# Semiannual Status Report
# for
# NAG 5–1612

Submitted to:
Instrument Division
Engineering Directorate
Goddard Space Flight Center
Greenbelt, MD 20771
Attn: Warner Miller and Pen Shu Yeh

K. Sayood, A. C. Hadenfeldt,
B. Gorjala and X. Wang

Department of Electrical Engineering
and
Center for Communication & Information Science
University of Nebraska-Lincoln
Lincoln, Nebraska 68588–0511

*Period: June 15, 1991 — December 15, 1991*

## Introduction

This is the semiannual status report for the NASA grant NAG 5–1612.

The work performed in the previous six months can be divided into three main areas:

- Transmission of images over local area networks (LANs)
- Coding of color mapped (pseudo-color) images.
- Low rate video coding.

The first two items were initiated in the previous period, and completed in the current period. Detailed reports on these items have been given in two MS theses. We will therefore provide only a brief overview of the work done in these two areas here. The third item is reported in somewhat more detail.

## 1 Coding for Local Area Networks

We have previously developed several simple image coding algorithms with good edge preservation properties [1,2]. In this work we examined how methods for implementing these schemes could be implemented over local area networks. In particular we examined implementation issues with respect to token ring networks.

The first coding scheme is an Adaptive Differential Pulse Code Modulation (ADPCM) scheme with a Jayant adaptive quantizer [1]. Consecutive expansions of the step-size of the adaptive quantizer indicate the presence of an edge in this scheme. This process is takes place at both the transmitter and the receiver; therefore, there is no need to alert the receiver. While an edge is present, information is sent through a side channel. This

information is used by the receiver to preserve the edge. Using a side channel channel is a drawback in convention channels because synchronization problems are introduced.

These synchronization problems can be overcome by implementing this scheme in a packet network environment. In a token ring network, any station may capture the token by removing it from the ring. After the removal of the token, the station may begin to transmit information frames. After transmitting the information, the station immediately issues a new token. The token ring network supports two classes of traffic:

1. Synchronous traffic: A data transmission service in which each requester is pre-allocated a maximum bandwidth and guaranteed a response time not to exceed a specific delay.

2. Asynchronous traffic: A class of data transmission service in which all requests for service contend for a pool of dynamically allocated ring bandwidth and response time.

A set of timers and several parameters limit a station's message transmission time before passing the token to the next station and the duration of information transmission of each class within a station. Each station maintains two timers, the Token Rotation Timer (TRT) and the Token Holding Timer (THT). The TRT at node j times the token's speed as it circulates around the ring. When node j recaptures the token, the TRT is immediately reset and restarted. Before resetting the TRT, its current value is assigned to the THT. Both the TRT and THT become active during message transmission at node j. The THT is reset when the token is passed to the next station. While the TRT continues to run, the THT remains active until the token arrives at node j again.

When the network is initialized, the stations decide the value of a target token rotation time (TTRT), so that the requirements for maximum access time are met. The extension from single to several priority classes is obtained by introducing a target rotation time

for each additional class, and by using that value to check whether or not the station is allowed to transmit frames of that class.

If a station captures the token before its token rotation timer reaches the TTRT value, it is called an *early* token. If it captures the token after the timer has exceeded the value of TTRT, then it is called a *late* token. An *early* token may be used to transmit both synchronous and asynchronous traffic, while a *late* token may only be used for synchronous traffic. The difference between the TTRT and the TRT is the available bandwidth for the asynchronous information. The THT limits the amount of time a station can transmit.

When using the proposed ADPCM scheme over the network, we view the regular Differential Pulse Code Modulator (DPCM) output as the synchronous message and the quantizer error information or side information as the asynchronous message. This system cannot afford to lose any of the regular DPCM output and, also, this information has timing constraints. The side information is not as important because the image can be reconstructed at the receiver, with some degradation, even without this information.

In the analysis of a token protocol, it is generally assumed that the queues of asynchronous messages ready to be sent are heavily loaded, so that messages are always available for transmission. In our case, the asynchronous information queue will not be loaded heavily because the side information needs to be sent only when there is an edge.

The size of the packet for the synchronous traffic is fixed. Whenever a node captures an *early* token, if there is asynchronous traffic waiting at the node, the size of the packet is increased to match the available capacity and the regular information, or synchronous traffic, is followed by as much of the asynchronous traffic as can be accommodated.

In the description of the protocol, we saw that the transmission ends when the THT is reached or when there is no more asynchronous information to transmit. In this

simulation, the transmission ends when the available bandwidth is utilized or when there is no more side information to send, because the asynchronous traffic is not heavily loaded. The most recent side information is transmitted in the bandwidth available for asynchronous traffic. If there is any side information left after the transmission, it is discarded.

It is assumed that the TTRT is the time taken by a token to return to the same node when all of the nodes have a packet to send. If any nodes do not have a packet to send, the token arrives earlier than expected. If node 1 captures an *early* token, i.e. the TRT is less than the TTRT, then (TTRT-TRT) becomes the time available for the side information. If node 1 captures a *late* token, i.e., the TRT is greater than the TTRT, then only the regular information is sent and the side information is held in the queue.

Whenever the receiver receives an increased size packet, it takes the bits received after the regular size of the packet as side information. This side information is used to correct the most recent edge pixels (as indicated by consecutive expansions of the adaptive quantizers).

The second coding scheme uses a recursively indexed quantizer instead of a Jayant quantizer [2]. In this case the algorithm shows a graceful degradation when used with different quantizer step-sizes [3]. We use this fact in a local area network with all traffic belonging to the same class.

In the timed token protocol, we had two classes of traffic; asynchronous and synchronous. Asynchronous messages do not have timing constraints and can be transmitted only when the bandwidth becomes available. Now consider a case where all of the traffic belongs to one class, but the size of the packet can be changed depending upon the available capacity. The amount of information sent in each time is fixed; only the length of the packet is variable. The coding scheme should be able to take advantage of the increased capacity when it becomes available and be able to decrease the size

of the packet when the available capacity decreases. To be able to adjust to varying bandwidths, the coder needs to know what the available bandwidth will be in the future at the time it is generating the packet. Therefore, the coder should be able to predict the available capacity for future packets based on present traffic conditions, because once the packet is processed, the system cannot afford to lose any information.

One simple way to predict available bandwidth for the future is to compare the buffer occupancy. The number of packets waiting in the queue is directly related to the traffic because a node can send only one packet with each token capture independent of how many packets are waiting in the queue. Therefore, depending upon the buffer occupancy, the size for the next coming packet can be decided. We can fix thresholds for the buffer, with which at different levels of occupancy the data generator will generate packets with different lengths.

The stepsize of the recursively indexed quantizer in the edge preserving scheme can be varied to get the desired bit rate. When the quantizer input falls in the inner levels, the rate obtained is fixed. For a fast changing input signal, the prediction error falls in the overload region, which usually happens when it encounters an edge in the image. When the quantizer input falls in the outer levels which are in the overload region, the error will be encoded repeatedly until the input falls into one of the inner levels of the quantizer. The bit rate corresponding to this rapidly changing input is variable and is higher than the usual rate. This scheme is especially useful in transmitting images over channels where the available bandwidth is variable, such as in the packet network environment.

Like the previous simulation, this scheme is implemented on a token ring network by taking the image information at node 1 and random information at all other nodes. In this simulation, the number of pixels per packet at node 1 is fixed, whereas the length of the packet is varied. The length of the packet is assumed to be fixed at all of the nodes except node 1. In this work seven different thresholds are fixed for the buffer occupancy.

Before processing the packet, the data generator observes the buffer occupancy at that node and then reads the length of the packet corresponding to that level of occupancy. The bit rate is then selected to match the packet length. This is the length of the packet which can be transmitted in the available bandwidth.

Once the packet arrives, the length of the packet cannot be changed and also the system cannot afford to lose any information. We assume that no information is lost, so all packets coming after the seventh threshold of buffer occupancy are processed at a rate of 1.0 bpp and stored in the buffer.

The stepsize of the quantizer chosen to process that packet is sent with each packet giving an overhead of three bits. The first three bits in the information of a frame represents the stepsize. At the receiver, the image is reconstructed with the corresponding stepsize. The network parameters used in this simulation were the same as the parameters used in the edge correcting scheme simulation.

A detailed report of these investigations is contained in [4].

## 2 Progressive Transmission of Color Mapped Images

To develop a progressive transmission scheme for color mapped images, we have to deal with two problems. One problem is of course the design of the progressive transmission algorithm. The other is the problem of quantization in the color map space. The latter is a problem because of the way color images are generated using a color map. In most color display devices, the color CRT uses three electron beams to produce the displayed color. Each beam excites a particular type of phosphor on the CRT which emits light in either the red green or blue range. In more expensive display terminals, there are three frame buffers which contain the red, green, and blue tristimulus values for the pixels. These are generally eight bit values that are converted to voltages by DACs and used to control the three electron beams. Because the amount of memory required for

these frame buffers is rather expensive, less costly systems use a single frame buffer along with a color map. The values stored in the frame buffer are no longer intensity values, but indices into a 24 bit table, generally called the colormap or lookup table (lut). The numerical values of these indices no longer have any direct relationship to the color of the pixel. If two indices are close in value, the pixels representing them will not necessarily be close in color. For this reason, the indices cannot withstand any coding distortion.

This is a severe problem in the context of coding, as at some stage of the coding process one generally requires some reduction in resolution or quantization. In a color mapped image distortion even in the least significant bit can result in an image which looks random. Because most progressive transmission schemes require intermediate approximations, an effect of the colormap structure is to prevent the user from seeing a coherent picture till the very last stages of the progressive transmission. This defeats the purpose of the progressive transmission approach which relies on the user being able to make out the image in the very early stages of the transmission.

Our initial efforts were therefore directed to developing colormaps in which the indices would, in some sense, correspond directly to the perceived colors. Thus when two indices were close together, this would generally mean that the colors represented by these values would also be (in some sense) close together. To obtain a measure of difference we examined the various color spaces proposed by NTSC [5] and CIE [5,6] to see, in which space euclidean distance corresponded most closely to perceived differences. We also evaluated measures proposed by Frei [7], and Faugeras [8] which involve the use of nonlinear transforms to map the color vectors into a "perceptual" space. The colormaps were sorted both as a circular list, and as a linear list. The sorting itself was performed using both a *greedy* algorithm, and *simulated annealing* [9,10]. Because the simulated annealing algorithm provided much better results, we used it in our study.

Once we had obtained both difference measures and a sorted colormap, we developed a progressive transmission scheme for the color mapped images. We evaluated several possibilities and finally selected a scheme based on a scheme by Driezen [11]

The details of this work were reported in an MS thesis included with this report [12].

# 3 Low Rate Video Coding

Transmitting images without removing redundant information is very expensive in terms of both transmission time and the bandwidth used. For instance, an uncoded image sequence 256x256 pixels in size, transmitted at 30 images per second, requires a bandwidth of more than 15 Mb/s. The available ISDN network has a 64 kb/s channel, making it possible to transmit an image sequence in real time only when the required transmission bandwidth can be reduced by a factor of more than 200.

There are various methods and strategies for removing redundancy in the image data. Among these, transform coding, especially cosine transform coding, is a very popular and efficient way to compress data. The basic motivation behind transform coding is to transform a set of data into another set of "less correlated" or "more independent" coefficients before coding. Traditionally as well as conveniently it is assumed that all of the *important* coefficients are packed into a specific area of the transform domain. The amount of compression depends on the number of coefficients retained in a given area. In cosine transform coding, the low frequency area is usually considered more important than the high frequency area. For this reason, image data is often compressed by coding and transmitting only the low frequency components.

Another possibility is to put a threshold on the transformed coefficient magnitude and set all coefficients below the threshold to zero. This approach is more realistic because we do not assume any fixed important area but consider this area dynamic depending on

the characteristics of images. This is a more complex coding strategy but it results in a very high compression rate while maintaining good picture quality.

In this report we present some of our tests and results by implementing cosine transform threshold coding. The motion picture sequence we use for the test is a typical video-conference sequence. We can see that with a bit rate as low as 0.1 bit/pixel the SNR of images is still more than 30 dB on the average. We plan to compress images with more efficient coding strategies and improve the picture quality within the bit rate limits.

Differential encoding schemes are very good for removing redundancy when the input sequence is highly correlated. As consecutive frames of a video sequence are generally very highly correlated, we use differential encoding for removal of interframe redundancy.

For intraframe redundancy removal we use the discrete cosine transform.

Each frame of our motion sequence is 256x256 in size. For convenience the frame is divided into smaller blocks with size 8x8. The two-dimensional discrete cosine transform for each block is:

$$F(u,v) = \frac{4c(u,v)}{N^2} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f(m,n) \cos \frac{(2m+1)u\pi}{2N} \cos \frac{(2n+1)u\pi}{2N}$$

The inverse cosine transform is given by

$$f(m,n) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} c(u,v) F(u,v) \cos \frac{(2m+1)u\pi}{2N} \cos \frac{(2n+1)v\pi}{2N}$$

where

$c(u,v)$ = 1/2 for $u = v = 0$

= sqrt(2) for only one of $u, v = 0$

= 1 for $u, v$=1,2, ... , $N$-1

$F(m,n)$ = the transformed coefficient

$f(i,j)$ = the grey level of each pixel.

Before the coefficients are quantized, they are compared against a preset threshold. The coefficients that have magnitude smaller than the threshold are set to zero.

The threshold setting is important. With a low threshold, the picture looks very similar to the original one. Many people would even say that the picture looks better because it is more smooth, less noisy and still has all of the features and details. We chose to use a comparatively high threshold so that we could compress the image as much as possible while maintaining reasonably good picture quality. Picture quality is defined by two criteria, a mathematical measurement of SNR (signal-to-noise ratio), and a subjective evaluation of the picture quality.

We chose a non-uniform quantizer after studying the characteristic of the nonzero coefficients. The stepsize and the coding for each level may effect the picture quality and the bit rate. Later in this report we show the result of using an 8-level quantizer.

We use variable-length coding for each quantizer output so on average we use less than three bits per level. Because the number of nonzero coefficients for each block is constantly changing and their positions are not fixed, efficient coding is very important. We are going to use run-length coding to encode the position of each nonzero coefficient, and Huffman codes will be used to encode the runs of coefficients with zero value.

### 3.1 Implementation

The first 256x256 image is divided into 8x8 blocks. For each block we perform discrete cosine transform. Then we set a threshold $t_0$. Any coefficient that is less than $t_0$ in absolute value is set to zero. Coefficients larger than the threshold are quantized and transmitted. At the receiver end the coefficient transmitted data is first decoded and then an inverse cosine transform is used to recover the picture. The inverse cosine transform is also determined by the sender because the recovered image is needed for processing the next frame.

The next frame is subtracted from the previously recovered image. This is the differential encoding procedure which is used to remove most of the redundancy due to the similarities of the two neighboring frames. For the differential image we repeat the process, divide the image into 8x8 blocks and determine the cosine transform, set all the coefficients less than threshold $t_1$ to zero, and quantize the remaining coefficients, which are then coded and transmitted. At the receiver end after the inverse cosine transform we get the differential image. This is added to the previous reconstructed frame to recover the current reconstructed frame. This procedure has to be duplicated at the transmitter in order to get a prediction for the next frame.

The following is a summary of the procedures described above:

Step 1. Divide the first frame into blocks of size 8x8.

Step 2. For each block obtain the cosine transform.

Step 3. Set each coefficient with an absolute value less than threshold $t_0$ to zero.

Step 4. Quantize the nonzero coefficients.

Step 5. Code the quantized coefficients and transmit them.

Step 6. Perform the inverse cosine transform and reconstruct the image for processing the next frame.

Step 7. Get the next frame and divide it into blocks of size 8x8.

Step 8. For each block, subtract it from the corresponding block of the reconstructed frame to obtain the differential block.

Step 9. Obtain the cosine transform for the differential block.

Step 10. Set coefficients with an absolute value less than threshold $t_1$ to zero.

Step 11. Quantize the nonzero coefficients.

Step 12. Code the quantized coefficients and transmit them.

Step 13. Obtain the inverse cosine transform.

Step 14. Add the reconstructed block to the block of previous frame at the same position and get the image.

Step 15. Go to step 7.

### 3.2 Results

The block size we used for this study is 8x8. A larger block size may work better, but the time spent computing the discrete cosine transform will be longer. We begin at frame #1 and run about 60 frames. Below are the results for frame #25.

In the initial trials no quantizer was used. The intent during these trials was to obtain values for the thresholds $t_1$ and $t_2$. Fig. 1 is the relationship curve of SNR versus threshold. Fig. 2 is a plot of the average number of nonzero coefficients per block versus the threshold. Figs. 3–7 are the images corresponding to different thresholds.

When the threshold is 5, we can still produce a good looking image with a SNR of about 33 dB. Next different quantizers were used to quantize the nonzero coefficients. We present some of the results using a non-uniform 8-level quantizer. The SNR from frame #4 to frame #30, when the threshold is 5 and an 8-level quantizer is used, is plotted in Fig. 8. Fig. 9 is the image after quantization.

We can see that the quantizer has had little effect on the picture quality. This means that the more coarse quantizer may probably be used. With two nonzero coefficients for each 8x8 block and three bits for each coefficient, the bit rate is about 0.1 bit/pixel.

From the trials described above, we found that the coding procedure works well if motion in the image is not very rapid. We are considering the use of a motion detector to further compress the image sequence. Issues which need study are whether it will not cost too many bits to transmit the motion vectors in the side information and whether the

picture quality will be affected. We are going to test the method developed with some other motion sequences to see whether it works well in general.

## 4 References


[1] S.M. Schekall, and K. Sayood, "An Edge Preserving DPCM Scheme for Coding Images," *Proc 31st Midwest Symposium on Circuits and Systems*, St. Louis, MO, Aug. 1988, pp. 904-907.

[2] M.C. Rost, and K. Sayood, "An Edge Preserving Differential Image Coding Scheme," *Proc. International Conf. on Computers and Communications*, Scottsdale, AZ, March 1990, pp.

[3] M.C. Rost, and K. Sayood, "An Edge Preserving Differential Image Coding Scheme," *IEEE Transactions on Image Processing vol. 1, April 1992*,

[4] B. Gorjala, *Compression of Images Over Local Area Networks*, MS Thesis, University of Nebraska - Lincoln, August 1991.

[5] A.K. Jain, *Fundamentals of Digital Image Processing*, Prentice-Hall, 1989.

[6] CIE Colorimetry Committee, "Proposal for the Study of Color Spaces and Color-Difference Evaluation," *Journal Opt. Soc. of Amer.*, vol. 64, pp. 896-897, June 1974.

[7] W. Frei, and B. Baxter, "Rate-Distortion Coding Simulation for Color Images," *IEEE Trans. Commun.*, vol. COM-25, pp. 1385-1392, Nov. 1977.

[8] O.D. Faugeras, *Digital Color Image Processing and Psychophysics Within the Framework of a Human Visual System Model*, PhD Dissertation, University of Utah, June 1976.

[9] E. Aarts, and J. Korst, *Simulated Annealing and Boltzman Machines*, New York, John Wiley and Sons, 1989.

[10] W.H. Press et. al., *Numerical Recipes in C*, New York: Cambridge University Press, 1988.

[11] H. Dreizen, "Content Driven Progressive Transmission of Grey-Scale Images," *IEEE Trans. Commun.*, vol. COM-35, pp. 289-296, March 1987.

[12] A.C. Hadenfeldt, *Progressive Transmission of Pseudo-Color Images*, MS Thesis, University of Nebraska - Lincoln, December 1991.

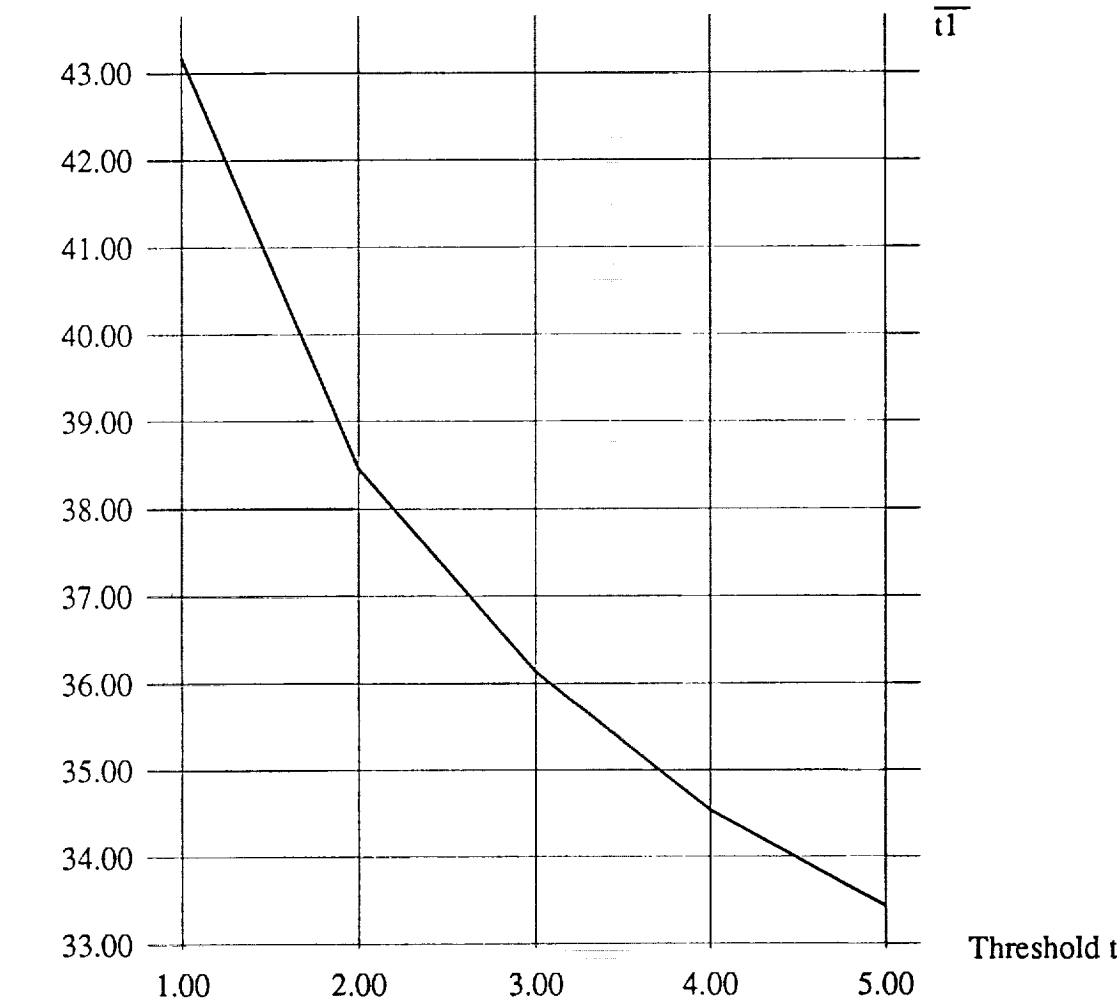


Fig. 1

Plot of PSNR vs. Threshold t, for Frame #25 of the Susie sequence.

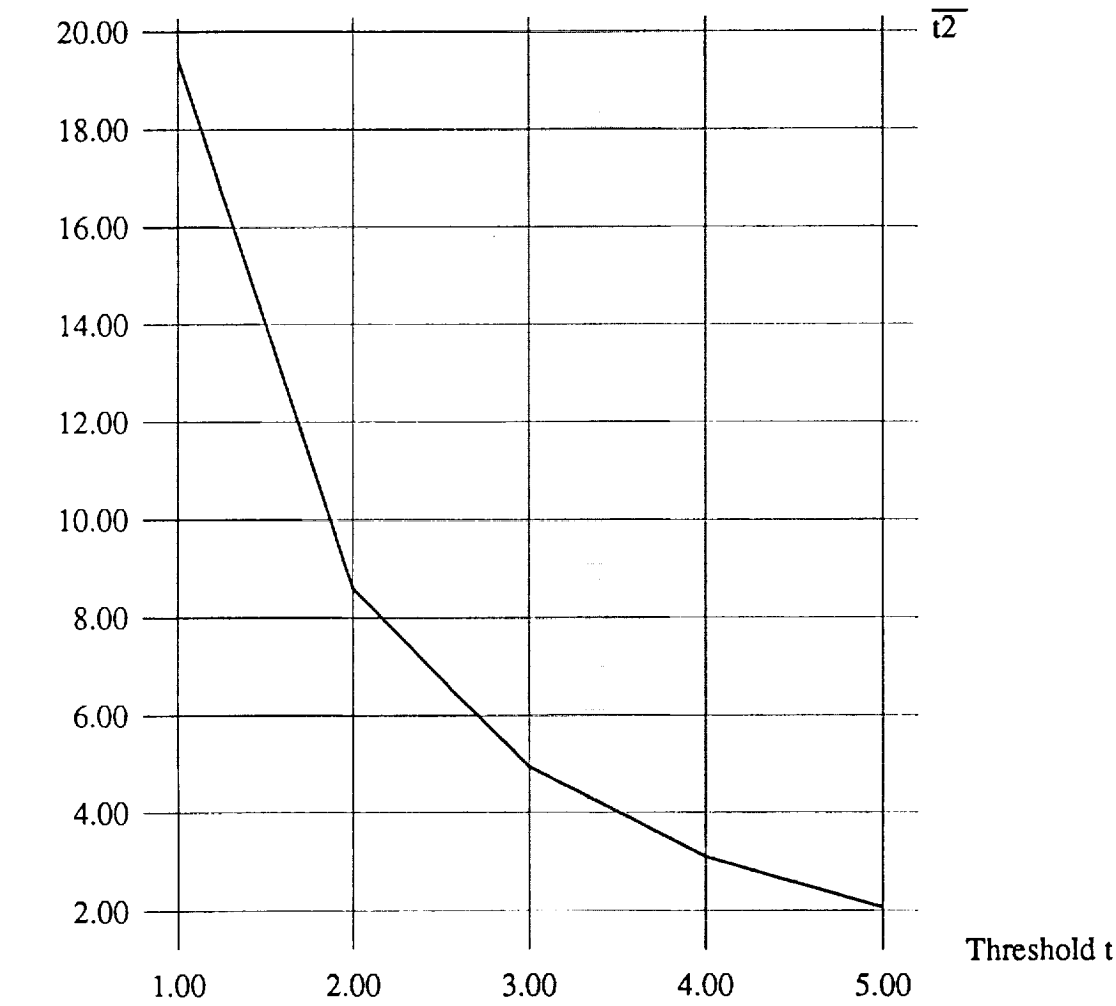


Fig. 2

Plot of the average number of non-zero DCT coefficients vs. threshold $t_2$ for Frame # 25 of the Susie sequence.

Fig. 3

Image reconstructed with $t_\gamma = 1$

Fig. 4

Image reconstructed with $t_1 = 2$

Fig. 5

Image reconstructed with $t_1 = 3$

Fig. 6

Image reconstructed with $t_1 = 4$

Fig. 7

Image reconstructed with $t_1 = 5$

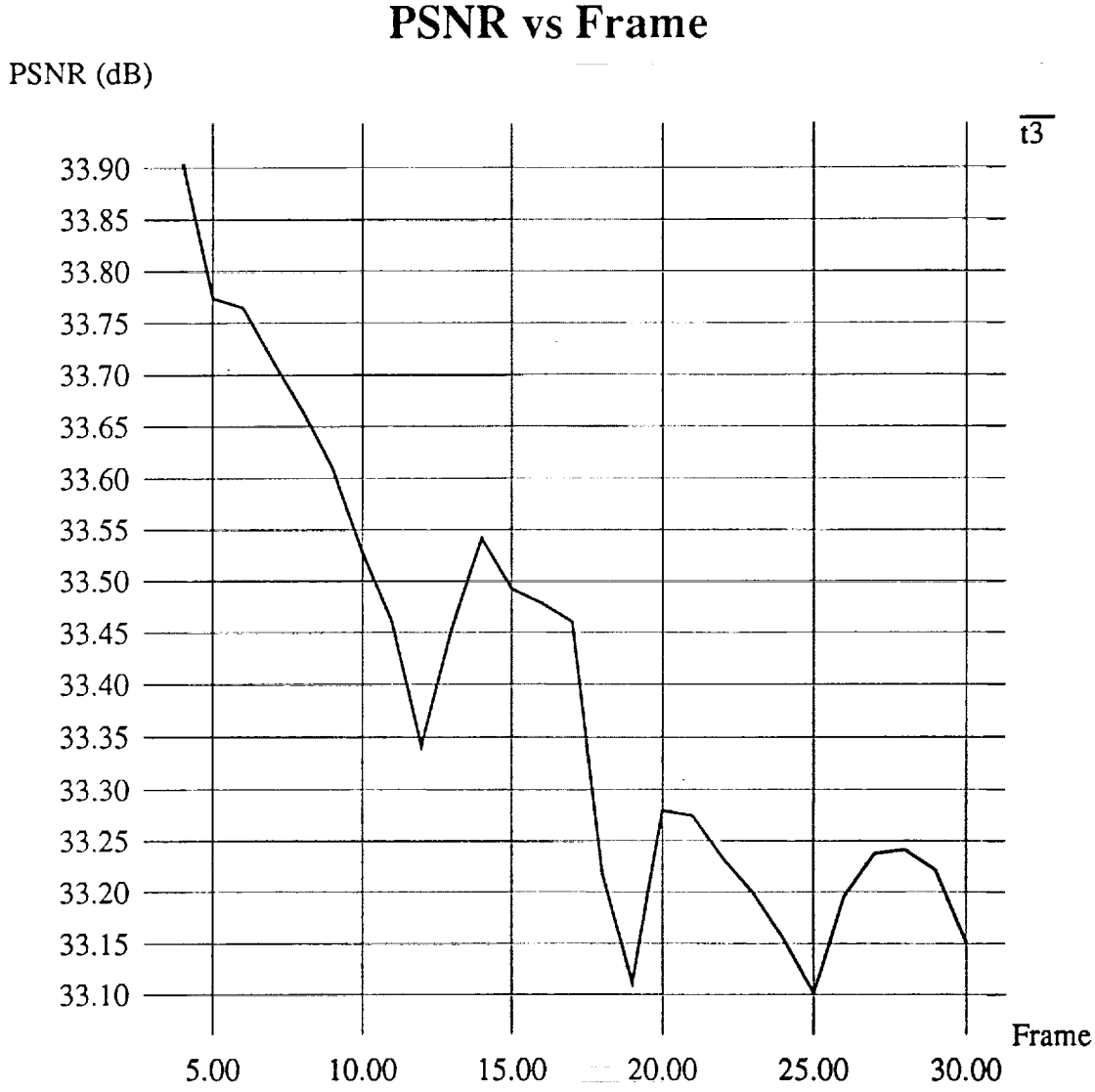


Fig. 8

Peak signal-to-noise ratio over a sequence of 30 frames.

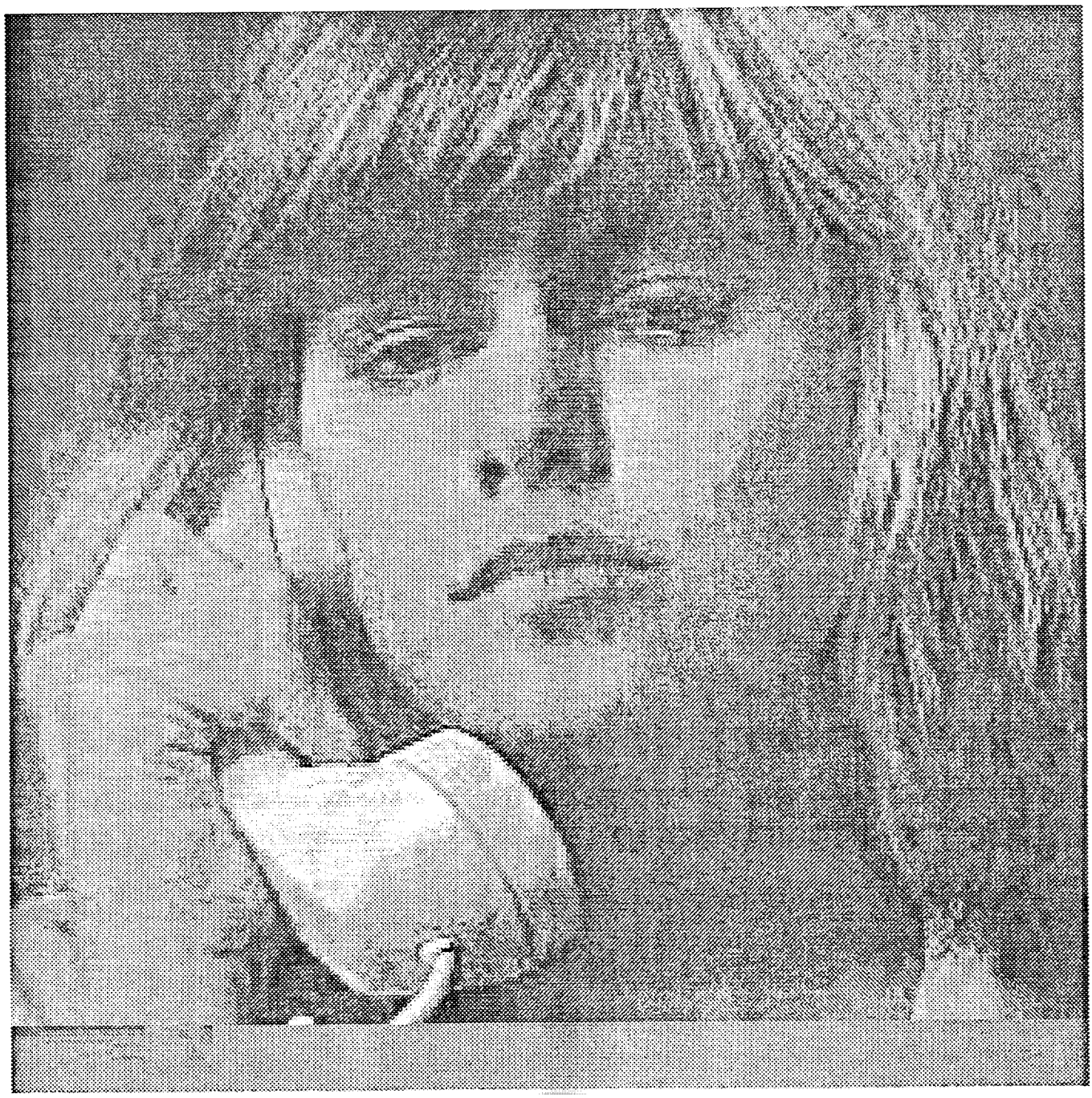

Fig. 9

Image coded at about 0.1 bit per pixel.