2001

Image quality reconstruction with neural networks

Stephen Edward Sipes
San Jose State University

Follow this and additional works at: https://scholarworks.sjsu.edu/etd_theses

Recommended Citation

Sipes, Stephen Edward, "Image quality reconstruction with neural networks" (2001). *Master's Theses*. 2159.

DOI: <https://doi.org/10.31979/etd.368v-gxbd>
https://scholarworks.sjsu.edu/etd_theses/2159

This Thesis is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Theses by an authorized administrator of SJSU ScholarWorks. For more information, please contact scholarworks@sjsu.edu.

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI[®]

Image Quality Reconstruction with Neural Networks

A Thesis Draft

Presented to

The Faculty of Human Factors and Ergonomics

San Jose State University

In Partial Fulfillment

Of the requirement for the Degree

Master of Science

By

Stephen Edward Sipes

May 2001

UMI Number: 1403992

Copyright 2001 by
Sipes, Stephen Edward

All rights reserved.

UMI[®]

UMI Microform 1403992

Copyright 2001 by Bell & Howell Information and Learning Company.

All rights reserved. This microform edition is protected against
unauthorized copying under Title 17, United States Code.

Bell & Howell Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

©2001

Stephen Edward Sipes

ALL RIGHTS RESERVED

APPROVED FOR THE PROGRAM OF HUMAN FACTORS AND ERGONOMICS

Dr. Kevin Corker, Chair
San Jose State University

Dr. Anthony D. Andre, Member
San Jose State University

Dr. John F. McGowan, Member
NASA Ames Research Center

APPROVED FOR THE UNIVERSITY

ABSTRACT

IMAGE QUALITY RECONSTRUCTION WITH NEURAL NETWORKS

by Stephen E. Sipes

This thesis addresses the problems that are associated with videoconferencing over limited bandwidth media. Bandwidth requirements limit frame size, frame rates and resolution. A method to manage video with bandwidth limitations involves a process known as compression. Compression reduces the amount of information at the expense of image quality, which is currently irreversible.

Many researchers are looking into ways to enhance the quality of the compressed image. This study evaluates whether some standard artificial neural networks can improve the quality of compressed image. Quality was measured using the standard image metric, the Mean Squared Error that is equivalent to the Peak Signal to Noise Ratio (PSNR) and visual inspection of the enhanced images.

DEDICATION

I would like to dedicate this to all the people in my life that had told me that this was never going to happen.

ACKNOWLEDGEMENTS

I would have been unable to make this thesis if it wasn't for the help and patience of many people. I would personally like to thank John F. McGowan Ph.D. for all of his insights, as well as the critical help in the software applications and explanations. Kevin Corker Ph.D. for his belief in my project and me, as well as his guidance in shaping this project from the beginning. And to my parents, Jim and Ann Sipes Ph.D., if it wasn't for their love, and guidance (and money) this would have never been possible.

I would also like to thank Roxana Greenman Ph.D. on her insight (patience) with explaining Neural Nets in the very beginning of this project. Dave Yaste and Kent Schiffer on their endless help and input on the computer related matters. And I would also like to thank Deanna Cote' for her help with the final editing and reassurance she provided me.

TABLE OF CONTENTS

ABSTRACT	IV
ACKNOWLEDGEMENTS	VI
TABLE OF CONTENTS	VII
STATEMENT OF PROBLEM.....	2
DELIMITATION	4
IMPORTANCE OF THIS STUDY	4
VIDEO-CONFERENCING	4
HUMAN FACTORS AND THE COMPRESSION MODEL.....	7
NEURAL NETWORKS	12
METHOD	20
EXPERIMENTAL DESIGN	20
APPARATUS	22
SOFTWARE.....	22
MATERIALS	23
PROCEDURES	23
ANALYSIS OF DATA	27
RESULTS	28
DISCUSSION	62

SUMMARY	65
RECOMMENDATIONS FOR FURTHER STUDIES	65
CONCLUSION	67
REFERENCES.....	69
APPENDIX A	73
APPENDIX B:	87

TABLE OF FIGURES

<u>Figure 1.</u> Original Image 43.3 kb.....	3
<u>Figure 2.</u> Compressed Image 2.5 kb.....	3
<u>Figure 3.</u> Step 1 of Image Compression.....	8
<u>Figure 4.</u> Step 2 of Image Compression	9
<u>Figure 5.</u> Step 3 of Image Compression.....	9
<u>Figure 6.</u> Step 4 of Image Compression	10
<u>Figure 7.</u> Step 5 of Image Compression.....	10
<u>Figure 8.</u> Biological Neuron	14
<u>Figure 9.</u> Back Propagation Neural Network Structure.....	16
<u>Figure 10.</u> Overall Method Process.....	26
<u>Table 1.</u> Independent Variables and Results for Experiments 1,2,3.	29
<u>Graph 1.1</u> Plotted Mean Squared Error from Experiment 1	30
<u>Figure 11.</u> Image Results From Experiment 1.....	31
<u>Graph 1.2</u> Plotted Mean Squared Error from Experiment 2.....	32
<u>Figure 12.</u> Image Results From Experiment 2.....	33
<u>Graph 1.3</u> Plotted Mean Squared Error from Experiment 3	34
<u>Figure 13.</u> Image Results From Experiment 3.....	35
<u>Table 2.</u> Independent Variables and Results for Experiments 4,5,6.	36
<u>Graph 2.1</u> Plotted Mean Squared Error from Experiment 4.....	37
<u>Figure 14.</u> Image Results From Experiment 4.....	38
<u>Graph 2.2</u> Plotted Mean Squared Error from Experiment 5	39

<u>Figure 15.</u> Image Results From Experiment 5	40
<u>Graph 2.3</u> Plotted Mean Squared Error from Experiment 6	41
<u>Figure 16.</u> Image Results From Experiment 6	42
<u>Table 3.</u> Independent Variables and Results for Experiments 7,8,9	43
<u>Graph 3.1</u> Plotted Mean Squared Error from Experiment 7	44
<u>Figure 17.</u> Image Results From Experiment 7	45
<u>Graph 3.2</u> Plotted Mean Squared Error from Experiment 8	46
<u>Figure 18.</u> Image Results From Experiment 8	47
<u>Graph 3.3</u> Plotted Mean Squared Error from Experiment 9	48
<u>Figure 19.</u> Image Results From Experiment 9	49
<u>Table 4.</u> Independent Variables and Results for Experiments 10,11,12.	50
<u>Graph 4.1</u> Plotted Mean Squared Error from Experiment 10	51
<u>Figure 20.</u> Image Results From Experiment 10	52
<u>Graph 4.2</u> Plotted Mean Squared Error from Experiment 11	53
<u>Figure 21.</u> Image Results From Experiment 11	54
<u>Graph 4.3</u> Plotted Mean Squared Error from Experiment 12	55
<u>Figure 22.</u> Image Results From Experiment 12	56
<u>Table 5.</u> Independent Variables and Results for Experiments 13,14	57
<u>Graph 5.1</u> Plotted Mean Squared Error from Experiment 13	58
<u>Figure 23.</u> Image Results From Experiment 13	59
<u>Graph 5.2</u> Plotted Mean Squared Error from Experiment 14	60
<u>Figure 24.</u> Image Results From Experiment 14	61

INTRODUCTION

Information technologies are devices or services that transfer information between people in order to decrease time, travel and distances. Examples of these technologies are the Internet, email, cellular phones, pagers and computers. These technologies enable people to be connected to information remotely, without face-to-face interaction that would otherwise be limited to specific settings, i.e. libraries, the workplace, and other business locations. Despite these recent advances in technology, contextual information that is relevant gets lost when passed through these types of media.

Research on this topic has shown that face-to-face communication is still the most effective form of interaction (Rosen, 1996). The effectiveness of the conversation can be attributed to the multiple types of information being transferred during the exchange. There is “explicit” information, what is actually being said and “implicit” information, what is being intimated through the voice and body language cues. Studies have shown that the addition of the implicit interaction is preferable and more meaningful. The conversation becomes more engaging to people mainly because the speaker or listener associates these types of cues with societal meanings, such as the roll of the eyes for sarcasm or yawning intimating boredom (Whittaker, O’Conaill, 1997).

Videoconferencing is a technology that tries to bridge the information gap that exists between face-to-face communication and remote communication. However, due to technological constraints, many negative perceptions exist in regards to video conferencing. These negative perceptions are created by, imperfect synchronization between video and audio, jerky movement from missing frames or uneven display of

frames, jagged edges and distorted features within the image and audio latency. These issues involve processing speeds to compress/decompress images and audio, memory requirements to cache sequential video frames and the limited bandwidth allocations of transmission media. A deficiency in any one of these components can negatively impact frame rates, transmission latency and reduce the resolution, which in turn degrades the end users' perception of this technology.

Statement of Problem

The most important factor that directly impacts the performance of videoconferencing is bandwidth. Bandwidth can be defined as the amount of information (measured in bits per second) that can be transferred over a medium, such as a telephone wire or Ethernet connection. Bandwidth becomes the important factor in video conferencing because images have the largest data requirement. Bandwidth is directly proportional to both resolution and frame rate. If either resolution or frame rate is increased, the overall bandwidth requirement will be increased.

Frame rates of at least 16 frames per second, presented evenly in time, achieve smooth believable motion to the human viewer (Apteker, Fisher, Kisimov, Neishlos, 1995). Spatial resolution of about 352 x 240 appears to be the lowest acceptable resolution to the human viewer. Currently, this is the spatial resolution of a single field of the analog NTSC television standard in the United States and the Video CD MPEG-1 digital video products. In contrast, the Plain Old Telephone System (POTS) videoconferencing products achieve a frame rate less than 15 frames per second and produce spatial resolutions of 176 by 120 or less with visible blocking artifacts and

distorted edges. The overall reduction of the frame rate and resolution is directly related to the low bandwidth limitation (56 kilobits per second) of the telephone wire.

Figure 1. Original Image 43.3 kb

Figure 2. Compressed image 2.5 kb

The current method of video transmission for limited bandwidth resources, such as the 56KBPS restriction of POTS, is managed by a process of “compression”. Video compression selectively reduces the amount of information within the video stream to fit the bandwidth available. This process reduces the overall frame rate and degrades the image quality (resolution). Most viewers react unfavorably to gross or unnatural visual artifacts such as the blocking and edge distortions seen in highly compressed video, as illustrated in figures 1 & 2.

Once this compression occurs, the original image quality cannot be restored. Recently, researchers have been studying the possibility of image reconstruction with the aid of Artificial Neural Networks.

Currently, the Plain Old Telephone System (POTS) bandwidth of 56kbps is the limiting factor in sending data. Because of this restriction, the amount of data that can be

sent can never produce high-resolution full-motion video conferencing, which requires 384 Kbps for acceptable resolution and frame rate.

Current technologies for the 56kbps market have been able to produce high-resolution images, but with low frame rates and considerable latency. The purpose of this study is to determine if Neural Networks can produce high-resolution image enhancements for the human viewer while using highly compressed images for low bandwidth transmissions.

Delimitation

This study will use single still images for testing and evaluation of the effectiveness from the Neural Network. By using still images the overall time involved is dramatically reduced as compared to embedding the program within the video system for evaluation. The results of the study are still valid and can be generalized to a series of frames that the video is comprised of.

Importance of this study

Many researchers are investigating alternative ways to reduce the amount of data transmitted while preserving image quality. This study evaluates the feasibility of enhancing the quality of compressed images using Artificial Neural Networks and measures the magnitude of any quality improvements using objective and subjective quality metrics. This study will help answer whether Artificial Neural Networks can be used to enhance videoconferencing over POTS lines (56 Kbits/second).

Video-Conferencing

In the early 1920s, Bell Labs began to investigate sending / receiving images and audio as part of telecommunications applications. Their primary technical focus was

television, but by the 1930's they had produced a two way interaction demonstration of this technology with a conference between participants in New York City and Washington D.C. (Winston, 1998).

Due to the beginning of World War II, research on videoconferencing stopped until the 1950's when Bell Labs began to research this topic again. Shortly after this, the first product aimed for the public acceptance was introduced in the 1964 Worlds Fair in New York. Bulky by today's standards, it used an asynchronous path for sending and receiving images and audio. It was estimated that it required up to 100 telephone lines to send the image alone (equivalent to 5.6 megabytes) and weighed 26 pounds (Trowt-Bayard, Wilcox, 1997).

In the late 1960's and early 1970's, major advances in satellite and microwave technology enabled larger data rates to be transmitted over greater distances, which were needed for video conferencing (Hilton, Jacobi, 1986). This new capability now allowed larger data rates to be transferred to areas that did not previously have the capacity to handle such a demand on the existing infrastructure. However, it was still impractical for many end users in the commercial market. The substantial investment in television production equipment, leasing of satellite time and manpower that was needed for these conferences could only be offset by the amount of employee travel; therefore, only large companies or government organizations could begin to realize a cost benefit opportunity.

In the mid 1970's, a software-based solution was developed to reduce the enormous amounts of raw data within the transmitted signal that was being used in satellite videoconferencing. This technique, called compression, reduces the amount of redundant information stored within the image being transmitted, while retaining the

quality of the image being displayed (Angiolillo, Blanchard, Israelski, Mane', 1997).

This new discovery lowered the requirements needed for satellite technologies and shifted the focus of transmission of video over to the existing telephone infrastructure.

This in turn reduced the huge monetary investments needed for video conferencing and allowed mid-level companies to participate on both the consumer and production side of this emerging market.

Despite the major reduction in associated costs with satellite technology, large computer systems were still needed for the process of compressing and decompressing of images. This hardware solution later became known as a "CODEC" for the processes that it involves; COmpression/DECompression (Angiolillo, et al. 1997).

By the early 1980's, the problems associated with frame rates were being addressed through the creation of allocated physical memory. Allocated physical memory, otherwise known as RAM memory, stored sequential images offline while a current image was being processed. Methods for rendering the images in "real time", even by today's technologies, take a certain amount of time to actually produce the image for viewing. Without a way to store the sequential image frames, the resultant video was "jerky" at best. With the ability to "cache" or store sequential data, the frame rates began increasing to the needed 30 FPS for smooth believable motion (Trowt et al. 1997).

The combination of these factors led to the sudden growth and interest in transmitted video technology in the mid 1980's to early 1990's. However, because of company trade secrets, most vendors' applications and technology would not interoperate with each other, creating problems for the end users and the industry in general.

In the early 1990's, the International Telecommunications Union (ITU) established standards for video conferencing and video compression (Rosen, 1996). These standards ensure that manufacturers and software developers will interoperate with other video products for that specific transmission protocol, while incorporating additional features. The ITU standards for video over specific transmission media are H.32x. For instance, transmission over ISDN line is known as H.320, transmission over IP (LAN) is H.324 and transmission over telephone (56k) is H.324. The ITU Standards for video compression are H.261, H.263 and H.263+ respectively.

Human Factors and the Compression Model

Current international standards for video-conferencing, the H.26x standards, the related MPEG digital video standard, and the JPEG still image compression standard, are all based on the block Discrete Cosine Transform (DCT) Algorithm. As a compression function, DCT integrates well with the behavior of the Human visual system.

Psychometric studies have shown that the visual system is less sensitive to fine detailed spatial patterns than the coarse spatial properties of an object. DCT reduces the data being transmitted by removing the high spatial frequencies (detailed visual patterns) from the image. The more that the image is "compressed" the more information within the picture will be lost (Pennebaker, Mitchell, 1993).

The DCT converts absolute pixel values of the original 8x8 block to Amplitudes of Spatial Frequency components within a new 8x8 block. The 8x8 block of pixels can be represented as the weighted sum of 64 DCT basis functions, which are superposition's of horizontal and vertical sinusoidal patterns of pixel intensity. The DCT coefficients, the output of the forward DCT, are the weights in the sum.

The DCT process can be mathematically expressed in the following way:

$$\frac{C(v)}{2} \frac{C(u)}{2} \sum_{x=0}^7 \sum_{y=0}^7 s(y, x) \cos[(2X+1)u\pi/16] \cos[(2y+1)v\pi/16]$$

However a step-by-step overview will show what the equation means and how the DCT process occurs.

23	56	78	122	155	155	155	101
23	56	77	130	155	155	248	249
12	16	54	25	28	74	88	98
12	16	54	25	27	74	87	98
23	56	78	122	155	155	155	101
23	56	77	130	155	155	248	249
12	16	54	25	28	74	88	98
12	16	54	25	27	74	87	98

Figure 3. Step 1.

Step 1: An 8x8 block of the image is converted to absolute pixel values.

Figure 4 Step 2.

Step 2: 8 random samples of pixel intensity values are taken from the 8x8 block of data.

These values are represented by the equation: $s(x)$

Figure 5 Step 3.

Step 3: Sampled values are now "zero shifted", or averaged, to achieve a new value.

Computers can only process binary values of 0's and 1's. Zero shifting is a way to allow the computer to generate negative values that are needed for processing the image.

Because the sample values are from the range of 0 to 255 the average value becomes 128.

Negative values can be represented as values closer to zero and the Positive values are closer to 1.

Step 4: Each value of the DCT function (u) is multiplied by the corresponding sample value.

Figure 6 Step 4.

Figure 7 Step 5.

Step 5: The products of the values are then summed to produce a new value that represents a waveform $S(u)$. (For illustrative purposes, only one value of this process is shown).

Step 6: Quantization is the process in which the newest values attained are processed into integers. Simply, the newest values that were created from the last step are now divided by 16. By dividing by 16, the amount of information to be sent has been reduced significantly and dividing by 16 would reduce the values that were close to zero, as a high spatial frequency might be, to zero. This is how the actual compression occurs.

Reducing the least significant values systematically enables the equation to remove the highest frequency while retaining the most salient attributes within the image. Unfortunately these steps must be carried out on each individual 8x8 block within the image, which increases the amount of time and processing overhead preparing the image for the actual compression and transmission.

H.26x video usually requires at least 384 Kbps to achieve the frame rates, spatial resolution, and image clarity most users find acceptable, which is only found in DSL or Intranet Network that is capable of that bandwidth. However, video-conferencing over the plain old telephone system, POTS, is desirable because of the large installed base of POTS lines, the traditional copper analog telephone lines connecting homes and offices to central offices or telephone switches. Upgrading to newer technologies such as Integrated Services Digital Network (ISDN), Digital Subscriber Line (DSL), or Cable modems requires substantial expense. Further ISDN, DSL and Cable modem technologies have suffered from competing standards and incompatible implementations of the standards. In general, the more ubiquitous the technology is, the overall value of that technology will increase. The downside to POTS is that the bandwidth is limited to 56 Kilobits per second (Kbps) or lower, even with state of the art modem technology.

Therefore, finding solutions to overcome the problems associated with low bandwidth technology is quickly becoming the focus of research for many companies. One research topic is in the area of Neural Networks and the ability for such a tool to be implemented in low bandwidth video technology.

Neural Networks

Artificial Neural Networks is a mathematical tool that is metaphorically associated to the structures of the brain because of the topographical similarities in design and actions (Taylor, 1993). Most, if not all of the initial investigators into Neural Networks were from a biological, physiological or psychological background. The term “Neural Network” comes from these early investigators trying to explain the process of this technology by using terminology they understood best (Eberhart, Simpson, Dobbins, 1996).

An Artificial Neural Networks, like the biological counterpart, can process data, adjust its parameters to change its eventual outcome with or without supervision, and recall previous examples to solve future problems (Kim, 1994). However, it should be noted that Artificial Neural Networks can only mimic some of the functions of the brain and, in comparison, are “simple associations” in relation to the complexity of interactions the biological processes can accomplish.

Warren McCulloch and Walter Pitts were the first to lay the groundwork on Artificial Neural Networks in a paper they published in 1943, on what they called Binary Decision Units (Taylor, 1993). Their work became a starting point for the study of Artificial Neural Networks. Their research had shown that Parallel Processing of a signal was now possible and these Parallel Processes could express logical functions (Eberhart,

Simpson, Dobbins, 1996). Even though most of their work is not used today, their basic assumptions underlie the current methods and theories for Neural Networks. Later researchers who have made major contributions to the field have had the distinction of having a certain type of “Network model” named after the process they had shown to work.

Biological Processing. For the purpose of this thesis, a brief overview of only three structures of the biological neuron cell will be discussed. This paper will show only the functional similarities that exist between Artificial Neural Networks and Biological Neurons. For a more in depth review of these processes, please refer to: Biological Psychology. (Kalat, 1992).

All biological neurons transmit and relay information that regulates internal and external stimuli for the body. It is through these mechanisms that we receive feedback, in the form of sensation and perception (Klien, 2000). Most, if not all, biological neurons are composed of dendrites, cell bodies and axons. It is the interaction of these structures and how they pass information along that plays an important function in feedback.

A simple illustration of this interaction would be to suppose an event occurs that will create a sensation. From this occurrence, signals have now been created into chemical and electrical translations which the brain can interpret. These chemical and electrical signals are carried through the dendrites to the Neuron Cell body. The Neuron Cell body then regulates the signal to be passed, depending on the strength of these

Figure 8. Biological Neuron

signals (action potentials). A positive signal increases the chance of the signal to be propagated further, while a negative signal will inhibit further transmission. The resultant signal is then sent out along the axon to be processed again.(Kalat, 1992). It is the cumulative total of all of these transmitted signals, from multiple neurons, that make perceptual sensations occur.

Feed Forward Artificial Neural Networks operate in a similar manner as the biological counterpart. A Neural Network is composed of Nodes, Biases, Weights, Connections and Logical Functions. The Connections of a Network can be associated to the function of the Dendrites within the Biological Neuron. Connections pass the information from one layer of the network to the next. The Weight (a numeric value) is added to the original information, which is the equivalent of the signal strength of the connection (Dendrite) to each Node. Layers of the Network have Nodes that process the data. These Nodes (Neurons) contain a Logical Function, a selection parameter, in which it will compute the added bias and weight to give an output value for the Network.

Multiple layers of Nodes that can exist in a Network are chained together by the Connections to increase the probability of the Network's output being close to its goal.

All data that is presented to the Network is converted into binary numbers values (0's and 1's) which can be understood by the Network. Floating point processing is a way to manage the large ranges of numeric values between 0 and 1, representing the range of input values between 1 and 0 as .999999999999 down to .000000000001.

Therefore, the computer that the Network is running on must be capable of processing this data as well as the storage for this large range of values. The following is an overview of the processes in a Feed Forward backpropagation Network. For more in depth information on the processes of backpropagation and Neural Networks please refer to Computational Intelligence PC Tools, (Eberhart, Simpson, Dobbins. 1996).

Figure 9. Back Propagation Neural Network Structure

A: Is considered the first layer of the Network. This representation is misleading in that it is actually your data that you are introducing to the Network. The data is presented to the Network in the form of vectors or matrices but can support arrays of data. The individual bits of data are formally known as a Processing Element or Elements (PE) depending on the size of your data.

B: Data is being sent to the second layer (a forward transition) to be processed.

Considered weights, values are added to the original data to be processed by the next layers of the Network.

C: Is the second layer of the Network. This layer, comprised of what is formally known as the Nodes of the Network (the circles within the layer), is part of the “hidden layer” within the Network. Considered “hidden” in that the processes it computes are not readily seen by the observer, but is carried out within the computer. Each Node within the layer sums the original values from A, the values introduced by B, and also another numeric value called the Bias. After this computation is completed, the new value is “selected” through a transfer function.

D: Output of C is fed forward to the next hidden layer within the Network. These connections between the layers are known as the “layer weights”. Additional weights or “values” are presented to these new outputs of C.

E: Is considered the third layer of the Network. This layer is also considered part of the hidden layer. The third layer sums the new values from C, values introduced by D, and also another Bias within its Nodes. After this computation is completed, another new value is “selected” through a different transfer function.

F: Is the fourth layer of the Network. This is the final output of the Network.

Completion to this point is called an iteration or epoch through the Network parameters.

G: The output of the Network is compared to the original data set. A learning rule is applied to the data, for example: $\text{Target-Input} = \text{Error}$

H: Updates to the Network weights and biases is backpropagated to the 3rd layer.

I: Updates to the Network weights and biases is backpropagated to the 2nd layer

These updates to the Network weights and biases enable the Network to adjust its parameters to achieve a local or global minimum error within the final output. After the local or global minimum error has been achieved, the training is stopped to test the

generalization of the Network. Generalization is the test of the Networks' ability to use the previous training examples to identify or solve a similar problem when presented with new data. This ability to generalize to different data sets makes the Neural Network a valuable research tool with its efficient error-reducing algorithms.

Researchers in the field of video and image processing have been investigating Neural Networks and the ability to reconstruct image quality. As mentioned previously, visual artifacts within the images appear because of the compression during low bandwidth transmissions. These can include jagged edges, loss of intricate patterns (high spatial frequencies) and visible rings around the object within the image. The application of Neural Networks is to detect the presence of these artifacts and remove them from the compressed images.

G. Qiu and H. P. Lee (1997) conducted a study on the process of eliminating the discontinuity within neighboring blocks (jagged edges) with Neural Networks. By isolating the "nearest neighboring" pixels in the discontinuity, the network was used to correct this anomaly. The Training process was offline (small samples) with an adaptive supervised learning Neural Network. The results achieved from this process were favorable in that they could remove some of the discontinuities from the image. However, they could not completely remove all artifacts from the image.

Random Neural Networks were used in a study by Erol Gelenbe, Hakan Bakircioglu and Taskin Kocak to remove the blocking artifacts that occur from image enlargement. Another focus of their study was that of image fusion. Image fusion was to combine sensory information images and visible images to produce one image. Overall,

their method of combining several layers of random neurons improved image quality. The metric for evaluating image quality in their study was the Mean Squared Error.

Studies were done by Igor N. Aizenberg and Naum N. Aizenberg (1998) on the degradation of satellite transmitted imagery and artifacts within hyper resolution images (image enlargement). Their work utilized multi-valued neurons within the Neural Network to increase the prediction of the highest frequency coefficients (spatial frequencies) within the image. The results show that high frequency coefficients can be predicted and used as a filtering method for transmitted imagery on relatively low compression ratios. Some image degradation was still present in some of the samples at higher rates of compression.

Method

The purpose of this experiment was to assess the quality of images that were post-processed using Neural Networks. The process of training a Neural Network entails collecting, compressing and converting sets of images into raw data. Two matrices of raw data must be generated, the desired outcome (good images) and the initial state of the images (compressed images), in order for the Network to make a “learning rule”. A learning rule is the basis on which the Network forms its parameters for reconstruction. To test the effectiveness of this training process, different images that have been compressed will be presented to the Network for reconstruction. The resultant images will then be assessed by a Mean Squared Error comparison using the original uncompressed image as the baseline.

Experimental Design

Past research has shown that parts of image reconstruction and the prediction of the high spatial frequencies are possible using Neural Networks. Being restricted to an off-the-shelf product for the Neural Network tool, a constant training algorithm will be assumed during the following experiments. Each independent variable will be manipulated to find a basis for building a suitable Neural Network that will reconstruct images. After the results from that training session have been assessed, the most appropriate variable will be kept a constant during the following experiments.

The first variable to be manipulated is the Epochs of the training session. Epochs are the cycles or iterations of the training sets through the entire Network. Weights and Biases within the Nodes are adjusted incrementally after each Epoch, slowly decreasing

the Network error. Conversely, more Epochs directly translate into longer training intervals.

The next independent variable to manipulate is the data requirement introduced within the training sets before testing. Increasing the amount of data in the training sets can decrease the probability for generalization error (which is needed for testing on dissimilar patterns). Two methods for introducing data into the Network consist of Adaptive and Batch Mode training. Adaptive training is the presentation of discrete variables to the Network. The Network must then identify the pattern for each individual change and must “adapt” to the differences between variables. Batch Mode training is the presentation of “concurrent” data to the Network. The result is that the Network identifies the general difference within the variables presented. Both methods should produce a Network that can identify patterns and correct needed changes. In regards of training time between the two choices, Batch Mode training is more efficient.

Following that experiment is the manipulation of the Nodes within the hidden layers of the Network. The number of Nodes is a measure of the computational power within the Neural Network. Probabilities of correctly identifying and reconstructing the image increase proportionately to the power of the Network. The Node is a combination of processes between Weights / Biases and logical functions (a selection process) that make the Network “powerful”. The Network compares the difference between the input and goal and then adjusts the Weights and Biases within the Nodes. This process, known as backpropagation, adjusts the Weights and Biases to overcome the Network error by updating the Weights and Biases in each layer in reverse consecutive order.

The fourth independent variable manipulated is the size of the input vectors to the Network. The size of the vector inputs is the actual number of pixels in the block, which has been converted to raw data. A determination can be made with regards to the ability of the Network to detect and correct patterns of irregularities based upon block sizes. The probability of the Network to generalize should increase proportionately to the size of the image data being presented at one time.

The final independent variable manipulated is the compression ratios within training and testing images. By reducing the compression ratios in a stepwise fashion, objective metrics can be made as to the amount of artifacts that can be present for a noticeable reconstruction to occur.

Apparatus

The equipment used to manipulate images and run the neural net software was a Windows 98 based personal computer; 600 Mhz Pentium III processor. AGP Riva TNT Graphics Blaster, with 782megabytes of PC-100 DRAM.

Software

The software used to compress and convert images from one format to the other was JASC Paint Shop Pro 5.0. JASC Paint Shop Pro can be purchased over the counter at any computer store that sells software. The student edition of Matlab (version 5.3) was the software program that was used to calculate the mathematic properties of the Neural Network. The Neural Network is an additional program that needs to be purchased in conjunction with Matlab. Both Matlab and the Neural Network toolbox are proven scientific software programs that are used in the general public for engineering and

modeling applications. Matlab can be purchased directly through MathWorks or from a University bookstore that sells scientific software.

Materials

Both sample images and test images used for this experiment were acquired from an online data base image archive from the Signal and Image Processing Institute (SIPI) at the University of Southern California computer server. This archive is for public use and can be accessed by the following URL; <http://sipi.usc.edu/>

Procedures

The first step of the process was to simplify the task of handling sets of images within Matlab. Matlab is a command prompt text based software package that was implemented in the course of this study because of the Neural Network functions it can perform. To load data within Matlab you must write a code for your operation. The following code is an example of the type of operation you must write to preprocess the images before running the Neural Network tool. (For the complete code, please refer to appendix A).

```
x=imread('cl.jpg');
```

“x” is now a variable that represents “cl.jpg”. The command ‘imread’ loads ‘cl.jpg’ into Matlab to be processed.

```
x=double(x)/255;
```

This command converts the data type to Floating-Point math. “Floating Point” is the only type of data that is accepted within the Neural Network tool. Values that were 0 to 255 are now between 0 and 1

```
x=rgb2gray(x);
```

This command converts color images to black and white scale, now with values between 0 and 255. (This was a glitch in the software, because the test images were already black and white)

```
x=im2col(x,[16 16],'distinct');
```

This command converts the image into separate but distinct 16x16 pixel areas in a column.

```
x=imread('c1.jpg');
```

```
y=imread('b1.jpg');
```

```
x=double(x)/255;
```

```
y=double(y)/255;
```

```
x=rgb2gray(x);
```

```
y=rgb2gray(y);
```

```
y=im2col(y,[16 16],'distinct');
```

```
x=im2col(x,[16 16],'distinct');
```

The finished codes, inclusive of the commands for the compressed images, are easier to manage in a sub-group for each pair.

After all the images were preprocessed into vectors (columns), they were then classified as either inputs (data to be evaluated) or targets (ideal data to be compared to).

```
p=[x x1 x2 .....];
```

p denotes that the data is the input. The brackets [] classify the data as a matrix.

“x x1...” are the groups of vectors to be evaluated.

```
t=[y y1 y2 .....];
```

“t” denotes that the data is the target. The brackets “[]” classify the data as a matrix.

“y y1...” are the groups of vectors to be compared to. A matrix configuration is needed for the input and the target data to be valid input structures for the Network to recognize.

At this point, a Neural Network must be created to accept the pre-processed images to train with. The following parameters must be in place before a Neural Network is operational and can be trained.

```
net=newff([minmax(p)],[256, 256],{'logsig' 'logsig'},'traincgb');
```

“net=newff” creates a new feedforward backpropagation Network. “[minmax (p)]” is the range of the input values. “[256, 256]” specifies that there are two hidden layers. The first hidden layer contains 256 nodes and the second layer contains 256. Both hidden layers have a specific function, ‘logsig’, which squashes the input and output between the values of 1 and 0. ‘Traincgb’ is a conjugate gradient Network, which is a derivative of the backpropagation Network. Traincgb includes an additional linear search function to increase the Networks probability of convergence at low error rates. The convergence is the disparity and the least minimum gradient change within the data. The addition of a search function also allows the Network to converge at a faster rate when running large data sets.

An overview of the overall method process is illustrated in the following example:

Figure 10. Overall Method Process

After the image is post-processed through the Neural Network. (the reconstructed image) analysis of the image quality will be assessed using the Mean Squared Error.

Analysis of data

The industry standard for objectively evaluating image quality is the Peak Signal to Noise Ratio (PSNR). The Peak Signal to Noise Ratio measures the absolute differences in decibels between an original image and the subsequent processed image. However, the Mean Squared Error (MSE), is equivalent to Peak Signal to Noise Ratio and has been used for the objective evaluation in this study. The Mean Squared Error is a statistical tool that measures the difference of image quality by Standard Deviations rather than decibels.

The Mean Squared Error is the mean of the sum of the squares of the differences between the values of pixels in two images.

$$MSE = 1 / n * (\text{Sum}(i)(j) |P(i)(j) - Q(i)(j)|^2)$$

Where P and Q are two images. i and j are the horizontal and vertical locations of a pixel. P(i)(j) is the value of the pixel at location (i)(j). n is the total number of pixels in the image.

(McGowan, 1999 as interpreted from Netravali, Haskell,

Puri 1996).

Both methods, PSNR and MSE, are generally accepted methods for objective evaluation of image quality.

Results

The results from the original hypothesis, that Neural Networks can reconstruct compressed images to improve the end user's perception of image quality, are displayed on the following pages. There are three parts to each result section.

The first page relates the Network parameters that were used for that specific test and the results from those variables. The most important feature to analyze from this page is the Mean Square Error compared with the original image.

The second page is the training record that was produced during the initial Network training. The blue line indicates the actual training error (Mean Squared Error) vs the Black line, which is the desired goal of the training exercise. The goal is a pre specified error rate that translates to an acceptable level of image reconstruction. The final overall Mean Squared Error is displayed at the top of the graph to show the last reduction at the end of the training. The final number of epochs reached during the training is displayed at the bottom.

The final page shows a side-by-side comparison of the Network Output image with the Original image and the Network input image, which is the compressed image.

Network Parameters	Independent Variable	Image	Pre Processed Mean Squared Error from Original	Network Images Mean Squared Error
Network Used Traincgb Transfer Functions Logsig Purelin Learning Rate 10.5 Images used 30 Nodes per hidden layer 256x256 Performance function Mean Squared Error Size of sample vectors 16x16 Compression ratio 95%	200 Epochs		Goal .0005 For training set	
		Airplane	.0015	.0016
		Tank	.0018	.0019
		Wall	.0097	.0102
	500 Epochs		Goal .0005 For training set	
		Airplane	.0015	.0016
		Tank	.0018	.0019
		Wall	.0097	.0102
	1000 Epochs		Goal .0005 For training set	
		Airplane	.0015	.0015
		Tank	.0018	.0016
		Wall	.0097	.0123

Table 1. Experiment 1.2.3 Mean Squared Error results from increasing/decreasing training iterations.

Graph 1. 1. The final Mean Squared Error achieved during Experiment 1. The horizontal axis is the training epochs and the vertical axis is the error rate. Epochs were the independent variable for this experiment.

Figure 11. Image results from Experiment 1. Comparative analysis of the final Network Output with the Original Image and Compressed Images.

Graph 1. 2. The final Mean Squared Error achieved during Experiment 2. The horizontal axis is the training epochs and the vertical axis is the error rate. Epochs were the independent variable for this experiment

Figure 12. Image results from Experiment 2.

Graph 1.3. The final Mean Squared Error achieved during Experiment 3. The horizontal axis is the training epochs and the vertical axis is the error rate. Epochs were the independent variable for this experiment.

Figure 13. Image results from Experiment 3.

Network Parameters	Independent Variable	Image	Pre Processed Mean Squared Error from Original	Post Processed Mean Squared Error from Original
Network Used Traincgb Transfer Functions Logsig Purelin Learning Rate 10.5 Epochs 200 Nodes per hidden layer 256x256 Performance function Mean Squared Error Size of sample vectors 16x16 Compression ratio 95%	60 Images		Goal .0005 For training set	
		Airplane	.0015	.0018
		Tank	.0018	.0017
		Wall	.0097	.0104
	30 Images		Goal .0005 For training set	
		Airplane	.0015	.0016
		Tank	.0018	.0019
		Wall	.0097	.0102
	20 Images		Goal .0005 For training set	
		Airplane	.0015	.0016
		Tank	.0018	.0017
		Wall	.0097	.0119

Table 2. Experiments 4,5,6 Mean Squared Error results from increasing/decreasing training sets of data.

Graph 2. 1. The final Mean Squared Error achieved during Experiment 4. The horizontal axis is the training epochs and the vertical axis is the error rate. Amount of data within the training sets was the independent variable for this experiment.

Figure 14. Image results from Experiment 4.

Graph 2. 2. The final Mean Squared Error achieved during Experiment 5. The horizontal axis is the training epochs and the vertical axis is the error rate. Amount of data within the training sets was the independent variable for this experiment.

Figure 15. Image results from Experiment 5.

Graph 2. 3. The final Mean Squared Error achieved during Experiment 6. The horizontal axis is the training epochs and the vertical axis is the error rate. Amount of data within the training sets was the independent variable for this experiment.

Figure 16. Image results from Experiment 6.

Network Parameters	Independent Variable	Image	Pre Processed Mean Squared Error from Original	Post Processed Mean Squared Error from Original
Network Used Traincgb Transfer Functions Logsig Purelin Learning Rate 10.5 Images used 20 Epochs 200 Performance function Mean Squared Error Size of sample vectors 16x16 Compression ratio 95%	Nodes per hidden layer 256x256		Goal .0005 For training set	
		Airplane	.0015	.0016
		Tank	.0018	.0017
		Wall	.0097	.0119
	Nodes per hidden layer 512x256		Goal .0005 For training set	
		Airplane	.0015	.0016
		Tank	.0018	.0017
		Wall	.0097	.0118
	Nodes per hidden layer 768x256		Goal .0005 For training set	
		Airplane	.0015	.0016
		Tank	.0018	.0017
		Wall	.0097	.0121

Table 3. Experiments 7,8,9 Mean Squared Error results from increasing/decreasing Nodes within the Network.

Graph 3. 1. The final Mean Squared Error achieved during Experiment 7. The horizontal axis is the training epochs and the vertical axis is the error rate. Nodes within the Network were the independent variable for this experiment.

Figure 17. Image results from Experiment 7.

Graph 3. 2. The final Mean Squared Error achieved during Experiment 8. The horizontal axis is the training epochs and the vertical axis is the error rate. Nodes within the Network were the independent variable for this experiment.

Figure 18. Image results from Experiment 8.

Graph 3. 3. The final Mean Squared Error achieved during Experiment 9. The horizontal axis is the training epochs and the vertical axis is the error rate. Nodes within the Network were the independent variable for this experiment.

Figure 19. Image results from Experiment 9.

Network Parameters	Independent Variable	Image	Pre Processed Mean Squared Error from Original	Post Processed Mean Squared Error from Original
Network Used Traincgb Transfer Functions Logsig Purelin Learning Rate 10.5 Images used 20 Nodes per hidden layer 512x256 Epochs 200 Performance function Mean Squared Error Compression ratio 95%	Size of sample vectors 8x8		Goal .0005 For training set	
		Airplane	.0016	.0015
		Tank	.0019	.0018
		Wall	.0105	.0112
			*Nodes in 2 nd hidden layer was 64	
	Size of sample vectors 16x16		Goal .0005 For training set	
		Airplane	.0015	.0016
		Tank	.0018	.0017
		Wall	.0097	.0119
	Size of sample vectors 32x32		Goal .0005 For training set	
		Airplane	.0013	.0023
		Tank	.0015	.0018
		Wall	.0084	.0199
			*Nodes in 2 nd hidden layer was 1024	

Table 4. Experiments 10, 11, 12 Mean Squared Error results from increasing/decreasing blocks of data from the image. (resolution)

Graph 4. 1. The final Mean Squared Error achieved during Experiment 10. The horizontal axis is the training epochs and the vertical axis is the error rate. Blocks of data (size of the vectors) were the independent variable for this experiment.

Figure 20. Image results from Experiment 10.

Graph 4.2. The final Mean Squared Error achieved during Experiment 11. The horizontal axis is the training epochs and the vertical axis is the error rate. Blocks of data (size of the vectors) were the independent variable for this experiment.

Figure 21. Image results from Experiment 11.

Graph 4. 3. The final Mean Squared Error achieved during Experiment 12. The horizontal axis is the training epochs and the vertical axis is the error rate. Blocks of data (size of the vectors) were the independent variable for this experiment.

Figure 22. Image results from Experiment 12.

Network Parameters	Independent Variable	Image	Pre Processed Mean Squared Error from Original	Post Processed Mean Squared Error from Original
Network Used Traincgb Transfer Functions Logsig Purelin Learning Rate 10.5 Images used 20 Nodes per hidden layer 256x256 Performance function Mean Squared Error Size of sample vectors 32x32	50% Compression ratio		Goal .0005 For training set	
		Airplane	1.2648e-004	7.8496e-004
		Tank	3.0647e-004	6.6764e-004
		Wall	.0020	.0102
	75% Compression ratio		Goal .0005 For training set	
		Airplane	2.5253e-004	.0012
		Tank	4.8592e-004	9.9308e-004
		Wall	.0032	.0133

Table 5. Experiments 13 & 14 Mean Squared Error results from increasing/decreasing compression ratios.

Graph 5. 1. The final Mean Squared Error achieved during Experiment 13. The horizontal axis is the training epochs and the vertical axis is the error rate. Compression ratios were the independent variable for this experiment.

Figure 23. Image results from Experiment 13.

Graph 5.2. The final Mean Squared Error achieved during Experiment 14. The horizontal axis is the training epochs and the vertical axis is the error rate. Compression ratios were the independent variable for this experiment.

Figure 24. Image results from Experiment 14.

Discussion

Hypothesis 1: Increasing the amount of epochs used during the training session improves the Network Generalization. Epochs are directly proportional to time and by increasing the epochs in training conversely increases the time to train.

Experiments 1, 2, and 3 were programmed for 200, 500 and 1000 epochs respectively during the initial training of the Network. Table 1 is the comparison of the Mean Squared Error for the Network Inputs and Outputs. Increasing the amount of epochs during training did not produce a significant variation of the Mean Squared Error between variables. Therefore, the trade off between epochs and change vs. time to run more data does not warrant more than 200 epochs for the following experiments.

Hypothesis 2: Training the Network with more images during the training sets would improve the Network generalization. The independent variable in experiments 4, 5 and 6 was to increase the amount of concurrent data within the training sets by 60, 30 and 20 images respectively. Table 2 compares the Mean Squared Error for the Network Inputs and Outputs of each of these results. It was found that by increasing the amount of images the Network trained upon does not prove to be statistically different across all test images. The lack of statistically significant results indicate that the independent variable under examination, does not effect the network generalization and can be minimized to 20 images during the following experiments.

Hypothesis 3: Increasing the Nodes in the Network improves the Network generalization. Nodes within the Network layers are directly related to the computational power of the Network. In experiments 7, 8, and 9, the first hidden layer of Nodes in the Network was increased by 256. However, the second layer of Nodes was kept at a

constant figure of 256. This was a limitation of the software and the final layer was to be the same amount as the input vector.

Table 3 shows the comparison of the Mean Squared Error for the Network Inputs and Outputs for experiments 7, 8 and 9. The results indicate the variation between Network Outputs and Inputs does not change significantly enough to determine a noticeable difference by increasing the Nodes in the Network. No statistical difference can be reported from these tests. The lack of statistically significant results indicate that the independent variable, under examination does not effect the network generalization and can be minimized to a vector of 256 during the following experiments.

Hypothesis 4: Increasing the portion size of the image (block size) to be analyzed by the Network increases the probability of the Network generalization. Larger blocks of data increase the pattern continuity within the image. The pattern continuity increases the information about the image, which allows the network to generate a learning rule based on the information available.

Experiments 10, 11, and 12 increased the block sizes by 8x8, 16x16 and 32x32 respectively. Table 4 shows the comparison of the Mean Squared Error for the Network Inputs and Outputs. Objectively, the results show that the variation of the Mean Squared Error between Network Outputs and Inputs does not change significantly. However, subjectively, to the unaided eye, experiment 12 shows a slight improvement in reconstructing the image over the previous experiments. The discontinuity of the edges has been changed within the Output images of the Network. Another reconstructed feature was that the small blocking patterns had been removed from the images. This, however, was replaced with larger visible nearest neighbor discontinuities. Also, more

artifacts had been added within each larger block. Known as Dithering, small spots appear throughout the entire image and are not specific to the actual objects within the picture. These new anomalies increased the overall Mean Square Error of the Network Output despite the reconstruction that occurred. In general, the Network Output for experiment 12 has shown more improvement than all of the prior experiments. Therefore, the parameters of using 32x32 in further experiments will be continued.

Hypothesis 5: Network generalization improves with decreasing the ratio of image compression. Artifacts that appear within the image are dependent upon the ratio of compression that the image has under gone. The presence of fewer artifacts might increase their different salient attributes from the attributes of the objects within the image. Previous baselines during all experiments had been at the 95% compression ratio. The 95% compression ratio, was an optimal setting to reduce image quality and frame rates that would be needed over POTS transmission.

Experiments 13. and 14, decreased the compression ratio of the image to 75% and 50% respectively. Table 5 is the comparison of the Mean Squared Error for the Network Inputs and Outputs. The results are similar to experiment 12, in their shared properties. Objectively the Mean Squared Error increased while subjectively there was some image reconstruction. First, the image did reduce the ringing and small blocking artifacts that were present, but continued to add small spots to the whole picture. While compression ratios manipulate the amount of artifacts that are present within the image, there seems to be no effect to the reconstruction process with Neural Networks.

Summary

An important aspect of the Neural Network is to generalize the learning parameters of the previous training sets to other unrelated images that would be presented to the Network. Training the Neural Network to reconstruct images that manipulate the Human visual system is extremely complex. The difficulty of the task is to identify the correct parameters within the training sets that will effect the generalization capabilities of the Network. In addition, different types of training algorithms are more efficient at certain mathematical problems than the Network used for this thesis. The specific Network that was used during testing was restricted by the computational limits that were imposed by the software and hardware capabilities. Lastly, it can be debated as to the appropriate and inclusive nature of the training data and whether it would be representative of all image parameters. Given the enormous amount of parameters that are involved in such a process, the likelihood is that a combination of several factors will be needed for reconstruction. Training a Neural Network to predict and restore the highest frequency coefficient removed from the image is possible.

Recommendations for further studies

Several possible reasons exist to explain the lack of significant results, which were attained from these experiments. All results were assessed by using the Mean Squared Error for the objective measure of differences between two images. The main constraint of this study was the restrictions imposed by a commercial product. Most researchers in the field of Neural Networks generate a Network with a programming language. This allows the researcher more flexibility and to isolate the exact parameters under examination. Furthermore, the technology limitations of computers to run the

Network application must be capable of handling larger amounts of data with faster processing speeds. At the time of testing, the fastest speed for a desktop computer was a 600MHz p.c. Increasing the processor speeds and memory requirements of RAM and the hard drive, would allow the evaluation of other Network Algorithms that are more efficient for this type of problem, such as the Levenberg-Marquardt algorithm.

Another possible reason for the lack of significant results was the way in which data was presented in the Network. Batch mode training was used during these experiments because of the commercial software parameters. These parameters only accepted large amounts of data to be represented as one concurrent matrix, not as a cell array. The use of cell arrays would enable the Network to process the information as discrete variables (adaptation) rather than a single variable (batch mode). Using Adaptation should return a local parameter rather than a global one, as is the case with Batch mode training.

Further studies should investigate these variables with the inclusion of the block sizes. The most noticeable differences found in the test images, occurred from the 32x32 block sizes. This was the only parameter on this network that reconstructed any part of the image. Although the Mean Squared Error increased due to a dithering effect, other artifacts such as jagged edges and blocks were removed. Layers of Competitive Nodes in the Network might decrease the dithering effect that was present. A Competitive Node will only select the only correct variable that passes through the Nodes of the Network. Implementing this feature within the Network would decrease the final error ratio and might improve the image quality.

Conclusion

Overall the lack of significant results from the independent variables indicates that they do not have an effect as to the reconstruction process with Neural Networks. There is one exception to this and that is the variable of block sizes. Increasing the block size to 32x32 within the neural Network seems to reconstruct some of the properties of the image. A confounding factor when using this independent variable is the dithering effects that occur from this process. The dithering effect might be attributed to the actual underlying properties of the Neural Network. Being that the Neural Network is based on probabilities, the spots are the best “guess” as to the image reconstruction. This would leave the possibility that the problem is inherent to the Network settings and would need to be investigated further.

It should also be noted that the topic of this thesis was to address the effects on the human perception with image reconstruction. The engineering aspect of this thesis was designed to affect the utility as well as the usability of videoconferencing. In the case of image reconstruction, the technological limitations were the limiting factor that affected the overall perception and performance of the end user. By addressing the engineering aspects of this technology, a change to the Human Performance and Perception, not just the interaction, can be made. From this viewpoint, the topic of this thesis was approached.

To test that theory, several testing procedures and software packages, such as NASA’s DCTune, were to be used during the course of this study to evaluate this criterion. However, due to the lack of significant results from the objective measurements and upon critical review from experts within the field, the subjective

measurement testing became unnecessary. Had there been sufficient results from the objective evaluations, the subjective testing would have been necessary to show the improvement of image quality to the human viewer. It is firmly believed that the continuance of this project with the recommended improvements will constitute the implementation of the psychometric evaluations.

References

Aizenberg, I. N., & Aizenberg, N.N., (1998). Application of the Neural Networks Based on Multi-Valued Neurons in Image Processing and Recognition in *Applications of Artificial Neural Networks in Image Processing III*, Nasser M. Nasrabadi, Aggelos K. Katsaggelos, Editors, Proceedings of SPIE Vol. 3307, pp. 88-97 (1998).

Apteker, R. T., Fisher, J. A., Kisimov, V. S., & Neishlos, H. (1995). Video acceptability and frame rate. IEEE Multimedia. 32-40.

Angiolillo, J.S., Blanchard, H.E., Israelski, E.W., & Mane', A. (1997). Technology constraints of video mediated communication. In Finn, K.E., Sellen, A.J., & Wilbur, S.B., (Eds.). Video mediated communication (pp.51-73). New Jersey: Erlbaum.

Eberhart, R., Simpson, P., & Dobbins, R. (1996). Computational intelligence PC tools: Chestnut Hill, MA: Academic Press Inc.

Gelenbe, E., Bakircioglu, K., & Kocak, T., Image Processing with the Random Neural Network in *Applications of Artificial Neural Networks in Image Processing III*, Nasser M. Nasrabadi, Aggelos K. Katsaggelos, Editors, Proceedings of SPIE Vol. 3307, pp.38-49 (1998).

Geszti. T. (1990). Physical models of neural networks. Teaneck, NJ: World Scientific.

Haskell. B. G., Puri, A., & Netravali, A. N. (1996). Digital video: An introduction to mpeg-2: New York: Chapman & Hall.

Hilton. J., & Jacobi, P. (1986). Straight talk about videoconferencing. New York: Prentice-Hall, Inc.

Kalat. J. (1992). Biological Psychology. (4th ed.). Belmont, Ca: Wadsworth Publishing Company.

Kies, J. K. (1997). A psychophysical evaluation of frame rate in desktop video conferencing. Proceedings of the Human Factors and Ergonomics Society 4th annual meeting. 310-314.

Kim. S. H. (1994). Learning and coordination: Enhancing agent performance through distributed decision making. Norwell, MA: Kluwer Academic Publishers.

Klien, S. B. (2000). Biological Psychology. (1sted.). Upper Saddle River, NJ: Prentice Hall Inc.

Pennebaker, W., & Mitchell, J. C. (1993). JPEG: Still Image Data Compression Standard: New York : Van Nostrand Reinhold.
pp. 38-49.

Qiu, G.. & Lee, H. P. (1997). JPEG Image Enhancement Based on Adaptive Learning. SPIE 3309, pp. 260-270. Derby DE22 1GB, United Kingdom.

Rosen, E., (1996). Personal Videoconferencing: Greenwich, CT: Manning Publications Co.

Sekuler, R., & Blake, R. (1994). Perception. (3rd ed.). New York: McGraw-Hill.

Taylor, J. G. (1993). The promise of Neural Networks. New York: Springer-Verlag.

Trowt-Bayard, T.. & Wilcox, J. (1997). Videoconferencing & Interactive media: The whole picture. New York: Flaitron Publishing, Inc

Weiss, S. M., & Kulikowski, C.A. (1991). Computer systems that learn: Classification and prediction methods from statistics, neural nets, machine learning, and expert systems. San Mateo, CA: Morgan Kaufmann.

Whittaker, S., & O'Conaill, B. (1997). The role of vision in face-to-face and mediated communication. In Finn, K.E., Sellen, A.J., & Wilbur, S.B., (Eds.), Video mediated communication (pp.23-46). New Jersey: Erlbaum.

Winston, B., (1998). Media Technology and Society: A History: from the Telegraph to the Internet. New York : Routledge.

Appendix A

```
x=imread('c1.jpg');
y=imread('b1.jpg');
x=double(x)/255;
y=double(y)/255;
x=rgb2gray(x);
y=rgb2gray(y);
y=im2col(y,[32 32],'distinct');
x=im2col(x,[32 32],'distinct');
x2=imread('c2.jpg');
y2=imread('b2.jpg');
x2=double(x2)/255;
y2=double(y2)/255;
x2=rgb2gray(x2);
y2=rgb2gray(y2);
y2=im2col(y2,[32 32],'distinct');
x2=im2col(x2,[32 32],'distinct');
x3=imread('c3.jpg');
y3=imread('b3.jpg');
x3=double(x3)/255;
y3=double(y3)/255;
x3=rgb2gray(x3);
y3=rgb2gray(y3);
y3=im2col(y3,[32 32],'distinct');
x3=im2col(x3,[32 32],'distinct');
x4=imread('c4.jpg');
y4=imread('b4.jpg');
x4=double(x4)/255;
y4=double(y4)/255;
x4=rgb2gray(x4);
y4=rgb2gray(y4);
y4=im2col(y4,[32 32],'distinct');
x4=im2col(x4,[32 32],'distinct');
x5=imread('c5.jpg');
y5=imread('b5.jpg');
x5=double(x5)/255;
y5=double(y5)/255;
x5=rgb2gray(x5);
y5=rgb2gray(y5);
y5=im2col(y5,[32 32],'distinct');
x5=im2col(x5,[32 32],'distinct');
x6=imread('c6.jpg');
```

```

y6=imread('b6.jpg');
x6=double(x6)/255;
y6=double(y6)/255;
x6=rgb2gray(x6);
y6=rgb2gray(y6);
y6=im2col(y6,[32 32],'distinct');
x6=im2col(x6,[32 32],'distinct');
x7=imread('c7.jpg');
y7=imread('b7.jpg');
x7=double(x7)/255;
y7=double(y7)/255;
x7=rgb2gray(x7);
y7=rgb2gray(y7);
y7=im2col(y7,[32 32],'distinct');
x7=im2col(x7,[32 32],'distinct');
x8=imread('c8.jpg');
y8=imread('b8.jpg');
x8=double(x8)/255;
y8=double(y8)/255;
x8=rgb2gray(x8);
y8=rgb2gray(y8);
y8=im2col(y8,[32 32],'distinct');
x8=im2col(x8,[32 32],'distinct');
x9=imread('c9.jpg');
y9=imread('b9.jpg');
x9=double(x9)/255;
y9=double(y9)/255;
x9=rgb2gray(x9);
y9=rgb2gray(y9);
y9=im2col(y9,[32 32],'distinct');
x9=im2col(x9,[32 32],'distinct');
x10=imread('c10.jpg');
y10=imread('b10.jpg');
x10=double(x10)/255;
y10=double(y10)/255;
x10=rgb2gray(x10);
y10=rgb2gray(y10);
y10=im2col(y10,[32 32],'distinct');
x10=im2col(x10,[32 32],'distinct');
x11=imread('c11.jpg');
y11=imread('b11.jpg');
x11=double(x11)/255;
y11=double(y11)/255;
x11=rgb2gray(x11);
y11=rgb2gray(y11);
y11=im2col(y11,[16 16],'distinct');

```



```

x11=im2col(x11,[16 16],'distinct');
x12=imread('c12.jpg');
y12=imread('b12.jpg');
x12=double(x12)/255;
y12=double(y12)/255;
x12=rgb2gray(x12);
y12=rgb2gray(y12);
y12=im2col(y12,[16 16],'distinct');
x12=im2col(x12,[16 16],'distinct');
x13=imread('c13.jpg');
y13=imread('b13.jpg');
x13=double(x13)/255;
y13=double(y13)/255;
x13=rgb2gray(x13);
y13=rgb2gray(y13);
y13=im2col(y13,[16 16],'distinct');
x13=im2col(x13,[16 16],'distinct');
x14=imread('c14.jpg');
y14=imread('b14.jpg');
x14=double(x14)/255;
y14=double(y14)/255;
x14=rgb2gray(x14);
y14=rgb2gray(y14);
y14=im2col(y14,[16 16],'distinct');
x14=im2col(x14,[16 16],'distinct');
x15=imread('c15.jpg');
y15=imread('b15.jpg');
x15=double(x15)/255;
y15=double(y15)/255;
x15=rgb2gray(x15);
y15=rgb2gray(y15);
y15=im2col(y15,[16 16],'distinct');
x15=im2col(x15,[16 16],'distinct');
x16=imread('c16.jpg');
y16=imread('b16.jpg');
x16=double(x16)/255;
y16=double(y16)/255;
x16=rgb2gray(x16);
y16=rgb2gray(y16);
y16=im2col(y16,[16 16],'distinct');
x16=im2col(x16,[16 16],'distinct');
x17=imread('c17.jpg');
y17=imread('b17.jpg');
x17=double(x17)/255;
y17=double(y17)/255;
x17=rgb2gray(x17);

```

```

y17=rgb2gray(y17);
y17=im2col(y17,[16 16],'distinct');
x17=im2col(x17,[16 16],'distinct');
x18=imread('c18.jpg');
y18=imread('b18.jpg');
x18=double(x18)/255;
y18=double(y18)/255;
x18=rgb2gray(x18);
y18=rgb2gray(y18);
y18=im2col(y18,[16 16],'distinct');
x18=im2col(x18,[16 16],'distinct');
x19=imread('c19.jpg');
y19=imread('b19.jpg');
x19=double(x19)/255;
y19=double(y19)/255;
x19=rgb2gray(x19);
y19=rgb2gray(y19);
y19=im2col(y19,[16 16],'distinct');
x19=im2col(x19,[16 16],'distinct');
x20=imread('c20.jpg');
y20=imread('b20.jpg');
x20=double(x20)/255;
y20=double(y20)/255;
x20=rgb2gray(x20);
y20=rgb2gray(y20);
y20=im2col(y20,[16 16],'distinct');
x20=im2col(x20,[16 16],'distinct');

x25=imread('c25.jpg');
y25=imread('b25.jpg');
x25=double(x25)/255;
y25=double(y25)/255;

y25=im2col(y25,[16 16],'distinct');
x25=im2col(x25,[16 16],'distinct');
x26=imread('c26.jpg');
y26=imread('b26.jpg');
x26=double(x26)/255;
y26=double(y26)/255;

y26=im2col(y26,[16 16],'distinct');
x26=im2col(x26,[16 16],'distinct');
x27=imread('c27.jpg');
y27=imread('b27.jpg');

```

```
x27=double(x27)/255;
y27=double(y27)/255;
```

```
y27=im2col(y27,[16 16],'distinct');
x27=im2col(x27,[16 16],'distinct');
x28=imread('c28.jpg');
y28=imread('b28.jpg');
x28=double(x28)/255;
y28=double(y28)/255;
```

```
y28=im2col(y28,[16 16],'distinct');
x28=im2col(x28,[16 16],'distinct');
x29=imread('c29.jpg');
y29=imread('b29.jpg');
x29=double(x29)/255;
y29=double(y29)/255;
```

```
y29=im2col(y29,[16 16],'distinct');
x29=im2col(x29,[16 16],'distinct');
x30=imread('c30.jpg');
y30=imread('b30.jpg');
x30=double(x30)/255;
y30=double(y30)/255;
```

```
y30=im2col(y30,[16 16],'distinct');
x30=im2col(x30,[16 16],'distinct');
```

```
x31=imread('c31.jpg');
y31=imread('b31.jpg');
x31=double(x31)/255;
y31=double(y31)/255;
```

```
y31=im2col(y31,[16 16],'distinct');
x31=im2col(x31,[16 16],'distinct');
x32=imread('c32.jpg');
y32=imread('b32.jpg');
x32=double(x32)/255;
y32=double(y32)/255;
```

```
y32=im2col(y32,[16 16],'distinct');
x32=im2col(x32,[16 16],'distinct');
```

```
x33=imread('c33.jpg');  
y33=imread('b33.jpg');  
x33=double(x33)/255;  
y33=double(y33)/255;
```

```
y33=im2col(y33,[16 16],'distinct');  
x33=im2col(x33,[16 16],'distinct');  
x34=imread('c34.jpg');  
y34=imread('b34.jpg');  
x34=double(x34)/255;  
y34=double(y34)/255;
```

```
y34=im2col(y34,[16 16],'distinct');  
x34=im2col(x34,[16 16],'distinct');  
x35=imread('c35.jpg');  
y35=imread('b35.jpg');  
x35=double(x35)/255;  
y35=double(y35)/255;
```

```
y35=im2col(y35,[16 16],'distinct');  
x35=im2col(x35,[16 16],'distinct');  
x36=imread('c36.jpg');  
y36=imread('b36.jpg');  
x36=double(x36)/255;  
y36=double(y36)/255;
```

```
y36=im2col(y36,[16 16],'distinct');  
x36=im2col(x36,[16 16],'distinct');  
x37=imread('c37.jpg');  
y37=imread('b37.jpg');  
x37=double(x37)/255;  
y37=double(y37)/255;
```

```
y37=im2col(y37,[16 16],'distinct');  
x37=im2col(x37,[16 16],'distinct');  
x38=imread('c38.jpg');  
y38=imread('b38.jpg');  
x38=double(x38)/255;  
y38=double(y38)/255;
```

```

y38=im2col(y38,[16 16],'distinct');
x38=im2col(x38,[16 16],'distinct');
x39=imread('c39.jpg');
y39=imread('b39.jpg');
x39=double(x39)/255;
y39=double(y39)/255;

```

```

y39=im2col(y39,[16 16],'distinct');
x39=im2col(x39,[16 16],'distinct');
x40=imread('c40.jpg');
y40=imread('b40.jpg');
x40=double(x40)/255;
y40=double(y40)/255;

```

```

y40=im2col(y40,[16 16],'distinct');
x40=im2col(x40,[16 16],'distinct');
x41=imread('c41.jpg');
y41=imread('b41.jpg');
x41=double(x41)/255;
y41=double(y41)/255;

```

```

y41=im2col(y41,[16 16],'distinct');
x41=im2col(x41,[16 16],'distinct');
x42=imread('c42.jpg');
y42=imread('b42.jpg');
x42=double(x42)/255;
y42=double(y42)/255;

```

```

y42=im2col(y42,[16 16],'distinct');
x42=im2col(x42,[16 16],'distinct');
x43=imread('c43.jpg');
y43=imread('b43.jpg');
x43=double(x43)/255;
y43=double(y43)/255;

```

```

y43=im2col(y43,[16 16],'distinct');
x43=im2col(x43,[16 16],'distinct');
x44=imread('c44.jpg');
y44=imread('b44.jpg');
x44=double(x44)/255;
y44=double(y44)/255;

```

```

y44=im2col(y44,[16 16],'distinct');
x44=im2col(x44,[16 16],'distinct');
x45=imread('c45.jpg');
y45=imread('b45.jpg');
x45=double(x45)/255;
y45=double(y45)/255;

```

```

y45=im2col(y45,[16 16],'distinct');
x45=im2col(x45,[16 16],'distinct');
x46=imread('c46.jpg');
y46=imread('b46.jpg');
x46=double(x46)/255;
y46=double(y46)/255;

```

```

y46=im2col(y46,[16 16],'distinct');
x46=im2col(x46,[16 16],'distinct');
x47=imread('c47.jpg');
y47=imread('b47.jpg');
x47=double(x47)/255;
y47=double(y47)/255;

```

```

y47=im2col(y47,[16 16],'distinct');
x47=im2col(x47,[16 16],'distinct');
x48=imread('c48.jpg');
y48=imread('b48.jpg');
x48=double(x48)/255;
y48=double(y48)/255;

```

```

y48=im2col(y48,[16 16],'distinct');
x48=im2col(x48,[16 16],'distinct');
x49=imread('c49.jpg');
y49=imread('b49.jpg');
x49=double(x49)/255;
y49=double(y49)/255;

```

```

y49=im2col(y49,[16 16],'distinct');
x49=im2col(x49,[16 16],'distinct');
x50=imread('c50.jpg');
y50=imread('b50.jpg');

```

```
x50=double(x50)/255;
y50=double(y50)/255;
```

```
y50=im2col(y50,[16 16],'distinct');
x50=im2col(x50,[16 16],'distinct');
x51=imread('c51.jpg');
y51=imread('b51.jpg');
x51=double(x51)/255;
y51=double(y51)/255;
```

```
y51=im2col(y51,[16 16],'distinct');
x51=im2col(x51,[16 16],'distinct');
x52=imread('c52.jpg');
y52=imread('b52.jpg');
x52=double(x52)/255;
y52=double(y52)/255;
```

```
y52=im2col(y52,[16 16],'distinct');
x52=im2col(x52,[16 16],'distinct');
x53=imread('c53.jpg');
y53=imread('b53.jpg');
x53=double(x53)/255;
y53=double(y53)/255;
```

```
y53=im2col(y53,[16 16],'distinct');
x53=im2col(x53,[16 16],'distinct');
x54=imread('c54.jpg');
y54=imread('b54.jpg');
x54=double(x54)/255;
y54=double(y54)/255;
```

```
y54=im2col(y54,[16 16],'distinct');
x54=im2col(x54,[16 16],'distinct');
x55=imread('c55.jpg');
y55=imread('b55.jpg');
x55=double(x55)/255;
y55=double(y55)/255;
y55=im2col(y55,[16 16],'distinct');
x55=im2col(x55,[16 16],'distinct');
```

```
x65=imread('c65.jpg');
```

```

y65=imread('b65.jpg');
x65=double(x65)/255;
y65=double(y65)/255;
x65=rgb2gray(x65);
y65=rgb2gray(y65);
y65=im2col(y65,[16 16],'distinct');
x65=im2col(x65,[16 16],'distinct');
x66=imread('c66.jpg');
y66=imread('b66.jpg');
x66=double(x66)/255;
y66=double(y66)/255;
x66=rgb2gray(x66);
y66=rgb2gray(y66);
y66=im2col(y66,[16 16],'distinct');
x66=im2col(x66,[16 16],'distinct');
x67=imread('c67.jpg');
y67=imread('b67.jpg');
x67=double(x67)/255;
y67=double(y67)/255;
x67=rgb2gray(x67);
y67=rgb2gray(y67);
y67=im2col(y67,[16 16],'distinct');
x67=im2col(x67,[16 16],'distinct');
x68=imread('c68.jpg');
y68=imread('b68.jpg');
x68=double(x68)/255;
y68=double(y68)/255;
x68=rgb2gray(x68);
y68=rgb2gray(y68);
y68=im2col(y68,[16 16],'distinct');
x68=im2col(x68,[16 16],'distinct');
x69=imread('c69.jpg');
y69=imread('b69.jpg');
x69=double(x69)/255;
y69=double(y69)/255;
x69=rgb2gray(x69);
y69=rgb2gray(y69);
y69=im2col(y69,[16 16],'distinct');
x69=im2col(x69,[16 16],'distinct');
x70=imread('c70.jpg');
y70=imread('b70.jpg');
x70=double(x70)/255;
y70=double(y70)/255;
x70=rgb2gray(x70);
y70=rgb2gray(y70);
y70=im2col(y70,[16 16],'distinct');

```



```

x70=im2col(x70,[16 16],'distinct');
x71=imread('c71.jpg');
y71=imread('b71.jpg');
x71=double(x71)/255;
y71=double(y71)/255;
x71=rgb2gray(x71);
y71=rgb2gray(y71);
y71=im2col(y71,[16 16],'distinct');
x71=im2col(x71,[16 16],'distinct');
x72=imread('c72.jpg');
y72=imread('b72.jpg');
x72=double(x72)/255;
y72=double(y72)/255;
x72=rgb2gray(x72);
y72=rgb2gray(y72);
y72=im2col(y72,[16 16],'distinct');
x72=im2col(x72,[16 16],'distinct');
x73=imread('c73.jpg');
y73=imread('b73.jpg');
x73=double(x73)/255;
y73=double(y73)/255;
x73=rgb2gray(x73);
y73=rgb2gray(y73);
y73=im2col(y73,[16 16],'distinct');
x73=im2col(x73,[16 16],'distinct');
x74=imread('c74.jpg');
y74=imread('b74.jpg');
x74=double(x74)/255;
y74=double(y74)/255;
x74=rgb2gray(x74);
y74=rgb2gray(y74);
y74=im2col(y74,[16 16],'distinct');
x74=im2col(x74,[16 16],'distinct');
x75=imread('c75.jpg');
y75=imread('b75.jpg');
x75=double(x75)/255;
y75=double(y75)/255;
x75=rgb2gray(x75);
y75=rgb2gray(y75);
y75=im2col(y75,[16 16],'distinct');
x75=im2col(x75,[16 16],'distinct');
x76=imread('c76.jpg');
y76=imread('b76.jpg');
x76=double(x76)/255;
y76=double(y76)/255;
x76=rgb2gray(x76);

```

```

y76=rgb2gray(y76);
y76=im2col(y76,[16 16],'distinct');
x76=im2col(x76,[16 16],'distinct');

```

```

p=[x x2 x3 x4 x5 x6 x7 x8 x9 x10 x11 x12 x13 x14 x15 x16 x17 x18 x19 x20 x25 x26
x27 x28 x29 x30 x31 x32 x33 x34 x35 x36 x37 x38 x39 x40 x41 x42 x43 x44 x45 x46
x47 x48 x49 x50 x51 x52 x53 x54 x55 x65 x66 x67 x68 x69 x70 x71 x72 x73 x74 x75
x76];
t=[y y2 y3 y4 y5 y6 y7 y8 y9 y10 y11 y12 y13 y14 y15 y16 y17 y18 y19 y20 y25 y26
y27 y28 y29 y30 y31 y32 y33 y34 y35 y36 y37 y38 y39 y40 y41 y42 y43 y44 y45 y46
y47 y48 y49 y50 y51 y52 y53 y54 y55 y65 y66 y67 y68 y69 y70 y71 y72 y73 y74 y75
y76];

```

Normal Trial

```

net=newff([minmax(p)],[256, 256],{'logsig' 'logsig'},'traincgb');
net.LW{2,1} = net.LW{2,1}*0.0001;
net.b{2} = net.b{2}*0.001;
net.trainparam.min_grad=1e-5;
net.trainParam.lr= .5;
net.trainparam.epochs=1000;
net.trainparam.show= 5;
net.trainparam.goal= .0005;
net=train(net,p,t);
end

```

Validation Trial

```

[pn,meanp,stdp,tn,meant,stdt]=prestd(p,t);
[ptrans,transMat] = prepca (pn, 0.001);
[r,q]=size(ptrans);
iitst=2:4:q;
iiival=4:4:q;
iitr=[1:4:q 3:4:q];
validation.P=ptrans(:,iiival);
validation.T=tn(:,iiival);
test.P=ptrans(:,iitst);
test.V=tn(:,iitst);
ptr=ptrans(:,iitr);
ttr=tn(:,iitr);

net=newff([minmax(ptr)],[256, 256],{'logsig' 'logsig'},'traincgb');
net.trainparam.max_fail=10;
net.LW{2,1} = net.LW{2,1}*0.0001;

```

```

net.b{2} = net.b{2}*0.001;
net.trainparam.min_grad=1e-5;
net.trainParam.lr= .5;
net.trainparam.epochs=1000;
net.trainparam.show= 5;
net.trainparam.goal= .0005;

[net,tr]=train(net,ptr,ttr,[],[],validation,test);

```

Test Image

```

x21=imread('c21.jpg');
y21=imread('b21.jpg');
x21=double(x21)/255;
y21=double(y21)/255;
x21=rgb2gray(x21);
y21=rgb2gray(y21);
y21=im2col(y21,[16 16],'distinct');
x21=im2col(x21,[16 16],'distinct');
p=[x21];
t=[y21];

a=sim(net,p);
a=uint8(round(a*255));
a=col2im(a,[16 16],[200 200],'distinct');
imshow (a)

pnewn=trastd(pnew,meanp,stdp);
pnewtrans=trapca(pnewn,transMat);
a=sim(net,pnewtrans);
a=uint8(round(a*255));
a=col2im(a,[16 16],[200 200],'distinct');
imshow (a)

x21=uint8(round(x21*255));
x21=col2im(x21,[16 16],[200 200],'distinct');
figure, imshow (x21)

x=imread('c1.jpg');
y=imread('b1.jpg');
x=double(x)/255;
y=double(y)/255;
x=rgb2gray(x);
y=rgb2gray(y);

```

```

y=im2col(y,[16 16],'distinct');
x=im2col(x,[16 16],'distinct');
p=[x];
t=[y];

```

```

x21=imread('c21.jpg');
y21=imread('b21.jpg');
x21=double(x21)/255;
y21=double(y21)/255;
x21=rgb2gray(x21);
y21=rgb2gray(y21);
y21=im2col(y21,[16 16],'distinct');
x21=im2col(x21,[16 16],'distinct');
p=[x21];
t=[y21];

```

```

a=sim(net,p);

```

```

e=t-p;
mse(e)

```

```

e=t-a;
mse(e)

```

```

a=uint8(round(a*255));
a=col2im(a,[16 16],[200 200],'distinct');
imshow(a)

```

```

x21=uint8(round(x21*255));
x21=col2im(x21,[16 16],[200 200],'distinct');
figure, imshow(x21)

```

Appendix B:

Images Used

Four main reasons for using a public image archive are A) that there are no royalties for using the images for private/public use. B) There are no copyright infringements for using images for private/public use. C) Ease of access to materials can greatly reduce time required for the project. D) Images within the Archive are recognized as industry standards.

Compressing images was achieved in the software program Paint Shop Pro. In order to reduce the amount of time it would take to individually compress each image, the utility “Batch Conversion” within Paint Shop Pro, was used. Batch Conversion compresses images or change file formats to a group of images within a user-selected folder. Under this application, the advanced option must be selected to set the ratio of compression for all of the images, otherwise compression will not occur. Also, the user must specify the target folder in which the newly compressed images must go in. Files under Microsoft Windows cannot have two of the same name and extension types within the same folder; otherwise the newly compressed images will copy over the originals.

The commands for this operation are as follows;

File> Batch Conversion> select folder where the images are located> files of type > select .jpg> output settings> select .jpg> options> select standard encoding> slide control to 95%> select ok> select output folder> choose SELECT ALL in the right hand corner to start the process.

It should be noted that some in the following pages have been reduced in size to actually fit on the pages.