applied sciences

MDPI

Article

Supplementary Materials

1. Video and File Links

Video S1: A video of SimRoach2 walking forward in a simulated environment can be found at the link below.

https://vimeo.com/214706759

File S2: A link to download the AnimatLab 2 files for SimRoach2 can be found at the link below.

https://drive.google.com/open?id=0Bw02uf2W7DPXSkpDaVA1TDg5T1U

File S3: A link to download the version of the KinematicOrganism Toolbox used for SimRoach2 can be found at the link below.

https://drive.google.com/open?id=1YBkv6uLMSZiGPbLHdn-fPKSvSyLSzMW-

2. Parameter Values

Tables S4-S8: Many parameter values were chosen as part of the design process. In this section, these parameter values will be explicitly given (Tables 1-5).

2.1. Rest Posture - Spring, Flexor, and Extensor Rest Lengths

One of the steps of the design process involved choosing a resting posture for the model. The chosen angles of the rest posture corresponded directly to specific resting lengths of the spring, flexor, and extensor based on the muscle attachment points. Table 1 provides these values. In each joint, the spring used the same attachment points as the flexor, thus the resting length was the same and that value is only provided once.

Table 1. Rest Lengths of Spring, Flexor, and Extensor.

Joint	Rest Length of Spring and Flexor (mm)	Rest Length of Extensor (mm)
LF ThC2	4.217	4.998
LF ThC1	4.331	4.778
LF ThC3	2.872	3.054
LF CTr	3.901	4.005
LF FTi	5.497	5.355
RF ThC2	4.217	4.999
RF ThC1	4.331	4.779
RF ThC3	2.869	3.054
RF CTr	3.901	4.005
RF FTi	5.490	5.359
LM ThC2	4.220	7.527
LM ThC1	3.548	3.841
LM CTr	5.167	5.490
LM TrF	5.464	5.211
LM FTi	8.602	8.109
RM ThC2	4.184	7.274
RM ThC1	3.548	3.84
RM CTr	5.167	5.490
RM TrF	5.464	5.211
RM FTi	8.610	8.102
LH ThC2	5.281	7.336
LH ThC1	5.177	5.774
LH CTr	7.565	8.243
LH TrF	4.595	4.597
LH FTi	8.672	8.692
RH ThC2	5.281	7.336
RH ThC1	5.177	5.774
RH CTr	7.565	8.243
RH TrF	4.596	4.597
RH FTi	8.672	8.692

2.2. Spring Stiffness and Damping Coefficients

Another step was to calculate each springs stiffness and damping coefficients. Table 2 provides the calculated values.

Table 2. Spring Stiffness and Damping Coefficients.

Joint	Stiffness Coefficient (N/m)	Damping Coefficient (mg/s)
LF ThC2	1009.9	2.414
LF ThC1	1009.9	2.414
LF ThC3	228.492	0.351
LF CTr	355.77	2.115
LF FTi	511.805	0.210
RF ThC2	1001.9	2.403
RF ThC1	1001.9	2.403
RF ThC3	231.762	0.353
RF CTr	355.77	2.115
RF FTi	434.372	0.209
LM ThC2	400.823	3.00
LM ThC1	708.231	2.238
LM CTr	549.196	3.640
LM TrF	612.759	3.503
LM FTi	289.236	0.399
RM ThC2	1008.99	3.00
RM ThC1	709.442	2.306
RM CTr	549.013	3.818
RM TrF	611.364	5.00
RM FTi	262.222	0.434
LH ThC2	1000.0	5.00
LH ThC1	2677.0	8.969
LH CTr	665.484	23.78
LH TrF	844.002	2.224
LH FTi	369.532	1.961
RH ThC2	1000.0	5.00
RH ThC1	2678.0	8.936
RH CTr	663.72	23.76
RH TrF	846.636	2.244
RH FTi	369.848	1.962

2.3. Muscle Parameter Values

For each muscle, the values of k_{se}, k_{pe}, and b were chosen to be 45 N/m, 11.24 N/m, and 0.1 Ns/m respectively. The x-offset and steepness of the muscle activation curve were chosen to be -50 mV and 300, respectively. The amplitude of the muscle activation curve was chosen as part of the optimization process. The y-offset of the curve was constrained such that there was zero active force when the motor-neuron was at -60 mV. In Table 3, values of the amplitude and y-offset for each muscle are given.

Table 3. Muscle Amplitude and Y-offset values.

Joint	Flexor Muscle Amplitude T_{ce} (N)	Muscle y-offset (mN)	Extensor Muscle Amplitude T_{ce} (N)	Extensor y-offset (mN)
LF ThC2	1.456	-69.036	3.805	-180.47
LF ThC1	1.489	-70.623	1.038	-49.234
LF ThC3	0.407	-19.308	0.802	-38.016
LF CTr	1.145	-54.325	0.754	-35.768
LF FTi	0.0674	-3.198	2.375	-112.66
RF ThC2	1.177	-55.818	1.899	-90.066
RF ThC1	1.332	-63.174	0.713	-33.792
RF ThC3	0.328	-15.544	0.665	-31.555
RF CTr	0.530	-25.140	0.571	-27.092
RF FTi	1.048	-49.695	2.218	-105.20
LM ThC2	0.201	-9.512	0.506	-23.99
LM ThC1	1.00	-50.00	0.200	-10.00
LM CTr	0.600	-28.45	1.60	-80.00
LM TrF	0.684	-32.45	0.220	-11.0
LM FTi	1.097	-52.02	1.257	-59.634
RM ThC2	0.040	-1.897	0.434	-20.581
RM ThC1	1.569	-74.391	0.364	-17.257
RM CTr	0.546	-25.886	1.594	-75.576
RM TrF	0.669	-31.748	0.279	-13.255
RM FTi	1.146	-54.347	1.761	-83.514
LH ThC2	3.013	-143.0	3.905	-185.2
LH ThC1	2.352	-111.5	0.800	-37.0
LH CTr	1.122	-53.21	1.104	-52.358
LH TrF	1.326	-62.861	0.472	-22.363
LH FTi	0.691	-32.754	0.966	-45.82
RH ThC2	3.097	-146.9	3.898	-184.85
RH ThC1	3.00	-142.0	1.20	-56.0
RH CTr	1.00	-47.426	1.117	-52.981
RH TrF	1.211	-52.166	0.400	-18.97
RH FTi	0.411	-19.471	0.541	-25.678

2.4. Proportional and Integral Synapse Parameter Values

As part of the optimization process, the synaptic strength between the proportional and integral parts of each joint position controller to the motor-neurons were chosen. These parameter values are given in Table 4. The equilibrium potential of all of these synapses is -20 mV.

Table 4. Proportional and Integral Synapse Parameter Values.

Joint	Integral Synaptic Conductance (uS)	Proportional Synaptic Conductance (uS)
LF ThC2	19.292	1.354
LF ThC1	13.544	1.128
LF ThC3	19.142	1.396
LF CTr	8.826	0.876
LF FTi	18.31	1.0
RF ThC2	29.777	1.333
RF ThC1	16.381	0.678
RF ThC3	24.737	1.868
RF CTr	19.917	0.901
RF FTi	11.984	0.796
LM ThC2	10.982	1.321
LM ThC1	12.0	1.00
LM CTr	13.0	1.70
LM TrF	9.5	1.3
LM FTi	8.68	1.267
RM ThC2	11.0	0.60
RM ThC1	9.707	0.692
RM CTr	15.04	1.797
RM TrF	8.603	1.954
RM FTi	7.534	0.835
LH ThC2	10.982	1.522
LH ThC1	14.0	0.982
LH CTr	10.0	4.0
LH TrF	14.02	1.089
LH FTi	11.064	1.059
RH ThC2	10.955	1.499
RH ThC1	9.0	0.796
RH CTr	11.0	4.50
RH TrF	14.912	1.211
RH FTi	19.067	2.415

2.5. Upper Level Synapse Parameter Values

Using our tool KinematicOrganism, the synaptic conductance of the synapses from the upper level to the intermediate level and were calculated. These parameter values are given in Table 5. The equilibrium potential of all of these synapses are 300 mV. The standard case is when PEP Positive Rotation and PEP Positive Translation neurons are connected to the PEP Sum neuron, and the PEP Negative Rotation and PEP Negative Translation neurons are connected to the AEP Sum neuron. In some joints, the rotation and/or translation parts are reversed. Table 5 provides this information.

Table 5. Upper Level Synapse Parameter Values.

Joint	Rotational Synaptic Conductance (nS)	Translational Synaptic Conductance (nS)	Reversed Synapses
LF ThC2	0.028411	51.592	Translation
LF ThC1	0.449	27.098	Neither
LF ThC3	1.356	18.47	Translation
LF CTr	1.275	16.241	Rotation
LF FTi	1.854	20.025	Rotation and Translation
RF ThC2	0.706	51.746	Rotation and Translation
RF ThC1	0.429	27.11	Rotation
RF ThC3	0.974	17.545	Rotation and Translation
RF CTr	1.188	16.322	Neither
RF FTi	2.224	19.697	Translation
LM ThC2	0.775	15.684	Rotation
LM ThC1	1.069	52.419	Rotation
LM CTr	0.578	28.093	Rotation
LM TrF	0.223	25.591	Rotation
LM FTi	0.728	6.392	Rotation
RM ThC2	0.664	16.963	Neither
RM ThC1	1.098	50.862	Neither
RM CTr	0.606	28.122	Neither
RM TrF	0.205	25.174	Neither
RM FTi	0.829	7.934	Neither
LH ThC2	0.2	17.032	Neither
LH ThC1	0.363	28.902	Rotation
LH CTr	0.574	46.953	Rotation
LH TrF	0.029312	1.947	Translation
LH FTi	0.329	25.562	Neither
RH ThC2	0.2	17.031	Rotation
RH ThC1	0.363	28.848	Neither
RH CTr	0.574	46.957	Neither
RH TrF	0.029212	1.934	Rotation and Translation
RH FTi	0.329	25.557	Rotation

3. Methods Explained

For the sake of paper length and readability, many details were left out of the main paper. For the purposes of research integrity, more details on the methods are explained in this section.

3.1. Modeling Overview: Description of Design Tools

The first design tool, FeedbackDesign, can simulate the dynamics of the system in both the time domain and the frequency domain [31]. Time domain simulations are useful because we can learn about the stability and equilibrium of a system. Frequency domain calculations are useful because we can use stability margins to learn about the robustness of our model.

The second design tool, KinematicOrganism, can read an AnimatLab 2 file and construct a model in MATLAB. A full model in MATLAB is very useful; this tool automates a lot of the calculations that were performed in the design process. Among many other things, KinematicOrganism can calculate the mapping between the length of the extensor and joint angle, calculate the AEP and PEP for walking, and find the resting posture.

The last design tool, SimScan, can run batches of AnimatLab 2 simulations at once [31]. SimScan does not display a graphics window and takes advantage of parallel computing capabilities, making it much faster to run simulations. Parameters can be varied between simulations, so SimScan can scan through permutations of parameter sets or parameters can be optimized against an objective function.

3.2. Neuron Dynamics

The primary dynamical variable of a neuron is the voltage across the cell membrane, V, with the dynamics:

$$C_m \dot{V}(t) = I_{leak}(t) + I_{syn}(t) + I_{NaP}(t) + I_{app}(t), \tag{1}$$

where

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

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

and

$$I_{NaP}(t) = G_{Na} \cdot m_\infty\big(V(t)\big) \cdot h \cdot \big(E_{Na} - V(t)\big). \tag{4}$$

and $I_{app}(t)$ is an external stimulus current. E_r is the resting potential of the neuron. C_m and G_m are the capacitance and conductance of the cell membrane, respectively. Instead of inputing the capacitance and conductance parameters values directly into AnimatLab 2, the neurons are defined by the parameters "time constant" and "relative size." The time constant is defined by $\tau_m = C_m / G_m$, and the relative size is equivalent to the conductance G_m.

Neurons communicate with each other via synapses. A neuron can, in general, have n incoming synapses. $G_{s,i}$ and $E_{s,i}$ are the conductance and resting potential of the i^{th} synapse, respectively. The conductance of the synapse is given by

$$G_{s,i}(t) = G_{max,i} \cdot \min\left(\max\left(\frac{V_{pre}(t) - E_{lo,i}}{E_{hi,i} - E_{lo,i}}, 0\right), 1\right), \tag{5}$$

where $G_{max,i}$ is the maximum conductance, $E_{lo,i}$ is the lower threshold, $E_{hi,i}$ is the upper threshold, and V_{pre} is the voltage of the pre-synaptic neuron.

Most neurons in our network have $G_{Na} = 0$, however, neurons that make up the CPGs will have $G_{Na} \neq 0$. For these neurons, h is the second dynamical variable with dynamics

$$\dot{h}(t) = \frac{h_\infty - h(t)}{\tau_h(V)}. \tag{6}$$

m_∞ and h_∞ are functions of the neuron's voltage, and are sigmoids of the form

$$z_\infty(V) = \frac{1}{1 + A_z \cdot \exp(S_z \cdot (V - E_z))}, \tag{7}$$

where z represents either m or h and parameters A, S, and E are constants specific to m or h. The time constant of h in Equation 6 is a function of the neuron's voltage and is given by

$$\tau_h(V) = \tau_{h,max} \cdot h_\infty(V) \cdot \sqrt{A_h \cdot \exp(S_h \cdot (V - E_h))}. \tag{8}$$

3.2.1. Units

Unless otherwise stated, voltages are measured in millivolts (mV), conductances are measured in microsiemens (µS)($1/\text{M}\Omega$), currents are measured in nanoamps (nA), capacitances are measured in nanofarads (nF), and time is measured in milliseconds (ms). The differential equation modeling a neuron with 0 incoming synapses and no persistent sodium channels, from combining Equations 1 and 2, is

$$C_m \dot{V}(t) = I_{leak} = G_m \cdot \big(E_r - V(t)\big). \tag{9}$$

Plugging in the units for each variable in place of the variable itself, we get

$$\text{nF} \cdot \frac{\text{mV}}{\text{ms}} = \text{nA} = \mu\text{S} \cdot \text{mV}. \tag{10}$$

It can be seen that units on each side of the equation match.

3.3. Physical Model - Changes

The physical model was adapted from the model that N. Szczecinski created for his M.S. Thesis [57]. As part of this thesis, Szczecinski dissected *Blaberus discoidalis* and measured the lengths of the bodies and the angles of rotation of the joints to create this model. The changes to this original model are detailed below.

First, an extra body was added to the back portion of the model. Cockroaches have large abdomens, and this fact was not correctly modeled in the original model. From a results standpoint, the purpose of adding the extra body was to bring the center of mass closer to the hind legs, which is where it is located in the animal.

Secondly, we added a passive spring and damper to each joint. The spring for each joint has the same attachment points as the flexor muscle for that joint. Insect exoskeletons possess elastic and dissipative tissues that exert very large passive forces on the body, and show over-damped motion even when all of their muscles are removed [35,36]. The spring stiffness and damping constants were tuned to mimic this observation. By adding the springs to each joint, we can model that biological observation and the result led to smoother dynamics overall.

Third, the modeling of the tarsus body segment was changed. In both models, the joint's degree of freedom was removed for simplification purposes. In Szczecinski's original model, the tarsus was attached to the tibia with a spring in between, allowing for some compliance via a passive degree of freedom. In the current model, the tarsus is rigidly attached to the tibia. The simulation freezes the foot in place when the leg is in stance phase and the foot is close to the ground. The tarsus is connected to the tibia with a free ball and socket joint to allow rotation around the frozen tarsus during stance phase. Actual cockroaches have claws that they use to grip the ground [38], so this technique, while only possible in a simulation environment, is justified. Future improvements to this mechanism will be discussed in future work.

The last major change to the physical model was to the muscle attachment points. Modeling the muscle attachment points is difficult. In Szczecinski's original model, muscle attachment points were based on major muscle groups. However, since the musculature is simplified to just one pair

of muscles for each joint, the muscle attachment points are also simplified. When going through the design process to calculate spring constants such that the model could stand without muscles, it was found that some of the springs required very large spring constants to meet the torque requirement for that joint because the muscle attachment points were too close to the joint's point of rotation. This led to requiring unrealistically strong muscle forces to control the joint. If that was the case for any joint, the muscle attachment points for that joint were modified slightly to increase the moment arm of that muscle to get more reasonable spring constant and muscle force values. Modeling of the muscle attachment points is a big area for improvement and will be discussed more in future work.

A small change was made to the way the modeled cockroach detects ground contact. Szczecinski's original model used AnimatLab 2's receptive field pair feature; the current model uses the program's Contact Sensor feature. The end result is identical: a binary signal telling us if the tarsus is in contact with the ground (or any body). Actual cockroaches have campaniform sensilla in their legs that effectively act as strain sensors and are used to detect load [4].

3.4. Neural Network as Control System

3.4.1. Central Pattern Generator

The study of Central Pattern Generators is a rich field from both a biological standpoint and a mathematical standpoint. A lot of work has been done to understand their non-linear dynamics and sensitivity to inputs, and tools have been developed to aid in the design process [31]. The CPGs in this model were designed using these tools. Therefore, an in-depth explanation of CPGs is left out of this paper. The purpose of this section is to provide a general description of their function and explain how we chose certain CPG parameters.

In insects, each joint of each leg is capable of independent oscillation, generated by a neural oscillator called a Central Pattern Generator (CPG) [2]. The purpose of the CPG is to provide the rhythmic oscillations useful for walking. The CPGs in each leg do not communicate with each other, but they are coordinated via sensory information. One CPG from each leg is connected to the same CPG in other legs to produce coordinated walking [3].

The CPG network used in this model is a non-spiking half-center oscillator. It is composed of four neurons: two half-centers and two interneurons (for a more in-depth description, see [31]). The half-centers are the oscillators: the "output" of the CPG. The interneurons relay the mutual inhibition that produces the oscillation, and also mediate sensory inputs to the CPG. The interneurons are standard non-spiking neurons similar to the neurons used for the rest of the neural networks. The half-centers differ in that they utilize optional persistent sodium channels.

The four neuron CPG model can oscillate by itself without input. However, increasing the strength of the inhibitory synapses will eventually stop oscillation [31]. This characteristic was desired because when an insect stands still, its CPGs do not oscillate. So, the strength of the inhibitory synapses were manually increased to just slightly above the threshold to stop oscillation. A CPG drive was added that inputs current to both sides of the CPG, causing oscillation. The CPG drive is turned on when the upper level signals walking and turned off when commanded to stand still.

Now that we have a CPG that can oscillate or not oscillate, it was desired to control the speed and duration of oscillation. A cockroach walks at approximately 5 steps per second [61]; so, with a tripod gait, one step lasts about 200 ms and therefore swing duration lasts approximately 100 ms. Therefore, it was desired that the CPG would oscillate every 100 ms by itself for a total period of 200 ms. We also wanted the oscillation transition to happen over a relatively quick period of time. Originally, we thought the CPG parameter values of SimRoach2 could be the same as the parameter values used for MantisBot, but it was found that the transition time of these CPGs was not fast enough for the much quicker steps that SimRoach2 takes. The strength of the CPG drive affects the speed of the transition time between oscillations, and it has been shown that the Sodium Deactivation Time Constant parameter has a large effect on the duration of oscillations. So, the process was to increase

the CPG drive strength until we got a transition speed between oscillation that was fast enough for SimRoach2, and then increase the Sodium Deactivation Time Constant until the CPG oscillated by itself approximately every 100 ms.

The last item of concern regarding the CPGs for SimRoach2 was their sensitivity to inputs. A CPG's oscillation must be stable and sensitive to inputs to allow sensory feedback to coordinate the CPGs. For the purposes of SimRoach2, the sensory feedback of interest is the load on a leg. If there is load on the leg, the CPG should be stable in the stance-phase motion (flexion or extension) and not continue to oscillate. Otherwise, the CPG should oscillate naturally as described earlier. Other inputs to the CPG include Enter Stance or Enter Swing commands that come from other parts of the network. Using previously developed design tools, the strength of the synapse of the sensory input signaling load was tuned so that the CPGs would be stable if the network signaled load [65]. The sensory synapses from the Enter Swing and Enter Stance command neurons were made stronger than the load sensory synapses to interrupt the stability of stance phase oscillation.

3.4.2. Intermediate Level Network

The Intermediate Level Network for this work was designed by N.Szczecinski for MantisBot and is explained in [60]. For these reasons, an in-depth explanation is left out of this paper and only a general description is provided.

There are two main purposes for this network. The first purpose is to receive sensory feedback and coordinate CPG oscillation. The network decides which side of the CPG to input this sensory feedback based on the direction of intended walking (forward or backward).

The second purpose is to provide a Commanded Position to the Joint Position Control network. Based on which side of the CPG is active, the network will provide a PEP or AEP as the Commanded Position. The process of calculating the PEP and AEP for each joint is based on kinematics and whether we want to walk forward or backward, and whether we want to turn. This process is integrated into our tool KinematicOrganism and is explained in [26].

3.4.3. Inter-leg Coordination

The network designed to replicate the first influence is shown in Figure 1. The Influence 1 Neuron is a copy of the influencing leg's CTr Flx HC. The CTr Flx HC activity indicates that that leg is in swing phase. So, for example, if this was the network for the left-middle leg, the Influence 1 Neuron would get a signal from the left-hind CTr Flx HC. The Influence 1 Neuron then excites the Leg Don't Enter Swing Neuron via a binary synapse with a threshold at -50 mV, halfway between the CPG equilibrium voltages of -60 mV and -40 mV. The Leg Don't Enter Swing Neuron is designed to keep the leg in stance phase if it is in stance phase, however, we do not want it to do anything if the leg is in swing phase. Therefore, the Leg's CTr Flx HC inhibits the Leg Don't Enter Swing Neuron, causing Influence 1 to do nothing if the leg is in swing phase. If the leg is in stance phase, the Leg CTr Flx HC is not active, which allows the Leg Don't Enter Swing Neuron to excite the Enter Stance Neuron and inhibit the Enter Swing Neuron. Exciting the Enter Stance Neuron sends and input current to the CPG Ext IN, keeping the leg on the ground in stance phase. Inhibiting the Enter Swing Neuron will disable this neuron. Influence 1 is stronger than the other influences. So, if this leg gets a command to Enter Swing from another inter-leg influence, and Influence 1 is active, the leg will not enter swing phase.

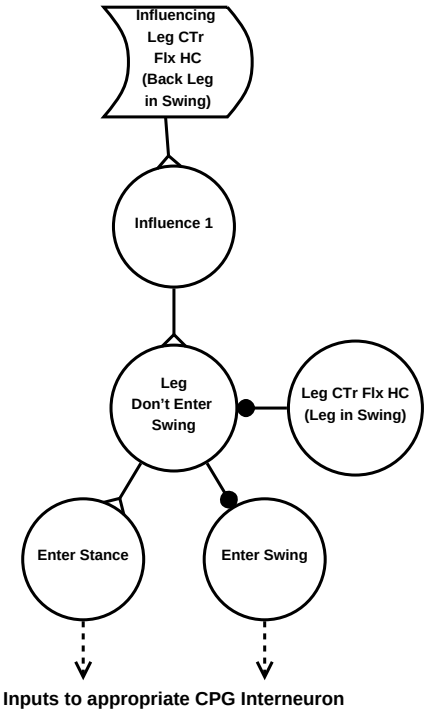

Figure 1. Influence 1 Neural Network.

Figure 2 shows how Influence 1 coordinates CTr CPG activity between a rostral (anterior) and a caudal (posterior) leg. To show Influence 1 working, data was taken from 0.4-1.0s such that a steady state tripod gate was already active. The first and last posterior leg swing phases were unperturbed, but the second posterior leg swing phase was enforced to be held for longer than nominal. This caused the anterior leg to remain in stance phase for longer than expected.

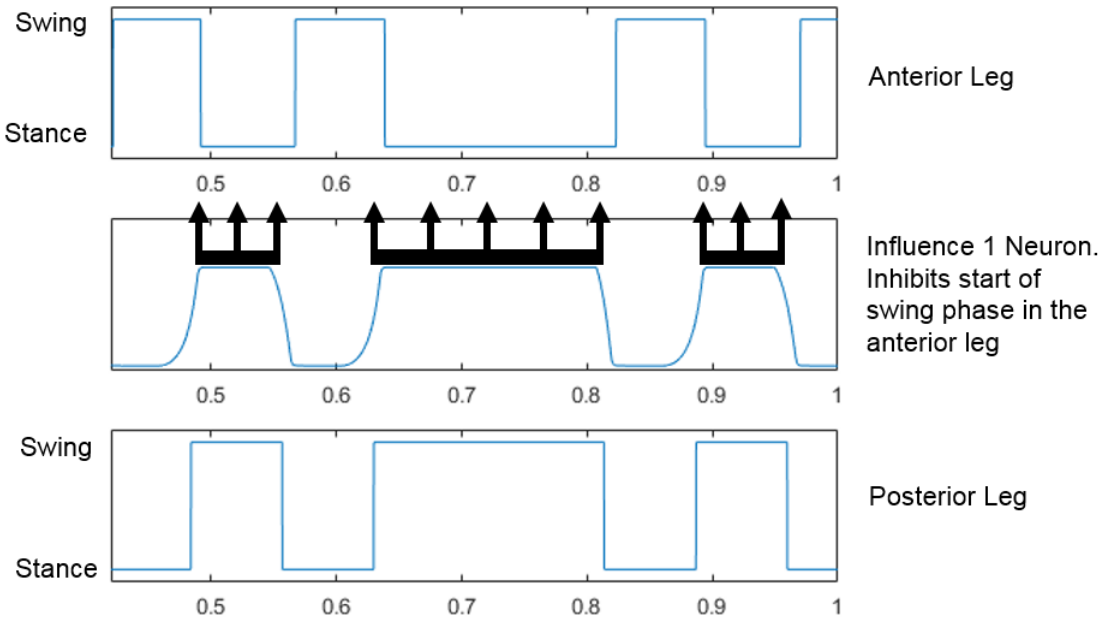

Figure 2. Influence 1 inhibits the start of swing phase in the anterior leg.

The network designed to replicate the Influence 2 is shown in Figure 3. This influence excites the start of swing phase in the rostral leg when the caudal leg enters its stance phase. It is important to note that, according to Cruse, this influence is not active throughout the duration of stance phase in the caudal leg; it is only a short burst that occurs at the start of stance phase. When the caudal leg enters stance, that leg's CPG oscillates such that CTr Ext HC becomes active. To convert the stance phase signal to a short spike, we use a differentiator network. The math behind the differentiator network is explained in [40]. The Influence 2 Neuron then sends that short spike as current to the Enter Swing Neuron for that leg. The Enter Swing Neuron will then make the leg enter swing via an input to the leg's CTr Flx HC Neuron.

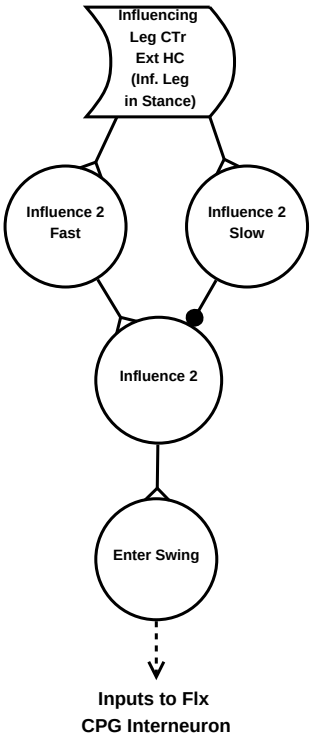

Figure 3. Influence 2 Neural Network.

Figure 4 shows how Influence 2 helps coordinate CTr CPG activity between a rostral (anterior) and a caudal (posterior) leg. To show Influence 2 working, data was taken from a steady state tripod gate; no perturbation was necessary. When the posterior leg finishes its swing phase and enters stance (around 0.7s), Influence 2 excites the start of swing phase in the anterior leg.

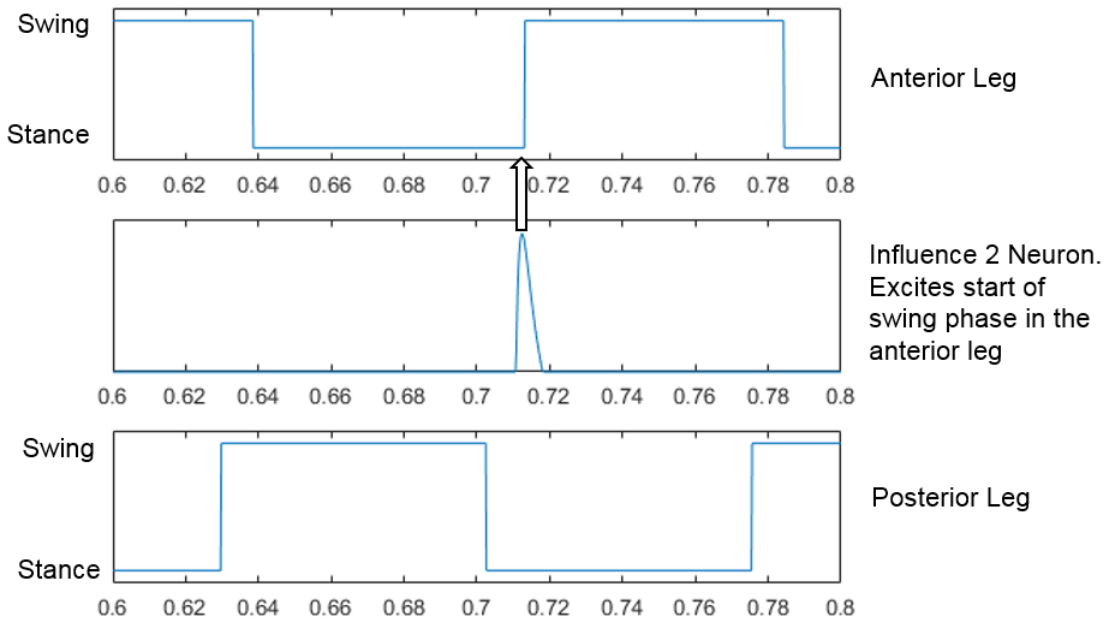

Figure 4. Influence 2 excites the start of swing phase in the anterior leg.

The network designed to replicate the last influence, Influence 3, is shown in Figure 5. This influence excites the start of swing phase in the caudal leg when the rostral leg is in its stance phase. According to Cruse, this influence increases in strength over time as the rostral leg moves farther in its stance phase. Therefore, we use an integrator network to convert the constant voltage signal of stance phase into a signal of increasing voltage. Then, the Influence 3 Neuron excites the Enter Swing Neuron for that leg. A network exists to reset the integrator when the influencing leg enters its swing phase. Also, because this influence only excites the start of swing phase, a network exists to disable the Influence 3 neuron if the influenced leg is already in swing phase.

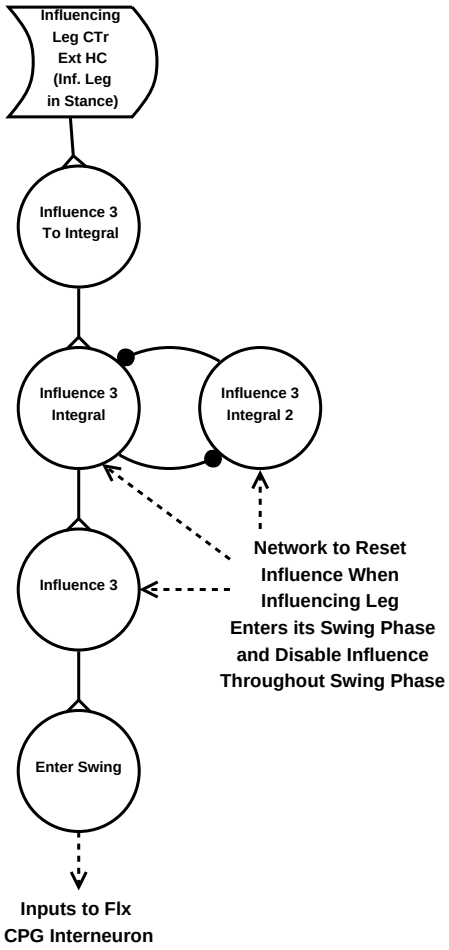

Figure 5. Influence 3.

Influences 2 and 3 are similar in that they excite the start of swing phase of an adjacent leg. The timing of these influences was important; if not properly tuned, it was possible that the influences would work against each other. For example, if a hind leg enters stance, it tells the middle leg to enter swing via Influence 2. However, during that time the middle leg was in stance, Influence 3 was building, telling the hind leg to enter swing. Therefore, if not properly timed, the hind leg could want to enter stance and tell the middle leg to enter swing with Influence 2, but Influence 3 would tell the hind leg to enter swing again, keeping the hind leg from entering stance at all. The timing of these two influences was manually tuned such that Influence 2 was “stronger" than Influence 3. Influence 2 is active during steady state walking and helps coordinate a steady state tripod gait (Figure 4). Influence 3 exists in the network, however, it only influences the gait if there is a perturbation to the gait.

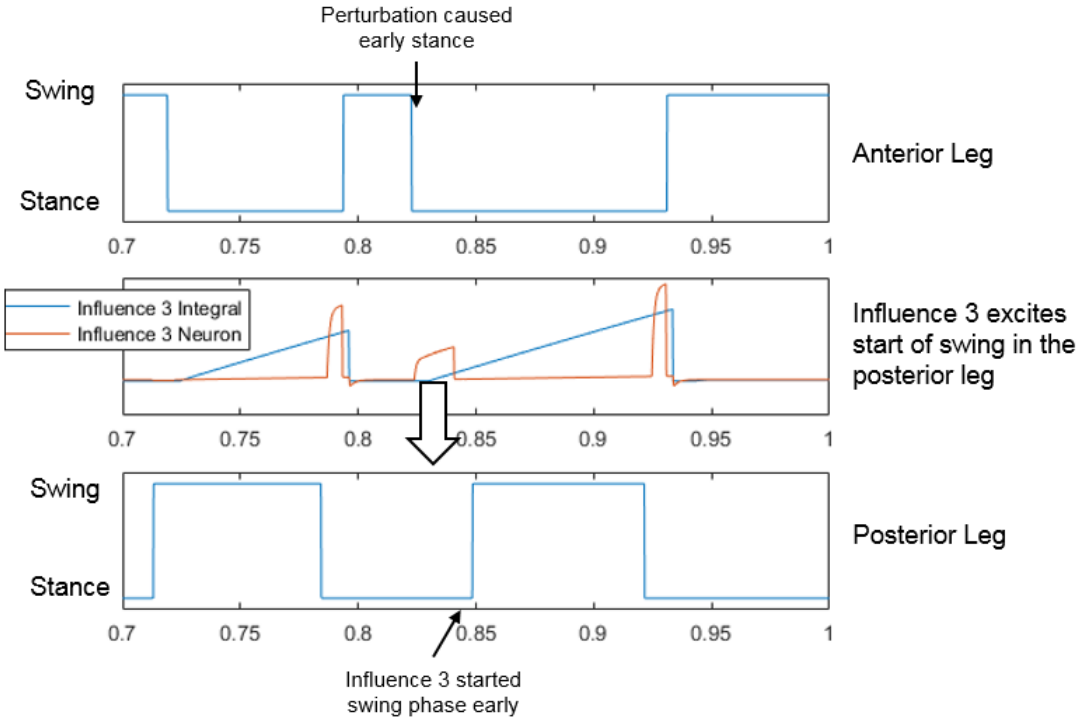

Figure 6. Influence 3 excites the start of swing phase in the posterior leg if steady state walking is perturbed.

Figure 6 shows how Influence 3 helps re-establish coordination of legs if the gait is perturbed. Around 0.82 s, we forced swing phase to end early in the anterior leg. Influence 3 then becomes active, and after a certain point causes swing phase to start early in the posterior leg. This influence only excites the start of swing phase, it does not keep a leg in swing phase; therefore, separate neurons, Influence 3 Neuron and Influence 3 Integral Neuron, are used and a network exists to disable the Influence 3 Neuron if the posterior leg is already in swing phase. The extra “bumps" of Influence 3 in Figure 6 are a result of the front leg transitions between stance and swing being slightly behind the other legs. This is a result of the rostrally directed influences and is a behavior that has been observed to exist in insects [61]. This data was taken from a front (anterior) and middle (posterior) leg. Those extra pulses telling the middle leg to enter swing are negated by Influence 1 from the hind legs. If data was taken from the middle and hind legs, those extra pulses would not exist.

As previously mentioned, Influence 3 does not activate during walking forward in this model. It would be useful if swing phase ended early; for example, climbing over and object. However, for this model swing phase is nearly constant for all legs.

3.4.4. Maintaining Ground Contact

Figure 7 shows the network designed to change the property adapter for each tarsus. If the leg is in swing phase, then the CTr Flx HC is active and the Leg Loaded and In Stance Neuron is inhibited regardless of whether there is load on the leg or not. If that is the case, the tarsus is not frozen. If the leg is in stance phase, then the CTr Flx HC is not active, so there is no inhibition on the Leg Loaded and In Stance Neuron. So, when the leg is loaded, the Leg Loaded Neuron excites the Leg Loaded and In Stance Neuron, causing the tarsus to be attached to the ground. The tarsus is unattached from the ground as soon as the leg enters swing phase (CTr Flx HC becomes active). This is similar to the positive force feedback control of tarsal claw grip observed in insects [37].

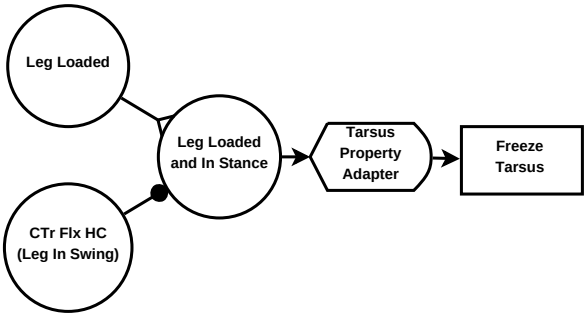

Figure 7. The Tarsus is frozen in place when the leg is loaded and when the leg is not in swing phase (i.e. leg is in stance phase).

3.4.5. Height Control

Figure 8 shows the neural network that was implemented in each leg. This network is only active during stance phase, and resets itself during every swing phase. The y-position (vertical height) of the tarsus and the coxa are linearly mapped to neuron voltages via the height adapter. Those heights are then compared via a subtraction network to get the Perceived Height Difference. The Commanded Height Difference is calculated based on the rest posture and the physical geometry of the model (coxa body attachment location for each leg are not all located at the same y-position relative to the main body). The Commanded Height Difference and Perceived Height Difference are then subtracted from each other to get two difference neurons: one if the Perceived Height Difference is greater than the Commanded Height Difference and one if the Perceived Height Difference is less than the Commanded Height Difference. This works exactly the same way as the Joint Control Network's Commanded Position and Perceived Position Neurons. Only one of these difference neurons can be active at a time. If one is active, current is injected into the marginally stable integral neuron, which increases or decreases the CTr PEP Position until the Perceived Height Difference and Commanded Height Difference are equal. Additionally, the Height Control loop should only be active during stance phase. Therefore, the CTr Flx HC was used to inhibit the height difference neurons. So, if the leg was in swing phase, the difference neurons would be deactivated, no current can be input to the integral loop, and the CTr PEP would not be adjusted. Lastly, we want to reset the CTr PEP each step, so a network was designed to reset the Height Control Integral Neuron's while the leg was in swing phase.

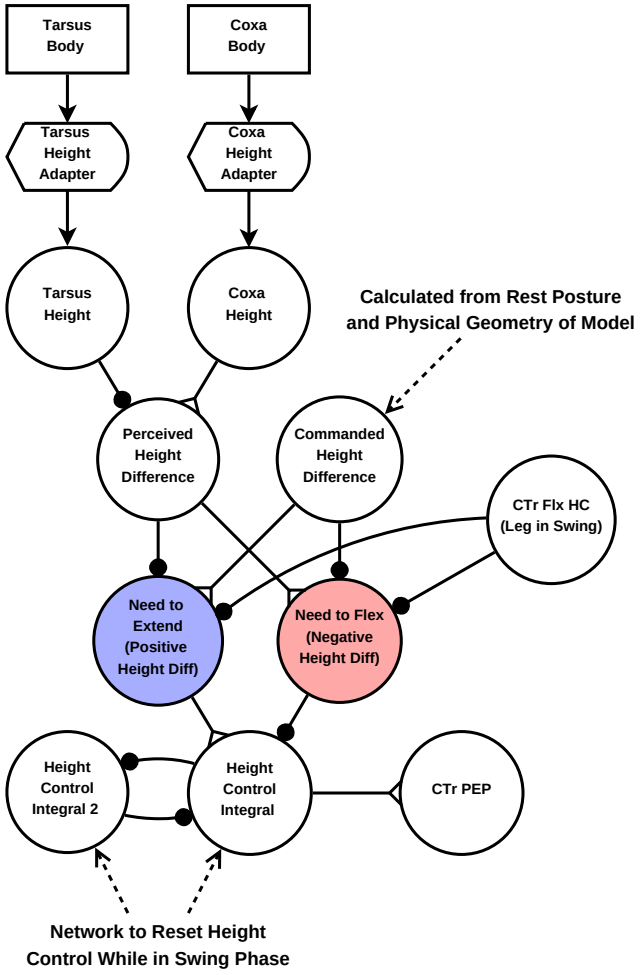

Figure 8. Height Control Neural Network.

3.4.6. Upper Level Network

One of the major concepts in the development of our synthetic nervous systems is to minimize descending commands from the upper level network. Therefore, this network is relatively simple. Essentially, it acts like a joy stick, telling the organism whether to go forward or backward and left or right [11].

Figure 9 shows a diagram of how this network is constructed. The PEP Positive Translation and PEP Negative Translation Neurons correspond to walking forward and backward, respectively. The PEP Positive Rotation and PEP Negative Rotation Neurons correspond to walking left and right, respectively. The inputs are received by the PEP Positive Translation and PEP Positive Rotation Neurons. If there is no input, all four of these neurons reach an equilibrium at -50 mV. The synapses between these four neurons and the No Rotation and No Translation Neurons are binary with a threshold of -49.9 mV. Therefore, if there are no inputs, the No Rotation and No Translation neurons are both active. If both of these neurons are active, than the Stand Still Neuron is active (the Stand Still Neuron has a negative tonic stimulus such that it will not be active unless both the No Rotation and No Translation neurons are active). A positive input to the PEP Positive Translation neuron corresponds to walking forward; a negative input corresponds to walking backward. If either of these is the case, either the PEP Positive Translation or PEP Negative Translation will be active, inhibiting the No Translation neuron and therefore deactivating the Stand Still neuron. The rotation part works in exactly the same way. Because of the negative tonic stimulus in the Stand Still neuron, it will only be active if both the No Translation and No Rotation neurons are active.

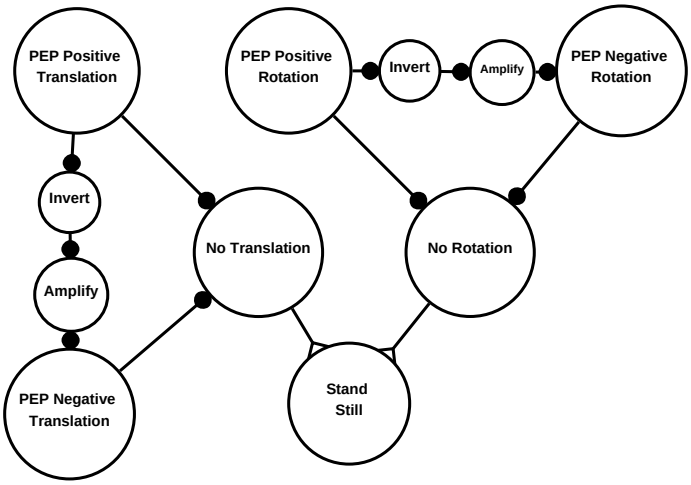

Figure 9. Upper Level Network.

Stand Still Neuron

The Stand Still Neuron is a descending neuron that inhibits as few parts of the thoracic ganglia as possible to halt walking. Functionally speaking, it causes every leg to enter stance phase, it inhibits rhythmic motor output, it inhibits excitatory influences between the ganglia, and it inhibits height control networks. It accomplishes these via the following: it inhibits CTr levation by stabilizing each CPG in the depression phase; it inhibits excitatory drive to the CPGs, halting their oscillation; it inhibits pathways that communicate inter-leg influence 3, which cause legs to enter swing phase; and it disables pathways that adjust CTr depression to control body height. Such a neuron may represent the bulk activity of the supra-esophageal ganglion.

3.5. Design Process for a Single Joint

3.5.1. Calculation of Passive Spring Stiffness and Damping Coefficients

Stiffness Coefficient

To begin the calculation of the stiffness coefficient using our method we first must calculate the required torque on each joint to hold the leg in place under the weight of gravity. Each leg is assumed to hold an equal amount of the weight of the body. We use the manipulator Jacobian method described in [49] to calculate these torques.

The spatial frame is defined at the center of mass of the front body (Figure 10). The tool frame for each leg is defined at the tarsus for each leg.

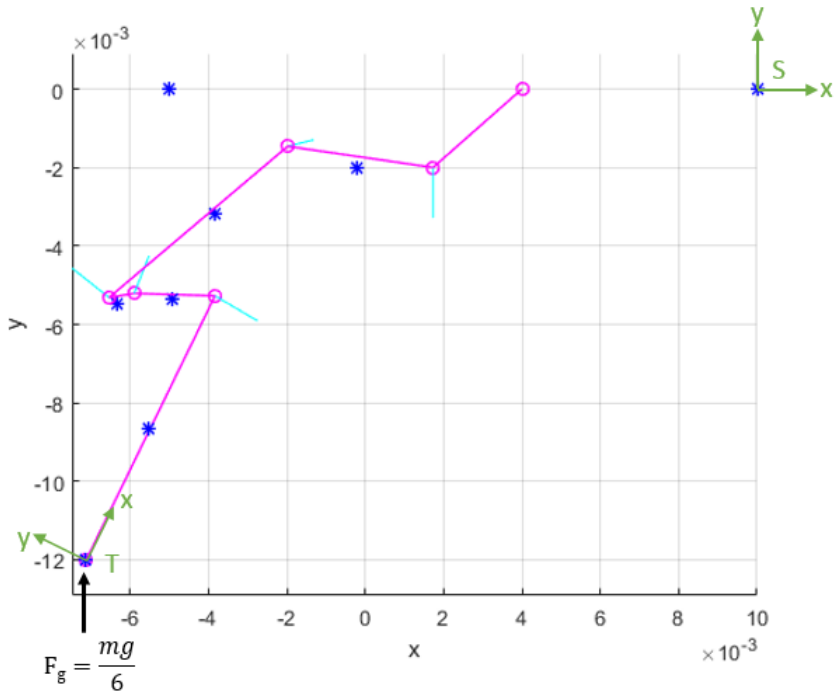

Figure 10. Diagram of one leg in rest posture with spatial and tool frames labeled. The tool frame is defined at the tarsus of each leg. The spatial frame is defined at the center of mass of the front body. An end-effector force is applied from the ground to the tarsus.

To calculate the torque in each joint to maintain a static posture under the load of an end effector force, we use Equation 3.60 from [49]

$$\tau = (J^s_{st})^T F_s, \tag{11}$$

where $(J^s_{st})^T$ is the transpose of the spatial Jacobian and F_s is a wrench applied at the end-effector in spatial coordinates. In this case, the end effector is the tarsus. Our tool KinematicOrganism uses $(J^s_{st})^T$ for other calculations and thus was already known. So, to find the required torque in each joint, we only need to calculate the end-effector wrench in spatial coordinates.

To do that, we start with the actual force acting on the end effector. This is the force from the ground pushing on the tarsus and is equal to one-sixth the weight of the body.

$$F_g = \begin{bmatrix} 0 \\ \frac{mg}{6} \\ 0 \end{bmatrix}. \tag{12}$$

This force acts at the center of the tool frame, so, the wrench in the tool frame has no rotational component. To construct the wrench in the tool frame, however, we need to convert the linear component, the force from the ground, from spatial frame to tool frame coordinates. The rotational matrices were previously incorporated into our tool KinematicOrganism. So, the wrench in tool coordinates is

$$F_t = \begin{bmatrix} R_{ts} F_g \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{13}$$

The wrench in the tool frame needs to be converted to the spatial frame to use Equation 11. To do this, we use Equation 2.65 from [49]

$$F_s = Ad^T_{g_{ts}} F_t. \tag{14}$$

After calculating the wrench in the spatial frame, we use Equation 11 to solve for the torques required for static posture. Because the springs' rest lengths are set such that the spring will produce zero force at the rest posture, we must rotate the joint a small amount for the spring to produce the required torque. This small amount was chosen to be 0.1 radians. From there, using our tool KinematicOrganism, we used an optimizer to increase the value of the spring constant k until each slightly stretched spring produced the calculated required torque.

Damping Coefficient

The damping coefficient for each spring was calculated with the idea that the joint should be at least critically damped for every possible movement of the system with just the spring (i.e. no muscles). The damping value for a critically damped joint is not only a function of the stiffness and inertia but also a function of the angle between the force that the spring is acting and the joint. This is ultimately then a function of joint angle θ.

For a rotational joint with a rotational spring and damper, the damping coefficient is related to the stiffness and inertia of the joint by

$$c_t = 2\xi\omega_n I, \tag{15}$$

where I is the inertia of limb and ω_n is the natural frequency of the system defined as

$$\omega_n = \sqrt{(k_T / I)}. \tag{16}$$

If we enforce $\xi = 1$, then if we know the inertia and equivalent torsional stiffness, we can calculate a torsional damping value to critically damp the system.

Our system, however, uses a linear spring and damper (Figure 11). To convert our linear spring to a torsional spring, we use the idea of equivalent energy.

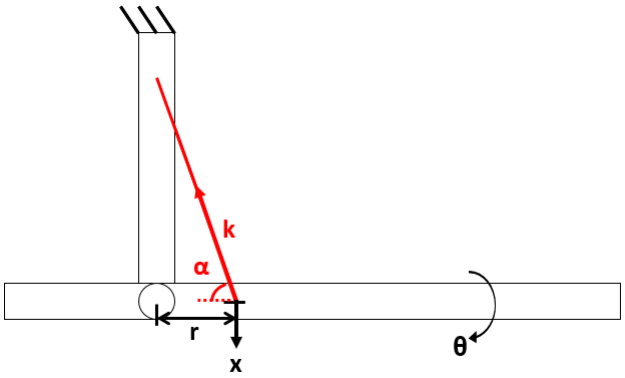

Figure 11. Concept of equivalent energy is used to convert between linear and torsional stiffness and damping values.

$$\frac{1}{2}k_t\theta^2 = \frac{1}{2}kx^2\sin^2(\alpha). \tag{17}$$

We need the $\sin(\alpha)$ term to account for the fact that the spring is not acting perpendicular to this joint. For small rotations, $x = r\theta$, so

$$\frac{1}{2}k_t\theta^2 = \frac{1}{2}kr^2\theta^2\sin^2(\alpha), \tag{18}$$

and by canceling terms,

$$k_t = kr^2\sin^2(\alpha). \tag{19}$$

Also, once we calculate a torsional damping value, it needs to be converted to a linear damping value. This is done using an identical process and it can be shown that

$$c = \frac{c_t}{r^2\sin^2(\alpha)}. \tag{20}$$

To enforce the idea that the joint should be critically damped for every movement, we calculate many values of c over the joint's angles of possible rotations and pick the largest one. This is important because the angle of each joint affects the $\sin(\alpha)$ term, which affects whether our system is critically damped or not. The calculated value of c also depends on the inertia of the joint. For the most distal joint, this value is constant, however, for other joints, it depends on the joint angle of distal joints.

So, the process starts with the most distal joint in each leg. The inertia of that joint is always constant. We rotate the limb over the joint's limits of rotations and use our tool KinematicOrganism to calculate α, the angle between the joint and the spring. We then use Equation 15, Equation 16, Equation 19, and Equation 20 to find the associated c value to critically damp the system at that joint angle. We then pick the largest calculated value of c for that joint.

For other joints, we must first find the joint angles of the more distal joints that maximize the inertia of the current joint. For most cases, this involves fully extending the distal joints. Then, the same process of rotating the current joint and picking the maximum calculated value of c is done.

Our calculated values were tested in AnimatLab 2 and worked as intended (Figure 12). Joints were forced to positions away from equilibrium and allowed to passively rotate to their equilibrium position. For these tests, the rest length of the spring was changed to allow to joint to rotate to joint positions that were more susceptible to becoming under-damped based on the angle between the spring and the joint α. The results of one of these tests is shown in Figure 12. Our calculated value critically damps the system at the equilibrium position that is most prone to becoming under-damped. Lowering the value of the damping coefficient slightly results in a slightly under-damped system.

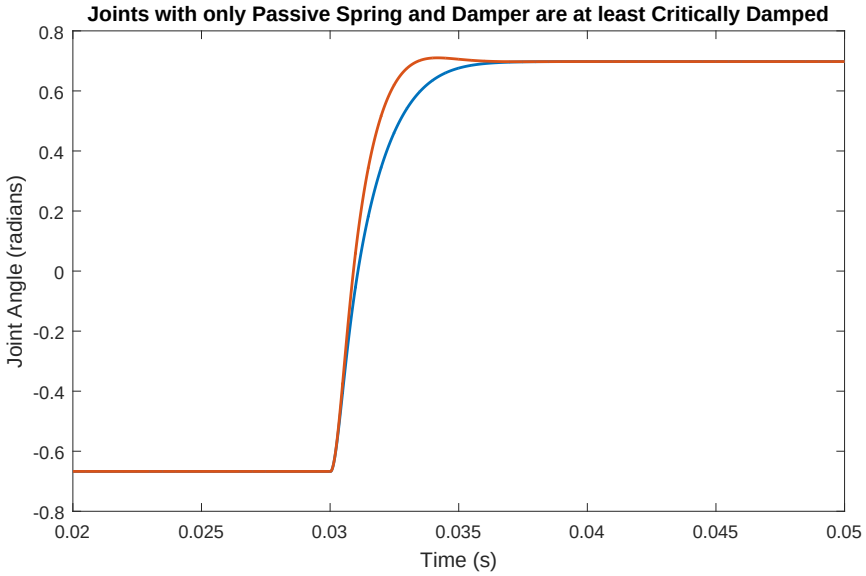

Figure 12. Joints with only Passive Spring and Damper are at least Critically Damped. Joints were forced to positions away from equilibrium and allowed to passively rotate to equilibrium. Our calculated value for the dampers are always at least critically damped (orange). Lowering the damping value by 10% resulted in a slightly under-damped system (blue). For this joint (LM FTi), because of the spring attachment point geometry it was found that the larger joint angles were more susceptible to becoming under-damped, so data is shown for that portion of the range of motion.

ThC2 Joints

For each leg, the calculated values for the stiffness and damping coefficients using the above process for the ThC2 joints were not used. Instead, the values for the ThC1 joint for that leg or smaller values were used. This is because the ThC2 joints are all nearly vertical in the model. Because these joints are oriented almost parallel to the applied force, the end-effector wrench and Jacobian method resulted in large torque requirements, leading to unrealistically large stiffness coefficients for these joints. It's interesting to note that the muscles and network could still be tuned to control these joints using the method outlined below, however, it took abnormally strong muscles producing unrealistically large forces to control these joints. Because of this, we decided to use the same values calculated from the ThC1 joints for each leg or other smaller values. The joints are close together, so we hypothesized that the biomechanics providing the passive stiffness were similar.

Muscle Attachment Points

The original muscle attachment points used in Szczecinski's model for his M.S. Thesis were adjusted at this stage in the process. For some joints besides the ThC2 joints, the numbers calculated using this method resulted in very large stiffness and damping values. For these joints, the muscle attachment points were adjusted to result in more realistic stiffness and damping coefficients. Generally, the muscle attachment points for these joints were very close to the limb, so the spring had a small moment arm and a larger stiffness was required to produce the torque necessary to hold a static posture without muscles. So, the muscle attachment points were moved slightly away from the joint's point of rotation until reasonable values for the stiffness and damping coefficients were calculated.

It was found that the muscle attachment points greatly affected the calculated stiffness and damping coefficients (and the entire system in general). Modeling of muscle attachment points is an important area of improvement and will be discussed more in future work.

3.5.2. Passive Muscle Force Parameters

After tuning the spring coefficients, the next step was to add passive muscle components. The parameters of interest for this section are muscle parameters k_{se}, k_{pe}, and b. That is, the stiffness of the series and parallel components and the damping in the muscle. Ideally, we could come up with a set of values that are biologically accurate, controllable, stable, and robust. In the paper, we state that we chose specific values for these coefficients. In this section, we provide more detail on why we simply chose specific values.

Values for these parameters vary widely from muscle-to-muscle and even from organism-to-organism [50]. Different sources cite different values of these parameters measured on the same muscle in the same joint (Figure 3 in [50], Figure 6 in [51]). In addition, large scale data is not available for every muscle in every joint in the cockroach, and even if there was, using average values as a model would lead to an inaccurate model [50]. When choosing these values, we need to think on an individual-to-individual basis, not a muscle-to-muscle basis and definitely not use large data averages. No individualized data is available for all of these parameter values for all of the muscles in one cockroach. Therefore, we needed to develop a way to pick these parameter values.

Adding passive muscles complicated the dynamics, however, it was not possible to destabilize the system. Also, it was found that after going through the rest of this design process, the performance of the system using different parameter combinations of k_{se}, k_{pe}, and b was almost identical. This was an important observation. Our Joint Position Controller can be optimized to control the muscles regardless of their passive parameter values. So, picking these values became less of a question of system performance and more of a question of picking parameter values that would induce the least number of additional problems. The biggest problem that a "bad" set of parameter values could result in was simulation instability.

It was found that if the value of b was too small or too large, the simulation required an extremely small time step to run a stable simulation. The system itself was stable, but the simulation was not. While this problem can be solved by simply lowering the time step, that solution was undesirable. Because we are dealing with a small cockroach will small inertia values, a small time step was already necessary to begin with. Lowering that time step even more would result in impractical simulation run times.

Looking at the equation for muscle tension, this make sense. If the value of b is too small compared to k_{se}, then the we would be multiplying everything in parenthesis by a very large value. This would result in a very large $\dot{T}$ that would destabilize the simulation at the next time step unless the time step was short enough.

Large b values alone are not the solution to this issue, though. Yes, a larger b value lowers the number that everything in parenthesis is multiplied by. But, it also increased the magnitude of the $b\dot{x}$ term. For small b values, this term was a non-issue, but for relatively larger b values, this term had the potential to blow up the simulation. These joints are moving fairly quickly, so the $\dot{x}$ terms can get relatively high. So, for larger b values, especially when paired with a relatively large k_{se} value, the simulation was also unstable.

The relative size of the passive parameters, especially the size between k_{se} and b, was an important factor in simulation stability. So, for SimRoach2, we picked values of k_{se} and k_{pe}, and then picked a reasonable value of b to keep the simulation stable at the current time step.

As previously mentioned, the values of k_{se} and k_{pe} vary greatly from muscle-to-muscle and from organism-to-organism. Guschlbauer cited k_{se} and k_{pe} values of approximately 6.32 N/m and 3.156 N/m (Figure 6 in [51]) while Blumel cited k_{se} and k_{pe} values of approximately 45 N/m and 11.24 N/m (Figure 3 in [50]). Knowing that either or any parameter choice would result in the same system perforance after optimization, we chose k_{se} and k_{pe} values of 45 N/m and 11.24 N/m.

These values of k_{se} and k_{pe} were small enough to not induce any of the simulation stability issues associated with a large b value. However, if b was too large, we would have an extremely over-damped system that would require large muscle forces to control. As mentioned before, if the b value was

too small relative to these values of k_{se} and k_{pe}, the simulation would be unstable. A b value of 0.1 Ns/m was chosen as a good value to keep the system from being too over-damped while keeping the simulation stable, eliminating the need to lower the time step.

3.5.3. Active Muscle Parameters

Figure 13 shows an example of what one of our muscle activation curves looks like.

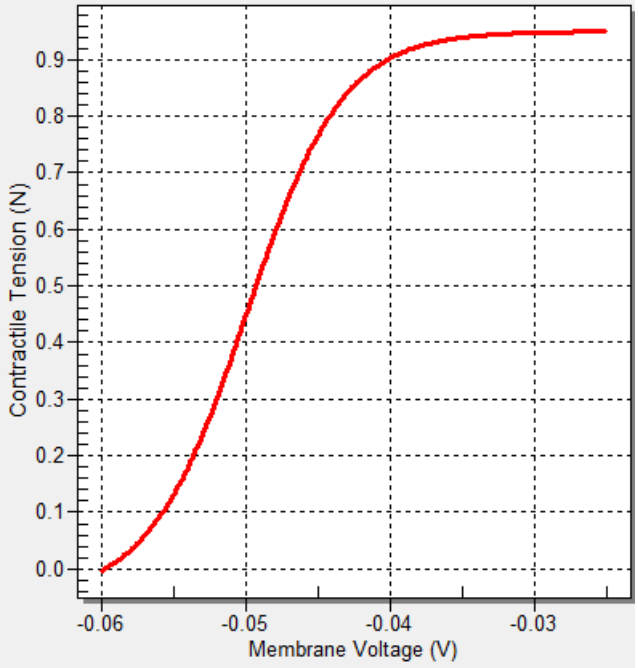

Figure 13. Example of one muscle activation curve used in this model. In this case, the amplitude of the curve is 1 N. The rest of the parameters are constrained such that the curve looks similar to biological data (See Figure 5 from [50]).

3.6. Results: Measuring the AEP and PEP During Walking

For each joint in each leg during walking, the joint angle at the AEP and PEP were measured. To get these measurements, we looked at the beginning point and ending point of stance phase for forward walking. Stance phase was defined as when that leg supported some body weight. Steps that were considered too short were removed from this analysis. Simply taking the data points at the exact point of stance to swing transitions was not robust and lead to large variances in the data. Therefore, we looked at a small window around the transition points and took either the maximum or the minimum value. If the joint extended during stance phase, we took the minimum of the AEP window and the maximum of the PEP window. If the joint flexed during stance phase, we took the maximum of the AEP window and the minimum of the PEP window. The $\Delta\theta$ row was calculated by subtracting the AEP from the PEP.

References

1. Ryckebusch, S.; Laurent, G. Rhythmic patterns evoked in locust leg motor neurons by the muscarinic agonist pilocarpine. *J. Neurophysiol.* **1993**, *69*, 1583–1595.
2. Büschges, A.; Schmitz, J.; Bässler, U. Rhythmic patterns in the thoracic nerve cord of the stick insect induced by pilocarpine. *J. Exp. Biol.* **1995**, *198*, 435–456.
3. Ryckebusch, S.; Laurent, G. Interactions between segmental leg central pattern generators during fictive rhythms in the locust. *J. Neurophysiol.* **1994**, *72*, 2771–2785.

4. Noah, J.A.; Quimby, L.; Frazier, S.F.; Zill, S.N. Sensing the effect of body load in legs: Responses of tibial campaniform sensilla to forces applied to the thorax in freely standing cockroaches. *J. Comp. Physiol. A* **2004**, *190*, 201–215.
5. Zill, S.; Schmitz, J.; Büschges, A. Load sensing and control of posture and locomotion. *Arthropod Struct. Dev.* **2004**, *33*, 273–286.
6. Akay, T.; Haehn, S.; Schmitz, J.; Büschges, A. Signals From Load Sensors Underlie Interjoint Coordination During Stepping Movements of the Stick Insect Leg. *J. Neurophysiol.* **2004**, *92*, 42–51.
7. Bucher, D.; Akay, T.; DiCaprio, R.a.; Buschges, A. Interjoint coordination in the stick insect leg-control system: The role of positional signaling. *J. Neurophysiol.* **2003**, *89*, 1245–55.
8. Hess, D.; Büschges, A. Role of Proprioceptive Signals From an Insect Femur-Tibia Joint in Patterning Motoneuronal Activity of an Adjacent Leg Joint. *J. Neurophysiol.* **1999**, *81*, 1856–1865.
9. Mu, L.; Ritzmann, R.E. Kinematics and motor activity during tethered walking and turning in the cockroach, Blaberus discoidalis. *J. Comp. Physiol. A* **2005**, *191*, 1037–1054.
10. Hellekes, K.; Blincow, E.; Hoffmann, J.; Buschges, A. Control of reflex reversal in stick insect walking: Effects of intersegmental signals, changes in direction, and optomotor-induced turning. *J. Neurophysiol.* **2012**, *107*, 239–249.
11. Martin, J.P.; Guo, P.; Mu, L.; Harley, C.M.; Ritzmann, R.E. Central-Complex Control of Movement in the Freely Walking Cockroach. *Curr. Biol.* **2015**, *25*, 2795–2803.
12. Buschmann, T.; Ewald, A.; von Twickel, A.; Büschges, A. Controlling legs for locomotion-insights from robotics and neurobiology. *Bioinspir. Biomim.* **2015**, *10*, 041001.
13. Daun-Gruhn, S. A mathematical modeling study of inter-segmental coordination during stick insect walking. *J. Comput. Neurosci.* **2011**, *30*, 255–278.
14. Szczecinski, N.S.; Brown, A.E.; Bender, J.A.; Quinn, R.D.; Ritzmann, R.E. A neuromechanical simulation of insect walking and transition to turning of the cockroach Blaberus discoidalis. *Biol. Cybern.* **2014**, *108*, 1–21.
15. Cruse, H.; Kindermann, T.; Schumm, M.; Dean, J.; Schmitz, J. Walknet - A biologically inspired network to control six-legged walking. *Neural Netw.* **1998**, *11*, 1435–1447.
16. Schilling, M.; Paskarbeit, J.; Hoinville, T.; Hüffmeier, A.; Schneider, A.; Schmitz, J.; Cruse, H. A hexapod walker using a heterarchical architecture for action selection. *Front. Comput. Neurosci.* **2013**, *7*, 126.
17. Ekeberg, Ö.; Blümel, M.; Büschges, A. Dynamic simulation of insect walking. *Arthropod Struct. Dev.* **2004**, *33*, 287–300.
18. Rutter, B.L.; Taylor, B.K.; Bender, J.A.; Blümel, M.; Lewinger, W.A.; Ritzmann, R.E.; Quinn, R.D. Descending commands to an insect leg controller network cause smooth behavioral transitions. In Proceedins of the IEEE International Conference on Intelligent Robots and Systems, San Francisco, CA, USA, 25–30 September 2011; pp. 215–220.
19. Daun-Gruhn, S.; Tóth, T.I. An inter-segmental network model and its use in elucidating gait-switches in the stick insect. *J. Comput. Neurosci.* **2011**, *31*, 43–60.
20. Toth, T.I.; Schmidt, J.; Büschges, A.; Daun-Gruhn, S. A Neuro-Mechanical Model of a Single Leg Joint Highlighting the Basic Physiological Role of Fast and Slow Muscle Fibres of an Insect Muscle System. *PLoS ONE* **2013**, *8*, e78247.
21. Tóth, T.I.; Grabowska, M.; Rosjat, N.; Hellekes, K.; Borgmann, A.; Daun-Gruhn, S. Investigating inter-segmental connections between thoracic ganglia in the stick insect by means of experimental and simulated phase response curves. *Biol. Cybern.* **2015**, *109*, 349–362.
22. Schilling, M.; Hoinville, T.; Schmitz, J.; Cruse, H. Walknet, a bio-inspired controller for hexapod walking. *Biol. Cybern.* **2013**, *107*, 397–419.
23. Schmitz, J.; Schneider, A.; Schilling, M.; Cruse, H. No need for a body model: Positive velocity feedback for the control of an 18-DOF robot walker. *Appl. Bionics Biomech.* **2008**, *5*, 135–147.
24. Dupeyroux, J.; Passault, G.; Ruffier, F.; Viollet, S.; Serres, J. Hexabot: A small 3D-printed six-legged walking robot designed for desert ant-like navigation tasks. In Proceedings of the IFAC Word Congress, Toulouse, France, 9–14 July 2017.
25. Szczecinski, N.S.; Getsy, A.P.; Martin, J.P.; Ritzmann, R.E.; Quinn, R.D. Mantisbot is a robotic model of visually guided motion in the praying mantis. *Arthropod Struct. Dev.* **2017**, *46*, 736–751.
26. Szczecinski, N.S.; Quinn, R.D. Template for the neural control of directed stepping generalized to all legs of MantisBot. *Bioinspir. Biomim.* **2017**, *12*, 045001.

27. Gruhn, M.; Hoffmann, O.; Dübbert, M.; Scharstein, H.; Büschges, A. Tethered stick insect walking: A modified slippery surface setup with optomotor stimulation and electrical monitoring of tarsal contact. *J. Neurosci. Methods* **2006**, *158*, 195–206.
28. Durr, V. The behavioural transition from straight to curve walking: Kinetics of leg movement parameters and the initiation of turning. *J. Exp. Biol.* **2005**, *208*, 2237–2252.
29. Cruse, H. What mechanisms coordinate leg movement in walking arthropods? *Trends Neurosci.* **1990**, *13*, 15–21.
30. Cofer, D.; Cymbalyuk, G.; Reid, J.; Zhu, Y.; Heitler, W.J.; Edwards, D.H. AnimatLab: A 3D graphics environment for neuromechanical simulations. *J. Neurosci. Methods* **2010**, *187*, 280–288.
31. Szczecinski, N.S.; Hunt, A.J.; Quinn, R.D. Design process and tools for dynamic neuromechanical models and robot controllers. *Biol. Cybern.* **2017**, *111*, 105–127.
32. Hodgkin, A.L.; Huxley, A.F. A quantitative description of membrane current and its application to conduction and excitation in nerve. *Bull. Math. Biol.* **1990**, *52*, 25–71.
33. Hill, A.V. The Heat of Shortening and the Dynamic Constants of Muscle. *Proc. R. Soc. B Biol. Sci.* **1938**, *126*, 136–195.
34. Shadmehr, R.; Arbib, M.A. A mathematical analysis of the force-stiffness characteristics of muscles in control of a single joint system. *Biol. Cybern.* **1992**, *66*, 463–477.
35. Hooper, S.L.; Guschlbauer, C.; Blumel, M.; Rosenbaum, P.; Gruhn, M.; Akay, T.; Buschges, A. Neural Control of Unloaded Leg Posture and of Leg Swing in Stick Insect, Cockroach, and Mouse Differs from That in Larger Animals. *J. Neurosci.* **2009**, *29*, 4109–4119.
36. Ache, J.M.; Matheson, T. Passive Joint Forces Are Tuned to Limb Use in Insects and Drive Movements without Motor Activity. *Curr. Biol.* **2013**, *23*, 1418–1426.
37. Zill, S.N.; Chaudhry, S.; Exter, A.; Büschges, A.; Schmitz, J. Positive force feedback in development of substrate grip in the stick insect tarsus. *Arthropod Struct. Dev.* **2014**, *43*, 441–455.
38. Paskarbeit, J.; Otto, M.; Schilling, M.; Schneider, A. Stick(y) Insects—Evaluation of Static Stability for Bio-inspired Leg Coordination in Robotics. In Proceedings of the Conference on Biomimetic and Biohybrid Systems, Edinburgh, UK, 19–22 July 2016; Volume 1, pp. 239–250.
39. Ramdya, P.; Thandiackal, R.; Cherney, R.; Asselborn, T.; Benton, R.; Ijspeert, A.J.; Floreano, D. Climbing favours the tripod gait over alternative faster insect gaits. *Nat. Commun.* **2017**, *8*, 14494.
40. Szczecinski, N.S.; Hunt, A.J.; Quinn, R.D. A functional subnetwork approach to designing synthetic nervous systems that control legged robot locomotion. *Front. Neurorobot.* **2017**, *11*, 1–19.
41. Bässler, D.; Büschges, A.; Meditz, S.; Bässler, U. Correlation between muscle structure and filter characteristics of the muscle-joint system in three orthopteran insect species. *J. Exp. Biol.* **1996**, *199*, 2169–83.
42. Wolf, H. Inhibitory motoneurons in arthropod motor control: Organisation, function, evolution. *J. Comp. Physiol. A* **2014**, *200*, 693–710.
43. Hooper, S.L.; Guschlbauer, C.; von Uckermann, G.; Buschges, A. Different Motor Neuron Spike Patterns Produce Contractions With Very Similar Rises in Graded Slow Muscles. *J. Neurophysiol.* **2007**, *97*, 1428–1444.
44. Garcia-Sanz, M. Chapter 3 P.I.D. control: Structure. In *EECS 475 Applied Control*, Case Western Reserve University: Cleveland, OH, USA, 2016.
45. Dürr, V.; Schmitz, J.; Cruse, H. Behaviour-based modelling of hexapod locomotion: Linking biology and technical application. *Arthropod Struct. Dev.* **2004**, *33*, 237–250.
46. Mantziaris, C.; Bockemühl, T.; Holmes, P.; Borgmann, A.; Daun, S.; Büschges, A. Intra- and intersegmental influences among central pattern generating networks in the walking system of the stick insect. *J. Neurophysiol.* **2017**, *118*, 2296–2310.
47. Cruse, H.; Riemenschneider, D.; Stammer, W. Control of body position of a stick insect standing on uneven surfaces. *Biol. Cybern.* **1989**, *61*, 71–77.
48. Cruse, H.; Schmitz, J.; Braun, U.; Schweins, A. Control of body height in a stick insect walking on a treadwheel. *J. Exp. Biol.* **1993**, *181*, 141–155.
49. Murray, R.M.; Li, Z.; Sastry, S.S. *A Mathematical Introduction to Robotic Manipulation*; CRC Press: Boca Raton, FL, USA, 1994.
50. Blumel, M.; Hooper, S.L.; Guschlbauer, C.; White, W.E.; Buschges, A. Determining all parameters necessary to build Hill-type muscle models from experiments on single muscles. *Biol. Cybern.* **2012**, *106*, 543–558.
51. Guschlbauer, C.; Scharstein, H.; Büschges, A. The extensor tibiae muscle of the stick insect: Biomechanical properties of an insect walking leg muscle. *J. Exp. Biol.* **2007**, *210*, 1092–108.

52. Blümel, M.; Guschlbauer, C.; Daun-Gruhn, S.; Hooper, S.L.; Büschges, A. Hill-type muscle model parameters determined from experiments on single muscles show large animal-to-animal variation. *Biol. Cybern.* **2012**, *106*, 559–571.
53. Hooper, S.L.; Guschlbauer, C.; Blümel, M.; von Twickel, A.; Hobbs, K.H.; Thuma, J.B.; Büschges, A. *Muscles: Non-Linear Transformers of Motor Neuron Activity*; Springer Series in Computational Neuroscience; Springer: New York, NY, USA, 2016; pp. 163–194.
54. Rassier, D.; MacIntosh, B.; Herzog, W. Length dependence of active force production in skeletal muscle. *J. Appl. Physiol.* **1999**, *86*, 1445–1457
55. Schroer, R.; Boggess, M.; Bachmann, R.; Quinn, R.; Ritzmann, R. Comparing cockroach and Whegs robot body motions. In Proceedings of the IEEE International Conference on Robotics and Automation, New Orleans, LA, USA, 26 April–1 May 2004; Volume 4, pp. 3288–3293.
56. Hughes, G.M. The Co-ordination of Insect Movements. *J. Exp. Biol.* **1952**, *29*, 267–285.
57. Szczecinski, N.S. Massively Distributed Neuromorphic Control for Legged Robots Modeled after Insect Stepping. Master's Thesis, Case Western Reserve University, Cleveland, OH, USA, 2013.
58. Rubeo, S.E. Control of Simulated Cockroach Using Synthethic Nervous Systems. Master's Thesis, Case Western Reserve University, Cleveland, OH, USA, 2017.
59. Rutter, B.L.; Lewinger, W.A.; Blumel, M.; Buschges, A.; Quinn, R.D. Simple Muscle Models Regularize Motion in a Robotic Leg with Neurally-Based Step Generation. In Proceedings of the 2007 IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14 April 2007; pp. 630–635.
60. Szczecinski, N.S.; Getsy, A.P.; Bosse, J.W.; Martin, J.P.; Ritzmann, R.E.; Quinn, R.D. MantisBot Uses Minimal Descending Commands to Pursue Prey as Observed in Tenodera Sinensis. In *Biomimetic and Biohybrid Systems*; Springer International Publishing: Cham, Switzerland, 2016; pp. 329–340.
61. Bender, J.A.; Simpson, E.M.; Tietz, B.R.; Daltorio, K.A.; Quinn, R.D.; Ritzmann, R.E. Kinematic and behavioral evidence for a distinction between trotting and ambling gaits in the cockroach Blaberus discoidalis. *J. Exp. Biol.* **2011**, *214*, 2057–2064.
62. Golowasch, J.; Goldman, M.S.; Abbott, L.F.; Marder, E. Failure of Averaging in the Construction of a Conductance-Based Neuron Model. *J. Neurophysiol.* **2002**, *87*, 1129–1131.
63. Cruse, H. Which Parameters Control the Leg Movement of a Walking Insect? I. Velocity Control during the Stance Phase. *J. Exp. Biol.* **1985**, *116*, 343–355.
64. Gruhn, M.; von Uckermann, G.; Westmark, S.; Wosnitza, A.; Buschges, A.; Borgmann, A. Control of Stepping Velocity in the Stick Insect Carausius morosus. *J. Neurophysiol.* **2009**, *102*, 1180–1192.
65. Szczecinski, N.S.; Quinn, R.D. MantisBot Changes Stepping Speed by Entraining CPGs to Positive Velocity Feedback. In *Biomimetic and Biohybrid Systems*; Springer: Cham, Switzerland, 2017; pp. 440–452.