

GPGPU ACCELERATED SIMULATION AND PARAMETER TUNING FOR NEUROMORPHIC APPLICATIONS

**Jeff Krichmar, Kris Carlson,
Michael Beyeler, Nikil Dutt**

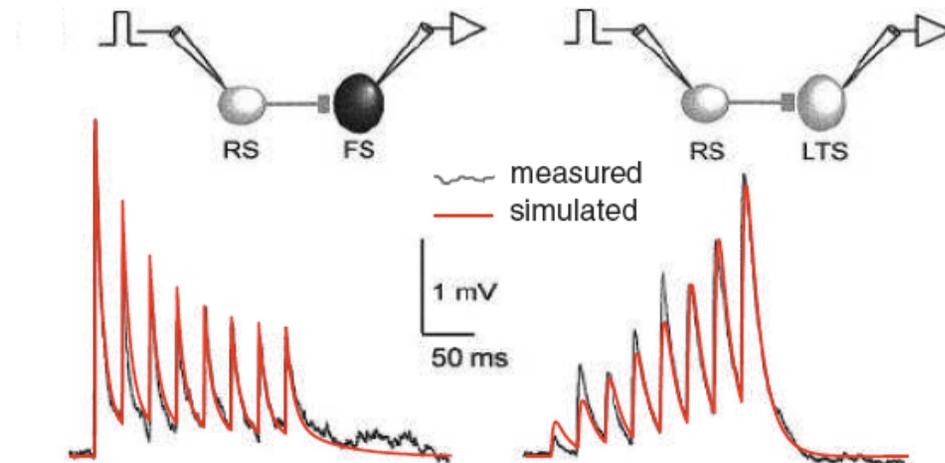
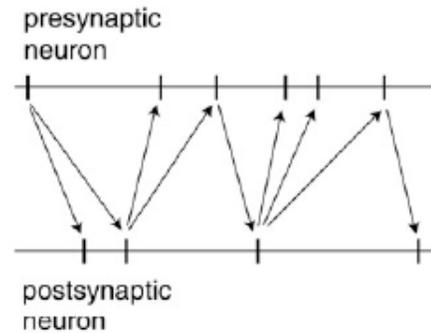
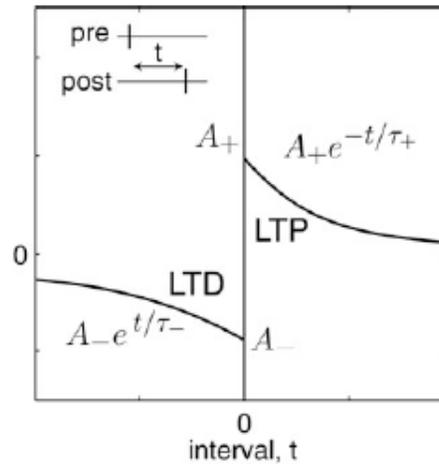
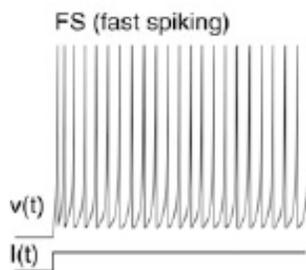
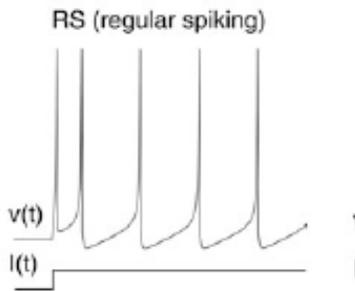
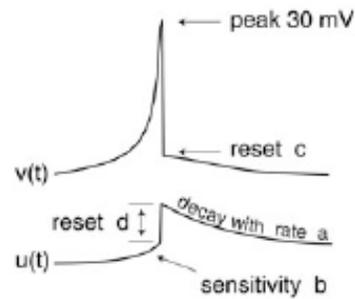
Cognitive Anteater Robotics Lab (CARL)
Department of Cognitive Sciences
Department of Computer Science
University of California, Irvine, USA



Spiking Neural Networks (SNNs)

- **What are SNNs?**
 - Neural Networks that model neuronal/synaptic temporal dynamics
 - Spike only when the membrane voltage exceeds a threshold
- **Why use SNNs?**
 - Spike events are rare: average brain activity 1-10 Hz
 - More energy efficient than sending an analog rate.
 - SNNs use temporal coding but can still use rate coding
 - Event-driven nature of SNNs fits well with neuromorphic hardware
 - Use “**Address Event Representation**” (**AER**) to minimize communication
 - SNNs support a biologically plausible learning algorithm: Spike Timing-Dependent Plasticity (STDP)
 - Unsupervised and stable learning rule.

Modeling Components at the Neural Circuit Level



Some Examples of Neuromorphic Hardware Devices

Hardware Project: Hardware Group	Hardware Description	Neuron Models	Synaptic Plasticity	Max Neurons	Max Synapses
SpiNNaker: Industry and UK universities	<ul style="list-style-type: none"> - Completely digital - Consists of array of nodes - Each node has 18 ARM9 cores - Final goal: 1,036,800 cores 	Spiking: Izhikevich and non-spiking	Yes: STDP	1,000 neurons per ARM9 core	10k synapses per ARM9 core
Neurogrid: Stanford University	<ul style="list-style-type: none"> - Analog/digital hybrid - Full board has 16 neurochips - Operates on only 5 W 	Spiking: Two-compartment neurons	No	65,536 neurons per neurochip	375M synapses per neurochip
True North Cog. Architecture: IBM SyNAPSE Team	<ul style="list-style-type: none"> - Completely digital - Consists of hierarchical design - Neurosynaptic core is basic building block 	Spiking: many behaviors including LIF	No	256 neurons per neuro-synaptic core	256K binary synapses per neuro-synaptic core
HRL neural chip: HRL Labs, SyNAPSE Team	<ul style="list-style-type: none"> - Analog/digital hybrid - Synaptic weights stored in memristors 	Spiking: Izhikevich	Yes: STDP	576 neurons per chip	70k virtual synapses per chip
HiCANN: BrainScaleS Team	<ul style="list-style-type: none"> - Analog/digital hybrid - Each wafer has 384 chips - Neurons are analog - Synapses are digital 	Spiking: AdExp and I&F	Yes: STDP	512 neurons per chip	16k synapses per chip

Some Examples of Neuromorphic Software Tools

Software Project	Features	Parallelized Implementations	Implementation Language	Parameter Tuning Tools
NENGO	<ul style="list-style-type: none"> - Uses neural engineering framework (NEF) - Set weights to perform specific computations - Uses both rate-based and spiking neurons - Uses neural plasticity rules (STDP) as well 	None	<ul style="list-style-type: none"> - Core: Java - Python scripting 	NEF
NEST	<ul style="list-style-type: none"> - Mature codebase for multiple platforms - Includes many neuron and plasticity models - Built-in simulation language interpreter - Module for creating complex networks 	Parallelized MPI CPU implementation	<ul style="list-style-type: none"> - Core: C++ - Interface: Python - PyNN support 	None
Brian	<ul style="list-style-type: none"> - Multiple integration methods - Multiple neuron and plasticity models - Uses Python plotting packages - Good documentation 	Parallelized CPU support Parallelized GPU support only for tuning component	<ul style="list-style-type: none"> - Core: Python - PyNN support 	Support for tuning neuron models
CARLsim	<ul style="list-style-type: none"> - Fast and efficient CUDA GPU implementation - Support for key ion channels - GPU parallelized general tuning framework - Includes highly optimized CUDA vision frontend 	Parallelized CUDA GPU implementation	<ul style="list-style-type: none"> - Core: C++ and CUDA - Syntax similar to PyNN 	General tuning framework using EAs

Some Examples of Neuromorphic Software Tools

Software Project	Features	Parallelized Implementations	Implementation Language	Parameter Tuning Tools
NENGO	<ul style="list-style-type: none"> - Uses neural engineering framework (NEF) - Set weights to perform specific computations - Uses both rate-based and spiking neurons - Uses neural plasticity rules (STDP) as well 	None	<ul style="list-style-type: none"> - Core: Java - Python scripting 	NEF
NEST	<ul style="list-style-type: none"> - Mature codebase for multiple platforms - Includes many neuron and plasticity models - Built-in simulation language interpreter - Module for creating complex networks 	Parallelized MPI CPU implementation	<ul style="list-style-type: none"> - Core: C++ - Interface: Python - PyNN support 	None
Brian	<ul style="list-style-type: none"> - Multiple integration methods - Multiple neuron and plasticity models - Uses Python plotting packages - Good documentation 	Parallelized CPU support Parallelized GPU support only for tuning component	<ul style="list-style-type: none"> - Core: Python - PyNN support 	Support for tuning neuron models
CARLsim	<ul style="list-style-type: none"> - Fast and efficient CUDA GPU impl. - Support for key ion channels - GPU parallelized general tuning framework - Includes highly optimized CUDA vision frontend 	Parallelized CUDA GPU implementation	<ul style="list-style-type: none"> - Core: C++ and CUDA - Syntax similar to PyNN 	General tuning framework using EAs

CARLsim Applications

- Used to create large-scale simulations of cognitive processes
 - 10k – 100K neurons with millions of synapses
- Sample Applications
 - **Visual Processing**
 - Large-scale model of cortical areas V1, V4, and area MT
 - Richert, M., Nageswaran, J.M., Dutt, N., and Krichmar, J.L. (2011). An efficient simulation environment for modeling large-scale cortical processing. *Frontiers in Neuroinformatics* 5, 1-15.
 - **Neuromodulation**
 - Top-down and bottom-up attention
 - Avery, M.C., Dutt, N., and Krichmar, J.L. (2013). Mechanisms underlying the basal forebrain enhancement of top-down and bottom-up attention. *The European Journal of Neuroscience*.
 - Working memory and behavior
 - Avery, M., Dutt, N., and Krichmar, J.L. (2013). A large-scale neural network model of the influence of neuromodulatory levels on working memory and behavior. *Frontiers in Computational Neuroscience* 7.
 - **Object Categorization**
 - Classifying handwritten digits, semi-supervised learning
 - Beyeler, M., Dutt, N.D., and Krichmar, J.L. (2013). Categorization and decision-making in a neurobiologically plausible spiking network using a STDP-like learning rule. *Neural Netw* 48, 109-124.
 - **Neural Plasticity**
 - Biologically plausible STDP and Homeostatis
 - Carlson, K.D., Richert, M., Dutt, N., and Krichmar, J.L. (2013). Biologically Plausible Models of Homeostasis and STDP: Stability and Learning in Spiking Neural Networks. Paper presented at: International Joint Conference on Neural Networks (Dallas, TX: IEEE Explore)
- Code available at: <http://www.socsci.uci.edu/~jkrichma/CARLsim>

Example of Large-Scale SNN Simulation Visual Motion Perception in Cortex

Neuroinform

DOI 10.1007/s12021-014-9220-y

ORIGINAL ARTICLE

Efficient Spiking Neural Network Model of Pattern Motion Selectivity in Visual Cortex

**Michael Beyeler • Micah Richert • Nikil D. Dutt •
Jeffrey L. Krichmar**

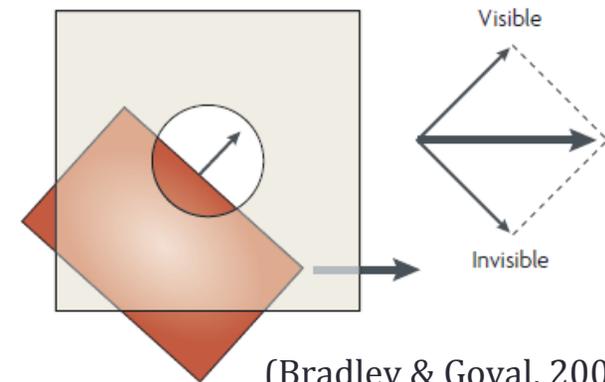
Neuroinformatics, February 2014

Visual Motion Perception

- Visual motion perception
 - Critical for navigating through the environment, while tracking objects.
 - Computationally expensive & memory-intensive.
- Goal / motivation:
 - Understand cortical machinery for visual motion perception through modeling
 - Build more powerful artificial vision systems
- Spiking neural networks (SNN) on GPUs
 - Low-cost yet high-performance approach
 - Enables real-time processing of visual motion
 - Potential application for neuromorphic hardware

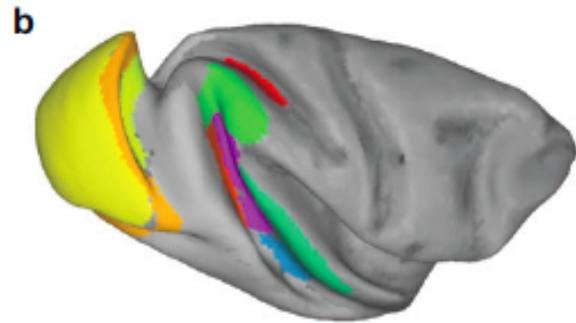


(Britten, 2008)

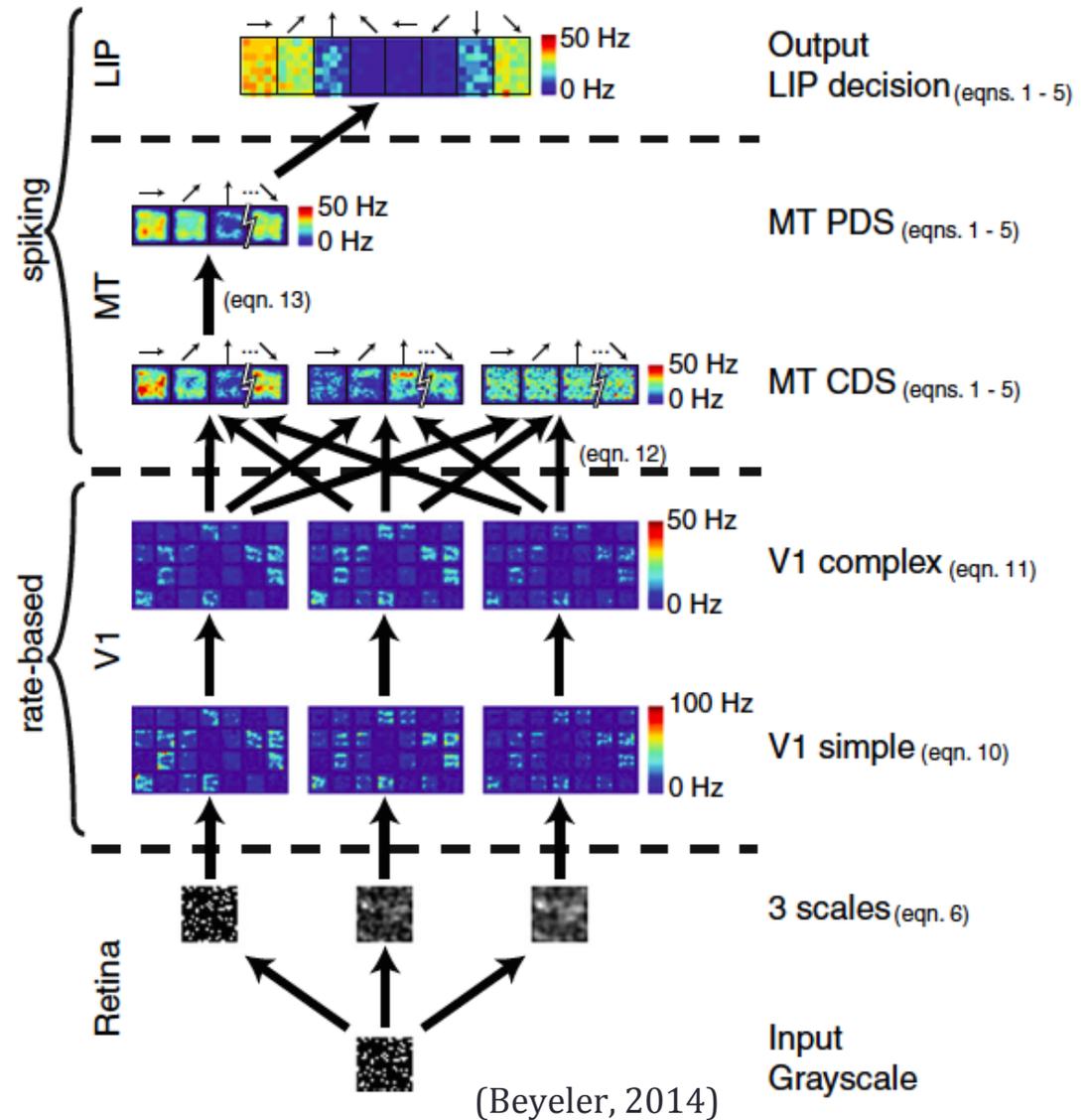
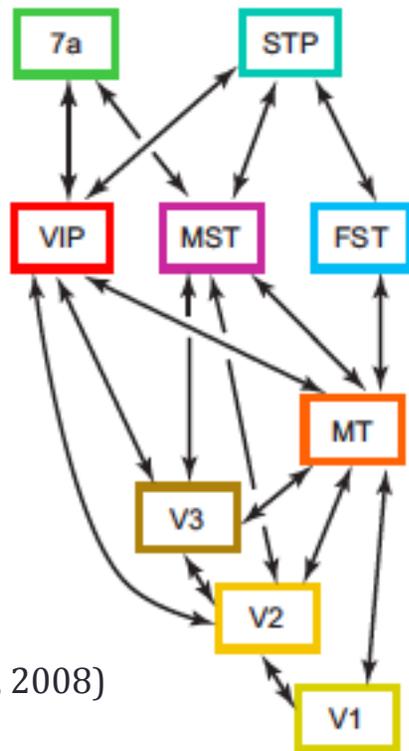


(Bradley & Goyal, 2008)

Visual Cortex and Motion Perception Model

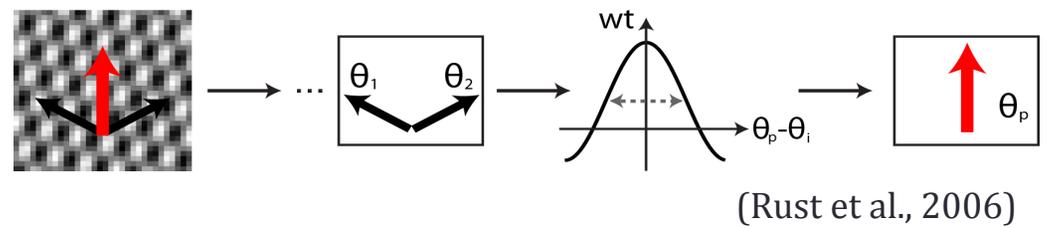
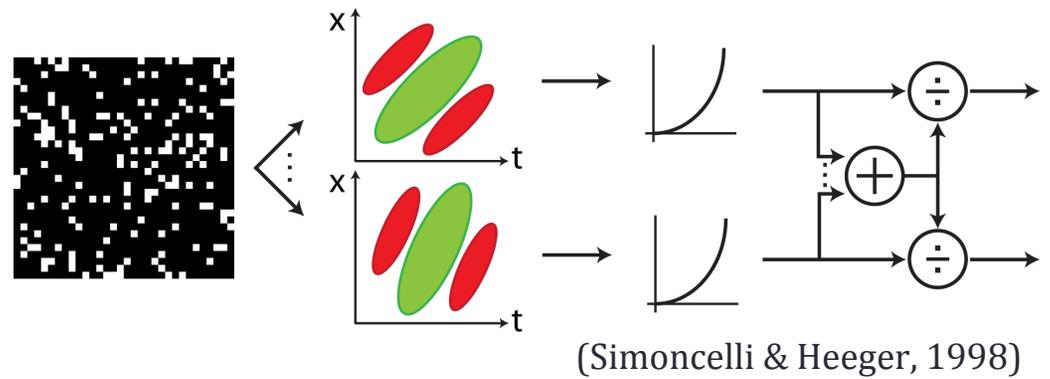


(Britten, 2008)



V1 and MT Model

- Spatiotemporal-energy model of V1
 - Bank of linear space-time oriented filters (rate-based)
 - Direction-selective cells
 - Fully realized in CUDA
- Two-stage spiking model of MT
 - Izhikevich spiking neurons: regular-spiking / fast-spiking
 - 153,216 neurons, ~40 million synapses
 - Runs in real-time with video (32x32 pixels).
 - Conductance-based synapses: AMPA, NMDA, GABA_A, GABA_B
 - Pattern-direction-selective cells:
 - direction pooling + opponent inhibition
 - signal the direction of global (pattern) motion
 - solve the aperture problem



Model Response to Motion Patterns

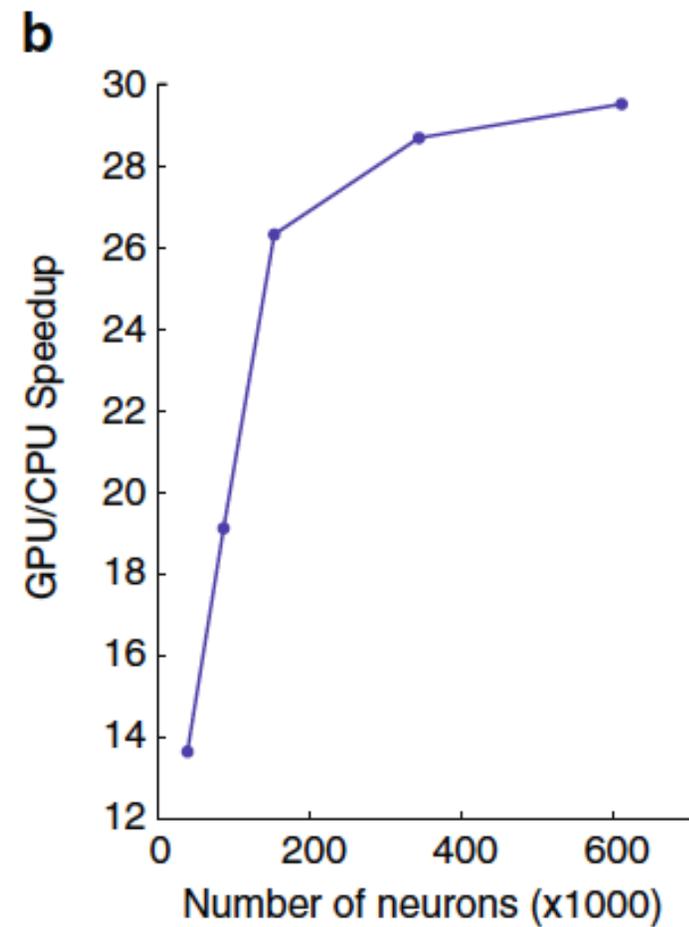
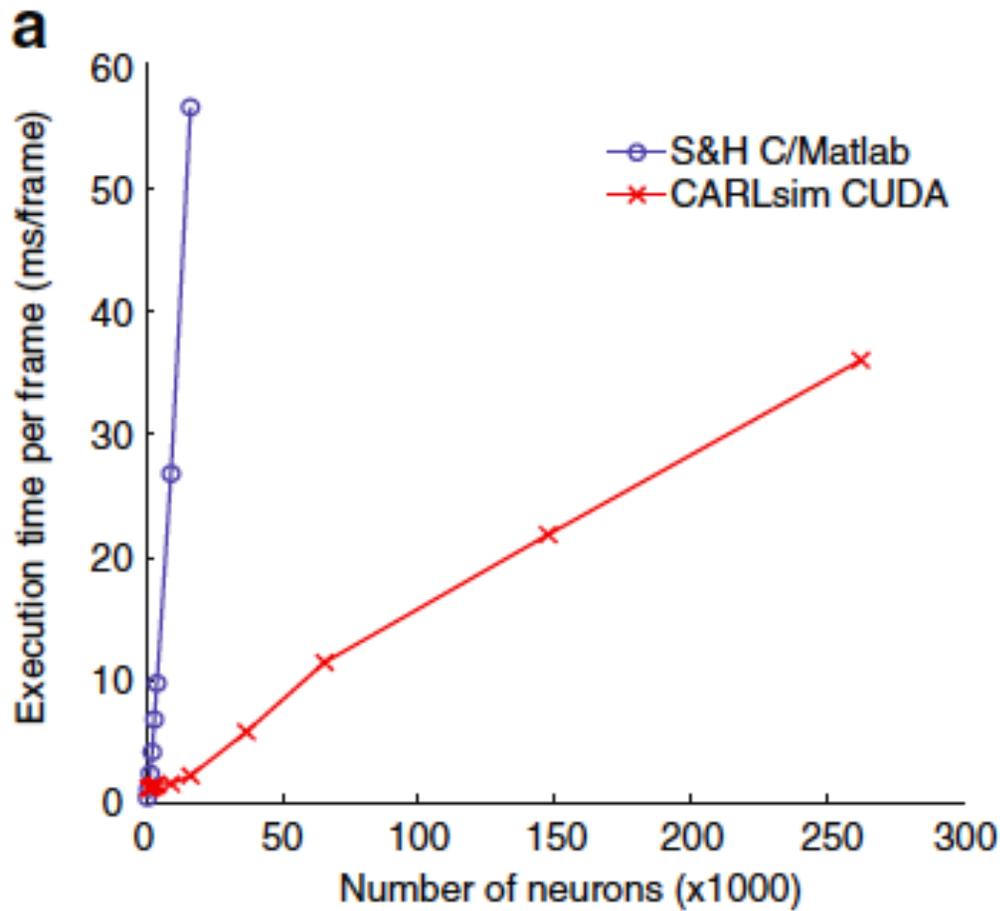
Component and Pattern Selectivity

**Component
-direction-
selective**

**Pattern-
direction-
selective**



Large-scale Cortical Model of Motion Perception: Computationally Efficient & Scalable

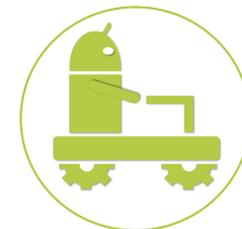


Possible Neuromorphic Application

Autonomous Vision Based Navigation



- LeCarl – Android Based Robot
 - Video frames from Samsung Galaxy Smartphone fed to CarlSim MT model
 - For more information on Android Based Robotics:
 - <http://www.socsci.uci.edu/~jkrichma/ABR/index.html>



Example – Automated Tuning of Large-Scale SNNs

frontiers in
NEUROSCIENCE

ORIGINAL RESEARCH ARTICLE

published: 04 February 2014
doi: 10.3389/fnins.2014.00010



An efficient automated parameter tuning framework for spiking neural networks

Kristofor D. Carlson¹, Jayram Moorkanikara Nageswaran², Nikil Dutt³ and Jeffrey L. Krichmar^{1,3*}

¹ Department of Cognitive Sciences, University of California Irvine, Irvine, CA, USA

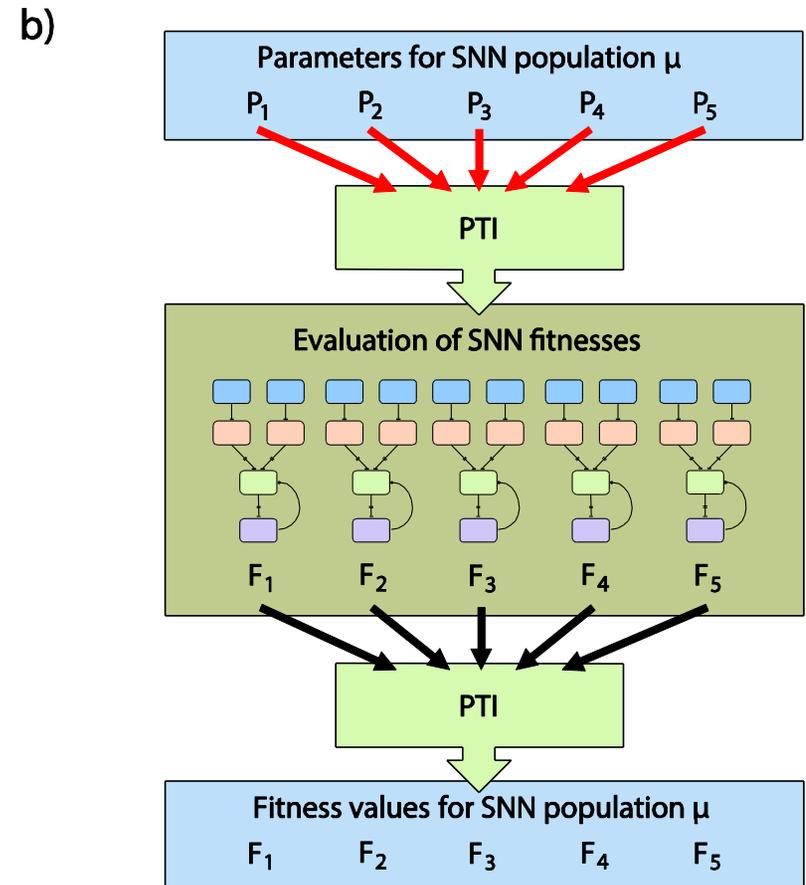
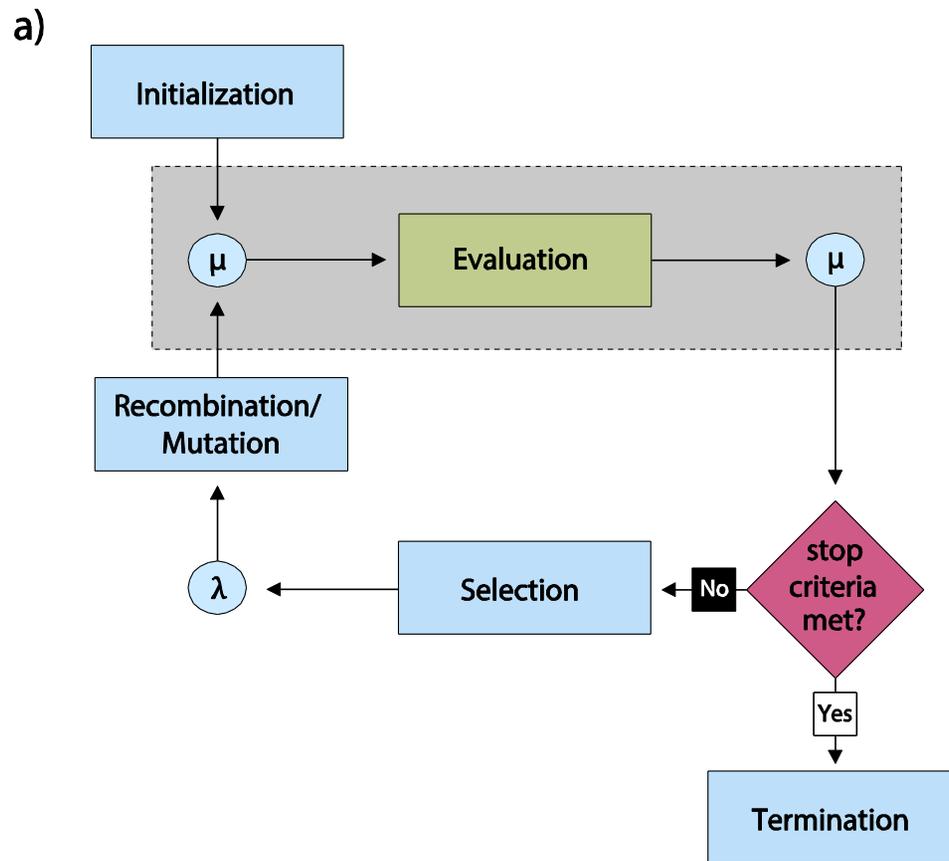
² Brain Corporation, San Diego, CA, USA

³ Department of Computer Science, University of California Irvine, Irvine, CA, USA

Automated Parameter Tuning Framework

- Tuning large-scale spiking neural networks is time consuming with many open parameters.
- Our approach to parameter tuning leverages:
 - Recent progress in evolutionary algorithms.
 - Optimization with off-the-shelf graphics processing units (GPUs)
- The parameter search is guided by principles of neuroscience
 - Biological networks adapt their responses to increase the amount of transmitted information, reduce redundancies, and span the stimulus space

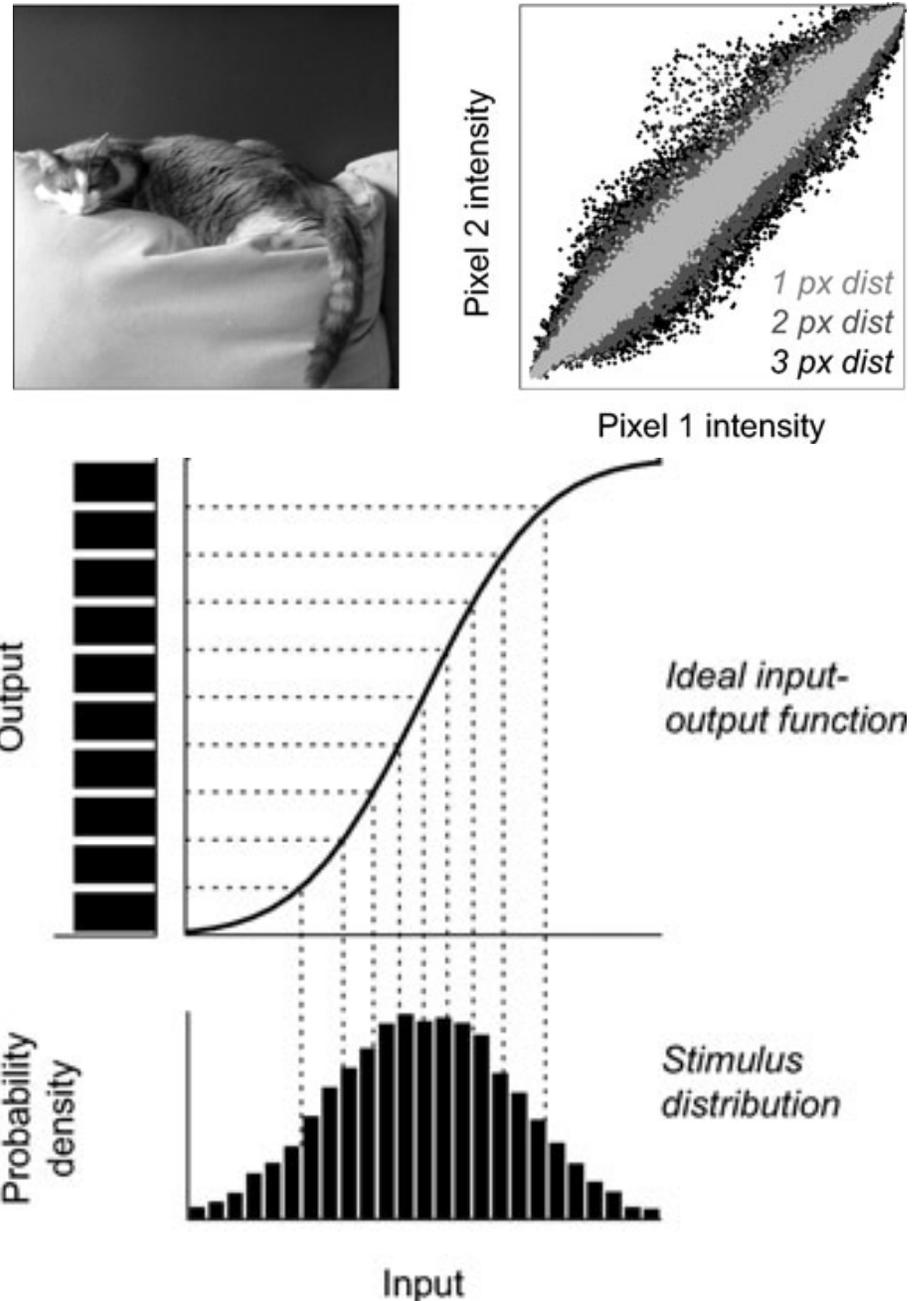
Automated Parameter Framework for Tuning Spiking Neural Networks (SNNs)



Efficient Coding Hypothesis

- Fundamental idea:
 - Sensory systems adapt their responses to the regularities of their input
 - Increase the amount of transmitted information at any given time

 1. Maximize efficiency (*reduce redundancy*)
 2. Responses should be independent of one another (*decorrelation*)
 3. A stimulus should involve only a small fraction of the available neurons (*sparse*)



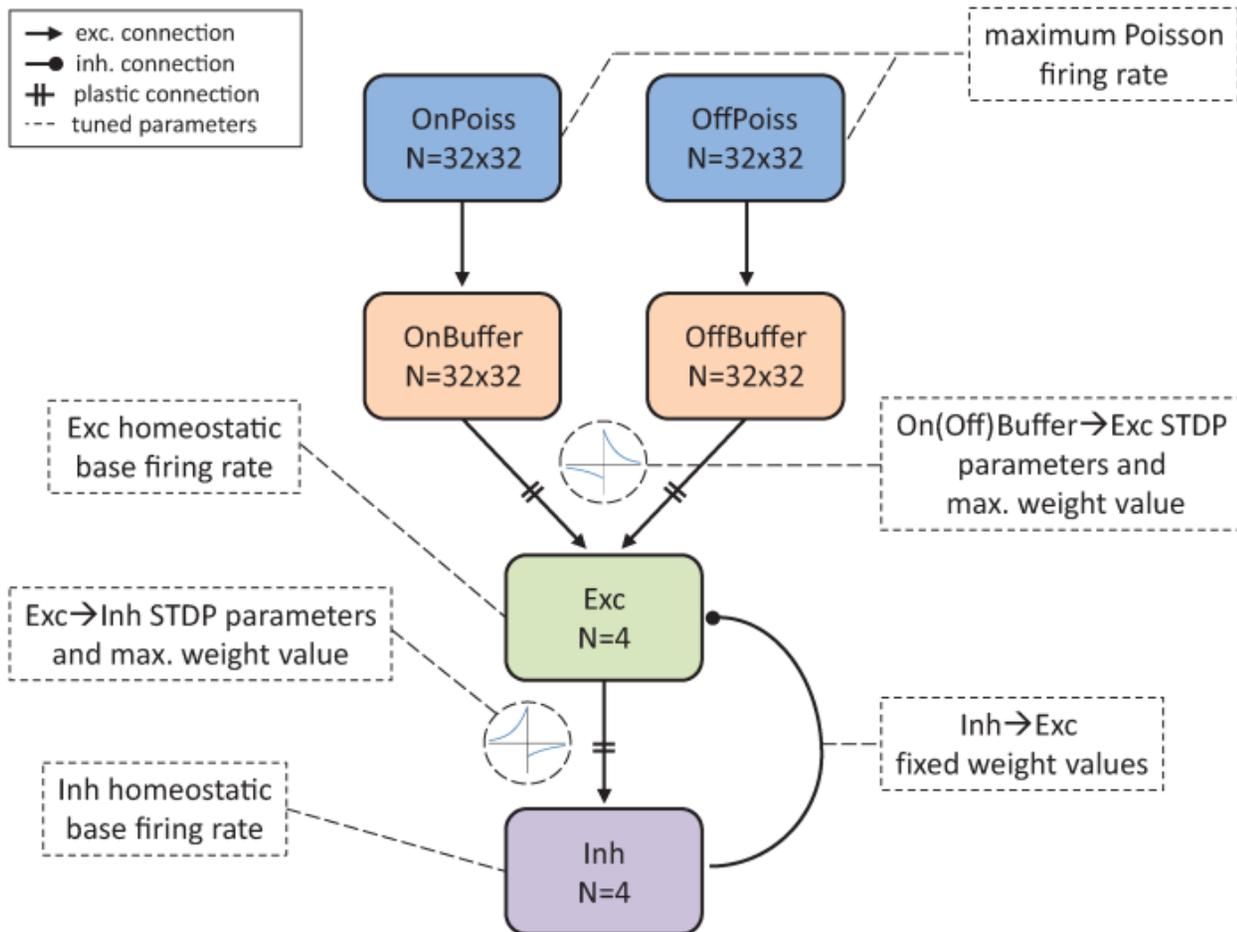
Louie, K., and Glimcher, P.W. (2012). Efficient coding and the neural representation of value. *Ann N Y Acad Sci* 1251, 13-32.

Fitness Function Based on the Efficient Coding Hypothesis

$$Fitness_{total} = \frac{1}{Fitness_{decorr} + Fitness_{Gauss} + K_{scaling\ factor} \cdot Fitness_{maxRate}}$$

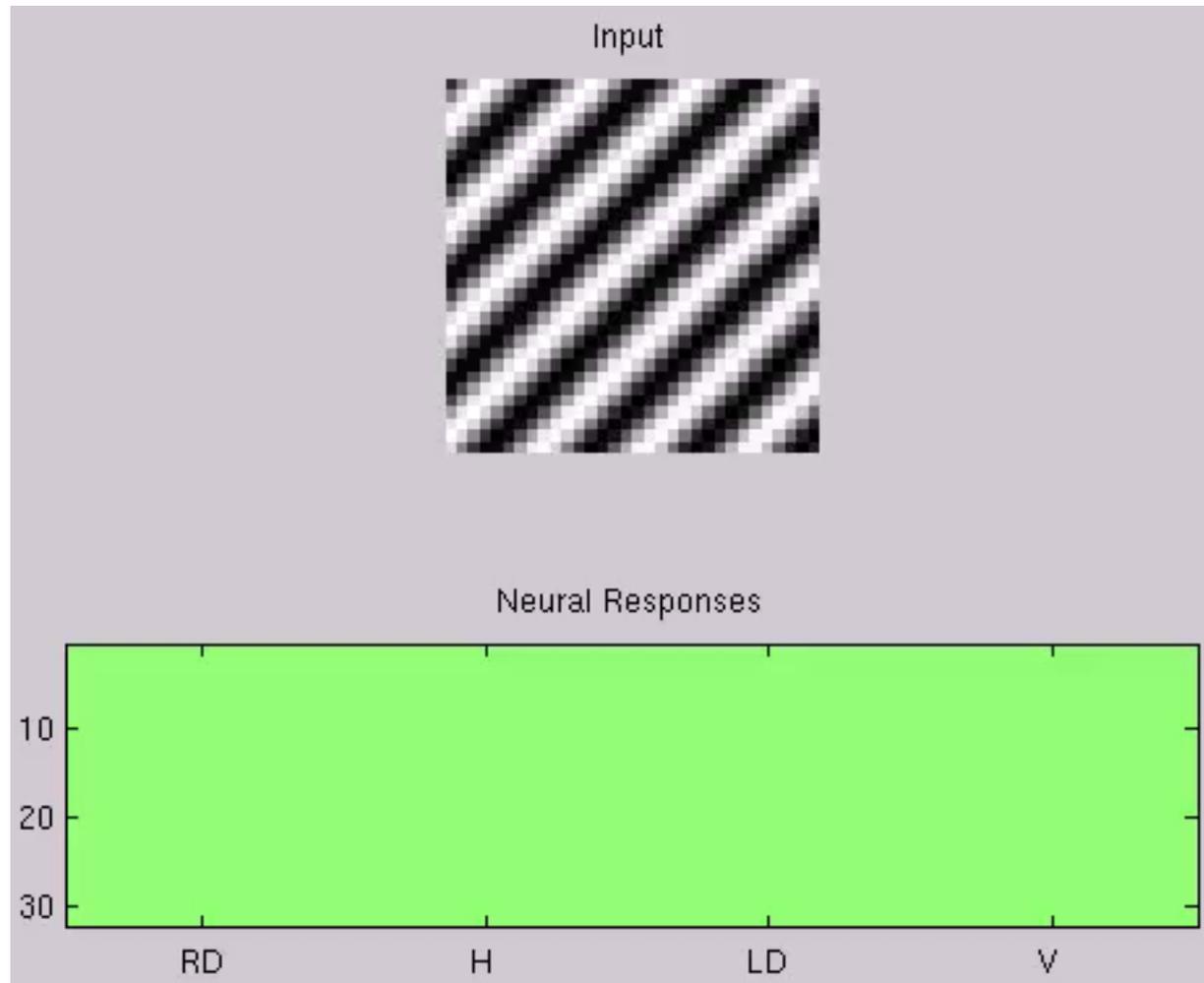
- $Fitness_{decorr}$ ensured decorrelation by forcing each neuron to respond maximally to different stimuli
- $Fitness_{Gauss}$ required Gaussian tuning curves. Also lead neurons to employ their full response range to describe the stimulus
- $Fitness_{maxRate}$ limited the maximum firing rate of each neuron which contributes to sparsity

Tuning SNNs that Generate Self Organizing Receptive Fields (SORF)

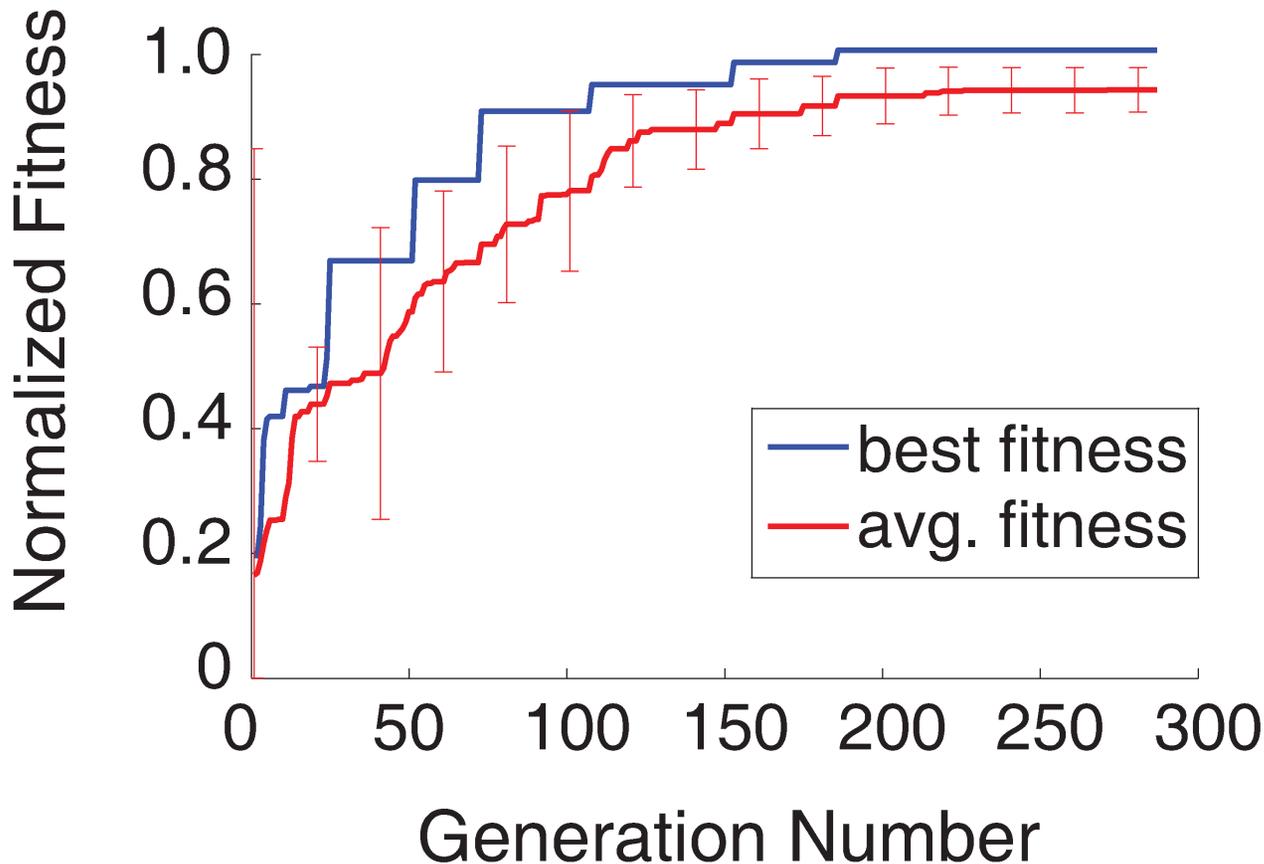


- Network size
 - 4104 neurons
- Indirect encoding
 - 14 parameters to search
- Training phase
 - 40 sinusoidal orientations presented
 - 2400 presentations
- Testing phase
 - 8 sinusoidal orientations presented to the network
 - Responses of the Exc neurons were evaluated using the ECH fitness function

Simulated Visual Cortex Responses to Sinusoidal Gratings



Best and Average Normalized Fitness vs. Generation Number



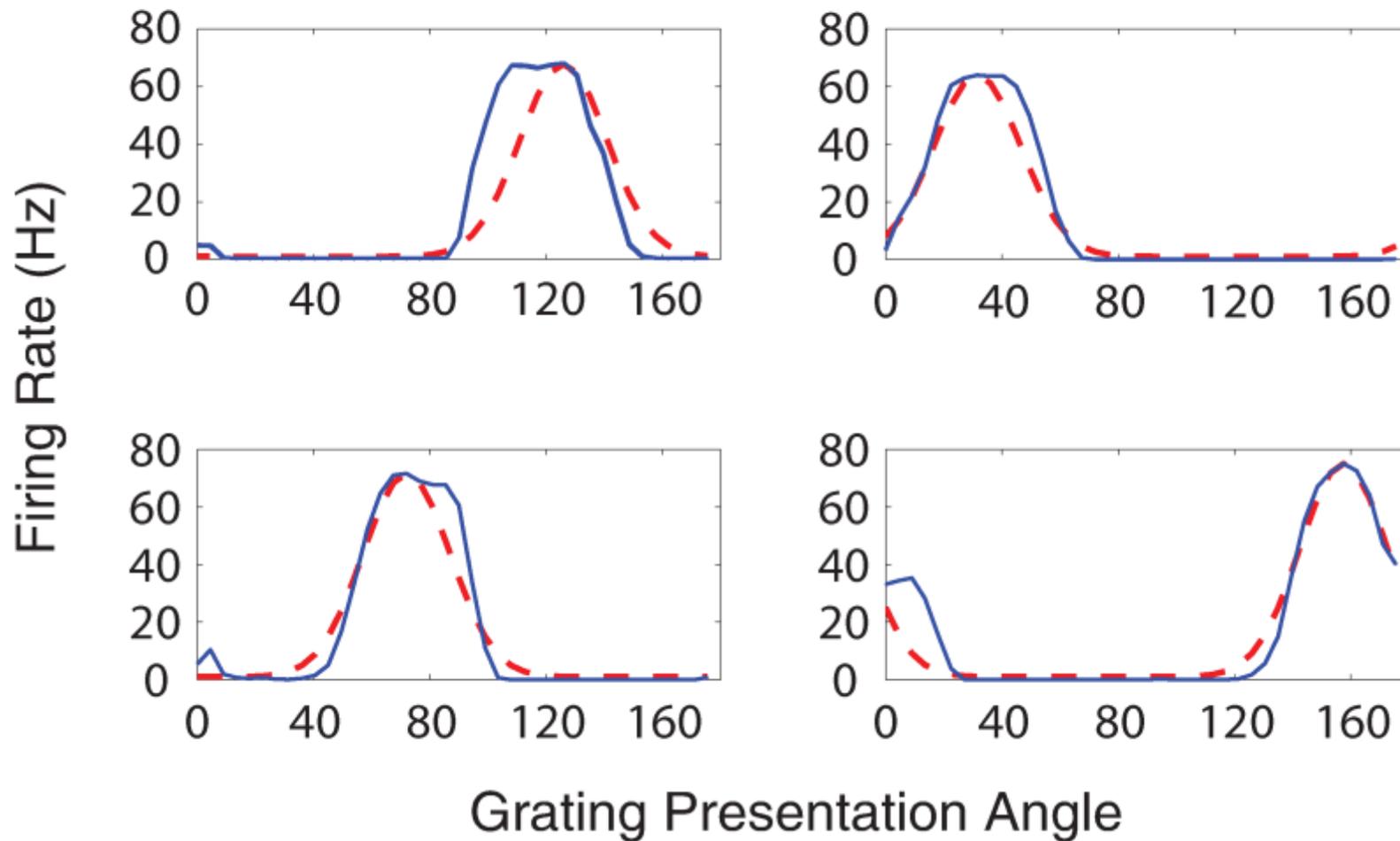
- 10 SNNs in parallel
 - Up to 40.
- 287 EA generations in 127 hours.
- Run on a single NVIDIA Tesla M2090

Synaptic Weight Progression During Training for a High Fitness Individual



- Synaptic weights for the On(Off)Buffer→Exc connections
 - Light regions denote strong weights, dark regions denote weak weights

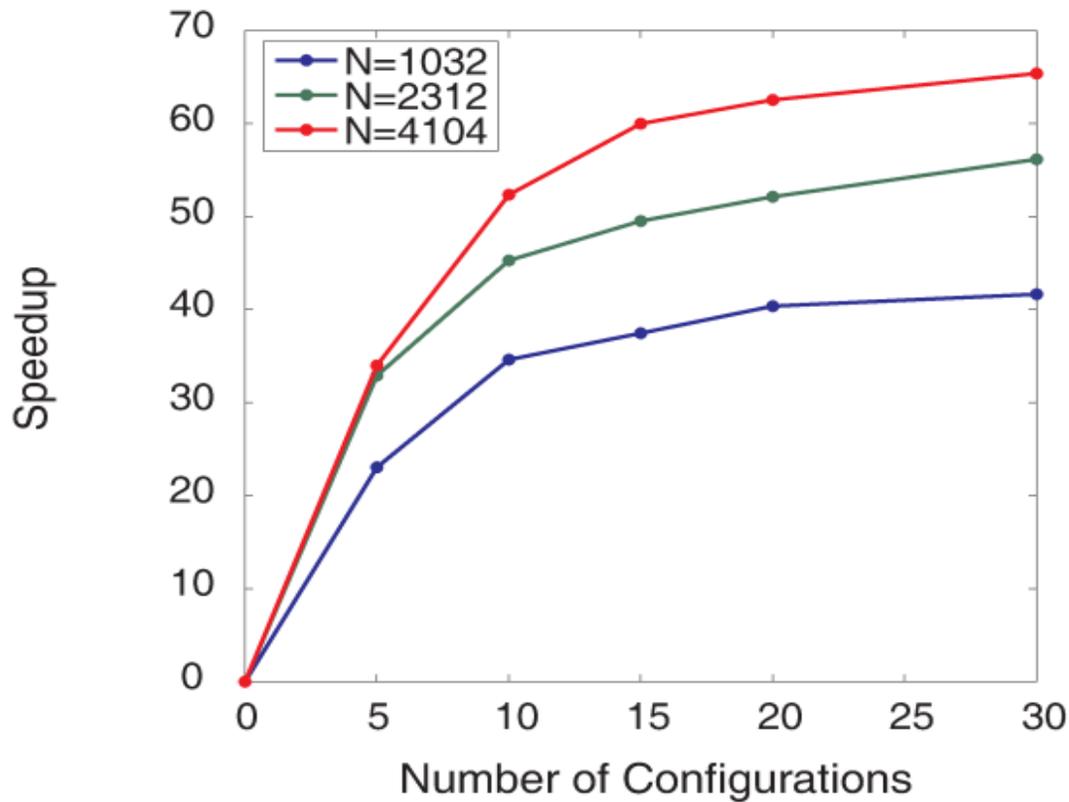
Response Of Neurons To Sinusoidal Gratings



- **Blue line:** firing rate of simulated individual Exc group neuron
- **Red line:** ideal Gaussian tuning curve firing rate response for V1

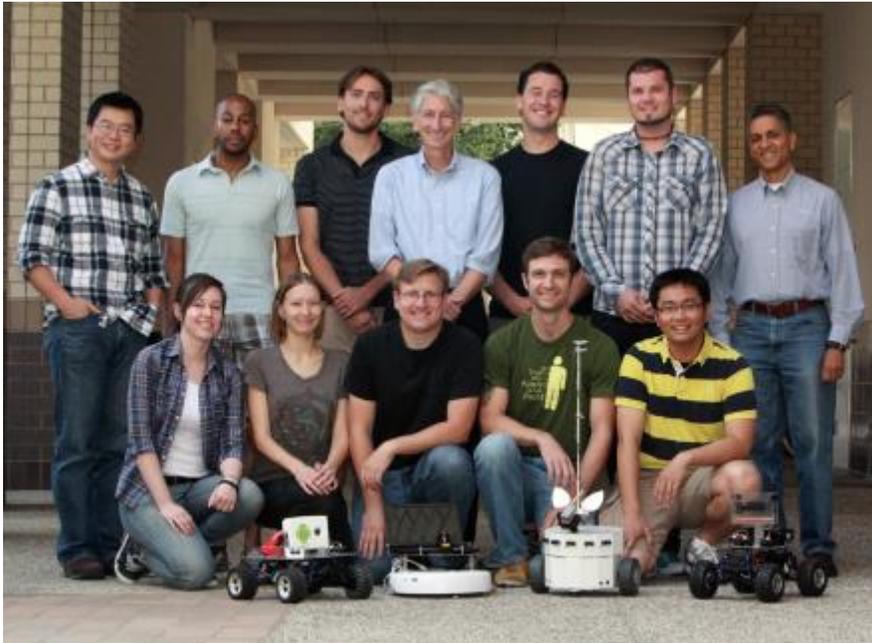
Automated Parameter Tuning Framework Performance

GPU Speedup Over CPU vs. Number of Networks Run in Parallel
For Networks of Size N=1032, 2312, and 4104



Thanks to...

Team CARL Now (2014) – UC Irvine



Front row – Alexis Craig, Emily Rounds, Kris Carlson, Liam Bucci, Ting-Shuo Chou
Back row – Feng Rong, Andrew Zaldivar, Nicolas Oros, Jeff Krichmar, Derrik Asher, Michael Beyeler, Nik Dutt

Team CARL Then (2009) – UC Irvine



Front row – Jay Nageswaran, Mike Avery, Chelsea Guthrie, Micah Richert
Back row – Andrew Zaldivar, Brian Cox, Michael Wei, Jeff Krichmar

Supported by the National Science Foundation, the Defense Advanced Research Projects Agency, and the Intelligence Advanced Research Projects Activity.

Conclusions

- Large-scale, complex, realistic brain simulations are necessary:
 - For the field of neuromorphic engineering to produce results and applications of practical value.
 - To help computational neuroscientists develop new theories of neural function.
- To address this challenge, our approach leverages:
 - Optimization capabilities of evolutionary computation.
 - Exploits graphical processing unit (GPU) parallelism.
- Implementation is compatible with neuromorphic hardware.
- Framework and examples are publicly available
 - <http://www.socsci.uci.edu/~jkrichtma/CARLsim>