SpikeNET: A simulator for modeling large networks of integrate and fire neurons

Article *in* Neurocomputing · February 1999

DOI: 10.1016/S0925-2312(99)00095-8 · Source: DBLP

4 authors, including:


Arnaud Delorme


Jacques Gautrais
Université Toulouse III - Paul Sabatier


Simon Jonathan Thorpe
Université Toulouse III - Paul Sabatier

Published in Neurocomputing, 38-40(1-4), 539-545, 2001.

SpikeNET : A Simulator for Modeling Large Networks of Integrate and Fire Neurons

Arnaud Delorme, Jacques Gautrais, Rufin van Rullen & Simon Thorpe
Centre de Recherche Cerveau & Cognition
133, route de Narbonne, 31062, Toulouse, France
arno@cerco.ups-tlse.fr, gautrais@cerco.ups-tlse.fr, rufin@cerco.ups-tlse.fr, thorpe@cerco.ups-tlse.fr

Abstract

SpikeNET is a simulator for modeling large networks of asynchronously spiking neurons. It uses simple integrate-and-fire neurons which undergo step-like changes in membrane potential when synaptic inputs arrive. If a threshold is exceeded, the potential is reset and the neuron added to a list to be propagated on the next time step. Using such spike lists greatly reduces the computations associated with large networks, and simplifies implementations using parallel hardware since inter-processor communication can be limited to sending lists of the neurons which just fired. We have used it to model complex multi-layer architectures based on the primate visual system that involve millions of neurons and billions of synaptic connections. Such models are not only biological but also efficient, robust and very fast, qualities which they share with the human visual system.

Keywords : Modeling software, Natural scenes, categorization, biological visual systems.

1. Introduction

There are currently a large number of different systems that can be used for simulating neural networks. Many have been designed for simulating networks of artificial neurons and make no attempt to model the detailed biophysics of neurons. The underlying units have no structure, and their outputs typically consist of a single continuous value (often in the range 0 to 1 or from -1 to +1). While such systems have been widely used, and have had applications in a wide range of engineering and financial areas, few would regard them as being useful as tools for the computational neuroscientist.

At the other end of the spectrum there are sophisticated programs such as GENESIS and NEURON which are good for performing detailed biophysical simulations that take into account factors like the dendritic structure and complex channel kinetics, but where the level of detail makes it difficult to simulate very large networks efficiently [2, 3].

In this paper we describe SpikeNET, a neural network simulation package written in highly portable C++ code which lies between these two extremes. It is sufficiently biologically realistic to make it possible to examine the role of temporal properties such as synchronous or asynchronous spiking in neurons, and yet sufficiently simple to allow real-time simulation of large scale networks of neurons.

2. Basic Organization

The basic objects in SpikeNET are two dimensional arrays of relatively simple leaky integrate-and-fire neurons. Each unit is characterized by a small number of parameters : a membrane potential, a threshold, and (in some cases) a membrane time constant. When an afferent neuron fires, the weight of the synapse between the two neurons is added to the target neuron's potential, and we test to see whether the neuron's potential has exceeded the threshold. If so, the neuron is reset (by subtracting the threshold) and the neuron is added to the list of neurons that have fired in the current time

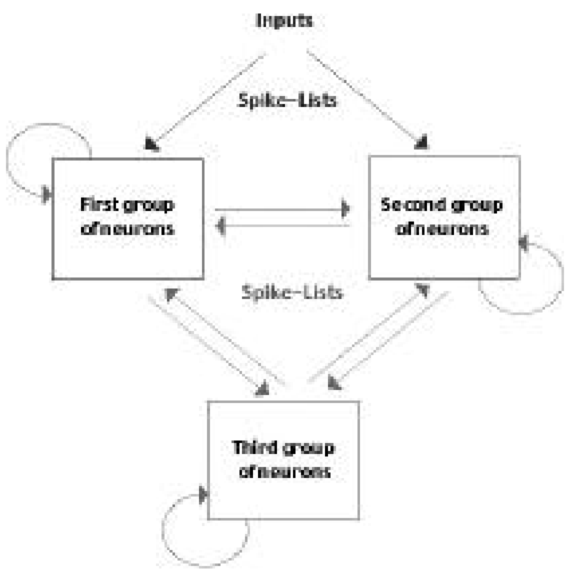

Figure 1: Basic organization of SpikeNET. SpikeNET redirects lists of spikes between different groups of neurons organized in two-dimensional arrays. Since only a small percentage of cells fire in each time-step, communication overheads are kept to a minimum.

step. Propagation of activity within SpikeNET involves sending lists of spikes between neuronal arrays as illustrated in figure 1. The event-driven nature of spike propagation is one of the reasons for the efficiency of SpikeNET as a modeling system.

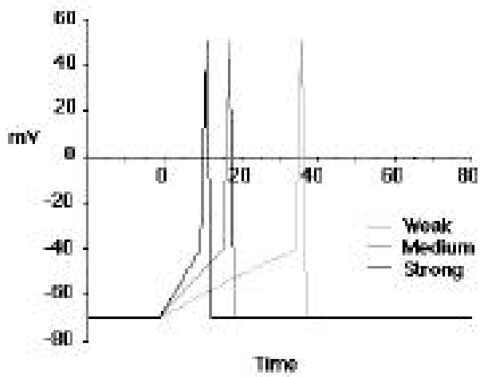

Figure 2: Basic behavior of an integrate and fire neuron. The latency of its discharge depends on the strength of the stimulation. With strong stimulation, the neuron will reach threshold quickly whereas with weak stimulation the latency will increase.

The basic cellular model can be made more complex by including a sensitivity parameter which modulates the effect of incoming action potentials. We have used this feature to implement a rank-order coding scheme which we have developed [6]. According to this scheme, the sensitivity parameter is initially fixed at 1.0, but decreases by a fixed percentage with each incoming impulse, resulting in a progressive desensitization of the post-synaptic neuron which can be thought of in terms of fast shunting inhibition [1]. The net result of this mechanism is that activation is maximal only when the spikes arrive in the order of the weights - with the highest weight synapses being activated first. If desired, this desensitization process can be made specific to particular sets of inputs such that, for example, inputs from the thalamic could mutually desensitize each other without affecting the efficacy of intra-cortical inputs to pyramidal cells. These more complex models for individual neurons are designed to mimic some of the effects of the dendritic structure of neurons while at the same time avoiding the computationally expensive detailed modeling that is normally required.

Most neurons are only affected by spikes in their afferent neurons. However, for certain "input" cells, corresponding for example to cells in the retina, we determine spike timing by a direct calculation that depends on the stimulus. Thus for retinal ganglion cells, we can perform a local "Mexican-hat" convolution on the image, and this value is used to calculate the latency of the unit's spike - the earliest latencies correspond to those cells for which the value of the convolution is highest, whereas lower activation levels result in progressively longer latencies (figure 2).

3. SpikeNET in Action

To illustrate how SpikeNET can be used, we will describe a multiscale face recognition network which extends the face-localization model described by Van Rullen et al [7], and uses an architecture loosely based on the organization of the primate visual system. Input images are first analyzed by arrays of ON-center and OFF-center cells in the "retina" at three different spatial scales. These cells send spikes to neurons in the next layer which contains neurons tuned for 8 different orientations at each spatial scale. Lateral interactions between cells in this layer were used to improve selectivity, and are similar to those described by Zhaoping Li [4]. A weak shunting inhibition was also included to make the neurons sensitive to the order of activation of their inputs. A third layer in the network contains neurons selective for faces at the three spatial scales. The connections between the level 2 orientation maps and these face-selective units were trained using a set of 200 photographs of faces and a supervised learning procedure which attaches high weights to inputs which are systematically among the first to fire, and progressively smaller weights to later firing inputs. Finally, a fourth layer of neurons contains neurons which integrate the information at the three different spatial scales in the previous layer.

As can be seen from Figure 3, the simulation is successful in that in the final map, the neurons fire if a face, at any scale, is present in the input image.

The model is clearly not very realistic. For example, no attempt was made to model change in resolution with retinal eccentricity, but the architecture illustrated here demonstrates how SpikeNET can be used to create quite complex multilayer architectures involving large numbers of units, and it shows how different hypotheses could be tested and integrated easily in a biologically plausible neural network.

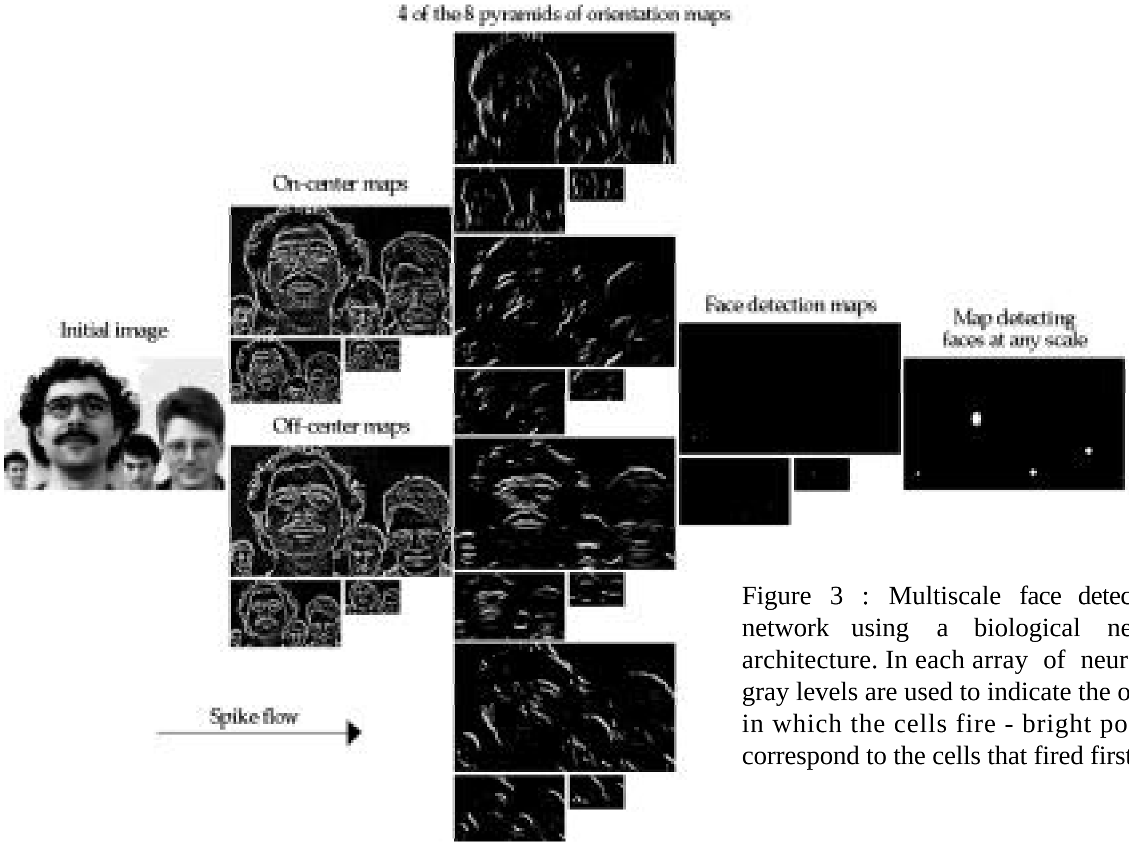

Figure 3 : Multiscale face detection network using a biological neural architecture. In each array of neurons, gray levels are used to indicate the order in which the cells fire - bright points correspond to the cells that fired first.

4. Performance of SpikeNET

SpikeNET has been designed to be computationally efficient. One of its advantages comes from the efficient use of RAM. Since the number of parameters per neuron is kept low, each neuron can require as little as 16 bytes of memory, depending on the type of precision required. More importantly, the use of shared weights means that one set of weights can be used for all the neurons in an array. As a result it is perfectly reasonable to simulate networks with tens of millions of neurons and billions of synapses on standard desktop computers.

The second advantage of SpikeNET is speed. Using a standard G3 Macintosh (PowerPC 750 processor at 266 Mhz), SpikeNET can update roughly 20 million connections per second, even when using the sensitivity parameter to modulate the effect of each synaptic input. This is sufficient to model a network of 400 000 neurons in real time, using a time step of 1 ms (assuming 49 connections per neuron, and an average firing rate of 1 spike per second, a value which is a reasonable estimate for the average firing rate of cortical neurons). Note that with a more conventional neural network simulation approach one has to recalculate every unit at every time step, and so the same computational power would only allow 20 000 connections to be calculated per millisecond, which with 49 connections per neuron would limit real-time simulation to around 400 neurons.

Performance is clearly optimal with shared weights, but even when each neuron has its own set of weights (which obviously increases RAM usage very considerably), speed only drops by a factor of around 2. Adding a decay to neurons to simulate the leaky nature of the synaptic integration process adds roughly 30-40% to the computation time (the exact value depends on the number of time steps in the simulation). Finally, note that increasing the time resolution from 1 ms to 0.1 ms has virtually no effect on computation time, since the number of spikes that are propagated does not change.

5. Parallel SpikeNET

Although running SpikeNET on a standard desktop machine is already reasonably quick, the very nature of SpikeNET makes it an ideal candidate for implementation on parallel hardware. The factor which usually prevents large scale use of parallel hardware in computing is the amount of communications needed between processors. For many problems, one sees little speed up once the computation has been split between more than 4 or 8 processors. However, with SpikeNET, the only information that needs to be transferred between processors are the Spike Lists. The format used by SpikeNET means that the identity of each neuron which fired can be transmitted using only around 1-2 bytes, and so even a network with 10 million neurons firing at an average of one spike per second could be simulated in real time without saturating the bandwidth of a cluster of processors linked by conventional fast Ethernet technology. We are currently developing multiprocessor PCI boards which will allow real time simulation of even larger networks of neurons.

6. Final Comments

Although primarily designed as a tool for modeling biological neural networks, the level of performance obtained with SpikeNET is such that in a variety of tasks, processing architectures developed using SpikeNET can perform at least as well and in many cases substantially better than more conventional image processing techniques. To the biologist, this may not be so surprising. We know that the processing strategies and architectures used in the human visual system (for example) are the end-product of hundreds of millions of years of intense natural selection. The levels of performance achieved by the human visual system are orders of magnitude better than even the most sophisticated artificial vision systems [5]. By elucidating the computational principles which make this level of performance possible, it may well be possible not only to demonstrate the power of computational neuroscience as a paradigm for understanding biology, but may reveal the potential of the discipline in areas as diverse as machine vision and artificial intelligence.

References

[1] L. J. Borg-Graham, C. F. Monier, Y. Frégnac, Visual input evokes transient and strong shunting inhibition in visual cortical neurons, Nature 393, (1998) 369-73.

[2] J. M. Bower, D. Beeman, The book of GENESIS: Exploring realistic neural models with the GEneral SImulation System. Second Edition Springer-Verlag, New York, (1998).

[3] M. L. Hines, N. T. Carnevale, The NEURON simulation environment, Neural Computation 9, (1997) 1179-1209.

[4] Z. Li, A neural model of contour integration in the primary visual cortex, Neural Computation 10, (1998) 903-40.

[5] S. Thorpe, D. Fize, C. Marlot, Speed of processing in the human visual system, Nature 381, (1996) 520-522.

[6] S. J. Thorpe, J. Gautrais, Rank Order Coding: A new coding scheme for rapid processing in neural networks, in: J. Bower, Ed, Computational Neuroscience: Trends in Research, (Plenum Press, New York, 1998) 113-118.

[7] R. Van Rullen, J. Gautrais, A. Delorme, S. J. Thorpe, Face detection using one spike per neurone, Biosystems (In press), (1998).