ON THE IMPLEMENTATION OF THE SPATIAL PRISONER'S DILEMMA GAME ON THE CNN UNIVERSAL MACHINE

by

Laszlo Nemes and Leon O. Chua

Memorandum No. UCB/ERL M96/78

5 December 1996

ON THE IMPLEMENTATION OF THE SPATIAL PRISONER'S DILEMMA GAME ON THE CNN UNIVERSAL MACHINE

by

Laszlo Nemes and Leon O. Chua

Memorandum No. UCB/ERL M96/78

5 December 1996

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

On the Implementation of the Spatial Prisoner's Dilemma Game on the CNN Universal Machine

Laszlo NEMES, Leon O. CHUA
Department of Electrical Engineering and Computer Sciences,
University of California
Berkeley, CA 94720, USA
December, 1996

Abstract

The prisoners dilemma (PD) game is a simple, still surprisingly efficient model of the basic mechanisms of several different areas of science, such as economics, ecology, sociology and game theory. The model, as opposed to the Darwinian view of evolution, incorporates the possibility of cooperation between individuals. A relatively younger branch of the PD research is that of the spatial PDs. In the present paper a CNN implementation of the spatial PD model presented in [1] will be proposed along with the introduction of a new selection strategy. The similarities and differences between the two models are illustrated by simulation results.

1. Introduction

The Prisoner's Dilemma game (henceforth called PD) has been an extensively investigated topic of economics, ecology, sociology and game theory in the recent years. The idea of the game originates from A. W. Tucker [5] who proposed it in the early fifties. Since the birth of the original notion it has provided several new insights to the basic mechanisms of the different areas of science mentioned above. In spite of the relative simplicity of the model it has several parameters and displays a big variety of qualitatively different behaviors.

The model, as opposed to the Darwinian view of evolution, incorporates the possibility of cooperation between individuals. It also represents a crossing point between individual interest versus group interest since the payoff for defection is never worse than cooperation whereas the sum of benefits from mutual cooperation are always greater than in the case when one defects while the other cooperates.

A relatively younger branch of the PD research is that of the spatial PDs. It has been addressed in several papers recently, among which [1] served as a basis for the present paper. The authors had shown that with the given configuration is very sensitive for certain parameters and also presented evidence for the survival of a cluster of cooperators in a strongly defector favored environment.

In the following a CNN (Cellular Neural Network) [2] implementation of the spatial PD model presented in [1] will be proposed using a new concept of mapping a spatial extremum problem into the temporal domain, along with the introduction of a new selection strategy. The similarities and difference between the two models are illustrated by various simulation results.

2. Definitions, Strategies and Framework of the Prisoner's Dilemma Game

Before proceeding to the implementation part, some of the basic features and configuration parameters of the PD will be introduced in the following sections.

2.1 General Concept of the Prisoner's Dilemma game

The basic setup is as follows: there are two suspects (prisoners) who are indeed guilty of a certain felony and well aware of each others' affairs are facing the dilemma whether to confess or withhold the truth. Without the informations only known by the suspects neither of them can be fully proven guilty whereas giving out information about the other individual results in significant benefits. The person who withholds the truth about his partner will henceforth be called cooperator and the one who bears witness against the other will be called defector. The interactions of the two individuals yield four different outcomes depending on their decision:

1-2. One of the two suspects cooperates while the other one defeats (and the symmetric case). The cooperator will be heavily sentenced, while the defector will get negligible punishment.

3. If both cooperate, then they cannot be fully proven guilty of felony; hence they will be sentenced only a few years.

4. If both suspects defect, then they will be heavily sentenced but not as much as a cooperator against a defector.

2.2 The spatial Prisoner's Dilemma

There are many possible arrangements to simulate the Prisoner's Dilemma game. One of them is the so called spatial prisoners dilemma which has drawn the attention of researchers recently.
Let us consider the following setup for the Prisoner's Dilemma game:
A number of players are placed in the nodes of a 2D finite grid, they can be either cooperators (C) or defectors (D). In each iteration (generation) the players interact with their immediate neighbors and are rewarded according to a so called transition matrix:

	C	D
C	P_{CC}	P_{CD}
D	P_{DC}	P_{DD}

More general definition of the neighborhood is also possible, but we restrict ourselves to the one described above.
The entries of the transition matrix are the payoffs for CC, CD, DC, DD interactions, respectively (row-wise), subject to following constraints:

$$P_{DC} \geq P_{CC} \geq P_{DD} \geq P_{CD} \qquad (2\text{-}1)$$

that is defection against a cooperator is the most rewarding behavior for an individual. The second highest payoff comes from mutual cooperation. Mutual defection is rather poorly rewarded but the lowest paid behavior is that of the cooperator against a defector. Furthermore usually stands:

$$2\,P_{CC} > P_{DC} + P_{DD} \qquad (2\text{-}2)$$

This latter inequality provides the "drive" to form cooperative clusters, since the joint payoff is bigger than in the the total payoff of a defector cooperator pair.
The payoffs of interaction with the neighbors and possibly self interactions are summed up in each generation and the next state of each player is determined according to a pre-defined strategy discussed in details in the following sections, considering the payoff values belonging to each cell. The updating process when each player selects its next state at the same time (synchronously) is the so called synchronous updating. There is an other way of updating, the so called asynchronous updating, when only one player's state is updated at a time and the game is played over again with its partners. In the next iteration an other player's state will be updated.
The payoff of the player in position *i,j* (represented by its Cartesian coordinates):

$$p_{ij} = \sum_{kl} P(u_{ij}, u_{kl}) \qquad (2\text{-}3)$$

where p_{ij} is the payoff in position ij, $P(u_{ij},u_{kl})$ is the *transition function* which returns the corresponding values of the transition matrix for each configuration of u_{ij} cell and u_{kl} neighboring players.
We can summarize the various conditions (configuration parameters) which significantly affect the spatio-temporal behavior of the PD :

- the initial configuration (number and location of cooperators and defectors)
- the values of the transition matrix entries
- the strategies
- the updating method
- the presence or absence of self interaction

2.3 Strategies

Amongst the configuration parameters given in the previous section strategies to decide the player's next step considering the achieved payoffs play a crucial role. These strategies can be deterministic or probabilistic.
A strategy is deterministic if there is a prescribed response for each situation possible to occur. In the probabilistic case e.g. a certain probability proportional to the payoff value can be assigned to each neighbor with which the next state is selected.

There are also strategies with non-zero memory length. This means that the player considers not only the outcome of the current interaction but also the past iterations up to certain depth (determined by the memory length).

2.4. A special case

In the following we will consider a special case of the transition matrix . This transition matrix was thoroughly examined in [1]. Note that all matrix entries comply with inequalities (2-1) and (2-2). Despite its simplicity the matrix is general in a sense that it allows us to explore the full P_{DC}/P_{CC} range.
Let the transition matrix **P** be:

	C	D
C	1	0
D	b	0

where $2>b>1$ (*i.e.* $P_{CD}=0$ $P_{DD}=0$, $P_{CC}=1$, $P_{DC}=b$).

3. CNN Implementation

The spatial arrangement described in the previous section bears remarkable resemblance to that of the Cellular Neural Networks. In the following we show two examples of the CNN implementation of the spatial Prisoner's Dilemma.

3.1 Payoff calculation

Clearly, this specific type of the game consists of two consecutive steps:

1. Calculating payoffs
2. Selecting next state of the player (cell) accordingly

Step 1 is well suited for CNN implementation. Let us assign -1.0 to cooperators (C) and 1.0 to defectors (D), respectively and load the binary field containing the players' state to the input of the CNN . Introducing the following nonlinear function (Fig. 1),

$$\tilde{f}_1(u_{ij}, u_{i+k,j+l}) = \tilde{f}_1(2u_{ij} + u_{i+k,j+l}) \qquad (3\text{-}1)$$

This function implements the actual transition function for the PD.
Let us define the feedforward template:

$$\tilde{B}_{11}=\begin{bmatrix}\tilde{f}_1 & \tilde{f}_1 & \tilde{f}_1\\ \tilde{f}_1 & \tilde{f}_1 & \tilde{f}_1\\ \tilde{f}_1 & \tilde{f}_1 & \tilde{f}_1\end{bmatrix},$$

and, let A=0 , z=0

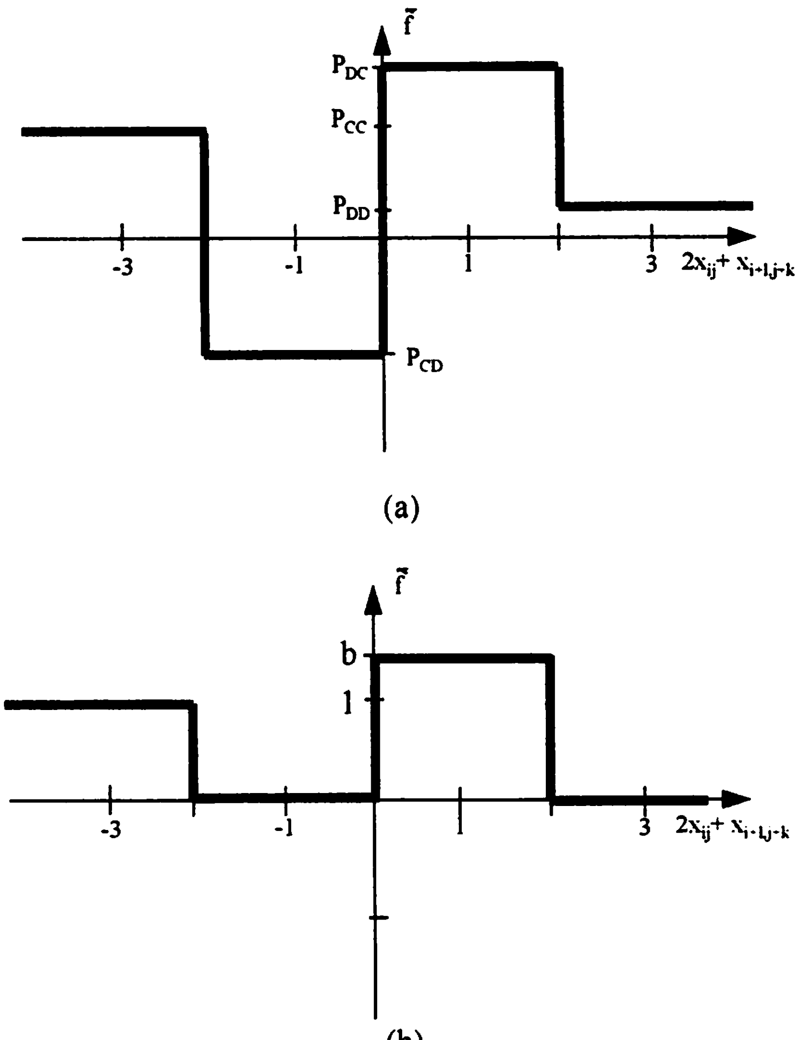


Fig. 1 Nonlinear function representing the transition function of the PD in the general case (a) and for the special case given in section 2.4

the state of the CNN will contain the sum of payoffs for each player in the corresponding position.
In the following sections, the implementation of two different selection strategy implementations will be presented.

3.2 The "Best Individual" Strategy

Step 2: The implementation of the selection strategy.

Definition:
After calculating the payoffs in each generation the state of the player belonging to the highest payoff in the neighborhood will be selected as the next state of each player in the next generation. This strategy will henceforth be referred as best individual strategy (BIS).

This is the strategy that was used in the experiments in [1].
The implementation of selection strategy described above is not trivial. The task to find the position of a maximum for the payoff function constrained by the local neighborhood for each cell is rather "ill-posed" for the regular CNN structure. It will be shown however, that with the proper selection of the output nonlinearity of the CNN cell, the spatial extremum problem can be translated into a more tractable temporal problem.
Let us assume that all entries of the transition matrix are non-negative which stands for the transition matrix given in [1]. Let us modify the nonlinear function (3-1) (transition function) given in the previous section the following way:

$$\tilde{f}'(u) = \mathrm{sgn}(u) \cdot \tilde{f}(u) \qquad (3\text{-}2)$$

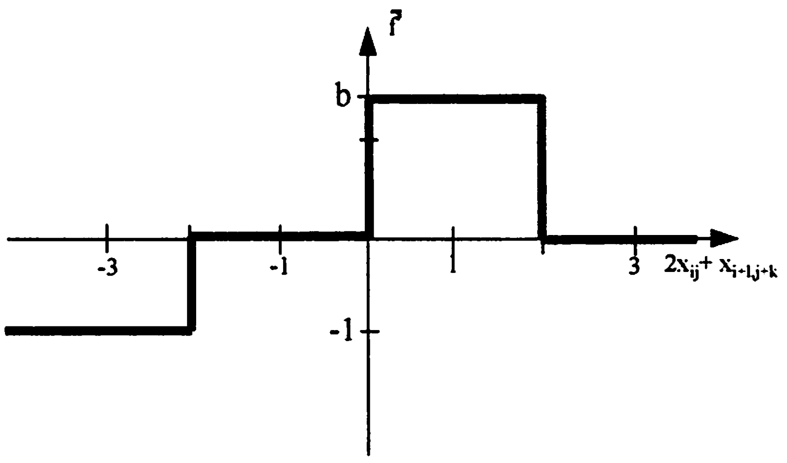
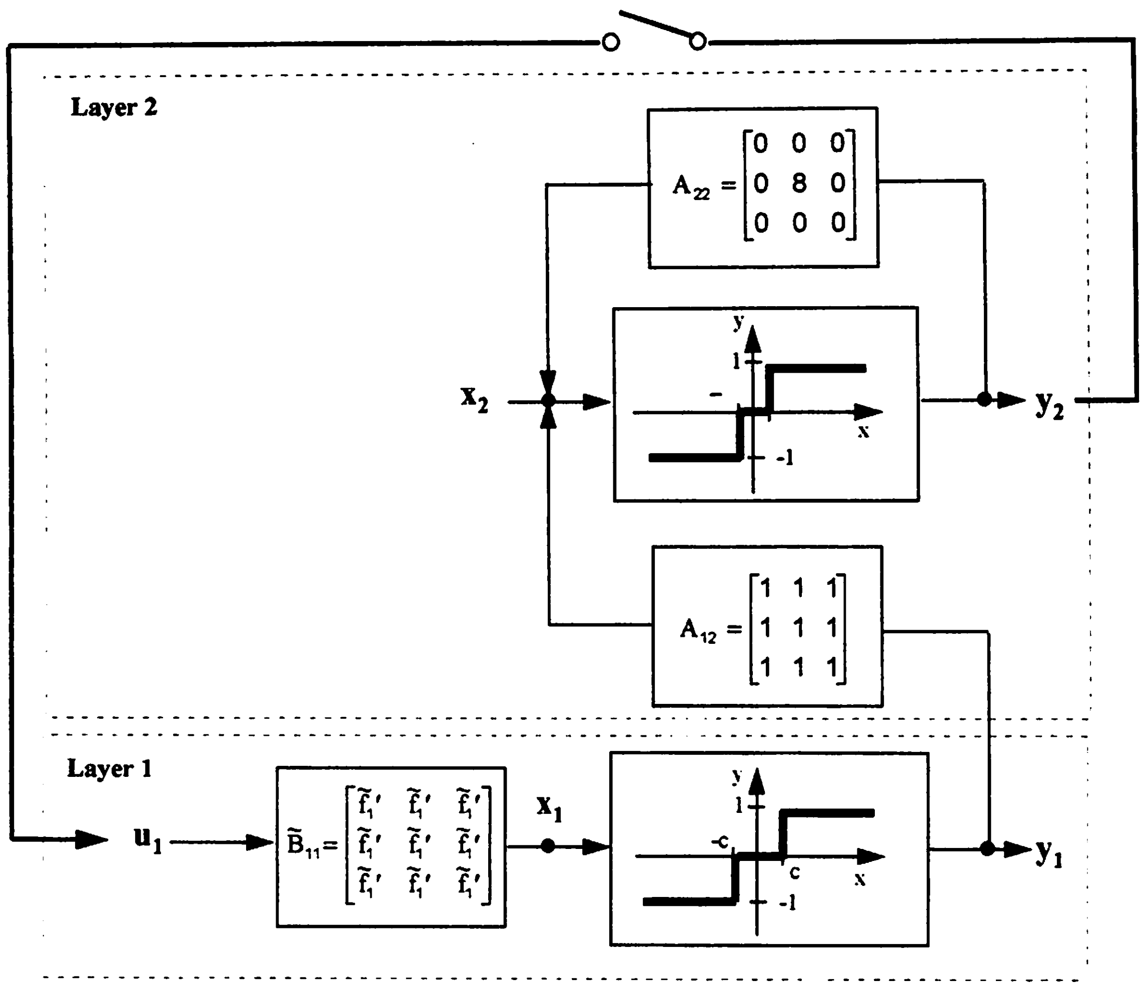


Fig. 2 Modified nonlinear function for the implementation of "best individual" strategy

The sign of the function argument determines the state of the central element. This modified function will provide negative payoffs for cooperators and positive payoffs for defectors.
Let us consider the two-layer arrangement in Fig. 3.



Layer 2
$A_{22} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 8 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
x_2
y
1
x
-1
y_2
$A_{12} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$
Layer 1
u_1
$\tilde{B}_{11} = \begin{bmatrix} \tilde{f}_1' & \tilde{f}_1' & \tilde{f}_1' \\ \tilde{f}_1' & \tilde{f}_1' & \tilde{f}_1' \\ \tilde{f}_1' & \tilde{f}_1' & \tilde{f}_1' \end{bmatrix}$
x_1
y
1
-c
c
x
-1
y_1

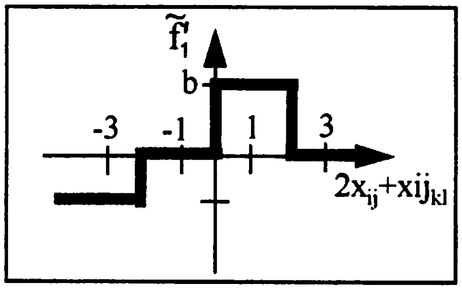


Fig. 3 CNN implementation of PD for "best individual" strategy using modified output nonlinearities. The nonlinear function implementing the transfer function is shown in the bottom of the figure

The CNN state equation for the first layer:

$$\dot{\mathbf{x}}_1 = \tilde{\mathbf{B}}'_{11} \ast \mathbf{u}_1 = \sum_r \tilde{f}_1'(\mathbf{u}_1, \mathbf{u}_{1r}) \qquad (3\text{-}3)$$

where: u_l the input matrix of layer 1 and u_{lr} contains the corresponding neighborhood values. $\ast$ denotes spatial convolution

Since the argument of the nonlinear function is constant throughout the transient (static input) and there is no feedback for the first layer the derivative of the state variable also remains constant. This implies that if the state of the CNN had initially been set to zero the order of the state values remains unchanged during the transient and complies with the order of the resulting payoffs . Let us consider the output nonlinearity of layer 1 in Fig. 2. If threshold value *c* is set to a positive value smaller than the lowest non-zero payoff, then all cell states whose derivatives are different from zero will reach this level sooner or later, though the order of reaching this threshold will concur with the final order of payoff values. This way we have translated the 2D spatial extremum problem into a 1D temporal problem. As soon as one of the cell state values reaches the threshold the corresponding output value will be the sign of the state variable due to the threshold type nonlinearity. The second layer serves as a "watchdog" circuit. It captures the first non-zero output of the first layer in the immediate neighborhood for each cell and fixes it on the output of layer two, thus yielding the state belonging to the highest payoff in the neighborhood.
(see Fig. 3 for an example of a transient and Fig 4 for the flowchart of the algorithm. In Fig 4 "field" is the "playground" of cooperators and defectors. In our case *field$_i$* is the input of the first layer while *field$_{i+1}$* is calculated as the output of the second layer)
The state equation of the second layer:

$$\dot{\mathbf{x}}_2 = \mathbf{A}_{12} * \mathbf{y}_1 + \mathbf{A}_{22}\mathbf{y}_2 \qquad (3\text{-}4)$$
$$y_2 = \tilde{f}_{out2}(x_2)$$

where $\tilde{f}_{out2}$ is the output nonlinearity for layer 2. This is practically a sign function with 2ε insensitivity band around zero to filter out noise and make the arrangement sufficiently robust.
since $\mathbf{A}_{12} * \mathbf{y}_1 = \sum_r y_{1,r}$ and $\mathbf{A}_{22} = 8$:

$$\dot{x}_2 = \sum_r y_{1,r} + 8\tilde{f}_{out2}(x_2) \qquad (3\text{-}5)$$

There are 4 different stages can be distinguished in the operation of the structure in Fig 3. corresponding to t_0-t_3 in Fig 4. Let us consider the cell characterized by its Cartesian coordinates i,j and its neighboring cells:

1. $t = t_0$ At the very beginning of the transient all cell state and output in layer 1 and layer 2 are zero
2. $t = t_1$ The state values corresponding to non zero payoffs in layer 1 are different from zero, though one of them reached threshold level *c*. Therefore the output of layer 1 is 0 and $\dot{x}_2 = 0$ according to the state equation.
3. $t = t_3$ One of the neighboring cell states in layer 1 passes *c* or *-c*. In the first layer's output a 1.0 or -1.0 value appears, respectively . The first term of the state equation (3-5) becomes strongly positive or negative depending on the first layer's output. As

the state value on the second layer passes ε the strong feedback "locks" this value on the output of layer 2.

4. $t = t_4$ More neighboring cells reach *c* on the first layer, but it has no effect on the locked values, because the second term of the state equation (3-5), once its differs from zero, is always greater than the first one.

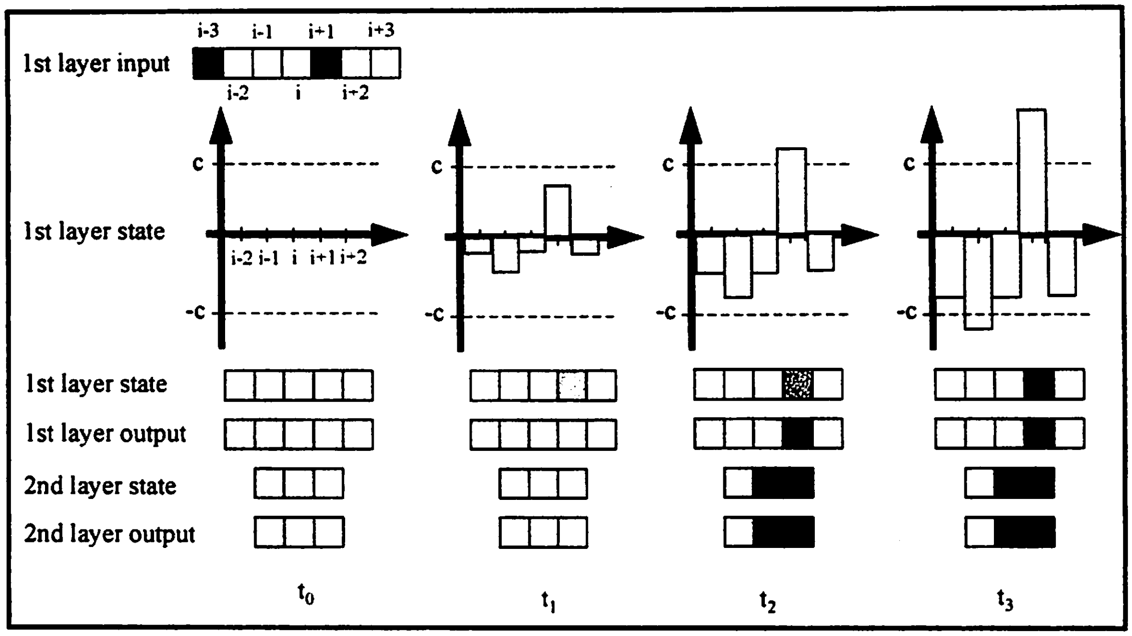


Fig. 4 A 1 dimensional example for transient in the structure implementing the BIS.

$t = t_0 = 0$ All layers' state and output values are 0

$t = t_1$ The state absolute values in layer 1 start to increase. Because of the modified transition function in Fig. 2 the state values corresponding to cooperators are changing toward -∞ whereas defector values are changing toward ∞. The order of their absolute values corresponds to the order of the final payoffs. e.g. in Fig. 4 the highest state value belongs to cell i+1 because this is a defector surrounded by two cooperators. The second is cell i-1, which is a cooperator surrounded by two other cooperators. The smallest values belong to cell i-2 , i and i+2. They are all cooperator cells surrounded by one defector and one cooperator neighbors

Since none of them has reached c output threshold the output of layer 1 and therefore the state and output of layer 2 are 0..

$t = t_2$ The state of cell i+1 has reached threshold c and output 1 at i+1 changes to 1. The state values of layer 2 at i+1 and i (which is a neighbor of i+1) start to increase and via the threshold type nonlinearity locks 1 on the output

$t = t_3$ *The state of cell i-1 has reached threshold -c and output 1 at i-1 changes to -1. At position i-1 on layer 2 the state value decreases toward -∞ and flips output at i-1 to -1. It has no effect on cell i though, because 1 has already been locked by the strong feedback at i during the previous phase*

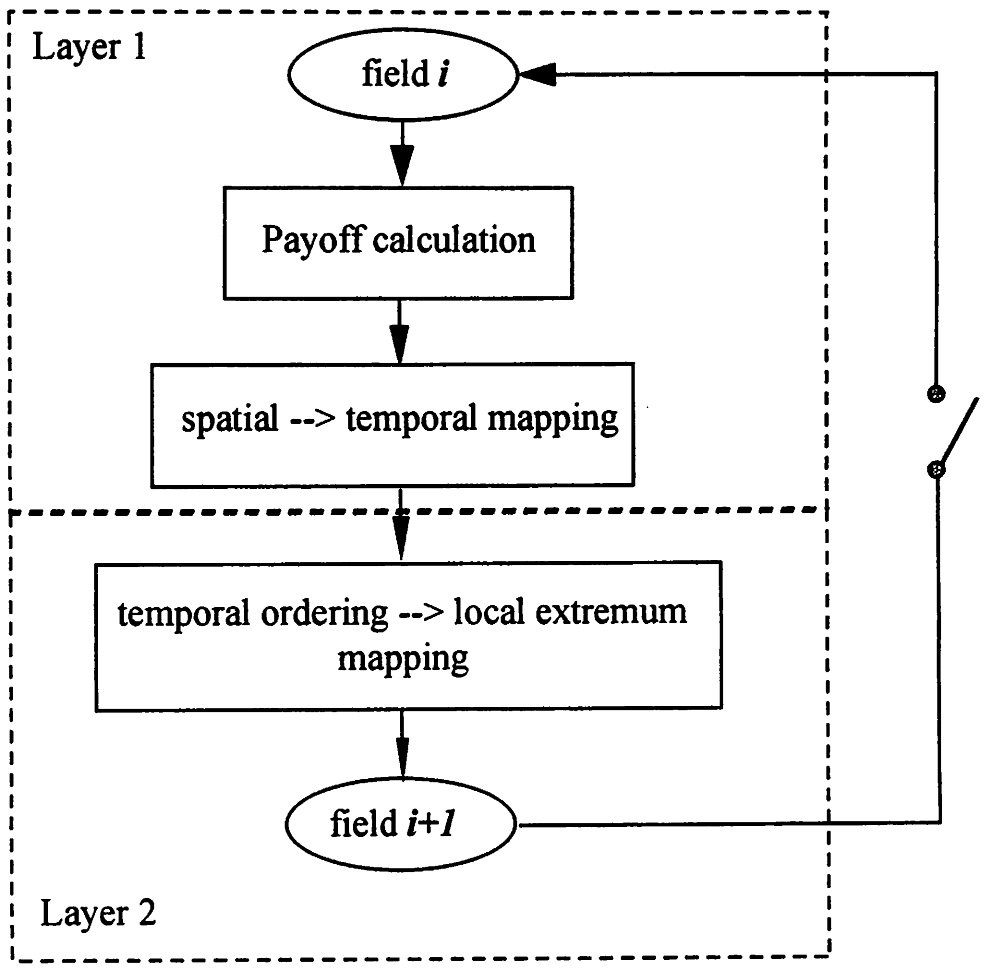
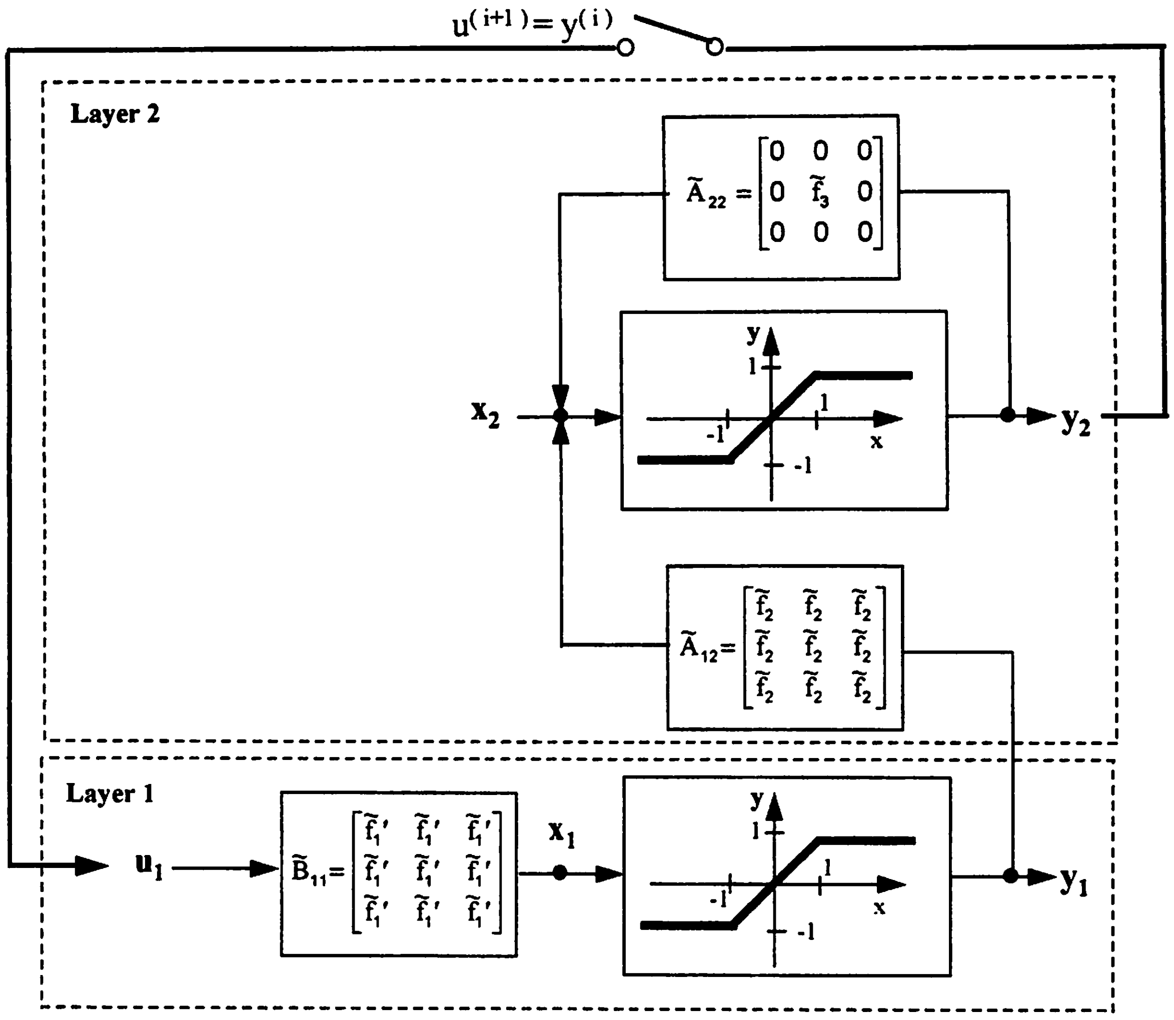

Layer 1
field *i*
Payoff calculation
spatial --> temporal mapping
temporal ordering --> local extremum mapping
field *i+1*
Layer 2

Fig. 5 Operational flowchart of CNN for BIS

Note that in the practical implementation it is impossible for more than one state values to reach threshold *c* at the same time due to random parameter variation of the particular implementation. This introduces a probabilistic selection between the cells that have theoretically the same payoffs. The final output of layer two will be the result of the "best individual" selection strategy. For those cells which don't have neighbors with non-zero payoffs the output will be zero.

This arrangement realizes one generation of PD for the "best individual strategy". In order to simulate more generations the output of the second layer must be loaded to the input of the first layer (The cell in locations with zero values in the second layer output keep their previous input values. See auxiliary decide circuit in section 3.4).

The same idea can be exploited using regular CNN cells with (regular output nonlinearities), though this requires more nonlinear templates and somewhat less effective in terms of computation speed. This arrangement can be seen in Fig. 6



$u^{(i+1)} = y^{(i)}$
Layer 2
$\tilde{A}_{22} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \tilde{f}_3 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
x_2
y_2
$\tilde{A}_{12} = \begin{bmatrix} \tilde{f}_2 & \tilde{f}_2 & \tilde{f}_2 \\ \tilde{f}_2 & \tilde{f}_2 & \tilde{f}_2 \\ \tilde{f}_2 & \tilde{f}_2 & \tilde{f}_2 \end{bmatrix}$
Layer 1
u_1
$\tilde{B}_{11} = \begin{bmatrix} \tilde{f}_1' & \tilde{f}_1' & \tilde{f}_1' \\ \tilde{f}_1' & \tilde{f}_1' & \tilde{f}_1' \\ \tilde{f}_1' & \tilde{f}_1' & \tilde{f}_1' \end{bmatrix}$
x_1
y_1

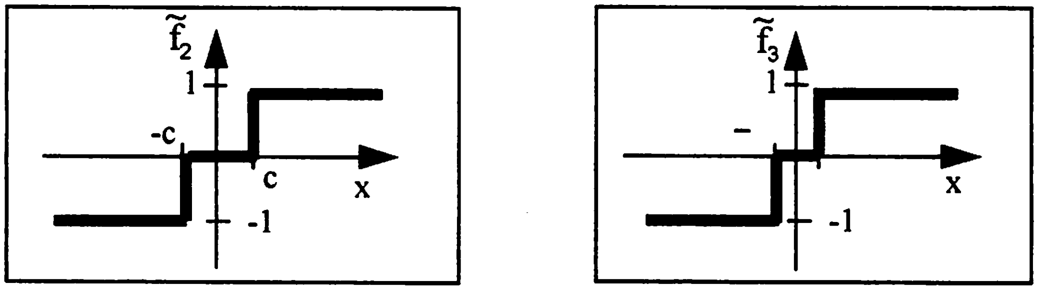


Fig 6 CNN implementation of PD for "best individual" strategy using regular CNN output nonlinearities.

3.3 "Better Average" Strategy

Let us consider a following strategy:

Definition:
The state for which better ***average*** *payoff has been achieved in the neighborhood is to be selected as next state. This selection strategy will henceforth be referred as better average strategy (BAS).*

This strategy is quite similar, though not identical to the original. This is also a valid strategy from the practical point of view (possibly even more effective) and unlike the previous one it can be implemented serially on single layer CNN Universal Machine.

Let us decompose the nonlinear function used in the previous section into two functions, one of which will reward only cooperators ($\tilde{f}_1^{(-)}$) and the other one only defectors ($\tilde{f}_1^{(+)}$) in central position. Let us introduce the following three layers (Fig 4(a)):

1 The output of this layer contains the negative payoffs for cooperators after a fixed time:

$$\begin{aligned} \dot{x}_1 &= \tilde{B}_{11} * u^{(i)} \\ y_1 &= \tilde{f}_{out1}(x_1) \end{aligned} \qquad (3\text{-}6)$$

Furthermore $\mathbf{y_1 = x_1} = c(t)\,\mathbf{p}^{(-)}$ stands throughout the transient, where $\mathbf{p}^{(-)}$ is the negative sum of the payoffs achieved by cooperators in the neighborhood supposing we have set the template and timing values in order to keep state values in the linear domain of the standard output nonlinearity and c(t) is time dependent coefficient.

2 The output of this layer contains the payoffs for defectors. Similarly to the previous:

$$\begin{aligned} \dot{x}_2 &= \tilde{B}_{22} * u^{(i)} \\ y_2 &= \tilde{f}_{out2}(x_2) \end{aligned} \qquad (3\text{-}7)$$

and $\mathbf{y_2 = x_2} = c(t)\,\mathbf{p}^{(+)}$ stands throughout the transient, where $\mathbf{p}^{(+)}$ is the sum of the payoffs achieved by defectors in the neighborhood

3 The state of this layer contains the sum of the cells in the binary input for the given neighborhood in each cell position. This discrete values (ranging from -9 to 9) enable us to determine the ratio of the cooperators and defectors for the given neighborhood in each cell position. We use two different output nonlinearities for this layer to calculate the proper weighting coefficients for cooperators and defectors:

$$\begin{aligned} \dot{x}_3 &= \sum_r u_r^{(i)} \\ y_3 &= \tilde{f}_3^{(-)}(x_3) = w^{(-)} \\ y_4 &= \tilde{f}_4^{(-)}(x_3) = w^{(+)} \end{aligned} \qquad (3\text{-}8)$$

where $\mathbf{w}^{(-)}$ and $\mathbf{w}^{(+)}$ are the weighting coefficients for cooperators and defectors, respectively and

$$\frac{w^{(-)}}{w^{(+)}} = \frac{\text{number of cooperators}}{\text{number of defectors}} \qquad (3\text{-}9)$$

and

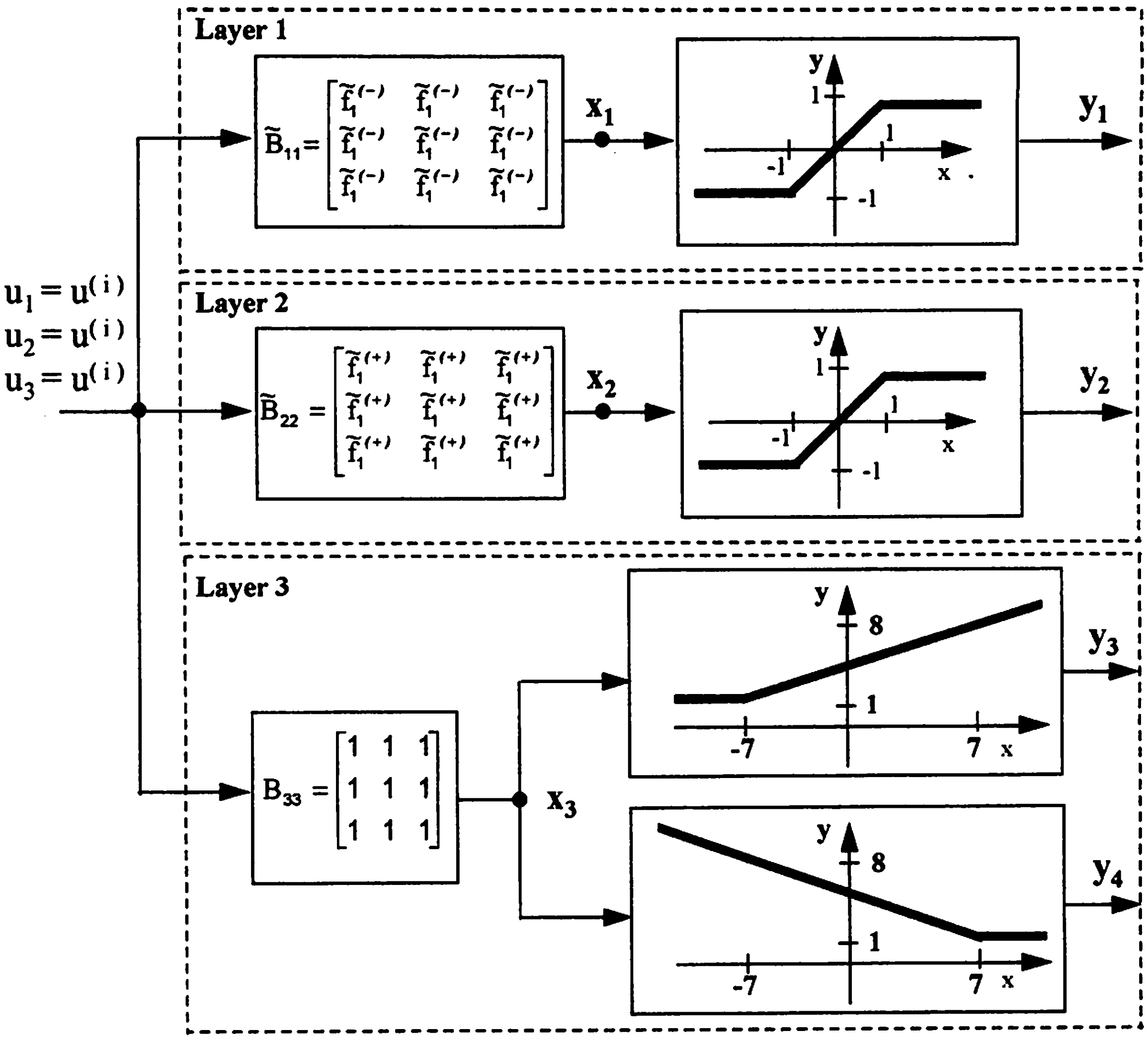
$$\tilde{f}_3^{(-)}(x) = \begin{cases} 0.5(x+9) \text{ if } x \geq -7 \\ 1 \text{ if } x < -7 \end{cases} \qquad (3\text{-}10)$$

$$\tilde{f}_4^{(-)}(x) = \begin{cases} 0.5(9-x) \text{ if } x \leq 7 \\ 1 \text{ if } x > 7 \end{cases} \qquad (3\text{-}11)$$

(Note that x can take only discrete values.)
Throughout the discussion we assume that the timing values have been set to comply with the previous equations (e.g. c(t) = 0). If this assumption does not hold all equation and function values have to be normed accordingly.
As a second step we have to multiply the weighting coefficients with the proper payoff values then sum the results in order to obtain the weighted average payoff. The sign of this value will whether cooperator or defectors achieved better within the neighborhood of each cell. Thus using a sign type nonlinearity for the layer which sums up the weighed payoffs provides with the desired output which will serve as the input configuration for the next generation (Fig. 4(b)):

$$u^{(i+1)} = y^{(i)} = \boldsymbol{sgn}\left(w^{(-)}p^{(-)} + w^{(+)}p^{(+)}\right) \qquad (3\text{-}12)$$

Note that though we talk about a three layer parallel structure, all calculations can be done sequentially thus can be implemented on a single layer CNNUM with proper data transfer between the analog memories. Nevertheless this approach results in definitely slower simulation (which might still not be a problem considering the tremendous computational speed of the analog circuitry).



Layer 1
$\tilde{B}_{11}=\begin{bmatrix}\tilde{f}_1^{(-)} & \tilde{f}_1^{(-)} & \tilde{f}_1^{(-)}\\ \tilde{f}_1^{(-)} & \tilde{f}_1^{(-)} & \tilde{f}_1^{(-)}\\ \tilde{f}_1^{(-)} & \tilde{f}_1^{(-)} & \tilde{f}_1^{(-)}\end{bmatrix}$
x_1
y_1
Layer 2
$\tilde{B}_{22}=\begin{bmatrix}\tilde{f}_1^{(+)} & \tilde{f}_1^{(+)} & \tilde{f}_1^{(+)}\\ \tilde{f}_1^{(+)} & \tilde{f}_1^{(+)} & \tilde{f}_1^{(+)}\\ \tilde{f}_1^{(+)} & \tilde{f}_1^{(+)} & \tilde{f}_1^{(+)}\end{bmatrix}$
x_2
y_2
$u_1=u^{(i)}$
$u_2=u^{(i)}$
$u_3=u^{(i)}$
Layer 3
$B_{33}=\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$
x_3
y_3
y_4
y 1 -1 1 -1 x
y 8 1 -7 7 x

Fig 7(a) The first stage of CNN structure (or analogic program) implementing "Better average" strategy for spatial Prisoner's Dilemma game

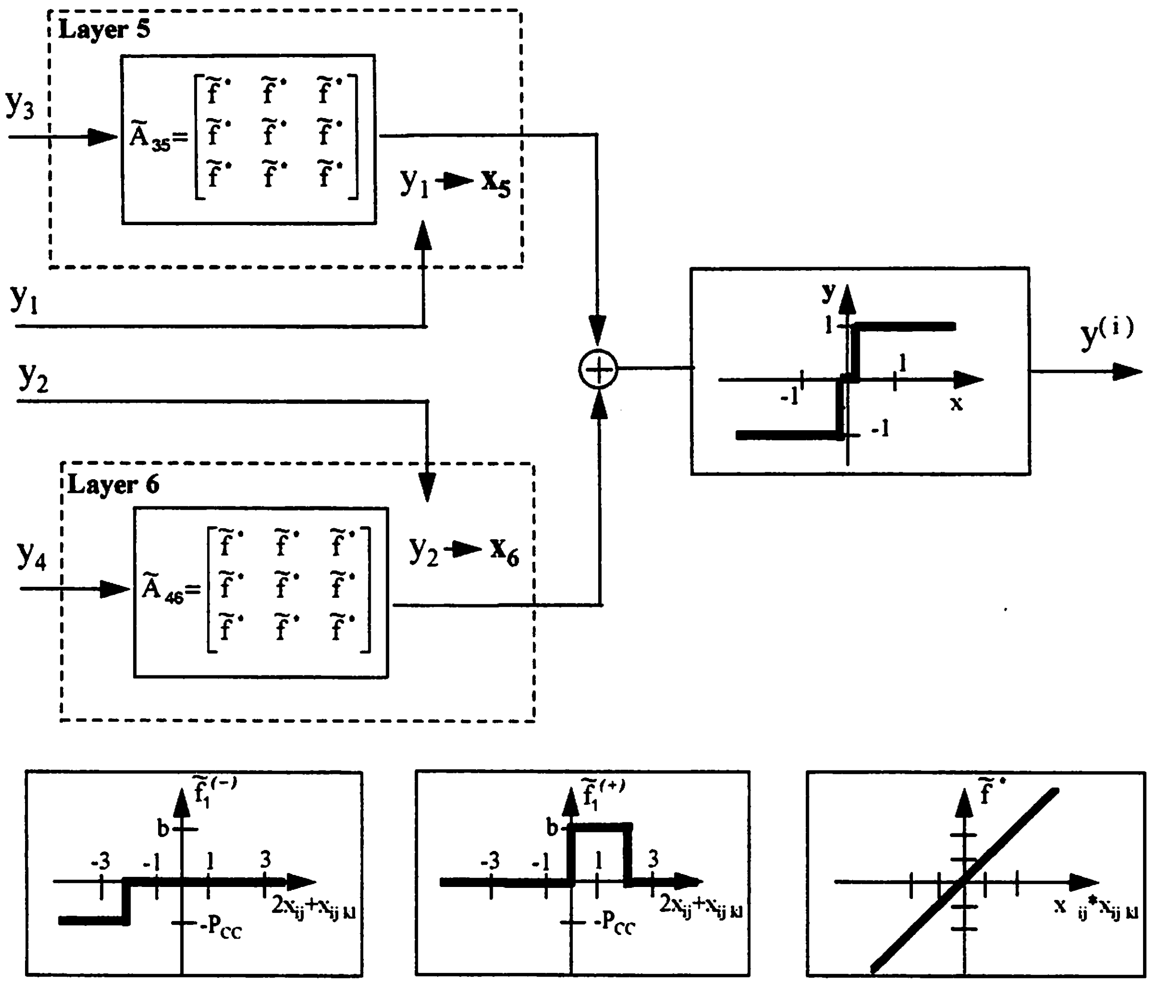


Fig 7(b) The second stage of CNN structure (or analogic program) implementing "Better average " strategy for spatial Prisoner's Dilemma game

3.4 Remarks on the implementation

Note that in the implementation of both strategies we used sign type output nonlinearities with insensitivity band. This is because in some cases the resulting state is zero and the unavoidable asymmetry in the implementation of the sign function would lead to the destruction of the PD's symmetry . The resulting output will not be binary but will also contain values close to zero. It seems to be reasonable to keep the previous value of the player in these positions. For this purpose the following auxiliary operation is proposed:

$$u_{i+1}=sgn(u_i+2y_i) \qquad (3\text{-}13)$$

The scheme of the corresponding CNN operation can be seen in Fig. 5

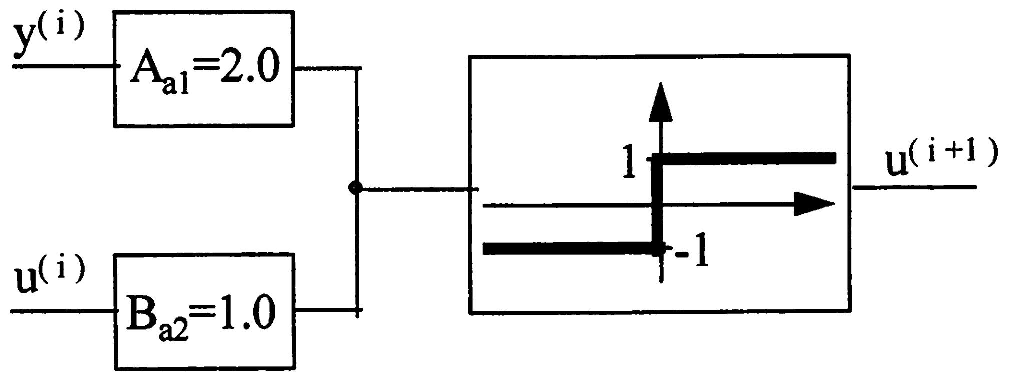


Fig. 8 Auxiliary circuit to assign value to cell inputs for the next generation undetermined by the given strategy

4. Simulation results

Both arrangements have been tested for several different values of *b* as well as for different initial conditions. The results for "best individual" strategy concur with the results published in [1].
In Fig. 9 simulation results can be seen for different values of b = P_{DC}. There are perceptible qualitative differences : for b=1.77 after 30 iteration a rather static structure seems to evolve whereas for b=1.85 a dynamic equilibrium develops with significantly larger proportion of defectors (the color coding of the last row in Fig 9 shows the dynamics of the fields. Red and Blue pixels denote static values. The color coding is explained in detail in the caption of Fig. 9 and in the Appendix)
Fig. 10 and Fig. 11 show The process as a defector invades the field of cooperators for "best individual" strategy (b=1.85). Fig. 12 is the same for "better average" strategy.
The resulting patterns have different structures. The BIS output consists of large blocks, while results obtained by using BAS have fractal-like structures.
Fig. 13 and 14 demonstrates that a sufficiently large cluster of cooperators can survive starting in the middle of a field of cooperators.
Fig. 15 shows the state of large fields after 100 iteration for both defector and cooperator invasion.

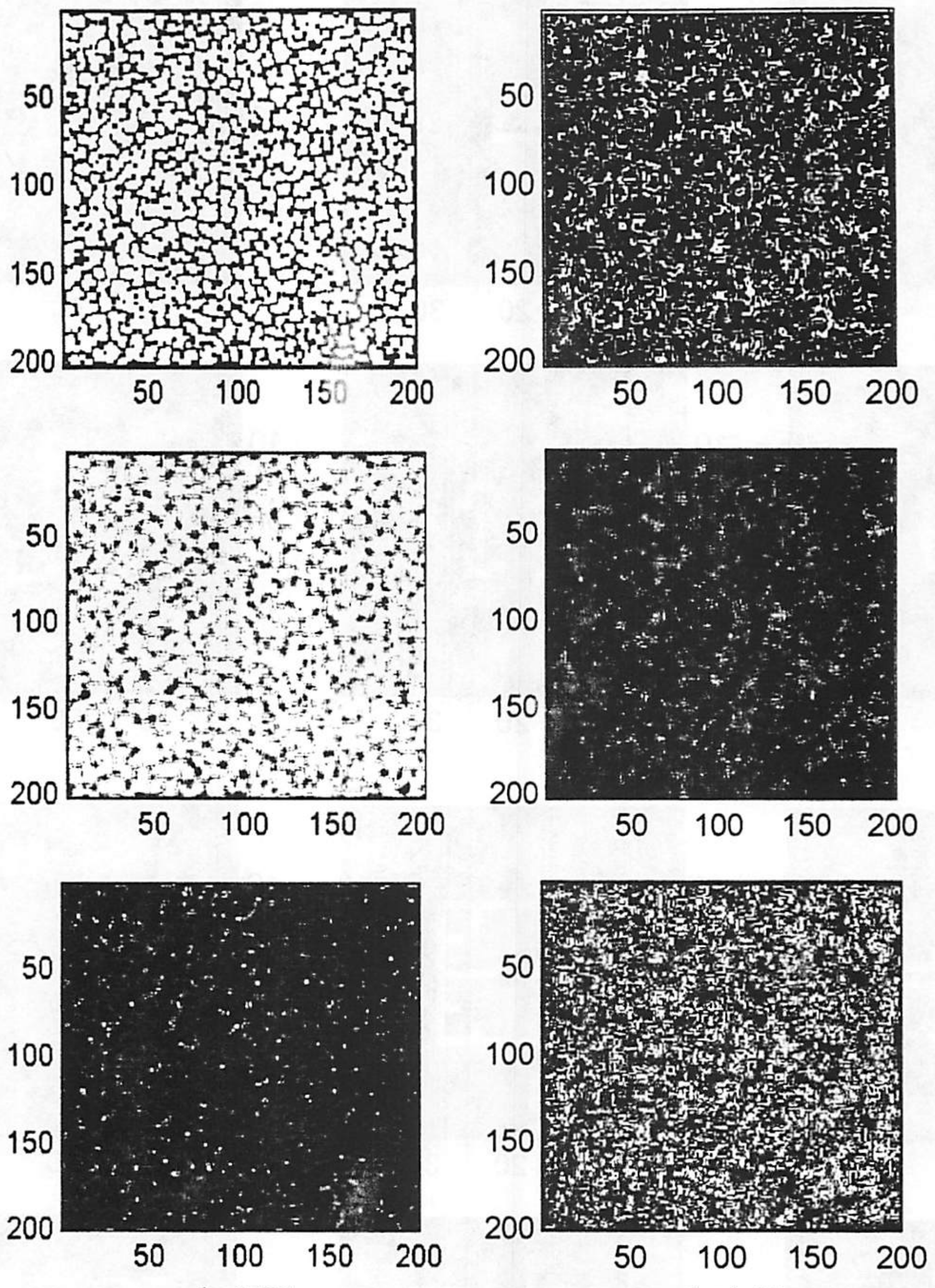


Fig. 9

Simulation results for "best individual" strategy for b=1.77 (left column) and b=1.85 (right column) after 33 iteration. The fist row (static fields) shows the distribution of cooperators (white pixels) and defectors (black pixels). The second row (payoff fields)shows the payoff distribution in hot colormap (yellow corresponds to higher values, red and black correspond to lower values). The last row's (dynamic fields) color coding is as follows: blue represents a cooperator (C) which was already a C in the preceding generation; red is a defector (D) following a D; yellow is a D following a C and green is a C following a D. All images are 200x200, the initial configuration was random with cooperator frequency 0.9 (For color coding see Appendix).

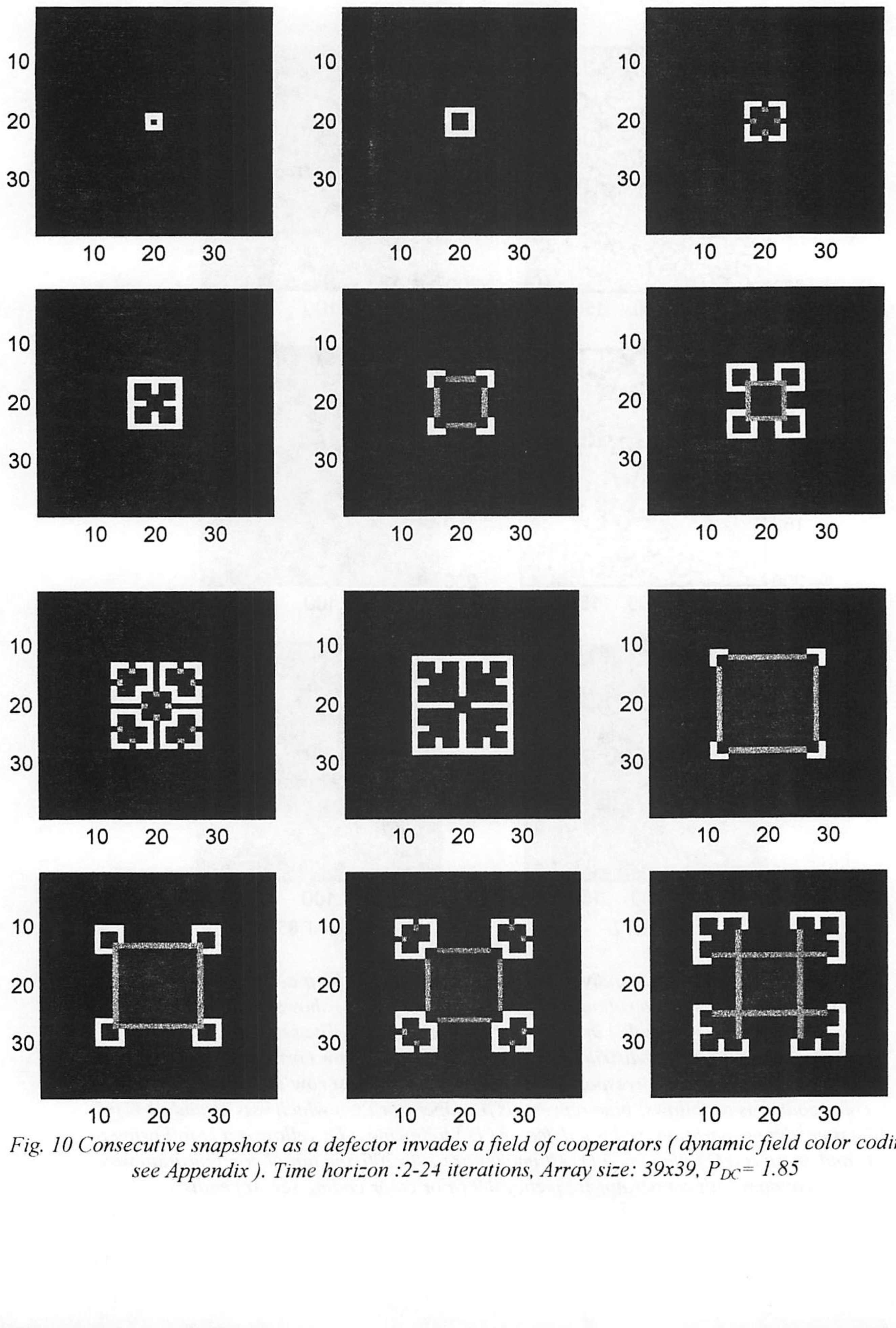


Fig. 10 Consecutive snapshots as a defector invades a field of cooperators (dynamic field color coding. see Appendix). Time horizon :2-24 iterations, Array size: 39x39, P_{DC}*= 1.85*

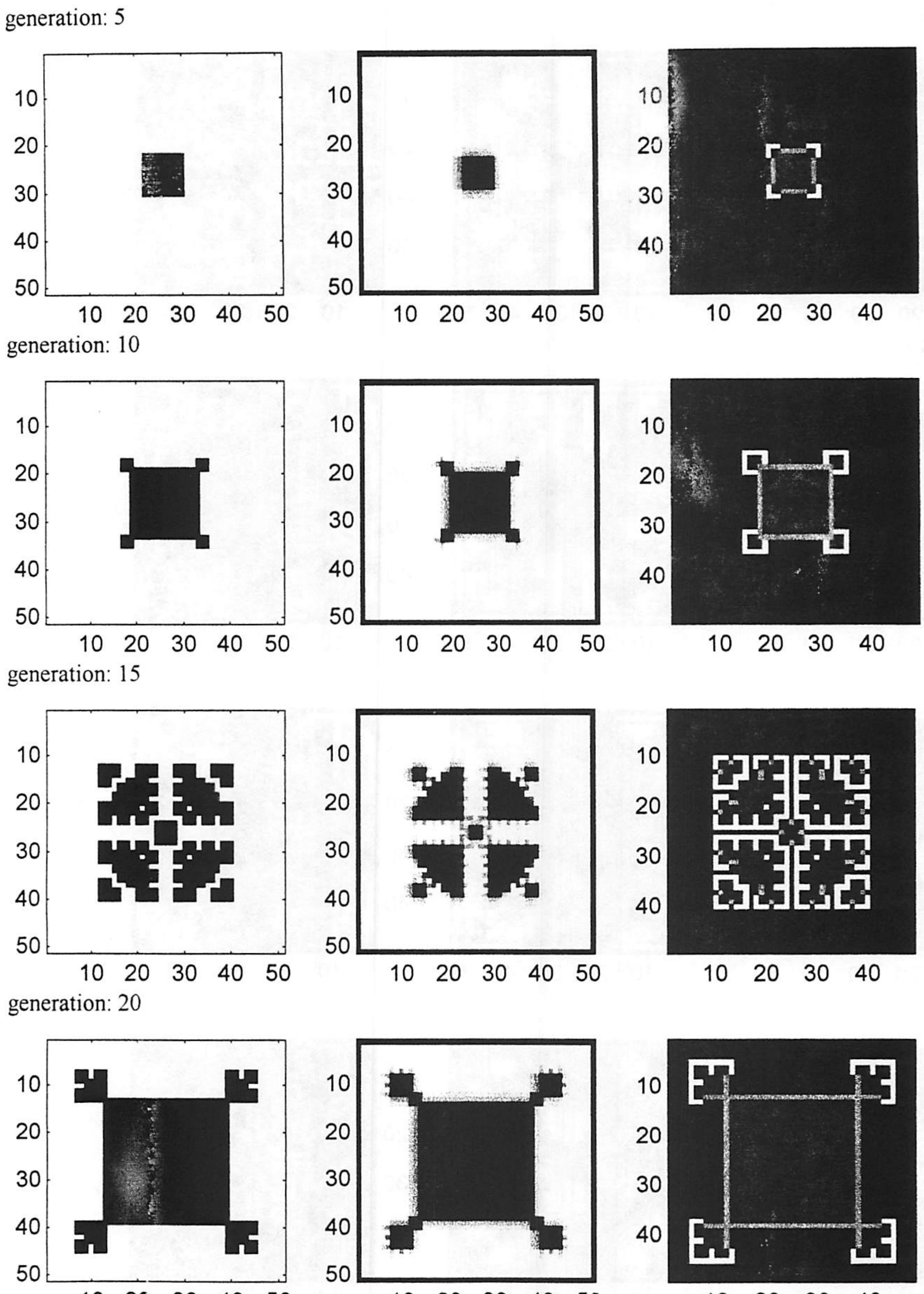
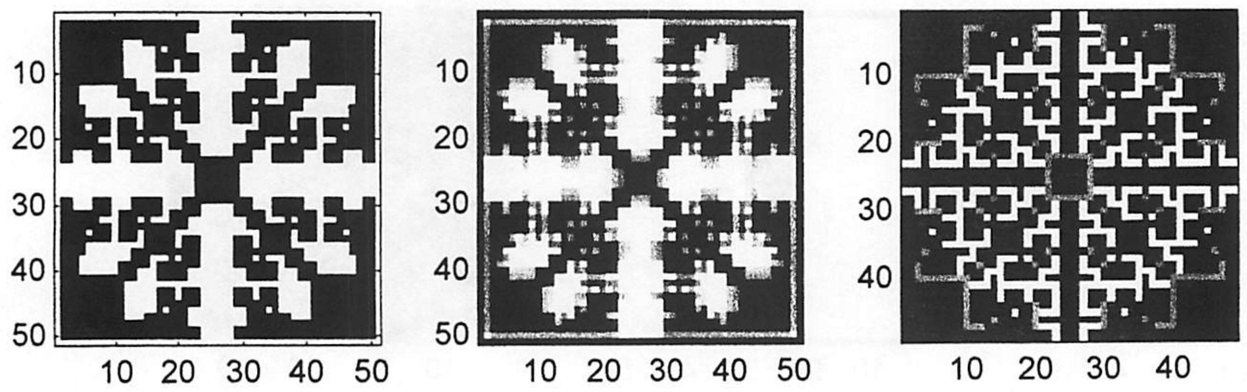
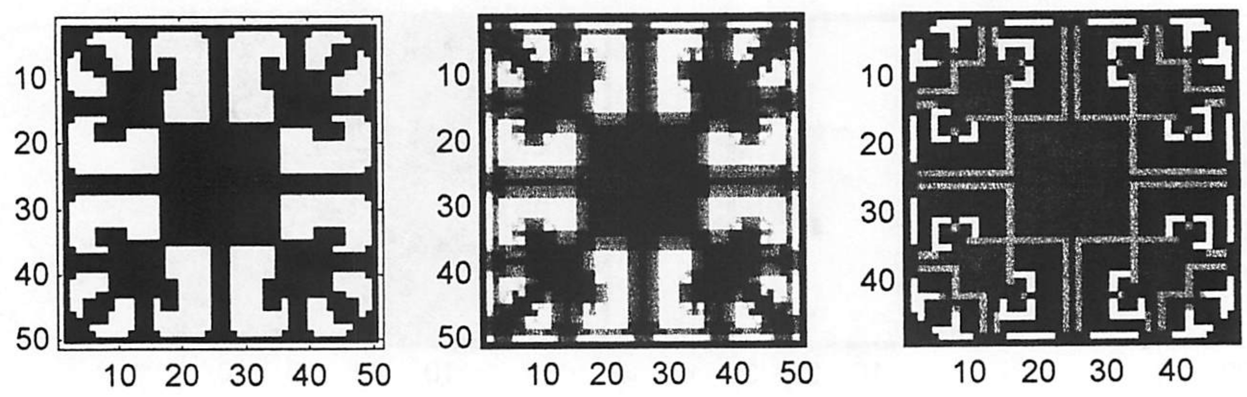


Fig. 11 Simulation results for "best individual" strategy

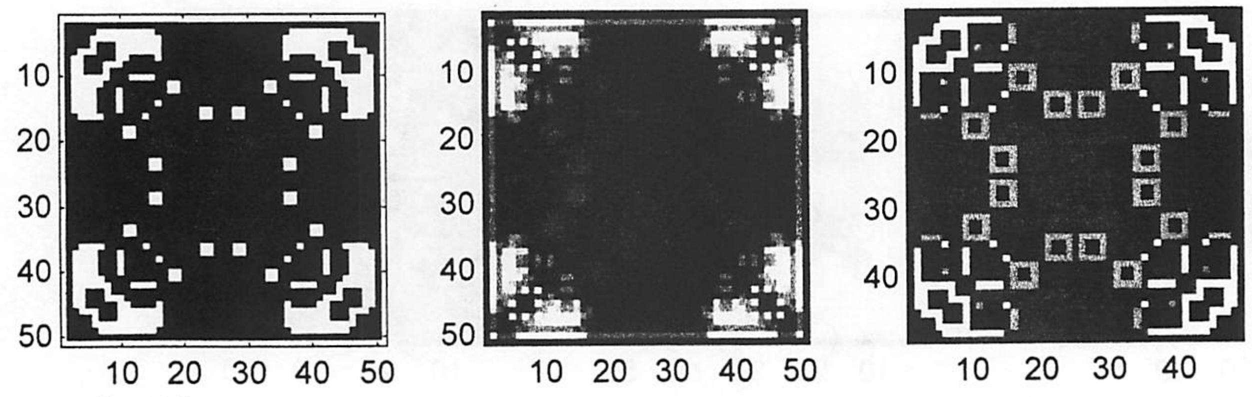
generation: 30



generation: 40



generation: 50



generation: 70

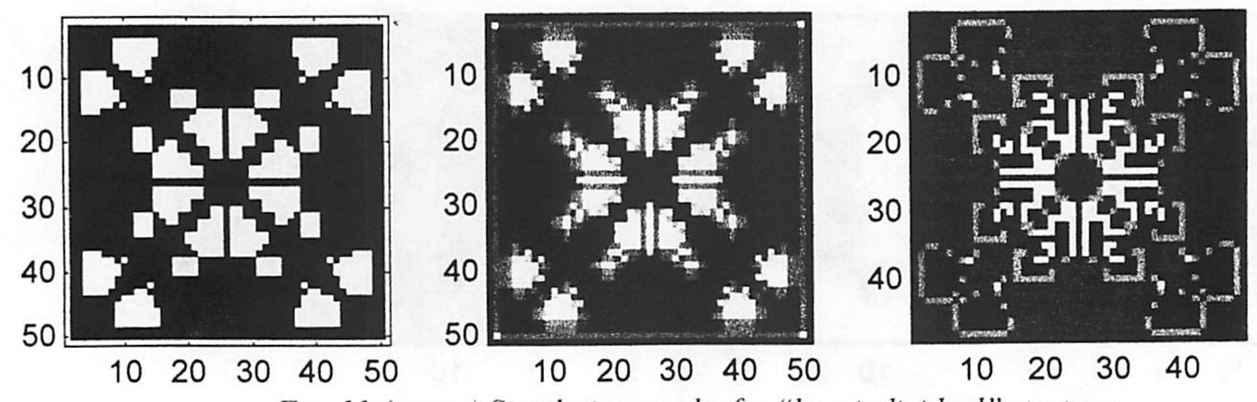


Fig. 11 (cont.) Simulation results for "best individual" strategy

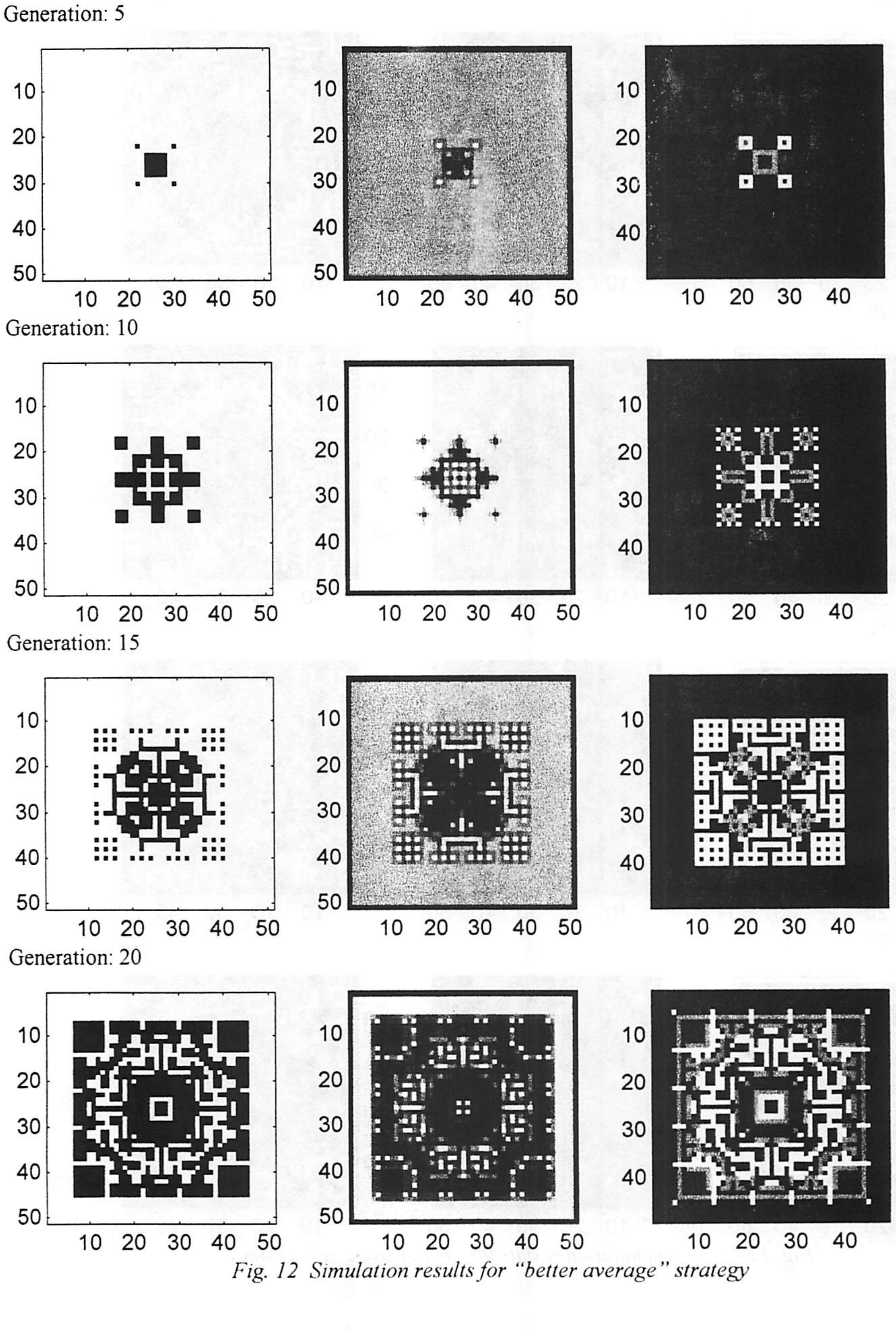
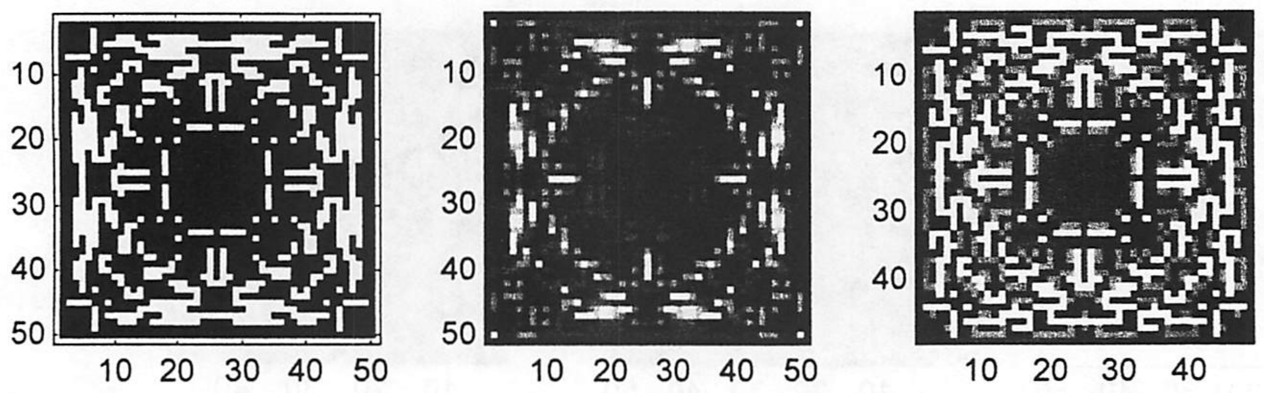
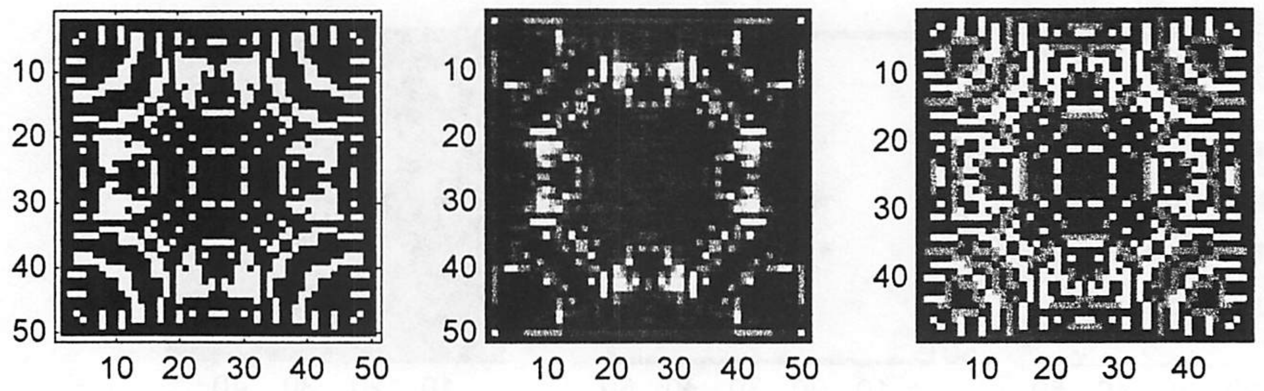


Fig. 12 Simulation results for "better average" strategy

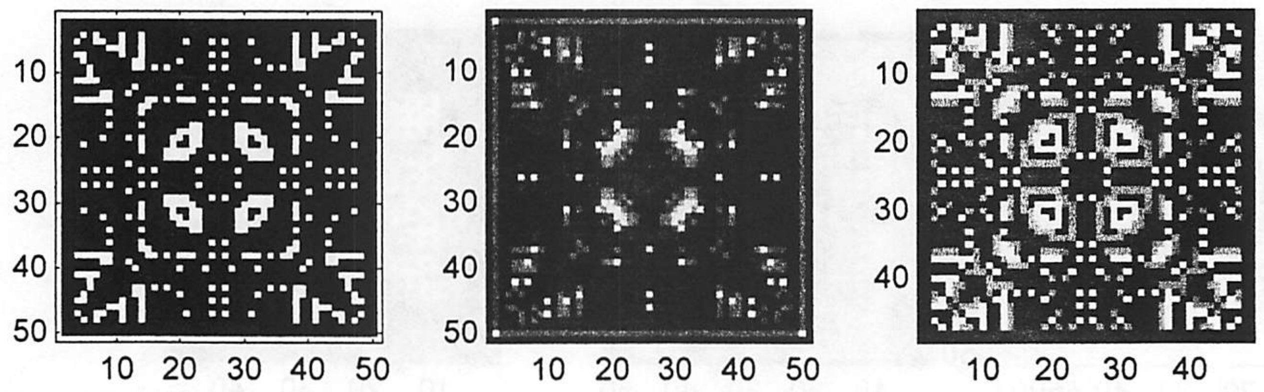
Generation: 20



Generation: 40



Generation: 50



Generation: 70

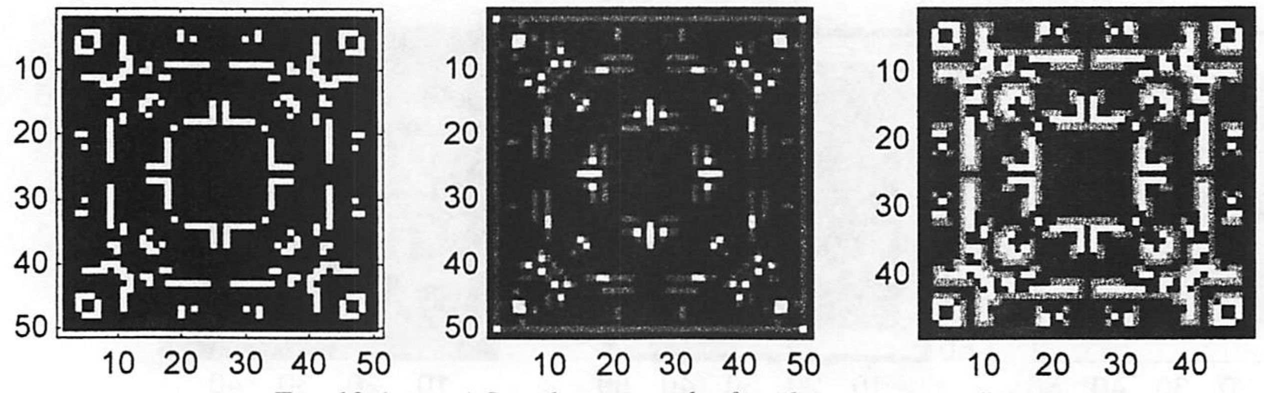


Fig. 12 (cont.) Simulation results for "better average" strategy

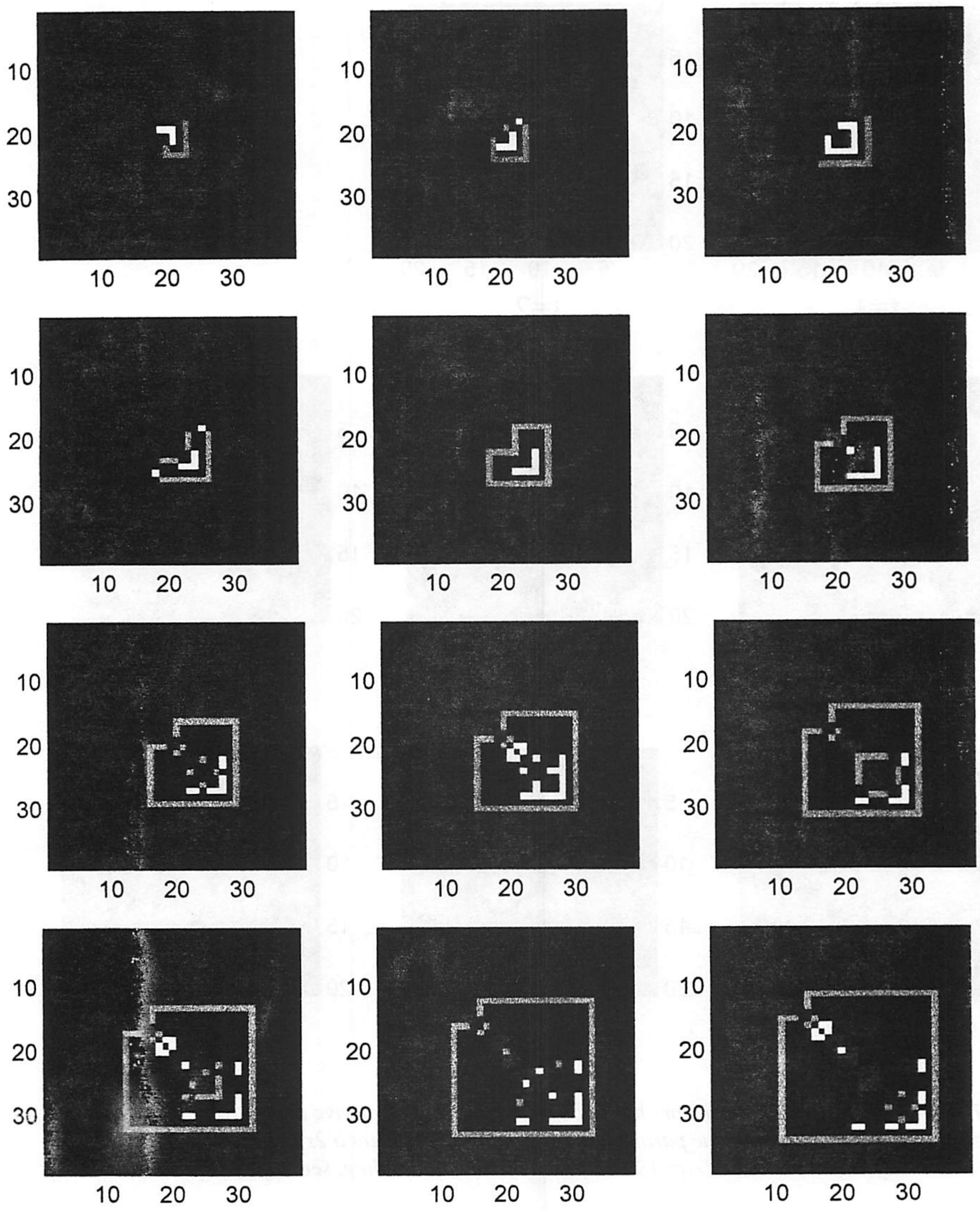


Fig. 13 A cluster of cooperators invades a field of defectors.(dynamic field color coding. see Appendix).
Array size: 39x39, Cluster size: 11, P_{DC}*= 1.79, Time horizon : 2-14*

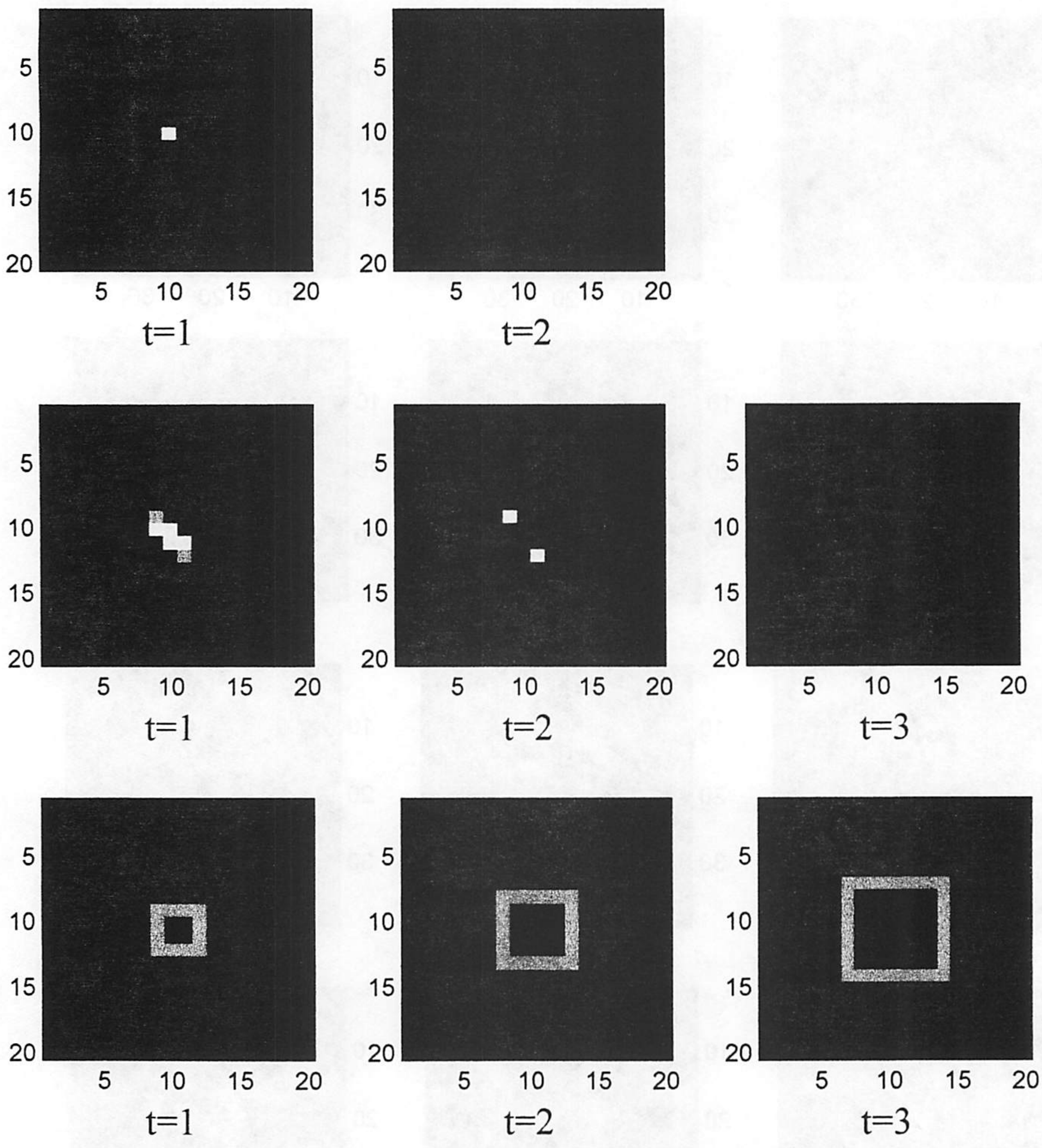
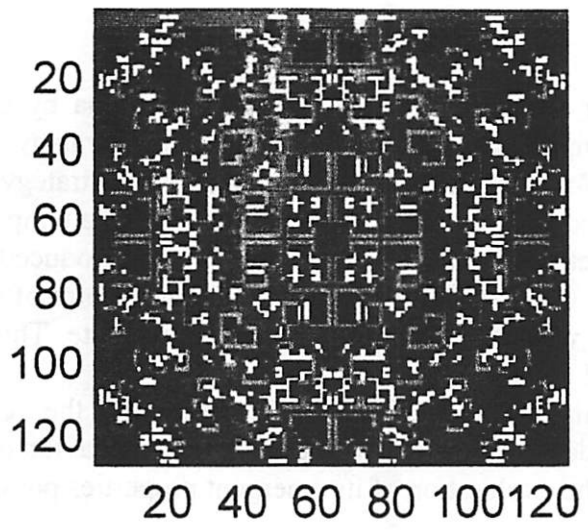
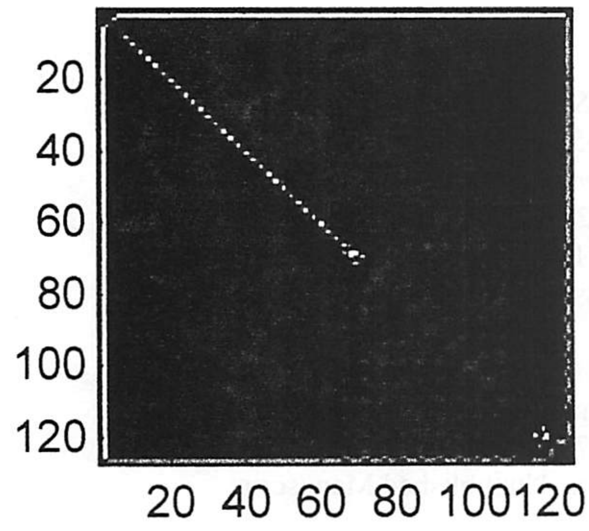


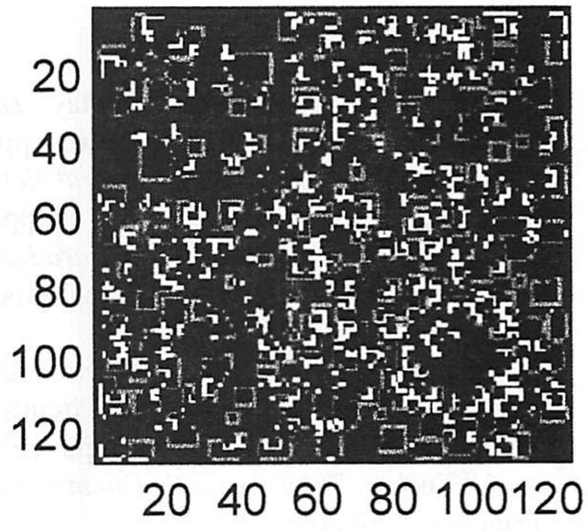
Fig. 14 The cluster of cooperators has to be sufficiently big to survive and grow in the hostile environment of defectors. For the parameters used in this experiment a 2x2 cluster of cooperators is virulent enough. P_{CD}= *1.79 (dynamic field color coding. see Appendix).*



(a) Single defector invades a field of cooperators $P_{DC}= 1.85$



(b)



(c)

Fig. 15

(b) A cluster of cooperators invades a field of defectors $P_{DC}= 1.79$
(c) A cluster of cooperators invades a field of defectors $P_{DC}= 1.85$

In experiments (b) and (c) the same cluster was used. Cluster size: 11
All images: image size: 127x127, number of iterations: 100
(dynamic field color coding. see Appendix).

5. Conclusion

Two different solutions for simulating spatial prisoners dilemma by CNN have been proposed. The CNN implementation of the "best individual" strategy yields the same results as it was published in [1]. In order to implement this strategy on a CNN we translated the spatial local extremum problem into a more tractable temporal problem.

A new strategy called "better average" strategy has been introduced for its greater conformity with the regular CNN structure. From the practical point of view it provides an at least equally efficient way of selecting the player's next state. This strategy yields similar, though not identical results.

Beside the theoretical importance of these experiments the CNN's superior computational speed provides new insight to the long term behavior of the prisoner's dilemma game and makes the exploration of its emergent properties possible.

References

[1] Martin A. Nowak and Robert M. May: *The Spatial Dilemmas of Evolution* , I. J. of Bifurcation and Chaos, Vol. 3 No. 1 pp. 35-78, February 1993

[2] L. O. Chua, L. Yang: *Cellular Neural Networks: Theory* , IEEE Transactions on Circuits and Systems, Vol. 35, No. 10, pp. 1257-1290, October 1988

[3] L. O. Chua , T. Roska: *The CNN Paradigm*, IEEE Transactions on Circuits and Systems-I: Analog and Digital Signal Processing, Vol. 40, pp. 147-156, March 1993

[4] T. Roska, L. O. Chua: *The CNN Universal Machine: An Analogic Array Computer*, IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing, Vol. 40, No. 3, pp. 163-173, March 1993

[5] A. W. Tucker: *The Prisoner's Dilemma Game*, Unpublished Manuscript, Princeton University

[6] Tamas Kis : *Predictable Properties of the Spatial Prisoner's Dilemma Game*, Research Report 1995, Computer and Automation Institute of the Hungarian Academy of Sciences

Appendix

Legend of color coding of simulation results:

Static field:	cooperator	□
	defectors	■
Payoff field:	Hot colormap	MIN ▬ MAX
Dynamic field:	cooperator --> cooperator	▨
	defector--> defector	■
	cooperator --> defector	□
	defector--> cooperator	▩