Cognitive Processing Hardware Elements

Sponsored by

Defense Advanced Research Project Agency (DoD)
(Controlling DARPA Office)

DARPA Order S016-37

Issued by U.S. Army Aviation and Missile Command Under
Contract No. W31P4Q-04-C-R279

Name of Contractor: Cardinal Research LLC
Dr. Bernard Widrow, President
Business Address: 860 Lathrop Dr.
Stanford CA. 94305-1053

Effective Date of Contract: June 1, 2004

Short Title of Work: Development of a Cognitive Memory System

Contract Expiration Date: January 31, 2005
Reporting Period: June 1, 2004 to January 30, 2005

Disclaimer
The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

Declaration of Technical Data Conformity
The contractor, Cardinal Research, LLC, hereby declares that, to the best of its knowledge and belief, the technical data delivered herewith under Contract No. W31P4Q-04-C-R279 is complete, accurate, and complies with all requirements of the contract.

Date: January 31, 2005
Name and title of Authorized Official: signed by Dr. Bernard Widrow, President.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

REPORT DATE *(DD-MM-YYYY)*	**1. REPORT TYPE**	**3. DATES COVERED** *(From – To)*
31-01-2005	Final report	01-06-2004 to 31-01-2005

4. TITLE AND SUBTITLE
Cognitive Processing Hardware Elements

5a. CONTRACT NUMBER
W31P4Q-04-C-R279

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Widrow, Bernard, Dr., Eliashberg, Victor, Dr., Kamenetsky, Max, Mr.

5d. PROJECT NUMBER
RTWBF04

5e. TASK NUMBER
Tasks 1,2,3,4,5,6

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Cardinal Research, LLC
860 Lathrop Dr.
Stanford, CA. 94305-1053

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Defense Advanced Research Project Agency (DoD)/ US Army Aviation & Missile Command AMSAM-AC-CS-RAY Redstone Arsenal AL 35898-5280

DCMA San Francisco P O BOX 232 700 East Roth Road Bldg 330 (Lathrop, CA) French Camp CA 95231-0232

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Distribution unlimited

DISTRIBUTION STATEMENT A
Approved for Public Release
Distribution Unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The purpose of this research is to identify and develop cognitive information processing systems and algorithms that can be implemented with novel architectures and devices with the goal of achieving high-speed decision making and cognitive responses to sensor inputs, based on current and prior experience. The research focuses on memory, i.e. the development of a "human-like" cognitive memory. Although memory is only a single component of a cognitive information processing system, it is a critical and central component, one that is essential for learning. A cognitive memory has been computer simulated, tested, and applied to four potentially high-value DOD problems. They are, location and tracking an airplane's position by comparing telescopic images of the ground with a previously recorded aerial photo, location and identification of aircraft at an airfield from satellite imagery, reading Chinese characters even if they are distorted, and human face recognition. Future work will involve improvements in the application techniques and the development of hardware for high-speed cognitive memories.

15. SUBJECT TERMS
Cognitive information processing, autoassociative memory, pattern recognition, parallel search.

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Dr. Bernard Widrow
U	U	U	SAR	63	**19b. TELEPHONE NUMBER** *(include area code)* 1-650-857-9151

Final Report
U.S. Army Contract No.W31P4Q-04-C-R279

by
Cardinal Research LLC
Dr. Bernard Widrow
Dr. Victor Eliashberg
Mr. Max Kamenetsky

January 31, 2005

This report describes research performed under subject contract during June 1, 2004–January 31, 2005. In reference to Proposal No. D041–019–0270 entitled Cognitive Processing Hardware Elements, the reported work pertains to Task 1 of section 4.2.1, Task 2 of section 4.2.2, Task 3 of section 4.2.3, and Task 4 of section 4.2.4, Task 5 of section 4.2.5, and Task 6 of section 4.2.6.

Task 1

This Task has the objective of identifying candidate DOD applications for cognitive information processing. A cognitive memory system design has been proposed. The objective of Task 1 is therefore to identify likely DOD applications for the proposed cognitive memory. A preliminary set of applications have been selected, and work has been done toward their realization. These applications are:

1. Location and tracking of the position of an aircraft relative to a previously obtained aerial photo. We believe that the precise location of an aircraft can be determined by using a combination of optical images of the ground below and a previously obtained wide area aerial photo of the ground below. Aircraft locations have been determined to within one pixel, within several feet. Results are presented below.

2. Satellite surveillance of an airfield, with a cognitive memory used to automatically detect and locate aircraft and determine their type. Results are presented below and in an accompanying CD–ROM that demonstrates by means of a video the detection, location, and identification of various aircraft types at an airfield of Diego Garcia Island.

3. Use of a cognitive memory to identify large number of Chinese characters. Results will be presented below.

4. Recognition of human faces.

There are many other possible applications such as facial recognition and facial feature identification, fingerprint analysis and recognition, etc., but the above three applications have taken our attention.

Task 2

The objective of this task is to identify the algorithms and structures that would be required for efficient real–time implementation. As we worked with selected applications, we realized that the proposed cognitive memory structure would be suitable for finding solutions. At the outset, all work was able to be done by implementing a small cognitive memory with a conventional PC. A full–scale system could be implemented with a cluster of PC's. However, we believe that more efficient and more parallel operation could be obtained with a bank of DSP boards. The relative economies are not clear at the moment. The PC architecture may not be the best, but PC's are in high–volume production and are relatively cheap. Custom ASIC's would be another approach. It is too early now to decide on final cognitive memory architecture. We must have a great deal more experience with applications and know of their speed requirements before we could design an efficient general–purpose cognitive memory.

Task 3

The objective of this task is to computer simulate a small but practical cognitive memory that could be used to implement the selected applications. This was done. We have a MATLAB simulation of a cognitive memory that runs on a conventional PC. Application results will be presented below.

Task 4

The objective of this task is to apply the computer simulated cognitive memory to the selected applications. Very good results were obtained, and they are explained below. This experience convinces us that the area of application for cognitive memory will be very wide indeed.

Task 5

The objective of this task is to design an implementation of the cognitive memory by means of custom parallel hardware. A design has been made based on FPGA technology.

Task 6

The objective of this task is to analyse the speed of performance of the custom parallel hardware and compare to that of computer implementation. This has been done and

the speed of the system based on custom hardware would be about 1,000 times greater than that of software implementation on a state-of-the-art 64-bit PC.

The Cognitive Memory Concept

The current design for the cognitive memory has been changed somewhat based on our experience with this work, but the basic ideas are essentially the same as originally proposed.

The cognitive memory design described herein is based on concepts derived from life experience, from the literature of psychology, psychiatry, and neurobiology [5, 9, 20, 30, 34], and from years of experience in working with artificial neural networks and adaptive and learning systems. Certain conjectures about human memory are key to the central idea. The design of a practical and useful memory system is contemplated, a memory system that may also serve as a model for many aspects of human memory.

The cognitive memory design would be able to store in a unified electronic memory system visual inputs (pictures and sequences of pictures), auditory inputs (acoustic patterns and sequences of patterns), tactile inputs, inputs from other kinds of sensors such as radar, sonar, etc., and to retrieve stored content as required.

The memory would not function like a computer memory where specific data is stored in specific numbered registers and retrieval is done by reading the contents of the specified memory register, or done by matching key words as with a document search. The stored sensory data would neither have key words nor would it be located in known or specified memory locations. Incoming sensory data would be stored at the next available empty memory location, and indeed could be stored redundantly at several empty locations. In any event, the location of any specific piece of recorded data would be unknown.

Retrieval would be initiated by a prompt signal from a current set of sensory inputs or patterns. A search through the memory would be made to locate stored data that correlates with or relates to the present real-time sensory inputs. The search would be done by a retrieval system that makes use of autoassociative artificial neural networks [13].

Applications of cognitive memory systems to analysis of aerial imagery, human facial images, sounds, rote-learning for game-playing, low-level battlefield decision making, adaptive control systems, pattern recognition, and to other practical problems are being explored.

The proposed cognitive memory architecture would be scalable so that larger memories could store more sensory data, but storage and retrieval time would not increase with memory size. How this works will be seen below.

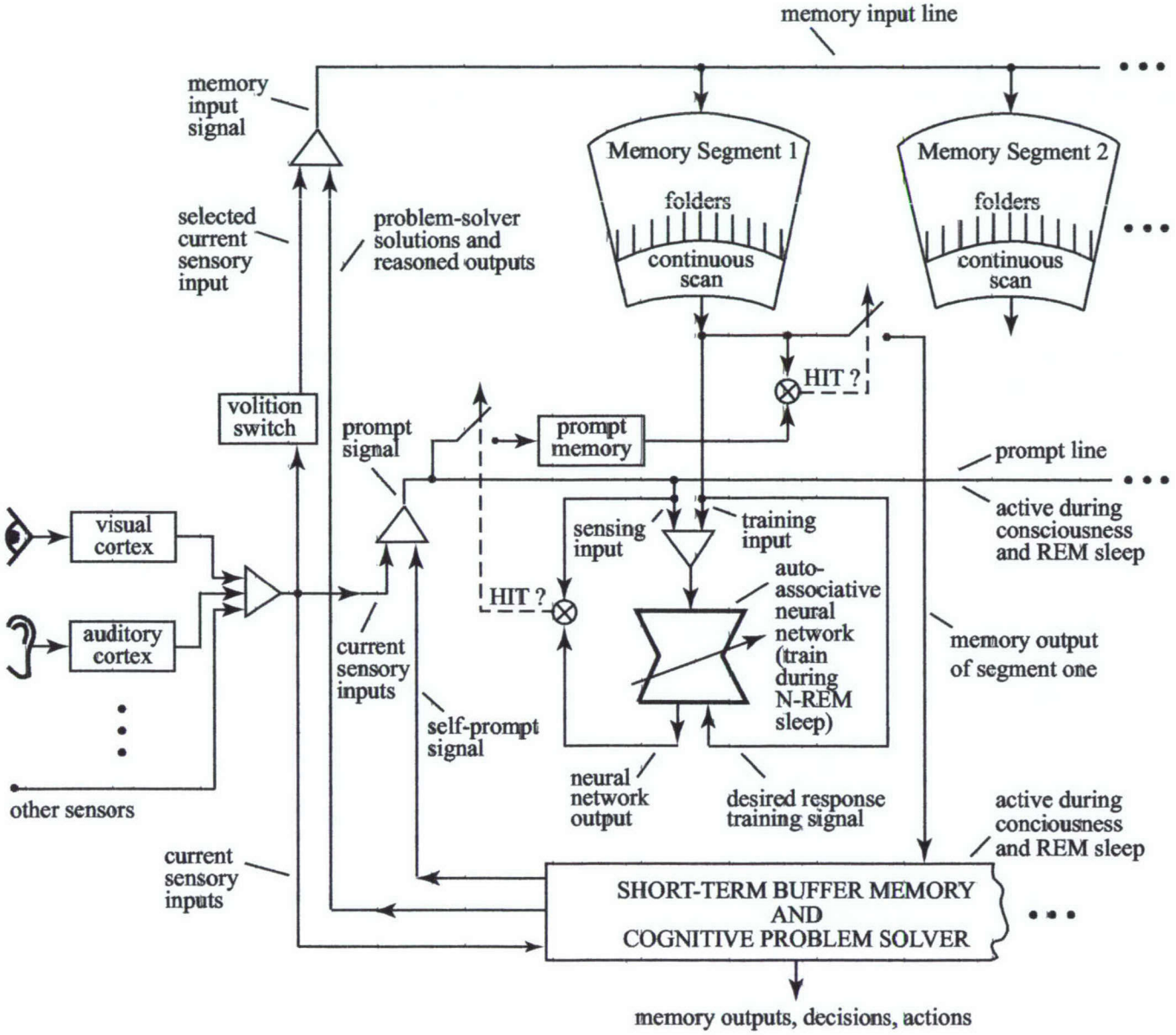


Figure 1: A cognitive memory system.

Design of a Cognitive Memory System

Figure 1 shows architectural elements and structures of a memory system that could behave to some extent like human memory. It would have practical engineering value and would be useful in solving military and commercial problems. This memory system is intended to model human memory function, and its workings will be described in human terms. The anatomical locations in the brain where the various architectural elements and components might be contained are mostly unknown. What is important is that the functions of these elements and components be performed.

The design of the cognitive memory system of Figure 1 is based on the following hypotheses about human memory:

1. During a lifetime, images, sounds, tactile inputs, etc. are stored permanently, if they were of interest when the sensory inputs were experienced. Human memory has enough storage capacity for a long lifetime. Old recordings are not deleted for lack of storage space.

2. Sensory inputs concerning a single object or subject are stored together as vectors in a single "file folder" or "memory folder." When the contents of the folder are retrieved, sights, sounds, tactile feel, smell, etc., are obtained all at the same time. Sensor fusion is a memory phenomenon. The sensory signals are not fused, but they are simply recorded together in the same folder and retrieved together.

3. Thoughts, conclusions, and problem solutions are also stored in memory, just like sensory signals.

4. The same information stored in a memory folder may be stored redundantly in a number of separate folders.

5. There may be many folders storing different information about the same subject, recorded on different days. Suppose you have a Bernard Widrow folder containing many different images of his face taken during a one hour visit, with various lighting conditions, scale, perspective, rotation, translation, with zoom-in images of his eyes, nose and other facial details. The folder also contains the sounds of the conversation. After many visits, there will be many Bernard Widrow folders. During one of the conversations, the name of his wife is mentioned. During retrieval, the contents of that particular folder would need to be read in order to recall the name of his wife.

6. Retrieval of stored information results from reading the contents of a folder when triggered by a prompt from a set of current sensory inputs, or by a thought process. Recalling the name of Widrow's wife would require a prompt, such as seeing Widrow, and the need at the end of the conversation of saying, "please give my best regards to Ronna Lee."

7. Current sensory inputs would have very little meaning and would be puzzling if they did not trigger the reading of the contents of folders containing related information. Current sensory inputs would trigger or prompt the delivery of the contents of folders containing experience that is related to the present input environment. For example, listening to and understanding the speech of another person requires access to the memory folders storing the sounds and associated meaning of each word and each combination of words or phrases. Without memory and memory access, one could hear speech but not understand it, similar to hearing a person speak an unknown foreign language.

8. Retrieval of the contents of the sought after folder or folders is done by association of the current sensory input or prompt signal with the folder contents. One would need to scan through the folders to make the association and find the right folder or folders. This needs to be done rapidly, using a method that allows the size of the memory to be increased without increasing the retrieval time.

9. When a search is prompted by current sensory inputs and a folder containing related information is found, all of the folder contents could provide self prompt signals to find additional related folders that were not found in the initial search.

10. A problem-solving process could create new patterns from sensory inputs. These new patterns could be stored in memory and could prompt new searches.

11. Associations are made by pattern matching or vector matching.

12. Features of patterns are portions of the patterns themselves, often zoomed-in portions.

13. The memory is organized in segments. Each segment contains a finite number of folders. Each segment contains its own retrieval system for searching its folders. When a search is prompted, separate but parallel searches take place in all memory segments simultaneously. Thus, search time does not increase with the number of segments or with the total size of the memory.

The Workings of the Cognitive Memory System of Figure 1

The cognitive memory system of Figure 1 has the capability of performing in accord with all of the above hypotheses. Sensory inputs are brought into the system in the lower left of the figure. These inputs could be visual, auditory, tactile, olfactory, etc. or, in a mechanistic system, optical, radar, sonar, etc. These current sensory inputs are made available to a short-term buffer memory that is part of a cognitive problem-solver, to be described below. The sensory inputs go through a "volition switch" before being stored. This allows only "interesting" inputs to go to permanent storage. For Phase I research, the volition switch will be bypassed. Ultimately, it may or may not be needed for a mechanistic system. This will be studied during Phase II.

The memory input is a signal vector that goes to all memory segments simultaneously. This same signal vector serves as a prompt signal for searching memory. Separate prompt and memory input lines are shown in the figure because there is evidence that in human memory, these are separate circuits [29]. The memory input vector to be stored goes from segment to segment looking for an empty folder. The memory input vector may be stored in more than one folder and in more than one segment for redundancy.

The recalled memory output vector is delivered to the short–term buffer memory and, along with the sensory input vector, feeds data to a cognitive problem-solver. The recalled memory output vector may actually be the desired memory output. Or it may be used as an important input to the problem-solver. A simple form of reasoning can be done by a problem-solver which, at the present time, is envisioned to be based on the classical work of Arthur Samuel. His checker-playing program embodies a reasoning process that plays by the rules, plays tentative moves ahead, and makes optimized decisions in order to win the game of checkers.

Game playing is a good model for a general reasoning process. Samuel's checker-player dating back to the 1950's and 1960's is still recognized as one of the finest pieces of work done in the field of artificial intelligence [27, 28]. The problem-solver can best be implemented by conventional off the shelf computer components (COTS) and will be a research subject for Phase II.

Computed outputs from the problem–solver can be stored and later retrieved from memory. Inputs to be stored by the memory come from both the current sensory inputs and outputs from the problem solver. The prompt signal can come from current sensory inputs or from self–prompt signals that result when the contents of a memory folder are delivered to the problem solver and some of the contents are determined to be of interest as prompts for further searches.

The entire cognitive memory system of Figure 1 could be built under Phase II. For Phase I, a part of the system, the cognitive memory of Figure 2, has been built in software and tested.

The applications to be presented below have been performed and implemented with the cognitive memory of Figure 2. The details of how this memory works will be discussed next. The block diagram of Figure 2 is simpler than that of Figure 1, making an explanation of how this works much easier.

Details of the Working of the Cognitive Memory Itself

The cognitive memory itself is shown in Figure 2. This has been built and tested, and has been used to implement the applications described below. This memory is the principal focus of the Phase I effort. The goal is to simulate it and analyze its behavior, and to

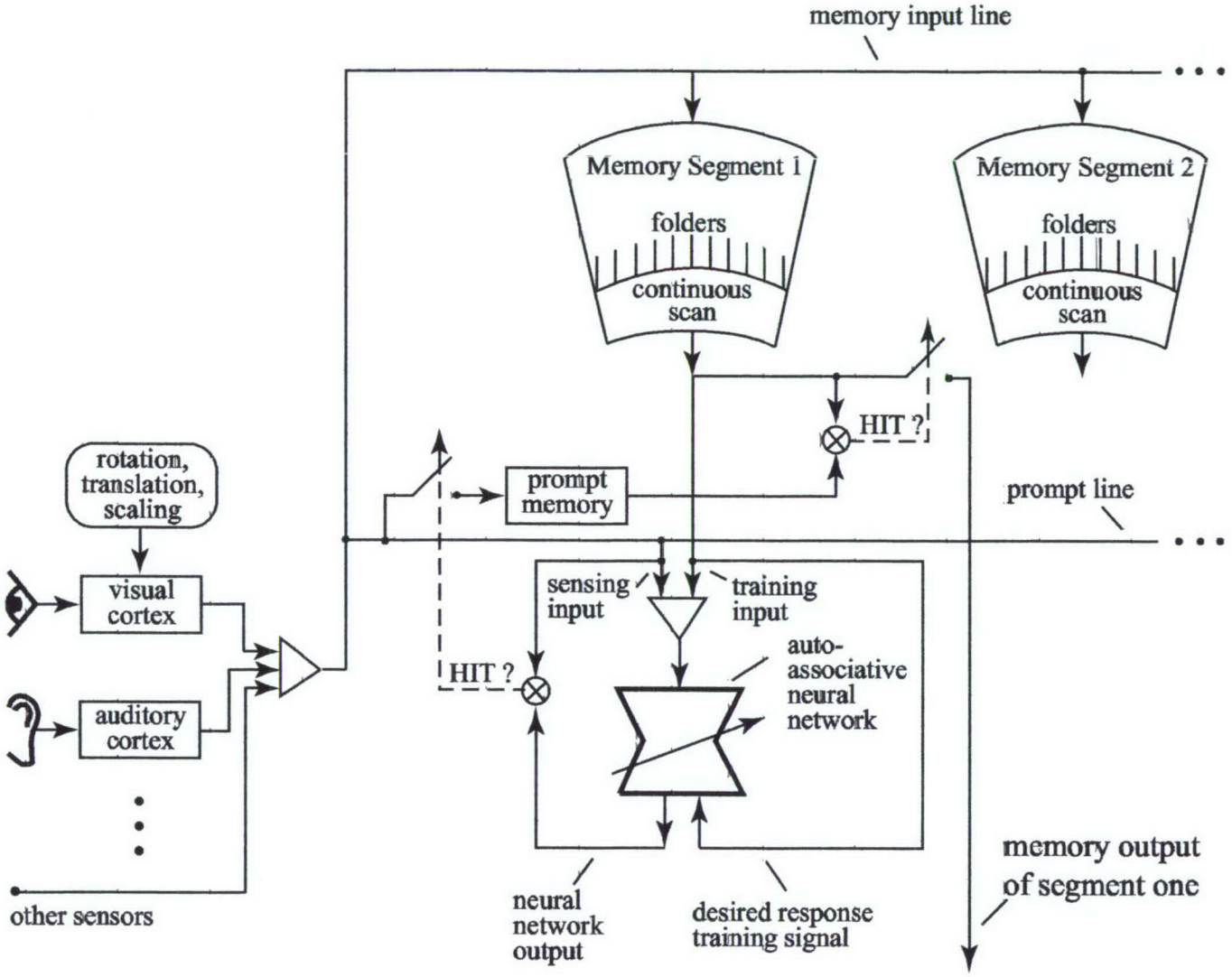


Figure 2: The cognitive memory itself.

find applications for it, and to devise hardware means for its practical implementation

In Figure 2, the memory input line delivers input vectors for permanent storage, looking for empty folders in memory segment 1,2,... etc. Some of the memory folders are large, some small, depending on the amount of storage space needed for the given memory input. Everything sensed, seen, heard, smelled, or felt, etc. is recorded in a folder.

The autoassociative neural networks are trained to produce output patterns that are identical to their input training patterns. During training, their desired output responses are taken from their inputs. After training this way, input patterns that have been trained–in will reproduce themselves at the neural network output, and input patterns that were not trained–in will not reproduce themselves at the neural network output. A trained–in pattern will produce a low error when input and output of the neural network are compared, and a high error results with patterns that were not previously trained–in. When a trained–in pattern is presented to the autoassociative neural network, the error is low and there is a hit. The training patterns for the autoassociative neural networks come from the patterns contained in the folders of the respective memory segments.

Each folder could contain visual, auditory, tactile, etc. patterns that were recorded simultaneously. Each memory segment has many folders. A commutator continuously scans the folder of the memory segment extracting every pattern present and using these patterns to train the connected neural network. During training, these patterns are recycled and trained over and over again, so that the neural network would be able to reliably identify patterns that have been trained in and would be able to separate them from patterns that were not seen before.

A prompt vector from the prompt line can initiate a search among all of the memory segments simultaneously. The prompt signal is actually a sensory input signal in the cognitive memory of Figure 2. The prompt signal goes to the inputs of all the autoassociative neural networks of all the memory segments. The neural networks deliver output vectors that are compared with the prompt vector. If the difference is small, ie the correlation is high, there is a hit, which means that the prompt signal has been seen before and has been previously trained into the neural network that reported a hit. Therefore, the associated memory segment has a folder that contains this pattern vector and other related information. Once there is a hit, a search for this folder takes place. This is a search through all the folders of the memory segment looking for a folder containing a pattern that matches the prompt pattern. Once this folder has been found, its contents are delivered to the memory output line.

Visual prompt signals are translated, rotated, and scaled in trying to make a hit with the autoassociative neural network. Once a hit is made, one has just the right form of the prompt pattern that will match one of the patterns in one of the folders. Responses of an autoassociative neural network are very fast (particularly with parallel implementation).

The neural network allows quick testing of various forms of the prompt signal, rotations, translations, etc., to find the right form for searching the folders.

The neural networks do not undergo training when a prompt signal is present. The prompt signal senses the response of the neural networks. These network are not trained while they are being sensed.

We can speculate about human memory, about the need for sleep, and for different kinds of sleep. We begin by speculating that the cognitive memory system of Figure 1 behaves like human memory. The sensors, the eye, ears, etc. are turned on during wakefulness and are continuously generating prompt signals. Everything seen or heard is being tested to determine possible relationships to things seen before. This requires sensing all of the neural networks. Therefore training does not take place during wakefulness. But training is necessary because new interesting inputs are often observed and need to be recorded in folders and to be used as training patterns. Also, in the human, the neural networks are analog devices and can only maintain their trained responses if they are continually trained or at least go through a training sessions each day. Therefore training is done during each night's sleep. REM (rapid eye movement sleep, dream sleep) is special, and probably does not involve neural network training. This is a time when the brain dreams and fantasizes and the brain, without sensory inputs, prompts itself to recall stored images in random sequences, and REM sleep is probably necessary for creativity. During this time, the neural networks are sensed and not trained. Training therefore goes on during N–REM (non–REM) sleep. Sensing goes on during wakefulness and during REM sleep. This paragraph is speculative. Further discussion with scientists on the biological side will be necessary for confirmation and experimentation.

Returning now to Figure 2, it is clear that the purpose of the autoassociative neural networks is to identify patterns that have been seen before. This seems to be fundamental to memory and thinking.

When a sensory input is received and, being a prompt, causes an autoassociative neural network to indicate that there is a hit, this prompt vector is stored in prompt memory (see Figure 2) as a result of a switch closing because of a hit signal. Now, as the folders are being scanned, when a pattern from one of the folders correlates with the stored prompt, there is another hit and this closes a switch that connects and downloads the contents of the folder to the segment's memory output line. Thus, if a visual, auditory, or other sensory input vector corresponds to a pattern seen before, the contents of the folder containing the original pattern are delivered as memory output. This could be the output that is desired or it could be the input to a cognitive problem solver, as in Figure 1. Prompting the cognitive memory causes it to output information related to the prompt, regardless of where it might be stored in the memory.

The cognitive memory connected to a conventional computer (COTS) with a conven-

tional memory makes what seems to be a powerful combination. The question is, what kind of problems can be solved by making use of this combination? It is too early to exhaustively categorize these problems and impossible to see what they will be, at this time. From limited experience, it seems that the range of problems to make use of the cognitive memory will be very broad indeed. Several applications have been worked on, and the results follow.

Aircraft Location Based on Ground Imaging

A candidate DOD application for cognitive memory is that of location and tracking of aircraft from optical imaging of the ground below. One might raise a question about this: What does cognitive memory have to do with location? This is a good question, but one should realize that a person waking up in the morning would not know how to go from the bedroom to the kitchen without memory. A person walking on a path through the woods would not know how to find the way home without memory. In one's own home town, a person would not know how to drive from point A to point B without memory. So memory of visual cues is critical to being able to navigate in every day life. This should be kept in mind as we explore this candidate DOD application.

Figure 3 shows a portion of Manhattan island in New York. This is an aerial photo taken some years ago.The photo was obtained from a web site. Superposed on the photo is a grid of squares. Each square is 81x81 pixels. Within each square is a picture of a small portion of Manhattan. Using Matlab, a 25x25 pixel picture was made from each 81x81 pixel picture. Matlab does this by interpolation and re-sampling.

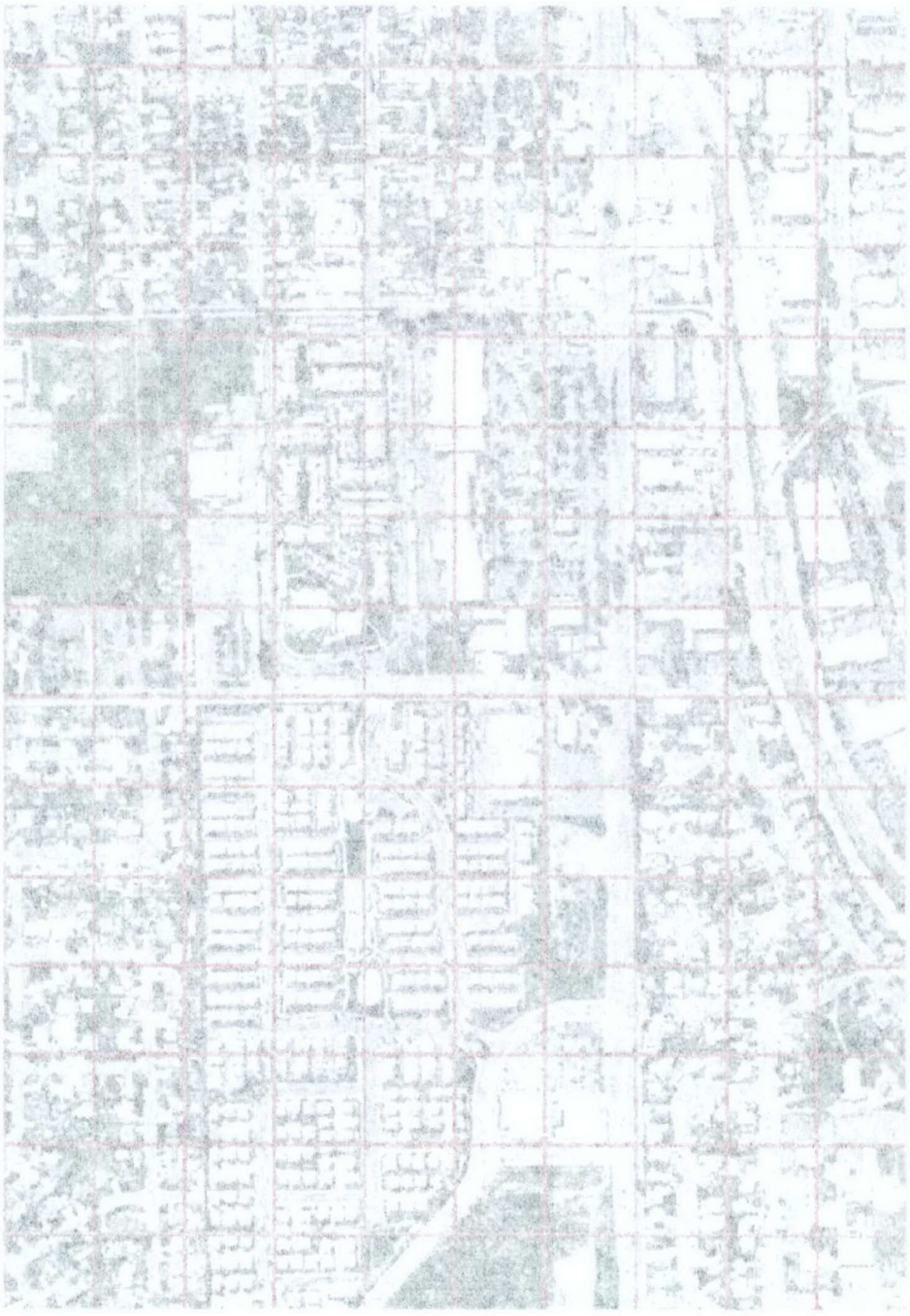
Each 25x25 pixel picture was then recorded in a folder in memory segment 1 of the cognitive memory of Figure 2. In each folder, the 25x25 pixel picture was recorded along with a piece of additional information, the x,y location of the pixel in the center of the corresponding 81x81 pixel picture. Distances along the x and y directions were recorded not in meters, but in pixel count.

In Figure 3, twenty five small pictures of 81x81 pixels are shown. The entire Manhattan area had 900 small pictures. All 900 pictures were reduced to 25x25 pixels and were trained into the autoassociative neural network.

In Figure 4, an aerial photo of a portion of Simi Valley, CA is shown. This was obtained from a web site. Superposed is a grid of squares, each square being 81x81 pixels. This picture has 2700x2200 pixels. The number of small pictures shown in the photo is 160. Covering the rest of Simi Valley, there were a total of 918 small pictures. Using Matlab, a 25x25 pixel version of each 81x81 picture was made by interpolation and re-sampling. The nine hundred and eighteen 25x25 pixel small pictures were recorded in individual folders in memory segment 2. In each folder, the position of the center pixel of the 81x81 pixel small picture was also recorded. The 918 pictures were trained into the

Figure 3: Scanning aerial photo of Manhattan, NY, (Color photo)

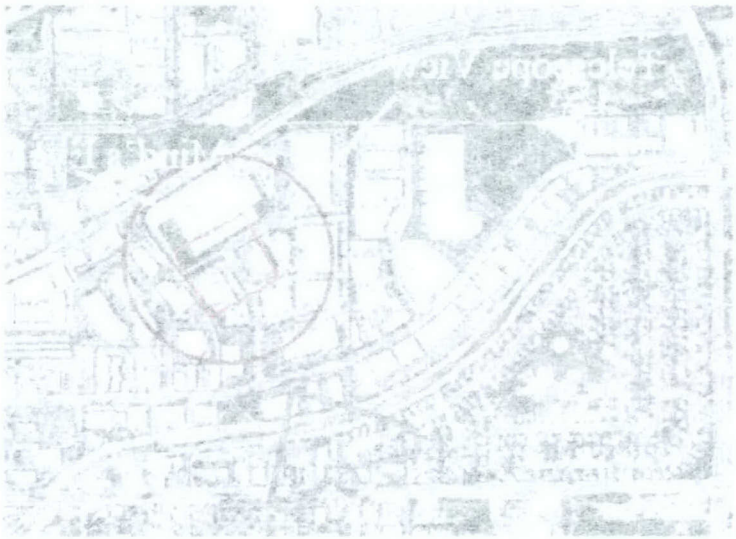
Figure 4: Scanning aerial photo of Simi Valley, CA, (Color photo)



a) Original Aerial Photograph

(b) Actual Groundscape Under the Airplane

Figure 5: Shooting the ground to find airplane's position, (Color photo)



autoassociative neural network of memory segment 2.

The autoassociative neural network of memory segment 1 and 2 both had 250 neurons in their first layer. Since the input patterns had 25x25 pixels, each neuron had 25x25=625 weights. For the first layer, there were 625x250=156,250 weights. The second layer had 420 neurons, with 250x420=105,000 weights. The third layer had 625 outputs to correspond with the 625 inputs, so the third layer had 625 neurons, each having 420 weights. Therefore, the third layer had 420x625=262,500 weights. The total number of neurons in the network was 1,295, and the total number of weights was 523,750. Implemented on a Sun Blade 2000 workstation, the training time for the Manhattan patterns was 5.6 hrs., and the training time for the Simi Valley patterns also was 5.6 hrs.

The backpropagation algorithm was used with Matlab to train the autoassociative networks. By careful programming, the training time was reduced, but by no means optimized. A next step will be to do the neural network with C- code, and that should result in a 20 times speed–up. With really large problems, this would be implemented with special hardware designed to do backpropagation neural networks. The speed–up would be many orders of magnitude.

More important in most cases would be the sensing time not the training time. Training would generally be done off–line. Sensing the autoassociative networks, ie applying an input pattern and seeing if the network output does or does not match the input, would be done very frequently, and generally in real time. Sensing time for the above network was 83 milliseconds. This could be greatly speeded–up by implementation with special hardware.

Returning next to Figure 5, we can see the original aerial photo of Simi Valley in Figure 5(a). In Figure 5(b), we see a blow–up of a portion of Figure 5(a). We see the image of an airplane symbolizing the direction and path of flight.

The scenario that we present goes like this. Some time after an aerial photo was taken, an airplane travels over the region carrying a telescope. By looking at the ground through the telescope and comparing the image to the original aerial photo image, it should be possible to locate the position of the flying airplane relative to the aerial photo and to determine its heading direction. This has been done using the cognitive memory. The process is like using a map to find your way. The map here is the original aerial photo.

The telescopic picture indicated within the red circle in Figure 5(b) is supposed to be the same as would be obtained from a real telescope in a real airplane. We used the aerial photo instead of flying a telescope–mounted airplane over Simi Valley. The telescopic image within the red circle has a center that represents the exact location of the airplane when shooting an image through the telescope. This is in effect a simulation of an airplane shooting pictures of the ground through a telescope that is mounted to look straight down when the airplane is in level flight. The red square within the circle

is oriented in the direction of the flight path, in the direction from noise to tail of the airplane. When shooting pictures with a real airplane, it is assumed, in accord with this model, that the airplane would be in level flight at the moment that each picture is shot. If not, the pitch, yaw, and roll would need to be measured and recorded and taken into account. In any event, for the simulated airplane flight, we assume that the airplane flies with level flight.

The red square within the telescopic circle bounds an image which we call the "mind's eye". This image will be fed as a prompt to the autoassociative neural network. The size of this square corresponds exactly to that of the grid square of Figure 4. For the moment, assume that this is the case. For this to be so however, one would need to know the altitude of the airplane that took the original photo of Simi Valley, and how big the squares were in Figure 4 In reality one would not know these details, but if the correspondence were not exact, image scaling would be necessary, and this could be done. Assume that scaling is not necessary for the present experiment.

When the telescopic picture is shot, the telescopic image is feed to an onboard computer. The mind's eye can be moved with software control. This would be the function of the "visual cortex" in Figure 2. The mind's eye could be translated left–right, up–down, and rotated relative to the telescopic image. The objective is to make a hit, ie to get coincidence between the image in the mind's eye and just the right small picture from one of the 918 small pictures of Simi Valley. These pictures have been trained into the autoassociative neural network, and when the mind's eye is adjusted with the right position and angle to get an image like the right original small picture, then there is a hit.

Once there is a hit, the mind's eye image is stored in the prompt memory, and a search by scanning through the folders is made to find a matching image. When the correct folder is found, the x,y pixel coordinates of the center of the small picture is retrieved. The position of the airplane at the moment of shooting the telescopic picture is then determined by the amount of left–right, up–down translation that was made to the x,y pixel location of the relevant small picture. The heading of the airplane relative to the original aerial photo can be determined from the amount of rotation that was also needed to make the hit.

This procedure was performed using the aerial photo of Manhattan. The simulated airplane flight was a straight line path and telescopic photos were assumed to be shot uniformly in distance over Manhattan. For each photo, a hit was obtained by rotating and translating the mind's eye. From the coordinates of each small picture that was hit and correcting for the number of pixels of translation and the number of degrees of rotation required for each hit, the location of the airplane for each telescopic shot was calculated and plotted with red crosses on the original aerial photo, shown in Figure 6.

The heading of the airplane at the time of each shot is indicated by yellow arrows. The arrows, determined independently, point along a straight that turns out to be the

Figure 6: Straight–line airplane track over Manhattan, (Color photo)

Figure 7: Circular airplane track over Simi Valley, CA, (Color photo)

exact airplane path. The red crosses are at the exact locations of the airplane at the shot times. The result is that the position and the heading of an airplane relative to a previously taken aerial photo can be determined down to the very pixel in x and y, and the heading determined to within about one degree. From the aerial photo in Figure 3, one can see that a city block is about 20x20 pixels. Position along each coordinate can therefore be located to within a distance of one twentieth of a city block with this aerial photo.

If the size of the mind's eye square did not match those of the original grid in Figure 3, then scaling would be necessary in addition to translation and rotation, in order to make a hit. The autoassociative neural network would need to be able to respond rapidly in sense mode to a prompt input. Zooming in and out in addition to translating and rotating would greatly increase the number of prompt images that would be imputed to the neural network in order to make a hit. A fast sensing response would make this feasible in real–time applications.

All of the hits were done independently for the airplane track shown in Figure 6. These hits were perfect in this case. If there were places on the ground where the texture was not strong enough to make accurate hits, location points would be missed. One could use larger small pictures in order to encompass a bigger area and to thereby gain some texture in overall image.

The size of the small pictures and the size of the mind's eye have not been optimized. Nor has the design of the autoassociative neural network in terms of number of neurons, number of weights, and number of layers. There is a wide range for these parameters that would give excellent results. This is a subject under study.

Figure 7 shows the track of an airplane flying in a circle over Simi Valley. The same method was used to determine the position and heading of the airplane at the time of each hit. The positions are indicated by blue crosses, and the airplane headings are indicated by the red arrows. These arrows turned out to be tangent to the circular path and uniformly spaced along the arc. Each location and heading was precisely the true position and heading at the shot times.

By observing the sizes of houses, industrial buildings, roads, cul–de–sacs, and highways in Figure 4, the spacing from pixel to pixel can be estimated to be approximately ten feet. Location accuracy to within a position of a pixel corresponds to accuracy within about 10 feet. With higher resolution photography, even greater accuracy could be achieved. This kind of accuracy is similar to that of GPS.

GPS locates things relative to absolute latitude and longitude on the earth surface. The cognitive memory locates things relative to a previously taken aerial photo. By using the cognitive memory, targets could be located on the aerial photo, and an airplane could

find the targets with pinpoint accuracy by telescoping the ground. Moveable as well as stationary targets could be found wherever they are.

Figure 8 pertains to one of the hits along the Simi Valley circular path. It shows how position can be located at the exact pixel. Scanning the mind's eye in left–right, up–down translation one pixel at a time, and scanning in angle with 10 degree increments around all 360 degrees and subsequently in 1 degree increments over a small angular range when getting close to optimal, is being done. The plots in Figure 8 show percent mean square error (MSE) as the mind's eye position is adjusted. Mean square error is the sum of squares of the pixel differences between the input and output patterns of the autoassociative neural network. A percent MSE below a set threshold indicates a hit. Figure 8(c) shows percent MSE versus mind's eye rotation and translation along the x–direction with the y–position optimized. Notice the deep sharp dip in MSE. This indicates a hit. With the y–position set to one pixel less than optimum in Figure 8(b), there is still a dip when angle and x–position are varied. The dip is not as deep, and does not indicate a hit, but does indicate proximity to a hit. With the y–position two pixels less than optimum in Figure 8(a), there is still a small dip, but not very pronounced. Figures 8(d) and (e) show this effect for y–positions one and two pixels greater than optimum. Blow–ups of these figures follow.

Figure 9 shows the dip effect in a different way. With the y–position optimized, curves are plotted for percent MSE versus rotation angle for various x–positions. The deepest dip, indicating a hit, corresponds to the x–position optimized. The other curves correspond to the x–position one to five pixels away from optimum, in both directions. From these curves one can see that it is possible to pinpoint position at the exact pixel, and to sense direction within one degree.

Satellite Surveillance of Diego Garcia Island

Diego Garcia is a British–owned island in the Indian Ocean. On this island is a U.S. Air Force base used primarily for B52 and B2 bombers and KC135 tanker planes. A satellite photo of Diego Garcia was obtained from a web site, and it shows aircraft parked in a tarmac area adjacent to the principal runaway of the island. The objective of this experiment is to locate and identify the parked aircraft, making use of the cognitive memory of Figure 2. Being able to do this would allow constant automatic satellite surveillance of Diego Garcia, and of any other airfield in the world.

Figure 10 is a satellite photo of a major portion of Diego Garcia Island. Zooming in, the aircraft parked near the main runway are clearly visible in Figure 11. The smaller 4–engine aircraft are believed to be KC135 tankers. The larger aircraft, each with 4 engine pods, are believed to be B52's. The B52 actually has 8 engines, 2 engines per pod. That much detail cannot be seen in the photos. Looking at the upper row of aircraft, counting from left to right, there are two KC135's, then a space, then two B52's, then what looks like a big grease spot on the tarmac but is believed to be a B2 stealth bomber, then two

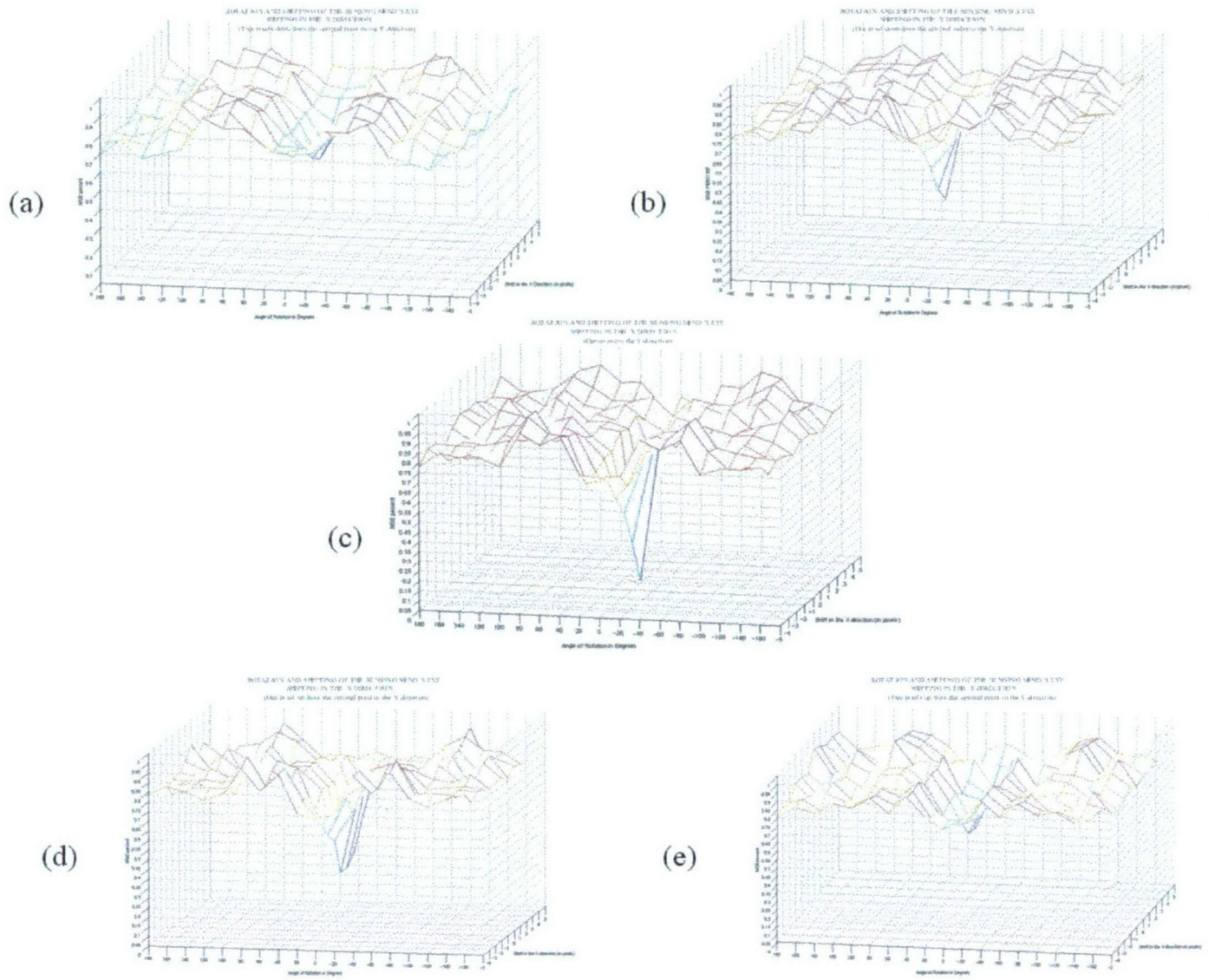


Figure 8: Rotation and translation of "Mind's Eye", (Color drawings)

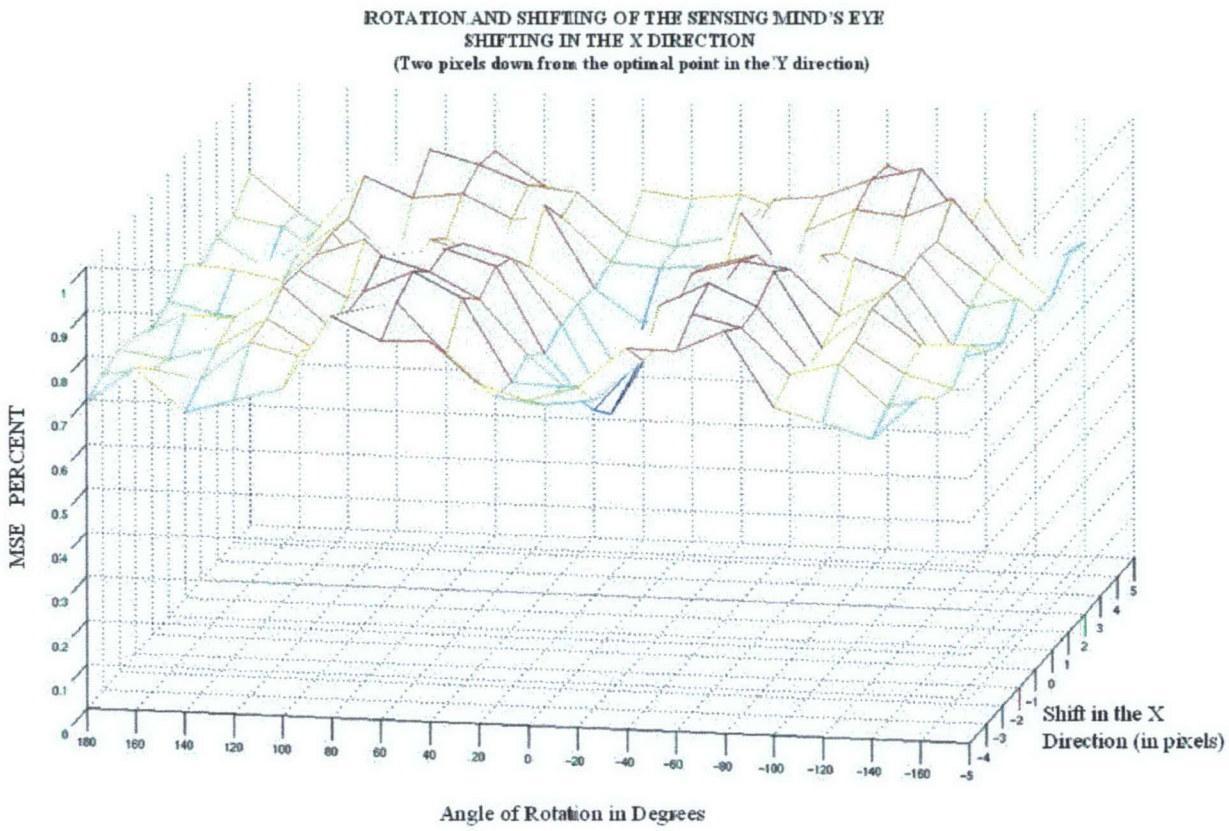


Figure 8: (a), (Color drawing)

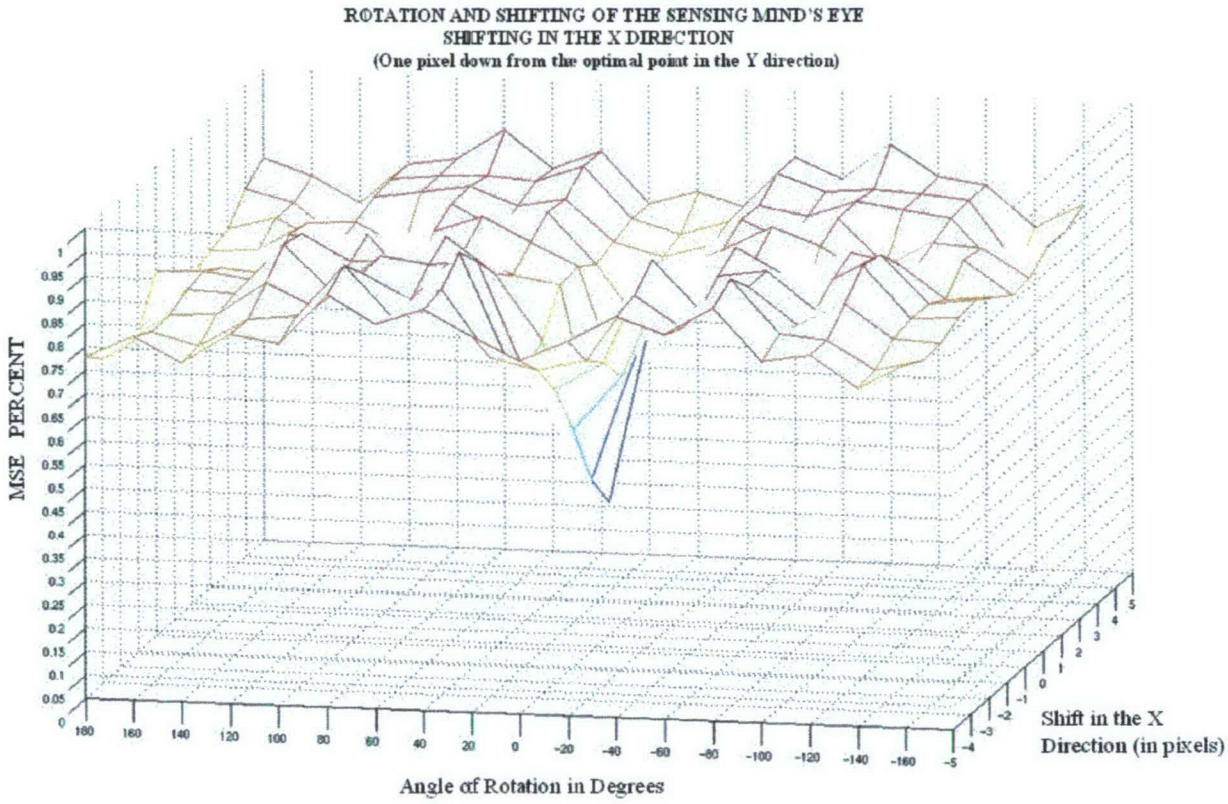


Figure 8: (b), (Color drawing)

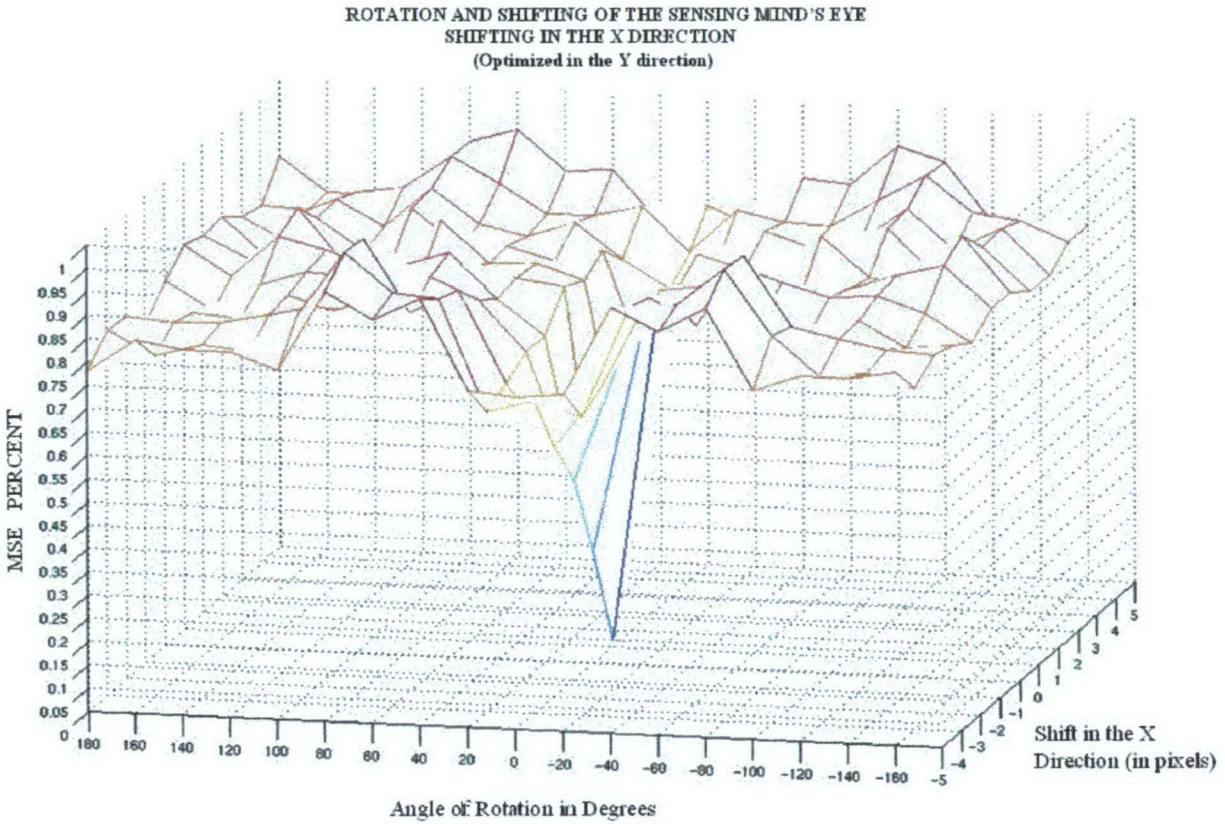


Figure 8: (c), (Color drawing)

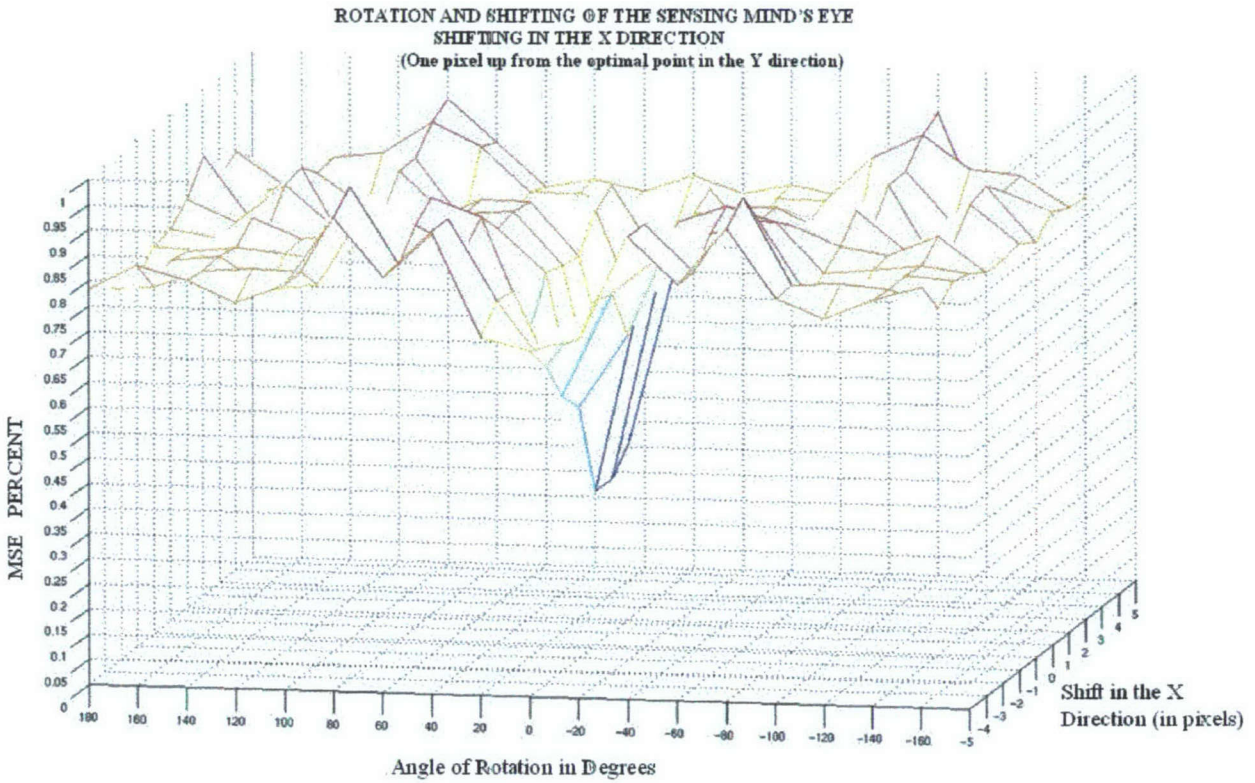


Figure 8: (d), (Color drawing)

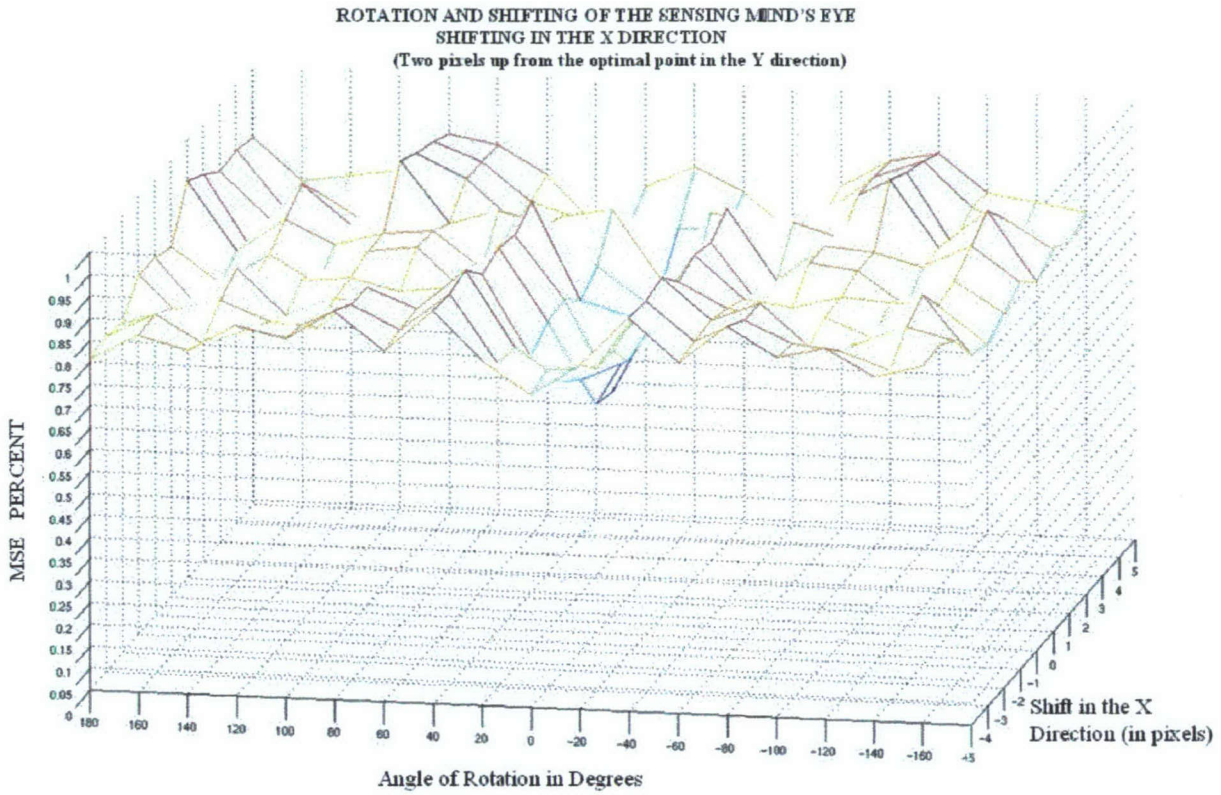


Figure 8: (e), (Color drawing)

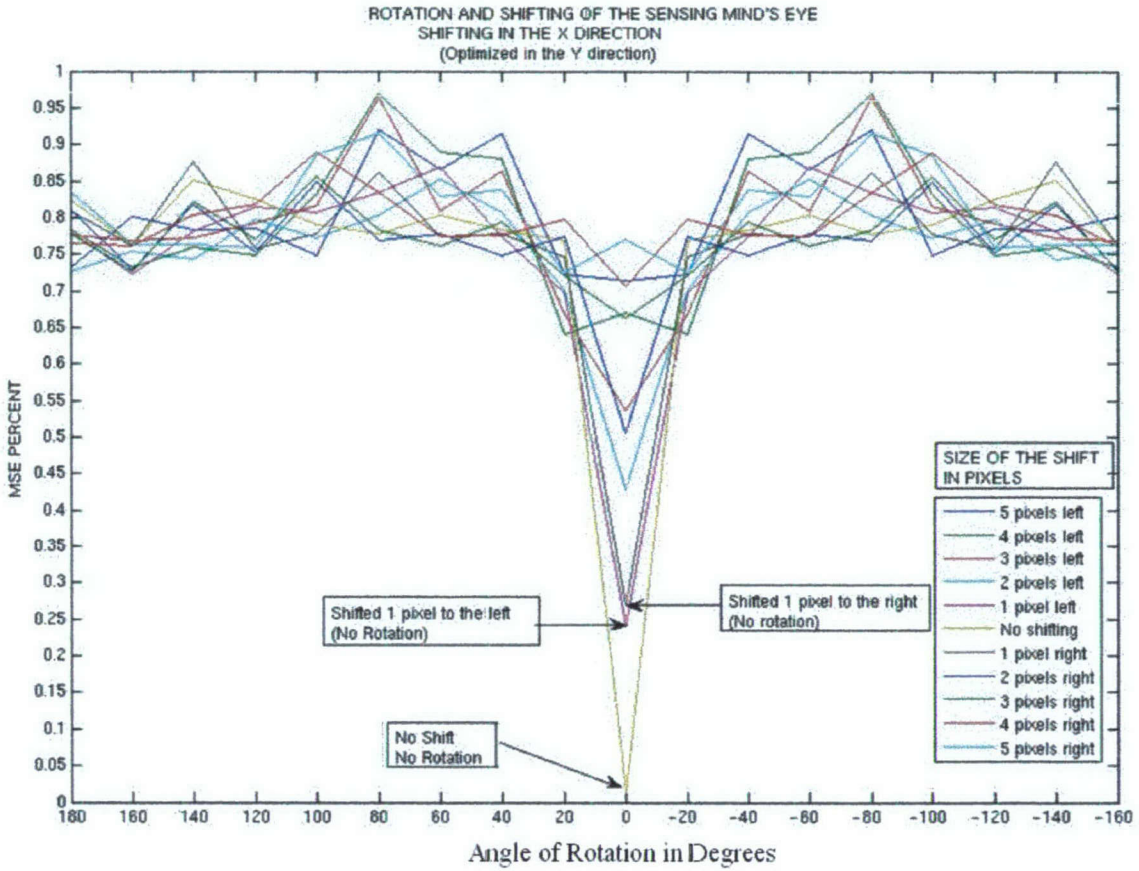


Figure 9: Rotation and translation in the x–direction with y optimized, (Color drawing)

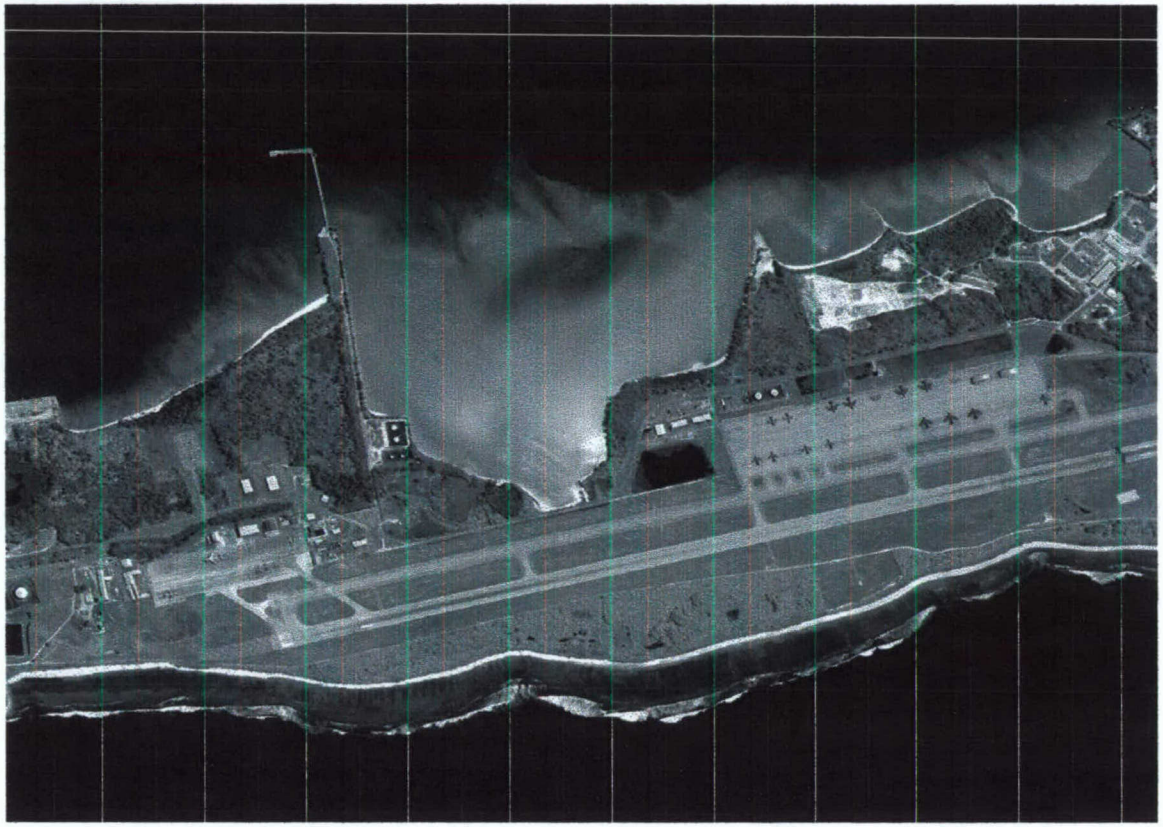


Figure 10: Satellite photo of Diego Garcia Island showing U.S. Air Force base

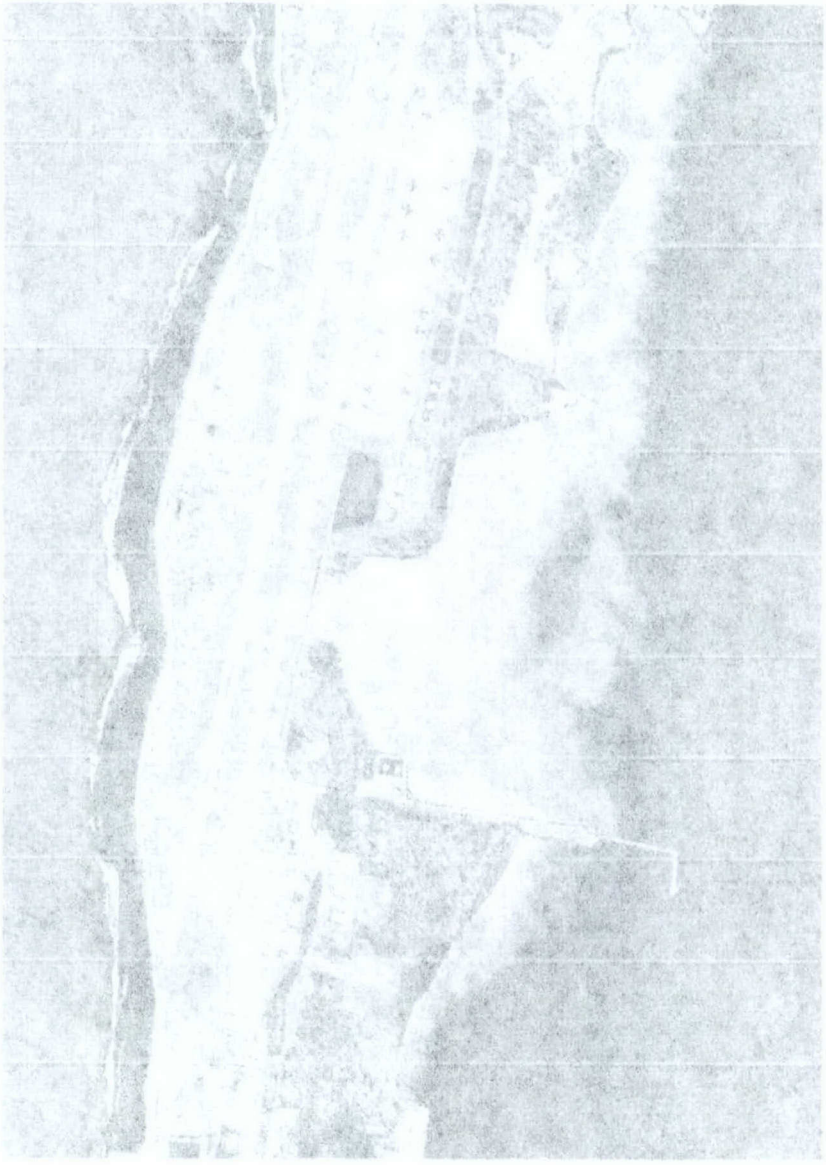
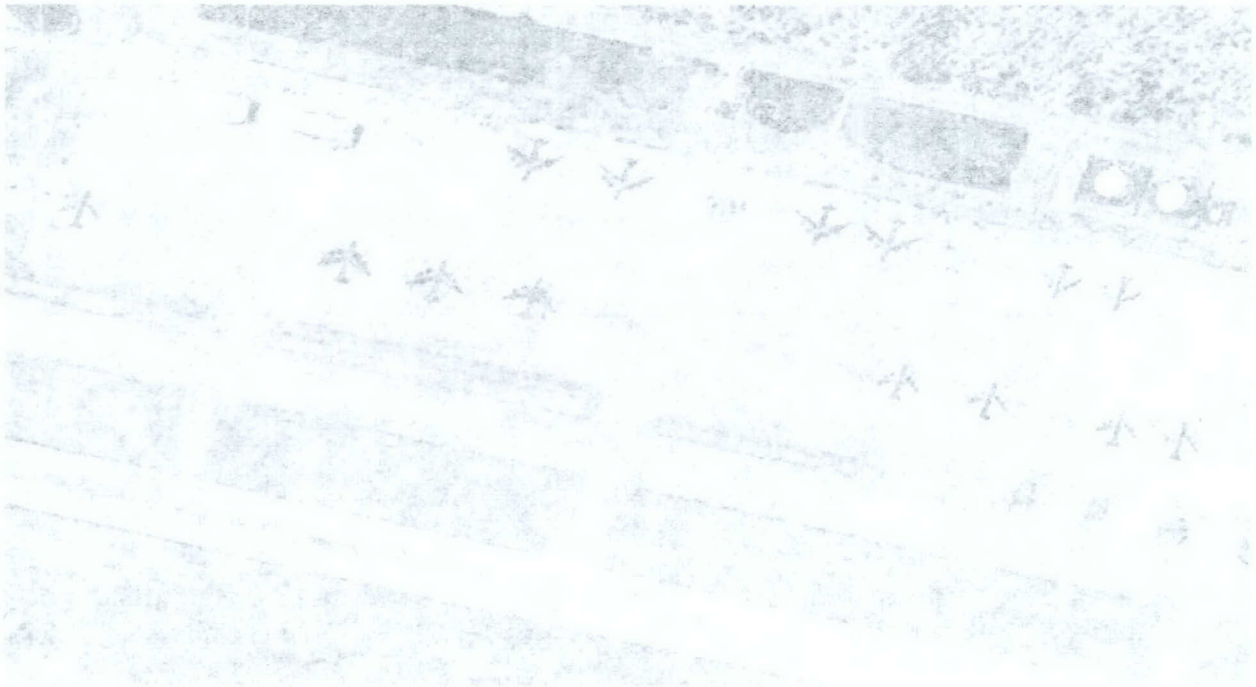


Figure 11: Aircraft parked in an area near the main runway



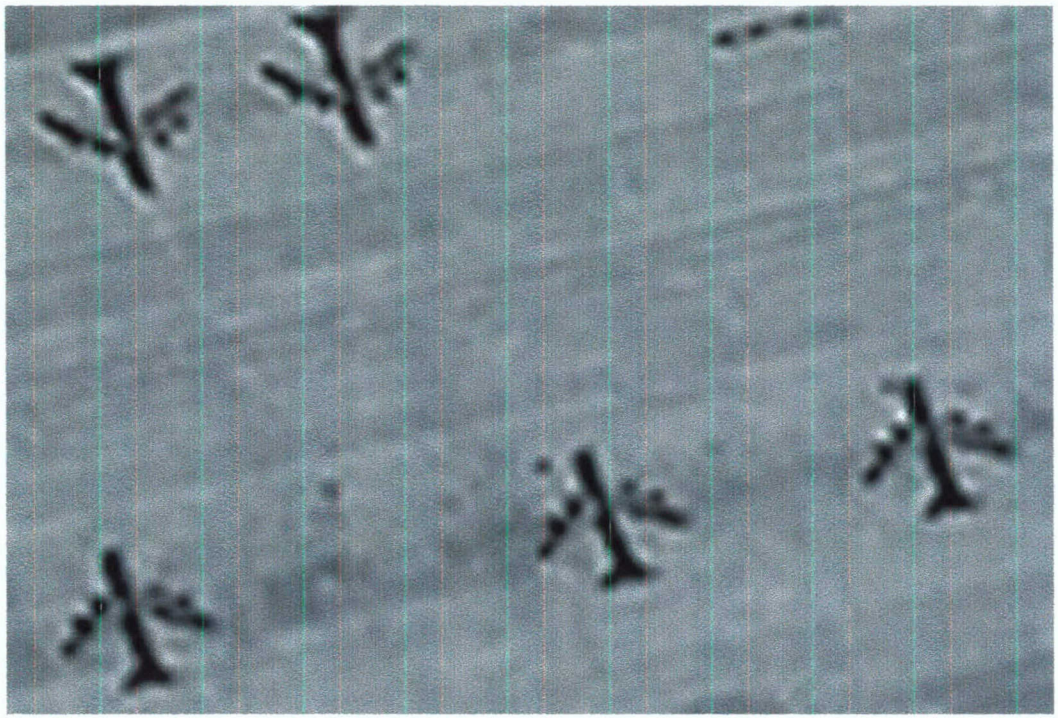


Figure 12: Zoomed–in photos of five KC135 tankers

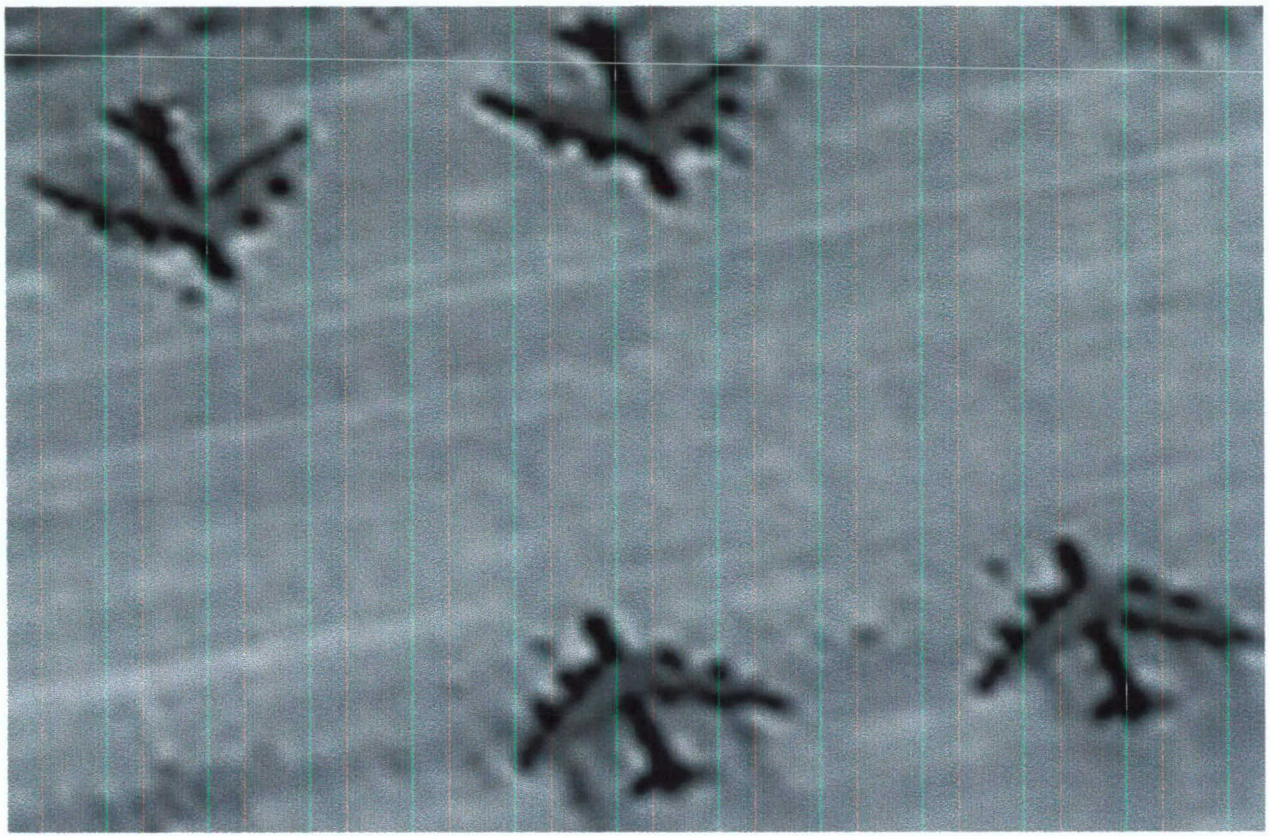


Figure 13: Zoomed–in photos of four B52 bombers

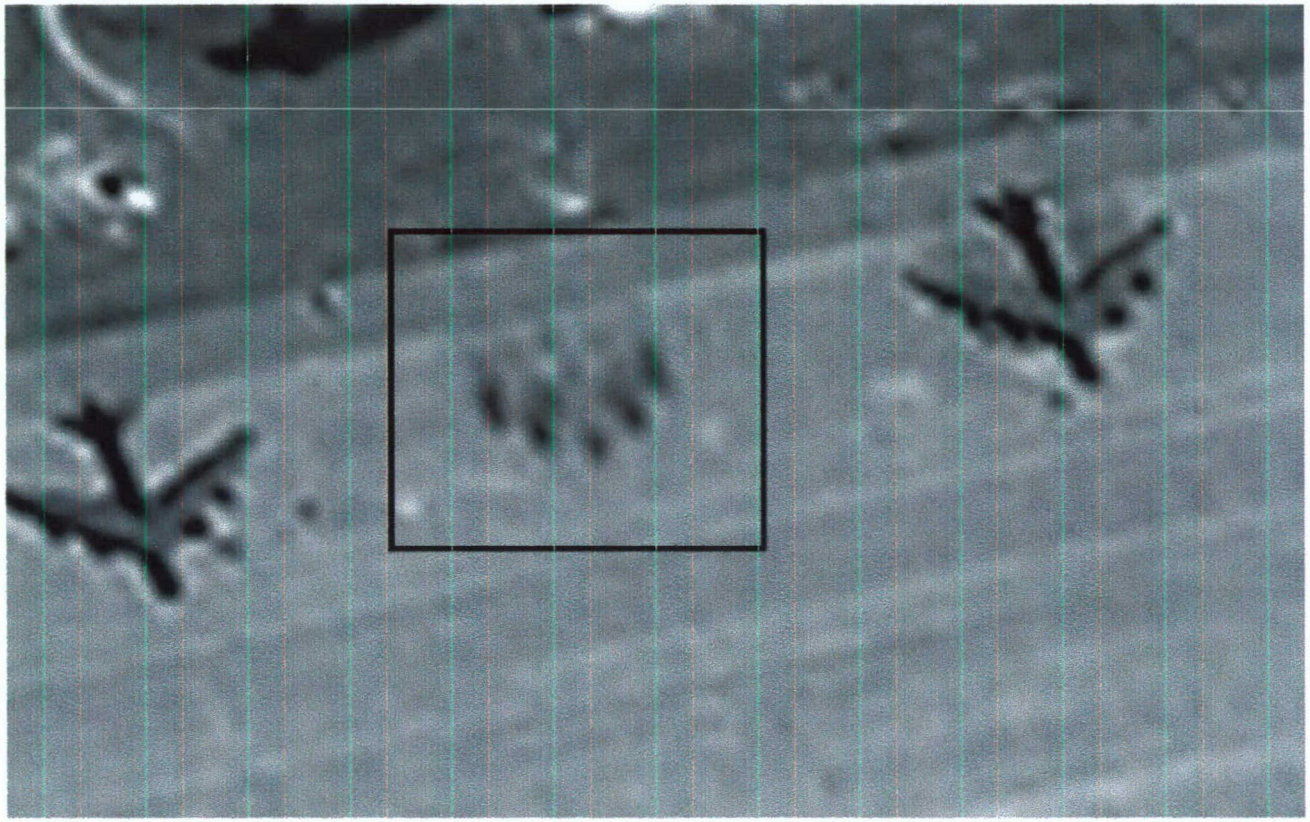


Figure 14: Zoomed–in photo of B2 stealth bomber and neighboring B52's

more B52's, then another big grease spot that is probably another B2, and finally two buildings that are believed to be B2 shelters. Now looking at the lower row of aircraft, from left to right, there are two KC135's, then a space, then two more KC135's, then a big space, then three B52's, then a big space, and finally a KC135.

A zoomed–in picture of five KC135's is seen in Figure 12. Two are from the upper row, and three are from the lower row. The upper row and lower row airplanes were parked in opposite directions, and the shadows and perspectives are opposite for these planes. The image of an upper row KC135 is not exactly a rotated version of the image of a lower row KC135. They are somewhat different images. The same effects can be seen regarding shadow and perspective for the B52's. From Figure 13 four zoomed–in images of B52's can also be seen. Note the dark shadows from the wings and fuselages.

Two of the B52 images from Figure 13 were selected, one from the upper row and one from the lower row. These were treated as two separate images because of the differences in perspective and shadow. More images were created by translating these images left–right and up–down, so that their centers were moved over a (5pixel x 5pixel) grid. The result was 25x2=50 different B52 patterns. Each of these patterns was recorded in a folder of memory segment 1. In addition, each folder had the identification of the aircraft, ie., B52, and its x,y location in pixels. The airplane images in the folder were trained into the autoassociative neural network.

Images from the satellite pictures of Figure 12 taken within the mind's eye (a square area somewhat bigger than the B52 image) acted as prompts. The mind's eye was programmed to scan the entire satellite photo to look for hits. In each scanned position, the mind's eye was rotated in 5 degree increments, looking for a hit, looking to minimize the mean square error of the autoassociative neural network. When a hit took place, the memory segment was searched for correspondence with the mind's eye image that caused the hit, the folder was found, and the identification of the aircraft and its x,y location was downloaded from the folder.

A hit corresponded to the MSE of the autoassociative neural network being very low. As the mind's eye scanned and came upon the image of a B52, it scanned about this image up and down, left and right, making many hits because the B52 patterns were trained with 25 different positions. For each position of the mind's eye, many rotations with 5 degree increments were tried to find the best fit. Getting many hits and identifying them as B52, the computer screen showed "B52 Bomber." When the mind's eye scanned over a KC135, various positions and rotations gave low MSE, but not really low, as when encountering a B52 image. Then the computer screen indicated "unidentified aircraft." The detection threshold can be pre–set. The detection criterion can be relaxed or tightened.

The structure of the autoassociative neural network was identical to that of the network in the cognitive memory used to track aircraft positions over Manhattan and Simi

Valley, the same number of layers, neurons per layer, and weights per neuron.

A video of the searching for B52's has been made and is part of this report in electronic form. The video has been recorded on a CD disc that is included with the hard copy version of the report.

Another experiment was made using upper row and lower row images of B52's and KC135's and the image of the B2 located on the upper row, between the B52's. The B52 and KC135 images were taken from Figures 12 and 13. The B2 image was taken from Figure 14. Each image was translated over a (5pixel x 5pixel) grid. The total number of images was 50 B52's, 50 KC135's, and 25 B2's. These were all trained into the cognitive memory.

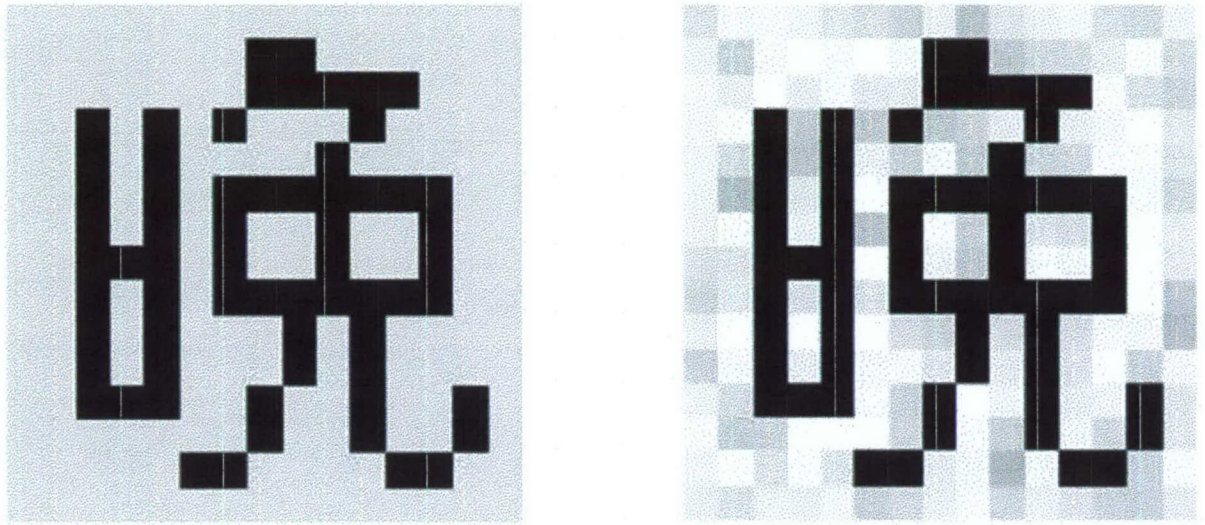
The aerial photo of Figure 11 was again scanned by mind's eye, with pictures taken with 5 degree rotational increments. With this experiment, all of the B52's, the KC135's, and the B2's were located and identified. Whenever a hit was made, the computer screen indicated the type of aircraft and its location, x–pixel, y–pixel. A video of this search and detection was made and is included on the CD as part of this report.

The cognitive memory could have been trained to locate and identify many other types of objects on the island in addition to the aircraft, scanning over the entire aerial photo. Once the cognitive memory has been trained to seek out and identify the types of objects of interest, scanning and detections can be made at very high speed with parallel hardware implementation.

Reading Chinese Characters

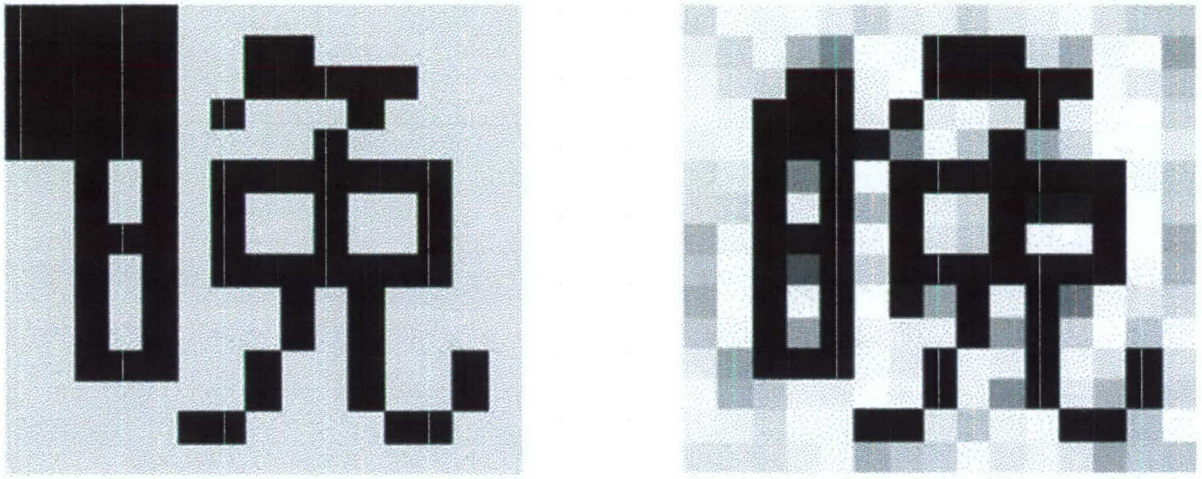
Experiments in reading Chinese characters were made. We had available a data base of 20,000 Chinese characters. Ten thousand of these were trained into the autoassociative neural network, and when testing the trained network, the trained patterns were all correctly identified as having been trained–in, and 97% of the non–trained patterns were correctly identified as not having been trained–in. 3% of the non–trained patterns gave false–alarm indications that they were trained–in. One of the problems is that there are close similarities between certain of the Chinese characters, and especially between major portions of Chinese characters. Although the results have been good, we are working to improve the score.

Further experiments were made with a group of 1000 randomly selected Chinese characters. They were trained into a cognitive memory. Figure 15(a) shows one of these characters. Figure 15(b) shows the output of the autoassociative neural network. Ideally, the input pattern should be perfectly reproduced at the neural network output. In this case, the output was close to perfect, but there was an MSE of 0.25%. This error would have been lower if the network had been trained longer. Certainly, the Chinese character



(a) (b)

Figure 15: A Chinese character applied to a trained autoassociative neural network
(a) The original character, (b) Output of the neural network



(a) (b)

Figure 16: A Chinese character partially obscured, applied to the trained autoassociative neural network (a) The partially obocured character, (b) Output of the neural network (26.82% MSE)

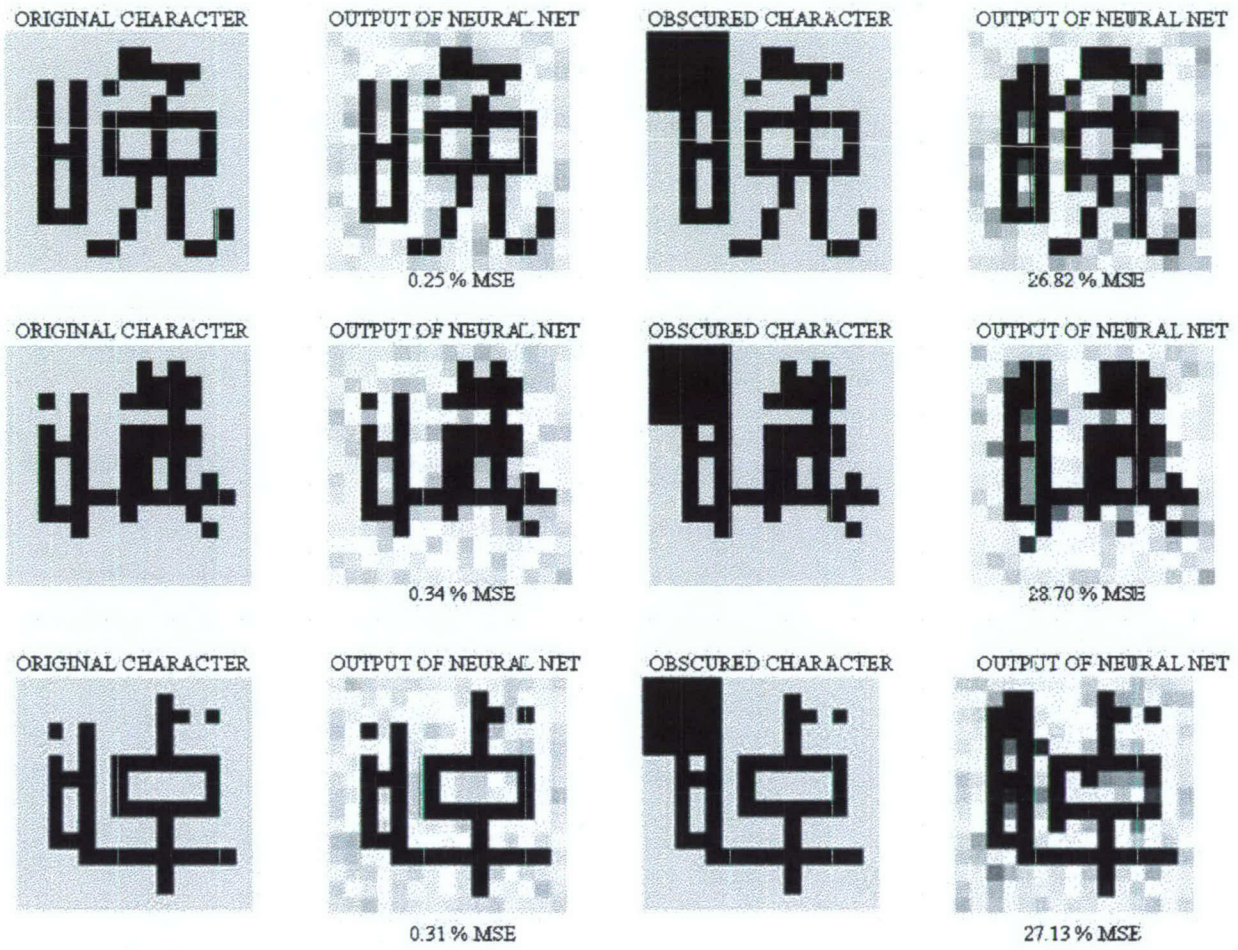


Figure 17: Chinese characters applied to the trained Autoassociative Neural Network

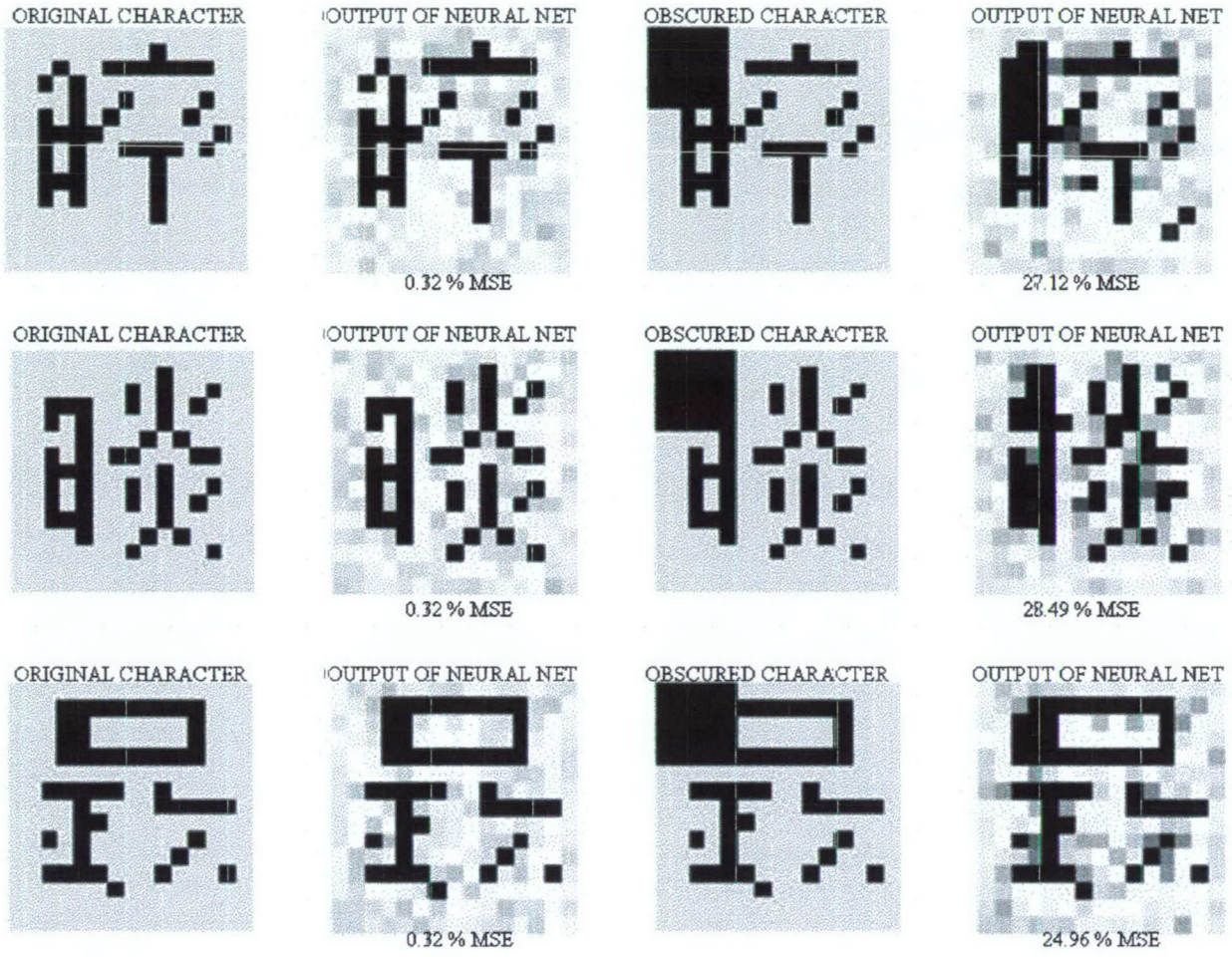


Figure 18: More Chinese characters applied to the trained Neural Network

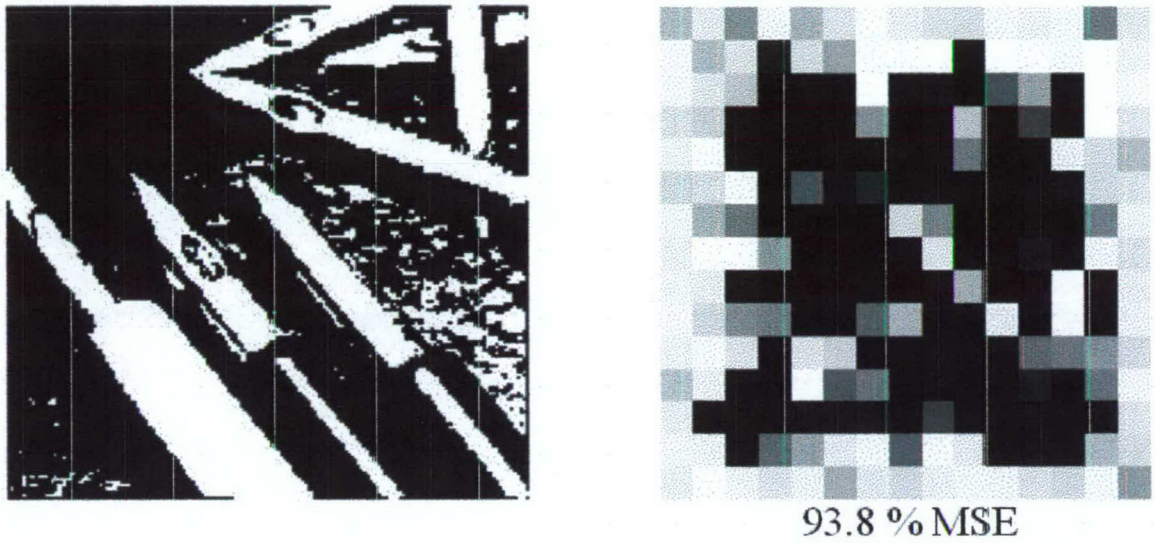


Figure 19: Non-trained pattern applied to the trained Autoassociative Neural Network, (a) Input Pattern, (b) Output of Neural Network

of Figure 15(a) was recognized and identified.

The Chinese character of Figure 15(a) was then partially obscured, as shown in Figure 16(a). This obscured pattern was then inputted to the already–trained neural network, and the output, shown in Figure 16(b), does not perfectly match the original input of Figure 15(a). This is to be expected. There is an MSE of 26.82%. In spite of the error, the pattern of Figure 16(b) contains much of the original information. One could guess which character this is. The distorted character could probably be recognized, but one would need to accept a high MSE when calling this a hit.

All of the trained Chinese characters gave almost perfect responses at the autoassociative neural network output. The partially obscured patterns gave surprisingly good outputs from the neural network, as can be seen from the examples of Figures 17 and 18.

The implications of this experiment are that it would be possible to read Chinese printing even if the printing were not perfect. Also, if a cognitive memory were used to find aircraft, tanks, etc. in aerial or satellite photos and these objects were partially obscured, this may also be possible. This is a question being studied.

Figure 19(a) shows a non–trained–in pattern that was inputted to the neural network that was trained on the Chinese characters, and Figure 19(b) shows the neural network output. The output does not match the input in any way, indicating that the pattern in Figure 19(a) was not trained–in. This is a check on the autoassociative network. It worked the way it should have.

Recognition of Human Faces

Another possible application for the cognitive memory is that of face recognition. A capability for doing this would be very valuable for military security and for homeland defence. On the civilian side, thousands of surveillance cameras all over the U.S. are producing continual video output data, and a means for automatic evaluation and analysis would make these cameras much more useful. Very high speed image analysis would be needed for this type of work, and a means for doing this with parallel processing is described below.

We begin with a very large number of photos containing people's faces stored in folders of the cognitive memory. Each folder contains the face and the name of the person. The prompting photo is presented, and the question is, what is the name of the person in the photo? We need to associate the photo with photos of the same person that may be stored in the folders. The associations are not easy to make because the size of the facial images are not the same, the orientation and position of the faces are not the same, and the brightness or luminance of the faces are not the same. Nevertheless, we shall make these associations and connect the unknown prompting pattern to one or more of

Figure 20: Three photos of Bernard Widrow used for training

the known patterns stored in the folders. So do this, it would be necessary to translate, rotate, scale, and adjust the luminance of the patterns.

The images of all the faces stored in the folders were trained into the autoassociative neural network. This is the training data. The trained neural network was sensed by applying to it the facial image in the prompting pattern. By adjusting the sensing pattern with translation, rotation, scaling, and luminance, a hit was being sought. If the error, the difference between the input and output patterns of the autoassociative network, was below a threshold, then a hit was established. The folders were then searched for the exact pattern that caused the hit. Once the folder or folders were found that corresponded to the hit pattern, then the identification of the unknown prompt pattern could be made.

Figure 21: A photo of Juan Carlos Aragon, Victor Eliashberg and Bernard Widrow used for sensing

Figure 20 shows three photos of Bernard Widrow. All three were used to train the autoassociative network. From each photo, many new ones were created by rotation, translation, and scaling. There were seven different rotations, two degrees apart. There were twenty five different left/right, up/down positions, one pixel apart, in a square array. There were three different sizes or scaled versions of each picture. For each of the original three pictures, there were now 525 pictures. Therefore the total number of training patterns was 1,575 pictures. All of these patterns were stored in the folders and labeled "Bernard Widrow". Training time was approximately 26 hours to bring the mean square error to 0.25% or less for all the training patterns.

Figure 21 is a photo of Juan Carlos Aragon, Victor Eliashberg and Bernard Widrow. This photo was the prompting image causing memory recall. This photo was scanned with small moving window in an attempt to find an image that matched one of the 1,575 training patterns. Sensing the prompting image to find a hit was done with six different window sizes, allowing scaling of the prompt image. The sensing window was translated with twenty five different left/right, up/down positions, two pixels apart in a square array, allowing accurate alignment. The image intensity within the window was varied with six

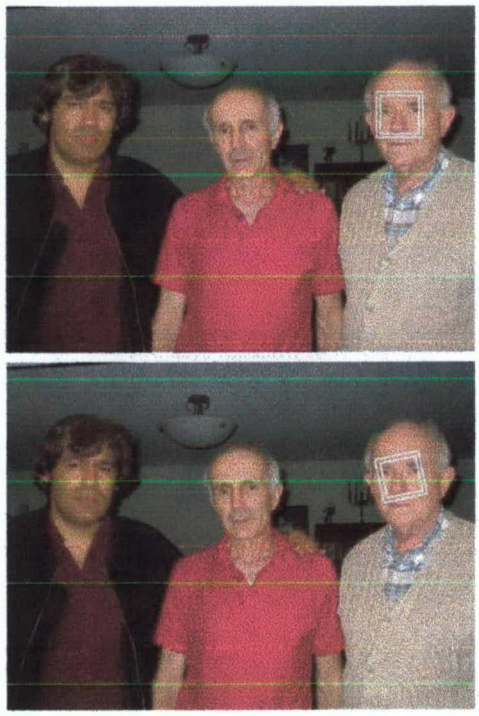


Figure 22: Sensing patterns obtained from Widrow's face with two window sizes a) straight up, and b) rotated

different intensity settings, allowing match of luminance with that of training patterns. The window was rotated to three different angular positions.

Figure 22 shows the prompting photo with an illustration of how the window was varied during sensing. Five patterns per second were sensed by the autoassociative neural network. Figure 22(a) shows two windows of different sizes on Widrow face, and Figure 22(b) shows these windows rotated. Each image within the window was applied to autoassociative neural network in seeking a hit. The entire photo was scanned by moving and changing the window. The error for the best fit to Widrow's face was 0.37%. The error was about four times greater for the best fit to the face of Aragon and Eliashberg. The error was about ten to twenty times greater for best fit to a typical background scene. The system easily chooses Widrow's face for the lowest error. This is not a surprise since three different images of Widrow's face were trained in.

Figures 23–27 show how the mean square error varies as the sensing parameters are varied. These parameters are x–position of sensing window, y–position, rotation of the window, scaling the window size, and brightness or luminance within the window. The plots were made by first optimizing the window on Widrow's face, and varying each

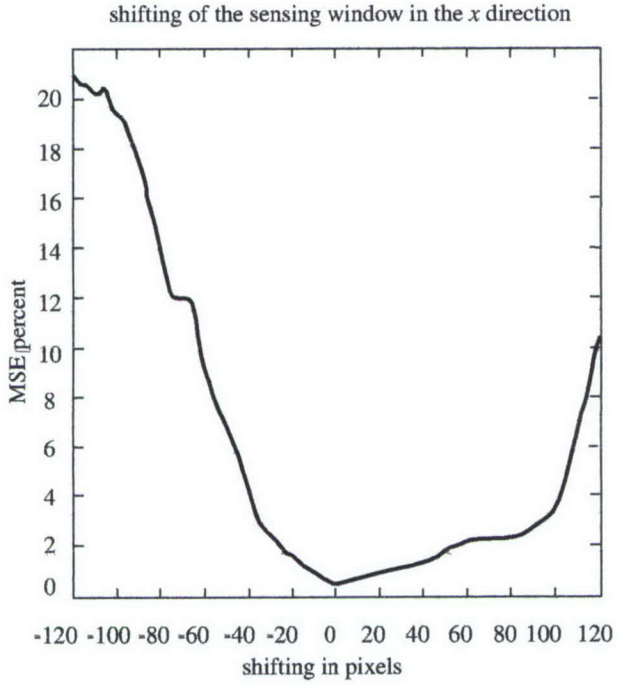


Figure 23: Effects on mean square error due to left/right translation from the optimal position

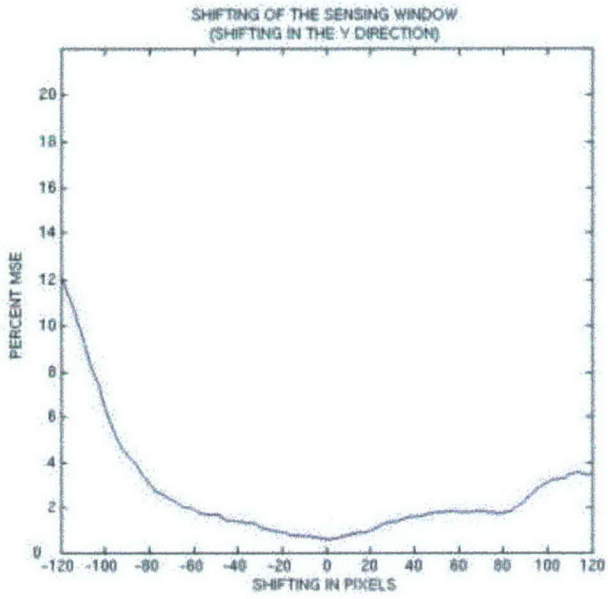


Figure 24: Effects on mean square error due to up/down translation from the optimal position

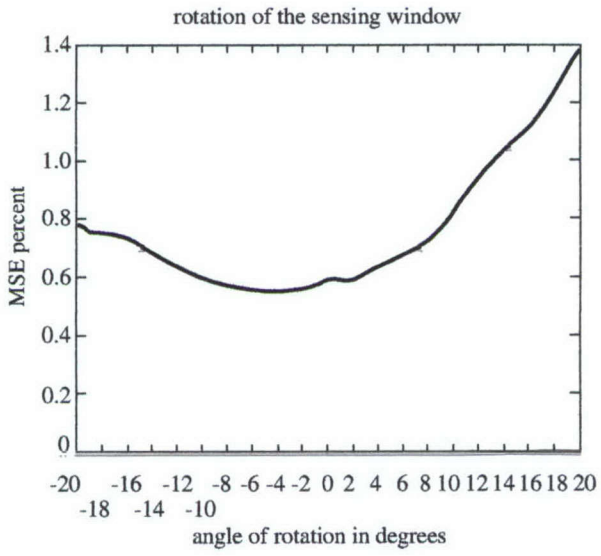


Figure 25: Effects on mean square error due to rotation from the optimal orientation

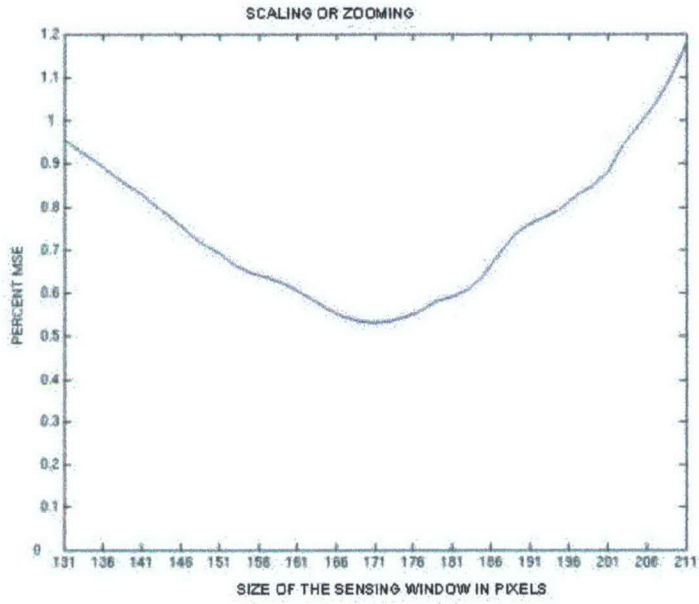


Figure 26: Effects on mean square error due to zooming in and out from the optimal orientation

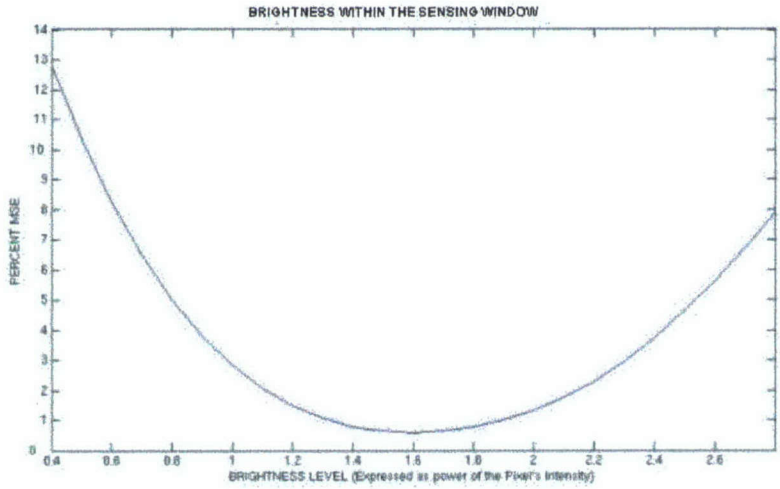


Figure 27: Effects on mean square error due to variation of image brightness from the optimal orientation

parameter separately from the optimum point. These curves have a characteristic "V" shape. They resemble the curves of Figure 9 that were obtained from navigation over Simi Valley. These curves are smooth and essentially monotonic. Seeking the optimal setting has been accomplished automatically by using the Sonthwell Relaxation Method, adjusting one parameter at a time seeking its optimal value, then going to the next parameter, etc. When all of the parameters have been optimized, the cycle is repeated. The cycle is repeated again and again until further reduction of error no longer happens.

For the experiments in facial recognition, we have used a 3–layer autoassociative neural network having the following specifications:

- The input patterns were 50x50 pixels.
- The first layer had 1800 neurons, each with 2500 weights.
- The second layer had 1500 neurons, each with 1800 weights.
- The third layer had 2500 neurons, each with 1500 weights.
- There were 5800 neurons total.
- There were 10,950,000 weights total.

This network was implemented in software on PC with a 64–bit AMD Athlon chip with a 2.6GHz clock. The software used was Matlab64, from the Matlab neural network toolbox for a 64–bit CPU, MATLAB 7 (R14). Once trained, this system can sense through the 10,950,000 weight network at a rate of 5 patterns per second. This same network could be realized with external hardware connected to a host PC computer with a speed–up of about a factor 1,000 for both training and sensing. How this could be done is explained next.

High–Speed Hardware Realization

Many more applications for cognitive memory will open up if the speed of operation could be increased. We have thought long and hard about this situation and propose a two–step approach to high–speed realization. The first step would be based on the use off the shelf FPGA chips mounted on circuit boards which are connected to a host computer. The second step would involve the design and development of ASIC's that would be mounted on circuit boards that may also be connected to a host computer. We have been using the facial recognition system described in the section above as an example. This system uses an autoassociative neural network containing over 10 million weights and about six thousand neurons in three adaptive layers.

Implementation with a 64–bit Athlon PC computer has given us a good deal of experience with the cognitive memory. The bottleneck in the system has been the autoassociative neural network. Precision and retrieval speed is limited by the ability of this network

to sense input patterns to determine if a given input pattern has been seen before. Training the neural network has been done with the backpropagation algorithm, and the speed of adaptation is limited by the rate at which the computer can perform this algorithm.

Implementation of the neural network with FPGA chips will give a speed–up in sensing by approximately a factor of 1,000 over implementation with the Athlon PC, and will give a speed–up in training by approximately a factor of 500 over the PC. Implementation with custom ASIC's will give an additional factor of 100 in speed–up for both sensing and training. A speed–up of a factor of 1,000 would allow automatic real–time analysis of airport surveillance camera video, for example. It would be possible to sense 5,000 patterns per second, with the ability of scanning and analyzing approximately 50 total video images per second. Using ASIC's will not only be faster but will allow the circuit boards to be smaller and in volume production, be cheaper.

We do not propose at this time to build an autoassociative network with ASIC's. It will be essential to get a first experience with FPGA's. Having a neural network constructed with FPGA's and connected to the PC will give us a high–speed digital realization with which we will not only learn how to architect the system hardware, but it will also give us a platform for experimentation and research into the fundamental properties of cognitive memory. Questions of memory capacity, association, and retrieval will be able to be explored efficiently. Building an autoassociative network with FPGA's is an important first step.

A preliminary design for a multi–layer neural network interfaced to a host PC computer will be described next. We begin with a brief discussion of the well known multi–layer network and the backpropagation learning algorithm. Figure 28 shows a two–layer neural network with three neurons in the first layer and two neurons in the second layer. The first layer is fully connected to the input signals, and the second layer is fully connected to the first layer. The sensing path for signal flow from the inputs to the outputs is shown with solid heavy lines. Training of the weights is done by backpropagating the output errors, the differences between the output responses and the desired output responses. The backpropagation paths are shown with dashed lines.

When each input pattern is presented, the output errors of the second layer are calculated and are used to determine the errors for the neurons of the first layer. By this means, a gradient is calculated for every weight in every neuron in the network. These gradient components are partial derivatives of the sum of the squares of the output errors with respect to the individual weights. The *sgm* operators are sigmoid functions, and the *sgm'* operators are functions that are derivatives of sigmoids. How this works is explained in detail in references [12,13,37,41].

To build the hardware, we only need to follow the scheme illustrated in Figure 28. The sensing paths involve multiplying signals by weights, and summing. Each sum is applied

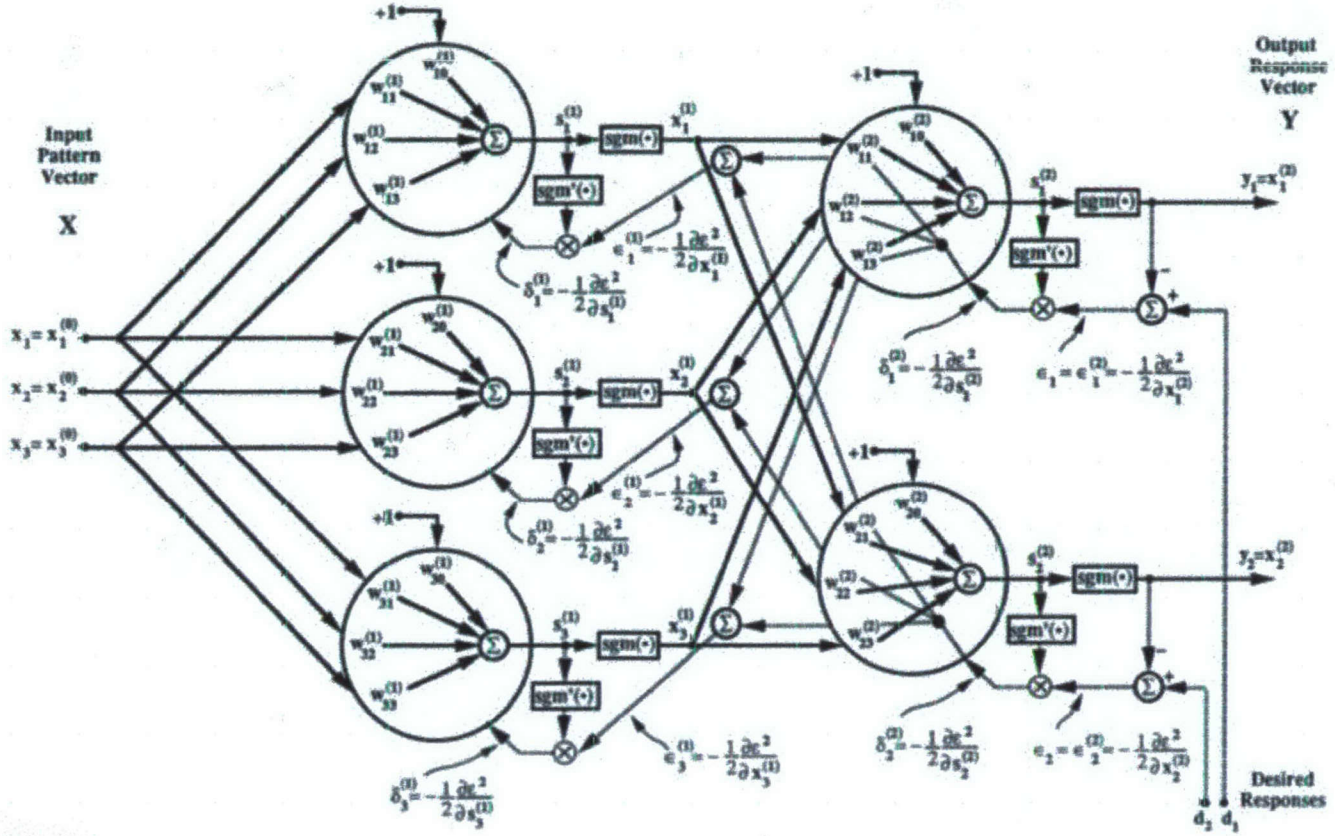


Figure 28: A two–layer neural network trained with the backpropagation algorithm

to *sgm* and *sgm'* functions. The *sgm* and *sgm'* operations can be done in an attached PC computer. This would simplify the hardware, yet would require transfer of signals from the neurons summation points to the computer. The transfers take time, but their number is equal to the number of neurons, which is in the thousands. Transferring weights from the computer to the hardware is another story, since the weights number into the millions.

Transferring neuron input signals and output signals to and from the computer involves thousands of transfers, and this is practical. Transferring weights can be also done practically, as long as this is not done very often.

Transferring neuron input signals and output signals to and from the computer is a good thing to do since the computer would then be able to control the neuron–to–neuron connections. The architecture of the neural network could thereby be controlled by software in the host computer. This kind of flexibility would be needed if the memory system were used for research, and this flexibility would also be necessary in order to customize the autoassociative network for specific applications. One would have a general purpose neural network controlled with software in the host PC. It should be noted that the host PC would also be able to do input pattern manipulations such as rotation, translation, scaling, etc., as well as store the input patterns in folders. The neural hardware and PC computer could work together as a system comprising a cognitive memory. Each piece of the system would be doing what it is best suited for. The neural hardware would break the bottleneck limiting the speed of the entire system.

A particular form of the backpropagation algorithm has been found to be most effective in this research. It is called "batch learning". Suppose for example that we are training 10,000 patterns into the neural network. Each pattern would be presented and the gradients for all the weights would be calculated by the backpropagation algorithm. Instead of changing the weights in proportion to the gradients, the gradients are stored in the computer. The next pattern is then presented, and the gradients are calculated. They are then added to the first set of gradients. Then the next pattern is presented and the gradients are found. They are added to the accumulated sums, and the process goes on. After the 10,000 patterns have been presented, the gradients are averaged and the weight changes are calculated. Then a new set of weights are calculated, and the entire batch of training patterns has been presented and trained for one cycle. The batch cycle is repeated over and over until the mean square error is brought down to the desired level.

With the hardware realization of the neural network, the new set of weights would be calculated in the computer and then transferred to the weight memory of the external hardware. For the present example, over 10,000,000 weights would need to be transferred from the computer. This is a lot of data, but the transfer would need to be done only at the end of each batch cycle, not after the presentation of each training pattern, but after the presentation of 10,000 training patterns. Thus, by doing batch training, the time for transfer of weights has been reduced by a factor of 10,000 with this example. Transfers of

neuron inputs and outputs back and forth to the computer involve thousands of pieces of data. Now, transfers of the weights only involve thousands of pieces of data rather than millions. The numerical operations in the neural circuits are comprised of multiplication and some addition. This can be done in a highly parallel manner by the FPGA's. Thus, the whole process of sensing and training can be done at very high speed, about 1,000 times faster than the computer alone for sensing and about 500 times faster for training which involves a double operation for each training pattern, sensing and backpropagating of errors.

The basic numerical operation for sensing involves multiplying by weights and forming sums of products. The basic numerical operation for backpropagation of error involves multiplying by weights and forming sums of products. This is the same as for sensing. The hardware for sensing and error backpropagation can be the same, first utilized for sensing and then for error backpropagation. The wiring is different, as can be observed from inspection of Figure 28, and this can be achieved by the computer with software corresponding to the network architecture.

The operations of sensing and error backpropagation are similar, using the same hardware. In the remainder of this section, we will explain with a series of diagrams how the neural hardware performs sensing. The understanding is that the workings of error backpropagation are very similar.

Figure 29 shows a 3–layer network, fully connected. Figure 30 shows this network and the design specifications that would allow it to function like the computer simulated

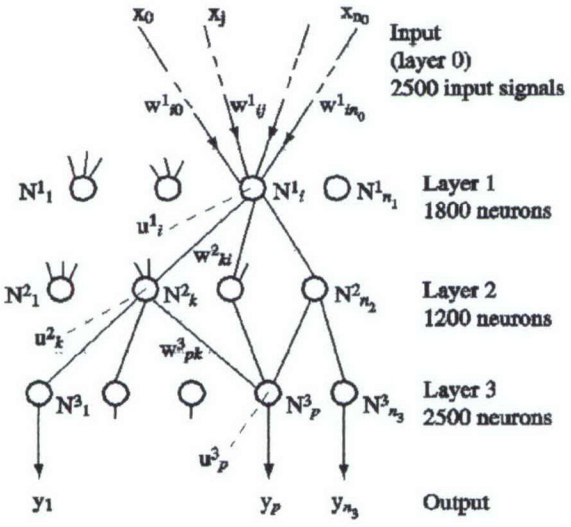


Figure 29: A tree–layer network

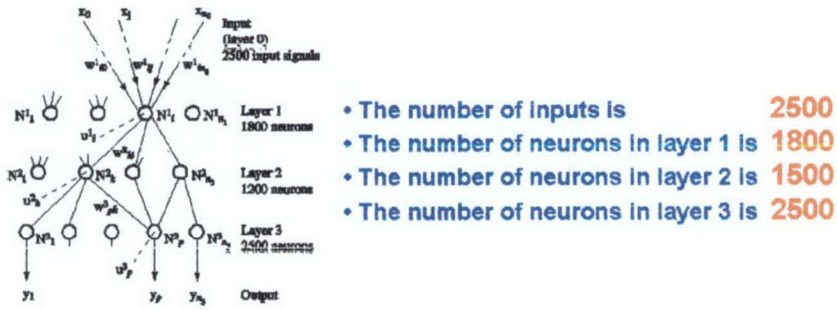


Figure 30: The network that was used for facial recognition

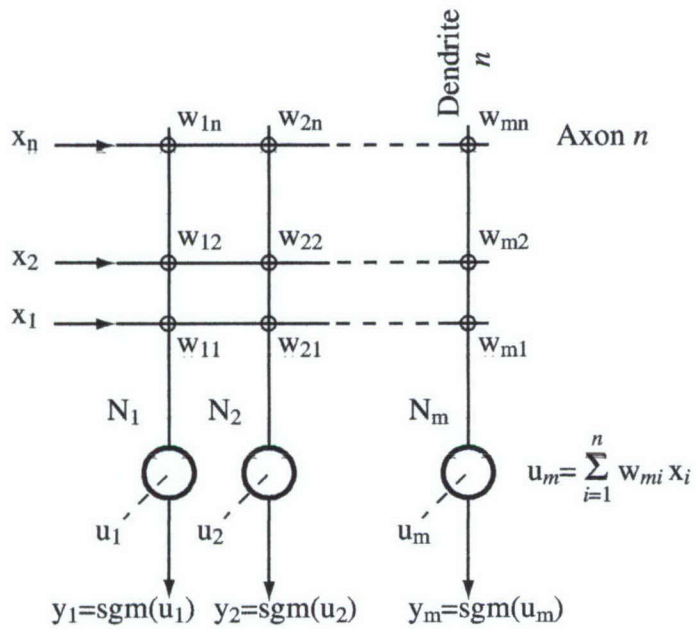


Figure 31: A single layer shown with "programmable logic device" notation

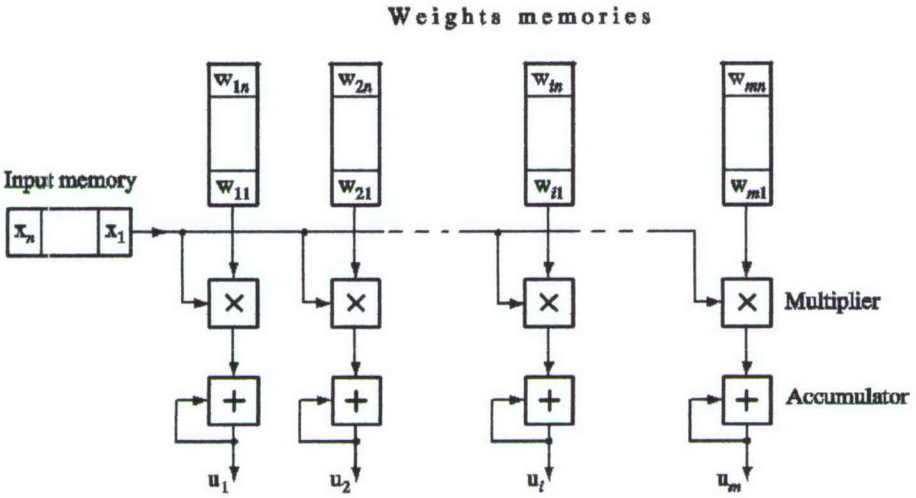


Figure 32: Input pattern memory, weight memories, and multipliers for a single neuron layer

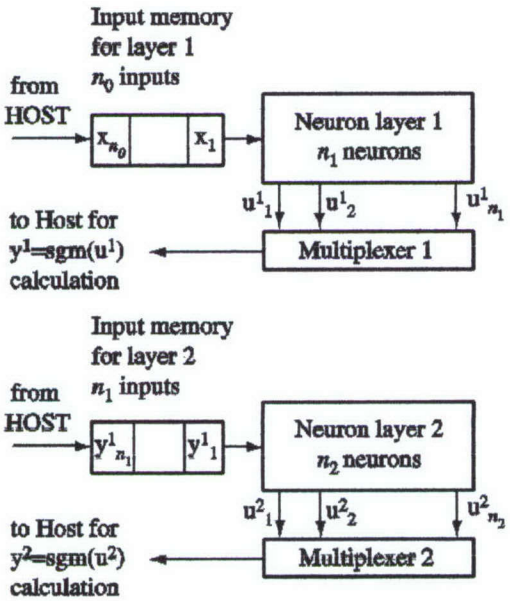


Figure 33: Connections between the host computer and the neural hardware for a 2–layer network

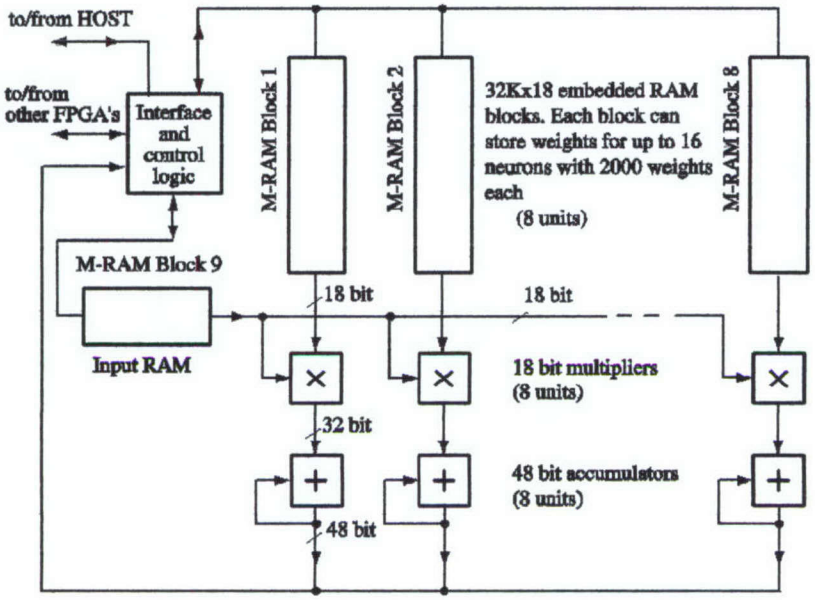


Figure 34: Implementation with FPGA's

network that was used for facial recognition. Figure 31 shows the functioning of a single layer. The sigmoid operators will be realized by the host computer. Figure 32 shows how the weight's memories and input pattern's buffer memory are connected to allow the components of pattern vector and simultaneously the components of the weight vector be shifted simultaneously, multiplied in parallel, and accumulated to produce the neuron's summed outputs. These outputs are fed to the computer for application of the sigmoid operator. Figure 33 shows how the host computer is interfaced to the neural hardware. Each layer of the neural network could be mounted on a single printed circuit board. All the boards could be made identical to each other. The computer will be able to control which neurons are to be used if more than the required number of neurons are mounted on a given board. The boards are designed to have the maximum number of neurons that will be required for any of network layers. Surplus neurons will be ignored by the computer, under software control. Figure 34 indicates how a set of neurons could be implemented with an FPGA. Each circuit board would have fifteen FPGA's, extra memory chips, and control logic.

A Preliminary Proposal for Phase II Research

This section presents a preliminary proposal for a two–year Phase II research program. We would propose a four–part project:

High–speed hardware implementation. This involves building an autoassociative neu-

ral network to connect to a PC computer. The neural hardware would be based on FPGA's. Software to drive this hardware would also need to be developed.

Applications of cognitive memory. This development would be based on the Phase I work on facial recognition, aircraft identification, aircraft navigation, extending this work and testing and verifying with large data sets. Additional application areas will also be developed based on the original work.

Commercial/Military product development. A set of products would be developed based on Phase I experiences. A search engine based on facial images as prompts rather than key–words will be formulated and developed for searching for photos in a personal computer and on the web sites. An airport surveillance camera images of people's faces. Similar systems could provide workplace and public building security. These products would have military use as well as civilian use.

Basic Science. The basic science aspects related to better understanding of the cognitive memory as a new means of computation and as a model for human memory.

An outline of our proposal can be discerned from a set of slides that were presented by Dr. Bernard Widrow to DARPA on January 27, 2005. These slides are self explanatory.

PHASE II RESEARCH

A PRELIMINARY PHASE II RESEARCH PROPOSAL

- **HIGH-SPEED HARDWARE IMPLEMENTATION**

- **APPLICATIONS OF COGNITIVE MEMORY**

- **COMMERCIAL/MILITARY PRODUCTS**

- **BASIC SCIENCE**

A PRELIMINARY PHASE II RESEARCH PROPOSAL

HIGH-SPEED HARDWARE IMPLEMENTATION

- **DESIGN AND BUILD NEURAL CIRCUITS BASED ON FPGA'S**
- **DESIGN DRIVER SOFTWARE FOR NEURAL CIRCUITS**
- **GAIN OPERATING EXPERIENCE WITH COGNITIVE HARDWARE**
- **BENCHMARK SENSING AND TRAINING SPEED**
- **HAVING HAD DESIGN AND OPERATING EXPERIENCE WITH FPGA HARDWARE, DESIGN A FASTER, SIMPLER, CHEAPER SYSTEM BASED ON ASIC'S**

A PRELIMINARY PHASE II RESEARCH PROPOSAL

APPLICATIONS OF COGNITIVE MEMORY

- **FACIAL RECOGNITION**
- **AERIAL PHOTO ANALYSIS**
- **SATELLITE PHOTO ANALYSIS**
- **DRONE AIRCRAFT PHOTO ANALYSIS**
- **AUTOMATIC HIGHWAY SURVEILLANCE**
- **VIDEO IMAGE ANAYLSIS**

A PRELIMINARY PHASE II RESEARCH PROPOSAL

COMMERCIAL/MILITARY PRODUCTS

- **SEARCH ENGINE FOR PHOTOS**
- **AIRPORT SURVEILLANCE SYSTEM**
- **WORKPLACE SECURITY**
- **PUBLIC PLACE/BUILDING SURVEILLANCE**

A PRELIMINARY PHASE II RESEARCH PROPOSAL

BASIC SCIENCE

DEVELOP AND REFINE COGNITIVE MEMORY CONCEPT

- **CONTINUE STUDY OF ARCHITECTURE OF AUTOASSOCIATIVE NEURAL NETWORK AS IT RELATES TO GENERALIZATION AND MEMORY CAPACITY**
- **DEVELOP GRADIENT METHODS TO SPEED OPTIMIZATION PROCESS FOR MINIMIZING MEAN SQUARE ERROR**
- **INVENT A SHORT TERM MEMORY CONCEPT TO WORK IN CONJUNCTION WITH THE COGNITIVE MEMORY**
- **CONFER WITH PSYCHOLOGISTS AND NEUROSCIENTISTS:**
  - **DR. BILL NEWSOME, NEUROBIOLOGY**
  - **DR. JED BLACK, PSYCHIATRY AND BEHAVIORAL SCIENCES**
  - **DR. JEROME YESAVAGE, PSYCHIATRY AND BEHAVIORAL SCIENCES**
  - **DR. PATRICK SUPPES, PHILOSOPHY**

References

[1] J. A. Barnden. High-level reasoning, computational challenges for connectionism, and the Conposit solution. *Appl. Intell.*, 5(2):103–135, Apr. 1995.

[2] B. E. Burnside, D. L. Rubin, and R. Shachter. A Bayesian network for mammography. Technical Report SMI-2001-0867, Stanford Medical Informatics, 2000.

[3] G. A. Carpenter and S. Grossberg. Adaptive resonance theory. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 87–90. MIT Press, Cambridge, MA, 2nd edition, 2003.

[4] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* John Wiley & Sons, New York, 2nd edition, 2001.

[5] H. Eichenbaum. *The Cognitive Neuroscience of Memory: An Introduction.* Oxford University Press, New York, 2002.

[6] L. Esserman, H. Cowley, C. Eberle, A. Kirkpatrick, S. Chang, K. Berbaum, and A. Gale. Improving the accuracy of mammography: volume and outcome relationships. *J. Natl. Cancer Inst.*, 94(5):321–323, 6 Mar. 2002.

[7] K. Fukushima. Cognitron: a self-organizing multilayered neural network. *Biol. Cybern.*, 20(3-4):127–136, 5 Nov. 1975.

[8] K. Fukushima and S. Miyake. A self-organizing neural network with a function of associative memory: feedback-type cognitron. *Biol. Cybern.*, 28(4):201–208, 3 Mar. 1978.

[9] J. M. Fuster. *Cortex and Mind: Unifying Cognition.* Oxford University Press, New York, 2002.

[10] S. Grossberg. Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors. *Biol. Cybern.*, 23(3):121–134, 30 July 1976.

[11] S. Grossberg. Adaptive pattern classification and universal recoding: II. Feedback, expectation, olfaction, illusions. *Biol. Cybern.*, 23(4):187–202, 30 Aug. 1976.

[12] S. Haykin. *Neural Networks: A Comprehensive Foundation.* Prentice Hall, Upper Saddle River, NJ, 2nd edition, 1999.

[13] R. Hecht-Nielsen. *Neurocomputing.* Addison Wesley, Reading, MA, 1989.

[14] R. Hecht-Nielsen. A theory of cerebral cortex. Technical Report 03.01, UCSD Institute for Neural Computation, La Jolla, CA, 24 Oct. 2003.

[15] R. Hecht-Nielsen. A theory of thalamocortex. In R. Hecht-Nielsen and T. McKenna, editors, *Computational Models for Neuroscience: Human Cortical Information Processing*, pages 85–124. Springer Verlag, London, 2003.

[16] R. Hecht-Nielsen and T. McKenna, editors. *Computational Models for Neuroscience: Human Cortical Information Processing.* Springer Verlag, London, 2003.

[17] J. H. Holland. Genetic algorithms. *Sci. Am.*, 267(1):44–50, July 1992.

[18] T. Kohonen. *Self-Organization and Associative Memory.* Springer Verlag, Berlin, 3rd edition, 1989.

[19] T. Kohonen. *Self-Organizing Maps.* Springer, Berlin, 3rd edition, 2001.

[20] I. B. Levitan and L. K. Kaczmarek. *The Neuron: Cell and Molecular Biology.* Oxford University Press, New York, 2001.

[21] D. E. Meyer and D. E. Kieras. A computational theory of executive cognitive processes and multiple-task performance: I. Basic mechanisms. *Psychol. Rev.*, 104(1):3–65, Jan. 1997.

[22] D. E. Meyer and D. E. Kieras. A computational theory of executive cognitive processes and multiple-task performance: II. Accounts of psychological refractory-period phenomena. *Psychol. Rev.*, 104(4):749–791, Oct. 1997.

[23] D. E. Meyer and D. E. Kieras. An overview of the EPIC architecture for cognition and performance with application to human-computer interaction. *Hum.-Comp. Interact.*, 12(4):391–438, 1997.

[24] C. L. Nash, K. O. Perlmutter, and R. M. Gray. Evaluation of Bayes risk weighted vector quantization with posterior estimation in the detection of lesions in digitized mammograms. In *Proc. Asilomar Conf. Signals Syst. Computers*, volume 1, pages 716–720, Pacific Grove, CA, Oct–Nov 1994.

[25] A. Newell. *Unified Theories of Cognition.* Harvard University Press, Cambridge, MA, 1990.

[26] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychol. Rev.*, 65(6):386–408, 1958.

[27] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM J. Res. Dev.*, 3(3):210–229, 1959.

[28] A. L. Samuel. Some studies in machine learning using the game of checkers II – recent progress. *IBM J. Res. Dev.*, 11(6):601–617, 1967.

[29] W. B. Scoville and B. Milner. Loss of recent memory after bilateral hippocampal lesions. *J. Neurol. Neurosurg. Psychiatry*, 20(1):11–21, 1957.

[30] G. M. Shepherd, editor. *The Synaptic Organization of the Brain.* Oxford University Press, New York, 5th edition, 2003.

[31] L. R. Squire. Memory and the hippocampus: a synthesis from findings with rats, monkeys and humans. *Psychol. Rev.*, 99(2):195–231, Apr. 1992.

[32] K. Steinbuch. Die Lernmatrix. *Kybernetik*, 1(1):36–45, 1961.

[33] K. Steinbuch and U. A. W. Piske. Learning matrices and their applications. *IEEE Trans. Electron. Computers*, 12:846–862, 1963.

[34] L. W. Swanson. *Brain Architecture: Understanding the Basic Plan.* Oxford University Press, New York, 2003.

[35] E. Thurfjell, M. G. Thurfjell, E. Egge, and N. Bjurstam. Sensitivity and specificity of computer-assisted breast cancer detection in mammography screening. *Acta Radiol.*, 39(4):384–388, July 1998.

[36] M. P. Walker. A refined model of sleep and the time course of memory formation. *Accepted for publication in Behav. Brain Sci.*, 2004.

[37] P. J. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences.* PhD thesis, Harvard University, Cambridge, MA, Aug. 1974.

[38] B. Widrow and M. E. Hoff. Adaptive switching circuits. In *IRE WESCON Conv. Rec.*, volume 4, pages 96–104, 1960.

[39] B. Widrow and M. Kamenetsky. On the efficiency of adaptive algorithms. In S. Haykin and B. Widrow, editors, *Least-Mean-Square Adaptive Filters.* John Wiley & Sons, New York, 2003.

[40] B. Widrow and M. Kamenetsky. Statistical efficiency of adaptive algorithms. *Neural Netw.*, 16(5–6):735–744, June–July 2003.

[41] B. Widrow and M. A. Lehr. 30 years of adaptive neural networks: perceptron, Madaline, and backpropagation. *Proc. IEEE*, 78(9):1415–1442, Sep. 1990.

[42] B. Widrow and S. D. Stearns. *Adaptive Signal Processing.* Prentice Hall, Upper Saddle River, NJ, 1985.

[43] M. Young, R. G. Eggleston, and R. Whitaker. Direct manipulation interface techniques for interaction with software agents. In *Proc. RTO Human Factors Med. Panel Symp.*, pages 19–1–19–10, Oslo, Norway, Apr. 2000.

[44] L. A. Zadeh. Fuzzy logic, neural networks, and soft computing. *Commun. ACM*, 37(3):77–84, Mar. 1994.