

The oscillatory dynamic link matcher for spiking-neuron-based pattern recognition

Ramin Pichevar^{a,*}, Jean Rouat^a, Le Tan Thanh Tai^{a,b}

^a*Department of Electrical and Computer Engineering, Université de Sherbrooke, QC, Canada*

^b*Institut de la francophonie pour l'Informatique, Hanoi, Vietnam*

Received 25 February 2005; received in revised form 31 August 2005; accepted 22 November 2005

Available online 7 June 2006

Abstract

In this paper we show that an unsupervised two-layered oscillatory neural network with inter- and intra-layer connections, and a learning rule based on stimulus difference can behave as a dynamic link matching machine for invariant pattern recognition. We show that this architecture is robust to affine transformations. We call this architecture oscillatory dynamic link matching (ODLM). We use DEVS (discrete-event simulation) to increase simulation speed by updating the network only at instants when an internal or external stimulus is applied to neurons.¹

© 2006 Elsevier B.V. All rights reserved.

Keywords: Oscillatory dynamic link matching; Spiking neurons; Pattern recognition; Discrete-event simulation; Facial recognition

1. Introduction

Pattern recognition robust to noise, symmetry, homothety (size change with angle preservation), etc. has long been a challenging problem in artificial intelligence. Many solutions or partial solutions to this problem have been proposed using expert systems, neural networks, or statistical approaches. In general, three different approaches are used to perform invariant pattern recognition:

- *Normalization:* In this approach the analyzed object is normalized to a standard position and size by an internal transformation. The advantages of this approach are: the coordinate information (the “where” information) is retrievable at any stage of the processing and there is a minimum loss of information. The disadvantage of this approach is that the network

should find the object in the scene and then normalize it. This task is not as obvious as it can appear [26,14].

- *Invariant features:* In this approach some features that are invariant to the location and the size of an object are extracted. The disadvantages of this approach is that the position of the object may be difficult to extract after recognition and information is lost during the process. The advantage is that the technique does not require to know where the object is and unlike normalization in which other techniques should be used after this stage to recognize patterns, the invariant features approach already does some pattern recognition by finding important features [10].
- *Invariance learning from temporal input sequences:* The assumption is that primary sensory signals, which in general code for local properties, vary quickly while the perceived environment changes slowly. If one succeeds in extracting slow features from the quickly varying sensory signal, he/she is likely to obtain an invariant representation of the environment [38,33].

*Corresponding author. Tel.: +1 613 998 7203; fax: +1 613 993 9950.

E-mail addresses: Ramin.Pichevar@usherbrooke.ca (R. Pichevar), Jean.Rouat@usherbrooke.ca (J. Rouat), letanthanhtai@yahoo.com (L.T.T. Tai).

¹A preliminary version of this work has been presented at the Brain Inspired Cognitive Systems (BICS) Conference 2004, Stirling, UK. Now with CRC, Ottawa, Canada.

Based on the *Normalization* approach, the “dynamic link matching” (DLM) has been first proposed by Konen et al. [14]. This approach consists of two layers of neurons

connected to each other through synaptic connections constrained to some normalization. The saved pattern is applied to one of the layers and the pattern to be recognized to the other. The dynamics of the neurons are chosen in such a way that “blobs” are formed randomly in the layers. The formation of blobs is guaranteed by the continuum neural field theory (CNFT) [2,32]. If the features in these two blobs (from different layers) are similar enough, some weight strengthening and activity similarity will be observed between the two layers, which can be detected by correlation computation [14,4]. These blobs can or cannot correspond to a segmented region of the visual scene, since their size is fixed in the whole simulation period and is chosen by some parameters in the dynamics of the network [14]. The apparition of blobs in the network has been linked to the attention process present in the brain by the developers of the architecture (for a more recent work, see [34]). The dynamics of the neurons used in the original DLM network is not the well-known spiking neuron dynamics. In fact, its behavior is based on rate coding (average neuron activity over time, for details see Section 6) and can be shown to be equivalent to an enhanced dynamic Kohonen Map in its fast dynamic link matching (FDLM) form [14]. Here, we propose the oscillatory dynamic link matching (ODLM) algorithm, which uses conventional spiking neurons and is based on phase (place) coding. In a more general way, our proposed network can solve the correspondence problem, and at the same time, perform the segmentation of the scene, which is in accordance with the Gestalt theory of perception [12] and it is very useful when pattern recognition should be done in multiple-object scenes. In other words, the network does normalization, segmentation, and pattern recognition at the same time. It is also self-organized and requires no learning phase, in contrast with [43,19]. In addition, if only for relatively simple scenes, the segmentation phase can be bypassed, if the speed of convergence is the only concern (Section 8). Since the architecture is robust to affine transforms, it is capable of doing motion analysis, but neither it computes optical flow nor it performs additional signal processing between the layers, unlike in [42]. The application of this network is not limited to visual scene analysis, it can be used in sound source segregation problem and may act as a top-down (schema-driven) processor in the computational auditory scene analysis (CASA) [23].

2. The oscillatory dynamic link matcher

The building blocks of this network are oscillatory neurons [37]. The dynamics of this kind of neurons is governed by a modified version of the Van der Pol relaxation oscillator (called the Wang–Terman oscillator, for a similar approach with different dynamics see [5]). There is an active phase when the neuron spikes and a relaxation phase when the neuron is silent. The dynamics of the neurons follows the following state-space equations,

where x_i is the membrane potential (output) of the neuron and y_i is the state for channel activation or inactivation

$$\begin{aligned} \frac{dx_{ij}}{dt} &= 3x_{ij} - x_{ij}^3 + 2 - y_{ij} + \rho + H(p_{ij}^{\text{input}}) + S_{ij}, \\ \frac{dy_{ij}}{dt} &= \varepsilon[\gamma(1 + \tanh(x_{ij}/\beta)) - y_{ij}], \end{aligned} \quad (1)$$

where $H(\cdot)$ is the Heaviside function, ρ denotes the amplitude of a Gaussian noise, p_{ij}^{input} the external input to the neuron, and S_{ij} the coupling from other neurons (connections through synaptic weights). ε , γ , and β are constants. Initial values are generated by a uniform distribution between the interval $[-2; 2]$ for x_{ij} and between $[0; 8]$ for y_{ij} (these values correspond to the whole dynamic range of the equations).

It can be shown that the integrate-and-fire neuron (Eq. (2)) is a fixed-point approximation of the Wang–Terman oscillator [11]. From now on, we use equivalently these two models for simulation and theoretical analysis

$$\begin{aligned} \frac{dx_{ij}}{dt} &= -x_{ij} + H(p_{ij}^{\text{input}}) \\ \text{when } x_{ij} > \text{threshold} &\rightarrow x_{ij} = 0. \end{aligned} \quad (2)$$

A neighborhood of 4 is chosen in each layer for the connections. Each neuron in the first layer is connected to all neurons in the second layer and vice versa. A global controller is connected to all neurons in the first and second layers as in [6]. In a first stage, segmentation is done in the two layers independently (with no extra-layer connections) as explained in Section 4, while dynamic matching is done with both intra-layer and extra-layer couplings in the second stage. The intra-layer and extra-layer connections are defined as follows:

$$w_{i,j,k,m}^{\text{int}}(t) = \frac{w_{\text{max}}^{\text{int}}}{\text{Card}\{N^{\text{int}}(i,j) \cup N^{\text{ext}}(i,j)\} e^{\lambda|p(i,j;t) - p(k,m;t)|}}, \quad (3)$$

$$w_{i,j,k,m}^{\text{ext}}(t) = \frac{w_{\text{max}}^{\text{ext}}}{\text{Card}\{N^{\text{ext}}(i,j) \cup N^{\text{int}}(i,j)\} e^{\lambda|p(i,j;t) - p(k,m;t)|}}, \quad (4)$$

where $w_{i,j,k,m}^{\text{int}}(t)$ are intra-layer connections and $w_{i,j,k,m}^{\text{ext}}(t)$ are extra-layer connections (between the two layers) and $w_{\text{max}}^{\text{int}} = 0.2$ and $w_{\text{max}}^{\text{ext}} = 0.2$ are constants equal to the maximum value of the synaptic weights. $\text{Card}\{N^{\text{int}}(i,j)\}$ is a normalization factor and is equal to the cardinal number (number of elements) of the set $N^{\text{int}}(i,j)$ containing neighbors connected to the $neuron(i,j)$ and can be equal to 4, 3 or 2 depending on the location of the neuron on the map, i.e. center, corner, etc., and the number of active connections. A connection is active when $H(w_{i,j,k,m} - 0.01) = 1$, which is true both for intra-layer and extra-layer connections. $\text{Card}\{N^{\text{ext}}(i,j)\}$ is the cardinal number for extra-layer connections and is equal to the number of neurons in the second layer with active connection to $neuron_{i,j}$ in the first layer. Note that normalization in Eq. (4) is mandatory if someone wants to associate similar pictures with different sizes. If the aim is to match objects with exactly the same size the normalization

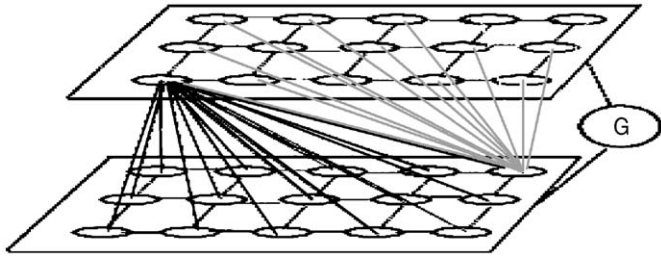


Fig. 1. The architecture of the oscillatory dynamic link matcher. The number of neurons in the figure does not correspond to the real number of neurons. The global controller has bidirectional connections to all neurons in the two layers.

factor should be set to a constant for all neurons. The reason for this is that with normalization even if the size of the picture in the second layer was the double of the same object in the first layer the total influence to the $neuron_{i,j}$ would be the same as if the pattern was of the same size.

The schematic of the network is shown in Fig. 1. The image to be recognized is applied to the first layer (bottom) of the network and the model (template) image is applied to the second layer (top). Biologically speaking, the first layer may correspond to the primary visual cortical areas, and the second layer to the infero-temporal cortex [14].

3. Motivations behind using the ODLM

The ODLM is based on the “temporal coding” paradigm. Temporal coding is different from the hierarchical coding [15] in which each face is identified by a single neuron, and is different from the sparse coding scheme [20]. Our “temporal coding” approach uses a population of neurons and the relative spiking phase among them to code information. Temporal coding is slower than the hierarchical coding. However, the number of neurons does not increase exponentially with the complexity of the task in “temporal coding”. Furthermore, the architecture of the network does not change with a change in the classification problem (for more details see [22,35]).

4. Behavioral description of the network

The network has two different behavioral mode: segmentation and matching. During the segmentation phase the images applied to the two layers of the network are segmented. This stage creates binary masks that are associated with different segments of the image and multiplexed in time (see for example Figs. 14 and 15). As neurons belonging to different regions of the segmented multi-part image cannot fire at the same time, snapshots of the network activity are binary and show only two regions at the same time. Thus, this is equivalent to observing binary images evolving through time. For two different times t_i and t_j ($t_i \neq t_j$ modulo T , T is the oscillation period) the activity will be different and can be assigned to artificial binary images presented during the matching phase. By

analogy with the TDM (time-division multiplexing), each mask would correspond to one information channel of the TDM and the period T of neural oscillations would correspond to the frame length of the TDM. In the remaining of this section, the segmentation and matching phases are described in detail.

- *Segmentation*: In the segmentation stage, there is no connection between the two layers. The two layers act independently (unless for the influence of the global controller) and segment the two images applied to the two layers, respectively. The global controller forces the segments on the two layers to have different phases. At the end of this stage, the two images are segmented but no two segments have the same phase (Fig. 3). The results from segmentation are used to create binary masks that select one object in each layer in multi-object scenes.

The coupling strength S_{ij} for each layer as defined in Eq. (1) is computed by

$$S_{ij}(t) = \sum_{k,m \in N^{\text{int}}(i,j)} w_{ij,k,m}^{\text{int}}(t) H(x^{\text{int}}(k, m; t)) - \eta G(t), \quad (5)$$

where $G(t)$ is the influence of the global controller defined by the following equation. η should be set to a value smaller than the maximum value of synaptic weights, i.e. 0.2 in our case

$$G(t) = \alpha H(z - \theta), \quad (6)$$

$$\frac{dz}{dt} = \sigma - \zeta z, \quad (7)$$

where σ is equal to 1 if the global activity of the network is greater than a pre-defined ζ and is zero otherwise. ζ is a constant.

- *Dynamic matching*: In the matching phase, synchronization results from the segmentation phase are applied to the network.

Extra-layer connections (Eq. (4)) are established. If there are similar objects in the two layers, these extra-layer connections will help them synchronize. In other words, these two segments are bound together through these extra-layer connections [35]. In order to detect synchronization double-thresholding can be used [3]. This stage may be seen as a folded oscillatory texture segmentation device as the one proposed in [37]. The coupling strength S_{ij} for each layer in the matching phase is defined as follows:

$$S_{ij}(t) = \sum_{k,m \in N^{\text{ext}}(i,j)} \{w_{ij,k,m}^{\text{ext}}(t) H(x^{\text{ext}}(k, m; t))\} + \sum_{k,m \in N^{\text{int}}(i,j)} \{w_{ij,k,m}^{\text{int}}(t) H(x^{\text{int}}(k, m; t))\} - \eta G(t). \quad (8)$$

5. Geometrical interpretation of the ODLM

We know that an object can be represented by a set of points corresponding to its corners, and any affine transform is a map $T: \mathbf{R}^2 \rightarrow \mathbf{R}^2$ of these points defined by the following matrix operation:

$$\mathbf{p}' = \mathbf{A} * \mathbf{p} + \mathbf{t}, \quad (9)$$

where \mathbf{A} is a 2×2 non-singular matrix, $\mathbf{p} \in \mathbf{R}^2$ is a point in the plane, and \mathbf{p}' is its affine transform, \mathbf{t} is the translation vector. The transform is linear if $\mathbf{t} = \mathbf{0}$. Affine transformation is a combination of several simple mappings such as rotation, scaling, translation, and shearing. The similarity transformation is a special case of affine transformation. It preserves length ratios and angles while the affine transformation, in general does not. In this paragraph we show that the coupling $S_{i,j}$ is independent of the affine transform used. We know that any object can be shattered into its constituent triangles (three corners per triangle). Now suppose that the set $\{a, b, c, d\}$ is mapped to the set $\{T(a), T(b), T(c), T(d)\}$, and that the objects formed by these two sets of points are applied to the two layers of our neural network. Suppose also that points inside the triangle $\{a, b, c\}$ (resp., $\{T(a), T(b), T(c)\}$) have values equal to \mathbf{B} (corresponding to the gray-level value of the image at that points) and points inside $\{a, b, d\}$ (resp., $\{T(a), T(b), T(d)\}$) have values equal to \mathbf{A} .

We know that for an affine transform (Fig. 2):

$$\frac{\Delta_{abc}}{\Delta_{abd}} = \frac{\Delta_{T(abc)}}{\Delta_{T(abd)}}, \quad (10)$$

where Δ_{abc} is the area of the triangle $\{a, b, c\}$ (expressed in number of neurons). For $neuron_{i,j}$ belonging to $\{a, b, c\}$ and $neuron_{k,m}$ belonging to $\{T(a), T(b), T(c)\}$, Eq. (4) is equivalent to (neglecting the effect of intra-layer connections, since $N^{\text{ext}} \gg N^{\text{int}}$)

$$N^{\text{ext}} = \Delta_{T(abc)} + \Delta_{T(abd)}. \quad (11)$$

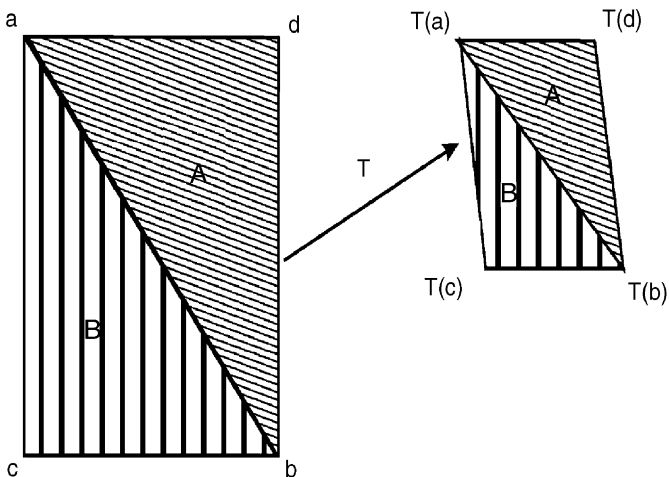


Fig. 2. An affine transform T for a four-corner object.

Hence,

$$w_{i,j,k,m}^{\text{ext}}(t) = \frac{f(p(i,j;t) - p(k,m;t))}{\Delta_{T(abc)} + \Delta_{T(abd)}} \quad (12)$$

with $f(x - y) = \frac{w_{\text{max}}^{\text{ext}}}{e^{2|x-y|}} \quad \forall x, y.$

There are $\Delta_{T(abc)}$ connections from the region with gray-level value \mathbf{B} (triangle $\{T(a), T(b), T(c)\}$) and $\Delta_{T(abd)}$ connections from the region with gray-level value \mathbf{A} (triangle $\{T(a), T(b), T(d)\}$) to the $neuron_{i,j}$ belonging to the triangle $\{a, b, c\}$ with gray-level value \mathbf{B} . Therefore, the external coupling for $neuron_{i,j}$ from all $neuron_{k,m}$ becomes

$$S_{i,j}(t) = \frac{\Delta_{T(abc)}f(\mathbf{A} - \mathbf{A})\psi(t, \phi_1)}{\Delta_{T(abc)} + \Delta_{T(abd)}} + \frac{\Delta_{T(abd)}f(\mathbf{A} - \mathbf{B})\psi(t, \phi_2)}{\Delta_{T(abc)} + \Delta_{T(abd)}} \quad (13)$$

$$\text{with } \psi(t, \phi) = H(x_{k,m}^{\text{ext}}(t)), \quad (14)$$

where $\psi(t, \phi_2)$ and $\psi(t, \phi_1)$ are, respectively, associated to spikes with phases ϕ_2 and ϕ_1 that appear after segmentation. After factorization and using Eq. (10) we obtain

$$S_{i,j}(t) = \frac{f(0)\psi(t, \phi_1)}{1 + \Delta_{abd}/\Delta_{abc}} + \frac{f(\mathbf{A} - \mathbf{B})\psi(t, \phi_2)}{1 + \Delta_{abc}/\Delta_{abd}}. \quad (15)$$

This means that the extra-layer connections are independent of the affine transform that maps the model to the scene (first and second layer objects) and can be extended to more than four points. Note that this is a more straightforward and less computationally complex affine invariance solution than the original DLM by Konen et al. [14] for which a matching between different scales of the gabor jets (used as feature extractors) should be used to obtain scale invariant recognition. If there are several objects in the scene and we want to match patterns, we can use the results from the segmentation phase to break the scene into its constituent parts (each synchronized region corresponds to one of the objects in the scene) and apply the objects one by one to the network, until all combinations are tested. This is not possible in the averaged DLM case where no segmentation occurs.

6. Rate coding vs. phase coding

The aim in this section is to show that the original DLM is a rate coding approximation of our proposed ODLM. In the following we illustrate the equivalence for layer 2 connected to layer 1. The same applies for layer 1 connected to layer 2.

Aoinishi et al. [4] have shown that a canonical form of rate coding dynamic equations solve the matching problem in the mathematical sense. The dynamics of a neuron in one of the layers of the original Dynamic Link Matcher proposed in [14] is as follows:

$$\frac{dx}{dt} = -\alpha x + (k * \sigma(x)) + I_x, \quad (16)$$

where $k(\cdot)$ is a neighborhood function (Mexican hat), I_x is the summed value of extra-layer couplings, σ is the sigmoidal function, x is the output of the rate coded neuron, and $*$ is the convolution. On the other hand, as mentioned in Section 2, we know that the Wang–Terman oscillator can be approximated by the integrate-and-fire neuron (for details see [11]), we can write for a single neuron of our network:

$$\frac{dx_a^{\text{two}}}{dt} = -x_a^{\text{two}} + \sum_{k,m \neq ij} w_{ij,k,m}^{\text{int}} H(x_{k,m}^{\text{two}}) + \sum_{k,m} w_{ij,k,m}^{\text{ext}} H(x_{k,m}^{\text{one}}) + H(p^{\text{input}})$$

$$\text{when } x > \text{threshold} \rightarrow x = 0, \quad (17)$$

where x^{two} stands for neurons in layer 2 and x^{one} stands for neurons in layer 1. There are synaptic connections (w^{int}) in layer 2 and synaptic connections from layer 1 to layer 2 (w^{ext}).

If we neglect the influence of intra-layer connections, therefore Eq. (17) becomes

$$\frac{dx_{ij}^{\text{two}}}{dt} = -x_{ij}^{\text{two}} + \sum_{k,m} w_{ij,k,m}^{\text{ext}} H(x_{k,m}^{\text{one}}) + H(p^{\text{input}})$$

$$\text{when } x_{ij} > \text{threshold} \rightarrow x_{ij} = 0. \quad (18)$$

Note that for an integrate-and-fire neuron the approximation $H(x) = x$ holds, since the output of an integrate-and-fire neuron is either 0 or 1 (it emits spikes or delta functions), therefore Eq. (18) can be further simplified to

$$\frac{dx_{ij}^{\text{two}}}{dt} = -x_{ij}^{\text{two}} + \sum_{k,m} w_{ij,k,m}^{\text{ext}} x_{k,m}^{\text{one}} + H(p^{\text{input}})$$

$$\text{when } x_{ij} > \text{threshold} \rightarrow x_{ij} = 0. \quad (19)$$

By averaging the two sides of Eq. (19) we get ($H(p^{\text{input}})$ is considered constant over T)

$$\frac{dx_a^{\text{two}}}{dt} = -x_a^{\text{two}} + \sum w^{\text{ext}} x_a^{\text{one}} + H(p^{\text{input}}),$$

$$x_a(t) = \langle x(t) \rangle_T = \frac{1}{T} \int_t^{t+T} x(t) dt, \quad (20)$$

where $\langle x(t) \rangle_T$, the averaged version of x over a time window of length T . For the sake of simplicity, the indices are omitted throughout Eqs. (20) to (24).

From [17], we know that the averaged output x_a^{two} of an integrate-and-fire neuron is related to the averaged-over-time inputs of a neuron ($\sum w^{\text{ext}} x_a^{\text{one}}$) by a continuous function (sigmoidal, etc.). Let name this function φ (note that β is a proportionality constant), we then get

$$\langle x_{ij}^{\text{two}} \rangle = \beta \varphi \left(\sum w^{\text{ext}} \langle x_{k,m}^{\text{one}} \rangle \right). \quad (21)$$

Note that in Eq. (20) we need $\langle x_{ij}^{\text{one}} \rangle$ in function of $\langle x_{k,m}^{\text{two}} \rangle$. Note further that Eq. (21) is a set of linear equations in w^{int}

and that we can deduce x_{ij}^{one} from that set of equations:

$$x_{ij}^{\text{one}} = \sum_{k,m} \sigma(x_{k,m}^{\text{two}}), \quad (22)$$

where $\sigma(x) = \varphi^{-1}(x)$. Replacing the above result in Eq. (20) gives (note that for the sake of simplicity we omitted again the indices)

$$\frac{dx_a^{\text{two}}}{dt} = -x_a^{\text{two}} + \sum \sum w^{\text{ext}} \sigma(x_a^{\text{two}}) + H(p^{\text{input}}). \quad (23)$$

On the other hand:

$$\sum \sum w^{\text{int}} \sigma(x_a^{\text{two}}) = k(x_a^{\text{two}}) * \sigma(x_a^{\text{two}}), \quad (24)$$

where $*$ is a 2-D convolution. Comparing Eqs. (16)–(23) and Eq. (24), we conclude that $H(p^{\text{input}})$ is equivalent to I_x , $\alpha = -1$, and the Mexican hat $k(\cdot)$ in Eq. (16) is replaced by the 2-D rectangular $k(\cdot)$ in Eq. (24).

Hence, we have proved that the DLM is an averaged-over-time approximation of the ODLM.

7. Discrete-event-based simulation of the ODLM

7.1. Discrete event modeling and simulation

Discrete-event models can be distinguished along at least two dimensions from traditional dynamic system models—how they treat passage of time (stepped vs. event-driven) and how they treat coordination of component elements (synchronous vs. asynchronous). Event-based simulation is inherently efficient since it concentrates processing attention on events—significant changes in states that are relatively rare in space and time—rather than continually processing component at every time step [41].

In continuous models, a simulation stepsize is defined at the beginning of the simulation and no matter how the dynamics of the system evolves, the update is done at each step. This is not an optimal way of simulation, since the update is performed even when there is not any change in the state of the system. In event-based simulation, we perform an update only at instants when we anticipate a change in the system.

Different DEVS toolboxes are proposed in the literature. For discrete-time event simulations a JAVA-based toolbox (DEVJAVA) is proposed by Zeigler and Sarjoughian [41], while a C++ based toolbox is proposed in [40]. For continuous-time event simulation an approach is proposed in [13].

7.2. Event-based simulation of neural networks

Our neural network is based on a home-made JAVA library of the DEVS (see also [9] for another implementation) and its home-made C++ version, which runs faster. Our simulator can handle up to 100,000 neurons on a Pentium 4-based computer with 512 MB of RAM. This performance is superior to the 10,000 neurons used in SimSPiNN [36] and to the 15,000 neurons used in [18]. In [7] a cluster of eight machines is used to simulate 10^6 neurons

(see also [29] for clustered simulations). Note that in all the techniques mentioned above connections are local, while in the ODLM connections between the two layers are global (fully connected). In this fully connected mode, the simulator can handle up to 300 million synaptic connections. We also use the shared weights technique as used in SpikeNET [8] to optimize connections. For discrete-event simulation of noisy and stochastic neurons see [27,18].

Our technique based on DEVS has some advantages in comparison with other proposed techniques. For example, in SpikeNET [8], a list of active neurons is kept in memory at each update. At each time step, only neurons that are on the list are updated and other neurons are left unchanged. Thus, if no neuron is active at a given time, no update occurs. Note that even if no update occurs at a given time, the simulator will still go

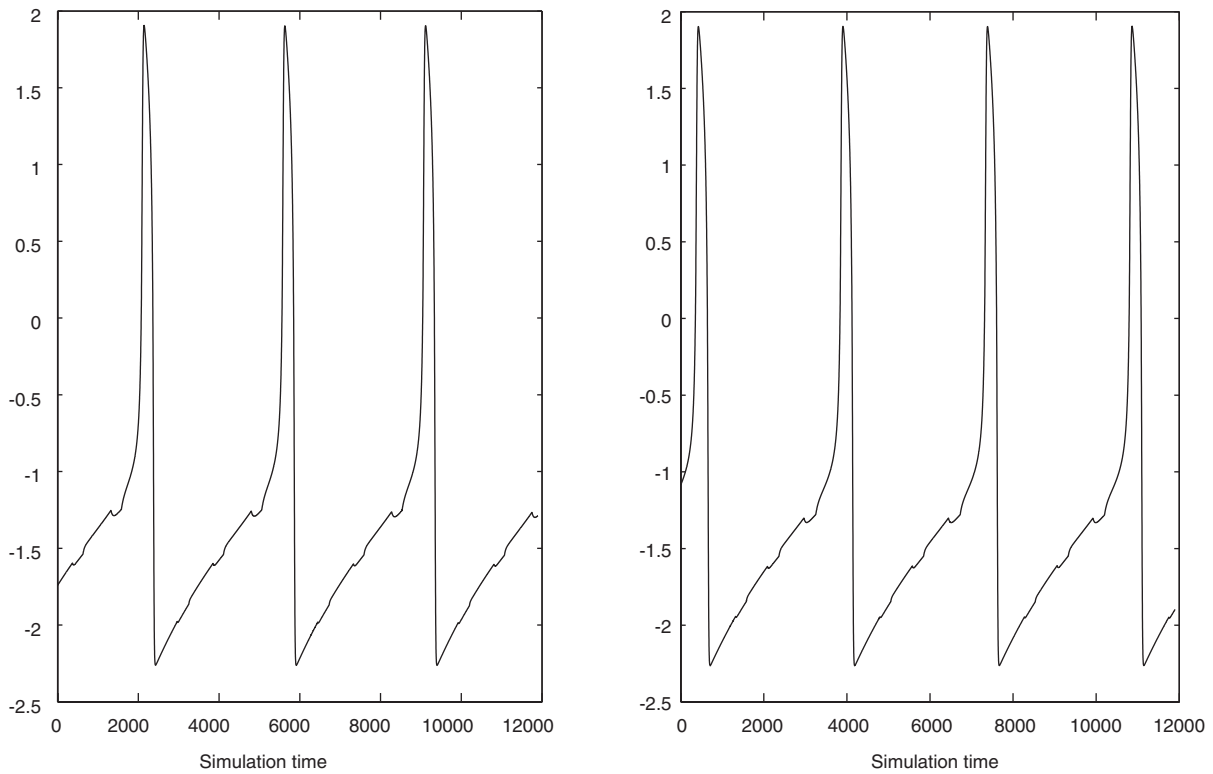


Fig. 3. Left: activity of one of the neurons associated with the triangle in the first layer after segmentation. Right: activity of one of the neurons associated with the background in the same layer.

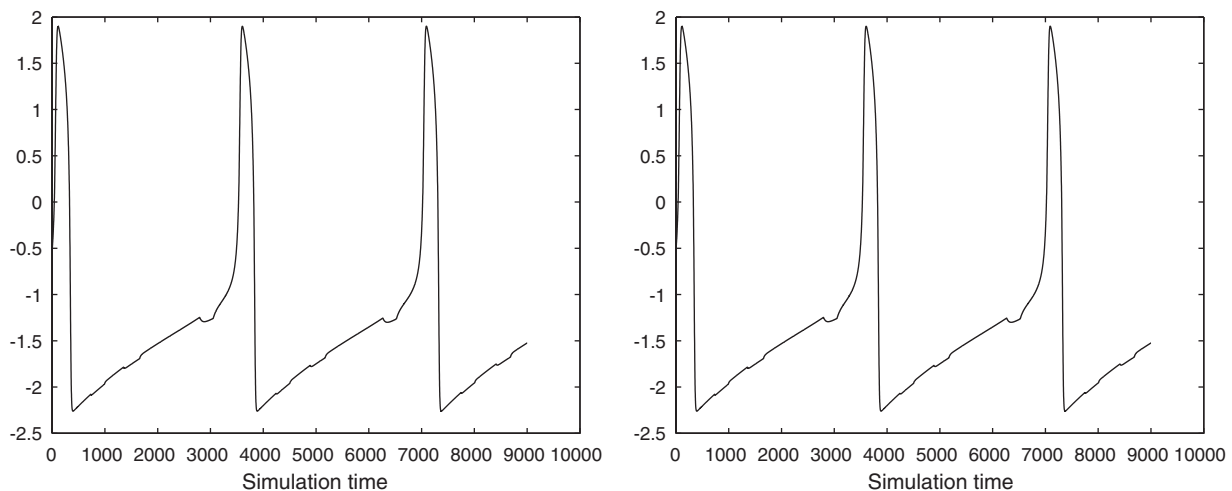


Fig. 4. Left: activity of one of the neurons associated with the triangle in the first layer after dynamic matching. Right: activity of one of the neurons associated with the triangle after dynamic matching in the second layer. Both neurons synchronize showing that they belong to the same type of object (related with an affine transform).

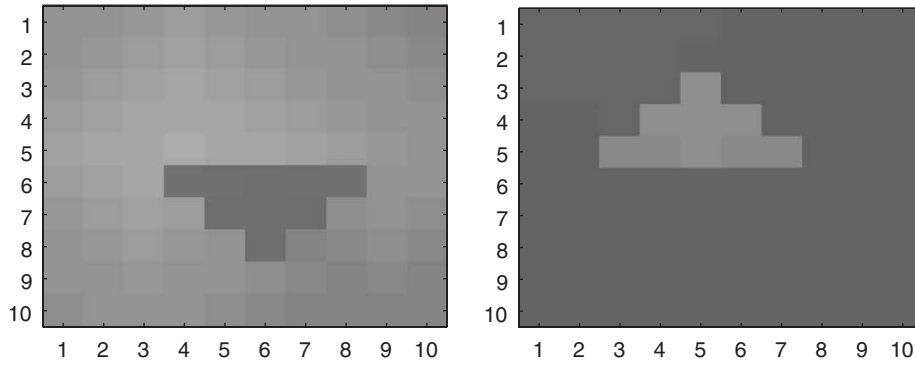


Fig. 5. Left: snapshot of neurons’ activities in the first layer when a triangle is applied during the segmentation phase. Right: snapshot of neurons’ activities in the second layer when a triangle is applied during the segmentation phase (the triangle in the second layer is translated and rotated).

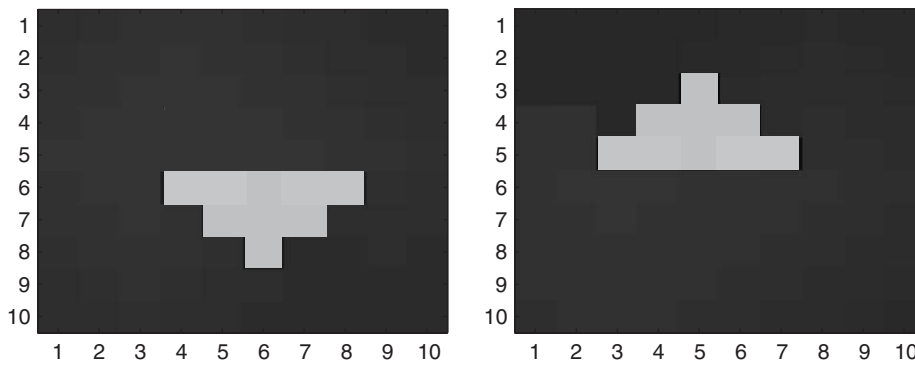


Fig. 6. Left: snapshot of neurons’ activities in the first layer when a triangle is applied during the matching phase. Right: snapshot of neurons’ activities in the second layer when a triangle is applied during the matching phase.

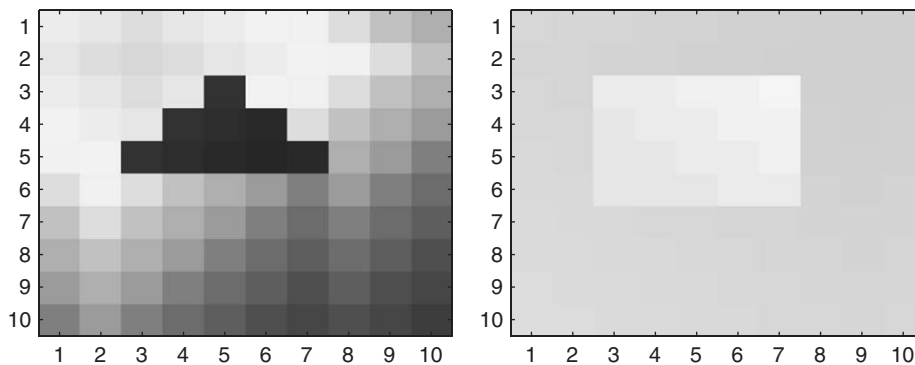


Fig. 7. Left: snapshot of neurons’ activities in the first layer when a triangle is applied during the matching phase. Right: snapshot of neurons’ activities in the second layer when a rectangle is applied during the matching phase. No synchronization occurs between the triangle in the first layer and the rectangle in the second layer.

through that time step, spending some time (even if it is minor) idling in that step. In DEVS, at each update the next expected update time is computed and put on an event list. The simulator directly goes to the time when the next update is supposed to be performed without idling in transitional steps (when no update is necessary). Therefore, DEVS more optimally manages the computational time.

7.3. DEVS simulation of the ODLM

To perform a DEVS simulation of the ODLM, the integrate-and-fire approximation of Eq. (2) is used. Roughly speaking, the neuron integrates its input until a threshold is reached. At this moment, the neuron emits a spike and the neuron’s internal potential is reset. If the input to the neuron is known at instant t , and nothing

changes between that instant and the instant at which a spike is emitted, we can predict the spiking time, i.e. the time at which the internal potential of a neuron crosses the threshold (for details see [29]).

8. Results

As stated earlier, this network can be used to solve the correspondence problem. For example, suppose that in a

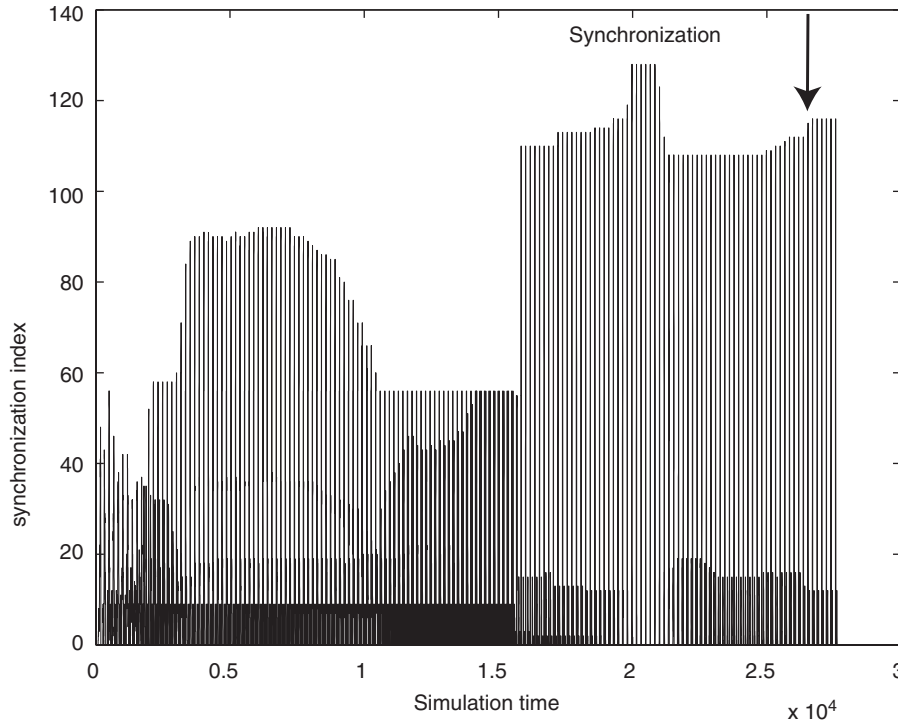


Fig. 8. The evolution of the thresholded activity of network through time when the segmentation precedes the matching phase. The synchronization takes 155 oscillations (spikes).

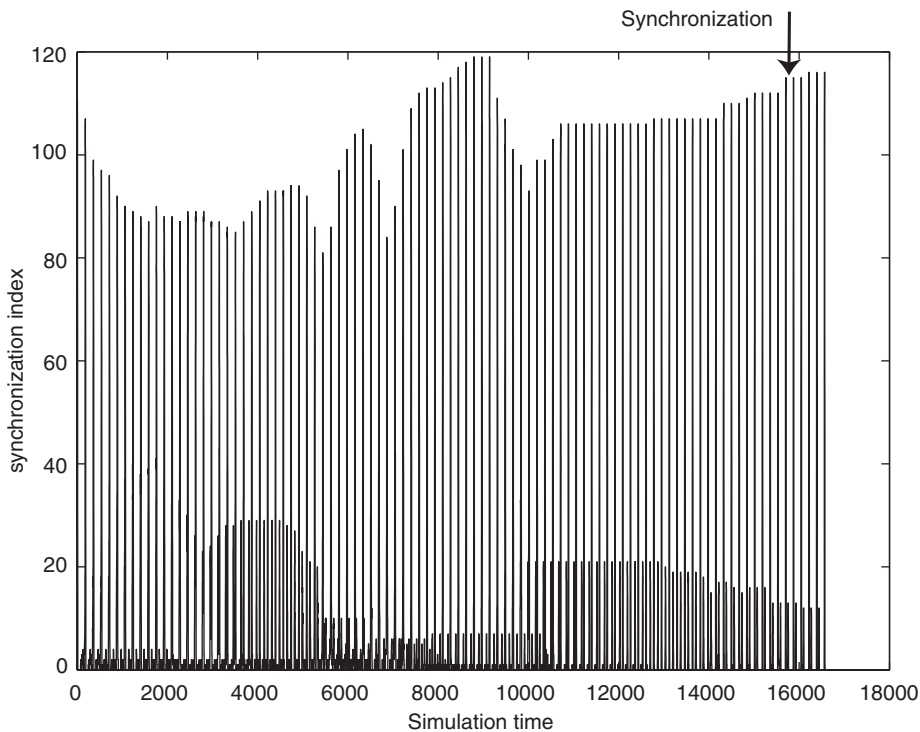


Fig. 9. The synchronization index of the vertical–horizontal bar scene when the segmentation step is bypassed. The synchronization takes 85 oscillations (spikes) or 15 600 simulation steps.

factory chain, someone wants to check the existence of a component on an electronic circuit board. All he/she has to do is to put an image of the component on the first layer and check for synchronization between the layers. Ideally, any change in the angle or the location of the camera or even the zoom factor should not influence the result. One of the signal processing counterparts of our proposed



Fig. 10. A Toy image for demonstrating the behavior of the ODLM in face recognition tasks.

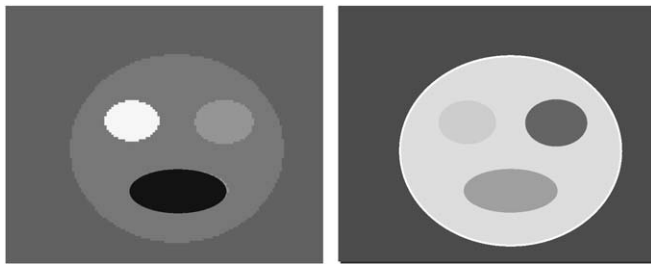


Fig. 11. Snapshot of the synchronized activity of layer 1 (left) and layer 2 (right) at the end of the segmentation phase. Each gray level represents a distinct phase. No two regions have the same gray level.

technique is the morphological processing. Other partial solutions such as the Fourier (resp., Mellin) transform could be used to perform matching robust to translation (resp., scaling). A more recent approach based on part-based representations is proposed in [1].

There is no need to train or configure our architecture to the stimulus we want to apply. The network is autonomous and flexible to not previously seen stimuli. This is in contrast with associative memory-based architectures that are limited in stocking capacity and in which stimuli must be applied and saved into memory before retrieval [33]. It does not require any pre-configured architecture adapted to the stimulus, like in the hierarchical coding paradigm [28]. DLM can play an important role in structuring memory, e.g. finding structural similarities between stored information during sleep [39].

8.1. Segmentation and matching for invariant pattern recognition of toy objects

A rectangular neuron map is chosen. There are 10×10 neurons in each layer. We use two objects for the test: a triangle and a rectangle. In this simple case, since the computational burden is low, the original relaxation oscillators are used and the simulation is continuous (not event-driven). In the first set of tests, a triangle is applied to the first layer and a triangle to the second layer. The two layers synchronize during the matching phase (Fig. 4) after segmentation (Fig. 3). However, when a triangle is applied to the first layer and a rectangle to the second layer no synchronization between the two layers is achieved during

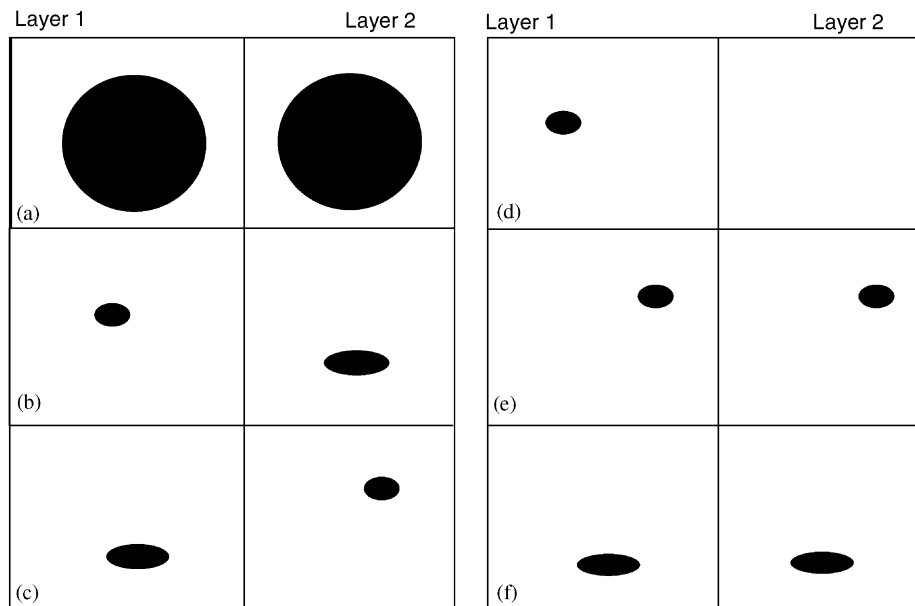


Fig. 12. Different matched regions of the smiley-like face. Each image consists of a snapshot of the activity of the first and second layers side-by-side: (a) visage; (b) and (c) eyes and mouth; (d) and (e) eyes; (f) mouth. Regions that popped out in the images are synchronized. Eyes and mouth synchronize according to the ODLM theory, since they have the same pixel intensity and are affine transform versions of each other. If the intensities were not the same (real-world images), synchronization between the mouth and the eye would not have happened.

the matching phase. This is due to the fact that no affine transform relates a rectangle to a triangle (Figs. 3 and 4).

Fig. 5 shows an activity snapshot (instantaneous values of $x(i,j)$) in the two layers after segmentation (first phase) for triangles. Note that same-colored neurons have similar spiking phases in the figure. On the other hand, different segments on different layers are desynchronized (see Fig. 5).

In the dynamic matching stage, similar objects among different layers are synchronized (Fig. 6). However, when the objects applied to the different layers are not linked by an affine transform, the objects on the two layers do not synchronize (Fig. 7).

8.1.1. Dynamical evolution of the ODLM for toy objects

The aim here is to show the dynamical evolution of the ODLM through time. For this purpose we used a 5×5 ODLM network with bars as objects to be recognized. A vertical bar is applied to the first layer and a horizontal bar to the second (a rotated version of the object on the first layer). In Fig. 8, the dynamical evolution of the network is shown when the segmentation precedes the matching. In another experiment the segmentation phase has been bypassed (Fig. 9). Results show that the synchronization time is longer when the two phases (segmentation and matching) are done separately (155 oscillations compared to 85 oscillations).

8.2. Face recognition with ODLM

8.2.1. Toy images

A smiley-like binary image is applied to the two layers of the network. The image consists only of black and white regions and no adjacent regions have the same intensity (Fig. 10). The image is applied to the first and second layer of our proposed ODLM network. During the segmentation phase regions synchronize and no two regions have the same phase (Fig. 11).

During the matching phase (Fig. 12), similar regions synchronize. Note that the eyes synchronize with the mouth. This is due to the fact that a simple affine transform maps the two ellipses (eyes and mouth). This is in total accordance with the theory described in Section 5. Note also that this phenomenon happened just because the two ellipses (eye and mouth) have the same intensity, which is not the case for real-world image where lips and eyes do not have the same color. In fact, the circle corresponding to the face did not synchronize with eyes or mouth, since they have different intensities.

8.2.2. Real-world face images

Results for simple objects described in the previous subsections led us to undertake experiments with more complicated scenes and objects. For this purpose, the ColorFERET database of facial images collected under the FERET program [21] is used. The approach described in this paper is different from the general framework of face



Fig. 13. Left and middle: two photo shots of the same person from the FERET database. Note the different facial expression between the two pictures. Right: a shot from another person in the FERET database (image 00002_930831_fa). Tags on the left top of pictures are only added for referencing purposes and are not present when simulations have been undertaken.

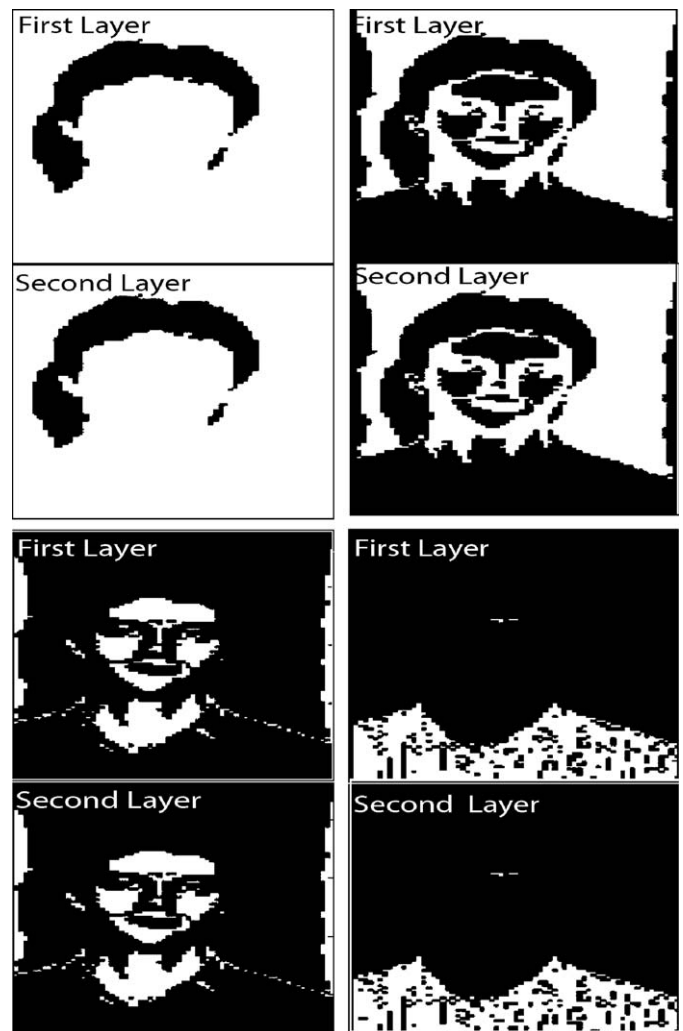


Fig. 14. Four different snapshots (binary masks) of the activity (taken at four different instants) of the two layers for the image 00002_930831_fa of the FERET database. The intensity level of each pixel indicates its output activity level. Regions of the same intensity level shows synchronized zones. Layer 1 and layer 2 are synchronized. Image segmentation and matching have been performed simultaneously. (Note that snapshots are flipped over their vertical axis of symmetry compared to photos shown in Fig. 13).

recognition, in which localization and normalization are performed prior to recognition [21]. One can see the ODLM as a system that can achieve the three goals of localization, normalization and recognition at the same time. Note that the coordinates of the centers of the eyes and other important features of the face are not given, as is the case for partially automatic algorithms.

The goal in this section is to compare an image of an unknown individual (called a “probe”) with a set of known images (called the “gallery”). The algorithm is supposed to be robust to geometric transformations of the image (i.e. rotation, symmetry, etc.) and to different facial expressions (i.e. laugh, grimace, etc.) (Fig. 13).

Three sets of experiments have been undertaken. In the first set, the same picture is applied to the two layers (Fig. 14). Segmentation in each layer is achieved and segmented regions are matched among the two layers. This means that similar regions in different layers have the same synchronization phase. In the second set, two pictures of the same person with different facial expressions have been applied to the network (Fig. 15). For the sake of simplicity, segmentation and matching are done in one shot as described in Section 8.1.1. Matching is achieved in all parts except for the region encircling the lips. We therefore define the matching ratio as follows:

$$r = \frac{S_m}{S_t} \times 100\%, \quad (25)$$

where S_m is the total surface of matched regions and S_t is the total surface of pictures. We assign this ratio to the confidence by which we identify a person. In the case of Fig. 15, this ratio is equal to 95%. Therefore, the person has been identified with a confidence of 95%. In the third set of experiments, pictures from different persons have been applied to the network. No matching has been achieved. Hence, the photo on the second layer does not represent the same person as the one on the first layer.

9. Conclusion and further work

We proposed the oscillatory DLM as a mean to segment images and solve the correspondence problem, as a whole system, using a two-layered oscillatory neural network. We showed that our network is capable of establishing correspondence between images and is robust to translation, rotation, noise and homothetical transforms. More experiments with complex objects and more general transforms like shearing, etc. are under investigation. Pattern recognition of occluded objects is another challenge for this proposed architecture and will be presented in further works. In order to increase the computational speed a pre-processing based on part-based representations [1,16,20] is under investigation.

We are investigating the possibility of the insertion of this architecture in our bottom-up sound segregator [23,24,30,25] as a top-down processor. In fact, in this application, visual images will be replaced by CAM (Cochleotopic/AMtopic) and CSM (Cochleotopic/Spectrotopic) Maps proposed in [23]. The approach could also be used as a separate discrete-word recognizer [31].

The introduction of noise in the neural dynamics is another aspect that will be investigated in the future. The discrete-event simulation of noisy neurons is more tricky [27,18] and should be adapted to our network.

Acknowledgments

Many thanks to DeLiang Wang for fruitful discussions on the dynamics of the relaxation oscillators. The authors would also like to thank Stéphane Loiseau and Hoang Hai

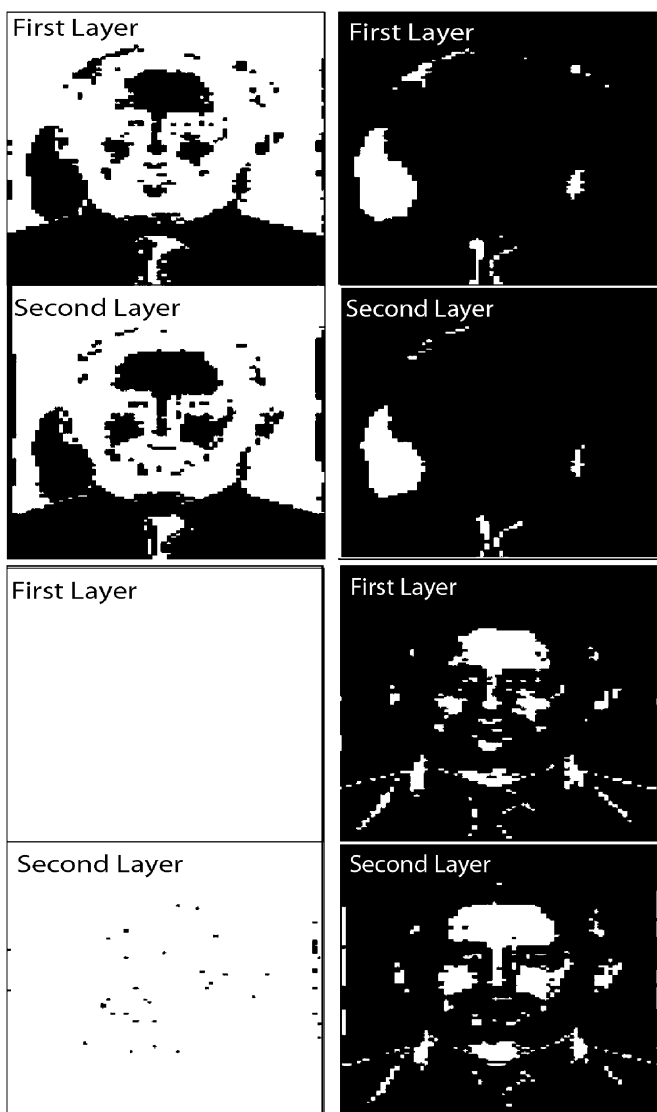


Fig. 15. Four different snapshots (binary masks) of the activity of the two layers for the two images 0001_930831_fa_a and 0001_930831_fb_a of the same person with different facial expressions (FERET database). Segmentation and matching have been performed accurately even if the visages are not exactly the same. Due to differences in facial expression and pose, some neurons are not synchronized among the two layers (as seen at the bottom left part of the picture).

Anh for discussions and implementations. This work was partly supported financially by NSERC, and the University of Sherbrooke.

References

- [1] S. Agarwal, A. Awan, D. Roth, Learning to detect objects in images via a sparse, part-based representation, *IEEE Trans. Pattern Anal. Mach. Intell.* 26 (2004) 1475–1490.
- [2] S.I. Amari, Dynamical study of formation of cortical maps, *Biol. Cybernet.* (1977) 77–87.
- [3] H. Ando, N. Takashi Morie, M. Nagata, A. Iwata, A nonlinear oscillator network circuit for image segmentation with double-threshold phase detection, in: *ICANN 99*, 1999.
- [4] T. Aoinishi, K. Kurata, T. Mito, A phase locking theory for matching common parts of two images by dynamic link matching, *Biol. Cybernet.* 78 (4) (1998) 253–264.
- [5] R.M. Borisjuk, Y. Kazanovich, Oscillatory neural network model of attention focus formation and control, *Biosystems* 71 (2003) 29–36.
- [6] E. Cemel, D. Wang, Motion segmentation based on motion/brightness integration and oscillatory correlation, *IEEE Trans. Neural Networks* 11 (4) (2000) 935–947.
- [7] E. Claverol, A. Brown, J. Chad, Scalable cortical simulations on Beowulf architectures, *Neurocomputing* 47 (2002) 277–297.
- [8] A. Delorme, J. Gautrais, R. VanRullen, S.J. Thorpe, Spikenet: a simulator for modeling large networks of integrate and fire neurons, *Neurocomputing* 26–27 (1999) 989–996.
- [9] J.B. Filippi, P. Bigambiglia, The JDEVS environmental modeling and simulation environment, *Environ. Modeling Software*, to appear.
- [10] K. Fukushima, A neural network model for selective attention in visual pattern recognition, *Biol. Cybernet.* (1986) 5–15.
- [11] W. Gerstner, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*, Cambridge University Press, Cambridge, MA, 2002.
- [12] L.E. Gordon, *Theories of Visual Perception*, Wiley, New York, 1997.
- [13] E. Kofman, M. Lapadula, E. Pagliero, Quantized state systems. A DEVS approach for continuous systems simulation, *Trans. SCS (The Society for Modeling and Simulation International)* 18 (3) (2003) 123–132.
- [14] W. Konen, T. Maurer, C. Von der Malsburg, A fast dynamic link matching algorithm for invariant pattern recognition, *Neural Networks* (1994) 1019–1030.
- [15] Y. LeCun, F.J. Huang, L. Bottou, Learning methods for generic object recognition with invariance to pose and lighting, in: *IEEE Computer Vision and Pattern Recognition Conference*, Washington, USA, 2004.
- [16] D.D. Lee, H.S. Seung, Learning the parts of objects by non-negative matrix factorization, *Nature* 401 (1999) 788–791.
- [17] W. Maass, C.M. Bishop, *Pulsed Neural Networks*, MIT Press, Cambridge, MA, 1998.
- [18] M. Mattia, P.D. Giudice, Efficient event-driven simulation of large networks of spiking neurons and dynamical synapses, *Neural Comput.* 12 (2000) 2304–2329.
- [19] B. Moghaddam, A. Pentland, Probabilistic visual learning for object detection, *IEEE Trans. Pattern Anal. Mach. Intell.* 17 (1997) 696–710.
- [20] B.A. Olshausen, D.J. Field, Emergence of simple-cell receptive field properties by learning a sparse code for natural images, *Nature* 381 (1996) 607–609.
- [21] P.J. Philipps, H. Moon, P.J. Rauss, S. Rizvi, The FERET evaluation methodology for face recognition algorithms, *IEEE Trans. Pattern Anal. Mach. Intell.* 22 (2000) 1090–1104.
- [22] R. Pichevar, Neural and anthropomorphic processing of the “Cocktail Party” effect and its applications in information technology, Ph.D. Thesis, University of Sherbrooke, 2004.
- [23] R. Pichevar, J. Rouat, Cochleotopic/AMtopic (CAM) and Cochleotopic/Spectrotopic (CSM) map based sound source separation using relaxation oscillatory neurons, in: *IEEE Neural Networks for Signal Processing Workshop*, Toulouse, France, 2003.
- [24] R. Pichevar, J. Rouat, A Quantitative Evaluation of a Bio-inspired Sound Segregation Technique for Two- and Three-source Mixtures Sounds, *Lecture Notes in Computer Science*, vol. 3445, Springer, Berlin, 2005, pp. 430–435.
- [25] R. Pichevar, J. Rouat, C. Feldbauer, G. Kubin, A bio-inspired sound source separation technique in combination with an enhanced FIR gammatone analysis/synthesis filterbank, in: *EUSIPCO Vienna*, 2004.
- [26] E.O. Postma, H.J. Van der Herik, P.T.W. Hudson, SCAN: a scalable neural model of covert attention, *Neural Networks* 10 (1997) 993–1015.
- [27] J. Reutimann, M. Giugliano, S. Fusi, Event-driven simulation of spiking neurons with stochastic dynamics, *Neural Comput.* (2003) 811–830.
- [28] M. Riesenhuber, T. Poggio, Are cortical models really bound by the binding problem?, *Neuron* 24 (1999) 87–93.
- [29] O. Rochel, A discrete-event approach to the simulation of spiking neural networks (in French), Ph.D. Thesis, LORIA, 2003.
- [30] J. Rouat, R. Pichevar, Source separation with one ear: Proposition for an anthropomorphic approach, *EURASIP J. Appl. Signal Process.* 9 (2005) 1365–1373.
- [31] J. Rouat, R. Pichevar, S. Loisselle, Auditory Scene Analysis in Link with Non-linear Speech Processing and Spiking Neural Networks, *Lecture Notes in Computer Science*, vol. 3445, Springer, Berlin, 2005, pp. 351–371.
- [32] J.G. Taylor, Neural bubble dynamics in two dimensions: foundations, *Biol. Cybernet.* (1999) 5167–5174.
- [33] T. Vinh Ho, J. Rouat, Novelty detection based on relaxation time of a network of integrate-and-fire neurons, in: *Proceedings of IEEE International Joint Conference on Neural Networks*, Alaska, USA, 1998.
- [34] J. Vitay, N. Rougier, Using Neural Dynamics to Switch Attention, *IJCNN*, Montreal, Canada, 2005.
- [35] C. Von der Malsburg, The what and why of binding: the modeler’s perspective, *Neuron* (1999) 95–104.
- [36] M. Walker, H. Wang, S. Kartamihardjo, U. Roth, SimSPINN—a simulator for spike-processing neural networks, *TIMACS’97*, 1997.
- [37] D.L. Wang, D. Terman, Image segmentation based on oscillatory correlation, *Neural Comput.* (1997) 805–836.
- [38] L. Wiskott, T. Sejnowski, Slow feature analysis: unsupervised learning of invariances, *Neural Comput.* (2002) 715–770.
- [39] L. Wiskott, C. Von der Malsburg, A. Weitzenfeld, *The Neural Simulation Language: A System for Brain Modeling*, MIT Press, Cambridge, MA, 2002, pp. 343–372 (Chapter 18).
- [40] B. Zeigler, *Object Oriented Simulation with Hierarchical, Modular Models: Selected Chapters Updated for DEVS-C++*, 1990.
- [41] B. Zeigler, H. Sarjoughian, *Theory of Modeling and Simulation*, Academic Press, New York, 2000.
- [42] X. Zhang, A. Minai, Detecting corresponding segments across images using synchronizable pulse-coupled neural networks, in: *IJCNN2001*, 2001.
- [43] W. Zhao, R. Chellappa, A. Krishnaswamy, Discriminant analysis of principal components for face recognition, in: *Third International Conference on Automatic Face and Gesture Recognition*, 1998, pp. 336–341.



Ramin Pichevar was born in March 1974, in Paris, France. He received his bachelor of science degree in electrical engineering (electronics) in 1996 and his master of science in electrical engineering (telecommunication systems) in 1999, both in Tehran, Iran. He received his Ph.D. in electrical and computer engineering from Université de Sherbrooke, Québec, Canada in 2004. During his Ph.D., he taught courses on signal processing and computer hardware as a

lecturer. In 2001 and 2002 he performed two summer internships at Ohio State University (USA) and at the University of Grenoble (France), respectively.

From November 2004 to July 2006, he was a postdoctoral and research associate in the computational neuroscience and signal processing laboratory at the University of Sherbrooke under an NSERC (Natural Sciences and Engineering Council of Canada) Idea to Innovation (I2I) grant.

He is presently a research engineer at the Communications Research Center (CRC), Ottawa, Canada. His domains of interest are signal processing, Computational Auditory Scene Analysis (CASA), neural networks with emphasis on bio-inspired neurons, speech recognition, audio and speech coding, sparse coding, digital communications, discrete-event simulation, and image processing.



Jean Rouat holds a master degree in Physics from Univ. de Bretagne, France (1981), an E. & E. master degree in speech coding and speech recognition from Université de Sherbrooke (1984) and an E. & E. Ph.D. in cognitive and statistical speech recognition jointly from Université de Sherbrooke and McGill University (1988). From 1988 to 2001 he was with the Université du Québec à Chicoutimi (UQAC). In 1995 and 1996, he was on a sabbatical leave with the Medical Research Council, Applied Psychological Unit, Cambridge, UK and the Institute of Physiology, Lausanne, Switzerland. In 1990 he

founded the ERMETIS, Microelectronics and Signal Processing Research Group from UQAC. He is now with Université de Sherbrooke where he founded the Computational Neuroscience and Signal Processing Research group. He regularly acts as a reviewer for speech, neural networks and signal processing journals. He is an active member of scientific associations (Acoustical Society of America, Int. Speech Communication, IEEE, Int. Neural Networks Society, Association for Research in Otolaryngology, ACM, etc.). He is member of the IEEE technical committee on Machine Learning for Signal Processing.



Le Tan Thanh Tai graduated with a B.Eng. in Information Technology in 2001, from the Polytechnic University of Ho Chi Minh city, Vietnam. From 2001 to 2002, he worked as a developer, specialized in Windows programming. He received his master degree in Computer Science in 2005, from the French-Speaking Institute of Informatics (Institut de la Francophonie pour l'Informatique) and the Polytechnic University of Ha Noi, Vietnam.

In 2004, he was on internship in the “Signal Processing and Computational Neuroscience Group” at the Université de Sherbrooke, Quebec, Canada, under the supervision of professor Jean Rouat.

Since 2005, he has been working as a team leader, focused on the Web services and SOA (Service-Oriented Architecture) management spaces at Systinet Corporation, Vietnam.