Exploration of spatial-temporal dynamic phenomena in a 32×32-cells stored program 2-layer CNN Universal Machine Chip Prototype

István Petrás¹, Csaba Rekeczky¹, Tamás Roska¹, Ricardo Carmona², Francisco Jiménez-Garrido², Angel Rodríguez-Vázquez²

¹ Analogical and Neural Computing Laboratory, Computer and Automation Research Institute, Hungarian Academy of Sciences, Kende u.11, Budapest, 1111 - Hungary
² Instituto de Microelectrónica de Sevilla-CNM-CSIC
Avda. Reina Mercedes s/n, 41012 Sevilla (SPAIN)
Tel. +34955056666 Fax. +34955056686

Abstract. This paper describes a full-custom mixed-signal chip that embeds digitally programmable analog parallel processing and distributed image memory on a common silicon substrate. The chip was designed and fabricated in a standard 0.5µm CMOS technology and contains approximately 500,000 transistors. It consists of 1024 processing units arranged into a 32×32 grid. Each processing element contains two coupled CNN cores, thus, constituting two parallel layers of 32×32 nodes. The functional features of the chip are in accordance with the 2nd Order Complex Cell CNN-UM architecture. It is composed of two CNN layers with programmable inter- and intra-layer connections between cells. Other features are: cellular, spatial-invariant array architecture; randomly selectable memory of instructions; random storage and retrieval of intermediate images. The chip is capable of completing algorithmic image processing tasks controlled by the user-selected stored instructions. The internal analog circuitry is designed to operate with 7-bits equivalent accuracy. The physical implementation of a CNN containing second order cells allows real time experiments of complex dynamics and active wave phenomena. Such well-known phenomena from the reaction-diffusion equations are traveling waves, autowaves, and spiral-waves. All of these active waves are demonstrated on-chip. Moreover this chip was specifically designed to be suitable for the computation of biologically inspired retina models. These computational experiments have been carried out in a development environment designed for testing and programming the analogic (analog-and-logic) programmable array processors.

1 Introduction

Various phenomena of the world around us are the result of local interactions of particles. This is true for both micro and macro world, for molecules, cells and even for members of a population. Two large classes of such infinitely many phenomena are the pattern formation in biology (e.g. spots of animals or plants) or in chemistry and active wave propagation (e.g. action potential in the nervous system, propagation of cardiac muscle excitation, blood coagulation process). Fig. 1 shows examples of pattern formation and active wave phenomena.

The expression “active wave” means that the wave propagation occurs in an “energetically active” medium, i.e. the energy conservation law does not hold because the media can inject energy into the system, therefore the amplitude and the waveform are preserved during the propagation. These are the most important differences when compared with classical waves. To make any reasonable complex pattern formation, we need the local activity of the cells [34].

Up to now the typical way to carry out experiments related to these phenomena was either laboratory work (e.g. with chemicals) or through time-consuming computer simulation. Other option is to build dedicated hardware circuitry. Although it is fast, its disadvantage is the lack of easy programmability.

The Complex Cell CNN programmable array computer [5][11] is an extension of the CNN Universal Machine [1]-[4]. The multilayer or higher order elementary cells are or-
ganized into a 32×32 square grid. Each cell has second order dynamics and local inter-
connections to its neighbors. This structure is especially suitable for computing a certain
set of ordinal differential equations. Simple PDEs can be transformed so that they can be
easily programmed on the array computer. With the programmable Complex Cell CNN
Universal Machine scientists are provided with a unique opportunity to study active-wave
propagation and other reaction-diffusion differential equation based phenomena in a
programmable manner. Exploiting the inherent programmability of the CNN architec-
ture, complex wave-computing analogic algorithms can be designed.

In this paper we show how to transform the mathematical model of the phenomena
outlined above into a form suitable for parallel processing on the Complex Cell CNN
chip and we present the first measurement results.

Fig. 1: spiral- and autowaves and patterns measured on the CACE1k chip (image size: 32×32)

The following section describes the mathematical model of the core of the complex cell
chip. In Section 3 the prototype chip is introduced. Section 4 describes the relation be-
tween the reaction-diffusion equations and the structure of the complex cell chip. Finally,
Section 5 contains the real-time chip measurements. We should emphasize that no simu-
lation results are presented here. This chip was designed also for computing biologically
inspired retina modeling. The implementation of the main effect of a retina model will be
published in the near future elsewhere.

2 Computational architecture

The architecture of the chip follows the design of the first order CNN-UM chip [10], but
its first order cell core is replaced by a second order one [5]. With this change we are able
to reproduce basic active wave phenomena (See Fig. 1). Fig. 2 shows the functional
architecture of the 2nd order, two layer CACE1k core.

Fig. 2: Functional diagram of the CACE1k chip

The evolution law of the complex cell CNN is the following differential equation system:
\[
\begin{align*}
\tau_1 \frac{dx_{1,ij}(t)}{dt} &= -g(x_{1,ij}(t)) + \sum_{k,l \in N_i} A_{1,kl} x_{1,kl}(t) + a_{12} x_{2,ij}(t) + b_{1} u_{ij} + z_{1,ij} \\
\tau_2 \frac{dx_{2,ij}(t)}{dt} &= -g(x_{2,ij}(t)) + \sum_{k,l \in N_i} A_{2,kl} x_{2,kl}(t) + a_{21} x_{1,ij}(t) + b_{2} u_{ij} + z_{2,ij} 
\end{align*}
\]

(2.1)

Where:

\[
g(x_{n,ij}) = \lim_{m \to \infty} \begin{cases} 
mx_{n,ij} & \text{if } x_{n,ij} > 1 \\
x_{n,ij} & \text{if } |x_{n,ij}| \leq 1 \\
-mx_{n,ij} & \text{if } x_{n,ij} < -1 
\end{cases} 
\]

(2.2)

Function \(g(\cdot)\) regulates the state so that it stays within the +1..-1 interval. Variables \(u_1, u_2\) are the independent inputs, \(b_1, b_2\) are their weight factors respectively; \(z_{1,ij}, z_{2,ij}\) are space variant bias maps. Variables \(x_1, x_2\) denote the state variables of the layers. Each \(x_{ij}\) corresponds to one cell; it is one pixel of a M by N image if we consider the system’s output as a picture. \(A_1, A_2\) are the weights of the intra-layer couplings, \(a_1, a_2\) are the inter-layer weights. Equation (2.1) utilizes the so-called full-range model (FSR)[7], where the voltage of the state variable is always the same as the output.

3 Chip description

3.1 Architecture of the chip and basic processor structure

The prototype chip consists of an analog programmable array processor of 32×32 identical cells (Fig. 3), surrounded by the boundary conditions for the CNN dynamics. There is also an I/O interface, a timing and control unit and a program memory. The interface consists of a serializing-deserializing analog multiplexor. The program memory is composed of the analog program memory, and the logic program memory and the switch configuration registers. The analog instructions and reference signals need to be transmitted to every cell in the network in the form of analog voltages. Thus, a bank of D/A converters interfaces the analog program memory with the processing array. Distributing analog references across large distances within a chip is not a trivial task. Apart from the problems caused by electromagnetic interference, voltage drops in long metal lines carrying currents can be quite noticeable. Thus, in order to avoid this signal buffering and low-resistance paths must be provided. Finally, the timing unit is made of an internal clock/counter and a set of FSMs that generate the internal signals which control the processes of images up/downloading and program memory accesses.
Every elementary processor of the Complex Cell CNN programmable array chip includes two coupled continuous-time CNN cores (Fig. 4(a)) belonging to each of the two different layers of the network. The synaptic connections between processing elements of the same or different layer are represented by arrows in the diagram. The basic processor also contains a programmable local logic unit (LLU) and local analog and logic memories (LAMs and LLMs) to store intermediate results. All the blocks in the cell communicate via an intra-cell data bus, which is multiplexed to the array I/O interface. Control bits and switch configuration are passed to the cell directly from the global programming unit.

The internal structure of each CNN core is depicted in Fig. 4(b). Each core receives contributions from the rest of the processing nodes in the neighborhood which are summed and integrated in the state capacitor. The two layers differ in that the first layer has a scalable time constant, controlled by the appropriate binary code, while the second layer has a fixed time constant. The evolution of the state variable is also driven by self-feedback and by the feed forward action of the stored input and bias patterns. There is a voltage
limiter for implementing the FSR CNN model [7]. The state variable is transmitted in voltage form to the synaptic blocks, in the periphery of the cell, where weighted contributions to the neighbors’ are generated. There is also a current memory that will be employed for cancellation of the offset of the synaptic blocks. Initialization of the state, input and/or bias voltages is done through a mesh of multiplexing analog switches that connect to the cell’s internal data bus.

3.2 Prototype chip data

The prototype chip has been designed and fabricated using 0.5µm single-poly triple-metal CMOS technology. Its dimensions are 9.27 x 8.45 mm² (Fig. 5). The cell density achieved is 29.24cells/mm². The time constant of the layers is around 100ns (unscaled). The programmable dynamics of the chip permit the observation of different phenomena of the type of propagation of waves, pattern generation, etc (See Section 5 for examples). Table 1 summarizes the most relevant data of the prototype chip.

<table>
<thead>
<tr>
<th>Technology</th>
<th>0.5 µm CMOS 1-P 3-M</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of cells</td>
<td>32 x 32</td>
</tr>
<tr>
<td>Die area</td>
<td>9.27 x 8.45 mm²</td>
</tr>
<tr>
<td>Die area (w/o pads)</td>
<td>8.77 x 7.94 mm²</td>
</tr>
<tr>
<td>Array area</td>
<td>5.98 x 5.93 mm²</td>
</tr>
<tr>
<td>Package</td>
<td>ceramic PGA – 100</td>
</tr>
<tr>
<td>Power supply voltage</td>
<td>3.3V</td>
</tr>
<tr>
<td>Logic “0” / Logic “1”</td>
<td>0V / 3.3V</td>
</tr>
<tr>
<td>Accuracy on the weights</td>
<td>8b</td>
</tr>
<tr>
<td>Image samples resolution</td>
<td>7-8b</td>
</tr>
<tr>
<td>I/O rates</td>
<td>10Ms/s</td>
</tr>
<tr>
<td>CNN time constant</td>
<td>below 100ns</td>
</tr>
</tbody>
</table>

**Table 1**: Characteristic parameters of the CACE1k prototype chip

![Microphotograph of the prototype chip](image)
4 Complex cell CNN-UM chip as a “PDE solver” – typical waves

4.1 General PDE formulation

Let us consider the following PDE formulation of a coupled vector valued nonlinear reaction-diffusion system (examples for nonlinear PDEs could be found in [26]-[32]):

\[
\frac{d}{dt} \phi_i(x,t) - \text{div}(c_i(x,t) \text{grad}(\phi_i(x,t))) = \alpha_i(\phi_i(x,t)) + \beta_i(\phi_i(x,t)) + \gamma_i(\phi_i(x,t))
\]

\[
\frac{d}{dt} \phi_j(x,t) - \text{div}(c_j(x,t) \text{grad}(\phi_j(x,t))) = \alpha_j(\phi_j(x,t)) + \beta_j(\phi_j(x,t)) + \gamma_j(\phi_j(x,t))
\]

(4.1)

where \([\phi_i(x,t) \phi_j(x,t)]\) can be interpreted as the time evolution of a vector valued image intensity (\([\phi_0(x) \phi_0(x)]\) are the initial images), the vector \(x\) represents the spatial coordinates, the time variable \(t\) can also be interpreted as the scaling parameter and \(c\) is the conductance parameter in the diffusion term. The right hand side of the equations consists of three reaction terms: (i) \(\alpha(.)\) is the “self-reacting term”; (ii) \(\beta(.)\) is a “spatial constraint” (calculated from the initial data); and (iii) \(\gamma(.)\) is the “cross-coupling” term. Some proposed functions for \(g(.)\) are:

\[
g_1 = \exp\left[-\left(\text{grad}(I(x,t)) \cdot K^2\right)\right], \quad K > 0
\]

\[
g_2 = \left(1 + \left(\text{grad}(I(x,t)) \cdot K^2\right)^{1+\alpha}\right)^{-1}, \quad \alpha > 0, \quad K > 0
\]

(4.2)

Decoupling the two equations \((\gamma_1 = \gamma_2 = 0)\) and choosing \(\alpha(\xi) = -\xi\) the PDE formulation is given in the form of the so-called “biased” nonlinear anisotropic diffusion equation [Nordström [27]] an extended version of the Perona-Malik formulation [26]. Gérig et. al [29] first proposed the application of coupled nonlinear diffusion systems for vector-valued image processing that has been further studied by many others [32]. For our purposes – motivated by silicon implementation – we focus on a simplified version of (4.1) fixing the diffusion parameter to a constant value.

4.2 Spatial discretisation

In two spatial dimensions and assuming \(c = \text{const.}\) (4.1) and (4.2) reduces to:

\[
\frac{d}{dt} \phi_i(x,y,t) - c_1 \text{div}(\text{grad}(\phi_i(x,y,t))) = \alpha_i(\phi_i(x,y,t)) + \beta_i(\phi_i(x,y,t)) + \gamma_i(\phi_i(x,y,t))
\]

\[
\frac{d}{dt} \phi_j(x,y,t) - c_2 \text{div}(\text{grad}(\phi_j(x,y,t))) = \alpha_j(\phi_j(x,y,t)) + \beta_j(\phi_j(x,y,t)) + \gamma_j(\phi_j(x,y,t))
\]

(4.3)

After spatial discretisation using the finite difference approach one obtains:

\[
\frac{d\phi_i(x,y,t)}{dt} = \gamma_i(\phi_i(x,y,t)) + \alpha_i(\phi_i(x,y,t)) - c_i\phi_i(x,y,t) + \frac{c_i}{4}(\phi_{i-1}(t) + \phi_{i+1}(t) + \phi_{i,j}(t) + \phi_{i,j+1}(t)) + \beta_i(\phi_{i,j})
\]

\[
\frac{d\phi_j(x,y,t)}{dt} = \gamma_j(\phi_j(x,y,t)) + \alpha_j(\phi_j(x,y,t)) - c_j\phi_j(x,y,t) + \frac{c_j}{4}(\phi_{j-1}(t) + \phi_{j+1}(t) + \phi_{i,j}(t) + \phi_{i,j+1}(t)) + \beta_j(\phi_{i,j})
\]

(4.4)

Finally, with \(\alpha(.) = \alpha_0\) and \(\beta(.) = \beta_0\) a simplified form (with spatial symmetry and isotropy) of the CNN complex cell equation is derived:
\[
\frac{d\phi_{ij}(t)}{dt} = \gamma \phi_{ij}(t) + (\alpha_i - c_i) \phi_{ij}(t) + \frac{c_i}{4} (\phi_{i-1,j}(t) + \phi_{i+1,j}(t) + \phi_{i,j-1}(t) + \phi_{i,j+1}(t)) + z_{ij},
\]

\[
\frac{d\phi_{2ij}(t)}{dt} = \gamma \phi_{2ij}(t) + (\alpha_i - c_i) \phi_{2ij}(t) + \frac{c_i}{4} (\phi_{2i-1,j}(t) + \phi_{2i+1,j}(t) + \phi_{2i,j-1}(t) + \phi_{2i,j+1}(t)) + z_{2ij},
\]

\[z_{ij} = \beta_i \phi_{2ij} + z_i; \quad z_{2ij} = \beta_i \phi_{22ij} + z_i \]

Making \( \phi \) explicitly depend on a state variable \( \xi \) such as \( \phi = f(\xi) = \text{sigm}(\xi) \) leads a good approximation to the full-range CNN circuit model ([7], \text{sigm}(.)) could be a piecewise linear or monotonic continuous smooth function playing the role of a signal-limiter). The form of the corresponding 2nd order CNN template is as follows:

\[
A_1 = \begin{bmatrix}
0 & c_i/4 & 0 \\
c_i/4 & \alpha_i - c_i & c_i/4 \\
0 & c_i/4 & 0 \\
\end{bmatrix}; \quad A_2 = \gamma I; \quad B_1 = \beta; \quad \tau_1 = 1; \quad z_1 = 0
\]

\[
A_2 = \begin{bmatrix}
0 & c_i/4 & 0 \\
c_i/4 & \alpha_i - c_i & c_i/4 \\
0 & c_i/4 & 0 \\
\end{bmatrix}; \quad A_1 = \gamma I; \quad B_2 = \beta; \quad \tau_2 = 1; \quad z_2 = 0
\]

Analysis: It should be noted that the diagonal terms could also be added resulting in a much better spatial isotropy during the diffusion process. Symmetry and spatial isotropy of these templates are due to the fact that there are no convection terms in the original formulation (4.1)–(4.2). With these modifications the sign and magnitude of the template entries in (4.6) could be changed in any spatial directions. The time constant is also fixed and equal for the two layers. Having two different time variables (say \( t'' \neq t' \)) in (4.1) leads to a "double time-scale" in the discretized system \( t'' / t' = \tau_2 / \tau_1 \neq 1 \) which has practical relevance when looking at various second order models.

4.3 Typical wave classes

There are a number of parameter settings of special interest that lead to very different qualitative behaviors in this symmetric second order system. We have examined and explored primarily the following simple cases (see also Sections 5.2):

(i) trigger-wave generation (diffusion process in "saturation" at various speeds)
\( \alpha_i > 0; \quad \alpha_i - (c_i+1) > 0; \quad \gamma = \gamma > 0; \quad \beta_i = \beta_i = 0; \quad \tau_2 / \tau_1 \neq 1 \)

(ii) traveling-wave, spiral-wave and auto-wave generation (spatially interacting trigger-waves)
\( \alpha_i > 0; \quad \alpha_i - (c_i+1) > 0; \quad \gamma = \gamma > 0; \quad \beta_i = \beta_i = 0; \quad \tau_2 / \tau_1 \neq 1 \)

(iii) retina effects (spatially interacting receptive fields)
\( \alpha_i = \alpha_i > 0; \quad \alpha_i = 0; \quad \gamma = \gamma > 0; \quad \gamma < 0; \quad \beta_i = \beta_i = 0; \quad \tau_2 / \tau_1 \neq 1 \)

5 Measurement and computation examples

In the following we present chip measurement of the different active wave phenomena: trigger wave, traveling wave, autowave, and spatio-temporal edge detection. The movie files of the measurements are available at [33]

5.1 Stored programmability

The CACE1k chip can be programmed using the so-called AMC (ANALOGIC MACHINE CODE) language. This is similar to the assembler language but it contains additional high-level instructions, such as: image loading, frame grabber instructions, etc. and built in im-
age processing functions. In the Appendix is a simple AMC program which computes the snapshots of the autowave example (Subsection 5.2.4). The actual computation is performed in line 40. Before this line is the preparation for the computation: setting up the parameters. After the computation is done, the results are read out and displayed from the chip’s local analog memories.

The elementary program of the chip is the template. It contains the weight factors of the coupling between the cells and weights for the input and the bias map.

\[
\begin{align*}
A_1 &= \begin{bmatrix}
  a_{1,-1} & a_{1,0} & a_{1,1} \\
  a_{0,-1} & a_{0,0} & a_{0,1} \\
  a_{-1,-1} & a_{-1,0} & a_{-1,1}
\end{bmatrix} \\
A_2 &= \begin{bmatrix}
  a_{2,-1} & a_{2,0} & a_{2,1} \\
  a_{1,-1} & a_{1,0} & a_{1,1} \\
  a_{0,-1} & a_{0,0} & a_{0,1}
\end{bmatrix}
\end{align*}
\]  

(5.1)

The operation of the array computer is completely determined by the 25 template values, the initial states and boundary conditions. In (5.1) \(A_1\) and \(A_2\) matrices denotes the weights inter layer connections of the slower and the faster layer respectively. The strength of the influence of the second layer on the first is controlled by \(a_{21}\) and \(a_{12}\) stands for the reverse case. Symbols \(b_1, b_2, z_1\) and \(z_2\) are the weights of the independent inputs and the space variant bias maps. The ratio of the time constants of the two CNN layers are controlled by \(\tau_1, \tau_2\) is fixed. An analogic algorithm is made up of combination of template executions, logic instructions and spatial arithmetic operations. The example shown in the Appendix is a very simple one, it contains only template execution operation.

5.2 Wave phenomenon
This subsection contains the on-chip measurements of the active wave phenomena.

5.2.1 Trigger wave
This example shows a very simple effect: an active (nonlinear) wave in the faster layer initiates a second wave in the slower layer through the positive interlayer coupling. The input is fed to the faster layer. Fig. 6 shows snapshots of the wave evolution.

\[
\begin{align*}
A_1 &= \begin{bmatrix}
  1 & 1 & 1 \\
  1 & 1.5 & 1 \\
  1 & 1 & 1
\end{bmatrix} \\
A_2 &= \begin{bmatrix}
  0.6 & 0.8 & 0.6 \\
  0.8 & -0.6 & 0.8 \\
  0.6 & 0.8 & 0.6
\end{bmatrix}
\end{align*}
\]

\(a_{21} = 3.3\quad a_{12} = 0\)

(5.2)

\(b_1 = 0\quad b_2 = 3\quad z_1 = 0.6\quad z_2 = -1.5\quad \tau_1 : \tau_2 = 1 : 16\)

Fig. 6: Trigger wave propagation in the faster and in the slower layer. The first film shows the output of the faster layer.
5.2.2 Erasure effect

This example is an “enhancement” of the trigger wave example: a negative coupling is introduced from the slower layer back to the faster one. As a result, after a while, the trigger wave on the faster layer is erased by the trigger wave from the slower layer. As in the case of the trigger wave the input is given to the faster layer. Fig. 7 shows the snapshots of the wave evolution on the faster layer and Fig. 8 shows the evolution on the slower layer.

\[
A_1 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1.5 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0.6 & 0.8 & 0.6 \\ 0.8 & -0.6 & 0.8 \\ 0.6 & 0.8 & 0.6 \end{bmatrix}, \quad a_{11} = 3.3, \quad a_{12} = -5.4
\]

\( b_1 = 0, \quad b_2 = 3, \quad z_1 = 0.6, \quad z_2 = -2.7, \quad \tau_1 : \tau_2 = 16 : 1 \) (5.3)

Fig. 7: Retina like wide-field erasure effect, faster layer.

Fig. 8: Retina like wide-field erasure effect, slower layer.

5.2.3 Traveling wave

The characteristic of this wave is that a single wave front travels across the active medium. The autowave is simply formed from the trigger wave template by adding a negative intra-layer coupling from the slower to the faster layer. In Fig. 9 observe the annihilation property of this wave type.
5.2.4 Autowave
The autowaves possess all of the properties of the traveling waves; the difference is that in this case multiple wave fronts are generated. See Fig. 10.

\[
\begin{align*}
A_1 &= \begin{bmatrix}
1 & 1 & 1 \\
1 & 1.5 & 1 \\
1 & 1 & 1
\end{bmatrix},
A_2 &= \begin{bmatrix}
0.6 & 0.8 & 0.6 \\
0.8 & -0.6 & 0.8 \\
0.6 & 0.8 & 0.6
\end{bmatrix},

a_{21} &= 4.5, \quad a_{12} &= -4.2
\end{align*}
\]

\[
\begin{align*}
b_1 &= 0, \quad b_2 &= 5 \\
z_1 &= 0.6, \quad z_2 &= -2.7
\end{align*}
\]

\[
\tau_1 : \tau_2 = 16 : 1
\]

Fig. 10: Autowave propagation CACE1k chip.

5.2.5 Spiral wave
This wave type received its name from the spiral like autowave structure. Fig. 11 shows measurement results. The annihilation of two wave fronts occurs in this case as well.
\begin{equation}
A_1 = \begin{bmatrix}
0.6 & 0.8 & 0.6 \\
0.8 & -0.3 & 0.8 \\
0.6 & 0.8 & 0.6 \\
\end{bmatrix} \quad A_2 = \begin{bmatrix}
0.6 & 0.8 & 0.6 \\
0.8 & -0.9 & 0.8 \\
0.6 & 0.8 & 0.6 \\
\end{bmatrix} \quad a_{31} = 6 \quad a_{12} = -3
\end{equation}

\begin{equation}
b_1 = 0 \quad b_2 = 3.9 \quad z_1 = -4.5 \quad z_2 = -3.6 \quad \tau_1 : \tau_2 = 8:1
\end{equation}

Fig. 11: Spiral wave formation in the CACE1k chip.

5.2.6 Spatio-temporal edge detection

The basic idea of edge detection in the image processing is to somehow extract or enhance the high frequency components of the image. Here this is done with two layers. The slower layer, which gets its input from the faster one, computes the diffusion. The result is subtracted from the faster layer. As a result we get the edge image of the original image. The input was fed to the faster layer. See snapshots of the transient in Fig. 12.

\begin{equation}
A_1 = \begin{bmatrix}
0.5 & 0.8 & 0.5 \\
0.8 & -3.6 & 0.8 \\
0.5 & 0.8 & 0.5 \\
\end{bmatrix} \quad A_2 = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
\end{bmatrix} \quad a_{31} = 4.5 \quad a_{12} = -5.7
\end{equation}

\begin{equation}
b_1 = 0 \quad b_2 = 3 \quad z_1 = -9 \quad z_2 = 0 \quad \tau_1 : \tau_2 = 6:1
\end{equation}

Fig. 12: Edge detection with second order dynamics on the CACE1k chip.

Acknowledgement

This project was supported by: MTA-SZTAKI - Hungarian Academy of Sciences, Spanish CCICYT (SIVA-2, TICC99-0826), ESPIRIT V (DICTAM-II, IST-1999-19007), ONR/NICOP (N00014-00-1-0429).
Appendix

**Typical AMC Program Example**

```plaintext
; Autowave Computing AMC Program for the CACE1x Complex Cell Chip

; Initialize Cycle Variable
mov.gam_f.gam_f 10 gamf1

; Load Cloning Templates
host.load.tem autowave.tem tem33 1
host.load.tem autowave.tem tem34 2

; Load Image
host.load.pic autowave.bmp lam5

Loop:

; Fill Up the Necessary Memories
ar.layer.param 1 fill_bias -1
ar.layer.param 2 fill_bias -1
ar.layer.param 1 fill_state 0
ar.layer.param 2 fill_state 0

; Set the Time Constant Ratio of the Two Layers (1..16)
ar.layer.param 1 layer_c 11

; Set Boundary Conditions for the Two Layers
ar.layer.param 1 layer_boundary zeroflux
ar.layer.param 2 layer_boundary zeroflux

; Assign the Local Analog Memories (LAM) to the Inner Memories
ar.layer.image 1 img_input LAM1
ar.layer.image 2 img_input LAM1
ar.layer.image 1 img_output LAM3
ar.layer.image 2 img_output LAM4

; Assign Template to Be Executed
mov.tem.tem tem33 TEM1

; Copy Input Image to the Chip Memory
mov.lam.lam lam5 LAM1

; Compute the Transient
ar.tem.run TEM1 gamf1 1

; Read Out the Result from the Chip
mov.lam.lam LAM3 lam6
mov.lam.lam LAM4 lam7

; Display It
host.display lam6 1
host.display lam7 2

; Increase Cycle Counter
sc.add.gam_f 50 gamf1 gamf1
sc.rel.gt 3500.0 gamf1 glm1
jumpc glm1 Loop

end
```
References


[33] http://www.sztaki.hu/~petras/cace1k