Implementing Neural Models in a Virtual Bug: An Exploration in Neuro-Robotics

Ian Jackson
Duke University, Dept. of Biomedical Engineering

Introduction

Unmasking the neuron

Understanding and modeling the biophysics of the neuron — and, subsequently, the behavior that its activity generates — has been of great interest to biologists, chemists, and engineers alike for centuries. As far back as the late 18th century, scientists like Luigi Galvani understood that muscles move in response to electrical impulses. [5] It was, however, not until 20th century developments in recording technology and materials science that scientists were able to precisely quantify the activity of individual neurons, adding an unprecedented depth to our understanding of the nervous system. Most notably, the works of Hodgkin and Huxley in the 1950s established the building blocks of modeling neural dynamics that are used today. [1] With the power of computing, we are now able to model the activity of not only individual neurons with great precision, but also networks of neurons. This, in turn, allows us to simulate the behavior that neural networks give rise to.

Biologically-inspired machines

The study of modeling this behavior, both internal and external, in machines or simulation environments is known as neuro-robotics. One can attempt to solve this problem using a top-down approach, which focuses on the directly recreating higher functioning from the level of logic structures and linguistic models, or a bottom-up approach, which starts from modeling networks of neurons to generate such behavior. This study explores only the bottom-up approach to neuro-robotics. In the past, researchers have used the bottom-up approach to simulate processes like visuo-spatial attention and motor planning. [10]

While using models of neural networks to recreate the complex internal processes that are typical in humans are at the forefront of this field, such applications are beyond the scope of this study.

We have instead turned to insects to be the model organism for this introduction of neural modeling, due to their very simple nervous system in comparison to humans and other mammals.

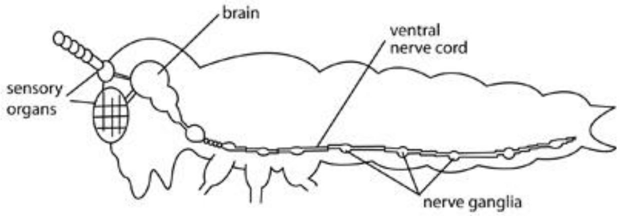

Figure 1. Simplified model of a generalized insect nervous system [3]

The sensorimotor outlined network shown in Figure 1 above allows the bug to perform complex behaviors in response to exogenous stimuli. This system can be simplified even further by assuming that the sensory organs connect directly to the motor neurons to which the muscles are connected. Without this simplification, even state-of-the-art CPUs struggle to model the complexity of the entire insect nervous system!

Using such a circuit, one can implement the same network in a robot or simulation environment, giving rise to biologically-inspired machines. In the 1960s, Dr. Valentino Braitenberg used such an implementation in his conception of Braitenberg vehicles, which utilize a combination of photosensors connected to motors to generate movement in response to light. [6]

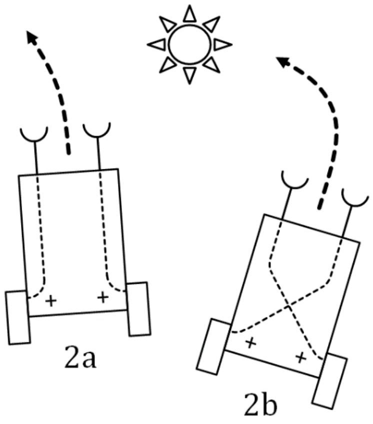

Figure 2. An example of Braitenberg vehicles responding to a light stimulus.
2a is known as the Coward, while 2b is known as the Aggressor.

The circuits used by these vehicles are modeled after the excitatory and inhibitory connections in the nervous system and give rise to a wide range of behaviors depending on the nature and number of sensor-motor connections. Braitenberg vehicles are explored in more detail in subsequent sections of this paper.

The present study is both a presentation of the fundamentals of neural modeling as well as an exploration of the applications of sensorimotor network models that simulate biologically-inspired behavior. All applications explored in this study are implemented in the Python package, Brian 2, and can be found at GitHub.com/orbitalhybridization/BasicNeuralModeling. [4]

Methods & Background

Models of neural activity can be categorized by both their biophysical accuracy and their computational efficiency/tractability. The modeling approach used for a specific application typically depends on the context in which it is being used. For example, a model that seeks to generate a large network of neurons in as little time as possible might forgo biophysical accuracy for efficiency. The figure below shows several models and their places on the spectrum of biophysical accuracy and efficiency. [17]

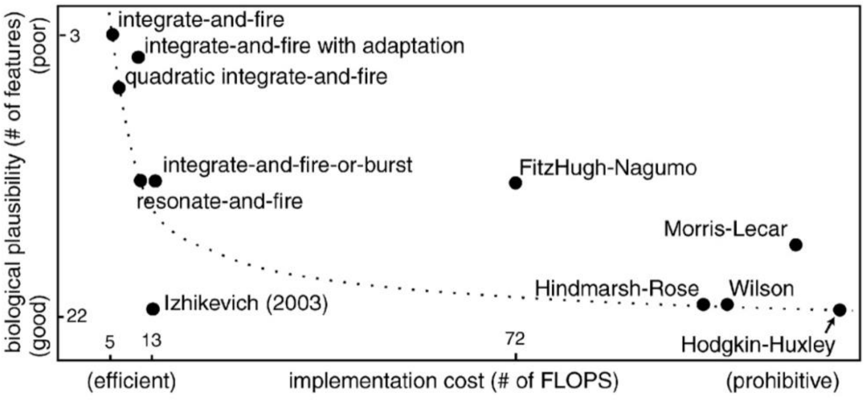

Figure 3. "The Simplicity-Realism Tradeoff"

The first aim of this study is to survey models that cover a range of both biological plausibility and the cost of implementation. The following sections detail the background, implementation, and motivation of each model used in this study.

Firing Models

Pyramidal Cell Model

The pyramidal cell is one of the most common neurons in the brains of mammals, comprising roughly two-thirds of the cerebral cortex. [11] In general, these cells are excitatory in nature (for a definition of excitatory, see the next section on synapses).

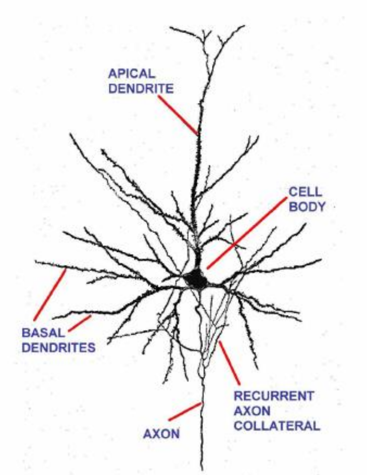

Figure 4. Typical morphology of a pyramidal cell. [12]

In a previous exploration, we implemented the pyramidal cell model based on that proposed in Lowet et al. (2015). [7] In their study, Lowet et al. investigated the gamma-frequency phase-locking properties of an excitatory-inhibitory network model of pyramidal cells known as E-cells (regular-spiking excitatory cells) and I-cells (fast-spiking inhibitory cells). Using the original paper's equation for the biophysical dynamics of an E-cell, we can replicate the spiking activity of the proposed cell:

$$C_m \frac{dV}{dt} = -g_{Leak}(V - E_{Leak}) - I_{Na} - I_K \quad - I_M$$

Where E_{Na} = 50mV, E_K = -90mV, E_{Leak} = -70mV, g_{leak} = 0.0205msiemens/cm^2, and I_{Na} and I_K follow the same formulation as their Hodgkin-Huxley equivalents. Furthermore, the added term, I_M (the non-inactivating potassium current), is calculated with the following equations:

$$I_M = \bar{g}_M \, p \ \ (V - E_k)$$

$$\frac{dp}{dt} = (p_\infty(V) - p)/\tau_{p(V)}$$

$$p_\infty(V) = \frac{1}{1 + \exp\left[-\frac{V+35}{10}\right]}$$

$$\tau_p(V) = \frac{\tau_{max}}{3.3\exp\left[\frac{V+35}{20}\right] + \exp\left[-\frac{V+35}{20}\right]}$$

This initial example of a biophysical model is meant to show that, with the power of several differential equations, this model of the pyramidal cell generates research-grade simulations of actual cells, but at the cost of computability.

Izhikevich Model

The Izhikevich model was proposed by Eugene Izhikevich in his 2004 paper "Simple Model of Spiking Neurons." [2] Unlike the pyramidal cell model, the Izhikevich model is less biophysically accurate but makes up for this with its efficiency and flexibility. These strengths come from its simple composition, utilizing just two simple differential equations (as opposed to the four used in the Hodgkin-Huxley equation) modulated by four parameters: a, b, c, and d.

$$\boxed{\begin{aligned} &v' = 0.04v^2 + 5v + 140 - u + I \\ &u' = a(bv - u) \end{aligned}}$$

Izhikevich equation, where $v = c$ and $u = u+d$ if v reaches a threshold voltage.

With the equation for v' defining activation and the equation for u' defining decay, this formulation results in the following curves:

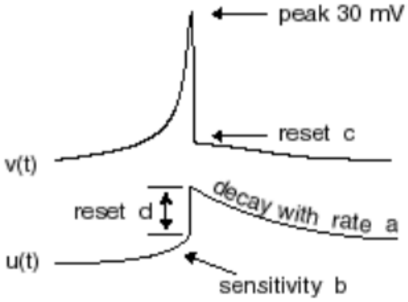

Figure 5. Activation and decay curves for the Izhikevich model. Figure from Izhikevich, 2004.

For the purposes of this study, the Izhikevich model was used to generate several different patterns of spiking activity: chattering (CH), fast spiking (FS), regular spiking (RS), and intrinsically bursting (IB). The table below shows the parameters used to tune the model to each of the generated spiking patterns. With the generated activity, both membrane potential and firing pattern (as a function of input current) were observed.

	a	b	c	d
Chattering (CH)	0.05	0.2	-50	2
Fast Spiking (FS)	0.1	0.2	-66	2
Regular Spiking (RS)	0.02	0.2	-66	8
Intrinsically Bursting (IB)	0.02	0.2	-65	4

The actual implementation involved defining a single neuron with the chosen parameters in Brian2 and monitoring its activity as a constant suprathreshold current of 8A was applied for 200ms. Finally, a 2ms refractory period was included as well.

Linear Integrate-and-Fire Model

The final model of neuronal spiking explored in this study is the Linear Integrate-and-Fire model (LIAF). Like other members of the IAF family, the LIAF model also boasts a high efficiency at the cost of a lower biophysical plausibility. The concept for this model was introduced in Lapicque (1907), before Hodgkin and Huxley's fundamental experiments in electrophysiology, and generally captures the most basic functioning of a cortical neuron. [13] As the name implies, the two main properties of any IAF model are to 1) integrate input from one or more sources, and

2) fire once a threshold is reached. In this model, the voltage returns to rest (or some preset value below rest), instantaneously, upon reaching the threshold.

Below is the general equation for an IAF model, implemented with the Hodgkin-Huxley form. In the LIAF case, F(V) = 0.

$$\tau_m \frac{dV}{dt} = (E_L - V) + F(V) + R_m I_{stim}$$

with

$$F(V) = \Delta_E \exp\left(\frac{(V - V_T)}{\Delta_E} \right)$$

Where E_L = -65mV, τ_m = 10ms, R_m = 10MOhm, Δ_E = 5.0mV, and V_T = -55mV.

In a previous study, the firing activity and firing rate of the LIAF model were observed. To generate these patterns, a group of 100 neurons was defined, from which each neuron was given a constant amount of current ranging from 0.0nA to 10.0nA. Our observations of membrane potential singled out the activity of a neuron well above the firing threshold, which happened to be the 75th neuron. The input current was applied for 2s, then the simulation was stopped. The activity and firing rate of this model was observed both with and without a 2ms refractory period. A threshold of -40mV (E_{Rest} = -65mV) was used throughout the experiment.

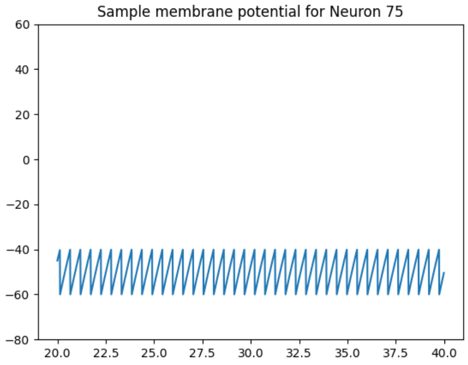

Figure 6. Sawtooth-shaped activity characteristic of the integrate-and-fire model.

The above graph shows the activity of the LIAF model previously implemented. Though this model does not have the same flexibility as the Izhikevich model, it is very simple to implement and maintains a low computational complexity.

Synaptic Models

Beyond the action potentials that play a key role in the functionality of neurons, the dynamics at the synapse – the space *between* neurons are the slower, yet equally important drivers of neural activity.

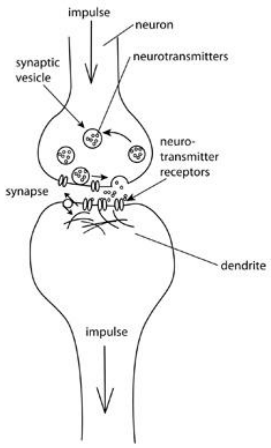

Figure 7. Diagram of a synapse. [3]

A synapse is a unidirectional electrochemical interface between a neuron that sends a signal (the presynaptic neuron) to a neuron that receives the signal (the postsynaptic neuron). There are two main types of synapses: excitatory and inhibitory. Excitatory synapses increase the membrane potential of the postsynaptic neuron (known as an excitatory postsynaptic potential, or EPSP), making it more likely to fire, while inhibitory synapses decrease that potential (creating an inhibitory postsynaptic potential, or IPSP), making it less likely to fire. Networks of these connections in various combinations between neurons are what give rise to the complex behavior of the nervous system.

Synapses are also defined by the types of neurotransmitters that bind to them, and the nature of their biophysical membrane dynamics. For example, an AMPA synapse is different from an NMDA synapse, although the two are both excitatory and receive glutamate as a neurotransmitter. The reason for this is that the time constants of activation and decay of the AMPA synapse are much faster than that of the NMDA synapse. Furthermore, the activation of

NMDA channels also relies on the concentration of magnesium at the membrane. The general formula for the neurotransmitter release from the presynaptic neuron, a function of the presynaptic spike is:

$$T = 0.25 * \tanh\left(\frac{(t - t_{spike})}{0.005}\right) - \tanh\left(\frac{(t - t_{spike} + transwidth)}{0.005}\right)$$

Where transwidth denotes the duration of the pulse.

On the other hand, the general form of the postsynaptic current is as follows:

$$I_{syn}^{post} = g_{syn} s(t)(V_{post} - E_{syn})$$

Where s(t) is a function of time constants α and β, representing the fraction of open receptors, and g_{syn} is the maximal conductance of the synapse.

Like firing models, synaptic models can vary in their place in the efficiency vs. biophysical accuracy trade-off. In this study, models on both ends of the spectrum are utilized. Of course, the more ideal model to use depends on the application for which it is being implemented.

AMPA & GABAa

For this study, we chose two biophysical models of synapses to observe. The first of the two, AMPA, is a fast-acting excitatory synapse. Being an excitatory synapse, its reversal potential is well above a typical resting potential – around 0mV. For the AMPA synapse, $\alpha = 1.1 mM^{-1} ms^{-1}$ and $\beta = 0.19 ms^{-1}$. [14]

The second model, a model of GABAa, simulates an inhibitory synapse. Because of its inhibitory nature, the reversal potential of the GABAa synapse is lower than rest – roughly -80mV. With $\alpha = 5 mM^{-1} ms^{-1}$ and $\beta = 0.18 ms^{-1}$, this synapse is fast-acting, at a similar scale to AMPA. [14]

Both models were run using a two-neuron group, where Neuron A was presynaptic to Neuron B. First a 0nA current was inputted for 5ms, followed by an 8nA current input into Neuron A for 0.5ms, causing a spike, then another 0nA current for 5ms. The metrics observed for this experiment were the post- and presynaptic membrane potentials, and the postsynaptic conductance.

Alpha & Modified Alpha

Now we turn to the non-biophysical models. Being on the other side of the accuracy-efficiency tradeoff, the original alpha model of postsynaptic conductance change utilizes just two differential equations. In response to a single presynaptic spike, the change in conductance is represented as:

$$\dot{z} \equiv \frac{dz}{dt} = \frac{-z}{\tau} + \bar{g}_{syn} u(t)$$

$$\dot{g} = \frac{-g}{\tau} + z(t)$$

$$\bar{g}_{syn} = g_{peak} \Big/ \left(\tau \exp(-1)\right)$$

Where τ is the time constant for the increase and decrease in conductance, g is the post-synaptic conductance, g_{syn} is the maximal conductance, g_{peak} is where the conductance peaks. The alpha function, however, is limited by its use of a single time constant for rise and decay. Therefore, we made use of a modified alpha function, which uses two time constants (one for rise and one for decay), to generate a more flexible model of postsynaptic conductance. This the formulation below is discussed in a chapter on synaptic modeling written by Roth & van Rossum. [14, 16]

$$\dot{z} \equiv \frac{dz}{dt} = \frac{-z}{\tau_{rise}} + \bar{g}_{syn2} u(t)$$

$$\dot{g} = \frac{-g}{\tau_{decay}} + z(t)$$

where

$$\bar{g}_{syn2} = g_{peak} \Big/ \left(\left(\frac{\tau_{decay} \tau_{rise}}{\tau_{decay} - \tau_{rise}} \right) \left(\exp\left(-\frac{peaktime}{\tau_{decay}} \right) - \exp\left(-\frac{peaktime}{\tau_{rise}} \right) \right) \right)$$

As the equation shows, the modification considers distinct rise and decay times. This change slightly increases the complexity of the model, which again demonstrates the tradeoff between efficiency of computation and the flexibility needed to recreate a more biophysically accurate model.

Like the AMPA and GABAa models, the metrics observed for this experiment were the post- and presynaptic membrane potentials, and the postsynaptic conductance. In addition, we tuned τ_{rise} and τ_{decay} to recreate the conductance curves observed from the biophysical models.

	τ_{rise} (ms)	τ_{decay} (ms)
AMPA	0.09	5.0
GABAa	0.09	5.0

The formulation of the modified alpha model can be added to the membrane potential equations of a model to incorporate the effect of synaptic inputs. In the Izhikevich equations, one can simply add the postsynaptic current equation to the Izhikevich equation to include the effects of an inhibitory or excitatory synapse. Additionally, the same can be done for an LIAF model equation, which is equivalent to the Hodgkin-Huxley equations without the sodium and potassium currents. To demonstrate this combination, several custom network models were made using the Izhikevich firing model and the alpha synapse model, each with by the amount and type of connections. The parameters used to run these models, as well as their outcome, are described in the Results section.

Bug Applications

Combining the models of firing activity and synaptic conductance in Brian2 opens the door to creating simulated networks of neural activity that respond to input currents. In the context of this study, this combination came in the form of Braitenberg-vehicle-inspired “bugs.”

To achieve a simplified structure of the bug nervous system, we simulated sensor neurons synapsed with motor neurons, which would be the main driver of the bug’s movement. The bug's environment was a simple blank map generated in the Python library, Matplotlib. [15] Within the environment, a "food" object (represented by a blue asterisk) is placed randomly at the beginning of a simulation session. Depending on the type of bug, the food would move to a random location when it was consumed (The Aggressor) or do nothing at all (The Lover). All bugs were implemented with Izhikevich neurons connected by alpha-type synaptic models.

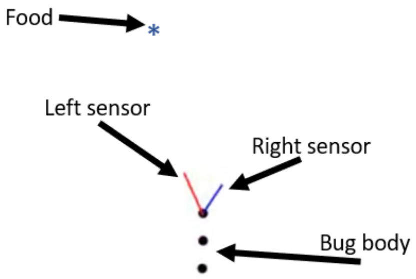

Figure 8. General outline of bug implementation.

The Aggressor

The Braitenberg vehicle characterized by its propensity to run directly to the source of stimulation is known as the Aggressor. As Figure 2 in the Introduction section shows, the cross-connection between the sensors of the vehicle and the motor is part of what causes this behavior. To emulate this, we recreated the same connections, making the right sensor presynaptic to the left motor, and the left sensor presynaptic to the right motor. Furthermore, these synapses are designed as excitatory, in order to increase motor activity with more stimulation.

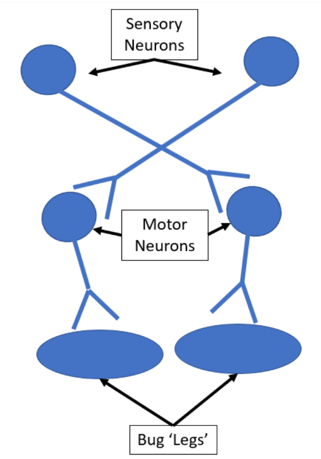

Figure 9. Circuit Diagram of the Aggressor.

I0 (Input Current Factor)	1700
Tau_ampa (conductance time constant)	0.5ms
Taum (“Leg” movement time constant)	4.5ms
Base_speed	1
Turn_rate	33Hz
Alpha (speed modifier)	0.1
w (Post-synaptic motor movement modifier)	10
g_{synpk} (maximum conductance)	5

Key parameters used to for the the Aggressor.

The Lover

The Lover, on the other hand, is first designed by connecting the right sensor to the right motor, and the left sensor to the left motor. Also, unlike the Aggressor, the synapses between the Lover’s neurons are inhibitory. The combination of same-side connections and inhibitory synapses gives this bug a movement that slows down as it approaches the stimulus, after which it finally stops, as if to admire it. Hence the term “the Lover.”

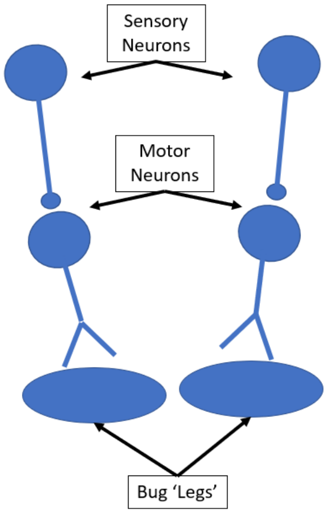

Figure 10. Circuit Diagram of the Lover.

I0 (Input Current Factor)	1250
Tau_ampa (conductance time constant)	0.5ms
Taum (“Leg” movement time constant)	4.5ms
Base_speed	1
Turn_rate	10Hz
Alpha (speed modifier)	0.01
w (Post-synaptic motor movement modifier)	10
g_{synpk} (maximum conductance)	5

Key parameters used to for the The Lover.

Custom Application: The Very Hungry Bug

The custom application implemented in this study utilized the same design as the Aggressor, but with a twist to give it more of a life-like feeling. In “The Very Hungry Bug,” the bug is given a hunger meter, which is displayed next to it on the map. Once the bug eats a predefined maximum number of food objects, it enters a “sleep” mode, during which it is unable to move. After a short regeneration period, the hunger meter resets, and the Very Hungry Bug returns to eating.

I0 (Input Current Factor)	1300
Tau_ampa (conductance time constant)	0.5ms
Taum (“Leg” movement time constant)	4.5ms
Base_speed	1
Turn_rate	27Hz
Alpha (speed modifier)	0.1
w (Post-synaptic motor movement modifier)	10
g_{synpk} (maximum conductance)	5

Key parameters used to for the Very Hungry Bug. More details can be found on the linked Github page.

This addition simply adds a dynamic element to the otherwise infinitely uniform process of chasing and eating food objects. The purpose of this addition was also to show the realistic effects that sustenance intake has on a biological organism. Animals can't eat forever, and usually enter a low-energy state after eating too much.

Results

Izhikevich Model

Using the differential equations for activation and recovery, the implementation of the Izhikevich model was successfully able to replicate the findings of the original paper in which it was proposed.

The following figures show the resulting spiking activity obtained from each parameter set:

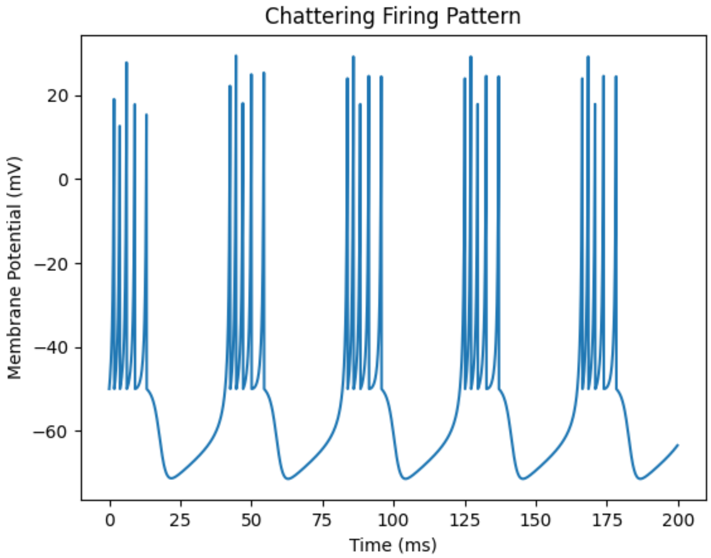

Figure 11. Chattering firing pattern

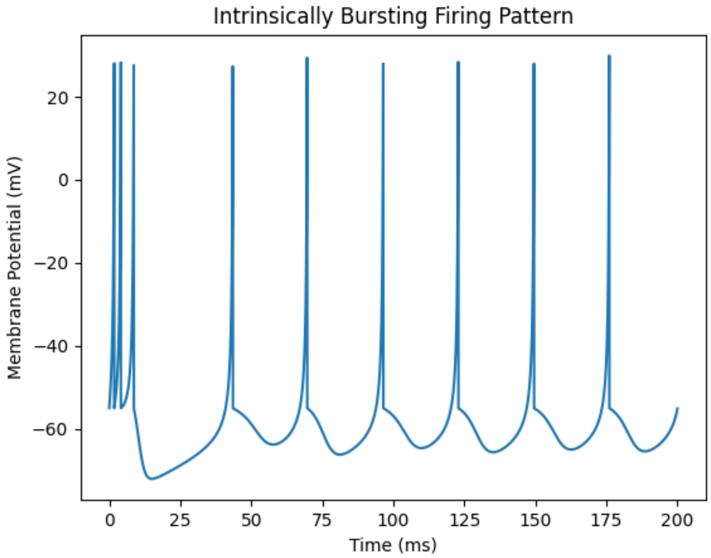

Figure 12. Intrinsically bursting firing pattern

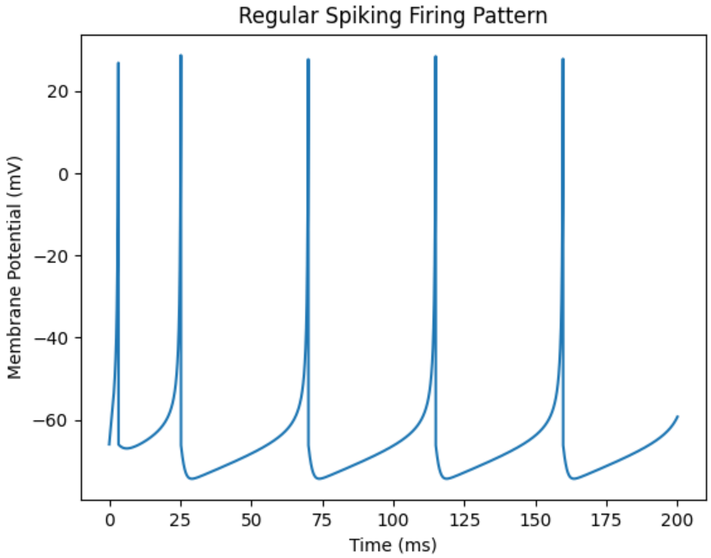

Figure 13. Regular spiking firing pattern

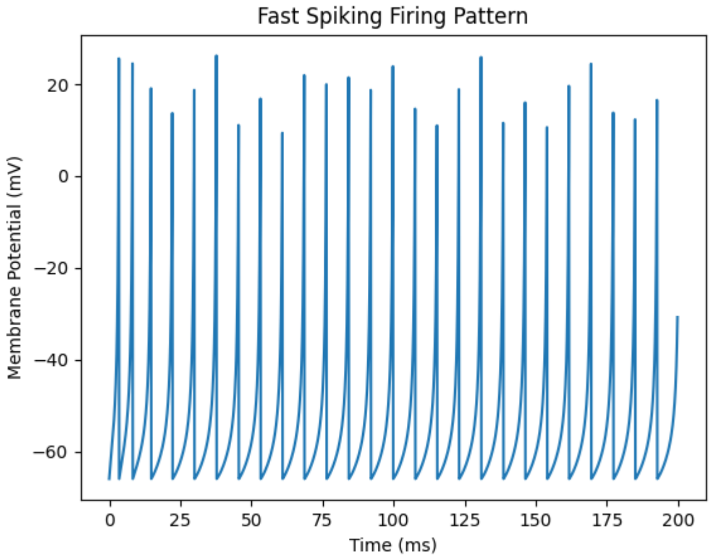

Figure 14. Fast spiking firing pattern

Interestingly, the chattering firing pattern was the only pattern to require a change from the original input parameters used to generate it. Namely, the ‘a’ parameter was changed from 0.02 to 0.05, and the input current had to be decreased from 10 units to 8 (though input current is typically in nA, the model implemented was unitless). This could be because of slight discrepancies between the method of implementation for this paper (Brian2) and the original paper. The other firing patterns were generated using the same values from the original paper. The activity depicted in each plot does not quite reach the maximum voltage of 30mV due to the timestep not being small enough to capture the extremely rapid rise.

The figures above demonstrate one of the primary benefits of the Izhikevich firing model: it has an incredible flexibility. Not only that, but its use of only two differential equations makes it a computationally efficient model as well. While this model has its strengths, however, it is limited in its biophysical accuracy. Because the model cannot take into account real neuronal properties like the voltage-gated channels for sodium and potassium, it will fall short in its ability to truly replicate biological activity. In the case of exact measurements of neural activity, this model is not ideal. As always, the best model for any given implementation depends on the demands that need to be met.

Modified Alpha Synapse

After implementing the modified alpha model from Roth & van Rossum (2009), experimentation with different time constant combinations led to generating activity that nearly mirrored each individual synapse type.

Figures 15 and 16 depict the activity for each of the modified alpha parameter combinations. Visually, they are nearly indistinguishable from the biophysical models shown in Figures 17 and 18. Looking at the general shape of the plots for the AMPA synapse, the EPSP and increase in postsynaptic conductance in response to the presynaptic spike is clear. Similarly, the IPSP and subsequent decrease in post-synaptic conductance upon the presynaptic spike are also visible for the GABAa synapse.

The point at which the conductance reaches its peak is calculated analytically with the *peaktime* equation from Roth & van Rossum (2009) and superimposed as a dotted line on each conductance plot. [16] In each plot, the conductance does reach its peak at or very near to the dotted line, which further solidifies the validity of the implemented model. The striking similarity between the modified alpha model and the biophysical models calls to the flexibility that the modified model adds simply by incorporating different speeds for rise and decay. Combining the firing pattern flexibility of the Izhikevich model and the conductance rise/decay flexibility of the modified alpha model appears to be a useful method of modeling neuronal networks for both computational efficiency and versatility, at the expense of exact biophysical accuracy.

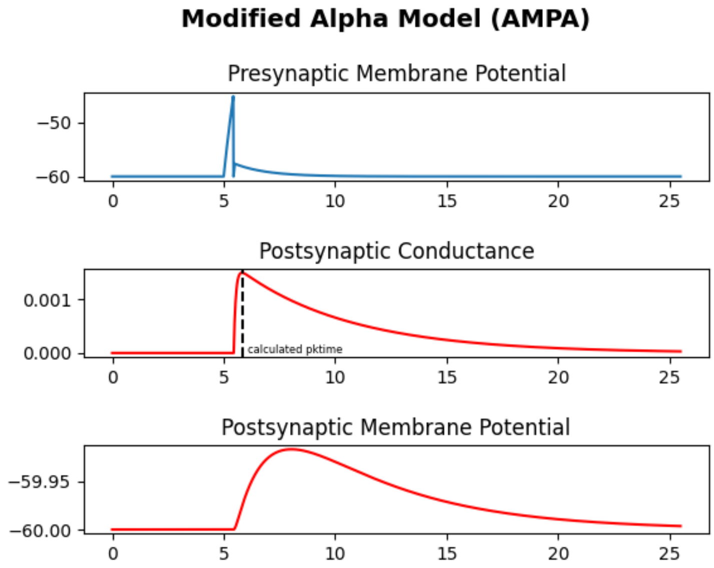

Figure 15. Recreating AMPA conductance model with the modified alpha model

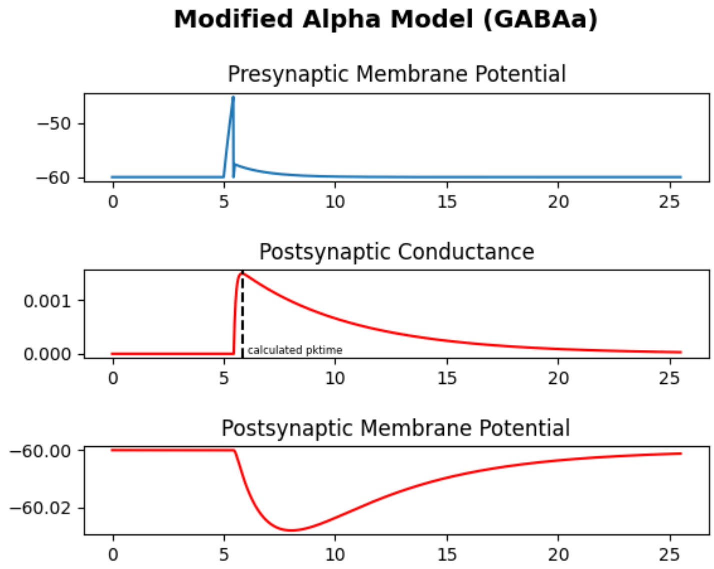

Figure 16. Recreating GABAa conductance model with the modified alpha model

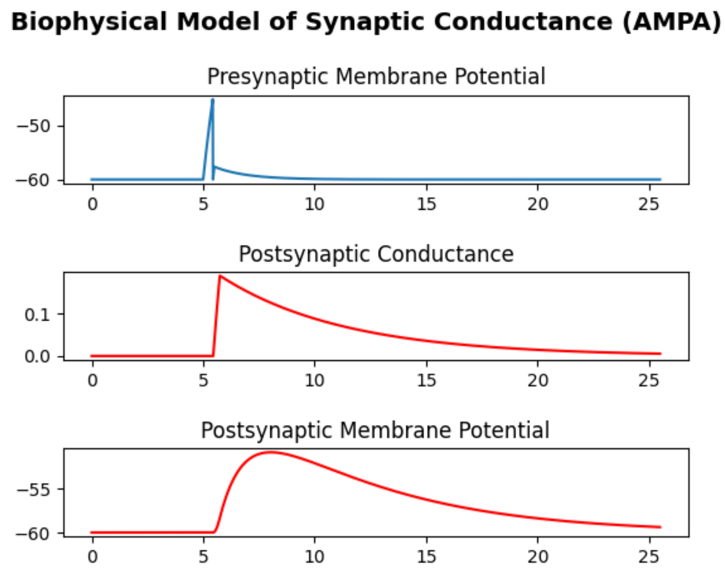

Figure 17. Modeling synaptic conductance of AMPA synapse

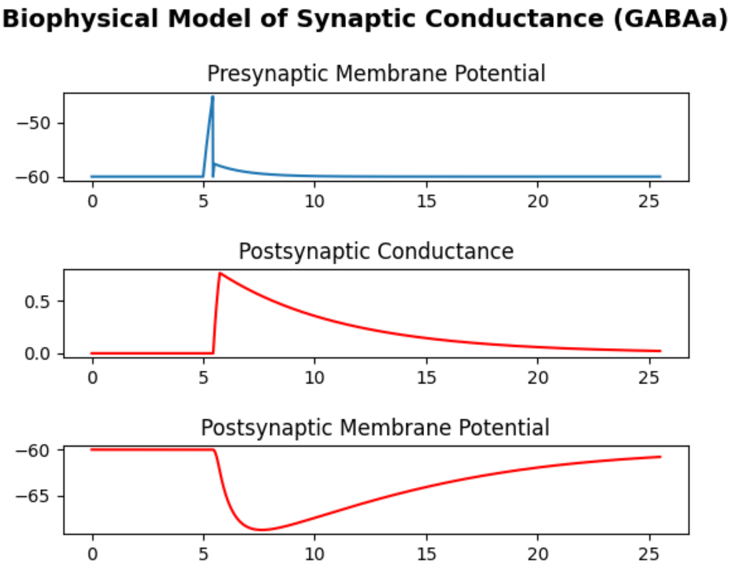

Figure 18. Modeling synaptic conductance of GABAa synapse

Custom Networks: Design 1

The first custom-designed network contained four neurons connected by a mix of inhibitory and excitatory synapses. Figure 19 below outlines the design of the network:

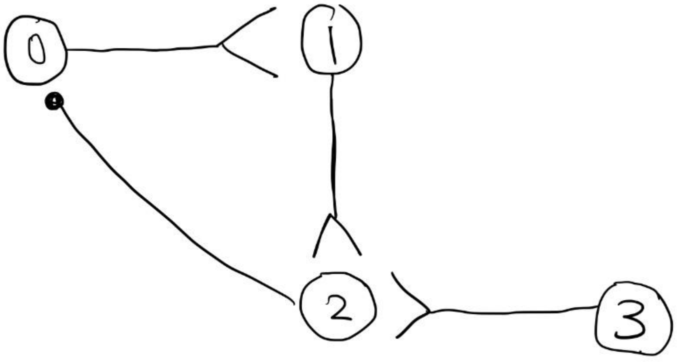

Figure 19. Custom Design 1

The activity of Custom Design 1 was tested using an initial zero-input for 10ms to establish a steady-state, after which a suprathreshold current I_{sup} was input into Neuron 0 for 20ms. Next, I_{sup} was then input into Neuron 0 and Neuron 2 for another 20ms. Finally, the input current was dropped to zero for another 10ms.

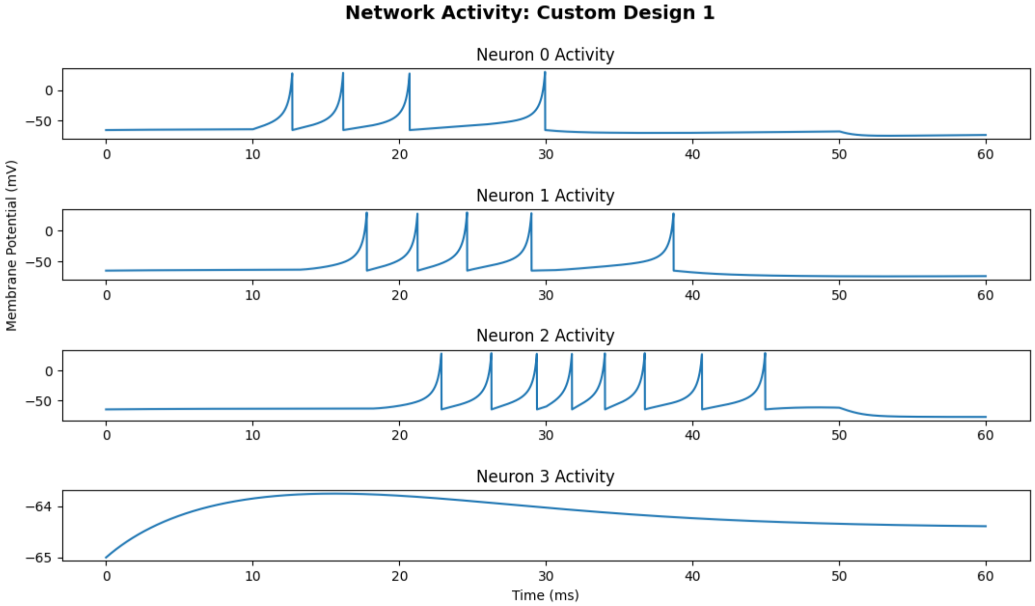

Figure 20. Network Activity of Custom Design 1

The results above show, firstly, the initial spiking of Neuron 0 after I_{sup} was applied at the 10ms mark, which is followed by spiking activity in Neuron 1 due to their excitatory connection. This also results in the spiking of Neuron 2 due to the same excitatory synapse. Neuron 0's firing rate decreases suddenly around the 23ms point, which is when Neuron 2 begins to fire. This makes sense due to Neuron 2 having an inhibitory connection with Neuron 0. Things get interesting around the 30ms mark, where Neuron 2's firing rate increases due to the input, and both Neuron 0 and Neuron 1 drop down to their rest potential (Neuron 0 dipping a little bit below it). This exemplifies the power of inhibitory connections in a network of neurons. Throughout the experiment, Neuron 3 never reached its firing threshold, remaining at rest due to the fact that it received no input.

Custom Networks: Design 2

The next design, like the others, utilizes a mix of excitatory and inhibitory synapses. This time, however, the design includes a single neuron that has both excitatory and inhibitory connections to other neurons. The focus of this experiment is the effect that this conflicting relationship has on a fourth neuron: Neuron 3 (in Figure 21 below).

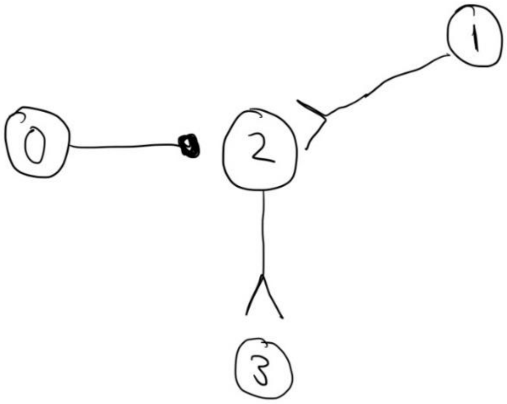

Figure 21. Custom Design 2

The design was tested using a baseline zero-input for 10ms, followed by a 20ms suprathreshold input current I_{sup} to Neuron 1. Next I_{sup} was applied to both Neuron 1 and Neuron 0 for 20ms. Finally, another baseline zero-input current was applied for 10ms.

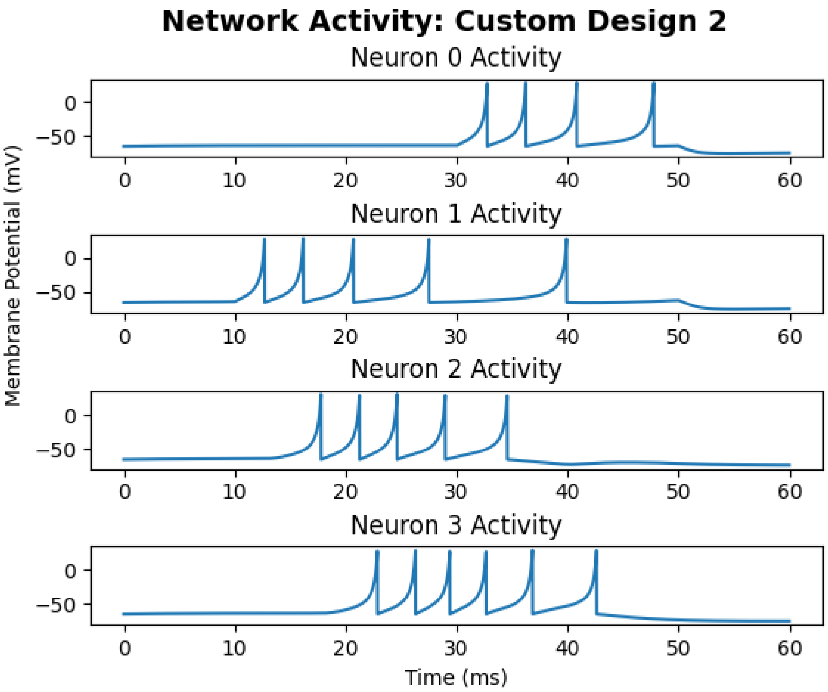

Figure 22. Network Activity of Custom Design 1

Breaking down Figure 22, it is clear that the initial spiking of Neuron 1 successfully excited Neuron 2, causing it to spike slightly afterwards. This, in turn, caused Neuron 3 to spike after a delay as well. At the 30ms mark, Neuron 0 begins to fire, which causes Neuron 2 to drop back to baseline shortly after. Not long after, Neuron 3 ceases firing due to the lack of input from Neuron 2. Interestingly, the firing rate of Neuron 1 also decreases, which is unexpected as it receives the same amount of current throughout stimulation and has no synaptic connections other than its connection to Neuron 2. This could be due to the 2ms refractory period stifling the firing rate, or an unexpected factor of the neurodynamics like the Izhikevich recovery variables.

Custom Networks: Design 3

The third and final design in this set considers a network with more inhibitory connections. More specifically, this design observes inhibitory synapses in series: the inhibition of an inhibitory connection. The main player here is the effect that this circuit has on Neuron 1 below.

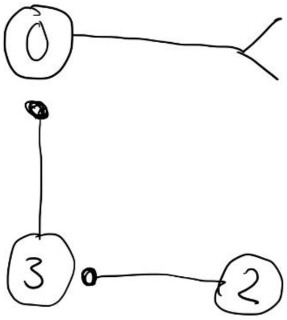

Figure 23. Custom Design 3

This network was tested using the same 10ms zero-input baseline current, followed by the same I_{sup} current applied to Neuron 0 for 20ms. Next, I_{sup} was applied to Neuron 0 and Neuron 3 for 20ms. Lastly, I_{sup} was applied to Neuron 0, Neuron 2, and Neuron 3 for 20ms, and a 10ms zero-input was included at the end of the experiment.

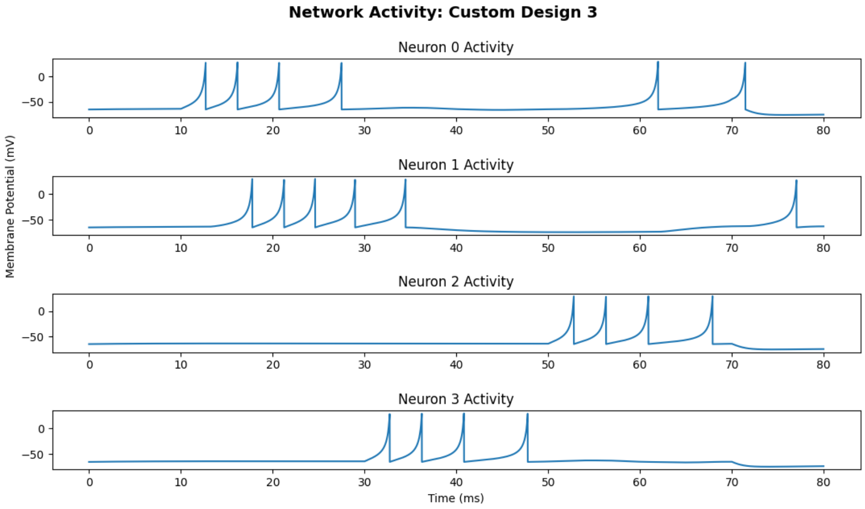

Figure 24. Network Activity of Custom Design 3

The story told in Figure 24 starts with the firing of Neuron 0, causing excitation and subsequent firing of Neuron 1. When Neuron 3 is turned on at the 30ms mark, this causes an immediate cease in firing of Neuron 0, which then has the same effect on Neuron 1. Now, what happens when that inhibition is inhibited? As expected, at the 50ms mark when Neuron 2 is hit with an input current, the effect of Neuron 3 is diminished as it returns to rest potential. This allows Neurons 0 and 1 to return to firing, which can be observed in the latter end of the top two subplots in the figure.

The Aggressor

Using the Izhikevich spiking model and alpha synapses to connect sensory neurons to motor neurons to motors on the bug's body, the creation of The Aggressor was somewhat of a success. The activity of the virtual creature can be seen as a YouTube video here: https://www.youtube.com/watch?v=yhx7gaEIUZ0.

Looking at the montage, the expected activity – a brazen aggression toward the stimulus – can be observed. For the most part, the bug appears to wiggle its way toward the food stimulus. However, the bug does exhibit some unexpected behavior. Most strikingly, it sometimes doesn't notice the stimulus when too far away or placed too far to the side. This could be caused by a mix of a base stimulus response factor being too low or a turn rate that is too low. The latter of these is unlikely, as increasing the turn rate created a bug that turned too quickly to accurately acquire food.

The Lover

A short video of the Lover design in action can be found at this YouTube link: https://youtu.be/at3o7KmvqlM.

Like the Aggressor, this implementation was a partial success. While the bug appears to approach the stimulus, and slow down while it does so, it does not fully stop as expected. Instead, the bug continues moving in the direction of the original stimulus, passing it entirely. This could be a result of a mistuning of the stimulus strength. However, turning down the modifier of the stimulus strength relative to distance from the food did not change the behavior much. Furthermore, tweaking the maximum synaptic conductance to a lower threshold also yielded little change. More investigation into the parameters governing when the bug might come to a complete stop – possibly the motor neuron modifier, taum – must be explored in future iterations of this experiment in order to build a more accurate simulation.

The Very Hungry Bug

As it utilized the same design, The Very Hungry bug shared both the successes and failures of the Aggressor. Although the bug did seem to detect and move directly toward the food sources for the most part, there were some instances when it didn’t seem to detect the food at all. On the other hand, the hunger meter and “sleep mode” modifications seemed to work quite well. After eating the predefined maximum number of food objects, the bug did cease movement entirely, entering a resting state. After its resting period ended, the bug reliably resumed pursuing the food sources on the map.

A short video of the behavior can be found here on YouTube: https://youtu.be/WNCb4-n_yew. The state of the bug (“hungry”, when pursuing food, and “sleepy” when full) can be seen at the top of the screen recoding window throughout the simulation.

Discussion

This study in neuro-robotics was done to both summarize the foundations of the field and to explore its basic implementations. One of such fundamentals is difference between biophysical and non-biophysical models, and the tradeoff that is considered when choosing which route to take. In our results from implementing the Izhikevich and biophysical firing models, we see that high flexibility and computability comes at the price of biological accuracy. The same is seen in the models of synaptic conductance. Building on top of this, we then saw how firing models and synapse models can combine to form networks of neurons that modulate one another in response to incoming connections. These building blocks culminated in the implementation of these networks into a behaving virtual creature modeled after the Braitenberg vehicles. While the bugs had variable results, the activity they exhibit truly do resemble that of a living organism reacting to its environment.

Finally, the custom bug design added an even more realistic twist to the standard Braitenberg paradigm. Much like the refractory period of a neuron, the bug’s resting state staunched how much (and how quickly) the bug can eat. These simple implementations are just the beginning, with many potential improvements that could be made. For example, the behavior of the Hungry Bug could be made even more realistic by causing a slowing-down in the bug’s movement as it eats more and more, rather than the speed dropping to zero upon reaching its hunger limit. The paradigm in general could also benefit from the inclusion of additional bugs interacting with one another, which would add a social element to the simulation. This was actually attempted, but issues with Brian2’s MagicNetwork class forced the effort into a potentially separate project. The implementation of neural activity itself could also be improved by adding factors like synaptic plasticity to the bug, allowing it to learn. Finally, as a more computational improvement, the bug implementation could make use of parallel processing to increase its efficiency. Instead of calculating the state of the bug at each timestep, multiple cores could be run on a CPU to calculate a set of states simultaneously to allow for a smoother design. Future iterations of this project may investigate the benefits of these modifications.

It is also worth noting that, while this exploration was done with an emphasis on sensorimotor networks, implementations of the neural activity behind cognitive processes like memory and decision making exist as well. Like sensorimotor networks, these networks rely on the same fundamentals discussed in this paper: firing models connected by synapse models. More advanced implementations, of course, may incorporate additional factors to models – such as modeling myelination and plasticity – to achieve a more biophysically accurate model. Due to the efficiency/biophysicality tradeoff, one of the primary limits to these advanced implementations lie within the sheer computational complexity of modeling even a fraction the billions of neurons and trillions of connections in the human central nervous system. However, with the leaps and bounds that innovations in technology afford, it is likely that we will one day be able to model in extreme detail entire networks related to cognitive processes that make us human.

References

1. A. L. Hodgkin and A. F. Huxley, A quantitative description of membrane current and its application to conduction and exciation in nerve, Journal of Physiology, 117:500-544, 195.
2. Izhikevich, E, Simple Model of Spiking Neurons, IEEE Transactions of Neural Networks, 14: 1569-1572.
3. Purdue Extension Entomology, Insect Physiology, Purdue University Department of Entomology, https://extension.entm.purdue.edu/401Book/default.php?page=insect_physiology.
4. M. Stimberg, R. Brette, D.F.M. Goodman, Brian 2, an intuitive and efficient neural simulator, eLife, 8, 2019.

5. Galvani, L, De viribus electricitatis in motu musculari commentarius, Typographia Instituti Scientiarum, 1791.
6. Braitenberg, V, Vehicles: Experiments in synthetic psychology, MIT Press, 1984.
7. Lowet, E., Roberts, M., Hadjipapas, A., Peter, A., van der Eerden, J., De Weerd, P., Input-Dependent Frequency Modulation of Cortical Gamma Oscillations Shapes Spatial Synchronization and Enables Phase Coding, PLOS Computational Biology, 11(2): e1004072.
8. Dayan, P., Abbott, LF., Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems, The MIT Press, 2001.
9. Taraban, R., Limits of Neural Computation in Humans and Machines, Sci Eng Ethics, 2020 Oct; 26(5): 2547-2553.
10. Jeong, S., Arie, H., Lee, M., Tani, J., Neuro-robotics study on integrative learning of proactive visual attention and motor behaviors, Cogn Neurodyn. 2012 Feb;6(1):43-59.
11. Spruston, N., Pyramidal neurons: dendritic structure and synaptic integration, Nat Rev Neurosci, 2008, 9: 206–221.
12. Ditlevsen, S., Samson, A., Jacobsen, M., Wainrib, G., Sacerdote, L., Giraudo, M.T., Tuckwell, H.C., Thieullen, M., Stochastic Biomathematical Models with Applications to Neuronal Modeling, Springer Lecture Notes in Mathematics, 2012.
13. Lapicque, L., Recherches quantitatives sur l’excitation électrique des nerfs traitée comme une polarization, J. Physiol. Pathol. Gen., 1907, 9: 620-635.
14. Henriquez, C., (2022), BME 503 – Computational Neuroengineering: Lecture Notes, [PowerPoint Slides].
15. Hunter, J.D., Matplotlib: A 2D Graphics Environment, Computing in Science & Engineering, 2007, 9(3): 90-95.
16. De Schutter, E., Computational Modeling Methods for Neuroscientists, 2013, MIT Press Scholarship Online.
17. Von Kugelgen, J., On Artificial Spiking Neural Networks: Principles, Limitations and Potential, [Working Paper], 2017.