Far East J. Math. Sci. 1(2) (1993), 139–155

CELLULAR NEURAL NETWORKS, GENETIC ALGORITHMS AND OBJECT EXTRACTION

SANKAR K. PAL, DINABANDHU BHANDARI, P. HARISH and MALAY K. KUNDU

(Received October 4, 1993)

Submitted by P. B. Gibbons

Abstract

A cellular neural network (CNN) is an information processing system with a large scale nonlinear analog circuit. Setting up a CNN for a particular task needs a proper selection of circuit parameters (cloning template) which determines the dynamics of the network. The present paper provides a methodology, demonstrating the capability of Genetic Algorithms, for automatic selection of cloning templates when CNN is used in extracting object regions from noisy images. This relieves the CNN from using heuristics for template selection procedure and performs consistently well in noisy environments for both synthetic and real images.

1. Introduction

A cellular neural network (CNN) is an unsupervised neural network proposed by Chua and Yang [2,3]. It is made up of a massive aggregate of analog circuit components called cells. Any cell in a cellular neural network is connected only to its neighboring cells and interacts only with them. Key features of this network are asynchronous processing, continuous time dynamics and local interaction between the cells (processing elements). Cells not directly connected together affect each other due to propagation effects. The continuous time feature of a CNN imparts it real time signal processing capability while the local interconnections make it amenable to VLSI implementation. Its application in image processing problems, viz., edge

1991 Mathematics Subject Classification : 68U10, 68T05
Key words and phrases : cellular neural network, genetic algorithm, automatic selection, cloning templates, image processing, object region extraction.

detection, noise removal, horizontal and vertical line detection has been reported in [3,5].

Setting up a CNN needs a proper selection of circuit parameters of cells. The set of circuit parameters is called a *cloning template* which determines the dynamics of the network and its application domain. There is no standard method for selecting a cloning template automatically for a CNN, although some heuristics have been suggested [2, 3] for this task.

In this article an attempt is made to develop a methodology for demonstrating an application of a CNN to another image processing operation, namely, image segmentation and object extraction where the required cloning templates are automatically selected using genetic algorithms (GAs) [4]. Some guidelines for selecting heuristically the circuit parameters for object extraction problem have also been provided. The study has been conducted with both noisy synthetic and real images using different cloning templates. The results obtained using the heuristically selected cloning templates are compared with those obtained using the template selected by GA. It has been found that the performance of the later one is consistently better under different noisy conditions. The performance has also been quantitatively evaluated. Although the method developed for selection of cloning template, based on GAs, has been used for the object extraction problem, it can also be applied to other image processing operations.

2. Cellular Neural Network

An $M \times N$ cellular neural network can be considered as an ordered set $\{C_{ij}\}$ of MN cells which are arranged in a pattern shown in Figure 1. For a two dimensional CNN, the r–neighbourhood $\left(N_{ij}^r\right)$ can be defined as

$$N_{ij}^r = \left\{C_{kl} \mid \max(|k-i|, |l-j|) \le r,\ 1 \le k \le M; 1 \le l \le N\right\}.$$

The state υ_{xij} of any cell C_{ij} in a CNN is described by the differential equation :

$$C\frac{d\upsilon_{xij}(t)}{dt} = \frac{-1}{R_x}\upsilon_{xij}(t) + \sum_{C_{kl}\in N_{ij}^r} A_{ij;kl}\upsilon_{ykl} + I_{ij},\ 1 \le i \le M;\ 1 \le j \le N \quad (1)$$

where $A_{ij;kl}$ represents the conductance of the link between C_{ij} and C_{kl}, and I_{ij} is the bias to the cell C_{ij}. R_x and C (constants for a CNN) are the resistance and capacitance of the cell. $C.R_x$ is the time constant of the circuit and R_x determines the power dissipation.

The output of the cell C_{ij} is a nonlinear function shown in Figure 2. It can be

expressed as

$$v_{yij}(t) = 0.5\left(\left|v_{xij}(t) + 1\right| - \left|v_{xij}(t) - 1\right|\right), \quad 1 \le i \le M; 1 \le j \le N. \tag{2}$$

The state of each cell in CNN is bounded and after the transients settle down, a CNN approaches a stable state [2]. Moreover, if the circuit parameters satisfy

$$A_{ij;ij} > \frac{1}{R_x}$$

then

$$\lim_{t\to\infty} \left|v_{xij}(t)\right| \ge 1, \quad 1 \le i \le M;\ 1 \le j \le N;$$

or equivalently

$$\lim_{t\to\infty} v_{yij}(t) = \pm 1, \quad 1 \le i \le M;\ 1 \le j \le N.$$

Before explaining the methodology for object extraction using a CNN along with automatic selection of the parameters $A_{ij;kl}$ a brief description of GAs is given in the next section.

3. Genetic Algorithms : Basic Principles and Features

Genetic algorithms [4] are highly parallel adaptive search and machine learning processes based on the mechanics of natural selection in natural genetic system. They exploit structured information to solve a wide range of complex optimization problems using genetic operators (reproduction/selection, crossover and mutation) on coded solutions (strings/chromosomes) in an iterative fashion. GAs deal simultaneously with multiple points (comprising the population), not a single point, which helps to find the global optimal solution without getting stuck at local optima. Genetic algorithms are blind, that is, they use only the payoff or penalty (i.e., objective) function and do not need any other auxiliary information. To solve an optimization problem, genetic algorithms start with the chromosomal (structural) representation of a parameter set. The parameter set is to be coded as a finite length string over an alphabet of finite length. Usually, the chromosomes are strings of 0's and 1's. For example, let $\{a_1, a_2, \ldots, a_p\}$ be a realization of the parameter set and the binary representation of $a_1, a_2, \ldots, a_p$ be 10110,00100, ..., 11001 respectively. Then the string 10110 00100 ... 11001 is a chromosomal representation of the parameter set $\{a_1, a_2, \ldots, a_p\}$. Strings are then processed using three simple genetic operations.

Reproduction/selection is a process in which individual strings are copied according to the values of their objective function, f (called the fitness function), into a mating pool. Therefore, highly fit strings have a higher number of offsprings in the succeeding generation.

The *crossover* operation may proceed in two steps. First, pairs of the reproduced strings in the mating pool are selected for mating at random. Second, each selected pair of strings undergoes crossing over as follows : an integer position k is selected uniformly at random between 1 and $l - 1$, where l is the string length greater than 1. Two new strings are created by swapping all characters from position $k + 1$ to l. Let

$$a = 11000\ 10101\ 01000 \ldots 01111\ 10001$$
$$b = 10001\ 01110\ 11101 \ldots 00110\ 10100$$

be two strings (parents) selected for the crossover operation and the generated random number be 11. Then the newly produced offsprings (swapping all characters after position 11) will be

$$a' = 11000\ 10101\ 01101 \ldots 00110\ 10100$$
$$b' = 10001\ 01110\ 11000 \ldots 01111\ 10001.$$

In genetic algorithms, *mutation* is the occasional (with small probability) random alteration of the value of a string position. It helps to prevent the irrecoverable loss of potentially important genetic material. A random bit position of a random string is selected and is replaced by another character from the alphabet. For example, let the third bit of string a, given above, be selected (randomly) for mutation. Then the transformed string after mutation will be

$$a'' = 11100\ 10101\ 01000 \ldots 01111\ 10001$$

The effectiveness of this efficient searching technique has been demonstrated in the next section for automatic extraction of objects from images using CNN.

4. CNN for Object Extraction

Object extraction involves segmenting the whole image into two region types, namely object region and background region. Two adjacent pixels of an image belong to the same region if they have similar gray value properties. So, both gray level and positional properties play an important role in making a decision as to whether a pixel belongs to the object or to the background.

In order to explain the principle of object extraction using CNN, let us consider equation (1) which can be approximated by a difference equation as :

$$\upsilon_{xij}(n+1) = \upsilon_{xij}(n) + \frac{h}{C}\left[\frac{-1}{R_x}\upsilon_{xij}(n) + \sum_{C_{kl}\in N_{ij}^r} A_{ij;kl} f\left(\upsilon_{xkl}\right) + I_{ij}\right] \tag{3}$$

$$1 \le i \le M;\ 1 \le j \le N$$

where h is a constant time step and $\upsilon_{xij}(n)$ is the state of the cell C_{ij} at the nth instant of time.

A CNN processes signals by mapping from one signal space to another one. If the initial state space is $[-1.0, 1.0]^{M\times N}$ and the output space is $\{-1, 1\}^{M\times N}$, then the dynamic map F can be defined as

$$F : [-1.0, 1.0]^{M\times N} \rightarrow \{-1, 1\}^{M\times N}. \tag{4}$$

Object extraction in an image $X = \{x_{mn} : m = 1, 2, \ldots, M; n = 1, 2, \ldots, N\}$ of size $M \times N$ can be considered as a mapping

$$E : [a, b]^{M\times N} \rightarrow \{A, B\}^{M\times N} \tag{5}$$

where the range of gray levels of the image is $[a, b]$, and A, B are gray levels of object and background respectively or vice versa. Equations (4) and (5) are analogous. Thus if we can transform the gray level of the input image into $[-1.0, 1.0]$ it is possible to achieve a transformation of the image into $\{-1, 1\}$ by using a cellular neural network.

Equation (3) can be rewritten, for an unit time step ($h = 1$), as

$$\upsilon_{xij}(n+1) = \upsilon_{xij}(n) + \frac{1}{C}\left[\frac{-1}{R_x}\upsilon_{xij}(t) + \sum_{C_{kl}\in N_{ij}^r} A_{ij;kl} f\left(\upsilon_{xkl}(t)\right) + I_{ij}\right], \tag{6}$$

$$1 \le i \le M;\ 1 \le j \le N.$$

or

$$\upsilon_{xij}(n+1) = \upsilon_{xij}(n) - \frac{1}{R_x C}\upsilon_{xij}(t) + \frac{1}{C}\left[T * \upsilon_{yij}(t) + I_{ij}\right] \tag{7}$$

where

$$T = \begin{bmatrix} T_{-r,-r} & \cdots & T_{-r,0} & \cdots & T_{-r,r} \\ T_{0,-r} & \cdots & T_{0,0} & \cdots & T_{0,r} \\ T_{r,-r} & \cdots & T_{r,0} & \cdots & T_{r,r} \end{bmatrix}$$

is the cloning template and * is a convolution operator. For any cloning template T, the convolution operator * is defined as

$$T * v_{ij} = \sum_{C_{kl} \in N_{ij}^r} T_{k-i,l-j} v_{kl} \tag{8}$$

In the above definition $A_{ij;\,kl}$ is assumed to be position invariant.

4.1 Selection of Cloning Template

The selection of cloning template T plays an important role in defining the dynamic rule of a CNN for performing a particular operation. The guidelines for selection of T for some operations like noise removal, edge detection and line detection are mentioned in [3]. For example, in noise removal of an image, T can be expressed as an averaging operator. For edge detection, T is expressed by a Laplacian or a difference operator so that the pixels of homogeneous and boundary regions can possess values –1 and +1 respectively at the stable state of the net. The way one can select T for object extraction problem is described below.

As mentioned earlier, in object extraction both gray level and positional properties should be taken into account and one way of achieving this operation is to use an weighted average operator. Therefore one can choose the weighted averaging operator in defining the dynamic rule of a CNN in this regard. This operator can be expressed by a cloning template like C_2 (2–connected), C_3 (3–connected), C_4 (4–connected) etc. as shown below :

$$C_2 = \begin{bmatrix} 0.0 & 1.0 & 0.0 \\ 1.0 & 2.0 & 1.0 \\ 0.0 & 1.0 & 0.0 \end{bmatrix} \tag{9}$$

$$C_3 = \begin{bmatrix} 0.5 & 1.0 & 0.5 \\ 1.0 & 2.0 & 1.0 \\ 0.5 & 1.0 & 0.5 \end{bmatrix} \tag{10}$$

$$C_4 = \begin{bmatrix} 0.0 & 0.0 & 0.5 & 0.0 & 0.0 \\ 0.0 & 1.0 & 2.0 & 1.0 & 0.0 \\ 0.5 & 2.0 & 4.0 & 2.0 & 0.5 \\ 0.0 & 1.0 & 2.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.5 & 0.0 & 0.0 \end{bmatrix} \tag{11}$$

Let us consider C_2, as an example, for explaining the object extraction operation in terms of dynamic equation. Here for cell time constant $C.R_x = 1$ microsecond and

cell bias $I_{ij}=0$ we have

$$\frac{d\upsilon_{xij}(t)}{dt}=10^6\left[-\upsilon_{xij}(t)+\upsilon_{yi-1j}(t)+\upsilon_{yij-1}(t)+2\upsilon_{yij}(t)+\upsilon_{yij+1}(t)+\upsilon_{yi+1j}(t)\right] \quad (12)$$

and

$$\upsilon_{yij}(t)=0.5\left(\left|\upsilon_{xij}(t)+1\right|-\left|\upsilon_{xij}(t)-1\right|\right) \quad (13)$$
$$1\le i\le 8, 1\le j\le 8$$

Let the initial state of the CNN be the same as the pixel values as shown in Figure 3(a). Then we see that for the pixel at position, (4,4) the derivative $\frac{d\upsilon_{xij}(t)}{dt}$ is positive, whereas the derivative at the point (2,2) is negative. As a result, the pixel values at (4,4) and (2,2) at the next iteration will tend to increase and decrease respectively. After several iterations when the network becomes stable, these values will tend to +1 and –1, denoting object and background, respectively. The dynamic rule (equation (12)) is therefore able to detect the homogeneous object and background regions in the image by making their values +1 or –1 (Figure 3 (b)). It is only the corner pixels of the object which may possess some intermediate value.

–1.0	0.4	–0.8	–1.0	–0.6	–0.7	0.2	–0.8
–0.3	–0.7	–0.2	0.3	–0.7	–0.7	–1.0	–0.7
–0.1	–0.9	0.8	0.8	0.5	0.6	–0.9	–0.3
0.2	–1.0	1.0	0.8	0.6	0.3	–0.2	0.4
–0.3	0.1	0.8	0.1	0.7	0.4	0.2	–0.4
–0.7	–1.0	0.2	1.0	0.9	1.0	–0.2	–0.9
–0.5	0.2	–1.0	–0.6	–0.8	–0.7	0.2	–1.0
–0.2	0.1	–0.5	–1.0	–0.7	–1.0	–0.2	–0.6

Fig. 3(a)

–1.0	–1.0	–1.0	–1.0	–1.0	–1.0	–1.0	–1.0
–1.0	–1.0	–1.0	–1.0	–1.0	–1.0	–1.0	–1.0
–1.0	–1.0	0.7	1.0	1.0	–0.5	–1.0	–1.0
–1.0	–1.0	1.0	1.0	1.0	1.0	–1.0	–1.0
–1.0	–1.0	1.0	1.0	1.0	1.0	–1.0	–1.0
–1.0	–1.0	0.0	1.0	1.0	0.5	–1.0	–1.0
–1.0	–1.0	–1.0	–1.0	–1.0	–1.0	–1.0	–1.0
–1.0	–1.0	–1.0	–1.0	–1.0	–1.0	–1.0	–1.0

Fig. 3(b)

It is therefore clear from the aforesaid discussion that different cloning templates are required for different image processing operations. Again, for a particular operation, the same cloning template may not be applicable for all kinds of images. For example, C_2 (equation (9)) may not be applicable for segmenting thin or noisy image regions.

There is no standard method for seleting automatically a correct set of cloning template parameters in setting up the dynamics of a CNN for a particular operation on a given image. In the next section, we develop a methodology using genetic algorithms to provide a solution to this problem. Here, the problem of choosing correct parameters (for a r–neighbourhood cloning template) of a CNN for object extraction has been considered to be equivalent to searching for an appropriate set of parametrs $\{T_{i,j} : -r \le i \le r; -r \le j \le r\}$ from a complex space which gives the best object background classification. (Note that the same framework may be used for other image processing operations).

4.2 Optimum selection of Cloning Template using genetic Algorithms

Selection of fitness function of a GA for selecting the optimum parameters of a CNN is an important task. Here, the parameters of cloning templates are selected using a training (labeled) pair consisting of an input image and the desired segmented (target) image. For directing the search in GAs we have used a fitness function that measures the discrepancy between the obtained and the target images. Considering the obtained and the target images as fuzzy sets representing some imprecise property, we have used divergence [1] between them as the fitness function. Then we try to minimize the divergence to direct the search.

Let X_1 and X_2 be the obtained and target images respectively. Let $A = \{x_{mn} / \mu^A_{mn}\}$ and $B = \{x_{mn} / \mu^B_{mn}\}$ be the fuzzy sets characterizing the property that "(m, n)th pixel belongs to the object/background region" corresponding to X_1 and X_2. Then the divergence $D(A, B)$ between A and B is defined as

$$D(A, B) = \frac{1}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} \left(\mu^A_{mn} - \mu^B_{mn}\right) \ln \frac{\mu^A_{mn}}{\mu^B_{mn}} + \left(\mu^B_{mn} - \mu^A_{mn}\right) \ln \frac{1 - \mu^A_{mn}}{1 - \mu^B_{mn}} . \quad (14)$$

$D(A,B)$ is minimum if two sets are similar, i.e., $B = A$ and is maximum if B is the farthest non fuzzy set of A. Although, we have used divergence measure as a fitness function, one may use any other similarity measure e.g., mean square error, for this purpose.

As stated earlier, for an r neighbourhood cellular neural network the total number

of circuit parameters to be selected in a cloning template is $(2r + 1)^2$. The method of selecting these parameters is written in algorithmic form as follows.

1. Generate $p(= (2r + 1)^2)$ random strings of 0's and 1's each of length 'l' corresponding to the parameters $x_1, x_2,\ldots,x_p$. Each of these strings represent a parameter of the cloning template in a coded form. 'p' strings are concatenated to form a chromosomal/string representation of the parameter set. Generate 'm' such chromosomes to form the initial pool.
2. Input a noisy or gray version (X_1) of a known binary image (X_T).
3. Decode the strings representing the parameters of the cloning template and use these to set up a cellular neural network. The gray level range of the input image (X_1) is scaled to [–1, 1].
4. Compute the fitness function value corresponding to the output image (X').

Table 1 : Percentage of correct classification using C_2, C_3 and C_4

σ	C_2	C_3	C_4
10	99.71	99.74	99.80
15	98.93	99.31	99.46
20	96.16	98.26	98.61
24	92.58	96.04	97.38
32	84.46	89.62	91.88

5. If specified number of iterations is reached then STOP.
6. The strings are reproduced or selected to create a new mating pool as described in section 3.
7. Generate a new population by crossover and mutation.
8. Goto step 3.

Though the object extraction is an unsupervised classification problem, we have adopted here a supervised approach (in the sense of known X_T) for automatic selection of the set of optimal cloning template parameters of a CNN. Once the template parameters are selected by this GA based algorithm the unknown images can be processed for segmentation.

5. Computer Simulation and Results

The methodologies developed in Section 4 is tested on a set of 128 × 128 synthetic images (Figures 4 (a–d)) consisting of geometric objects corrupted by Gaussian noise of different standard deviatioins (σ = 10, 20 and 32) and zero mean, and a real 64 × 64 image of a biplane (Figure 5) on a VAX–8650 computer. The experiment has two parts. In the first part, the effectiveness of a CNN has been

demonstrated for 2–connected (C_2), 3–connected (C_3) and 4–connected (C_4) cloning templates (equations (9–11)) which were selected heuristically. The second part consists of an investigation where we have demonstrated the adaptivity of GAs in selecting automatically the circuit parameters (clonig template). Throughout the experiment, we assumed $C.R_x$ (cell time constant) = 1 micro second, R_x (resistance of a cell) = $10^3\Omega$ and I_{ij} (cell bias) = 0.

Figures 6(a–c), 7(a–c) and 8(a–c) depict the outputs corresponding to the images with various noise levels (σ = 10, 20 and 32) when C_2, C_3 and C_4 have respectively been used to define the dynamic rule of CNN. Table 1 shows the corresponding percentages of correct classification. The results, as expected, show that the performance of the algorithm deteriorates with σ and improves with the increase in size of the cloning template. The template C_4 is less immune to noise and provides better performance as compared to C_2 and C_3 due to the greater neighbourhood effect. This experiment was repeated on the biplane image also and it has been found that the classification accuracy increases with the size of the cloning template for thick and compact regions. Thin and elongated regions get destroyed by the cloning templates of larger size due to greater neighbourhood effects (Figures 9(a–c)). Note that the cost of realizing the circuit increases with the size of the neighbourhood. It may therefore be pragmatic to choose cloning templates depending upon the need.

Table 2 : Percentage of correct classification using C_3 and C_{GA} (templatc obtained by Genetic Algorithm)

σ	C_3	C_{GA}
t0	99.74	98.97
15	99.31	98.56
20	98.26	98.02
24	96.04	97.74
32	89.62	94.69

For demonstrating the capability of genetic algorithms to obtain optimum cloning template parameters for the above mentioned task, we considered the template C_3. Here, for the selection of template parameters we have used a noisy version (Figure 10(b)) of a 32 × 32 image of character '*B*' (Figure 10(a)). Figure 10(a) and 10(b) represent X_T and X_1 (Section 4.2) respectively for computing divergence (equation (14)) as fitness value between the target image X_T and an obtained image. The cloning template (C_{GA}) obtained after 50 generations is shown in (15) (assuming population size = 10 and mutation probability = 0.005).

$$C_{GA} = \begin{bmatrix} 0.37 & 0.97 & 0.51 \\ 0.94 & 0.95 & 0.95 \\ 0.54 & 0.92 & 0.47 \end{bmatrix} \quad (15)$$

This cloning template was then applied on the noisy images of geometric objects (Figures 4(a–d)). The percentages of correct classification are given in Table 2. From Table 2, and Figures 11(a–c) and 12 it is clear that the cloning template obtained using genetic algorithms performs uniformly well under different signal to noise ratios as compared to C_2, C_3 and C_4.

In a part of the experiment we have also investigated the effect of cell bias on network performance by considering (i) $I_{ij} = 0$ (zero bias) and (ii) $I_{ij} = \upsilon_{xij}$ (proportional bias)

Table 3 : Effect of bias on classification accuracy (in percentage)

σ	C_2		C_3	
	Proportional bias	Zero bias	Proportional bias	Zero bias
10	98.45	99.71	99.76	99.74
15	87.51	98.93	89.46	99.31
20	85.15	96.16	87.30	98.26
24	74.83	92.58	78.37	96.04
32	69.34	84.46	70.20	89.62

For C_3 cloning template. Table 3 gives the percentage of correct classification achieved at various σ levels. The results indicate the deterioration of performance at higher noise levels in case of proportional bias, which is not so significant in case of zero bias.

6. Conclusion

The problem of automatic selection of cloning template of cellular neural networks which can perform parallel signal processing in real time has been considered. A solution has been provided by developing a methodology based on Genetic Algorithms. Its success has been demonstrated in extracting objects from images for various sizes of cloning templates. As expected, a template of larger size is less noise sensitive but affects thin and elongated regions due to greater neighborhood effect. The results show that the template selected by GA performs consistently well and less immune to noise as compared to the heuristically selected templates. Though we have used here the fuzzy divergence measure as the fitness function, one can

choose any other discrimination measure to direct the search.

Although we have adopted here a partially supervised technique for object extraction, one can develop a unsupervised approach by defining an appropriate fitness function which will quantify the segmented image. The framework, based on genetic algorithms, developed here in selecting cloning template for object extraction may also be applicable for some other image processing operations.

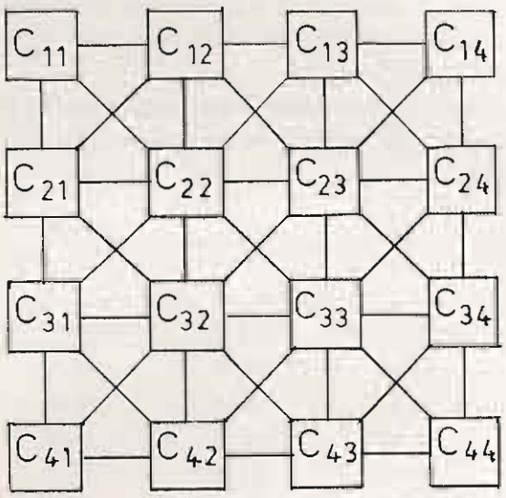

Figure 1. A two dimensional cellular neural network.

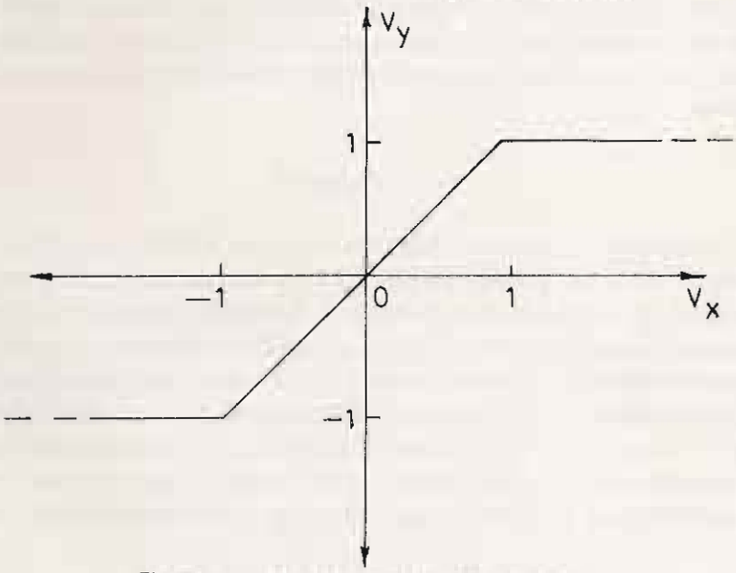

Figure 2. The nonlinear input output function.

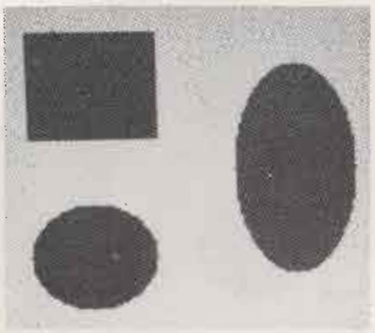

Figure 4(a). Original image of geometric objects (128 × 128).

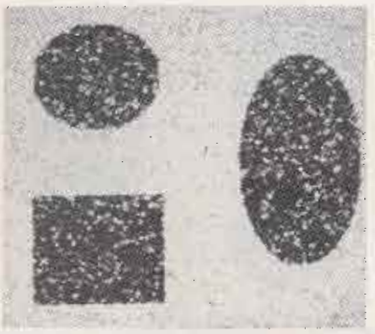

Figure 4(b). Image of geometric objects with Gaussian noise ($\sigma = 10$).

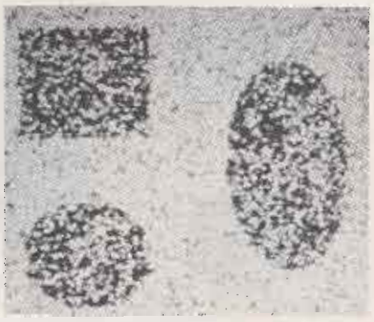

Figure 4(c). Image of geometric objects with Gaussian noise ($\sigma = 20$).

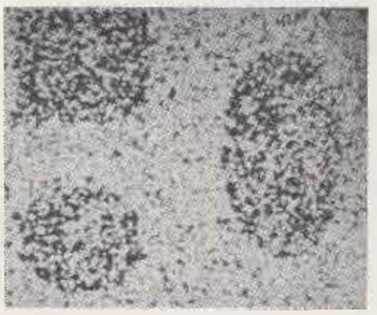

Figure 4(d). Image of geometric objects with Gaussian noise ($\sigma = 32$).

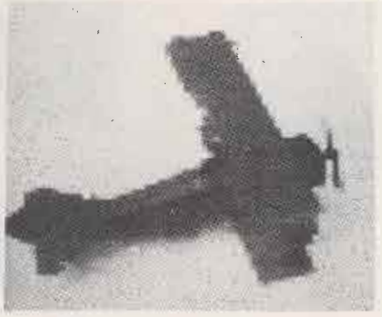

Figure 5. Original image of biplane (64 *X* 64).

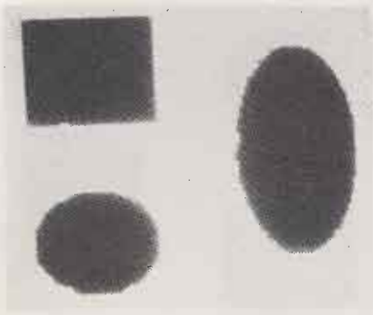

Figure 6(a). Extracted objects from noisy image ($\sigma = 10$) using C_2.

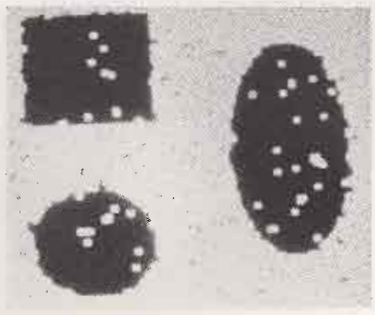

Figure 6(b). Extracted objects from noisy image ($\sigma = 20$) using C_2.

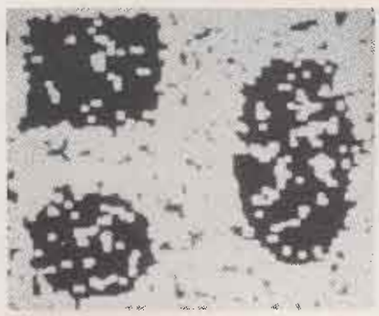

Figure 6(c). Extracted objects from noisy image ($\sigma = 32$) using C_2.

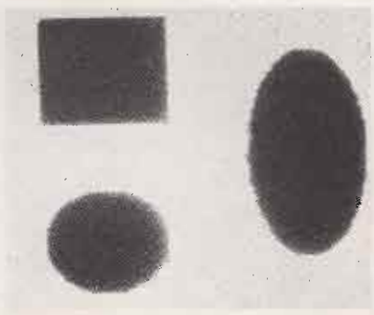

Figure 7(a). Extracted objects from noisy image ($\sigma = 10$) using C_3.

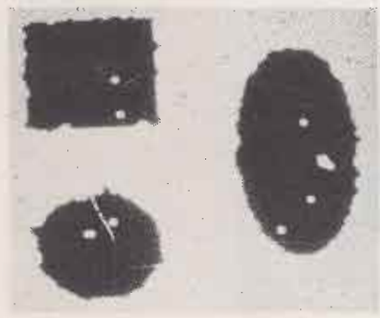

Figure 7(b). Extracted objects from noisy image ($\sigma = 20$) using C_3.

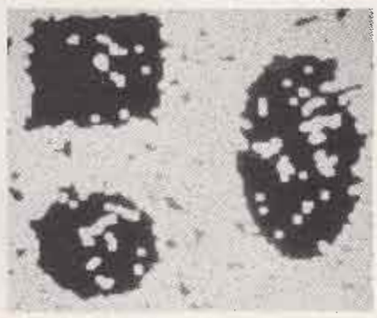

Figure 7(c). Extracted objects from noisy image ($\sigma = 32$) using C_3.

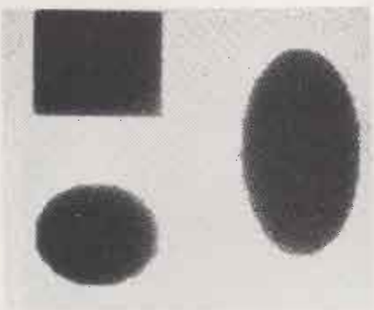

Figure 8(a). Extracted objects from noisy image ($\sigma = 10$) using C_4.

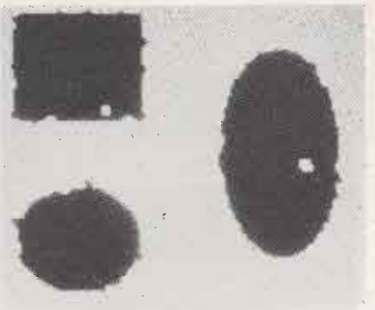

Figure 8(b). Extracted objects from noisy image ($\sigma = 20$) using C_4.

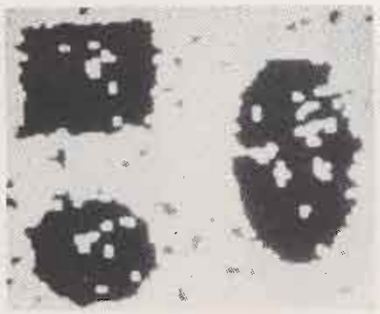

Figure 8(c). Extracted objects from noisy image ($\sigma = 32$) using C_4.

Figure 9(a). Extracted object from biplane image using C_2.

Figure 9(b). Extracted object from biplane image using C_3.

Figure 9(c). Extracted object from biplane image using C_4.

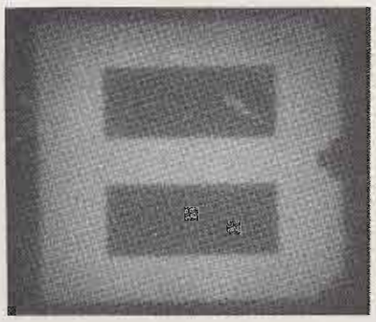

Figure 10(a). Original image of character '*B*' (32 × 32).

Figure 10(b). Noisy '*B*' (σ = 2) used for obtaining C_{GA}.

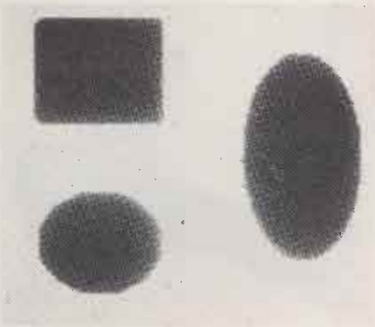

Figure 11(a). Extracted objects from noisy image (σ = 10) using C_{GA}.

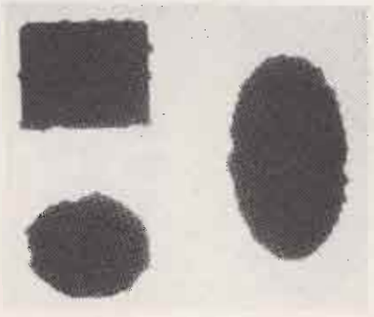

Figure 11(b). Extracted objects from noisy image (σ = 20) using C_{GA}.

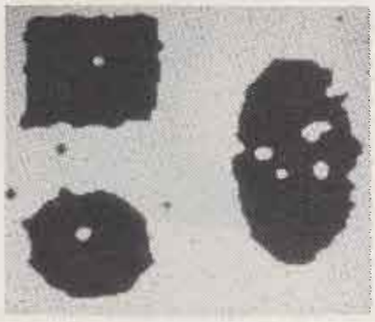

Figure 11(c). Extracted objects from noisy image (σ = 32) using C_{GA}.

Figure 12. Extracted objects from biplane image using C_{GA}.

References

[1] D. Bhandari and N. R. Pal, Some new information measures for fuzzy sets, Information Sciences 67 (1993), 304–323.

[2] L. O. Chua and L. Yang, Cellular neural networks : theory, IEEE trans on Circuits and Systems 35 (1988), 1257–1272.

[3] L. O. Chua and L. Yang, Cellular neural networks : applications, IEEE trans on Circuits and Systems 35 (1988), 1273–1290.

[4] D. E. Goldberg, Genetic Algorithms : Search, Optimizatioin and Machine Learning, Addison Wesley Publishing Company, Inc., 1989.

[5] L. Vandenberghe, S. Tau, and J. Vandewalle, Cellular neural networks : dynamic properties and adaptive learning algorithm, in Lecture Notes in Computer Science, No. 412 : Neural Networks, (L. B. Almeida and C. J. Wellekens, eds.), 141–150, Springer Verlag, 1990.

Machine Intelligence Unit
Indian Statistical Institute
203 B. T. Road
Calcutta - 700 035
India