ELSEVIER

BioSystems 40 (1997) 11–19

BioSystems

Biologically plausible neural computation

Guido Bugmann*

Neurodynamics Research Group, School of Computing, University of Plymouth, Plymouth PL4 8AA, UK

Abstract

The function of a neuron can be described simultaneously at several levels of abstraction. For instance, a spike train represents the result of a computation done by a single neuron with its inputs, but it also represents the result of a complex function realized by the network in which the neuron is embedded. When models of large parts of the brain are considered, it may be desirable to use computational modules operating at a very abstract level. However, it is shown here that abstract neural functions depend on detailed features of the single neuron model used in the network reproducing the abstract function. Examples are given of the multiplicative function, motion detection, short-term memory and timing. All these operations rely on one or another feature of the extended Leaky Integrate-and-Fire neuron used in this paper, e.g. probabilistic synapses, post-synaptic currents modelled with alpha functions or partial reset of the somatic membrane. Consequently it is suggested that neural modelling at an abstract level does not obviate the need for a clear statement on the nature of the underlying model of biological neuron. In that sense, not many abstract functions are convincingly grounded, not even the standard formal neurons used in most artificial neural networks.

Keywords: Neuron model; Neural code; Neural modelling; Leaky integrator; Motion detection; Short-term memory; Timing; Multiplication

1. Introduction

The biological neuron has numerous complex features which can be exploited in various ways for computational purposes. For instance, non-linear interactions in dendrites can be used to perform a series of logical functions (Koch and Poggio, 1987; Shepperd and Brayton, 1987). In this paper short examples are given of the use of other features. In Section 3, leaky integration in the soma is used for *multiplication*. In Section 4, the dependence of the time course of dendritic currents on the electrotonic distance of the synapse from the soma is used for *motion detection*. Partial reset of the somatic membrane potential after a spike is used to *control the gain* of the neuron.

The computation performed by a neuron can be described at several levels. For instance, although a single neuron may perform the multiplication of its inputs, its output may represent the probability that a complex image is in the visual field. Therefore, the computational meaning of

* Corresponding author, e-mail: gbugmann@soc.plym.ac.uk

PII S0303-2647(96)01625-5

the output of a neuron depends on the network in which the neuron is embedded. In Section 5, an example is given of a small network behaving as a stochastic *short term memory* which can then be used within a larger network with a *timing function*.

Currently there is an urge to simulate more and more complex neural systems. In the past, the most complex models represented small parts of primary sensory systems. Recently, new data provided by brain imagery techniques, such as PET (Corbetta, 1993) call for models describing interactions between several brain areas (Taylor, 1994). To be computationally tractable, such a task requires the use of computational modules whose functions are many degrees abstracted from biological systems. However, to simulate these interactions, abstract modules should be firmly grounded in biological plausibility. A possible approach to ensure grounding is initially to construct a set of medium level functions representing medium size clusters of neurons which are characterized by simulation or mathematical analysis. These may be combined into more complex structures at a moderate computational cost, and so on. Generally, there has not been a careful verification that abstract computational units are decomposable into simpler functions compatible with biological neurons.

A concern throughout this paper is to make clear that a number of functions of neurons or a network of neurons are made possible only by the exploitation of detailed physiological properties of biological neurons. Therefore, a relatively detailed model has been used. It is an extension of the standard leaky integrator neurons, including in particular a model of dendritic propagation, probabilistic synapses, and the partial reset of the membrane potential (Section 2). In each section the problem of grounding the formal neurons is illustrated.

2. Extended Leaky Integrate- and-fire neuron model.

The neuron model used in this investigation is an extended version of the standard Leaky Integrate-and-Fire (LIF) neuron (Harmon, 1961; Bugmann, 1991). The detail of this model is described in Bugmann (1995). In summary, the four basic differences with the LIF are (starting with the input side of the neuron in Fig. 1):

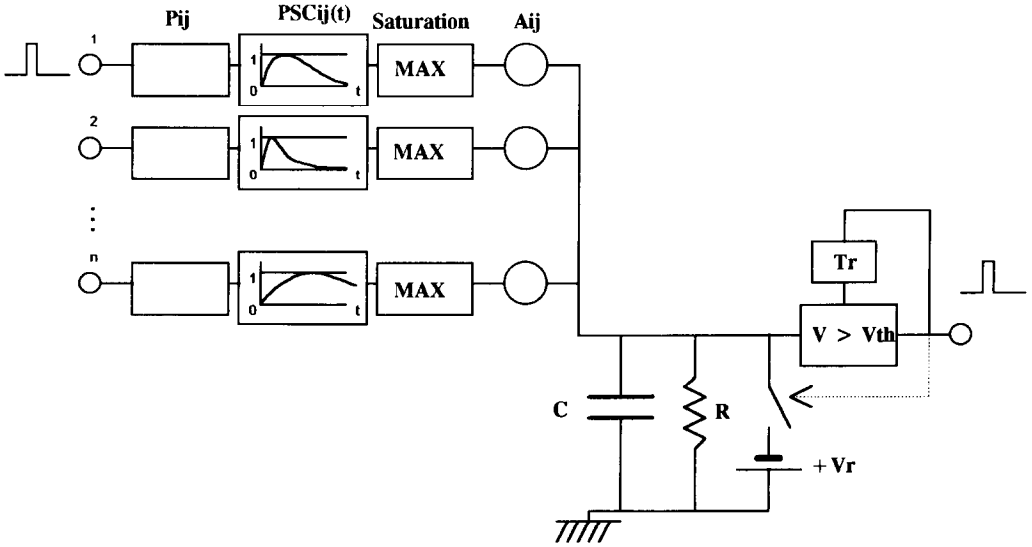

Fig. 1. Extended Leaky Integrate- and-Fire neuron. The features are explained in the Section 2. The simulation results shown in the paper are obtained on a PC with the simulation package CORTEX-PRO.

(1) Synapses have probabilistic transmission properties. An input spike generates a post-synaptic response with a probability P_{ij}. This is based on direct physiological observations (Pun et al., 1986) and gives interesting decaying memory properties to self-excitatory networks of spiking neurons (see Section 5) (Bugmann and Taylor, 1997).

(2) Input spikes do not cause a simple step-like increase in membrane potential as in (Softky and Koch, 1993) but initiate a post-synaptic current ($PSC_{ij}(t)$) extended in time and modelled by an alpha function which represents the dendritic current at its arrival into the soma:

$$PSC_{ij}(t) = \frac{t - t_0}{T_{max,ij}} \exp\left(\frac{t - t_0}{T_{max,ij}} + 1 \right) \qquad (1)$$

where t_0 is the arrival time of the spike. $T_{max,ij}$ is the time taken by the alpha function to reach its maximum. This function reproduces qualitatively the current pulse widening effect of dendritic propagation (cf. Stratford et al., 1989). The single parameter, $T_{max,ij}$, allows to simulate proximal inputs, which have a fast rise and decay of the PSC, e.g. by using $T_{max,ij} = 2$ ms, or distal inputs which have a slow time course of the PSC, for instance by using $T_{max,ij} = 50$ ms. The main advantage of this feature is that temporal effects associated with synaptic inputs at different electrotonic distances of the soma of the same neuron can be simulated. The PSCs and their hardware version (Christodoulou et al., 1992) can be used to realise motion detectors with realistic response properties (Section 4), whereas with the standard LIF model, all inputs are at the same electrotonic distance from the soma (Bugmann, 1991). In principle, any form of postsynaptic current can be accommodated by the model, such as the filter functions used in Bialek et al. (1991).

(3) Successive PSCs on the same synapse do not integrate temporally but saturate at a maximum value A_{ij} (used in conjunction with $T_{max,ij}$ to set the synaptic weight). However, PSCs from *different* synapses are integrated in the soma (the RC circuit of standard LIFs). Saturating synapses are biologically plausible and allow neurons to operate as coincidence detectors despite long membrane time constants (Bugmann, 1992). Excitatory synapses are simulated with $A_{ij} > 0$, and hyperpolarizing inhibitory synapses with $A_{ij} < 0$.

(4) The membrane is partially reset after a spike has been produced. The membrane potential V is not reset to zero but to a value $V_r = \beta V_{th}$ intermediate between zero and the threshold potential $V_{th} = 15$ mV. This increases the gain of the neuron and enables it to produce spike trains with an irregularity similar to that of biological spike trains (Bugmann, 1995; Bugmann, Christodoulou and Taylor, 1996). When values of β are close to one, as in Section 4 and 5, the membrane potential stays initially close to the threshold just after a spike. If a strong input current persists, a response in the form of bursts of spikes results (see Figs. 4 and 5).

3. Multiplication

Multiplication is based on a coincidence detection operation (Bugmann, 1991). It depends crucially on the leaky nature of biological integration, which make the LIF neuron sensitive to spike timing. It also requires a neural code made of discrete pulses with a probabilistic behaviour: Assume that a neuron with n inputs fires a spike during the time window Δt *only if* one input spikes arrives at that time from one of the inputs *and* if $n - 1$ input spikes have already arrived beforehand during a larger time window, τ, from each of the other inputs (see Fig. 2). This means that the $n - 1$ spikes have incremented the membrane potential in such a way that the n^{th} spike causes the firing. It requires weights to be set in a narrow range (Bugmann, 1991).

If the input spike trains are random, as produced by a Poisson process, then the probability

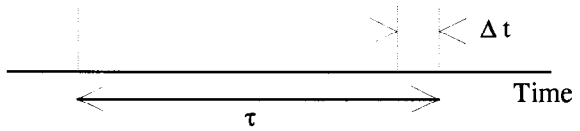

Fig. 2. Illustration of the time windows used to develop the multiplication Eqs. (2) and (3).

$P(n,\tau,\Delta t)$ that a spike be fired during Δt is given by:

$$P(n, \tau, \Delta t) = \Delta t\, \tau^{n-1} n \prod_{i=1}^{n} f_i \qquad (2)$$

Where f_i is the firing rate of the i^{th} input. Hence the output firing rate f_{out} is:

$$f_{out} = \frac{P(n, \tau, \Delta t)}{\Delta t} = \tau^{n-1} n \prod_{i=1}^{n} f_i \qquad (3)$$

This expression[1] shows that the output firing rate is proportional to the product of the input firing rates, under the stated conditions. Especially, each input must be able to provide at most one spike within a time window τ. This condition is impossible to satisfy with random inputs, but can be approached when the input firing frequencies are small and the value of τ is small, for instance due to short decay time RC of the membrane and the use of PSCs of short duration (proximal synapses). Under these conditions, the average input current to a neuron is small. As a result, 'multiplicative behaviour' is generally observed in the lower part of the transfer function. In this region a relatively linear plot of the output frequency as a function of the product of the input frequencies can be observed (Bugmann, 1991, 1992).

A perfect multiplier should stop firing when one of its inputs is silent. This is only observed in finely tuned models (Bugmann, 1991, 1992). However, in general, LIF neurons show a great sensitivity to the *number* of active inputs in the lower part of the transfer function. It can be seen in Fig. 3 that the response to a given average input current is larger when all inputs are actively contributing to that current.

This points to one of the differences between models of biological neurons and formal neurons. In formal neurons with sigmoidal transfer func-

[1] This expression corrects Eq. 4.4 in (Bugmann, 1991).

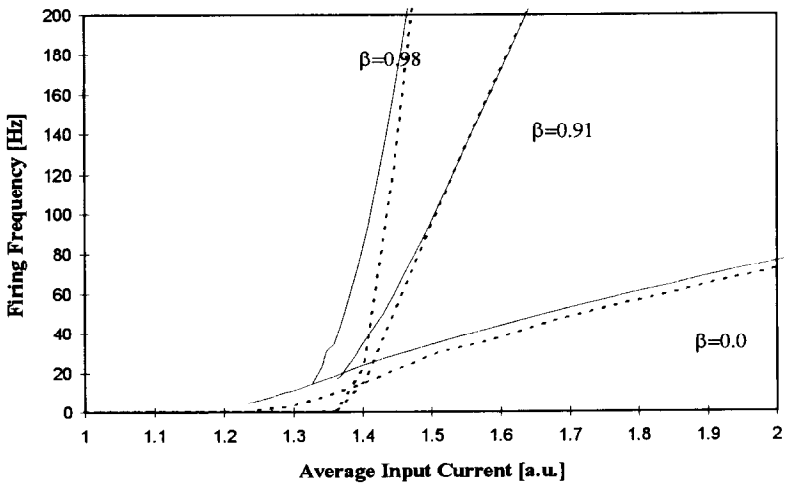

Fig. 3. Low firing frequency part of the transfer function of the extended LIF neuron, for 3 values of the reset parameter β. The reset parameter controls the gain of the neuron. **Full lines:** The average input current is scanned by progressively increasing the input firing frequency of all 50 inputs from 150 Hz to 200 Hz. **Dotted lines:** The average input current is varied by progressively increasing the number of active inputs up to 50, all inputs firing at 400 Hz. **Simulation conditions:** A refractory time of $Tr = 1$ ms and the simulation time steps of 1 ms result in a saturation firing rate of 500 Hz. $A_{ij} = 0.05$, $T_{\max,ij} = 2$ ms, $P_{ij} = 1$, $V_{th} = 15$.

tions[2], the weighted sum of the inputs carries no information about the number of active inputs, which therefore can not be reflected in the output firing rate. For a better reproduction of the multiplicative behaviour one may attempt to use an artificial 'Π-neuron' with a multiplicative transfer function, but this would not reproduce the behaviour of the upper parts of the curves (not shown in Fig. 3), first linear in the input current, and then saturating. A full analytical expression for the neuronal transfer function does not exist yet, even for the simplest LIF model (for further references see Lansky and Rospars, 1995).

4. Motion detection

This model for motion detection implements the correlation principle (Poggio and Reichardt, 1973) which is based on the product between the input signal and a delayed version of the input signal from a shifted retinal position. Here the delays are produced internally by exploiting the different time-to-maximum $T_{max,ij}$ associated with different electrotonic distances of the synapse from the soma (Fig. 4). Multiplication is done by coincidence detection, by setting weights (A_{ij}) in such a way that only signals adequately delayed can cause firing.

As the delay due to the dendritic propagation causes a widening of the EPSC, the motion detector has a wide response curves as shown in Fig. 4 and cannot give precise information regarding velocity. Although this scheme does not require inhibition for direction selectivity, apparently in contradiction with biological motion detectors (Hildreth and Koch, 1987), it accurately reproduces the shape of the response curve as measured in the fly (Fig. 4, right).

[2]A formal neuron i produces a real number between 0 and 1 as its output y_i by realizing following function over its n inputs y_j:

$$y_i = 1/\left(1 + \exp\left(-\sum_{j=1}^{n} y_j W_{ij} + W_{i0}\right)\right)$$

where W_{ij} are synaptic weights and W_{i0} is a bias input.

In this example, the neural information is in form of a burst of spikes in which the number of spikes depends on the velocity of the stimulus. Formal neurons have no temporal properties which would allow them to exhibit such a function. It is not clear at present how to define an abstract analytical function describing this motion detector, especially since it has not yet been characterized with more general repeated stimuli, for instance emulating the movement of a random dot pattern.

5. Short-term memory and a neuronal timer

In this example (Fig. 5), the response of the neuron reflects the network dynamics. Lateral connections cause a sustained prolonged firing, which is a form of short-term memory of the initial stimulus having activated the network. The neurons in a recurrent network 'Cluster' are activated by a stimulus taking the form of random firing of neurons in the layer 'Input'. The recurrent connections in 'Cluster' sustain the activity of the network after the stimulus has disappeared. Due to the probabilistic nature of the synaptic transmission, there is a gradual decrease of the total activity in 'Cluster' and of the firing rate of each neuron. Thus, the firing frequency indicates the delay since stimulus termination.

The temporal decay of the activity in 'Cluster' depends critically on the duration of the PSCs and the lateral synaptic transmission probability. For instance, using shorter EPSCs leads to a more abrupt termination of the sustained activity, although it can still be of long average duration. In the limit of very short EPSCs, no information about the offset time of the stimulus is carried by the firing rate of individual neurons or the activity of the network as a whole, both having only two states, active or silent. In these cases, a population of such networks, having probabilistic durations of sustained firing, is needed to extract timing information (Bugmann and Taylor, 1997). In the example shown here, prolonged EPSCs are used and the decaying activity of a single network carries sufficient timing information.

A simple network architecture is shown which can produce a burst of spikes when the activity in

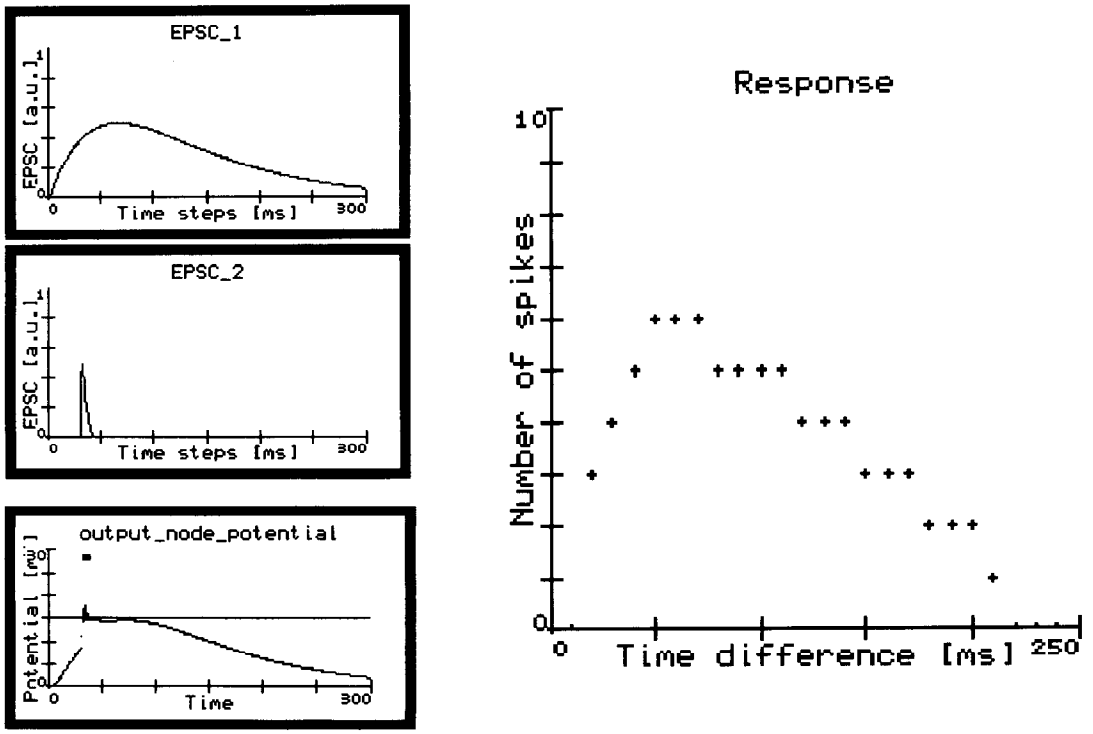

Fig. 4. Motion detector. **Left:** This example is based on the neuron model of Fig. 1. A moving stimulus is assumed to activate successively the synapses 1 and 2. The graphs EPSC_1 and EPSC_2 show a slow EPSC ($T_{max,1} = 65$ ms, $A_1 = 1.4$, $P_1 = 1$) triggered first at synapse 1, then, 40 ms later, a fast EPSC ($T_{max,1} = 2$ ms, $A_2 = 3$, $P_2 = 1$) being triggered at synapse 2. Both EPSCs integrate in the soma and result in a potential increase, shown in the graph output_node_depotential, such that a burst of spikes is fired. The production of a burst is due to partial reset with a reset parameter $\beta = 0.98$ (Bugmann, 1995). **Right:** Number of spikes in the burst produced for different time differences between initiation of EPSC_1 and EPSC_2. This curve is similar to the one observed in the cell H1 of the visual system of the fly (Franchescini, 1985). The simulation is done with 1 ms time steps. $V_{th} = 15$ mV.

the short-term memory 'Cluster' has decayed sufficiently. As shown in Fig. 5, an excitatory and an inhibitory neuron receive projections from the same sub-population of neurons in 'Cluster' (see caption for details). Both neurons have the same input weights and duration of the EPSCs. They excite the neuron 'Timer' with long EPSCs and inhibit it with short IPSCs. When the 'Cluster' is sufficiently active, 'Excit' and 'Inhib' produce their first spike and the fast IPSC initially dominates the potential in 'Timer'. When the activity of the 'Cluster' diminishes, 'Excit' and 'Inhib' become silent, but the long EPSCs are still active and cause the 'Timer' to fire. As in Section 4, the neuron 'Timer' is using a partial reset (with $\beta =$ 0.9) to produce bursts of spikes.

Regarding the design of an abstract model of the timer, the decaying curve of the memory activity of Cluster can be reproduced with a single formal neuron with a self-feedback loop with gain < 1. The timer based on the decaying activity of this 'neuron' would be precise and reproducible, unlike the timer based on our network, which shows a distribution of firing times (Timer_histogram in Fig. 5). However, biological timers are also stochastic (Jasselette et al., 1990) and can, therefore, not be modelled using formal neurons. A simple random number generator with a given probability distribution could constitute an acceptable abstract timer module within a more complex biological model, because it can be traced back to a network of biological neurons. However, the fact that animal timing behaviour follows stochastic rules, does not alone justify the

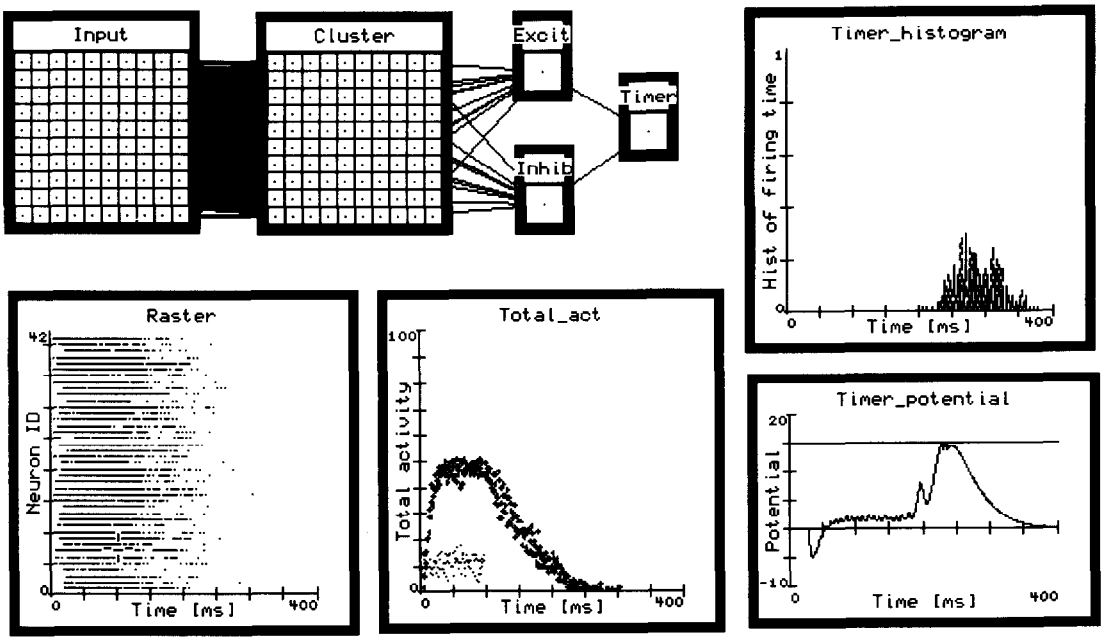

Fig. 5. A neuronal timer. **Short-term memory component (Input and Cluster):** The graph 'Total_act' represents the total number of neurons firing at each time step (1 ms). The dots correspond to the layer 'Input'. The activity ends after 100 ms. The crosses correspond to the layer 'Cluster'. The activity persists and decays slowly after the input ends. The graph 'Raster' shows spike trains of 40 neurons in 'Cluster'. Their firing rate decreases progressively after end of the input. The jitter in starting time is due to the stochastic arrival of input spikes and the probabilistic recurrent synaptic transmission. **Timer component (Excit, Inhib and Timer):** the neurons 'Excit' and 'Inhib' received inputs from the same neurons in Cluster and fire at the same time. The neuron 'Timer' receives short IPSCs ($T_{max} = 5$ ms) from 'Inhib' and long EPSCs ($T_{max} = 25$ ms) from 'Excit'. The graph 'Timer potential' shows the somatic membrane potential of 'Timer'. When the activity in 'Cluster' becomes too small to fire 'Inhib' and 'Excit', the slow EPSCs are still active and raise the potential to the firing threshold. A burst is produced due to partial reset. Sometimes a second burst occurs which explains the two peaks of the distribution of firing times shown in the graph 'Timer_histogram'. This timer fires approximately 180 ms after stimulus offset. **Simulation parameters:** Input: 10×10 units firing random spikes with a probability $P_0 = 0.105$ per time step, between step 1 and 100. One time step represents 1 ms. In all other layers, neurons of the type shown in Fig. 1 are used, with $\beta = 0$, except for Timer where $\beta = 0.9$. Cluster: 10×10 neurons receiving projections form input units selected at random with a connection probability = 0.1. Same scheme for the recurrent connections within Cluster. No self connections. Idem for connections from cluster to Inhib and Excit with the constraint of having identical source neurons for both neurons. Connection details: Input → Cluster: $A_{ij} = 5$, $P_{ij} = 0.1$; $T_{max} = 5$; Cluster → Cluster: $A_{ij} = 3$, $P_{ij} = 0.055$; $T_{max} = 5$; Cluster → (Inhib, Excit): $A_{ij} = 0.27$, $P_{ij} = 1$; $T_{max} = 5$; Inhib → Timer: $A_{ij} = -1.8$, $P_{ij} = 1$; $T_{max} = 5$; Excit → Timer: $A_{ij} = 1.8$, $P_{ij} = 1$; $T_{max} = 25$.

use of a stochastic timer, since other mechanisms could possibly also lead to a stochastic timing behaviour. Theoretical models which produce timing probabilities instead of stochastic behaviours are not models of the subject but models of the hybrid system comprising the subject *and* the off-line data evaluation process, which produces average responses in form of probabilities.

6. Conclusion

A number of examples have been shown of functions depending crucially on detailed features of the model of biological neurons. The biological brain does not necessarily use these functions, but it may do so. In each of these examples, the output spike trains encode information in a different way: the product of input frequencies is encoded in the firing frequency, or the probability of observing a spike within a given time window. Motion velocity is encoded by the number of spikes in a burst. Elapsed time since an event is encoded in a firing rate, or in a binary code signalling that the event has happened or not

(with short EPSCs). The timer produces a burst with a binary signification at a given time. This multiplicity of codes makes it unlikely that a unique and simple coding scheme may be designed for the communication between abstract computational modules.

Such modules can be implemented in ways avoiding the computational costs of a direct simulation of a network of biological neuron models. A simple implementation is suggested in the case of the timer. In general, the analysis of detailed simulations of a module is needed to provide the justification for a more abstract form of implementation.

Concerning formal neurons, it is shown that their function is not an abstraction of any of the functions of the biological neuron model described here (Section 2). Nevertheless it is tempting to use formal neurons because they are computationally less costly to simulate than extended LIF neurons. However, the number of formal neurons needed to emulate a given biological network may be much larger than the number of biological neurons. For instance, it is expected that a network of formal neurons reproducing the motion detection function described in Section 4 would be of large size. It can not be excluded, however, that there may be a critical size for a larger network of biological neurons for which formal neuron based models may become more economical. It is generally assumed that formal neurons reproduce the average response of a large number of biological neurons (Amit, 1989), but it is not yet clear for which range of network architectures and neuronal parameters this assumption holds. Further comparative studies along these lines are needed. Therefore, despite some interesting learning results with formal neurons (Zipser and Anderson, 1988) it is currently difficult to justify their use in biological models.

In general, the effort of establishing the biological grounding of an abstract neural function may be rewarding in itself: It would increase our understanding of the algorithmic constraints faced by the brain, and clarify the computational significance of the brain circuitry.

Acknowledgements

The author is thankful for valuable comments by Jack Boitano and Raju Bapi on earlier versions of this paper.

References

Amit, D.J., 1989, Modelling brain function: the world of attractor neural networks. Cambridge University Press, Cambridge, England.

Bialek, W., Rieke, F., De Ruyter van Steveninck, R.R. and Warland, D., 1991, Reading a neural code. Nature 252, 1854–1857.

Bugmann, G., 1991, Summation and multiplication: Two distinct operation domains of leaky integrate- and-fire neurons. Network: Comput. Neural Sys. 2, 489–509.

Bugmann, G., 1992, Multiplying with neurons: Compensation for irregular input spike trains by using time dependent synaptic efficiencies. Biol. Cybern. 68, 87–92.

Bugmann, G., 1995, Controlling the irregularity of spike trains Research Report NRG-95-04 (available from http: /www.tech.plym.ac.uk/soc/research/neural/index.html).

Bugmann, G., Christodoulou, C., and Taylor, J.G., 1996, Role of temporal integration and fluctuation detection in the highly irregular firing of a leaky Integrator Neuron with partial reset. Neurol Computation, in press.

Bugmann, G. and Taylor, J.G., 1997, A stochastic short-term memory using a pRAM neuron and its potential applications, in: Recent advances in neural networks, R. Beale and M.D. Plumbley (eds.) (Ellis Horwood Publishing) (to appear, available on the web server above).

Christodoulou, C., Bugmann, G., Taylor, J.G. and Clarkson, T., 1992, An extension to the temporal noisy-leaky integrator neuron and its potential applications. Proc. IJCNN'92 (Beijing) vol. III, 165–170.

Corbetta, M., 1993, Positron emission tomography as a tool to study human vision and attention. Proc. Natl. Acad. Sci. USA 90, 10901–10903.

Francescini, N., 1985, Early processing of colour and motion in a mosaic visual system. Neurosci. Res. Suppl. 2, s17–s49.

Jasselette, P., Lejeune, H. and Wearden, J.H., 1990, The perching response and the laws of animal timing. J. Exp. Psychol: Animal Behav. Proc. 16, 150–161.

Harmon, L.D., 1961, Studies with artificial neurons, I: properties and functions of an artificial neuron. Kybernetik 3, 89–101.

Hildreth, E.C. and Koch, C., 1987, The analysis of visual motion: From computational theory to neural mechanisms. Annu. Rev. Neurosci. 10, 477–533.

Koch, C. and Poggio, T., 1987, Biophysics of computation: Neurons, Synapses and Membranes, in: Synaptic function,

G.M. Edelman, W.E. Gall and W.M. Cowan (eds.) (John Wiley and Son, New York) pp. 637–697.

Lansky, P. and Rospars, J.P., 1995, Ornstein-Uhlenbeck model neuron revisited. Biol. Cybern. 72, 397–406.

Poggio, T. and Reichardt, W.E., 1973, Considerations on models of movement detection. Kybernetik 13, 223–227.

Pun, R.Y.K., Neale, E.A., Guthrie, P.B. and Nelson, P.G., 1986, Active and inactive synapses in cell culture. J. Neurophysiol. 56, 1242–1256.

Shepperd, G.M. and Brayton, R.K., 1987, Logic operations are properties of computer-simulated interactions between excitable dendritic spines. Neuroscience 21, 151–165.

Softky, W.R. and Koch, C., 1993, The highly irregular firing of cortical cells is inconsistent with temporal integration of random EPSPs. J. Neurosci. 13, 334–350.

Stratford, K., Mason, A. Larkman, A., Major, G. and Jack, J., 1989, The modelling of pyramidal neurons in the visual cortex, in: The computing neuron, R. Durbin et al. (eds.) (Addison-Wesley, Wokingham), pp. 296–321.

Taylor, J.G., 1994, Global modelling and testing of models of brain and mind, in: Supercomputing in brain research: From tomography to neural networks, H.J. Hermann, D.E. Wolf and E. Pöppel (eds.) (World Scientific, Singapore), pp. 11–26.

Zipser, D. and Andersen, R.A., 1988, A back-propagation programmed network that simulates response properties of a subset of posterior parietal neurons. Nature 331, 679–684.