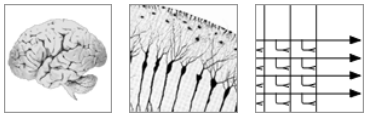
coneural
Center for Cognitive and Neural Studies



Biologically inspired neural networks for the control of embodied agents

Răzvan V. Florian

Center for Cognitive and Neural Studies (Coneural)
Str. Saturn 24, 400504 Cluj-Napoca, Romania
www.coneural.org
florian@coneural.org

Technical Report Coneural-03-03
Version 1.0
November 30, 2003

Abstract

This paper reviews models of neural networks suitable for the control of artificial intelligent agents interacting continuously with an environment. We first define the characteristics needed by those neural networks. We review several classes of neural models and compare them in respect of their suitability for embodied agent control. Among the classes of neural network models amenable to large scale computer simulation, recurrent spiking neural networks result to be the most suited for this task. We present next several models of spiking neural networks and review models of unsupervised learning for them. Finally, we review current implementations of spiking neural networks as controllers for embodied agents.

Contents

1 Introduction

There is an increasing awareness among the scientific community that genuine artificial intelligence (AI) can emerge only in embodied, situated agents (Steels & Brooks, 1995; Pfeifer & Scheier, 1999; Florian, 2003*a*). The sensorimotor interaction of an embodied agent with the environment may ground its intelligence, thus eliminating the fundamental problems of classical AI systems.

Artificial neural networks (ANNs) offer an appropriate framework for the development of controllers for artificial intelligent agents. They are parallel, loosely-coupled processes; their activity is decentralized and distributed. Thus, they are fault tolerant and robust with respect to noise. They may self-organize, thus being well adapted for learning and generalization, and potentially supporting emergent cognitive capabilities. Because of their many degrees of freedom, they incorporate a sufficient amount of redundancy for adapting to novel situations. Being biologically inspired, parallels may be drawn between the activity of some ANN models and the activity of biological neural systems.

This paper reviews the current state of the art in the domain of neural models suitable to be used for the control of artificial embodied agents. First, we will establish the characteristics needed for the neural networks that can be used for this purpose (embodied neural networks). We will review the different classes of neural models and compare their capabilities. We will present next in detail several models of spiking neural networks (SNNs) and review learning methods for these networks. We will also present the existing implementations of SNNs as controllers for embodied agents, as well as suggestions for future work.

2 The specific of neural networks controlling embodied agents

2.1 Characteristics

Input and output. Embodied autonomous agents perceive the surrounding environment through their sensors, and act on the environment through their effectors. Thus, the input of their neural control system is the activation of their sensory neurons, and the output of the neural network is the motor command that will be executed by the agent's effectors.

Temporal characteristics. In general, the input and output are continuously varying in time. Embodied agents are in a continuous interaction with the environment. Their body can both be influenced by the environment and act on it. Some actions of the agents can change the environment, thus

changing the influence of the environment over it, in a closed loop structural coupling.

There is thus a spatiotemporal structure in the input of the neural network, that should be detected and exploited by the control system of the agent. The network does not process static patterns, but a continuous informational flux, which depends on both the output of the network and on other factors from the environment.

Learning and memory. The control system of an intelligent agent should be capable of learning and memorizing. Neural networks can memorize both through plasticity (changes in the parameters of the neural units, e.g. weights or axonal delays), and through the dependence of their activation state on past states. Given the usual timescales of these changes of the state of the network, plastic memory usually persists on long term, while activation memory usually persists for relatively shorter durations.

The output of a feedforward neural network with continuous, spontaneous activation depends exclusively on the current input. The memory of a feedforward network with neurons whose activation is the result of the integration in time of the inputs has a temporal extent of the order of the integration time constant of the units. The most complex type of activation memory arises in recurrent neural networks, where the activity reverberates in loops and the current state of the network may depend significantly on past input. Recurrent neural networks are thus preferable for building control systems for embodied agents.

A genuinely intelligent agent should be adaptive, flexible, robust: it should adjust its operation to unexpected changes that influence it, and should be creative in finding solutions for completing its tasks, rather than conforming exclusively to preprogrammed behavior. Its categories and concepts should not be imposed by its designer, but should be developed by the agent itself, as a consequence of its interaction with the environment, its goals and its corporal capabilities. According to this perspective, the designer should not implement exclusively supervised learning schemes where the response of the agent in particular situations is predefined at the level of particular states of the output vector. The designer or the user of the agent may guide its behavior primarily through reinforcement learning, and occasionally by demonstration–imitation or through physical guidance in the environment. Self-organizing mechanisms should also be implemented.

Neural networks possess many known self-organizational capabilities for unsupervised learning, which will be discussed below for some specific types of networks. Reinforcement learning may also be implemented in neural networks, for example by feeding the reinforcement signal throughout the network and regulating learning mechanisms, such as local plasticity, through this neuromodulatory signal.

Biological resemblance. The resemblance of neural network models with their biological counterparts is highly desirable. Current artificial neural systems for the control of embodied agents do not reach the versatility and robustness of animal brains. Biological examples may suggest architectures and learning mechanisms for artificial models that would improve their performances. In the reverse direction, theories developed during the study of embodied ANNs may lead to new concepts and explanations regarding the activity of real neural networks (Ruppin, 2002).

2.2 Embodied neural networks vs. classical neural networks

Given the needed characteristics of embodied neural networks described above, they are quite different from classical ANNs described in popular introductory textbooks (e.g. Hertz et al., 1991; Haykin, 1998) or commonly used for engineering problems.

No static patterns. Many classical ANNs (e.g., Hopfield associative memory networks) are designed for processing static patterns. In contrast, embodied NNs process an input that is, in general, continuously changing in time, being static only in particular cases.

No time for convergence, no symmetric weights. Some classical ANNs function by convergence to an attractor. The result of the computation is given by the particular attractor to which the state converges, after an evolution in time. Given that the input of embodied network, in general, varies continuously, the embodied network does not have time to converge. Renouncing to this paradigm, we can also eliminate the artificial constraint of symmetric weights, needed in some classical models in order to the energy of the network to be defined and convergence to a point attractor to be guaranteed.

No predefined representations, no supervised learning, no backpropagation. Many ANNs output the result of their processing as a vector that is then interpreted by the human user, through a particular decoding. Sometimes, the input is also preprocessed and encoded by a human. Often, the goal is to have a particular predefined output for a specified range of inputs.

In contrast, embodied networks are fed the sensory state, and output the motor action of the agent. What is needed is a pertinent behavior of the agent, and not a predefined output for a particular input. Interesting environments are much too complex for a programmer to predefine the output for each possible situation. We are thus not interested in supervised learning procedures. In particular, we are not interested in backpropagation, which is a widely used supervised learning scheme.

No learning epochs. Classical learning methods for ANNs (such as back-propagation) need learning epochs where the parameters of the network are changed, separated from the actual functioning of the network for the task it was designed. Once the network was trained, its parameters are frozen during the performance. In contrast, embodied networks should be able to learn continuously during their interaction with the environment.

Recurrent, rather than feedforward architecture. Feedforward networks are predominant in classical textbooks and in scientific literature. However, recurrent neural networks (RNNs) are preferred for the control of embodied agents, for their capability of retaining short-term memory through their activation state.

Unfortunately, by eliminating these simplifications or features, we also have to renounce to many analytical tools and methods that gave the popularity of the classic NN models. The gain is a greater potential for emergent cognitive capabilities.

3 Biological neurons and classes of neural models

3.1 Real neurons

Biological neurons are a special type of cells, capable of generating and transmitting electrical signals. The cell body (or soma) of the neuron gives rise to two kinds of processes: several short dendrites and one long, tubular axon. Dendrites branch out like a tree and receive incoming signals from other neurons. The axon transmits to the network the signals generated by the neuron. There are about 10^{11} neurons in the human brain, each receiving about $10^3 - 10^5$ contacts from other neurons.

The electrical signals emitted by the neurons (called action potentials, or spikes) are rapid, transient, all-or-none (binary) impulses, with a duration of about 1 ms. They are initiated at a specialized region at the origin of the axon and propagate along the axon without distortion. Near its end, the tubular axon divides into branches that connect to other neurons through synapses. When the action potential emitted by a presynaptic neuron reaches the terminal of its axon, it triggers the emission of chemical transmitters in the synaptic cleft (the small space separating the two neurons at a synapse). These transmitters bind to receptors in the postsynaptic neuron, causing processes that depolarize or hyperpolarize its membrane (exciting or, respectively, inhibiting the neuron). These changes of the polarization of the membrane relative to the extracellular space spread passively from the synapses on the dendrites across the cell body. Their effects are integrated, and when there is a large enough depolarization at the origin of

the axon, a new action potential is generated (for details, see Kandel et al., 2000). An action potential is typically followed by a brief refractory period, where no further action potentials can be fired by the same neuron.

Real neurons are extremely complex biophysical and biochemical entities. In order to understand the activity of neurons and neural networks, analytical and computational models have to be developed. These models necessarily are a simplified approximation of their real counterparts. There are several classes of neural models, with various degrees of sophistication, which we will briefly present next.

3.2 Detailed neuron models

Detailed neuron models describe the activity of the neuron at the level of the various ionic currents flowing across the membrane, also considering the spatial circulation of the currents trough the cell.

The neural membrane may contain several types of sodium, potassium and calcium channels, that allow the passage of these ions. There is also a leakage current of chlorine ions. This flow of currents results in changes in the voltage across the membrane. The probabilities that a type of ionic channel is open depend nonlinearly on the voltage and the current state of the channel. All these dependencies result in a set of several coupled nonlinear differential equations, that describe the electrical activity of a part of the neural cell.

The Hodgkin and Huxley model is a classical such model, which they developed in 1952 as a result of their extensive experimental studies on the giant axon of the squid. A piece of cell membrane is considered to have a capacitance C. The ionic currents charge this capacitance:

$$C\frac{\mathrm{d}u}{\mathrm{d}t} = -\sum_k I_k + I(t) \tag{1}$$

where u is the voltage across the membrane, $I(t)$ is an external driving current and $\sum_k I_k$ is the sum of the ionic currents through the membrane. In this model, there are three types of ion current, indexed by Na, K, and L (leakage):

$$\sum_k I_k = g_{Na}\, m^3\, h\, (u - V_{Na}) + g_K\, n^4\, (u - V_K) + g_L\, (u - V_L) \tag{2}$$

The parameters g_k are conductances, V_k are reversal potentials, and m, n, h are additional variables that evolve according the the differential equations:

$$\begin{aligned} \dot{m} &= \alpha_m(u)\,(1-m) - \beta_m(u)\,m \\ \dot{n} &= \alpha_n(u)\,(1-n) - \beta_n(u)\,n \\ \dot{h} &= \alpha_h(u)\,(1-h) - \beta_h(u)\,h \end{aligned} \tag{3}$$

The α and β are empirical functions of u.

The spatial flow of the current inside the neuron may also be considered. This may be modelled by dividing the neuron in many uniform cylindrical compartments that approximate its shape (whence the name compartmental models). The cable equation may be used to describe the change in time of voltage and longitudinal current, which depends on the spatial gradient of these quantities (for details, see Koch, 1999; Koch & Segev, 1997).

These coupled differential equations may be integrated numerically, and the behavior of some real neurons can be recovered, including the emission of action potentials. Unfortunately, the numerical integration of these equations is computationally demanding, and is impractical to model more than several neurons at once at such level of detail. These neuronal models are thus currently not useful for creating large neural networks to be used as controllers for intelligent agents.

3.3 Formal spiking neuron models

Formal spiking neuron models preserve several important characteristics of real neurons: the emission of action potentials, and the leaky integration of inputs. They neglect the spatial structure of the neuron, but however they may approximate it to a certain degree. They also neglect the different nature of the different ionic currents, and may neglect nonlinearities in the integration of the input postsynaptic potentials. Thus, they loose some of the complexity and the computational power of real neurons (Koch & Segev, 2000). However, the models are computationally simple enough to be used in simulations of networks containing of the order of 10^4 such neurons.

Because of their special properties related to the processing of temporally varying input, networks of formal spiking neurons are extremely attractive for the modelling of control systems for embodied agents. For this reason, this class of networks will be described in detail later (Section 5).

3.4 Rate models

Rate models neglect the fact that neurons communicate through brief action potentials. In these highly simplified models, neurons are simple integrators with no spatial structure. Their resulting activity after integration is usually a real continuous value, which sometimes is said to represent the firing rate of real neurons. However, it is currently known that biological neural systems encode information not only in the firing rate, but also in the relative timing of the spikes.

In some models, the continuous dynamics may be also reduced to discrete updates of the state of the network.

The rate model closest to the biological inspiration is the one using continuous time and leaky temporal integration of the input. The equation

governing the neural activity is:

$$\tau_i \frac{\mathrm{d}u_i}{\mathrm{d}t} = -u_i + g(\sum_j w_{ij} u_j + I_i) \tag{4}$$

where $u_i(t)$ is the activation of neuron i (corresponding to the voltage across the membrane in real neurons), τ_i is its membrane time constant, $I_i(t)$ is an external input applied to the neuron (in the case of sensory neurons), and w_{ij} is a weight that characterizes the synaptic transmission of the activity of neuron j to neuron i. It may be positive or negative, corresponding to an excitatory or, respectively, inhibitory synapse. The sum runs over the neurons that connect to neuron i. The function $g(u)$ has a saturation nonlinearity; it is usually taken as a sigmoidal function with $g(u) \to 0$ for $u \to -\infty$ and $g(u) \to 1$ for $u \to \infty$. Commonly used functions are the logistic function,

$$\sigma(u) = \frac{1}{1 + e^{-u}} \ , \tag{5}$$

and the linear saturated function,

$$\pi(u) = \begin{cases} 0 \, , & \text{if } u < 0 \\ u \, , & \text{if } 0 \leqslant u \leqslant 1 \\ 1 \, , & \text{if } 1 < u \, . \end{cases} \tag{6}$$

A variation of this model is based on the following equation:

$$\tau_i \frac{\mathrm{d}u_i}{\mathrm{d}t} = -u_i + \sum_j w_{ij} \, g(u_j + \theta_i) + I_i \tag{7}$$

with $g(u)$ being the logistic function, and θ_i is a bias factor. Recurrent networks composed by such neurons are known as continuous time recurrent neural networks (CTRNNs).

The above models can be further simplified by eliminating the continuous variation of the activation in time, and by using discrete time. The equation governing such a model is:

$$u_i(t + \Delta t) = g(\sum_j w_{ij} \, u_j(t) + \theta_i) + I_i(t) \tag{8}$$

The updating may be synchronous (all neurons are updated in the same time), or asynchronous (in random order, each neuron being updated with equal probability).

3.5 The McCullogh-Pitts model

If the continuous sigmoid function is replaced in Eq. 8 by the unit step function (or Heaviside function) Θ,

$$\Theta(u) = \begin{cases} 1 \, , & \text{if } u \geqslant 0 \\ 0 \, , & \text{otherwise} \end{cases} \tag{9}$$

we get the McCullogh-Pitts neuron model (McCullogh & Pitts, 1943), also known as the threshold gate model. The activity is now a binary value that is supposed to represent the all-or-none nature of the action potential. However, this interpretation may have a certain validity only if the neural spikes would be synchronized. This may happen in real brains, but is not their typical mode of operation. This interpretation also neglects the importance of the relative timing of neural spikes, which was shown to carry information in some biological neural systems.

Many variants of the models presented above also exist.

4 Comparison of classes of neural networks

4.1 Computational capabilities in a classical context

First generation of NNs (threshold gates). Historically, networks consisting of McCullogh-Pitts neurons were the first to be studied analytically and computationally. Although simple, these networks are already computationally powerful devices. A synchronous feedforward network with a hidden layer can compute any Boolean function (McCullogh & Pitts, 1943). Recurrent threshold gates networks are analogous to finite automata, and, in general, they can perform universal computation (Minsky, 1967). Fully connected networks of this type may function as an autoassociative memory (the Hopfield model or the Boltzmann machine). Feedforward networks, known as perceptrons, were also used for tasks like heteroassociation and classification (see Hertz et al., 1991, for an introductory review).

Second generation (analog networks). The second generation of neural networks consisted of neurons with continuous (analog) activation function (Eq. 4,7,8). They are computationally more powerful than McCullogh-Pitts neural networks, being able to compute classes of Boolean functions with fewer neurons (DasGupta & Schnitger, 1996; Maass et al., 1994). Moreover, they are universal for analog computation, a network with a single hidden layer being able to approximate arbitrarily well any continuous function with compact domain and range. The differentiability of the activation function of the neurons allows the application of powerful supervised algorithms, like the backpropagation algorithm, for feedforward networks, and also for some types of recurrent networks (Pearlmutter, 2001).

The computational capabilities of analog discrete time recurrent NNs were reviewed by Siegelmann (1993) and Orponen (2000). Finite analog RNNs with rational weights employing a saturated linear activation function can simulate arbitrary Turing machines step by step. If the weights are real, these networks may have, in theory, super-Turing computational capabilities (Siegelmann & Sontag, 1995).

Continuous time recurrent NNs are universal dynamics approximators, in the sense that, for any finite interval of time, they can approximate the trajectories of any smooth dynamical system on a compact subset of $\mathbb{R}^n$ arbitrarily well (Funahashi & Nakamura, 1993). A systematic study of the dynamics of small CTRNNs was performed by Beer (1995). CTRNNs can also simulate discrete RNNs (Orponen, 1999; Sima & Orponen, 2000).

Third generation (spiking networks). Networks of spiking neurons are, with regard to the number of neurons that are needed, computationally more powerful than both McCullogh-Pitts and continuous activation function neural networks. There are example of functions that can be computed by a single spiking neuron, but which require hundreds of hidden units on a sigmoidal neural net. However, any function that can be computed by a small sigmoidal NN can also be computed by a small spiking NN (Maas, 1997*a*). Even very noisy spiking neural networks can perform digital computations with any desired degree of reliability (Maas, 1996). Noisy SNNs can also simulate sigmoid neural networks and have a larger computational power for the same number of units, using temporal coding of information (Maas, 1997*b*).

Supervised learning algorithms were also devised for spiking neural networks (Bohte et al., 2002; Bohte, 2003; Moore, 2002). They can also implement associative memory (Sommer & Wennekers, 2000; Maass & Natschlaeger, 1998).

Conclusion. In conclusion, spiking neural networks are computationally more powerful than sigmoidal (analog) neural networks, which are also computationally more powerful than McCullogh-Pitts neural networks. A more detailed review of the computational capabilities of the different classes of neural networks can be find in a paper by Sima & Orponen (2003).

4.2 Discussion: computational paradigms

The results presented above, regarding the computational capabilities of different network types, are established mostly in the framework of a classical paradigm of digital computation. For this, when networks having continuous analog internal states were used (e.g., sigmoidal or spiking networks), their inputs or outputs were digitized. This paradigm does not fit well with the needed characteristics of embodied networks, which have to process a continuous analog input in real time (Section 2.1).

Another computational paradigm is based on the continuous dynamics of an analog dynamical system, usually converging to attractors. Continuous time Hopfield nets are a particular instance of this type of systems. Few theoretical results concerning the general computational capabilities of such

systems are available (Gori & Meer, 2002; Siegelmann & Fishman, 1998; Orponen, 1997). They may be more efficient than Turing machines for solving some types of problems (Siegelmann et al., 1999). However, they have a finite (possibly long) delay between the input of data and the output of the results of the computation, needed for the systems to converge to attractors. This does not correspond to the needed characteristics of embodied control systems.

An alternative computational paradigm that satisfies the need for real time computation, in the condition of a continuous input-output flow, is the liquid state machine, presented next.

4.3 Liquid state machine computation

The liquid state machine (LSM) was recently proposed as a new, biologically plausible, computational paradigm (Maass et al., 2002). A LSM processes a continuous analog input stream, and its output is also a continuous analog input stream. The processing is performed in real time, there is no computation delay other than the delay given by the propagation of the signal through the system. Thus, this kind of system is suitable for the control of embodied agents.

The LSM is composed of a nonlinear filter L (a function that maps time-varying input streams $u(\cdot)$ to other time-varying streams $x(\cdot)$, also known as operator in mathematics), and a memoryless readout function f that maps the output $x(t)$ of the filter to a target output $y(t)$. The filter L has a fading memory: $x(t)$ may depend not only on $u(t)$, but also, in a quite arbitrary nonlinear fashion, on previous inputs $u(t - \tau)$. The LSM has universal computational power for time series, if two simple conditions are fulfilled. The first condition is that the filter L separates output-relevant differences in two inputs $u_1(t)$ and $u_2(t)$ to different states $x_1(t)$ and $x_2(t)$ (the pointwise separation property). The second condition is that the readout map is able to distinguish and transform different states $x(t)$ into given target outputs $y(t)$ (the approximation property).

The initial implementation of the LSM (Maass et al., 2002) used a recurrent spiking neural network as the nonlinear filter L, and a small population of spiking neurons, trained with a perceptron-like local learning rule, as a readout map. However, the nonlinear filter may be implemented also by sigmoid neural networks, either in continuous or discrete time. Many other types of nonlinear filters are possible. Recent implementations included a finite element simulated model of a liquid (Goldenholz, 2003) or even real water in a bucket (Fernando & Sojakka, 2003) for the nonlinear filter, with classic perceptrons for the readout map. The filter transforms the temporal dynamics of the input stream into a high dimensional spatial pattern. The readout neurons, which may receive inputs from hundreds or thousands neurons from the neural filter, can learn to extract salient information from the

high dimensional transient states of the filter, and can transform them to stable readouts. Invariant readout is possible despite the fact that the neural filter may never revisit the same state. The readout map may be linear, and thus extremely easy to train with classic supervised learning methods.

The LSM computational paradigm has certain properties that make it extremely interesting. Multiple readout modules can be trained to perform different tasks on the same neural filter, thus enabling parallel real-time computing. The filter may be a generic circuit, to be used for many tasks, only the readout map needs to be trained for each task. Computational power may be added to the LSM by just adding generic neural circuits to the neural filter, without changing the existing wiring. The system is extremely robust to noise.

4.4 Performance in embodied agent control

As previously discussed, we are interested in recurrent NN architectures. Recurrent discrete and continuous time sigmoid networks, and also spiking networks were successfully used for the control of embodied agents. Discrete time RNNs are usually trained by supervised learning (e.g. Tani, 2002), or by evolutionary methods (e.g. Floreano & Mondada, 1996). CTRNNs developed through evolutionary algorithms were used for the control of agents interacting with simple environments (e.g. Ijspeert, 2001; Slocum et al., 2000; Yamauchi & Beer, 1994). Embodied controllers consisting of spiking networks started to be used only recently; they are usually evolved (Di Paolo, 2003, 2002*a*,*b*; French & Damper, 2002; Floreano et al., 2002*a*,*b*; Floreano & Mattiussi, 2001); see Section 7 for more details.

Very few studies attempted to compare directly the performance of different classes of neural networks in the same context. Blynel & Floreano (2002) compared evolved CTRNNs (without epigenetic plasticity) and plastic neural networks (PNNs, discrete time sigmoidal networks without unit memory, but with plasticity) in two behavioral tasks aimed at exploring their capabilities to display reinforcement learning-like behaviors, and adaptation to unpredictable environmental changes. In this study, the networks had similar performances on both tasks, but PNNs displayed significantly better performance when sensorimotor readaptation is required after the evolutionary process. It is possible, however, that CTRNNs with plasticity would outperform PNNs.

Two studies (Floreano & Mattiussi, 2001; Di Paolo, 2002*a*,*b*) compared evolved CTRNNs to spiking neural networks. In both these studies SNNs were shown to be superior to CTRNNs. In both studies, the fitness of the evolved individuals based on sigmoid networks were systematically lower than the fitness of those based on spiking nets, if both types of networks were evolved in the same conditions.

4.5 Biological plausibility

We have seen in the presentation of the classes of neural networks that the spiking NNs are the class of ANNs suitable for large scale experiments that is the most similar to the biological reality. Spiking networks allow the implementation of interesting phenomena (spike timing dependent plasticity, temporal coding) that are present in biological neural systems and cannot be implemented with sigmoidal networks or threshold gates.

4.6 Conclusion

From the above comparison, spiking neural networks appear to be the most powerful class of neural networks suitable for embodied agent control. They are computationally more powerful than other classes even in a classical digital context, they can support liquid state machine-type computation, they compare better in applications, and they are the closest to the biological reality. Their simulation on common personal computers is more costly than that of sigmoidal or threshold networks, but dedicated hardware may be built which can simulate spiking networks extremely fast. CTRNNs seem to be the second best choice.

Thus, the advantage of using SNNs for the control of embodied agents is not only their biological plausibility, which may allow a bidirectional transfer of concepts and methodologies between neuroscience and embodied cognitive science, but also their computational power. In the following, we will present in detail several models of SNNs.

5 Spiking neural models

5.1 Leaky integrate-and-fire neurons

The leaky integrate-and-fire model is the most common formal model of a spiking neuron, being the easiest to simulate computationally. The presentation below follows Chapter 4 of the review of Gerstner & Kistler (2002).

A neuron is modelled as a leaky capacitor, i.e. a capacitor C in parallel with a resistor R. This is suggested by the electrical characteristics of a piece of membrane of a real neuron; however, the spatial structure of a real neuron is neglected. The model neuron is driven by a current $I(t)$, which may be synaptically transmitted by other neurons from the network or an externally injected current. The conservation of charge requires that $I = I_R + I_C$, where I_C is the current charging the capacitor, and I_R is the leaking current through the resistor. We have $I_R = u/R$ from Ohm's law, where u is the voltage across the resistor. From the definition of capacity, $C = q/u$, where q is the charge on the capacitor, and thus $I_C = \mathrm{d}q/\mathrm{d}t = C\,\mathrm{d}u/\mathrm{d}t$. Thus, we

have:

$$I = \frac{u}{R} + C\frac{\mathrm{d}u}{\mathrm{d}t}. \tag{10}$$

Defining the time constant of the neuron $\tau_m = RC$, we get the standard equation that defines the dynamics of the leaky integrator:

$$\tau_m \frac{\mathrm{d}u}{\mathrm{d}t} = -u(t) + RI(t). \tag{11}$$

This equation describes the dynamics of the membrane potential u only between the spikes. As this model neglects the characteristics of the different ionic channels that pump the ions through the membrane, this simple equation does not account by itself for the emission of spikes. Spikes are modelled phenomenologically considering that, when the membrane potential u becomes greater than a threshold θ, a spike is emitted and u is immediately reset to a base value $u_r < \theta$.

In a more detailed version, an absolute refractory period may be implemented, considering that, after a spike is emitted at $t^{(f)}$, u stays at the base value u_r for an absolute refractory time Δ^{abs}, and the integration restarts at $t^{(f)} + \Delta^{abs}$.

The state of a neuron is defined by its membrane potential u. The spikes are identical, stereotyped events, and thus the activity of a neuron i is fully characterized by the set of firing times $\mathcal{F}_i = \{t_i^{(1)}, \ldots, t_i^{(n)}\}$.

If a presynaptic neuron j fires a spike at $t_j^{(f)}$, a current pulse $\alpha(t - t_j^{(f)})$ will be transmitted to postsynaptic neurons. If the efficacy of the synapse between presynaptic neuron j and postsynaptic neuron i is w_{ij}, the total input current to neuron i is

$$I_i(t) = \sum_j w_{ij} \sum_f \alpha(t - t_j^{(f)}) + I_i^{ext}(t), \tag{12}$$

where $I_i^{ext}(t)$ is an external input, for example a sensory activation for sensory neurons.

The simplest choice for the time course of the postsynaptic current α is a Dirac pulse, $\alpha(s) = q\,\delta(s)$. More realistically, the postsynaptic current should have a finite duration. An example is an exponential decay with time constant τ_s,

$$\alpha(s) = \frac{q}{\tau_s}\, e^{-\frac{s}{\tau_s}}\, \Theta(s), \tag{13}$$

where Θ is the Heaviside step function (Equation 9). A more detailed version may include a finite rise time τ_r of the postsynaptic current and an axonal transmission delay Δ^{ax}:

$$\alpha(s) = \frac{q}{\tau_s - \tau_r} \left(e^{-\frac{s-\Delta^{ax}}{\tau_s}} - e^{-\frac{s-\Delta^{ax}}{\tau_r}} \right) \Theta(s - \Delta^{ax}). \tag{14}$$

In the limit of $\tau_r \to \tau_s$, this transforms to

$$\alpha(s) = q\frac{s-\Delta^{ax}}{\tau_s^2}\, e^{-\frac{s-\Delta^{ax}}{\tau_s}}\, \Theta(s-\Delta^{ax}). \tag{15}$$

The dynamics of the leaky integrate-and-fire neuron may be illustrated for the case of a constant input current $I(t) = I_0$. For simplicity, we take $u_r = 0$ in this example. If we assume that a spike was fired at t_1, the following trajectory of the membrane potential can be found by integrating (11):

$$u(t) = RI_0\left(1 - e^{-\frac{t-t_1}{\tau_m}}\right). \tag{16}$$

For $RI_0 < \theta$ the membrane potential approaches asymptotically to $u(\infty) = RI_0$ and no further spike can occur. For $RI_0 > \theta$, the membrane potential may reach the threshold θ at time t_2, which is given by

$$\theta = RI_0\left(1 - e^{-\frac{t_2-t_1}{\tau_m}}\right). \tag{17}$$

and thus we found the time interval $T = t_2 - t_1$ as

$$T = \tau_m \ln\frac{RI_0}{RI_0 - \theta}. \tag{18}$$

After the emission of a spike at t_2, the membrane potential is reset again at u_r and we have the same state as at t_1. Thus, the neuron will spike regularly with period T, given a constant input current.

5.2 The Izhikevich simple model (ISM)

The integrate-and-fire model described above captures the essence of the spiking behavior and is simple to simulate computationally. However, its behavior (periodic spiking in the presence of a constant input current) does not reflect the richness of the dynamics of biological neurons. More biologically plausible behavior may be displayed by more complex models, like the Hodgkin-Huxley model (Section 3.2), but they are usually computationally more demanding to simulate. A recent model by Izhikevich (2003*a*) succeeds at displaying rich dynamics (bursting, chattering, adaptation, resonance; see Figure 5.2), while also having a moderate computational complexity. A comparison with ten other models (Izhikevich, 2003*b*) shows that, among the compared models, the ISM offers the best biological plausibility over computational simulation complexity ratio.

The model is based on a two-dimensional system of ordinary differential equations of the form

$$\begin{aligned} \frac{\mathrm{d}u}{\mathrm{d}t} &= m\,u^2 + n\,u + p - v + I(t) \\ \frac{\mathrm{d}v}{\mathrm{d}t} &= a(b\,u - v). \end{aligned} \tag{19}$$

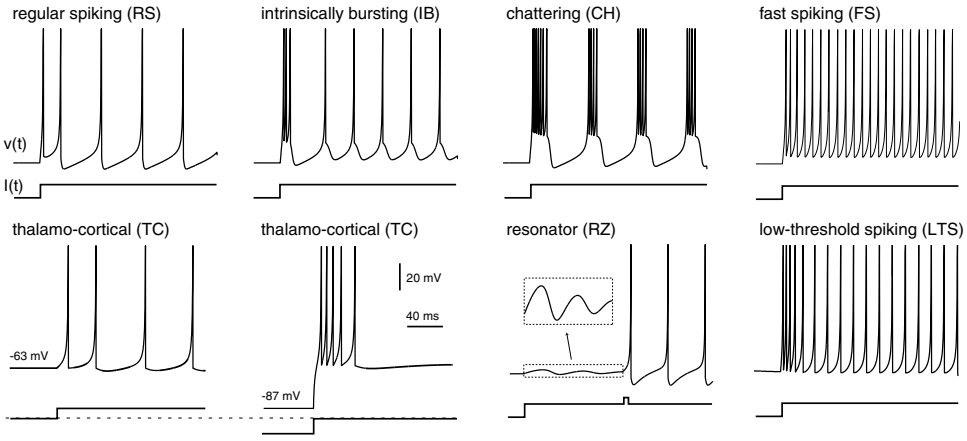


Figure 1: Types of dynamics displayed by the Izhikevich simple model. Electronic version of the figure and reproduction permissions are freely available at www.izhikevich.com.

The variable u represents the membrane potential of the neuron. The membrane recovery variable v accounts for the activation of K^+ ionic currents and the inactivation of Na^+ currents, and provides negative feedback to u. The parameters m, n, p are constants defining the model, a, b are constant parameters determining the intrinsic firing pattern of the neuron, and $I(t)$ is the external current driving the neuron. A spike is emitted if u reaches the threshold θ. Immediately after a spike, the variables are reset to

$$\begin{aligned} u &\leftarrow c \\ v &\leftarrow v + d. \end{aligned} \tag{20}$$

Again, c and d are constant parameters.

By fitting the spike initiation dynamics of a cortical neuron so that u has mV scale and the time has ms scale, Izhikevich has obtained the values $m = 0.04$, $n = 5$, $p = 140$. Typical values for the other parameters are $a = 0.02$, $b = 0.2$, $c = -65$, $d = 2$. As a function of the $a..d$ parameters, the neuron can display a large variety of behaviors that are also displayed by biological neurons (regular spiking, intrinsically bursting, chattering, fast spiking, low-threshold spiking, thalamo-cortical, resonator; see Figure 5.2). The postsynaptic current following a spike emitted by a presynaptic neuron can be modelled as for the integrate-and-fire model, described in the previous section. Networks having 10^4 such neurons and 10^6 synapses can be simulated in real time, with 1 ms resolution, on a regular PC. For details, please see the original paper (Izhikevich, 2003*a*).

5.3 The spike response model (SRM)

The spike response model (Gerstner & Kistler, 2002; Gerstner, 1999) is a generalization of the leaky integrate-and-fire model, in which the variables that characterize the dynamics of neuron i depend on the time since its last output spike, $\hat{t}_i$. After firing, the evolution of the membrane potential u_i is given by

$$\begin{aligned} u_i(t) &= \eta(t-\hat{t}_i) + \sum_j w_{ij} \sum_f \epsilon_{ij}(t-\hat{t}_i, t-t_j^{(f)}) \\ &\quad + \int_0^\infty \kappa(t-\hat{t}_i, s) I^{ext}(t-s)\mathrm{d}s. \end{aligned} \tag{21}$$

The function η describes the refractory period that follows the spike. The kernel $\epsilon_{ij}(t-\hat{t}_i, t-t_j^{(f)})$ models the time course of a postsynaptic potential evoked in the postsynaptic neuron i by the emission of a spike at $t_j^{(f)}$ by presynaptic neuron j. The synapse between i and j is considered to have an efficacy w_{ij}. The kernel $\kappa(t-\hat{t}_i, s)$ is the linear response of the membrane potential to an input current I^{ext}.

As in previous models, the neuron fires when the membrane potential u_i crosses a threshold θ_i from below.

It can be shown that, for a particular form of the kernels η, ϵ and κ, this model is equivalent to the leaky integrate-and-fire model. For other forms of the kernels, this model can approximate with a high degree of accuracy (up to 90 %) the detailed Hodgkin-Huxley model (Chapter 4, Gerstner & Kistler, 2002). Thus, this model offers another simple possibility to account for some of the complex dynamics of biological neurons, while also being convenient for computational simulation.

A simpler version (SRM_0) of this model can be obtained by neglecting the dependence of kernels ϵ and κ on the time from the last spike emitted by the neuron, $t-\hat{t}_i$.

5.4 Modelling noise

Biological neurons are noisy, due to thermal effects and to the finite numbers of transmitter molecules and ion channels. Noise can cause failures in synaptic transmission and different responses of a neuron to the same input current. If biological plausibility is a concern, noise should thus also be implemented in neural models. However, there might be also other reasons for introducing noise. If we would like our controller to drive a robot, whose sensors necessarily have noisy input, or to implement the controller in analog hardware that is subject to noise, we would like to design the controller to function robustly even under noise. Noise can also be a source of behavioral variability, and can drive the controller out of locked oscillations. More-

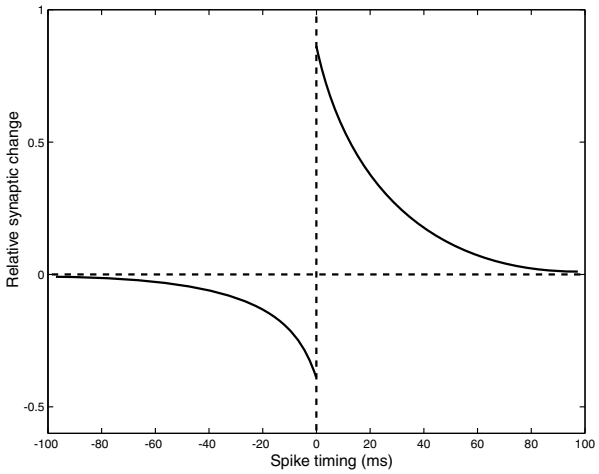


Figure 2: Typical example of spike-timing dependent plasticity. The relative change in synaptic efficacy $\Delta w/w$ is plotted against the delay Δt between the postsynaptic and presynaptic spike. Synaptic potentiation ($\Delta w/w > 0$) appears when the postsynaptic spike follows (is caused by) the presynaptic spike ($\Delta t > 0$) (after Bi, 2002).

over, noise can, under certain circumstances, improve signal transmission properties of neural systems, through stochastic resonance.

There are several ways to introduce noise in threshold-fire neural models, such as the models presented above.

Noisy threshold. We may consider that the neuron can fire stochastically even though the threshold θ was not yet reached, with a probability that depends on the difference between the membrane potential and the threshold.

Noisy reset. In this model of noise, the duration of the refractory period following a spike may vary stochastically.

Noisy integration. We may also consider that a small stochastic current is injected in the neuron, thus adding noise to the trajectory of the membrane potential.

6 Learning in spiking neural networks

As discussed above (Section 2.1), embodied neural systems should be capable of learning and memorizing. Learning should be based mostly on unsupervised and reinforcement learning mechanisms, and not on supervised

learning.

An important biologically inspired unsupervised learning mechanism is hebbian learning, where changes in a synapse depend on the activity of the neurons joined by the synapse. In classical sigmoid networks, models of hebbian learning depend on the activation rates of the neurons. In spiking neural networks, the precise firing time of the spikes can offer useful information, available locally at the synapse, that should be exploited by the learning mechanism.

Actually, spike timing dependent plasticity (STDP) of the synapse strengths is a phenomenon that was experimentally observed in biological neural systems (Bi, 2002). The changes Δw of the synapse efficacies depend on the relative timing Δt between the postsynaptic and presynaptic spikes (see Fig. 6). The synapse is strengthened if the postsynaptic spike occurs shortly after the presynaptic neuron fires, and is weakened if the sequence of spikes is reversed, thus enforcing causality. Notably, the direction of the change depends critically, on a millisecond time scale, on the relative timing.

This can be modelled phenomenologically by the following equation:

$$\begin{aligned}\frac{\mathrm{d}w_{ij}}{\mathrm{d}t} = a_0 \quad & +S_i(t)\left[a_1^{post} + \int_0^\infty a_2^{post,pre}(s)\, S_j(t-s)\mathrm{d}s\right] \\ & +S_j(t)\left[a_1^{pre} + \int_0^\infty a_2^{pre,post}(s)\, S_i(t-s)\mathrm{d}s\right]. \end{aligned} \tag{22}$$

Here $S_j(t) = \sum_f \delta(t - t_j^{(f)})$ and $S_i(t) = \sum_f \delta(t - t_i^{(f)})$ are, respectively, the pre- and postsynaptic spike trains. The term a_1^{post} reflects a change in the synaptic efficacy triggered by a postsynaptic spike, even without additional presynaptic action potentials. The kernel $a_2^{post,pre}(s)$ gives the weight change if a presynaptic action potential is followed by postsynaptic spike arrival with delay s. The terms a_1^{pre} and $a_2^{pre,post}(s)$ are analogous. Together, the two a_2 terms define a "learning window" such as the one from Fig. 6. A possible analytical form for these terms is

$$a_2^{post,pre}(s) = w_{max}\, A^+\, e^{-\frac{s}{\tau^+}}, \quad \text{with } s > 0, \text{ and} \tag{23}$$

$$a_2^{pre,post}(s) = -w_{max}\, A^-\, e^{\frac{s}{\tau^-}}, \quad \text{with } s < 0. \tag{24}$$

The term a_0 reflects a activity-independent weight decay or consolidation.

An agent must be able to learn and adapt continuously to its environment. However, it should also have a memory, where a trace of important past events should subsist. There is thus a conflict between the consolidation of memory and the continuous synaptic change determined by continuous neural activity (the stability-plasticity dilemma). A possible solution is the bistability of synaptic weights: in the absence of stimulation, weights should converge to two stable fixed points (0 and $\pm w_{max}$). Synaptic activity may lead to a transition from one stable point to the other, but only if the duration and amplitude of this activity exceeds a certain learning threshold.

Learning thus affects only a few synapses at a time so that previously stored information is mostly preserved. Such a dynamics can be implemented by setting

$$a_0(w_{ij}) = -\gamma\, w_{ij}\, (w_{max} - |w_{ij}|)(w_\theta - |w_{ij}|), \tag{25}$$

where $0 < w_\theta < w_{max}$ and $\gamma > 0$. In the absence of stimulation, small weights ($|w_{ij}| < w_\theta$) decay to 0 and large weights ($|w_{ij}| > w_\theta$) increase in absolute value towards w_{max}. Synaptic activity has thus to drive the absolute efficacy across the threshold w_θ before long term changes take place (Gerstner & Kistler, 2002, Section 10.3.2).

Spike timing dependent plasticity has several advantages over classical hebbian models of firing rate dependent plasticity. STDP is sensitive to both spatial and, importantly for embodied agents, temporal correlations in the neural activity. An asymmetric spike time dependent learning rule is capable of detecting early events that may serve as predictors for others. Considering a particular neuron, synapses from neurons that are activated before a postsynaptic spike are strengthened, while other synapses are depressed. In subsequent trials in the same environmental conditions, the firing threshold of the neuron is reached earlier. After many trials, those presynaptic neurons that fire first have developed strong connections with the neuron, while other connections are depressed. Thus a temporally asymmetric learning rule favors connections that can serve as earliest predictors of spike events. An asymmetric learning window can also selectively strengthen synapses that deliver precisely timed spikes at the expense of others that deliver spikes with a broad temporal jitter (Gerstner & Kistler, 2002, Chapter 12).

Combined with short-term depression caused by presynaptic spikes, STDP can redistribute synaptic plasticity, such that a synapse responds more strongly to the first presynaptic event in a spike train than to subsequent events, as also observed experimentally in biological systems (Shon & Rao, 2003). Also, single neurons are able to learn to predict temporal patterns of presynaptic activity (Rao & Sejnowski, 2003). Temporal asymmetric plasticity rules were also shown theoretically to maximize the mutual information between the output and the input of the neurons (Chechik, 2003).

7 Examples of spiking embodied agent controllers

There currently exist few implementations of embodied agent controllers based on spiking neural networks. A reason may be the current relative lack of analytical methods for developing such controllers and for mastering the complexity of their behavior. In this situation, the solution used by the existing implementations is to use artificial evolution in order to develop such controllers. Evolution may lead to the discovery of controllers that implement the desired behavioral abilities of the agents, without the

need of imposing predefined constraints on their architecture and functioning modality.

In interfacing a spiking neural network with an embodied agent, a conversion of the analog input and output signal to binary spikes has to be performed. There exist several methods for this. The first alternative is to map the analog signal to the firing rate of a neuron. This is inspired by classical experiments that observed such correlations in some biological sensory neurons. A second alternative is to map the analog value to the proportion of firing neurons of a neural population (population coding). There are classical experimental results that have observed such correlations in motor cortices, and also in some visual processing neural systems. A third alternative is to code the value in the firing delay of the neuron.

Floreano & Mattiussi (2001) have evolved vision based controllers for a Khepera robot, for a navigation task in a rectangular arena. The walls of the area were textured with random black and white vertical stripes. The controller of the robot consisted of a SNN with 10 fully-connected neurons, each connected to the 18 sensory receptors (16 visual, and 2 proprioceptive), and 4 of them being motor neurons that drove the wheels of the robot. Noise was added to the neural activity. In the absence of noise, it was observed that the network would go quickly into locked oscillations for a very large range of connectivity patterns and parameters. The chosen fitness function selected the ability of the robot to go as fast as possible while avoiding the walls.

This experiment demonstrated that evolution can easily discover functional networks that accomplish the desired behavior, despite the complex nonlinear dynamics of SNNs. It was observed that the functional importance of the neurons in the network varies, and that the controller is robust to synaptic decay. The presence of coding strategies based on coincidence detection was not clearly observed. This may be the result of the low membrane threshold and the interfacing mechanism between the controller and the sensorimotor system used in the experiment. The input and output of the controller were updated once every 100 time steps, potentially biasing the evolution of a firing rate-based coding strategy.

The experiment also compared the SNN controller with a controller based on sigmoidal RNNs. It was observed that it is harder to evolve, in the same conditions, a network of sigmoid networks with equal fitness.

Similar experiments were performed using an Alice microrobot driven by an evolved SNN controller implemented in onboard hardware (Floreano et al., 2002*a*) and using flying robots (blimps) (Floreano et al., 2002*b*).

French & Damper (2002) have evolved a SNN controller of 6-8 neurons that permitted a Khepera robot to perform phototaxis, distinguishing between lamps of different flashing frequencies. No systematic analysis was performed on the evolved networks.

Di Paolo (2002*a*,*b*) has also evolved SNN controllers for embodied agents.

For the first time, these evolved controllers also benefited of spike timing dependent plasticity. Additionally, an activity dependent synaptic scaling (ADS) mechanism was implemented to insure the homeostatic regulation of the firing rate of the neurons. If the postsynaptic activity is above a certain target, excitatory synapses are scaled down; otherwise, they are scaled up. Directional damping is implemented to keep the synaptic weights in a limited range. If a weight value is near the boundary, changes that push this value towards the boundary are slowed down, but changes that push it away are not. The equilibrium weight distribution in this case tends to be unimodal and centred around the point where potentiation and depression equilibrate.

The network consisted of 6 neurons and controlled a simulated circular agent that was able to perform phototaxis in a 2D environment. It was observed that plastic networks (with STPD and ADS) were easier to evolve than non-plastic ones. If low neural noise was added to the network, the evolved coding strategies depended on the precise firing time of the neurons. This was tested with a randomisation of the spikes of evolved networks, which kept constant the firing rates. Networks evolved under moderate noise developed coding strategies that are robust to spike randomisation, which implies that they did not use precise firing times, although the plasticity rules still used precise firing times. Also, these later networks used functionally the random noise, the removal of the noise leading to decreased fitness.

A comparison with rate-based CTRNNs evolved in the same conditions was performed, and it was found that their performance was significantly lower for the studied task.

8 Discussion and directions for future work

Existing implementations of recurrent SNN for the control of embodied artificial agents implied the evolution of very small networks (comprising no more than 10 neurons). Only one implementation included plasticity, with promising results. Such small networks cannot achieve the range of complex behaviors exhibited by biological networks. Future work should thus consider the evolution of larger networks, using fast simulators for implementing the networks (e.g., Marian et al., 2002; Mureşan, 2003) and the agent and the environment (e.g., Florian, 2003*b*).

The evolution of intelligent controllers is limited by the computational power of current computers, given that the search space is extremely large (Grand, 1998). An alternative is to design developmental models that will allow the controller to integrate the structure of the environment by interacting with it. This is an important area for future research, which should probably draw upon biologically inspired models of plasticity, such as the STDP presented above. A complementary research area is the integration of reinforcement learning mechanisms with unsupervised learning, probably

by feeding diffusely a neuromodulatory reinforcement signal throughout the network.

Another important research direction for future work is the development of methods for the analysis of the complex activity of recurrent SNNs. The liquid state machine paradigm, presented above, allows the probing of the network activity, informing us about the information present in the network. Other suitable analytical tools, coming from information theory or from the statistical mechanics of non-equilibrium systems should also be developed.

9 Conclusion

Among the classes of neural network models amenable to large scale computer simulation, recurrent spiking neural networks are the most suitable for implementing control systems for embodied artificial intelligent agents. They allow the extraction of temporal structure from the sensorimotor data flow, an important feature for agents interacting continuously with the environment. We have reviewed several models of spiking neural networks, and the few existing implementations of SNNs as controllers for embodied agents. There is much work to be done in this research area, but the importance of this research, potentially leading to epigenetically developed, embodied and grounded artificial intelligence, fully motivates this work.

10 Acknowledgements

This work was supported by Arxia SRL (www.arxia.com).

References

Beer, R. D. (1995), 'On the dynamics of small continuous-time recurrent neural networks', *Adaptive Behavior* **3**, 471–511. Available from: http://vorlon.ces.cwru.edu/~beer/Papers/CTRNNDynamics.pdf.

Bi, G.-Q. (2002), 'Spatiotemporal specificity of synaptic plasticity: cellular rules and mechanisms', *Biological Cybernetics* **87**, 319–332. Available from: http://www.neurobio.pitt.edu/2_bi2002BCreview.pdf.

Blynel, J. & Floreano, D. (2002), Levels of dynamics and adaptive behavior in evolutionary neural controllers, *in* (Hallam et al., 2002), pp. 272–281. Available from: http://asl.epfl.ch/aslInternalWeb/ASL/publications/uploadedFiles/blynel_sab02.pdf.

Bohte, S. B. (2003), *Spiking neural networks*. Available from: http://homepages.cwi.nl/~sbohte/pub_thesis.htm.

Bohte, S. M., Poutré, H. L. & Kok, J. N. (2002), 'SpikeProp: Error-backpropagation for networks of spiking neurons', *Neurocomputing* **48**, 17–37. Available from: http://homepages.cwi.nl/~sbohte/pub_spikeprop2k2.htm.

Chechik, G. (2003), 'Spike time dependent plasticity and information maximization', *Neural Computation* **15**, 1481–1510. Available from: http://www.cs.huji.ac.il/~ggal/ps_files/chechik_stdp.pdf.

DasGupta, B. & Schnitger, G. (1996), 'Analog versus discrete neural networks', *Neural Computation* **8**, 805–818. Available from: http://citeseer.nj.nec.com/dasgupta96analog.html.

Di Paolo, E. A. (2002*a*), 'Evolving spike-timing dependent plasticity for robot control', *EPSRC/BBSRC International Workshop: Biologically-inspired Robotics, The Legacy of W. Grey Walter, WGW'2002. HP Labs, Bristol, 14 - 16 August 2002*. Available from: http://www.cogs.susx.ac.uk/users/ezequiel/dipaolo_wgw2002.ps.

Di Paolo, E. A. (2002*b*), 'Spike timing dependent plasticity for evolved robots', *Adaptive Behavior* **10**, 243–263. Available from: http://citeseer.nj.nec.com/598020.html.

Di Paolo, E. A. (2003), 'Evolving spike-timing dependent plasticity for single-trial learning in robots', *Philosophical Transactions of the Royal Society A* **361**, 2299–2319.

Fernando, C. & Sojakka, S. (2003), Pattern recognition in a bucket, *in* 'Proceedings of the Seventh European Conference on Artificial Life (ECAL 2003)'. Available from: http://www.cogs.susx.ac.uk/users/sampsas/bucket.pdf.

Floreano, D. & Mattiussi, C. (2001), Evolution of spiking neural controllers for autonomous vision-based robots, *in* T. Gomi, ed., 'Evolutionary Robotics IV', Springer-Verlag, Berlin. Available from: http://asl.epfl.ch/aslInternalWeb/ASL/publications/uploadedFiles/er2001b.pdf.

Floreano, D. & Mondada, F. (1996), 'Evolution of homing navigation in a real mobile robot', *IEEE Transactions on Systems, Man, and Cybernetics–Part B: Cybernetics* **26**, 396–407. Available from: http://citeseer.nj.nec.com/floreano96evolution.html.

Floreano, D., Schoeni, N., Caprari, G. & Blynel, J. (2002*a*), Evolutionary bits'n'spikes, *in* R. K. Standish, M. A. Bedau & H. A. Abbass, eds, 'Artificial Life VIII: Proceedings of the Eight International Conference on Artificial Life', MIT Press, Boston, MA. Available from: http://asl.epfl.ch/aslInternalWeb/ASL/publications/uploadedFiles/alife02.pdf.

Floreano, D., Zufferey, J. C. & Mattiussi, C. (2002*b*), 'Evolving spiking neurons from wheels to wings', *Proceedings of the 3rd International Symposium on Human and Artificial Intelligence Systems, Fukui, Japan*. Available from: http://asl.epfl.ch/aslInternalWeb/ASL/publications/uploadedFiles/FloreanoEtAl_HARTS2002_color.pdf.

Florian, R. V. (2003*a*), 'Autonomous artificial intelligent agents', *Technical Report Coneural-03-01*. Available from: http://www.coneural.org/reports/Coneural-03-01.pdf.

Florian, R. V. (2003*b*), 'Thyrix: A simulator for articulated agents capable of manipulating objects', *Technical Report Coneural-03-02*. Available from: http://www.coneural.org/reports/Coneural-03-02.pdf.

French, R. L. B. & Damper, R. I. (2002), Evolution of a circuit of spiking neurons for phototaxis in a Braitenberg vehicle, *in* (Hallam et al., 2002), pp. 335–344.

Funahashi, K. & Nakamura, Y. (1993), 'Approximation of dynamical systems by continuous time recurrent neural networks', *Neural Networks* **6**, 801–806.

Gerstner, W. (1999), Spiking neurons, *in* (Maas & Bishop, 1999), pp. 3–53.

Gerstner, W. & Kistler, W. M. (2002), *Spiking neuron models*, Cambridge University Press, Cambridge, UK. Available from: http://diwww.epfl.ch/~gerstner/BUCH.html.

Goldenholz, D. (2003), 'Liquid computing: A real effect'. Available from: http://www.lsm.tugraz.at/papers/Goldenholz-report.pdf.

Gori, M. & Meer, K. (2002), 'A step towards a complexity theory for dynamical systems', *Mathematical Logic Quarterly* **48**, 45–58. Available from: ftp://ftp.imada.sdu.dk/pub/papers/pp-2001/22.ps.gz.

Grand, S. (1998), 'Battling with GA-Joe', *IEEE Intelligent Systems* **13**, 18–20. Available from: http://www.cyberlife-research.com/articles/ieee/ieee3.htm.

Hallam, B., Floreano, D., Hallam, J., Hayes, G. & Meyer, J.-A., eds (2002), *From animals to animats 7: Proceedings of the Seventh International Conference on Simulation of Adaptive Behavior*, MIT Press, Cambridge, MA.

Haykin, S. S. (1998), *Neural networks: A comprehensive foundation*, Prentice Hall.

Hertz, J., Krogh, A. & Palmer, R. G. (1991), *Introduction to the theory of neural computation*, Perseus Books, Cambridge, MA.

Ijspeert, A. J. (2001), 'A connectionist central pattern generator for the aquatic and terrestrial gaits of a simulated salamander', *Biological Cybernetics* **84**, 331–348. Available from: http://lslwww.epfl.ch/birg/PAPERS/BC.pdf.

Izhikevich, E. M. (2003*a*), 'Simple model of spiking neurons', *IEEE Transactions on Neural Networks*. Available from: http://www.nsi.edu/users/izhikevich/publications/spikes.htm.

Izhikevich, E. M. (2003*b*), 'Which model to use for cortical spiking neurons?', *IEEE Transactions on Neural Networks*. Available from: http://www.nsi.edu/users/izhikevich/publications/whichmod.htm.

Kandel, R., Schwartz, J. H. & Jessell, T. M., eds (2000), *Principles of neural science*, McGraw-Hill.

Koch, C. (1999), *Biophysics of Computation*, Oxford University Press, New York.

Koch, C. & Segev, I. (1997), *Methods in neuronal modeling: From synapses to networks*, MIT Press, Cambridge, MA.

Koch, C. & Segev, I. (2000), 'The role of single neurons in information processing', *Nature Neuroscience* **3**, 1171–1177. Available from: http://icnc.huji.ac.il/Files/Koch_Segev_NN2000.pdf.

Kolen, J. F. & Kremer, S. C., eds (2001), *A field guide to dynamical recurrent neural networks*, IEEE Press, New York.

Maas, W. (1996), On the computational power of noisy spiking neurons, *in* D. Touretzky, M. C. Mozer & M. E. Hasselmo, eds, 'Advances in Neural Information Processing Systems', Vol. 8, MIT Press, Cambridge, MA, pp. 211–217. Available from: http://www.cis.tugraz.at/igi/maass/psfiles/80.ps.gz.

Maas, W. (1997*a*), 'Networks of spiking neurons: the third generation of neural network models', *Neural Networks* **10**, 1659–1671. Available from: http://www.cis.tugraz.at/igi/maass/psfiles/85a.pdf.

Maas, W. (1997*b*), Noisy spiking neurons with temporal coding have more computational power than sigmoidal neurons, *in* M. Mozer, M. I. Jordan & T. Petsche, eds, 'Advances in Neural Information Processing Systems', Vol. 9, MIT Press, Cambridge, MA, pp. 211–217. Available from: http://www.cis.tugraz.at/igi/maass/psfiles/90.pdf.

Maas, W. & Bishop, C. M., eds (1999), *Pulsed neural networks*, MIT Press, Cambridge, MA.

Maass, W. & Natschlaeger, T. (1998), Associative memory with networks of spiking neurons in temporal coding, *in* L. S. Smith & A. Hamilton, eds, 'Neuromorphic Systems: Engineering Silicon from Neurobiology', World Scientific, pp. 21–32. Available from: http://www.cis.tugraz.at/igi/maass/psfiles/htc-ewns1.ps.gz.

Maass, W., Natschlaeger, T. & Markram, H. (2002), 'Real-time computing without stable states: A new framework for neural computation based on perturbations', *Neural Computation* **14**, 2531–2560. Available from: http://www.cis.tugraz.at/igi/maass/psfiles/ms2469-figs.pdf.

Maass, W., Schnitger, G. & Sontag, E. D. (1994), A comparison of the computational power of sigmoid and boolean threshold circuits, *in* V. P. Roychowdhury, K. Siu & A. Orlitsky, eds, 'Theoretical Advances in Neural Computation and Learning', Kluwer Academic Publishers, pp. 127–151. Available from: http://citeseer.nj.nec.com/maass94comparison.html.

Marian, I., Reilly, R. & Mackey, D. (2002), 'Efficient event-driven simulation for spiking neural networks', *Proceedings of the 3rd WSES International Conference on Neural Networks and Applications, Interlaken, Switzerland*.

McCullogh, W. S. & Pitts, W. (1943), 'A logical calculus of the ideas immanent in nervous activity', *Bulletin of Mathematical Biophysics* **5**, 115–133.

Minsky, M. L. (1967), *Computation: Finite and infinite machines*, Prentice Hall, Englewood Cliffs.

Moore, S. C. (2002), *Back-propagation in Spiking Neural Networks*. Available from: http://www.simonchristianmoore.co.uk/back.htm.

Mureşan, R. C. (2003), 'RetinotopicNET: An efficient simulator for retinotopic visual architectures', *Proceedings of the European Symposium on Artificial Neural Networks, Bruges, Belgium* pp. 247–254. Available from: http://www.raulmuresan.home.ro/Papers/RetinotopicNET.pdf.

Orponen, P. (1997), A survey of continuous-time computation theory, *in* D.-Z. Du & K.-I. Ko, eds, 'Advances in Algorithms, Languages, and Complexity', Kluwer Academic Publishers, Dordrecht, The Nederlands, pp. 209–224. Available from: http://citeseer.nj.nec.com/orponen97survey.html.

Orponen, P. (1999), 'The computational power of continuous time asymmetric neural networks', *Technical report*. Available from: http://citeseer.nj.nec.com/338971.html.

Orponen, P. (2000), An overview of the computational power of recurrent neural networks, *in* 'Proc. of the Finnish AI Conference (Espoo, Finland, August 2000), Vol. 3: "AI of Tomorrow"', Finnish AI Society, Vaasa, pp. 89–96. Available from: http://citeseer.nj.nec.com/orponen00overview.html.

Pearlmutter, B. A. (2001), Gradient calculations for dynamic recurrent neural networks, *in* (Kolen & Kremer, 2001), pp. 179–205.

Pfeifer, R. & Scheier, C. (1999), *Understanding intelligence*, MIT Press, Cambridge, MA.

Rao, R. P. N. & Sejnowski, T. J. (2003), 'Self-organizing neural systems based on predictive learning', *Philosophical Transactions of the Royal Society London A*. Available from: http://www.cs.washington.edu/homes/rao/royal_03.pdf.

Ruppin, E. (2002), 'Evolutionary embodied agents: A neuroscience perspective', *Nature Reviews Neuroscience* **3**, 132–142. Available from: http://www.cs.tau.ac.il/~ruppin/npaper10.ps.gz.

Shon, A. P. & Rao, R. P. N. (2003), 'Learning temporal patterns by redistribution of synaptic efficacy', *Neurocomputing*. Available from: http://www.cs.washington.edu/homes/aaron/pub/CNS02_ShonRao.ps.gz.

Siegelmann, H., Ben-Hur, A. & Fishman, S. (1999), 'Computational complexity for continuous time dynamics', *Physical Review Letters* **83**, 1463–1466. Available from: http://www.technion.ac.il/~asa/Papers/prl.ps.gz.

Siegelmann, H. T. (1993), 'Foundations of recurrent neural networks', *PhD thesis*. Available from: http://citeseer.nj.nec.com/173827.html.

Siegelmann, H. T. & Fishman, S. (1998), 'Computation by dynamical systems', *Physica D* **120**, 214–235. Available from: http://citeseer.nj.nec.com/144584.html.

Siegelmann, H. T. & Sontag, E. D. (1995), 'Computational power of neural networks', *Journal of Computer System Sciences* **50**, 132–150. Available from: http://citeseer.nj.nec.com/siegelmann91computational.html.

Sima, J. & Orponen, P. (2000), A continuous-time Hopfield net simulation of discrete neural networks, *in* 'Proceedings of NC2000: Second International ICSC Symposium on Neural Computing (Berlin, May 2000)', International Computer Science Conventions, Wetaskiwin, Canada, pp. 36–42. Available from: http://citeseer.nj.nec.com/101040.html.

Sima, J. & Orponen, P. (2003), 'General-purpose computation with neural networks: A survey of complexity theoretic results', *Neural Computation*. Available from: http://www.uivt.cas.cz/~sima/nntax.ps.

Slocum, A. C., Downey, D. C. & Beer, R. D. (2000), Further experiments in the evolution of minimally cognitive behavior: From perceiving affordances to selective attention, *in* J.-A. Meyer, A. Berthoz, D. Floreano, H. L. Roitblat & S. W. Wilson, eds, 'From animals to animats 6: Proceedings of the Sixth International Conference on Simulation of Adaptive Behavior', MIT Press, Cambridge, MA, pp. 430–439. Available from: http://vorlon.ces.cwru.edu/~beer/Papers/SAB2000.pdf.

Sommer, F. T. & Wennekers, T. (2000), 'Associative memory in networks of spiking neurons', *Neural Networks* **14**, 825–834. Available from: http://citeseer.nj.nec.com/sommer01associative.html.

Steels, L. & Brooks, R., eds (1995), *The artificial life route to artificial intelligence: Building embodied, situated agents*, Lawrence Erlbaum Associates, Hillsdale, NJ.

Tani, J. (2002), Articulation of sensory-motor experiences by "forwarding forward model": from robot experiments to phenomenology, *in* (Hallam et al., 2002).

Yamauchi, B. M. & Beer, R. D. (1994), 'Sequential behavior and learning in evolved dynamical neural networks', *Adaptive Behavior* **2**, 219–246. Available from: http://citeseer.nj.nec.com/yamauchi94sequential.html.