Which Model to Use for the Liquid State Machine?

Beata J. Grzyb, Eris Chinellato, Grzegorz M. Wojcik, and Wieslaw A. Kaminski

***Abstract*— The properties of separation ability and computational efficiency of Liquid State Machines depend on the neural model employed and on the connection density in the liquid column. A simple model of part of mammalians visual system consisting of one hypercolumn was examined. Such a system was stimulated by two different input patterns, and the Euclidean distance, as well as the partial and global entropy of the liquid column responses were calculated. Interesting insights could be drawn regarding the properties of different neural models used in the liquid hypercolumn, and on the effect of connection density on the information representation capability of the system.**

I. Introduction

In biological circuits a single neuron would not be able to process complex input information, and neural microcircuits constitute a computational base [1]. A new idea on microcircuit computing was suggested by Maass [2] and, since then, it has been called Liquid State Machine (LSM). In general, the brain cortex (or one of its areas) is treated as a "liquid". Cortical microcircuits turn out to be very good "liquids" for computing perturbations. Microcircuits are characterized by the large diversity of elements, neurons, synapses, and variety of mechanisms and time constants characterizing their interactions, involving recurrent connections. As an illustration for the LSM approach, a series of transient perturbations caused in an excitable medium (i.e., liquid) can be considered as a sequence of external disturbances ("inputs") like sounds, stones dropped into the liquid or wind [3]. If the liquid is treated as an attractor neural network, it has only one possible attractor - its resting state. Nevertheless, perturbed states of the liquid, represent both present as well as past inputs, and the information they carry can be used for analysis of the surrounding environment.

Liquid computing efficiency has been investigated principally for integrate-and-fire neurons [2], and some experiments have been conducted for the sophisticated and numerically demanding Hodgkin-Huxley cells [4][5]. In this work we perform a systematic analysis and comparison of LSM computational performance for various neuron models. The integrate-and-fire, resonate-and-fire, FitzHugh-Nagumo, Morris-Lecar, both versions of Hindmarsh-Rose and Izikevich's neural models are examined and assessed.

Beata J. Grzyb and Eris Chinellato are with the Robotic Intelligence Lab, Computer Science and Engineering Department, Jaume I University, 12071 Castellon, Spain (email: {grzyb,eris}@icc.uji.es). Grzegorz M. Wojcik and Wieslaw A. Kaminski are with the Institute of Computer Science, Division of Complex Systems and Neurodynamics, Maria Curie-Sklodowska University, 20-601 Lublin, Poland.

Support for the UJI Robotic Intelligence Lab is provided in part by the European Commission under project EYESHOTS (FP7 ICT-217077) and by the Spanish Ministry of Science and Innovation (FPU grant AP2007-02565). Support for this research was provided also by the Polish State Committee for Scientific Research (grant N N519 403734).

Sec. II of the paper introduces the general architecture of a liquid state machine and its properties, as well as measures of its computational efficiency. In Sec. III we overview neuron models used in this investigation. Sec. IV presents the details of our experimental setup. Finally, in Sec. V we discuss and compare the different neuron models used for the hypercolumn implementation.

II. Liquid State Machine

The model of a Liquid State Machine is based on a strict mathematical framework that guarantees, under ideal conditions, universal computational power. A detailed description of the LSM theory can be found in [2], [6].

A liquid state machine consists of three main components. The first component is a layer of input neurons where an external stimulus is introduced. The signal from this component is transmitted to the selected neurons of the second component, called the liquid column, where the proper neural computation takes place. In mathematical terms, the liquid state is simply the current output of some operator or filter L^M that maps input functions $u(\cdot)$ onto functions $x^M(t)$:

$$x^M(t) = (L^M u)(t). \tag{1}$$

The liquid part receives a continuous input stream $u(s)$, and at later time $t > s$ the current internal state of $x_M(t)$ holds a substantial amount of information about recent inputs $u(s)$. The current liquid state $x^M(t)$ is passed to the last component of a liquid state machine, that is a memory-less readout map f^M (in a sens that it has no access to the states prior to time t), which performs an analysis and interpretation of liquid calculations. The signal is transformed by a readout map into the output:

$$y(t) = f^M(x^M(t)). \tag{2}$$

A readout map f^M structure is not given explicitly, and all available statistical analysis or pattern recognition methods can be used.

The most important property of a LSM is its ability to generate different responses to different input patterns: "the separation property" of the liquid. The ability of read-out maps to differentiate these responses, their generalization and relation with the given output was called the "approximation property". This property mainly depends on the adaptation ability of the read-out map, whereas the separation property depends directly on the complexity of the liquid.

To identify the difference between two liquid states or the difference in the LSM responses to two stimuli is a hard task. There is no concrete measure of the liquid states distance or of the input patterns distance. The liquid states are highly dynamic and non-linear, therefore it is easy to

notice that a simple "one to one" mapping of two states is not possible. Even a small difference in input pattern can cause a huge difference in liquid response. However, some simple comparisons, such as the Euclidean norm can provide clues about the distance between liquid trajectories for two input patterns. Formally, the Euclidean distance between liquid states for all recorded time steps is calculated according to the formula:

$$E_{S1S2}(t) = \sqrt{\sum_{i=1}^{n}(y_i^{S1}(t) - y_i^{S2}(t))^2}, \tag{3}$$

where n stands for the number of observed neurons, $y_i^{S1}(t)$ and $y_i^{S2}(t)$ are the $i-th$ neuron activity measured in time t after presenting $S1$ and $S2$ stimuli respectively.

In order to investigate the information flow in the liquid state machine an entropy concept based on the classical definition of Shannon's informational entropy can be used [7]. Simulating $1000ms$ of biological system's work, N spikes on the readout are obtained, and the number of spikes varies depending on the neuron model used in the liquid column and the patterns given in input. The probability of having n_i spikes occurring during a period $t_i = 10ms$ is:

$$p_i = \frac{n_i}{N}. \tag{4}$$

Such a probability can be interpreted as a chance of giving the whole information in t_i part of time T. Of course, all probabilities for time T sum up to 1.

$$\sum_{i=1}^{100} p_i = \frac{n_1 + n_2 + \cdots + n_{100}}{N} = 1. \tag{5}$$

Using probability p_i, partial entropy can be defined as:

$$S_o^i = -p_i \cdot ln(p_i) \tag{6}$$

and global entropy as:

$$S^i = \sum_i S_o^i = -\sum_i p_i \cdot ln(p_i). \tag{7}$$

In general, the measure of entropy helps in evaluating the capability of the system to represent properly the information contained in the data, and its high index is a cue that the system is able to capture the useful information available in input.

III. Spiking models

A neuron can be modeled at various levels of abstraction. In this section, we shortly review some neuron models which are used in our investigation. An extended review of biological neuron models can be found in [8].

A. Integrate-and-Fire

The integrate-and-fire model is probably the best-known and widespread used spiking model, mainly because of its properties, such as clarity of coding, possibility of mathematical analysis of network dynamics, and relatively efficient simulation of large networks. The detailed model description is given in [9]. The basic circuit of an integrate-and-fire model consists of a capacitor C in parallel with a resistor R driven by a current $I(t)$. The model dynamics is described by the following equation:

$$\tau_m \frac{du}{dt} = -u(t) + RI(t), \tag{8}$$

where τ_m is the membrane time constant of the neuron, and u is the membrane potential. The form of an action potential in the integrate-and-fire model is not given explicitly. Spikes are events characterized by a firing time $t^{(f)}$:

$$t^{(f)} : u(t^{(f)}) = \nu. \tag{9}$$

In other words, the membrane potential ***u(t)*** is compared with the threshold value ν. If $u(t) = \nu$ in time $t_i{}^{(f)}$, then an action potential $\delta(t - t_i{}^{(f)})$ is sent in output. Immediately after $t^{(f)}$, the potential is reset to a new value $u_R < \nu$.

$$\lim_{t \to t^{(f)}, t > t^{(f)}} u(t) = u_R. \tag{10}$$

B. Resonate-and-Fire

A resonate-and-fire neuron is obtained by linearization of the non-linear neuron model with the membrane potential suppression. The resulting linear system:

$$\dot{x} = bx - \omega y$$

$$\dot{y} = \omega x + by$$

can be written in the equivalent complex form:

$$\dot{z} = (b + i\omega)z, \tag{11}$$

where $z = x + iy \in \mathbb{C}$ is a complex-valued variable that describes the oscillatory activity of the neuron. The real part, x, is the current-like variable, whereas the imaginary part, y, is the voltage-like variable. Here, $(b+i\omega) \in \mathbb{C}$ is a parameter, where $b < 0$ is the rate of attraction to the rest state and $\omega > 0$ is the oscillation frequency. For more details please refer to [10].

C. FitzHugh-Nagumo

The FitzHugh-Nagumo model [11] is a two-dimensional simplification of the Hodgkin-Huxley model inspired by spike generation in squid giant axons. In the Hodgkin-Huxley system the variables v (voltage) and m (sodium activation), as well as the variables n (potassium activation) and h (sodium inactivation) have similar time courses. In the Fitzhugh-Nagumo model, v and m are regarded as mimicked by a single variable $v(t)$, called the voltage, and n and h are mimicked by a single variable $w(t)$ called the recovery variable. The model can be written in the general form:

$$\epsilon \frac{dv}{dt} = F(v) - w + I,$$

$$\frac{dw}{dt} = v - \gamma w, \tag{12}$$

where: $F(v) = v(1-v)(v+a), \epsilon \ll 1$, a, γ are constants, and I is the current injected into the neuron.

D. Morris-Lecar

Similar to the Hodgkin-Huxley, the Morris-Lecar [12] model has two ionic conductances (g_{Ca}, g_K), that underlie action potential activity. However, in the Morris-Lecar model the inward current is calcium rather than sodium. Additionally, to implement bursting, the model was extended to include an intracellular pool of calcium and a second, calcium-dependent potassium conductance. The kinetics of the calcium pool are relatively slow, and determine the activation kinetics of the calcium-dependent potassium conductance. The ordinary differential equations that define the Morris-Lecar model are as the following:

$$C_m \frac{dV_m}{dt} = I_{stim} - g_{Ca} m_\infty (V_m - E_{Ca}) - g_K n (V_m - E_K) - g_L (V_m - E_L) - g_{K(Ca)} \frac{[Ca]}{[Ca] + K} (V_m - E_K),$$

$$\frac{d[n]}{dt} = \frac{n_\infty - n}{\tau_n},$$

$$\frac{d[Ca]}{dt} = k_1 I_{Ca} - k_2 [Ca]. \quad (13)$$

Equations (13) for the membrane potential V_m and for the potassium activation are similar to those defining the Hodgkin-Huxley model. The novelty in this model is the calcium-dependent potassium current $I_{K(Ca)}$. The activation of $g_{K(Ca)}$ is determined by a function of calcium concentration $\frac{[Ca]}{[Ca]+K}$, and $[Ca]$ is defined by an equation in which calcium current $(k_1 I_{Ca})$ contributes calcium to the internal calcium pool and $(-k_2 I_{Ca})$ removes calcium from this pool.

E. Hindmarsh-Rose

Hindmarsh and Rose proposed several models among which the most popular is the model of bursting neurons. This model is based on a modified version of the FitzHugh-Nagumo model and is described by the following equations:

$$\frac{dx}{dt} = y - f(x) - z + I,$$

$$\frac{dy}{dt} = g(x) - y, \quad (14)$$

$$\frac{dz}{dt} = \epsilon (h(x) - z),$$

where I represents the applied current, $f(x)$ and $g(x)$ are functions experimentally deduced:

$$f(x) = ax^3 - bx^2,$$

$$g(x) = c - dx^2, \quad (15)$$

ϵ is the time scale of the slow adaptation current, and $h(x)$ is the scale of the influence of the slow dynamics, which determines whether the neuron fires in a tonic or in a burst mode when it is exposed to a sustained current input. (x^*, y^*) are the coordinates of the leftmost equilibrium point of the model without adaptation.

Hindmarsh-Rose'84 and Hindmarsh-Rose'85 models are both described by equations (14), but function $h(x)$ is defined differently. In the Hindmarsh-Rose'84 model it is:

$$h(x) = s(x - x^*),$$

and in the Hindmarsh-Rose'85 it is:

$$h(x) = K \frac{\mu^{(x+1+k)} - 1}{\mu^{(1+2k)-1}}, \quad (16)$$

where K and μ are constants.

F. Izhikevich's model

Equations (17) describe the Izhikevich model [14]:

$$\frac{dv}{dt} = 0.04v^2 + 5v + 140 - u + I,$$

$$\frac{du}{dt} = a(bv - u) \quad (17)$$

with the auxiliary after-spike resetting:

$$\text{if } v \geq 30mV, \text{ then } v \leftarrow c,\ u \leftarrow u + d, \quad (18)$$

In the equations above, a, b, c and d are parameters, I represents the synaptic current or the current directly introduced to the system. Variable v represents the membrane potential of the neuron and u represents a membrane recovery variable, which accounts for the activation of the potassium ionic currents and inactivation of sodium ionic currents. When the spike reaches its apex (+30 mV), membrane voltage and recovery variable are reset according to equation (18).

IV. Experimental setup

The proposed neural network is a simple simulation of part of mammalians visual system, and consists of two modules. Because the idea of LSM calls for such an architecture, our model includes an "Input" (Retina) and a "Liquid" (brain hypercolumn) structures (Fig. 1).

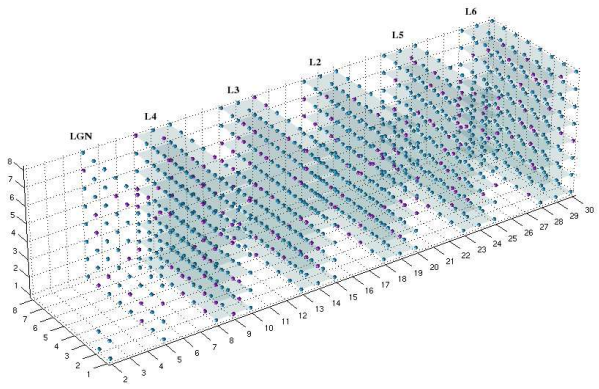
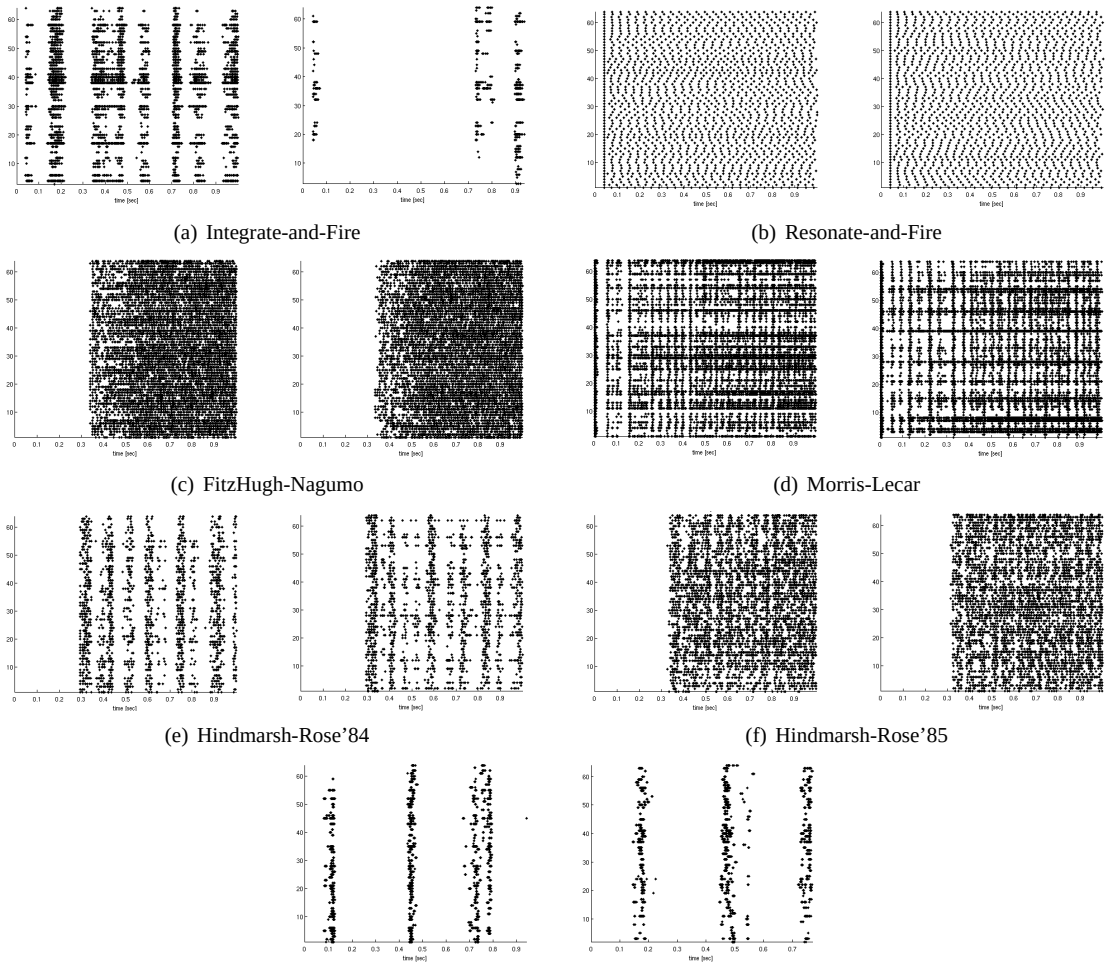


Fig. 1. The simplified model of mammalian visual system

The Retina (input module) is built up of four neurons put on a square shaped grid 2×2. Each neuron of the Retina is connected with one cell of the LGN layer (the 1st layer of the hypercolumn). The LGN consists of 64 neurons put on a 1×8×8 grid. The other cells of the hypercolumn are



(a) Integrate-and-Fire (b) Resonate-and-Fire

(c) FitzHugh-Nagumo (d) Morris-Lecar

(e) Hindmarsh-Rose'84 (f) Hindmarsh-Rose'85

(g) Izhikevich

Fig. 2. Liquid State Machine responses for two different stimuli S1 and S2 and different neuron models used for the hypercolumn implementation.

positioned on a 15×8×8 grid (5 layers 3×8×8 neurons: L4, L3, L2, L5, L6, approximately simulating the layer structure of the primary visual cortex). This makes a total of 1024 neurons in the hypercolumn module. Cells of each layer are connected with the neurons of its closest neighbor layers, in addition cells from L6 are recurrently connected with LGN (this also happens in the real visual system of mammals). Connections between two particular cells are arranged with probability calculated according to the rule:

$$P(D) = C * e^{\frac{-D(A,B)}{\lambda}}, \tag{19}$$

where $D(A, B)$ stands for the Euclidean distance between two cells A and B, and λ is the density of connections. Parameter C depends on the type of pre-synaptic and post-synaptic neurons, that is, whether they are the excitatory (Ex) or inhibitory (Inh) cells. In our simulations the parameter C is set as: $C_{Ex-Ex} = 0.3$ (for connections between two excitatory neurons), $C_{Ex-Inh} = 0.2$ (for connections between excitatory and inhibitory neurons), $C_{Inh-Inh} = 0.1$ (for connections between two inhibitory neurons), $C_{Inh-Ex} = 0.4$ (for connections between inhibitory and excitatory neurons). There are 80% of excitatory and 20% of inhibitory connections in the hypercolumn. All simulations discussed in this paper are conducted in the *CSim* package[1] developed for MATLAB. For the purpose of our investigation we extended this software package with the following neuron models: resonate-and-fire, FitzHugh-Nagumo, Morris-Lecar, Hindmarsh-Rose'84, and Hindmarsh-Rose'85.

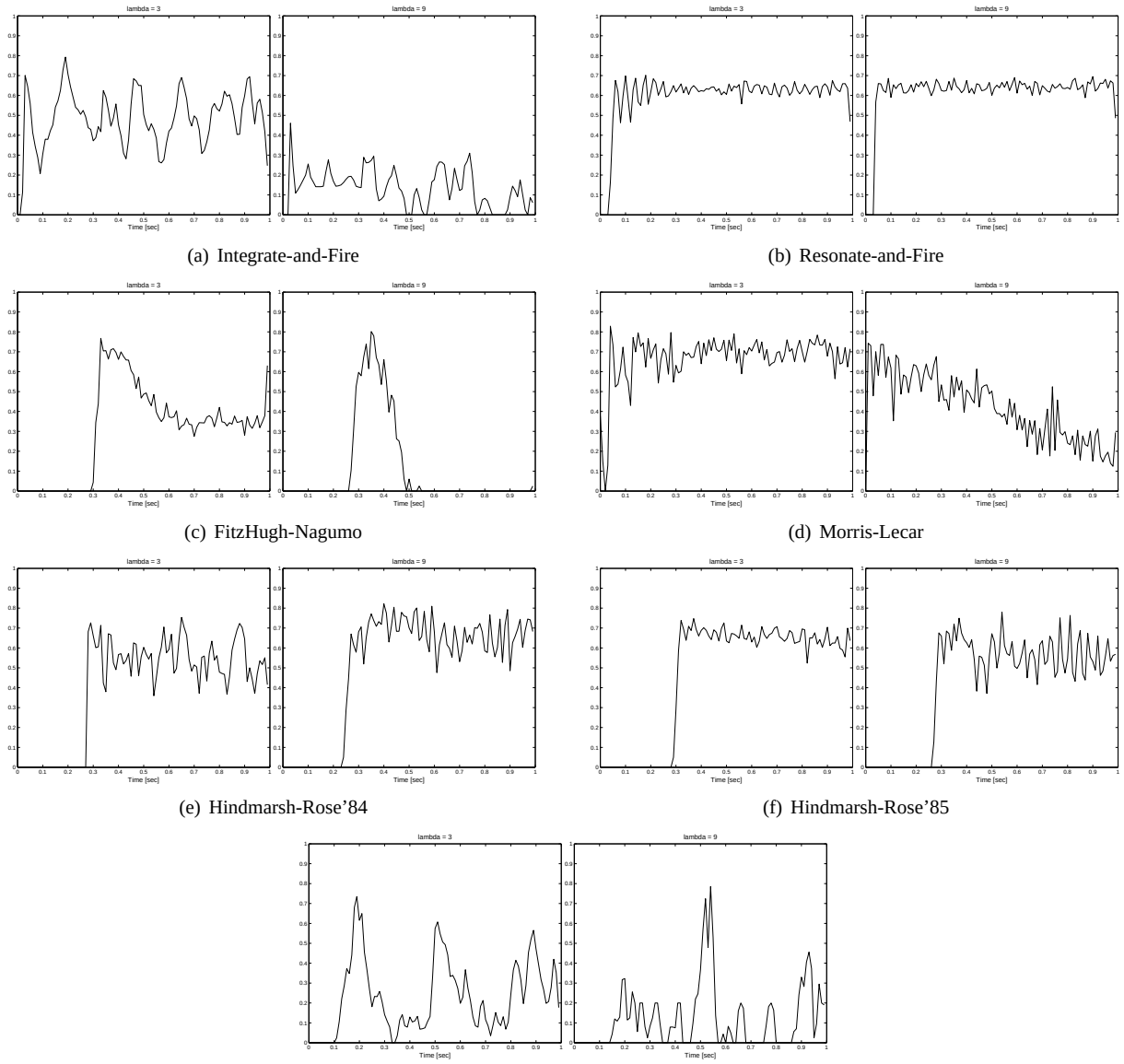
V. Results and discussion

On the Retina, two different and randomly chosen, time varying stimuli were provided. The simulation was repeated five times for each introduced input pattern, and the average of the five trials was computed for each condition.

A. Neural activity

The responses of Liquid State Machine for two different stimuli S1 and S2 and different models used in a hypercolumn are depicted in Fig. 2. In Fig. 2(a) the activity of the integrate-and-fire neurons LSMs is shown. A clear difference can be observed between responses with S1 and S2 inputs: whereas for the former input pattern neural activity shows very short inhibition periods, for the latter a long inhibition

[1]http://www.lsm.tugraz.at



(a) Integrate-and-Fire

(b) Resonate-and-Fire

(c) FitzHugh-Nagumo

(d) Morris-Lecar

(e) Hindmarsh-Rose'84

(f) Hindmarsh-Rose'85

(g) Izhikevich

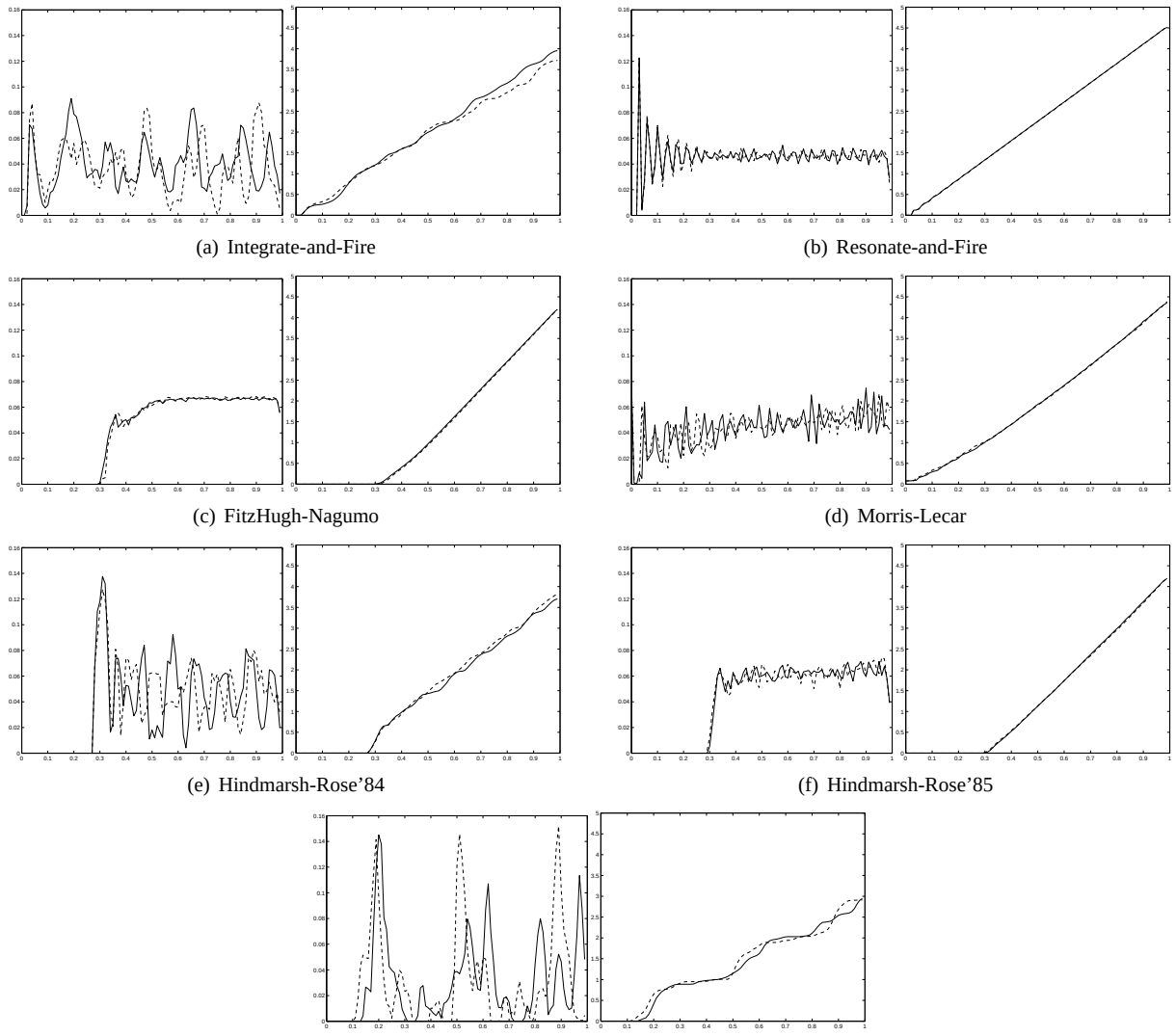
Fig. 3. Detected difference between two input stimuli for Liquid State Machines and different models of neuron used for the hypercolumn implementation, for $\lambda = 9$ and $\lambda = 3$ respectively.

appears after a short neural activity. The difference in activity for the resonate-and-fire LSM (Fig. 2(b)) is less clear, and some periodical behavior can also be observed. For the FitzHugh-Nagumo (Fig. 2(c)) and Morris-Lecar (Fig. 2(d)) LSMs, the difference between the two responses is again not clearly visible. In the FitzHugh-Nagumo case, there is a $350ms$ period of inhibition preceding high activity. For what concerns the Hindmarsh-Rose LSM neural implementation, completely different situations can be observed comparing the 1984 (Fig. 2(e)) and 1985 (Fig. 2(f)) versions of the model. For the former, the spike timing differences between two stimuli can easily be noticed, but not for the latter. Also in the Izhikevich's model spike timing between responses with S1 and S2 inputs are different. Moreover, the periodical activity is mixed with inhibitions (Fig. 2(g)).

B. Euclidean distance

The Euclidean distance was used to measure the hypercolumn difference between the states that appeared after feeding into the LSM the input patterns S1 and S2. The simulation time $T = 1000ms$ was divided into intervals, $t_i = 10ms$. The state of the hypercolumn was characterized in each interval by a 64-dimensional vector with coordinates equal to 1 for active and 0 for inactive neurons. Only the activity of the neurons belonging to the last sublayer of layer L6 ($1\times8\times8$) was investigated as this corresponds to the output layer of the whole network.

In Fig. 3, the Euclidean distance between LSM states as a percentage of maximal distance is presented for different values of the connection density parameter λ. For the integrate-and-fire neurons, the distance measured for $\lambda = 9$



(a) Integrate-and-Fire

(b) Resonate-and-Fire

(c) FitzHugh-Nagumo

(d) Morris-Lecar

(e) Hindmarsh-Rose'84

(f) Hindmarsh-Rose'85

(g) Izhikevich

Fig. 4. Partial (on the left) and global (on the right) entropy for a Liquid State Machine built of different neuron models.

is much higher than for $\lambda = 3$, which means that the net has a weaker pattern separation ability when the density of connections is high (Fig. 3(a)). In case of resonate-and-fire neurons, the separation ability is similar for both values of λ and oscillates around 65% (Fig. 3(b)). This means that the density of connections has no influence on the computational power of the LSM built with the resonate-and-fire model.

Totally different values of distance can be observed for the hypercolumn constructed of FitzHugh-Nagumo neurons. In case of $\lambda = 9$ the distance rapidly grows, starting from $250ms$ for about $100ms$ and then drops to zero. This behavior may suggest that neurons got synchronized, and the input patterns are indistinguishable. A different situation can be observed for $\lambda = 3$, the distance value rapidly grows at about $300ms$, reaches its maximum and then falls down but still keeps a small, positive value till the end of the simulation (Fig. 3(c)). A good separation ability on a constant level of 70% for $\lambda = 3$ was observed in the column consisting of the Morris-Lecar neurons. However, for $\lambda = 9$ the separation ability is high at the beginning, and slowly but sistematically falls (Fig. 3(d)). As far as Hindmarsh-Rose neurons are concerned, the difference between distances for both values of λ are very similar (Fig. 3(e), Fig. 3(f)). Therefore, it can be assumed that the density of connections has little influence on the computational power of the Hindmarsh-Rose LSMs. Poor separation ability characterizes the Izikevich neuron (Fig. 3(g)), as usually its value is lower than 30%.

Considering the hypercolumn configurations discussed above, for $\lambda = 3$ the Euclidean distance value is the highest for the Morris-Lecar model, and the lowest for the Izhikevich. This indicates that the Morris-Lecar neuron has the best separation ability and the Izhikevich's model the worst. Resonate-and-fire model and Hindmarsh-Rose'85 have also a quite good separation ability. On the other hand, the best separation ability for the connection density parameter $\lambda = 9$ observed from Hindmarsh-Rose'84, which is the only case

when performance gets better after increasing the synaptic connection density. The lowest distance values for $\lambda = 9$ have the integrate-and-fire and Izhikevich's model. Overall, the LSM seems to distinguish more efficiently, and thus separate better the input states, for $\lambda = 3$ (lower density).

Apart from the high separation property, the stability of the behavior should also be taken into account while selecting the neuron model for LSM simulations. The resonate-and-fire model, having high and stable Euclidean distances for both λ parameters, seems to be a good choice. Also possible are Morris-Lecar and Hindmarsh-Rose'85 neurons due to their relatively stable and high separation ability, but they should be used in LSMs with lower density of connections.

C. Partial and global entropy

We measured the entropy of the system for each model used in the hypercolumn in every instant, for both input patterns S1 and S2. Fig. 4 shows partial (left graphs) and cumulative entropy (right graphs) for each neuron model.

The global entropy varies for all models, being the lowest for Izhikevich's model (around 3), and the highest for resonate-and-fire, FitzHugh-Nagumo, and Morris-Lecar models (around 4.5), which means that some neuron models are more active during stimulation and encode input information using larger number of spikes (see Fig. 2). It can be noticed, that for FitzHugh-Nagumo, both Hindmarsh-Rose, and Izhikevich's models so-called dead state appears, that is the time in which the neurons are not active, which explains the initial 0 value of partial, as well as global entropy.

The changes in partial entropy for the input patterns S1 and S2 are small for the resonate-and-fire (Fig. 4(b)), FitzHugh-Nagumo (Fig. 4(c)), and Hindmarsh-Rose'85 (Fig. 4(f)). This means that the number of spikes is equally spread over the simulation time, maximizing the efficiency of transmitted information. The same conclusion can be also drawn from the plots of the global entropy, which for these neurons grows proportionally. On the other hand, the biggest difference in partial entropy is noticed for the Hindmarsh-Rose'84 (Fig. 4(e)), Izhikevich (Fig. 4(g)), and integrate-and-fire (Fig. 4(a)) models. This suggests that these models do not code information optimally, and their unstable behavior makes them rather unreliable, at least for these inputs. Higher entropy is reached by FitzHugh-Nagumo, Hindmarsh-Rose'85, and this can be interpreted as a cue that the LSM coded with such models is especially suitable for coding the given input patterns.

VI. Conclusions

The dynamics of the simulated neural hypercolumn of LSM was analyzed. Wide variety of neuron models were used for the cells implementation. Different input patterns shown to the simulated Retina usually resulted in different activities of the hypercolumn. Only in the case of Izikevich's model of neuron the separation ability was low. Density of connections λ affected in different ways the separation ability of the neuron models, and with the exception of Hindmarsh-Rose'84 model, the separation ability fell after the λ increased. The separation ability for parameter $\lambda = 3$ was the best for the Morris-Lecar neurons, while for $\lambda = 9$ was the best for Hindmarsh-Rose'84. Quite stable and similar for both values of λ was the Euclidean distance for resonate-and-fire neuron, which also proves its good separation ability.

In conclusion, taking into account both the entropy and separation ability, we postulate that Morris-Lecar, resonate-and-fire, and Hindmarsh-Rose'85 (for lower density of connections) neurons are the most suitable for LSM implementation. Although variations of the partial entropy for Morris-Lecar model are bigger than in the case of resonate-and-fire and Hindmarsh-Rose'85 models, they are constant and in our opinion acceptable. All aforementioned models have high and stable separation ability. The neuron's activity for these neuron models is highly symmetric and have a regular spike rate during the whole simulation.

References

[1] J. Obrach, *The Neuropsychological Theories of Lashley and Hebb*, University Press of America, 1998.

[2] W. Maass, T. Natschlager, H. Markram, "Real-time Computing Without Stable States: A New Framework for Neural Computation Based on perturbations," *Neural Computations*, vol. 14, pp. 2531-2560, 2002.

[3] A.V. Holden, J.V. Tucker, B.C. Thompson, "Can excitable media be considered as computational systems?," *Physica D*, vol. 49, pp. 240-246, 1991.

[4] G.M. Wojcik, W.A. Kaminski, "Liquid State Machine Built of Hodgkin-Huxley Neurons and Pattern Recognition," *Neurocomputing*, vol. 239, pp. 245-251, 2004.

[5] G.M. Wojcik, W.A. Kaminski, "Liquid State Machine and its separation ability as function of electrical parameters of cell," *Neurocomputing*, vol. 70, pp. 2593-2697, 2007.

[6] T. Natschlger, W. Maass, H. Markram, "The "Liquid Computer": A Novel Strategy for Real-Time Computing on Time Series," *Special Issue on Foundations of Information Processing of TELEMATIK*, vol. 8, 2002, pp. 39-43, 2002.

[7] W.A. Kaminski, G.M. Wojcik, "Liquid State Machine Built of Hodgkin-Huxley Neurons - Pattern Recognition and Informational Entropy," *Annales Informatica UMCS*, vol. 1, pp. 107-113, 2003.

[8] E.M. Izhikevich, "Which Model to Use for Cortical Spiking Neurons," *IEEE Transactions on Neural Networks*, vol. 15, pp. 1063-1070, 2004.

[9] W. Gerstner, W.M. Kistler, *Spiking Neuron Models, Single Neurons, Populations, Plasticity*, Cambridge 2002.

[10] E.M. Izhikevich, "Resonate-and-fire neurons," *Neural Networks*, vol. 14, pp. 883-804, 2001.

[11] R.A. FitzHugh, "Impulses and physiological states in theoretical models of nerve membrane," *Biophysical Journal*, vol. 1, pp. 445-466, 1961.

[12] E. Av-Ron, J.H. Byrne, D.A. Baxter, Teaching Basic Principles of Neuroscience with Computer Simulations, The Journal of Undergraduate Neuroscience Education, vol. 4, pp. 40-52, 2006.

[13] Korn H., Faure P., Is there chaos in the brain? II. Experimental evidence and related models, C.R. Biologies 326 (2003) 787-840

[14] E.M. Izhikevich, "Simple Model of Spiking Neurons," *IEEE Transactions on Neural Networks*, vol. 14, pp. 1569-1572, 2003.