A Simple Dynamic Controller for Emulating Human Balance Control

J. Stu McNeal(✉) and Alexander Hunt

Department of Mechanical and Materials Engineering, Maseeh College of Engineering and Computer Science, Portland State University, Portland, OR 97201, USA
mcneal@pdx.edu

Abstract. This paper presents a biologically inspired control system developed for maintaining balance in a simulated human atop an oscillating platform. This work advances our previous research by adapting a human balance controller to an inverted pendulum and controlled by linear-Hill muscle models. To expedite neuron/synapse parameter value selection, we employ a novel two-stage process that pairs a previously developed analytic method with particle swarm optimization. Using the parameter values found analytically as inputs for particle swarm optimization (PSO), we take advantage of the benefits of each method while avoiding their pitfalls. Our results show that PSO optimization allowed improved balance control from modest (<10%) changes to the synaptic parameters. The improved performance was accompanied by muscle coactivations, however, and further refinement is needed to better align overall behavior of the neural controller with biological systems.

Keywords: Neural Controller · Balance Control · Functional Subnetwork Approach · Particle Swarm Optimization

1 Introduction

Robots and neural networks are gaining in popularity, but studies that make use of synthetic nervous systems (SNSs) as robotic controllers, especially in motor control applications, have thus far not found much traction in the research community. Most studies are preoccupied with image processing, pattern recognition, or decision-making (e.g. [1–3]). Here we pause to draw a distinction between SNS models such as the one used in this study, and artificial- or recurrent neural networks (ANNs or RNNs), namely that the former features neurons and synapses governed by differential equations (see §2.1 below) that make them behave similarly to the structures they mimic. Of the works interested in the motor control of SNS-driven robots, most explore the utility of CPGs (e.g. [4–6]), with few concerned with balance control. This work presents a novel approach in developing a muscle-actuated adaptive balance controller driven by a synthetic nervous system.

This work was supported by NSF DBI 2015317 as part of the NSF/CIHR/DFG/FRQ/UKRI-MRC next Generation Networks for Neuroscience Program

F. Meder et al. (Eds.): Living Machines 2023, LNAI 14158, pp. 227–239, 2023.
https://doi.org/10.1007/978-3-031-39504-8_16

It is our hypothesis that the addition of numerical optimization to our previously developed analytic methods [7], will enable a synthetic nervous system (SNS) to dynamically balance a muscle-actuated inverted pendulum. We test our hypothesis by adapting our previously developed SNS-driven balance controller [8] to be more biorealistic by replacing the torque-controlled motor with simulated muscles and refining the values of the parameters with particle swarm optimization. Our results show the viability of the method and allow us to more quickly and accurately find parameter values that produce desirable model behaviors.

2 Background

One of the primary inspirations for this work is the balance studies of Peterka (2002, 2003) [23]. During the experiments, the balance of patients with severe vestibular conditions was measured by situating them atop an oscillating platform and securing their trunk and legs to a vertical plank that restricted bulk motion to bending at the ankle (Fig. 1, left). The patients were subjected to a pseudorandom stimulus that tilted their feet back and forth along the sagittal plane. Angle and velocity data were collected from sensors situated at the ankle. Their results introduced the common "Independent Channel" bipedal balance control model that connects weighted sensory inputs to muscle activations, and was later validated by [24] using PostuRob II. Peterka's subsequent 2003 study [25] repeated the analysis on a subset of the original study focusing on patients whose inputs were limited to proprioception and found that the best control equation that described the human balance data was a PD controller with force feedback and passive muscle dynamics (see Fig. 2,[25]). This control equation was used as the basis for the Hilts et al., (2018) SNS model [8], which was adapted for use here. The Hilts SNS incorporated force feedback but not passive muscle dynamics since it drove a torque motor rather than muscles.

The present study extends our previous work, in which an SNS emulated proprioception-based human balance control on an inverted pendulum controlled by a torque motor [8, 9]. Here, we employ a novel two-stage approach to parameter value selection. To understand the theoretical framework used in this study, an understanding of two subjects is required: the functional subnetwork approach (FSA) constraint equations that begin in §2.1 with a description of the neurons and synapses used, and particle swarm optimization (PSO).

2.1 Neurons and Synapses

The neuron model used in this study is a conductance-based non-spiking variant [10, 11], chosen because it captures neuronal subthreshold behavior while keeping computational costs low. Another benefit of this model is that it allows a single neuron to represent the aggregate behavior of a population of neurons [12]. The mathematical framework starts with calculating the membrane voltage V as the sum of various currents I:

$$C_m \frac{dV}{dt} = I_{leak} + I_{syn} + I_{app}, \tag{1}$$

where the leak and synaptic currents can be modeled as:

$$I_{leak} = G_m \cdot (E_r - V) \tag{2}$$

$$I_{syn} = \sum_{i=1}^{n} G_{s,i} \cdot \left(E_{s,i} - V\right) \tag{3}$$

and I_{app} is an optional applied current. Here C_m is the membrane capacitance, G_m and G_s are the neuronal and synaptic conductance, respectively. The reversal potentials, E, are in units of mV and the subscripts indicate whether they apply to the membrane (i.e. E_r) or the synapse (E_s). The synaptic conductance, $G_{s,i}$, is envisioned as a piecewise-linear function for simplicity:

$$G_{s,i} = \begin{cases} 0, & \text{if } V_{pre} \leq E_{lo}, \\ g_{s,i} \cdot \frac{V_{pre} - E_{lo}}{E_{hi} - E_{lo}}, & \text{if } E_{lo} \leq V_{pre} \leq E_{hi}, \\ g_{s,i}, & \text{if } V_{pre} \geq E_{hi}. \end{cases} \tag{4}$$

where $g_{s,i}$ represents the synapse's maximum conductance, E_{lo} and E_{hi} represent the lower threshold and saturation, respectively. Here we introduce the variable $R = E_{hi} - E_{lo}$ that describes the operating range of the network, in mV. For this study, we assign a value of $R = 20$ mV. Also, we introduce $\Delta E_s = E_{s,i} - E_{r,post}$, where $E_{r,post}$ is the resting potential of the post-synaptic neuron. Based on the design rules set forth in [7], parameter value selection is primarily a matter of estimating the values of $g_{s,i}$ and ΔE_s.

2.2 Functional Subnetwork Approach

The functional subnetwork approach (FSA) refers to a set of constraint equations analytically derived by [7], that provide design rules that can be used to build subnetworks for specific mathematical functions (i.e. addition, subtraction, etc.). The constraints are based on the observation that small clusters of neurons can be envisioned as mathematical operators, leading to their classification of synapses as being either for signal transmission or signal modulation. FSA allows designers to analytically develop networks that perform mathematical operations and to quickly find parameter values for the model. We refer the reader to [7] for the full derivation.

While the FSA provides a vehicle with which to find parameter values, the output constraint equations rely on simplifications that can introduce error [7]. As a result, some additional tuning of parameters is often required for optimal model behavior.

2.3 Particle Swarm Optimization

To further tune the SNS, we implement particle swarm optimization (PSO), a metaheuristic genetic algorithm created in 1995 by Kennedy and Eberhart [13]. It was originally envisioned for deployment in describing natural systems [14], as a simplified social milieu of collision-proof birds and fish [13]. PSO was quickly applied to optimization problems with success and has since been the subject of much interest and study. PSO

makes use of parameter sets (swarms) called particles (X_t) that move through the solution space at some velocity (V) determined by some error measurement (X_i). This study makes use of the PSO framework where

$$V_{t+1} = w * V_t + c_1 r_1 \left(P_{best,i} - X_i\right) + c_2 r_2 \left(g_{best,i} - X_i\right) \tag{5}$$

$$X_{t+1} = X_t + V_{t+1} \tag{6}$$

with Eq. (4) having three terms that we call the inertia term, personal term, and group term, from left to right, respectively. The inertia term, which describes the particle's previous velocity, is weighted by w, and the personal and group terms contain a random number $0 \leq r_{1,2} \leq 1$ and a weighting variable $c_{1,2}$ that scale the calculated error in the solution space. The new velocity V_{t+1} is the sum of the weighted inertial term plus weighted inputs based on comparisons of the current error X_i with the particle's *personal* best configuration, $P_{best,i}$, and the best *group* configuration, $g_{best,i}$.

Rapid convergence is a hallmark characteristic of PSO but can lead to swarm stagnation if the early solution is sub-optimal [15]. Some modified methods have attempted to address this, with varying degrees of success. Some efforts, for example, have found means of guaranteeing convergence (GCPSO) (see [16]), but these algorithms require a priori solution knowledge and have shown to perform similarly to PSO in multimodal problems [15]. Riget and Vesterstrom (2002) implemented a scheme that alternated attraction and repulsion phases, resulting in better algorithm performance that scaled with model dimensionality [17]. Silva et al., 2002 utilized a predator particle that forces other particles to disperse [18]. While these latter strategies proved effective, they introduce further complexity and computational cost to the effort. Veeramachaneni et al. (2003) sought to avoid stagnation by slowing convergence using at fitness-distance-ratio-PSO [19]. This study addresses stagnation by setting the weights such that increment is slowed by small parameter values. Here w was set to 0.8 and both c values in Eq. (5) were set to 0.1 so that the overall influence of the personal and group terms was 10% per epoch.

Although PSO converges quickly, the algorithm's continuous reliance on g_{best} means that particles naturally move toward convergence. This behavior makes PSO ill-suited to deployment in vast solution spaces, which is a significant barrier for SNSs and ANNs alike. To combat this and access PSO's rapid convergence, several studies have shown the utility of pre-seeding the particles so that they are proximal to solution troughs. Parsopolous and Vrhatis (2002) [20] initialized particles using Nelder-Mead (1965)'s Nonlinear Simplex Method NSM [21]. While the NSM overhead added to the overall cost, the benefits could be significant. Trelea (2003) added to this effort by adding evidence-based guidance on this while investigating the exploration-exploitation tradeoff inherent to optimization algorithms [22]. This study bypasses this limitation by using FSA output parameter values as a seed for PSO.

3 Experimental Methods

Physical Model. A physical model was developed that approximates the humans tested during the balance trials (Fig. 1, right) [23]. Wherever possible, physical model parameters represent the average physical characteristics of the patients who participated in the

original study. Average height (165 cm), mass (60 kg), and center of mass (80 cm) were used. The model, henceforth called the block person, consists of a single body and a foot situated atop an oscillating platform. A simple hinge joint between the body and the foot represented the ankle joint and was controlled by two muscles, situated contralaterally on the posterior and anterior sides of the lower portion of the body. Thus, balance is maintained by actuating either the anterior or posterior muscle, controlled by the SNS. Note that no effort was made to consider human physiology when creating or placing the linear-Hill muscle models (see [26]) between the lower section of the body and the feet. However, the model does capture the basic mechanics of actuation that connect an input signal to muscle activation. Angle data from the ankle joint is transformed to an applied current and sent to the SNS and used as an input for balance control.

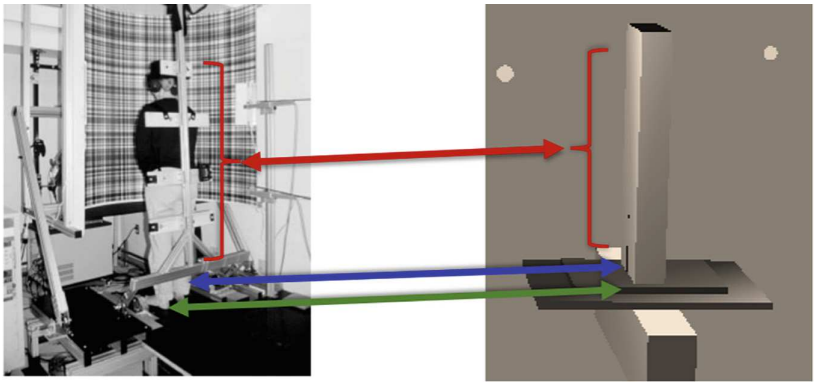

Fig. 1. (Left) Human on platform during balance experiments. (Right) Human balance control model generated in AnimatLab [27]. Lines indicate analogous elements. Red: body, blue: ankle joint, green: foot. Image on left courtesy of [23]

Hill Muscles. AnimatLab makes use of a linear-Hill muscle model. While the modeled muscles do not precisely mimic the detailed properties of the muscles they emulate, they readily capture the essential relationships between force, length, and velocity that underlie their biological counterparts. The muscles consist of a serial spring with coefficient k_{se} with a parallel combination of a force actuator A, dashpot with damping coefficient b, and a parallel spring with coefficient k_{pe}. Contractile tension T is generated by the model when one or more motor neurons depolarize the muscle membrane, according to

$$\frac{dT}{dt} = \frac{k_{se}}{b}\left(k_{pe}x + b\dot{x} - \left(1 + \frac{k_{pe}}{k_{se}}\right) \cdot T + A\right) \tag{7}$$

where x is the displacement of the resting muscle length, $x = l - l_{rest}$. Actuation A is the product of a sigmoid adapter, A_m, and a length-tension component, A_l, such that $A = A_m * A_l$. The sigmoid adapter equation,

$$A_m = \frac{F_{max}}{1 + e^{C(V_o - V)} + B}, \tag{8}$$

relates the maximum muscle force F_{max} to the motor neuron membrane voltage V by a slope coefficient C. V_o and B control the V and F offsets, respectively. The length-tension relation scales A according to

$$A_l = 1 - \frac{x^2}{l_{width}^2} = 1 - \frac{(l - l_{rest})^2}{l_{width}^2} \tag{9}$$

where $l_{width}$is the length at which the muscle can no longer produce force.

Implementation of the linear-Hill muscle model in AnimatLab was accomplished by setting the necessary coefficient values. Wherever possible, reference values were chosen as the average from values representing people in the 45–69-year range, the demographic containing the patients in the balance experiments. For this study, F_{max} (2300 N) was estimated based on data presented in Thelen (2003) [28]. Since the SNS model operates between −40 mV and −60 mV, the upper and lower limits, respectively, V_o was set to −50 mV. Steepness (530 N/mV) is simply the slope of the sigmoid at V_o. l_{rest} (40 cm) is estimated as the length of the lower leg, deduced by subtracting the average length of the upper leg (~41.4 cm)[1] from half of the overall body height (82.5 cm). l_{width} (2.6 cm) was calculated by manipulating Eq. (9) and represents the length at which <1% tension can be attained. Finally, k_{se} (575 KN/m) was set according to data from [29], and k_{pe} (9.75 KN/m) was set based such that the muscle deflects 4% of its upper limit under maximum loading conditions [30].

SNS. The Hilts SNS model is a faithful representation of the transfer function from Peterka (2003) describing a PD controller + force feedback [25]. A passive time delay was incorporated into the model in the form of lengthened time constants in the feedback gain loop. The model was adapted for use in this study and is shown in Fig. 2. The ankle joint angle θ serves as the input and is mapped such that $-R/2 \leq \theta \leq R/2$, and is transformed to an applied current that bounds the receiving neuron, θ_{now}, between −40 mV and −60 mV. Since the calculations made by the SNS model must be "positive" and align with the relative value of the input and output neurons, error calculations are made in parallel such that the reference voltage (−50 mV) is subtracted from the incoming θ voltage, and vice versa. The difference is conveyed along excitatory synapses, and out-of-bounds (i.e. inhibitory) values are ignored. To account for this limitation, the model is symmetrical about the horizontal midline (Fig. 2) with most structures featured twice. The SNS model is configured such that the neurons in the top half are active when the body is rotated counterclockwise relative to the foot and the neurons in the bottom half are active when the body is rotated clockwise relative to the foot. Structures appearing equivalently in the CW and CCW halves were grouped during the optimizations. The network outputs muscle activation to the CW and CCW muscles from the Sum neurons that represent the sum of the *Kp* and *Kd* pathways. The *Kt* force feedback circuit functions to minimize steady state errors (see Eq. 6, [31]). Parameter values were initially set to those reported in [8] and according to the FSA constraints.

Optimization. Gain in the Hilts SNS is the product of a tonic input current and the parameter it attenuates, manifested as multiplication subnetworks as described by [7].

[1] See Table 29, https://www.cdc.gov/nchs/data/series/sr_03/sr03-046-508.pdf.

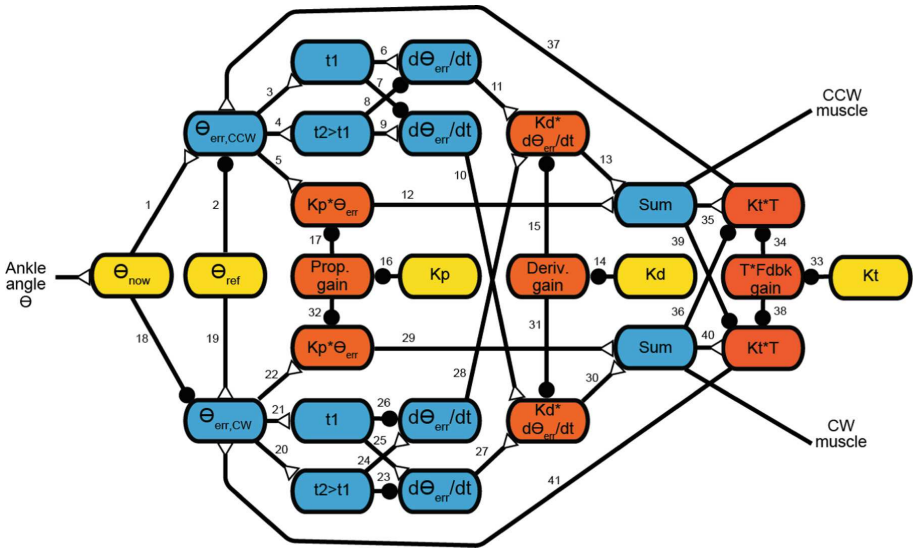

Fig. 2. The synthetic nervous system (SNS) developed by Hilts (2018) and modified for use in this study. For this study, a new system identification was performed using PSO.

For example, the *Kp* neuron (Fig. 2) receives a tonic stimulus and is multiplied times θ_{err} at$Kp \cdot \theta_{err}$. The multiplications subnetworks are comprised of a single excitatory neuron and two inhibitory neurons with synapse E_s-g_s values set identically (e.g., synapses $16 = 17 = 32$). Consequently, each SNS gain circuit is comprised of two sets of E_s-g_s values and a tonic stimulus. To avoid mathematically ambiguous solutions, optimization of the gain circuits was done in two steps, beginning with finding optimized tonic stimulus values. In total, two optimizations were performed. The parameters optimized during the experiment are shown with their start/end values in §4 in Tables 1 and 2.

To perform the PSO, python code was written that implements PSO and iterates AnimatLab simulation files in a standalone capacity, saving time and computational cost. Optimizations are divided into epochs that begin by assigning new parameter values to each particle, using Eqs. (5) and (6). For each particle during an epoch, the code.

- Writes a standalone AnimatLab file using the newly calculated parameter values.
- Runs an AnimatLab iteration using the file generated in the previous step.
- Reads from the AnimatLab text file and fetches the output orientation data for the platform and the block person.
- Calculates the particle error. Error estimations are comprised of three calculations that measure $\theta_{platform}$- θ_{person} variance, muscle tensions $T_{\{CW,CCW\}}$ (a proxy for preventing muscle coactivation), and a muscle activation roughness metric that addresses SNS instabilities. Each error element is scaled such that muscle activation is the most heavily penalized component, and curve roughness is the least. The error for each particle is calculated after each iteration according to:

$$Error = e_{angles} + e_{tensions} + e_{rough} \tag{10}$$

where

$$e_{angles} = \frac{1}{\theta_{max}} \cdot sqrt\Big(mean\Big(\big(\theta_{platform} - \theta_{person}\big)^2\Big)\Big) \tag{11}$$

$$e_{tensions} = \frac{1}{T_{max}} \cdot sqrt\Big(mean\Big(\big(T - (T_{CW} + T_{CCW})\big)^2\Big)\Big) \tag{12}$$

$$e_{rough} = (mean(abs(diff\,(T_{CW}))) + mean(abs(diff\,(T_{CW})))) \cdot \frac{1}{2} - \phi \tag{13}$$

Here θ_{max} represents the largest difference in angle allowed by the physical model (8 °), T_{max} is the maximum value of T, calculated based on the static torque needed at $\theta = \theta_{max}$. For e_{rough}, an offset ϕ is imposed to scale output values (i.e. $e_{rough} \rightarrow 0$) for target performance.

- Returns the error data matrix to the PSO manager.

The PSO manager then calculates new parameter values (velocities) according to Eqs. (5) and (6) and a new epoch begins. This code is on the AARL GitHub repository and available to the public.

Test Conditions. During the experiments, the oscillating platform, which mapped to oscillations described by a cosine wave $\theta = \sin(t)$, where θ is in degrees and t is in seconds, from $\pi \leq t < 60$. This configuration differs from the original experiments in that it only captures a single frequency, although it represents the maximum point-to-point variance of 4 degrees used for the patients with vestibular issues. Error was calculated on values during the interval from $\pi \leq t < 60$ seconds according to Eq. (10). Since PSO convergence conditions are difficult to establish analytically [32], tests were performed to establish convergence based on parameter variance across the particles optimized. Bench tests showed the lowest variance occurred at about 60 epochs and 20 particles. Trials performed during the experiments incorporated 20 particles and 200 epochs, and the output solution was the particle configuration with the lowest overall error, g_{best} from Eq. (5).

4 Results

Figure 3 shows a comparison the block body orientation in various solution stages plotted with the platform orientation. During the simulations, the overall goal of the model was to actuate the muscles such that the body and platform maintained alignment. For reference, line (2) shows the output orientation of the block body when the muscles failed to actuate: the body slumped to one side and remained stationary for the duration of the run. Line (3) shows orientation after FSA-guided parameter value selection. The muscles actuated, but the behavior is erratic and the muscles are overcompensating. Line (4) shows orientation after FSA and PSO optimization. The body tracks well with the platform as it oscillates and the phase lag appears to show a delay similar to that seen in human balance control.

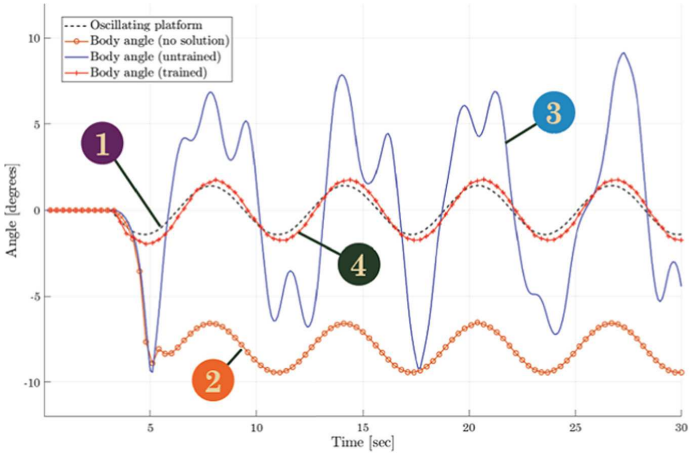

Fig. 3. Orientation of the oscillating platform (1) plotted with block person orientation for various solutions. (2) shows the results where the muscles failed to support the body and slumped to one side, (3) shows orientation after setting synapse parameters according to FSA constraints, indicating a partial solution, and (4) shows body orientation after FSA and PSO solution.

Tables 1 and 2 show the optimized parameter values before and after PSO. Since the mass and overall dimensions of the block body are much larger than that of Hilts' inverted pendulum, gain values are expected to be lower in this study. The tonic stimulus' shown in Table 1 decreased overall, with a 2/3 reduction to *Kd* and about an order of magnitude for the *Kd* and *Kt* inputs. Parameter value change during PSO was modest overall ($<10\%$), with the largest shift to g_s in the inhibitive side of the *Kt* circuit.The modesty of the changes by PSO lend further validate the analytic methods performed ahead of this work.

Table 1. Neuronal parameters from Hilts SNS optimized by during the first PSO of this study.

Neuron	Parameter	Hilts	PSO
Kd	Stim Current, I_{App} (nA)	12.5	4.2
Kp	Stim Current, I_{App} (nA)	13.5	1.42
Kt	Stim Current, I_{App} (nA)	0.8	0.08

Table 2. Synapse parameters before and after PSO optimization. Type refers to whether the synapse is excitatory (+) or inhibitory (-). * hand-tuned prior to PSO to stabilize SNS behavior.

Circuit	Synapses	Type	E_s(mV)			$g_s(\mu$ S)		
			Hilts	PSO	diff (%)	Hilts	PSO	diff (%)
Kp	5, 22	+	134	138.92	3.7	2.2	2.29	4.3
	16–17, 32	−	−60	−59.29	1.2	20	19.64	1.8
Kd	10–11, 27–28	+	134	138.24	3.2	40*	43.48	4.3
	14–15, 31	−	−60	−61.37	3.2	20	20.19	0.9
Kt	35, 40	+	134	132.49	1.1	0.558	0.585	4.8
	33–34, 38	−	−61	−61.37	1.1	20	21.68	8.4
	36, 39	−	−100	−96.17	3.8	0.558	0.560	0.3

5 Discussion

We hypothesized that the methods presented here would produce dynamic balance control using muscle activations. We demonstrated that by adopting the FSA-PSO method to a modified SNS, the model successfully found a solution that resulted in good body-platform alignment. This proof-of-concept exercise shows that balance can be maintained by actuating muscles in a fashion that approximates the basic mechanics of proprioception-based human balance control.

Although the method found a viable solution that produces the target behavior, the solution it found relies on high muscle coactivation (Fig. 4), which is not very efficient. The expectation is that only one muscle should be activated during peak output, and during the simulation minimum muscle activation only fell to about 20% of the peak rather than to zero. This is the result of an underconstrained cost function that permitted unrealistic neural activity that caused the muscles to remain activated during most of the simulation. To better approximate the energy minimizing nature of the biology, our next steps will refine the PSO cost calculations to further penalize muscle activation and produce behavior that better mimics that found in humans.

The methods used in this study can be generalized to encompass a broad range of balance and control applications where muscle actuation is required. Increasingly realistic SNSs that incorporate features like type Ib feedback can now be tuned with minimal increases to optimization time and computational cost. Additionally, increases in tuning efficiency allow modelers to devote more time to other aspects of biomimetic modeling, like hardware implementation.

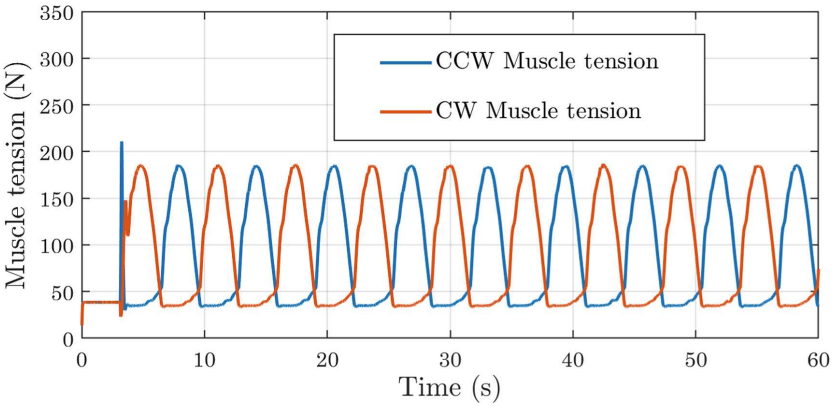

Fig. 4. Muscle activation data output from AnimatLab simulation of the FSA-PSO solution. While the controller maintained good alignment with the platform, it did so by keeping both muscles actuated throughout the simulation.

References

1. Franco, J.A.G., Padilla, J.L. del V., Cisneros, S.O.: Event-based image processing using a neuromorphic vision sensor. In: 2013 IEEE International Autumn Meeting on Power Electronics and Computing (ROPEC), pp. 1–6 (2013). https://doi.org/10.1109/ROPEC.2013.6702715
2. Chu, M., et al.: Neuromorphic hardware system for visual pattern recognition with memristor array and CMOS NEURON. IEEE Trans. Ind. Electron. **62**, 2410–2419 (2015). https://doi.org/10.1109/TIE.2014.2356439
3. Corradi, F., You, H., Giulioni, M., Indiveri, G.: Decision making and perceptual bistability in spike-based neuromorphic VLSI systems. In: 2015 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 2708–2711 (2015). https://doi.org/10.1109/ISCAS.2015.7169245
4. Chen, W., Ren, G., Wang, J., Liu, D.: An adaptive locomotion controller for a hexapod robot: CPG, kinematics and force feedback. Sci. Chin. Inf. Sci. **57**(11), 1–18 (2014). https://doi.org/10.1007/s11432-014-5148-y
5. Deng, K., et al.: Neuromechanical model of rat hind limb walking with two layer CPGs and muscle synergies. In: Vouloutsi, V., et al. (eds.) Living Machines 2018. LNCS (LNAI), vol. 10928, pp. 134–144. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-95972-6_15
6. Endo, G., Morimoto, J., Matsubara, T., Nakanishi, J., Cheng, G.: Learning CPG-based biped locomotion with a policy gradient method: application to a humanoid robot. Int. J. Robot. Res. **27**, 213–228 (2008). https://doi.org/10.1177/0278364907084980
7. Szczecinski, N.S., Hunt, A.J., Quinn, R.D.: A Functional subnetwork approach to designing synthetic nervous systems that control legged robot locomotion. Front. Neurorobotics. **11**, (2017). https://doi.org/10.3389/fnbot.2017.00037
8. Hilts, W.W., Szczecinski, N.S., Quinn, R.D., Hunt, A.J.: Emulating balance control observed in human test subjects with a neural network. In: Vouloutsi, V., et al. (eds.) Living Machines 2018. LNCS (LNAI), vol. 10928, pp. 200–212. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-95972-6_21
9. Hilts, W.W., Szczecinski, N.S., Quinn, R.D., Hunt, A.J.: Simulation of human balance control using an inverted pendulum model. In: Mangan, M., Cutkosky, M., Mura, A., Verschure,

P.F.M.J., Prescott, T., Lepora, N. (eds.) Living Machines 2017. LNCS (LNAI), vol. 10384, pp. 170–180. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63537-8_15
10. Hopfield, J.J.: Neurons with graded response have collective computational properties like those of two-state neurons. Proc. Natl. Acad. Sci. **81**, 3088–3092 (1984). https://doi.org/10.1073/pnas.81.10.3088
11. Beer, R.D., Gallagher, J.C.: Evolving dynamical neural networks for adaptive behavior. Adapt. Behav. **1**, 91–122 (1992). https://doi.org/10.1177/105971239200100105
12. Wilson, H.R., Cowan, J.D.: Excitatory and inhibitory interactions in localized populations of model neurons. Biophys. J. **12**, 1–24 (1972). https://doi.org/10.1016/S0006-3495(72)86068-5
13. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the ICNN95 - International Conference Neural Network (1995)
14. Kennedy, J.: The particle swarm: social adaptation of knowledge. In: Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC 1997), pp. 303–308 (1997). https://doi.org/10.1109/ICEC.1997.592326
15. Banks, A., Vincent, J., Anyakoha, C.: A review of particle swarm optimization. Part I: background and development. Nat. Comput. **6**, 467–484 (2007). https://doi.org/10.1007/s11047-007-9049-5
16. van den Bergh, F., Engelbrecht, A.P.: A new locally convergent particle swarm optimiser. In: IEEE International Conference on Systems, Man and Cybernetics, vol. 3, p. 6 (2002). https://doi.org/10.1109/ICSMC.2002.1176018
17. Riget, J., Vesterstrøm, J.S.: A diversity-guided particle swarm optimizer-the ARPSO. Dept. Comput. Sci Univ Aarhus Aarhus Den. Technical report. 2, 2002 (2002)
18. Silva, A., Neves, A., Costa, E.: An empirical comparison of particle swarm and predator prey optimisation. In: O'Neill, M., Sutcliffe, R.F.E., Ryan, C., Eaton, M., Griffith, N.J.L. (eds.) Artificial Intelligence and Cognitive Science. AICS 2002. Lecture Notes in Computer Science, vol. 2464, pp. 103–110. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45750-X_13
19. Veeramachaneni, K., Peram, T., Mohan, C., Osadciw, L.A.: Optimization using particle swarms with near neighbor interactions. In: Cantú-Paz, E., et al. (eds.) GECCO 2003. LNCS, vol. 2723, pp. 110–121. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-45105-6_10
20. Parsopoulos, K.E., Vrahatis, M.N.: Initializing the particle swarm optimizer using the nonlinear simplex method. Adv. Intell. Syst. Fuzzy Syst. Evol. Comput. **216**, 1–6 (2002)
21. Nelder, J.A., Mead, R.: A simplex method for function minimization. Comput. J. **7**, 308–313 (1965). https://doi.org/10.1093/comjnl/7.4.308
22. Trelea, I.C.: The particle swarm optimization algorithm: convergence analysis and parameter selection. Inf. Process. Lett. **85**, 317–325 (2003). https://doi.org/10.1016/S0020-0190(02)00447-7
23. Peterka, R.J.: Sensorimotor integration in human postural control. J. Neurophysiol. **88**, 1097–1118 (2002)
24. Pasma, J.H., Assländer, L., van Kordelaar, J., de Kam, D., Mergner, T., Schouten, A.C.: Evidence in support of the independent channel model describing the sensorimotor control of human stance using a humanoid robot. Front. Comput. Neurosci. **12**, 13 (2018)
25. Peterka, R.J.: Simplifying the complexities of maintaining balance. IEEE Eng. Med. Biol. Mag. **22**, 63–68 (2003). https://doi.org/10.1109/MEMB.2003.1195698
26. Hill, A.V.: The heat of shortening and the dynamic constants of muscle. Proc. R. Soc. Lond. B Biol. Sci. **126**, 136–195 (1938). https://doi.org/10.1098/rspb.1938.0050
27. Cofer, D., Cymbalyuk, G., Reid, J., Zhu, Y., Heitler, W., Edwards, D.: AnimatLab: a 3D graphics environment for neuromechanical simulations. J. Neurosci. Meth. **187**, 280–288 (2010). https://doi.org/10.1016/j.jneumeth.2010.01.005

28. Thelen, D.G.: Adjustment of muscle mechanics model parameters to simulate dynamic contractions in older adults. J. Biomech. Eng. **125**, 70–77 (2003). https://doi.org/10.1115/1.1531112
29. Pearson, K.G., Ekeberg, Ö., Büschges, A.: Assessing sensory function in locomotor systems using neuro-mechanical simulations. Trends Neurosci. **29**, 625–631 (2006). https://doi.org/10.1016/j.tins.2006.08.007
30. Meijer, K., Grootenboer, H.J., Koopman, H.F.J.M., van der Linden, B.J.J.J., Huijing, P.A.: A Hill type model of rat medial gastrocnemius muscle that accounts for shortening history effects. J. Biomech. **31**, 555–563 (1998). https://doi.org/10.1016/S0021-9290(98)00048-7
31. Hilts, W.W., Szczecinski, N.S., Quinn, R.D., Hunt, A.J.: A Dynamic neural network designed using analytical methods produces dynamic control properties similar to an analogous classical controller. IEEE Control Syst. Lett. **3**, 320–325 (2019). https://doi.org/10.1109/LCSYS.2018.2871126
32. Clerc, M., Kennedy, J.: The particle swarm - explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. Evol. Comput. **6**, 58–73 (2002). https://doi.org/10.1109/4235.985692