reflect the arms innate resistance to extreme angles. The spring at \( \pi \) -2.4 radians models the natural resistance while fully contracting a human arm. The spring at \( \pi \) -0.4 is meant to model the behavior of the arm while muscles are relaxed. At rest there is a small amount of bicep tension resulting in a slight inclined angle. The connections and placement of the bicep is shown as I₁ in Fig. 1.

The movable limb is a uniform cylinder with parameters shown in Table I.

<p>| TABLE I |</p>
<table>
<thead>
<tr>
<th>JOINT MODEL PARAMETERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movable Limb Radius</td>
</tr>
<tr>
<td>Movable Limb Mass</td>
</tr>
<tr>
<td>Timestep</td>
</tr>
<tr>
<td>L₁</td>
</tr>
<tr>
<td>L₂</td>
</tr>
<tr>
<td>L₃</td>
</tr>
<tr>
<td>L₄</td>
</tr>
<tr>
<td>Spring Constant</td>
</tr>
</tbody>
</table>

The mass and dimensions of the arm model are somewhat greater than that of an actual human arm because lighter limbs are more sensitive to the simulated muscle contractions. In this model, slightly heavier limbs are easier to actuate.

B. Hill Muscle

The Hill model is commonly used to simulate muscle systems and their nonlinearities. The muscle has three independently triggerable contractile elements connected to a common tendon. Each contractile element receives a binary neural signal from its alpha-motoneuron (α-MN) and contracts according to the Hill equations. A contractile element contracts if its α-MN is active and relaxes otherwise. The tendon component also stretches depending on the activity of each of the three contractile elements. The muscle connects to the upper portion of the arm and to the forearm near the joint as indicated in Fig. 1. The total contractile force of the muscle is calculated with Equation 1

\[
T = \sum_{\gamma} \left( T_{\gamma_{sec}} \right) + T_{pec}
\]

where \( \gamma \) is a given contractile element, \( T_{\gamma_{sec}} \) is the force due to a given contractile element (serial elastic component), and \( T_{pec} \) is the force due to the common tendon (parallel elastic component). To match the joint simulation (Table 1), the Hill muscle timestep is also 0.02 seconds.

C. Integrate-and-Fire Neuron

Each neuron in the network uses the pulse-coded integrate-and-fire (IF) model. This is a logical choice for biologically realistic neural networks because IF neurons integrate potential over time just as biological neurons. It is important to note that all of the IF neurons are either excitatory or inhibitory. That is, a given IF neuron can only have positive outgoing synaptic weights or negative outgoing synaptic weights. Classical artificial neurons do not have time-dependant behavior and do not retain phase information which is present in biological networks. However, IF neurons do have time dependant behavior and retain phase information.

With the exception of the α-MNs and Renshaw cells, each IF neuron outputs a single pulse when firing. The α-MNs and Renshaw cells fire two consecutive pulses during an action potential. This is to better simulate these neuron types. Renshaw cells are known to have burst firing patterns [15] and certain classes of α-MNs are easily excited leading to burst-like behavior [21]. Each neuron is modelled with the equation 2

\[
f(t + \Delta t) = f(t) + \Delta t \left( \frac{1}{\tau} \left( g(t) - s(t) \right) \right)
\]

where \( s(t) \) is the current input sum to the neuron and \( g(t) \) is the current level of stored potential. The constant \( \tau \), 0.6, determines the decay rate of the neuron. This decay is electronically similar to that of a leaky integrator [23]. To match the joint and muscle simulations, \( \Delta t \) is 0.02 seconds. This timestep allows for the α-MN to have a maximum firing frequency of \( \frac{1}{0.02} = 50 \) hertz which is reasonably close to biological firing rates [21].

After a neuron fires it enters a refractory period of a single timestep during which it cannot build up any potential. After
Fig. 2. This figure illustrates the descending neural pathways to the α-MN and Renshaw Cell. The nodes labelled dp represent synfire nodes and the input to the subnet is the node labelled I. The Renshaw cell interconnections are also indicated.

the refractory period, the neuron can build potential and fire again.

D. Neural Network

The neural network is composed of three subnets, one for each of the three contractile elements in the Hill muscle. Each subnet is identical and can receive independent inputs (Fig. 2). The individual subnets contain neural chains which model individual descending pathways to the spinal cord neurons.

Each dp node in Fig. 2 is implemented as a synfire node [25]. Each synfire node is a collection of five neurons which are fully connected to the preceding and following node. It is difficult for a simple chain of IF neurons to propagate a signal, however a chain of synfire nodes models the robustness of neural connections in the SNMS. The use of synfire nodes also allows for complex variations in descending signal patterns.

The node labelled I represents the input to the subnet and is implemented as five synchronous binary inputs. Placement of the α-MN and Renshaw cell are also indicated in the figure. Each Renshaw cell has inhibitory links to all three α-MNs in the network and each α-MN signals to its respective contractile element. Despite the simplicity of this model, training the network requires adjusting 261 real-valued weights. We use the genetic algorithm for training because adjusting this number of weights by hand would be prohibitive.

E. Genetic Algorithm

We used a steady state Genetic Algorithm (GA) to train the synaptic links of the network. The GA maintains a population of individuals (chromosomes) where each chromosome encodes a string of synaptic weight values which corresponds to a given network. In our GA, these real values range from 0 to 30 and are interpreted as negative numbers if they correspond to inhibitory links as defined by the known biology incorporated into the network model. During each iteration, the GA selects two parents and generates two offspring. The two offspring then replace the two least fit individuals in the population.

Our goal is to produce a particular movement measured in response to the input from the simulated descending pathways. Fitness, or error, is calculated by summing the difference between the desired behavior and the actual behavior at each timestep t. The smaller the error associated with a given network, the higher the fitness. Specifically, fitness is calculated with equation 3.

\[ F = -\sum_{t} (\Theta(t) - \text{target}(t))^2 \]  

At the end of execution, the most fit individual is taken as the best solution. Specific parameters for the GA are listed in Table II.

<table>
<thead>
<tr>
<th>GA Parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Trials Per Network</td>
<td>50</td>
</tr>
<tr>
<td>Population Size</td>
<td>500</td>
</tr>
<tr>
<td>Iterations</td>
<td>75000</td>
</tr>
<tr>
<td>Parent Selection</td>
<td>2-way Tournament</td>
</tr>
<tr>
<td>Representation</td>
<td>String of 64-bit IEEE Doubles</td>
</tr>
<tr>
<td>Weight Range</td>
<td>[0,30] ([[-30,0] Inhibitory Links])</td>
</tr>
<tr>
<td>Crossover Type</td>
<td>Arithmetic</td>
</tr>
<tr>
<td>Crossover Probability</td>
<td>1.0</td>
</tr>
<tr>
<td>Mutation Type</td>
<td>( N(0,0.75) )</td>
</tr>
<tr>
<td>Mutation Probability</td>
<td>( \frac{1}{n} ) (n = Number of Weights)</td>
</tr>
</tbody>
</table>

IV. Experiments

In these experiments, our goal is to train a biologically realistic neural network to control a simulated neuromuscular system. The general research methodology is to make changes to the network and observe how these changes affect the network’s ability to control movement.

The training is based on a target behavior for the arm which involves the arm raising itself from a resting position, holding at a fixed angle, and returning to the resting position. The target velocity for the upward and downward motion is -0.5 and 0.5 radians per second, respectively, which is much less than the maximum possible. The target holding period is 4.5 seconds.
The descending neural signals are modelled with the three binary inputs to the neural network. The input signals are single timestep bursts and are evenly distributed over the three subnets over a given number of timesteps. The training input to raise the arm is three spiking inputs rotating over the subnets. For example, the pattern for upward motion is as follows: at timestep 0 subnet 1 receives an input signal, at the next timestep the subnet 1 signal deactivates and the subnet 2 signal activates, and so on for the third subnet. The input signals continue to cycle for the duration of the upward motion. The input pattern to lower the arm are pulses evenly distributed over 70 timesteps (1.4 seconds). Thus, during every 70 timesteps each subnet receives a single input pulse in intervals of about 23 timesteps. In general, rapidly pulsed inputs are signals (1 pulse per 3 timesteps) to raise the arm and slower pulses (1 pulse per 70 timesteps) are signals to lower the arm. Gravity provides most of the force needed to lower the arm, however a low frequency input is required to control the downward motion. The input patterns are timed with when the target angle increases and decreases. The full duration of each training run is 20 seconds (1000 timesteps).

The experiments involve training three network configurations.
A. Renshaw Cells with Neighboring $\alpha$-MN Connections
B. Renshaw Cells with Single $\alpha$-MN Connection
C. No Renshaw Cells

Experiment A involves training the neural model with connections from each Renshaw cell to each $\alpha$-MN. Therefore, it is possible for a single $\alpha$-MN to inhibit the other two indirectly through its own Renshaw cell. In experiment B, the network is trained with no Renshaw cell links between subnets. Thus, the $\alpha$-MNs are only inhibited as a function of their own firing patterns. Experiment C removes all the Renshaw cells so the $\alpha$-MN firing patterns are only a function of their own subnet input patterns and the descending synaptic links.

V. RESULTS

On average, removing Renshaw cell interconnections increases training error by approximately 70% and removing them altogether increases error by a much larger amount (Fig. 8 and 9). This change to the network results in a statistically significant difference in fitness (Student’s t-test, $p < 0.0005$).

A. Renshaw Cells with Neighboring $\alpha$-MN Connections

Fig. 3 shows that the GA can successfully train the network to move the arm upward and downward at a controlled rate. The arm’s behavior closely matches the target behavior during most of the upward and downward motions. Fig. 4 shows the individual firing patterns of the $\alpha$-MNs. Note that each $\alpha$-MN has its own unique firing pattern. Each Renshaw cell periodically inhibits the three $\alpha$-MNs (Fig. 5) which causes gaps in the $\alpha$-MN firing patterns. Note that the frequency of gaps varies among the three $\alpha$-MNs. This variance is possible because the IF neurons in the network retain phase and time-dependant information.

B. Renshaw Cells with Single $\alpha$-MN Connections

Fig. 6 shows the result of removing interconnecting Renshaw cell links. In the sample, upward movement becomes more irregular and it takes longer to reach the uppermost angle. The upward motion has distinct shifts in velocity resulting in large deviations from the target behavior. The holding motion is stable and the downward motion appears unchanged from experiment A.

C. No Renshaw Cells

The trained networks with no Renshaw cells nearly always moves the joint in an s-shaped path upward and rapidly drops
while moving down. It appears that with no Renshaw cells, the GA selects a weight set which makes the descending pathways behave like a central pattern generator (CPG). That is, the α-MNs produce a regular pattern of signals that is constant despite the changes in gravitational torque as the arm moves. This produces the s-shaped path during the upward motion because movement slows when the gravitational torque is greatest. The arm suddenly drops down because without feedback the network can not effectively compensate for the changing velocity.

VI. CONCLUSIONS

These results show that it is possible to use a GA to train a biologically realistic pulse-coded neural network to control a complex muscle system. Of the three tested network variations, the data shows that the GA takes advantage of the more effective and the more biologically natural neural pathways to achieve the target motion. The GA achieves this by giving the interconnecting connections large enough values which takes advantage of the links to neighboring α-MNs. If an interconnecting link has a small weight value, perhaps zero, it is equivalent to not having that connection. Furthermore, The Renshaw cell interconnections to other α-MNs appear to be important in the upward motion when the α-MNs fire at a high frequency. When compared with the networks with interconnections, networks trained without them results in significantly higher deviations from the target behavior. These results support the claim that Renshaw cells and their interconnections to other α-MNs contribute to controlled motion of muscle systems.

VII. FUTURE WORK

We have shown that our SNMS model is trainable with an evolutionary algorithm, thus future work would include expanding on this network model. In biological muscle systems afferent feedback mechanisms in the muscle fibers stimulate afferent neurons. These neurons feed information back to the α-MN and other spinal neurons. Future work includes training a single network with afferent neurons and testing that network with forearms of various weights. Ideally, the network would compensate for the change in weight. Our training method and SNMS model generalizes to accommodate this addition.
Fig. 8. This figure shows the average best fit individual of 50 trials over 75,000 GA iterations. In training the network with interconnecting Renshaw cells, the GA continues to find better individuals until approximately 60,000 iterations reaching an average fitness of -10.2 (± 0.9). The GA also continues to find better individuals in training the network single Renshaw connections, though on average it only reaches fitness values of -17 (± 5.5). The average best fit of networks with no Renshaw cells only reaches -114 (± 8.9) (not shown).

Fig. 9. This figure shows the average mean fit individual of 50 trials over 75,000 GA iterations. The GA progressively improves its average individual in all cases. However, the fitness of the trained networks with no Renshaw cells are significantly lower than the other two. The average mean fitness at the end of the GA training for the first, second, and third experiments are -11.6 (± 1.4), -19.4 (± 5.5), and -115 (± 9.3), respectively.

ACKNOWLEDGMENT

This work is supported by NSF EPSCoR EPS-0132626. These experiments were performed on a Beowulf cluster built with funds from NSF grant EPS-80935 and a generous hardware donation from Micron Technologies.

REFERENCES


