Contractor Report ARFSD-CR-91007

ARTIFICIAL NEURAL SYSTEM FOR OBJECT CLASSIFICATION AND ORIENTATION ESTIMATION

Jeffrey D. Johnson
Timothy A. Grogan
4208 Forsythia Dr.
Cincinnati, OH 45245

Frank P. Kuhl
Project Engineer
ARDEC

April 1991

U.S. ARMY ARMAMENT RESEARCH, DEVELOPMENT AND ENGINEERING CENTER

Fire Support Armaments Center

Picatinny Arsenal, New Jersey

The views, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

The citation in this report of the names of commercial firms of commercially available products or systems does not constitute official endorsement by or approval of the U.S. Government.

Destroy this report when no longer needed by any method that will prevent disclosure of contents or reconstruction of the document. Do not return to the originator.

REPORT DOCUMENTATION PAGE

Form Approved
OMB N0. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	April 1991	8 Jun 89 to 28 Feb 90

4. TITLE AND SUBTITLE
ARTIFICIAL NEURAL SYSTEM FOR OBJECT CLASSIFICATION AND ORIENTATION ESTIMATION

5. FUNDING NUMBERS

6. AUTHOR(S)
Timothy A. Grogan and Jeffrey D. Johnson
Frank Kuhl, ARDEC Project Engineer

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Timothy A Grogan
4208 Forsythia Dr.
Cincinnati, OH 45245

ARDEC, FSAC
Fire Control Division (SMCAR-FSF-RC)
Picatinny Arsenal, NJ 07806-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
ARDEC, IMD
STINFO Br
ATTN: SMCAR-IMI-I
Picatinny Arsenal, NJ 07806-5000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
Contractor Report ARFSD-CR-91007

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

The usefulness of self-organizing neural systems for the problems of object recognition and orientation estimation are discussed. Self-organizing neural systems, like unsupervised cluster algorithms from classical pattern recognition, are most useful when no predetermined labels are available to attach to input patterns which must be categorized by the system. However, this makes it necessary to assign a "natural" category to the input stimuli from the environment. Only with some type of supervisory feedback will this natural category be associated with the proper label for the input pattern. Without this teaching input, the internal, self-organizing principles of the system must be used to assign categories. These assigned categories may or may not coincide with the unique labels of an external supervisory system.

14. SUBJECT TERMS				15. NUMBER OF PAGES
Neural networks	Self-organizing	ART	Neocognition	61
shape recognition				16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAR

Contents

List of Figures

List of Tables

1 Introduction

The intention of this study was to investigate the usefulness of self-organizing neural systems to the problems of object recognition and orientation estimation. Self-organizing neural systems like unsupervised cluster algorithms from classical pattern recognition are most useful when no predetermined labels are available to attach to input patterns which must be categorized by the system. This is an important requirement for natural systems which must assign labels to environmental stimuli that will possibly later take on some importance. However, this makes it necessary to assign a "natural" category to the input stimuli from the environment. Only with some type of supervisory feedback will this natural category be associated with the "proper" label for the input pattern. Without this teaching input, the internal, self-organizing principles of the system must be used to assign categories. These assigned categories may or may not coincide with the unique labels of an external supervisory system. If such a supervisory system is available, then this information should be utilized to improve the performance of the system *as seen by the supervisor* during training. This is because the performance criteria or categories of the supervisory system may or may not coincide with those corresponding to the self-organizing principles of the unsupervised learning system. Therefore, if supervisory inputs are available, a supervised learning system will in general produce more appropriately labeled patterns.

The experimental results described in this report indicate that for the low resolution aircraft silhouettes, it is difficult to obtain self-organized "natural" categories that coincide with the known object labels supplied by a supervisory system. Results are also provided that illustrate some of the difficulties with the ART 1 and Neocognitron neural systems maintaining consistent categorization with translation, rotation, and noise.

The report begins by providing an introduction to artificial neural networks in section 2. In section 3, a special class of artificial neural networks, those utilizing unsupervised learning, are discussed. This type of learning is contrasted with two types of supervised learning, learning with a teacher and learning with a critic. A classical self-organizing system for feature categorization, the K-nearest means algorithm, is described as a point of reference in section 4. A self-organizing neural network using competitive learning, Kohonen's self-organizing topological feature maps is described in section 5. Networks of this type are useful for feature extraction or vector-quantization.

Another competitive learning self-organizing system with many interesting properties, the adaptive resonance theory (ART) model, is presented in section 6. Some experiments that illustrate the properties of this network are also presented. The general self-organizing learning principle of Hebbian learning is discussed in section 7. This leads to the principle network chosen for close scrutiny in this study. The Neocognitron model for visual pattern categorization is described in section 8. Also in section 8, aircraft classification experimental procedures and results are provided for the Neocognitron. The computational complexity of the Neocognitron model is also presented in section 8. Concluding remarks are given in the final section 9.

2 Artificial Neural Networks

Investigations into neural network models of human perception have taken place since the 1940's [15,29]. Artificial neural networks (ANNs) are also known as connectionist models, parallel distributed processing, or neuromorphic systems. However, the success of recent discoveries using new architectures and training procedures have caused renewed interest in this approach. New technology on the horizon providing massively parallel implementations (e.g. optical computers) of these theories make them even more attractive.

The elementary processing element for the artificial neural network consists of a simple processing node having numerous inputs which are weighted (according the connections strength) and summed. After subtraction of a

von Neumann Computer	Artificial Neural System
Few, complex PEs	Many, simple PEs
Limited interconnections	Massive interconnection
Inherently fault intolerant	Inherently fault tolerant
Programmed	Learn by experience
PEs fast (10 nsec)	PEs slow (10 msec)
Excellent symbolic processing	Excellent sensory processing

Table 1: Comparison between von Neumann computers and artificial neural systems.

threshold, a non-linear function is performed to produce an output, i.e.

$$y = f(\sum_{i=0}^{N-1} w_i x_i - \theta),$$

where $x_i \equiv$ inputs from the previous layer or the stimulus input, $w_i \equiv$ the connections weights, $\theta \equiv$ threshold, and

$$f(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

Sigmoidal non-linearities are also used. Different models vary in the connection pattern and the computation of the connections weights.

The Hopfield model [16,17,18] is a single-layer recursive neural network with symmetric connection weights. These connection weights are specified a priori by the problem to be solved. This network has been used to find (near) optimal solutions to some n-p complete problems, in particular, the traveling salesman problem. This network can also be used as a pattern classifier or content-addressable memory. The Hopfield network has problems including convergence to "spurious" outputs corresponding to a misclassification of the input pattern. Also, a large number of nodes (N) are required to recognize M classes, experience has shown that it is necessary for $M < 0.15N$.

The three-layer perceptron is a feedforward network with two "hidden" layers of neurons between the stimulus input and the final output layer. This multi-layered perceptron overcomes many of the limitations of the first-order perceptrons that were throughly studied by Minsky and Papert [31]. Recently developed algorithms have been successful in solving a number of interesting problems [33]. The most successful is the recently reported error back-propagation training algorithm (BPN) [33,35]. The network consists of three or more layers. Each input is connected to every node of the first hidden layer. The outputs of the first hidden layer are connected to every node in the second hidden layer. Similarly, the outputs of the second hidden layer are connected to every node of the final output layer. To begin training the network, all the connection weights are initialized to small random values. Then, for each training input pattern, the input feature is feedforward from the input units through the hidden layer units and through the output units. Each input training pattern is paired with a desired output pattern. The back-propagation training algorithm is a gradient search technique in the

space of the connection weights that seeks to minimize the mean-square error between the actual output of the network and the desired output. Beginning at the final output layer, the error between the output and the desired output is propagated back towards the input. At each layer, the weights are adapted to reduce the error. The weights for both layers are adapted after the error is feedback.

A three-layer perceptron when used as a pattern classifier can, in theory, form arbitrarily complex boundaries in the input feature space. A theorem proved by Kolmogorov and described in [28] provides an existence theorem for this kind of neural network. It states that any continuous mapping of n variables to m output variables can be implemented exactly with a three-layer neural network having n neurons on the first layer, $2n+1$ in the middle layer, and m in the final layer. However, this existence proof gives no clue as to how the weights are to be calculated. The back-propagation algorithms were among the first to dramatically improve the search time required for obtaining near optimal connection weights.

3 Unsupervised Learning

Most of the artificial neural networks are constructed to produce a desired output given a particular input. This is accomplished by repeatedly presenting pairs of inputs with its associated desired output. The difference or error between the networks actual output and the desired output is used to modify the weights in such a way that will (eventually) reduce this error. This type of learning is called *learning-with-a-teacher.* This is because the network must be supplied with the correct response for a given input by the *teacher.* Popular networks of this type are the various perceptron models, e.g. first & higher order perceptrons with the perceptron, least-mean-squares, or back-propagation learning rules. These learning algorithms essentially perform some variant of gradient descent in the weight space to attain a global minimum on the error surface.

Two other kinds of learning are also possible. *Unsupervised learning* is when the system produces it own output representation for each input it is presented. The universe of input signals are categorized according to principles that emerge as a consequence of the input processing rules of each individual neuron (processing node) and the inter-neuronal architecture (in-

terconnections with associated weights). The network uses principles of *self-organization* to assign subsets of inputs to the same output class. ANNs with this type of learning are similar to clustering algorithms in traditional pattern recognition. Here no teacher supplies what is the correct or desired output for the network. This is useful when either no labeled training data is available or when it is useful to determine what is the "natural" clustering of the input patterns.

Learning-with-a-critic [1] or *graded learning* is a type of learning which fits somewhere in between the previous two. In this case the network is supplied with inputs and produces outputs or responses. The *critic* supplies a signal to the system which simply grades the performance of the network. The desired output is not supplied to the network. This reward or punishment signal is used to improve the performance of the network. This type of learning is similar to the reinforcement learning experiments conducted on mammals. These models simulate the learning behavior exhibited in the famous experiments by Pavlov. For example, a dog learns the conditioned response of salivation by repeated sounding of a bell prior to the presentation of food. This type of learning is being used by the authors as a mechanism for producing the selection of visual attention.[19] These drive-reinforcement models [24] utilize the temporal difference of neuronal inputs and outputs to determine the strength and direction of changes of the neuronal input synaptic weights.

This report will focus on the usefulness of unsupervised learning in neural networks as it applies to the the recognition of simple shapes in imagery. In particular, networks with this type of learning are in general at a disadvantage in comparison to the supervised learning algorithms. Unsupervised learning or self-organizing systems are most useful when no pattern categories are available. The self-organizational principles in operation in the neural networks are used by the system to form its own categorization of the input patterns. If the user of the system already has a fixed idea as to the classes to which are to be mapped, the supervised networks will undoubtedly provide a better mapping function. This mapping function provides the best approximation in some sense from the input patterns to the user assigned label or output vector. As a point of reference, we will first describe a traditional method of unsupervised learning for pattern recognition, K-nearest means.

4 K-Nearest Means

The K-nearest means algorithm is a method of partitioning a set of input training vectors, $\{\underline{x}(n)\}$, into K clusters C_i. The procedure begins first with an initialization phase, where the K vectors are initialized to describe initial cluster centers, $\underline{y}_i(0)$, $1 \leq i \leq K$. Next, the set of input training vectors are classified into one of K sets. The nearest neighbor rule is used to assign each input vector to the closest cluster C_i:

$$\underline{x} \in C_i(t), \text{ iff } d[\underline{x}, \underline{y}_i(t)] \leq d[\underline{x}, \underline{y}_j(t)], \text{ all } j \neq i.$$

Next the cluster centers are updated to be the centroid of all the training vectors currently assigned to that cluster. The classification and cluster center updating is continued until either few training vectors change their to a particular cluster, or a maximum number of iterations has been performed. Variations of the algorithm allow for the merging or splitting of a cluster using a measure of the cluster dispersion or overlap. Once these cluster centers are formed, the intra-class sample mean vector and covariance matrix can be estimated to obtain a parametric classifier to classify new unknown input vectors.

Of significance here, is to note that the the input patterns are not labeled by a teacher to indicate the "proper" class assignment. Instead, the algorithm self-organizes a partitioning of the input pattern space by assigning an index indicating one of K classes, according to its own classification and cluster updating rules.

5 Self-Organizing Topological Feature Maps

An artificial neural network for performing categorization of multi-dimensional input vectors has been advanced by Tuevo Kohonen[20,21,22,23]. This network performs it task in a way very similar to the K-nearest means classifier. This network is organized as a single layer of either an linear or two-dimensional array of neurons. Each neuron receives each element of the input vector, $\underline{x}_i$. Associated with each input element is a weight, w_{ij}. The output or activation of the network is obtained as the weighted sum of the input

vector elements and the associated weight. For the i_{th} neuron

$$y_i = \sum_{j=1}^{n} w_{ij} x_j, \quad x_j \in \Re, \; j = 1, \ldots, n.$$

or

$$y_i = \underline{w}_i \cdot \underline{x}$$

The network uses a self-organizing principle to learn a mapping between the input vector and an output vector. This network learns a mapping or transformation between the input vector, $\underline{x} \in \Re^n$ and the output vector, $\underline{y} \in \Re^1 or \Re^2$. If the inputs are ordered with respect to a metric in the input vector space, $\Re^n$, then the outputs retain order relative to some metric in the output vector space, $\Re^1$ or $\Re^2$. In other words, the mapping is topology preserving. It is this property however that would make this network unsuitable for the recognition of the images of three-dimensional objects.

During the learning phase, each neuron also has local competitive interactions. This can be accomplished by connecting the output of each neuron to neurons in its local neighborhood through inhibitory weights. Each neuron also has an excitatory connection between its own output and input as well as those neurons within a smaller local neighborhood. If an input pattern persists, then this local competitive interaction causes a strongly responding neuron to suppress the activity of its neighbors while enhancing its own output (as well as those in a small neighborhood). Eventually a "bubble" of activity forms around the strongest responding neuron. The neuronal weights are then updated according to the differential equation

$$\frac{d\underline{w}_i}{dt} = \begin{cases} \alpha \underline{x} - \beta \underline{w}_i, & \text{inside bubble} \\ 0, & \text{outside bubble} \end{cases}$$

This leads to a discrete time simulation with discrete time variable t_k having both training and processing phase. The training phase has two steps. First a similarity matching is performed to determine the position, C, of the maximally responding neuron in the network.

Similarity Matching

$$\|\underline{x}(t_k) - \underline{w}_C(t_k)\| = \min_i \{\|\underline{x}(t_k) - \underline{w}_i(t_k)\|\}$$

Updating

The learning or updating of the weights then takes place according to whether the neuron is within the local neighborhood, N_C, as

$$\underline{w}_i(t_k + 1) = \underline{w}_i(t_k) + \alpha(t_k)[\underline{x}(t_k) - \underline{w}_i(t_k)], \quad \text{for } i \in N_C$$

or outside the local neighborhood as

$$\underline{w}_i(t_k + 1) = \underline{w}_i(t_k), \quad \text{for } i \ni N_C$$

The radius of the neighborhood shrinks linearly with each time. The learning constant, $\alpha(t_k)$, decreases linearly with time after an initial phase where $\alpha \approx 1$. The elements of the output vector, $\underline{y}_i$, are the outputs of each of the neurons in the network.

After the learning phase is complete, the input patterns, $\underline{x}$ are processed to provide an output vector, $\underline{y}$, whose elements are obtained from

$$y_i = \underline{w}_i \cdot \underline{x}.$$

5.1 Applications

Kohonen provides several examples of how the self-organizing topological feature maps (SOMs) can be used. The example is called "The Magic TV." A two- dimensional array of neurons is used. The simple sensing device consists of a circular photosensitive device divided into three equal area sectors. A crude optical focusing device images a spot of light as a large spot exciting one or more sectors of the sensing device. The electrical output from each sectors forms a three-dimensional input vector supplied to all the network neurons. After sufficient learning cycles have been performed, it is found that the position of the maximally responding neuron corresponds to the position of the point light source in the input plane.

A second example has a robotics application. In this case, a feeler mechanism is simulated. The mechanism consist of two jointed arms with the endpoint of each connected. Each arm as two joints whose movement is restricted in the plane. The relative angle of each joint is provided as input to a two-dimensional network. The network is trained by positioning the mutual endpoint of the feeler at random positions. After suitable training cycles, the position of the maximally responding neuron in the network corresponds

to relative spatial position of the feeler endpoint. Note that this transformation from joint angle to position is a very non-linear (albeit continuous) transformation.

A third example has been widely reported. This is the use of the SOM as a speech phoneme indicator. In this case the multi-dimensional input is related to the magnitude of the Fourier transform of a segment of speech. After suitable training, the network responds with the maximally responding neuron that is related to the phonetic content of the speech segment. Kohonen calls this a tonotopic map.

The self-organizing topological feature map of Kohonen is not ideally suited to pattern classification. Instead, this network is better used as a feature extraction or feature transformation processor. This is because of the topology preservation properties of the processing transformation. Categorization of input patterns often requires discontinuous transformations.

6 Adaptive Resonance Theory

The self-organizing neural network of Carpenter and Grossberg [2] is based on their Adaptive Resonance Theory (ART). ART 1 is used to categorize binary input patterns. ART 2 [3] has been developed for the recognition of analog inputs. These networks perform unsupervised clustering of sequential inputs. Learning of this kind is often described as competitive learning. As the first input is applied, the first cluster center is formed. As subsequent inputs patterns are applied, a distance measure is obtained. If the distance to the first cluster is greater than a threshold, a new cluster center (or critical feature pattern) is formed. If the distance is less than the threshold, the nearest matching cluster center is adapted. As more inputs are applied, they are either used to adapt an existing critical feature pattern or to form a new one. The matching scores between input and cluster centers (so called critical feature patterns) are computed using a feedforward network. The maximum matching score is selected using lateral inhibition among the output nodes. Feedback connections are provided to deselect the maximum output node and to compare the input to the critical feature pattern. The threshold or vigilance parameter determines how close an input must match an existing critical feature pattern. This regulates the number of critical features patterns learned by the system, i.e. whether the categories are fine or coarse.

6.1 Algorithmic Description

The dynamics of the network are complete described by a set of differential equations. An algorithmic description of the network is provided by Lippmann[27] and reproduced below:

Step 1: Initialization

$$\begin{gathered} t_{ij} = 1 \\ b_{ij} = \frac{1}{1+N} \\ 0 \leq i \leq N-1, \\ 0 \leq j \leq M-1 \\ \text{Set} \rho.\ 0 \leq \rho \leq 1, \end{gathered}$$

where $b_{ij}(t)$ and t_{ij} are the bottom-up and top-down connection weights or long term memory traces (LTMs), respectively. The ij^{th} weight corresponds to the i^{th} input element projecting to the j^{th} output node. The value ρ is the *vigilance* which indicates how close a match must be to be recognized or stored in LTM.

Step 2: Apply New Input

The input vector with element values x_k equal to ± 1 is applied to the network.

Step 3: Compute Matching Scores

$$\mu_j = \sum_{i=0}^{N-1} b_{ij}(t) x_i, \ \ 0 \leq j \leq M-1$$

Step 4: Select Best Matching Exemplar

$$\mu_j^* = \max_j \{\mu_j\}$$

Step 5: Vigilance Test

$$\text{If } \|\underline{t}_j \cdot \underline{x}\|/\|\underline{x}\| > \rho, \quad \text{then go to Step 7,} \quad \text{else go to Step 6.}$$

where

$$\|\underline{x}\| = \sum_{i=1}^{N-1} x_i \text{ and } \|\underline{t}_j \cdot \underline{x}\| = \sum_{i=0}^{N-1} t_{ij} x_i$$

Step 6: Disable Best Matching Exemplar

The output of the selected output node is set to zero and no longer is allowed to be considered in the selection of a maximum in step 4. Then go to Step 3.

Step 7: Adapt Best Matching Exemplar

$$t_{ij^*}(t+1) = t_{ij^*}\, x_i$$
$$b_{ij^*}(t+1) = \frac{t_{ij^*}(t)\, x_i}{.5+\sum_{i=0}^{N-1} t_{ij^*}(t) x_i}$$

Step 8: Repeat

Repeat by going back to Step 2, but first enable any nodes disabled in Step 6.

6.2 Properties

A simple implementation of this model was used to verify its fast learning capabilities. Overhead views of three aircraft (the same used in the investigation of the Neocognitron model) were used. The network quickly learned the three input patterns. The fast online learning is a distinct advantage of the ART networks. However, the ART1 and ART2 networks do not have any position or orientation invariance properties. This makes this kind of network unsuitable for shape recognition. However, with some fixed shape invariant transformation on the input image, it may be possible to perform invariant pattern recognition using this network. Some important advantages this network possesses are:

1. Real-time (or on-line) learning

   This network is designed to allow "recoding" or updating of the stored (average) templates in the long-term memory traces or weights. This is very much unlike the supervised networks such as the feed-forward networks with error back-propagation learning. For the BPNs, when a new pattern is added to the training set, the network must make many learning cycles over the *entire* training set. As for the ART network, if a pattern is sufficiently close to the stored pattern, the new information in the pattern is used to update the stored pattern. This is possible when the network is not required to assign input patterns to a category selected by some external user or teacher.

2. Effective use of memory capacity

3. Fast direct access to familiar patterns

6.3 Experiments & Results

A simple implementation of the ART 1 network was used to illustrate the capabilities and limitations of this approach. Overhead views of three aircraft; B57, F104, Phantom, were generated as silhouettes within a 16x16 image array. An ART 1 network with six category nodes or neurons was used. The three aircraft were input to the network having vigilance parameter, ρ, of 0.9. The output of the simulation is shown in Figures 1-3. With the vigilance at this high level the network quickly stores the three patterns individually. Applying the same three aircraft shapes causes no change to the stored patterns.

The same network after initialization was again presented with the three overhead views of the aircraft. However, in this case the vigilance parameter has been set to a lower value, $\rho = 0.75$. In this case, the third pattern (Phantom) causes recoding over the first pattern (B57), i.e. both patterns are assigned to the same category. This simulation results are shown in Figure 4-5.

Next, translated version of the same views of the aircraft are input to the network. With the same vigilance of 0.9, the translated shapes are considered

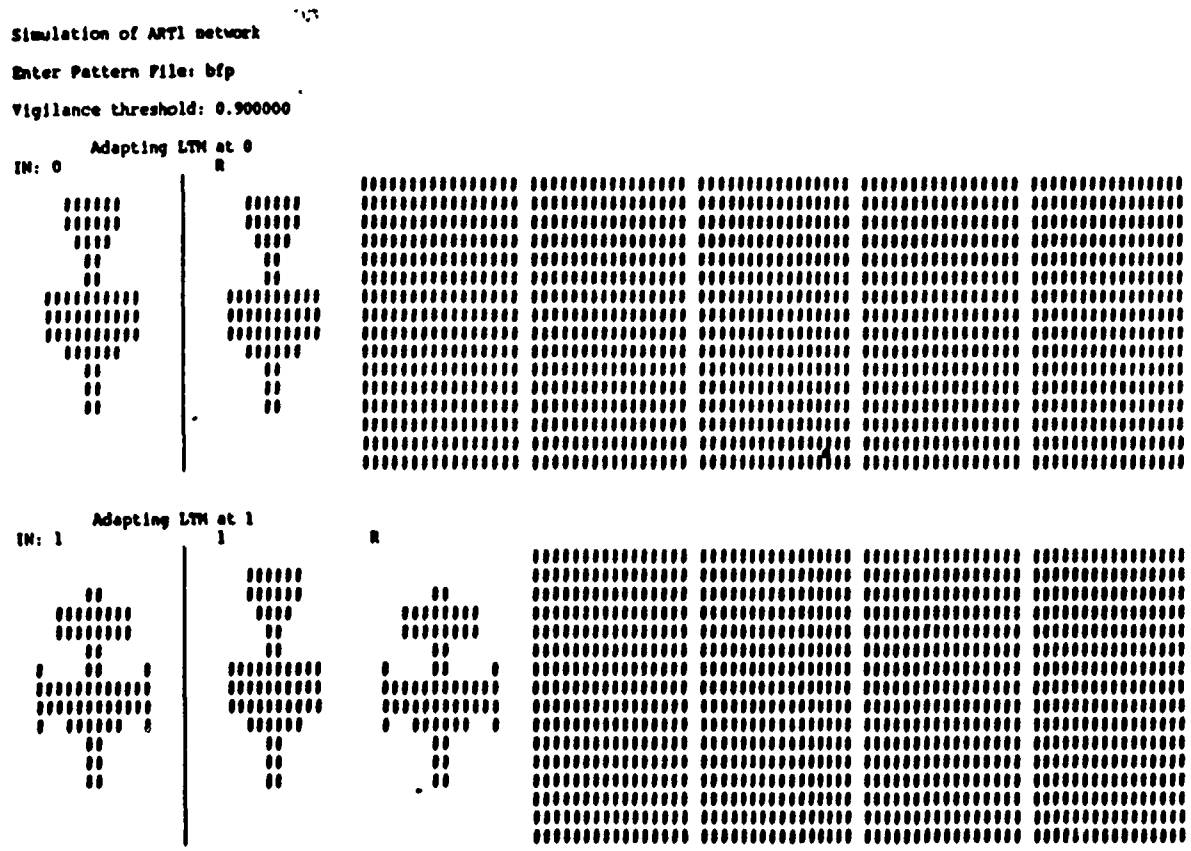

Figure 1: ART 1: Three aircraft shapes categorization quickly stabilizes with $\rho = 0.9$.

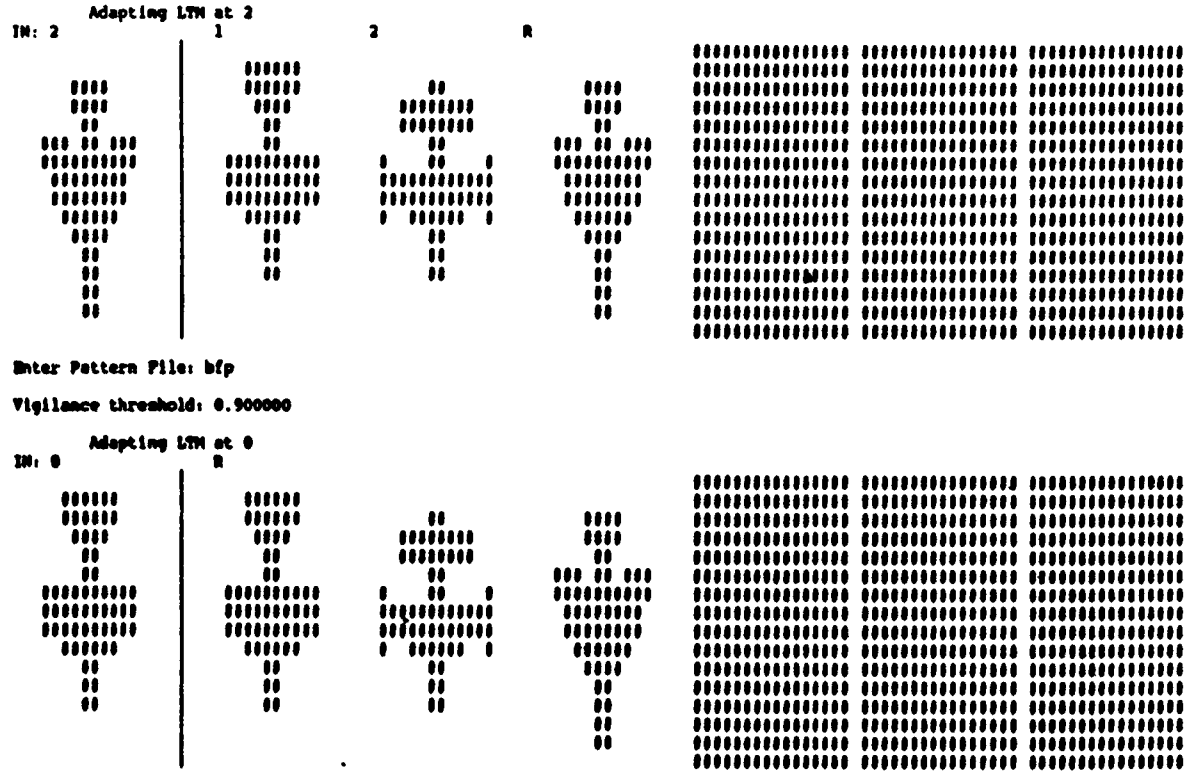

Figure 2: (Part b) ART 1: Three aircraft shapes categorization quickly stabilizes with $\rho = 0.9$.

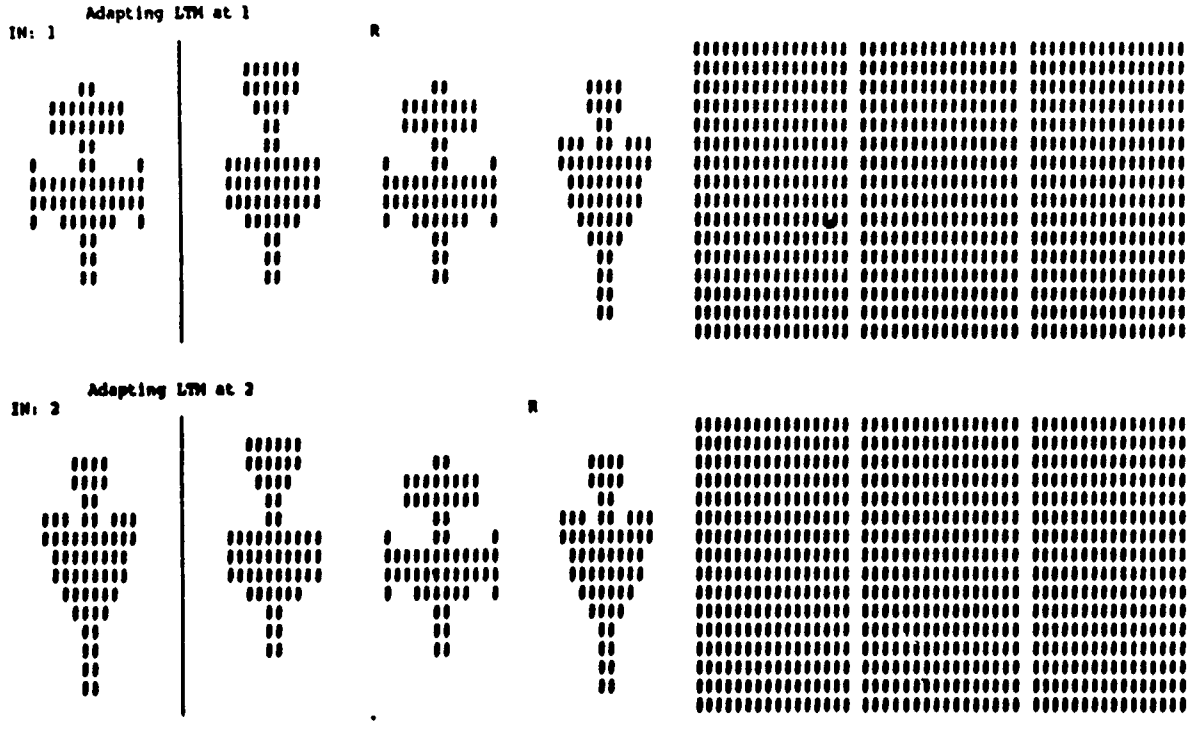

Figure 3: (Part c) ART 1: Three aircraft shapes categorization quickly stabilizes with $\rho = 0.9$.

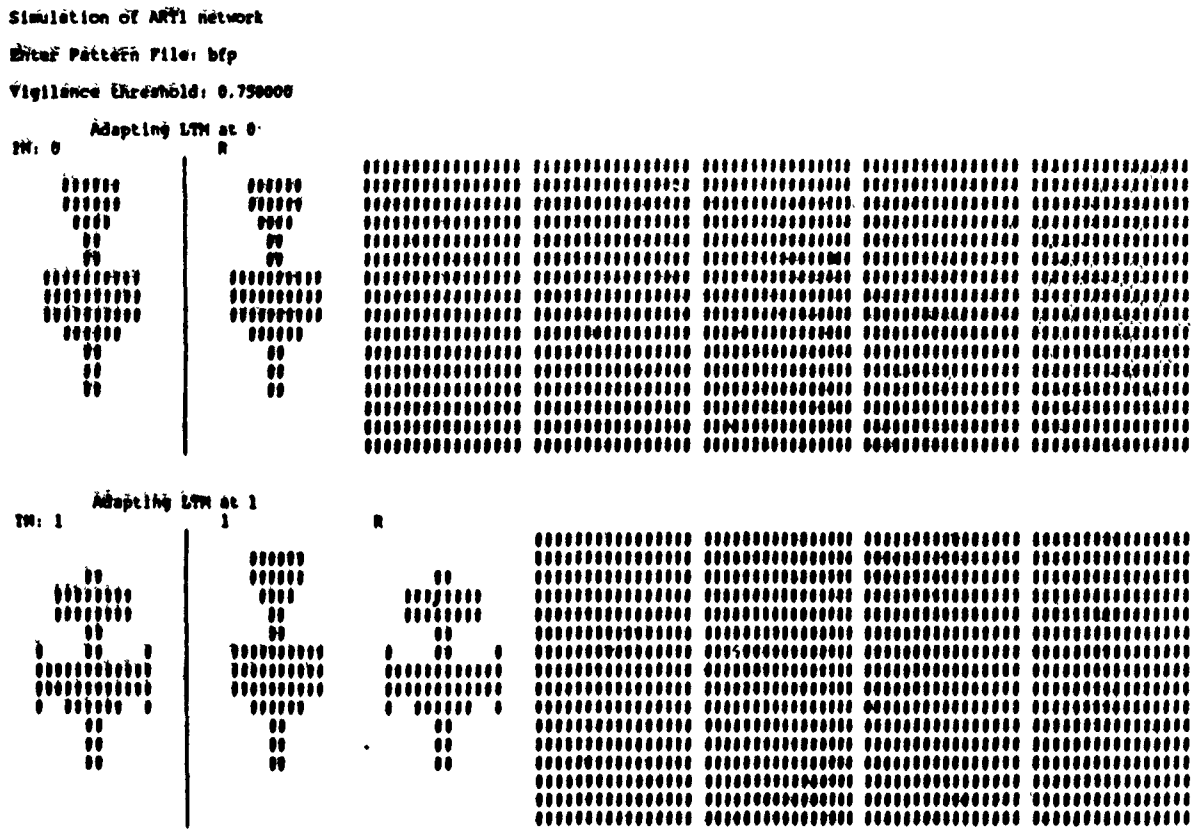

Figure 4: ART 1: Three aircraft shapes categorization as only two patterns with $\rho = 0.75$.

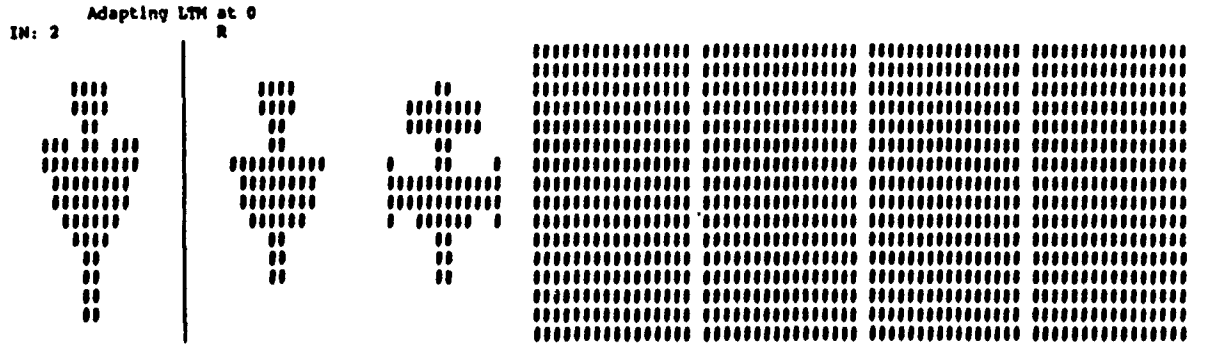

Figure 5: (Part b) ART 1: Three aircraft shapes categorization as only two patterns with $\rho = 0.75$.

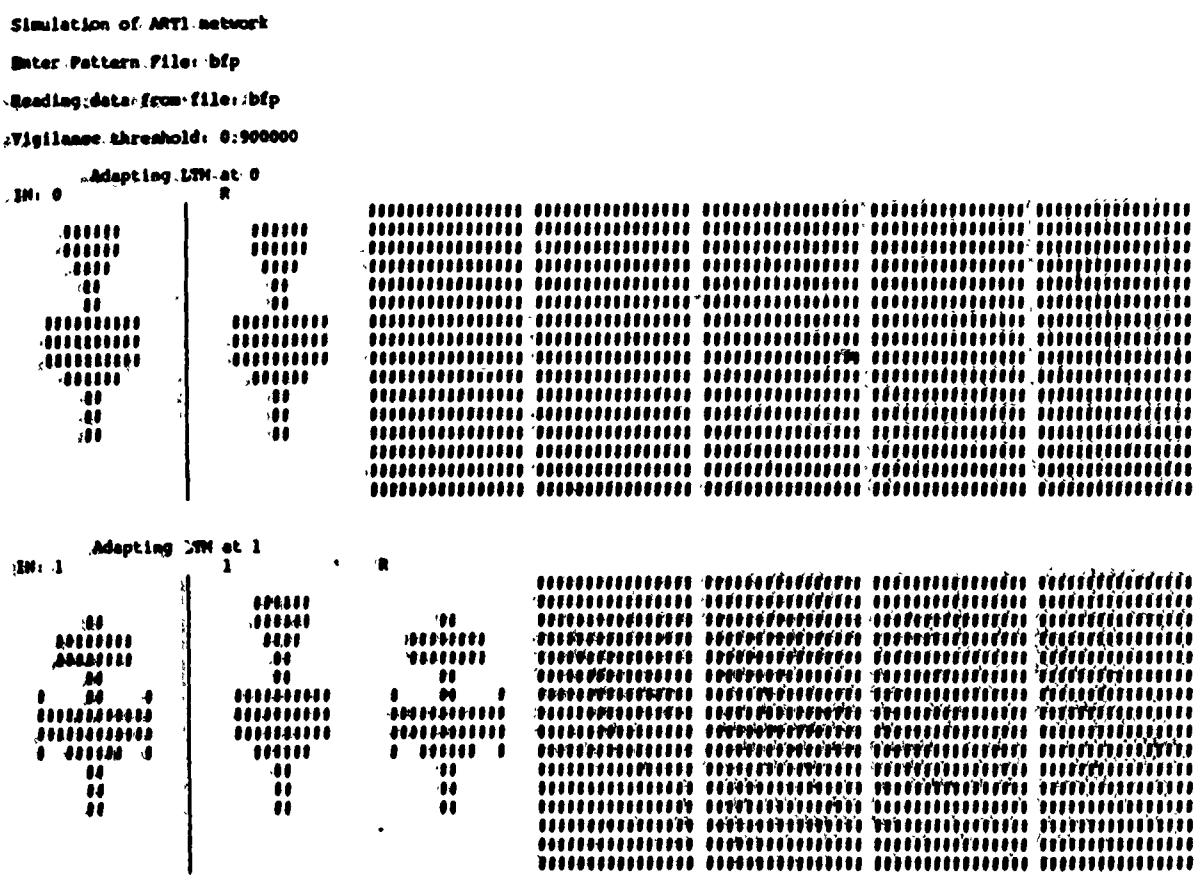

Figure 6: ART 1: Translated shapes are stored as new patterns with $rho = 0.9$.

to be different patterns are are stored separately in the network as shown in Figure 6-8.

The network is again presented with the same sequence of patterns, but in this case the vigilance parameter is much lower, $\rho = 0.75$. Instead, as might be hoped, the three translated shapes are assigned to different categories, while two of the non-translated shapes are assigned to the same category. The simulation results are shown in Figure 9-11.

The results of the ART 1 simulation in categorizing rotated views are depicted in Figures 12-13 and 14-16. The network organizes the three original view as well as the rotated views into separate when the vigilance parameter is at a high value, $\rho = 0.9$. When the vigilance parameter is reduced to a lower value of 0.75, the new rotated views of the Phantom are place in two separate, where as the original view has been placed in the same pattern category as the B57.

7 Self-Organization using Hebbian Learning

Donald O. Hebb in his book *The Organization of Behavior* [15] was the first to explicitly state an important principle in unsupervised learning in biological systems. He states

> When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased, (p.50)

In mathematical terms, the weight or efficacy of a connection between the input, x_j, and the output, y_i, increases in proportion to the joint occurrence (or correlation), i.e.

$$\Delta w_{ij} = \alpha y_i x_j.$$

So, learning takes place without a separate specific teaching input. Hebbian or modifications of Hebbian learning have been proposed by many investigators as the central principal for performing the self-organization in technical systems.

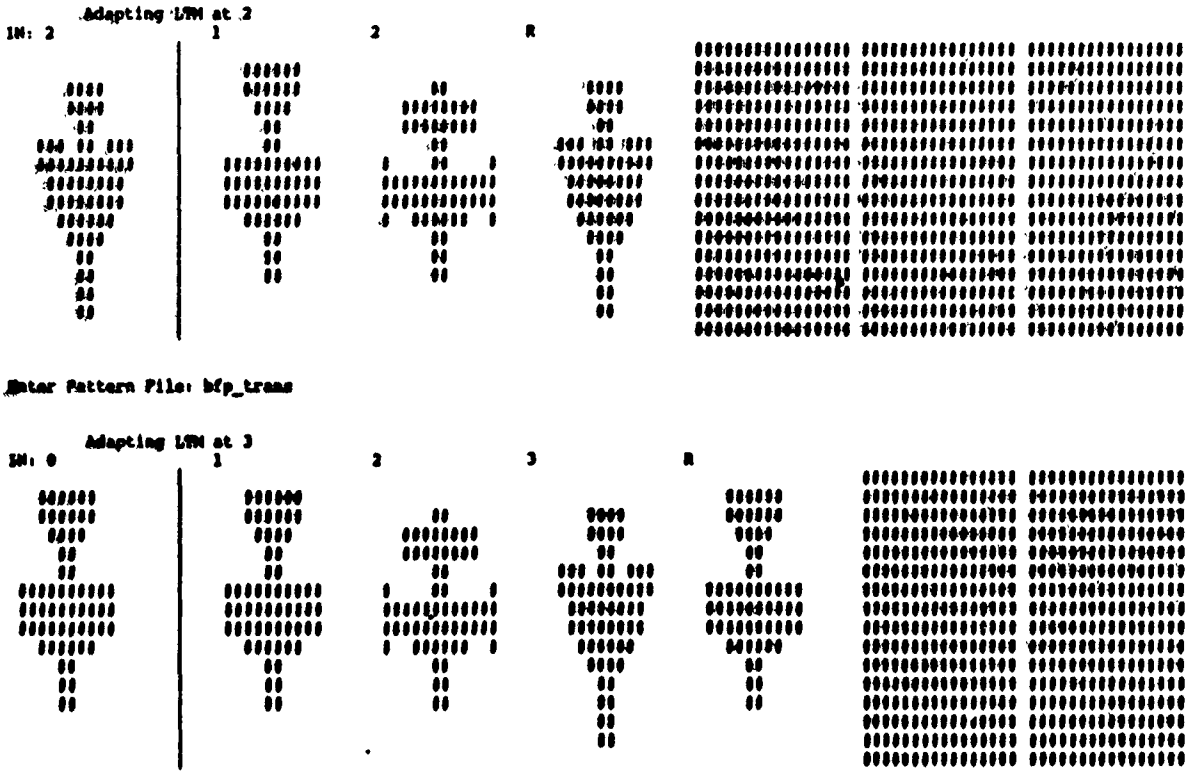

Figure 7: (Part b) ART 1: Translated shapes are stored as new patterns with $rho = 0.9$.

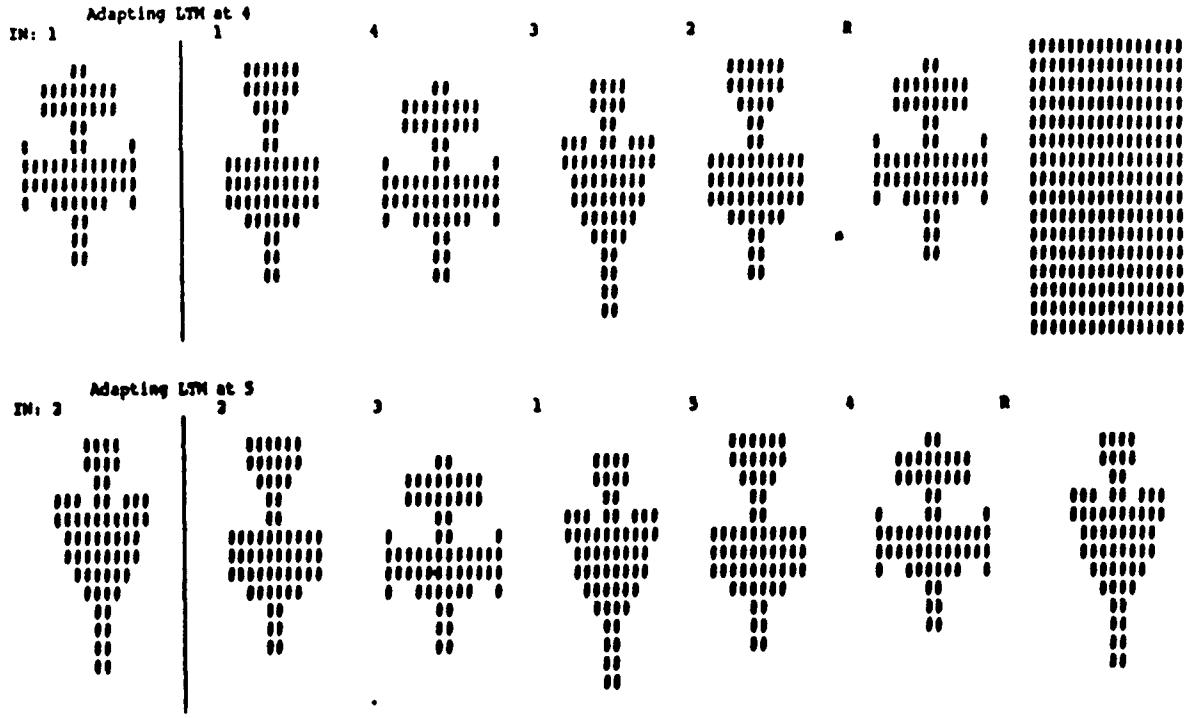

Figure 8: (Part c) ART 1: Translated shapes are stored as new patterns with $rho = 0.9$.

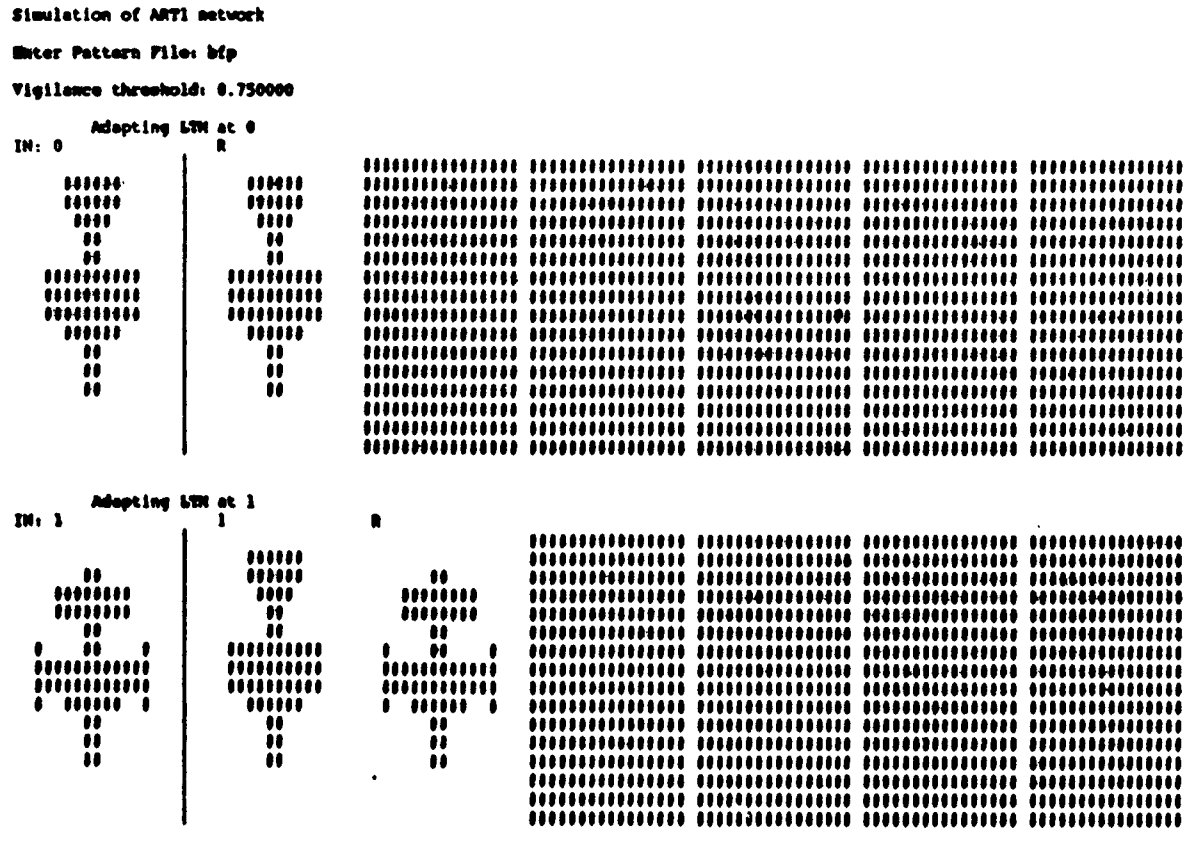

Figure 9: ART 1: Translated shapes are stored as new patterns with rho = 0.75, while two non-translated assigned to same category.

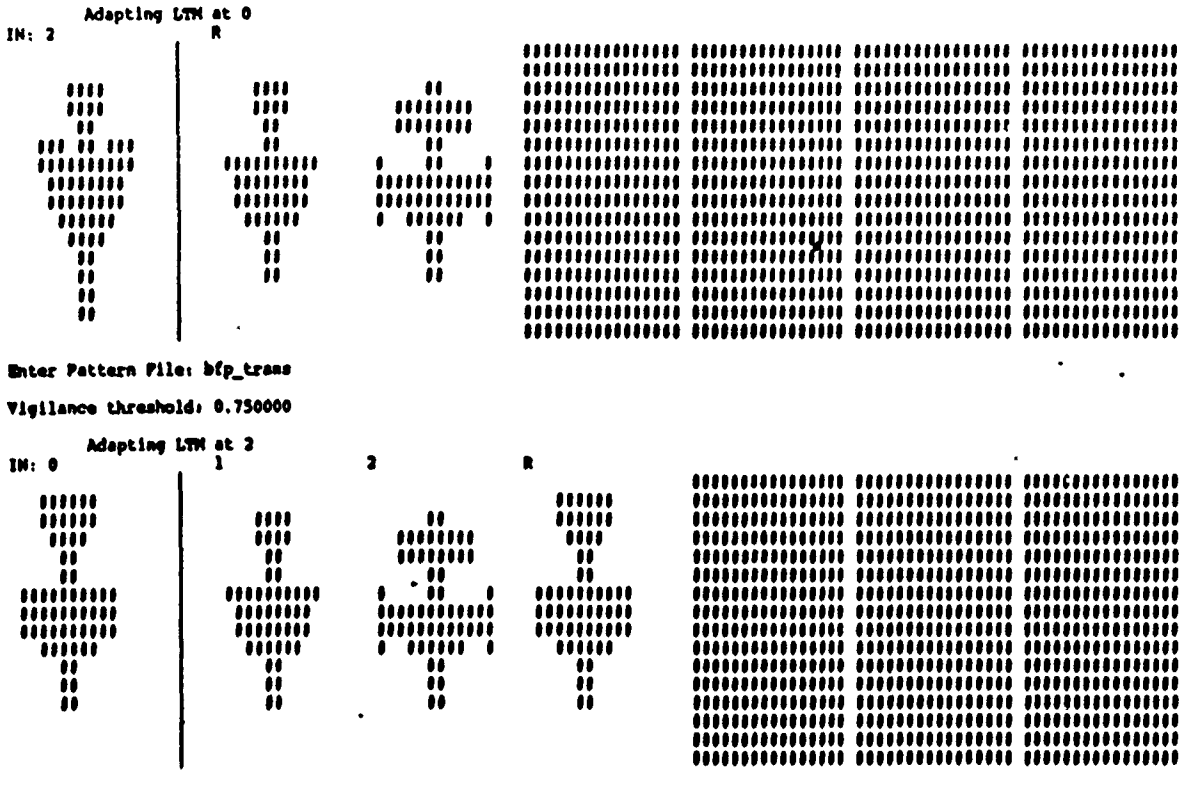

Figure 10: (Part b) ART 1: Translated shapes are stored as new patterns with $rho = 0.75$, while two non-translated assigned to same category.

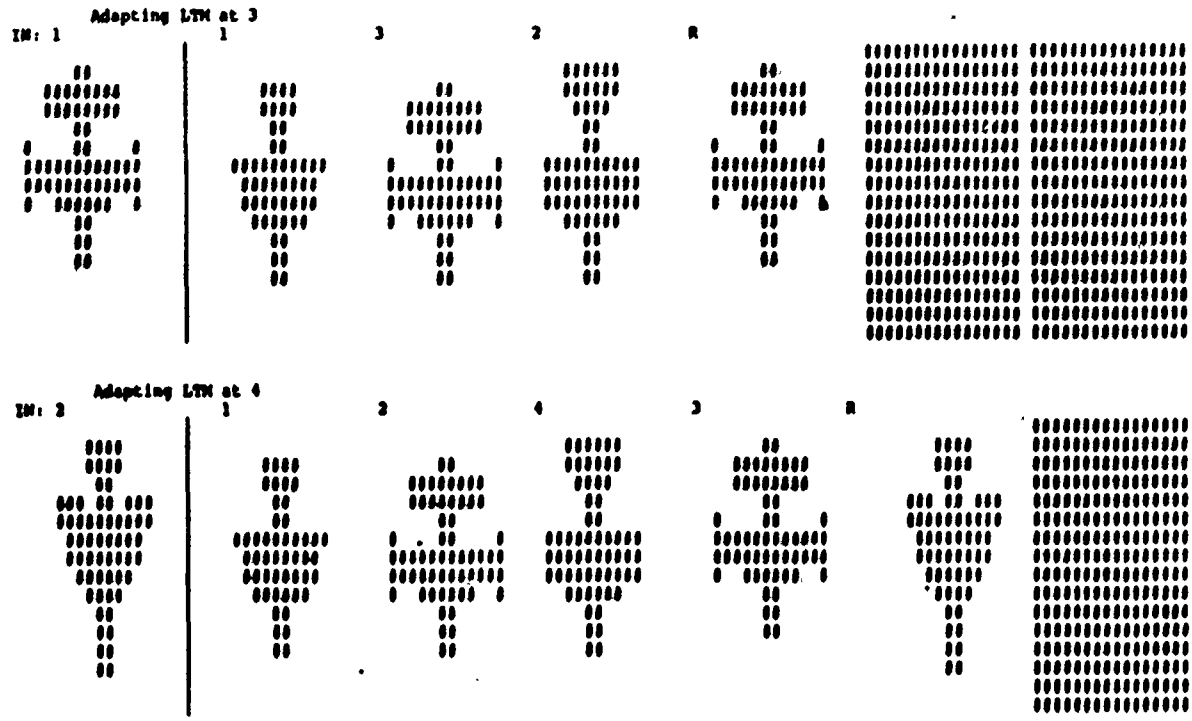

Figure 11: (Part c) ART 1: Translated shapes are stored as new patterns with $rho = 0.75$, while two non-translated assigned to same category.

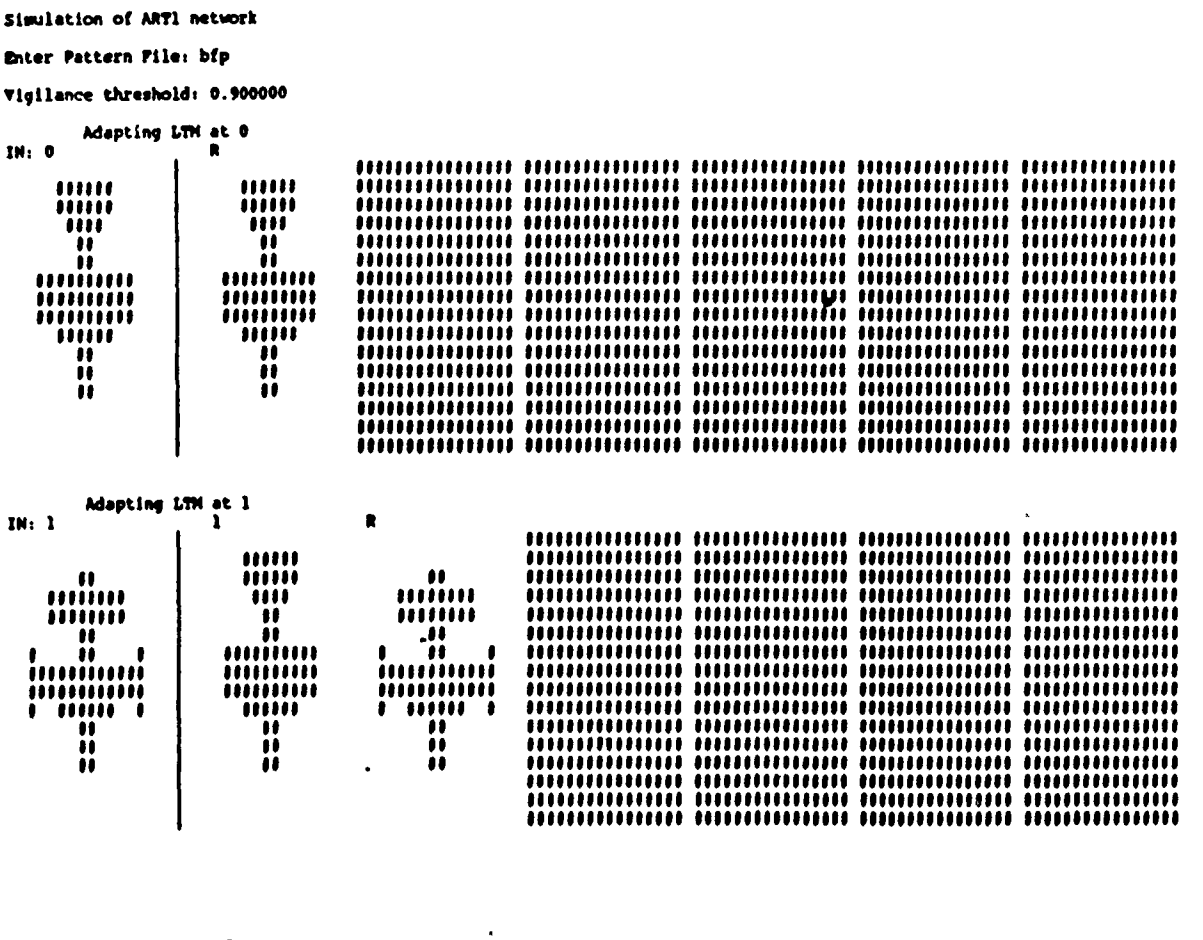

Figure 12: ART 1: Rotated shapes are stored as new patterns with $rho = 0.9$.

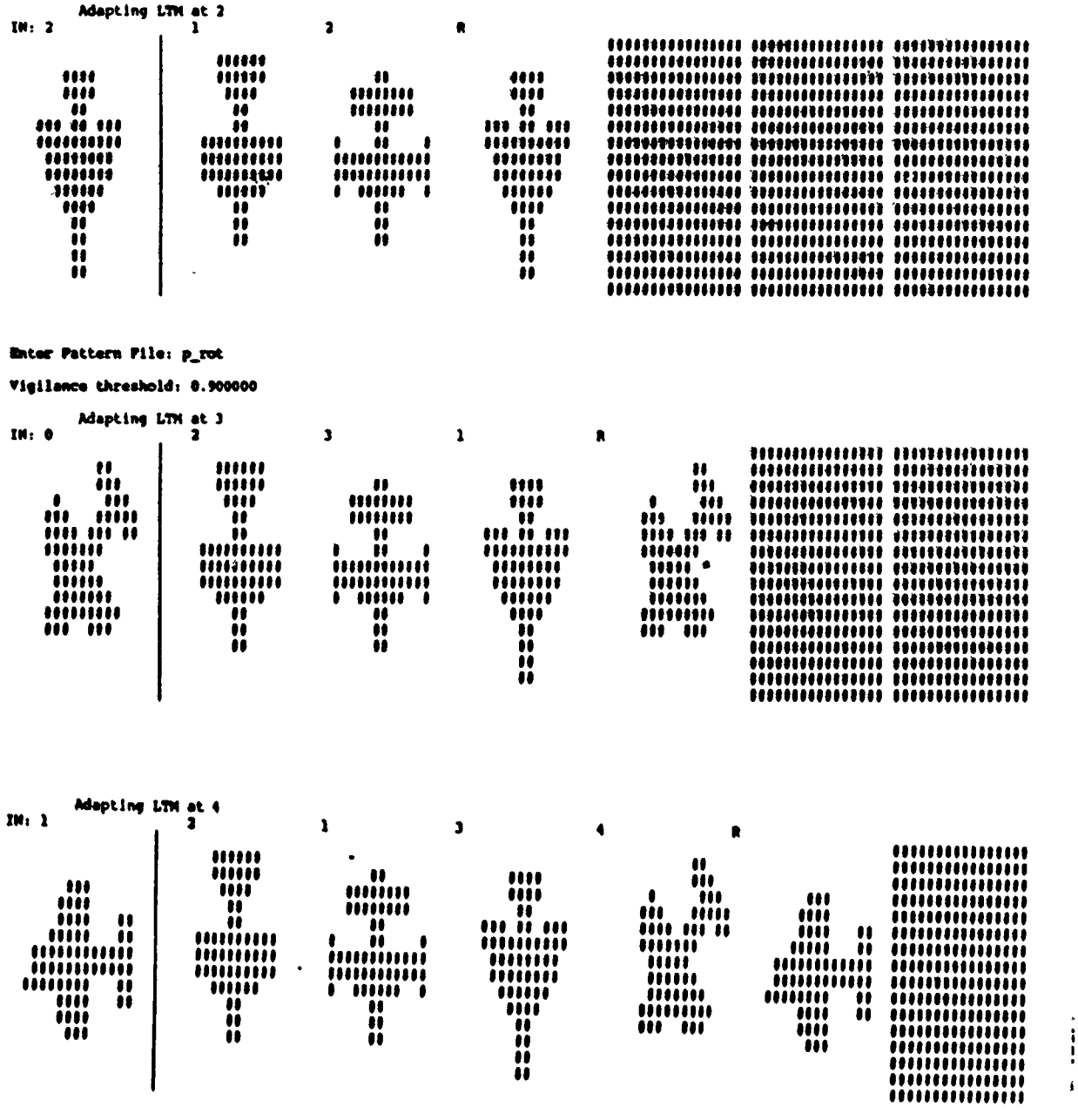

Figure 13: (Part b) ART 1: Rotated shapes are stored as new patterns with $rho = 0.9$.

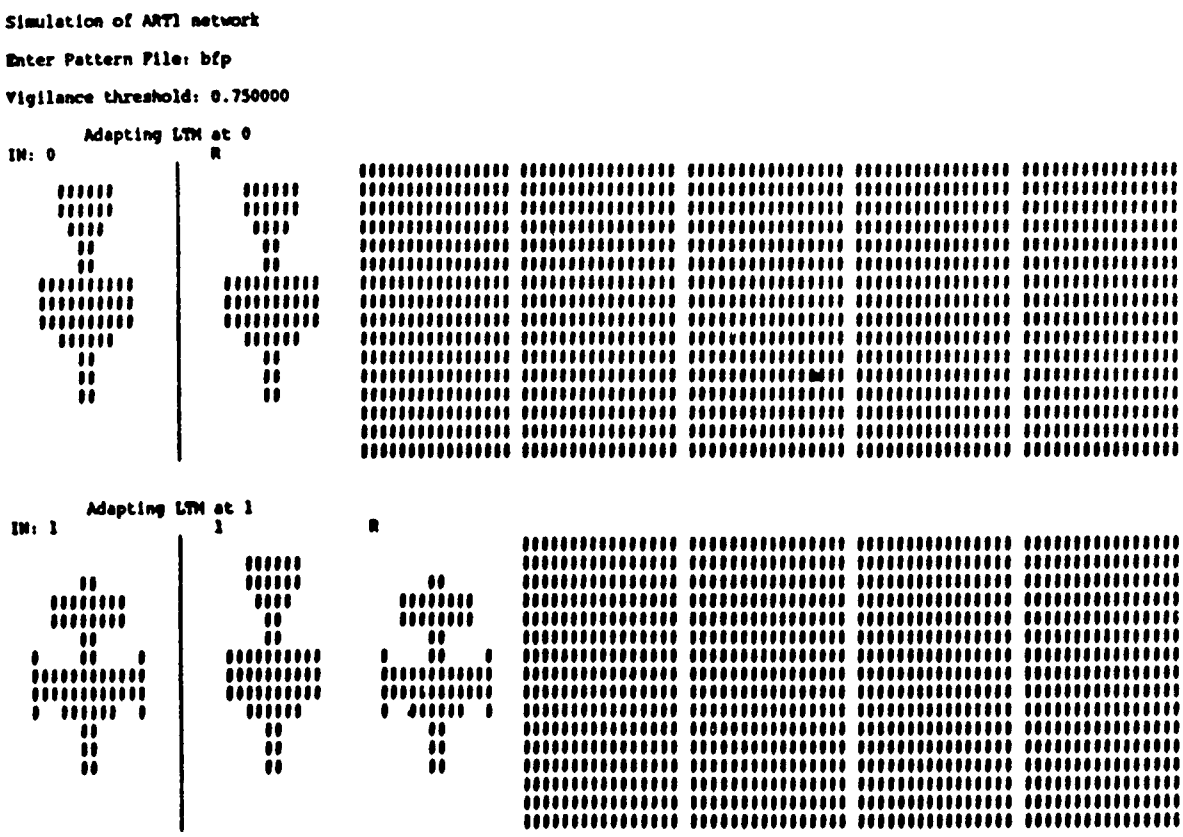

Figure 14: ART 1: Rotated shapes are stored as new patterns with rho = 0.75, while two non-rotated assigned to same category.

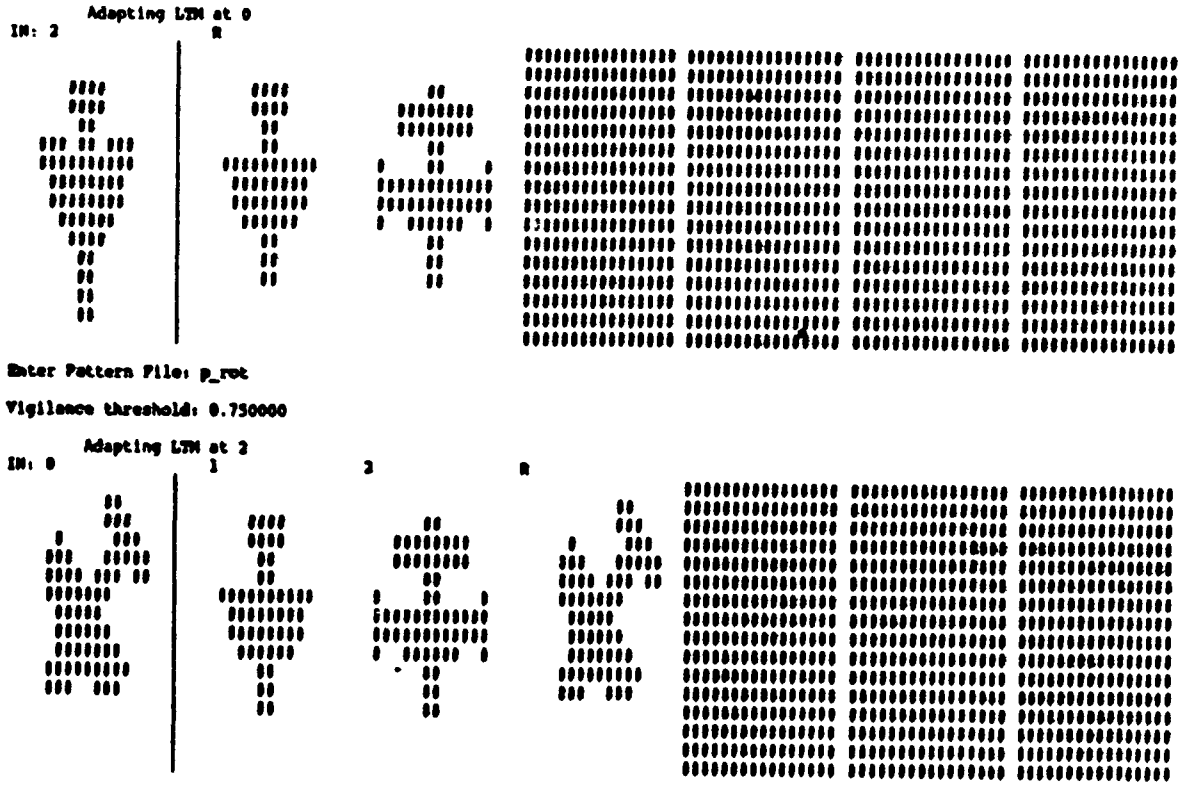

Figure 15: (Part b) ART 1: Rotated shapes are stored as new patterns with $rho = 0.75$, while two non-translated assigned to same category.

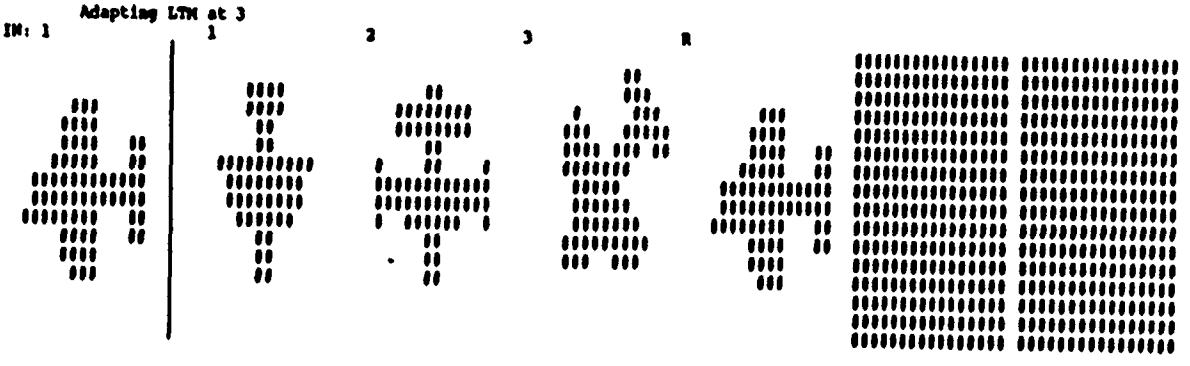

Figure 16: (Part c) ART 1: Rotated shapes are stored as new patterns with $rho = 0.75$, while two non-rotated assigned to same category.

7.1 Networks of Oja & Linsker

Oja [32] has investigated the information processing capabilities for a simple feedforward neuron having a linear output activation function,

$$y_i = a_1 + \sum_{j=1}^{n} w_{ij} x_j \ ,$$

using Hebbian type adaptation of the weights. The Hebb synaptic weights are constrained by the total resources available for the neuron to the form connection, e.g.

$$w_{ij}(t+1) = \frac{w_{ij}(t) + \gamma y_i(t) x_j(t)}{\sqrt{\sum_{j=1}^{n} [w_{ij}(t) + \gamma y_i(t) x_j(t)]^2}}$$

This leads to a weight vector update or adaptation rule that can be approximated as

$$\underline{w_i}(t+1) = \underline{w}(t) + \gamma y_i(t)[\underline{x}(t) - y_i(t)\underline{w}(t)].$$

Let the correlation matrix for a stationary input vector $\underline{x}$ be $C_x = E\{\underline{x}(t)\underline{x}^T(t)\}$. Oja found that a neuron so constructed develops a weight vector, $\underline{w}$, that is the normalized first eigenvector of the correlation matrix C_x corresponding to the largest eigenvalue. The output of the linear neuron is a linear combination of the inputs which maximizes the variance of the output. In other words, the neuron performs the first principle components transformation on the input.

In a similar, but larger study, Ralph Linsker [25,26,88] investigated the self-organizing abilities of a multi-layer feedforward network that employs local connections from one layer to the next with Hebb type learning. The network was motivated by the visual systems of early mammals. Learning was performed first at the lowest levels. Then the later levels where then trained. The network was stimulated with totally unstructured random input images. The network forms "feature analyzing cells" at each layer. The first layer of neurons form excitatory connections. The second layer develops "opponent cells" having a on-center off-surround receptive fields. Layers three, four, and five form "on-center" cells with the correlation between outputs within each layer having the "Mexican-hat" function as a function of intercellular distance only. The sixth layer forms bi-lobed or tri-lobed "edge" detectors. Islands of cells having similar orientations are formed. The spatial

layout or topology of these orientation selective cells is very similar to that observed in the macaque monkey.

This suggests that multi-layered feedforward networks with Hebbian type learning can self-organized to extract those features necessary for visual pattern recognition.

8 Neocognitron

Fukushima has proposed a self-organizing network called the Neocognitron[4,5,6,7,8,9,10,11,12,13]. This network is a multi-layer feedforward network. The network was constructed in a manner to categorized input patterns unaffected by shifts in position and some distortion of the shape.

8.1 Architecture

The network topology or architecture is multi-layered in order to obtain hierarchical recognition or clustering of detected features. The first plane, $u_{c0}(x,y) = u_{c0}(\underline{n})$, consists of the two-dimensional input image. This provides the inputs or stimuli to the first layer of processed elements or cells.

The network was devised in such a way as to grossly model the early vision processing of mammals. As such, the input is processing by a 'layer' of cells whose output is passed to the next layer. Processing continues up until the final output layer. The cell-plane or position of the maximally responding cell in this final layer is an indication of the category into which the input image pattern has classified. Each layer of cells is actually a pair of cell layers each performing a different function. The first layer of cells are called *S-cells* and model the simple cells of the mammalian visual system. The output of the S-cells are used as inputs to the *C-cells* which model the complex cells. Each of these pairs of cell layers are subdivided into cell planes. A cell plane is a two-dimensional array of cells. Each cell in the cell plane uses the same feature detecting receptive field weights. The receptive field is the local neighborhood of cells in the previous layer that provide the excitatory input to a cell. The number of cells in each cell planes decreases at higher layers. In this way, a cell at the highest layer has a receptive field that effectively covers the entire original input cell plane or image.

The Neocognitron model was implemented in the C programming language. This implementation utilizes window management utilities to provide an interactive environment for investigating the behavior of this network. The details of the program are described in the Neotool User's Manual[14].

8.2 Processing

In the following sections the processing by the cells of each type are described.

8.3 S-cells

Using the notation $u_{Sl}(k_l, \underline{n})$, to denote the simple or S-cell in the k^{th} cell-plane in the l^{th} layer at position $\underline{n}$, each S-cell receives inputs from the previous C-cell planes in a local neighborhood (or receptive field) about the same position $\underline{n}$. The output of the S-cells is obtained by the function

$$u_{Sl}(k_l, \underline{n}) = r_l \cdot \phi\left[\frac{1 + \sum_{k_{l-1}=1}^{K_{l-1}} \sum_{\underline{\nu} \in B_l} a_l(k_{l-1}, \underline{\nu}, k_l)\, u_{C\,l-1}(k_{l-1}, \underline{n} + \underline{\nu})}{1 + \frac{r_l}{1+r_l} b_l(k_l) v_{C\,l-1}(\underline{n})}\right]$$

where a_l are the variable weights in the receptive field B_l, for the l^{th} layer. The weight b_l is the strength of the variable weight for the inhibitory input obtained as the the output of the v_c-cell. The gain constant r_l controls the selectivity of the S-cell. K_l is the number of cell planes in the l^{th} cell layer. The activation function, ϕ is defined as

$$\phi(x) = \begin{cases} x, & x \geq 0 \\ 0 & x < 0 \end{cases}$$

8.4 V_c-cells

The v_c-cells provides an inhibitory input to the S-cells. A single plane of v_c-cell exist at each layer. The output of the v_c-cell is calculated as

$$V_{C\,l-1}(\underline{n}) = \sqrt{\sum_{k_{l-1}=1}^{K_{l-1}} \sum_{\underline{\nu} \in B_l} c_{l-1}(\nu) u^2_{C\,l-1}(k_{l-1}, \underline{n} + \underline{\nu})}$$

The receptive weights, c_l, for the v_c-cells are fixed.

$$c_l(\underline{\nu}) = \frac{1}{C(l)}\alpha_l^{|\underline{\nu}|}$$

where $|\underline{\nu}|$ is the distance between the position $\underline{\nu}$ and the center of the receptive field. This indicates that the receptive field weights, c_l, should be peaked towards the center of the receptive field. For the aircraft shape recognition experiments described latter, this proved to adversely affect the ability to learn features extended over the entire receptive field. Therefore, for our implementation of the model, $c_l(\underline{\nu})$, are constant over the size of the receptive field, B_l. They are normalized by $C(l)$ so that their sum is unity, i.e.

$$\sum_{k_{l-1}=1}^{K_{l-1}} \sum_{\underline{\nu}\in B_l} c_{l-1}(\underline{\nu}) = 1.$$

This inhibitory signal is used to shunt the output of all the S-cells at the same position.

8.5 C-Cells

The C-cells are used to detect the occurrence of features detected by the S-cells. Summing the responses from the S-layer over a small receptive field or neighborhood, D_l, makes it possible to detect the occurrence of features even with moderate spatial shifts of the features. Similar to the S-cell, the C-cell receives inputs from the previous corresponding S-cell plane within the receptive field as well as an inhibitory input derived from the same S-cell layer outputs aggregated as the output of the v_s-cell at the same position. The C-cell output is calculated as

$$U_{Cl}(k_l, \underline{n}) = \psi\left[\frac{1 + \sum_{\underline{\nu}\in D_l} d_l(\underline{\nu}) u_{Sl}(k_l, \underline{n}+\underline{\nu})}{1 + v_{Sl}(\underline{n})} - 1\right]$$

The activation function, ψ is defined as

$$\psi(x) = \begin{cases} \frac{x}{\beta+x}, & x \geq 0 \\ 0 & x < 0 \end{cases}$$

where β is typically chosen to be 0.5.

8.6 V_s-Cells

The v_s-cells provide the inhibitory input to the C-cells. The output of the V_s-cells is calculated as

$$v_{Sl}(\underline{n}) = \frac{1}{K_l} \sum_{k_l=1}^{K_l} \sum_{\underline{\nu} \in D_l} d_l(\underline{\nu}) U_{Sl}(k_l, \underline{n} + \underline{\nu})$$

The receptive weights, d_l, for the v_s-cells are fixed and for our implementation of the model monotonically decrease with increasing $\|\underline{\nu}\|$. In particular,

$$d_l(\underline{\nu}) = \frac{1}{D(l)} r_l^{|\underline{\nu}|}$$

The constant $D(l)$ is chosen such that the sum is unity, i.e.

$$\sum_{\underline{\nu} \in D_l} d_l(\underline{\nu}) = 1.$$

8.7 Learning

The network self-organizes by reinforcing the weights in response to inputs to the network. The networks learns without a teacher. During the learning phase, input patterns are repeatedly applied as stimuli. A Hebbian type learning rule is used update the individual weights. In order for the network to be capable of responding in a unique way to input stimuli, all the variable weights must not be simultaneously updated. Therefore, a competition is set up so the receptive field weights of the most strongly responding cells to a stimulus are reinforced. Representative cells for reinforcement are selected by finding those cells with the maximum response with a local neighborhood. The individual S-cell planes can be interpreted as being stacked into S-columns similar to the hypercolumns discovered in mammalian visual cortex. Cells in the same S-column defined as those cells at the same position $\underline{\nu}$ but belong to different cell planes k_l compete. Only the strongest responding cell in the S-column can be reinforced. If a cell is suppressed by another in another cell plane, then another cell in the same plane can become the candidate for reinforcement. Let a representative cell $u_s(\hat{k}_l \hat{\underline{n}})$, for the l^{th} layer on the $\hat{k}_l^{th}$ cell-plane, at position $\hat{\underline{n}}$. be chosen for reinforcement. Corresponding to each S-cell plane, k_l, in each layer, l, are a set of excitatory

weights with a receptive field in each cell plane of the previous layer, k_{l-1}, and provides input to the S-cell at position $\underline{\nu}$, i.e. $a_l(k_{l-1}, \underline{\nu}, k_l)$. In addition, each cell plane has an inhibitory weight associated with the v_c-cell, $b_k(k_l)$. These variable weights are reinforced according to the rule

$$\Delta a_l(k_{l-1}, \nu, \hat{k}_l) = q_l \cdot c_{l-1}(\underline{\nu} \cdot u_{C\,l-1}(k_{l-1}, \hat{\underline{n}} + \underline{\nu}),$$

and

$$\Delta b_l(\hat{k}_l) = q_l \cdot v_{C\,l-1}(\hat{\underline{n}}),$$

where q_l is a the positive learning constant for layer l. The values of the excitatory variable weights are initially set to small values but with differing orientation sensitivity. The variable inhibitory weights were initialized to zero.

8.8 Computational Complexity

The computational complexity of the Neocognitron model when processing the input image as it is feed forward to the output level can be closely approximated by the number of multiply-accumulates required. This is summarized in the formula below:

$$\text{number of Mult./Accum.} \doteq \sum_{l=1}^{L} K_l \left[|S_l| \left(2 \cdot |B_l| \cdot K_{l-1} + 1\right) + |C_l| \left(2 \cdot |D_l| + 1\right) \right]$$

where K is the number of layers, $|S_l|$ and $|C_l|$ are the number of cells (or x,y positions) in the l_{th} layer of the S and C cell-planes, respectively. $|B_l|$ and $|D_l|$ are the number of cells in the receptive fields for the S and C cells for the l_{th} layer, respectively.

8.9 Experiments & Results

In the following section a set of classification experiments are described. The network is trained with overhead views of three aircraft derived from planar patch models. The three aircraft used are the B57, F104, and the Phantom. The images are two-level of size 16x16. These shape images are shown in Figure 17. An important aspect is that the shapes do not fill a large fraction of the total image area (62 pixels out of 256.) For all the experiments described below the same network parameters where used. These parameters are given in Table 2.

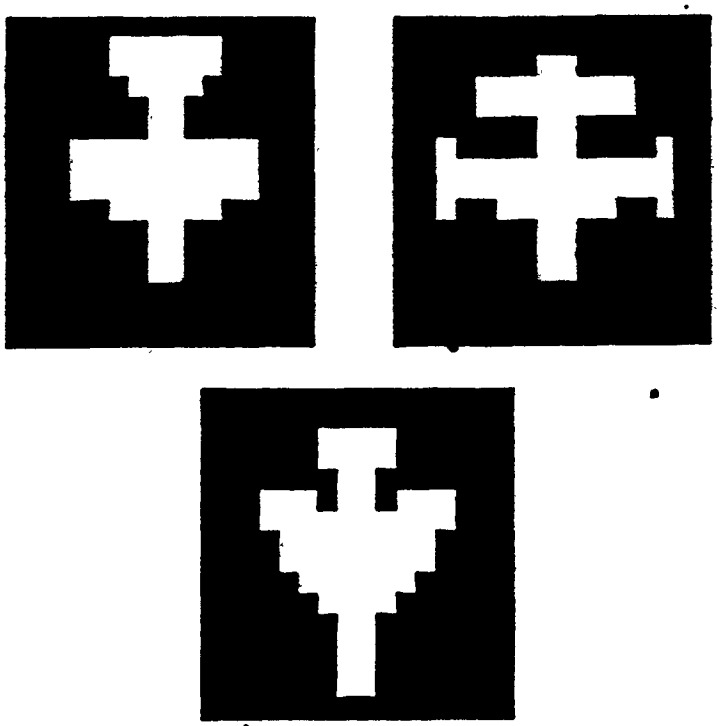

Figure 17: The three aircraft silhouettes used to train the Neocognitron.

Level	No. Planes	S-plane Size	C-plane Size	S-rec. Area	C-rec. Area	r_l	q_l	S-col. Size
1	6	14x14	12x12	5x5	3x3	20	2	7x7
2	6	10x10	8x8	5x5	3x3	15	16	5x5
3	6	6x6	4x4	5x5	3x3	10	16	3x3
4	6	2x2	1x1	3x3	3x3	10	20	2x2

Table 2: Neocognitron network parameters for three aircraft silhouettes.

8.9.1 Initial Training

The three image of the aircraft were used to train the network. The initial variable weights are shown in Figure 18.

This is accomplished by sequential applying each image to the input layer. Representative cells from the first S-layer are selected automatically and the corresponding receptive fields from the input layer used to update the variable weights. Training of the first layer variable weights is performed before learning is begun on the second layer. This is the case for layers three and four as well. Training is completed at the lower layers before training occurs at the next layer. After ten cycles of training, the variable weights have stabilized. The final values of the excitatory variable weights are shown in Figure 19.

After the learning of the receptive fields had stabilized, the image of each aircraft maximally excited a unique cell in the output layer. This is summarized in Table 3.

The processing of the F104 aircraft silhouette image through each of the four levels is shown in Figures 20 - 23.

Considerable effort was required to adjust the network parameters to assign unique categories to the three aircraft and provide some invariance to translation, and noise as described in the experiments below. The v_c-cell fixed inhibitory weights, $c_l(\underline{\nu})$, are described by Fukushima to be peaked at the center of the receptive field. During the course of this investigation, adjustment of the network parameters clearly indicates that the peaked fixed receptive field weights caused a reduction in the network's capability to discriminate among spatially distributed patterns. For this reason, the v_c-cell fixed weights were constant over the receptive field. This implies then that the S-columns should be of at least the size of the variable receptive field

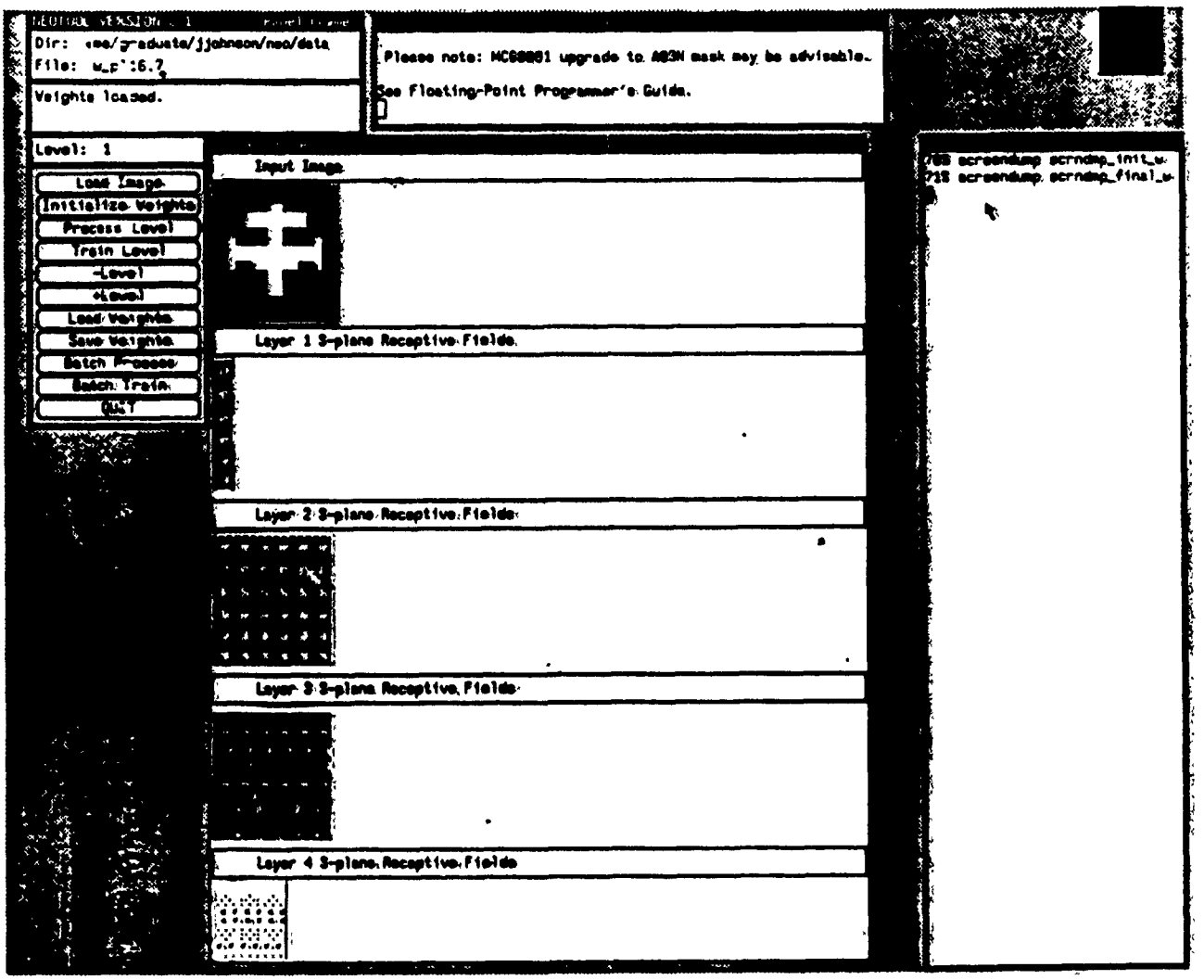

Figure 18: Neocognitron: Initial variable excitatory receptive field weights for all four layers.

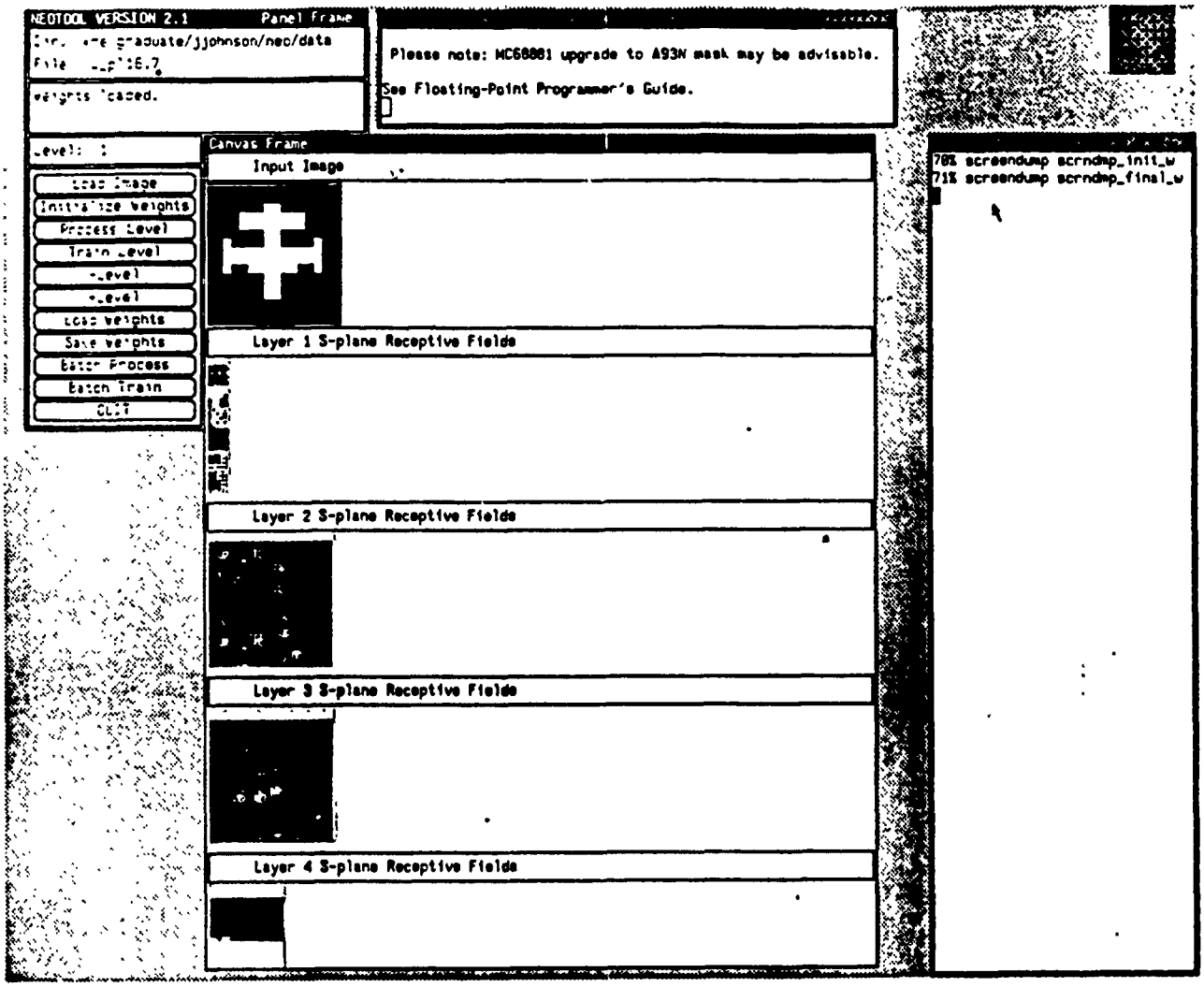

Figure 19: Neocognitron: Final variable excitatory receptive field weights for all four layers.

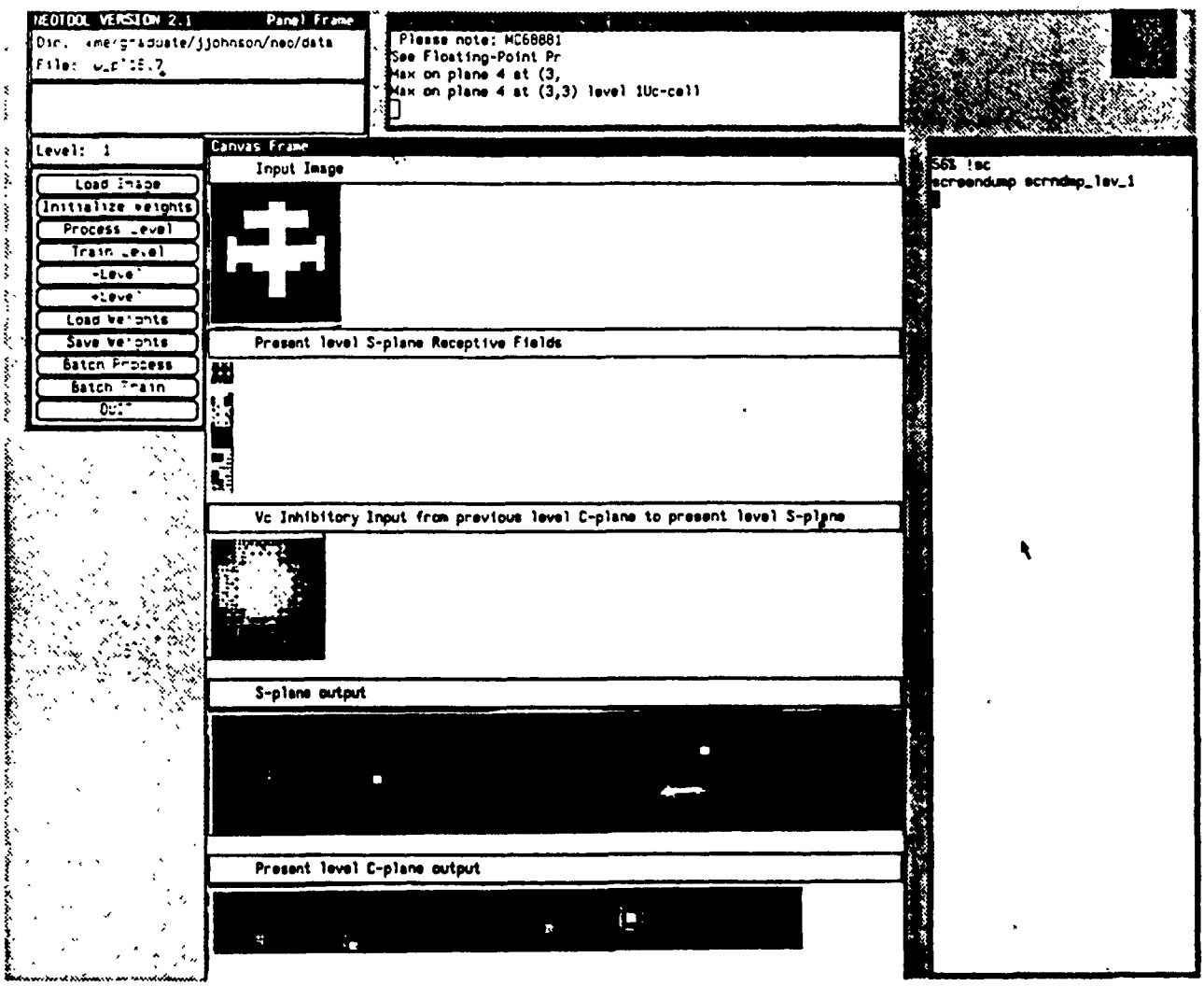

Figure 20: Neocognitron: Processing of the F104 silhouette through the first layer of cells.

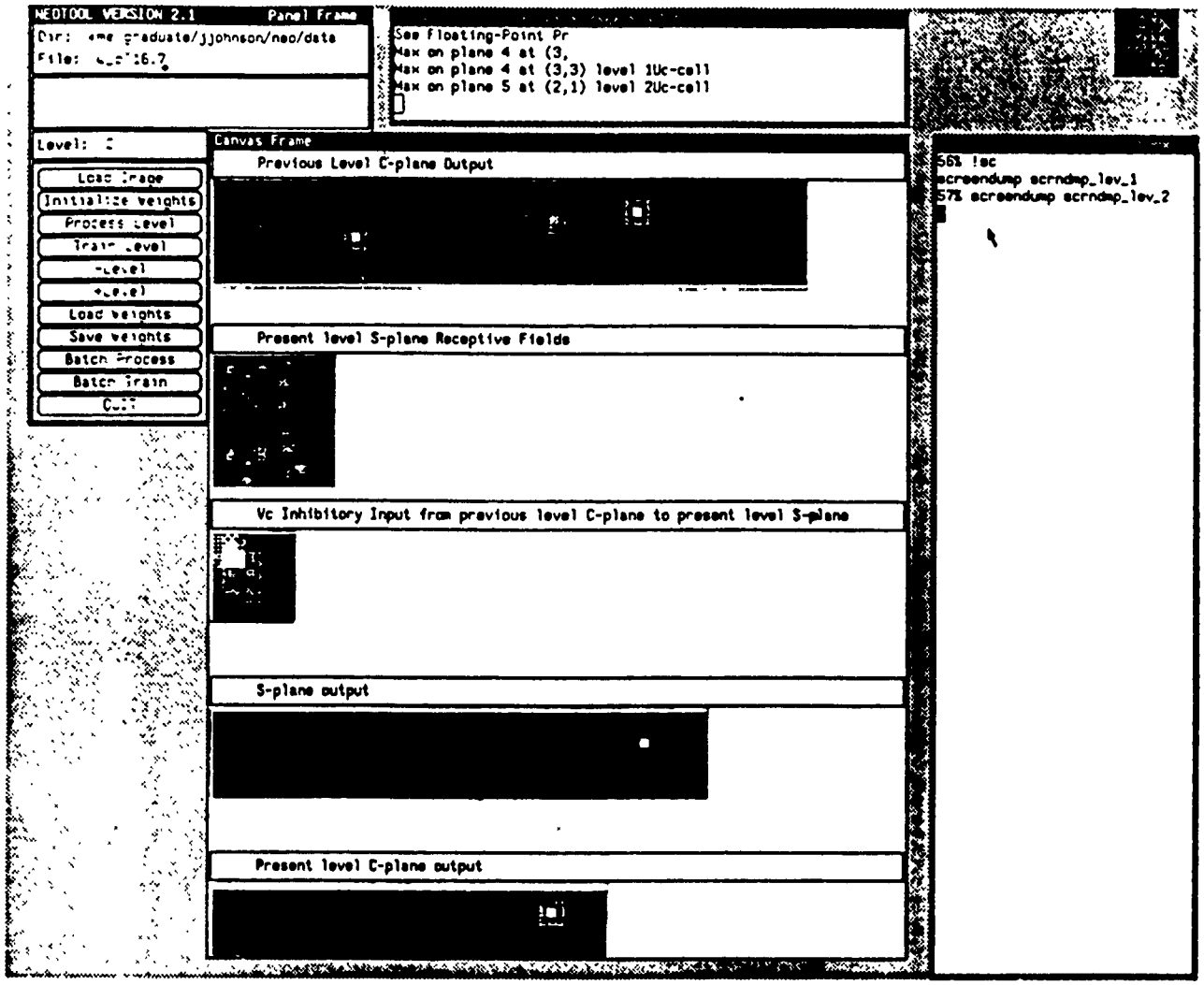

Figure 21: Neocognitron: Processing of the F104 silhouette through the second layer of cells.

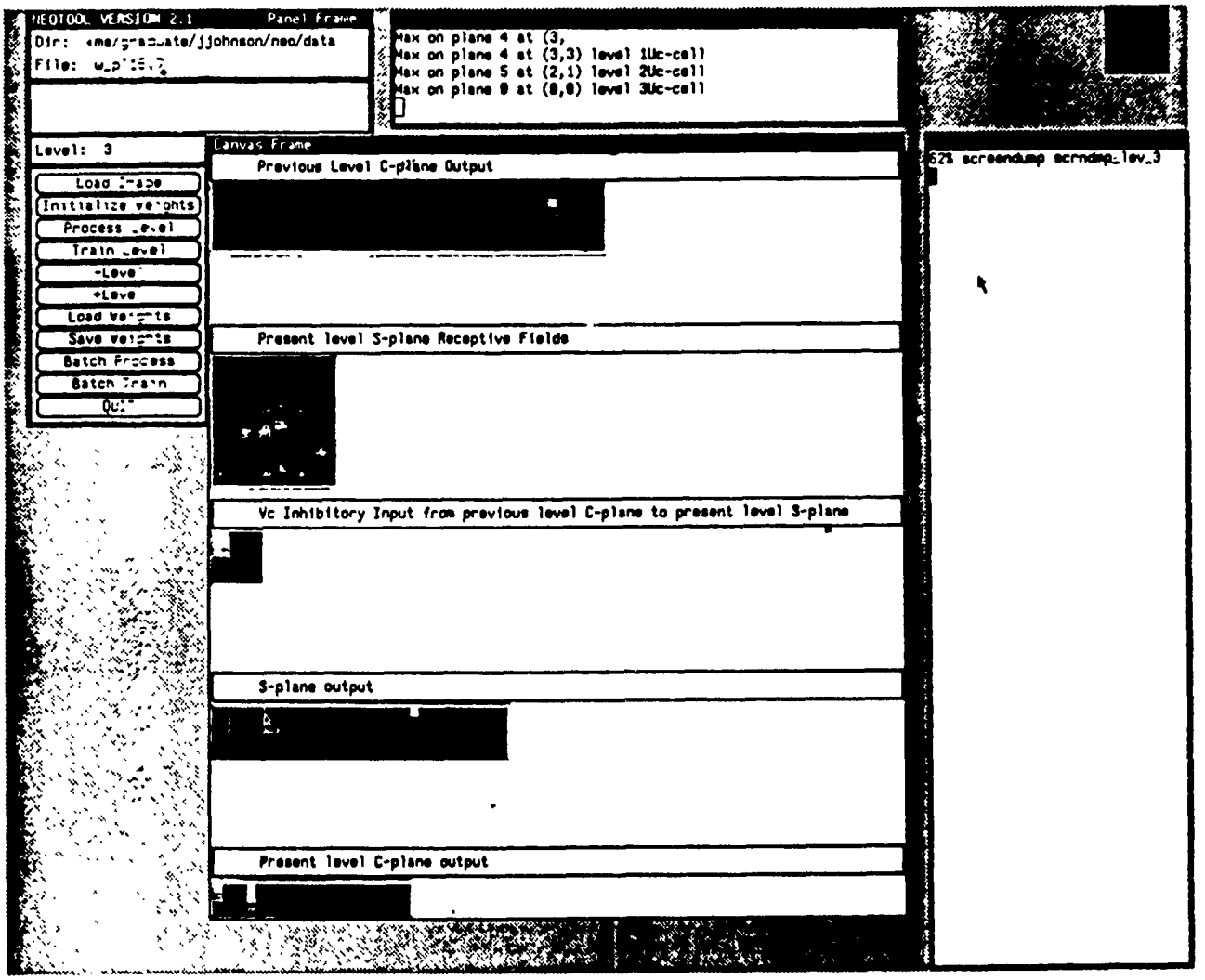

Figure 22: Neocognitron: Processing of the F104 silhouette through the third layer of cells.

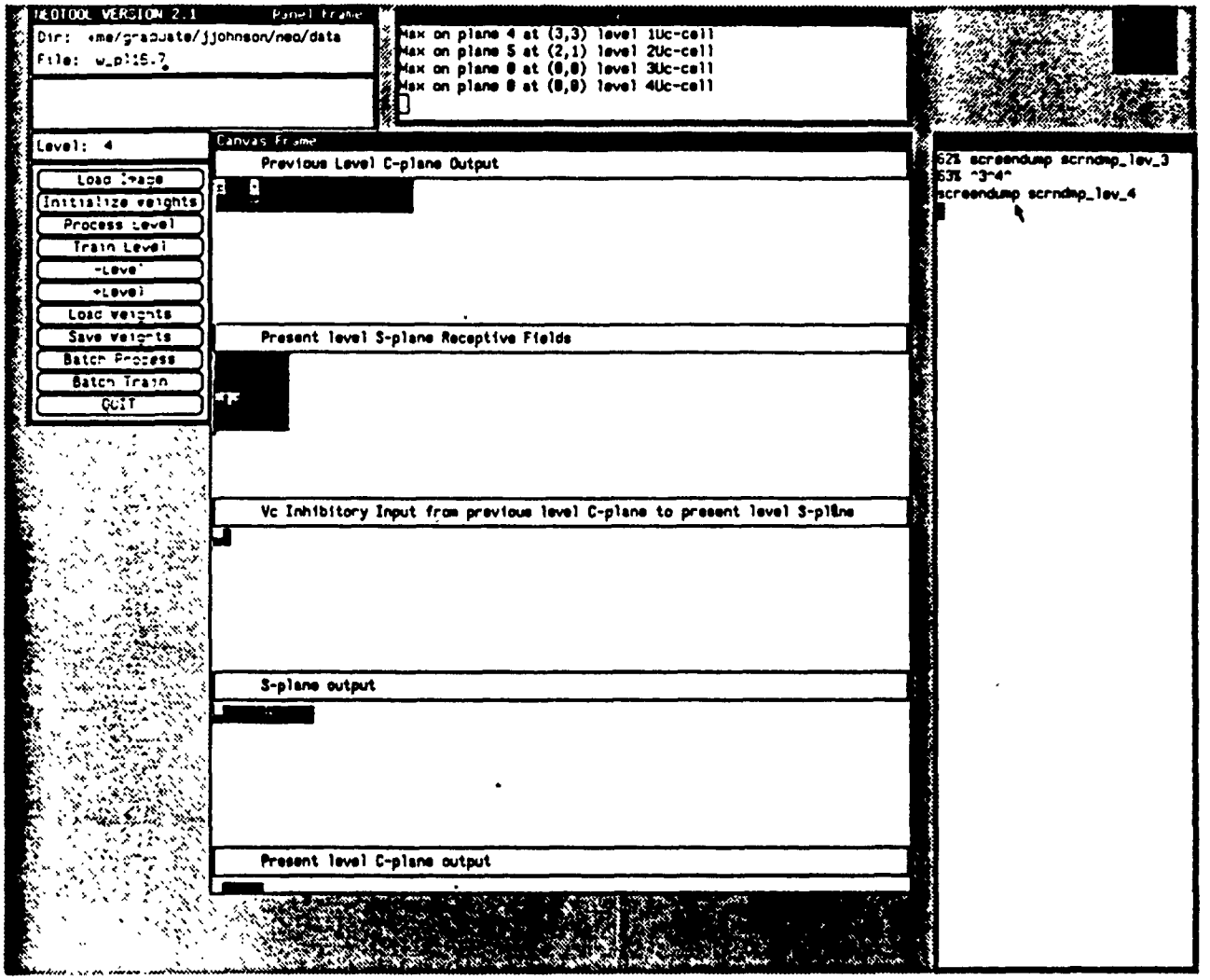

Figure 23: Neocognitron: Processing of the F104 silhouette through the fourth layer of cells.

Aircraft Image	Output C-plane with Max Response
B57	0
F104	1
Phantom	3

Table 3: Self-organized categorization of three aircraft images by Neocognitron.

weights do not learn overlapping local spatial features.

8.9.2 Translation Invariance

An experiment was performed to test the translation invariance of the network. Translated version of the three aircraft images were applied to the network that had been previously organized using the original three aircraft images. These translated shapes are shown in Figure 24. The results of this experiment are summarized in Table 4. below. The translated F104 aircraft shape assigned to the same category as the B57. The Neocognitron is reported in the literature to be translation invariant. However, even with our numerous attempts at adjusting the network parameters we were unable to uniquely categorize the shapes the same in both the untranslated and translated images. After carefully scrutinizing the model behavioral description, it becomes evident that there is a trade-off in the network's ability to discriminate and to provide translation invariance. The profile of the C-cell plane fixed receptive field weights control the ability to recognize shifted patterns in the previous layer. The flatter (and larger in spatial extent) of these fixed weights the better the translation detection ability. However, this has a unfortunate side-effect. It also causes a blurring of that C-cell response. This results in the next layer being force to work with quite indistinct features, and thereby reducing the discriminability of the network overall. For the aircraft shapes, at a (less than) 16x16 resolution, the shape are already very similar. The network is unable to simultaneously provide the necessary discriminability and translation invariance. Menon and Heinemann mention this problem [30], but the two vehicle shapes (a tank and a truck at a resolution of approximately 64x64 were already quite different and posed much less a difficulty. They reported that they could shift the shapes 50 percent

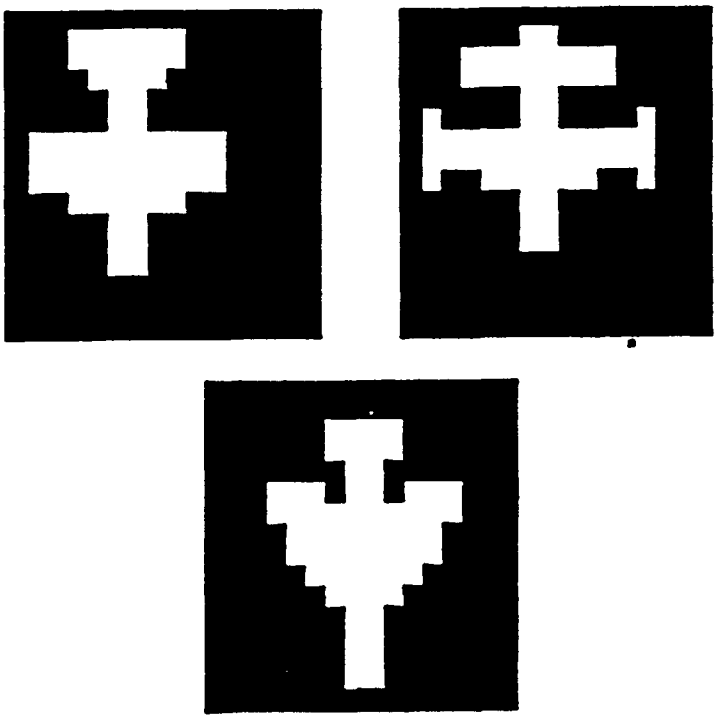

Figure 24: The three translated aircraft silhouettes used to test the Neocognitron.

Aircraft Image	Output C-plane with Max Response	Original Output C-plane with Max Response
B57	0	0
F104	0	1
Phantom	3	3

Table 4: Self-organized categorization of the three translated aircraft images by the Neocognitron.

Aircraft Image	Output C-plane with Max Response	Original Output C-plane with Max Response
B57	5	0
F104	0	1
Phantom	0	3

Table 5: Self-organized categorization of the three rotated aircraft images by the Neocognitron.

of the total image size.

8.9.3 Rotation

Next an experiment was designed to test the ability of the network to categorize rotated shapes. The rotated shapes were *not* used to train the network. Fukushima [6] has reported results of good categorization of distorted shapes, in particular handprinted (strokes) characters. The network has *not* been reported to perform rotation invariant categorization. The three aircraft shapes were rotated 10 degrees in the plane of the image. The rotated silhouettes are shown in Figure 25. These rotated shape images were then categorized by the previously trained network. The results are summerized in Table 5.

The rotated B57 is assigned to a completely new category. The F104 and the Phantom are assigned to the category previously assigned to the B57. It is clear that some other preprocessing of the original image is necessary to provide the capability of rotation invariant categorization.

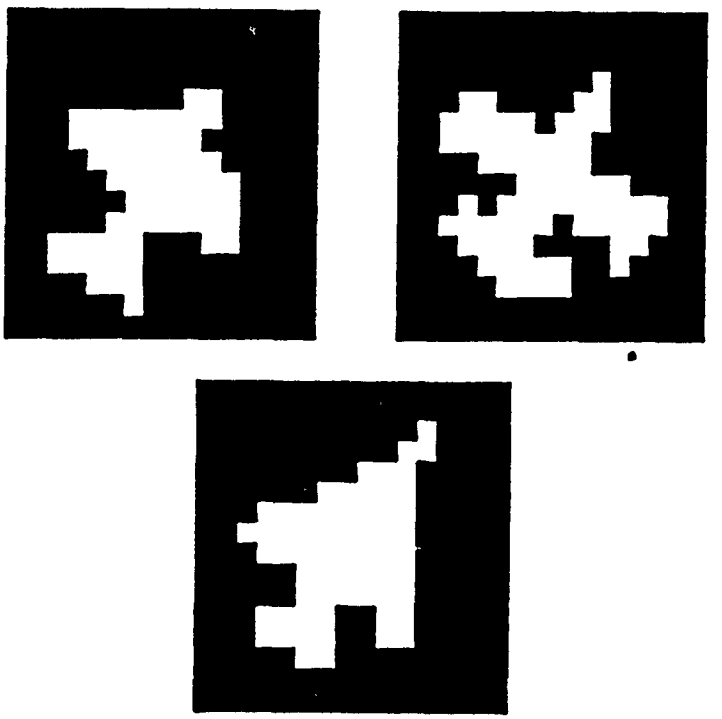

Figure 25: The three rotated aircraft silhouettes used to test the Neocognitron.

Aircraft Image	Output C-plane with Max Response	Original Output C-plane with Max Response
B57	0	0
F104	1	1
Phantom	1	3

Table 6: Self-organized categorization by the Neocognitron of the three noisy aircraft images, $Pr(0 \rightarrow 1) = 1/256$.

Aircraft Image	Output C-plane with Max Response	Original Output C-plane with Max Response
B57	0	0
F104	1	1
Phantom	3	3

Table 7: Self-organized categorization by the Neocognitron of the three noisy aircraft images, $Pr(0 \rightarrow 1) = 4/256$.

8.9.4 Noise

An experiment was performed to investigate the sensitivity of the network to noisy patterns. Noisy two-level images were synthesized by adding noise and then thresholding in such a way to change random background pixels into foreground pixels. The noise levels are described as the probability of a background pixel changing to a foreground pixel ($0 \rightarrow 1$). The noisy B57 silhouette shapes are shown in Figure 26. The results are summarized in Tables 6 - 8.

At all three noise levels, the B57 and F104 are assigned their original

Aircraft Image	Output C-plane with Max Response	Original Output C-plane with Max Response
B57	0	0
F104	1	1
Phantom	2	3

Table 8: Self-organized categorization by the Neocognitron of the three noisy aircraft images, $Pr(0 \rightarrow 1) = 39/256$.

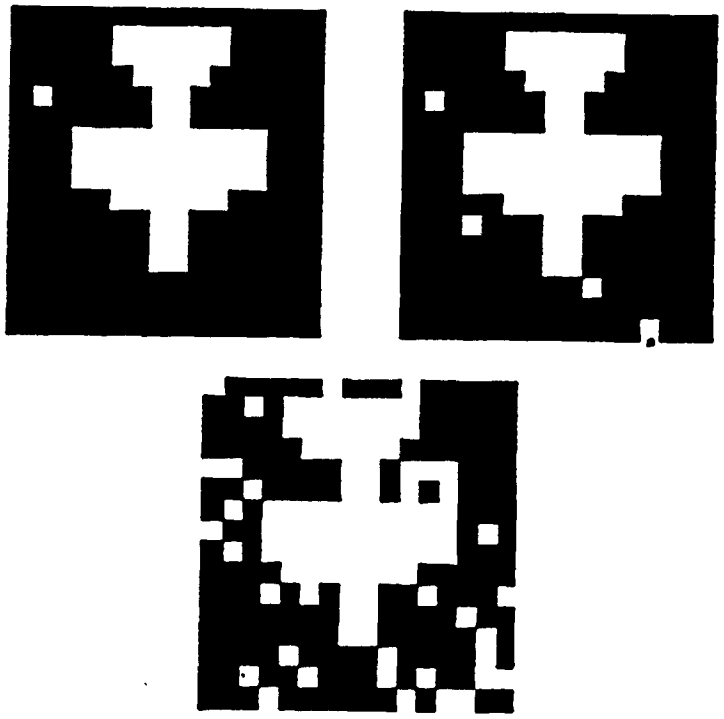

Figure 26: The B57 aircraft silhouette shown at the three noise levels.

shape category. The Phantom assigned the same category at the lowest, the original category at the middle, and a unique category at the highest noise level.

9 Conclusions

The experimental data described in this report indicates that if the input patterns have already labels of known significance, then supervised learning neural network paradyms should be utilized in place those network employing an unsupervised learning technique. The unsupervised networks should be used in those situations where no informative label is available and it is the task of the system to organize or to induce an order on the input patterns.

Both the Neocognitron and the ART networks would be more useful in organizing spatial patterns if preceded by processing making the patterns invariant to rigid geometric transformations. The Neocognitron can be made invariant to translation, but only at the cost of reduced sensitivity to pattern shape variability. The on-line learning property of the ART network could then be used in those scenarios where adaptability to a changing pattern environment was important. Most of the supervised learning networks must be retrained on all the original training data set as well as the new patterns or the old patterns will be forgotten.

One of the authors at the time of this final report has implement a three-layer error back-propagation supervised learning network. Network training is being carried for the identification of three aircraft from arbitrary viewing angles.

This study indicates that for the task of aircraft identification and orientation estimation unsupervised learning does not offer the required performance. However, the self-organizing systems might be useful for feature extraction or reduction. However, some information content is lost in the categorization process. Care must therefore be taken to insure that information pertinent to the ultimate recognition task is not eliminated.

10 References

1. A. G. Barto, R.S. Sutton, and C.W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Trans. Syst, Man, and Cybern.*, Vol. SMC-13, no. 5, pp 834-846, 1983.

2. G. Carpenter and S. Grossberg, "A massively parallel architecture for a self-organizing neural pattern recognition machine," *Computer Vision, Graphics, and Image Processing*, Vol. 37, pp. 54-115, 1987.

3. G. Carpenter and S. Grossberg, "ART2: self-organization of stable category recognition codes for analog input patterns," *Applied Optics*, Vol. 26, no. 23, pp. 4919-4930, 1 December, 1987.

4. K. Fukushima, "Cognitron: a self-organizing multilayered neural network," *Biological Cybernetics*, Vol. 20, pp. 121-136, 1975.

5. K. Fukushima, "Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, Vol. 36, pp. 193-202, 1980.

6. K. Fukushima, "Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position," *Pattern Recognition*, Vol. 15, no. 6, pp. 455-469, 1982.

7. K. Fukushima, "Neocognitron: a neural network for a mechanism of visual pattern recognition," *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. SMC-13, pp. 826-834, 1983.

8. K. Fukushima, "A hierarchical neural network model for associative memory," *Biological Cybernetics*, Vol. 50, pp. 105-113, 1984.

9. K. Fukushima, "A neural network model for selective attention in visual pattern recognition," *Biological Cybernetics*, Vol. 55, pp. 5-15, 1986.

10. K. Fukushima, "Neural network model for selective attention in visual pattern recognition and associative recall," *Applied Optics*, Vol. 26, no. 23, pp. 4985-4992, 1987.

11. K. Fukushima, "A neural network for visual pattern recognition,", *IEEE Computer*, pp. 65-75, March, 1988.

12. K. Fukushima, "Neocognitron: A hierarchical neural network capable of visual pattern recognition," *Neural Networks*, Vol. 1, pp. 119-130, 1988.

13. K. Fukushima, "Analysis of the process of visual pattern recognition by the Neocognitron," *Neural Networks*, Vol.2, pp. 413-420, 1989.

14. T.A. Grogan and J.D. Johnson, "Neotool User's Manual: An implementation of the Neocognitron," *Univ. of Cincinnati, Dept. of ECE, Technical Report*, in preparation, 1990.

15. Donald O. Hebb, *The Organization of Behavior*, Wiley, New York, 1949.

16. J.J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities," *Proc. Natl. Acad. Sci.*," USA, Vol. 79, pp.2554-2558, April, 1983.

17. J.J. Hopfield, "Neurons with Graded Response Haved Collective Computational Properties Like of Two-State Neurons," *Proc. Natl. Acad. Sci.*," USA, Vol. 81, pp.3088-3092, May, 1984.

18. J.J. Hopfield and D.W. Tank, "Computing with Neural Circuits: A Model," *Science*, Vol. 233, pp. 625-633, August, 1986.

19. J.D. Johnson and T.A. Grogan, "Neural network controlled visual saccades," *Proceedings of the SPIE - Image Understanding and the Man-Machine Interface*, Vol. 1076, pp. 44-49, 17-18 January, 1989, Los Angeles, California.

20. Tuevo Kohonen, *Self-Organization and Associative Memory*, Springer-Verlag, New York, 1983 and 1987.

21. Tuevo Kohonen, "Self-organized formation of topologically correct feature maps," *Biological Cybernetics*, Vol. 43, pp. 59-69, 1982.

22. Tuevo Kohonen, "Adaptive, associative, and self-organizing functions in neural computing," *Applied Optics*, Vol.26, no. 23, pp. 4910-4918, 1 December, 1987.

23. Tuevo Kohonen, "The "neural" phonetic typewriter," *IEEE Computer*", pp.11-22, March 1988.

24. A.H. Klopf, "A neuronal model of classical conditioning," *AFWAL-TR-87-1139*, USAF Wright Aeronautical Laboratories, Dayton, OH, 1987.

25. Ralph Linsker, "From basic network principles to neural architecture: Emergence of spatial-opponent cells", *Proc. Natl. Acad. Sci. USA*, Neurobiology, Vol. 83, pp. 7508-7512, October, pp.8390-8394, November, and pp. 8779-8783, November, 1986.

26. Ralph Linsker, "Self-organization in a perceptual network," *IEEE Computer*, pp.105-117, March, 1988.

27. R. P. Lippmann, "An introduction to computing with neural networks," *IEEE ASSP Magazine*, Vol. 3, no. 4, pp. 4-22.

28. G.G. Lorentz, "The 13th Problem of Hilbert," in *Mathematical Developments Arising from Hilbert Problems*, F.E. Browder (Ed.), American Mathematical Society, Providence, R.I., 1976.

29. W.S. McCulloch and W. Pitts, "A Logical Calculus of the Ideas Imminent in Nervous Activity," *Bull. of Mathematical Biophysics*, Vol.5, pp. 115-133, 1943.

30. M. M. Menon and K. G. Heinemann, "Classification of patterns using a self-organizing neural network," *Neural Networks*, Vol. 1, pp. 201-215, 1988.

31. M. Minsky and S. Papert, *Perceptrons: An Introduction to Computational Geometry*, MIT Press, 1969.

32. E. Oja, "A simplified neuron model as a principal component analyzer," *J. Math. Biology*, Vol. 15, 1982, pp. 267-273.

33. D.E. Rummelhart, J.L. McClelland, and R.J. Williams, "Learning Internal Representations by Error Propagation", in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol.1: Foundations*, D.E. Rummelhart and J.L. McClelland (eds.), MIT Press, 1986.

34. T. D. Sanger, "Optimal unsupervised learning in a single-layer linear feedforward neural network," *Neural Networks*, Vol.2, pp. 459-473, 1989.

35. T. Sejnowski and C.R. Rosenberg, "NETtalk: A Parallel Network That Learns to Read Aloud,", *Johns Hopkins Univ. Technical Report JHU/EECS-86/01*, 1986.

DISTRIBUTION LIST

Commander
Armament Research, Development and Engineering Center
U.S. Army Armament, Munitions and Chemical Command
ATTN: SMCAR-IMI-I (5)
SMCAR-FSF-RC (15)
Picatinny Arsenal, NJ 07806-5000

Commander
U.S. Army Armament, Munitions and Chemical Command
ATTN: AMSMC-GCL(D)
Picatinny Arsenal, NJ 07806-5000

Administrator
Defense Technical Information Center
ATTN: Accessions Division (12)
Cameron Station
Alexandria, VA 22304-6145

Director
U.S. Army Materiel Systems Analysis Activity
ATTN: AMXSY-MP
Aberdeen Proving Ground, MD 21005-5066

Commander
Chemical Research, Development and Engineering Center
U.S. Army Armament, Munitions and Chemical Command
ATTN: SMCCR-MSI
Aberdeen Proving Ground, MD 21010-5423

Commander
Chemical Research, Development and Engineering Center
U.S. Army Armament, Munitions and Chemical Command
ATTN: SMCCR-RSP-A
Aberdeen Proving Ground, MD 21010-5423

Director
Ballistic Research Laboratory
ATTN: AMXBR-OD-ST
Aberdeen Proving Ground, MD 21005-5066

Chief
Benet Weapons Laboratory, CCAC
Armament Research, Development and Engineering Center
U.S. Army Armament, Munitions and Chemical Command
ATTN: SMCAR-CCB-TL
Watervliet, NY 12189-5000

Commander
U.S. Army Armament, Munitions and Chemical Command
ATTN: SMCAR-ESP-L
Rock Island, IL 61299-6000

Director
U.S. Army TRADOC Systems Analysis Activity
ATTN: ATAA-SL
White Sands Missile Range, NM 88002