Analogical and Neural Computing Laboratory
Computer and Automation Institute, Hungarian Academy of Sciences
H-1111 Budapest, Kende-u.13-17.

DISTANCE PRESERVING 1D TURING-PATTERN MODELS VIA CNN, IMPLEMENTING OF COMPLEX-VALUED CNN, AND SOLVING A SIMPLE INVERSE PATTERN PROBLEM (DETECTION)

G. Tóth, P. Földesy, and T. Roska

DNS-3-1996

MTA SZTAKI

***Abstract** - In this paper several examples for pattern generation and detection using cellular neural networks (CNNs) are explained. In the first part a CNN implementation of a reaction-diffusion system is described that produces periodic, so-called Turing patterns. In the second part the CNN with complex-valued templates are introduced, presenting two applications for pattern generation. In the third part a method for black-and-white pattern detection will be described.*

1. Introduction

The implementation of PDEs using CNN has been reported recently [5,6,11]. *Turing*-patterns are introduced by *A. M. Turing* [1] in 1952. They appear in physics, chemistry, biology. The hypothetical molecular mechanism is called *reaction-diffusion* system, and develops periodic patterns from the initially inhomogeneous state. Practically, the initial state always contains inhomogeneity, and this is enough to start pattern generation.

Shigeru Kondo and *Rihito Asai* [2] used *Turing*-patterns to simulate the behavior of the skin of the marine angelfish *Pomacanthus*. On the skin of this fish the width of the stripes is independent of the length of the fish. As the fish grows, new vertical stripes appear between two old stripes, so these stripes are not fixed in the skin. Unlike mammal skin patterns which simply enlarge proportionally during the body grows, these stripes maintain the spaces between the lines. This is shown in Figure 1. First the tissue contains 60 cells. The growing was modeled by duplicating *5%* of the cells once in a given number of iterations. A_i denotes the concentration of the so-called *Activator* molecules that will be described soon. Three snapshots are shown in the case of different array sizes. It can be easily seen how new stripes appear.

In this paper, in section 2 we show a CNN model of this phenomenon. In section 3 and 4 we show how a complex valued CNN can be implemented with two layers and used for pattern generation. In section 5 we show a single 1D pattern detection mechanism.

2. CNN model of a one-dimensional *Turing*-type reaction-diffusion system found in Angelfish

2.1. Stripes arising from Turing-type reaction-diffusion equations

In the following section we will summarize the mathematical background of pattern formation described in [2]. The system can be easily given by two partial differential equations. Starting the system from *any* initial condition eventually one spatial harmonic will be dominant. This property makes forming patterns possible.

The following reaction-diffusion equations were identified as the governing equations for forming patterns:

$$\frac{dA}{dt} = c_1 A + c_2 I + c_3 + D_A \frac{d^2 A}{dx^2} - g_A A \qquad \frac{dI}{dt} = c_4 A + c_5 + D_I \frac{d^2 I}{dx^2} - g_I I \qquad (1)$$

Here x is the coordinate for the one-dimensional space, A and I are the concentrations of the two so-called *morphogens*, the Activator and Inhibitor molecules. Parameters c_i ,g_i and D_i are constants:

c_1=0.08 c_2=-0.08 c_3=0.05 c_4=0.1 c_5=-0.15 (2)
D_A=0.007 D_I=0.1 g_A=0.03 g_I=0.06

The first terms of the equations, called the *synthesis rates* were limited:

$0<(c_1A+ c_2I+ c_3)<0.18$ (3)
$0<(c_4A+ c_5)<0.5$

With this nonlinear effect we can avoid the unlimited increasing of the amplitude. The equation is discretized in space to model it with a 1D cellular neural network [5]:

$$\begin{aligned} \frac{dA_i}{dt} &= c_1 A_i + c_2 I_i + c_3 + \frac{D_A}{h^2}(A_{i-1} - 2A_i + A_{i+1}) - g_A A_i \\ \frac{dI_i}{dt} &= c_4 A_i + c_5 + \frac{D_I}{h^2}(I_{i-1} - 2I_i + I_{i+1}) - g_I I_i \end{aligned} \qquad (4)$$

where h is the grid unit and we used a simple approximation of the spatial second order derivative. (It should be discretized in time, value and parameters to model it in digital computers.)

Turing proved that one of the spatial frequency components grows faster than the others so it will dominate. In other words, a spatial sine wave will appear. For an infinite cell array, the wave-length of this dominating frequency component can be computed knowing the constants c_i, g_i, D_i. Of course, in case of a finite cell array it can occur that the ratio of the number of the cells to the computed wave-length is not an integer. In this case the real wave-length will be near the computed wave-length. (For example, suppose the computed wave-length is 4 units and there are 18 cells in the array. Either 4 waves with period length of 4.5 units, or 5 waves with period-length of 3.6 units will appear.) Thus the wave-length is *almost* independent of the array size. While increasing the size of the array, we reach a limit size where the real wave-length immediately changes, i.e., a new stripe appears.

2.2. Realization using a 1D double-layer first-order CNN

In this section the double-layer one-dimensional CNN model of the previous system is constructed and the phenomenon found in Angelfish is also demonstrated. The growth of the fish is modeled by increasing the size of the CNN cell array.

Equation (4) can be easily realized using CNN. (See also [5] and [6].) The coefficients of the differential equations given by *Kondo* and *Asai* will be transformed because of the saturation region of the cells' nonlinearity. (The cell state must be between *-1* and *+1*.)

$$\begin{aligned} &A_{old}=1.25A_{new}+2.75 \\ &I_{old}=I_{new}+2.5 \end{aligned} \qquad (5)$$

where A_{old}, I_{old} are the concentrations of the two morphogens in (4), A_{new}, I_{new} are the transformed concentrations. Substituting (5) in equations (4) we get new equations for A_{new} and I_{new}.

Moreover, we do not use the previously described nonlinear effect (limiting the synthesis rates describing by equations (3)). The saturation feature of the cells' nonlinearity will stop the growing of the stripes instead and causes deformation at the peaks of the weaves. In this case, it is not necessary to use the ring structure, that was applied by Turing [1]. If we force a constant value to the edges (fixed frame) then we can get also an appropriate result. The increase of the array can be also solved easier. We need not duplicate the cells inside the cell array, as in [2]. It is enough to add a new cell to the network at the edges.

The general form for the equations (4) given by *Turing*:

$$\begin{aligned} \frac{dA_i}{dt} &= aA_i + bI_i + c + \mu(A_{i-1} - 2A_i + A_{i+1}) \\ \frac{dI_i}{dt} &= dA_i + eI_i + f + \nu(I_{i-1} - 2I_i + I_{i+1}) \end{aligned} \qquad (6)$$

The parameters *a, b, c, d, e, f,* μ and ν can be easily given with c_i ,g_i and D_i. (In fact, Turing did not use the *c* and *f* constant terms, yet.) The cellular neural network templates implementing equations (6) are:

$$\boxed{\begin{array}{llll} A_{1to1}=[\ \mu \quad (a-2\mu+1) \quad \mu\ ] & A_{2to1}=[\ 0 \quad b \quad 0\ ] & I_1=c & \\ A_{1to2}=[\ 0 \quad d \quad 0\ ] & A_{2to2}=[\ \nu \quad (e-2\nu+1) \quad \nu\ ] & I_2=f & (7) \end{array}}$$

Hence, the following CNN realizes equations (4) with the parameters specified above:

$$\begin{array}{ll} A_{1to1}=[\ 0.112\ \ 0.826\ \ 0.112\ ] & A_{2to1}=[\ 0\ \ -0.064\ \ 0\ ] \\ A_{1to2}=[\ 0\ \ 0.125\ \ 0\ ] & A_{2to2}=[\ 1.6\ \ -2.26\ \ 1.6\ ] \\ I_1=-0.01 & I_2=-0.025 \end{array} \qquad (8)$$

The phenomenon found in Angelfish can be modeled in the following way. We start the network that produces *Turing*-patterns. (The initial state is a random noise.) When the network reaches the steady state a new cell is added at the right edge of the cell array. Then we start the network and wait again. This sequence can be repeated several times.

This can be seen in Figure 2. First, as shown in Figure 2.a, the array consists of 50 cells. There are only two peaks. Then the size of the CNN block grows. After every 37.5τ a new cell is added at the right edge. In Figure 2.b reaching array size of 51 cells there are still only two peaks, but the second one begin to change. When reaching 52 units array size, the second peak splits, and after a time two new peaks evolve from it. This is shown in Figure 2.c and 2.d.

2.3. *Realization using a single-layer CNN with second order cells*

Turing-type reaction-diffusion equations or Turing-patterns can be generated by a single layer CNN with second-order cells, too. ([7]) Consider equation (6) without any nonlinear effect. The structure of the cell solving these equations can be seen in Figure 3.a. The two state variables A_i and I_i, are the voltages of the two capacitors. The capacitances are not equal. The coupling between the two state variables is implemented by a resistor. Two current sources play the role of the constants *c* and *f.* Figure 3.b shows how the cells are connected to each other. Fortunately, the template matrices are symmetrical so resistors can be used for the intercell-coupling.

3. Complex valued CNN templates and their implementation

Complex neural cells and networks were introduced in [10,12], where so-called *multi-valued* cells and complex templates were used. In the realization presented here *both the templates and the states are complex*. All complex variables are represented by their real and imaginary parts.

The state equation of the complex-valued CNN is:

$$\frac{d\left(v_{Rx,ij}+jv_{Ix,ij}\right)}{dt}=-\left(v_{Rx,ij}+jv_{Ix,ij}\right)+\sum_{(k,l)\in Nr(i,j)}\left(A_R+jA_I\right)_{kl}\left(v_{Ry,kl}+jv_{Iy,kl}\right)+\sum_{(k,l)\in Nr(i,j)}\left(A_R+jA_I\right)_{kl}\left(v_{Ru,kl}+jv_{Iu,kl}\right)+I_R+jI_I \tag{9}$$

where $j=\sqrt{-1}$. The templates of the complex-valued CNN are:

layer	feedback	control	current
Complex:	A_R+jA_I	B_R+jB_I	I_R+jI_I

The complex equation can be separated into two real equations:

$$\begin{aligned}
\frac{dv_{Rx,ij}}{dt} &= -v_{Rx,ij} + \sum_{(k,l)\in Nr(i,j)} \left(A_R v_{Ry,kl} - A_I v_{Iy,kl} \right) + \sum_{(k,l)\in Nr(i,j)} \left(B_R v_{Ru,kl} - B_I v_{Iu,kl} \right) + I_R \\
\frac{dv_{Ix,ij}}{dt} &= -v_{Ix,ij} + \sum_{(k,l)\in Nr(i,j)} \left(A_I v_{Ry,kl} + A_R v_{Iy,kl} \right) + \sum_{(k,l)\in Nr(i,j)} \left(B_I v_{Ru,kl} + B_R v_{Iu,kl} \right) + I_I
\end{aligned} \tag{10}$$

The complex-valued CNN can be realized by a double-layer real-valued CNN given below:

layer	feedback	control	current
Real:	Self: A_R	B_R	I_R
	From Imaginary: $-A_I$	$-B_I$	
Imaginary:	Self: A_R	B_R	I_I
	From Real: A_I	B_I	

4. Pattern generation with complex-valued templates

Here some applications of the complex-valued CNN templates will be used to generate one-dimensional periodic patterns and we will compare these solutions to the single-layer real-valued CNN implementations. We will see that in the case of complex-valued CNN the template size is only 3×1, in the case of the real-valued CNN it is 5×1.

4.1. Sine wave generation with a band-pass filter A template

We have seen in the case of *Turing*-patterns that one of the spatial harmonic was amplified stronger than others, after a lapse of time it was dominant. In other words, a sine wave appeared. This idea can be used in other structures, too. We design a CNN (with a band-pass filter *A* template) to achieve that starting from *any* initial state eventually only one spatial harmonic remains ([9,13]).

An autonomous CNN is described, that has a zero *B* template and *I* bias. Suppose that we are in the linear domain of the output nonlinearity. In this case the equation of a CNN cell for the one-dimensional case is the following:

$$C\frac{dv_{xi}}{dt} = -\frac{1}{R}v_{xi} + \sum_{|k-i|\le r} A(i;k)v_{xk} \tag{11}$$

Here *r* is the neighborhood in which the cells have interconnected neighbors. For the sake of simplicity, choose capacitance *C=1* and resistor *R=1*. Introducing the so-called convolution mask a_i the template *A* can be given in the following way:

$$A = \left[a_r \quad \ldots \quad a_2 \quad a_1 \quad (a_o + 1) \quad a_{-1} \quad a_{-2} \quad \ldots \quad a_{-r} \right] \tag{12}$$

equation (11) can be expressed using spatial convolution denoted by ‘*’:

$$\frac{d}{dt} v_{xi}(t) = a_i * v_{xi}(t) \tag{13}$$

Let *a[ω] be* the spatial spectrum of a_i and *v[ω](t)* the spatial spectrum of the state v_{xi} at time *t*. Then *v[ω](t)* can be expressed in the following way :

$$v[\omega](t)=v[\omega](0)e^{a[\omega]t} \tag{14}$$

If the real part of *a[ω]* is positive then the amplitude of the *ω* spatial frequency component will increase. If the real part is negative then the amplitude of this frequency component will decrease. If the real part of *a[ω]* is positive only in interval $[\omega_0\text{-}\Delta\omega, \omega_0+\Delta\omega]$ then the network will increase only the amplitude of the frequency components around ω_0. Practically, in a physical system the initial state contains all of the frequency components. If only the previously mentioned ones increase then a complex harmonic with frequency ω_0 will appear. The network operates as a band-pass filter amplifying only the spatial frequency components of the initial state around ω_0.

Next, we design a template *A* having such an *a[ω]* spatial spectrum. Consider the following complex-valued CNN template:

$$\boxed{A(\omega_0)=[\ a\cdot e^{-j\omega_0} \quad (b+1) \quad a\cdot e^{j\omega_0}\ ] \qquad B=[\ 0\ ] \qquad I=0 \qquad (15)}$$

where *a, b,* ω_0 are constants to be computed in the following part. The spectrum of a_i will be

$$a[\omega]=2a\cdot\cos(\omega\text{-}\omega_0)+b \tag{16}$$

If a>0 and b>0 then the spectrum is maximum at $\omega=\omega_0$, and the value of the maximum is

$$\max a[\omega]=a[\omega_0]=2a+b \tag{17}$$

We require that $a[\omega]>0$ fulfills only around ω_0. This can be achieved with the proper settings of a and b, i.e., *max* $a[\omega]$ must be a small positive value. The larger *max* $a[\omega]$, the faster the growing of the waves.

Assume, we generate sine waves with period-length $L=10$. The frequency ω_0 can be computed in this well-known way:

$$\omega_0=2\pi /L \tag{18}$$

Thus $\omega_0=0.628$. Choose $a=0.25$, $b=-0.4$. Thus *max* $a[\omega]=2a+b=0.1>0$. The complex-valued templates:

$$A=[\ 0.202-0.147j \quad 0.6 \quad 0.202+0.147j\ ] \qquad B=[\ 0\ ] \qquad I=0 \tag{19}$$

In Figure 4 and 5 the generation process can be seen, in case of random initial state and a single spike, respectively. Only the output of the first layer is presented, because the graphs for the second layer would be qualitatively similar. At settled state both layers contain a spatial sine wave.

With a greater template size the sine wave generation can be realized also by a real-valued CNN with 5×1 templates:

$$\boxed{A=[\ -a \quad 4a\cdot\cos(\omega_0) \quad (b+1) \quad 4a\cdot\cos(\omega_0) \quad -a\ ] \qquad B=[\ 0\ ] \qquad I=0 \qquad (20)}$$

where a and b are positive constants. Similar to the previous case, they must be chosen knowing that the maximum of the spectrum must be a small positive value.

$$\max a[\omega]=a[\omega_0]= 2a(1+2\cos^2(\omega_0))+b \tag{21}$$

Thus the realization with real-valued templates needs a larger template size.

4.2. Sine wave generation with series given by recursive formula

In this section a new method will be introduced to generate sine waves with complex-valued templates. Here again an autonomous CNN will be presented. In this structure due to the one-directional coupling between cells the steady-state of a cell can be computed only if the previous cell reached the steady-state, so a cell must "wait" for its neighbor. Thus the structure presented here provides inherently serial execution.

The following series describes a spatial complex harmonic:

$$v_i=Ke^{j\omega_i} \tag{22}$$

where K is a constant. Given the first element, it can be equivalently described by a recursive formula:

$$v_i = e^{j\omega} v_{i-1} \qquad (23)$$

This recursive definition can be used to construct a CNN capable of producing this series:

$$\boxed{A=[\, e^{j\omega} \;\; 0 \;\; 0 \,] \qquad B=[\, 0 \,] \qquad I=0 \qquad (24)}$$

We fill K into the first cell and keep this cell fixed. Then after the transient settled the second cell will contain $Ke^{j\omega}$, the third $Ke^{j2\omega}$, etc. The operation of the network is shown in Figure 6.

The sine wave generation with series given by recursive formula can be solved also by a single-layer CNN with real-valued 5×1 templates:

$$\boxed{A=[\, -1 \;\; 2\cos(\omega) \;\; 0 \;\; 0 \;\; 0 \,] \qquad B=[\, 0 \,] \qquad I=0 \qquad (25)}$$

Here the recursion involves the two left neighbors not only one. If K is filled into the first two cells of the network and their states are kept fixed then after a lapse of time a sine wave with frequency ω will appear. Both methods can be used to generate stripes with controlled stripe-density. This can be seen in Figure 7.

5. 1D black and white pattern detection with 3×1 templates

In this section a new method for pattern detection on a one-dimensional black-and-white image is given. The main problem is to measure the length of a black or a white stripe with the CNN that contains only locally interconnected cells. Thus we have to measure a stripe with length of 20-30 units with a CNN that has a template size of 3 units. First an algorithm independent of the implementation will be presented then the realization with CNN will be described. The solution proposed here uses a series given by a recursive formula.

As mentioned above, first an algorithm independent of the implementation will be explained. Suppose, that we want to detect a stripe series containing black stripes with length of *5* units and *3* units space between them. (Thus the required pattern consists of black and white stripes with length of 5 and 3 units, respectively.)

■	■	■	■	■	□	□	□	■	■	■	■	■	□	□	□	■	■	■	■	□	□	■	■	■	1D input image
1	2	3	4	5	1	2	3	1	2	3	4	5	1	2	3	1	2	3	4	1	2	1	2	3	1. Counting from left to right
5	4	3	2	1	3	2	1	5	4	3	2	1	3	2	1	4	3	2	1	2	1	3	2	1	2. Counting from right to left
6	6	6	6	6	4	4	4	6	6	6	6	6	4	4	4	5	5	5	5	3	3	4	4	4	3. Addition of row 2 and 3
6	6	6	6	6	4	4	4	6	6	6	6	6	4	4	4	6	6	6	6	4	4	6	6	6	4. Required value for row 4
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	- 1	- 1	- 1	- 1	- 1	- 1	- 2	- 2	- 2	5. Subtraction of row 5 from 4

Table II. Algorithm for stripe detection. The pattern to detect contains black stripes with length of 5 units and 3 units space between them. If there is a zero in the last row then there the algoritm detected the desired stripes.

The steps of the detecting algorithm are the following (See also Table II):

Step 1 Count from left to right and put the result in row 1. Restart counting, if the actual element and the previous one of the black and white input image are different (i.e., if the border of a single-colored area is reached).
Step 2 Count from right to left in a similar way and put the result in row 2.
Step 3 Add row 1 and the row 2 and put the result in row 3. There is under a black or a white area its length plus one. To decide whether there are the desired stripes or no, we must compare the elements of the fourth row with *5+1=6* under a black stripe and with *3+1=4* under a white stripe
Step 4 Write in row 4 the value with which we should compare the elements of row 3 to detect the stripes with the desired length.
Step 5 Subtract of the row 5 from row 4 and put the result in row 5. If it is zero then there the required stripe series was found.

Next, the implementation in CNN will be described. It will be a multi-layer network, although it can be implemented in CNN Universal Machine, too. In the case of a multi-layer network additional data transfer and digital hardware are not necessary, the execution speed can be higher.

First a method is described to implement a “counter” with CNN. The following single-layer network realizes this function:

$$A=[\ 1\ \ 0\ \ 0\ ] \qquad B=[\ 0\ ] \qquad I=U \qquad (26)$$

Here the constant *U* is the unit of incrementation. In Table II. it was one, but now it is chosen to avoid reaching the cells’ saturation region. For example, it can be *0.05*. Fill *0.05* into the cell v_0

and keep it fixed, then after a lapse of time we get *0.1* in v_1, *0.15* in v_2, etc. This network produces a series given by the following recursive formula:

$$v_i=v_{i-1}+0.05 \qquad (27)$$

Note, that in this case a cell must wait for the previous cells, so the execution will be serial. The more cells we have the more time the transient takes. (In the final solution it does not cause a problem.)

Next, a solution is explained to control this incrementation implementing the function of the first step of the algorithm presented in Table II. The following network restarts counting at the borders of the stripes. (The analyzed black and white image is applied onto the input of the network. The black color is described by +1.0, the white color by -1.0. The value of the frame is 0.0. It means, that we force zero to the inputs of the cells at the edges, that are out of the regular CNN structure.)

$$A=[\ n(v_{ui}-v_{u,i-1})\ \ 0\ \ 0\ ] \qquad B=[\ 0\ ] \qquad I=U \qquad (28)$$

$$n(x)=\begin{cases} 1, \ if \ |x| < 0.1 \\ 0, \ if \ |x| \geq 0.1 \end{cases}$$

This type of nonlinear template (so-called *switched template*) is introduced in [8]. The variable element of the *A* matrix depends on the actual cell states. It is 1, if the colors of the actual and the previous cell are identical, and 0, if they differ. Thus the count restarts at the border of the stripes. The steady state of the network is:

$$v_{xi}=\begin{cases} v_{x,i-1}+U, \ if \ v_{u,i}=v_{u,i-1} \\ \qquad U, \ if \ v_{u,i} \neq v_{u,i-1} \end{cases} \qquad (29)$$

Some snapshots of the output during the transient can be seen in Figure 8.

Next, a three-layer structure is presented, that measures the length of a black or a white stripe. The analyzed image is applied onto the input of the first layer. In steady state the output of the third layer contains information about the length of the single-colored stripes of the input pattern, realizing the first three steps of the algorithm in Table II. The templates of the network are the following:

$$A_{1to1}=[\ n(v_{1ui}-v_{1u,i-1})\ \ 0\ \ 0\ ] \qquad B_{1to1}=[\ 0\ ] \qquad I_1=U$$

$$A_{2to2}=[\ 0\ \ 0\ \ n(v_{1ui}-v_{1u,i+1})\ ] \qquad B_{2to2}=[\ 0\ ] \qquad I_2=U \qquad (30)$$

$$A_{1to3}=[\ 0\ \ 1\ \ 0\ ] \qquad B_{3to3}=[\ 0\ ] \qquad I_3=0$$

A_{2to3}=[0 1 0]
A_{3to3}=[0]

The first layer counts from left to right, the second one from right to left, the third one adds the previous two layers. If a pixel of the input image is a part of an *L* long single-colored stripe then in the third layer the steady state of the corresponding cell will be *(L+1)U*:

$$v_{3xi} = v_{1xi} + v_{2xi} = (L + 1)U \qquad (31)$$

In the following part a three-layer network detecting black and white patterns is described, in other words, the whole algorithm shown in Table II. is realized. The pattern to find contains L_b unit long black stripes and L_w unit long white stripes. To detect this pattern the state of the third layer must be compared with $(L_b+1)U$ in the case of black stripes, and with $(L_w+1)U$ in the case of white stripes. Fortunately, this comparing function can be built in the third layer, hence we get the following network:

A_{1to1}=[$n(v_{1ui}-v_{1u,i-1})$ 0 0]	B_{1to1}=[0]	I_1=U
A_{2to2}=[0 0 $n(v_{1ui}-v_{1u,i+1})$]	B_{2to2}=[0]	I_2=U
A_{1to3}=[0 1 0]	B_{1to3}=[0 $-(L_b-L_w)U/2$ 0]	$I_3=-(L_b+L_w+2)U/2$
A_{2to3}=[0 1 0]	B_{3to3}=[0]	
A_{3to3}=[0]		

(32)

Consider a pixel of a black or a white stripe of length *L* in the input. In steady state the corresponding point of the third layer will contain

$$v_{3xi} = v_{1xi} + v_{2xi} - (L_b - L_w)\frac{U}{2}v_{1ui} - (L_b + L_w + 2)\frac{U}{2} = \begin{cases} v_{1xi} + v_{2xi} - (L_b + 1)U, \text{ if } v_{1ui} = +1 \\ v_{1xi} + v_{2xi} - (L_w + 1)U, \text{ if } v_{1ui} = -1 \end{cases} \qquad (33)$$

Taking

$$v_{1xi} + v_{2xi} = (L + 1)U \qquad (34)$$

into consideration the following equations are obtained:

$$v_{3xi} = \begin{cases} LU - (L_b + 1)U, \text{ if } v_{1ui} = +1 \\ LU - (L_w + 1)U, \text{ if } v_{1ui} = -1 \end{cases} \qquad (35)$$

In the case of a white stripe the state of the third layer is zero if the length is L_w, and in the case of a black stripe it is zero if the length is L_b. We compare L with different values depending on the color of the stripe. This is due to the usage of B_{1to3} that affects from the input of the first layer to the third layer.

To detect zero level the third layer has a special output nonlinearity:

$$f(x) = \begin{cases} 1, \ if \ |x| < (n+0.5)U \\ -1, \ if \ |x| \geq (n+0.5)U \end{cases} \qquad (36)$$

With the parameter *n* the tolerance of the required stripe length can be determined. If it is zero then stripes having *exactly* the desired length will be detected. If it is a positive integer then ±*n* units deviance is allowed in stripe-length.

The operation of the network can be seen in Figure 9. The input of the network is the picture to examine. The result is obtained at the output of the third layer. The black areas show that there an L_b unit long black stripe or an L_w unit long white stripe was found.

In fact this network uses a series given by a recursive formula to measure the length of a stripe, and it means *partly* serial execution. The longer the sought stripe is the longer the execution time is, however, the execution time does not depend on the size of the examined area. Thus we can exploit the higher speed of parallel processing.

4. Conclusions

In this paper several examples for pattern generation and detection were introduced. First the CNN implementation of a reaction diffusion system was described that produced *Turing*-patterns. It was also used to model the behavior of the skin pattern of the marine angelfish *Pomacanthus*. This example showed that the CNN paradigm could serve as a general frame for several problems, e.g., modeling partial differential equations. These kinds of problems can be easily expressed in this way and then the CNN simulator gives an easy-to-use way for numerical computations and experiments.

Second the CNN with complex-valued templates was introduced. Unlike usual CNN now all the templates, the states, the inputs and the outputs were complex. It was shown how a complex-valued CNN could be built of two real-valued CNN layers. This new structure is sufficient in case of several problems, especially when complex quantities are used (e.g., filters, problems concerning the frequency domain). Here two applications was explained: sine wave generation with a band-pass filter *A* template and another method generating sine waves on the basis of a series given by a recursive formula.

Third a method for black-and-white pattern detection was described. This example shows how to avoid the drawbacks of the CNN's local nature. This structure can detect 20-30 unit long black or white stripes with templates of size only 3x1. The algorithm used to measure the length of a stripe can be implemented in digital computers, too.

5. Acknowledgement

I would like to thank Tibor Kozek and Károly László for the review of my paper and for their valuable advices.

6. References

[1] A. M. Turing, "The chemical basis of morphogenesis", *Philos. Trans. R. Soc. London*, ser. B, vol. 237, pp. 37-72, 1952

[2] S. Kondo, R. Asai, "A reaction-diffusion wave on the skin of the marine angelfish Pomacanthus", *Nature*, vol. 376, pp. 765-768, 1995

[3] L. O. Chua, L. Yang, "Cellular neural network: Theory and Application", *IEEE Trans Circuits Syst.*, vol. 35, pp. 1257-1290,1988

[4] L. O. Chua, T. Roska, "The CNN Paradigm", *IEEE Trans. Circuits Syst.*, vol. 40(I), pp. 147-156, 1993

[5] Kozek, L. O. Chua, T. Roska, D. Wolf, R. Tetzlaff, F. Puffer, K. Lotz, "Simulating nonlinear waves and partial differential equations via CNN-Part II: Basic Techniques", *IEEE Trans. Circuits Syst.*, vol. 42(I), pp. 807-815, 1995

[6] Kozek, L. O. Chua, T. Roska, D. Wolf, R. Tetzlaff, F. Puffer, K. Lotz, "Simulating nonlinear waves and partial differential equations via CNN-Part II: Typical examples", *IEEE Trans. Circuits Syst.*, vol. 42(I), pp. 816-820, 1995

[7] L. O. Chua, M. Hasler, G. Moschytz, J. Neirynck, "Autonomous Cellular Neural Networks: A Unified Paradigm for Pattern Formation and Active Wave Propagation", *IEEE Trans. Circuits Syst.*, vol. I., pp. 559-577, 1995

[8] L. Nemes, G. Tóth, T. Roska and A. Radványi, "Analogic CNN algorithms for 3D interpolation-approximation and object rotation using controlled switched templates", *Int. Journal of Circuit Theory and Application,* vol. II., May 1996

[9] P. Thiran, K. R. Crounse, L. O. Chua, M. Hastler, “Pattern Formation Properties of Autonomous Cellular Neural Networks”, *IEEE Trans. Circuits Syst.*, vol. 42(I), pp. 757-775, 1995

[10] N. N. Aizenberg, I. N. Aizenberg, “CNN based on multi-values neuron as a model of associative memory for grey-scale images”, *Proc. of IEEE CNNA’92 Conferecnce*, pp. 36-41

[11] P. Szolgay, G. Vörös, Gy. Erôss, “ On the implementation of the cellular neural network paradigm in mechanical vibrating systems”, *IEEE Trans. Circuits Syst.,* vol. 40, pp. 222-227, 1993

[12] N. N. Aizenberg, Y. L. Ivaskiv, “Multi valued treshold logic”, Naukova Dumka Publisher House, Kiev, 1977

[13] K. R. Crounse, L. O. Chua, “Method for Image Processing and Pattern Formation in Cellular Neural Networks: A Tutorial”, *IEEE Trans. Circuits Syst.*, vol. 42(I), pp. 583-601, 1995

7. Figures

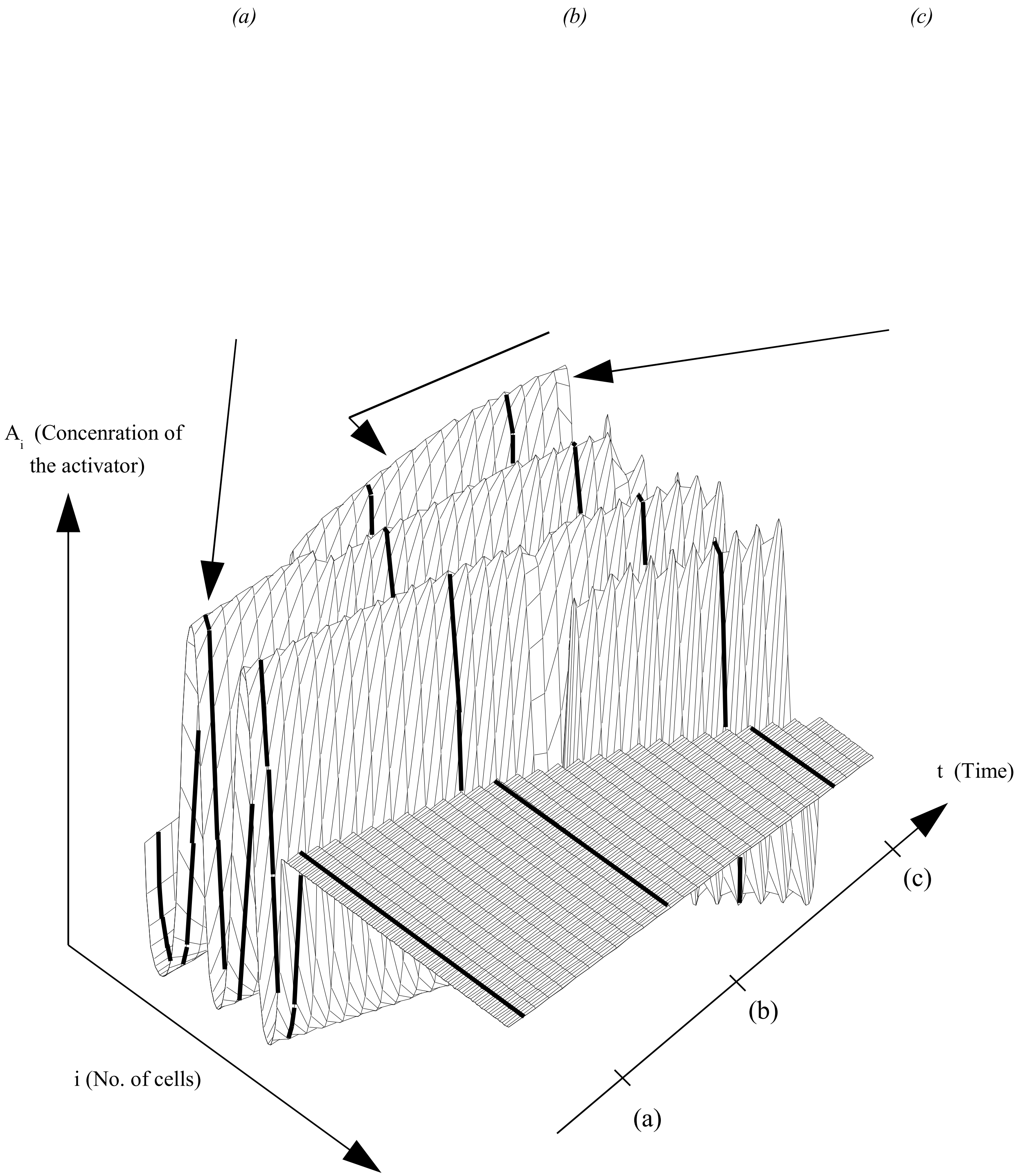
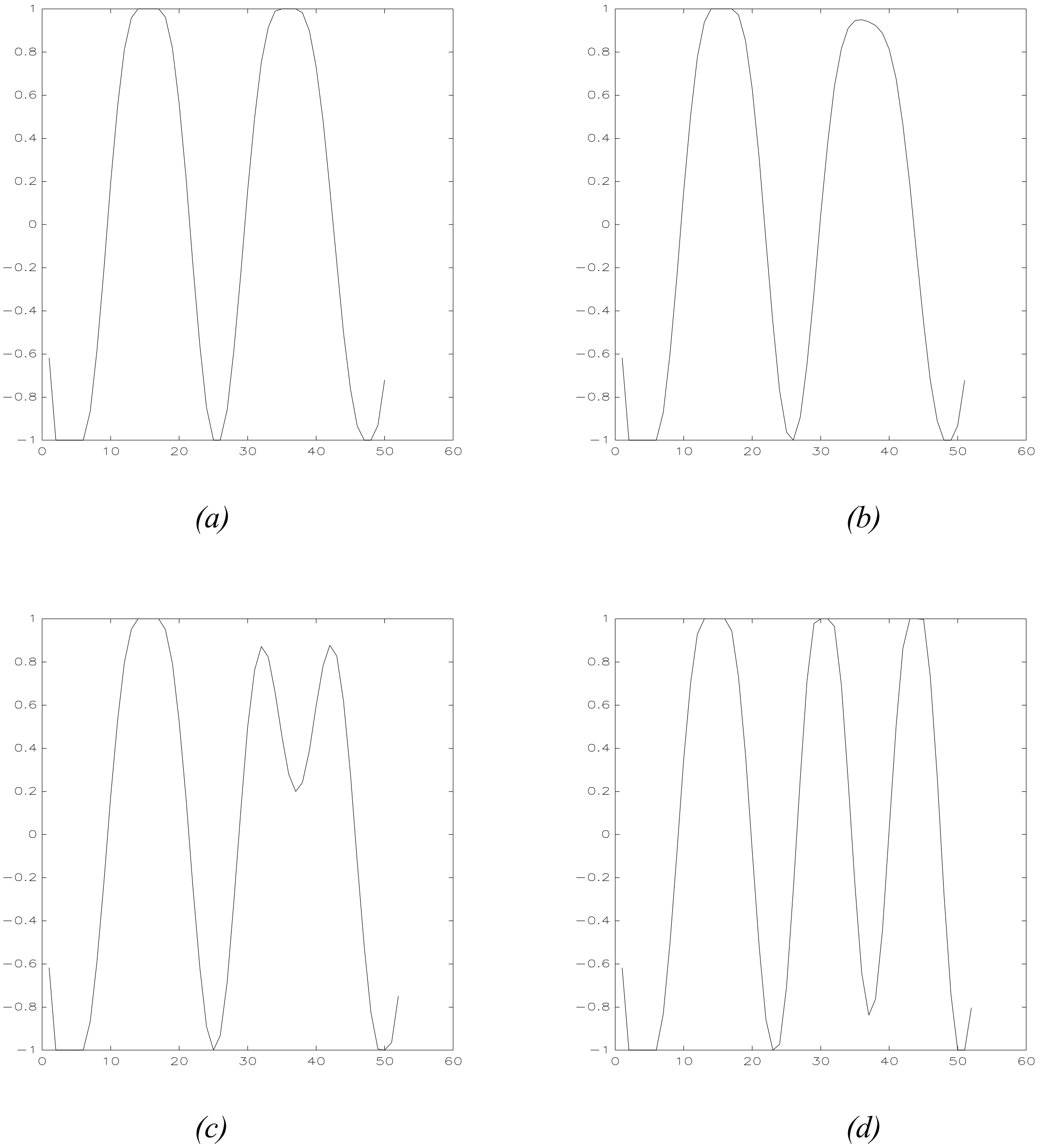


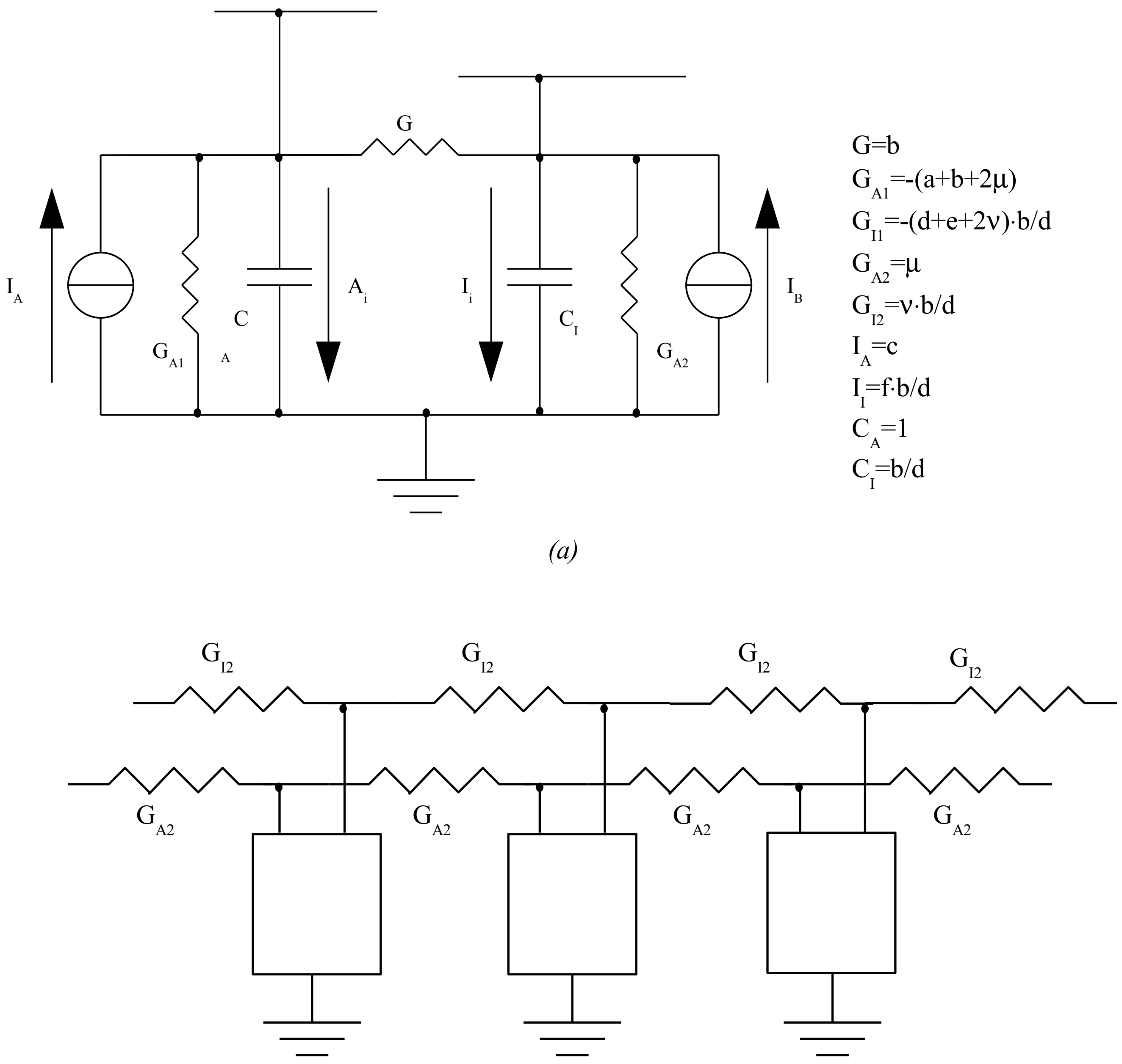
Figure 1. Stripes in an increasing cell array. At start the length of the cell block is 60. After a given time it increases by 5%. (One inner cell of a 20 cell block is duplicated.) Three snapshots are shown. (a) Two stripes and 60 unit long array. (b) Three stripes and 90 unit long array. (c) Four stripes and 115 unit long array.



(a) *(b)*

(c) *(d)*

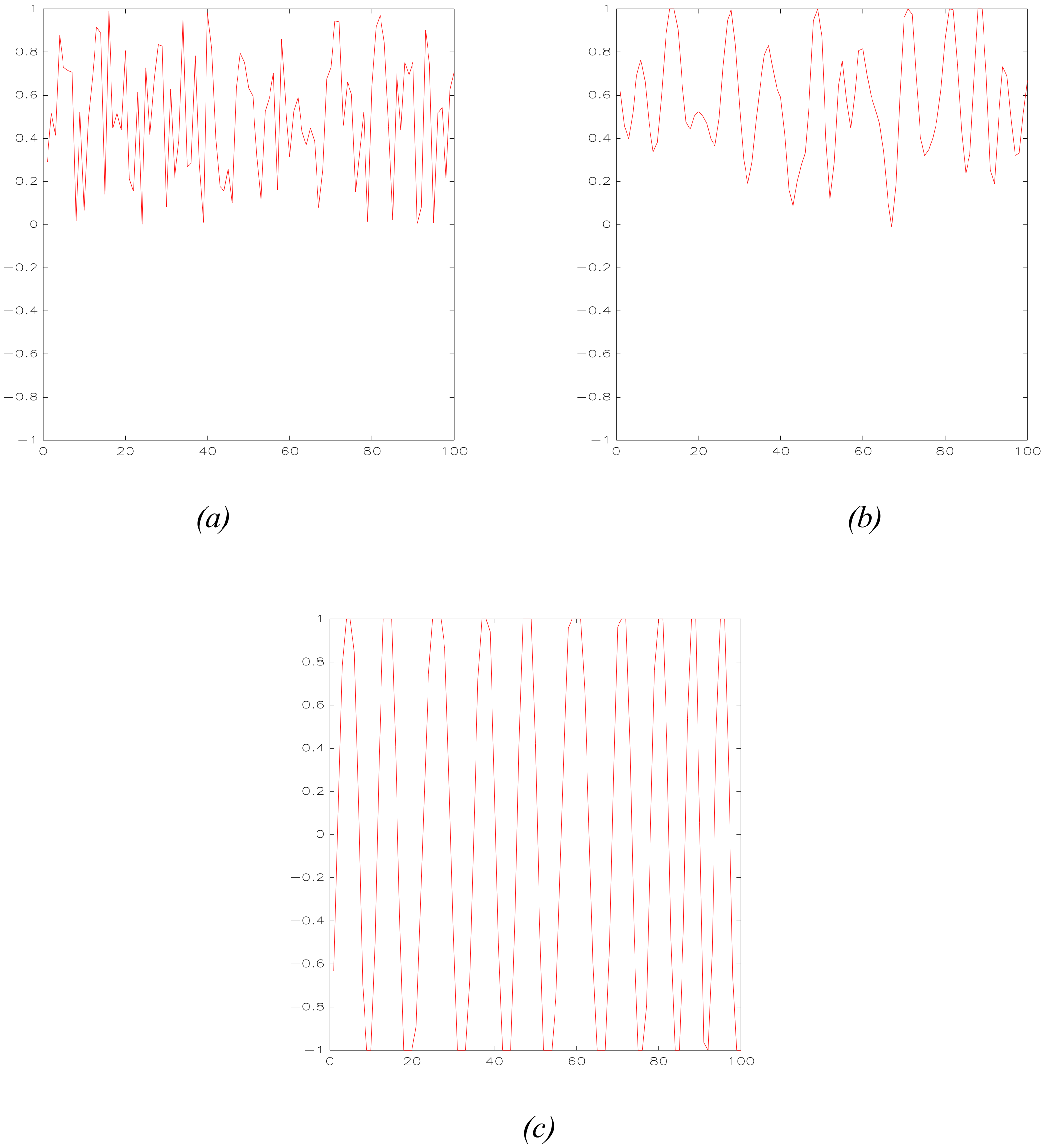
Figure 2. Stripes generated by CNN. The size of the CNN block grows. After every 37.5τ a new cell is added at the right edge. The sizes of the array are (a) 50, (b) 51, (c) and (d) 52. When the size of the array reaches 52 a new stripe appears.



(a)

(b)

Figure 3. Implementing Turing-equations using a CNN with second-order cells. (a) One cell and (b) intercell-connections The two state variables are the voltages of the cpacitors, denoted by A_i and I_i, respectively.

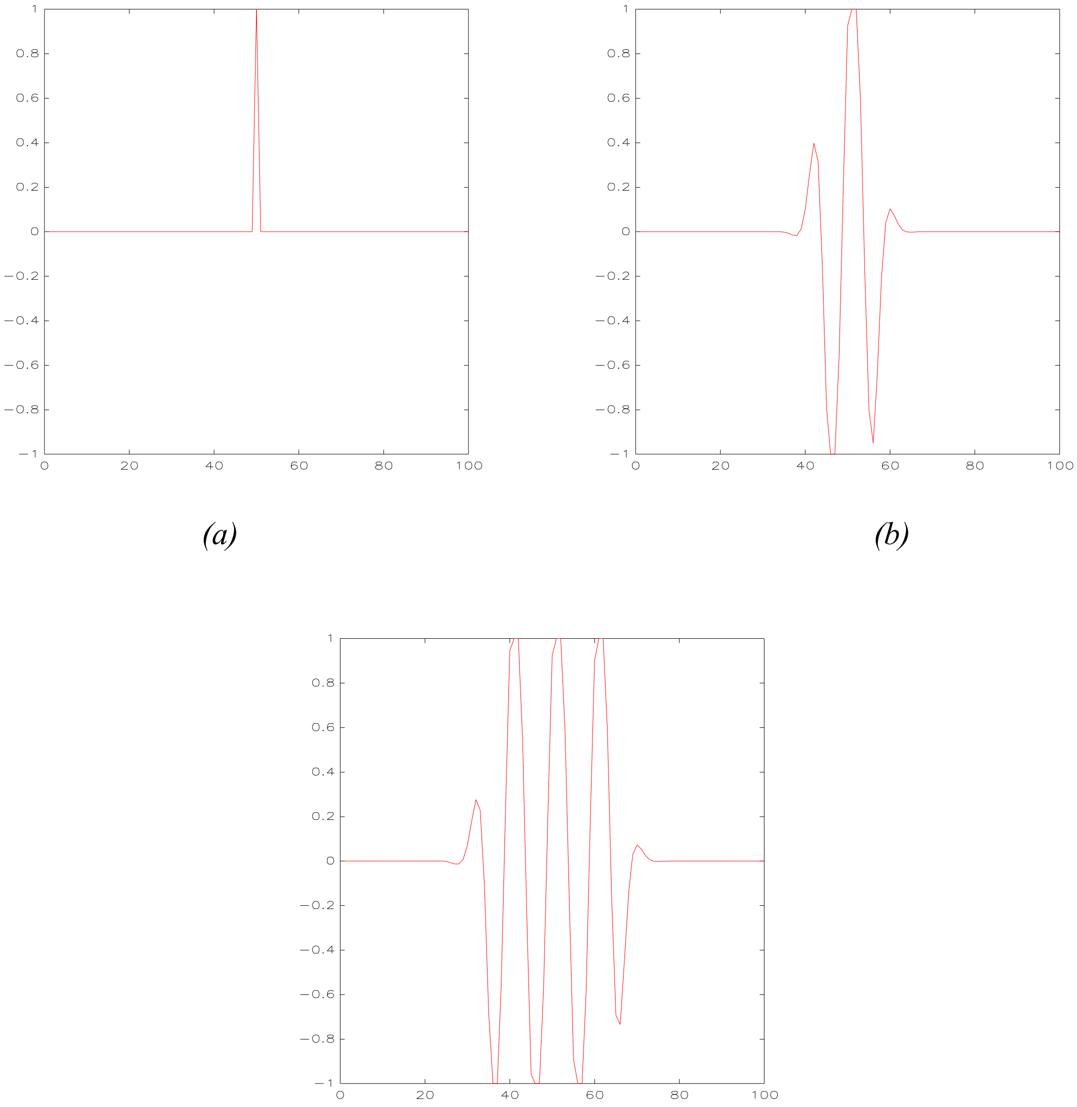


(a)

(b)

(c)

Figure 4. Generating spatial sine waves (with period-length of 10 units) designing in the frequency domain. The cells reached the saturation region, and it deforms the sinusoidal shape. (a) The random initial state, and the output of the real layer after (b) 10 and (c) 100τ can be seen.

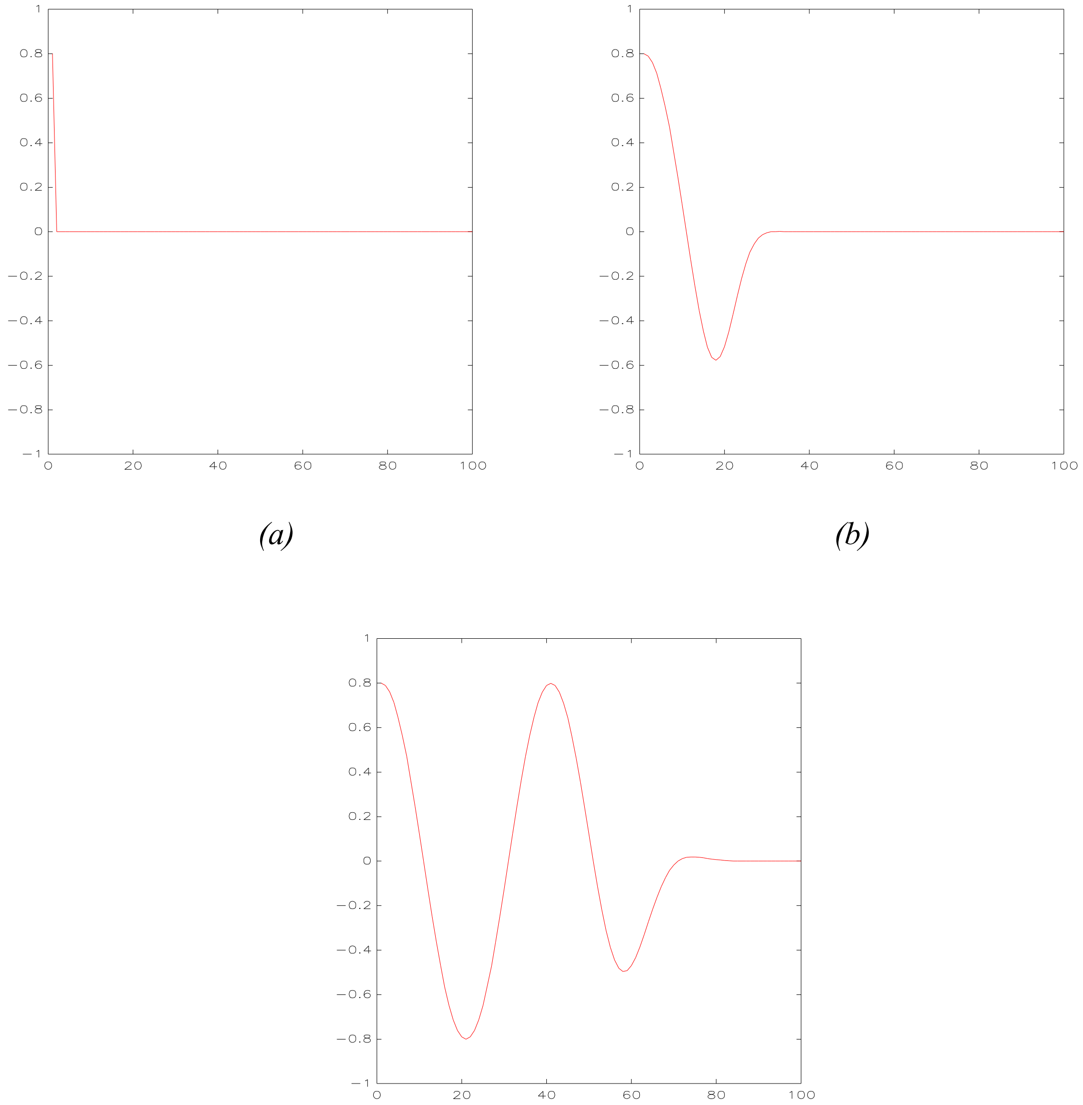


(a)

(b)

(c)

Figure 5. Generating spatial sine waves (with period-length of 10 units) designing in the frequency domain. The cells reached the saturation region, and it deforms the sinusoidal shape. (a) The initial state (that is a peak), and the output of the real layer after (b) 30 and (c) 60 τ can be seen. After 100τ the whole area is filled with sine peaks.



(a)

(b)

(c)

Figure 6. Generating sine waves (with period-length of 40 units) with complex-valued CNN templates. The state of the first cell is set to 0.8, and it is kept fixed. Zero is filled into the states of the other cells. It can be seen that the wave propagates from right to left. (a) The initial state and the output of the real layer can be seen after (b) 20 and (c) 60τ. After 100τ the whole area is filled with sine peaks.

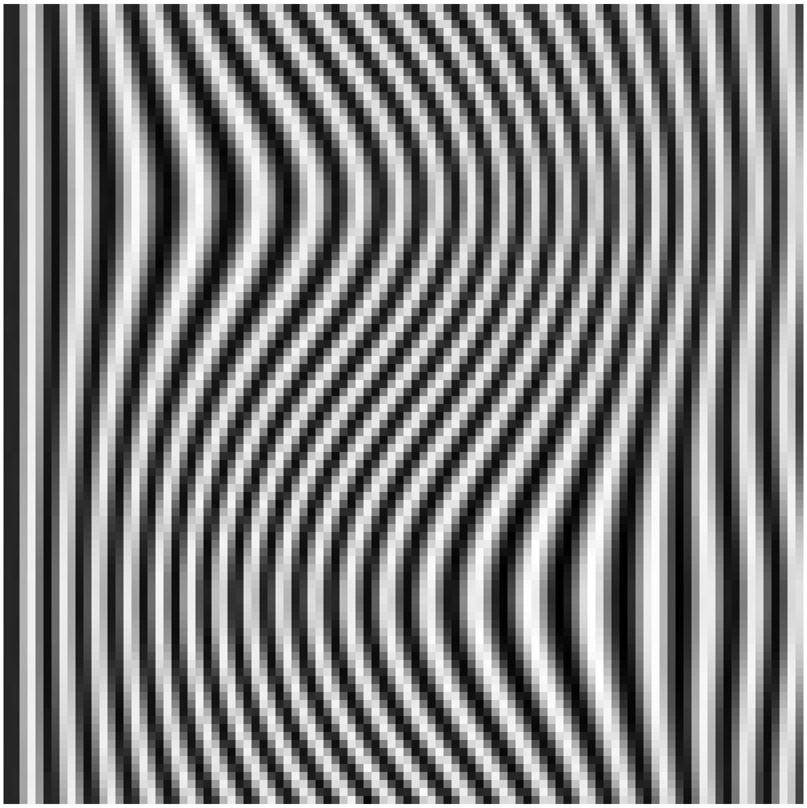
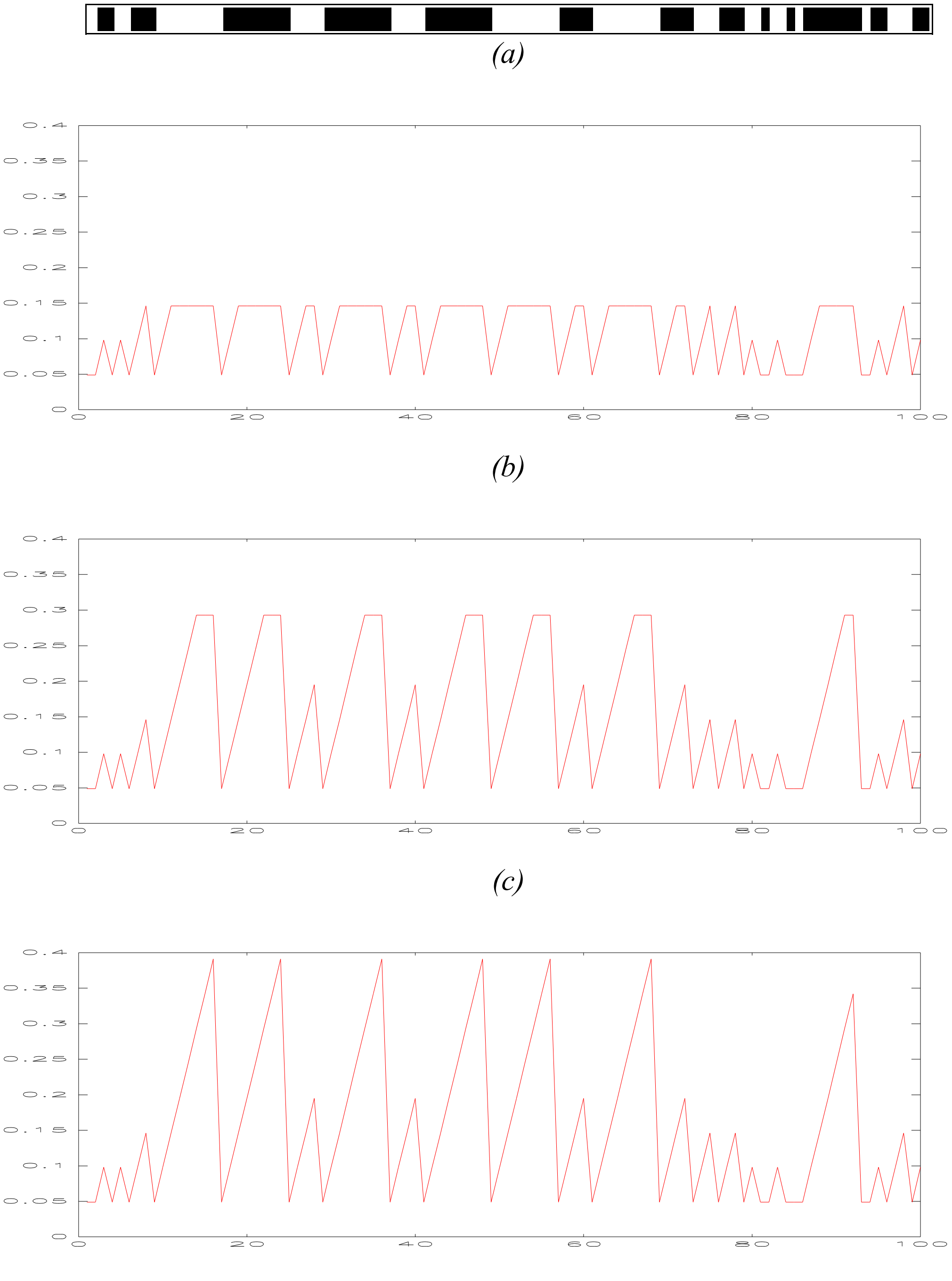


Figure 7. Generating stripes using frequency modulation by a control image. The image seems to be two-dimensional, in spite of this the network contains independent one-dimensional networks in the lines. The two-dimensional feature is due to the control image.



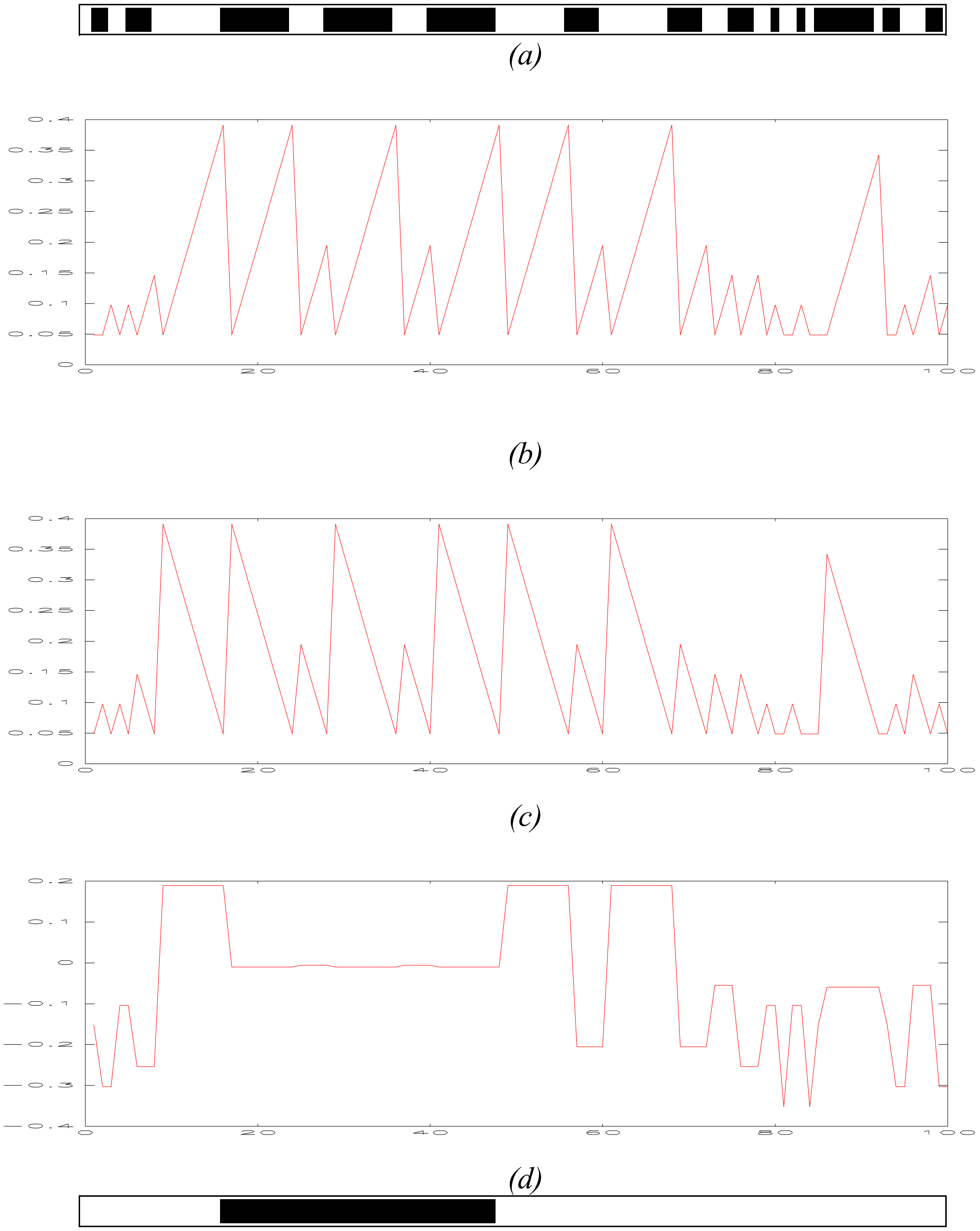
(a)

(b)

(c)

(d)

Figure 8. "Counting" from left to right The settled state of a cell is larger by 0.05 than the settled state of the previous cell and the count restarts at the border of a black or a white stripe. (a) The input. The output after (b) 3τ, (c) 6τ and (d) at settled state.



(a)

(b)

(c)

(d)

(e)

Figure 9. Detecting a pattern containing 8 unit long black stripes and 4 unit long space. (a) The input contains the black and white pattern to examine. (b) The output of the first layer shows "counting" left to right. This counting restarts at the borders of a black or a white area. (c) The output of the second layer shows counting right to left. (d) The state of the third layer is zero only under a black stripe of 8 units or a white stripe of 4 units. (e) The output of the third layer is black only if the state is near zero. It is achieved by a special output nonlinearity to detect zero level.

TABLE OF CONTENTS

1. Introduction....................1
2. CNN model of a one-dimensional Turing-type reaction-diffusion system found in Angelfish....................1
   2.1. Stripes arising from Turing-type reaction-diffusion equations....................1
   2.2. Realization using a 1D double-layer first-order CNN....................3
   2.3. Realization using a single-layer CNN with second order cells....................4
3. Complex valued CNN templates and their implementation....................4
4. Pattern generation with complex-valued templates....................5
   4.1. Sine wave generation with a band-pass filter A template....................5
   4.2. Sine wave generation with series given by recursive formula....................7
5. 1D black and white pattern detection with 3×1 templates....................8
4. Conclusions....................12
5. Acknowledgement....................13
6. References....................13
7. Figures....................14