TR-H-277 026

CoDi Technique: Cellular Automata-Based Neural Networks.

Andrzej BULLER (ATR-HIP/Univ. Gdansk),
Tomasz CHODAKOWSKI (Univ. Gdansk),
Hitoshi HEMMI (NTT CS Lab.) and Katsunori SHIMOHARA (NTT CS Lab./ATR-HIP)

1999.11.26

ATR人間情報通信研究所
〒619-0288 京都府相楽郡精華町光台2-2 TEL: 0774-95-1011

ATR Human Information Processing Research Laboratories
2-2, Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0288, Japan
Telephone: +81-774-95-1011
Fax : +81-774-95-1008

Andrzej Buller

ATR Human Information Processing Research Laboratories
2-2 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-02 Japan
abuller@hip.atr.co.jp
&
Technical University of Gdańsk
Faculty of Electronics, Telecommunications & Informatics
ul. G.Narutowicza 11/12, 80-952 Gdańsk, Poland
buller@pg.gda.pl

Tomasz Chodakowski

Gdańsk University, Department of Mathematic and Physics
ul. Wita Stwosza 57, 80-952 Gdańsk-Oliwa, Poland
tch@manta.univ.gda.pl

Hitoshi Hemmi

NTT Communication Science Laboratories
2-4 Hikaridai, Soraku-gun, Kyoto 619-02 Japan
hemmi@cslab.kecl.ntt.co.jp

Katsunori Shimohara

NTT Communication Science Laboratories
2-4 Hikaridai, Soraku-gun, Kyoto 619-02 Japan
katsu@cslab.kecl.ntt.co.jp
&
ATR Human Information Processing Research Laboratories
2-2 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-02 Japan
katsu@hip.atr.co.jp

November 17, 1999

CoDi Technique: Cellular Automata-Based Neural Networks

Abstract—We present an extraordinary approach to building very large-scale neural systems, using cellular automata. The key concept is the CoDi cell—a simple binary signal-processing device to be represented by a single cell in a cellular automata workspace. A set o appropriately configured CoDi-cells form a neural network. We provide a formal definition of the CoDi cell, as well as definitions of the CoDi neuron and the CoDi module. We also present results of tests of some CoDi modules evolved using a software simulator of cellular automata, as well as a basic description of the CBM—a special-purpose hardware for evolution of CoDi modules and running multi-module structures. In some concluding remarks we argue that the CoDi technique is a key to strong Artificial Intelligence.

I. INTRODUCTION

The CoDi technique, developed in the framework of the ATR's CAM-Brain Project [3], like CABANN [8] combines neural engineering with cellular automata (CA) paradigm [15], and like ECANS [10][11] employs evolutionary computation [6] to obtain cellular-neural systems. The CoDi ("collect" and "distribute") model is based on CoDi cells—processing units defined in cellular automata workspace. Every CoDi cell collects 1-bit signals obtained from some of its neighboring cells, processes them, and distributes among other neighboring cells. A set of CoDi cells is organized in such a way that they form the CoDi module—a model of a neural network receiving and producing spatio-temporal binary patterns [5]. Hence, the CoDi module belongs to the class of Pulsed Neural Networks [12].

A configuration of CoDi cells in a CoDi module is a function of a binary chromosome, and, this way, it may be a subject to an evolutionary process based on mutation and crossover.

Using a software simulator of CA and a genetic algorithm [4] several useful devices have been obtained, as for example, a timer triggered by a single pulse, frequency-to-delay converter, or an erasable-memory unit. Based on the last one, several kinds of learning systems can be built, including brain-like systems [1][2].

Using a special purpose hardware, called CBM, an evolution of a desired CoDi module should be possible in seconds, while a designed neural structure consisting of 64,640 CoDi modules (an equivalent of over 75 million neurons) should be updated almost 150 times per second. This means, that the CBM can serve an artificial brain controlling a behavior of a robot-pet, as, for example, a kitten [9].

Formal definitions of objects used in the CoDi technique will be based on some basic notions listed below:

General notions:

$\mathbf{B} = \{0, 1\}$.
$\mathbf{F} = \{-1, 0, 1\}$.
$\mathbf{E}_e = \{i \mid i \in \mathbf{I}_+, i < e \in \mathbf{I}_+\}$.
$\mathbf{I}$ – space of integers.
$\mathbf{I}_+$– space of non-negative integers.
$\mathbf{L} = \{ i \mid i \in \mathbf{I}, i_{min} \leq i \leq i_{max}, i_{min} \in \mathbf{I}, i_{max} \in \mathbf{I} \}$,
$\mathbf{R}_+$ – space of non-negative real numbers
$\mathbf{T}$ – space of discrete values of flowing time (non-negative integers).
$\mathbf{V} = \{(v_x, v_y, v_z) \in \mathbf{F}^3 \mid v_x^2 + v_y^2 + v_z^2 = 1\}$.

CoDi-specific notions:

$\boldsymbol{A_m}$ – the active surface of the CoDi module $\boldsymbol{m}$.
$\boldsymbol{C}$ – set of CoDi cells.
e – length of the edge of a CoDi-module.
$\boldsymbol{J_m}$ – set of input points of the CoDi module $\boldsymbol{m}$.
$\boldsymbol{K_m}$ – set of output points of the CoDi module $\boldsymbol{m}$.
$\boldsymbol{M}$ – set of CoDi modules.
$\boldsymbol{N}$ – set of CoDi neurons.
$\boldsymbol{P}$ – set of CoDi pseudo-modules
$\boldsymbol{R}$ – space of CoDi-module total reactions.
$\boldsymbol{S}$ – space of CoDi-module total stimuli.
$\boldsymbol{U}$– space of CoDi-cell total stimuli.
$\boldsymbol{X}$ – space of CoDi-cell states.
$\boldsymbol{Y}$ – space of CoDi-cell reactions.
$\boldsymbol{Z}$ – space of CoDi-module states.

II. CoDi CELL

A basic unit in CoDi technique is CoDi cell—a binary-signal-processing device having 6 input/output points. Let X be the space of all possible functions $x \mid \mathbf{T} \rightarrow \mathbf{L}$. Let U be the space of all possible functions $u \mid \mathbf{V} \times \mathbf{T} \rightarrow \mathbf{B}$. Let Y be the space of all possible functions $y \mid \mathbf{T} \rightarrow \mathbf{B}$.

Definition 1: *CoDi cell*, denoted c, is a triple $\langle p, w, f \rangle$, where $p \in \mathbf{I}^3$, $w \mid \mathbf{V} \rightarrow \mathbf{F}$, $f \mid U \rightarrow X \times Y$.

f is a CoDi-cell's transition function. A convenient metaphor of a CoDi cell is a cube of the size $1 \times 1 \times 1$ with its facets labeled using $\mathrm{i} \in \mathbf{V}$, where i may be interpreted as a vector pointing towards a given facet from the center of the cell. Thus p may be interpreted as the location of the cell in Euclidean 3-D space, $w(\mathrm{i})$ as a binary value being an attribute of the cell's facet labeled i.

There are three types of CoDi cells: axonic, dendritic, and neuron-body cells. The axonic and dendritic ones constitute, respectively, axons and dendrites of a CoDi neuron. A neuron-body cell acts alone as a body of a CoDi neuron.

Let $x_c \in X$ will be interpreted as a function returning a state of the CoDi cell $\boldsymbol{c}$ for a given time. Let $u_c \in U$ will be interpreted as a function returning a stimulus of a given facet of the CoDi cell $\boldsymbol{c}$ for a given time. Let $y_c \in Y$ will be interpreted as a function returning a reaction of the CoDi cell c for a given time. Let $\boldsymbol{C}$ be the space constituted by all possible CoDi cells.

Definition 2: *Axonic CoDi cell* is a triple $\langle p, w, \alpha \rangle \in \boldsymbol{C}$, such that

$$\exists_{\mathrm{i} \in \mathbf{V}}\ w(\mathrm{i}) = 1, \quad \forall_{\mathrm{k}}\ \mathrm{k} \neq \mathrm{i}\ \ w(\mathrm{k}) = 0;$$

$$\langle x, y \rangle = \alpha(u) \Rightarrow \forall_{\mathrm{t} \in \mathbf{T}}\ x(\mathrm{t}+1) = y(\mathrm{t}+1) = \mathbf{OR}_{\mathrm{i} \in \mathbf{V}}\, w(\mathrm{i})u(\mathrm{i}, \mathrm{t}).$$

Definition 3: *Dendritic CoDi cell* is a triple $\langle p, w, \delta \rangle \in \boldsymbol{C}$, such that

$$\exists_{\mathrm{i} \in \mathbf{V}},\ \ w(\mathrm{i}) = 0; \quad \forall_{\mathrm{k}}\ \mathrm{k} \neq \mathrm{i}\ \ w(\mathrm{k}) = 1;$$

$$\langle x, y \rangle = \delta(u) \Rightarrow \forall_{\mathrm{t} \in \mathbf{T}}\ x(\mathrm{t}+1) = y(\mathrm{t}+1) = \mathbf{OR}_{\mathrm{i} \in \mathbf{V}}\, w(\mathrm{i})u(\mathrm{i}, \mathrm{t}).$$

Definition 4: *Neuron-body CoDi cell* is a triple $\langle p, w, \eta\rangle \in \boldsymbol{C}$, such that

$$\exists_{\mathrm{i,j} \in \mathbf{V}},\ \mathrm{i}\neq\mathrm{j}\quad w(\mathrm{i}) = 0,\ w(\mathrm{j}) = 0\quad \forall_{\mathrm{k}}\ \mathrm{k}\neq\mathrm{i},\ \mathrm{k}\neq\mathrm{j}\quad w(\mathrm{k}) \neq 0\ );$$

$$\langle x, y\rangle = \eta(u) \Rightarrow \forall_{\mathrm{t}\in\mathbf{T}}$$

$$(\ x_c(\mathrm{t}+1) = net(\mathrm{t}) \Leftrightarrow net(\mathrm{t}) \in \mathbf{L} \text{ while otherwise } x(\mathrm{t}+1) = 0;$$

$$y_c(\mathrm{t}+1) = 1 \Leftrightarrow net(\mathrm{t}) > \sup(\mathbf{L}) \text{ while otherwise } y(\mathrm{t}+1) = 0\ ),$$

where $net(\mathrm{t}) = x\,(\mathrm{t}) + \Sigma_{\mathrm{i}\in\mathbf{V}}\, w(\mathrm{i})u(\mathrm{i},\, \mathrm{t})$.

Each of the CoDi cells collects signals through its facets that have assigned values -1 or 1 and sends signals thorough facets having assigned the value 0. Some facets of a CoDi cell can be distinguished and referred as *gates* i.e. facets where cell of given type points to the object of it's activity. A task of a *dendritic* cell is to collect signals from its five neighbor cells and deliver signals to a neighbor cell its facet having assigned zero value points to, hence in the *dendritic* cell the gate is a facet i such that with $w(\mathrm{i}) = 0$. A task of an *axonic* cell is to get signal from the neighbour its non-zero gate points to, and distribute them to its other neighbors, hence, in the *axonic* cell the gate is the facet i such that $w(\mathrm{i}) = 1$. In a *neural-body* cell there are two and only two zero facets, which means that a neural-body cell can pass signals to exactly two neighbours. So in a neuron-body cell the gates are the facets i and j such that $w(\mathrm{i}) = 0$ and $w(\mathrm{j}) = 0$.

A CoDi cell delivers signals to another cell when certain conditions are satisfied. First, the distance between sender's center and receiver's center must equal 1. Second, the facet of contact in case of the cell-sender must have assigned value equal to 0, while in case of the cell-receiver the value must equal -1 or 1.

III. CoDi NEURON

CoDi neuron is a rough imitation of its biological counterpart. It has its body, dendritic tree, and axonic tree. CoDi-neuron formal definition uses such auxiliary notions as CoDi-level sender-receiver relationship and the relationship's characteristic function λ.

Definition 5: Two CoDi cells $c = \langle p, w, f\rangle$, $\boldsymbol{c'} = \langle p', w', f'\rangle \in \boldsymbol{C}$ are in the *CoDi-cell-level sender-receiver relation*, denoted $\langle c \rightarrow c'\rangle$, where c is assumed to be a sender, while $\boldsymbol{c'}$ is assumed to be a receiver, when $\exists_{\,\mathrm{i} \in \mathbf{V}}\ \ p' = p + \mathrm{i}$, $w(\mathrm{i}) = 0$, $w'((0, 0, 0) - \mathrm{i}) \neq 0$ and $(f, f') \in \{\ (\delta, \delta), (\delta, \eta), (\eta, \alpha), (\alpha, \alpha), (\alpha, \delta)\ \}$.

Definition 6: *CoDi-cell-level sender-receiver-relation characteristic function*

$\lambda \mid \boldsymbol{C}^2 \rightarrow \mathbf{B}$ is such that:

$\lambda_{f\rightarrow f'}(c, c') = 1 \Leftrightarrow c = \langle p, w, f\rangle, \boldsymbol{c'} = \langle p', w', f'\rangle \in \boldsymbol{C}$ and $\langle c \rightarrow c'\rangle$,

while otherwise $\lambda_{f\rightarrow f'}(c, c') = 0$.

For a given CoDi cell c, the stimulus $u_c(\mathrm{i,t})$ is a reaction $y_{c'}(\mathrm{t})$ of CoDi cell $c' = \langle p', w', f'\rangle$ such that $p' = p + \mathrm{i}$ and $\langle c' \rightarrow c\rangle$, i.e. $u_c(\mathrm{i,t}) = y_{c'}(\mathrm{t})$.

Definition 7: *CoDi neuron* $\boldsymbol{n}$ is a processing device consisting of CoDi cells configured in such a way that: $\boldsymbol{n} = \langle n, \boldsymbol{D}, \boldsymbol{A}\rangle$, where: $\boldsymbol{D} = \{d_1, d_2, \ldots, d_{\mathrm{n}}\}$ is a set of dendritic cells forming a dendritic tree of a neuron body cell $n = \langle p, w, \eta\rangle \in \boldsymbol{C}$ in a way such that: $\forall_{\mathrm{m} = 1, \ldots, \mathrm{d}}\ \lambda_{\delta\rightarrow\eta}(\ d_{\mathrm{m}}, n\ )$ and $\forall_{\mathrm{k} = \mathrm{d}+1, \ldots, \mathrm{n}}\ \exists_{\ \mathrm{k0, k1, \ldots, kj, m}},\ \ \mathrm{m} \in \{1, .., \mathrm{d}\} \mid \lambda_{\delta\rightarrow\delta}(\ d_{\mathrm{k}}, d_{\mathrm{kj}}\ ) \wedge \lambda_{\delta\rightarrow\delta}(\ d_{\mathrm{kj}}, d_{\mathrm{k-1}}\ ) \wedge \ldots \wedge \lambda_{\delta\rightarrow\delta}(\ d_{\mathrm{k0}}, d_{\mathrm{m}}\ )$, where d is a number of dendrities of in the neuron $\boldsymbol{n}$, $\boldsymbol{A} = \{a_1, a_2, \ldots, a_{\mathrm{n}}\}$ is a set of axonic cells which form an axon of the neuron body cell $n = \langle p, w, \eta\rangle \in \boldsymbol{C}$ in a way such that: $\forall_{\mathrm{m} = 1, \ldots, \mathrm{a}}\ \lambda_{\eta\rightarrow\alpha}(\ n, \mathrm{a_m})$ and $\forall_{\mathrm{k} = \mathrm{a}+1, \ldots, \mathrm{n}}\ \exists_{\ \mathrm{k0, k1, \ldots, kj, m}}\ ,\ \mathrm{m} \in \{1, .., \mathrm{a}\} \mid \lambda_{\alpha\rightarrow\alpha}(\ a_{\mathrm{m}}, a_{\mathrm{k0}}\ ) \wedge \lambda_{\alpha\rightarrow\alpha}(\ a_{\mathrm{k0}}, a_{\mathrm{k1}}\ ) \wedge \ldots \wedge \lambda_{\alpha\rightarrow\alpha}(\ a_{\mathrm{kj}}, a_{\mathrm{k}}\ )$, where a is a number of axons of the neuron $\boldsymbol{n}$.

Remark 1: It can be noted that $\mathrm{a} > 0$, $\mathrm{d} > 0$, and $2 \leq \mathrm{a+d} \leq 6$.

Definition 8: Two neurons $\boldsymbol{n}$ and $\boldsymbol{n'}$ are in neuron-level sender-receiver relationship, what is denoted by $\langle \boldsymbol{n}, \boldsymbol{n'}\rangle$, where the neuron $\boldsymbol{n} = \langle n, \boldsymbol{D}, \boldsymbol{A}\rangle$ is called sender while $\boldsymbol{n'} = \langle n', \boldsymbol{D'}, \boldsymbol{A'}\rangle$ receiver, when $\exists_{a\in A}, \exists_{d\in D'}\ \lambda_{\alpha\rightarrow\delta}(\ a, d\ )$.

Remark 2: It may be noted, that both $\langle \boldsymbol{n}, \boldsymbol{n'}\rangle$ and $\langle \boldsymbol{n'}, \boldsymbol{n}\rangle$ can occur simultaneously, what is impossible in case of cell-level sender-receiver relationship.

IV. CoDi MODULE

CoDi module is a set CoDi cells configured is such a way, that all of them serve as parts of CoDi neurons and belong to cube of the size e $\times$ e $\times$ e. This way a CoDi module imitates a neural network going to a defined cube (Fig. 1).

Let N be the set of all possible CoDi neurons.

Definition 9: *CoDi module* $\boldsymbol{m}$ is a processing device, such that

$\boldsymbol{m} = \{ \boldsymbol{n} \mid \boldsymbol{n} = \langle n, \boldsymbol{D}, \boldsymbol{A} \rangle \in \boldsymbol{N}, (\langle p, w, \eta \rangle = n, \langle p', w', \delta' \rangle \in \boldsymbol{D}, \langle p'', w'', \alpha'' \rangle \in \boldsymbol{A}) \Rightarrow p, p', p'' \in \mathbf{E}_e^3 \}$; $\boldsymbol{J}_m \neq \varnothing$, $\boldsymbol{K}_m \neq \varnothing$, where e is the length of the CoDi module's edge, $\boldsymbol{J}_m \subseteq \mathbf{E}_e^3$ is the *CoDi-module input-point set* assumed as a set of places to which external signals can be provided, $\boldsymbol{K}_m \subseteq \mathbf{E}_e^3$ is the *CoDi-module output-point set* assumed as a set of places from which signals can be taken by an external device.

Let all possible CoDi modules form the set of CoDi modules, denoted $\boldsymbol{M}$. A *CoDi module* is subject to stimulation, to which it reacts in a specific manner, depending of the module internal state.

Definition 10: *CoDi-module local stimulus* is, for a given CoDi module $\boldsymbol{m} \in \boldsymbol{M}$, a function $s_m \mid \boldsymbol{J}_m \times \mathbf{T} \rightarrow \mathbf{B}$, such that for all $\boldsymbol{i} \in \boldsymbol{J}_m$, $s_m(\boldsymbol{i}, \mathrm{t}) = \boldsymbol{u}_i(\mathrm{t})$, where $\boldsymbol{u}_i$ is the total stimulus of the CoDi cell $\boldsymbol{i}$.

Definition 11: *CoDi-module total stimulus* is, for a given CoDi module $\boldsymbol{m} \in \boldsymbol{M}$, a function $s_m \mid \mathbf{T} \rightarrow \mathbf{B}^{\mathrm{J}}$, such that $s_m(\mathrm{t}) = (s(\boldsymbol{i}_1, \mathrm{t}), s(\boldsymbol{i}_2, \mathrm{t}), \ldots s(\boldsymbol{i}_{\mathrm{J}}, \mathrm{t}))$, where J is the number of input points. All possible CoDi-module total-stimuli form the space of *CoDi-module total stimuli*, denoted $\boldsymbol{S}$.

Definition 12: *CoDi-module state* is, for a given CoDi module $\boldsymbol{m} \in \boldsymbol{M}$, a function $\boldsymbol{x}_m \mid \mathbf{T} \rightarrow \mathbf{I}^{\mathrm{C}}$, such that $\boldsymbol{x}_m(\mathrm{t}) = ((x_{c1}(\mathrm{t}), x_{c2}(\mathrm{t}), \ldots x_{c\mathrm{C}}(\mathrm{t}))$, where C is the number of all CoDi cells forming the CoDi-module neurons. All possible CoDi-module states form the *space of CoDi-module states*, denoted $\boldsymbol{Z}$.

Definition 13: *CoDi-module local reaction* is, for a given CoDi module $\boldsymbol{m}$, a function $r_m \mid \boldsymbol{J_m} \times \mathbf{T} \rightarrow \mathbf{B}$, such that for all $\boldsymbol{o} \in \boldsymbol{K_m}$, $r_m(\boldsymbol{o}, \mathrm{t}) = y_o(\mathrm{t})$, where y_o is the reaction of the CoDi cell $\boldsymbol{o}$.

Definition 14: *CoDi-module total reaction* belonging to the space $\boldsymbol{R}$ of *CoDi-module total reactions* is for a given CoDi module $\boldsymbol{m}$, a function $\boldsymbol{r_m} \mid \mathbf{T} \rightarrow \mathbf{B}^{\mathsf{K}}$, such that $\boldsymbol{r_m}(\mathrm{t}) = (r(o_1, \mathrm{t}), r(o_2, \mathrm{t}), \ldots\, r(o_{\mathsf{K}}, \mathrm{t}))$, where K is the number of output points. All possible CoDi-module total reactions form the *space of CoDi-module total reactions*, denoted $\boldsymbol{S}$.

Remark 3: Since CoDi module is a finite-state dynamical system, for a given module $\boldsymbol{m} \in \boldsymbol{M}$, its transition function $f_m \mid \boldsymbol{S} \rightarrow \boldsymbol{Z} \times \boldsymbol{R}$, is state-based, i.e. $\exists f$ that for all $\boldsymbol{s_m} \in \boldsymbol{S}$, $\boldsymbol{x_m} \in \boldsymbol{Z}, \boldsymbol{r_m} \in \boldsymbol{R} : \langle x_m, r_m \rangle = f_m(s_m) \Rightarrow \forall_{\mathrm{t} \in \mathbf{T}} \langle \boldsymbol{x_m}(\mathrm{t}+1), \boldsymbol{r_m}(\mathrm{t}+1) \rangle = f(\boldsymbol{x_m}(\mathrm{t}), \boldsymbol{s_m}(\mathrm{t}))$,

A *CoDi module* is to process a set of binary stimuli incoming in discrete moments over a period of time. A quality of the processing, assumed to be equal to a quality of the CoDi module itself, may be measured based on observed reactions of the module to arbitrary given stimuli. A set of obtained reactions of a given module $\boldsymbol{m}$ depends on its internal structure which determines the module transition function f_m. Def. 13. provides a formal description of one of possible CoDi-Module quality measurements.

Definition 15: *CoDi module quality measurement* is, for a given module $\boldsymbol{m}$, spike-train length Δ, 'technological' delay τ, and an arbitrary designed set $\boldsymbol{\Phi} = (\Phi_1, \Phi_2, \ldots, \Phi_{\mathrm{n}})$ of testing series of total stimuli, given for $\mathrm{t} = 1, 2, \ldots, \Delta$, the value of the expression

$$\Sigma^{\mathrm{n}}_{i=1} \left(1 - \left(\Sigma^{\mathsf{K}}_{k=1} \mathrm{H}(\omega_{m,k,i}, \theta_{m,k,i}, \Delta) / \Delta\right) / \mathsf{K}\right) / \mathrm{n}$$

where K is a number of used output points, $\theta_{m,k,i}$ is a desired spike-train to be produced at the k-th output point of $\boldsymbol{m}$ as a reaction to Φ_i measured for $\mathrm{t} = \tau + 1, \tau + 2, \ldots, \tau + \Delta$, $\omega_{m,k,i}$ is a spike-train obtained at the k-th output point of m as a reaction to

Φ_i, measured for $t = \tau + 1, \tau + 2, \ldots, \tau + \Delta$, and H is the Hamming distance between the two spike-trains.

V. WHAT CoDi MODULE CAN DO?

A number of CoDi modules have been obtained using a software simulator of cellular automata. The following constraints was taken for the simulation experiment:

$\mathbf{L} = \{ i \mid i \in \mathbf{I} \mid -8 < i < 2 \}$; for each simulated CoDi module $\boldsymbol{m}$:

for all $\langle p, \mathrm{w}, \eta\rangle \in \boldsymbol{m}$, $(p \in \{ (x, y, z) \in \mathbf{I}^3 \mid x \bmod 3 = 0, y \bmod 2 = 0, z \bmod 2 = 0\}$;

$\boldsymbol{J_m} = \{ \boldsymbol{c} = \langle p, \mathrm{w}, \alpha\rangle \in \boldsymbol{C} \mid p \in \boldsymbol{A_m} ,\ \forall v \in \mathbf{V} ,\ p+v \notin \boldsymbol{J_m} \}$; and

$\mathbf{K_m} = \{ \mathbf{c} = \langle p, \mathrm{w}, \delta\rangle \in \mathbf{C} \mid p \notin \mathbf{A_m} \}$, where

$\mathbf{A_m} = \{p \mid p \in \{(0, y, z) \mid y, z \in \mathbf{E}_{24}\} \cup \{(x, 0, z) \mid x, z \in \mathbf{E}_{24}\} \cup \{(x, y, 0) \mid x, y \in \mathbf{E}_{24}\}$.

A binary chromosome contained full information about a role of each of $24 \times 24 \times 24$ cells forming a CoDi module. Based on such a chromosome a neural network could grow inside the module. In order to obtain a CoDi module behaving in a desired way a genetic algorithm was implemented. And so, a population of random chromosomes was generated, based on the chromosomes CoDi modules were grown and tested, the most promising solutions could mate and have their offspring inheriting appropriate parts of their parental chromosomes. Usually after passing of some hundreds of generations a satisfactory module appeared. This way we evolved, among others, a neural frequency-to-delay converter and a neural erasable memory unit. The behavior of the converter is showed in the Fig.2. As it can be seen, the lower frequency of incoming spikes, the longer time of waiting for start of spike production. The quality measurement for the converter was taken based only on training examples and its value was equal to 0.974.

The erasable memory unit was intended to start producing a dense spike-train when a not interrupted series of 4 spikes are provided to one of its inputs, and stop producing spikes when a not interrupted series of 4 adjacent spikes are provided to the second of its inputs. In fact each of the inputs consisted of 6 arbitrary selected input points to the CoDi module, to which the same signal was provided. For the erasable memory unit a quality measurement was taken based on a set of 32 testing examples not used during evolution and its value was equal to 1.0. Some examples of the module reactions to the test spike-trains are showed in the Fig. 3. Further investigation revealed an interesting property of the evolved memory unit. It proved to react properly to not interrupted series of three spikes or even to two adjacent spikes (Fig. 4. and Fig. 5.).

In order to use CoDi to a processing of 2D moving pictures, we employ the SIIC (Spike Interval Information Coding) [13] and HAS (Hough Spiker Algorithm for Deconvolution) [7] which convert analog waveforms to spike-trains and vice versa [4].

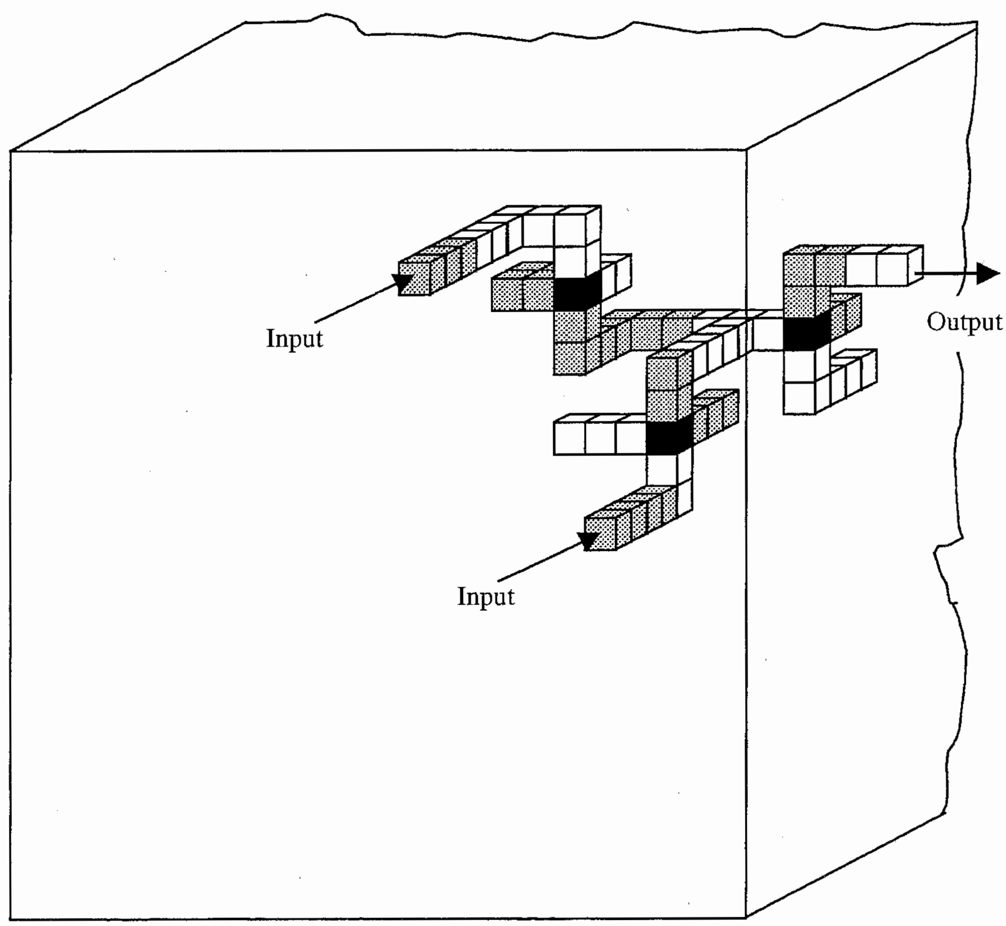

Fig.1. CoDi Module. In a cube of 24×24×24 cellular automata cells a network of up to 1152 neurons grows. In this figure axons are represented by grey cells, black cells represent neural-body cells, while white cells form dendrites. Using current version of Korkin CBM (Cellular [Automata-based] Brain Machine) a simulation of a structure consisting of 64,640 such modules is possible with the speed comparable with biological brains. (Adapted from [2]).

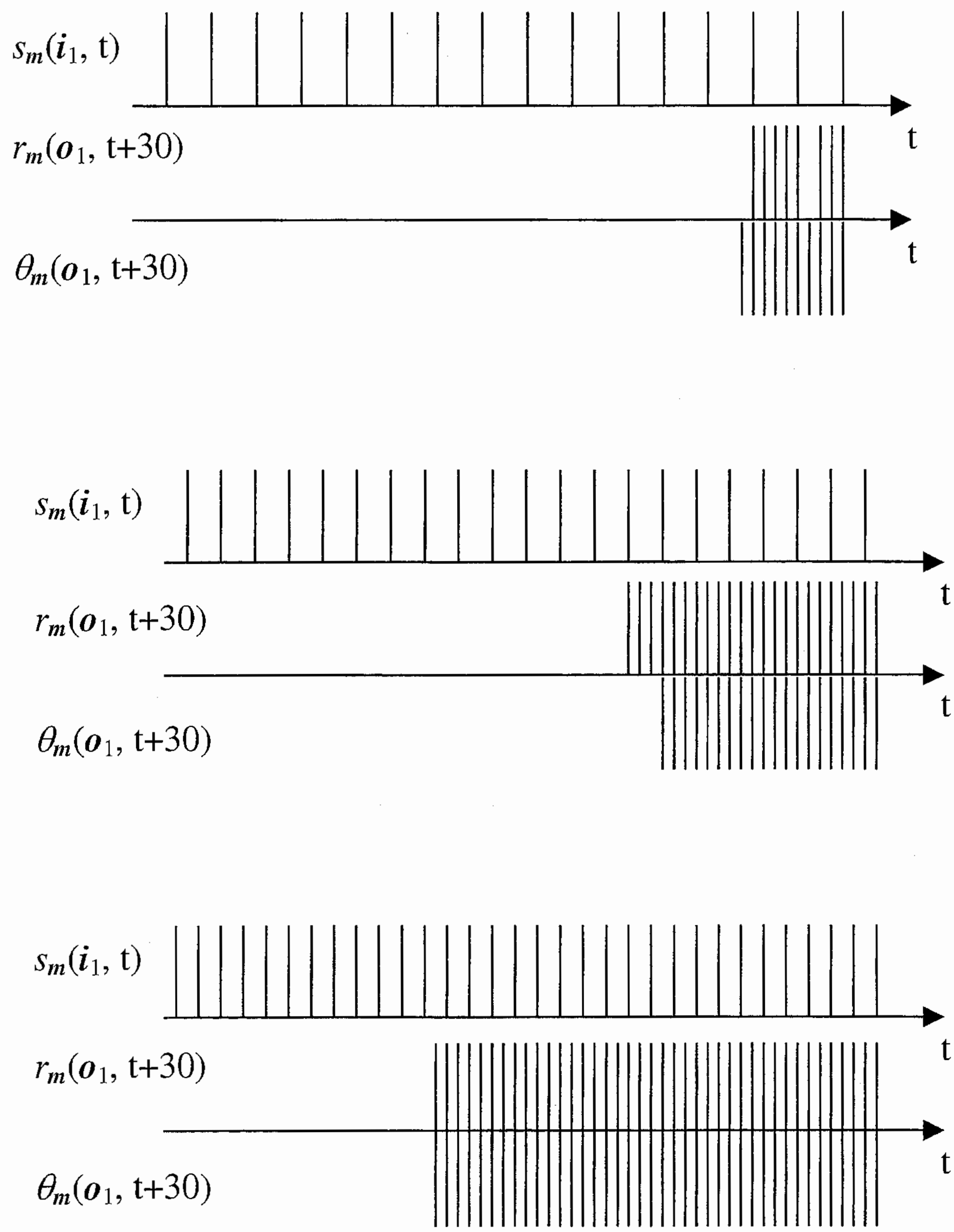

Fig. 2. A behavior of a CoDi module evolved to serve as a Frequency-to-Delay converter. $s_m(i_j, t)$ – a stimulus of the j^{th} input point, $r_m(o_1, t)$ – actual reaction for the given stimuli at the k^{th} output point. $\theta_m(o_k, t)$ – target reaction at the k^{th} output point.

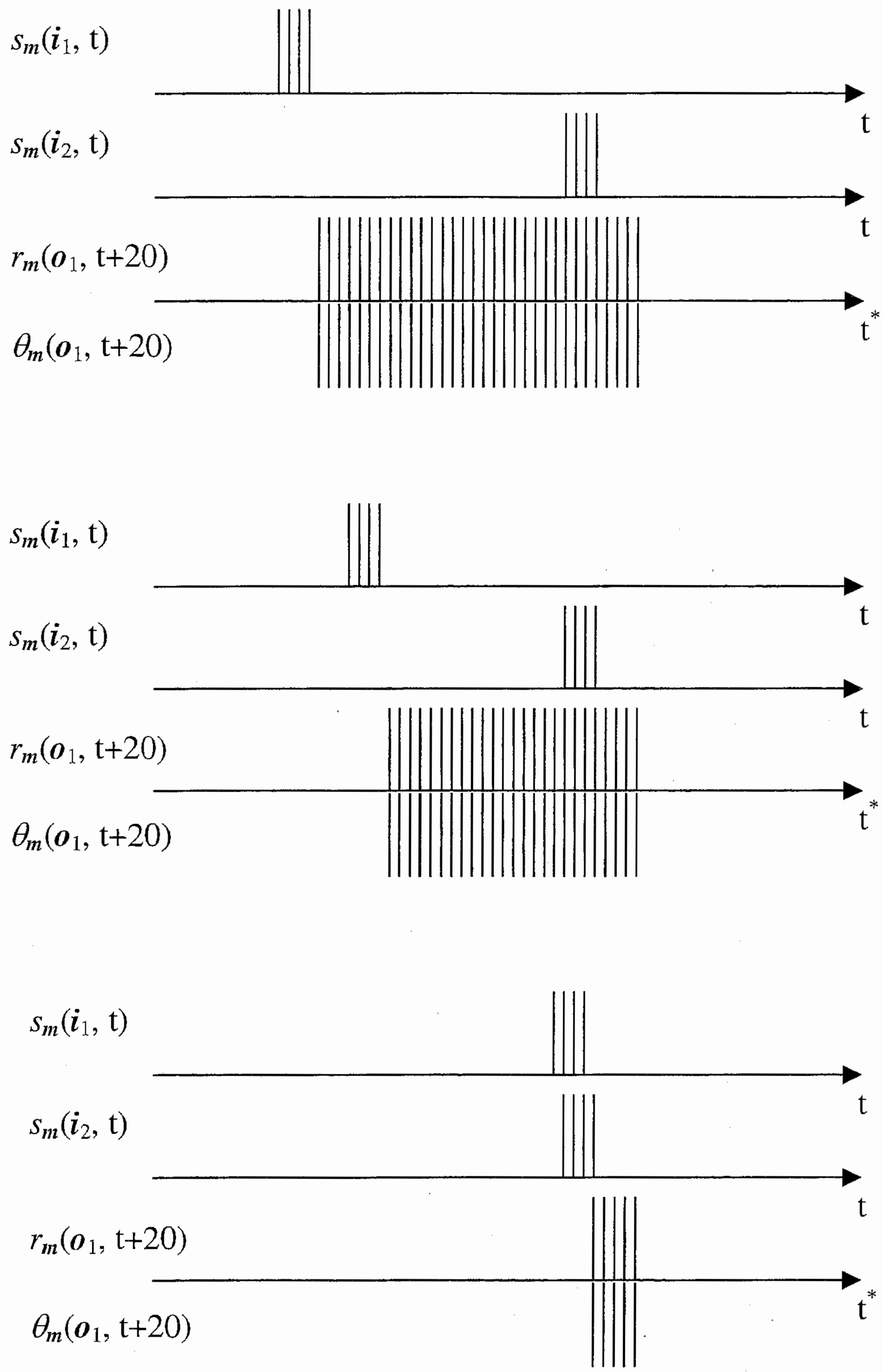

Fig. 3. A behavior of a CoDi module evolved to serve as a memory unit (Flip-Flop). A series of 4 consecutive spikes serves as the memory loader or discharger. $s_m(\boldsymbol{i}_\mathrm{j}, \mathrm{t})$ – a stimulus of the j^{th} input point, $r_m(\boldsymbol{o}_1, \mathrm{t})$ – actual reaction for the given stimuli at the k^{th} output point. $\theta_m(\boldsymbol{o}_\mathrm{k}, \mathrm{t})$ – target reaction at the k^{th} output point.

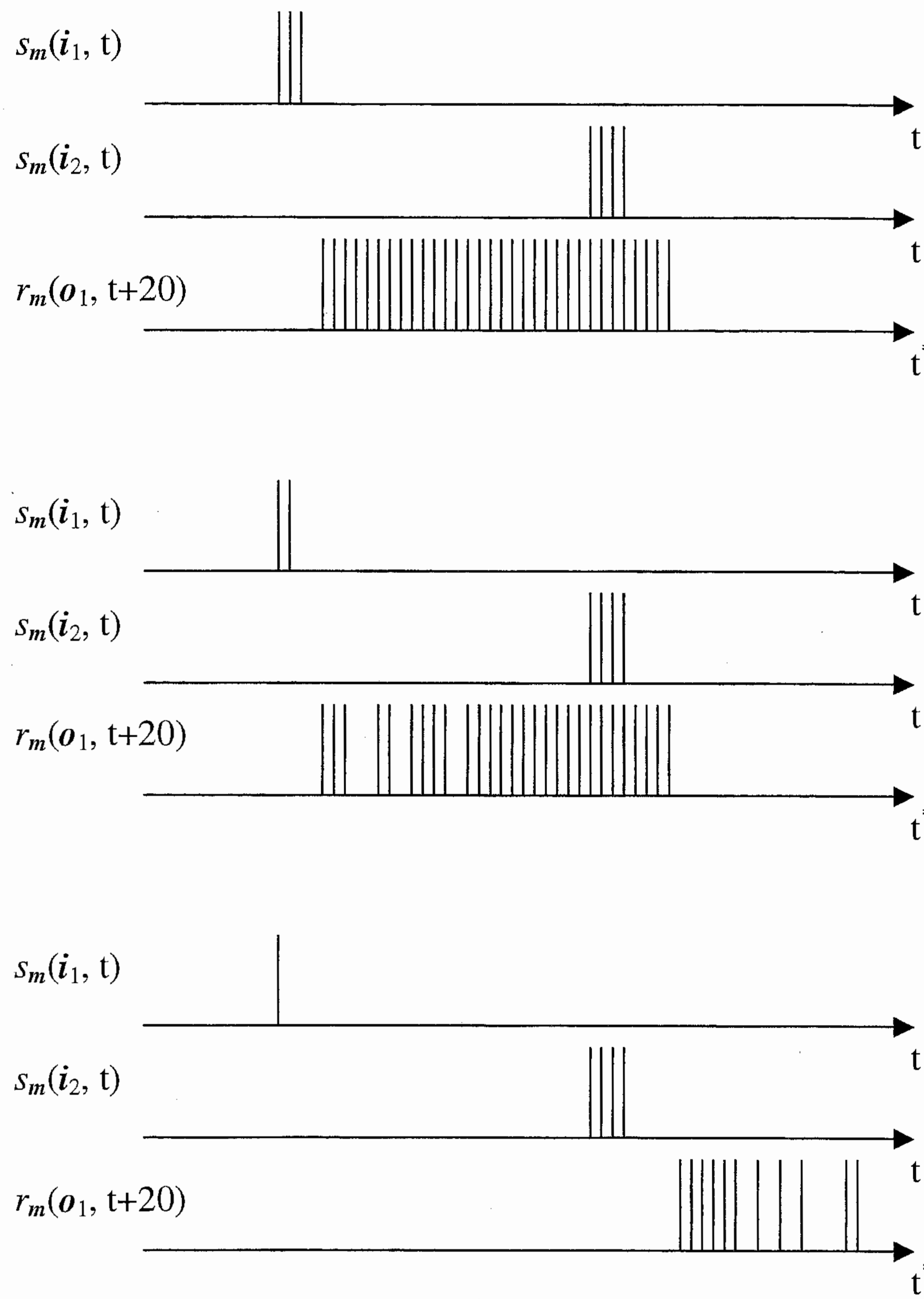

Fig 4. The CoDi module evolved as a Flip-Flop to be set/reset by four-spike series works properly when set by three-spike series, works almost properly when set by two-spike series, but does not work when one tries to set it using a single spike. $s_m(\boldsymbol{i}_j, t)$ – a stimulus of the j^{th} input point, $r_m(\boldsymbol{o}_1, t)$ – actual reaction for the given stimuli at the k^{th} output point.

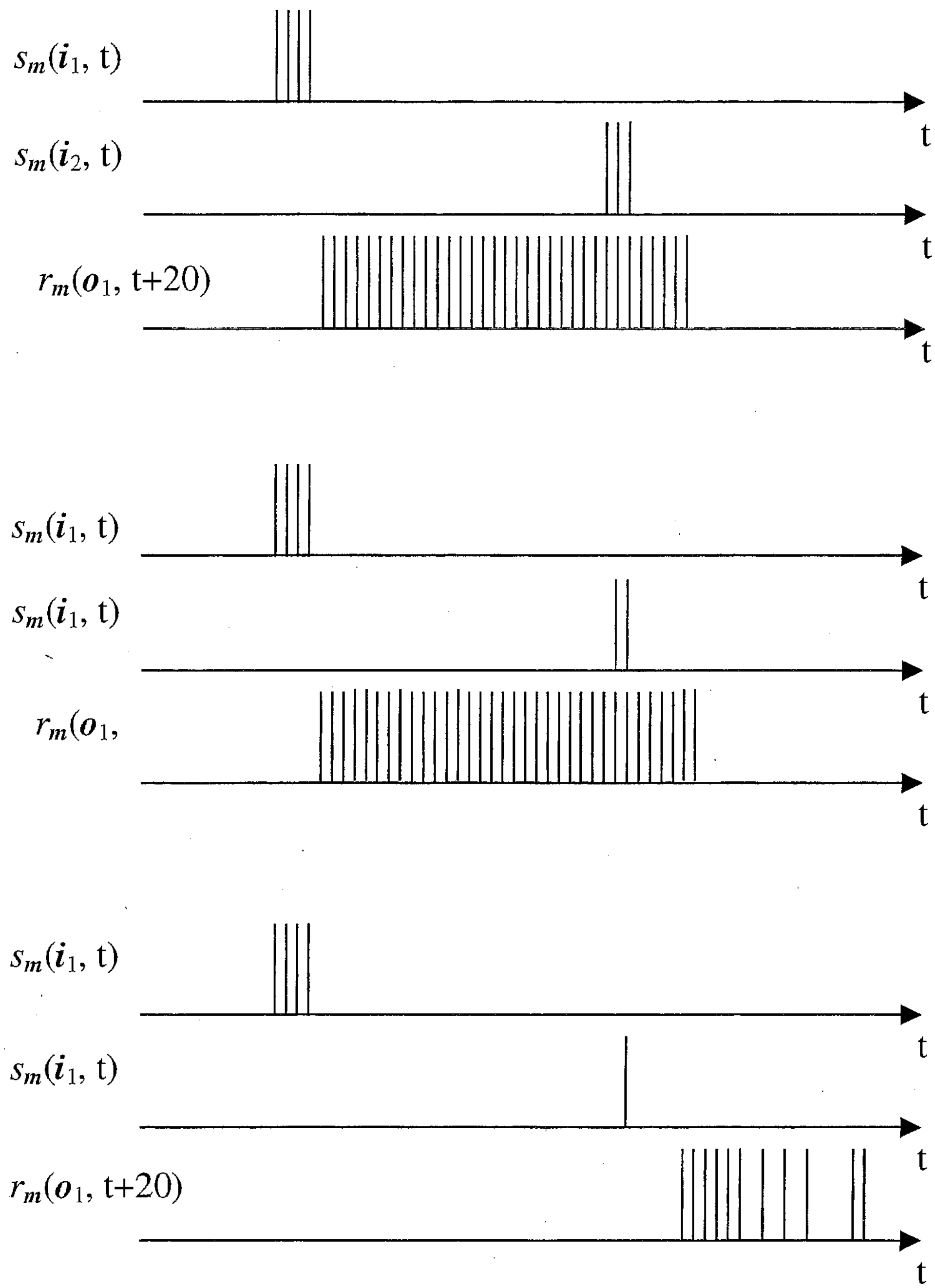

Fig 5. The CoDi module evolved as a Flip-Flop to be set/reset by four-spike series works properly when reset by three-spike series, works almost properly when reset by two-spike series, but does not work when one tries to reset it using a single spike. $s_m(\boldsymbol{i}_\mathrm{j}, \mathrm{t})$ – a stimulus of the j^{th} input point, $r_m(\boldsymbol{o}_1, \mathrm{t})$ – actual reaction for the given stimuli at the k^{th} output point.

VI. CBM - A HARDWARE FOR BUILDING CoDi-BASED SYSTEMS

The CBM (Cellular [Automata-Based] Brain Machine) is a research tool for rapid CoDi-module evolving, as well as for the simulation of multi-module systems. An original set of ideas for the CAM-Brain project was developed by Dr. Hugo de Garis at the Evolutionary Systems Department of ATR HIP (Kyoto, Japan), and is currently implemented as a dedicated research tool by Genobyte, Inc. (Boulder, Colorado). Genobyte is licensed by ATR International and Japan's Key Technologies Center to manufacture and sell CBMs to third parties [3].

An artificial brain, supported by the CBM, consists of up to 64,640 CoDi modules, each populated by 1,152 neurons, for a total of 74.5 million neurons. Within each module, neurons are densely interconnected with branching dendritic and axonic trees in a three-dimensional space, forming an arbitrarily complex interconnection topology. A neural module can receive afferent axons from 188 other modules of the structure, with each axon being capable of multiple branching in three dimensions, forming hundreds of connections with dendritic branches inside the module. Each module sends efferent axon branches to up to 64,640 other modules.

A critical part of the CBM approach is that CoDi modules are not "manually designed" or "engineered" to perform a specific function, but rather evolved directly in hardware, using genetic algorithms.

Genetic algorithms operate on a population of chromosomes, which represent neural networks of different topologies and functionalities. Better performers for a particular function are selected and further reproduced, using chromosome recombination and mutation. After hundreds of generations, this approach produces very complex neural networks with a desired functionality. The evolutionary approach can create a complex functionality without any a priori knowledge about how to achieve it, as long as the desired input/output function is known.

A summary of CBM technical parameters are showed in Tab. 1. The next section provides a detailed description of CBM architecture.

Cellular Automata Update Rate (max.):	131 billion cells/s
Number of Supported Cellular Automata Cells (max.):	893 million
Number of Supported Neurons per CoDi Module (max.):	1152
Number of Supported CoDi Modules:	64,640
Number of Supported Neurons (max.):	74,465,280
Neural Module Chromosome Length:	91,008 bits
Information Flow Rate on Neuronal Level (max):	1.8 Gbytes/s
Information Flow Rate on Intermodular Level (max.):	64 Mbytes/s
Number of FPGAs (Xilinx XC6264BG560):	72
Number of FPGA Reconfigurable Function Units:	1,179,648
Phenotype/Genotype Memory:	1.18 Gbytes
Power Consumption:	1.5 KWatt (5V, 300 A)
Computational Power (estimated):	10,000 × Pentium III 500 MHz

Tab. 1. Summary of CBM technical parameters [by Korkin, M., personal communication, November 1999].

VII. CBM ARCHITECTURE

This section consists of a fairly detailed description of the CBM, which is used to evolve the cellular automata-based neural net circuit modules at electronic speeds, using evolvable hardware techniques. The CBM consists of the following six major blocks: (A) Cellular Automata, (B) Genotype/Phenotype Memory, (C) Fitness Evaluation Unit, (D) Genetic Algorithm Unit, (E) Module Interconnection Memory, and (F) External Interface. Each of the blocks is discussed in detail below.

A. Cellular Automata

The Cellular Automata (CA) is the hardware core of the CBM. It is intended to accelerate the speed of brain evolution through a highly parallel execution of cellular state updates. The CA consists of an array of identical hardware logic circuits arranged as a 3D structure of 24 * 24 * 24 cells (a total of 13,824 cells). Cells forming the top layer of the module are recurrently connected with the cells in the bottom layer. A similar recurrent connection is made between the cells on the north and south, as well as east and west vertical surfaces. Thus, a fully recurrent toroidal cube is formed. This feature allows a higher axonic and dendritic growth capacity by effectively doubling each of the three dimensions of the cellular space.

The CBM hardware core is time-shared between multiple modules forming a very large neural system during simulation of the system. Only one CoDi module is instantiated at a time. The FPGA firmware design is a dual-buffered structure, which allows simultaneous configuration of the next module while the current module is being run (i.e., signals are propagated through the dendrites and axons between neurons). Thus, the FPGA core is run continuously without any idle time between CoDi modules for reconfiguration.

The surfaces of the cube have external connections to provide signal input from other modules. Each surface has a matrix of 64 signals, which is repeated on the opposite surface due to wraparound connections. Thus, a total of 192 different connections is available. Four connections, one on each of the surfaces, are used as output points.

The CA is implemented with new Xilinx FPGA devices XC6264. These devices are fully and partially reconfigurable, feature a new co-processor architecture with data and address bus access, in addition to user inputs and outputs, and allow the reading and writing of any of the internal flip-flops through the data bus. An XC6264 FPGA contains 16,384 logic function cells, each cell featuring a flip-flop and Boolean logic capacity, capable of toggling at a 220 MHz rate. Logic gates are interconnected with neighbors at several hierarchical levels, providing identical propagation delay for any length of connection. This feature is very well-suited for a 3D CA space configuration. Additionally, clock routing is optimized for equal propagation time, and power distribution is implemented in a redundant manner.

To implement the Cellular Automata, a 3D block of identical logic cells is configured inside each XC6264 device, with CoDi specified 1-bit signal buses interconnecting the cells. Given the FPGA internal routing capabilities and the logic capacity needed to implement each cell, the optimal arrangement for a XC6264 is 4 * 6 * 8 (192 cells). This elementary block of cell requires 208 external connections to form a larger 3D block, by interconnecting with 6 neighbor FPGAs on the south, north, east, west, top and bottom sides in a virtual 3D space. A total of 72 FPGAs, arranged as a 6 * 4 * 3 array are used to implement a 24 * 24 * 24 cellular cube.

The CBM implements interconnections between 72 FPGAs, each placed on a small individual printed circuit board, in the form of one large back-plane board, carrying all 72 FPGA daughter boards.

The CBM clock rate for cellular update is selected between 8.25 MHz, and 9.47 MHz. At this rate, all 13,824 cells are updated simultaneously, which results in the update rate of 114 or 131 billion cells per second.

B. Genotype/Phenotype Memory

Each of the 72 FPGA daughter boards includes 16 Mbytes of EDO DRAM to be used for storing the genotypes and phenotypes of the CoDi modules, for a total of 1,188 Mbytes. There are two modes of CBM operation; namely, evolution mode and run mode. The evolution mode involves the growth phase and signaling phase. During the growth phase, memory is used to store the chromosome bit-strings of the evolving population of CoDi modules (module genotypes). For a module of 13,824 cells there ore over 91 Kbits of genotype memory needed. For each CoDi module, the genotype memory also stores information concerning the locations and orientations of the neurons inside the module, and their synaptic masks.

During the run mode, memory is used as a phenotype memory for the evolved modules. The phenotype data describes the grown axonic and dendritic trees and their respective neurons for each module. The phenotype data is loaded into the CA to configure it according to the evolved function. The genotype/phenotype memory is used to store and rapidly reconfigure (reload) the FPGAs. Reconfiguration can be performed in parallel with running a module, due to a dual pipelined phenotype/genotype register provided in each cell. This guarantees the continuous running of the FPGA array at full speed with no interruptions for reloading in either evolution or run modes. The phenotype/genotype memory can support up to 64,640 interconnected CoDi modules at a time. An additional memory will be based in the main memory of the host computer (Pentium III 500 MHz) connected to the CBM through a PCI bus, capable of transferring data at 132 Mbytes/s.

C. Fitness Evaluation Unit

Signaling in the CBM is accomplished with 1-bit spiketrains, a sequence of ones separated by intervals of zeros, similar to those of biological neural networks. Information, representing external stimuli, as well as internal waveforms, is encoded in spike-trains using a co-called "Spike Interval Information Coding (SIIC)". This method of coding is implemented by nature in animal neural networks, and is very efficient in terms of information capacity per spike. Conversion from spike-trains into "analog" waveforms representing external stimuli, or internal signaling, is accomplished by convolving the spike-train with a special multi-tap linear filter.

When a CoDi module is being evolved, it must be evaluated in terns of fitness for a targeted task. During the signaling phase, each module receives up to 188 different spike-trains, and produces up to four different output spike-trains, which is compared with a target array of spike-trains in order to guide the evolutionary process. This comparison gives a measure of performance, or fitness, of the module.

Fitness evaluation is supported by a hardware unit, which consists of an input spike-train buffer, a target spike-train buffer, and fitness evaluator. During each clock cycle, an input vector is read from its stack and fed into the module's inputs. At the same time, a target vector is read from its buffer to be compared with the current module outputs by the evaluator. The fitness evaluator performs a convolution of the spike-trains with the convolution filter, and computes the sum of the waveform's absolute deviations for the duration of the signaling phase. At the end of the signaling phase, a final measure of the module's fitness is instantly available.

D. Genetic Algorithm Unit

To evolve a CoDi module, a population of modules is evaluated by computing every module's fitness measure, as described above. A subset of the best modules is then selected for further reproduction. In each generation of modules, the best are mated and mutated to produce a set of offspring modules to become the next generation. Mating and mutation is performed by the CBM hardware core at high speed, configured for the genetic phase. During this phase, each cell's firmware implements crossover and mutation masks, two parent registers, and an offspring register. Thus, each offspring chromosome is generated in nanoseconds, directly in hardware. The selection algorithm is performed by the host computer in software, using access to the CBM via a PCI interface.

E. Module Interconnection Memory

In order to support the run mode of operation, which requires a large number of evolved CoDi modules to function as an united very large neural system, a module interconnection memory is provided. Since each CoDi module can receive inputs from up to 188 other modules, a list of these source modules referenced to each module is stored in a CBM netlist memory (64 Mbytes) by the host computer. This list is compiled by CBM software, using a module interconnection netlist in EDIF format. This netlist reflects the module interconnections as designed by the user, using off-the-shelf schematic capture tools.

The length of module interconnections is 96 cells (clock cycles). For each of the 64,640 modules, a Signal Memory stores up to four 96-bit long output spike-trains.

During the run mode, at the time each module of the system is configured in the CA hardware core (by loading its phenotype), a signal input buffer is also loaded with up to 188 spike-trains, according to the netlist in the netlist memory. The spiketrains are the signals saved from the previous instantiation and signaling of the 188 sourcing modules. At the same time, the four output spike-trains of the currently instantiated module are saved back to the Signal Memory. This repetitive cycling through all the modules which form the neural system, results in a repetitive saving and retrieving of the spike-trains to/from the Signal Memory. It provides the signaling between CoDi modules according to the brain interconnection structure reflected in the schematics, designed by the user.

In a maximum-size neural system, with 64,640 modules, the CBM update rate is such that each cell propagates approximately 120 bit-long spike-trains per second. A 120 bit-long spike-train can carry on the order of 5 bytes of signal information, using the SIIC coding method. Each neuron receives up to five spike-trains, so there are up to 372 million spiketrains per second being processed by neurons in the overall system. Thus, the maximum information-processing rate by all neurons in the system is on the order of 1.86 Gbytes/sec.

Additional spike-train processing in multiple dendritic branches can be estimated by assuming 50% of the total cellular space to be occupied by dendrite cells, each cell on average having 2.5 branches out of 5 possible. Informational throughput of dendrite cells is then on the order of 5.6 Gbyte/s.

F. External Interface

The CBM architecture can receive and send spike-trains not only from/to the Signal Memory, but also from/to the external CBM interface. Any CoDi module can also receive up to 188 incoming spike-trains and send up to four spike-trains to an external device, such as a vehicle, a speech processing system, or even an autonomous robot. In the last case, the CBM would serve as an artificial brain.The information flow rate between CBM and an external device can be sustained at up to 20 MByte/s.

VIII. CONCLUDING REMARKS

Combining neural engineering with CA and evolutionary computation gave us the CoDi technique—an extraordinary way of obtaining very large neural systems, consisting potentially of hundreds of millions of neurons. The systems are based on CoDi modules bred in a workspace of Cellular Automata (CA) using a genetic algorithm.

We have presented a formal description of the CoDi module and the behavior of some modules we evolved using a software simulator of CA. The results look quite promising, especially important is the evidence, that the CoDi module can serve as an erasable memory unit.

We have described the CBM—a special purpose hardware for rapid evolution of CoDi modules and running very large-scale multi-module systems. Because of the number of simulated neurons and the speed of the simulation, the systems can control autonomous robots or imitations of animals in near real-time regime. The potential scalability of CoDi-based systems makes them a key to human-like robots, or, one day, maybe super-humans.

We are aware that the presented technique and suggested application can be a subject to severe criticism. Our opponents may argue: What is the superiority of the CoDi technique over, say, well known backpropagation networks? Artificial neural networks can learn to recognize hand-written script, to drive cars, to make medical diagnoses... What can CoDi do today? To convert frequency onto delay, or to serve as a kind of flip-flop only. What are grounds for enthusiasm for CoDi?

Such objections resemble the situation which took place near one hundred years ago. The idea of the airplane also had its opponents. Famous scientists, including physicist Lord Kelvin or astronomer Simon Newcomb, asserted that flying machines were impossible. The latter even provided a mathematical "proof" of this. After the Wright brothers' success, the opponents could mock: What is the superiority of the airplane over, say, the old good train? A train could carry hundreds of passengers for thousands of miles, while the Wright airplane carried a single person a couple of hundred feet! It was true. But, it is also true, that it was not the good old train, but the airplane that opened the gate towards the conquest of the Cosmos. The same seems to apply to the CoDi vs. Backpropagation debate.

It may be noted, that some applications of hand-designed AI systems are more impressive than CoDi modules we have evolved so far. Nevertheless, despite the learning capacity of the systems, they have their limits. We say, neither the wonderful recognizers of hand-written script, nor hand-designed networks for artificial drivers, but rather the CoDi technique, implemented using evolvable hardware, will open the gate to an artificial thinking entity. The reason is two-fold. First, the CoDi-based systems are enormously scalable. The possibility of adding new sets of modules to working CoDi-based systems is practically unlimited. Second, let us note, that CoDi modules have their binary chromosomes, hence, any CoDi-based autonomous agent has its binary 'genome'. This means that a population of CoDi-based agents can evolve over generations. In any case, the CoDi engineering has a good chance to become a revolution in computer engineering, while the CoDi modules have a good chance to become as ubiquitous as the microprocessor today.

AKNOWLEDGEMENTS

Let us thank Prof. Dr Hugo de Garis—the inventor of the CAM-Brain Project, as well as to Dr Michael Korkin—the builder of the CBM, for reviewing of the draft of this paper, sharing with us seminal ideas, and providing valuable remarks, which strongly influenced the final text. Let us also appreciate the efforts of anonymous volunteers—members of Gdansk Artificial Brain Research Initiative (GABRI), who, using their home PCs, performed hundreds of CoDi-module evolutions, obtaining finally some of the results we have quoted.

REFERENCES

[1] Buller A & de Garis H (1998) Brain Building Strategy: Some Remarks and Questions, *Intelligent Information Systems VII, Proceedings of the Workshop, Malbork, Poland, June 15-19*, 188-193.

[2] Buller A (1999) Fuzziness, Mega-networks, and CoDi Technique, *4th Conference on Neural Networks and Their Applications, May 18-22, Zakopane, Poland.*

[3] de Garis H (1995) "CAM-Brain": ATR's Billion Neuron Artificial Brain Project. A Three Year Progress Report, in: Furuhashi T & Uchikawa Y (Eds.) *Fuzzy Logic, Neural networks, and Evolutionary Computation*, Springer: Berlin, 215-243.

[4] de Garis, Nawa NE, Buller A, Korkin M, Gers F & Hough (1999) Evolution of Neural Network Modules: ATR's Artificial Brain Project, in: Jain LC (Ed.) *Evolution of Engineering and Information Systems and Their Applications*, CRC Press: Boca Raton, 231-274.

[5] Gers F, de Garis H & Korkin M (1997) CoDi-1Bit: A Simplified Cellular Automata Based Neuron Model, *Evolution Artificiele 97, 22 Octobre 1997, Nimes, France*, 211-229.

[6] Goldberg DE (1989) *Genetic algorithms in search, optimization, and machine learning*, Addison-Wesley: New York.

[7] Hough M, de Garis H, Korkin M, Gers F & Nawa NE (1999) SPIKER: Analog Waveform to Digital Spiketrain Conversion in ATR's Artificial Brain (CAM-Brain) Project, *Proc. of the 4th Intl. Symposium on Artificial Life and Robotics (AROB'99), January 19-22, Oita, Japan*, 610-613.

[8] Jo Y-G, Kang J (1999) Cellular Automata-Based Neural Networks (CABANN) for Optimal Path Planning, *Proc. of the Fourth Int. Symp. On Artificial Life and Robotics (AROB'99), Oita, Japan, January 19-22*, 220-223.

[9] Korkin M, de Garis H, Nawa NE & Rieken WD (1999) ATR's Artificial Brain Project: CAM-Brain Machine (CBM) and Robot Kitten (Robokoneko) Issues, in: Dobnikar A, Steele NC, Pearson DW & Albrecht R (Eds.) *Artificial Neural Nets and Genetic Algorithms*, Springer: Wien, 107-110.

[10] Lee D-W, Sim K-B (1999) Evolving Cellular Automata Neural System 1 (ECANS 1), *Proc. of the Third Asian Fuzzy System Symp., June*, 158-163.

[11] Lee D-W, Sim K-B (1999) Evolving Cellular Automata Neural Systems 2, *Proc. of the Fourth Int. Symp. On Artificial Life and Robotics (AROB'99), Oita, Japan, January 19-22*, 224-227.

[12] Maass W & Bishop CM (Eds.)(1999) *Pulsed Neural Networks*, A Bradford Book/The MIT Press: Cambridge, Mass.

[13] Rieke F, Warland D, de Ruyter van Steveninck & Bialek W (1997) *Spikes: Exploring the Neural Code*, A Bradford Book/The MIT Press: Cambridge, Mass.

[14] Shimohara K & Hemmi H (1999) [Evolving Artificial Brain] (in Japanese) *Computer Today*, 4, 4-9.

[15] Toffoli T & Margolus N (1987) *Cellular Automata Machines. A new environment for modeling*, The MIT Press: Cambridge, Mass.