Modular Cellular Neural Network Structure for Wave-Computing-Based Image Processing

Mojtaba Karami, Reza Safabakhsh, and Mohammad Rahmati

This paper introduces the modular cellular neural network (CNN), which is a new CNN structure constructed from nine one-layer modules with intercellular interactions between different modules. The new network is suitable for implementing many image processing operations. Inputting an image into the modules results in nine outputs. The topographic characteristic of the cell interactions allows the outputs to introduce new properties for image processing tasks. The stability of the system is proven and the performance is evaluated in several image processing applications. Experiment results on texture segmentation show the power of the proposed structure. The performance of the structure in a real edge detection application using the Berkeley dataset BSDS300 is also evaluated.

Keywords: Cellular neural network (CNN), modular cellular neural network (MCNN), wave computing, diffusion, trigger wave, edge detection.

Manuscript received Apr. 21; revised Oct. 8, 2012; accepted Oct. 26, 2012.

Mojtaba Karami (phone: +98 912 398 4662, m.karami@aut.ac.ir), Reza Safabakhsh (corresponding author, safa@aut.ac.ir), and Mohammad Rahmati (rahmati@aut.ac.ir) are with the Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran.

http://dx.doi.org/10.4218/etrij.13.0112.0107

I. Introduction

Computational intelligence approaches such as neural networks and fuzzy systems have been used to model dynamic phenomena and applications in different areas. They have proven to be powerful and effective. Several of the more recent works that used these approaches are [1]-[10].

Cellular neural networks (CNNs), introduced by Chua and Yang in 1988 [11], [12], consist of a grid of cells with nearest neighbor interconnections. Based on the CNN structure, the CNN universal machine (CNN-UM) architecture was proposed by Roska and Chua [13], which is suitable for VLSI implementation and is a powerful tool in parallel processing for signal and image processing applications [14], [15]. The CNN-based parallel computing relies on analog signals and connections using templates. This computing approach leads to a processing method called "cellular wave computing" [16].

Since the fully stored programmable analogic CNN-UM was introduced, spatiotemporal continuous nonlinear dynamics have been implemented on CNNs, and nonlinear partial differential equations (PDEs) have become applicable, achieving new results for different applications based on wave computing on the CNN-UM (a cellular wave computer [17]). The CNN-UM can be programmed as an analogic computing device by analogic algorithms, using analog operations in sequence combined with local logic at the cell level [16]-[18]. CNN-UM-based wave computing is a new kind of logic or reasoning that is applicable in parallel and allows researchers to study natural phenomena in diverse fields of physics, chemistry, biology, and so on. This kind of processing, unlike the Boolean logic, is a spatiotemporal logic defined by spatiotemporal patterns [16]. Using this method, new types of algorithms can be utilized for robot navigation [19], automated detection of a preseizure state [20], target detection [21], object comparison [22], learning of spatiotemporal behavior [23], and auditory scene analysis [24]. In some applications, the method shows the ability to mimic physical phenomena such as in the artificial retina model [25], [26].

The autowave principle for parallel image processing was proposed in [27] and autowave for image processing on two-dimensional CNN was introduced in [28]. Solving such PDEs as reaction diffusion type systems and systems of ordinary differential equations (ODEs) by CNN has been investigated [29], [30]. Diffusion-based image processing as a PDE type of autowave computing based on CNN was discussed by Rekeczky [31]. He investigated PDE-based (constrained linear and nonlinear) diffusion models and a non-PDE-based diffusion model and introduced analogic algorithms for segmentation and edge detection on CNNs. He also investigated the qualitative properties of a trigger wave as a computational tool and its capability in segmentation and shape and structure detection [32].

Most of the wave computing research carried out on CNNs is based on a single-layer CNN. There are, however, a few studies based on the multilayer CNN architecture. Majorana and Chua proposed a unified notation for multilayer and higher-order CNNs to improve the understanding and applications of this architecture [33]. Balya and others proposed a three-layer CNN structure for modeling the mammalian retina model and discussed the stability of their structure [25]. Yang and others presented mutually coupled two-layer CNNs for image processing applications. They showed the usefulness of their structure for center point detection and skeletonization and discussed its stability [34]. Shi proposed the eight-layer CNN architecture to implement spatiotemporal filters [35].

Previous cellular wave computing studies based on a single-layer CNN are called "wave computing algorithms" [17]. Each step in these algorithms can be seen as a CNN architecture that receives the input and initial state from its previous steps and generates an output for use as the input or initial state for the next steps (CNNs). The CNN has cell interactions within itself, but there is no intercell interaction between CNN modules in the stages of an algorithm.

In this paper, a new CNN structure that is useful for feature extraction from images is proposed. The proposed structure has a new topography of interactions between cells in different modules. Based on the topographic interactions, the new structure is referred to as the modular CNN (MCNN). The new structure achieves desirable results in several new image processing operations. Stability analysis of dynamic systems has been investigated using different approaches [36]-[39]. The stability condition of the MCNN is discussed using eigenvalues of the coefficient matrix of the system in the discrete space Fourier transform (DSFT).

In the next section, the CNN and wave computing are briefly introduced. The proposed structure, the MCNN, is presented in section III. The stability of the MCNN is discussed in section IV. Some examples from the experiment results are presented in section V, and the conclusions are summarized in section VI.

II. Cellular Neural Network and Wave Computing

A standard CNN architecture is a continuous-time network of locally interconnected similar dynamic cells. The cells $C(i,j)$ are arranged in an $M\times N$ rectangular array with Cartesian coordinates (i,j), $i=1, 2,\ldots, M$, $j=1, 2,\ldots, N$, to form a one-layer CNN (Fig. 1). Neighbors of cell $C(i, j)$ that are connected to $C(i,j)$ form a set $N_{ij}(r)$ specified by

$$N_{ij}(r)=\{C(k, l)\ |\max(|k-i|, |l-j|)\le r,\ 1\le k\le M;\ 1\le l\le N\},$$
$$i=1, 2,\ldots, M,\quad j=1, 2,\ldots, N,$$

where r is a positive integer (Fig. 1).

A cell is a dynamical nonlinear system with a state equation:

$$\begin{aligned}\frac{d}{dt}x_{ij} &= -x_{ij}+\sum\nolimits_{C(k,l)\in N_{ij}(r)} A(i,j,k,l)y_{kl}\\ &\quad +\sum\nolimits_{C(k,l)\in N_{ij}(r)} B(i,j,k,l)u_{kl}+z_{ij},\\ y_{ij} &= f(x_{ij})=\frac{1}{2}|x_{ij}+1|-|x_{ij}-1|,\end{aligned} \tag{1}$$

where u_{ij}, x_{ij}, and y_{ij} are the input, state, and output voltage of the cell (i, j) in the CNN grid, respectively. The feedback matrix A, control matrix B, and z can be space invariant. In this case, the fixed templates A, B, and z are used for a specific task and $\{A, B, z\}$ is called the cloning template.

The CNN architecture and dynamic behavior of its cells show the capability of the structure to mimic the brain operation. Based on this, some principles were extracted from neuroscience, genetics, and immunology that have been used in CNNs to solve several difficult problems. These principles include the twin wave principle, push-pull principle, immune response inspired principle, embedded grammar principle, selective spatial modulation principle, and fusion of multimodal features [40].

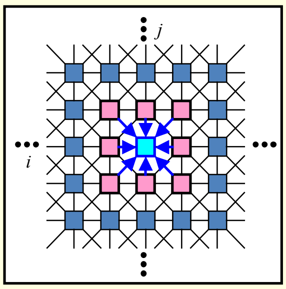

Fig. 1. CNN structure and indicated cell with coordinates (i, j) and its neighbors in sphere with radius $r=1$.

The behavior of dynamic phenomena can be described by continuous time and space PDEs. To solve these PDEs using CNN, a PDE is approximated by a grid of cells in which the behavior of each cell is characterized by ODEs. Each cell works as an ODE solver with continuous state and time and has a sphere of influence of cells. Therefore, the PDE is discretized in space with continuous time. This structure of cells is similar to the nervous organs in creatures.

Wave computing is based on solving the PDE defining a dynamic phenomenon. This point indeed demonstrates the difference between logic computation and wave computing. In digital computers, algorithms are defined on integers, whereas, in wave computing, algorithms are defined on the solution of a nonlinear wave equation (typically, a reaction diffusion equation). The formal definition of an algorithm in a cellular wave computer was introduced by Roska [40].

The reaction-diffusion type of PDE used by Rekeckey and others for image processing [41] is in the form of

$$\frac{\partial \varnothing(x,y,t)}{\partial t} - \mathrm{div}(\mathrm{grad}(\varnothing(x,y,t))) = \Im_1(\varnothing_0(x,y,t_0)) + \Im_2(\varnothing(x,y,t)), \tag{2}$$

where $\varnothing$ is the image intensity, $\varnothing_0$ is the initial state, and $\Im_1(.)$ and $\Im_2(.)$ are nonlinear functions.

Subclasses of the reaction-diffusion type of PDE can be obtained based on different choices of the right-hand side of the equation. If the right-hand side is zero, the linear diffusion equation is obtained by ignoring $\Im_2(.)$, and the constrained linear diffusion equation is obtained. If $\Im_2(.) = \mathrm{sigm}(.)$, the trigger wave equation is obtained, and, if $\Im_2(.) = \mathrm{sigm}(.)$ and $\Im_1(.) \neq 0$, the constrained trigger wave equations will be obtained.

To map the above PDE on the CNN, the following reaction-diffusion type of nonlinear ODE was used in [40]-[42].

$$\frac{d\phi_{ij}(t)}{dx} = g(\phi_{ij}(t)) - \phi_{ij}(t)$$
$$+\frac{c_1}{4}(\phi_{i-1j}(t) + \phi_{i+1j}(t) + \phi_{ij-1}(t) + \phi_{ij+1}(t)) + z_{ij}, \tag{3}$$
$$\phi_{ij}(t) = f(x_{ij}(t)), g(.) = c_0 f(.), z_{ij} = z_0 + \sum_{kl\in N} b_{kl}\phi_{kl}(t_0),$$

where z_{ij} and $g(.)$ take the places of $\Im_1(.)$ and $\Im_2(.)$ in (2), respectively. Here, N defines the nearest neighbors for the cell (i, j). By using z_{ij}=0 and $f(\phi)=\phi$, the linear diffusion equation is obtained. By setting $z_{ij}\neq 0$ and $f(\phi)=\phi$, the constrained linear diffusion equation is obtained. By using z_{ij}=0 and $f(\phi)=\mathrm{sigm}(\phi)$, the nonlinear trigger wave equation is obtained. By setting $z_{ij}\neq 0$ and $f(\phi)=\mathrm{sigm}(\phi)$, the constrained nonlinear trigger wave equation is obtained.

The equivalent CNN templates for the diffusion filter and trigger wave have the following form:

$$A = \begin{bmatrix} 0 & a_1 & 0 \\ a_1 & a_0 & a_1 \\ 0 & a_1 & 0 \end{bmatrix}, B = \begin{bmatrix} b_2 & b_1 & b_2 \\ b_1 & b_0 & b_1 \\ b_2 & b_1 & b_2 \end{bmatrix}, z_0, \tag{4}$$

Diffusion filter: a_0=0 and a_1>0; Trigger wave: a_0>a_1>0.

The diffusion wave and trigger wave have been used in some applications, such as texture segmentation and detection, change detection in video flow, object comparison, contrast enhancement, noise suppression, and shape enhancement [43].

III. Proposed CNN Structure (MCNN)

Existing research on cellular wave computing uses the CNN as a computing device that runs cellular wave computing algorithms. Each step in these algorithms can be seen as one module (CNN), which receives its input and initial state from the previous steps and generates output for the next steps (CNNs). The CNNs have been used as computing algorithms in a wave computing algorithm with cell interactions within itself but without intercell interactions among CNN modules in the various stages of an algorithm.

In this paper, we propose new topographic cell interactions among modules, which results in new capabilities for the wave computing structure. The proposed MCNN consists of nine one-layer modules arranged in a plane. Interactions between cells in different modules are defined in a way that the outputs of the nine modules are very similar except for slight differences in some directions. Comparing these nine images, one can see the differences as a shift in diffusion. Based on this, the output images are called "directed diffusion."

The nine modules of this system produce nine images. Every module consists of a grid of dynamic cells, each of which has interactions with the cells in its four neighboring modules. Considering circular neighborhoods, the neighboring modules for module (−1, −1) are (−1, 0), (−1, +1), (+1, −1), and (0, −1), as illustrated in Fig. 2. Here, the self-feedback coefficient is a_0

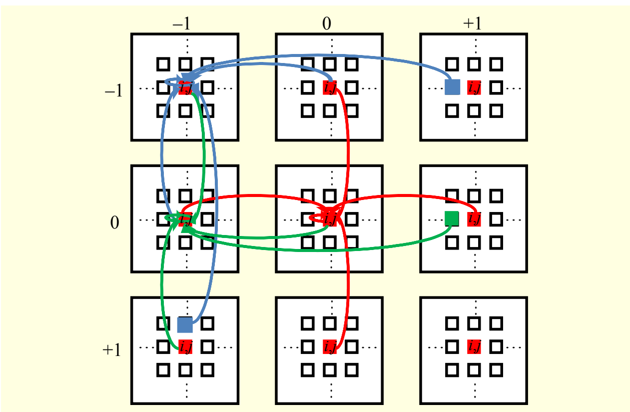

Fig. 2. Structure of MCNN.

and the coefficients of the other four pieces of feedback are a_1, as in the reaction diffusion filters. The equation giving the dynamics of each cell (i, j) in module (0, 0) is

$$\begin{aligned}\frac{d}{dt}x_{0,0}^{t}(i,j) &= -x_{0,0}^{t}(i,j)+a_0 y_{0,0}^{t}(i,j)\\ &\quad +a_1(y_{0,-1}^{t}(i,j)+y_{-1,0}^{t}(i,j)+y_{0,+1}^{t}(i,j)+y_{+1,0}^{t}(i,j)),\\ i&=1,\ldots,N,\ j=1,\ldots,M,\\ y_{m,n}^{t}(i,j) &= f(x_{m,n}^{t}(i,j))=\frac{1}{2}|x_{m,n}^{t}(i,j)+1|-|x_{m,n}^{t}(i,j)-1|,\end{aligned} \tag{5}$$

where $x_{m,n}^{t}(i,j)$ and $y_{m,n}^{t}(i,j)$ indicate the state and output of the cell (i, j) in module (m, n), respectively, and t denotes time. Coefficients a_0 and a_1 are the self and neighboring cell feedback coefficients, respectively.

For other modules, the neighborhoods can be assumed vertically and horizontally circular and each cell in a module receives feedback from cells belonging to modules in its circular neighborhood. Depending on the coordinates of a circular neighboring module, there exists a displacement in coordinates of the cell belonging to the circular module that issues feedback. Suppose that $C_K(i, j)$ indicates a cell with coordinates (i, j) belonging to module K. If module K has a left (or right) circular module L, then $C_K(i, j)$ receives feedback from the cells $C_L(i, j–1)$ (or $C_L(i, j+1)$). If K has an up (or down) circular module L, then $C_K(i, j)$ receives feedback from the cells $C_L(i+1, j)$ (or $C_L(i–1, j)$) (Fig. 2).

The dynamics of each cell (i, j) in module (–1, 0) is given by

$$\begin{aligned}\frac{d}{dt}x_{0,-1}^{t}(i,j) &= -x_{0,-1}^{t}(i,j)+a_0 y_{0,-1}^{t}(i,j)+a_1(y_{0,+1}^{t}(i,j-1)\\ &\quad +y_{1,-1}^{t}(i,j)+y_{0,0}^{t}(i,j)+y_{+1,-1}^{t}(i,j)),\\ i&=1,\ldots,N,\ j=1,\ldots,M,\end{aligned} \tag{6}$$

and, in module (–1, –1), is presented by

$$\begin{aligned}\frac{d}{dt}x_{-1,-1}^{t}(i,j) &= -x_{-1,-1}^{t}(i,j)+a_0 y_{-1,-1}^{t}(i,j)+a_1(y_{-1,+1}^{t}(i,j-1)\\ &\quad +y_{+1,-1}^{t}(i-1,j)+y_{-1,0}^{t}(i,j)+y_{0,-1}^{t}(i,j)),\\ i&=1,\ldots,N,\ j=1,\ldots,M.\end{aligned} \tag{7}$$

The general equation can therefore be derived as

$$\begin{aligned}\frac{d}{dt}x_{m,n}^{t}(i,j) &= -x_{m,n}^{t}(i,j)+(a_0-2a_1)y_{m,n}^{t}(i,j)\\ &\quad +a_1\left(\sum_{k=-1}^{+1} y_{m,n}^{t}\left(i-\frac{|m||m-k|k}{2},j\right)\right.\\ &\quad \left.+\sum_{l=-1}^{+1} y_{m,n}^{t}\left(i,j-\frac{|n||n-l|l}{2}\right)\right),\\ m,n&=-1,0,+1 \quad i=1,\ldots,N,\ j=1,\ldots,M,\end{aligned} \tag{8}$$

where (i, j) gives the cell coordinates in a module, (m, n) gives the module coordinates in the system plane, and t denotes time.

IV. Stability Analysis of MCNN

The stability of the MCNN is proved in this section using the DSFT. To eliminate the argument displacements in the term $x_{m,n}^{t}(i,j)$ of (7), we will use the two-dimensional DSFT used by Crouse and others [44]. Using the definition and boundary assumption proposed by Crouse and others and using the activation function $y_{m,n}^{t}(i,j)=x_{m,n}^{t}(i,j)$, the DSFT of (8) is

$$\begin{aligned}\frac{d}{dt}\boldsymbol{X}_{m,n}^{t}(\omega_1,\omega_2) &= (a_0-2a_1-1)\boldsymbol{X}_{m,n}^{t}(\omega_1,\omega_2)\\ &\quad +a_1\left(\sum_{k=-1}^{+1}\boldsymbol{X}_{m,n}^{t}(\omega_1,\omega_2)e^{-j\omega_1\frac{|m||m-k|k}{2}}\right.\\ &\quad \left.+\sum_{l=-1}^{+1}\boldsymbol{X}_{m,n}^{t}(\omega_1,\omega_2)e^{-j\omega_2\frac{|n||n-l|l}{2}}\right),\\ -\pi&\le\omega_1,\omega_2\le+\pi,\end{aligned} \tag{9}$$

where $\boldsymbol{X}_{m,n}^{t}(\omega_1,\omega_2)$ is the DSFT of $x_{m,n}^{t}(i,j)$.

To change the above equation to matrix differential equation form, the following definitions will be used:

$$\boldsymbol{X}(\omega_1,\omega_2)=\begin{bmatrix}\boldsymbol{X}_{-1,-1}(\omega_1,\omega_2)\\ \boldsymbol{X}_{-1,0}(\omega_1,\omega_2)\\ \boldsymbol{X}_{-1,+1}(\omega_1,\omega_2)\\ \boldsymbol{X}_{0,-1}(\omega_1,\omega_2)\\ \boldsymbol{X}_{0,0}(\omega_1,\omega_2)\\ \boldsymbol{X}_{0,+1}(\omega_1,\omega_2)\\ \boldsymbol{X}_{+1,-1}(\omega_1,\omega_2)\\ \boldsymbol{X}_{+1,0}(\omega_1,\omega_2)\\ \boldsymbol{X}_{+1,+1}(\omega_1,\omega_2)\end{bmatrix},$$

$$W=a_1\begin{bmatrix}\frac{a_0-1}{a_1} & 1 & e^{-j\omega_2} & 1 & 0 & 0 & e^{-j\omega_1} & 0 & 0\\ 1 & \frac{a_0-1}{a_1} & 1 & 0 & 1 & 0 & 0 & e^{-j\omega_1} & 0\\ e^{+j\omega_2} & 1 & \frac{a_0-1}{a_1} & 0 & 0 & 1 & 0 & 0 & e^{-j\omega_1}\\ 1 & 0 & 0 & \frac{a_0-1}{a_1} & 1 & e^{-j\omega_2} & 1 & 0 & 0\\ 0 & 1 & 0 & 1 & \frac{a_0-1}{a_1} & 1 & 0 & 1 & 0\\ 0 & 0 & 1 & e^{+j\omega_2} & 1 & \frac{a_0-1}{a_1} & 0 & 0 & 1\\ e^{+j\omega_1} & 0 & 0 & 1 & 0 & 0 & \frac{a_0-1}{a_1} & 1 & e^{-j\omega_2}\\ 0 & e^{+j\omega_1} & 0 & 0 & 1 & 0 & 1 & \frac{a_0-1}{a_1} & 1\\ 0 & 0 & e^{+j\omega_1} & 0 & 0 & 1 & e^{+j\omega_2} & 1 & \frac{a_0-1}{a_1}\end{bmatrix}.$$

Using this notation, (9) is transformed to

$$\frac{d}{dt}\boldsymbol{X}^{t}(\omega_1,\omega_2)=W.\boldsymbol{X}^{t}(\omega_1,\omega_2). \tag{10}$$

The behavior of the system depends upon the eigenvalues of matrix W [44]. The procedure is to determine the eigenvalues and eigenvectors and use them to construct the general solution. The solution of (10) is in the exponential form [45]:

$$\boldsymbol{x}^t(\omega_1, \omega_2) = e^{Wt} \cdot \boldsymbol{x}^0, \tag{11}$$

where $\boldsymbol{x}^0 = \boldsymbol{x}^0(\omega_1, \omega_2)$.

The exponential can be calculated using the infinite series representation [45]:

$$e^{Wt} = \sum_{k=0}^{+\infty} \frac{(Wt)^k}{k!}, \tag{12}$$

where W is a Hermitian matrix [46] that can be diagonalized as $W=D\Lambda D^*$, where $\Lambda=\mathrm{diag}(\lambda_1, \lambda_2,\ldots, \lambda_n)$, λ_i is the eigenvalue, D constitutes a set of eigenvectors of W and $W^k=D\Lambda^k D^*=D\,\mathrm{diag}(\lambda^k_1, \lambda^k_2,\ldots, \lambda^k_n)\, D^*$, and * denotes the conjugate transpose. Thus,

$$e^{Wt} = \sum_{k=0}^{+\infty} \frac{(Wt)^k}{k!} = \sum_{k=0}^{+\infty} \frac{(D\Lambda D^* t)^k}{k!} = D\sum_{k=0}^{+\infty} \frac{(\Lambda t)^k}{k!} D^* .$$

In other words, we do not have to use the infinite series to calculate e^{Wt}. Instead, we define

$$e^{\Lambda t} = \mathrm{diag}(e^{\lambda_1 t}, e^{\lambda_2 t},\ldots, e^{\lambda_n t}) = \begin{bmatrix} e^{\lambda_1 t} & & & 0 \\ & e^{\lambda_2 t} & & \\ & & \ddots & \\ 0 & & & e^{\lambda_n t} \end{bmatrix},$$

and thus (12) can be written as $e^{Wt}=De^{\Lambda t}D^*$.

The eigenvalues of W are real and its eigenvectors are completely orthonormal [46]. As a result, the solution of the system of (10) consists of nine functions in the Fourier domain. To prove the stability of the system, the following stability criterion is used.

Let $u'=Au$, $u(0)=c$, where A is diagonalizable with eigenvalues λ_i. If $Re(\lambda_i)<0$ for each i, then $\lim_{t\to\infty} e^{At}=0$, and $\lim_{t\to\infty} u(t)=0$ for every initial vector c. In this case, $u'=Au$, $u(0)=c$ is said to be a stable system, and A is called a stable matrix [46].

Since W is a Hermitian matrix, it has n real eigenvalues. To prove the stability of the system, we first determine the eigenvalues of W and then determine the conditions required to make the eigenvalue negative.

1. Eigenvalues of Matrix W

Matrix W can be rewritten in the Kronecker sum form [47] by defining A_3 and B_3 as

$$A_3 = a_1 \begin{bmatrix} 0 & 1 & e^{-j\omega_1} \\ 1 & 0 & 1 \\ e^{+j\omega_1} & 0 & 1 \end{bmatrix},$$

$$B_3 = a_1 \begin{bmatrix} \frac{a_0-1}{a_1} & 1 & e^{-j\omega_2} \\ 1 & \frac{a_0-1}{a_1} & 1 \\ e^{+j\omega_2} & 1 & \frac{a_0-1}{a_1} \end{bmatrix}, \quad I_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Then, the Kronecker products $I_3 \otimes B_3$ and $A_3 \otimes I_3$ are

$$I_3 \otimes B_3 = \begin{bmatrix} B_3 & 0 & 0 \\ 0 & B_3 & 0 \\ 0 & 0 & B_3 \end{bmatrix}, \quad A_3 \otimes I_3 = \begin{bmatrix} 0 & I_3 & e^{-j\omega_1} I_3 \\ I_3 & 0 & I_3 \\ e^{+j\omega_1} I_3 & I_3 & 0 \end{bmatrix},$$

and W is the Kronecker sum of A_3 and B_3 as

$$W = A_3 \oplus B_3 = A_3 \otimes I_3 + I_3 \otimes B_3.$$

If the eigenvalues of the matrices A_3 and B_3 are λ_i, $i=1,\ldots, 3$, and μ_j, $j=1,\ldots, 3$, respectively, then the eigenvalues of the Kronecker sum of A_3 and B_3 can be calculated as

$$\lambda_{kl} = \lambda_k + \mu_l, \; k, \; l = 1,\ldots,3. \tag{13}$$

Based on the above, nine eigenvalues can be calculated using the eigenvalues of the matrices A_3 and B_3. The eigenvalues of matrix A_3 can be calculated as

$$(A_3 - \lambda I_3) = \begin{bmatrix} -\lambda & a_1 & a_1 e^{-j\omega_1} \\ a_1 & -\lambda & a_1 \\ a_1 e^{+j\omega_1} & a_1 & -\lambda \end{bmatrix} = a_1 \begin{bmatrix} -\frac{\lambda}{a_1} & 1 & e^{-j\omega_1} \\ 1 & -\frac{\lambda}{a_1} & 1 \\ e^{+j\omega_1} & 1 & -\frac{\lambda}{a_1} \end{bmatrix}.$$

By defining x as

$$x = -\frac{\lambda}{a_1} \Rightarrow -a_1 x \tag{14}$$

and calculating the determinant of ($A_3- \lambda I_3$), a cubic root function is obtained:

$$| A_3 - \lambda I_3 | = a_1.(x^3 - 3x + 2\cos(\omega_1)) = 0 .$$

If we use the Chebyshev cube roots, (14) can be defined as a root (depending on t) of the following polynomial equation:

$$a_1 \neq 0, \quad x^3 - 3x = t, \quad t = -2\cos(\omega_1).$$

Then, the Chebyshev roots are given as

$$\begin{cases} x_1 = 2\cos\left(\dfrac{\pi - \omega_1}{3}\right), \\ x_2 = -2\cos\left(\dfrac{2\pi - \omega_1}{3}\right), \\ x_3 = 2\cos\left(\dfrac{3\pi - \omega_1}{3}\right). \end{cases}$$

By substituting these results in (14), the closed form relation for eigenvalues of matrix A_3 can be obtained as follows:

$$\lambda_k = -2a_1.(-1)^{k+1}.\cos\left(\frac{k\pi-\omega_1}{3}\right), k=1,...,3\,. \tag{15}$$

To obtain the eigenvalues of B_3, the determinant of $(B_3–\mu I_3)$ should be calculated as follows:

$$(B_3-\mu I_3)=a_1\begin{bmatrix}\frac{(a_0-1)-\mu}{a_1} & 1 & e^{-j\omega_2}\\ 1 & \frac{(a_0-1)-\mu}{a_1} & 1\\ e^{+j\omega_2} & 1 & \frac{(a_0-1)-\mu}{a_1}\end{bmatrix}.$$

By defining x as

$$x=\frac{(a_0-1)-\mu}{a_1}\Rightarrow \mu=(a_0-1)-a_1x \tag{16}$$

and calculating the determinant of $(B_3–\mu I_3)$, a cubic root function results:

$$|B_3-\mu I_3| = a_1^3.(x^3-3x+2\cos(\omega_2))=0.$$

Using the method utilized for matrix A_3, we have

$$a_1\neq 0, \quad x^3–3x=t, \quad t=–2\cos(\omega_2).$$

Then, the Chebyshev roots are given as

$$\begin{cases} x_1 = 2\cos\left(\frac{\pi-\omega_2}{3}\right),\\ x_2=-2\cos\left(\frac{2\pi-\omega_2}{3}\right),\\ x_3=2\cos\left(\frac{3\pi-\omega_2}{3}\right).\end{cases}$$

By using these results as substitutes in (16), the eigenvalues of matrix B_3 can be determined as follows:

$$\mu_l=(a_0-1)-2a_1.(-1)^{l+1}.\cos\left(\frac{l\pi-\omega_2}{3}\right), l=1,...,3\,. \tag{17}$$

By using (15) and (17) as substitutes in (13), the eigenvalues of matrix W are then given as

$$\lambda_{kl}=(a_0-1) -2a_1\left\{(-1)^{k+1}.\cos\left(\frac{k\pi-\omega_1}{3}\right)+(-1)^{l+1}.\cos\left(\frac{l\pi-\omega_2}{3}\right)\right\}, \quad k, l=1,\dots,3. \tag{18}$$

2. Stability Based on Eigenvalues of Matrix W

Let us denote $\alpha=(-1)^{k+1}\cdot\cos(k\pi–\omega_1)/3$ and $\beta=(-1)^{l+1}\cdot\cos(l\pi–\omega_2)/3$. Then, we have $|\alpha|$, $|\beta|\leq 1$. Using the inequality $\alpha+\beta\leq|\alpha|+|\beta|$ and assuming $a_1>0$, we have

$$\lambda_{ij}=(a_0-1)-2a_1\{\alpha+\beta\} \geq (a_0-1)-2a_1\{|\alpha|+|\beta|\}\geq(a_0-1)-4a_1.$$

As mentioned, if $Re(\lambda_{kl})<0$, then $\lim_{t\to\infty} e^{Wt}=0$ and $\lim_{t\to\infty} u(t)=0$, resutling in the stability of the system. Therefore, the criterion that must be satisfied for the stability of the MCNN is $0>\lambda_{kl}\geq(a_0–1)–4a_1$, which results in

$$a_0 < 4a_1+1. \tag{19}$$

By setting $a_0=0$, the stability criterion is reduced to $a_1>0$ and the operation of the system is referred to as directed diffusion.

$$\text{Directed diffusion: } a_0=0 \text{ and } a_1>0, \tag{20}$$

$$\text{Trigger wave: } a_0<4a_1+1 \text{ and } a_1>0. \tag{21}$$

The experiment results presented in the next section show the capability of the MCNN in wave computing algorithms for concavity detection, texture segmentation, and edge detection.

V. Applications

In all experiments of this section, the MCNN is used to process an image, used as input, and produces nine images that are used for further computing by CNNs in specific wave computing algorithms. This section demonstrates the potentials of the proposed MCNN-based wave computing algorithms for texture segmentation and edge detection.

1. Texture Segmentation Using Twin Wave Principle

The performance of the MCNN in texture segmentation is shown in this subsection. Considering the directions inherent in the MCNN structure, this system can be used to segment texture in four directions: horizontal, vertical, 45°, and –45°. The texture presented in Fig. 3(a) shows an example of this application. Figure 3(b) shows the output of the presented algorithm.

The difference images will be used to emphasize the slight differences between the center image and the other images. We set the MCNN parameters according to (21) (trigger wave), as $a_0=1$ and $a_1=0.25$. Then, we use the image in Fig. 3(a) as the input and initial state of the nine modules in the MCNN architecture and calculate the eight difference images from the nine shifted diffusion images of the MCNN.

The eight images are used as the input and initial state of eight standard CNNs to calculate the diffusion of each corresponding image. The vertical and horizontal differences between diffusion images from the eight CNNs can be used for textures with horizontal and vertical patterns in the input image. Two images result from this computation (Fig. 4). We use these images as excitation and inhibition waves (Fig. 5) and use the twin wave principle [40] to segment the area into the target areas and the non-target areas. By computing the subtraction of excitation and inhibition waves and using the recall computation in the next CNN, the segmentation of the area is completed.

Figure 6 compares the results of the proposed approach and the methods discussed in [48], [49]. It is evident that the proposed method produces better results: the character boundaries are smoother and they are separated more accurately. However, the result of the proposed method contains four spurious spots.

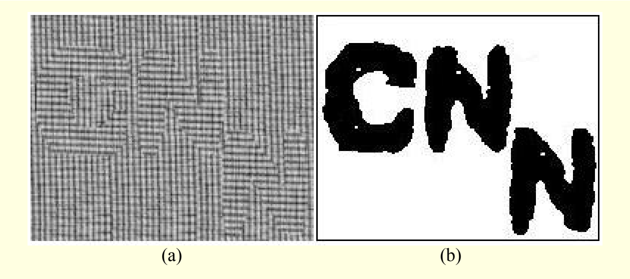

Fig. 3. Texture segmentation: (a) input image and (b) resulting image.

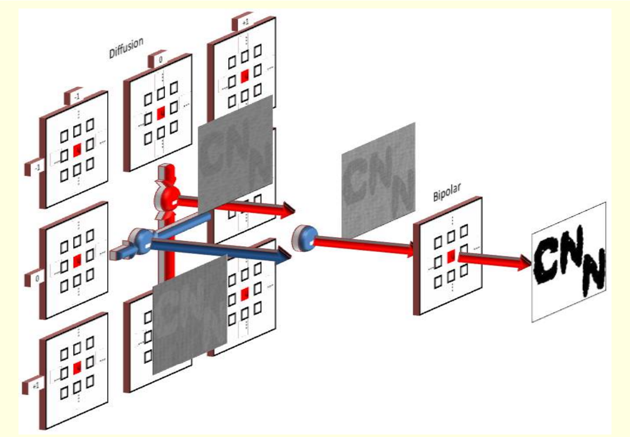

Fig. 4. Wave computing algorithm for texture segmentation using twin wave principle.

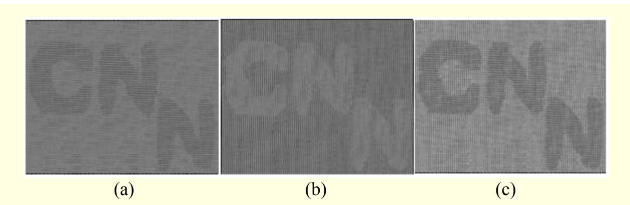

Fig. 5. (a) Excitation wave, (b) inhibition wave, and (c) excitation and inhibition difference.

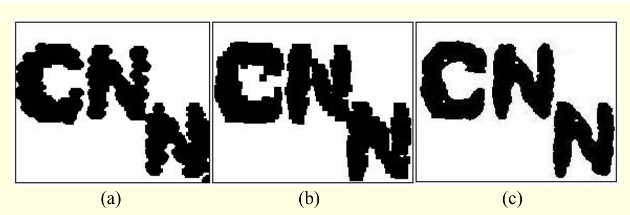

Fig. 6. Result from (a) reference [49], (b) reference [48] based on morphological computation on CNN, and (c) proposed approach based on MCNN.

2. Edge Detection

The features in the outputs of the MCNN can be used for edge detection. The algorithm for this operation is shown in Fig. 7. In the first stage of the algorithm, the eight difference images are computed using the outputs of the MCNN. As illustrated in Fig. 7, the eight difference outputs from the previous stage are used as the inputs to eight standard CNNs using the trigger template. The trigger output images are fed to eight standard CNNs to run erosion, and the output of this stage sums up to produce the final output as the boundary of the input image.

To show the power of the proposed edge detection algorithm, we compare it with the methods presented in [50] and [51] and use the Berkeley segmentation dataset (BSDS300) [52]. This dataset consists of 200 training and 100 testing images, each with multiple ground-truth boundaries marked by different humans. This dataset has also been used for evaluating other new boundary detection and segmentation algorithms [53]-[55].

We set the MCNN parameters according to (20), as a_0=0 and a_1=0.25, and run the wave computing algorithm of Fig. 7 on the 100 test images used as the input and initial state for the MCNN. To evaluate the proposed method, the precision-recall curve is obtained using the Berkeley benchmark with 50 levels for the threshold (Fig. 8).

The edge detection method proposed by Hernandez and others [51] finds an optimal threshold for the input image. Then, the threshold and the black and white edge detection templates

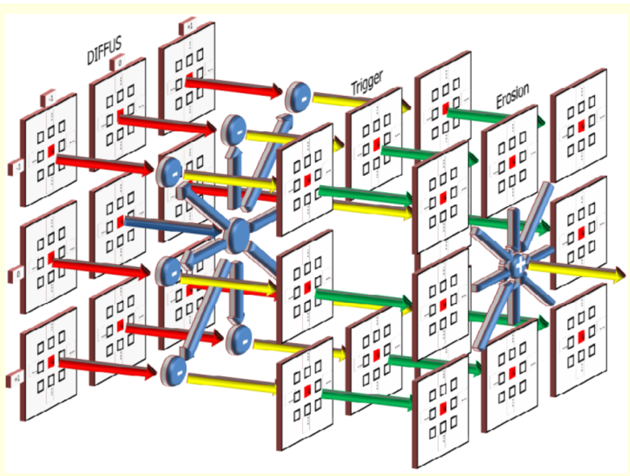

Fig. 7. Wave computing algorithm for boundary edge detection.

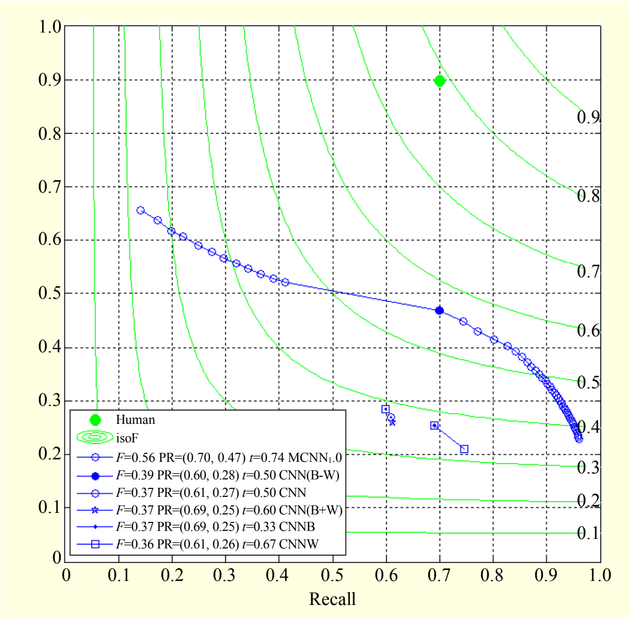

Fig. 8. Comparison of edge detection methods.

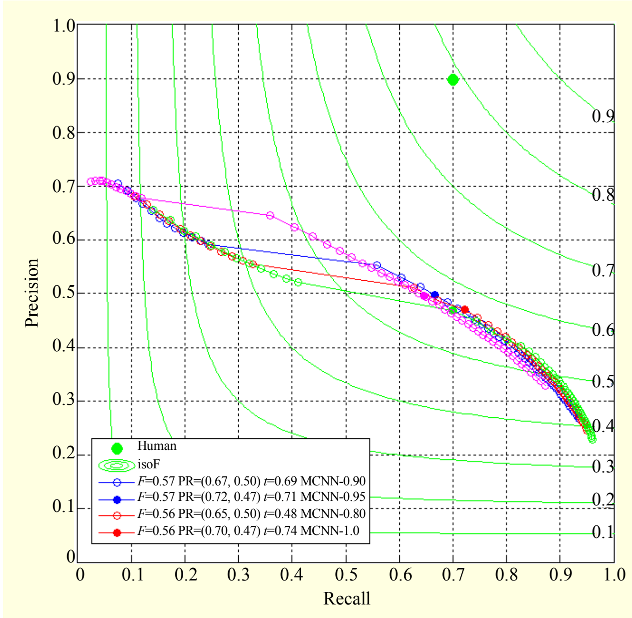

Fig. 9. Comparison of results by setting *b* to 0.8, 0.9, 0.95, and 1.0.

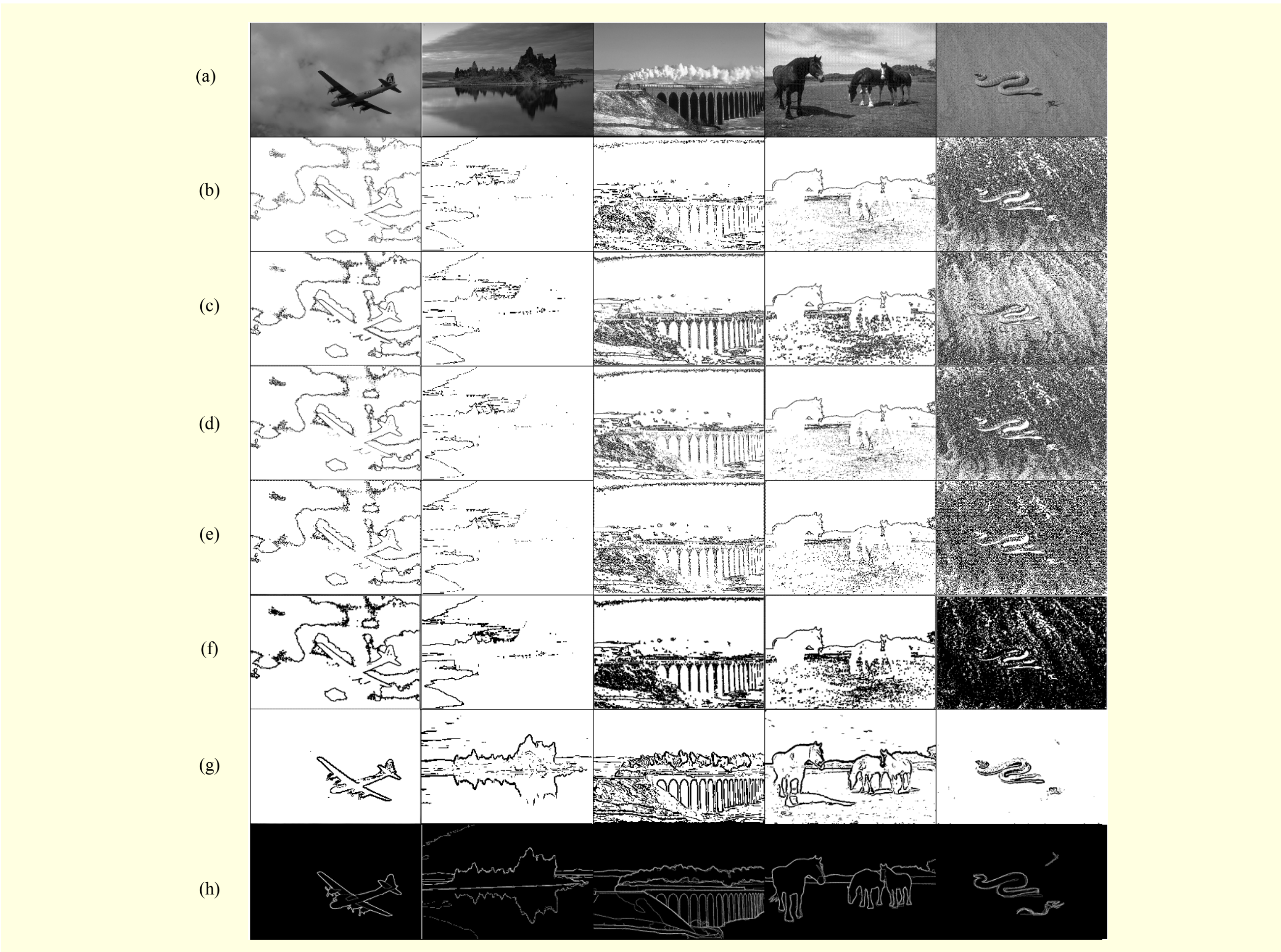

Fig. 10. Comparison of results of algorithms: (a) original images, (b) CNN, (c) CNNB, (d) CNNW, (e) CNN(B+W), (f) CNN(B–W), (g) MCNN, and (h) ground truth images.

are applied to the image to detect the image edges (CNN method). The black and white edge detection template is

$$A=\begin{bmatrix}-0.29 & 0.06 & -1.10\\ 0.06 & 9.72 & -1.38\\ -1.10 & -1.38 & -0.46\end{bmatrix}, B=\begin{bmatrix}0.41 & -1.57 & 0.48\\ -1.57 & 6.84 & -1.07\\ 0.48 & -1.07 & -0.15\end{bmatrix},$$

$z=-4.98.$

Another implementation for edge detection in grayscale images has been presented in the MatCNN toolbox [56]. This implementation can be run in two ways by inverting the sign of z in the edge detection template. In this paper, this method is referred to as CNNW (using z=+1.5) and CNNB (using z= –1.5). Furthermore, we use the addition and subtraction of the results of these two implementations: CNN(W+B) and CNN(W–B). The edge detection template used in this method is

$$A=\begin{bmatrix}0 & 0 & 0\\ 0 & 2 & 0\\ 0 & 0 & 0\end{bmatrix}, B=\begin{bmatrix}-0.25 & -0.25 & -0.25\\ -0.25 & 2 & -0.25\\ -0.25 & -0.25 & -0.25\end{bmatrix}, z=\pm 1.5.$$

We run the above methods using 100 test images in BSDS300 and evaluate the result using the Berkeley benchmark algorithm. Figure 8 illustrates the precision-recall curve and the maximum F-measure for these methods. Iso-F curves show the precision-recall relation for specific values of the F-measure. Edge detection approaches are ranked according to their maximum F-measure with respect to human ground-truth boundaries. As seen in Fig. 8, edge detection approaches based on CNN have F-measures between iso-F 0.3 and 0.4 curves. The proposed approach might have an F-measure between 0.1 to 0.6, depending on the desired precision and recall. The best F-measure for the proposed method based on the MCNN is 0.56 with (precision, recall)=(0.70, 47) at threshold t=0.74. Clearly, the proposed method is more accurate than the CNN methods.

The trigger template used in the proposed MCNN approaches bears the following values: a=0.25 and b=1. Another experiment is carried out by changing b. In this experiment, we set b to values 0.8, 0.9, 0.95, and 1.0 and run the proposed method on the BSDS300 and then run the benchmark to evaluate edge detection results for each value. The results are shown in Fig. 9. A maximum F-measure of 0.57 with the (precision, recall)=(0.67, 50) at t=0.69 is achieved using b=0.90 in the trigger template.

$$A=\begin{bmatrix}0 & a & 0\\ a & b & a\\ 0 & a & 0\end{bmatrix}, B=0, z=0.$$

Figure 10 shows the comparison of the results of our algorithm and other competing algorithms on five sample images. Clearly, the proposed method shows better performance than the CNN methods show.

VI. Conclusion

Computational intelligence methods have proven effective in modeling the dynamic phenomena and solving practical problems in different applications. This paper introduced a new CNN architecture based on topographic intercell interactions called "modular CNN." The stability of the proposed architecture was proven and the power of the system in texture segmentation and edge detection was illustrated. The results of the proposed approach in texture segmentation revealed desirable performance of the proposed method. Moreover, the MCNN was used for edge detection in grayscale images. Experiment results using the BSDS300 and Berkeley evaluating benchmark showed desirable performance of the MCNN in a real application. Our future work will concentrate on developing algorithms for texture segmentation and concave curve detection using the MCNN.

References

[1] M.B. Shahram and S. Mehdi, "An Imperialist Competitive Algorithm Artificial Neural Network Method to Predict Oil Flow Rate of the Wells," *Int. J. Comput. Appl.*, vol. 26, no. 10, July 2011, pp. 47-50.

[2] C.W. Chen, "Modeling and Control for Nonlinear Structural Systems via a NN-Based Approach," *Expert Syst. Appl.*, vol. 36, no. 3, 2009, pp. 4765-4772.

[3] M.L. Lin and C.W. Chen, "Application of Fuzzy Models for the Monitoring of Ecologically Sensitive Ecosystems in a Dynamic Semi-arid Landscape from Satellite Imagery," *Eng. Comput.*, vol. 27, no. 1, 2010, pp. 5-19.

[4] C.W. Chen, "Application of Fuzzy-Model-Based Control to Nonlinear Structural Systems with Time Delay: An LMI Method," *J. Vibration Control*, vol. 16, no. 11, 2010, pp. 1651-1672.

[5] C.W. Chen and J. Balthazar, "Modeling and Fuzzy PDC Control and Its Application to an Oscillatory TLP Structure," *Mathematical Problems Eng.*, vol. 2010, 2009, p. 39.

[6] C.W. Chen, K. Yeh, and K.F.R. Liu, "Adaptive Fuzzy Sliding Mode Control for Seismically Excited Bridges with Lead Rubber Bearing Isolation," *Int. J. Uncertainty, Fuzziness Knowl. Syst.*, vol. 17, no. 5, 2009, p. 705.

[7] C.W. Chen and P.C. Chen, "GA-Based Adaptive Neural Network Controllers for Nonlinear Systems," *Int. J. Innov. Comput. Inf. Control*, vol. 6, 2010, pp. 1793-1803.

[8] C.P. Tseng, C.W. Chen, and K.F.R. Liu, "Risk Control Allocation Model for Pressure Vessels and Piping Project," *J. Vibration*

Control, vol. 18, no. 3, 2012, pp. 385-394.

[9] K. Yeh et al., "Neural-Network Fuzzy Control for Chaotic Tuned Mass Damper Systems with Time Delays," *J. Vibration Control*, vol. 18, no. 6, 2011, pp. 785-795.

[10] T. Roska, "Computational and Computer Complexity of Analogic Cellular Wave Computers," *J. Circuits Syst. Comput*., vol. 12, no. 4, 2003, pp. 539-556.

[11] L.O. Chua and L. Yang, "Cellular Neural Networks: Theory," *IEEE Trans. Circuits Syst.*, vol. 35, no. 10, 1988, pp. 1257-1272.

[12] L.O. Chua and L. Yang, "Cellular Neural Networks: Applications," *IEEE Trans. Circuits Syst.*, vol. 35, no. 10, 1988, pp. 1273-1290.

[13] T. Roska and L.O. Chua, "The CNN Universal Machine: An Analogic Array Computer," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 40, no. 3, 1993, pp. 163-173.

[14] L.O. Chua and T. Roska, "The CNN Paradigm," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 40, no. 3, 1993, pp. 147-156.

[15] L.O. Chua and T. Roska, *Cellular Neural Networks and Visual Computing: Foundation and Applications*, Cambridge, UK: Cambridge University Press, 2002.

[16] T. Roska, "Circuits, Computers, and Beyond Boolean Logic," *Int. J. Circuit Theory Appl.*, vol. 35, no. 5-6, 2007, pp. 485-496.

[17] T. Roska, "Analogic CNN Computing: Architectural, Implementation, and Algorithmic Advances — A Review," *Proc. IEEE Int. Workshop Cellular Neural Netw. Appl.*, 1998, pp. 3-10.

[18] M. Pavone et al., "Climbing Obstacle in Bio-Robots via CNN and Adaptive Attitude Control," *Int. J. Circuit Theory Appl.*, vol. 34, no. 1, 2006, pp. 109-125.

[19] R. Tetzlaff, C. Niederhöfer, and P. Fischer, "Automated Detection of a Preseizure State: Non-linear EEG Analysis in Epilepsy by Cellular Nonlinear Networks and Volterra Systems," *Int. J. Circuit Theory Appl.*, vol. 34, no. 1, 2006, pp. 89-108.

[20] G. Cserey, A. Falus, and T. Roska, "Immune Response Inspired Spatial-Temporal Target Detection Algorithms with CNN-UM," *Int. J. Circuit Theory Appl.*, vol. 34, no. 1, 2006, pp. 21-47.

[21] I. Szatmári, "Object Comparison Using PDE-Based Wave Metric on Cellular Neural Networks," *Int. J. Circuit Theory Appl.*, vol. 34, no. 4, 2006, pp. 359-382.

[22] S. Xavier-de-Souza, J.A. Suykens, and J. Vandewalle, "Learning of Spatiotemporal Behaviour in Cellular Neural Networks," *Int. J. Circuit Theory Appl.*, vol. 34, no. 1, 2006, pp. 127-140.

[23] Z. Fodróczi and A. Radványi, "Computational Auditory Scene Analysis in Cellular Wave Computing Framework," *Int. J. Circuit Theory Appl.*, vol. 34, no. 4, 2006,pp. 489-515.

[24] D. Bálya et al., "A CNN Framework for Modeling Parallel Processing in a Mammalian Retina," *Int. J. Circuit Theory Appl.*, vol. 30, no. 2-3, 2002, pp. 363-393.

[25] D. Bálya et al., "Implementing the Multilayer Retinal Model on the Complex-Cell CNN-UM Chip Prototype," *Int. J. Bifurc. Chaos*, vol. 14, no. 2, 2004, pp. 427-451.

[26] V.I. Krinsky, V.N. Biktashev, and I.R. Efimov, "Autowave Principles for Parallel Image Processing," *Physica D: Nonlinear Phenomena*, vol. 49, no. 1-2, 1991, pp. 247-253.

[27] V. Perez-Munuzuri, V. Perez-Villar, and L.O. Chua, "Autowaves for Image Processing on a Two-Dimensional CNN Array of Excitable Nonlinear Circuits: Flat and Wrinkled Labyrinths," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 40, no. 3, 1993, pp. 174-181.

[28] T. Roska et al., "Simulating Nonlinear Waves and Partial Differential Equations via CNN — Part I: Basic Techniques," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 42, no. 10, 1995, pp. 807-815.

[29] T. Kozek et al., "Simulating Nonlinear Waves and Partial Differential Equations via CNN — Part II: Typical Examples," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 42, no. 10, 1995, pp. 816-820.

[30] C. Rekeczky, "CNN Architectures for Constrained Diffusion Based Locally Adaptive Image Processing," *Int. J. Circuit Theory Appl.*, vol. 30, no. 2-3, 2002, pp. 313-348.

[31] C. Rekeczky and L.O. Chua, "Computing with Front Propagation: Active Contour and Skeleton Models in Continuous-Time CNN," *J. VLSI Signal Process.*, vol. 23, no. 2, 1999, pp. 373-402.

[32] S. Majorana and L.O. Chua, "A Unified Framework for Multilayer High Order CNN," *Int. J. Circuit Theory Appl.*, vol. 26, no. 6, 1999, pp. 567-592.

[33] Z. Yang, Y. Nishio, and A. Ushida, "Characteristic of Mutually Coupled Two-Layer CNN and Its Stability," *J. Circuits Syst. Comput.*, vol. 12, no. 4, 2003, pp. 473-490.

[34] B.E. Shi, "An Eight Layer Cellular Neural Network for Spatio-Temporal Image Filtering," *Int. J. Circuit Theory Appl.*, vol. 34, no. 1, 2006, pp. 141-164.

[35] M.L. Lin and C.W. Chen, "Stability Analysis of Community and Ecosystem Hierarchies Using the Lyapunov Method," *J. Vibration Control*, vol. 17, no. 13, 2011, pp. 1930-1937.

[36] K. Yeh, C.W. Chen, and C. Cattani, "Stability Analysis of Interconnected Fuzzy Systems Using the Fuzzy Lyapunov Method," *Mathematical Problems Eng.*, vol. 2010, 2009, pp. 34-43.

[37] K. Yeh, C.Y. Chen, and C.W. Chen, "Robustness Design of Time-Delay Fuzzy Systems Using Fuzzy Lyapunov Method," *Appl. Mathematics Comput.*, vol. 205, no. 2, 2008, pp. 568-577.

[38] C.W. Chen et al., "Stability Analysis of an Oceanic Structure Using the Lyapunov Method," *Eng. Comput.: Int. J. Computer Aided Eng.*, vol. 27, no. 2, 2010, pp. 186-204.

[39] P.C. Chen, C.W. Chen, and W.L. Chiang, "Linear Matrix Inequality Conditions of Nonlinear Systems by Genetic Algorithm-Based H∞ Adaptive Fuzzy Sliding Mode Controller," *J. Vibration Control*, vol. 17, no. 2, 2011, pp. 163-173.

[40] T. Roska, "Cellular Wave Computers for Brain-like Spatial-

Temporal Sensory Computing," *IEEE Circuits Syst. Mag.*, vol. 5, no. 2, 2005, pp. 5-19.
[41] C. Rekeczky et al., "Analogic Cellular PDE Machines," *Proc. IJCNN*, 2002, pp. 2033-2038.
[42] C. Rekeczky and T. Roska, "Calculating Local and Global PDEs by Analogic Diffusion and Wave Algorithms," *Proc. European Conf. Circuit Theory Des.*, vol. 2, 2001, pp. 17-20.
[43] L. Kék, K. Karacs, and T. Roska, *Cellular Wave Computing Library: Templates, Algorithms, and Programs, V. 2.1*, Budapest, Hungary: MTA-SZTAKI, 2007.
[44] K.R. Crounse and L.O. Chua, "Methods for Image Processing and Pattern Formation in Cellular Neural Networks: A Tutorial," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 42, no. 10, 1995, pp. 583-601.
[45] T.M. Apostol, *Calculus, Volume 2, Multi-Variable Calculus and Linear Algebra with Applications*, New York: Wiley, 1969.
[46] C.D. Meyer, *Matrix Analysis and Applied Linear Algebra*, Philadelphia, PA: Society for Industrial Mathematics, 2000.
[47] W.H. Steeb and T.K. Shi, *Matrix Calculus and Kronecker Product with Applications and C++ Programs*, Hackensack, NJ: World Scientific Pub Co Inc., 1997.
[48] G.E. Pazienza, E. Gomez-Ramirez, and X. Vilasis-Cardona, "Genetic Programming for the CNN-UM," *10th Int. Workshop CNNA*, 2006, pp. 1-6.
[49] T. Szirányi and M. Csapodi, "Texture Classification and Segmentation by Cellular Neural Networks Using Genetic Learning," *Comput. Vision Image Understanding*, vol. 71, no. 3, 1998, pp. 255-270.
[50] T. Roska, *Software Library for Cellular Wave Computing Engines V. 3.1*, Budapest, Hungary: MTA-SZTAKI and Pazmany University, 2010.
[51] J.A. Hernandez, F.G. Castaneda, and J.A. Cadenas, "A Method for Edge Detection in Gray Level Images, Based on Cellular Neural Networks," *52nd IEEE Int. MWSCAS*, 2009, pp. 730-733.
[52] D. Martin et al., "A Database of Human Segmented Natural Images and Its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics," *Proc. 8th IEEE Int. Conf. Comput. Vision*, vol. 2, 2001, pp. 416-423.
[53] P. Arbelaez et al., "Contour Detection and Hierarchical Image Segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, 2010, pp. 898-916.
[54] D. Martin, C. Fowlkes, and J. Malik, "Learning to Detect Natural Image Boundaries Using Brightness and Texture," *Neural Inf. Process. Syst.*, 2002, pp. 1255-1262.
[55] D.R. Martin, C. Fowlkes, and J. Malik, "Learning to Detect Natural Image Boundaries Using Local Brightness, Color, and Texture Cues," *IEEE Trans. Pattern Anal.*, vol. 26, no. 5, 2004, pp. 530-549.
[56] C. Rekeczky, *MATCNN – Analogic CNN Simulation Toolbox for Matlab*, Version 1.0, DNS-11-1997, technical report, Analogical and Neural Computing Laboratory, Computer and Automation Institute of the Hungarian Academy of Sciences, Budapest, Hungary, Sept. 1997.

Mojtaba Karami received his BS in computer engineering from Sharif University of Technology, Tehran, Iran, in 1998 and his MS in computer engineering from Amirkabir University of Technology, Tehran, Iran, in 2002. He is currently a PhD candidate in computer engineering at Amirkabir University of Technology. His research interests include neural networks, evolutionary computing, and image processing.

Reza Safabakhsh received his BS in electrical engineering from Sharif University of Technology, Tehran, Iran, in 1976 and his MS and PhD in electrical and computer engineering from the University of Tennessee, Knoxville, TN, USA, in 1980 and 1986, respectively. He worked at the Center of Excellence in Information Systems, Nashville, TN, USA, from 1986 to 1988. Since 1988, he has been with the Computer Engineering Department, Amirkabir University of Technology, Tehran, Iran, where he is currently a professor and the director of the Computer Vision Laboratory. His current research interests include neural networks, computer vision, and information hiding. Dr. Safabakhsh is a member of the IEEE and several honor societies, including Phi Kappa Phi and Eta Kapa Nu. He was the founder and a member of the Board of Executives of the Computer Society of Iran and was the president of this society for the first four years.

Mohammad Rahmati received his MSc in electrical engineering from the University of New Orleans, New Orleans, LA, USA, in 1997 and his PhD in electrical and computer engineering from the University of Kentucky, Lexington, KY, USA, in 2003. Currently, he is an associate professor in the Computer Engineering Department at Amirkabir University of Technology, Tehran, Iran. His research interests include pattern recognition, image processing, bioinformatics, video processing, and data mining.