Accelerating Biophysical Neural Network Simulation with Region of Interest based Approximation

Yun Long, Xueyuan She, Saibal Mukhopadhyay
School of Electrical and Computer Engineering, Georgia Institute of Technology, GA, USA
Email: {yunlong, xshe, saibal.mukhopadhyay}@gatech.edu

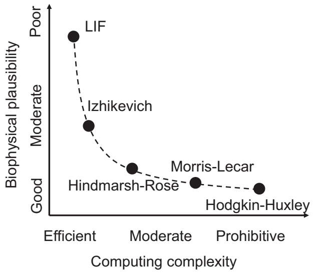
***Abstract*—Modeling the dynamics of biophysical neural network (BNN) is essential to understand brain operation and design cognitive systems. Large-scale and biophysically plausible BNN modeling requires solving multiple-terms, coupled and non-linear differential equations, making simulation computationally complex and memory intensive. This paper presents an adaptive simulation methodology in which neurons in the region of interest (ROI) follow high biological accurate models while the other neurons follow computation friendly models. To enable ROI based approximation, we propose a generic template based computing algorithm which unifies the data structure and computing flow for various neuron models. We implement the algorithms on CPU, GPU and embedded platforms, showing 11x speedup with insignificant loss of biological details in the region of interest.**

I. Introduction

Biophysical neural network (BNN) modeling provides an avenue for exploring hypotheses about how human brain works and how to realize neuronal coding [1]. Moreover, it is a critical step towards developing cognitive system [2], artificial intelligence (AI) [3], and emerging computer architecture [4], etc. However, the BNN modeling is challenging as the dynamics of neurons and synapses are regulated by complex, coupled non-linear differential equations, making it computation intensive.

For BNN simulation, there is a well-known tradeoff between the computing efficiency and the biology accuracy. Biologically accurate models always require more computation while less computing intensive models normally lack biophysical plausibility. For example, Hodgkin-Huxley model [5] presents high degree of biological credibility but is very computationally complex. Simplified mathematical models, such as leaky integrate-and-fire model (LIF model) and Izhikevich models [6], improve the computing efficiency but lack biology accuracy. Fig. 1 shows the tradeoff with several commonly used neuron models.

To accelerate BNN simulation, parallel computing frameworks such as CPU clusters and general-purpose graphics processing units (GPGPUs) are being actively explored [7, 8]. However, the performance is ultimately limited by the algorithm's ability to leverage parallel hardware and the memory bandwidth. There are efforts in developing specialized application-specific integrated circuit (ASIC) which provide significant improvements on performance and energy efficiency [4, 9]. However, the ASICs normally implement a single neuron model, which lack the flexibility to support different types of neuron dynamics. Moreover, programming the ASIC also require specialized knowledge compared to standard CPU or GPU based platforms.



Models computation complexity

Models	# of equations and parameters	# of FLOPS [6]
LIF	2/5	5
Izhikevich	4/5	13
Hindmarsh-Rose	5/7	120
Morris-Lecar	5/12	600
Hodgkin-Huxley	10/7	1200

Fig 1. The tradeoff between computing efficiency and biophysical plausibility [6].

In this work, we develop an adaptive simulation methodology that maintains high biological accuracy with more complex neuron models (e.g. Hodgkin-Huxley model) in the region of interest (ROI), while simplifying the neuron models (e.g. LIF and Izhikevich model) elsewhere to improve the overall computation speed. At the interested regions, the ROI based approximation retains all the biophysical information such as the sodium (Na^+) and potassium (K^+) ion channel conductance which is missing in LIF and Izhikevich models. In the proposed algorithm, the ROI can be statically defined as a specific region or determined dynamically during simulation based on factors such as spiking frequency. The key challenge in ROI based BNN simulation is the need to solve a system of hybrid neurons where the differential equations determining the dynamics of a neuron change over space as well as time. To address the preceding challenge, we propose a generic template based computing model, where both data and computing are stored/defined as templates. The proposed template based processing algorithm provides a uniform data structure and computing flow for various neuron models. Therefore, models can be easily switched with no programming overhead.

We implement the proposed ROI based computing algorithm on CPU, GPU and embedded system. Our baseline simulator with template based processing (without ROI approximation) provides accuracy and performance comparable to the state-of-the-art BNN simulators. The effectiveness of ROI based adaptive simulation is demonstrated through a visual cortex simulation. The experimental results measured from CPU, GPU, and embedded platforms demonstrate 11x speed-up on average

978-3-9819263-0-9/DATE18/©2018 EDAA

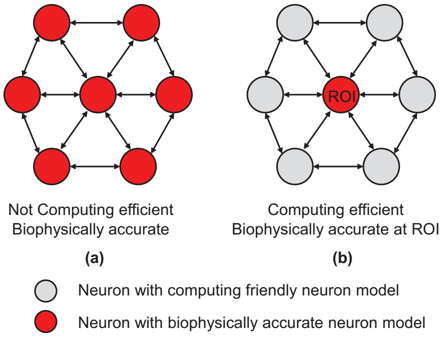

ROI
Not Computing efficient
Biophysically accurate
(a)
Computing efficient
Biophysically accurate at ROI
(b)
Neuron with computing friendly neuron model
Neuron with biophysically accurate neuron model

Fig. 2. BNN simulation with (a) homogenous neuron models (all biophysical accurate model) and (b) ROI based heterogeneous models with biophysical accurate model at concerned region and simpler model for the others.

with insignificant biological accuracy loss in the regions of interest.

II. Problem Formulation

A. The opportunities for ROI based BNN simulation

Large-scale cortical modeling is one of the most critical scientific challenges in 21st century. The cortical simulation facilitates better understanding of human brain, exploring hypotheses of neuroscience, and developing human-like artificial intelligence. The central nervous system (CNS) of human contains 10^{11} neurons and 10^{14} synapses, coupling together and regulated by complex neuronal dynamics, which is challenging for simulation even with the most powerful supercomputers [1, 4]. On the other hand, it is well known that only a small portion of CNS are responsible to functions such as vision, sound, and motion control. Modern imaging techniques such as PET (positron-emission tomography) and fMRI (functional magnetic resonance imaging) also indicate that neurons in different regions have varying activation levels. This inspires us to develop a ROI based approximation algorithm to accelerate cortical simulation where the highly-active or critical regions are modeled with biophysical accurate neuron models while the rest are modeled with simpler models.

B. Challenges in ROI based BNN Simulation models

The ROI based trade-off in accuracy and computation speed is a well-known concept in scientific computation. For example, adaptive mesh refinement (AMR) algorithm imposes finer sub-grids (denser grids) at the regions that require higher resolution to achieve better accuracy. However, unlike the conventional ROI based computations that mainly focus on the data-level approximation, the proposed algorithm for ROI based BNN simulation is to change the underlying neuron dynamics being solved at each node (i.e. model-level approximation). To be more specific, rather than emulating the detailed neuronal dynamics with the same model for all the neuron groups (Fig. 2(a)), we can simplify the simulation by using computing friendly model for the less important regions and biophysically accurate model for the critical regions (Fig. 2(b)). Therefore, *we need to simulate a system of coupled differential equations where the equations are not only different at different nodes, but also might changes over time (ROI changes over time)*. This is a unique approach for performance-accuracy trade-off in BNN simulation. A few of prior works implement heterogeneous simulation where different neuronal dynamics are modeled simultaneously [10], however, our approach is to dynamically change the model of the neurons, for example, depending on their spiking activity. Moreover, our objective is to accelerate the simulation while maintaining the biology accuracy of the interested region. In this work, we target on three neuron models: LIF model, Izhikevich model, and Hodgkin-Huxley model (Table I). We focus on the ROI approximation for neuron dynamics and assume synaptic weights are fixed. In the future work, we will implement the ROI approximation for synaptic dynamics.

Table I: Neuron models and descriptions

LIF model	Izhikevich model
$\frac{dv}{dt} = a + bv + cI$ $v = v_{reset}\ if\ v > v_{thres}$	$\frac{dv}{dt} = 0.04v^2 + 5v + 140 - u + I$ $\frac{du}{dt} = a(bv - u)$ $v = c \quad u = u + d \quad if\ v > v_{thres}$
Hodgkin-Huxley model	
$\frac{dv}{dt} = \frac{1}{C_m} \cdot [I - G_{Na}m^3h(v - E_{Na}) - G_K n^4(v - E_k) - G_l(v - E_l)]$ $\frac{dn}{dt} = \alpha_n(v)(1-n) - \beta_n(v)n \quad \alpha_n = \frac{0.01(v+50)}{1 - e^{-0.1(v+50)}} \quad \beta_h = \frac{1}{1 + e^{-0.1(v+30)}}$ $\frac{dm}{dt} = \alpha_m(v)(1-m) - \beta_m(v)m \quad \alpha_m = \frac{0.1(v+35)}{1 - e^{-0.1(v+35)}} \quad \beta_n = 0.125e^{-0.125(v+60)}$ $\frac{dh}{dt} = \alpha_h(v)(1-h) - \beta_h(v)h \quad \alpha_h = 0.07e^{-0.05\,(v+60)} \quad \beta_m = 4e^{-0.0556(v+60)}$	

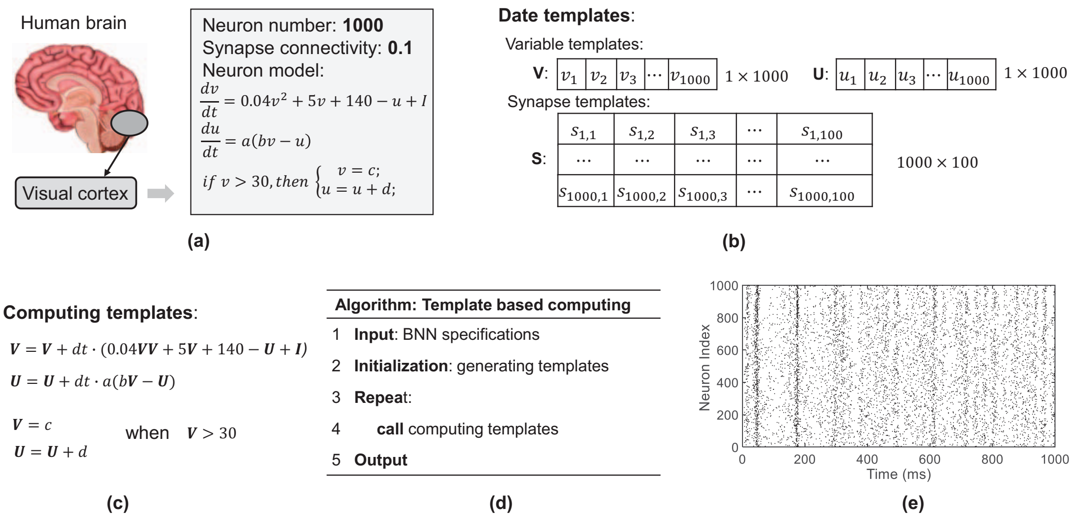
LIF model: v is the membrane potential, I is the input current, v_{thres} is the neuron firing threshold, and a, b, c are the parameters.
Izhikevich model: v represents the membrane potential of the neuron and u represents a membrane recovery variable which provides negative feedback to v. v_{thres} is the neuron firing threshold, and a, b, c, d are the parameters.
Hodgkin-Huxley model: Four differential equations in Hodgkin-Huxley model describe the membrane potential, activation of Na+ and K+ currents, and leakage current, respectively. C_m *is* the membrane capacitance, G_{Na}, G_K and G_l are the coefficients for ion channels conductance. m and n are the channels conductance regulators since the ion channels are voltage-gated. α and β are membrane potential controlled parameters.

III. Computing Algorithms

A. Template based computing

We propose a template based computing algorithm where variables (e.g. neuron membrane potential) and synaptic weights are stored in *data templates*, neuron dynamics are defined in *computing templates*. The template based processing allows us to configure and simulate a BNN where all the data and computing are represented in a unified form. Therefore, ROI based adaptive BNN simulations consisting of different neuron models can be easily realized by simply utilizing different data and computing templates.

An example is used to show how the template based processing works. As shown in Fig. 3(a), the BNN is utilized for visual cortex simulation, which contains 10^3 Izhikevich neurons and the synapse connectivity is 0.1 (i.e. each neuron has 10^2 synapses). Fig. 3(b) shows the corresponding data templates including V, U and synaptic weights. Storing data into templates provides a compact data structure and enhance


Human brain
Visual cortex
Neuron number: **1000**
Synapse connectivity: **0.1**
Neuron model:
$\frac{dv}{dt} = 0.04v^2 + 5v + 140 - u + I$
$\frac{du}{dt} = a(bv - u)$
$if\ v > 30, then \begin{cases} v = c; \\ u = u + d; \end{cases}$
(a)
Date templates:
Variable templates:
V: v_1 v_2 v_3 ⋯ v_{1000} 1×1000
U: u_1 u_2 u_3 ⋯ u_{1000} 1×1000
Synapse templates:
S: $s_{1,1}$ $s_{1,2}$ $s_{1,3}$ ⋯ $s_{1,100}$ / ⋯ / $s_{1000,1}$ $s_{1000,2}$ $s_{1000,3}$ ⋯ $s_{1000,100}$
1000×100
(b)
Computing templates:
$\boldsymbol{V} = \boldsymbol{V} + dt \cdot (0.04\boldsymbol{VV} + 5\boldsymbol{V} + 140 - \boldsymbol{U} + \boldsymbol{I})$
$\boldsymbol{U} = \boldsymbol{U} + dt \cdot a(b\boldsymbol{V} - \boldsymbol{U})$
$\boldsymbol{V} = c$
$\boldsymbol{U} = \boldsymbol{U} + d$
when $\boldsymbol{V} > 30$
(c)
(d)
Neuron Index
0
200
400
600
800
1000
Time (ms)
0
200
400
600
800
1000
(e)

```
Algorithm: Template based computing
Input: BNN specifications
Initialization: generating templates
Repeat:
    call computing templates
Output
```

Fig. 3. Template based processing. (a) BNN specifications. (b) Data templates. (c) Computing templates. (d) Algorithm for template based processing (no ROI). (e) Output results as a raster plot.

the computing efficiency for element-wise functions, especially for parallel computing platforms such as GPU [8].

The concept of computing template is similar with the computational graph in popular deep learning frameworks such as Tensorflow [11]. A computing template consists of a series of matrix operations. Each operation in a computing template takes two or more data templates (vectors and matrix) as inputs. The computing templates are pre-define based on the differential equations in the neuron models. As shown in Fig. 3(c), the computing templates are defined based on the differential equations of Izhikevich model in Fig. 3(a).

During computing, templates are first generated based on network specifications. Then the computing templates are called repeatly to perform matrix operations. The whole algorithm is shown in Fig. 3(d). Meanwhile, neuron activities such as membrane potential and spiking frequency can be recorded for post analyses. As shown in Fig. 3(e), we plot the recorded spikes (i.e. raster plot) for neurons in the network.

B. *ROI based approximation*

Benefiting from the template based computing, the neuron models can be switched easily by calling different computing templates, enabling ROI based approximation. As mentioned earlier, the ROI can be statistically defined by identifying a spatial region where higher accuracy is desired during the simulation. Alternatively, the ROI can be dynamically defined using certain criteria. For example, the information theory of a rate-based network suggests that the more a neuron fires, the more information it propagates [12]. Hence, the spiking frequency can be a criterion to keep the 'busy' neurons in biological plausible model (defined as ROI).

Fig. 4 illustrates the proposed algorithm considering 5 randomly connected neurons, and assuming the neuron models can switch between LIF and Izhikevich models (in practice, we consider model switching between the LIF and the Hodgkin-Huxley models). We first initialize data and computing templates for both models. Then we create a *regulator vector* which contains the spiking frequency for each neuron. During inference, system first accesses the regulator vector and determines which model the neuron should follow, which data templates and computing templates should be called. For example, if we define the model switching threshold to be 20 Hz, the corresponding neuron will follow Izhikevich model once the spiking frequency is higher than 20 Hz, otherwise it will follow LIF model.

Figure 5 illustrates the computation flow for ROI based adaptive simulation. First, during network initialization, data and computing templates for different neuron models are created. Second, during computation/inference, computing templates are called and neuron spiking frequency are updated with a given time window. After each iteration, program checks the regulator vector and determine which model the neuron should follow.

C. *Parameter tuning*

Even though the internal neuronal dynamics are different

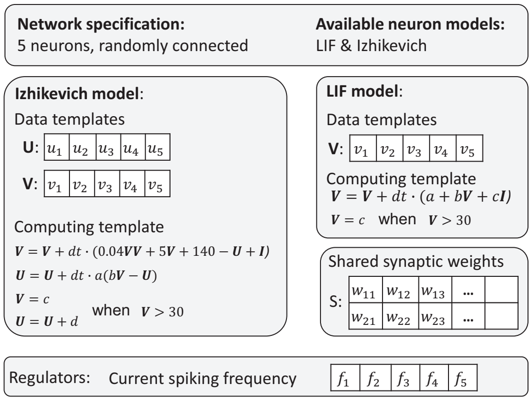

Network specification:
5 neurons, randomly connected
Available neuron models:
LIF & Izhikevich
Izhikevich model:
Data templates
U: u_1 u_2 u_3 u_4 u_5
V: v_1 v_2 v_3 v_4 v_5
Computing template
$\boldsymbol{V} = \boldsymbol{V} + dt \cdot (0.04\boldsymbol{VV} + 5\boldsymbol{V} + 140 - \boldsymbol{U} + \boldsymbol{I})$
$\boldsymbol{U} = \boldsymbol{U} + dt \cdot a(b\boldsymbol{V} - \boldsymbol{U})$
$\boldsymbol{V} = c$
$\boldsymbol{U} = \boldsymbol{U} + d$
when $\boldsymbol{V} > 30$
LIF model:
Data templates
V: v_1 v_2 v_3 v_4 v_5
Computing template
$\boldsymbol{V} = \boldsymbol{V} + dt \cdot (a + b\boldsymbol{V} + c\boldsymbol{I})$
$\boldsymbol{V} = c$ when $\boldsymbol{V} > 30$
Shared synaptic weights
S: w_{11} w_{12} w_{13} … / w_{21} w_{22} w_{23} …
Regulators: Current spiking frequency f_1 f_2 f_3 f_4 f_5

Fig. 4. A simple network with 5 neurons. Neurons can switch models dynamically between LIF and Izhikevich model according to the spiking frequency.

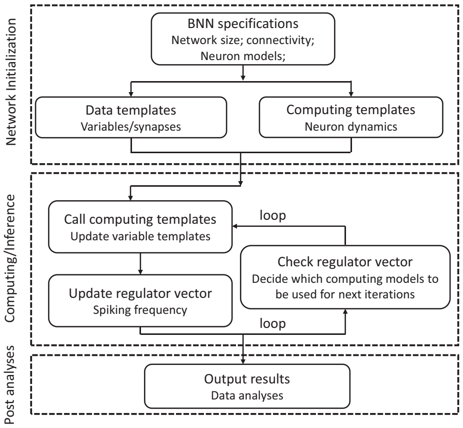


Fig. 5. Computation flow for ROI based adaptive simulation algorithm.

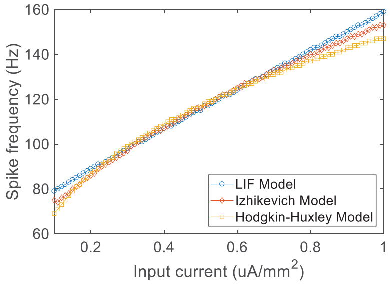
for different neuron models, ROI based approximation requires that the output (represented with spikes) of a neuron should remain the same after changing models. To acheve this goal, we carefully tune the parameters in computing friendly models (i.e. LIF model and Izhikevich model) with the Hodgkin-Huxley model as a baseline, to match the spiking frequency for different input current. Fig. 6 shows the tuned results. The spike frequencies match with each other very well in the explored region (The input current scope for Hodgkin-Huxley neuron model is 0.1 to 1, unit is *uA/mm^2*).

IV. Experimental Results

We implement the proposed BNN simulator on three platforms: CPU, GPU, and embedded system, targeting three different application environments. For CPU implementation, we use MATLAB since it is highly optimized for matrix-vector operation which is the main type of computing for BNN simulation. For GPU implementation, we utilize NVIDIA CUDA programming language. For embedded system, we use the popular Raspberry Pi, a single-board micro-computer which promotes Python as the main programming language. Table II summarizes the parameters for these platforms.

A. Performance analysis without ROI

First, we compare the accuracy of the proposed BNN simulator (without ROI) with state-of-the-art GPU based BNN simulator, CARLsim [8]. Fig. 7(a, b) shows the raster plots and spiking frequency distributions measured from CARLsim and our work. The network contains 10^3 Izhikevich neurons and 10^4 synapses, excitatory and inhibitory neurons in a 4:1 ratio [1]. Neurons receive uniformly distributed external input current and spikes from the pre-neurons. We observe good match for the spiking pattern and spiking frequency distribution. We further compare the computing speed of our simulator with existing simulators: Brian [13], Nest [10], and CARLsim [8]. The first two simulators are CPU based and CARLsim implements GPU version. To ensure the best performance, Data for these simulators are measured from the sample codes provided by the tools developer. As shown in the insert table of Fig. 7, our simulator provides the state-of-the-art performance compared with prior CPU and GPU based implementation (3.4x and 1.4x speedup than other CPU and GPU based simulators, respectively).



Tuned Parameters:

LIF model:

$a = -6.77 \quad b = -0.099$

$c = 0.315$

Izhikevich model:

$a = 5.0 \quad b = -0.273$

$c = -56.0 \quad d = 2.0$

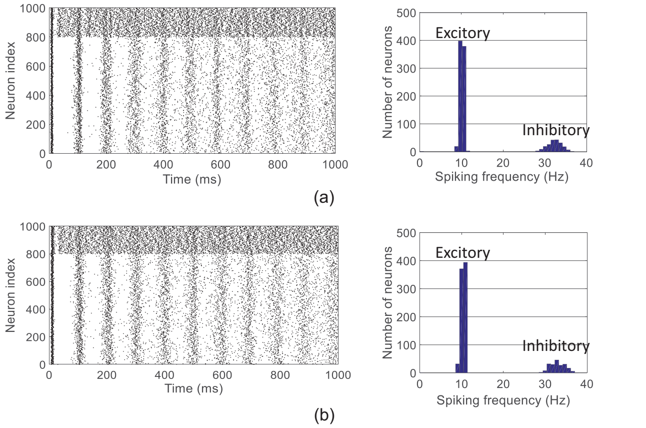
Fig 6. Neurons with different models generate similar spiking frequency after parameter tuning.

Table II. Parameters for CPU, GPU, and Embedded system.

Parameters	CPU	GPU	Embedded system
Name	Intel i7-7700k	NVIDIA GTX 1080TI	Raspberry PI 3 (ARM Cortex-A53)
RAM	32 GB (DDR4)	11 GB (GDDR5X)	1 GB (DDR3)
Maximum power	75 W	250 W	4.8 W
Maximum throughput	320 GFLOPS	11.3 TFLOPS	3.62 GFLOPS
Programming tools	MATLAB	CUDA	Python

B. Performance analysis with ROI approximation

We first evaluate the proposed ROI approximation computing algorithm with a randomly connected BNN containing 10^4 neurons (all excitatory) and 10^5 synapses. At the beginning, all neurons follow the Hodgkin-Huxley model. The spiking frequency distribution is plotted in Fig. 8(a). Fig. 8(b) shows the running time with different model switching threshold (high activity neurons are in Hodgkin-Huxley



Simulator	Brian	Nest	Our work	CARLsim	Our work
Speed (10^3 iterations)	1.89s	1.25s	0.46s	0.26s	0.18s
Platform/model	CPU/LIF			GPU/Izhikevich	

Fig. 7. Correctness verification and computing speed comparison. Raster plot and spiking frequency distribution for (a) our work and (b) CARLsim. Insert table lists the speed for different simulators. For CPU and GPU implementation, LIF and Izhikevich model are considered, respectively. Data are measured for BNN containing 10^3 neurons and 10^4 synapses.

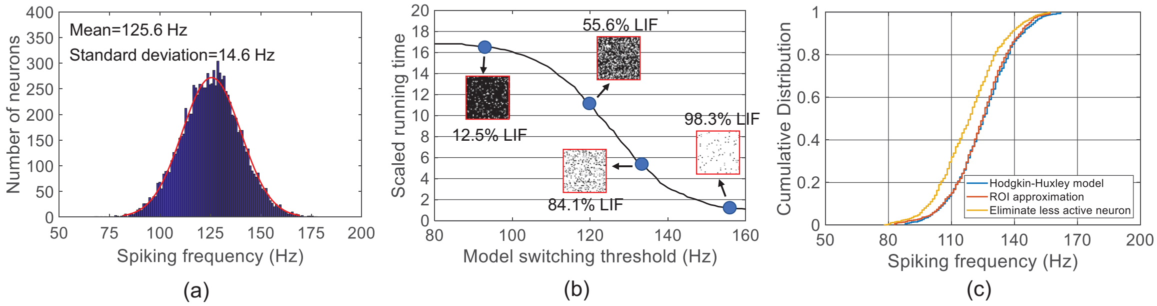


Fig. 8. (a) Spiking frequency distribution for all Hodgkin-Huxley scenario. (b) Normalized running time for ROI approximation with different threshold, inserts are the neuron model map with black for Hodgkin-Huxley and white for LIF. (c) Cumulative distribution function (CDF) for all Hodgkin-Huxley, ROI approximation, and eliminating the less active neurons.

model, low activity neurons are in LIF model). Data are normalized with the running time when all the neurons are switched to LIF model. The insert figures in Fig. 8(b) show the neuron activities with black representing Hodgkin-Huxley and white representing LIF, respectively. Simulation also indicates that the spiking frequency distribution after implementing ROI approximation is almost identical to the Hodgkin-Huxley baseline (blue and red lines in Fig. 8(c)). However, if we eliminate the less active neuron (disconnect them from the networks), the distribution changes a lot (yellow line in Fig. 8(c)). We conclude that even though the internal dynamics of the unconcerned region is less important, we cannot remove them as they are still critical for signal propagation.

C. Accelerate visual cortex modeling with ROI approximation

With the proposed computing algorithms, we implement the visual cortex modeling based on the theories of color vision: Young-Helmholz theory and opponent-process theory [14, 15]. Young-Helmholz theory states that there are three types of cone photoreceptors that are sensitive to short-wavelength (blue light), medium-wavelength (green light), and long-wavelength (red light), respectively. The opponent-process theory states that the cone photoreceptors are linked together to form three opposing color pairs: blue/yellow, red/green, and black/white. Activation of one member of the pair inhibits the other. Fig. 9(a) shows the structure of BNN for visual cortex simulation. The neurons in the first layer are based on Young-Helmholz theory and are divided into three groups, sensitive to blue light, green light, and red light, respectively. The second layer contains four groups of opponent-process theory based neurons. The third layer is the output layer, similar with the first layer, sensitive to different colors. The connectivity and neuron numbers are specified in the figure. Here we assume the output layer as ROI region.

We run simulation with two types of input signals: static input using an image and time-varying input using a video, as shown in Fig. 9(b, c). We plot the spiking frequency map for the last group of neurons (sensitive to red color) in the third layer considering the BNN with LIF neuron, Izhikevich neuron, Hodgkin-Huxley neuron, and ROI approximation (Hodgkin-Huxley for ROI and LIF for the rest). The results are shown in Fig. 10. Neurons with highest activity are white and neurons with lowest activity are black. Note that we flip the light intensity for the video input to intensify the dancer. All neuron models and ROI approximation can seasonably represent the color sensitivity. However, the dynamics of individual neuron are very different. We randomly select a neuron in the ROI (3rd layer) and plot the membrane potential (Fig. 11). We observe that the ROI based approach matches well with the results from pure Hodgkin-Huxley baseline.

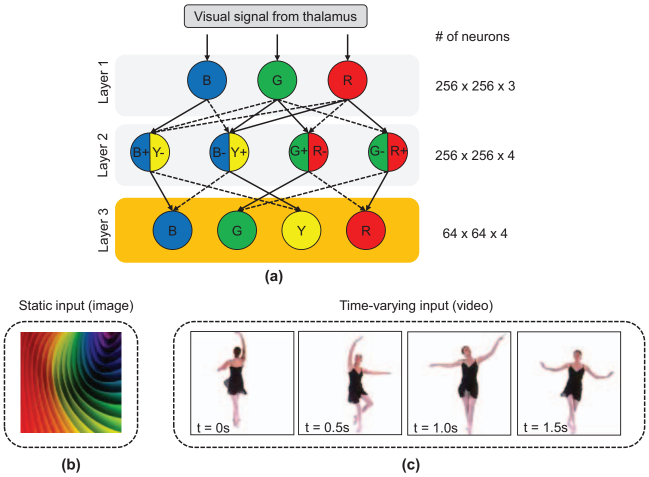


Fig. 9. (a) Visual cortex model considering Young-Helmholz theory and opponent-process theory. Solid lines represent synaptic connections from excitatory pre-neurons. Dash lines represent synaptic connections from inhibitory pre-neurons. (b) Static input with an image. (c) Time-varying input with video, here we show 4 frames.

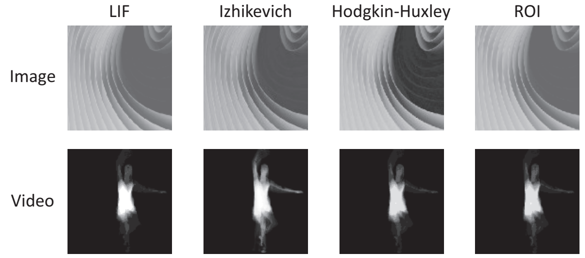


Fig. 10. Spiking frequency map for (a) static input with an image; and (b) time-varying input with a video (the spiking frequency is sampled at t=0.5s).

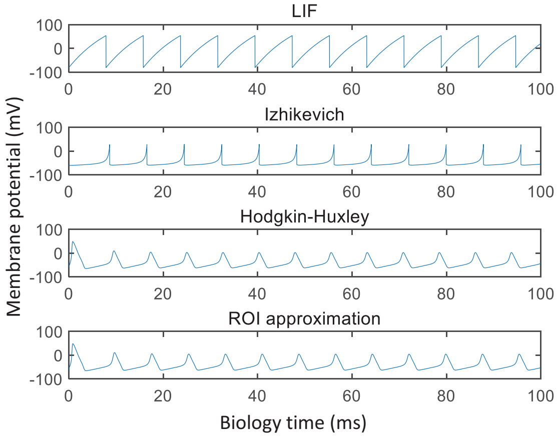
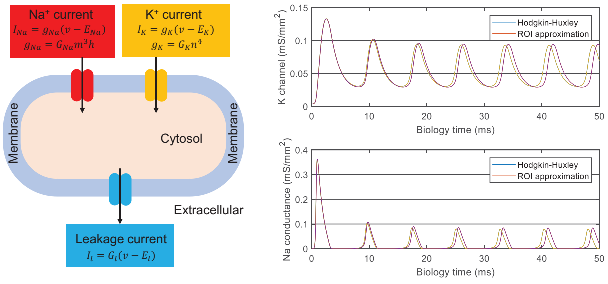


Fig. 11. Membrane potential for the neuron in the same location with different neuron models.

Moreover, the proposed ROI based approximation fully maintains the biology details such as ion channel (Fig. 12(a)) conductance in the interested regions. We plot the Na^+ and K^+ channel conductance for Hodgkin-Huxley based simulation and ROI approximation in Fig. 12(b, c). We observe a phase delay between Hodgkin-Huxley and ROI approximation which is caused by the minor models output mismatch after the parameter tuning. The phase delay can be eliminated by a time shifting.

Finally, we evaluate the computing speed of the proposed algorithms running on different platforms with image and video as BNN input, shown in Fig. 13. We observe that ROI based approximation can significantly enhance the computing efficiency with an average speed up of 11x over the pure Hodgkin-Huxley baseline. Therefore, in comparison to prior simulators, the template based ROI approximation computing algorithm achieves equivalent speedup of 37x and 16x for CPU (Nest and Brian) and GPU (CARLsim), respectively.



(a)

(b)

Fig. 12. (a) Voltage-gated ion channel and leakage channel for a neuron cell based on Hodgkin-Huxley model. (b) Ion channels conductance simulated from Hodgkin-Huxley and ROI approximation.

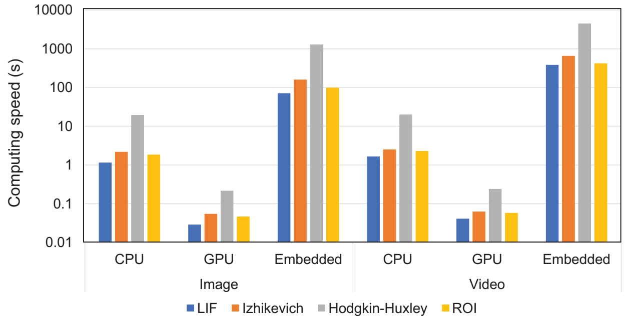


Fig. 13. Running time for visual cortex simulation. The time step is 0.01 ms and we run 20 ms (1000 iterations) biology time.

V. Conclusion

This paper presents an adaptive BNN simulation methodology that maintains high biological accuracy with more complex neuron models in the ROI while simplifies the neuron models elsewhere to improve the computation speed. A template based computing approach is presented to enable the simulation of heterogeneous BNN with dynamically varying neuron model. The proposed template based computing coupled with ROI approximation demonstrates more than one order of magnitude speedup over existing BNN simulators.

References

[1] R. Ananthanarayanan, S. K. Esser, H. D. Simon *et al.*, "The cat is out of the bag: cortical simulations with 109 neurons, 1013 synapses," in *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, 2009, pp. 1-12.

[2] G. Indiveri, E. Chicca, and R. J. Douglas, "Artificial Cognitive Systems: From VLSI Networks of Spiking Neurons to Neuromorphic Cognition," *Cognitive Computation,* journal article vol. 1, no. 2, pp. 119-127, June 01 2009.

[3] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Frontiers in Computational Neuroscience,* vol. 9, p. 99, 2015.

[4] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science,* vol. 345, no. 6197, pp. 668-673, 2014.

[5] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of physiology,* vol. 117, no. 4, pp. 500-544, 1952.

[6] E. M. Izhikevich, "Which model to use for cortical spiking neurons?," *IEEE transactions on neural networks,* vol. 15, no. 5, pp. 1063-1070, 2004.

[7] R. Brette, M. Rudolph, T. Carnevale *et al.*, "Simulation of networks of spiking neurons: a review of tools and strategies," *Journal of computational neuroscience,* vol. 23, no. 3, pp. 349-398, 2007.

[8] M. Beyeler, K. D. Carlson, T.-S. Chou *et al.*, "CARLsim 3: A user-friendly and highly optimized library for the creation of neurobiologically detailed spiking neural networks," in *2015 International Joint Conference on Neural Networks (IJCNN)*, 2015, pp. 1-8: IEEE.

[9] B. V. Benjamin, P. Gao, E. McQuinn *et al.*, "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations," *Proceedings of the IEEE,* vol. 102, no. 5, pp. 699-716, 2014.

[10] M.-O. Gewaltig and M. Diesmann, "Nest (neural simulation tool)," *Scholarpedia,* vol. 2, no. 4, p. 1430, 2007.

[11] M. Abadi, A. Agarwal, P. Barham *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467,* 2016.

[12] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *Selected Papers of Hirotugu Akaike*: Springer, 1998, pp. 199-213.

[13] D. Goodman and R. Brette, "Brian: a simulator for spiking neural networks in Python," *Frontiers in neuroinformatics,* vol. 2, 2008.

[14] T. Young, "The Bakerian lecture: On the theory of light and colours," *Philosophical transactions of the Royal Society of London,* vol. 92, pp. 12-48, 1802.

[15] R. L. Solomon, "The opponent-process theory of acquired motivation: The costs of pleasure and the benefits of pain," *American psychologist,* vol. 35, no. 8, p. 691, 1980.