METHODS FOR THE ASSESSMENT OF MULTINEURAL AND MULTIVARIATE ACTIVITY PATTERNS AND SOME CNN ARCHITECTURE IMPLEMENTATIONS

Ph.D. Dissertation

Viktor Gál

Supervisor:

József Hámori, *D.Sc.*
ordinary member of the HAS

Consulting adviser

Tamás Roska, D.Sc.
ordinary member of the HAS

Analogical and Neural Computing
Systems Laboratory
Computer and Automation Institute
Hungarian Academy of Sciences

Faculty of Information Technology
Péter Pázmány Catholic University

Budapest, 2007

Acknowledgements

In the first place, I would like to thank my supervisors, Professor József Hámori, and Professor Tamás Roska for their non-fading invaluable support. Their helpful guidance led me to the field I was always dreaming about, the fascinating interdisciplinary field between biology and engineering. It is a great privilege to be involved in the work of their exceptional research groups.

I am very grateful to Prof. Ronald Tetzlaff for the year I could spend in his group at the Applied Physics department of the Goethe University in Frankfurt am Main in a collaborative work with PD Dr. Sonja Grün from the Max Planck Institute for Brain Research. The institute - where I learned a lot about neurophysiology – is headed by the legendary icon of the modern neurobiology Prof. Dr. Wolf Singer, whose encouraging words determined the direction of my research.

Thanks to my research collaborators in the international LOCUST project: Dr. Claire Rind, Prof. Ángel Rodríguez-Vázquez, Martti Soininen, Dr. Gustavo Liñán-Cembrano, Jorge Cuadri Carvajo, Dr Richard Stafford and Dr. Matthias S. Keil.

It is hard to make a complete list of all those who I owe special thanks. I recall Dávid Bálya, István Petrás, Ákos Zarándy, Zoltán Vidnyánszky, Barna Garay, Csaba Rekeczky, Gergely Timár, László Orzó, István Szatmári, Mátyás Brendel, Károly László and the others whom I also wish a good luck. I owe a lot to Ms Katalin Keserű for her practical and official aid.

1 Introduction

This doctoral work is dealing with three different multivariate pattern analysis problems, and what encapsulates them into one work is the 'analog' and 'logic' (analogic) approach in their background and their direct or indirect relation to neurobiology.

Cellular Nonlinear/Neural Network Universal Machine (CNN-UM) [Chua & Roska, 1993; Roska & Chua, 1993; Chua & Roska, 2001] is the platform for practical implementations of the analogic ideas: a brief mathematical introduction of this framework (in Section 2) is inevitable because of the central role it plays in my thesis.

The structural organization of the Cellular Nonlinear/Neural Network Universal Machine makes this computer an ideal tool to study multiple spatially ordered signal flows. CNN is a locally connected array of nonlinear dynamical systems called cells [Chua & Yang, 1988]. It is discrete in space but continuous in time. Its connections or couplings determine the dynamics of the system. CNN provides a well-defined mathematical and a hardware feasible physical framework to study the emergence of patterns encoded by the local interactions of identical cells. The easy VLSI (Very Large Scale Integration) implementation of the cell array is the most important advantage compared to other modeling frameworks.

In the vast majority of considered applications the relative position of signals within that array–defining possible local interactions–has undoubted significance. Although this principle is the guideline in two of the topics covered by this dissertation, using CNN for the analysis of multidimensional systems where no embedded spatial information is expected seems to be unusual. Through the example of multiple neural activity data I try to show that problems with spatially independent–or independently processed–signals can also be translated effectively to the language of CNN.

One of the fundamental questions in neuroscience concerns the nature of the neural code: how the information, or signal that one neuron communicates to others is embodied in its biophysical processes. A basic controversy in investigations of this topic involves the rate-coding hypothesis. This is roughly the hypothesis that the information conveyed by a neuron in a sequence of action potentials (or spikes) is contained wholly in the local spiking frequency – number of spikes which occur in coarse temporal intervals on the order of several tens or hundreds of milliseconds [Singer, 1993, 1999; Singer et al 1997]. Alternatively, the temporal coding hypothesis holds that the precise (e.g. up to msec) location of spikes – especially in relation to spikes of other neurons belonging to one specific functional group - conveys information.

The role of synchronous firing of visual neurons to bind together the responses to components of an object is just such an attractive but controversial idea. Visual processing initially breaks up the visual scene into isolated fragments that are detected by many individual neurons in V1 and higher visual areas. Yet visual perception of objects somehow reassembles the isolated fragments into complete objects. The problem of creating a unified percept from the responses of many separate neurons is referred to as "the binding problem." In the early 1990s, a theoretical solution emerged from the work of Wolf Singer and others [Singer & Gray, 1995], who found that neurons responding to portions of the same object fired synchronous action potentials. They suggested that correlated firing could be the mechanism for binding together responses from multiple, dispersed neurons.

I describe statistical models for spike trains which bear on rate versus temporal coding distinctions, and develop hypothesis tests for the purpose of relating the models to experimental data. My goal is to develop definitions and methodology to assess the existence and nature of fine temporal structure in the neural records using assumptions which are compatible with the theoretical goals of neuroscience. Implementing these methods, beside the design of traditional computational algorithm, I also considered the reformulation of the problem of spatially independent signals enabling the much faster parallel computation.

In the second thesis I prove that effective CNN-UM implementation of a special 'fine-grained' type genetic algorithm is feasible [Bálya & Gál, 2006]. A genetic algorithm (GA) is a search technique to find approximate solutions to combinatorial optimization problems: classical domains are characterized by fitness (optimization) functions with complex fitness landscape (many local minima) on high dimensional data sets. Typical examples include timetabling, scheduling problems, automated design, but here I show the suitability of binary GA in connection with the analysis of multi-dimensional (multi-channel) neural activity data.

The last thesis group is dealing with collision prediction: analysis of situations when the motion of an object could end up with a useful (for a predator) or disadvantageous (for a prey) collision with the observer/sensor.

A growing body of evidence makes neurobiologists suppose that for some animals (and humans) low-level monocular visual information alone is enough for a remarkable estimation of the time left till an impending collision [Rind & Simmons, 1999], [Schiff & Detwiler, 1979]. Fast calculation even in an unfamiliar environment may have a life-saving impact. The locust LGMD neuron is evolutionary honed to detect potential collisions with predators and provide a collision avoidance reaction. In Section 5 I show analogic collision avoidance systems partly derived from the computer model of the locust visual system and demonstration of their suitability for use in automotive situations as a means of detecting a collision. The basic building blocks of the models are based on simple visual receptive field interactions implemented on CNN-UM [Gál et al., 2004].

2 Cellular Neural Networks and the CNN Universal Machine

2.1 Description of the standard CNN

According to the somewhat formal definitions, a standard cellular neural network is defined to be a

(1) two-dimensional array of
(2) identical dynamic systems, called cells,
(3) with local interactions between the cells, and
(4) continuous valued state variables.

Cells are indexed by their row (i) and column (j) coordinates. The neighborhood N of a cell (i,j) is the finite set of indices (k,l) for which the state of cell (i,j) explicitly depends on the set of cells $\{(i+k, j+l)((k,l)\in N\}$. The dynamic behavior of the CNN can be described by the collection of single cell ordinary differential equations. A simple CNN cell has a single state variable x_{ij}, an input value u_{ij} and an output y_{ij}. The output y_{ij} depends on the state variable according to the nonlinear function

$$y(x)=\frac{1}{2}(|x-1|-|x+1|). \tag{2.1}$$

The (normalized) first-order differential equation for the cell (i,j) is defined as

$$\frac{d}{dt}x_{i,j}(t)=-x_{i,j}(t)+\sum_{k,l\in N}A_{k,l}y_{i+k,j+l}(t)+\sum_{k,l\in N}B_{k,l}u_{i+k,j+l}+z$$

The outputs (y) and inputs (u) of neighboring cells are coupled by means of weighted sums. These $A_{k,l}$ and $B_{k,l}$ weights together with the bias *z* form the CNN *cloning template* (or simply CNN *template*). A CNN cell is shown in Figure 1. The block diagram of signal flow in a CNN cell is shown in Figure 2.

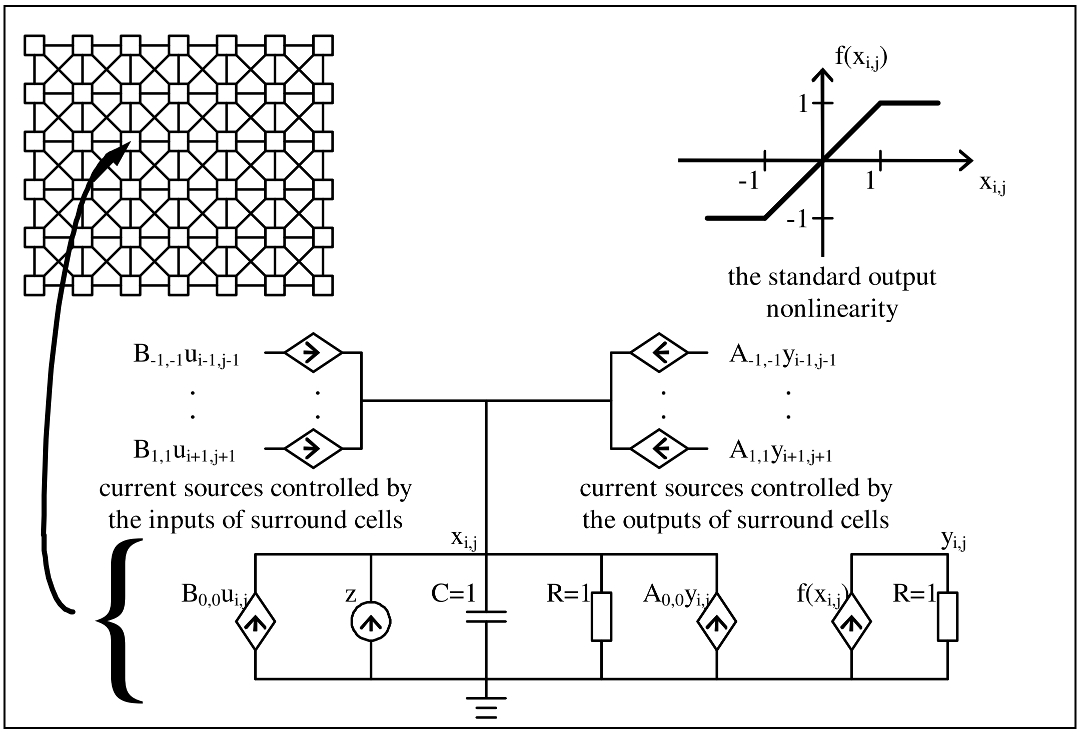


Figure 1 Realization of a CNN cell. Realization of a CNN cell. Diamond-shape symbols denote voltage-controlled current sources injecting currents proportional to the indicated controlling voltage $u_{i+k,j+l}$ or $y_{i+k,j+l}$, except for the diamond indicated by $f(x_{ij})$ which is a nonlinear voltage controlled current source.

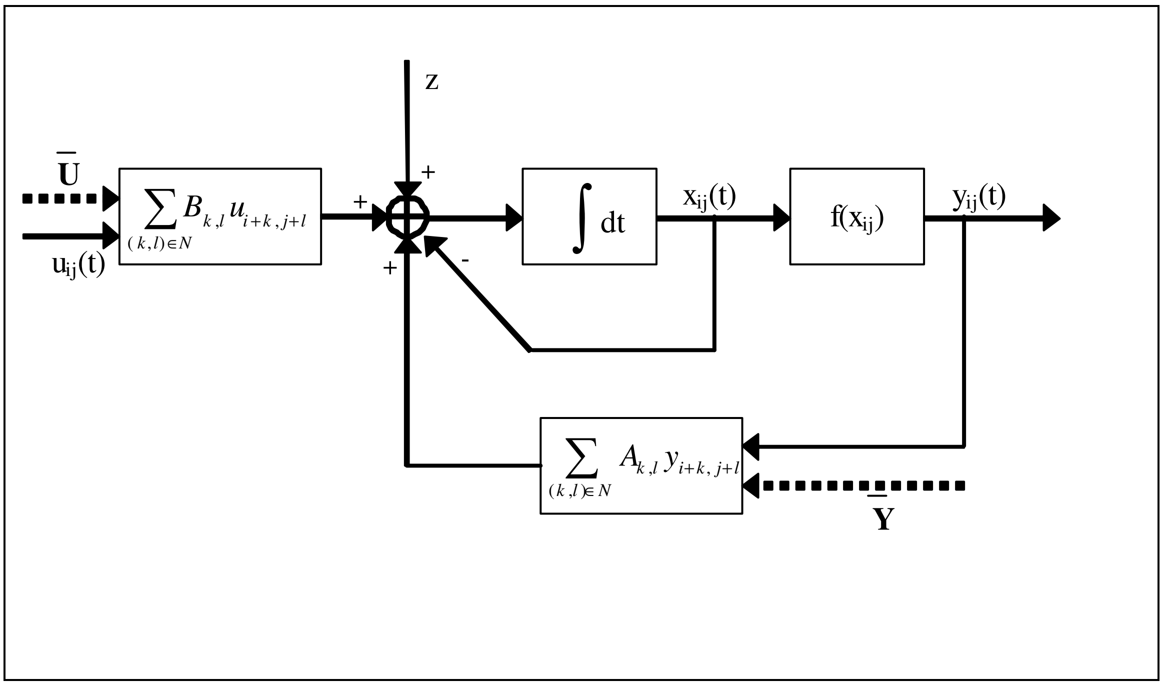

z
$\bar{U}$
$\sum_{(k,l)\in N} B_{k,l} u_{i+k,j+l}$
$u_{ij}(t)$
+
+
+
-
$\int dt$
$x_{ij}(t)$
$f(x_{ij})$
$y_{ij}(t)$
$\sum_{(k,l)\in N} A_{k,l} y_{i+k,j+l}$
$\bar{Y}$

Figure 2 Signal flow in a CNN cell. Dashed arrows mark parallel data paths from the input and the output of surrounding cells.

In order to fully specify the dynamics of the array, the boundary conditions have to be defined. Cells along the edges of the array may see the value of cells on the opposite side of the array (circular boundary), a fixed value (Dirichlet-boundary) or the value of mirrored cells (zero-flux boundary). The choice of boundary condition can have a fundamental effect on the global behavior of the array as it will be demonstrated in Section 5.3.

2.2 Analogic algorithms and the CNN Universal Machine

When used for image processing, the input and state of the CNN array is loaded with the image data, and the result of CNN computation is the steady state after the transient of the network. If each cell of the CNN array is equipped with programmable weight coefficients, analog and logic memory units, logic operations can be defined between these logic memory units, and these logic functions along with the templates become programmable, we arrive at the concept of the CNN Universal Machine (CNN-UM) [Roska et al, 1993]. This (re)programmability makes the CNN a real microprocessor, the first algorithmically programmable *analogic* (i.e., both analog and logic) computer. In this framework, each particular CNN operation (analog transient computation, or local logic operation) can be thought of as an *analogic instruction* of this computer. This allows to store intermediate results of the processing, and to build up and run complex image processing algorithms on a CNN chip using some control hardware.

In the course of CNN template and algorithm design, many useful specialized templates and simple template combinations have been found, many of which are compiled in a CNN Software Library [Roska et al., 1999]. These provide basic components of several standard image processing techniques. Beyond that, analogic CNN algorithms may utilize a number of spatio-temporal effects in their basic operations which can hardly be applied if conventional image processing technology is used. These may be neurobiological receptive field effects[Gál et al. 2004], mathematical morphology or different PDE-based techniques.

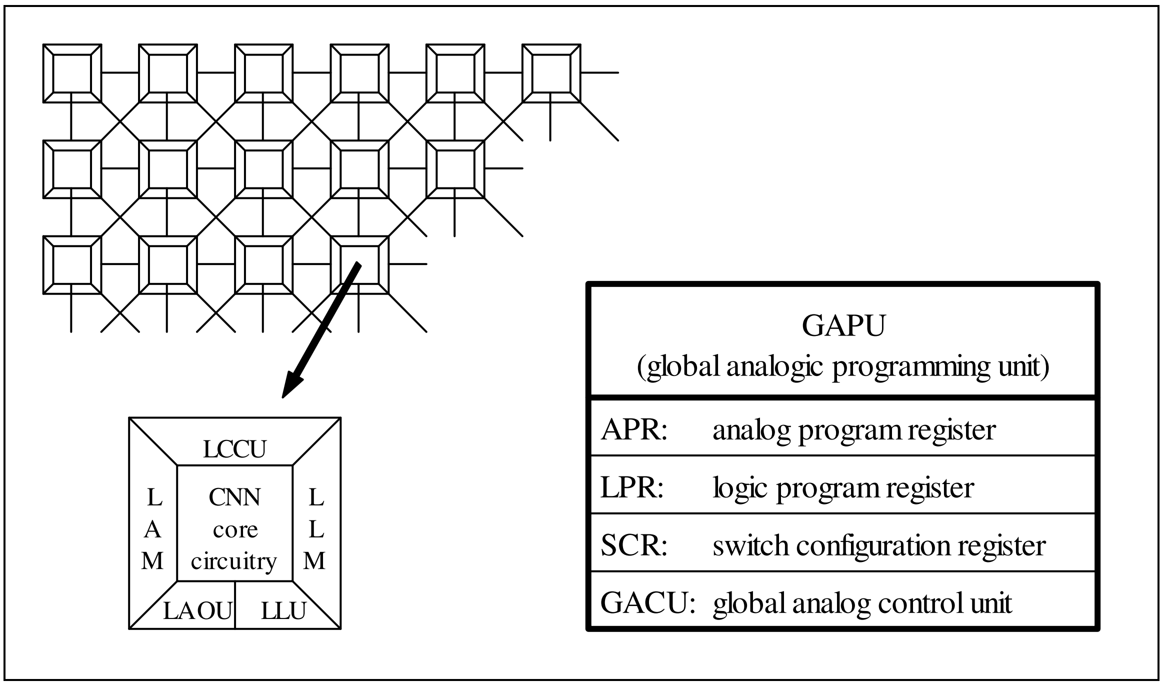

LCCU
L A M
CNN core circuitry
L L M
LAOU
LLU
GAPU
(global analogic programming unit)
APR: analog program register
LPR: logic program register
SCR: switch configuration register
GACU: global analog control unit

Figure 3. The structure of the CNN Universal Machine. The extended standard CNN cell has various components providing for programmable array computing. LCCU: local communications and control unit, LAM: local analog memory, LLM: local logic memory, LAOU: local analog output unit, LLU: local logic unit. The LCCU receives the programming instructions (in each cell) from the global analogic programming unit (GAPU).

In Figure 3, the global architecture of a CNN-UM [Roska et al., 1999] and the local units added to its cells are shown which provide for programmability and combination of intermediate results in analog and logic operations. Intermediate results can be analog and binary, these can be stored in local analog memories (LAMs) and local logic memories (LLMs) respectively. Pixel-by-pixel arithmetic and logic is performed by the local analog output unit (LAOU) and a local logic unit (LLU). A global analogic programming unit (GAPU) provides the central controlling of the CNN computer. This unit contains

- registers for storing template values (analog program register - APR),
- control sequences for logic operations (local program register - LPR),
- a switch configuration register (SCR) to access the different functional units, and
- a global analog control unit (GACU) storing the instruction sequence of the main program.

3 Assessment of multiple neural activity patterns

First thesis

Assessment of temporal scale and patterns in multi-neuron recordings

I/a. Effective and biologically relevant surrogate data generation algorithm development and characterization

I proposed a computationally effective method for the generation of surrogate data from an original discretized bioelectric signal, typically multiple single neuron recordings. Original spikes of the neurons are shifted randomly and independently back and forth resulting in surrogate spike-trains. This method satisfies the most typical requirements for surrogate data: it preserves the main structure of the individual neuron activity records following changes in the firing rates (non-stationary) but disrupts fine interdependence between them. The most important features to preserve:

•total number of spikes

•non-stationarity; even sudden, transient changes in frequency

The technique can take physiological relevance and accuracy assumptions into account.

I studied and explained the role and possible significance of the parameters controlling the surrogate generation, and I also showed adjustment methods for optimal performance.

I/b. A bootstrap technique for finding and evaluating higher order temporal patterns in multi neuron recordings

I introduced a statistic that can be evaluated on original multiple single neuron recordings and their surrogates. The statistic calculates the frequency of all possible synchrony patterns in the data.

Calculating the bootstrap significance of pattern frequencies based on their empirical distribution indicates if occurrence of a pattern is higher than expected by chance: decision about rejection of H_0 (there are only independent processes) can be based on the statistical rank of the original statistic within the order statistics of bootstrap distribution. I showed that manipulating the parameters of the surrogate generation process one can

- define extra limits
- tell if the occurrence of a certain pattern is statistically significant, thus a functional transient cell group is formed
- determine if there are transient formations of functional cell groups

I/c. Implementation of the surrogate data generation on CNN-UM: an analogic approach

I found an effective solution for the CNN-UM architecture adaptation of the surrogate data generation method described in the first part of the thesis. The original algorithm is modified to be suitable for fast parallel computing. Emulating a certain type of cellular automata (CA) by the CNN-UM I was able to generate good quality pseudo-random patterns with controllable features that suit the needs of high throughput surrogate data generation. The algorithm was tested on an industrial framework embedding the ACE4k. Cellular Visual Microprocessor serving as a feasibility test-bed.

3.1 Introduction: the temporal structure of the neural code

In the last few decades the attention of neurophysiologists has been shifted from single-neuron activity recordings to multi-cell studies. Although there are efficient analytical tools targeting single-neuron and neuron pair activity problems (autocorrelogram, crosscorrelogram, peri-stimulus time histogram (PSTH), Joint-PSTH etc. [Gerstein et al, 2001]), a great number of questions remain unanswered until we observe several cells simultaneously. Current scientific trends consider the information processing and transmission capacity of individual neuron cells rather poor. The idea is that even simple pieces of information in our brain would be represented by spatio-temporal activity patterns across a number of neurons.[Singer, 1993, 1999] In other words, only cell-groups with coordinated activity are able to carry complex signals effectively [Hebb, 1949].

There has been a long debate on the rate-coding (via individual cells)[Barlow, 1972; Barlow, 1992] versus temporal-coding (in terms of coordinated activity of functional groups of cells) theories among numerous experts from different interdisciplinary fields. The central question is whether precise relative timing of spikes matters: whether the precise temporal position of spike times in spike-patterns matters in the functioning of the brain.

One method of investigation involves presenting a stimulus repeatedly to a subject, measuring the neural response over many such presentations, and examining the resulting empirical rate: the empirical frequency of firing as a function of time, relative to stimulus onset (or for time-varying stimuli, relative to a fixed time in the stimulus presentation). If the neuron was coding for the identity of stimulus and environmental context precisely in its spike positions, one would expect that the rate would appear to be changing very rapidly: spikes are laid down with great precision relative to each other. This is, for example, referred to as temporal coding in [Abeles et al., 1993a], but its delectability depends on the width of the rate-increase, on the number and complexity of induced correlated spikes and whether the patterns are stimulus locked etc. [Pauluis & Baker, 2000; Roy et al., 2000; Gütig et al, 2001]. If, on the other hand; the spikes are not laid down precisely, one would expect the empirical rate function to be slow-varying, or flatter; this would be consistent with rate-coding.

How to explore and prove the existence of temporal patterns, or high relative (and not stimulus locked) timing precision? The idea is that we should set a precision limit, and prove that relative spike timing is more accurate than the limit we defined. For that end one can randomly and independently shift each spike in a recorded train, using perturbations with a predefined magnitude δ–e.g. δ=±5 milliseconds–, and compute the value of some statistic on the altered train. Roughly, perturbing the spike times should have the effect of preserving only those temporal structures with timing precision coarser than δ milliseconds. This leads to the following strategy for assessing the existence of fine temporal structure: repeating the surrogate generation process many times, one can create the bootstrap (sampling) distribution on the value of the statistic for surrogates of the spike train, and then compare the value of the statistic on the original, unchanged spike train. If the original statistic is then atypical with respect to the surrogates (for example, if it lies in one of the tails of the sampling distribution), then one is led to suspect that the presence of fine temporal structure is necessary to account for the observed spike train. Specifically, one concludes that the data is incompatible with temporal patterns the precision of which is coarser than δ millisecond.

What kind of estimator (statistic) should we choose for the method above? On the one hand, observing even a few neurons produces so many patterns–combinations of different number of spikes and inter-spike intervals across the cells–that makes computationally infeasible to detect all repeating sequences.

On the other hand limiting our search to synchronized events can be a sensible approach for a couple of reasons: firstly, this consumes moderate amount of computing power, secondly, that sort of patterns have already known biological relevance. In contrast with complicated temporal patterns, the theoretical significance and the neural mechanisms of keeping track and using of coincident spikes by the brain have been studied thoroughly. As a very simple example, synchronized firings of converging neurons facilitate the transmission of signals in the target cell via temporal summation [Softky & Koch, 1993; Diesmann et al., 1999].

A more ambitious but well established hypothesis says that synchronized firings could be the physical base of binding discrete pieces of information encoded by distinct cells but describing the same entity [Engel et al., 1992; Roelfsema et al, 1996; Singer, 1993; Singer & Gray, 1995]. Synchronization requires a mechanism that precisely coordinates the timing of individual spikes generated by neurons which transiently form functional cell assemblies [Abeles, 1991; Gerstein et al., 1989; Palm, 1990; Singer, 1993; Sakurai, Y. 1996] and would behave otherwise more or less independently. Our main interest is whether having only the output history of randomly selected cells one could reveal the presence of such a coordinating force working in the background, localize the neurons involved and prove possible significant interdependence between them.

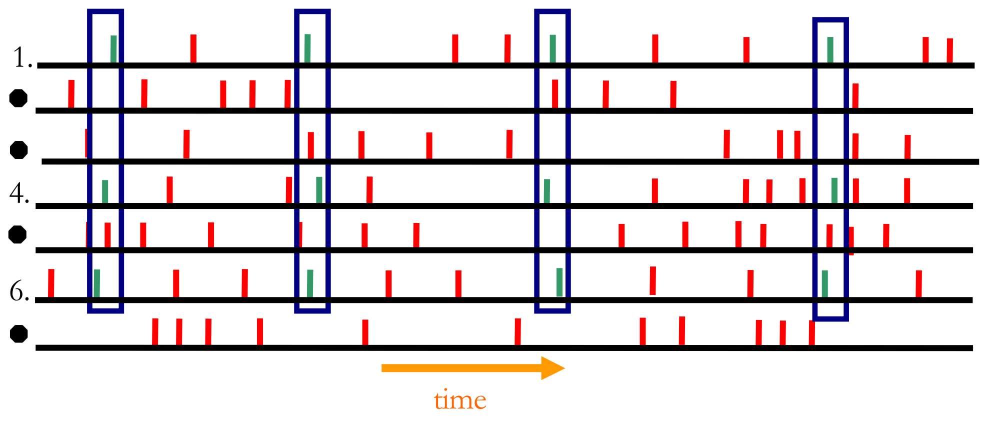


Fig. 4. Activity timeline of 7 simulated neurons. From the seven cells neuron 1,4 and 6 tend to fire in synchrony from time to time. Spikes are represented by the short vertical lines. The blue rectangles are the jitter-windows within that spikes accepted to be coincident.

The task is not straightforward, since some incidental coincidences can be found when looking at absolutely independently working neurons, and these have to be reliably differentiated from observations where real functional cell groups emerge. The critical point is to define a threshold after each experiment: a threshold in the number of detected coincidences–in fact a threshold for each coincidence pattern–that is exceeded only (at 5% confidence level) when activities of certain neurons are strongly coordinated, that is, the timing precision exceeds a certain limit. Choosing the right value may depend on the (non-stationary) background firing frequencies of the neurons involved, but some other parameters should also be considered.

One of these parameters is the level of tolerance towards the imprecision of the synchronizing mechanisms, namely, the width of a time window within that spikes emerging from distinct cells regarded to be coincident ("jitter", see Fig. 4). Exact values for the parameters are difficult to get, due to the fact that they are partly theoretical numbers, and the calculation should be based upon a lot of recordings and analysis trying with different values.

Though a lot of promising attempts have been made (likelihood, frequency-domain and information theory based methods, reverse correlation and Bayes' neural spike train decoding, gravitational clustering) no widely accepted method has been developed for this computationally demanding problem so far [Martignon et al., 1995; Grün et al,1999; Brown et al., 2004].

3.2 Spike time perturbation method: bootstrap statistic for the assessment of the temporal structure in the neural code

3.2.1 Multiple single cell recordings: the data

Although neurons are analog systems, here–like most large-scale modelers–we handle them as tiny processors with binary output function: either they are active (discrete firings, spikes), or silent; thus, their operation is represented by sequences of firings and inactive periods.

The spike activities of the observed neurons are recorded on a certain time scale. The first step is to rescale/discretize them based on a sensible resolution, which is usually τ=1 msec. Time discretization is implemented via "exclusive binning", thus each spike is sorted into time bins: one bin represents a time window with a predefined width (τ), and its value is 1, if at least one spike occurred during its respective time window, otherwise 0.

Accordingly, the activity of the i^{th} neuron of the observed cells in the k^{th} time bin–with respect to a certain event used as reference–can be formulated as

$$s_i(k) = \begin{cases} \text{1 if spike in } \left[\mathrm{k}\tau, (\mathrm{k}+1)\tau\right) \\ \text{0 if no spike in} \left[\mathrm{k}\tau, (\mathrm{k}+1)\tau\right) \end{cases} \tag{3.1}$$

where τ defines the length of the time-window used for discretization. Including all of the observed processes into one equation we can write:

$$\overline{\mathbf{s}}(k) = \begin{bmatrix} s_1(k) \\ s_2(k) \\ \vdots \\ s_N(k) \end{bmatrix} \tag{3.2}$$

where $s_i(k)$ is the spike activity function of time, i stands for the index of the respective electrode (or neuron). Thus, from N electrodes we get N parallel binary trains [Cox & Isham, 1980], and these can build up a vector-process, where a vector-valued function $\overline{\mathbf{s}}(k)$ represents the (0,1) pattern of activity across the studied neurons at the k^{th} time bin.

$\bar{\mathbf{s}}(k)$ at any k can take $m = 2^N$ possible constellations of 0s and 1s (coincidence patterns). cp_θ will denote one specific pattern of the m possible constellations, where θ is an arbitrary index $\theta \in \{1,..,m\}$ of the available patterns.

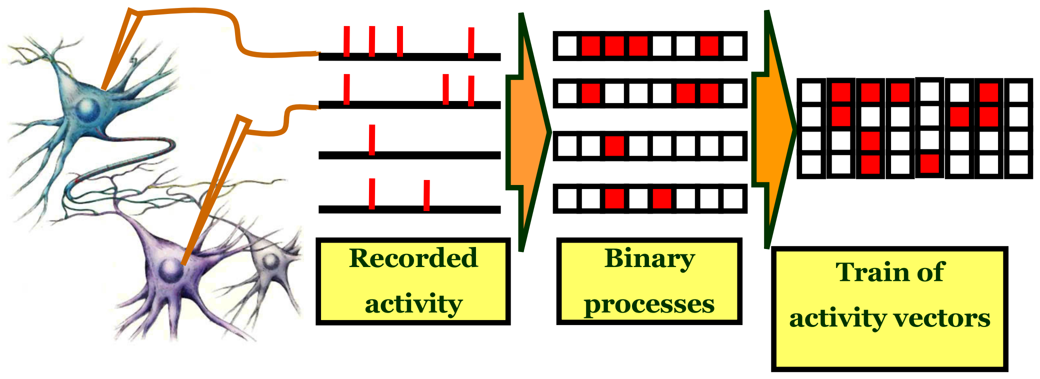


Fig. 5. Mapping multi-electrode recordings onto binary arrays.

Alternatively we can describe the processes by directly specifying the timings of each spikes in the data. If there are $\mathrm{P_i}$ spikes fired by the i^{th} neuron, the neuron's activity is defined by a vector $\bar{\mathbf{X}}_i$ with $\mathrm{P_i}$ entries specifying its spike-times, respectively:

$$\bar{\mathbf{X}}_i := \left(x_i(1), x_i(2), \dots, x_i(P_i)\right), x_i \in \mathbb{N} \tag{3.3}$$

where $x_i(q)$ specifies the time (time-bin index) of the q^{th} spike in the spike train of the i^{th} neuron. Spike-times are expressed in time-bin indexes using the same resolution (and notation for time) as in Eq. (3.1).

3.2.2 Patterns to detect

There is a lot of spatial-temporal activity patterns we are interested in [Dayhoff & Gerstein 1983; Nadasdy et al., 1999], but there is much less we can analyze due to computational power limits. Patterns like "there is a tendency that 11ms after a neuron 1 spike there is neuron 4 and neuron 5 coincident spike followed by neuron 2 activity after 3 ms" is extremely complex to find, since computing all the possible combinations means combinatorial explosion. For this reason we are going to limit our search for synchronous events or very simple spatial-temporal patterns.

In our mathematical term a synchronous event is expressed by a binary activity vector (cell column) that has more than one nonzero entry. The complexity ξ of such a pattern is defined by the number of nonzero entries in the vector.

We are also interested in less precise coincidences Figure 7, e.g. nonzero entries in subsequent activity vectors (neighboring columns).

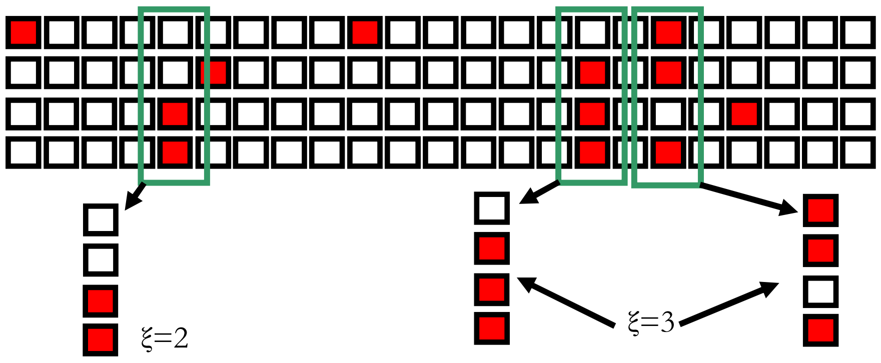


Figure 6 Precise synchronization patterns with complexity ξ=2 and ξ=3

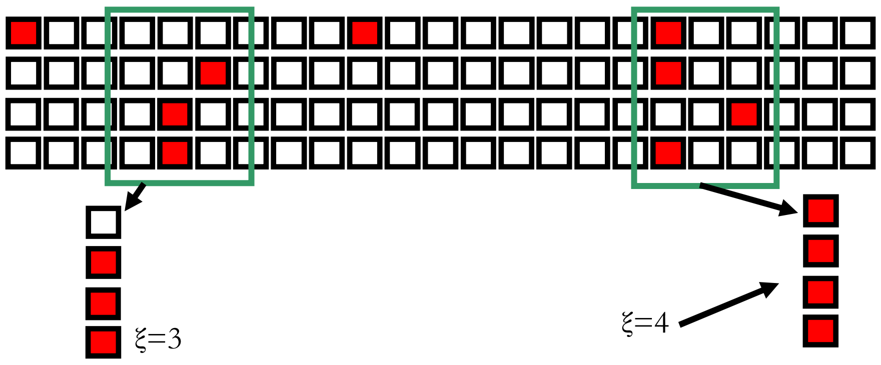


Figure 7 Detecting inaccurate coincidences: enabling a jitter window with width of 3 ms we can explore less accurate synchronization mechanisms

It is not rare, that the so called "capture" effect (a spike occurs in conjunction with a pattern by chance), more complex patterns occur than the number of correlated processes may explain, and some low complexity patterns remain undetected. To take all those low complexity patterns into account, on should care about subpatterns of the detected patterns (see Figure 8).

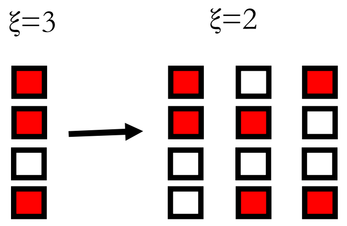


Figure 8 Subpatterns: there are 3 complexity ξ=2 subpatterns of each pattern ξ=3

3.2.3 *Statistical considerations*

To test the hypothesis that there is fine temporal structure –coordinated activity- amongst the neural processes, we define our null hypothesis, H_0, that there is only coarse temporal interplay between the investigated neurons, and

- Starting from the *N* original spike trains $\bar{\mathbf{X}}_1^0 \ldots \bar{\mathbf{X}}_N^0$ –the superscript zero(0) will refer to the original, 'recorded' data–create J 'surrogate' data for each of the *N* neurons ($\bar{\mathbf{X}}_i^1 \ldots \bar{\mathbf{X}}_i^J$ for the i^{th} neuron) that are very similar to the original recorded activity in terms of preserving the main structure of the individual neuron activity records but disrupt fine interdependence between them

- define an appropriate statistic $\mathrm{g}\left(\bar{\mathbf{X}}_1, \bar{\mathbf{X}}_2, \ldots, \bar{\mathbf{X}}_N\right)$ as a function of *N* processes for the assessment of fine temporal interdependence in the data

- calculate the statistic for the original activities $\bar{\mathbf{X}}_1^0 \ldots \bar{\mathbf{X}}_N^0$

- calculate the statistic for each of the J surrogates $\bar{\mathbf{X}}_1^j \ldots \bar{\mathbf{X}}_N^j, \forall 1 \le j \le J$ of the *N* spike-trains to create a sampling (bootstrap) distribution

- define a threshold value γ within the distribution for the rejection of H_0 at α (=0.05) significance level.

- based on the relation between the statistic $\mathrm{g}\left(\bar{\mathbf{X}}_1^0, \bar{\mathbf{X}}_2^0, \ldots, \bar{\mathbf{X}}_N^0\right)$ of the original activity and γ threshold make a statistical decision with significance p

Let $g(\bar{\mathbf{X}}_1, \bar{\mathbf{X}}_2, \ldots, \bar{\mathbf{X}}_N)$ a function of *N* discrete spike trains describing a relevant feature (statistics) of the *N* parallel processes. In our work it will be a measure of synchrony, i.e., the occurrence of a certain coincidence pattern within the interval we study. Selecting two parallel processes (e^{th} and f^{th} neurons with P_e and P_f spikes respectively) it can be written:

$$g(\bar{\mathbf{X}}_e, \bar{\mathbf{X}}_f) = \sum_{q=1}^{P_e} \pi_q(x_e(q)), \tag{3.4}$$

and

$$\pi_q(x_e(q)) = \begin{cases} 1 \text{ if } \min\limits_{1 \le p \le P_f} |x_e(q) - x_f(p)| \le w \\ 0 \text{ if } \min\limits_{1 \le p \le P_f} |x_e(q) - x_f(p)| > w \end{cases} \tag{3.5}$$

where $x_e(q)$ and $x_f(p)$ specify the time-bin indexes of the q^{th} and p^{th} spikes of the e^{th} and f^{th} neurons respectively (as in Eq. (3.3)), and w is the magnitude of inaccuracy we accept in the definition of synchrony. π_q is the (0,1) indicator function of the coincidence event.

The method of surrogate data generation will be described later in this chapter, here we define the j^{th} surrogate of the i^{th} neuron as a (random) u transformation of the original activity:

$$\bar{\mathbf{X}}_i^j = u_j(\bar{\mathbf{X}}_i^0) \tag{3.6}$$

Then compute the synchrony measure for each surrogate data (typically J =1000):

$$Z^j = g(\bar{\mathbf{X}}_1^j, \bar{\mathbf{X}}_2^j, \ldots, \bar{\mathbf{X}}_N^j), 1 \le j \le J \tag{3.7}$$

and also for the original data

$$Z^0 = g(\bar{\mathbf{X}}_1^0, \bar{\mathbf{X}}_2^0, \ldots, \bar{\mathbf{X}}_N^0). \tag{3.8}$$

If we rearrange the Z^j statistics (from j =0 to $j = J$) according to their values in an increasing order $Z_1, Z_2, \ldots, Z_{J+1}$ (the lower index introduced here denotes the rank order of the calculated Z^j statistics), a decision about H_0 can be based on the statistical rank of the original Z^0 within the order statistics. The lower index β in Z^0_β is the rank order of the calculated statistic of the original data within the surrogate data set. Comparing β and J (the total number of surrogates) one can define the statistical significance of Z^0:

$$p = 1 - \frac{\beta}{J+1} = \frac{J+1-\beta}{J+1} \tag{3.9}$$

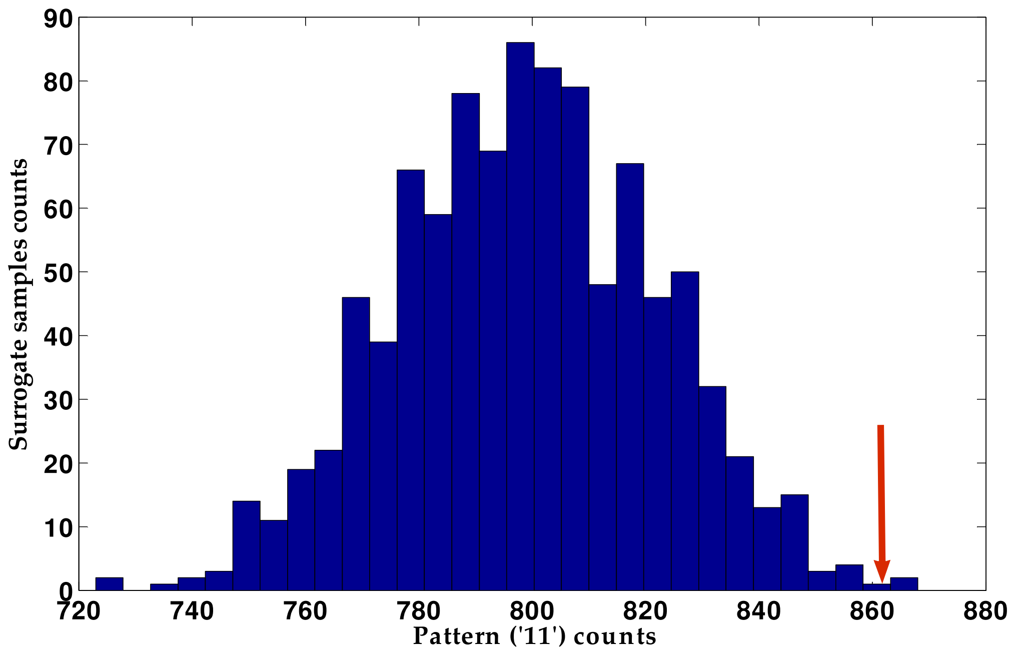


Figure 9 Bootstrap statistic: blue bars describe the sampling distribution of the surrogate data. The distribution of the synchrony counts within the samples is tabulated: the numbers of samples are presented as a function of the coincidences they contain. Red arrow: the count of the synchrony in the original spike train located at the tail of the distribution (p=0.006)

p is the estimated probability of rejecting the H_0–there is only coarse temporal interplay between the neurons–when that hypothesis is true. Thus with increasing β (decreasing p) we are getting more and more confident that the alternative hypothesis–precise temporal interplay exists between the neurons– has to be accepted.

For statistical decision, we may define the maximum of the p value, and above that we accept the H_0 hypothesis: $p >= \alpha$. Otherwise ($p < \alpha$) we must reject the null hypotheses at α significance level.

One can look at the other tail of the distribution at the opposite side as well: very low rank order of Z_{β}^{0} means that the feature we described by the statistic is much less frequent than that of the H_0 hypothesis suggests. Thus we can reject H_0 and accept an alternative hypothesis: a certain synchrony pattern is less frequent in the original data than we expected based on the sampling distribution.

Following the terminology of S. Grün "unitary event analysis" method [Grün, 2002] we are going to handle the cumulative probability from (3.9) as a measure for statistical significance of the observed feature (here: joint spiking), calling it joint-p-value $\Psi\left(Z^{0},\overline{\mathbf{Z}}'\right)$. $\overline{\mathbf{Z}}'$ is the bootstrap sampling distribution calculated from surrogate data.

To enhance the visual resolution at the relevant low values for Ψ (or $1-\Psi$), we choose a modified logarithmic scaling as was done for the surprise measure in other studies [Grün, 2002, Legendy, 1975; Palm, 1981; Legendy & Salcman, 1985; Palm, Aertsen, & Gerstein, 1988; Aertsen, Gerstein, Habib, & Palm, 1989]. This transformation, the joint-surprise, which approximates the conventional surprise measure for the relevant values of Ψ and yields a continuous and differentiable function around Ψ =0.5:

$$S\left(\Psi\right)=\log\left(\frac{1-\Psi}{\Psi}\right) \tag{3.10}$$

For excessive coincidences, this function is dominated by the numerator Ψ; for lacking coincidences, it is dominated by the denominator $1-\Psi$. Equation (3.10) is comparable to measure significance on a dB scale and yields positive numbers for excessive coincidences (e.g., S =1 for Ψ =0.1, or S =2 for Ψ =0.01), negative numbers for lacking ones, while changing sign at chance level Ψ =0.5 (see Table 1). S =1,2788 at the typical significance level Ψ =0.05.

On the basis of the joint surprise as a measure of significance, we now define unitary events as the occurrence of those spike patterns cp_{θ} in a given interval Δ that can be found much more often than expected by chance, or from the sampling distribution. To that end, we set a threshold S_{α} on the joint-surprise measure and denote the occurrences of those cp_{θ} for which as unitary events. The arguments of S $\left(\mathrm{Z}^{0},\overline{\mathbf{Z}}'\right)$ are pattern-dependent, consequently this test is performed separately for each coincidence pattern cp_{θ} within the interval Δ.

Ψ <0.5 (more than expected)	**S>0**
Ψ =0.5 (according to expectations)	**S=0**
Ψ >0.5 less than expected	**S<0**

Table 1 Relation between the joint-surprise and joint-p-value

3.2.4 Generation of Surrogate Data

From the previous section it must be clear, that the aim of surrogate data generation from the original data is to mimic "identical" or similar physiology experiments over and over again with the constraint that the processes operating in the background are independent, or their precision, or correlation in time is coarser than a predefined level. Thus the surrogate data must follow the local spike frequency–and spike counts–of the original data with a certain precision we define. Then the bootstrap statistic can make a decision whether the real neural processes producing the recorded, original data operate on a finer or coarser time scale than the precision at which the surrogates follow the original local frequencies.

We developed a computationally effective algorithm to perturb the exact timings of the spikes in a local manner by an amount we can parameterize. The algorithm is based on a special random walk process, where each spike can move one time-bin after each iteration with probability μ (see Figure 10). In half of the iterations spikes may move forward and in the other half backward.

The method respects some physiological constrains on spike intervals: spikes will not overlap or cross each other, and depending on implementation details a certain refractory period can be introduced.

Trying to formularize mathematically the algorithm we introduce a random variable d that is sampled from the distribution:

$$p(d = \delta | I) = \eta(\delta, I) \tag{3.11}$$

where $\eta(\delta, I)$ is the probability, that during a random walk, after I iterations the walker is at distance δ from its origin considering the random walk process with μ step probability [Feller, W. (1968)]:

$$\eta(\delta, I) = \sum_{|\delta| \le v \le 0.5(I+|\delta|), v \in \mathbb{N}} (\mu/2)^{2v-|\delta|} (1-\mu)^{I-(2v-|\delta|)} \binom{I}{v} \binom{I-v}{v-d} \tag{3.12}$$

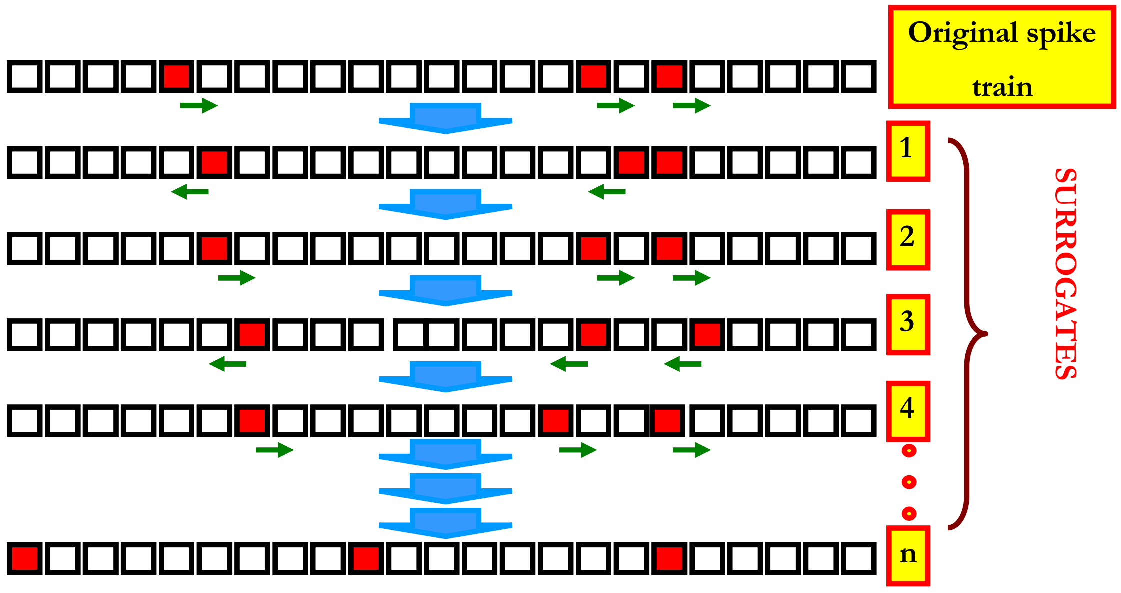


Figure 10 Surrogate data generation: shifting randomly selected spikes forward and backward in turn. Green arrows depict a shift with probability μ.

In our implementation, where during half of the iterations the spikes can move only in one direction (thus the maximal value of $\delta = I/2$) equation (3.12) is modified according to:

$$\eta(\delta, I) = \sum_{|\delta| \le v \le I/2, v \in \mathbb{N}} \mu^{2v-|\delta|} (1-\mu)^{I-(2v-|\delta|)} \binom{I/2}{v} \binom{I/2}{v-d} \tag{3.13}$$

provided there are no spikes in the $\pm I/2$ neighborhood. If there are spikes close to each other, the description of the distribution become increasingly complex. In Figure 11 and Figure 12 we present the features and performance of the random walk process as a function of different parameters.

$\bar{\mathbf{X}}_i^j$, the j^{th} surrogate sample of the i^{th} neuron is created from the original $\bar{\mathbf{X}}_i^0$ (for notations see Section 3.2.3) and from $\bar{\mathbf{D}}_{P_i}$, that is a sequence of $\mathrm{P_i}$ (number of spikes the i^{th} neuron showed) random variables d:

$$\bar{\mathbf{X}}_i^j = \bar{\mathbf{X}}_i^0 + \bar{\mathbf{D}}_{P_i} = \left(x_i^0(1)+d_1, x_i^0(2)+d_2, \ldots, x_i^0(P_i)+d_{P_i}\right) \tag{3.14}$$

$$\bar{\mathbf{D}}_{P_i} = \left(d_1, d_2, \ldots d_{\mathrm{P_i}}\right) \tag{3.15}$$

Creating an appropriate amount (typically 1000 samples) of surrogate data the coincidence patterns should be counted(see Figure 13) to create a sampling distribution.

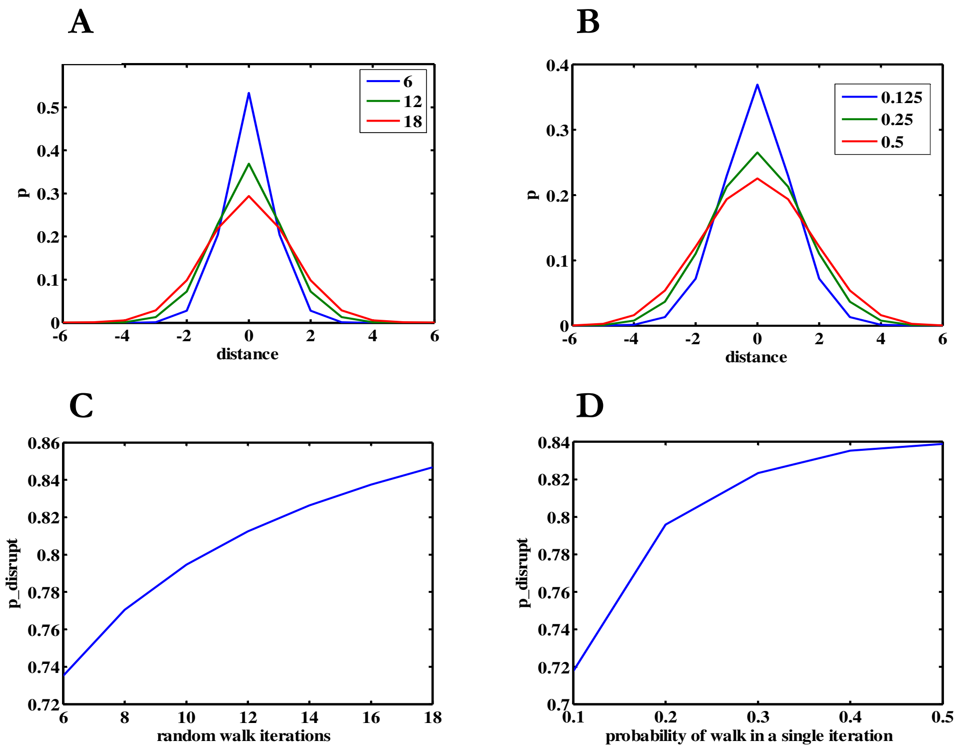


Figure 11 Role of different parameters of the random walk. **A** The probability distribution of the walk distance of shifted spikes from their original location after 6,12 and 18 iterations. The probability of shift by iterations: μ =0.125 **B** as in A, for different values of μ, after 12 iterations. **C** The probability that a given coincident pattern between 2 processes disappears after the random independent walk of the constituting spikes as a function of the number of iterations: μ =0.125 . **D** The probability that a given coincident pattern between 2 processes disappears after the random independent walk of the constituting spikes as a function of step probability (μ) after 12 iterations.

Constellations of complexity higher than 2 in independent multiple parallel processes are relatively rare. However, if they occur, they are very likely to be detected as false positives in data sets of finite length [Roy et al., 2000]. In order to account for that, it is advisable to apply an additional test at a meta-level, for example, by requiring a minimal absolute number of occurrences of the high-complexity event. According to (3.13) and Figure 12 one can set such a limit by varying the parameters of random walk: decreasing the walk probability μ, the probability that a single synchrony pattern will not be disrupted is increasing.

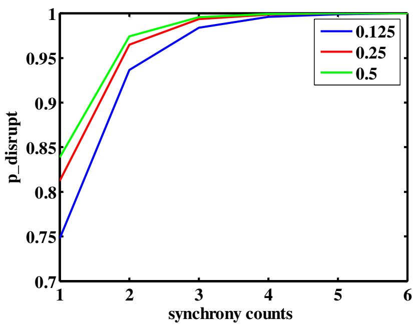


Figure 12 The probability that at least one coincident firing pattern between 2 processes disappears after the random independent walk of the constituting spikes as a function of the total number of synchrony patterns in the data. The different colors denotes three different step probabilities μ.

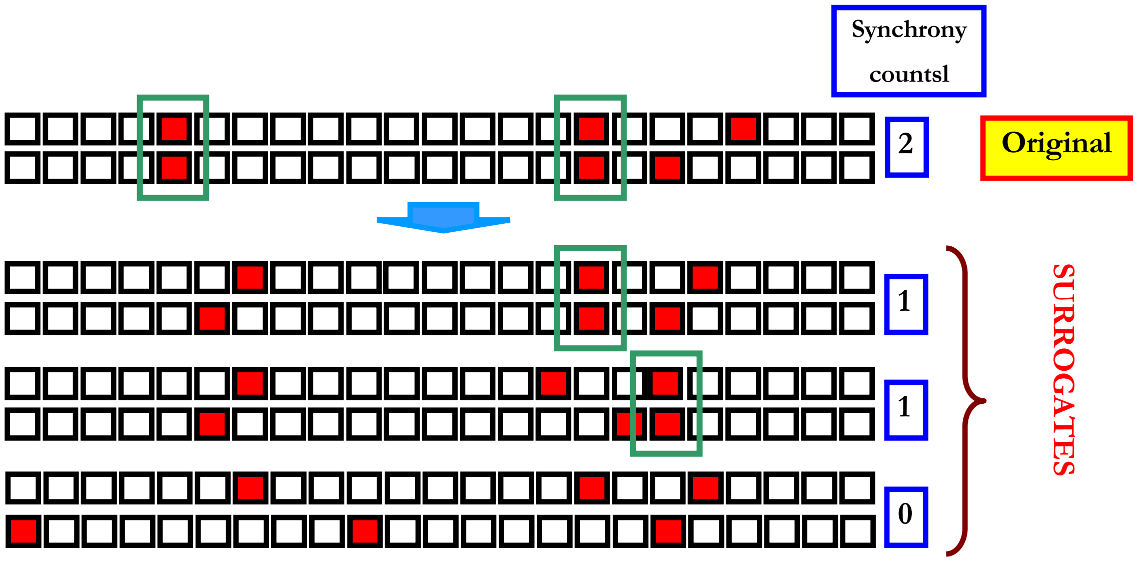


Figure 13 Creation of sampling distribution: original data consists of 2 dependent parallel processes, the surrogates are generated from it via random spike shifts. The number of coincidences is decreased in each surrogate comparing to the original train.

3.2.5 Detection of near coincidences

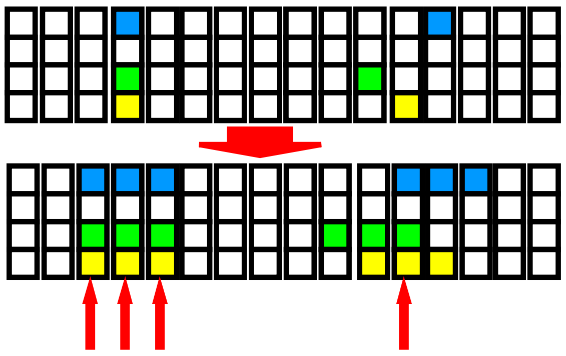


Figure 14 Spike elongation for the detection of near coincidences

To detect coincidences on a broader time scale – searching for less accurate synchrony (jitter) – spike representing pixels have to be elongated along the time-axis (see Figure 14). Thus, spikes (of different neurons) close to each other will overlap in time which enables us to recognize near-coincidences. The amount that spikes are expanded (*el*)is the function of the proposed jitter window – the level of inaccuracy we accept. From Figure 14 it is not difficult to understand that precise patterns are weighted stronger than less accurate ones in the analysis. When we apply this technique, the subsequent processing steps should be adapted and tuned to the modified input structure.

Figure 15 shows the integration of this spike elongation step into the coincidence pattern analysis explained above.

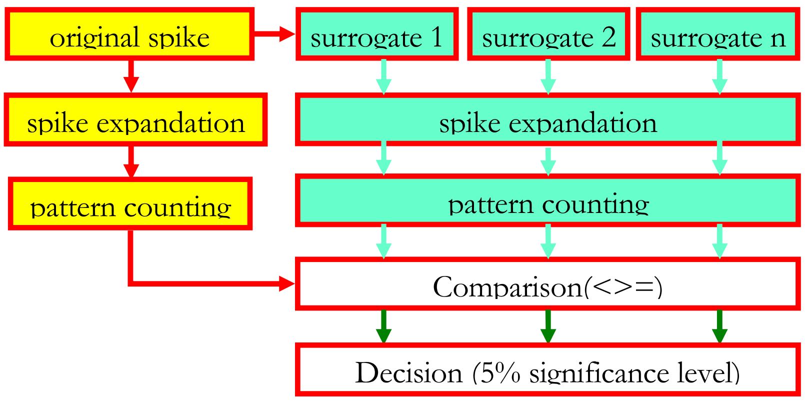


Figure 15 Outlines of the algorithm

3.2.6 Detection of higher order spatial-temporal patterns

Here I call the special constellations of spikes 'higher order patterns' where not only coincident firings but the temporal order of spikes matters as well. Although I mentioned in the previous sections that concerning more than two neurons assessment of all the possible higher order patterns become increasingly complex in an exponential manner, there is a simple solution within the framework of surrogate data generation to decide whether there is such a pattern within a time window with a moderate width.

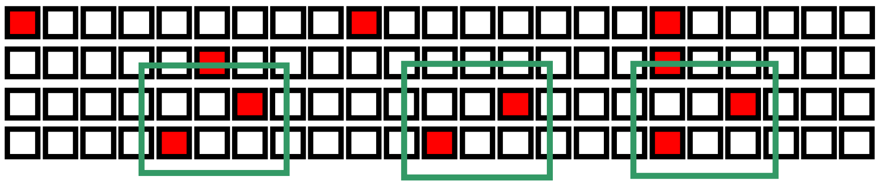


Figure 16 Higher order patterns: there is a clear tendency that spikes of neuron 4 are followed by spikes of neuron 3 after 2msec

Hypothesis:

Coincidence patterns that are less frequent in the original data than in the surrogates are suspicious to be part of a higher order pattern.

Proof:

To understand this idea we have to look into the mechanism of surrogate data generation. Normally, if there are totally independent processes, then the independent perturbation of spike times results in approximately equal number of pattern formation and destruction.

$$p_{formation}(cp) = p_{destrucion}(cp)$$

In other words, concerning a pair of cells if there are j spots in the spike-trains where spike-pairs (one from each neuron) are close to each other (within the maximum walk distance) to form higher order patterns, existence of all possible patterns (all possible delays between the spikes) have the same probability, provided the two cells fire independently. Denoting the number of all possible delays with m, we probably find approximately j/m events of all patterns – including exact coincidence. Randomly shifting of the spikes will not change this even distribution, the expected numbers for different patterns are j/m still.

However, when there are excessive numbers of coincidence patterns due to highly correlated activity – the distribution is biased towards coincidence patterns (Ncp –number of coincidence):

$$j/m < Ncp$$

and

$$j/m > Nhp$$

for higher order (other than coincidence) patterns hp. Random shift of the spikes will tend to reverse the even distribution, so the expected numbers for different patterns will be j/m again. Thus the destruction will dominate over formation.

$$p_{formation}(cp) < p_{destrucion}(cp)$$

There are many more ways a (near) coincidence can be disrupted than created from spikes close to each other. It is true provided that the jitter in the synchronous firings has even distribution (there is no bias towards a certain spike order, but a symmetrical variance around precise synchrony).

Provided, that there is a strong tendency that spikes of one neuron precede spikes of an other neuron –that is there is a strong bias in the variation of the relative spike times across neurons – the distribution is biased towards a specific higher order pattern hp_θ:

$$j/m < Nhp_\theta$$

and

$$j/m > Ncp$$

Random shift of the spikes will tend to reverse the even distribution, so the expected numbers for different patterns will be j/m again. Thus the formation of coincidence patterns will dominate over destruction:

$$p_{formation}(cp) > p_{destrucion}(cp)$$

From the equations above it follows that in certain situations detection of higher order temporal patterns is possible by the exploration of only coincident patterns: patterns that are less frequent in the original data than in the surrogates are suspicious to be part of a higher order pattern.

3.2.7 Evaluation of the method

Stationary Data

Dependence of joint-Surprise on Physiological Parameters

Having derived the joint-surprise (see Eq. (3.10)) as a measure for statistical significance of joint spiking events, we now investigate its performance with respect to various physiologically relevant parameters: the firing rates of the neurons under consideration, the time resolution (spike elongation length) chosen for the analysis, the rate of spike coincidences, their coincidence accuracy (allowing the biological system some degree of noise), and the number of neurons involved. To this end, we calibrate the performance of the joint-surprise by applying it to appropriately designed sets of simulated data. As before, the control data sets consist of independently generated Poisson trains of varying base rates. These are compared to different data sets, containing additionally injected coincidences of varying complexities and coincidence rates. Typically, the simulated data consisted of M=10 trials of 1000ms each and a time resolution of h=1ms. The rates of the random Poisson trains were chosen to cover a physiologically realistic range for cortical neurons-between 10 and 90Hz.

Influence of the Firing Rate.

To investigate the influence of the neurons' firing rates, we studied two parallel spike trains generated as independent Poisson processes, with both the same and constant rate. We varied this rate from f=10 to f=90Hz in steps of 10Hz, in the presence of different constant injection rates λ_c. Expectation values for the number of coincidences in the data set were estimated based on surrogate data generation.

To visualize the effect of statistical fluctuations, we generated 10 data sets for each rate level. Figure 18 shows that the empirical numbers of coincidences in the original data set indeed match the mean of the bootstrap distribution assuming independence ($S=0, \Psi=0.5$), apart from small statistical fluctuations. The number of coincidences exhibits a convex dependence on background firing rate. The increase is quadratic, being the coefficient of the leading power.

Closely related to the probability of obtaining false positives (significant outcome in the absence of excess coincidences) is the question of how precisely the distribution of equivalent independent processes can be estimated when no coincidences are injected. We present an analysis of the relation of significance level to the percentage of false positives obtained in independent data sets later in this section.

Figure 17 shows the joint-surprise values corresponding to the original and surrogate pairs. Without injected coincidences, the joint surprise fluctuates around 0, independent of the rate, due to the fluctuations in the generated sampling distribution. Because we necessarily have fluctuations in the surrogate samples the percentage of experiments in which the coincidence count is significant may differ from the theoretical value determined by S. In the case of injected coincident events, the measured and expected coincidence counts deviate from each other, and the more so the higher the injection rate (see Figure 17 for λ_c=0.5, 1 and 2Hz). The joint surprise declines hyperbolically with increasing background rate. At very low background firing rate, the expected number of coincidences (mean of the sampling distribution) is practically 0, while the number of measured coincidences remains considerable. Therefore, the joint surprise obtains a large value.

For the injected coincident rate of λ_c=0.5Hz (Figure 17), the joint surprise falls below the significance level of 0.05 (horizontal line in bottom graph) at a rate of about 60 Hz. For the injected rate of λ_c=1Hz (Figure 17), this occurs only at a considerably higher background rate (above 90Hz). At higher firing rates, more excess coincident events are needed to escape from the statistically expected fluctuation range. Clearly, this behavior imposes a severe limit on the detectability of excess coincidences at high firing rates. Before the expectation of the joint-surprise falls below the significance threshold, some of the joint-surprise values obtained in the individual experiments has already reached it (see deviation bars in Figure 17 A and C, 40Hz and 70Hz, respectively).

However, there is a broad range where fluctuations in the joint-surprise are well separated from the significance threshold, and, hence, excess coincidences can reliably be detected. When injected coincidences are present, the difference between the mean of bootstrap and actual number of patterns increases linearly with Δ (observed time-interval), while the width of the bootstrap distribution increases with $\sqrt{\Delta}$. Thus when data is long enough, excess coincidences can always be detected.

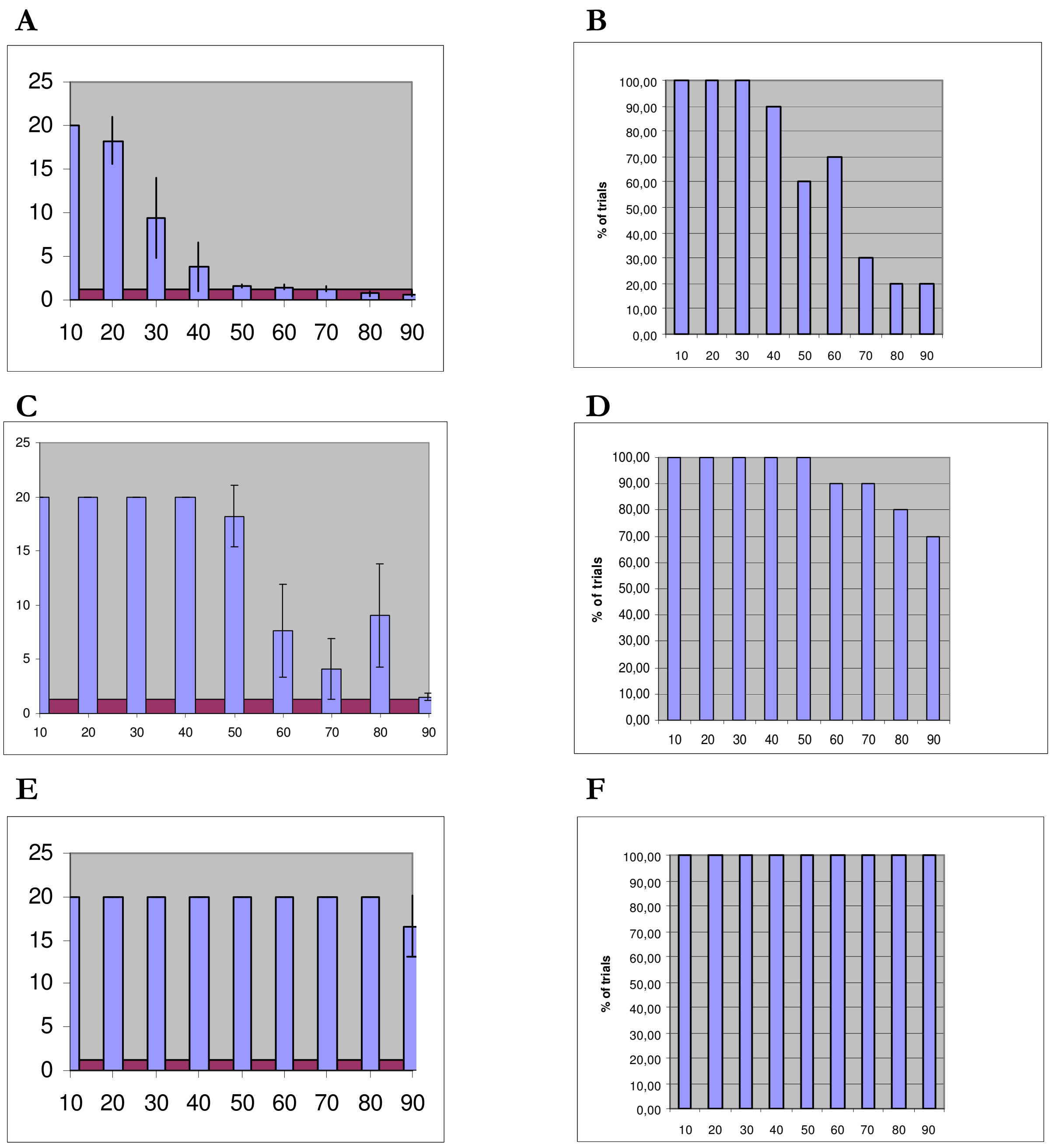


Figure 17: Detection of coincident events under different firing-rate conditions. Simulated data from two parallel processes were analyzed for the presence of coincident events. Realizations consisting of 100 trials, each of duration 1000 ms, were generated with a time resolution of 1 ms. Rates of both processes were varied from 10 to 90Hz in steps of 10Hz. The experiment was repeated 10 times at each rate, to visualize statistical variations. Coincident events, also generated by Poisson processes, were injected into each of the 10 data sets at one of three coincidence rates (**A**,**B**: 0.5Hz, **C**,**D**: 1Hz **E**,**F**: 2Hz). Data were analyzed for the number of empirical occurrences versus the sampling distribution from surrogates. The corresponding joint-surprise is shown in the left panels, percent of trials with successful pattern detection in the right panels. Results for the 10 realizations per firing rate are averaged together,. Horizontal red band in the panels indicate the significance threshold (S =1,2788 Ψ =0.05).

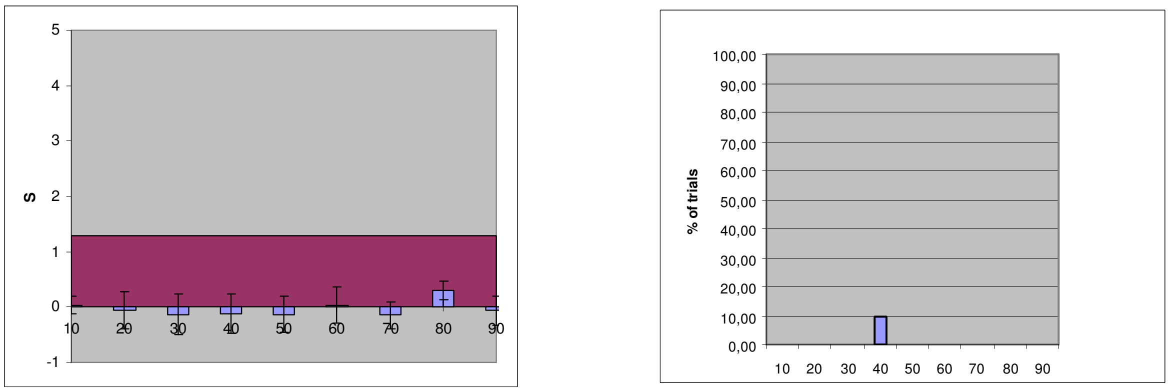


Figure 18 The results of the control experiments without injected events.(for the interpretation see Figure 17).

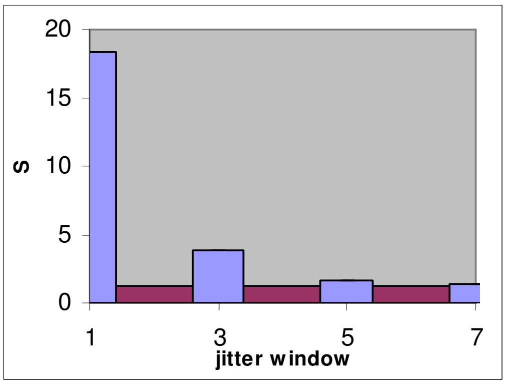
Influence of spike elongation.

The time resolution of data acquisition in extra-cellular spike recordings is typically 1 ms or better. There is recent experimental evidence from cross-correlation, joint peristimulus time histogram JPSTH, and, particularly, from spike pattern analysis, that the timing accuracy of spiking events that might be relevant for brain function can be as precise as 1-5 ms [Abeles et al., 1993a; Riehle et al., 1997]. Similar suggestions come from modeling studies (Diesmann et al., 1999). Here, we want to investigate whether, by choosing the spike elongation length el in that time range, we may be able to detect coincidences with corresponding accuracy. Therefore, we will first study the general influence of elongation on the outcome of joint-surprise analysis and then address the effect of varying elongation size on the detection of coincidences with a finite temporal jitter.

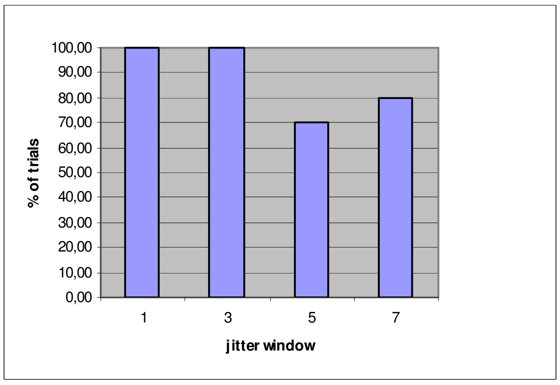
We generated a set of simulated data as before. While the rate of the independent processes was maintained constant (*f*=20Hz), we injected additional coincident events at various rates. Two examples for coincident rates of λ_c=0.5Hz and 1.0 Hz are shown in Figure 19; the control set is shown in Figure 20. In the analysis, we gradually increased the elongation length from *el*=1 to *el*=7. This newly generated process formed the basis of our investigation.

Elongation increases the probability to observe an event in a time step, compared to the original probability.

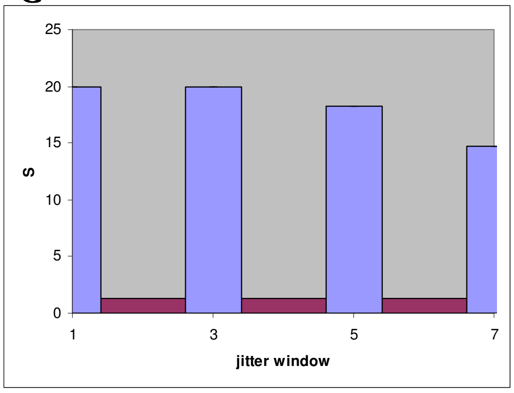
A



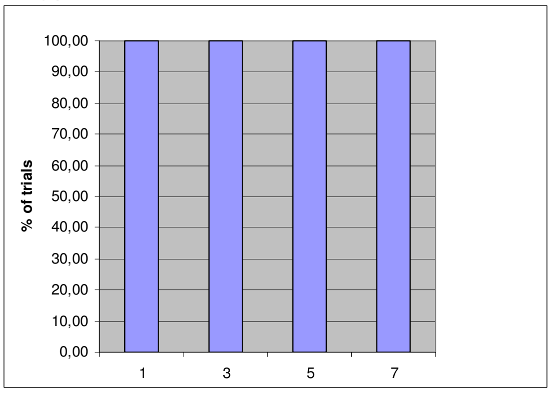
B



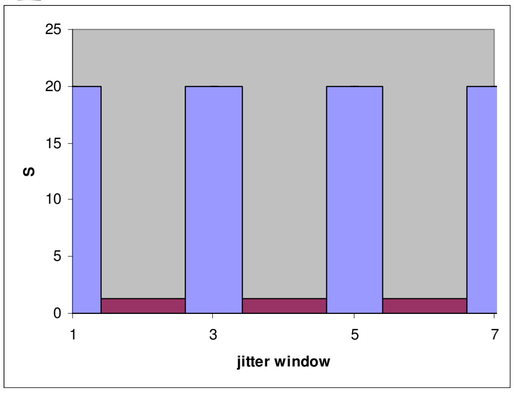
C



D



E



F

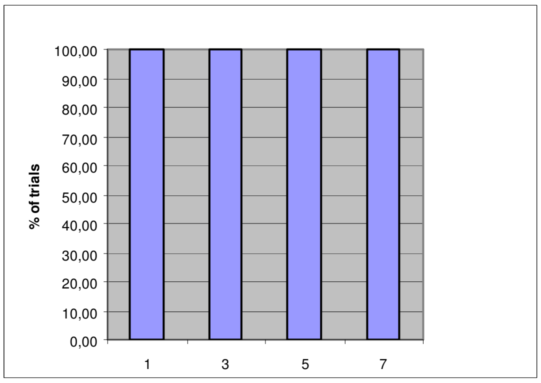


Figure 19: Detection of coincident events using different spike elongation lengths el (or jitter window). Simulated data from two parallel processes were analyzed for the presence of coincident events. Realizations consisting of 10 trials, each of duration 1000 ms, were generated with a time resolution of 1ms. Rates of both processes were kept constant at 20Hz. Coincident events were injected at one of three coincidence rates (**A**,**B**: 0.5Hz, **C**,**D**: 1Hz **E**,**F**: 2Hz). The experiment was repeated 10 times at each elongation length, to visualize statistical variations. Elongation length was varied from 1 to 7 ms. Data were analyzed for the number of empirical occurrences versus the sampling distribution from surrogates. The corresponding joint-surprise is shown in the left panels, percent of trials with successful pattern detection in the right panels. Results for the 10 realizations per firing rate are averaged together. Horizontal red band in the panels indicate the significance threshold (S =1,2788 Ψ =0.05).

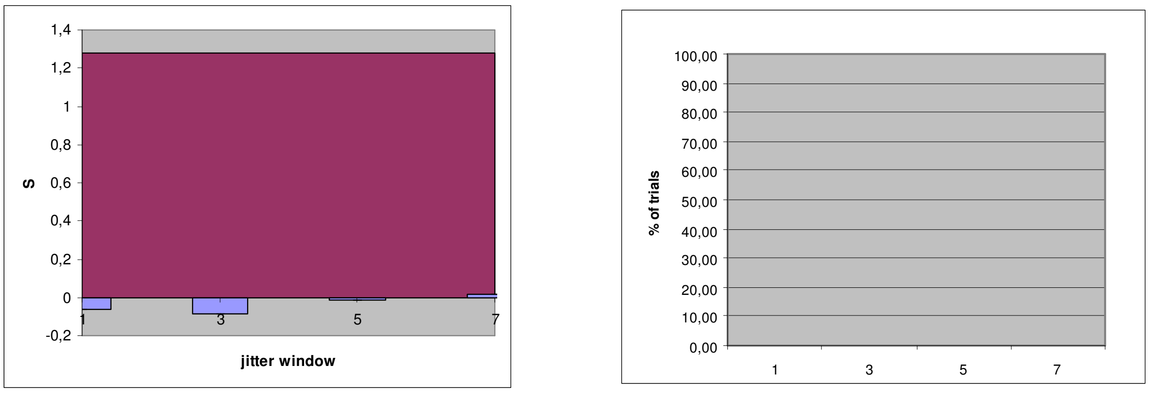


Figure 20 The results of the control experiments without injected events.(for the interpretation see Figure 19)

The dependence of the joint-surprise (see Figure 19) on the elongation size is similar to the above-described dependence on the rate. In the absence of injected coincidences, S fluctuates around 0 (see Figure 20). For injected coincidences, S decreases with increasing bin size. The lower the injection rate is, the sooner S starts to decrease and the faster it decays: for λ_c=0.5Hz, joint-surprise values start to fall below the 0.05 significance level at *el*=5-7 (see Figure 19), while for λ_c=1Hz, significance is maintained even at *el*=7 (see Figure 19). The similarity between the dependences of S on spike rate and on elongation size is not surprising, considering that elongation has the net effect of an apparent increase in firing probability.

Detection of Near-Coincidences.

In the next step, we investigate whether it is also possible to detect noisy (i.e., imprecise) coincidences. This question arises naturally, since neurons are usually considered to exhibit some degree of "noise" or uncertainty in the timing of their action potentials. Note, however, that the degree of this temporal noise has long been questioned [Abeles, 1983] and is still under debate, [Shadlen & Newsome, 1998; Diesmann et al., 1999]. While keeping both the independent background rate and the injection rate constant, we increase the temporal jitter of the injected near-coincident events stepwise from 0 to 5 ms, such that in each case, the difference in spike times is uniformly distributed within the chosen jitter range. The question is whether, by choosing an appropriate elongation length (*el*), we can improve the detection of such near-coincident events. To this end, we analyze the simulated data with varying *el* sizes and for each bin size compute the joint-surprise.

Figure 21 shows the results for a background rate of *f*=20Hz and a rate of injected near-coincidences λ_c=2Hz. Each of the curves in Figure 21 represents data with a particular temporal jitter ν=0 to ν=6ms, analyzed with elongation length increasing from 1ms to 7ms. Values of S are averages of 100 repetitions of the simulation experiment at constant parameters. Curves representing temporal jitter ν=2-6ms raise with increasing *el*, up to *el*=5, and then fall with different rates: the larger the temporal jitter the smaller the fall. That is at ν=5ms the value of S will not drop significantly (compared to ν=3ms) when increasing *el* from 5 to 7 ms .

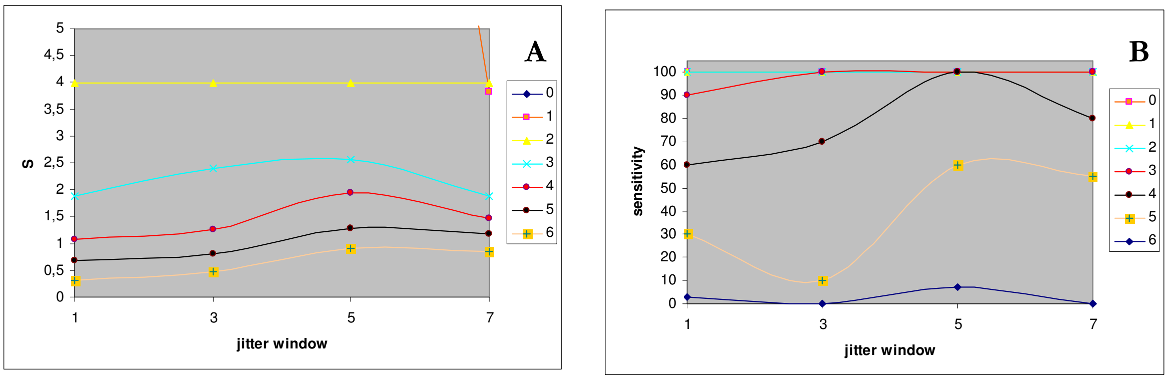


Figure 21: Detection of near-coincidences for different degrees of coincidence precision. Two parallel spike trains were generated with background rates f=30Hz the rate of the injected coincidences was λ_c=2Hz, for Δ=10000 time steps, *h*=1 ms. The temporal jitter of coincident events was varied from ν=0 to ν=6 time steps. Each simulation was repeated 100 times, and data were analyzed for the number of observed coincidences by varying the spike elongation size (jitter window) from *el*=1 to *el*=7, and compared to the bootstrap distribution based on surrogate data. (A) Each curve shows the resulting average joint-surprise as a function of the el for a given temporal jitter.

We can conclude that at elongation lengths larger than the coincidence accuracy, the rate at which the number of excess coincidences grows drops, and the probability that an injected coincidence is detected reaches saturation. Thus, S is decreasing again, because the bootstrap width (standard deviation) is increasing. The comparison of different joint-surprise curves shows that the higher the temporal jitter (i.e., the lower the coincidence accuracy) is, the lower the joint-surprise is. Hence, for a given el, the number of near coincidences that can be detected increases with decreasing temporal jitter.

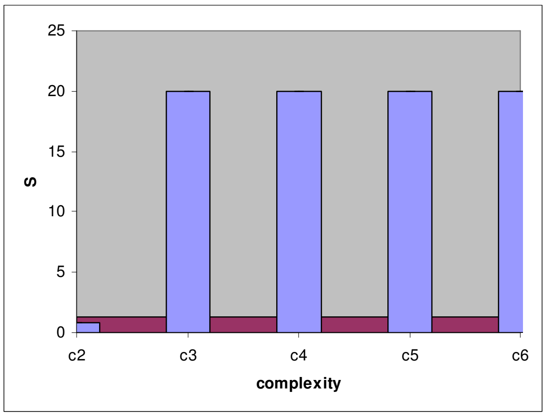
Multiple Parallel Processes.

When the number of simultaneously observed neurons N is increased, the variety of coincidence patterns grows strongly, due to the nonlinear increase in combinatorial possibilities. Each complexity ξ (activity pattern with ξ spike and N- ξ non-spike entries) can occur in

$$\vartheta = \binom{N}{\xi}$$

variations. On the other hand, the occurrence of higher-order constellations depends in a nonlinear fashion on the rates. The probability for a pattern with complexity ξ to occur among N neurons, all firing independently with probability p, is given by $p^{\xi}(1-p)^{N-\xi}$. For low firing rates, constellations of high complexity are actually expected to occur rarely, so here the measure of S could be subject of fluctuations. A solution to this problem is to collect more data for such an experimental situation, where the discrete nature of the distribution is of less importance.

High ξ constellations, if occurring at all, are typically accompanied by high joint-surprise values (see Figure 22). It is therefore not surprising that in simulations where we varied the complexity of the injected coincidence patterns from ξ=2 to ξ=6 –while keeping the number of processes (*N*=6), the background rate (*f*=20Hz) and the injection rate (λ_c=1Hz) constant–, all coincidences of complexity ξ=3 and above were detected with high significance (see Figure 22). For complexity ξ=2, the coincidence count is higher, because we get contributions from the background rate which is rapidly vanishing for higher complexities.



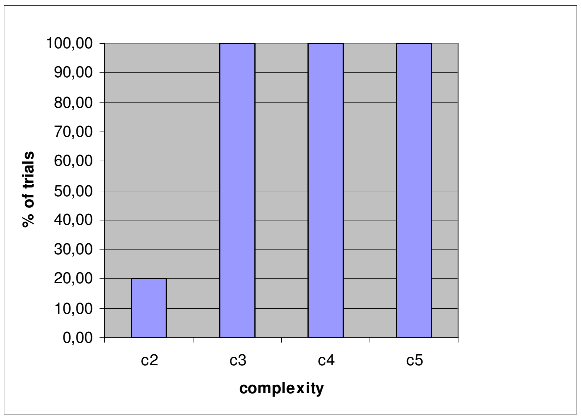


Figure 22 Complexity of joint spike patterns. The background rate of the independent processes was *f*=20Hz, and the injection rate was 1Hz (Δ=100000ms, 10 repetitions). The number of processes was kept constant (*N*=6), but the complexity of the injected coincidences was varied from 2 to 6. Thus, the pattern looked for was [110000], [111000], and so on, respectively. The corresponding joint-surprise was very high except for complexity ξ=2 (denoted by c2 in the figure).

When we increase the number of independent processes *N* (from 2 to 12), while keeping the complexity of injected coincidences constant (ξ=2) the joint-surprise at the given firing rates is practically independent of *N*. There would be a small decrease in the number of occurrences of this particular pattern, because with increasing *N*, more patterns containing the two spikes as a subpattern become available, but our algorithm counts the subpatterns as well. Thus, this effect does not affect the detectability of the injected coincidences at all.

We can conclude that to decide on the empirical relevance of coincidences of higher complexities (ξ>=3), given a moderate amount of data, it is advisable to set additional criteria, for example, by requiring a minimum absolute number of occurrences [Martignon & Vaadia, 1997; Martignon et al., 2000]: it is an inherent feature of the random walk process we tuned: we use a quite low step probability which results in a high probability that spike will not move at all.

False Positives

Up to now, we have studied the sensitivity of our method by exploring under which conditions excess coincident events are detectable. However, usually high sensitivity means low selectivity (or specificity). Thus when we have a low fraction of false negatives one can suspect high number of false positives. Thus, we have to establish conditions under which we reach a compromise between a sufficient degree of sensitivity and an acceptable degree of specificity. Therefore, we now analyze various sets of simulated data, with the combined requirement of attaining a high level (90%) of detection (only 10% false negatives), while securing a low level (10%) of false positives. As in the preceding sections, the simulations are described by biologically relevant parameters, varied over a physiologically realistic regime. 100 independent experiments were performed for each parameter value; from these, the percentage of experiments that crossed a certain threshold level on the joint-surprise was evaluated. This threshold level was varied in equidistant steps to cover the range of joint-surprise values between -15 and +15.

Influence of the Number of Parallel Processes.

Next, we varied the number of independent processes (from 3 to 12) while keeping the rates constant (*f*=20Hz). For each number of processes, the fractions of false positives and false negatives were evaluated at different threshold levels. We found that the fraction of false positives increased with decreasing threshold and with the number of processes involved (Figure 23, middle). Moreover, the sensitivity for excess coincidences (shown for complexity ξ =2 at coincidence rate of 1Hz in Figure 23, top panel) was independent of the number of processes. The intersection range of the joint-surprise, necessary to obtain maximally 10% false positives and maximally 10% false negatives, is shown in white in Figure 23 (bottom panel). Observe the narrow band for selective and sensitive detection, dependent on the number of observed processes. If more restrictive criteria (fewer false positives and/or fewer false negatives) are adopted, the band becomes accordingly smaller (not shown here).

Influence of the Firing Rate.

In the first step, we kept the number of independent processes constant ($N = 2$) and varied the rate of the processes. We found that for constellations of ξ=2, the percentage of false positives are practically independent of the background rates and has a linear relation to S threshold.

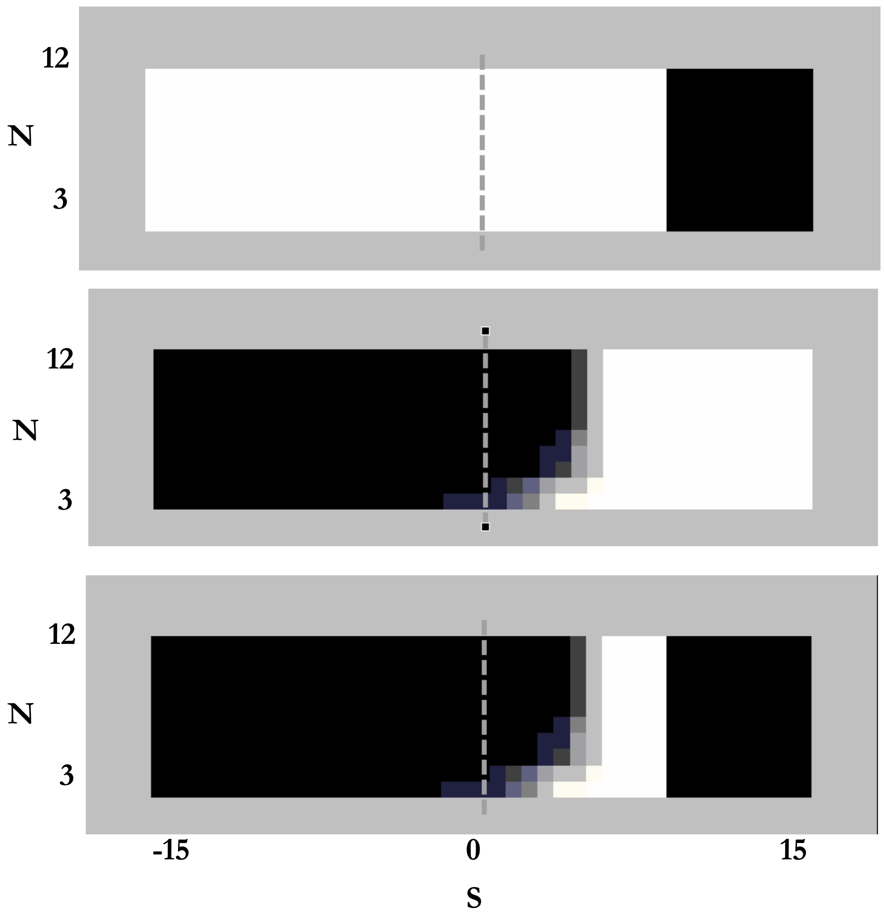


Figure 23 Sensitivity/specificity profile: white denotes min 90% correct as a function of Joint Surprise (S) and the number of processes (N) involved. Upper part: sensitivity (fraction of true positives/all positives) Middle part: selectivity (fraction of true negatives/all negatives) Bottom: combined performance: there is a secure white band where both sensitivity and selectivity are in an acceptable range.

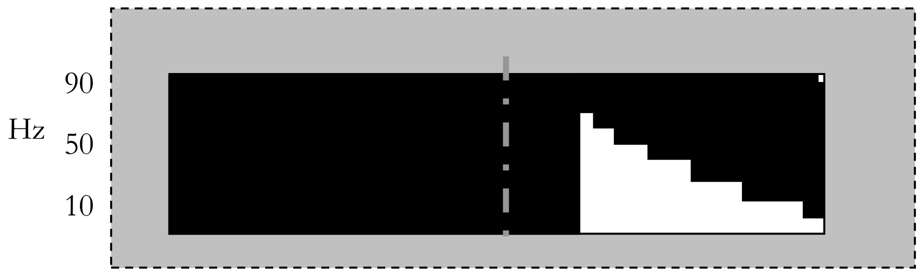


Figure 24 Combined specificity/sensitivity profile. white area denotes min 90% true positives and 90% true negatives at different joint surprise (*S*) levels. The number of processes (*N*) is varied from 10 to 90Hz.

By contrast, the sensitivity for detecting excess coincidences shows a clear dependence on background rates. At low rates, it is very high, but it decreases-rapidly at first, more slowly later-with increasing background rate (see Figure 24). At background rates above *f*=70Hz, the threshold for detecting the injected events has decayed to about α =0.05. Combining these two observations in a single graph, we obtain the intersection range of the joint-surprise, necessary to obtain both maximally 10% false positives and minimally 90% sensitivity (the white area in Figure 24 top). For low α, this region is bounded by an approximately straight vertical line at α =0.05; the lower boundary of the permissible significance measure is approximately independent of the background rate. The upper bound, however, is clearly curved: the threshold needed for reliable detection decreases with increasing background rate, reaching a level of only 0.05 at *f*=60Hz. Thus, the higher the rate is, the narrower is the bandwidth of α-values permissible to detect excess coincident events selectively and sensitively.

Detection of higher order spatial-temporal patterns

To test and evaluate the hypothesis we simulated data from two parallel processes and injected higher order patterns with different inter-spike intervals: 1,2 3 and 4msecs between the spikes of neuron 1 and neuron 2. Realizations consisting of 100 trials, each of duration 1000 ms, were generated with a time resolution of 1 ms. Rates of both processes were kept constant at 20 Hz. Higher order events were injected at different constant rates (A,B: 0.5 Hz, C,D: 1Hz E,F:2Hz). The experiment was repeated 10 times at each inter-spike intervals, to visualize statistical variations. Data were analyzed for the number of empirical occurrences versus the sampling distribution from surrogates. The corresponding joint-surprise is shown in Figure 25. Results for the 10 realizations per spike intervals are averaged together,. Horizontal red band in the panels indicate the significance threshold (S =-1,2788 Ψ =0.95). Negative S values crossing the threshold mean that coincident events occur less frequently than one could expect considering the given background rates or looking at the bootstrap distribution of surrogates. Figure 25 shows that patterns of up to 3ms width can be recognized applying this principle.

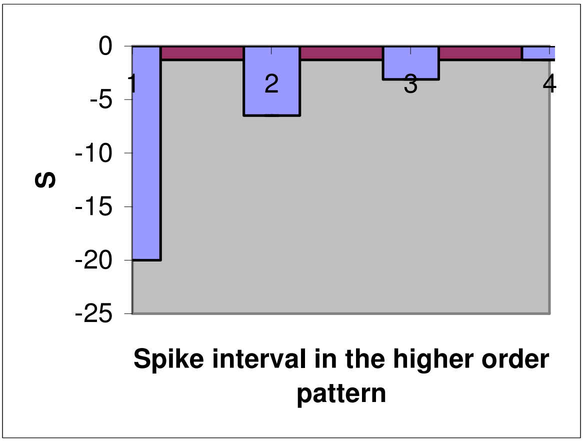


Figure 25 Performance of the surrogate method for the detection of higher order patterns

Non-stationary Data

Until now, we have considered only the case of neurons firing at a stationary rate and with stationary coincident activity among them. Physiological data, however, are usually not stationary. Firing rates vary considerably as a function of time, particularly when the subject is presented with adequate stimuli or is engaged in a behavioral task. A second type of non-stationarity is that coincident firing itself may be non-stationary for example, by being time-locked to a stimulus or behavioral event even if the rates of the neurons are constant (Vaadia et al., 1995).

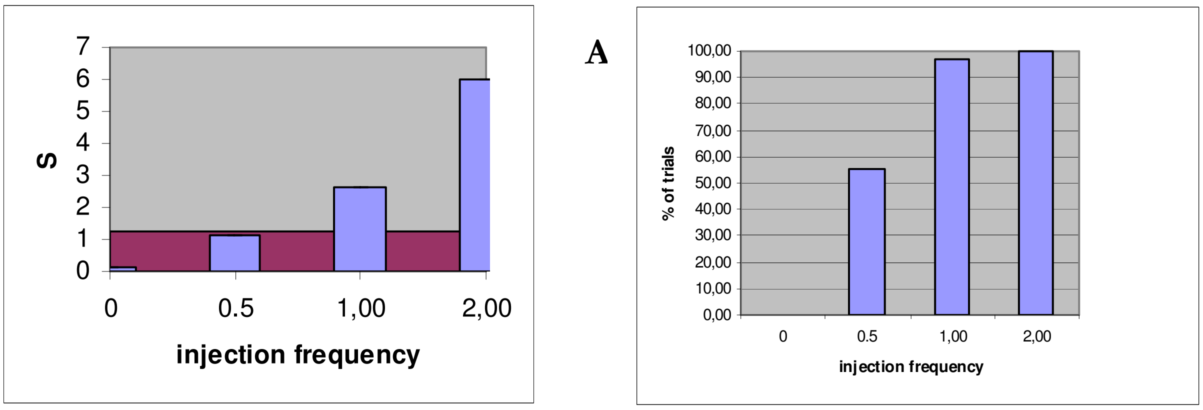


Figure 26 Non-stationary data: stepwise increase in background frequency. Influence of the injection rate of dependent patterns on performance.

One non-stationary example is shown in the Application Chapter 6, here we present the performance of the algorithm on stationary coincidence rates combined with stepwise increasing background rates (10, 20, 30, 40, 50, 60, 70, 80, 90 Hz). Coincidences are injected at rate 0, 0.5, 1 an 2 Hz. The experiment with 100 trials(1000ms long each) was repeated 100 times. As we can see (Figure 26) from the injection rate of coincidences λ_c=1Hz the method is quite robust on non-stationary data as well.

3.3 Generation of Surrogate Data via CNN-UM

Our primary intention is to preserve the main structure of the individual neuron activity records but disrupt fine interdependence between them. The most important features to preserve:

- total number of spikes
- non-stationarity; even sudden, transient changes in frequency

The key concept is explained in Section 3.2.4: repeatedly shifting single, randomly selected spikes in the original trials against each other in the time domain (see Fig. 27). This will result in independent random walk-like motions of the individual spikes.

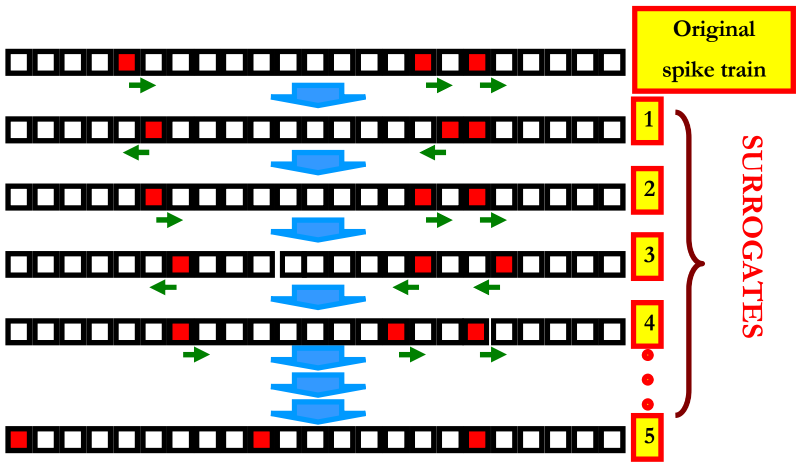


Fig. 27. Surrogate data generation: shifting randomly selected spikes forward and backward in turn. Green arrows depict a shift with a certain probability.

By means of this slight independent random modification we could destroy excess of coincidences keeping the actual frequency unchanged.

For repeated random spike (pixel) selection within the framework of CNN-UM we need a random binary pattern generator algorithm which works on the CNN-UM. An excellent method was proposed by Crounse and his colleges [Crounse et al., 1996]. Since CNN-UM can emulate certain classes of cellular automata (CA) [Wolfram, 1994] they examined a subset of two-dimensional binary CA rules, and found the following satisfactory:

$$c_{i,j}(n+1) = \left(c_{i+1,j}(n) + c_{i,j+1}(n)\right) \oplus c_{i-1,j}(n) \oplus c_{i,j-1}(n) \oplus c_{i,j}(n) \tag{3.16}$$

where $\oplus$ stands for the exclusive-or logical operator, $c_{i,j}(n)$ is the (0,1) state of the cell with (i,j) coordinates at step n. From that point CA refers to this specific operator throughout the following sections and chapters. This irreversible rule works within the nearest neighborhood; moreover, an iteration can easily be implemented by two templates and some local logical operations on an ACE4K based system [Szatmári et al., 2000; Liñán G. et al. 2002]. Even when one chooses a very simple, highly ordered initial condition (seed), the iterated application of the rule drives the system quickly to a high entropy state. The behavior of the system was thoroughly studied by the inventor, and passed a great number of statistical randomness tests: the subsequently generated black&white images revealed regularity neither in space nor in time.

Such an image – in the following it is denoted by R.–consists of independently selected black or white pixels. If the probability of a black pixel is p, then the logic operation between two Rs results a new random image with black pixel probabilities: XOR: p, AND: p^2, OR: $1-(1-p)^2 = p(2-p)$. Starting from a nearly arbitrary seeding image after a few (~100) iterations Eq.(3.16) generates evenly distributed random patterns with p=0.5. In the following R^i denotes the logic operation AND between i different random images, and the probability of having a black pixel in a given position is a function of the required number of iterations of the CA rule (i.e., the computational cost, see Figure 29):

$$p(i) = \left(\frac{1}{2}\right)^i \tag{3.17}$$

Given the above mentioned features it is reasonable to employ this CA as our pseudo-random pixel selector. R^i can be applied as a mask (see Figure 28) with a predefined density of black pixels: for our purpose it is typically 1/8 (p=0.125) which requires R^3 as it can be seen in Figure 29. The quality of randomness could be even better if some of the subsequently (iteratively) created patterns are ignored and only every second (or n^th) pseudo-random images are used for pixel-selection.

A possible way of usage: at every odd step only those spike-representing pixels are shifted to the right (forward in time), whose neighbors to the right are silent (white) and these neighbors are covered by the pixels in the actual random mask. The operation at even steps is basically the same, except its direction is changed to backward.

The actual steps of our procedure are as follows: starting from an original segment of recorded data (as seed) a couple of hundreds of CA iterations are executed according to Eq.(3.16). From this point R^3 samples from subsequent iterations will give properly distributed pseudo-random binary arrays. The shift operation – with alternating directions – is masked by the repeatedly generated random image, and following required number of spike elongation operations the binary activity patterns should be counted in the newly formed surrogate data (see Figure 30).

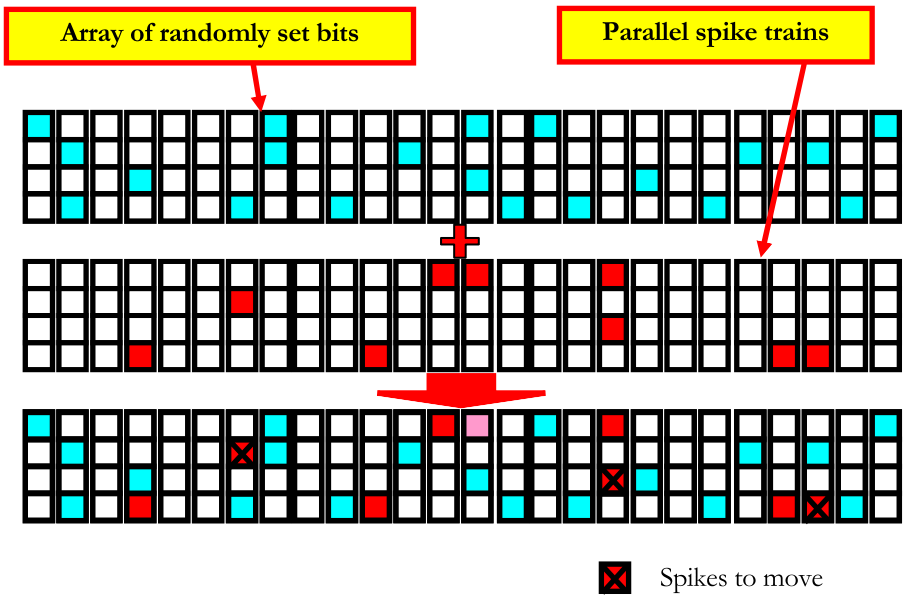


Figure 28 Randomly selecting spikes to shift forward in an array of 4 spike trains.

In that way the independent motion of individual pixels is not identical to but can be approximated by a ‘random walk’ process. To limit the distance they can travel from their starting point the input is reset to the original recordings after certain number of steps.

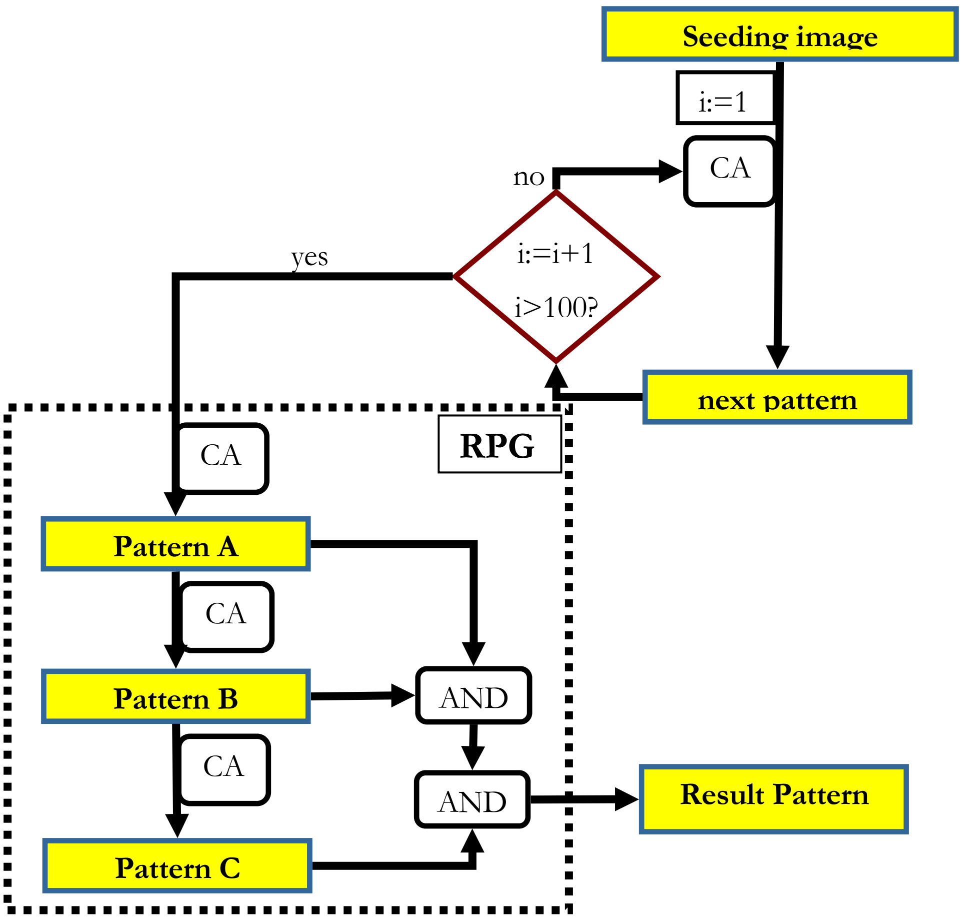


Figure 29 Creating pseudo-random patterns. RPG: pseudo-random pattern generator unit. CA is the cellular automata rule described by Eq (3.16).

Basically a 64X64 CNN can accommodate data with up to 64 msec duration, so switching between the 64msec segments means a very high data transfer overhead. Although using the periodic boundary condition with a meander-like arrangement this array can host up to 1024 samples of 4 neurons, it is still quite inefficient when processing data typically 10^5 samples long; beside periodic boundary is not safe to use when analyzing non-stationary activity.

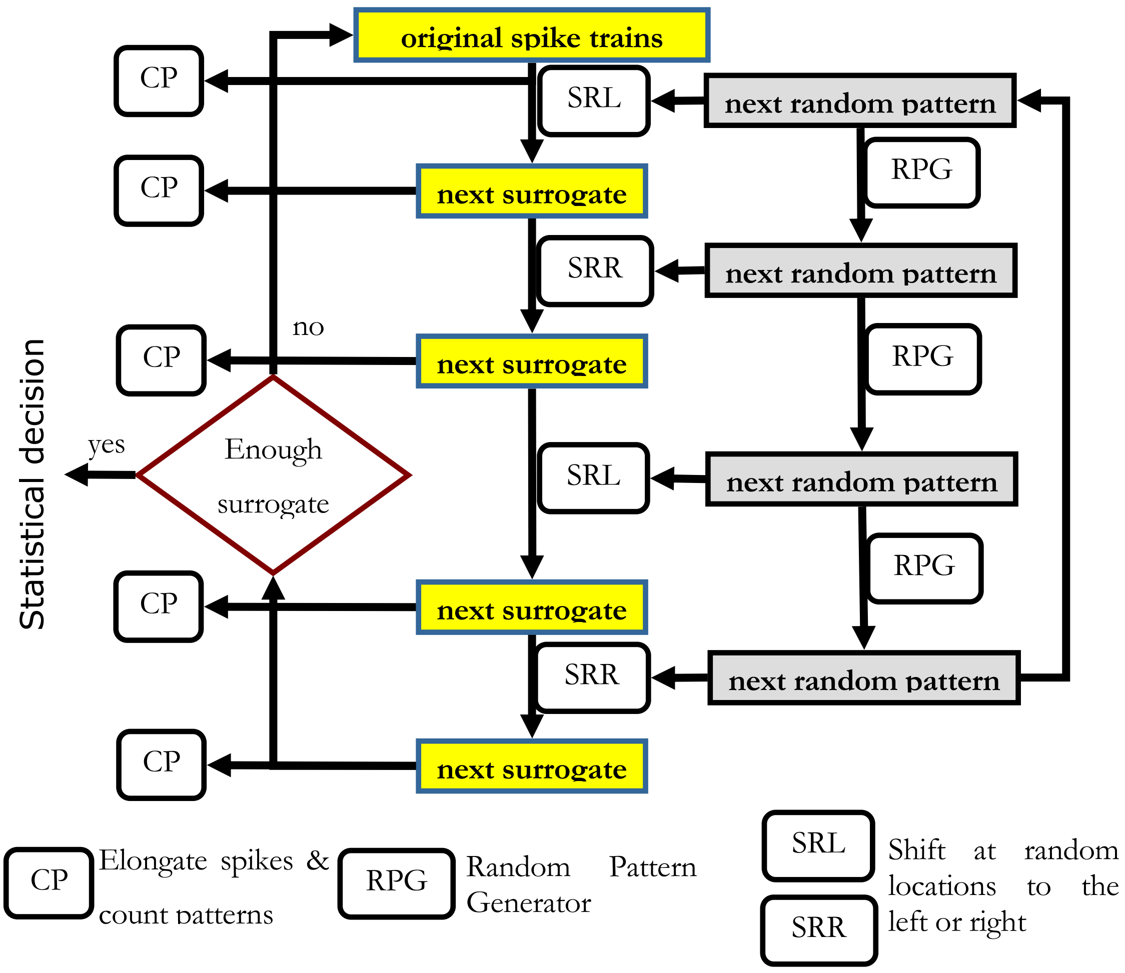


Figure 30 Flow chart of the algorithm. Figure 29 explains the operation of the pseudo-random pattern generator (RPG).

Results

The tests–using the Candy CNN-UM simulation software package of the Aladdin development system [Szatmári et al., 2002; Zarándy et al. 2003] and mathematical simulations on traditional PC–show that the analogic approach can perform equally good as traditional implementations of the surrogate data generation concerning short data segments. There were no significant differences between the quality of the statistics based on the surrogate data generated by standard conventional algorithms and created via the CNN-UM (experiments discussed in Section 3.2.7 were used for comparison).

The speed of a single iteration of surrogate data creation on traditional serial computers is not comparable to the ACE4K platform, especially in the fast DTCNN mode [Espejo et al.,1998]. However, the size of the current CNN-UM implementations makes the analogic computation ineffective mainly because of the communication overhead.

Table of the used templates:

Template name	A	B	Z	U	X_0
OR	2	1	1	P	Q
AND	2	1	-1	P	Q
DIF	2	-1	-1	P	Q
shift_right	0	[1 0 0]	0	P	0
shift left	0	[0 0 1]	0	P	0
match001	1	[0 0 1]	-0.5	P	P
match100	1	[1 0 0]	-0.5	P	P
dilation	1	[1 1 1]	1.5	P	P
CA1	1	0 0 0 0 1 -1 0 -1 0	-2.5	P	P
CA2	1	0 0 0 0 -1 1 0 1 0	-0.5	P	P

4 An Analogic implementation of the genetic algorithm

Second thesis:

Analogic implementation of a genetic algorithm

I took part in the development of an analogic method for an efficient implementation of a special 'fine-grained' type genetic algorithm. Beside the design and parameterization of the typical genetic algorithm operators a key issue was a scalable pseudorandom area selection within a CNN-UM array: I defined the optimal parameters for the method maximizing the probability that given an MxN array a single line can be selected minimizing the computational requirements at the same time.

4.1 Introduction

A genetic algorithm (GA) is a search technique to find approximate solutions to combinatorial optimization problems [Goldberg, 1989]. Genetic algorithms use techniques inspired by evolutionary biology such as inheritance, mutation, natural selection, and recombination (or crossover) [Mitchell, 1996]. A population of abstract representations (called chromosomes) of candidate solutions (called individuals) evolves toward better solutions. Traditionally, solutions are represented in binary as strings of 0s and 1s. The evolution starts from a population of completely random individuals and happens in generations. In each generation, the fitness of the whole population is evaluated; multiple individuals are selected, based on their fitness, from the current population, modified (mutated and recombined) to form a new population, which becomes current in the next iteration of the algorithm.

Genetic algorithms might be useful in problem domains that have a complex fitness landscape as recombination is designed to move the population away from local minima that a traditional hill climbing algorithm might get stuck in. Particularly appropriate problems for genetic algorithms include timetabling, scheduling problems, automated design and I will show that it can be effective in multi-dimensional (multi-channel) neural activity data analysis.

Parallel implementations of genetic algorithms come in two flavors. Coarse-grained parallel genetic algorithms assume a population on each of the computer nodes and migration of individuals among the nodes. Fine-grained parallel genetic algorithms assume an individual on each processor node which acts with neighboring individuals for selection and reproduction [Vose, 1999]. The presented algorithm is a special fine-grained implementation [Bálya & Gál, 2006].

4.2 Operation of a GA

An individual, or solution to the problem to be solved, is represented by a list of parameters, called chromosome. The algorithm represents each chromosome as a bit string. Typically, numeric parameters can be represented by integers, though it is possible to use floating point representations. These bit strings represent the chromosomes and each bit can be represented as a black or white pixel.

Initially several such chromosomes are randomly generated to form the first initial population. The population can be stored in an image, where each row is an independent chromosome.

During each successive generation, each chromosome is evaluated, and a value of fitness is returned by a fitness function. It requires an external evaluation process and we do not address the task specific fitness calculation question here. The fitness values are converted to a grayscale image, where each pixel in a given row has the same darkness. The darkness is determined as black is the best fitness value and the white as the least in the current population.

The next step is to create a second generation population of chromosomes, based on the processes of selection and reproduction of selected individuals through genetic operators: crossover and mutation. For each chromosome to be produced, a pair of parent chromosomes is selected for breeding. Selection is biased towards elements of the initial generation which have better fitness, though it is usually not so biased that poorer elements have no chance to participate, in order to prevent the population from converging too early to a sub-optimal or local solution. There are several well-defined organism selection methods e.g. roulette wheel or tournament selection.

Following selection, the crossover operation is performed upon the selected chromosomes. Commonly, genetic algorithms have a probability of crossover, typically over 70%, which encodes the probability that two selected organisms will actually breed. Organisms are recombined by this probability. Crossover results in new child chromosomes, which are added to the next generation population. The chromosomes of the parents are mixed during crossover, typically by simply swapping a portion of the underlying data structure. This process is repeated with different parent organisms until there are an appropriate number of candidate solutions in the next generation population.

The next step is to mutate the newly created offspring. Typical genetic algorithms have a fixed, very small probability of mutation around 1%. Based on this probability, the new chromosome is randomly mutated, typically by flipping bits in the chromosome data structure. The algorithm performs crossover and mutation at the bit level.

These processes ultimately result in the next generation population of chromosomes that is different from the initial generation. Generally the average fitness will have increased by this procedure for the population, since only the best organisms from the first generation are selected for breeding. The entire process is repeated for this second generation: each organism is evaluated, the fitness value for each organism is obtained, pairs are selected for breeding, a third generation population is generated, etc.

This generational process is repeated until a termination condition has been reached. Common terminating conditions are the combinations of these conditions:

- Fixed number of generations reached
- Allocated computation time reached
- An individual is found that satisfies minimum criteria
- The highest ranking individual's fitness is reaching or has reached a plateau such that successive iterations do no longer produce better results

- Manual inspection

4.2.1 Modifications

The fine-grained parallel genetic algorithms place the chromosomes in a fixed point and only neighboring chromosomes are used in selection and reproduction. It results an efficient implementation on the locally connected CNN-UM because the chromosomes are in interaction only at its nearest neighbors.

A slight, but very successful variant of the general process of constructing a new population is to allow some of the better organisms from the current generation to carry over to the next, unaltered. This strategy is known as elitist selection. Our algorithm must be elitist to ensure that the well-performing chromosomes could have more offspring despite its fixed place and local interactions.

Selection is clearly an important genetic operator, but opinion is divided over the importance of crossover versus mutation. Some argue that crossover is the most important, while mutation is only necessary to ensure that potential solutions are not lost. Others argue that crossover in a largely uniform population only serves to propagate innovations originally found by mutation, and in a non-uniform population crossover is nearly always equivalent to a very large mutation. In our implementation this innovation-propagation by the crossover operator is especially important because the chromosomes have fixed places.

Operating on dynamic data sets is difficult, as genomes begin to converge early on towards solutions that may no longer be valid for later data. Several methods have been proposed to remedy this by increasing genetic diversity somehow and preventing early convergence, either by increasing the probability of mutation when the solution quality drops (called triggered hypermutation), or by occasionally introducing entirely new, randomly generated elements into the chromosome pool (called random immigrants). These additions can be implemented easily by modifying a parameter or inserting a new chromosome in a random place.

We will follow a slightly different approach compared to the traditional approach and generate a completely new set of chromosomes using crossover with 100% probability and a small mutation probability. This intermediate population is merged with the old population to form a new: if the fitness of a given chromosome is below the fitness of the intermediate one the old chromosome is replaced with the new. Thus the selection is pair-based between the old and the new chromosomes and is not biased: poorer elements have chance to participate. Actually all low fitness chromosomes have two possibilities to have an offspring.

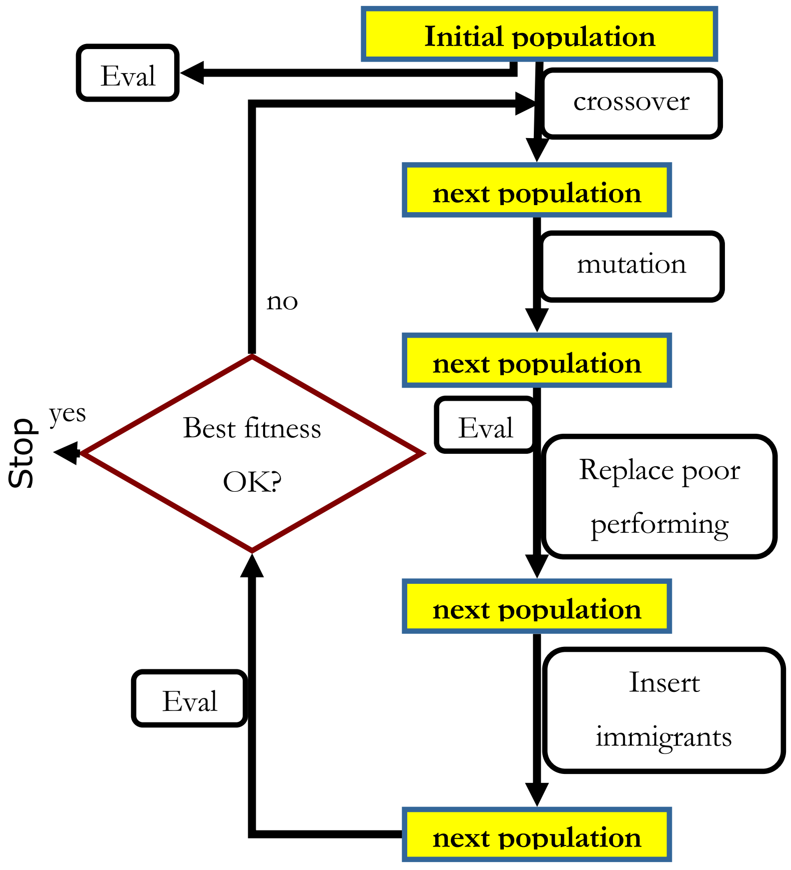


Figure 31 Flowchart of the genetic algorithm

The pseudo-code of our fine-grained genetic algorithm:

- Create initial population
- Evaluate the chromosomes
- Repeat
- Apply crossover operator
- Apply mutation operator with small probability
- Evaluate the individual fitnesses of the chromosomes

- Replace the poor-performing chromosomes
- Rarely insert random immigrants
- Until allocated time reached or other terminating condition is fulfilled

4.3 Implementation details

The mapping of individuals are straightforward, its chromosomes are composed of bits, which are mapped as one row in an image. The fitness calculation should be implemented independently. The crossover and mutation operations can be implemented parallel on the CNN-UM platform. It needs a random binary pattern, which is available using a 2D cell-automata template sequence. This uniform distribution should be transformed to reach the pre-described probability, it can be done using local logic. The black pixels can be used for the crossover through a mask image and for the mutation by the XOR operator. The replacement of the poor-performing chromosomes in the population is implemented using spatial logic.

Constructing a pseudo-random image

The interesting part of the analogic mapping is the different set of operators. The most important one is the creation of pseudo-random binary images. Such an image consists of independently selected black or white pixels.

The CNN-UM can be programmed to be a random binary pattern generator. The method, that was introduced in Section 3.3 can be used here as well. As I stated before, a cellular automata rule introduced by Eq. (3.16) can easily be implemented by four templates and the iterated application of the rule drives the system quickly to a high entropy state.

It was shown in Section 3.3 that in R^i–denoting the logic operation AND between i different random images–the probability of having a black pixel in a given position is a function of the required number of iterations of the CA rule (i.e., the computational cost): $p = (0.5)^i$.

The creation of an initial population is simply to select R as a new population P. Computational cost is 4 local operations.

Crossover operator

The recombination creates a new chromosome C from two parent chromosomes A and B. In our fine-grained parallel implementation, the order of the three chromosomes in the population is ACB in consecutive rows.

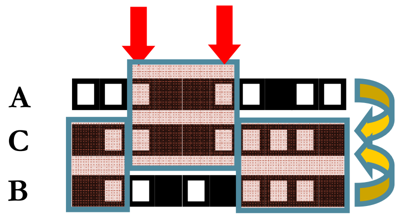


Figure 32 Crossing over. A block is defined by two randomly selected columns (indicated by red arrows): the segment of the new chromosome C within that block is constructed from chromosome A and the segments outside are inherited from chromosome B.

A possible method could be to define a given pixel in C so that to select it from A if R on the same position is black else from B. Although it is easy to implement, the method destroys the 'evolved local-solution blocks' so it results poor-performance according to the genetic algorithm theory. In our implementation, the new chromosome C is constructed by selecting two random columns with uniform probability, and within the block limited by the two columns the pixels from chromosome A is used else from chromosome B (see Figure 32). Below an analogic algorithm is defined that could transform R to one column black with the rest columns white in an effective way. In that case the position of the black column will follow uniform distribution because no spatial information is used to determine it.

Pseudo-random column selection

One possible method selects a random column by searching for a rare pattern (target pattern) in a random pattern image. The probability to have a white column with exactly k black pixels (see Figure 35):

$$Pr = 1- [1 – (N+1-k)\, p^{k}\, (1-p)^{(N-k)}]^{M} \qquad (4.1)$$

where N is the number of columns (size of chromosomes), M is the number of rows (size of the population) and in an R^{i} (AND operators between R random patterns) generated image the p probability has the form: $p = (0.5)^{i}$ (for the notations see pseudo-random pattern generation in

Section 3.3. As a solution, the first column having the target pattern (column with a single black pixel) can be selected (see Figure 35).

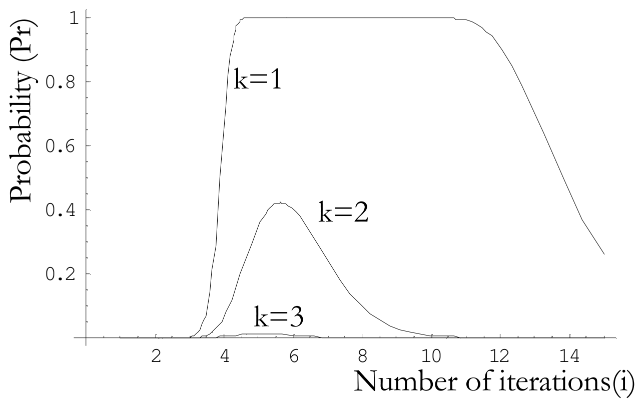


Figure 33 Pr–probability of finding one or more columns with k=1,2 or 3 black pixels– relative to i (number of required iterations for random pattern generation). The size of the CNN array is fixed at N = 100, M = 100. There is an optimum i and k parameter pair Pr value is dramatically decreasing as k is increased.

The computational cost of the random column selection is proportional to the number of the required random images: i. Because N and M are given by the task only the structure of the target pattern (parameter k) remains to tune so that the pattern emerges once in an image (but not more frequently) at a low computational cost. Thus Pr (probability of having the target pattern) should be high while keeping i (required random images) low. Surprisingly k=1 gives the best result, because when k > 1 Pr is small so the complete procedure should be repeated so doubling i. Figure 33 shows Pr relative to i by fixed N and M: as one can see there is an optimum i and the related Pr value is drastically shrinking as k is increased. Table 2 shows that at k=1 and $N=M=2^n$ i can always be chosen to have Pr nearly one. For example: 5 random images are enough for a 128x128 array to have a rare pattern (k=1) with 99.99% probability.

Array size	Required random images	Success probability
n = 2, 3	i = n	Pr > 88.82%
n = 4, 5	i = n-1	Pr > 99.35%
n = 6,7,8,9	i = n-2	Pr > 98.94%
9 < n < 21	i = n-3	Pr > 93.19%
n > 20	i = n-4	Pr > 97.7%

Table 2 Number of random images required as a function of the array size. Size of the array is defined as $N=M=2^n$.

A random column is selected twice for the start and end of the crossover. The same crossing points are used for all new chromosomes in a given iteration. A horizontal masked shadow creates the crossover mask image M that is used as a separator between two shifted population images. The result is an intermediate population C. A schematic representation of these steps is shown in Figure 34.

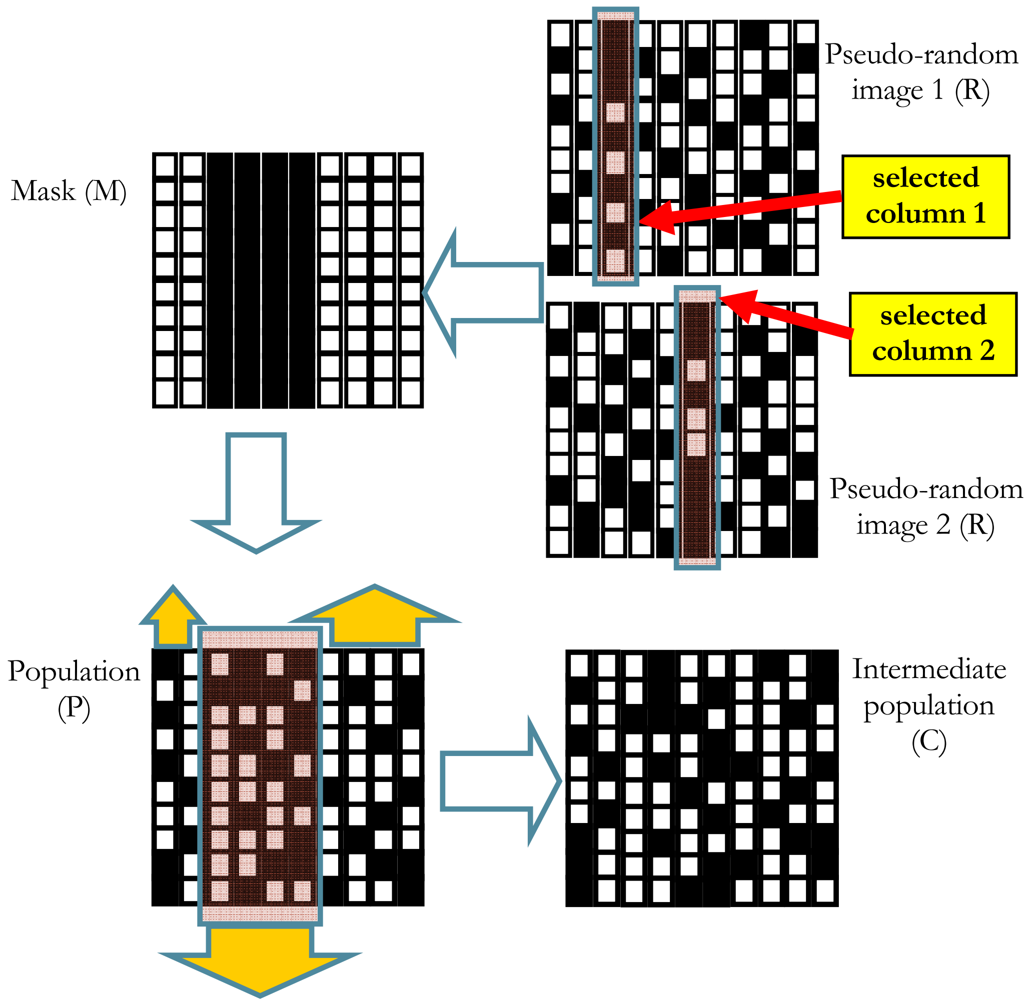

Pseudo-random image 1 (R)
Mask (M)
selected column 1
selected column 2
Pseudo-random image 2 (R)
Population (P)
Intermediate population (C)

Figure 34 Crossing over. A block of columns is selected based on two random column selection within two pseudo-random images ('start of block' and 'end of block'). Within the image representing the current population the selected block is shifted downwards and the remaining columns upwards resulting in a 'crossing over' between the rows representing chromosomes.

```
        M = hor_shadow( *select_col* $R^i$ ) masked with *select_col* $R^i$ .
        C = (M AND (shift_up P)) OR (NOT M AND (shift_down P))
where, *select_col* A:
        B = ccd (match010 A)
        B = shift_right(match001 B) DIF shift_left(match101 B)
        B = hor_shadow B
        if GW(B) than restart with A = $R^i$
        output = ver_shadow(A AND B)
```

Another method for selecting a column is to find the column with the most black pixels (see Figure 35). In this case only one random image is enough. The probability that the column containing the maximum number of black pixels in the random image is unique (there are no more columns having the same number of black pixels) could be about 80%, otherwise the column with the minimal black pixels is selected. The probability that either the column with the maximum black pixels or the minimum black pixels can be selected is shown in Figure 36. An extensive simulation (10000 random images) gave the data for the curve. For example having a 128x128 random array the probability to a unique column is selected about 96% and its average computational cost is 7.8 propagating and 13.6 local operations vs. with the pattern method 7 propagating and 60 local operations.

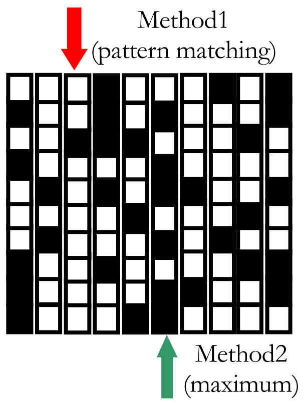


Figure 35 Pseudo-random column selection. Based on a pseudo-random image pattern matching method would select a column that contains a specific rare pattern; e.g. column with exactly one black pixel (third column), the 'maximum' searching method chooses the 6th column with the most black pixels.

This method changes the pseudo-code of the crossover operation only by giving a new selection method:

```
where, *select_col* A: (i=1)
        in cycle A = (v1 A) AND (v2 A)
        A = A DIF (ver_erosion (hor_shadow A))
```

B = match101 (ccd A)

If GW(B) output = A

Output = (NOT A) DIF (ver_erosion (hor_shadow NOT A))

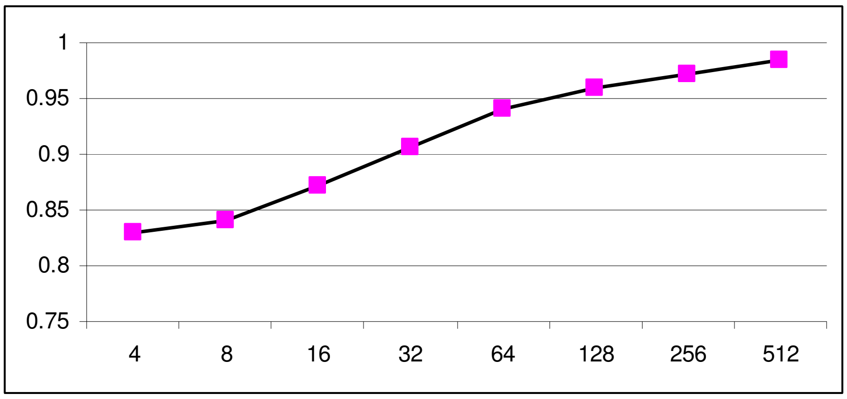


Figure 36 The probability of having a unique maximum or minimum column in a random image R (i =1) relative to the size of the array N = M. The curve is determined by extensive simulation (10000 cases per points).

Mutation operator

The mutation operator flips a certain percent of the bits in the new chromosome. Its implementation is an exclusive or with a random image. The random image is constructed depending on the mutation probability (see Figure 37). Computational cost is 21 local operations by 1%.

$$C = R^j \text{ XOR } C \tag{4.2}$$

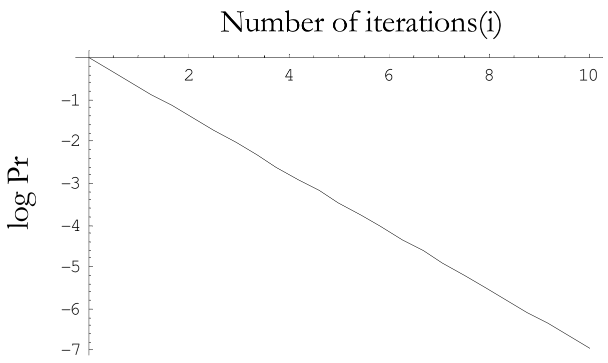


Figure 37 Probability that a certain pixel is black as a function of the number of pseudo-random images used. Pr is display in a logarithmic scale.

Selection

Selection equals replacing poor-performing individuals with new chromosomes. The fitness values are converted to a grayscale image, where each pixel in a given row has the same darkness. The darkness is determined as black is the best fitness value and the white as the least in the current population. The grayscale fitness images Fs are compared to create a mask image indicating the rows of the poor-performing chromosomes. Computational cost is 5 local operations

```
M = threshold F1 with bias map F2
P = (P AND M) OR (C AND NOT M)
```

Immigrants operator

Select a random row with very small probability and replace it with the same row in R. A random row can be selected as a random column in the crossover operation using the pattern matching method with k>1. The replacement is implemented as during selection. The two probabilities can be merged so the immigrants and row selection is determined in one step. Computational cost by 1‰ is 2 propagating and 32 local operations.

```
A = $R^i$
B = ccd( (match011 A) DIF (match111 A) )
B = shift_right(match001 B) DIF shift_left(match101 B)
M = hor_shadow B
P = (R AND M) OR (P AND NOT M)
```

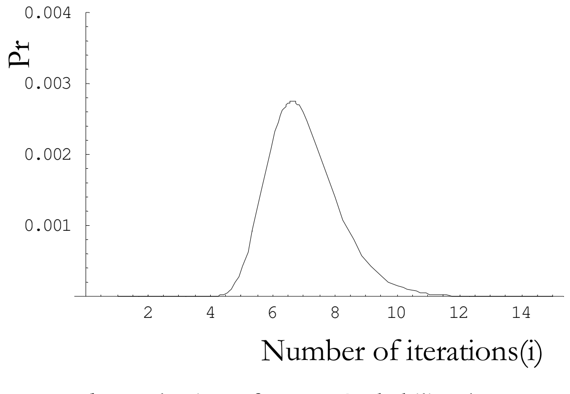


Figure 38 Immigrants operator.: random selection of a row. Probability that a certain pattern (k > 1) emerges in one of the rows is plotted against the number of pseudo-random images used creating R^i. N = 200, M = 1

4.4 Evaluation and summary

We implemented the algorithm by the Candy CNN-UM simulation software package of the Aladdin development system [Szatmári et al., 2002; Zarándy et al., 2003] using the templates shown in Table 3.

Evaluation involved typical test functions with complex fitness landscape (Banana, Camel, Rastrigin, Griewangk, Easom, Ackley) and I also examined the suitability of analogic binary GA in connection with the analysis of multi-dimensional (multi-channel) neural activity data in comparison with classical implementations of the GA (GAOT MATLAB toolbox [Houck et al., 1995]).

Tests showed that the analogic approach performs equally well or even better as its classical counterpart: both methodology find global minima after similar number of iterations. However, implementation of the analogic GA on CNN-UM hardware platforms would mean a much more efficient solution in terms of computation time. Detailed evaluation of the main test can be found in section 6.3.

Template name	A	B	Z	U	X_0
OR	2	1	1	P	Q
AND	2	1	-1	P	Q
DIF	2	-1	-1	P	Q
masked hor_shadow	[1.5 1.8 0]	-1.2	0	P	Q
hor_shadow	[2 3 2]	0	3	0	P
shift_right	0	[1 0 0]	0	P	0
Ccd	[1 2 –1]	0	0	P	P
ver_shadow	$[2\ 3\ 0]^T$	0	3	0	P
ver_erosion	0	$[0\ 1\ 1]^T$	-2	P	0
v1	1	$[0\ 1\ 1]^T$	1	P	P
v2	1	$[1\ –1\ 0]^T$	1	P	P
Match001	1	[0 0 1]	-0.5	P	P
Match011	1	[0 1 1]	-1.5	P	P
Match101	1	[1 0 1]	-1.5	P	P
Match101	1	[1 1 1]	-2.5	P	P
threshold (biased)	2	0	P	0	Q

Table 3 Used templates

5 Collision Warning algorithms

Third thesis

Bio-inspired and analogic collision warning algorithms

III/a. Collision warning algorithm inspired by the locust visual system

I took part in the design of a biologically inspired collision prediction algorithm based on the visual system of the locust. I modified and completed the model to match the requirements of robust operation in real-world car driving situations. We implemented the algorithm on a real-time visual computing system and I adjusted and tested its sensitivity/specificity parameters in real driving scenarios. The system can differentiate between a limited number of object classes according to the need of robust collision warning: the method prevents typical road signs to initiate false collision warnings and recognizes pedestrians and vehicles on an impending collision course, and warn reliably the driver 0.5-1 sec before the actual collision, with negligible amount of false alarms.

III/b. An analogic algorithm for parallel multiple collision prediction based on isotropic diffusion.

I designed a monocular collision warning method that efficiently approximates and can report on the so called time-to-contact variables belonging to different objects. Thus multiple separate warning signals are delivered in a parallel manner.

Beside typical analogic image processing operations the method requires an efficient implementation of 'isotropic diffusion'. Thus, it is an excellent candidate for implementation on CNN-UM computers equipped with locally switchable - mask controlled- resistive grid feature.

5.1 Introduction

Predicting dangerous or advantageous situations has similar importance for animals and for machines. One of these situations is when the motion of an object could end up with a useful (for a predator) or disadvantageous (for a prey) collision with the observer/sensor. A growing body of evidence makes neurobiologists suppose that for some animals (and humans) low-level monocular visual information alone is enough for a remarkable estimation of the time left till an impending collision [Schiff & Detwiler, 1979; Rind & Bramwell, 1996; Rind & Simmons, 1999]. Fast calculation even in an unfamiliar environment may have a life-saving impact. According to recent theories, calculation can be based upon only two optical variables of looming objects, ensuring the relative context-insensitivity of the process [Sun & Frost, 1998; Laurent & Gabbiani,1998, Gál & Roska, 2000]. Precise information about the size (diameter) of the two-dimensional projection of the object and the rate of its expansion are enough to predict the so called time-to-collision (TTC), provided that the motion is at a constant speed.

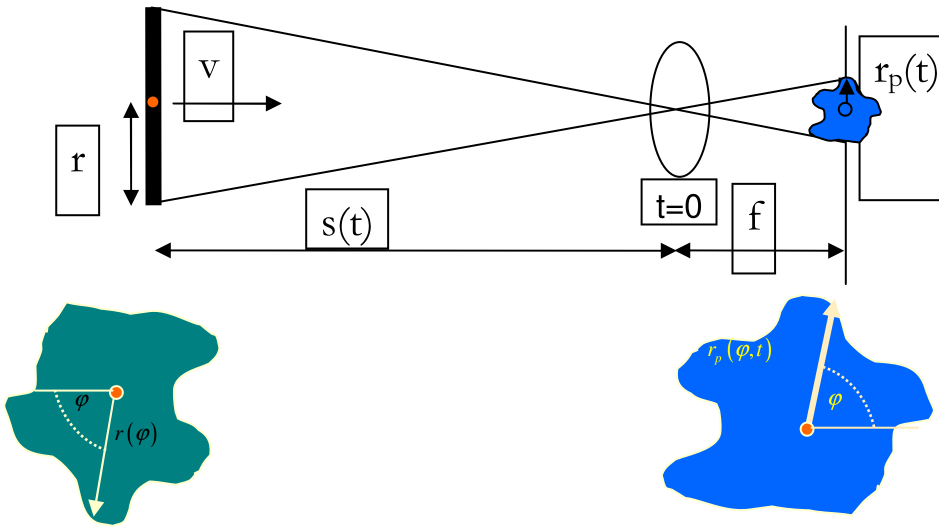


Figure 39 Image of an object approaching a sensor's lens on a direct collison coarse. r,r_p radius of the object and its corresponding image. s(t) is the distance between the object and the lens t secs before collision, approach speed is a constant v.

There is a law well known in optical geometry:

$$\frac{s(t)}{f} = \frac{r(\varphi)}{r_p(\varphi,t)}, \tag{5.1}$$

and

$$s(t) = -vt \tag{5.2}$$

Plugging (5.2) into (5.1) we get:

$$r_p(\varphi,t) = \frac{fr(\varphi)}{-vt} = \frac{fr(\varphi)}{-v}t^{-1} = k_r(\varphi)t^{-1} \tag{5.3}$$

The derivative of the function above is the speed of the projected image's edges at the image plane:

$$\dot{r}_p(\varphi,t)(=v_{edge}) = -k_r(\varphi)t^{-2} \tag{5.4}$$

Thus we get the so called time-to-contact (τ):

$$\tau = \frac{r_p(\varphi,t)}{\dot{r}_p(\varphi,t)} = -t \tag{5.5}$$

so the ratio between the diameter and the rate of its expansion is equal to the time left to a direct collision.

Although not directly, but a number of monocular collision warning systems make use of the principles of the time-to-contact estimation [Shi, 1996]. Here we propose two such algorithms. One has a biologically relevant origin: it is based on the locust's visual system, and its robustness is tested in real-world car driving situations. The background of the other method is purely computational approach, and has the advantage of monitoring more than one objects

5.2 A collision warning algorithm inspired by the locust visual system

5.2.1 Introduction

The locust LGMD neuron is evolutionary honed to detect potential collisions with predators and provide a collision avoidance reaction. This thesis presents computer models of the LGMD neuron and tests their suitability for use in automotive situations as a means of detecting a collision. Models respond well to colliding objects and are mainly successful in eliminating non colliding objects. Modifications, including low-level object classification and the incorporation of the fly EMD-like neuron [Harris et al. 1999; Harrison, 2000] for detection of translating objects are presented in the application chapter.

Locusts are known to detect looming objects via a large neuron in the brain called the Lobula Giant Movement Detector (LGMD) [Rind & Bramwell, 1996]. This neuron is tightly tuned to only respond to objects on a direct collision course and also appears to be tuned to only avoid objects of a certain size and approach velocity, such as avian predators. Any object present in an image will subtend an angle on the eye. Looming objects are characterized by the way the diameter (or angle subtended) changes with time; when far away the increase of angle is low but as the object approaches the increase in the angle is higher with a maximum during the final stages of the approach (Figure 40). The increase in the angle subtended on the eye looks similar to an exponential function, but is mathematically quite distinct(5.3). The LGMD receives excitation from a large number of afferents, which are arranged retinotopically, with neighbouring afferents associated with those of neighbouring photoreceptors in the eye. Excitation is passed down these afferents in the form of an action potential (spike) or, more commonly, a graded potential (voltage). The number of excited afferents depends on the angle of the object subtended on the eye, thus when a looming object approaches the excitation to the LGMD increase rapidly as the object comes closer. Non-colliding objects do not show the same increase in excitation (Figure 40) and as such are unlikely to trigger avoidance reactions.

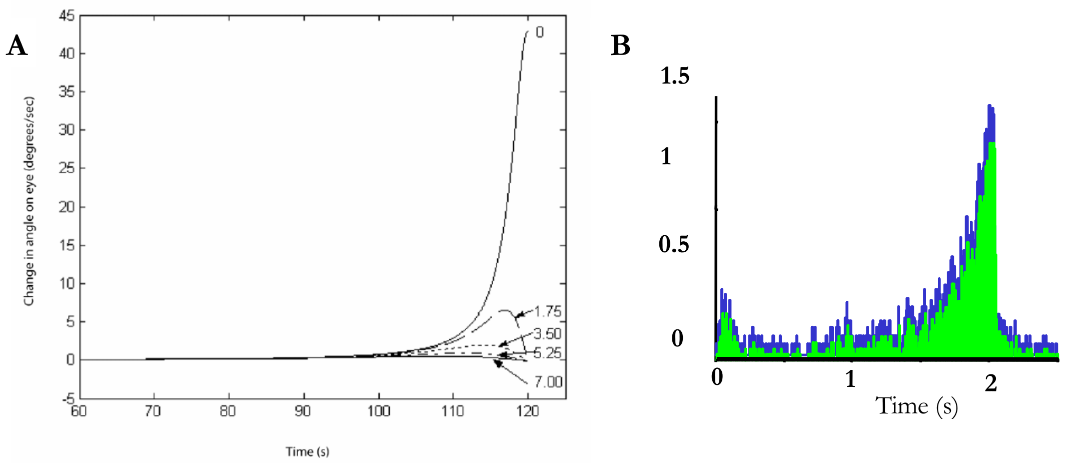


Figure 40 The increase in the diameter on the eye by an approaching object on direct and non direct collision courses. Data is for a black sphere moving towards the eye at 25 mm/s on a direct collision course or at 0, 1.75, 3.50, 5.25 or 7.00 degrees away from a direct collision course. B Single cell recordings from the LGMD neuron of the locust during presentation of an approaching object

The underlying principle of the LGMD is relatively simple; indeed it would be possible to calculate the increase in size of an object to determine if it was on a direct collision course. Such an approach, however, requires the detection and segmentation of the approaching object from an often complex background of other moving objects. There are no robust solutions for such tasks but I am going to present an algorithm in the following section which uses that kind of information – this is the base of my IIa thesis. The LGMD provides several methods to help eliminate responses to non approaching objects which are both cognitively and computationally simple [reviewed by Rind et al., 2003]. The best studied of these methods are lateral inhibition between afferents pre-synaptic to the LGMD and feed forward inhibition, post-synaptic to the LGMD.

We review what is known of the biology of the locust LGMD and details about further modifications that have been made to create models inspired by the locust LGMD.

5.2.2 An overview of the Rind Model

The LGMD and associated Descending Contralateral Movement Detector (DCMD) neurons in the locust and cockroach have been the subject of much research over the past 40 years [Rind and Simmons, 1999; Rind et al., 2003]. During this time extrapolations of the input architecture were made and basic computer models of this architecture were tested: e.g. [Edwards, 1982; Pinter, 1983; 1984]. Simmons and Rind [1992] discovered that the neuron showed a distinct preference for looming objects, these being objects moving towards the eye of the locust, with a constant velocity. Further electrophysiological recordings suggested the response of the neuron was tightly tuned to objects on a collision trajectory and did not respond in the same manner to objects on a near miss trajectory [Judge and Rind, 1997]. A computer model of the locust LGMD was produced by Rind and Bramwell [1996] which produced similar results to looming stimuli as found in the electrophysiological data from the locust. The complexity of the model was increased and was coupled with a mobile robot to obtain and process data in real time [Blanchard et al., 2000]. This experiment showed collision avoidance to real stimuli, albeit in a simplified environment. The proposed structure of the LGMD and its inputs as proposed by Rind et al. are outlined in Figure 41.

Light is captured by the photoreceptors and converted to electrical charge. This is passed to a range of retinotopically arranged neurons. The 'E' units pass excitation to the 'S' units directly. They also pass excitation to the 'I' units. The excitation is converted to inhibition at the synapse between the 'E' and 'I' units and inhibition is passed to the nearest neighbouring 'S' unit with a delay of one timestep and the next nearest neighbouring 'S' unit with a delay of two timesteps. At each 'S' unit the excitation is subtracted from inhibition and any remaining excitation is summed on the LGMD. In addition the total excitation of the photoreceptors is summed by the 'F' unit and if a threshold value is reached the output of the LGMD is inhibited. The LGMD output is either expressed as a graded potential (voltage) or as spikes, and collision avoidance behaviors are determined by this response. The interaction of the spreading excitation over the inputs of the LGMD due to a looming object, and the inhibition of the 'I' units (herein lateral inhibition), creates a critical race [Simmons & Rind, 1992]. If the object is on a direct collision course, the spread of excitation over time is less than the spread of lateral inhibition until the object is close to the eye, then excitation wins the race and passes to the LGMD causing an avoidance reaction.

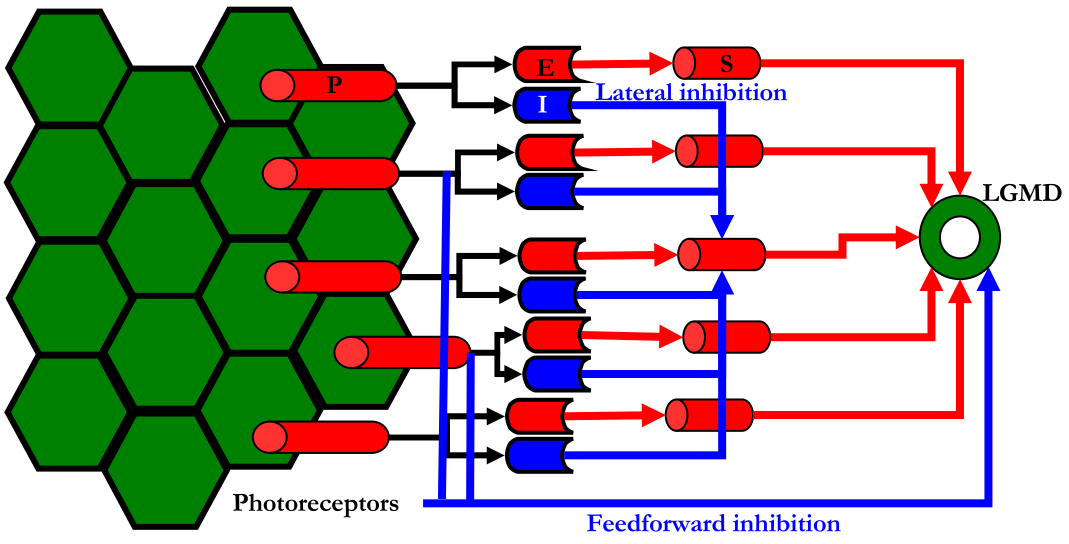


Figure 41 The LGMD neuron and input architecture as described by Rind et al. Light is captured in the photoreceptors (Layer 1) and is passed to excitatory 'E' units and inhibitory 'I' units (Layer 2). Interactions take place at the 'S' units (Layer 3) before the 'S' units are summed by the LGMD (Layer 4). See text for further details including the role of feed forward inhibition.

Objects not on a direct collision course are less likely to have the spread of excitation exceeding that of lateral inhibition and provide less excitation to the LGMD. Lateral inhibition can also help to suppress the background of an image, as background movements generally do not generate excitation which exceeds the range of lateral inhibition. The inhibition by the 'F' unit (feed-forward inhibition) responds to large scale changes in the image, for example changes in overall contrast of the image as might be encountered if flying from sunlight into shade or vice versa. It is thought to be triggered during the final stages of an approach by a looming object, however, its response is delayed so LGMD activity continues until after the point of collision. When an object recedes from the eye, on an otherwise identical trajectory, the locust will only show a short burst of activity from the LGMD, this is thought to be due to the delay of the feed forward inhibition. This process provides a useful mechanism to distinguish between approaching and receding objects. Work on collision sensors using the artificial LGMD network has proved successful [Rind et al., 2003], however, no direct attempt has been made to study the network in the types of environments encountered in automotive applications.

Bellow I describe modified biological models of the neuron layers which were the starting point of all other computational models. It was shown that the analogic implementation of the neuron models is feasible at different complexity levels [Gál et al., 2004].

ON-OFF neurons

OFF-units were the first stage in the original Rind et al. model. We substituted this stage by ON-OFF units. The membrane potential of an ON-OFF-cell at position (i, j) at time t is described by the state variable p_{ij} (t). The respective dynamics is governed by:

$$\dot{p}_{ij}(t) = g_{leak}\left(V_{rest} - p_{ij}\right) + L_{ij}(t)\left(E_{ex} - p_{ij}\right) + L_{ij}(t-1)\left(E_{in} - p_{ij}\right) \tag{5.6}$$

g_{leak} is the leakage conductance (or decay constant) which describes the total passive ion flow through the cell membrane. V_{rest} is the resting potential (or leakage reversal potential) which the cell will adopt if it does not receive any input. E_{ex} and E_{in} are the excitatory and inhibitory synaptic batteries, respectively, which confine a cell's dynamic range between E_{in} and E_{ex} (as long as V_{rest} is in the range of E_{in} ,E_{ex}). The On-Off units replace the single-contrast-polarity P-units in the original Rind et al. model. Input into the On-Off units is provided by luminance values L(t) and L(t-1).

Lateral Inhibition by Explicit Delay

This mechanism is according to Rind et al. [Blanchard et al., 2000] :

$$\dot{w}_{ij}(t) = g_{leak}\left(V_{rest} - w_{ij}\right) + \tilde{p}_{ij}\left(E_{ex} - w_{ij}\right) \tag{5.7}$$

Originally, P-unit activation excites inhibitory I-units. Here ON-OFF unit activity p_{ij} is fed into inhibitory interneurons characterized by membrane potential w_{ij}. Lateral inhibition is implemented by convolving the response of an inhibitory neuron w_{ij} with a nearest-neighborhood kernel K1 and a next-nearest-neighborhood kernel K2:

$$u_{ij}(t) = \tilde{w}_{ij}(t-1) \otimes K_1 + \tilde{w}_{ij}(t-2) \otimes K_2 \tag{5.8}$$

where the convolution K1 and K2 are Gaussian-weighted kernels.

Excitatory neurons

Excitatory neurons replace the S-units of the Rind et al. model. Excitatory neurons receive direct excitatory input from ON-OFF-cells, whereas in the Rind et al. model E-units are interposed between P-units and S-units. Excitatory neurons with membrane potential v_{ij} are defined as

$$\dot{v}_{ij}(t) = g_{leak}(V_{rest} - v_{ij}) + \beta_{ex}\tilde{p}_{ij}(E_{ex} - v_{ij}) + \beta_{in}\tilde{u}_{ij}(t)(E_{in} - v_{ij}) \tag{5.9}$$

where β_{ex} and β_{in} are gain factors (or synaptic weights).

The LGMD neuron

The LGMD neuron integrates the activity of all excitatory units according to:

$$\varepsilon = \sum_{i,j}^{n} \tilde{v}_{ij} \tag{5.10}$$

In contrast to the Rind et al. model, the present model of the LGMD neuron does not receive feed forward inhibition. The LGMD dynamics obeys:

$$\dot{l}(t) = g_{leak}(V_{rest} - l) + \gamma_{ex}\varepsilon(t)(E_{ex} - l) \tag{5.11}$$

where γ_{ex} is a gain factor and l is the membrane potential.

5.2.3 Biology inspired models

There are a lot of derived models based on the Rind model and the one described above. After exhaustive evaluation two models were selected to implement for test in real traffic situations. There are a lot of differences, modifications comparing to the original structure, like none of the input afferents produce spikes; all are simple graded potential neurons, more suitable for implementation in analogue VLSI chips. Also the feed forward inhibition has been removed from all of the models, and replaced by more effective solutions. Finally the 'E' units in the Rind et al. model have been replaced by On-Off units. These investigate changes in the contrast over time. In the following models the On Off response is captured within the framework of the LGMD.

Description of the implemented BSC Model

The basic building blocks of this model are based on simple visual receptive field interactions and neuron models like those implemented on CNN-UM [Gál et al., 2004]. It is aimed to increase the complexity of the LGMD network by adding a new summing layer immediately pre-synaptic to the LGMD. It also aimed to provide a detailed decision making process to decide when a collision was about to occur based on the neurobiology of escape behavior of the locust. The architecture of the model is shown in Figure 42.

As in the previous models the photoreceptors obtain a luminance value for a given point in space at a given time. The On Off units simply subtract the photoreceptor value at time t-1 from that at time t. As above, if there is no change in luminance the value of the On-Off unit is zero, although there is no persistence of membrane potential in this model. Excitation is passed though the layers of the model in both light and dark pathways until the separate pathways combine in the block sum cells. Inhibition is triggered by the On-Off pathway spreading from the position of the excited On-Off unit to all the effected retinotopical nearest neighbors inside one timestep.

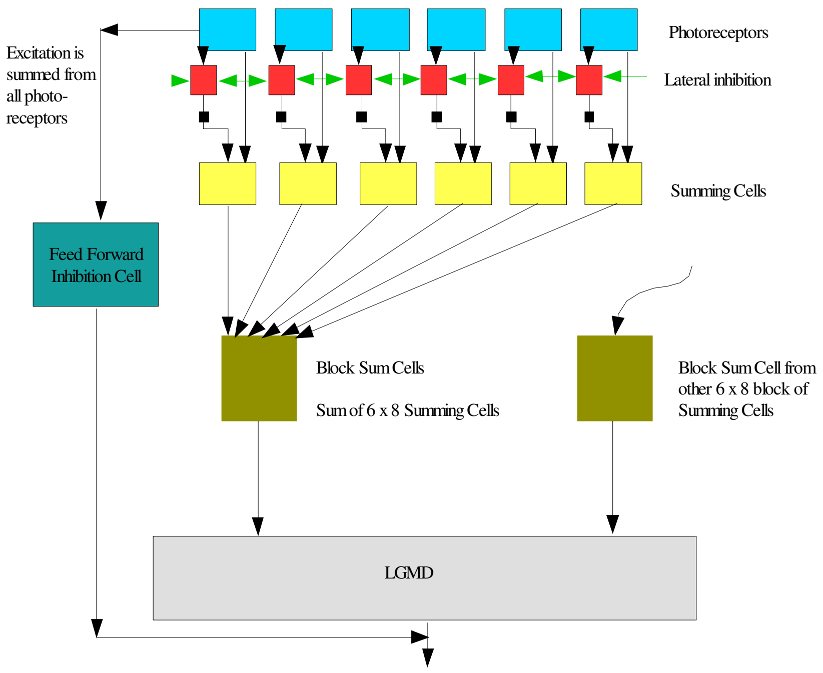

Photoreceptors
Excitation is summed from all photo-receptors
Lateral inhibition
Summing Cells
Feed Forward Inhibition Cell
Block Sum Cells
Sum of 6 x 8 Summing Cells
Block Sum Cell from other 6 x 8 block of Summing Cells
LGMD

Figure 42 The block summing cells model

The excitation interacted with inhibition on the 'S' units. The 'S' units were then summed in blocks of 10 x 10 onto a retinotopically arranged grid of 15 x 10 block sum cells. The result was full wave rectified before the block sum cells were summed onto the LGMD. The purpose of the block sum cells is to provide an antagonistic effect of light on and light off responses within a small spatial area, yet allow both pathways to contribute to the LGMD response is separated by a larger distance. This phenomenon occurs in the locusts' neural pathway [Simmons and Rind, 1992] and has clear advantages to automotive applications. The antagonistic effect over small spatial scales allows excitation caused by minor vibrations or turns of the car to be reduced. This can be explained by considering a vertical black line on a white background. If the car slightly turns or the black line appears to move to the left or right then there will be excitation caused by a light off response where the black line now appears and excitation caused by a light on response where the line previously was. These responses are likely to be present within the same block sum cell and the combination of positive light on and negative light off responses will produce only a little, if any excitation. If these separate pathways are maintained over the whole of the field of view then problems can occur. Approaching objects can appear lighter than some sections of the background (e.g. dark road surfaces) but darker than other sections of the background (e.g. light sky) creating little net excitation once the pathways are combined. Block sum cells allow excitation from both the light on and light off pathways to contribute to the LGMD activity and detect the moving object. In this model, as in the locust, the LGMD produces spikes if the membrane potential exceeds a threshold value. If this threshold is exceeded the membrane potential also falls to zero. The threshold is determined in 2 ways. Firstly no spikes are produced if the membrane potential is lower than a fixed threshold. If background scenes are complicated, however, more excitation passed through the processing of the On-Off, inhibition and block sum cell units. This could cause spikes to be generated in response to non colliding objects. In this case the threshold is raised by sampling the LGMD membrane potential at a selection of previous time steps, the mean value of these samples is raised by a small and is set as the LGMD threshold. If four or more spikes are produced within five timesteps then a collision is considered to be detected.

Formal description

Movement contrast

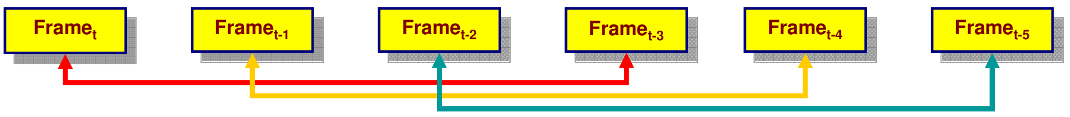
Movement contrast C(i,j,t) is computed by subtracting at each position (i,j),a previous luminance value L(i,j,t-f) from the current one L(i,j,t)

$$c(i,j,t) = \left| L(i,j,t) - L(i,j,t-f) \right| \tag{5.12}$$

where 0 < i < 100 is the row index, 0 < j < 150 is the column index, t are discrete time steps according to the frame rate(e.g. t=1,2,3 … denote 40, 80, 120 … msec at 25frames/second), and f is an adaptable parameter denoting the time interval between the compared luminance values:

$$f = \begin{cases} (60 - v(t))/10 & for \quad v(t) < 50 \\ 1 \quad for \quad v(t) \geq 50 \end{cases} \tag{5.13}$$

where v is the speed of the vehicle in km/h.



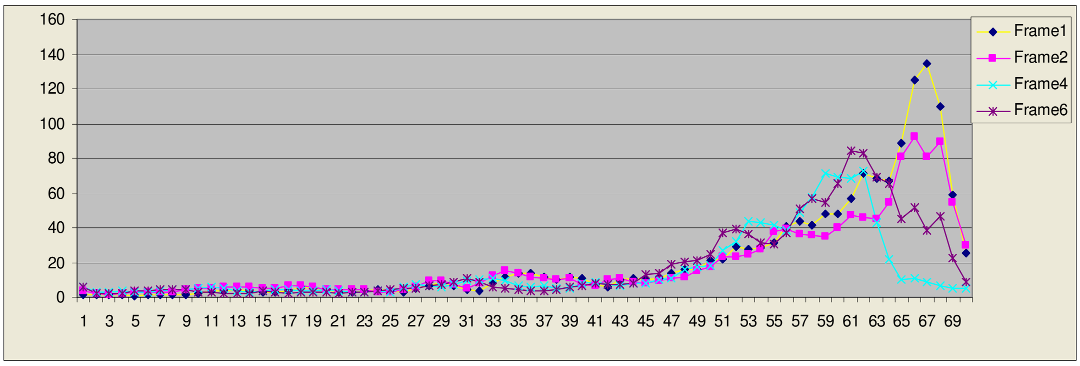


Figure 43 Impact of changing of the interframe interval on the time of collision warning. The time resolution (maximal framerate) itself is preserved: increasing frame intervals make earlier collisoin warning possible: Though their amplitudes are decreasing the threat level curves are shifted backwards

The explanation why we need this adaptivity parameter is that at lower speeds or relatively high frame rates the change between two subsequent frames are too small, so the growing image of approaching objects can not activate proper amount of On/Off cells for triggering alarm signals only when they are so close that the accident is inevitable (see Section "Optical constraints on motion detectability" later in this Chapter). For this reason it would be sensible to analyse the difference between the actual image and the nth preceding frame (see Figure 43): e.g. $frame_t$ is compared to $frame_{t-3}$ $frame_{t-1}$ to $frame_{t-4}$ etc. This is not "decimation" since the frame rate (and the available amount of information) is preserved: every single frame is analysed. The method requires a frame buffer which can store so many frames as the comparison requires adapted to the speed of the car. At lower speeds the buffer and the distance between the frames to compare should be increased.

After full wave rectification, contrast values *C(i,j,t)* are in the range:

$$0 \le L(i,j) \le 255 \tag{5.14}$$

Inhibition.

Inhibitory activity *I(i,j,t)* is given by the change in contrast generated by spatial filtering, and summing full wave rectified contrasts,

$$I(i,j,t) = \left|C(i,j,t-1) - C(i,j,t-2)\right| \otimes K \cdot w(t) \tag{5.15}$$

where *K(p,q)* is a 3x3 convolution kernel with values 1 at nearest-neighbor positions. *w(t)* are empirically defined, state-dependent weight defined later in this section. *w* is initialized with $w(t_0) = 0.75$. Inhibition *I(i,j,t)* is rounded before interacting with summing cells.

Summing cells.

The output of summing cells *S(i,j,t)* represents the outcome of the race excitation vs. inhibition:

$$S(i,j,t) = C(i,j,t) - I(i,j,t) \tag{5.16}$$

Block summing cells.

Let *B(p,q)* designate a set of indices selecting a block of 10 x 10 summing cells from mxn block (in this implementation m=10 rows and n=15 columns), where the set of rows are defined by:

$$\{(p-1)10+1,\ (p-1)10+2, ...,\ 10p-1, 10p\} \tag{5.17}$$

and the set of columns:

$$\{(q-1)10+1,\ (q-1)10+2, ..., 10q-1, 10q\} \tag{5.18}$$

Block summing cells(BSC) integrate summing cell activities within their corresponding block:

$$B(p,q,t) = \sum_{i=10(p-1)+1}^{i=10p} \sum_{j=10(q-1)+1}^{j=10q} S(i,j,t) \tag{5.19}$$

LGMD response.

Output of the summing cells is integrated to compute the net excitation

$$e(t) = scale_1 \sum_{\substack{p=1,\\ q=1}}^{\substack{p=m,\\ q=n}} B(p,q,t) \tag{5.20}$$

where scale$_1$ scales e(t) down to the range [0,255]

Raw LGMD state.

The state variable *l(t)* corresponding to the neuronal membrane potential of the *lobula giant movement detector* (LGMD):

$$l(t) = LGMDcoeff_1 \cdot e(t) + LGMDcoeff_2 \cdot l(t-1) + LGMDcoeff_3 \cdot l(t-2) \tag{5.21}$$

LGMDcoeff s are adjustable parameters, initially set to:

$$LGMDcoeff_1 = 0.71$$
$$LGMDcoeff_2 = 0.19$$
$$LGMDcoeff_3 = 0.1$$

The coefficients have always normalized values:

$$LGMDcoeff_1 + LGMDcoeff_2 + LGMDcoeff_3 = 1$$

When a spike is detected, then *l* is subjected to reset:

$$l(t) = 0; l(t-1) = 0; l(t-2) = 0\,.$$

Threshold computation.

Threshold computation involves a second LGMD variable $l_2(t)$:

$$l_2(t) = LGMDcoeff_1 \cdot e(t) + LGMDcoeff_2 \cdot l_2(t-1) + LGMDcoeff_3 \cdot l_2(t-2) \tag{5.22}$$

No reset takes place with $l_2(t)$, as opposed to $l(t)$. From l_2, a raw threshold *T(t)* is computed according to:

$$T(t) = Threshcoeff_1 \cdot l_2(t-5) + Threshcoeff_2 \cdot l_2(t-10) + Threshcoeff_3 \cdot l_2(t-14)$$

Threshcoeffs are adjustable parameters, initially set to:

$$Threshcoeff_1 = 0.1$$
$$Threshcoeff_2 = 0.8$$
$$Threshcoeff_3 = 0.1$$

The coefficients have always normalized values:

$$Threshcoeff_1 + Threshcoeff_2 + Threshcoeff_3 = 1$$

The actual threshold *W(t)* is then computed from $l_2(t)$:

$$W(t) = \max\{T(t), fixedthresh\} + threshoverload \tag{5.23}$$

where *threshoverload* is also an adjustable bias shifting the actual threshold with a fixed value (initial value: 10), and *fixedthresh* is a fixed minimum threshold value.

Spiking generator.

Let *L(t)* ∈ [0,1] be the output of the LGMD.

$$L(t) = \begin{cases} 1 & for \quad l(t) > W(t) \\ \\ 0 & for \quad l(t) \le W(t) \end{cases} \tag{5.24}$$

After firing a spike, the end of the LGMD response history is reset to zero:

$$l(t) = 0; l(t-1) = 0; l(t-2) = 0 \tag{5.25}$$

Otherwise, no reset takes place and *l(t)* keeps its original value.

Level of threat.

The model runs through a set of discrete states. These states are defined by the specific values of L*(t)*. Let *n(t)* be the number of spikes fired in an observation interval:

$$n(t) = L(t) + L(t-1) + L(t-2) + L(t-3) + L(t-4) \tag{5.26}$$

$n(t) \ge 4$ indicates a high probability of an impending collision.

w(t), the actual inhibitory coefficient is selected by n(t) from a set of adjustable parameters according to the table bellow:

Level of threat = n(t)	Name of the adjustable parameter	Default value
0	Inh_coeff1	10/18
1	Inh_coeff2	0
2	Inh_coeff1	10/18
3	Inh_coeff3	19/18
4	Inh_coeff4	7/18
5	Inh_coeff5	19/18

Table 4 Inhibitory weights *w(t)* as a function of spike count

Formal description of the Simplified Model (SM)

The simplified model is defined by a couple of modifications of the optimal model.

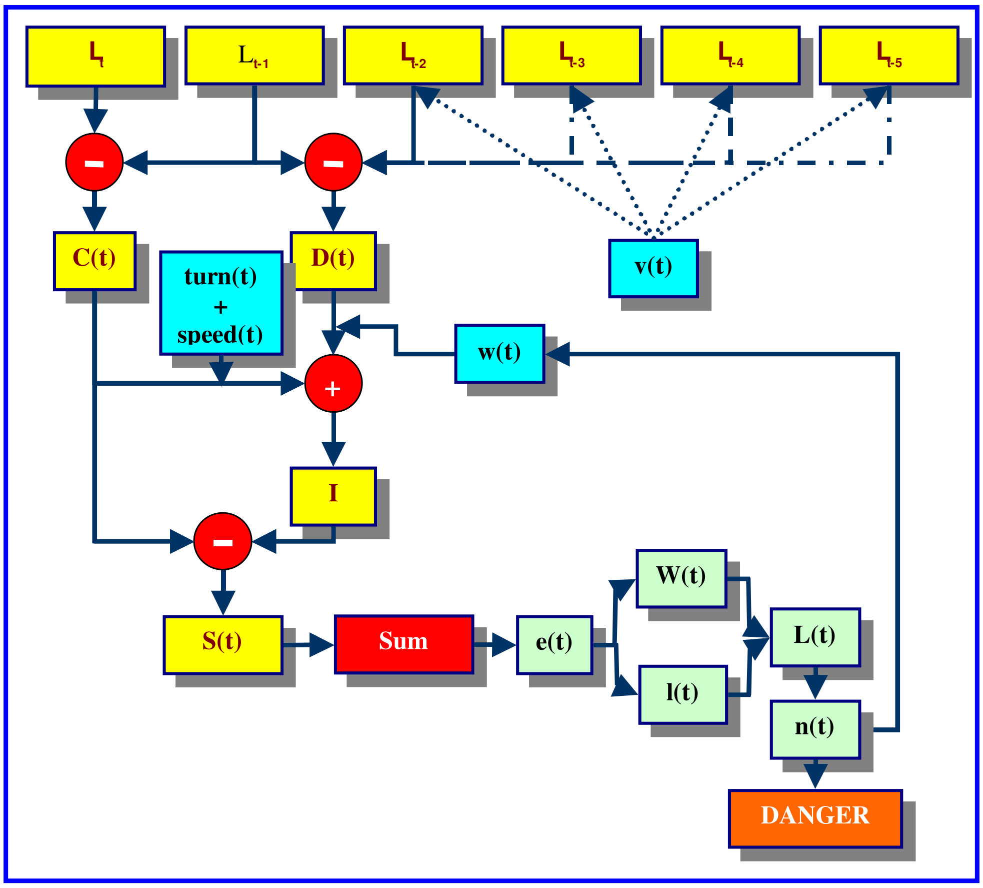


Figure 44. Simplified diagram of the implemented algorithm. For the symbols see the model description above.

Inhibition.

First, inhibition is replaced by

$$I(i,j,t) = C(i,j,t-1)w(t) + C(i,j,t)\cdot(turn(t) + speed(t)) \tag{5.27}$$

Thus, "lateral inhibition" is substituted by "inhibition" with no spread of activity between neighbours. As a second modification the inhibitory weights w(t) were slightly scaled up (factor 1.1).

$turn(t)$ is a coefficient defined by the actual wheel angle(angle(t) in degree):

$$turn(t) = \begin{cases} 1 \quad for \quad |angle(t)| > highturn \\ \\ |angle(t)| / highturn \quad for \quad |angle(t)| \le highturn \end{cases} \tag{5.28}$$

The position of the wheel (exceeding highturn threshold) can suppress the activity of the summing cells through the term $C(i,j,t)\cdot turn(t)$ avoiding false alarms induced by the huge optic flow induced by turning. Default value for highturn: 10 degree.

$speed(t)$ is a coefficient defined by the actual speed (v(t) in km/h):

$$speed(t) = \begin{cases} 0 \quad for \quad |v(t)| > lowspeed \\ \\ 1 \quad for \quad |v(t)| \le lowspeed \end{cases} \tag{5.29}$$

Bellow lowspeed the alarm system is suppressed: when the vehicle is not moving, or moving very slowly, other objects can come extremely close and cross the field of view safely.

Summing cells

The output of summing cells S(i,j,t) represents the outcome of the race between excitation vs. inhibition are rectified:

$$S(i,j,t) = \max\{C(i,j,t) - I(i,j,t), 0\} \tag{5.30}$$

Block summing cells

The block summing cells layer is comletely removed from this model.

LGMD response

Output of the summing cells is integrated to compute the net excitation

$$e(t) = scale_2 \sum_{\substack{i=1,\\ j=1}}^{\substack{i=100,\\ j=150}} S(i,j,t) \tag{5.31}$$

where $scale_2$ scales e(t) down to the range [0,255]

Threshold computation is separated from the state of LGMD cells l_1(t):

$$l_2(t) = Threshcoeff_1 \cdot e(t) + Threshcoeff_2 \cdot l_2(t-t_1) + Threshcoeff_3 \cdot l_2(t-t_2) \tag{5.32}$$

$$W(t) = \max\{l_2(t), fixedthresh\} + threshoverload \tag{5.33}$$

When a spike is detected, then *l* is subjected to repolarization:

$$l(t) = l(t) \cdot repolarize\_coeff; l(t-1) = l(t-1) \cdot repolarize\_coeff; \tag{5.34}$$

Inhibitory coefficients w(t), the actual inhibitory coefficient is selected by n(t) from a set of adjustable parameters according to the table bellow:

Level of threat = n(t)	Name of the adjustable parameter	Default value
0	Inh_coeff1	10/16
1	Inh_coeff2	0
2	Inh_coeff1	10/16
3	Inh_coeff3	19/16
4	Inh_coeff4	7/16
5	Inh_coeff5	19/16

Table 5 Inhibitory weights *w(t)* as a function of spike count

In Figure 45 the time course of the activation of different parts of the model: the observer is approaching a car with sudden break. As a result of the competition between LGMD and LGMD threshold –levels are crossing several times- a number of spikes are generated consecutively.

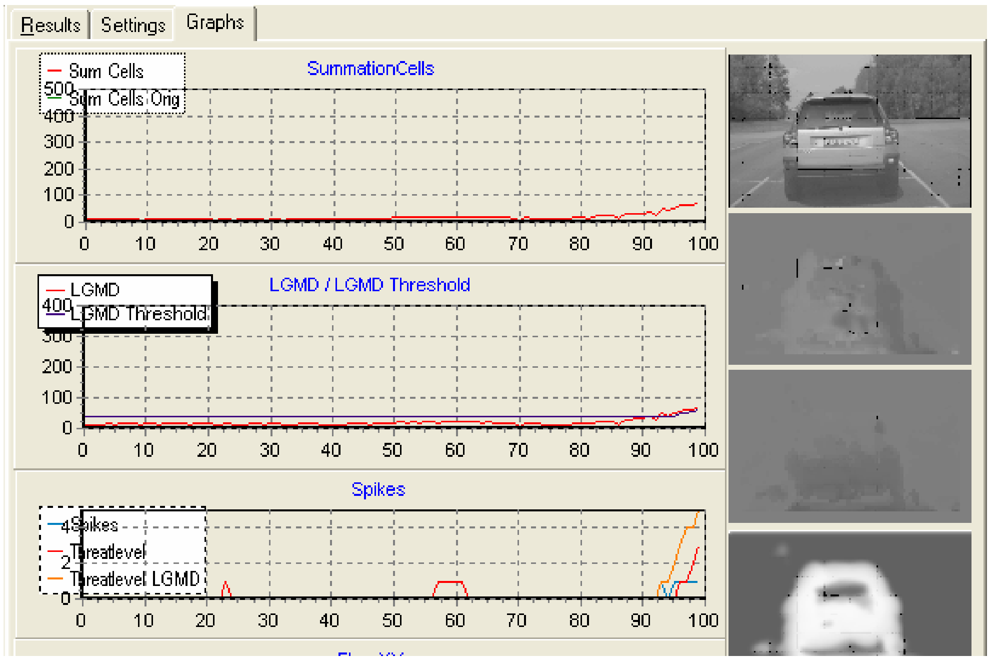


Figure 45 Time course of the activation of different layers of the model: the observer is approaching a car with sudden break.

Optical constraints on motion detectability

Provided that we have focal plane sensor and the width of the array is 7mm experience shows that we need a camera lens with a focal length of 25mm or shorter to have a proper view angle for monitoring the traffic in front of us.

The focal length and the sensor pitch (=70μm) defines the optical resolution of the system. The optical resolution imposes constraints upon the extent of change between subsequent frames to be above the detection threshold.

We've made a series of measurements and calculations to determine the required minimal relative speed of approaching vehicles causing detectable change between subsequent images: This parameter is a function of distance and frame rate. The chart below (Figure 46) shows the minimal speed of a car (width=2m) to be detected at different distances and frame rates. E.g. at a typical 25 frames/sec rate a car 15m away from the sensor has to approach the observer with 60km/h at least to cause detectable change between two subsequent frames.

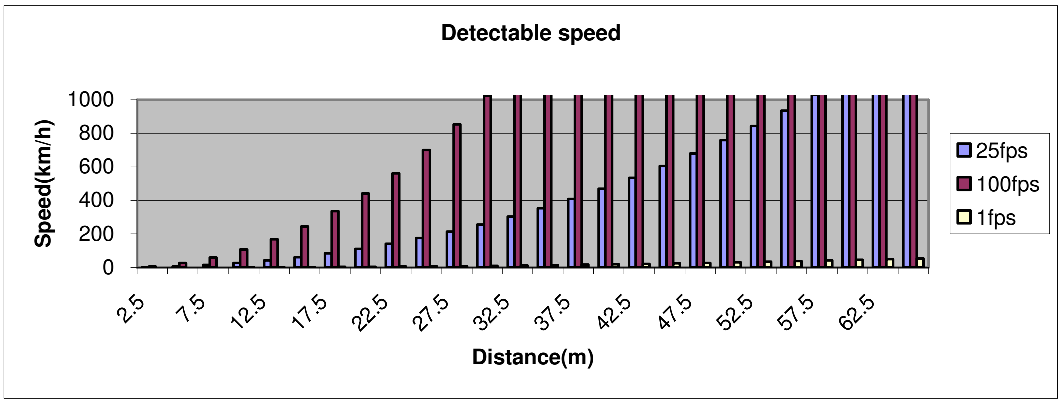


Figure 46 Required minimal speed of approaching vehicles that is necessary for a robust detection at different distances and frame rates.

This can be a serious issue making the implementation of adaptable interframe interval really important (see Eq. (5.13)).

5.2.4 Discussion

The models clearly show that the simple, biologically inspired design of the locust LGMD is useful in the detection of collisions in automotive situations. The main benefits of the LGMD are that there is no need to identify objects, their approach speeds or their angles of trajectory, all of which can be computationally expensive. The only data required is the luminance values of the photoreceptors over a period of time. The downside of the biologically inspired approach is that the locust LGMD has evolved to respond with a collision avoidance reaction in a certain situation. This is thought to be the approach of a small (~ 7 cm diameter), fast moving (~ 5 m/s) predator whilst in flight. In addition this reaction occurs only a short time before the collision would have occurred (~ 100ms). The differences in the colliding objects speed and size, slower refresh rates of images and lower temporal frequencies of the LGMD and the need to detect collisions sooner in automotive situations, due to the comparative sluggish reactions of a car, present challenges with the model design. In a locust the feed forward inhibition can be triggered by large, fast moving translating objects to suppress false collision alerts. Because of the size and speed of cars and the slower rate of growth of larger approaching objects no distinction can be made between an approaching object and a translating car using the feed forward mechanism. In addition the smaller sizes of road markings such as zebra crossings as compared to cars create a faster rate of growth near to the point of collision with the car. Even though they are not on a direct collision course with the sensor the excitation caused by the markings can be similar to that caused by a colliding car. Clearly the locust LGMD has not evolved to cope with large objects such as cars or to deal with road markings. This has resulted in the need to adapt the models to cope with these situations. To some extent this has been achieved through the use of the EMD to suppress translating and rotational stimuli; however other components to distinguish road lines are the most important supplements (see below and in Section 6.2, at application details).

In general translatory movement only triggers false collision detection alarms when the car is stationary or moving at very low speeds (< 5 km/h), as once moving beyond these speeds the translating object is a sufficient distance away, unless on a collision course, to allow lateral inhibition to cope with the translatory movement. Information on the speed of the car is used to suppress the responses of the collision detection alarm if the car is stationary or moving very slowly. In addition information combining the turning radius of the car and speed is also used to determine if the car is turning a very tight corner, which can also trigger false alarms and the alarm can again be suppressed in these situations. The work on the biological models was focused in two areas, and their implementation can be found in Section 6.2.

Firstly it was focused on mechanisms to differentiate between colliding objects, such as people, cars and non colliding objects such as road lines. This unit looks at the spatial patterns of excitation found in the biologically filtered image. Analysis of these patterns is able to determine what the object is and trigger the appropriate collision warning or avoidance action. For example, road lines will suppress the response of the LGMD causing no action to be taken where as detection of people or cars moving across the field of view, which are a potential collision threat, may provide an early warning audible alarm so drivers can take action to avoid the collision. Finally unavoidable collisions may be detected where actions such as pre-tensioning seatbelts or applying brakes could be triggered. Using the spatial patterns of the biologically filtered images will require considerably lower computational power than analysing unfiltered images and allow decisions of the type of threat to be made much faster.

5.2.5 System Tuning and Evaluation

The models were tuned and tested either by automotive video sequences or in real traffic environments. Images were taken at 25 frames per second with a spatial resolution of 150 x 100 pixels as defined by the sensor of our test bed. Commonly occurring automotive sequences along with collisions with full size model cars and people were captured and analyzed with the models. Real situation counterparts were tested at the VOLVO Car Company test drive area.

Image sequences involved actual collisions with a balloon car, and near miss with a person and other vehicles. Models were required to trigger collision alarm at least 0.5sec before the impending collision in sequences and situations that were classed as collisions or near hits- this area characterizes the sensitivity of the algorithms, while the selectivity is defined by other sequences that were required not to trigger the collision alarm.

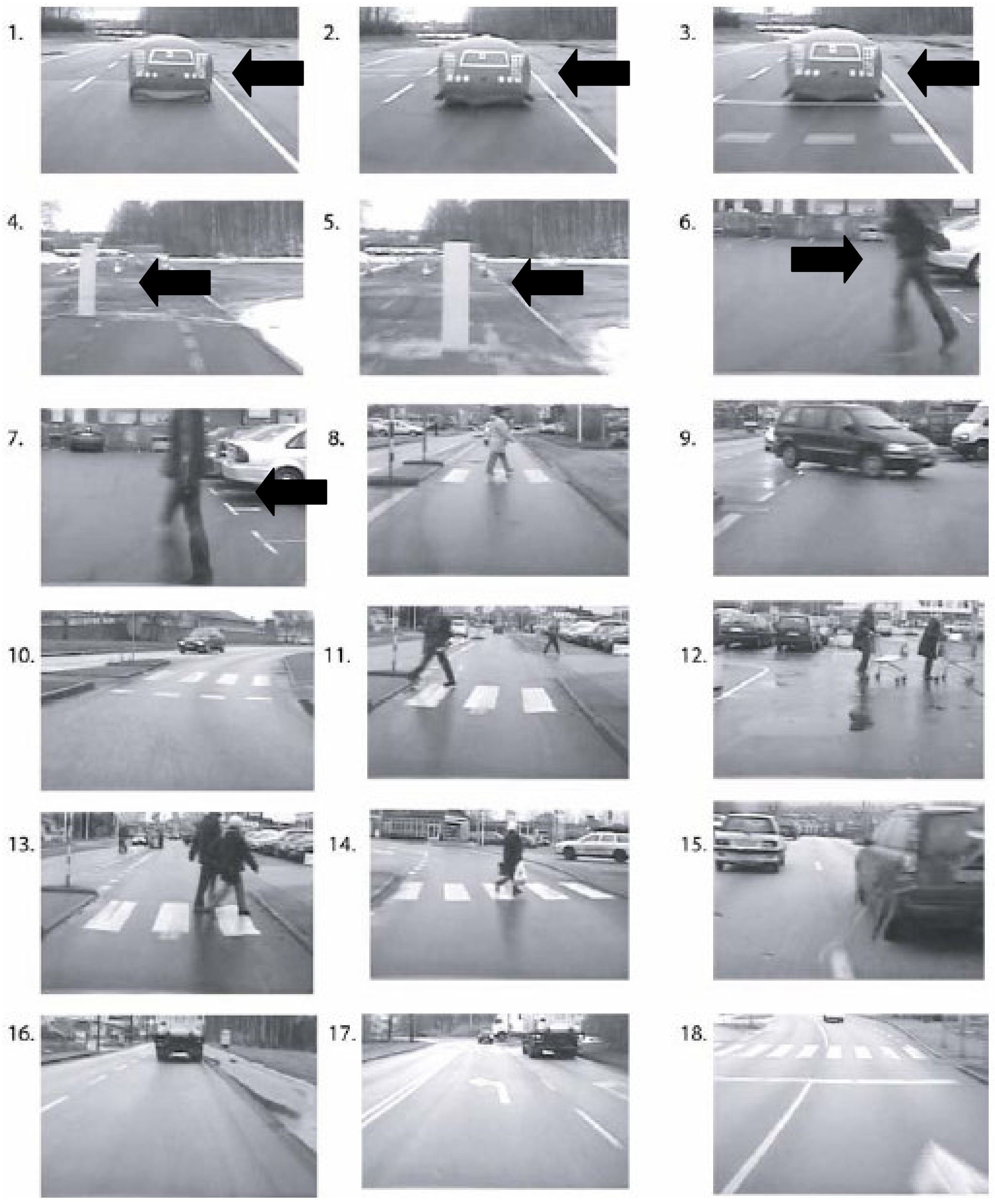


Figure 47 Typical snapshots of test sequences Black arrows indicate unambiguous collision threats with different objects and persons.

NO	Collision threat	Speed	Original BSC	AI	IFI120	AI+ IFI120
1	unambiguous	fast	520	520	840	720
2	unambiguous	fast	520	600	920	840
3	unambiguous	fast	520	520	1200	1200
4	unambiguous	fast	80	80	80	80
5	unambiguous	fast	-40	-40	480	-240
6	high	slow	880	920	880	880
7	high	slow	1080	1080	1080	1080
8	no threat	slow	N	N	N	N
9	ambiguous	slow	N	Alarm	alarm	alarm
10	no threat	turning	false alarm	false alarm	false alarm	false alarm
11	no threat	slow	N	N	false alarm	N
12	no threat	slow	N	N	N	N
13	ambiguous	slow	alarm	Alarm	alarm	alarm
14	no threat	fast	false alarm	false alarm	false alarm	False alarm
15	no threat	turn	false alarm	false alarm	false alarm	False alarm
16	No threat	fast, turn	N	N	N	N
17	No threat	turn	false alarm	N	false alarm	false alarm
18	No threat	fast	false alarm	false alarm	false alarm	false alarm

Table 6 An example of the evaluation tables: different video sequences processed by different versions of the BSC model. Numbers show the time in milliseconds by which the alarm signals precede the actual collisions. AI: denotes a model with space variant asymmetric inhibition; not explained in this thesis. IFI120: interframe interval is 120msec; AI+IFI120: interframe interval is 120msec+space variant asymmetric inhibition; N: no alarm signal was triggered; "-" denotes that the corresponding sequence was not tested by the specific model

Real traffic automotive situations

After giving examples of the behavior of the model in different situations we give the summary of tests in Table 7.

Figure 48 shows the that first the speed and the deceleration was low and thus it did not produced significant danger level.

The next figure, Figure 49Figure 49, shows a hard breaking at 20 km/h speed. It can be seen that a significant threat is signaled just before the driver started to break.

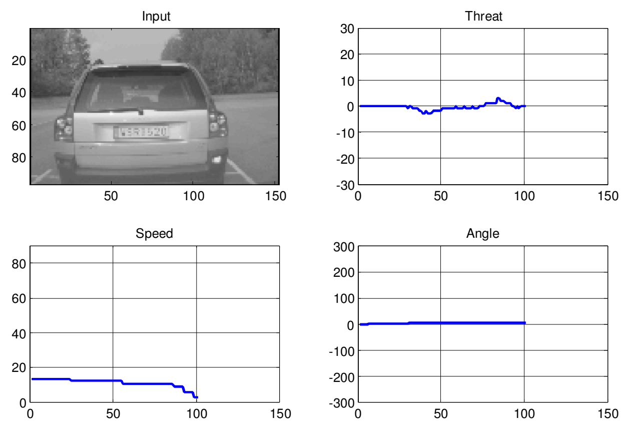


Figure 48.Low speed, soft breaking.

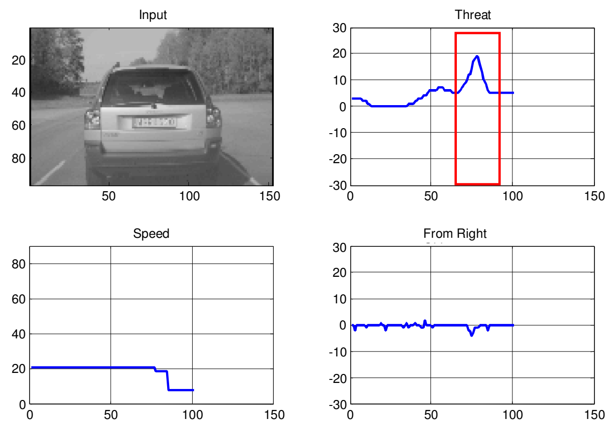


Figure 49. Low speed hard breaking.

The next figure shows that a negative threat is produced when departing from a car.

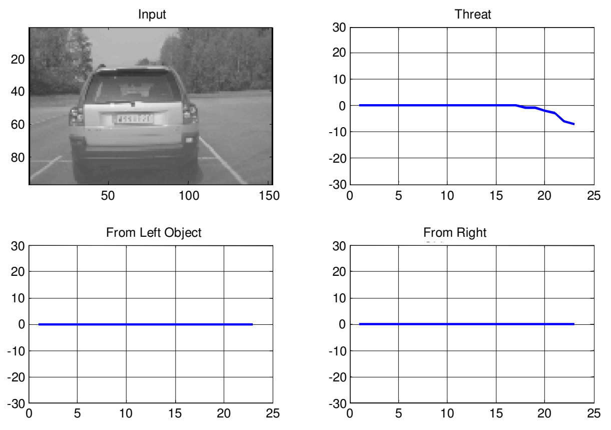


Figure 50. Departing backward from the standing car.

Shadow

The shadows produces false alarms because the expansion of shadow produced patterns produces virtual objects. To cancel this we implemented a simple object classification algorithm described later in the Chapter Applications.

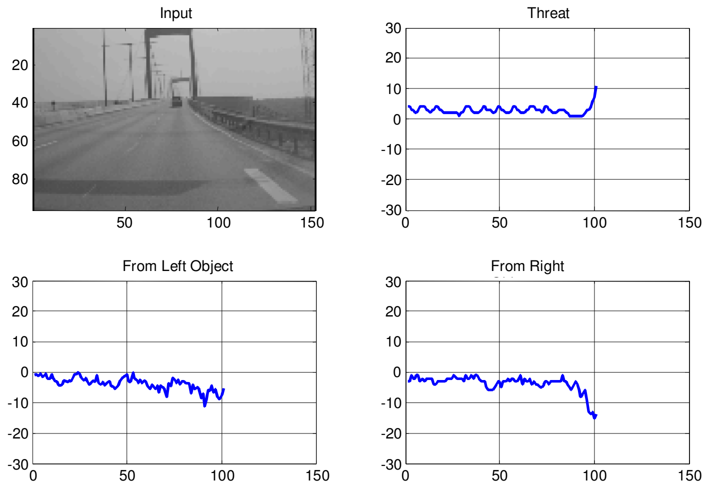


Figure 51. Shadow.

Balloon Car	100%
Real Car (optional)	100% at low speed Over 20km/h can not be tested well
Pedestrian test • Left • Right	50-100% Depending on the and distance background
Pedestrian test during turning	100%
Drive around to see general behavior	One false alarms/5 minutes

Table 7 Evaluation of sensitivity and selectivity of the locust system

5.2.6 Computational cost

For the details of the hardware platform we measured the performance see the Application chapter, section 6.2. The most time consuming part is the motion estimation. All of the computing is less than 7ms. There is still enough time frame for additional high level recognition tasks. It is around 30 ms considering 25 fps processing.

Image size	152×96
Optimised Locust model	44 μs
Original Locust model	188 μs
Noise Filtering	3.4 μs
Motion estimation	4.63 ms
Selectivity enhancement alg.	33 μs
Cycle time, excluding IO and sensor readings	6.65ms

Cycle time changes with sensor integration time. Maximal cycle time is 40ms that corresponds to 25 fps.

5.3 An analogic algorithm for parallel multiple collision prediction based on isotropic diffusion

The model described in the previous section is not using the knowledge in Eq. (5.5): when an observer has exact information about the growing diameter (and not only the rate of growing) of the projected image of an approaching object , one can calculate the exact time-to-contact. This requires robust, good quality image segmentation, where individual objects can reliably be identified. Although this is a prerequisite that is really difficult to fulfill, we have studied the possibility to compute more precise TTC provided we have black and white images of approaching objects after a correct segmentation.

5.3.1 Theoretical background

Here we are going to expand equation (5.5) to show that information about the area of the objects are enough for further calculations.

Area of a static object:

$$A = \frac{1}{2}\int_0^{2\pi} r^2(\varphi)\,d\varphi \tag{5.35}$$

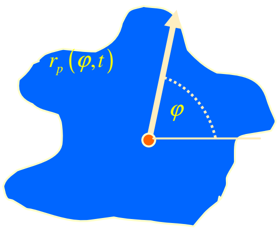


Figure 52 Area of a 2D object. For the notations see Figure 39

Area of an approaching object using Eq (5.3):

$$A_p(t) = \frac{1}{2}\int_0^{2\pi} r_p^2(\varphi,t)d\varphi = \frac{1}{2}\int_0^{2\pi}\left(k_r(\varphi)t^{-1}\right)^2 d\varphi = t^{-2}\frac{1}{2}\int_0^{2\pi} k_r^2(\varphi)d\varphi = k_a t^{-2} \tag{5.36}$$

Its derivative:

$$\dot{A}_p(t) = -2k_a t^{-3} \tag{5.37}$$

$$\frac{A_p(t)}{\dot{A}_p(t)} = -\frac{1}{2}t \tag{5.38}$$

And similarly to (5.5) we define time-to-contact (τ) as:

$$\tau = -t = -2\frac{A_p(t)}{\dot{A}_p(t)} \tag{5.39}$$

So our problem is solved as far as we can tell the ratio between the area and the rate of change in the area of an object.

Starting from an image sequence of an approaching object the difference between subsequent frames gives us the actual rate of change in the area in the form of a black ring(see Figure 53). The comparison of the area of that ring (change between frames) and the white hole –the area of the object – can be done with the help of space-limited isotropic diffusion (see Figure 54). Reaching its equilibrium state the level of the grey within the area of the object will be proportional to the ratio between the initial number of black and white pixels.

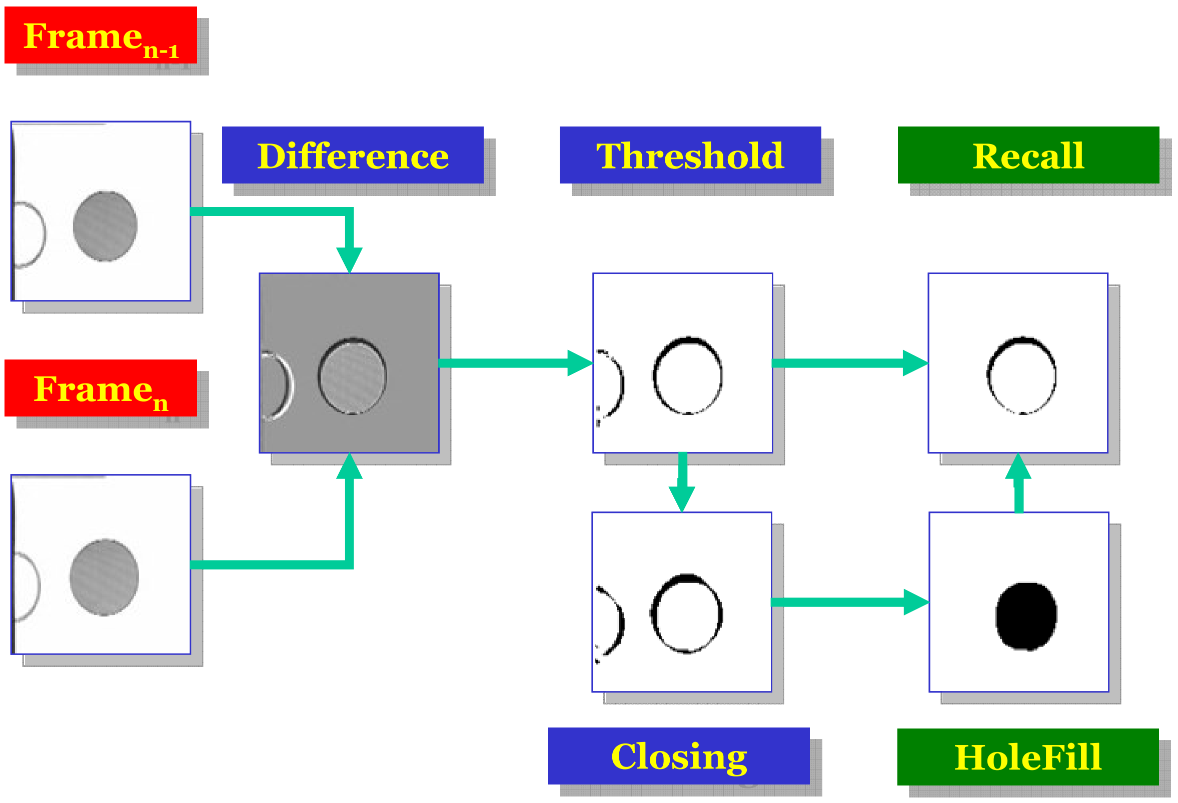


Figure 53 Identification and segmentation of an approaching object: the area and the change in the area between two consequtive frames are shown in the last column

The diffusion equation formula [Kimia & Siddiqi, 1996]:

$$\frac{d}{dt} I(\vec{x},t) = c\,div[\,grad(\,I(\vec{x},t\,))\,]\,, \qquad I(\vec{x},t_0\,) = I_0(\,\vec{x}\,) \qquad (5.40)$$

where $I(\vec{x},t\,)$ is the image intensity ($I_0(\,\vec{x}\,)$ is the original image), the vector $\vec{x} = [\xi\ \eta]$ represents the spatial coordinates, the time variable *t* can also be interpreted as the scaling parameter and *c* is the thermal conductivity. As pointed out by Witkin[1983], convolution of the original signal with Gaussians at each scale is equivalent to solving the heat equation (5.40) with the original signal as initial condition ($I(\vec{x},t\,) = G_\sigma * I_0(\,\vec{x}\,)$ where *G* is a Gaussian kernel with $\sigma^2(\,t\,)$ variance.

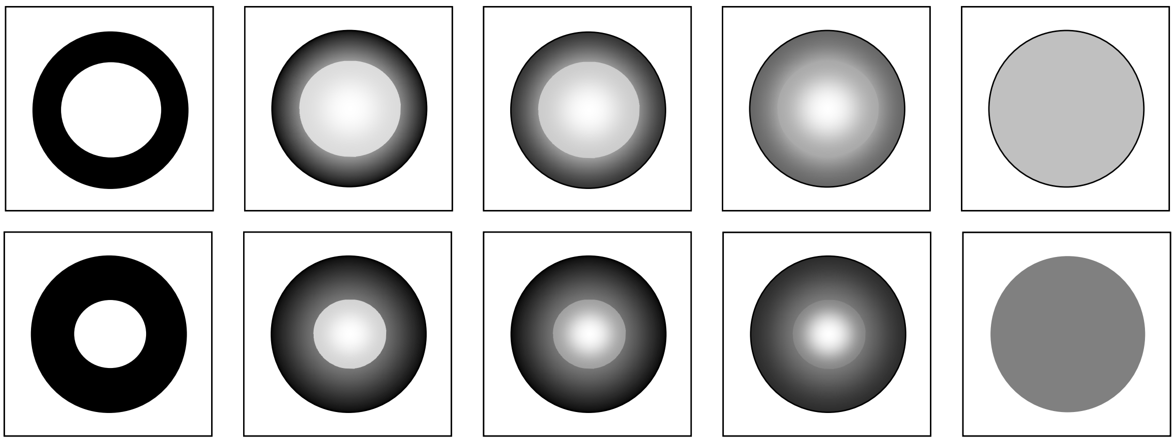


Figure 54 Area limited (masked) isotropic diffusion: subsequent samples from left to rigth starting from two different images. The ratio between the black rings and white holes are encoded in the grey level of the equilibrium state.

Using an eight-neighbor discretization of the Laplacian in 2D with the notation $I_{ij}(t) = I(i\Delta\xi, j\Delta\eta, t)$ and setting $\Delta\xi = \Delta\eta = 1$ (Δ is the difference operator) we obtain:

$$\begin{aligned}\frac{d}{dt} I(\xi,\eta,t) &= c\left(\frac{\partial^2 I(\xi,\eta,t)}{\partial\xi^2} + \frac{\partial^2 I(\xi,\eta,t)}{\partial\eta^2} \right) \\ &\approx c\frac{1}{8}\Big(I_{i-1j-1}(t) + I_{ij-1}(t) + I_{i+1j-1}(t) + I_{ij-1}(t) + I_{ij+1}(t) + \ldots \\ &\qquad + I_{i+1j-1}(t) + I_{i+1j}(t) + I_{i+1j+1}(t) \Big) - cI_{ij}(t)\end{aligned} \tag{5.41}$$

The heat equation can be directly mapped onto the CNN array resulting in the following simple template:

$$A = \begin{bmatrix} 0.125 & 0.125 & 0.125 \\ 0.125 & 0 & 0.125 \\ 0.125 & 0.125 & 0.125 \end{bmatrix}, \; B = 0, \; z = 0 \tag{5.42}$$

(5.42) first appeared in the original paper laying the groundwork of the CNN theory [Chua & Yang, 1988], in which the remarkable similarity between the spatially discretized PDEs and the nonlinear ODEs governing the CNN array was first pointed out. Since then, CNN approach has been discussed for solving various types of PDEs [Roska et al, 1993,1995; Gobovic & Zaghloul, 1994; Kozek et al., 1995; Kozek & Roska 1996], namely for Laplace, Poisson, Burger's, Navier-Stokes and some reaction-diffusion type equations.

To confine the diffusion to a predefined subspace, pixels not belonging to this subspace should be inactivated by masking. Strict zero-flux boundary conditions are required for the active pixels along the border of the masks – they must not interact with inactive cells at all. That means active cells must be cut from passives, or passives have to be isopotential(not feasible if there are two active neighbors with different potentials). The corresponding modified CNN equation (for eight direction diffusion) can be written as:

$$\frac{d}{dt}x_{i,j}(t)=m(i,j)\left(-x_{i,j}(t)+\sum_{\substack{k,l\in N\\ k\neq 0,l\neq 0}}\frac{m(i+p,j+q)\,y_{i+k,j+l}(t)}{\sum\limits_{\substack{p,q\in N\\ p\neq 0,q\neq 0}}m(i+p,j+q)}\right) \tag{5.43}$$

where $m(i,j)\in\{0,1\}$ is the space-variant mask, or indicator function defining pixel (i,j) to be active or inactive.

Concerning feasibility of implementation, we show the diffusion operation on a resistive grid, since technically this is quite simple to build, , and the Q-Eye (AnaFocus) focal-plane processor with QCIF resolution has a very similar function implemented.

The state equation of a resistive grid:

$$\frac{d}{dt}p_{i,j}(t)=\sum_{\substack{k,l\in N\\ k\neq 0,l\neq 0}}\frac{p_{i+k,j+l}(t)-p_{i,j}(t)}{r_{i,j,i+k,j+l}}=m_{i,j}\sum_{\substack{k,l\in N\\ k\neq 0,l\neq 0}}\frac{m_{i+k,j+l}}{r}\left(p_{i+k,j+l}(t)-p_{i,j}(t)\right) \tag{5.44}$$

Where $p_{i,j}$ is the potential at the grid point (i,j), N is the set of point coordinates defining the neighborhood and r(i,j,p,q) is the resistance between points (i,j) and (p,q). The indicator function $m(i,j)\in\{0,1\}$ defines that a grid point (i,j) is active(connected) or there is a break in the wires. Concerning space invariant r we can write:

$$\frac{d}{dt}p_{i,j}(t)=m_{i,j}\left[-p_{i,j}(t)\sum_{\substack{p,q\in N\\ p\neq 0,q\neq 0}}\frac{m_{i+p,j+q}}{r}+\sum_{\substack{k,l\in N\\ k\neq 0,l\neq 0}}\frac{m_{i+k,j+l}}{r}p_{i+k,j+l}(t)\right], \tag{5.45}$$

or

$$\frac{d}{dt}p_{i,j}(t)=\frac{m_{i,j}\sum_{\substack{p,q\in N\\p\neq 0,q\neq 0}} m_{i+p,j+q}}{r}\left[-p_{i,j}(t)+\sum_{\substack{k,l\in N\\k\neq 0,l\neq 0}}\frac{m_{i+k,j+l}}{\sum_{\substack{p,q\in N\\p\neq 0,q\neq 0}} m_{i+p,j+q}}p_{i+k,j+l}(t)\right] \tag{5.46}$$

So we have got an equivalent state equation to (5.43) at equilibrium:

$$\frac{d}{dt}p_{i,j}(t_e)=0=m_{i,j}\left[-p_{i,j}(t_e)+\sum_{\substack{k,l\in N\\k\neq 0,l\neq 0}}\frac{m_{i+k,j+l}}{\sum_{\substack{p,q\in N\\p\neq 0,q\neq 0}} m_{i+p,j+q}}p_{i+k,j+l}(t_e)\right] \tag{5.47}$$

And at equilibrium the potential is the average of the initial potentials:

$$p_{i,j}(t_e)=\frac{\sum_{k=1,l=1}^{k=M,l=N} m_{k,l}p_{k,l}(t_0)}{\sum_{k=1,l=1}^{k=M,l=N} m_{k,l}}, \forall i,j \,|\, m_{i,j}=1,$$

5.3.2 Design and evaluation of the algorithm

The flowchart of the algorithm from preprocessing till collision warning is shown in Figure 57. Preparation steps work as a filter: they remove objects not growing in a concentric manner. (see Figure 53). Finally we have got concentric contours: the contour itself represents the rate of change between frames n_k and n_{k+1}, that is:

$$\Delta A_{k+1,k}=A_{k+1}-A_k$$

We can approximate TTC from (5.39) via:

$$\tau=-t=-2\frac{A(t)}{\dot{A}(t)}=-2\frac{A_k}{\Delta A_{k+1,k}} \tag{5.48}$$

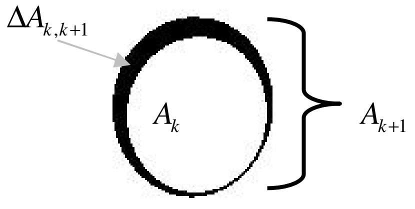


Figure 55 Area definitions in subsequent images of an approaching object

And that is the approximation isotropic diffusion (reaching its equilibrium state) calculates for us: (see Figure 55) creating a 'warning level map'(see Figure 56):

$$x_{i,j} = \frac{\Delta A_{k+1,k}}{A_{k+1}}, \tag{5.49}$$

where $x_{i,j}$ is the state of a cell belonging to the approaching object's image. To get (5.48) from (5.49):

$$1 - x_{i,j} = \frac{A_k}{A_{k+1}}, \tag{5.50}$$

$$\frac{1 - x_{i,j}}{x_{i,j}} = \frac{A_{k+1}}{A_k} \frac{\Delta A_{k+1,k}}{A_{k+1}} = \frac{\Delta A_{k+1,k}}{A_k} \tag{5.51}$$

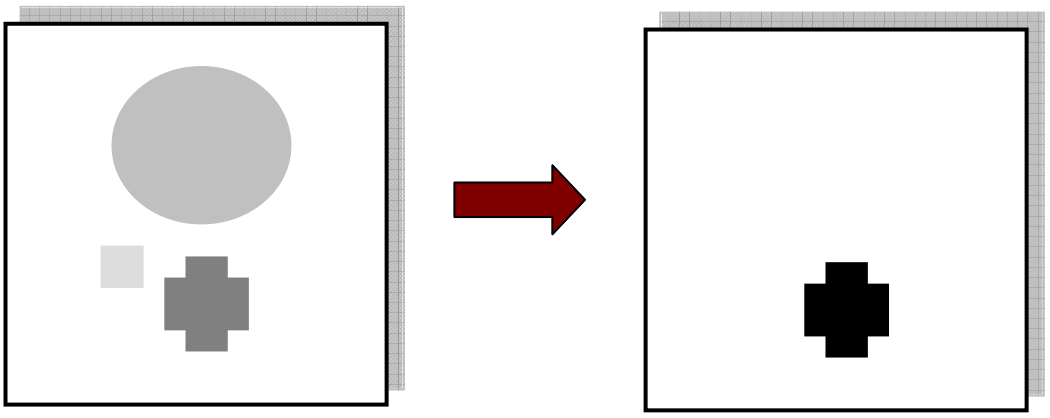


Figure 56 Warning level map and selection of the objects close enough to collision by a simple threshold operation

Thus τ can be approximated by:

$$\tau = -2\frac{A_k}{\Delta A_{k+1,k}} = 2\left(1 - x_{i,j}\right)/x_{i,j} \tag{5.52}$$

This calculation is necessary for setting the threshold level of warning, afterwards direct evaluation of the warning level maps by threshold is possible to make collision/no collision decisions.

For testing purpose we simulated the approach of 3 distinct objects with different sizes and speeds (arbitrary units: time in frames, size in sensor density). According to the simulation parameters collision would happen at the 21st frame. Following diffusion operation the gray levels of the pixels in the warning map belonging to the individual objects are shown in Figure 58. In Figure 59 the transformed levels are shown according to (5.52). The figures show quasi-linearity in the last 5-6 frames and excellent match between the expectation and real values.

Another test with a video-footage recorded by a camera fixed to a 6 wheeled robot-car is presented in Figure 60.

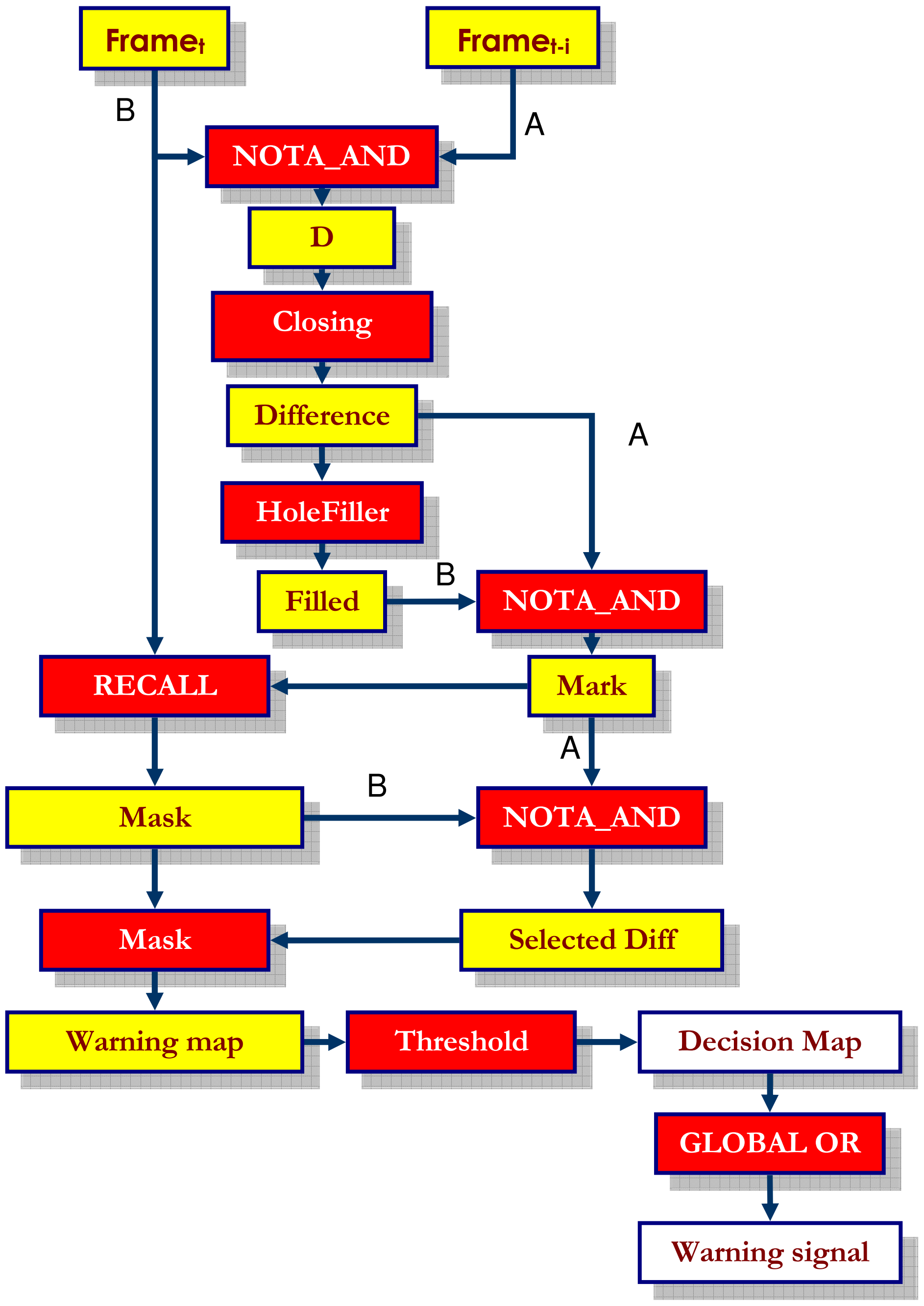


Figure 57 Flowchart of the diffusion based collision warning algorithm

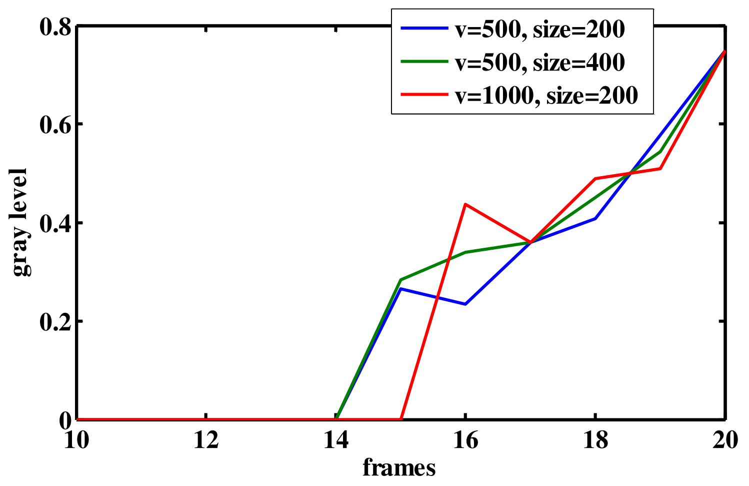


Figure 58 Test on simulated data. Approaching of 3 distinct objects are simulated with different sizes and speeds (arbitrary units). According to the simulation parameters collision happens at the 21st frame. Following diffusion operation the gray levels of the pixels belonging to the individual objects are shown.

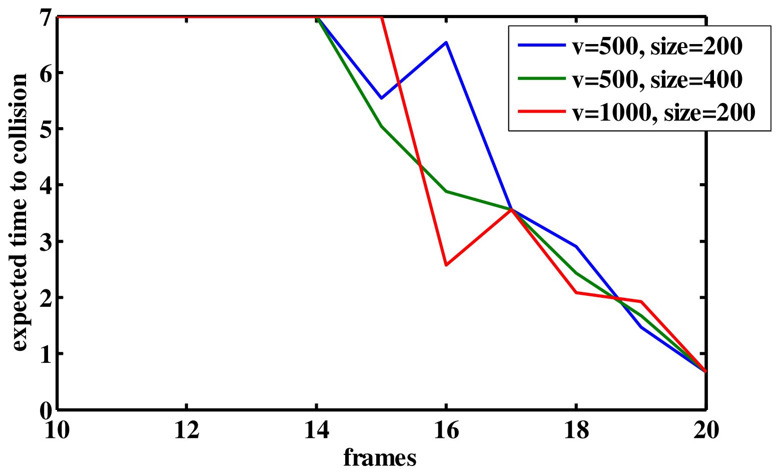
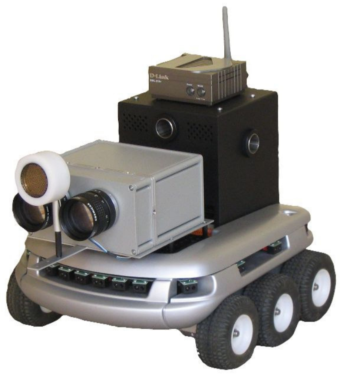
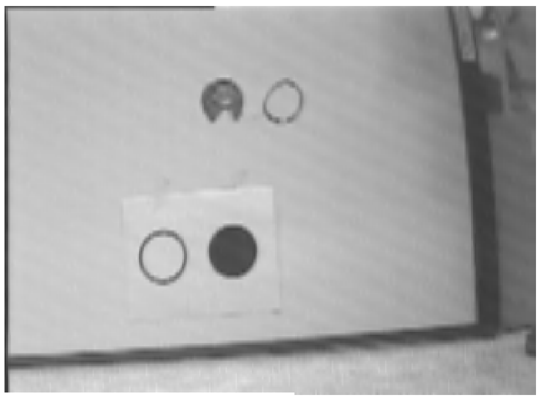


Figure 59 Test on simulated data (same data as in Figure 58). Expected time to collision is calculated by

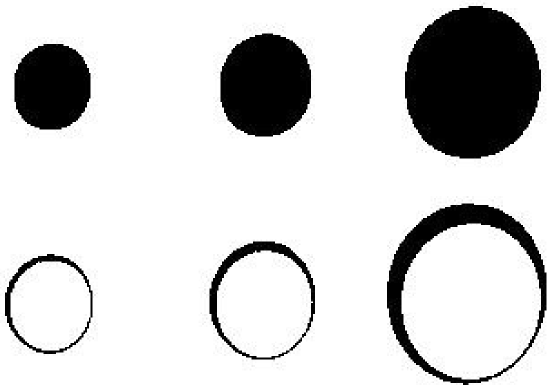
In Figure 60 we present a test of the algorithm: a video footage of approaching disc pattern was recorded and analyzed. Here we show the actual counts of pixels belonging to the object and to the difference between subsequent frames.

A

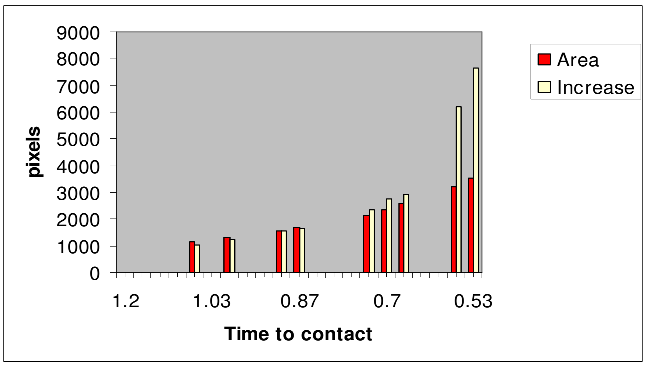




B



C



D

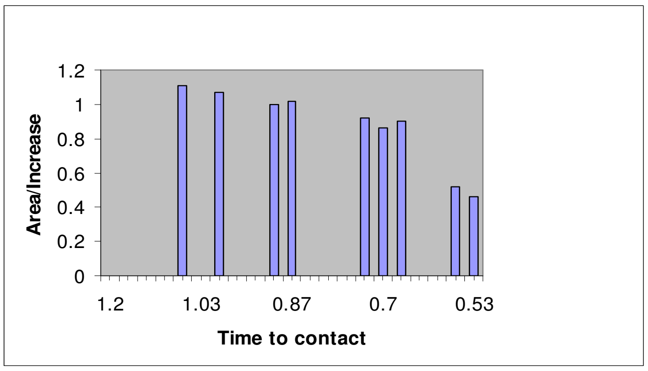


Figure 60 Experimentally recorded video footage and simulation of the diffusion based collision warning algorithm A The test vehicle and scenario B Segmented, preprocessed snapshots from the video: difference between subsequent frames are shown in the bottom row. D The size of the area and difference is shown in pixels as a function of time E Ratio between the area and the difference are shown as a function of time

6 Application

6.1 Analysis of spatial and temporal patterns in multiple single neuron recordings

The most trivial place of the statistical methods described here is at the offline analysis of simultaneous multi-electrode single-unit recording studies.

In the preparation phase expensive and time consuming studies can benefit from quasi real-time validation of the proper experimental setup. For example, before chronic implantation of electrodes in the scull the experimenter would like to see if the localization of the electrodes are optimal: whether the sites at they measure give rise to synchronization patterns expectedly.

Considering some future applications real-time processing and evaluation will be the minimum requirement: prediction and detection of epileptic seizures via intracranial electrodes are one of the task the generation and analysis of surrogate data may help.

Finally, the data exploration technique proposed in the thesis above has an ambitious – but not unrealistic – goal: recognition of important activity patterns seems to be a crucial step on the way of decoding the language of neurons, what in turn can ultimately be a synonym of "reading the mind". A fast, intelligent and reliable communication with the nervous system is the basis for advanced human-machine interfaces: on one hand, online translation of activity patterns may serve as a control signal for the actuators. On the other hand, sensory prosthesis have to "talk" to specific brain areas by means of sensible patterns.

6.1.1 An example for multiple single neuron data and analysis

One example is shown here: I studied multiple single-cell recordings from mammalian retinal ganglion-cells. The data was registered in the Friedrich Miescher Institute for Biomedical Research (Basel, Switzerland) by David Balya. In the experiment, a number of retinal cells (mostly ganglions) were stimulated by different video-sequences and their parallel activity were recorded by an 8x8 multi-electrode matrix: density was 200 microns, 2.2mikron/1 pixel. The recording session lasting 12 second was repeated four times with the same video footage. After spike sorting and discretization of spike times 20 channels were analyzed here. All the 2^{20} possible patterns have been searched for and tested for significance (see Figure 61).

A number of stimulus-locked and stimulus timing independent high order synchrony patterns were recognized and analyzed that traditional pair-wise correlation techniques could not detect. Evaluating the time scale of the non-stimulus locked patterns suggest that the accuracy of the coordination of certain retinal cells are in the milliseconds range. An example for $\xi=3$ pattern is shown in Figure 61. This particular pattern was found to be significant in every sessions, but none of its subcomponents of complexity two– this was validated by simple cross-correlation technique. Interestingly the pattern $\xi=4$ surrounded by the blue rectangle was not recognized to be significant, perhaps because there is a general coarse increase in firing rates around the pattern – thus the probability of the formation of that kind of pattern is increased.

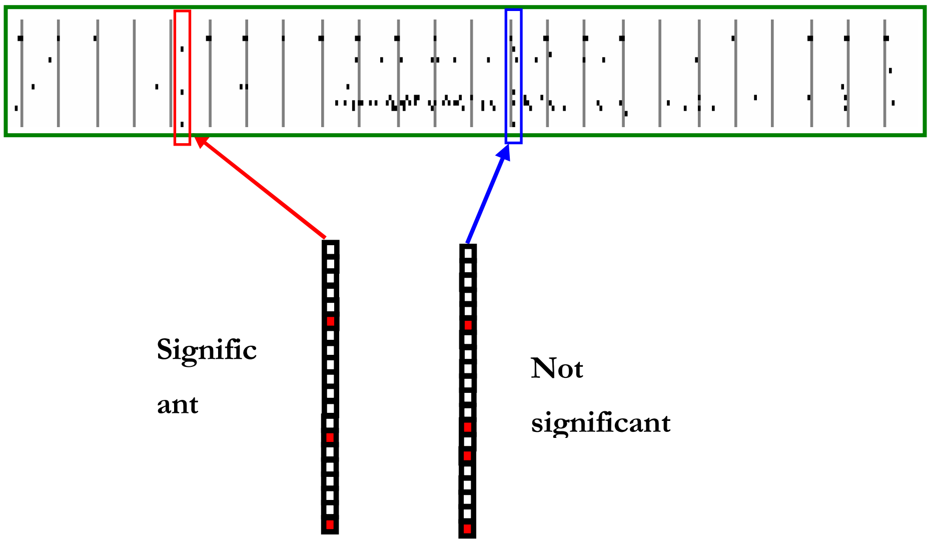


Figure 61 Rasterplot of high complexity patterns: pattern ($\xi=3$) indicated by red arrow found to be significant in every session of the experiment, while the pattern with higher complexity ($\xi=4$) is not recognized as significant.

Finally I simulated 10 sessions of stationary independent data with the same spiking frequences and length as in the experimental data. A sort of cross-validation of what we have found that analyzing it with the same parameters as the original no significant patterns could be detected.

6.2 Automotive application of the Collision warning system

Within the framework of an international project in a broad cooperation we have built and tested the proposed collision prediction system in a personal car. Figure 62 presents the communication path between a Volvo XC90, the Compact Vision System and a PC.

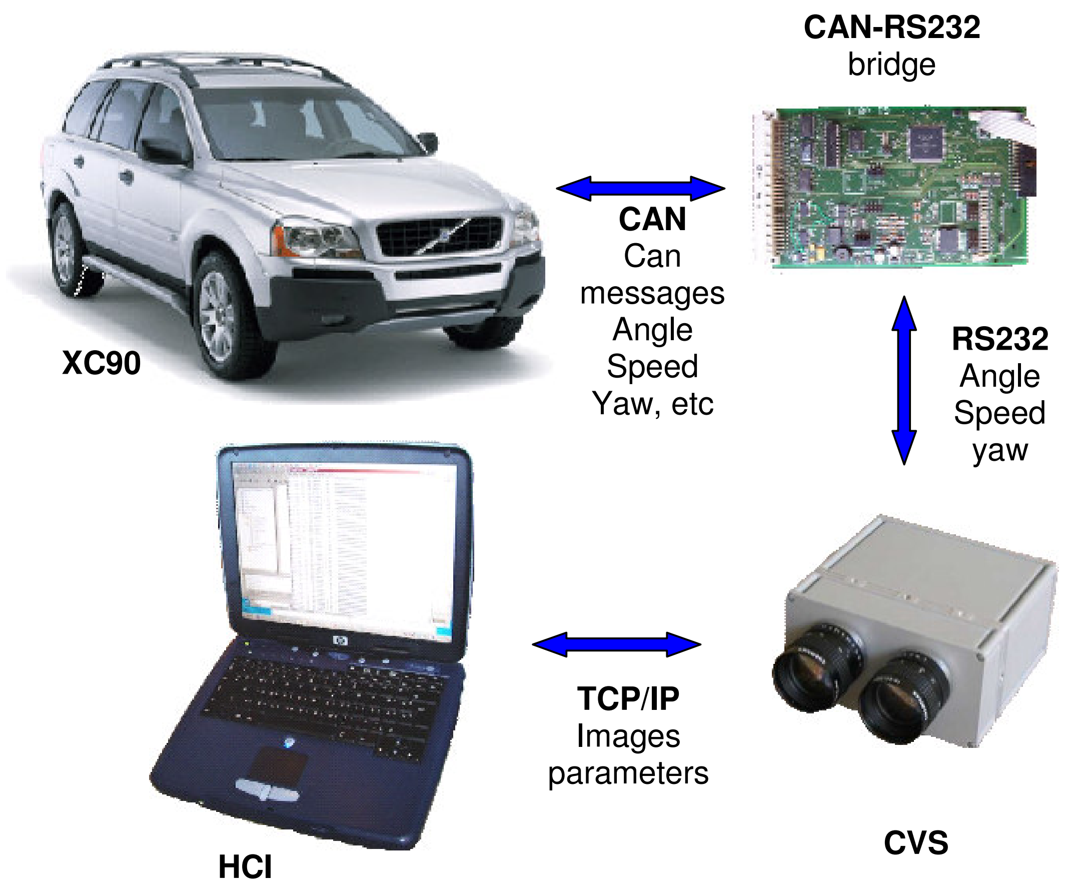


Figure 62 Locust inspired collision warning system is interfaced with a personal car

The Compact Vision System is a modification of the Bi-i intelligent camera specialized in image processing, which is the product of AnaLogic Computers (www.analogic-computers.com). It comes with software environment. This means SDK and API and configuration tools. CVS is not only the workhorse of the algorithmic experiments, but also the test bed of the LOCUST Vision Chip. Beside the LOCUST simplified model other functional units are integrated as well to get a robust, adaptive collision warning module (see Figure 63).

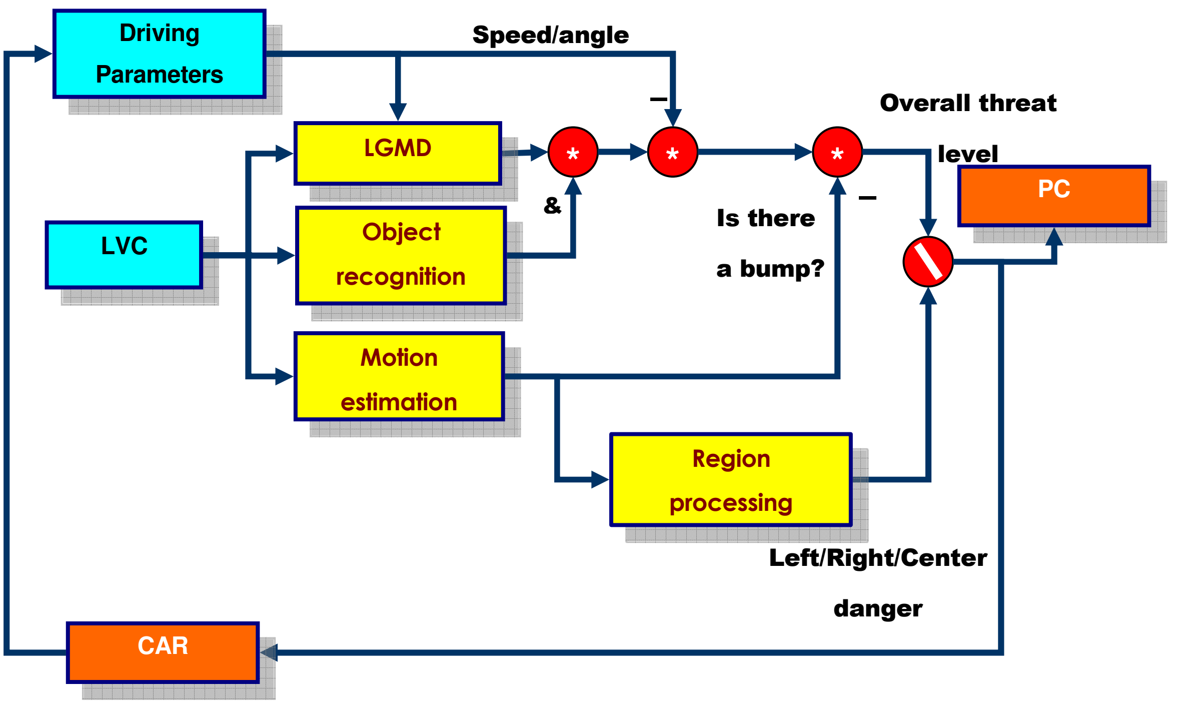
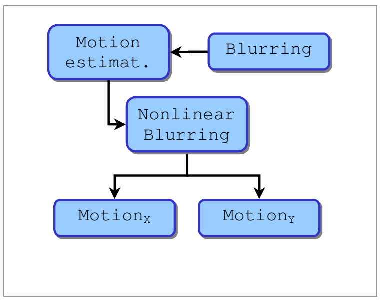


Figure 63 Structure of the complex integrated collision warning system

Motion Estimator

The motion estimation is based on the standard block matching method. It takes a block of pixels and compares it to displaced blocks in a certain neighborhood if the center pixel of the block to find the most probable displacement of the pixel neighborhood. Comparison is computed by taking the absolute difference for each pixel pair between two blocks and then summing the absolute value of the differences. The minimal difference between the reference and the displaced blocks gives the motion of a pixel. This is done for every second pixel in the image. This means resolution decrease, but this is acceptable.

The motion estimation yields two images. They are the two component of the motion vector for each pixel of the original image. The motion images are nonlinearly blurred. That means the blurring occurs only where the image is not zero (gray level 127).



As a final step the pixels of each motion images are averaged, yielding the two components of the general motion vector of the image. This estimates the self motion of the camera in the projected 2D coordinate frame.

Object classifier

A very simple but robust object classifier is defined and implemented to minimize the number of false alarms. It also makes possible to choose very low warning thresholds –enhancing the sensitivity while keeping selectivity- for the LGMD. The classifier uses elementary statistics of the activation patterns in the summation layer.: these are the means and deviations of the activities in the columns and rows of the array(see Figure 64). The subsystem classify a pattern as 'to be avoided' objects (pedestrian, car) or 'don't care' (road signs, shadows) etc.

$$t_{ROW} = \frac{Mean_{ROW}}{Std_{ROW}} \qquad t_{COL} = \frac{Mean_{COL}}{Std_{COL}} \qquad R_{COL/ROW} = \frac{t_{COL}}{t_{ROW}}$$

$$R_{COL} = \frac{Max_{COL}}{Mean_{COL}} \qquad R_{ROW} = \frac{Max_{ROW}}{Mean_{ROW}}$$

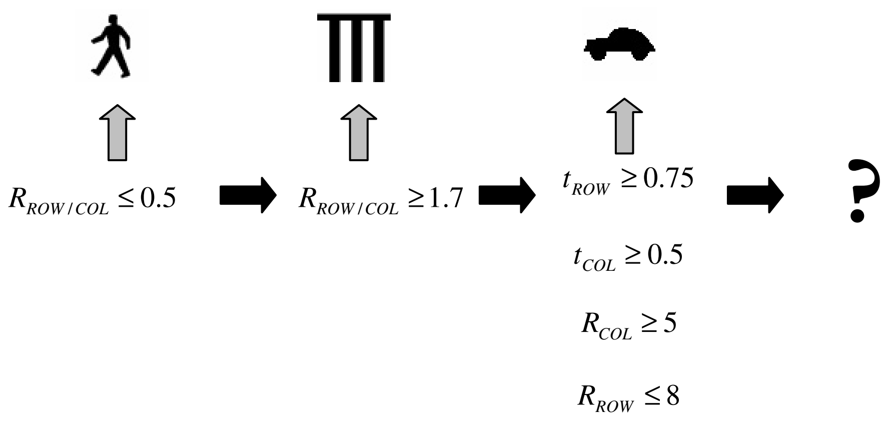


Figure 64 Object classifier using elementary statistical features of the activations in the S layer of the model.

Multiple field of view processing

To enhance the selectivity to objects moving into different directions, we divided the image into four regions (Figure 65).

1. Upper left region

   In this area moving objects are declared dangerous if they move to the right.

2. Upper right region

   In this area moving objects are declared dangerous if they move to the left.

3. Center left region and Center right region (3-4.)

   In this region motion is dangerous if in the left part the object moves to the left and in the right half the object moves to the right.

In each area the horizontal motion activity (i.e. the horizontal component of optic flow) is summed separately and average danger level is computed for regions 1., 2., and 3-4. These levels give additional threat information of objects coming form the left or right and objects being on direct collision course.

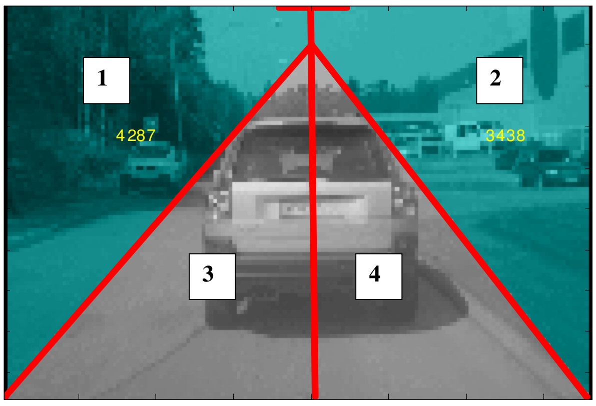


Figure 65 Multiple regions.

6.2.1 System level integration and evaluation

The whole system was fixed into a XC90 Volvo. The CVS was placed behind the windshield, in the middle (Figure 66). Power was provided by the car electrical system. A CAN-RS232 bridge was connected to the CAN bus of the car and to the RS232 connector of the CVS. It also receives power from the car. A laptop was connected through Ethernet cable to the CVS to control the processing and display the warning.

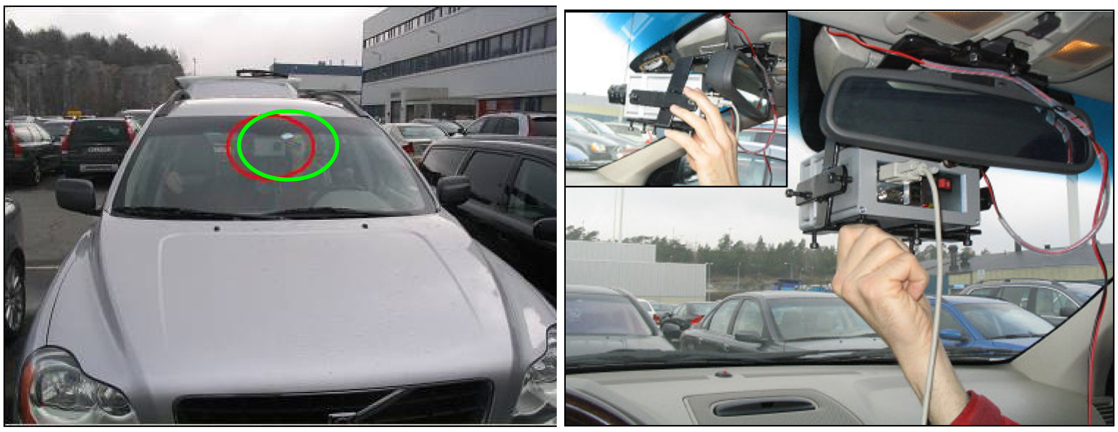
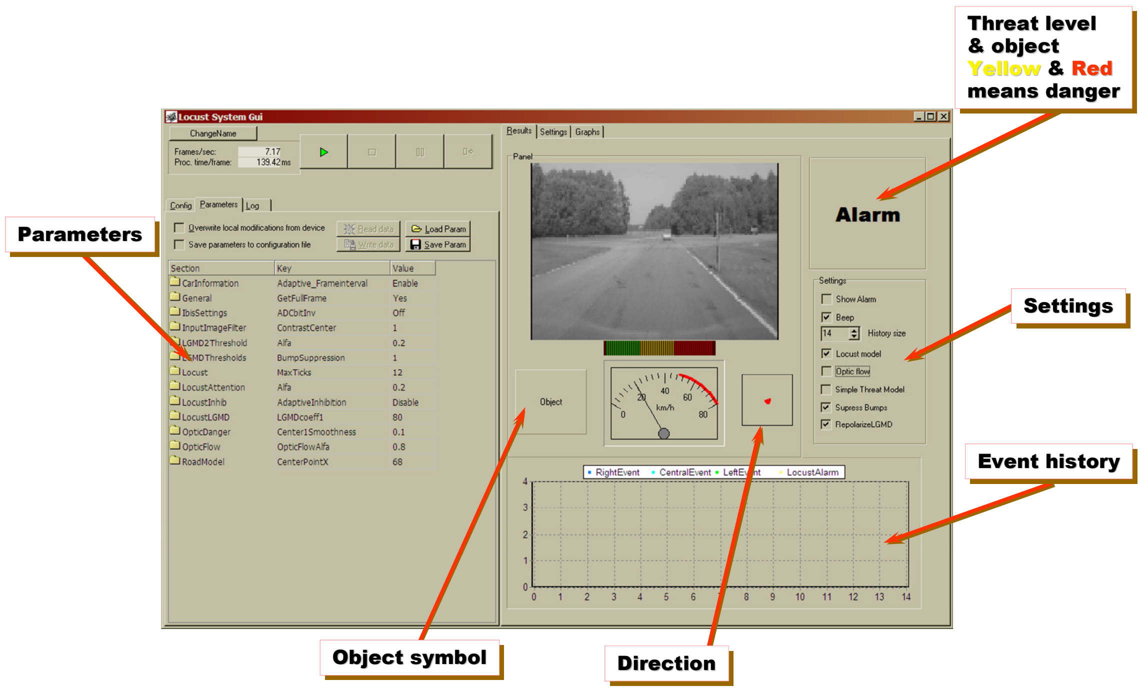


Figure 66 The CVS mounted in the XC90

The system uses data on speed and angle of the car. This makes it possible to reduce false alarms and to use speed sensitive Locust model and threat estimation. Obviously this is very important since the danger level of a situation differs very much when e.g. the car is standing or approaches slowly the zebra crossing and a pedestrian is crossing it.



Frames/sec: 7.17
Proc. time/frame: 139.42 ms

[ ] Overwrite local modifications from device
[ ] Save parameters to configuration file

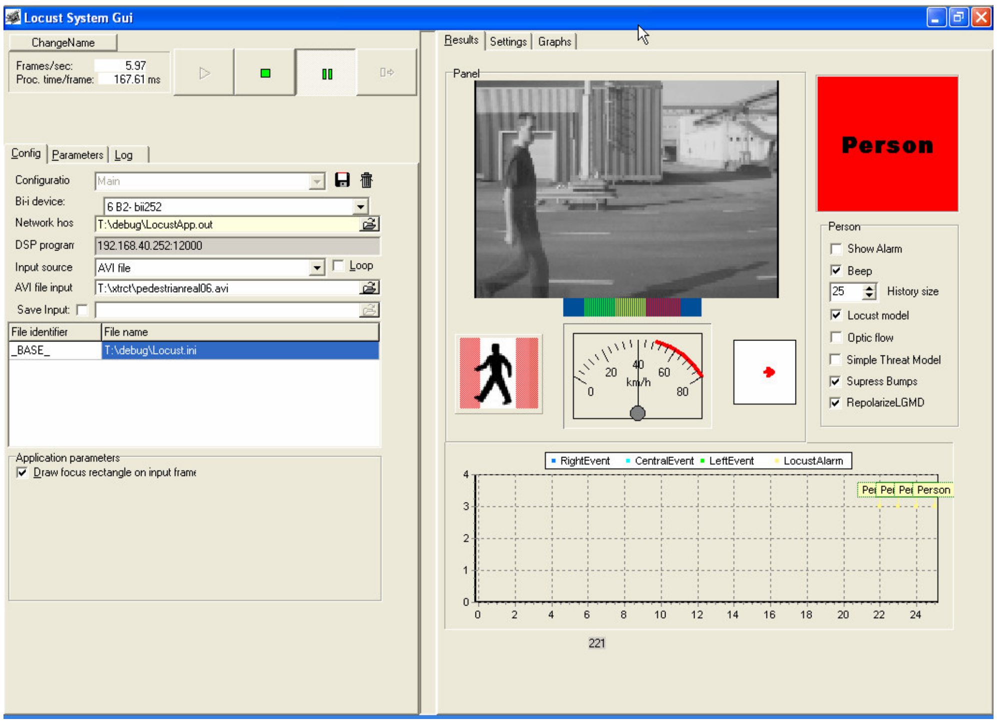
Section	Key	Value
CarInformation	Adaptive_Frameinterval	Enable
General	GetFullFrame	Yes
IbisSettings	ADCbitInv	Off
InputImageFilter	ContrastCenter	1
LGMD2Threshold	Alfa	0.2
LGMDThresholds	BumpSuppression	1
Locust	MaxTicks	12
LocustAttention	Alfa	0.2
LocustInhib	AdaptiveInhibition	Disable
LocustLGMD	LGMDcoeff1	80
OpticDanger	Center1Smoothness	0.1
OpticFlow	OpticFlowAlfa	0.8
RoadModel	CenterPointX	68

Settings
[ ] Show Alarm
[✓] Beep
14 History size
[✓] Locust model
[ ] Optic flow
[ ] Simple Threat Model
[✓] Supress Bumps
[✓] RepolarizeLGMD

Figure 67 Control Panel of the warning system

Traffic situations and behavior

A number of typical driving situations were tested. The control interface (Figure 67) made it possible to tune and optimize the parameters online. Concerning sensitivity, the system (primarily the locust module) behavior was excellent: crossing pedestrians within a specific sensible range and cars on direct collision course always triggered alarm signals. However, we can not be fully satisfied with the specificity of the algorithm: to avoid false alarms in every 5 minutes during everyday driving situations requires a lot more investment in the development of object classifiers.


Locust System Gui

ChangeName

Frames/sec: 5.97
Proc. time/frame: 167.61 ms

Config | Parameters | Log

Configuratio	Main
Bi-i device:	6 B2- bii252
Network hos	T:\debug\LocustApp.out
DSP prograr	192.168.40.252:12000
Input source	AVI file ☐ Loop
AVI file input	T:\xtrct\pedestrianreal06.avi
Save Input: ☐	

File identifier	File name
BASE	T:\debug\Locust.ini

Application parameters
☑ Draw focus rectangle on input frame

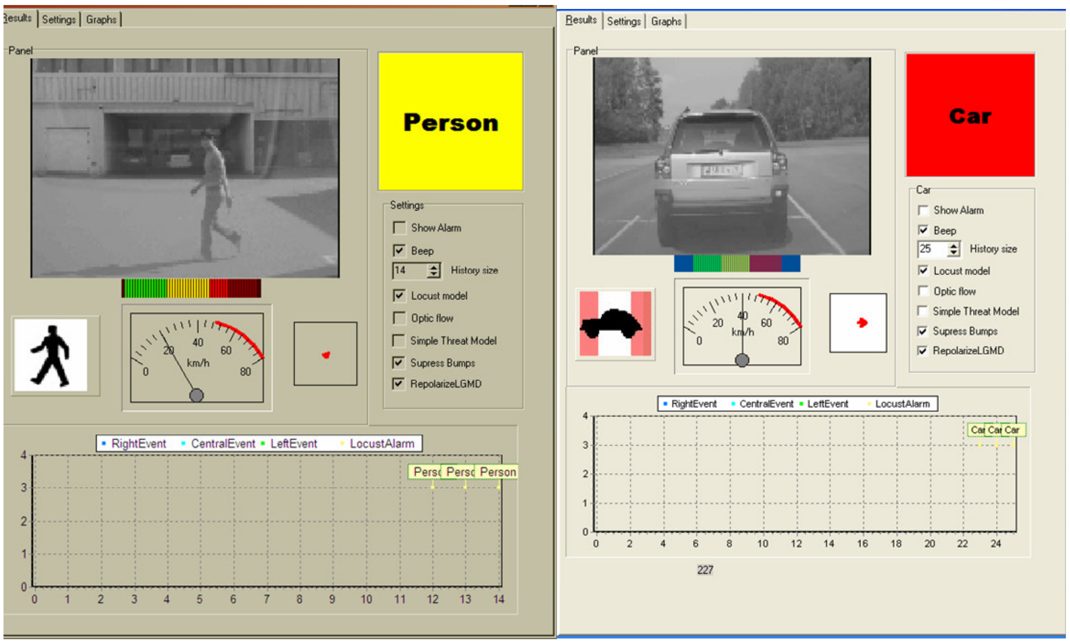
Results | Settings | Graphs

Panel

Person

Person
☐ Show Alarm
☑ Beep
25 History size
☑ Locust model
☐ Optic flow
☐ Simple Threat Model
☑ Supress Bumps
☑ RepolarizeLGMD

Figure 68 Crossing pedestrian gets too close to the car


Results | Settings | Graphs

Panel

Person

Settings
☐ Show Alarm
☑ Beep
14 History size
☑ Locust model
☐ Optic flow
☐ Simple Threat Model
☑ Supress Bumps
☑ RepolarizeLGMD

Results | Settings | Graphs

Panel

Car

Car
☐ Show Alarm
☑ Beep
25 History size
☑ Locust model
☐ Optic flow
☐ Simple Threat Model
☑ Supress Bumps
☑ RepolarizeLGMD

Figure 69 Left panel: walking pedestrian is on the border of acceptable range (medium level warn). Right panel: breaking in front of the car, almost collision.

6.3 Genetic Algorithm

As I pointed out in Section 4 typical application domains of genetic algorithms include timetabling, scheduling problems, automated design, but here I test the performance and examine the suitability of analogic binary GA in connection with the analysis of multi-dimensional (multi-channel) neural activity data, namely fMRI data. Nevertheless, the principles could be very similar in multichannel EEG recordings.

Over the past two decades, a variety of different BOLD (blood oxygenation dependent) functional Magnetic Resonance Imaging (fMRI) experiments have been done in order to understand the human brain activity pattern when doing some certain task. By recording the activity pattern of human brain as images of 3D voxels, it is possible to visualize the picture of the pattern, find statistical differences in bran activity during different tasks, and a more challenge problem is to train a classifier on the recorded data so as to predict cognitive states given any of the pattern [Haxby et al., 2001]: such as whether the human subject is reading a sentence or looking at a picture, or whether the subject is reading an ambiguous or non-ambiguous sentence, etc. (Probably too ambitious, but popular names for the process are 'decoding' and 'mind-reading'.) Machine Learning is a most powerful approach to train the classifier and then use the classifier to discriminate between different cognitive states.

Learning this series of brain data have many challenges, one of which is the extremely sparse noisy data with high dimensional features. This would cause the over-fitting problem for the classifier. Hence it is necessary to apply some feature selection method to make learning tractable and prevent over-fitting due to spurious correlations. The objective of feature selection is three-fold: improving the prediction performance (both sensitivity and selectivity) of the predictors, providing faster and more cost-effective predictors, and providing a better understanding of the underlying process that generated the data. There are a number of generic feature construction/selection/reduction methods, including univariate (voxel-wise) and multivariate statistics; clustering; basic linear transforms of the input variables (PCA/SVD, ICA); more sophisticated linear transforms like spectral transforms (Fourier, Hadamard), wavelet transforms or convolutions of kernels; and applying simple functions to subsets of variables, like products to create monomials[Isabelle & Andre, 2003].

According to univariate voxel selection methods one can select out the voxels that, considered individually, do the best job of discriminating between the conditions of interest

[Haxby et al., 2001; Polyn, S.M. et al., 2005]. Indeed, any univariate statistic used in conventional fMRI analysis can be used for feature selection. The main concern with univariate feature selection methods is that, even with a liberal threshold, it is possible that these methods are discarding voxels that (when taken in aggregate) would have provided useful information about the experimental conditions. We can avoid this problem if we replace univariate feature selection methods with multivariate feature selection methods that evaluate sets of voxels, based on the informativeness of patterns of activity expressed over those voxels. A challenge faced by this approach is that (because of combinatorial explosion issues) the space of voxel sets is much too large to search exhaustively. This issue can be addressed by constraining the search to sets of spatially adjacent voxels [Kriegeskorte et al., 2006], by adding voxels to the set one a time to maximize (a teach step) the multivariate goodness of the current voxel set, or perhaps using genetic algorithms to search through a number of evolutions of randomly initialized sets of voxels – and this is the method we explore here.

Beside testing of the analogic GA we introduced in section 4, we study its performance at selecting features (voxels) without any transformation from the original brain image data as the input for the classifiers to achieve best reconstruction of the data and be most efficient for making predictions.

Data for our example is taken from visual attention experiments carried out in the MR Research Center of the Szentagothai J. Knowledge Center - Semmelweis University. An fMRI scan produces a three-dimensional image related to the human subject's brain activity every 3 seconds. The aim and details of the experiment are beyond the scope of this work; here I explain what is necessary for understanding the role of the genetic algorithm in the analysis. The experiment consists of blocks of trials: 'active' blocks having 5 consecutive trials (3 seconds long) in them or 15 seconds long rest periods. In each active block subjects have to fixate at the centre and watch the same (or very similar) stimuli: coherently moving dots within a circular aperture and a parallel 'RSVP task' – rapid serial visual presentation of different letters – in the centre of the aperture. During 10 letters appeared after each other for a couple of milliseconds, two consecutive coherent dot motion interval are displayed within the 3 seconds of a trial.

In half of the blocks subjects are told to attend the RSVP task and indicate if letter X appears between the rapidly changing letters. In the other half they have to select the faster between the two intervals of dot motion ignoring the letters.

After normalization (percent signal change) the data was segmented and partitioned voxel-wise in two groups according to the tasks and aligned to the onset of the blocks. BOLD response amplitudes 6 and 9sec after stimulus onset was selected and averaged for each trial in each voxel. From this point BOLD response of the trials can be characterized by a single vector: the length of the vector (dimensionality) is defined by the total number of voxels (see Figure 70).

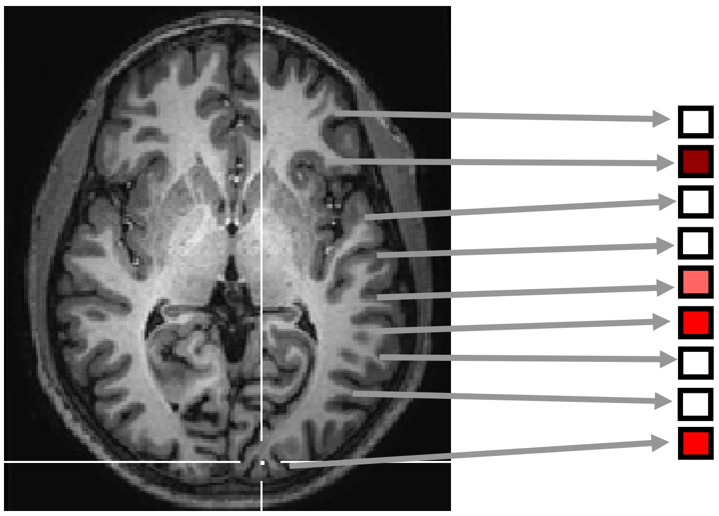


Figure 70 Feature vector built on activation pattern of voxels from different parts of the brain

For this analysis only voxels belonging to the primary visual area (V1) were selected based on an independent 'retinotopic mapping' experiment.

Each of the 6 subject's V1 visual areas contain approximately 500 voxels: thus initially the trials labeled either as coherent motion or rsvp trials are characterized by 500 elements feature vectors. The plan is that a linear classifier – popular choice are logistic regression and linear discriminant analysis (LDA)- should be trained on a randomly selected training set of BOLD activity vectors to discriminate between the two types of trials. The performance of the trained classifier is assessed on a test set of trials not taking part in the teaching process at all. The whole process is repeated over selecting the test set from the data via leave-one-out approach.

Before testing voxels are selected (the dimension of the vectors are reduced) by different types of selection for comparison purpose.

The method based on univariate statistics handled the voxels individually, and sorted them according to there ANOVA statistics reflecting the discrimination power between the two conditions within the training set. LDA discrimination performance was tested with different number of voxels: only with the first, the first two, first three etc. up to the first 50 ANOVA ordered voxels. In that way the best group was selected and the LDA was evaluated on the test set.

Inherent and seamless representation of selected and ignored voxels by binary genetic algorithms enables their simple and effective integration into this framework. We compared a classical implementation of GA (GAOT MATLAB toolbox [Houck et al., 1995]) to its analogic counterpart. After random population initialization – defining random voxel sets – every set is evaluated by LDA. Thus the GA fitness function takes the performance of the classification accuracy of LDA with the given voxel set on the training trials. After 200 evolution iterations the best individual -voxel group- is selected and the LDA classifies the test set after teaching on the trainng set. Table 8 summarizes the results of the analysis on the data of 6 subjects.

	Performance on training set(%) ± std deviation	**Performance on test set**	**Average number of selected voxels**
ANOVA	**85± 9**	**68± 9**	**40**
GA	**88 ± 4**	**69± 9**	**31**
Analogic GA	**89 ± 4**	**70± 9**	**31**

Table 8 Comparison of different voxel selection methods

Our 'analogic' GA performs significantly better than the univariate approach both on the training (t=8,83, p<0,001) and on the test set (t=5,41, p=0,002). Although the difference is small (2-4 percent), so within this context its actual application value is doubtful, this clearly shows that the proposed genetic algorithm do not stuck in local minima with lower performance than the solutions of the univariate method. It improves the accuracy for LDA and Logistic Regression Classifier, and it produced better accuracy as the other classical GA on the training set (t=6,83, p=0,001): actually the performance on training is the better indicator of the optimization power from a mathematical point of view; the test performance highly depends on the quality of the data and the generalization power of the classifier.

7 References

[The ALADDIN System, http://www.analogic-computers.com/

Abeles M. (1983). The quantification and graphic display of correlations among three spike trains. IEEE Trans. Biomed. Eng., 30, 236-239.

Abeles M. (1991). Corticonics: Neural circuits of the cerebral cortex. Cambridge: Cambridge University Press.

Abeles M., Bergman H., Margalit E. & Vaadia E. (1993a). Spatiotemporal firing patterns in the frontal cortex of behaving monkeys. J. Neurophysiol., 70(4), 1629-1638.

Abeles M., Vaadia E., Prut Y., Haalman I., & Slovin H. (1993b). Dynamics of neuronal interactions in the frontal cortex of behaving monkeys. Conc. Neurosci., 4(2),131-158.

Aertsen A., & Arndt M. (1993c). Response synchronization in the visual cortex. Curr. Op. Neurobiol., 3, 586-594.

ALADDIN: Analogic CNN Visual Microprocessor Application Environment, Analogic Computers Ltd. Homepage: http://www.analogic-computers.com

Bályа D. and Gál V. (2006) Analogic Implementation of the Genetic Algorithm in Proc. of the 10th IEEE International Workshop on Cellular Neural Networks and their Applications (CNNA 2006), Istambul

Barlow H. (1992). Single cells versus neuronal assemblies. In A. Aertsen & V. Braitenberg (Eds.), Information processing in the cortex (pp.169-173). Berlin: Springer-Verlag.

Barlow H. B. (1972). Single units and sensation: A neuron doctrine for perceptual psychology? Perception, 1, 371-394.

Blanchard M., Rind F.C. & Verschure P.F.M.J. (2000) Collision avoidance using a model of the locust LGMD neuron. Robotics and Autonomous Systems, 30, 17-38.

Brown E.N., Kass R.E., Mitra P. P. (2004). Multiple neural spike train data analysis: state-of-the-art and future challenges. Nat Neurosci. 7(5):456-61.

Chua L. O. and Yang L. (1988) Cellular Neural Networks: Theory, IEEE Trans. on Circuits and Systems, Vol. 35, pp. 1257-1272.

Chua L. O., & Roska T. (1993). The CNN Paradigm. IEEE Trans. on Circuits and Systems, Vol. 40, pp.147-156.

Chua L. and Roska T. (2001). Cellular Neural Networks and Visual Computing - Foundations and applications. Cambridge University Press, ISBN: 0521652472.

Cox D. R., & Isham V. (1980). Point processes. London: Chapman and Hall.

Crounse K.R., Yang T., Chua L.O. (1996). Pseudo-Random Sequence Generation Using the CNN Universal Machine with Application to Cryptography. Proceedings of IEEE Int.Workshop on Cellular Neural Networks and Their Applications,(CNNA '96),pp.133-138,Sevilla.

Dayhoff, J. E., & Gerstein, G. L. (1983). Favored patterns in spike trains. 11. Application. J. Neurophysiol., 49(6),1349-1363.

Diesmann, M., Gewaltig, M.O. & Aertsen, A. (1999). Conditions for stable propagation of synchronous spiking in cortical neural networks. Nature, 402, 529-533.

Edwards D.H. (1982). The cockroach DCMD neuron. II, Dynamics of response habituation and convergence of spectral inputs. Journal of Experimental Biology. 99, 91-107.

Engel, A. K., Körig P., Schilleri T. B. & Singer W. (1992). Temporal coding in the visual cortex: New vistas on integration in the nervous system. TINS, 15(6), 218-226.

Espejo S., R. Dominguez-Castro, Linan G. & Rodriguez-Vazquez A. (1998) A 64x64 CNN Universal Chip with Analog and Digital I/O. Proc. 5th IEEE Int. Conf. on Elec., Circ. and Sys., 203-206 (Lisboa).

Feller, W. (1968). An introduction to probability theory and its applications. (Vol. 1, 3rd ed.). New York: Wiley.

Gál V., Hámori J., Roska T., Bálya D., Borostyánkői Zs., Brendel M., Lotz K., Négyessy L., Orzó L., Petrás I., Rekeczky Cs, Takács J., Venetiáner P., Vidnyánszky Z. & Zarándy Á. (2004) Receptive Field Atlas and Related CNN Models. International Journal of Bifurcation and Chaos (IJBC), Vol. 14(2),pp. 551–584.

Gál V., Roska T. (2000). Collision Prediction via the CNN Universal Machine. Int. Workshop on Cellular Neural Networks and Their Applications (CNNA 2000), Catania, Italy, pp 105-110.

Gál V., Grün S. & Tetzlaff R. (2002) Analyzing Multidimensional Neural Activity via CNN-UM. Proc. of the 7th IEEE Int. Workshop on CNN and their Applications

Gál V., Grün S. & Tetzlaff R. (2003) Analysis of Multidimensional Neural Activity via CNN-UM. International Journal of Neural Systems, Vol. 13, No. 6 pp. 479-487

Gerstein G.L. and Kirkland K.L. (2001). Neural assemblies: technical issues, analysis, and modeling. Neural Networks, 14 ,589

Gerstein G. L., Bedenbaugh P. & Aertsen A. (1989). Neuronal assemblies. IEEE Trans. Biomed. Eng., 36,4-14.

Gobovic D. & Zaghloul M. E. (1994). Analog Cellular Neural Network with Application to Partial Differential Equations. Proceedings of International Symposium on Circuits and Systems ISCAS '94, Vol. 6, pp. 359-362, London.

Goldberg David E. (1989). Genetic Algorithms in Search, Optimization and Machine Learning. Kluwer Academic Publishers, Boston, MA.

Grün S., M. Diesmann and A. Aertsen (2002). Unitary events in multiple single-neuron spiking activity: II. Nonstationary data, Neural Comput., 14(1), 81-119.

Grün S., M. Diesmann and A. Aertsen(2002). Unitary events in multiple single-neuron spiking activity: I. Detection and significance. Neural Comput., 14(1), 43-80

Grün S., Diesmann M., Grammorrt F., Riehle A., & Aertsen A. (1999). Detecting unitary events without discretization of time. J. Neurosc. Methods., 94, 67-79.

Gutig R, Aertsen A, Rotter S. (2002). Statistical significance of coincident spikes: count-based versus rate-based statistics. Neural Comput. 14(1):121-53.

Harris R.A., O'Carroll D.C. & Laughlin S.B., (1999). Adaptation and the temporal delay filter of fly motion detectors. Vision Research. 39, 2603-2613.

Harrison R.R. (2000). An analog VLSI motion sensor based on the fly visual system. Ph.D. thesis: California Institute of Technology, California, USA.

Haxby, J.V., Gobbini M.I., Furey M.L., Ishai A., SchoutenJ.L. & Pietrini P. (2001) Distributed and overlapping representationsof faces and objects in ventral temporal cortex. Science 293, 2425–2429

Hebb, D. O. (1949). Organization of behavior. A neurophysiological theory. New York: Wiley.

Houck C., Joines J., Kay M. (1995). A Genetic Algorithm for Function Optimization: A Matlab Implementation. NCSU-IE TR 95-09, 1995.

Isabelle G. Andre E. (2003). An Introduction to Variable and Feature Selection. Journal of Machine Learning Research 3, 1157-1182.

Judge S.J. & Rind F.C. (1997). The locust DCMD, a movement detecting neuron tightly tuned to collision trajectories. Journal of Experimental Biology. 200, 2209-2216.

Kimia B. B. & Siddiqi K. (1996), "Geometric Heat Equation and Nonlinear Diffusion of Shapes and Images", Computer Vision and Image Understanding, 64(3), 305-322.

Kozek T. & Roska T. (1996). A double time-scale CNN for solving 2-D Navier-Stokes equations. International Journal of Circuit Theory and Applications, Vol. 24, pp. 49-56.

Kozek T., Chua L. O., Roska T., Wolf D., Tetzlaff R., Puffer F. & Lotz K. (1995). Simulating Nonlinear Waves and Partial Differential Equations via CNN - Part II: Typical Examples", IEEE Trans. on Circuits and Systems, Vol. 42, No. 10, pp. 816-821.

Kriegeskorte, N., Goebel R. & Bandettini P. (2006) Information-based functional brain mapping. Proc. Natl. Acad. Sci. U. S. A. 103, 3863–3868.

Laurent G. and Gabbiani F. (1998). Collision-avoidance: nature's many solutions. Nature Neuroscience, Vol.1, pp. 261-263.

Legendy, C. R. (1975). Three principles of brain function and structure. Intern. J. Neurosci., 6(5), 237-254.

Legendy, C.R., & Salcman, M. (1985). Bursts and recurrences of bursts in the spike trains of spontaneously active striate cortex neurons. J. Neurophysiol., 53(4), 926-939.

Liñán G., Espejo S., Dominguez-Castro R. & Rodríguez-Vázquez A. (2002). ACE4k: an Analog I/O 64x64 Visual Microprocessor Chip with 7-bit Analog Accuracy, International Journal of Circuit Theory and Applications, Vol.30, pp.89-116.

Martignon L., Deco G., Laskey K., Diamond M., Freiwald W. & Vaadia E. (2000). Neural coding: Higher-order temporal patterns in the neurostatistics of cell assemblies. Neural Comp., 12, 2621-2653.

Martignon L, Laskey K., Deco G., & Vaadia E. (1997). Learning exact patterns of quasi-synchronization among spiking neurons from data on multi-unit recordings. In M. Jordan & M. Mozer (Eds.), Advances in information processing systems, 9 (pp. 145-151). Cambridge, MA: MIT Press.

Martignon L., von Hasseln H., Grün S., Aertsen A., & Palm G. (1995). Detecting higher-order interactions among the spiking events in a group of neurons. Biol. Cybern., 73,69-81.

Mitchell M., (1996). An Introduction to Genetic Algorithms. MIT Press, Cambridge, MA.

Nadasdy Z., Hirase H., Czurko A., Csicsvari J., & Buzsaki G. (1999). Replay and time compression of recurring spike sequences in the hippocampus. J. Neurosci., 19(21), 9497-9507.

Palm, G. (1981). Evidence, information and surprise. Biol. Cybern., 42, 57-68. Palm, G. (1990). Cell assemblies as a guidline for brain reseach. Conc. Neurosci., 1,133-148.

Palm G., Aertsen A., & Gerstein G. L. (1988). On the significance of correlations among neuronal spike trains. Biol. Cybern., 59,1-11.

Palm, G. (1990). Cell assemblies as a guideline for brain reseach. Conc. Neurosci. 1, 133-148.

Pauluis Q. & Baker, S. N. (2000). An accurate measure of the instantaneous discharge probability, with application to unitary joint-event analysis. Neural Comp., 12(3), 647-669.

Pinter R.B, (1983) Product term nonlinear lateral inhibition enhances visual selectivity for small objects or edges. Journal of Theoretical Biology. 100, 525-531.

Pinter, R.B., (1984). Adaptation of receptive field spatial organisation via multiplicative lateral inhibition. Journal of Theoretical Biology. 110, 435-444.

Polyn S.M., Natu V.S., Cohen J.D. & Norman K.A. (2005) Category-specific cortical activity precedes recall during memory search. Science 310, 1963–1966

Riehle, A., Grün S, Diesmann M., & Aertsen A. (1997). Spike synchronization and rate modulation differentially involved in motor cortical function. Science, 278,1950-1953.

Rind F.C. & Bramwell D.I. (1996). Neural network based on the input organisation of an identified neuron signalling impending collision. Journal of Neurophysiology, 75, 967-984.

Rind F. C. & Simmons P.J. (1992). Orthopteran DCMD neuron: a reevaluation of responses to moving objects. I. Selective approaches to approaching objects. J. Neurophysiol. 68: 1654–1666.

Rind F.C., Simmons P.,J. (1999). Seeing what is coming: building collision-sensitive neurons. Trends in Neuroscience, Vol. 22, pp. 215-220.

Roelfsema P., Engel A., Körig P. & Singer W. (1996). The role of neuronal synchronization in response selection: A biologically plausible theory of structured representations in the visual cortex. J. Cogn. Neurosci., 8(6), 603¬625.

Roska T. & Chua L. O.(1993). The CNN Universal Machine: an Analogic Array Computer. IEEE Trans. on Circuits and Systems, Vol. 40, pp. 163-173.

Roska T., Wolf D., Kozek T., Tetzlaff R. & Chua L. O. (1993) Solving Partial Differential Equations by CNN. in Proceedings of European Conference on Circuit Theory and Design ECCTD '93, pp. 1477-1482, Davos.

Roska T., Zarándy Á., Zöld S., Földesy P. & Szolgay P. (1999). The Computational Infrastructure of Analogic CNN Computing - Part I: The CNN-UM Chip Prototyping System. IEEE Trans. on Circuits and Systems I: Vol. 46, No.2, pp. 261-268.

Roska T. Kék L., Nemes L. & Zarándy Á. (1999). CNN Software Library (Templates, subroutines, and Algorithms) Version 8.1 , Computer and Automation Institute of the Hungarian Academy of Sciences, Budapest

Roska T., Chua L. O., Wolf D., Kozek T., Tetzlaff R., & Puffer F.(1995). Simulating Nonlinear Waves and Partial Differential Equations via CNN - Part I: Basic Techniques. IEEE Trans. on Circuits and Systems, Vol. 42, No. 10, pp. 807-815.

Roy A., Steinmetz P. & Niebur E. (2000). Rate limitations of unitary event analysis. Neural Comp., 12, 2063-2082.

Schiff W. & Detwiler M.L. (1979). Information used in judging impending collision. Perception, Vol.8, pp. 647-658.

Shadlen M. N. & Newsome W. T. (1998). The variable discharge of cortical neurons: Implications for connectivity, computation, and information coding. J. Neurosci., 18(10),3870-3896.

Shi B. (1996). Second order CNN Arrays for Estimation of Time-to-Contact. Proc. CNNA-96, pp. 427-432., Seville, Spain.

Simmons P. J. & Rind F. C. (1992) Orthopteran DCMD neuron: a reevaluation of the of responses to moving objects. II. Critical cues for detecting approaching DCMD. objects. J. Neurophysiol. 68: 1667–1682.

Singer W. (1993). Synchronization of cortical activity and its putative role in information processing and learning. Annu. Rev. Physiol., 55,349-374.

Singer W. (1999). Neural synchrony: A versatile code for the definition of relations. Neuron, 24,49-65.

Singer, W., & Gray, C. (1995). Visual feature integration and the temporal correlation hypothesis. Annu. Rev. of Neurosci, 18, 555-586.

Singer W., Engel A.K., Kreiter A. K., Munk M. H. J., Neuenschwander S. & Roelfsema P. R. (1997). Neuronal assemblies: necessity, signature and detectability. Trends in Cognitive Sciences, 1(7), 252-261.

Softky W. R. & Koch C. (1993). The highly irregular firing of cortical cells is inconsistent with temporal integration of random EPSPS. T Neurosci., 13, 334-350.

Sun H. & Frost B.J. (1998). Computation of different optical variables of looming objects in pigeon nucleus rotundus neurons. Nature Neuroscience Vol.1, pp. 296-303.

Szatmári I., Földesy P., Rekeczky Cs. & Zarándy Á. (2002). Image Processing Library for the Aladdin Visual Computer. Pro-ceedings of the CNNA-2002 Frankfurt, Germany.

Szatmári, I., Zarándy Á., Földesy P. and Kék L. (2000). An analogic CNN engine board with the 64x64 analog I/O CNN-UM Chip," Proc. IEEE International Symposium on Circuits and Systems, 2, 124-127

Vaadia E., Haalman I, Abeles M., Bergman H., Pict Y, Slovin H., & Aertsen A. (1995). Dynamics of neuronal interactions in monkey cortex in relation to behavioural events. Nature, 373(6514), 515-518.

Vose M.D. (1999), The Simple Genetic Algorithm: Foundations and Theory, MIT Press, Cambridge, MA.

Witkin A. (1983). Scale-Space Filtering. Int. Joint Conference Artificial Intelligence, Karlsruhe, West Germany, pp. 1019-1021.

Wolfram S.(1994). Cellular Automata and Complexity. Westview press, Member of the Perseus Books Group.

Zarándy A, Rekeczky Cs, Földesy P. & Szatmári I. (2003) The New Framework of Applications - The Aladdin System. J. of Circuits, Systems, and Computers (JCSC), Vol 12(6)