Introducing Visual Latencies into Spin-Lattice Models for Image Segmentation: A Neuromorphic Approach to a Computer Vision Problem

Ralf Opara and Florentin Wörgötter
Dept. of Neurophysiology, Ruhr-Universität Bochum,
D-44780 Bochum

Abstract

In this study we will show how an algorithmic principle which might play a role in information processing in the brain of higher vertebrates - the so called visual latencies - can be transferred with high efficiency to a model system which is better suited for implementation on conventional computer hardware. To this end we assign luminance dependent temporal delays (latencies) to the individual pixels of an image. This temporal structure of the input data stream then accelerates and improves the relaxation of a spin-lattice labeling algorithm for scene segmentation.

1. Introduction

The goal of neuromorphic modeling is to bridge the gap between the complex information processing principles in the brain and their technical application in most cases on conventional computers. The basic "hardware" of both domains differs vastly such that a direct transfer of algorithms between both is almost always impossible. Thus, a successful technologically relevant application of an "algorithm taken from the brain" requires the extraction of the underlying mechanisms, the reduction to their basic algorithmic principles and finally the adaptation to the computer problem and the employed hardware.

The task of image segmentation may serve as an example. In the brain evidence exists that synchronization between the activity of neurons could subserve image segmentation [3], [6], [9]. On a conventional computer one would, however, rather not emulate the complex firing dynamics of nerve cells in order to segment a scene. Instead other more efficient labeling algorithms are employed and only the biological principle of sharing a label (i.e., "being synchronized") still indicates the initially existing relation between image segmentation in the brain and on the computer.

In a similar way, the objective of this study is to adapt a neuronal algorithm to the requirements of a computer vision problem. The physiological background we take into consideration is based on the fact that every sensorial neuron only responds after a certain delay (called "latency") to a stimulus [8], [1], [13]. In the visual system the latency of a neuron is shorter if the stimulus luminance increases. Thus, the population response of a group of neurons reacting to a visual scene containing objects of different luminance will get spread out in time. Recently we have shown experimentally and by means of an artificial neural network that this initial temporal spread of activity introduced by the response latencies could be used to improve segmentation of a scene [17], [12]. Obviously, the dynamics and the architecture of an artificial spiking neural network cannot be smoothly represented on regular hardware. On the other hand, here we show that the principle of *latency induced temporal spread of image information* can be introduced with high efficiency into spin-lattice models for scene segmentation, which are much better adapted to common computer architectures.

We will first give an overview of the model. Then we will show an example of how the system segments a visual scene and finally we will quantify the convergence speed of the algorithm.

2. The Model

The schematic diagram of our model is shown in Fig. 1. It consists of two main parts.

The first part of the system (Fig. 1) contains a luminance dependent delay line. Pixels with a high luminance are processed earlier than those with a low luminance. The temporal structure of this data stream determines the interaction between spins within the

0-7803-4122-8/97 $10.00

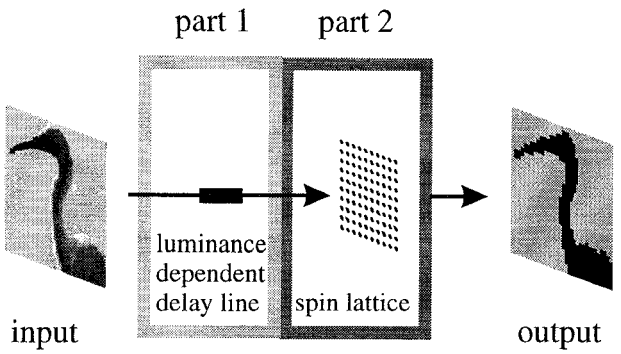

Figure 1. Schematic diagram of the system.

second part of the system. The second part consists of spins which are arranged on a two dimensional lattice of size $N = n_x n_y$. Each spin can take q different labels [5], [14]. Within this spin lattice the algorithm applied tends to assign the same label to spatially adjacent data points that reach the spin lattice with high temporal coherence (same latency) while data points with strong delays between them will get different labels. This is realized by minimizing the energy function which describes the system (Eq. 1). Two spins i and j are interacting with a coupling strength w_{ij}. The coupling strength w_{ij} depends on the temporal coherence of pixels i and j in the input data stream. The coupling strength is high within an object (high temporal coherence) while it is low or negative between objects (low temporal coherence).

The energy of the system is a function of the label configuration and the connection strength between spins.

$$E(t) = \sum_{i=1}^{N} [\sum_{j \in N_i} -w_{i,j}\delta_{\sigma_i\sigma_j}\eta(\Delta t_i)\eta(\Delta t_j)] + C(\sigma_i) \quad (1)$$

σ_i,σ_j	:	labels of spin i and j,
$w_{i,j}$	:	connection strength between spins at location i and j,
$\delta_{\sigma_i\sigma_j}$	:	kronecker function,
N_i	:	neighborhood of spin i,
$C(\sigma_i)$	:	global cost function for label σ_i,
$\eta(\Delta t_i)$	:	binary coupling term. Interaction of

two spins i and j is only possible if the time is larger than the latencies of both spins,
i.e., $\eta(\Delta t_i) = 0$ if $t - t_{lat}(i) < 0$ else $\eta(\Delta t_i) = 1$.

There are several approaches to minimize an energy function. Most approaches use local update algorithms [4], [5]. Local update algorithms are easy to implement, but converge slowly, because only single spins are updated. To avoid local minima a new spin configuration is only accepted with a certain probability (Metropolis algorithm [11])

$$P_{Metropolis} = min(1, \exp -\Delta E/T), \quad (2)$$

where T is the temperature of the system and ΔE the energy difference between the old and new configuration. At high temperatures ($T \to \infty$) all configurations are accepted, while at low temperatures ($T \to 0$) a new configuration is only accepted, if the energy of the system is decreased. The annealing schedule starts with a high temperature and for every iteration the temperature is decreased according to $T_{k+1} \geq C/log(1+k)$ [4],[5]. If the temperature is decreased too fast or the initial temperature is chosen too small, the system will reach only a local minimum.

Thus, the problems with local update algorithms are twofold: 1. A large number of iterations is needed to guarantee the convergence to a global minimum and 2. Very many iterations are necessary to escape from a local minimum for which already a large number of uniform labels has been assigned, because all those spins have to be flipped individually.

In order to circumvent these problems we use a different approach which is similar to cluster update algorithms known in statistical physics [15], [16]. Cluster update algorithms have the advantage, that large clusters of spins can be flipped simultaneously. Thus, to escape from a local minimum with large number of similar spins only one iteration is necessary.

The dynamic of the used cluster update algorithm is the following: Clusters can only be formed between spins which are in the same state (same label). The probability that two spins are bound together to one cluster depends on the temperature T and the coupling strength $w_{i,j}$ of the two spins and is given by:

$$P_B(i,j) = \eta(\Delta t_i)\eta(\Delta t_j)(1 - exp[-w_{i,j}\delta_{\sigma_i\sigma_j}/T]) \quad (3)$$

At high temperatures the average cluster size is small ($w_{i,j}/T \to 0$) while at lower temperatures the possible cluster size increases. The clusters, calculated according to Eq. 3, are flipped with a certain probability depending on the energy gain of the system (see above).

3 Results

As an example Fig. 2 shows snapshots of the label distributions of the model if a stimulus (Fig. 2, upper left) is given to the system. The stimulus consists of a 128x128 image, containing a paraglider, a shaded sky and some hills. The panels show the label distribution of the system at different times ($t \geq 0$). The labels are coded as gray values. Spins with the same label (same

gray value) belong to one object, while different gray values indicate the assignment to different objects.

The simulation starts with a random label configuration and each label is represented by nearly the same number of spins.

During the first iterations only the brightest objects (small latency) are processed (paraglider and lower part of the sky, Fig. 2 iteration 0-2). According to the competition included in the dynamics of the system the spins of the paraglider will receive the same label while spins representing the sky will get a different label.

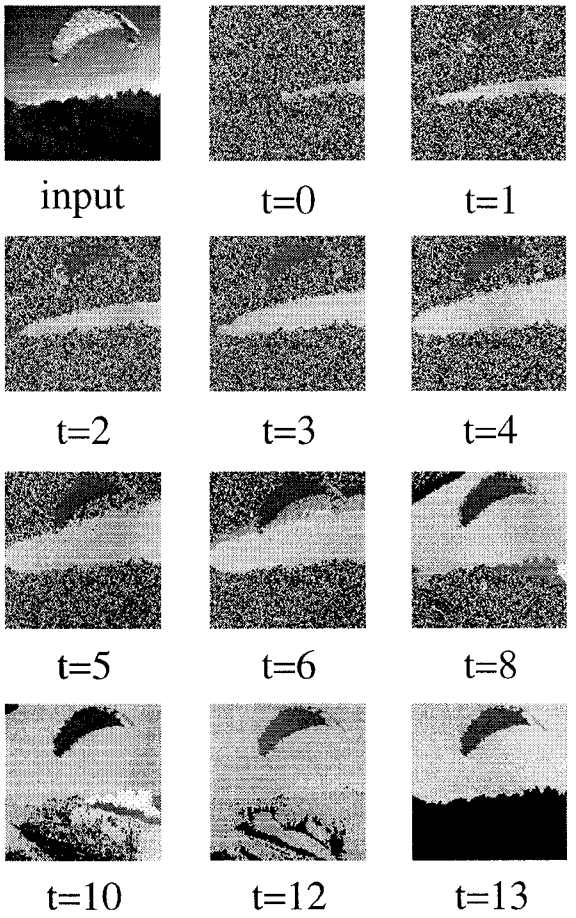

Figure 2. Segmentation result.

Due to the latency processing of the parts with a lower luminance does not start before iteration 8. After iteration 13 the whole image is segmented into four different areas.

The efficiency of the introduced algorithm can be judged from iterations 12 and 13, where a surface of nearly 40*128 pixels (hills) is flipped during only one iteration. As mentioned above local update algorithms on the other hand would need numerous updates and a careful annealing schedule to achieve this.

The next figure (Fig. 3) shows the influence of the latency differences on the convergence speed of the system. For this test a rather simple image is used (Fig. 3, inset), containing four square objects. The squares have different gray values which lead to different total latencies t_{lat}.

The brightest object is always processed at iteration zero. The second object is processed at iteration $t_{lat} = 1 \times L$ and the third object at iteration $t_{lat} = 2 \times L$ and so on. The unit-latency L is varied between 0 and 14 iterations (abscissa). With a unit-latency of $L = 0$ all objects are processed simultaneously.

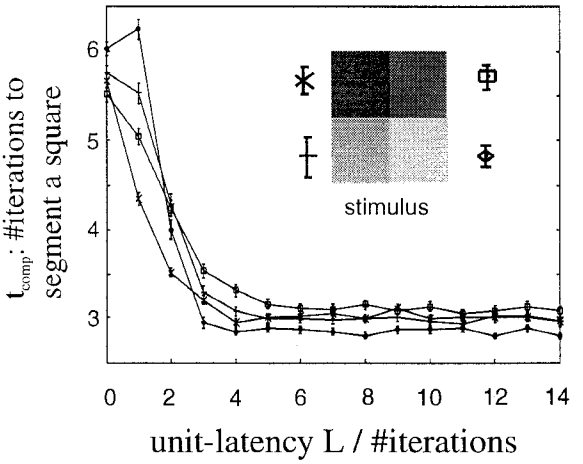

Figure 3. Convergence speed of the algorithm.

In 200 simulations the average computational time (t_{comp}, in number of iterations) is determined to segment a certain square and is plotted on the y-axis. We define $t_{comp} = t_{seg} - t_{lat}$, where t_{seg} is the actual iteration time reached when a given square is completely segmented and t_{lat} the total latency for this square. We use this particular measure, because at iterations $t < t_{lat}$ nearly no computer time is allocated for processing of that particular square.

In figure 3 one can see that at $L = 0$ the averaged number of iterations to segment a square is nearly the same for all four objects ($t_{comp} \approx 5.7$). With increasing latency the number of iterations necessary for the segmentation is decreased for all squares, until a plateau is reached ($t_{comp} \approx 3.1$). The number of iterations until a square is segmented is nearly reduced by 50% as compared to $L = 0$. The brightest square benefits from the

latency mechanism, because the relaxation of its spins is not disturbed by those of the other objects during the first iterations. It wins the competition for a certain label, because the spins of the other objects are still in disorder. The darker objects, on the other hand, take advantage in the ordered state of the brighter object, because in the dark objects all labels are punished that are already occupied by the bright object, leading to a faster convergence.

4 Discussion

Earlier approaches that utilize visual latencies have been described by Burgi and Pun [2]. In the filter model proposed by these authors latency is used in order to reduce the number of data points. Only the most relevant features and areas of the input image are processed, while the others are not. In the course of our work several steps have been undertaken to provide evidence that visual latencies can be used to improve image segmentation. To this end we have analyzed neuronal responses in the visual cortex by means of visually evoked potentials and demonstrated that this bulk neuronal signal synchronizes only after the visually induced latency [17]. In an artificial neural network latencies could be used to raise the number of discriminable objects and to improve the segmentation of simple scenes [12]. Only by the transfer of this algorithmic principle to a spin-lattice model, however, it was possible to achieve a high processing speed and a sufficient accuracy to be able to segment real images in a robust way.

Furthermore we were able to show, that cluster update algorithms can successfully be applied to an image segmentation task. These algorithms are well established in statistical physics to avoid the critical slowing down at a phase transition of spin lattices. Therefore they are well suited to overcome problems which are common in local update algorithms.

Acknowledgements: The authors acknowledge the support of the Deutsche Forschungsgemeinschaft (grants Wo 388/4-2, 5-2, 6-1).

References

[1] Bolz, J., Rosner, G. and Wässle, H. Response latency of brisk-sustained (X) and brisk-transient (Y) cells in the cat retina. H. *J. Physiol*, **328**, 171-190 (1982).

[2] Burgi, P. Y. and Pun, T. Asynchrony in image analysis: using the luminance-to-response-latency relationship to improve segmentation. *J. Opt. Soc. Amer. A* **11**/6, 1720-1726, (1994).

[3] Eckhorn, R., Frien, A., Bauer, R., Woelbern, T. & Kehr, H. High frequency (60-90 Hz) oscillations in primary visual cortex of awake monkey. *NeuroReport*, **4**, 243-246 (1993).

[4] Geman, S., Geman, D. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Trans. Pattern Analysis Machine Intelligence*, **6**, 721-741, (1984).

[5] Geman, D., Geman, S., Graffigne, C. & Dong, P. Boundary detection by constrained optimization. *IEEE Trans. Pattern Analysis Machine Intelligence*, **12**(7), 609-628, (1990).

[6] Gray, C.M., König, P., Engel, A.K. & Singer, W. Oscillatory responses in cat visual cortex exhibit intercolumnar synchronization which reflects global stimulus properties. *Nature*, **338**, 334-337 (1989).

[7] Hopfield, J. J. Pattern recognition computation using action potential timing for stimulus representation. *Nature*, **376**, 33-36, (1995).

[8] Levick, W.R. Variation in the response latency of cat retinal ganglion cells. *Vision Res.*, **13**, 837-853 (1973)

[9] v.d. Malsburg, C. The correlation theory of brain function. Int. report 81-2, Dept. of Neurobiol. Max-Planck-Institute for Biophysical Chemistry, Göttingen, (1981).

[10] McClurkin, J. W., Optican, L. M., Richmond, B. J. and Gawne, T. Concurrent processing and complexity of temporally encoded neural messages in visual perception. *Science* **253**, 675-677, (1991).

[11] Metropolis, N., Rosenbluth, A.W. Rosenbluth, M.N., Teller, A.H. and Teller, E., *J. Chem. Phys.*, **21** 1087-1091, 1953.

[12] Opara, R., Wörgötter, F., Using visual latencies to improve image segmentation. *Neural Computation* **8**, 1493-1520, (1996).

[13] Sestokas, A.K., Lehmkuhle, S. and Kratz, K.E. Visual latency of ganglion X- and Y-cells: a comparison with geniculate X- and Y-cells. *Vision Res.*, **27**, 1399-1408. (1987).

[14] Vorbrüggen, J.C., Zwei Modelle zur datengetriebenen Segmentierung visueller Daten, volume 47 of Reihe Physik. Verlag Harri Deutsch, Thun, Frankfurt am Main, 1995.

[15] Wang, S., Swendsen, R.H., Cluster monte carlo algorithms. *Physica A*, **167**, 565-579 (1990).

[16] Wolff, U., *Phys. Rev. Lett.*, **62** (1990) 361.

[17] Wörgötter, F., Opara, R., Funke, K. & Eysel, U. Utilizing latency for object recognition in real and artificial neural networks. *NeuroReport*, **7**, 741-744 (1996).