Image Compression and the Discrete Cosine Transform

Ken Cabeen and Peter Gent

Math 45
College of the Redwoods

Abstract. The mathematical equations of the DCT and its uses with image compression are explained.

Introduction

As our use of and reliance on computers continues to grow, so too does our need for efficient ways of storing large amounts of data. For example, someone with a web page or online catalog – that uses dozens or perhaps hundreds of images – will more than likely need to use some form of image compression to store those images. This is because the amount of space required to hold unadulterated images can be prohibitively large in terms of cost. Fortunately, there are several methods of image compression available today. These fall into two general categories: lossless and lossy image compression. The JPEG process is a widely used form of lossy image compression that centers around the Discrete Cosine Transform. The DCT works by separating images into parts of differing frequencies. During a step called quantization, where part of compression actually occurs, the less important frequencies are discarded, hence the use of the term "lossy." Then, only the most important frequencies that remain are used retrieve the image in the decompression process. As a result, reconstructed images contain some distortion; but as we shall soon see, these levels of distortion can be adjusted during the compression stage. The JPEG method is used for both color and black-and-white images, but the focus of this article will be on compression of the latter.

The Process

The following is a general overview of the JPEG process. Later, we will take the reader through a detailed tour of JPEG's method so that a more comprehensive understanding of the process may be acquired.

1. The image is broken into 8x8 blocks of pixels.
2. Working from left to right, top to bottom, the DCT is applied to each block.
3. Each block is compressed through quantization.
4. The array of compressed blocks that constitute the image is stored in a drastically reduced amount of space.
5. When desired, the image is reconstructed through decompression, a process that uses the Inverse Discrete Cosine Transform (IDCT).

The DCT Equation

The DCT equation (Eq. 1) computes the i,jth entry of the DCT of an image.

$$D(i,j) = \frac{1}{\sqrt{2N}} C(i)C(j) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} p(x,y) \cos\left[\frac{(2x+1)i\pi}{2N} \right] \cos\left[\frac{(2y+1)j\pi}{2N} \right] \qquad 1$$

$$C(u) = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } u = 0 \\ 1 & \text{if } u > 0 \end{cases} \qquad 2$$

$p(x,y)$ is the x,yth element of the image represented by the matrix p. N is the size of the block that the DCT is done on. The equation calculates one entry (i,jth) of the transformed image from the pixel values of the original image matrix. For the standard 8x8 block that JPEG compression uses, N equals 8 and x and y range from 0 to 7. Therefore $D(i,j)$ would be as in Equation (3).

$$D(i,j) = \frac{1}{4}C(i)C(j)\sum_{x=0}^{7}\sum_{y=0}^{7}p(x,y)\cos\left[\frac{(2x+1)i\pi}{16}\right]\cos\left[\frac{(2y+1)j\pi}{16}\right] \qquad 3$$

Because the DCT uses cosine functions, the resulting matrix depends on the horizontal, diagonal, and vertical frequencies. Therefore an image black with a lot of change in frequency has a very random looking resulting matrix, while an image matrix of just one color, has a resulting matrix of a large value for the first element and zeroes for the other elements.

The DCT Matrix

To get the matrix form of Equation (1), we will use the following equation

$$T_{i,j} = \begin{cases} \frac{1}{\sqrt{N}} & \text{if } i = 0 \\ \sqrt{\frac{2}{N}}\cos\left[\frac{(2j+1)i\pi}{2N}\right] & \text{if } i > 0 \end{cases} \qquad 4$$

For an 8x8 block it results in this matrix:

$$T = \begin{bmatrix} .3536 & .3536 & .3536 & .3536 & .3536 & .3536 & .3536 & .3536 \\ .4904 & .4157 & .2778 & .0975 & -.0975 & -.2778 & -.4157 & -.4904 \\ .4619 & .1913 & -.1913 & -.4619 & -.4619 & -.1913 & .1913 & .4619 \\ .4157 & -.0975 & -.4904 & -.2778 & .2778 & .4904 & .0975 & -.4157 \\ .3536 & -.3536 & -.3536 & .3536 & .3536 & -.3536 & -.3536 & .3536 \\ .2778 & -.4904 & .0975 & .4157 & -.4157 & -.0975 & .4904 & -.2778 \\ .1913 & -.4619 & .4619 & -.1913 & -.1913 & .4619 & -.4619 & .1913 \\ .0975 & -.2778 & .4157 & -.4904 & .4904 & -.4157 & .2778 & -.0975 \end{bmatrix}$$

The first row ($i = 1$) of the matrix has all the entries equal to $1/\sqrt{8}$ as expected from Equation (4).

The columns of T form an orthonormal set, so T is an orthogonal matrix. When doing the inverse DCT the orthogonality of T is important, as the inverse of T is T' which is easy to calculate.

Doing the DCT on an 8x8 Block

Before we begin, it should be noted that the pixel values of a black-and-white image range from 0 to 255 in steps of 1, where pure black is represented by 0, and pure white by 255. Thus it can be seen how a photo, illustration, etc. can be accurately represented by these 256 shades of gray.

Since an image comprises hundreds or even thousands of 8x8 blocks of pixels, the following description of what happens to one 8x8 block is a microcosm of the JPEG process;

what is done to one block of image pixels is done to all of them, in the order earlier specified.

Now, let's start with a block of image-pixel values. This particular block was chosen from the very upper- left-hand corner of an image.

$$Original = \begin{bmatrix} 154 & 123 & 123 & 123 & 123 & 123 & 123 & 136 \\ 192 & 180 & 136 & 154 & 154 & 154 & 136 & 110 \\ 254 & 198 & 154 & 154 & 180 & 154 & 123 & 123 \\ 239 & 180 & 136 & 180 & 180 & 166 & 123 & 123 \\ 180 & 154 & 136 & 167 & 166 & 149 & 136 & 136 \\ 128 & 136 & 123 & 136 & 154 & 180 & 198 & 154 \\ 123 & 105 & 110 & 149 & 136 & 136 & 180 & 166 \\ 110 & 136 & 123 & 123 & 123 & 136 & 154 & 136 \end{bmatrix}$$

Because the DCT is designed to work on pixel values ranging from -128 to 127, the original block is "leveled off" by subtracting 128 from each entry. This results in the following matrix.

$$M = \begin{bmatrix} 26 & -5 & -5 & -5 & -5 & -5 & -5 & 8 \\ 64 & 52 & 8 & 26 & 26 & 26 & 8 & -18 \\ 126 & 70 & 26 & 26 & 52 & 26 & -5 & -5 \\ 111 & 52 & 8 & 52 & 52 & 38 & -5 & -5 \\ 52 & 26 & 8 & 39 & 38 & 21 & 8 & 8 \\ 0 & 8 & -5 & 8 & 26 & 52 & 70 & 26 \\ -5 & -23 & -18 & 21 & 8 & 8 & 52 & 38 \\ -18 & 8 & -5 & -5 & -5 & 8 & 26 & 8 \end{bmatrix}$$

We are now ready to perform the Discrete Cosine Transform, which is accomplished by matrix multiplication.

$$D = TMT' \qquad 5$$

In Equation (5) matrix M is first multiplied on the left by the DCT matrix T from the previous section; this transforms the rows. The columns are then transformed by multiplying on the right by the transpose of the DCT matrix. This yields the following matrix.

$$D = \begin{bmatrix} 162.3 & 40.6 & 20.0 & 72.3 & 30.3 & 12.5 & -19.7 & -11.5 \\ 30.5 & 108.4 & 10.5 & 32.3 & 27.7 & -15.5 & 18.4 & -2.0 \\ -94.1 & -60.1 & 12.3 & -43.4 & -31.3 & 6.1 & -3.3 & 7.1 \\ -38.6 & -83.4 & -5.4 & -22.2 & -13.5 & 15.5 & -1.3 & 3.5 \\ -31.3 & 17.9 & -5.5 & -12.4 & 14.3 & -6.0 & 11.5 & -6.0 \\ -0.9 & -11.8 & 12.8 & 0.2 & 28.1 & 12.6 & 8.4 & 2.9 \\ 4.6 & -2.4 & 12.2 & 6.6 & -18.7 & -12.8 & 7.7 & 12.0 \\ -10.0 & 11.2 & 7.8 & -16.3 & 21.5 & 0.0 & 5.9 & 10.7 \end{bmatrix}$$

This block matrix now consists of 64 DCT coefficients, c_{ij}, where i and j range from 0 to 7. The top-left coefficient, c_{00}, correlates to the low frequencies of the original image block. As we move away from c_{00} in all directions, the DCT coefficients correlate to higher and higher frequencies of the image block, where c_{77} corresponds to the highest frequency. It is important to note that the human eye is most sensitive to low frequencies, and results from the quantization step will reflect this fact.

Quantization

Our 8x8 block of DCT coefficients is now ready for compression by quantization. A remarkable and highly useful feature of the JPEG process is that in this step, varying levels of image compression and quality are obtainable through selection of specific quantization matrices. This enables the user to decide on quality levels ranging from 1 to 100, where 1 gives the poorest image quality and highest compression, while 100 gives the best quality and lowest compression. As a result, the quality/compression ratio can be tailored to suit different needs.

Subjective experiments involving the human visual system have resulted in the JPEG standard quantization matrix. With a quality level of 50, this matrix renders both high compression and excellent decompressed image quality.

$$Q_{50} = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

If, however, another level of quality and compression is desired, scalar multiples of the JPEG standard quantization matrix may be used. For a quality level greater than 50 (less compression, higher image quality), the standard quantization matrix is multiplied by (100-quality level)/50. For a quality level less than 50 (more compression, lower image quality), the standard quantization matrix is multiplied by 50/quality level. The scaled

quantization matrix is then rounded and clipped to have positive integer values ranging from 1 to 255. For example, the following quantization matrices yield quality levels of 10 and 90.

$$Q_{10} = \begin{bmatrix} 80 & 60 & 50 & 80 & 120 & 200 & 255 & 255 \\ 55 & 60 & 70 & 95 & 130 & 255 & 255 & 255 \\ 70 & 65 & 80 & 120 & 200 & 255 & 255 & 255 \\ 70 & 85 & 110 & 145 & 255 & 255 & 255 & 255 \\ 90 & 110 & 185 & 255 & 255 & 255 & 255 & 255 \\ 120 & 175 & 255 & 255 & 255 & 255 & 255 & 255 \\ 245 & 255 & 255 & 255 & 255 & 255 & 255 & 255 \\ 255 & 255 & 255 & 255 & 255 & 255 & 255 & 255 \end{bmatrix}$$

$$Q_{90} = \begin{bmatrix} 3 & 2 & 2 & 3 & 5 & 8 & 10 & 12 \\ 2 & 2 & 3 & 4 & 5 & 12 & 12 & 11 \\ 3 & 3 & 3 & 5 & 8 & 11 & 14 & 11 \\ 3 & 3 & 4 & 6 & 10 & 17 & 16 & 12 \\ 4 & 4 & 7 & 11 & 14 & 22 & 21 & 15 \\ 5 & 7 & 11 & 13 & 16 & 12 & 23 & 18 \\ 10 & 13 & 16 & 17 & 21 & 24 & 24 & 21 \\ 14 & 18 & 19 & 20 & 22 & 20 & 20 & 20 \end{bmatrix}$$

Quantization is achieved by dividing each element in the transformed image matrix D by the corresponding element in the quantization matrix, and then rounding to the nearest integer value. For the following step, quantization matrix Q_{50} is used.

$$C_{i,j} = round\left(\frac{D_{i,j}}{Q_{i,j}}\right) \qquad 6$$

$$C = \begin{bmatrix} 10 & 4 & 2 & 5 & 1 & 0 & 0 & 0 \\ 3 & 9 & 1 & 2 & 1 & 0 & 0 & 0 \\ -7 & -5 & 1 & -2 & -1 & 0 & 0 & 0 \\ -3 & -5 & 0 & -1 & 0 & 0 & 0 & 0 \\ -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Recall that the coefficients situated near the upper-left corner correspond to the lower frequencies – to which the human eye is most sensitive – of the image block. In addition, the zeros represent the less important, higher frequencies that have been discarded, giving rise to

the lossy part of compression. As mentioned earlier, only the remaining nonzero coefficients will be used to reconstruct the image. It is also interesting to note the effect of different quantization matrices; use of Q_{10} would give C significantly more zeros, while Q_{90} would result in very few zeros.

Coding

The quantized matrix C is now ready for the final step of compression. Before storage, all coefficients of C are converted by an encoder to a stream of binary data (01101011...). In-depth coverage of the coding process is beyond the scope of this article. However, we can point out one key aspect that the reader is sure to appreciate. After quantization, it is quite common for most of the coefficients to equal zero. JPEG takes advantage of this by encoding quantized coefficients in the zig-zag sequence shown in Figure 1. The advantage lies in the consolidation of relatively large runs of zeros, which compress very well. The sequence in Figure 1 (4x4) continues for the entire 8x8 block.

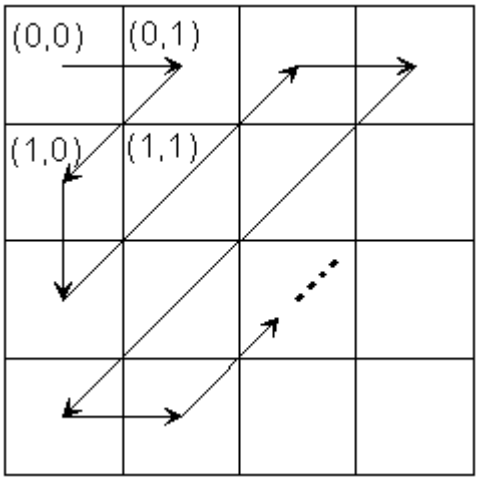


Figure 1

Decompression

Reconstruction of our image begins by decoding the bit stream representing the quantized matrix C. Each element of C is then multiplied by the corresponding element of the quantization matrix originally used.

$$R_{i,j} = Q_{i,j} \times C_{i,j} \qquad 7$$

$$R = \begin{bmatrix} 160 & 44 & 20 & 80 & 24 & 0 & 0 & 0 \\ 36 & 108 & 14 & 38 & 26 & 0 & 0 & 0 \\ -98 & -65 & 16 & -48 & -40 & 0 & 0 & 0 \\ -42 & -85 & 0 & -29 & 0 & 0 & 0 & 0 \\ -36 & 22 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The IDCT is next applied to matrix *R*, which is rounded to the nearest integer. Finally, 128 is added to each element of that result, giving us the decompressed JPEG version *N* of our original 8x8 image block *M*.

$$N = round(T' \; R \; T) + 128 \qquad 8$$

Comparison of Matrices

Let us now see how the JPEG version of our original pixel block compares.

$$Original = \begin{bmatrix} 154 & 123 & 123 & 123 & 123 & 123 & 123 & 136 \\ 192 & 180 & 136 & 154 & 154 & 154 & 136 & 110 \\ 254 & 198 & 154 & 154 & 180 & 154 & 123 & 123 \\ 239 & 180 & 136 & 180 & 180 & 166 & 123 & 123 \\ 180 & 154 & 136 & 167 & 166 & 149 & 136 & 136 \\ 128 & 136 & 123 & 136 & 154 & 180 & 198 & 154 \\ 123 & 105 & 110 & 149 & 136 & 136 & 180 & 166 \\ 110 & 136 & 123 & 123 & 123 & 136 & 154 & 136 \end{bmatrix}$$

$$Decompressed = \begin{bmatrix} 149 & 134 & 119 & 116 & 121 & 126 & 127 & 128 \\ 204 & 168 & 140 & 144 & 155 & 150 & 135 & 125 \\ 253 & 195 & 155 & 166 & 183 & 165 & 131 & 111 \\ 245 & 185 & 148 & 166 & 184 & 160 & 124 & 107 \\ 188 & 149 & 132 & 155 & 172 & 159 & 141 & 136 \\ 132 & 123 & 125 & 143 & 160 & 166 & 168 & 171 \\ 109 & 119 & 126 & 128 & 139 & 158 & 168 & 166 \\ 111 & 127 & 127 & 114 & 118 & 141 & 147 & 135 \end{bmatrix}$$

This is a remarkable result, considering that nearly 70% of the DCT coefficients were discarded prior to image block decompression/reconstruction. Given that similar results will occur with the rest of the blocks that constitute the entire image, it should be no surprise that

the JPEG image will be scarcely distinguishable from the original. Remember, there are 256 possible shades of gray in a black-and-white picture, and a difference of, say, 10, is barely noticeable to the human eye.

Pepper Example

We can do the DCT and quantization process on the peppers image.

Figure 2 – Peppers

Each eight by eight block is hit with the DCT, resulting in the image shown in Figure 3.

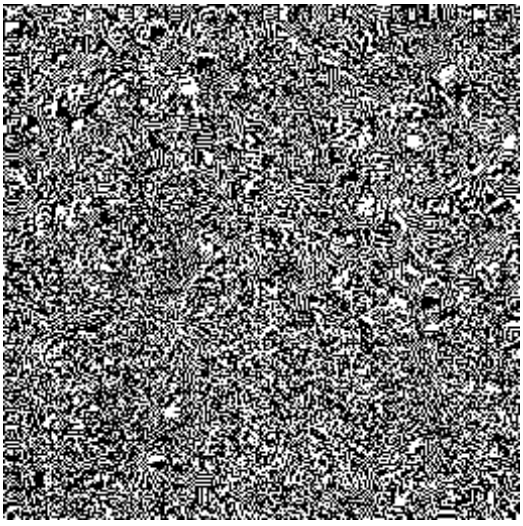


Figure 3 – DCT of Peppers

Each element in each block of the image is then quantized using a quantization matrix of quality level 50. At this point many of the elements become zeroed out, and the image takes up much less space to store.

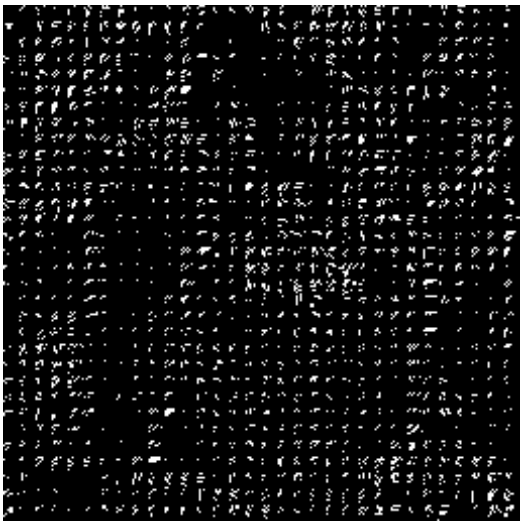


Figure 4 – Quantized DCT of Peppers

The image can now be decompressed using the inverse discrete cosine transform. At quality level 50 there is almost no visible loss in this image, but there is high compression. At lower quality levels, the quality goes down by a lot, but the compression does not increase very much.

Figure 5 – Original Peppers

Figure 6 – Quality 50 – 84% Zeros

Figure 7 – Quality 20 – 91% Zeros

Figure 8 – Quality 10 – 94% Zeros

More Examples

We can see what the compression does to other images. High contrast images, or images with a lot of high frequencies do not compress as well as smooth, low frequency images.

Figure 9 – Original

Figure 10 – Quality 15 – 90% Zeros

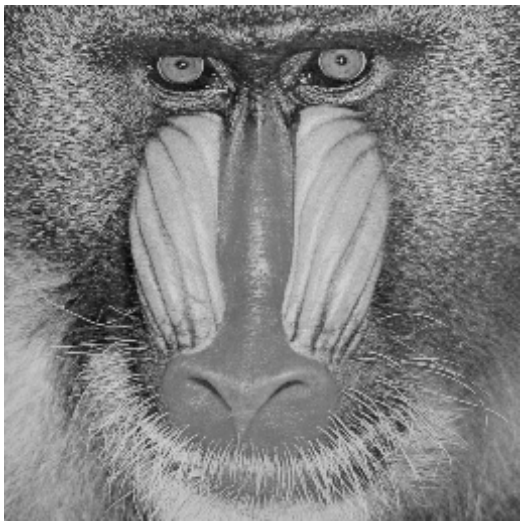


Figure 11 – Original

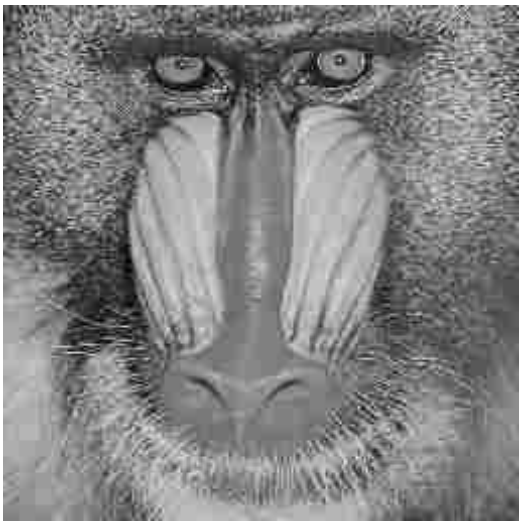


Figure 12 – Quality 15 – 88% Zeros

Bibliography

- Kesavan, Hareesh. *Choosing a DCT Quantization Matrix for JPEG Encoding*. Web page. http://www-ise.Stanford.EDU/class/ee392c/demos/kesavan/
- McGowan, John. *The Discrete Cosine Transform*. Web page. http://www.rahul.net/jfm/dct.html
- Wallace, Gregory K. *The JPEG Still Picture Compression Standard*. Paper submitted in December 1991 for publication in *IEEE Transactions on Consumer Electronics*.
- Wolfgang, Ray. *JPEG Tutorial*. Web page. http://www.imaging.org/tutorial/jpegtut1.html
- Our special thanks to David Arnold, math instructor extraordinaire.

Image Coding Using Wavelet Transform

Marc Antonini, Michel Barlaud, *Member, IEEE,* Pierre Mathieu, and Ingrid Daubechies, *Member, IEEE*

***Abstract*—Image compression is now essential for applications such as transmission and storage in data bases. This paper proposes a new scheme for image compression taking into account psychovisual features both in the space and frequency domains; this new method involves two steps. First, we use a *wavelet transform* in order to obtain a set of biorthogonal subclasses of images; the original image is decomposed at different scales using a pyramidal algorithm architecture. The decomposition is along the vertical and horizontal directions and maintains constant the number of pixels required to describe the image. Second, according to Shannon's rate distortion theory, the wavelet coefficients are *vector quantized* using a multiresolution codebook. Furthermore, to encode the wavelet coefficients, we propose a noise shaping bit allocation procedure which assumes that details at high resolution are less visible to the human eye. Finally, in order to allow the receiver to recognize a picture as quickly as possible at minimum cost, we present a progressive transmission scheme. It is shown that the wavelet transform is particularly well adapted to progressive transmission.**

***Keywords*—Wavelet, biorthogonal wavelet, multiscale pyramidal algorithm, vector quantization, noise shaping, progressive transmission.**

I. Introduction

In many different fields, *digitized images* are replacing conventional analog images as photograph or x-rays. The volume of data required to describe such images greatly slow transmission and makes storage prohibitively costly. The information contained in the images must, therefore, be compressed by extracting only the visible elements, which are then encoded. The quantity of data involved is thus reduced substantially.

A fundamental goal of data compression is to reduce the bit rate for transmission or storage while maintaining an acceptable fidelity or image quality. Compression can be achieved by transforming the data, projecting it on a basis of functions, and then encoding this transform. Because of the nature of the image signal and the mechanisms of human vision, the transform used must accept nonstationarity and be well localized in both the space and frequency domains. To avoid redundancy, which hinders compression, the transform must be at least biorthogonal and lastly, in order to save CPU time, the corresponding algorithm must be fast. The two-dimensional wavelet transform defined by Meyer and Lemarié [31], [24], [25], together with its implementation as described by Mallat [27], satisfies each of these conditions.

The compression method we have developed associates a wavelet transform and a vector quantization coding scheme. The wavelet coefficients are coded considering a noise shaping bit allocation procedure. This technique exploits the psychovisual as well as statistical redundancies in the image data, enabling bit rate reduction.

Section II describes the wavelet transforms used in this paper. After a quick review of wavelets in general, we explain in more detail the properties and construction of regular biorthogonal wavelet bases. We then extend this one-dimensional construction to a two-dimensional scheme with separable filters. The new coding scheme is next presented in Section III. We focus particularly in this section on the statistical properties of wavelet coefficients, on the asymptotic coding gain that can be achieved using vector quantization in the subimages, and on the optimal allocation across the subimages. Experimental results are given in Section IV for images taken within and outside of the training set.

Manuscript received February 7, 1990; revised March 26, 1991.

M. Antonini, M. Barlaud, and P. Mathieu are with LASSY 13S CNRS, Universite de Nice-Sophia Antipolis, 06560 Valbonne, France.

I. Daubechies is with AT&T Bell Laboratories, Murray Hill, NJ 07974.

IEEE Log Number 9106073.

II. Wavelets

A. A Short Review of Wavelet Analysis

Wavelets are functions generated from one single function ψ by dilations and translations

$$\psi^{a,b}(t) = |a|^{-1/2}\psi\left(\frac{t-b}{a}\right).$$

(For this introduction we assume t is a one-dimensional variable). The *mother wavelet* ψ has to satisfy $\int dx\, \psi(x) = 0$, which implies at least some oscillations. (Technically speaking, the condition on ψ should be $\int d\omega\, |\Psi(\omega)|^2\, |\omega|^{-1} < \infty$, where Ψ is the Fourier transform of ψ; if $\psi(t)$ decays faster than $|t|^{-1}$ for $t \to \infty$, then this condition is equivalent to the one above). The definition of wavelets as dilates of one function means that high frequency wavelets correspond to $a < 1$ or narrow width, while low frequency wavelets have $a > 1$ or wider width.

The basic idea of the wavelet transform is to represent any arbitrary function f as a superposition of wavelets. Any such superposition decomposes f into different scale levels, where each level is then further decomposed with a resolution adapted to the level. One way to achieve such a decomposition writes f as an integral over a and b of $\psi^{a,b}$ with appropriate weighting coefficients [22]. In practice, one prefers to write f as a discrete superposition (sum rather than integral). Therefore, one introduces a discre-

tization, $a = a_0^m$, $b = nb_0a_0^m$, with $m, n \in \mathbb{Z}$, and $a_0 > 1$, $b_0 > 0$ fixed. The wavelet decomposition is then

$$f = \sum c_{m,n}(f)\psi_{m,n} \tag{1}$$

with $\psi_{m,n}(t) = \psi^{a_0^m, nb_0a_0^m}(t) = a_0^{-m/2}\psi(a_0^{-m}t - nb_0)$. Decompositions of this type were studied in [14], [15]. For $a_0 = 2$, $b_0 = 1$ there exist very special choices of ψ such that the $\psi_{m,n}$ constitute an orthonormal basis, so that

$$c_{m,n}(f) = \langle \psi_{m,n}, f \rangle = \int dx\, \psi_{m,n}(x) f(x)$$

in this case. Different bases of this nature were constructed by Stromberg [36], Meyer [31], Lemarié [24], Battle [7], and Daubechies [16]. All these examples correspond to a multiresolution analysis, a mathematical tool invented by Mallat [27], which is particularly well adapted to the use of wavelet bases in image analysis, and which gives rise to a fast computation algorithm.

In a multiresolution analysis, one really has *two* functions: the mother wavelet ψ and a *scaling function* ϕ. One also introduces dilated and translated versions of the scaling function, $\phi_{m,n}(x) = 2^{-m/2}\phi(2^{-m}x - n)$. For fixed m, the $\phi_{m,n}$ are orthonormal. We denote by V_m the space spanned by the $\phi_{m,n}$; these spaces V_m describe successive approximation spaces, $\cdots V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \cdots$, each with resolution 2^m. For each m, the $\psi_{m,n}$ span a space W_m which is exactly the orthogonal complement in V_{m-1} of V_m; the coefficients $\langle \psi_{m,n}, f \rangle$, therefore, describe the information lost when going from an approximation of f with resolution 2^{m-1} to the coarser approximation with resolution 2^m. All this is translated into the following algorithm for the computation of the $c_{m,n}(f) = \langle \psi_{m,n}, f \rangle$ (for more details, see [27]):

$$c_{m,n}(f) = \sum_k g_{2n-k}a_{m-1,k}(f)$$

$$a_{m,n}(f) = \sum_k h_{2n-k}a_{m-1,k}(f) \tag{2}$$

where $g_l = (-1)^l h_{-l+1}$ and $h_n = 2^{1/2}\int dx\, \phi(x-n)\phi(2x)$. In fact the $a_{m,n}(f)$ are coefficients characterizing the projection of f onto V_m. If the function f is given in sampled form, then one can take these samples for the highest order resolution approximation coefficients $a_{0,n}$, and (2) describes a subband coding algorithm on these sampled values, with low-pass filter h and high-pass filter g. Because of their association with orthonormal wavelet bases, these filters give exact reconstruction, i.e.:

$$a_{m-1,l}(f) = \sum_n [h_{2n-l}a_{m,n}(f) + g_{2n-l}c_{m,n}(f)]. \tag{3}$$

Most of the orthonormal wavelet bases have infinitely supported ψ, corresponding to filters h and g with infinitely many taps. The construction in [16] gives ψ with finite support, and therefore, corresponds to FIR filters. It follows that the orthonormal bases in [16] correspond to a subband coding scheme with exact reconstruction property, using the same FIR filters for reconstruction as for decomposition. Such filters are well known since the work of Smith and Barnwell [35] and of Vetterli [37]. The extra ingredient in the orthonormal wavelet decomposition is that it writes the signal to be decomposed as a superposition of reasonably *smooth* elementary building blocks. The filters must satisfy the additional condition:

$$\prod_{k=1}^{\infty} H(2^{-k}\xi)$$

decay faster than $C(1 + |\xi|)^{-\epsilon - 0.5}$ as $|\xi| \to \infty$, for some $\epsilon > 0$, where

$$H(\xi) = 2^{-1/2}\sum_n h_n e^{-jn\xi}.$$

This extra regularity requirement is usually not satisfied by the exact reconstruction filters in the ASSP literature.

B. Applications of Wavelet Bases to Image Analysis

1) Biorthogonal Wavelet Bases: Since images are mostly smooth (except for occasional edges) it seems appropriate that an exact reconstruction subband coding scheme for image analysis should correspond to an orthonormal basis with a reasonably smooth mother wavelet. In order to have fast computation, the filters should be short (short filters lead to less smoothness, however, so they cannot be too short). On the other hand it is desirable that the FIR filters used be linear phase, since such filters can be easily cascaded in pyramidal filter structures without the need for phase compensation. Unfortunately, there are no nontrivial orthonormal linear phase FIR filters with the exact reconstruction property [35], regardless of any regularity considerations. The only symmetric exact reconstruction filters are those corresponding to the Haar basis, i.e., $h_0 = h_1 = 2^{1/2}$ and $g_0 = -g_1 = 2^{1/2}$, with all other h_n, $g_n = 0$.

One can preserve linear phase (corresponding to symmetry for the wavelet) by relaxing the orthonormality requirement, and using biorthogonal bases. It is then still possible to construct examples where the mother wavelets have arbitrarily high regularity.

In such a scheme, we still decompose as in (2), but reconstruction becomes

$$a_{m-1,l}(f) = \sum_n [\tilde{h}_{2n-l}a_{m,n}(f) + \tilde{g}_{2n-l}c_{m,n}(f)] \tag{4}$$

where the filters $\tilde{h}$, $\tilde{g}$ may be different from h, g. In order to have exact reconstruction, we impose:

$$\begin{aligned} \tilde{g}_n &= (-1)^n h_{-n+1} \\ g_n &= (-1)^n \tilde{h}_{-n+1} \end{aligned} \qquad \sum_n h_n\tilde{h}_{n+2k} = \delta_{k,0}. \tag{5}$$

So far, we have not performed anything differently from the usual exact reconstruction subband coding schemes with synthesis filters different from the decomposition filters. If the filters satisfy the additional condition that:

$$\prod_{k=1}^{\infty} \tilde{H}(2^{-k}\xi) \quad \text{and} \quad \prod_{k=1}^{\infty} H(2^{-k}\xi) \tag{6a}$$

decay faster than $C(1 + |\xi|)^{-\epsilon - 0.5}$ as $|\xi| \to \infty$, for some $\epsilon > 0$, where

$$\tilde{H}(\xi) = 2^{-1/2} \sum_n \tilde{h}_n e^{-jn\xi} \quad H(\xi) = 2^{-1/2} \sum_n h_n e^{-jn\xi} \tag{6b}$$

then we can give the following interpretation to (2) and (4). Define functions ϕ and $\tilde{\phi}$ by

$$\phi(x) = \sum_n h_n \phi(2x - n) \quad \text{and} \quad \tilde{\phi}(x) = \sum_n \tilde{h}_n \tilde{\phi}(2x - n).$$

Their Fourier transforms are exactly the infinite products (6a), and they are, therefore, well-defined square integrable functions, compactly supported if the filters h and $\tilde{h}$ are FIR. Define also

$$\psi(x) = \sum_n g_n \phi(2x - n) \quad \text{and} \quad \tilde{\psi}(x) = \sum_n \tilde{g}_n \tilde{\phi}(2x - n).$$

Then, the $a_{m,n}(f)$ and $c_{m,n}(f)$ in (2) can be rewritten as:

$$a_{m,n}(f) = \langle \phi_{m,n}, f \rangle = 2^{-m/2} \int dx\, \phi_{m,n}(x) f(x)$$

$$c_{m,n}(f) = \langle \psi_{m,n}, f \rangle = 2^{-m/2} \int dx\, \psi_{m,n}(x) f(x)$$

and reconstruction is simply:

$$f = \sum_{m,n} \langle \psi_{m,n}, f \rangle \tilde{\psi}_{m,n}. \tag{7}$$

The filter bank structure with the associating wavelets and scaling functions is depicted on the following subband coding scheme (Fig. 1).

If the infinite products in (6a) decay even faster than imposed above, then ϕ and $\tilde{\phi}$ and consequently ψ and $\tilde{\psi}$ will be reasonably smooth. Note that (7) is very similar to the orthonormal decomposition described in Section II-A; the only difference is that the expansion of f with respect to the basis $\tilde{\psi}_{m,n}$ uses coefficients computed via the dual basis $\psi_{m,n}$ with $\tilde{\psi}$ different from ψ. This interpretation is not possible for all exact reconstruction subband coding schemes; in particular, convergence of the infinite products (6a) is only possible if

$$\sum_n h_n = 2^{1/2} \quad \text{and} \quad \sum_n \tilde{h}_n = 2^{1/2}.$$

Moreover, (7) can only hold if

$$\sum_n (-1)^n h_n = 0 \quad \text{and} \quad \sum_n (-1)^n \tilde{h}_n = 0.$$

Most exact reconstruction subband coding schemes do not satisfy these conditions.

Biorthogonal bases of wavelets have recently been constructed, with regularity simultaneously but independently, by Cohen, Daubechies and Feauveau [12] and by Herley and Vetterli [38]. Reference [12] contains a detailed mathematical study, with proofs that, under the conditions stated above, the wavelets do indeed constitute numerically stable bases (Riesz bases) and a discussion of necessary and sufficient conditions for regularity. In [18] Feauveau explores the construction from the point of view of multiresolution spaces rather than from the filters. Basically one has two hierarchies of spaces in the biorthogonal case, each corresponding to one pair of filters.

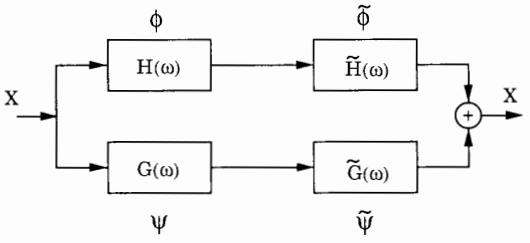


Fig. 1. Filter bank structure and the associating wavelets.

It is shown in [12] that arbitrarily high regularity can be achieved by both ψ and $\tilde{\psi}$, provided one chooses sufficiently long filters. In particular, if the functions ψ and $\tilde{\psi}$ are, respectively, $(k - 1)$ and $(\tilde{k} - 1)$ times continuously differentiable, then the trigonometric polynomials $H(\xi)$ and $\tilde{H}(\xi)$ have to be divisible by $(1 + e^{-j\xi})^k$ and $(1 + e^{-j\xi})^{\tilde{k}}$, respectively, so that the length of the corresponding filters h, $\tilde{h}$ has to exceed k, $\tilde{k}$.

By (5), divisibility of $\tilde{H}(\xi)$ by $(1 + e^{-j\xi})^{\tilde{k}}$ means that ψ will have $\tilde{k}$ consecutive moments zero:

$$\int dx\, x^l \psi(x) = 0, \qquad \text{for} \quad l = 0, 1, \cdots, \tilde{k} - 1.$$

For more details concerning this discussion, see [12].

It is well known (and it can easily be checked by using Taylor expansions) that if ψ has $\tilde{k}$ moments zero, then the coefficients $\langle \psi_{m,n}, f \rangle$ will represent functions f, which are $\tilde{k}$ times differentiable, with a high compression potential (many coefficients will be negligibly small).

Many examples of biorthogonal wavelet bases with reasonably regular ψ and $\tilde{\psi}$ can be constructed; for our applications, the regularity of the elementary building blocks $\tilde{\psi}_{m,n}$, which is linked to the number of zero moments of ψ, is more important than the regularity of the $\psi_{m,n}$ or the number of zero moments of $\tilde{\psi}$. Within the limits imposed by the support widths, we will, therefore, try to choose $\tilde{k}$ as large as possible.

In terms of trigonometric polynomials $H(\xi)$ and $\tilde{H}(\xi)$, the exact reconstruction requirement condition on h and $\tilde{h}$ given in (5) reduces to (for symmetric filters)

$$H(\xi)\tilde{H}(\xi) + H(\xi + \pi)\tilde{H}(\xi + \pi) = 1. \tag{8}$$

Together with divisibility of H and $\tilde{H}$, respectively, by $(1 + e^{-j\xi})^k$ and $(1 + e^{-j\xi})^{\tilde{k}}$, this leads to (see [12])

$$H(\xi)\tilde{H}(\xi) = \cos(\xi/2)^{2l} \left[\sum_{p=0}^{l-1} \binom{l - 1 + p}{p} \cdot \sin(\xi/2)^{2p} + \sin(\xi/2)^{2l} R(\xi) \right] \tag{9}$$

where $R(\xi)$ is an odd polynomial in $\cos(\xi)$, and where $2l = k + \tilde{k}$ (symmetry of h and $\tilde{h}$ forces $k + \tilde{k}$ to be even).

TABLE I
FILTER COEFFICIENTS FOR THE SPLINE FILTERS WITH $l = 3$, $k = 4$, $\tilde{k} = 2$

n	0	±1	±2	±3	±4
$2^{-1/2}h_n$	45/64	19/64	−1/8	−3/64	3/128
$2^{-1/2}\tilde{h}_n$	1/2	1/4	0	0	0

Many examples are possible. We have studied in particular the following three examples, which belong to three different families.

2) Spline Filters: One can choose, e.g., $R \equiv 0$, with $\tilde{H}(\xi) = \cos(\xi/2)^{\tilde{k}} e^{-j\kappa\xi/2}$ where $\kappa = 0$ if $\tilde{k}$ is even, $\kappa = 1$ if $\tilde{k}$ is odd. This corresponds to the filters called "spline filters" in [12] (because the corresponding function $\tilde{\phi}$ is a B-spline function) or "binomial filters" in [38] (because the $\tilde{h}$ are simply binomial coefficients). It then follows that:

$$H(\xi) = \cos(\xi/2)^{2l-\tilde{k}} e^{j\kappa\xi/2} \cdot \left[\sum_{p=0}^{l-1} \binom{l-1+p}{p} \sin(\xi/2)^{2p}\right]. \quad (10)$$

We have looked at one example from this family; it corresponds to $l = 3$, $\tilde{k} = 2$. The coefficients h_n and $\tilde{h}_n$ are listed in Table I; the corresponding scaling functions and wavelets are plotted in Fig. 2.

It is clear that the two filters in the first example have very uneven length. This is typical for all the examples in this family of "spline filters."

3) A Spline Variant with Less Dissimilar Lengths: This family still uses $R \equiv 0$ in (9), but factorizes the right-hand side of (9), breaking up the polynomial of degree $l-1$ in $\sin(\xi/2)$ into a product of two polynomials in $\sin(\xi/2)$ with real coefficients, one to be allocated to H, the other to $\tilde{H}$, so as to make the lengths of h and $\tilde{h}$ as close as possible.

The example presented here is the "smallest" one in this family (shortest h and $\tilde{h}$); it corresponds to $l = 4$ and $k = 4$. The filter coefficients are listed in Table II; the corresponding scaling functions and wavelets are plotted in Fig. 3.

Note that, unlike examples 1 and 3 where the $2^{-1/2}h_n$, $2^{-1/2}\tilde{h}_n$ are rational, the entries in Table II are truncated decimal expansions of irrational numbers. The functions ϕ in examples 1 and 2 look very similar (compare Figs. 2(a) and 3(a)); a more detailed analysis shows that the one in example 2 is more regular, however. Both correspond to 4 vanishing moments for $\tilde{\psi}$.

4) Filters Close to Orthonormal Filters: Finally, there exist many examples for which $R \neq 0$. In particular there exists a special choice of R for which the two filters are very close to each other, and both very close to an orthonormal wavelet filter.

Surprisingly, for the first example of this series, one of the two filters is a Laplacian pyramid filter proposed in [9]. It corresponds to $l = 2$, $k = 2$ and $R(\xi) = 48\cos(\xi)/175$. The filter coefficients are listed in Table III; the corresponding scaling functions and wavelets are plotted in Fig. 4. It is clear that the scaling functions ϕ and $\tilde{\phi}$ are very similar, corresponding to very similar ψ and $\tilde{\psi}$. Note that in this case, the filter coefficients are again rational.

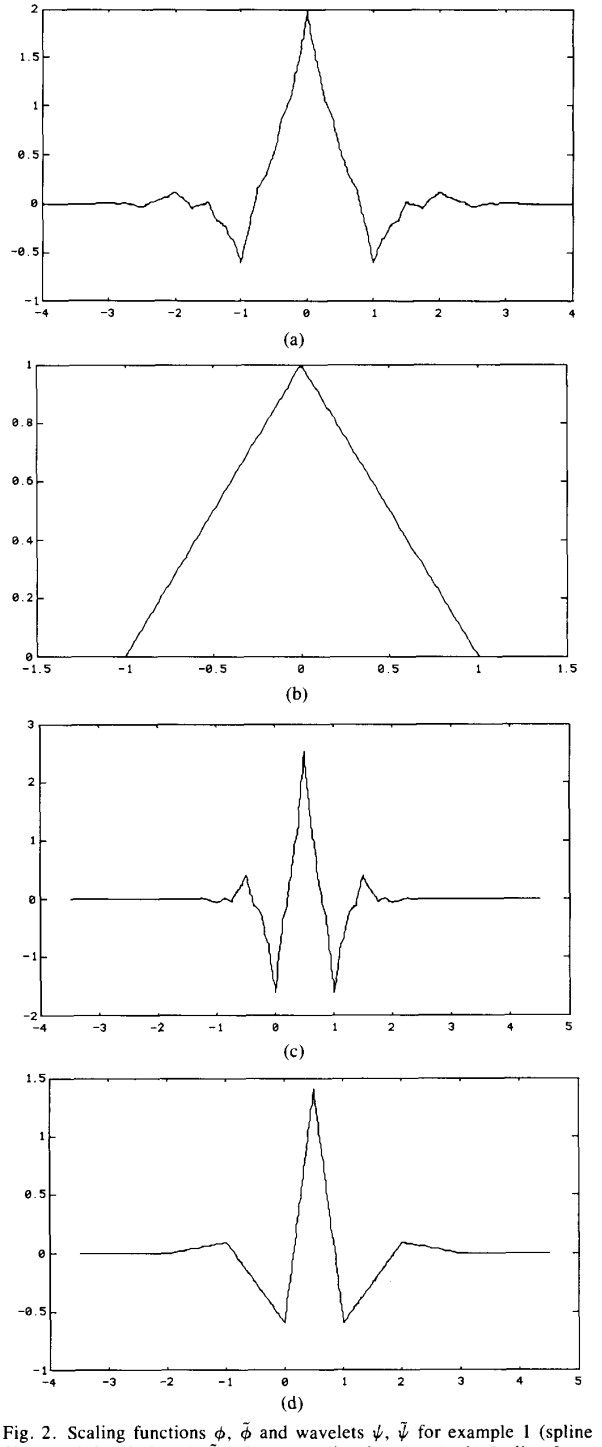


Fig. 2. Scaling functions ϕ, $\tilde{\phi}$ and wavelets ψ, $\tilde{\psi}$ for example 1 (spline filters with $l = 3$, $k = 4$, $\tilde{k} = 2$). (a) Scaling function ϕ. (b) Scaling function $\tilde{\phi}$. (c) Wavelet ψ. (d) Wavelet $\tilde{\psi}$.

TABLE II
FILTER COEFFICIENTS FOR THE SPLINE VARIANT WITH LESS DISSIMILAR LENGTHS, WITH $l = 4 = k$, $\tilde{k} = 4$

n	0	± 1	± 2	± 3	± 4
$2^{-1/2}h_n$	0.602 949	0.266 864	−0.078 223	−0.016 864	0.026 749
$2^{-1/2}\tilde{h}_n$	0.557 543	0.295 636	−0.028 772	−0.045 636	0

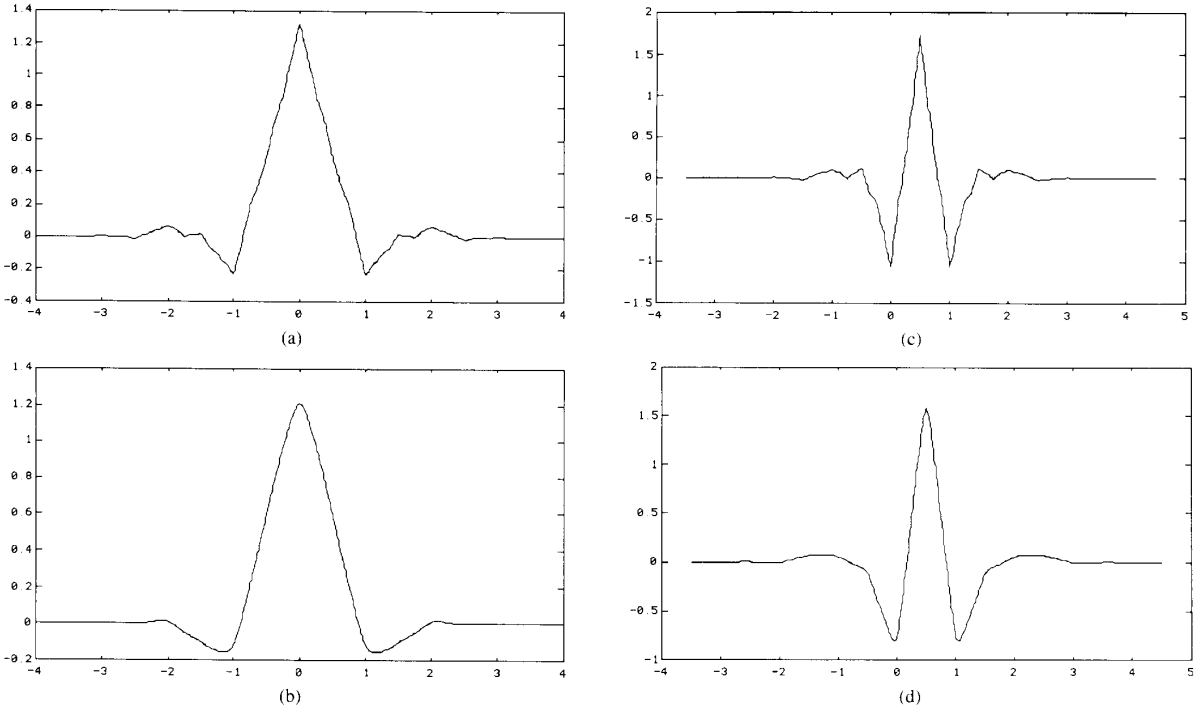


Fig. 3. Scaling functions ϕ, $\tilde{\phi}$ and wavelets ψ, $\tilde{\psi}$ for example 2 (spline variant with less dissimilar lengths; $l = 4 = k$, $\tilde{k} = 4$). (a) Scaling function ϕ. (b) Scaling function $\tilde{\phi}$. (c) Wavelet ψ. (d) Wavelet $\tilde{\psi}$.

TABLE III
FILTER COEFFICIENTS FOR EXAMPLE 3. THE ENTRIES ARE RATIONAL, AND THE TWO FILTERS ARE VERY CLOSE. THE h-FILTER COINCIDES WITH A LAPLACIAN PYRAMID FILTER PROPOSED IN [9]. IN THIS CASE $l = 2 = k$, $\tilde{k} = 2$

n	0	± 1	± 2	± 3	± 4
$2^{-1/2}h_n$	0.6	0.25	−0.05	0	0
$2^{-1/2}\tilde{h}_n$	17/28	73/280	−3/56	−3/280	0

The two biorthogonal filters in this example are both close to an orthonormal wavelet filter of length 6 constructed in [17], where it was called a "coiflet." Being an orthonormal wavelet filter, the coiflet is nonsymmetric. The filters in this example are shorter than in examples 1 and 2, but k is also smaller. The next example in this family corresponds to $k = 4$ (and $l = 4$); the filters h and $\tilde{h}$ then have length 9 and 15; they are both close to a coiflet of length 12.

5) Extension to the Two-Dimensional Case: There exist various extensions of the one-dimensional wavelet transform to higher dimensions. We follow Mallat [27] and use a two-dimensional wavelet transform in which horizontal and vertical orientations are considered preferential.

In two-dimensional wavelet analysis one introduces, like in the one-dimensional case, a scaling function $\phi(x, y)$ such that:

$$\phi(x, y) = \phi(x)\phi(y) \tag{11}$$

where $\phi(x)$ is a one-dimensional scaling function.

Let $\psi(x)$ be the one-dimensional wavelet associated with the scaling function $\phi(x)$. Then, the three two-dimensional wavelets are defined as:

$$\begin{aligned} \psi^H(x, y) &= \phi(x)\psi(y) \\ \psi^V(x, y) &= \psi(x)\phi(y) \\ \psi^D(x, y) &= \psi(x)\psi(y). \end{aligned} \tag{12}$$

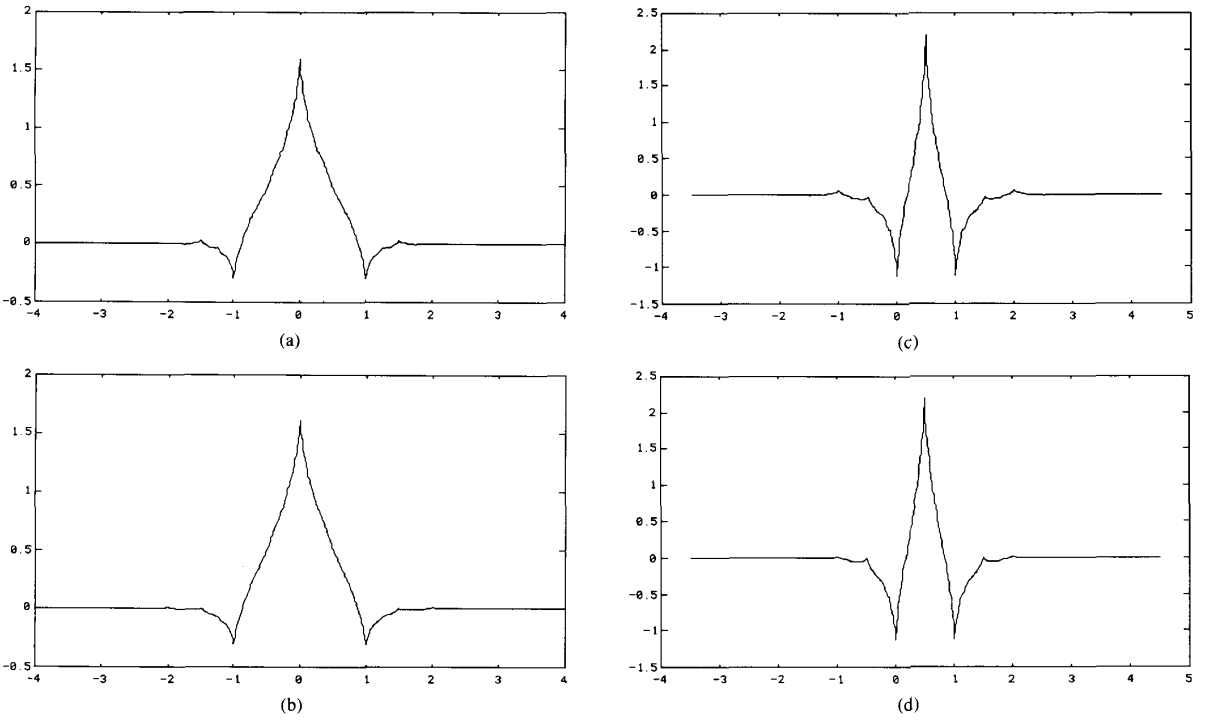


Fig. 4. Scaling functions ϕ, $\tilde{\phi}$ and wavelets ψ, $\tilde{\psi}$ for example 3 (biorthogonal filters close to an orthonormal wavelet filter, $l = 2 = k$, $\tilde{k} = 2$). (a) Scaling function ϕ. (b) Scaling function $\tilde{\phi}$. (c) Wavelet ψ. (d) Wavelet $\tilde{\psi}$.

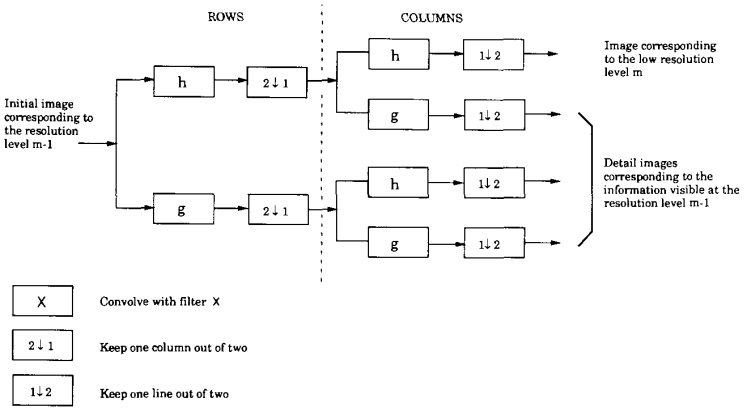


Fig. 5. One stage in a multiscale image decomposition.

Fig. 5 represents one stage in a *multiscale pyramidal* decomposition of an image: wavelet coefficients of the image are computed, as in the one-dimensional case (Sections II-A and II-B.1), using a subband coding algorithm. The filters h and g are one-dimensional filters. This decomposition provides subimages corresponding to different resolution levels and orientations (see Fig. 6). The reconstruction scheme of the image is presented Fig. 7.

To compare the three different filters presented in this paper, we have decomposed the image Lena (Fig. 16) with each of these filters. The results are presented in Fig. 8. In Fig. 8(a) we can see the normalized detail subimages at different resolution levels $m = 1$, $m = 2$, and $m = 3$ (wavelet coefficients) and in Fig. 8(b) the low resolution level subimages.

III. Image Coding Application

A. Statistical Properties of Wavelet Coefficients

The performance of a coder used for a given resolution and direction can be determined by the statistics of the corresponding subimage, i.e., its probability density function (PDF).

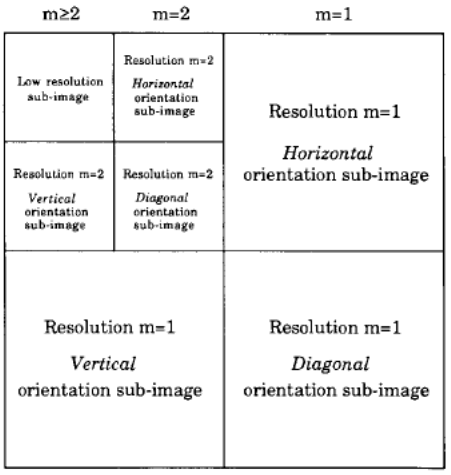


Fig. 6. Image decomposition.

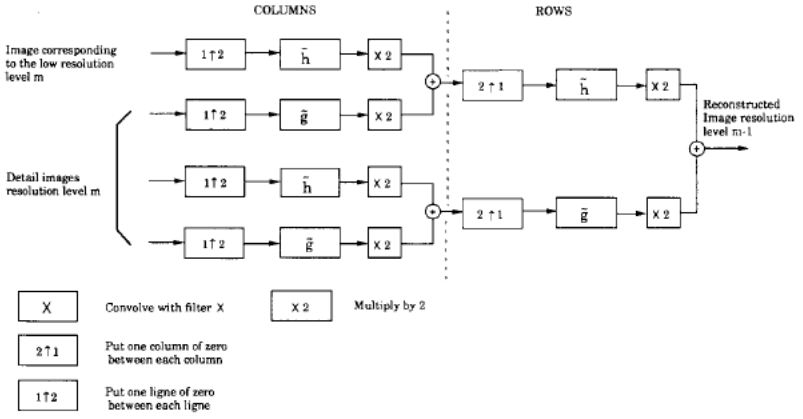


Fig. 7. One stage in a multiscale image reconstruction.

A typical PDF and different approximations are given in Fig. 9, where we plot the true PDF for resolution level $m = 1$ and direction d = vertical together with three model functions: a Gaussian, a Laplacian, and an intermediate function, the so-called generalized Gaussian [2].

This generalized Gaussian law is given explicitly by

$$p_{m,d}(x) = a_{m,d} \exp\left(-|b_{m,d}x|^{r_{m,d}}\right)$$

with

$$a_{m,d} = \frac{b_{m,d} r_{m,d}}{2\Gamma\left(\frac{1}{r_{m,d}}\right)} \quad \text{and} \quad b_{m,d} = \frac{1}{\sigma_{m,d}} \frac{\Gamma\left(\frac{3}{r_{m,d}}\right)^{1/2}}{\Gamma\left(\frac{1}{r_{m,d}}\right)^{1/2}} \tag{13}$$

where $\sigma_{m,d}$ is the standard deviation of the subimage (m, d), and $\Gamma(\cdot)$ is the usual Gamma function.

The general formula (13) contains the other two examples as particular cases:

- $r_{m,d} = 2$ leads to the well-known Gaussian PDF;
- $r_{m,d} = 1$ leads to a Laplacian PDF.

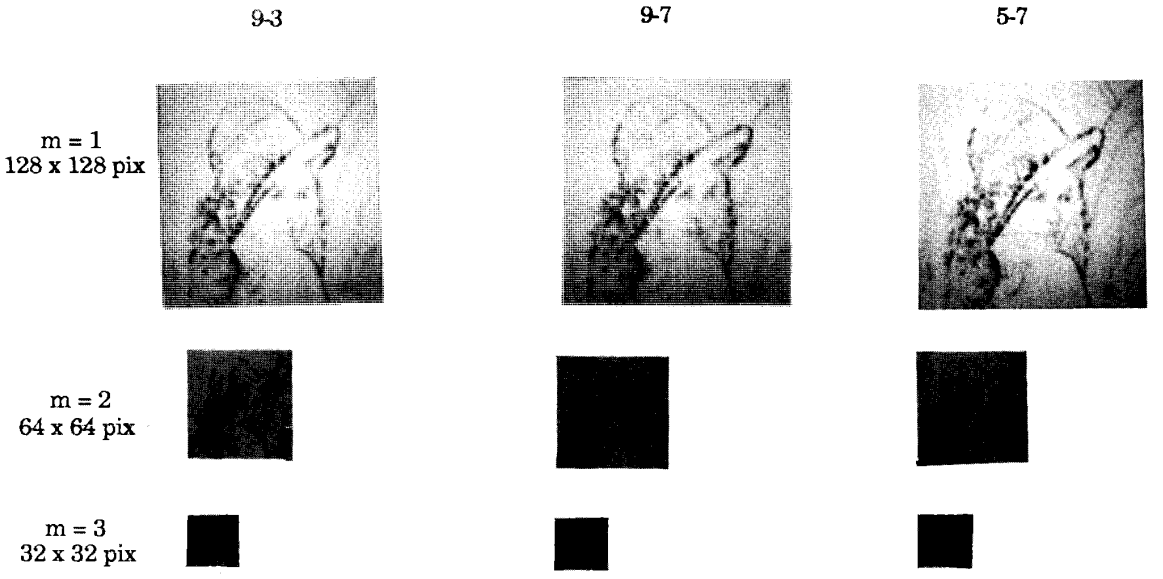
The variance of this approximation model is set equal to the variance of the corresponding subimage. Thus the parameter $r_{m,d}$ is computed in order to match the real PDF using the well-known chi-squared test. In this case the optimum parameter was 0.7. Other experiments for other resolutions (except the lowest resolution) lead to very similar results.

We can see in Fig. 9 that the real PDF (scale $m = 1$ and vertical orientation) is closely approximated by a generalized Gaussian law with parameter $r_{1,t} = 0.7$.

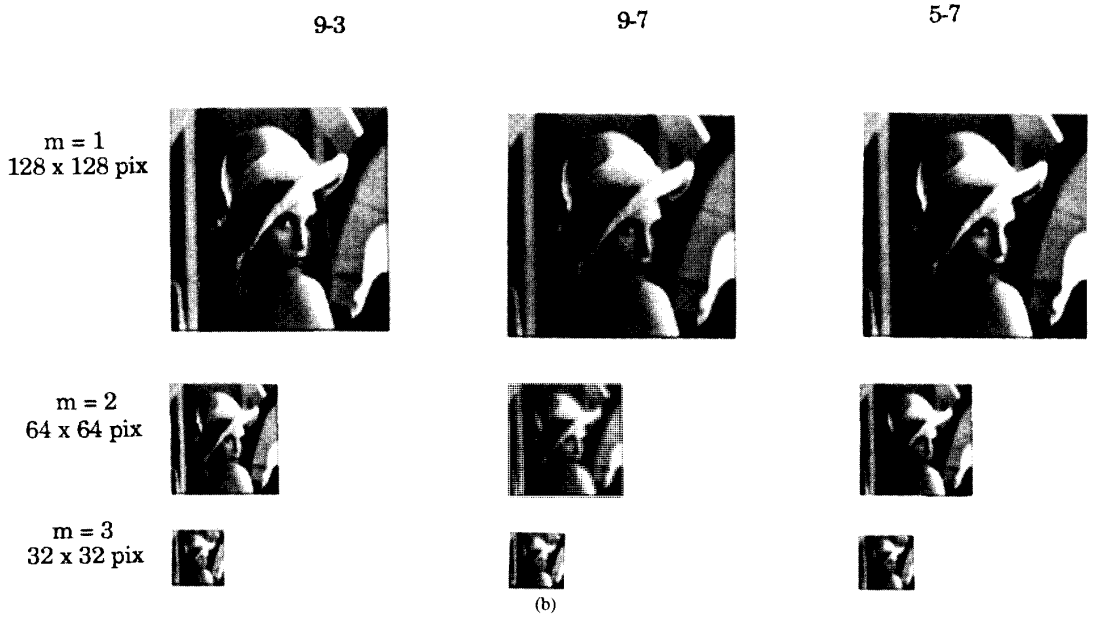
B. Encoding of Wavelet Coefficients Using Vector Quantization

Different techniques involving vector or scalar quantization can be used to encode wavelet coefficients.

According to Shannon's rate distortion theory, better results are always obtained when vectors rather than sca-



(a)



(b)

Fig. 8. Comparison among the different subimages. (a) Comparison among the normalized detail subimages. (b) Comparison among the low resolution level subimages.

lars are encoded. Therefore, the present application uses vector quantization.

1. Principle of Vector Quantization: Developed recently by Gersho and Gray (1980) [20], [21], vector quantization has proven to be a powerful tool for digital image compression [4], [29], [30], [32], [39]. The principle involves encoding a sequence of samples (vector) rather than encoding each sample individually. Encoding is performed by approximating the sequence to be coded by a vector belonging to a catalogue of shapes, usually known as a codebook.

The codebook is created and optimized using the well-known Linde–Buzo–Gray (LBG) [26] classification algorithm with a mean squared error (MSE) criterion. This algorithm is designed to perform a classification based on a training set comprised of vectors belonging to different images; it converges iteratively toward a locally optimal codebook.

Each of the vectors in the codebook is indexed. At the encoding stage, the index of the vector in the codebook most closely describing (in terms of MSE criterion) the sample set to be encoded is selected to represent this set. Of course, in order to reconstruct the sample set, the decoder must have the same codebook as the coder.

The encoding/decoding scheme depicted in Fig. 10 was proposed in [29] and [30] for orthonormal wavelets.

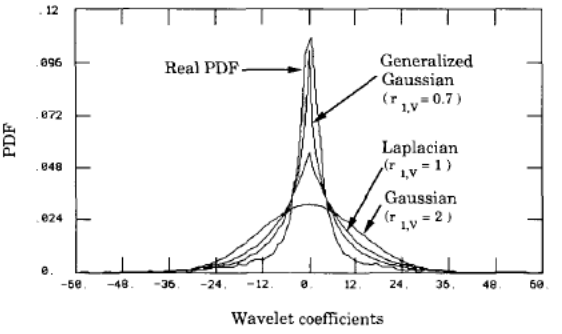


Fig. 9. Real PDF of subimage at scale $m = 1$ for vertical orientation, and its different approximations.

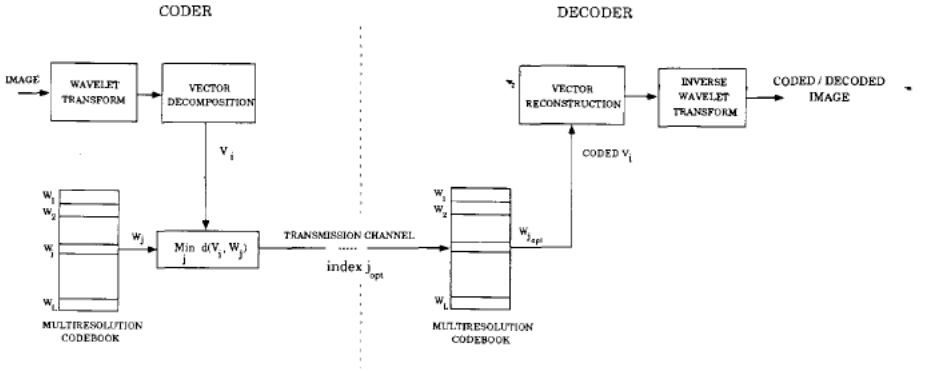


Fig. 10. Encoding/decoding scheme.

2) Comparative Performances of Vector Quantization (VQ) and Scalar Quantization (SQ): According to [3], [13], [19], [43], [30] the asymptotic lower bound distortion gain obtained when VQ, rather than SQ, is applied to a subimage is expressed as:

$$G_{m,d}^{VQ} \geq \frac{2^{-c}}{(c+1)A(k_{m,d}, c)} \times \frac{\left[\int [p_{m,d}(x)]^{1/(c+1)}\, dx\right]^{(c+1)}}{\left[\int [p_{m,d}(x)]^{k_{m,d}/(c+k_{m,d})}\, dx\right]^{(c+k_{m,d})}} \tag{14}$$

for a subimage corresponding to resolution m and direction d. $p_{m,d}(x)$ is the PDF of wavelet coefficients of the subimage with resolution m and direction d.

Here, the MSE criterion is used as a distortion measure ($c = 2$). The values of $A(k_{m,d}, 2)$ used are the upper bounds of the MSE computed and tabulated by Conway and Sloane for vector size $k_{m,d}$ [13]. This formula gives an indication of the minimum theoretical gain that can be obtained.

However, this approximation is valid only for small quantization errors, i.e., for a high bit rate $R_{m,d}$. Thus the gain $G_{m,d}^{VQ}$ only gives here an asymptotic indication.

In Fig. 11, the curves of $G_{m,d}^{VQ}$ are plotted as a function of the vector dimension $k_{m,d}$ for the Laplacian, Gaussian, and generalized Gaussian approximation laws, and for a subimage at scale $m = 1$ and vertical orientation. Experimental results are closely matched by the theoretical results for a generalized Gaussian law with $r_{m,d} = 0.7$ except for the lower subband. Therefore, all computations based on this approximation law show that, in each subband, VQ outperforms SQ (see Fig. 11).

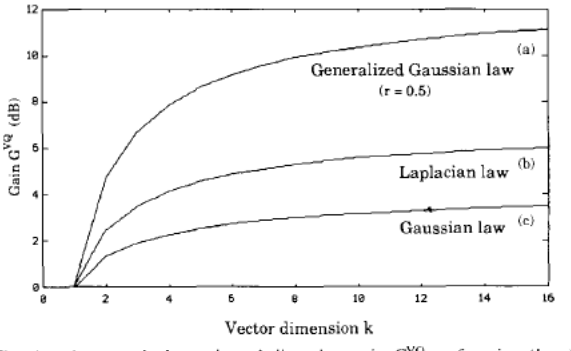


Fig. 11. Asymptotic lower bound distortion gain $G_{m,d}^{VQ}$ = function $(k_{m,d})$.

In summary VQ performs better for coding wavelet coefficients.

3) Generation of a Multiresolution Codebook: The preceding paragraph explained why VQ outperforms other methods. Nonetheless, major problems are encountered in the VQ of images.

- It is impossible to create a universal codebook (efficient for each image to be encoded).

- The LBG algorithm smooths high frequencies (loss of resolution).
- There is a trade-off between low distortion and high compression rate (computational cost).
- It is not easy to take into account the properties of the human visual system [28], [33].

The use of the wavelet transform (i.e., multiresolution) is one way of overcoming these different problems.

The wavelet decomposition of an image enables the generation of a codebook containing two-dimensional vectors for *each resolution level and preferential direction* (horizontal, vertical, and diagonal). Each of these subcodebooks (see Fig. 12) is generated using the LBG algorithm.

- The training set is comprised of vectors belonging to different images corresponding to the resolution and orientation under consideration.
- The initial codebook is generated by splitting the centroid (center of gravity) of this training set [21].

A multiresolution codebook can thus be obtained by assembling all of these resulting subcodebooks. Each subcodebook has a low distortion level and contains few words, which clearly facilitates the search for the best coding vector; the coding computational load is reduced, because only the appropriate subcodebook (resolution direction) of the multiresolution codebook is checked for each input vector. In addition, the quality of the coded image is better. The multiresolution codebook is depicted in Fig. 12.

Global codebook design has drawbacks in that it results in edge smoothing while the proposed method preserves edges. In fact, each subcodebook contains the shape of the wavelet coefficients which are most highly representative in terms of the MSE criterion.

Since the spatial and frequency aspects of the image are taken into account in the wavelet decomposition, the classification and search during the encoding of a subimage vector can be achieved using a simple criterion such as least mean squares. This frees us from using distortion measurements such as weighted least mean squares or other measurements involving perceptual factors. These algorithms are indeed costly in computation time.

C. Optimal Bit Allocation

Multiresolution exploits the eye's masking effects, and therefore, enables us to refine and select the type of coding according to the resolution level and the contour orientation. Although a flat noise shape minimizes the MSE criterion, it is generally not optimal for a subjective quality of image. To apply *noise shaping* across the VQ subimages, we define a total weighted MSE distortion $D_T^*(R_T)$ ((17)) for a total bit rate R_T ((18)).

Let us define $D_{m,d}(R_{m,d})$ the average distortion in the coding of the subimage (m, d) for $R_{m,d}$ bits per pixel:

$$D_{m,d}(R_{m,d}) = E(|x - q(x)|^c) = d(x, q(x)) \qquad c \geq 1 \tag{15}$$

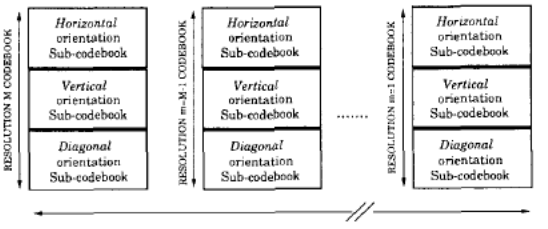


Fig. 12. Multiresolution codebook.

for all coefficients x belonging to the subimage, $q(x)$ being the quantization of x.

Total distortion of the image for a total rate of R_T bits per pixel is then given by:

$$D_T(R_T) = \frac{1}{2^{2M}} D_M^{SQ}(R_M^{SQ}) + \sum_{m=1}^{M} \frac{1}{2^{2m}} \sum_{d=1}^{3} D_{m,d}(R_{m,d}) \tag{16}$$

where $D_M^{SQ}(R_M^{SQ})$ corresponds to the distortion in the subimage of lowest resolution M (texture subimage).

The problem of finding an optimal bit assignment (in bits per pixel) for each subimage vector quantizer is then formulated as:

$$\operatorname*{Min}_{R_{m,d}} \left[D_T^*(R_T) = \frac{1}{2^{2M}} D_M^{SQ}(R_M^{SQ}) + \sum_{m=1}^{M} \frac{1}{2^{2m}} \sum_{d=1}^{3} D_{m,d}(R_{m,d}) \times B_{m,d} \right] \tag{17}$$

$$\textbf{subject to: } R_T = \frac{1}{2^{2M}} R_M^{SQ} + \sum_{m=1}^{M} \frac{1}{2^{2m}} \sum_{d=1}^{3} R_{m,d} \tag{18}$$

where R_M^{SQ} corresponds to the bit allocation, in bits per pixel, of lowest resolution M subimage.

Assignment of the weights is based on the fact that the human eye is not equally sensitive to signals at all spatial frequencies. On the basis of contrast sensitivity data collected by Campbell and Robson [10], and to obtain a controlled degree of noise shaping across the subimages, we consider a function $B_{m,d}$ such that:

$$B_{m,d} = \gamma^m \log(\sigma_{m,d}^{2\beta_{m,d}}) \tag{19}$$

where $\sigma_{m,d}$ is the standard deviation corresponding to subimage (m, d) and the values of γ and $\beta_{m,d}$ are chosen experimentally in order to match human vision.

$D_T^*(R_T)$ is the total weighted encoding distortion function, and M is the lowest resolution considered.

The expression of $D_{m,d}(R_{m,d})$ is given by [19]

$$D_{m,d}(R_{m,d}) = 2^{-cR_{m,d}} \times \alpha_{m,d}(p, c), \qquad c \geq 1$$

with

$$\alpha_{m,d}(p, c) = A(k_{m,d}, c) \times \left[\int [p_{m,d}(x)]^{k_{m,d}/(c + k_{m,d})} \, dx \right]^{(c + k_{m,d})}. \tag{20}$$

This minimization problem can be solved by using Lagrangian multipliers. Using this technique, we must solve the following equation:

$$\frac{\partial}{\partial R_{m,d}}\left[D_T^*(R_T) - \lambda\left(R_T - \frac{1}{2^{2M}} R_M^{SQ} - \sum_{m=1}^{M} \frac{1}{2^{2m}} \sum_{d=1}^{3} R_{m,d}\right)\right] = 0 \tag{21}$$

where λ is a Lagrangian multiplier.

Using (17) and (20), this equation becomes:

$$\frac{\partial}{\partial R_{m,d}}\left[\frac{1}{2^{2m}} D_M^{SQ}(R_M^{SQ}) + \sum_{m=1}^{M} \frac{1}{2^{2m}} \sum_{d=1}^{3} (2^{-cR_{m,d}} \alpha_{m,d}(p, c) B_{m,d}) - \lambda\left(R_T - \frac{1}{2^{2M}} R_M^{SQ} - \sum_{m=1}^{M} \frac{1}{2^{2m}} \sum_{d=1}^{3} R_{m,d}\right)\right] = 0. \tag{22}$$

Taking the partial derivative with respect to $R_{m,d}$ yields an expression for $R_{m,d}$ in terms of λ:

$$R_{m,d} = \frac{1}{c} \log_2 \left[\frac{(c \ln 2)\alpha_{m,d}(p, c) B_{m,d}}{\lambda}\right]. \tag{23}$$

By substituting (23) into the constraint (18) of the minimization problem we obtain an expression of the Lagrangian multiplier λ

$$\lambda = c \ln 2 \left[2^{-c(R_T - (1/4^M) R_M^{SQ})} \prod_{m=1}^{M} \prod_{d=1}^{3} \cdot [\alpha_{m,d}(p, c) B_{m,d}]^{1/4^m}\right]^{4^M/4^M - 1}. \tag{24}$$

Finally, substituting λ into (23) results in an expression of the optimal bit assignment $R_{m,d_{opt}}$ (in bits per pixel (bpp)) to the vector quantizer of subimage (m, d):

$$R_{m,d_{opt}} = \frac{4^M R_T - R_M^{SQ}}{4^M - 1} + \frac{1}{c} \log_2 \left[\frac{\alpha_{m,d}(p, c) B_{m,d}}{\left[\prod_{m'=1}^{M} \prod_{d'=1}^{3} [\alpha_{m',d'}(p, c) B_{m',d'}]^{1/4^{m'}}\right]^{4^M/4^M - 1}}\right]. \tag{25}$$

This expression requires the knowledge of the subimage's PDF's.

The optimal distortion of the quantizer, $D_{T_{opt}}^*(R_T)$, is then computed by combining (25) and (17). We find:

$$D_{T_{opt}}^*(R_T) = \frac{1}{2^{2M}} D_M^{SQ}(R_M^{SQ}) + \frac{4^M - 1}{4^M} 2^{-c(4^M R_T - R_M^{SQ})/4^M - 1} \cdot \left[\prod_{m=1}^{M} \sum_{d=1}^{3} [\alpha_{m,d}(p, c) B_{m,d}]^{1/4^m}\right]^{4^M/4^M - 1}. \tag{26}$$

Finally, bit allocation which is a function of the image will be transmitted as side information requiring only a few bits.

IV. Experimental Results

The images used are sampled 256 by 256 black and white images. The intensity of each pixel is coded on 256 grey levels (8 bpp).

The numerical evaluation of the coder's performance is achieved by computing the peak signal-to-noise ratio (PSNR) between the original image and the coded image.

For each coded image, we can use a variable length code. We also give the corresponding $\mathcal{R}_T$ if an optimal entropy coding was performed, defined as follows.

To the L codewords w_j; $j = 1, 2, \cdots, L$ of the vector quantizer corresponds to L regions (clusters) of $\mathbb{R}^k$, $\mathcal{P}_j$; $j = 1, 2, \cdots, L$. The jth region is defined by

$$\mathcal{P}_j = \{x \in \mathbb{R}^k / Q(x) = w_j\}$$

and represents the subset of vectors of $\mathbb{R}^k$ which are well matched by the codeword w_j of the codebook.

Thus for each resolution and direction, we can introduce the average information of the codebook, called the entropy measure:

$$\mathcal{R}_{m,d} = -\frac{1}{k_{m,d}} \times \sum_{j=1}^{L} p(w_j) \log_2 p(w_j) \text{ bpp}$$

where $p(w_j)$ is the probability of selecting the source vector w_j, belonging to the codebook at scale m and corresponding to the orientation d, during the coding of the image (m, d).

Then, as in (18), $\mathcal{R}_T$ is the sum of the estimated entropy in each subimage as follows:

$$\mathcal{R}_T = \frac{1}{2^{2M}} \mathcal{R}_M^{SQ} + \sum_{m=1}^{M} \frac{1}{2^{2m}} \sum_{d=1}^{3} \mathcal{R}_{m,d} \text{ bpp}.$$

The vector quantizer used is a *full search* quantizer, i.e., during the coding, all of the vectors in the subcodebook corresponding to the resolution and direction to be encoded are searched. The selection criterion used is the MSE criterion.

A. Comparison Between the Different Wavelets

In the following, we present results obtained with the Lena image (image within the training set) for a real bit rate of 1 bpp and using the three different filters proposed in Section II-B. (Fig. 13 corresponds to filters 9-3 presented in example 1, Fig. 14 corresponds to filters 9-7 presented in example 2, and Fig. 15 corresponds to filters 5-7 presented in example 3.) Here, the Lena image is taken as part of the training set in order to minimize the effects of quantization noise: this enables the influence of the filters to be taken into account.

For a given set of filters, separate codebooks are trained for each resolution-orientation subimage, and bit alloca-

Fig. 13. Filters no. 1 9-3, PSNR = 31.82 dB, $\mathcal{R}_T = 0.80$ bpp.

Fig. 15. Filters no. 3, 5-7, PSNR = 31.46 dB, $\mathcal{R}_T = 0.80$ bpp.

Fig. 14. Filters no. 2, 9-7, PSNR = 32.10 dB, $\mathcal{R}_T = 0.78$ bpp.

Fig. 16. Original 256 by 256 Lena, 8 bpp.

tion is carried out according to (25). For the Lena image, the bit assignment is represented in Fig. 17. Resolution 1 (diagonal orientation) is discarded. Resolution 1 (horizontal and vertical orientations) and resolution 2 (diagonal orientation) are coded using 256-vector codebooks (codeword size 4 by 4) resulting in a 0.5-b/pixel rate, while resolution 2 (horizontal and vertical orientations) is coded at a 2-b/pixel rate using 256-vector codebooks (codeword size 2 by 2). Finally, the lowest resolution is coded at 8 b/pixel.

B. Results as a Function of Regularity and Vanishing Moments

In Section II-B, we mentioned our belief that both the regularity of the reconstruction wavelet $\tilde{\psi}$ and the number

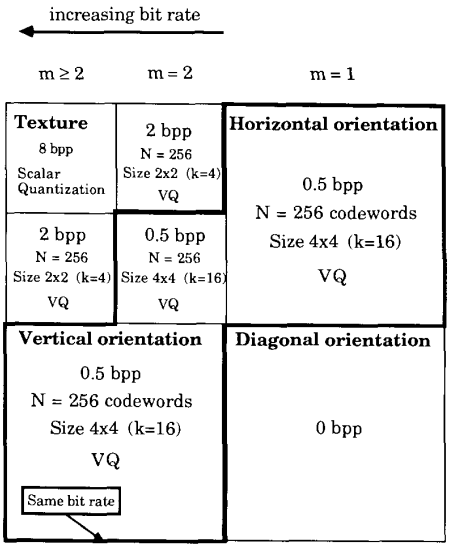


Fig. 17. Subimages bit rate allocation: example of a bit allocation for a total bit rate of 1 bpp and for the 256 by 256 Lena image.

of vanishing moments of the analyzing wavelet ψ are important in applications. To illustrate this we carried out the following experiments. For a given pair, h, $\tilde{h}$, we analyzed the same image twice: once as described above, and a second time after exchanging the roles of the filters h and $\tilde{h}$.

The filter pairs in example 2 both have the same number of vanishing moments, $k = \tilde{k} = 4$. However, $\tilde{\psi}$ is considerably more regular than ψ (see Fig. 3). With this filter pair, our experiment on the Lena image led to a PSNR of 32.10 dB in the first case, and to a PSNR of 31.51 dB if the roles of h and $\tilde{h}$ are inverted. The case where the reconstruction wavelet has the highest regularity, therefore, performs best.

In example 1 the functions ψ and $\tilde{\psi}$ have comparable regularity: both are continuous and neither has a continuous derivative. In fact $\tilde{\psi}$ is a bit more regular than ψ: $\tilde{\psi}$ is differentiable almost everywhere, and is Hölder continuous with exponent 1, while ψ is Hölder continuous with the exponent only at 0.83. On the other hand, ψ has 2 vanishing moments, while $\tilde{\psi}$ has 4 ($k = 4$, $\tilde{k} = 2$). The same experiment, again with the Lena image, now leads to a PSNR of 31.82 dB if h, $\tilde{h}$ are taken as in Table I, and to a PSNR of 31.13 dB when the roles of h and $\tilde{h}$ are reversed. The situation where $\tilde{\psi}$ is most regular but ψ has fewer vanishing moments, therefore, performs better (gain of 0.69 dB) than the case where ψ has more vanishing moments but $\tilde{\psi}$ is less regular. This seems to suggest that the regularity of $\tilde{\psi}$ has a larger effect than the number of vanishing moments of ψ. However, in this example the difference in overall regularity, as measured by the differences between Hölder exponents, is much smaller here than in example 2 (0.17 as compared to 0.63 in example 2), and it seems hard to explain how this smaller difference in Hölder exponent could account for a comparable gain in PSNR. In fact, the Hölder exponent is not a very good measure for the regularity of $\tilde{\psi}$ in this case: it is completely determined by the discontinuity of the derivative of $\tilde{\psi}$ in only a few points, and it is insensitive to the fact that $\tilde{\psi}$ is infinitely differentiable in all other points. If this is taken into account, then $\tilde{\psi}$ looks much more regular than ψ (the Hölder exponent of which is determined by its behavior near a dense set of points), which might explain the gain in PSNR.

We conclude from all this that: 1) for the same number of vanishing moments for ψ, the scheme with most regular $\tilde{\psi}$ is likely to perform best; and 2) increasing the regularity of $\tilde{\psi}$, even at the expense of the number of vanishing moments for ψ, may lead to better results.

Based on theoretical arguments (Taylor expansions) and results from numerical analysis [8], we also expect: 3) for comparable regularity of $\tilde{\psi}$, the scheme with largest vanishing moments for ψ is likely to perform best.

C. Comparison with Other Coders

If the PSNR is chosen as a criterion of comparison, these results are close to those obtained by Woods and O'Neil [42] and Westerink *et al.* [40]. However, in their subband coding algorithm, they use 32-taps Johnston filters, while only 9 or 7 taps are necessary for our method. According to Westerink's results in [41], the PSNR decreases by about 2 dB when using 8-taps Johnston filters. However, some others new QMF designs can also lead to good results with about 9 taps for image coding [1].

In this section, we present both numerical and qualitative comparison between our coding scheme and other previously published results. Since the most popular image in the recent literature has been the 512 by 512 Lena image, the comparison is made using this image taken outside the training set.

Among the different methods published, we consider the three following well-known methods: Ho and Gersho obtained a 30.93-dB PSNR at 0.36 bpp, result using "variable-rate multi stage VQ" [23]. Riskin and Gray improved on the full search VQ (PSNR = 29.29 dB, 0.32 bpp) using pruned tree structured VQ (PSNR = 30.92 dB, 0.32 bpp) [34]. High PSNR values were obtained by Woods and Cohen using entropy coded and predictive VQ (PSNR = 32.5 dB, 0.45 bpp) [11].

Our aim is not to optimize the PSNR but rather a weighted function of the MSE in order to match human vision. We give two examples at low bit rate using wavelet VQ.

Our initial result at 0.37 bpp presented Fig. 18 with a 30.85-dB PSNR is very close to those of Ho and Gersho [23] and Riskin *et al.* [34]. The perceptual quality of our coded images is better than indicated by the PSNR value

Fig. 18. 512 by 512 Lena image. Filters no. 2 9-7, PSNR = 30.85 dB, $\mathcal{R}_T = 0.37$ bpp.

Fig. 19. 512 by 512 Lena image. Filters no. 2 9-7, PSNR = 29.11 dB, $\mathcal{P}_T = 0.21$ bpp.

mainly due to the regularity of the wavelet and the bit allocation. These images do not suffer from the blocking effects obtained when using VQ in the spatial domain. No ringing effects can be observed.

The second result at 0.21 bpp presented in Fig. 19 with a 29.11-dB PSNR shows that a very low bit rate can be achieved with our method, without severe degradation.

Our method using a new class of filters derived from wavelet theory using full search VQ can be improved by any of the three above-mentioned methods.

In fact the LBG clustering algorithm is a very simple algorithm but not optimal for variable length code. The PSNR of the method could be improved by about 3 dB, for example, using ECVQ [34] but CPU time becomes prohibitively expensive.

D. Progressive Transmission Scheme

The main objective of progressive transmission is to allow the receiver to recognize a picture as quickly as possible at minimum cost, by sending a low resolution level picture first. Then, it can be decided to receiver further picture details or to abort the transmission. Further details of the picture are obtained by sequentially receiving the encoded wavelet coefficients at different resolution levels and directions.

Following the example of [40], we will display each picture level during the progressive transmission with a size that matches the resolution of that particular level.

To test the efficiency of the vector quantizer, the image to be coded is taken outside the training set.

Fig. 20 represents 5 stages in the progressive transmission of a 256 by 256 image using filters 9–7 given in example 2. According to the bit allocation procedure (Section III-C) with a generalized Gaussian PDF approximation law, only the wavelet coefficients corresponding to the $m = 1$ and $m = 2$ high resolution levels are vector quantized, while the low level subimages ($m \geq 2$) are scalar quantized.

V. Conclusion

This paper describes a new image coding scheme combining the wavelet transform and VQ.

A new family of filters has been derived from the wavelet theory. We have shown the importance of regularity and vanishing moments for image coding. Furthermore, these filters require few taps, unlike standard QMF methods.

The wavelet transform used here attempts to exploit the masking effect of the human eye, yielding encouraging results. Indeed, the proposed method enables high compression bit rates while maintaining good visual quality through the use of bit allocation in the subimages. The blocking effects seen when spatial VQ is performed are avoided.

This method is well adapted to progressive transmission as well as very low bit rate compression. Furthermore, using a simple full-search VQ provides good results, comparable to the best results published currently.

Further research should include some new derivation such as entropy constraint and predictive VQ. We would improve this coding scheme, if we accept a heavier computational load.

Resolution	*m=4*	*m=3*	*m=2*	*m=1*
Size	**16 × 16 pix**	**32 × 32 pix**	**64 × 64 pix**	**128 × 128 pix**
R_T	0.031 bpp	0.125 bpp	0.5 bpp	0.781 bpp
$\mathcal{R}_T$	0.0264 bpp	0.0919 bpp	0.3354 bpp	0.5039 bpp

Resolution m=0

256 × 256 pix

R_T = 1 bpp $\mathcal{R}_T$ = 0.6297 bpp

PSNR = 31.28 dB

Fig. 20. Progressive transmission-filters no. 2 9-7.

REFERENCES

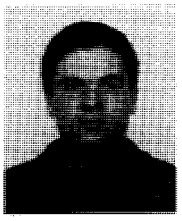
[1] E. H. Adelson and E. Simoncelli, "Non-separable extensions of quadrature mirror filters to multiple dimensions," *Proc. IEEE*, vol. 78, Apr. 1990.

[2] M. Abramowitz, I. A. Stegun, *Handbook of Mathematical Functions*. New York: Dover, 1965.

[3] V. R. Algazi, "Useful approximation to optimum quantization," *IEEE Trans. Commun.*, vol. COM-14, pp. 297-301, June 1966.

[4] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image

coding using vector quantization in the wavelet transform domain," in *Proc. IEEE ICASSP*, April 1990, pp. 2297-2300.
[5] M. Barlaud, L. Blanc-Féraud, P. Mathieu, J. Menez, and M. Antonini, "2D linear predictive image coding with vector quantization," in *Proc. EUSIPCO*, Grenoble, France, Sept. 5-8, 1988, pp. 1637-1640.
[6] M. Barlaud, P. Mathieu, and M. Antonini, "Wavelet transform image coding using vector quantization," presented at 6th Workshop on MDSP, Monterey, CA, Sept. 1989.
[7] G. Battle, "A block spin construction of wavelets. Part I Lemarié functions," *Comm. Math. Phys.*, vol. 110, pp. 601-615, 1987.
[8] G. Beylkin, R. Coifman, and V. Rokhlin, "Fast wavelet transforms and numerical analysis. I," to be published.
[9] P. Burt and E. Adelson, "The Laplacian pyramid as a compact image code," *IEEE Trans. Commun.*, vol. 31, pp. 482-540, 1983.
[10] F. W. Campbell and J. G. Robson, "Application of Fourier analysis to the visibility of gratings," *J. Phys.*, vol. 197, pp. 551-566, 1968.
[11] R. A. Cohen and J. W. Woods, "Sliding block entropy coding of images," in *Proc. IEEE ICASSP*, Glasgow, Scotland, May 23-26, 1989, pp. 1731-1733.
[12] A. Cohen, I. Daubechies, and J. C. Feauveau, "Biorthogonal bases of compactly supported wavelets," AT&T Bell Lab., Tech. Rep., TM 11217-900529-07, 1990.
[13] J. H. Conway and N. J. A. Sloane, "A lower bound on the average error of vector quantizers," *IEEE Trans. Inform. Theory*, vol. IT-31, pp. 106-109, Jan. 1985.
[14] I. Daubechies, A. Grossman, and Y. Meyer, "Painless nonorthogonal expansions," *J. Math. Phys.*, vol. 27, pp. 1271-1283, 1986.
[15] I. Daubechies, "The wavelet transform, time-frequency localization and signal analysis," to be published.
[16] ——, "Orthonormal bases of compactly supported wavelets," *Comm. Pure Appl. Math.*, vol. 41, pp. 909-996, 1988.
[17] ——, "Orthonormal bases of compactly supported wavelets. II. Variations on a theme," AT&T Bell Lab., Tech. Rep. TM 11217-891116-17, 1990.
[18] J. C. Feauveau, "Analyse multirésolution par ondelettes non orthogonales et bancs de filtres numériques," Ph.D. dissertation, Univ. Paris Sud, France, Jan. 1990.
[19] A. Gersho, "Asymptotically optimal block quantization," *IEEE Trans. Inform. Theory*, vol. IT-25, July 1979.
[20] ——, "On the structure of vector quantizers," *IEEE Trans. Inform. Theory*, vol. IT-28, Mar. 1982.
[21] R. M. Gray, "Vector quantization," *IEEE ASSP Mag.*, pp. 4-29, Apr. 1984.
[22] A. Grossman and J. Morlet, "Decomposition of hardy functions into square integrable wavelets of constant shape," *SIAM J. Math Anal.*, vol. 15, pp. 723-736, 1984.
[23] Y. Ho and A. Gersho, "Variable-rate multi-stage vector quantization for image coding," in *Proc. IEEE ICASSP*, New York, Apr. 1988.
[24] P. G. Lemarié, "Une nouvelle base d'ondelettes de $L^2(\mathbb{R})$," *J. Math. Pures et Appl.*, vol. 67, pp. 227-238, 1988.
[25] P. G. Lemarié and Y. Meyer, "Ondelettes et bases hilbertiennes," *Rev. Mat. Iberoamericana*, vol. 2, pp. 1-18, 1986.
[26] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COM-28, pp. 84-95, Jan. 1980.
[27] S. Mallat, "A theory for multiresolution signal decomposition: The wavelet representation," *IEEE Trans. Pattern Anal. Mach. Intel.*, vol. 11, July 89.
[28] D. Marr, *Vision*. New York: Freeman, 1982.
[29] P. Mathieu, M. Barlaud, and M. Antonini, "Compression d'Images par transformée en ondelette," *12ième colloque GRETSI, Juan les Pins*, June 12-16, 1989.
[30] P. Mathieu, M. Barlaud, and M. Antonini, "Compression d'Image par transformée en ondelette et quantification vectorielle," *Traitement du Signal*, vol. 7, no. 2, 1990.
[31] Y. Meyer, "Principe d'incertitude, bases hilbertiennes et algèbres d'opérateurs," *Seminaire Bourbaki*, no. 662, 1985-1986.
[32] N. M. Nasrabadi and R. A. King, "Image coding using vector quantization: A review," *IEEE Trans. Commun.*, vol. 36, Aug. 1988.
[33] W. K. Pratt, *Digital Image Processing*. New York: Wiley, 1978.
[34] E. Riskin, E. M. Daly, and R. M. Gray, "Pruned tree-structured vector quantization in image coding," in *Proc. IEEE ICASSP*, Glasgow, Scotland, May 1989, pp. 1735-1738.
[35] M. J. Smith and D. P. Barnwell, "Exact reconstruction for tree-structured subband coders," *IEEE Trans. Acoust., Speech, Signal Proc.*, vol. ASSP-34, pp. 434-441, 1986.
[36] J. O. Stromberg, "A modified haar system and higher order spline systems," in *Conf. in Harmonic Analysis in Honor of Antoni Zygmund*, Vol. II, pp. 475-493.
[37] M. Vetterli, "Splitting a signal into subsampled channels allowing perfect reconstruction," in *Proc. IASTED Conf. Appl. Signal Processing Digital Filtering*, Paris, France, June 1985.
[38] M. Vetterli and C. Herley, "Wavelets and filter banks: Relationships and new results," in *Proc. IEEE ICASSP*, Albuquerque, Apr. 1990.
[39] P. H. Westerink, D. E. Boekee, J. Biemond, and J. W. Woods, "Subband coding of image using vector quantization," *IEEE Trans. Commun.*, vo. 36, pp. 713-719, 1988.
[40] P. H. Westerink, J. Biemond, and D. E. Boekee, "Progressive transmission of images using subband coding," in *Proc. IEEE ICASSP*, 1989, pp. 1811-1814.
[41] P. H. Westerink, "Subband coding of images," Ph.D. dissertation Delft Univ., 1989.
[42] J. W. Woods and S. D. O'Neil, "Subband coding of images," *IEEE Trans. Acoust., Speech, Signal Proc.*, vol. ASSP-34, Oct. 1986.
[43] P. Zador, "Asymptotic quantization error of continuous signals and their quantization dimension," *IEEE Trans. Inform. Theory*, vol. IT-28, pp. 139-149, 1982.



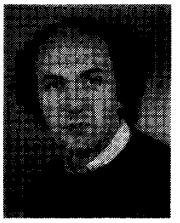
Marc Antonini was born in France on August 29, 1965. He received the DEA degree in signal processing in 1988 from the University of Nice-Sophia Antipolis, France, and the Ph.D. degree from the Laboratory of Signaux et Systèmes, URA I3S, CNRS and the University of Nice-Sophia Antipolis in 1991.

His research interests include multidimensional image processing, wavelet analysis, and image coding.

Michel Barlaud (M'88) was born in France on November 24, 1945. He received the "Doctorat d'Etat" degree from University of Paris XII.

He is currently a Professor and a member of the Laboratory of Signaux et Systèmes, URA I3S both from CNRS and University of Nice-Sophia Antipolis. After some work on non-stationary signal processing, his research interests move towards multidimensional image processing, wavelet analysis, image coding, inverse problems, image restoration, and edge detection.

Dr. Barlaud is member of the IEEE-ASSP MDSP committee.



Pierre Mathieu was born in Alger on May 10, 1956. He received the Ingenieur ENSEEIHT and Ph.D. degrees from INP Toulouse.

He is currently Maître de Conférences in the Laboratory of Signaux et Systèmes, URA I3S both from CNRS and University of Nice-Sophia Antipolis. His research interests include multidimensional image processing, wavelet analysis, image coding, and image restoration.

Ingrid Daubechies (M'89) received the B.S. and Ph.D. degrees from the Vrje Universiteit Brussel, Belgium in 1975 and 1980, both in physics.

She is currently a Member of Technical Staff in the Mathematics Center of AT&T Bell Laboratories, Murray Hill, NJ. Her current research interests include mathematical problems in connection with signal analysis, in particular applications of time-frequency representations.

US005321520A

United States Patent [19]

Inga et al.

[11] Patent Number: 5,321,520

[45] Date of Patent: Jun. 14, 1994

[54] **AUTOMATED HIGH DEFINITION/RESOLUTION IMAGE STORAGE, RETRIEVAL AND TRANSMISSION SYSTEM**

[75] Inventors: **Jorge J. Inga; Thomas V. Saliga,** both of Tampa, Fla.

[73] Assignee: **Automated Medical Access Corporation,** Tampa, Fla.

[21] Appl. No.: **915,298**

[22] Filed: **Jul. 20, 1992**

[51] **Int. Cl.**[5] **H04N 1/21;** H04N 1/41

[52] **U.S. Cl.** **358/403;** 358/426; 358/428

[58] **Field of Search** 358/403, 426, 261.1, 358/444, 428; 395/114

[56] **References Cited**

U.S. PATENT DOCUMENTS

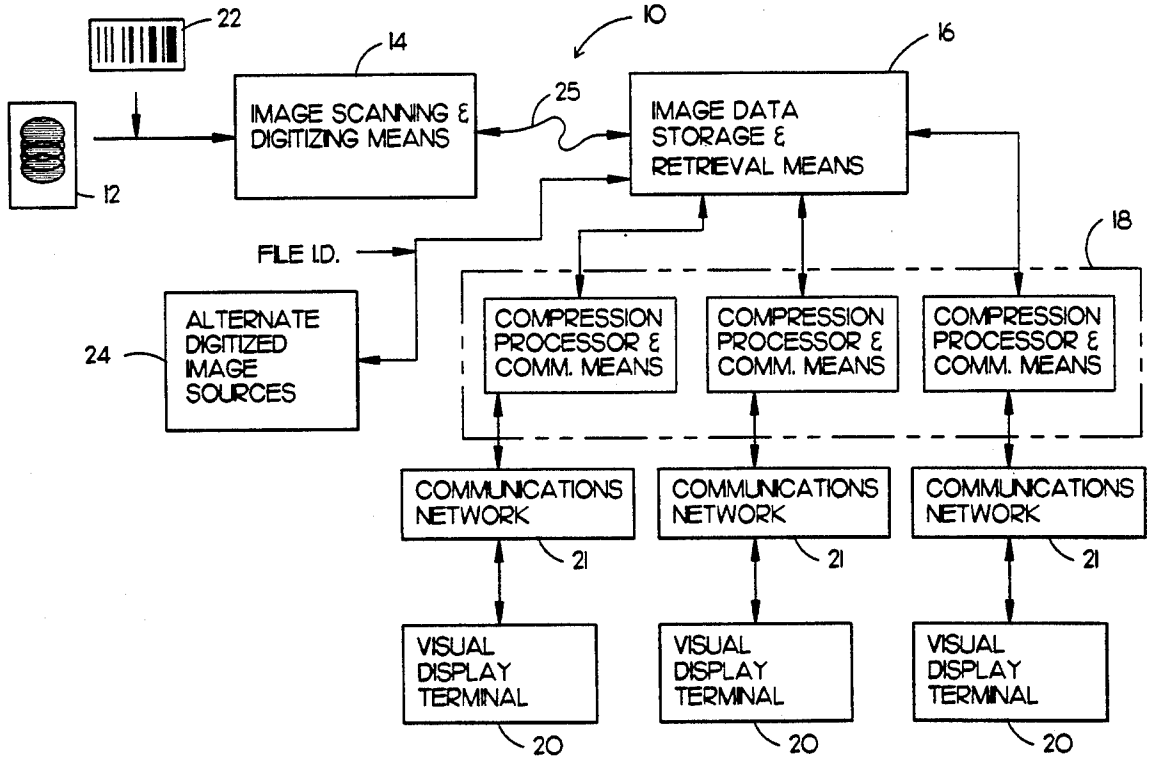
4,719,514	1/1988	Kurahayashi et al.	358/404
4,768,099	8/1988	Mukai	358/426
4,933,025	2/1991	Vesel et al.	370/94.1
4,958,283	9/1990	Tawara et al.	364/413.3
5,068,745	11/1991	Shimura	358/426
5,111,044	5/1992	Agano	250/327.2
5,111,306	5/1992	Kanno et al.	358/426
5,170,266	12/1992	Marsh et al.	358/426

Primary Examiner—Edward L. Coles, Sr.
Assistant Examiner—Scott A. Rogers
Attorney, Agent, or Firm—David Kiewit

[57] **ABSTRACT**

An automated high definition/resolution image storage, retrieval and transmission system for use with medical X-ray film or other documents to provide simultaneous automated access to a common data base by a plurality of remote subscribers upon request, the automated high definition/resolution image storage, retrieval and transmission system comprising an image scanning and digitizing subsystem to scan and digitize visual image information from an image film or the like; an image data storage and retrieval subsystem to receive and store the digitized information and to selectively provide the digitized information upon request from a remote site, a telecommunication subsystem to selectively transmit the requested digitized information from the image data storage and retrieval subsystem to the requesting remote visual display terminal for conversion to a visual image at the remote site to visually display the requested information from the image data storage and retrieval subsystem.

12 Claims, 9 Drawing Sheets



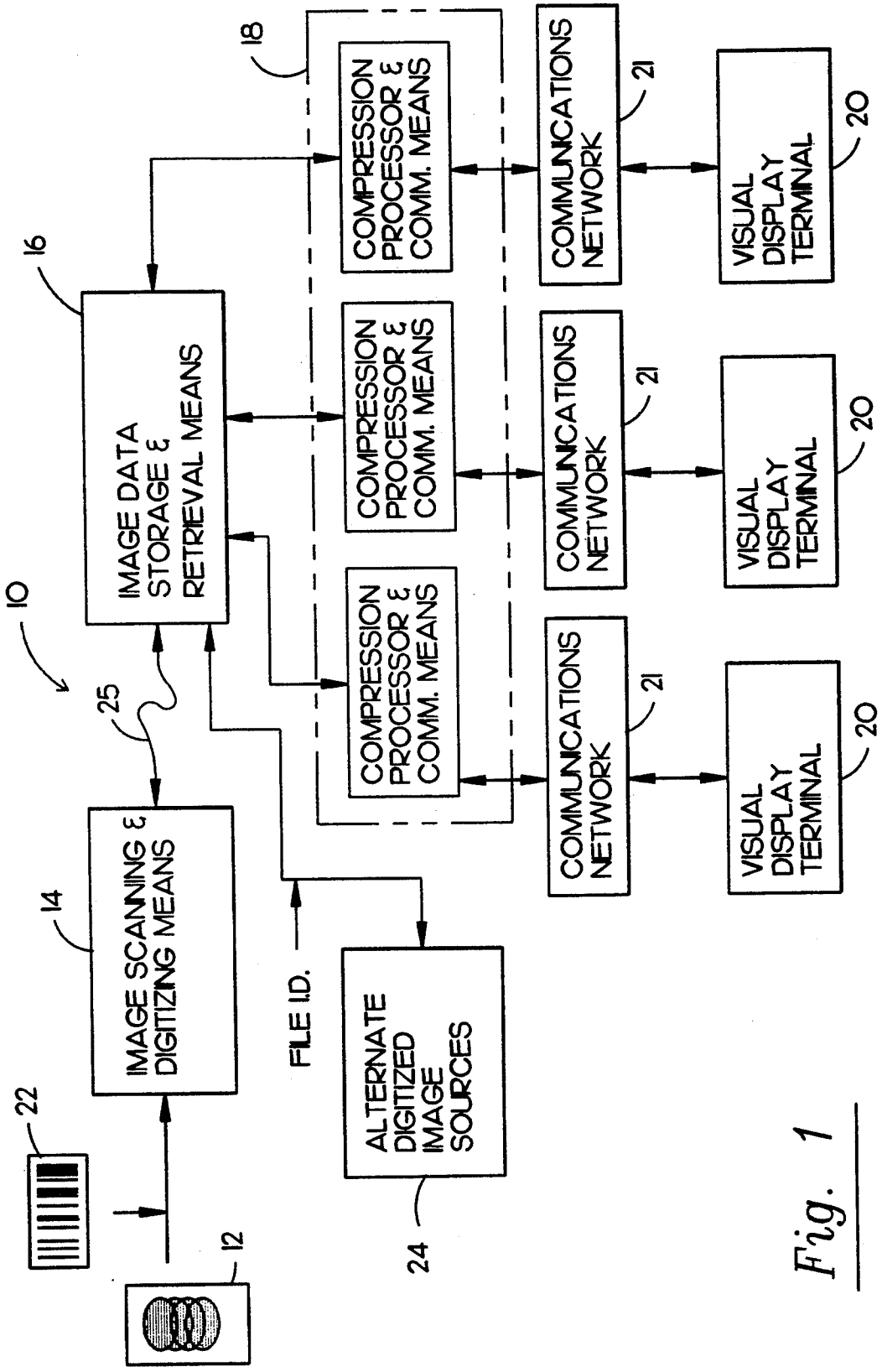

10
12
14
16
18
20
21
22
24
25
IMAGE SCANNING & DIGITIZING MEANS
IMAGE DATA STORAGE & RETRIEVAL MEANS
COMPRESSION PROCESSOR & COMM. MEANS
COMPRESSION PROCESSOR & COMM. MEANS
COMPRESSION PROCESSOR & COMM. MEANS
COMMUNICATIONS NETWORK
COMMUNICATIONS NETWORK
COMMUNICATIONS NETWORK
VISUAL DISPLAY TERMINAL
VISUAL DISPLAY TERMINAL
VISUAL DISPLAY TERMINAL
FILE I.D.
ALTERNATE DIGITIZED IMAGE SOURCES

Fig. 1

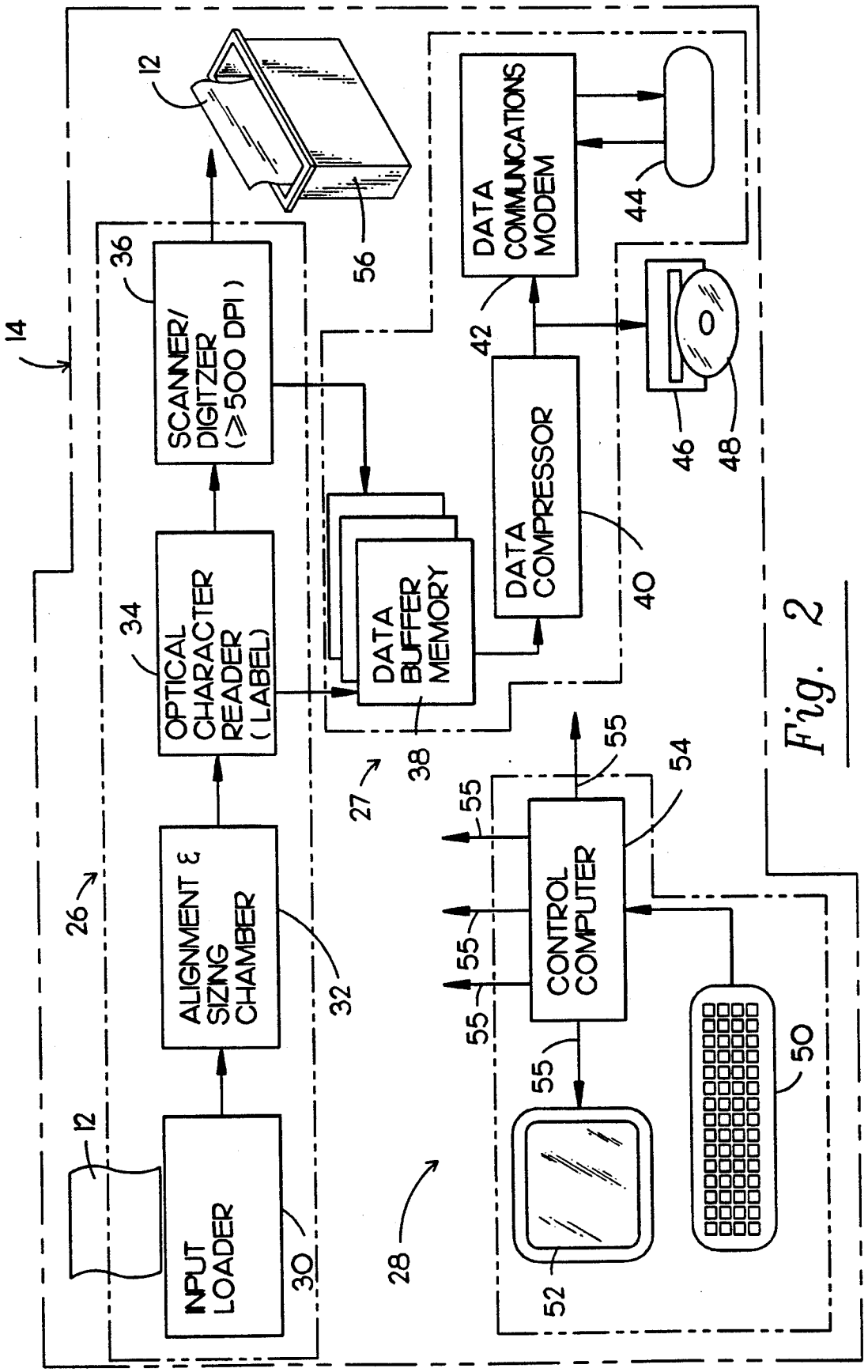

12
56
14
36
SCANNER/ DIGITZER (≥500 DPI)
34
OPTICAL CHARACTER READER (LABEL)
DATA BUFFER MEMORY
38
27
DATA COMPRESSOR
40
42
DATA COMMUNICATIONS MODEM
44
46
48
26
ALIGNMENT & SIZING CHAMBER
32
55
54
CONTROL COMPUTER
50
12
INPUT LOADER
30
28
52

Fig. 2

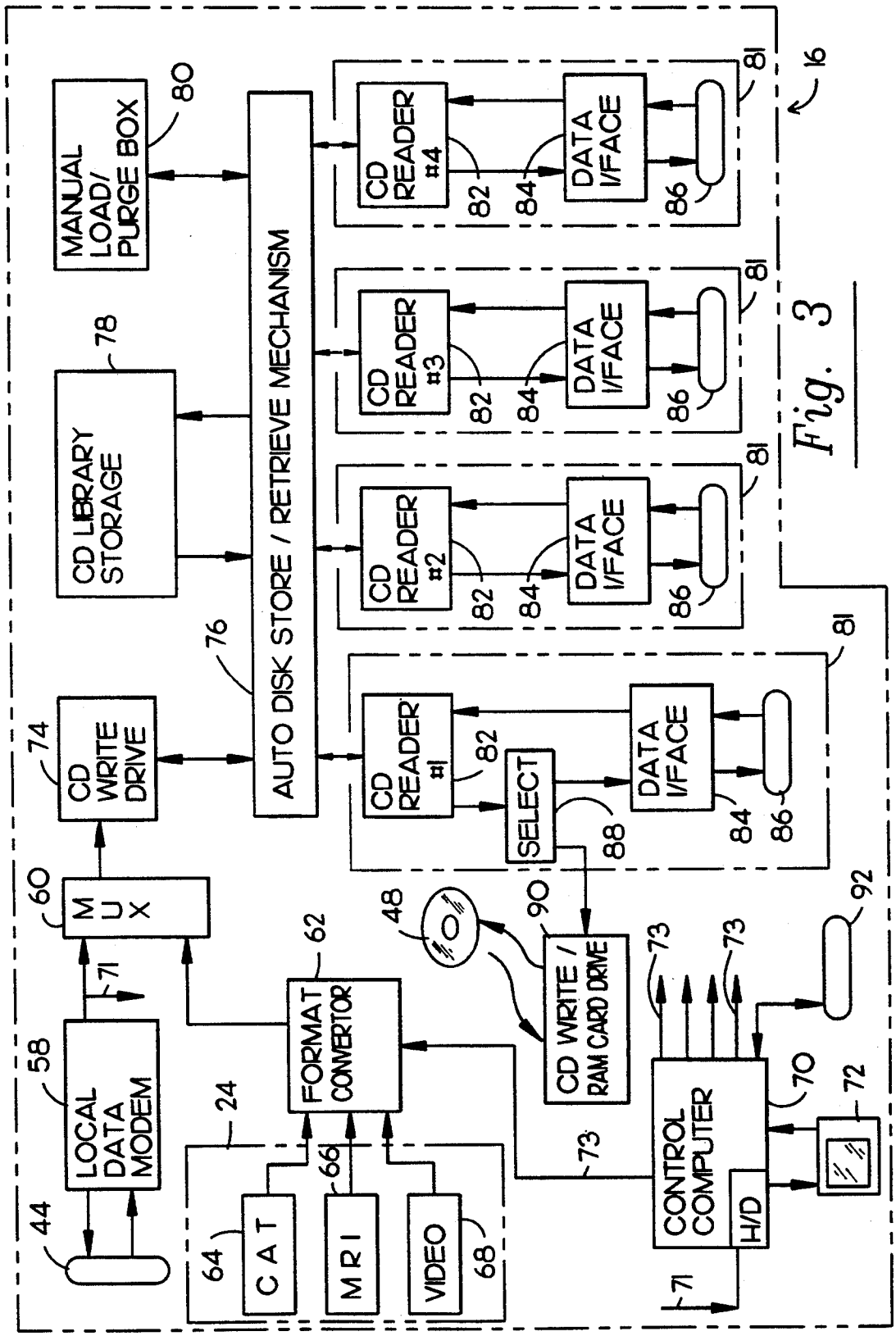

MANUAL LOAD/ PURGE BOX
80
CD LIBRARY STORAGE
78
AUTO DISK STORE / RETRIEVE MECHANISM
76
CD WRITE DRIVE
74
M U X
60
LOCAL DATA MODEM
58
44
71
FORMAT CONVERTOR
62
24
CAT
64
MRI
66
VIDEO
68
48
90
CD WRITE / RAM CARD DRIVE
73
CONTROL COMPUTER
H/D
70
72
92
CD READER #1
CD READER #2
CD READER #3
CD READER #4
SELECT
88
DATA I/FACE
82
84
86
81
16

Fig. 3

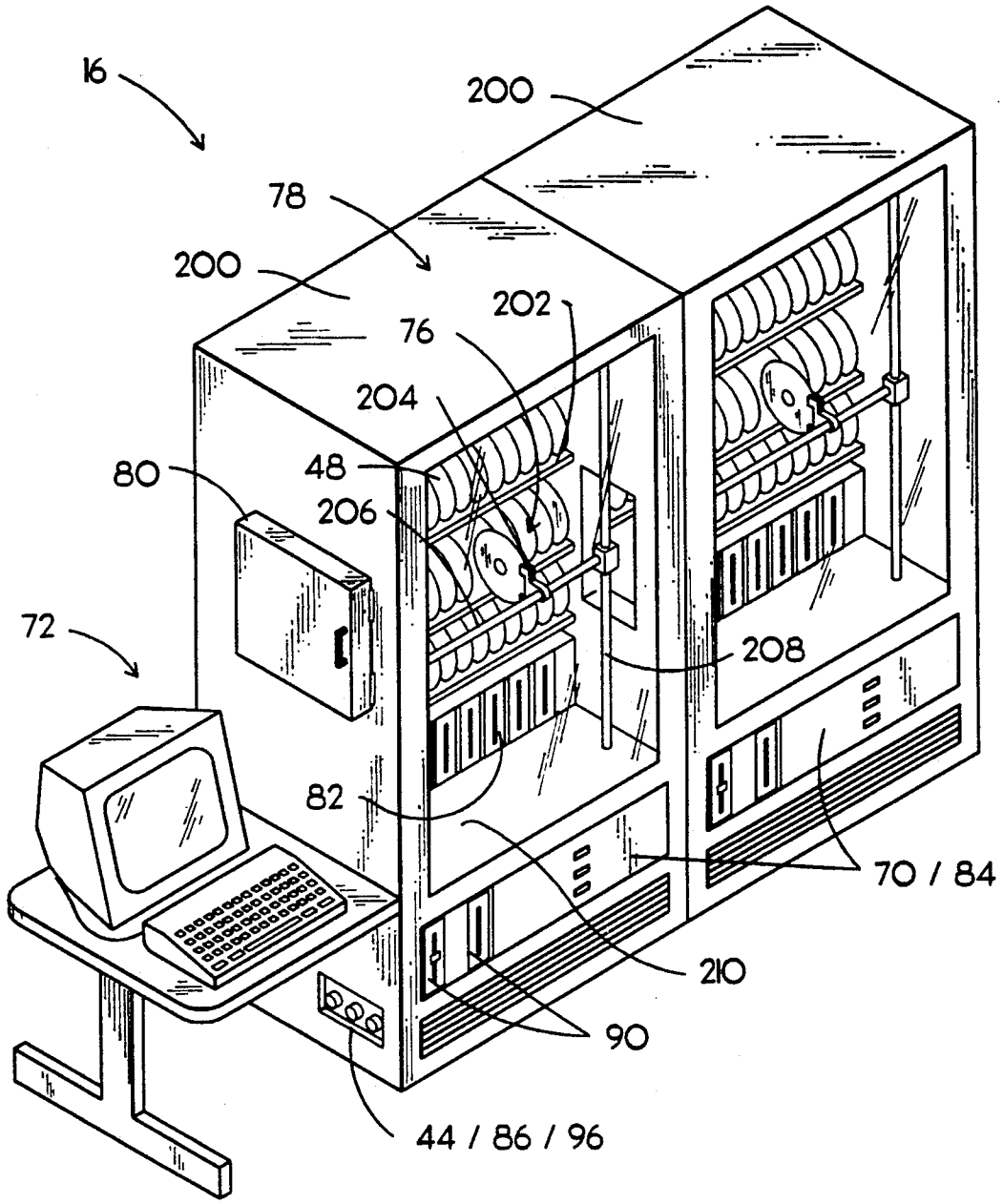


Fig. 4

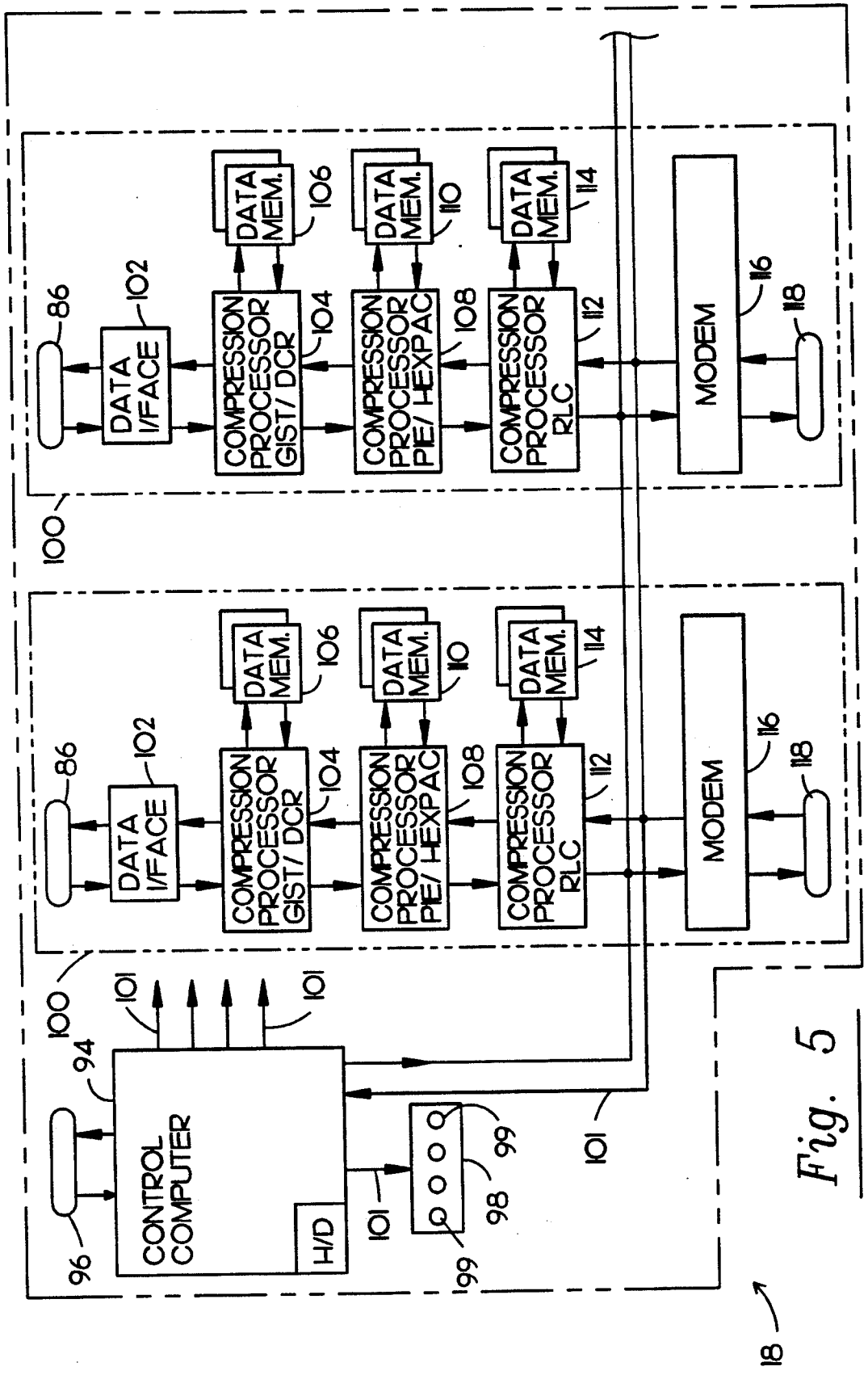

100
DATA I/FACE
102
86
COMPRESSION PROCESSOR GIST/ DCR
104
DATA MEM.
106
COMPRESSION PROCESSOR PE/ HEXPAC
108
DATA MEM.
110
COMPRESSION PROCESSOR RLC
112
DATA MEM.
114
MODEM
116
118
100
DATA I/FACE
102
86
COMPRESSION PROCESSOR GIST/ DCR
104
DATA MEM.
106
COMPRESSION PROCESSOR PE/ HEXPAC
108
DATA MEM.
110
COMPRESSION PROCESSOR RLC
112
DATA MEM.
114
MODEM
116
118
100
101
101
94
CONTROL COMPUTER
H/D
96
101
99
98
99
101
18

Fig. 5

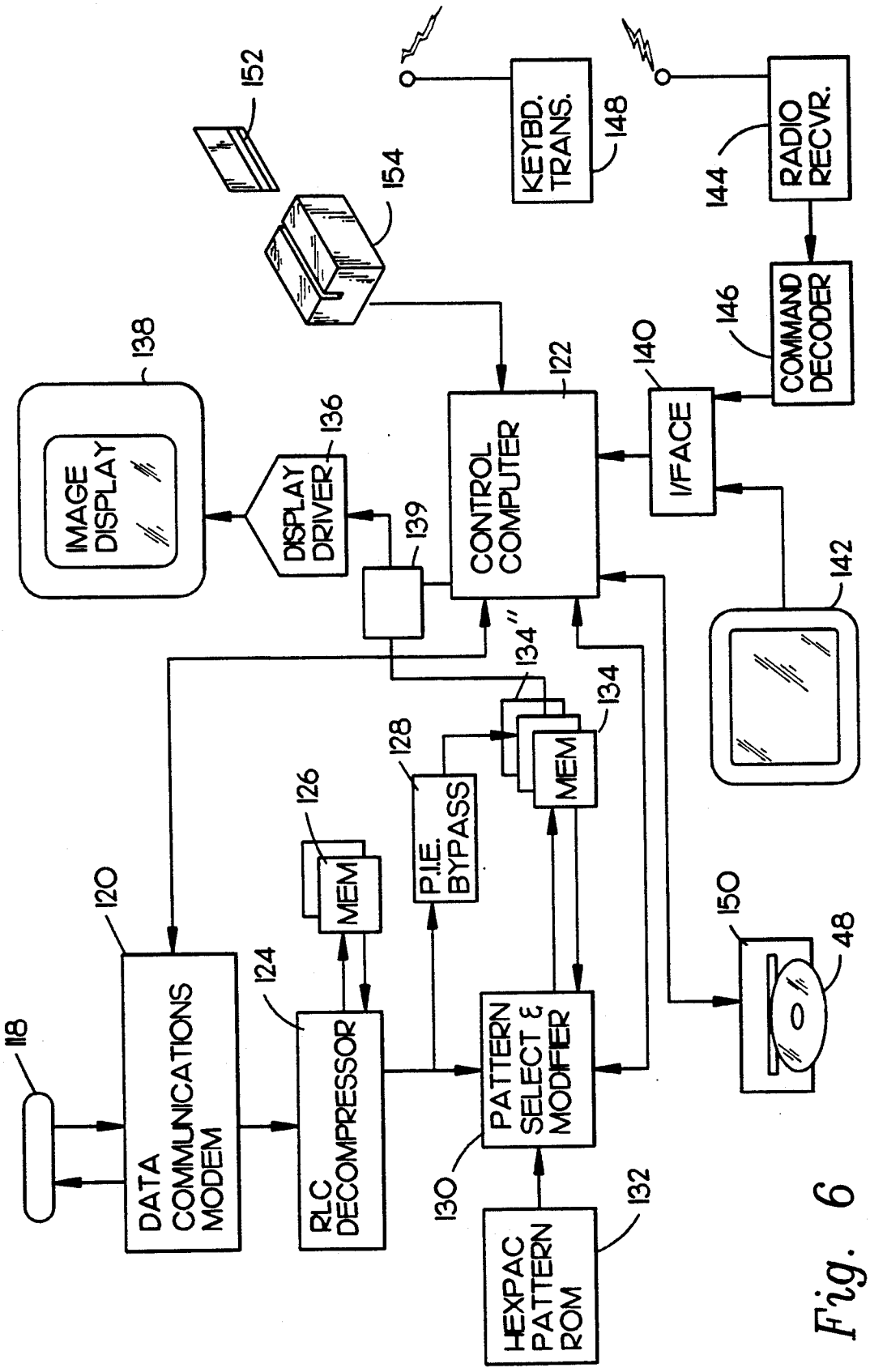

152
154
KEYBD. TRANS.
148
144
RADIO RECVR.
146
COMMAND DECODER
138
IMAGE DISPLAY
136
DISPLAY DRIVER
139
122
CONTROL COMPUTER
140
I/FACE
142
134"
134
MEM
128
P.I.E. BYPASS
126
MEM
120
124
150
48
118
DATA COMMUNICATIONS MODEM
RLC DECOMPRESSOR
PATTERN SELECT & MODIFIER
130
132
HEXPAC PATTERN ROM

Fig. 6

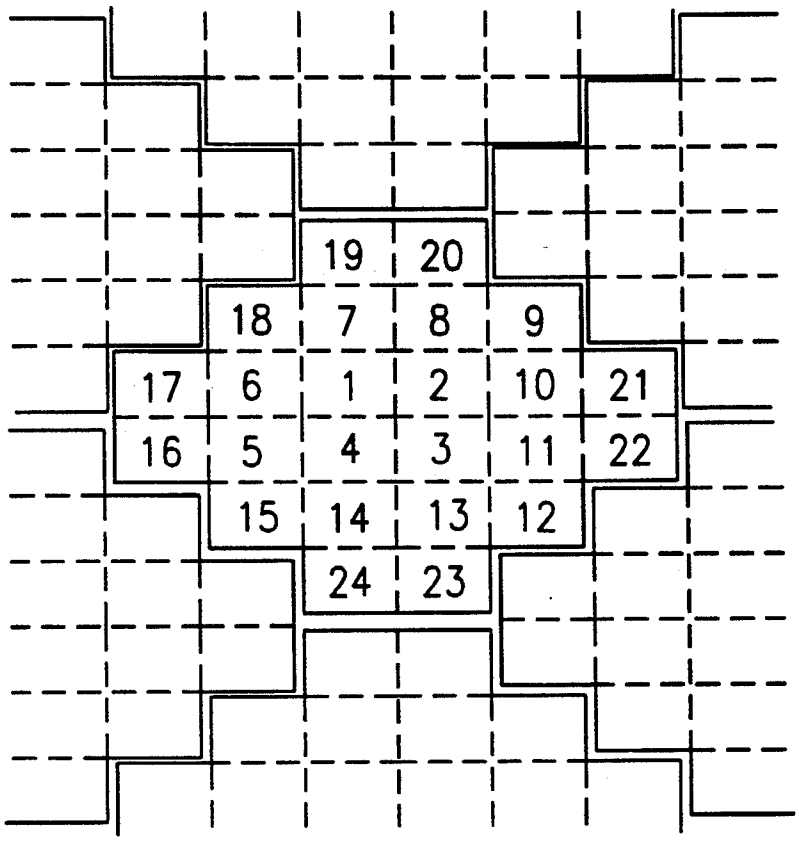


Fig. 7

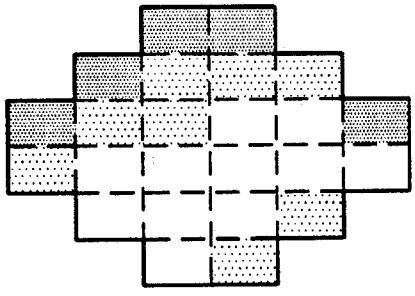


Fig. 8

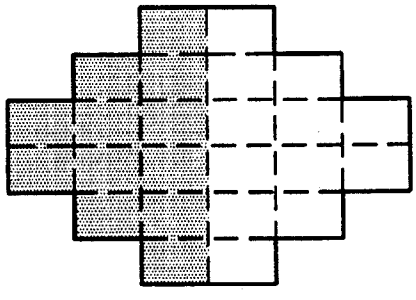


Fig. 9

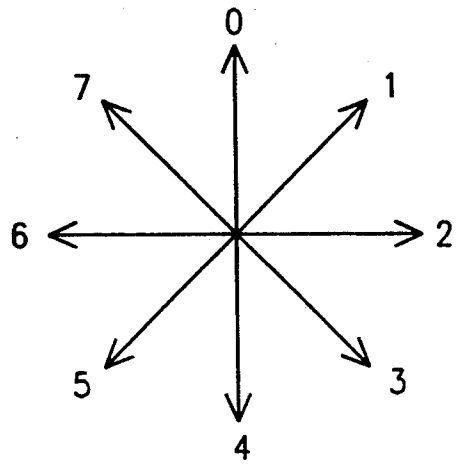


Fig. 10

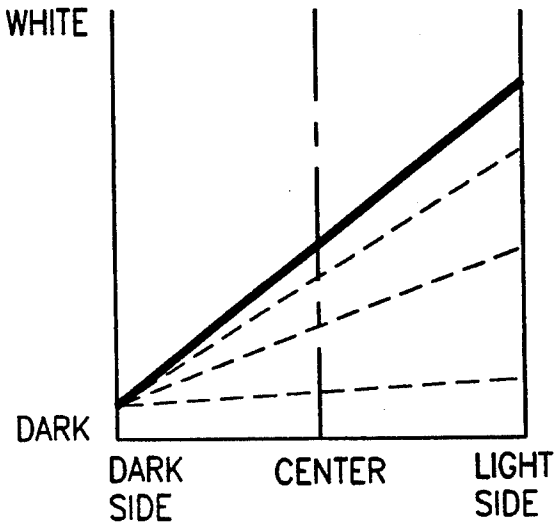


Fig. 11

Fig. 12

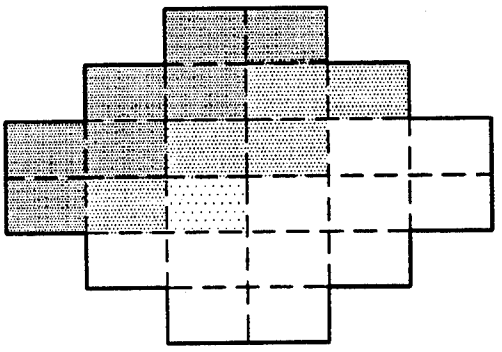


Fig. 13

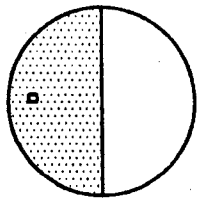


Fig. 14-A

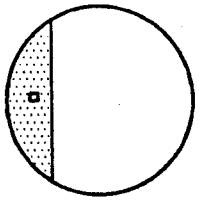


Fig. 14-B

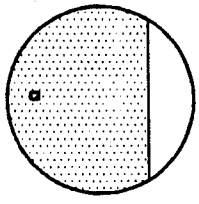


Fig. 14-C

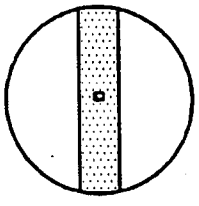


Fig. 14-D

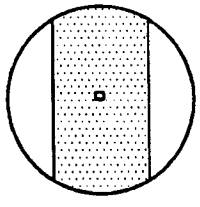


Fig. 14-E

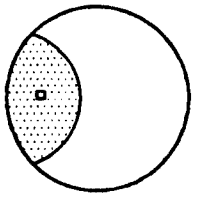


Fig. 14-F

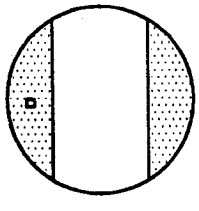


Fig. 14-G

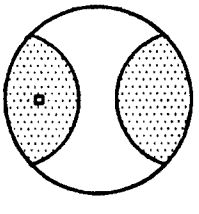


Fig. 14-H

AUTOMATED HIGH DEFINITION/RESOLUTION IMAGE STORAGE, RETRIEVAL AND TRANSMISSION SYSTEM

BACKGROUND OF THE INVENTION

1. Field of the Invention

An automated high definition/resolution image storage, retrieval and transmission system for use with medical X-ray films and the like.

2. Description of the Prior Art

Many industries have their own unique inventory control means for the particular needs of the respective trade. The food industry, for example, has largely standardized on bar-code technology to track food items. The medical profession, however, in many areas has not kept technological pace notwithstanding the obvious need therefor. Storage and retrieval systems for medical image data such as X-ray films, CAT scans, angiograms, tomograms and MRI are commonly antiquated and often employing methods of the 1920's. For example, when image films are used in the physician must display these photo films on a light box.

Moreover, due to the diffuse responsibilities of multiple attending physicians and treatment sites involving patients with particularly complex conditions, image data for such patients is often misplaced, lost, or at best, difficult to find when needed. Hospitals maintain large "file rooms" to store the patient image data. The film image data is typically stored in a large brown envelope approximately 14 by 17 inches and open at one end. These become bulky to handle and store especially in a complex situation in which several of these folders are needed. Weight alone can build up to 15 pounds. It has proven time consuming to obtain image data from file rooms either due to administrative backlogs, lack of specialized filing personnel and misfiling.

Typically, the physician examines the patient in his office after the radiographical studies have been made in a hospital or diagnostic facility. These films and the information contained therein are often unavailable at the time of the examination. Thus, there is a need for remote access to these image data for rapid patient assessment and therapy recommendation.

U.S. Pat. No. 4,603,254 teaches a stimulable phosphor sheet carrying a radiation image stored therein scanned with stimulating rays. The light emitted from the stimulable phosphor sheet in proportion to the radiation energy stored therein is detected and converted into an electric signal converted to a digital signal. Digital data is created to reproduce the radiation image for use in diagnosis and storage.

U.S. Pat. No. 4,764,870 describes a system for transferring medical diagnostic information from the diagnostic site to remote stations. An internal analog video signal from imaging diagnostic equipment such as a CAT scanner or MRI equipment, is converted to an analog video signal of different, preferably standard, format that is stored and transmitted in the reformatted image information to the remote terminal. The received signal is stored, decoded and applied in appropriate analog video form to an associated CRT display for reproduction of the diagnostic images.

U.S. Pat. No. 5,005,126 shows a system for transferring medical diagnostic information from the diagnostic site to remote stations similar to that found in U.S. Pat. No. 4,764,870.

U.S. Pat. No. 5,019,975 teaches a method for constructing a data base in a medical image filing system comprising the steps or recording information indicating the time at which each medical image is recorded and a rank of importance for each medical image as image retrieval signal data for image signals corresponding to each medical image; recording the number of times the image signals corresponding to each medical image have been retrieved as image retrieval signal data and incrementing the number each time the image signals are retrieved; and when the data base is full of image retrieval signal data, deleting the image retrieval signal data corresponding to the image signals of the medical image in which at least (1) the time at which the medical image was recorded earlier than a predetermined time and (2) the rank of importance of the medical image is lower than a predetermined value.

U.S. Pat. No. 4,611,247 describes a radiation image reproducing apparatus to read a radiation image from a first recording medium as a visible image. Input devices of the apparatus enter data which are associated with a method of exposing an object to a radiation and object's exposed part. In response to the input data, a processing condition determining unit determines conditions optimum for a gradation processing and a spatial frequency processing. A processor system is provided for reading the radiation image stored in the first recording medium and processing the radiation image on the basis of conditions which the processing condition determining unit determines in response to the input data associated with the radiation image.

U.S. Pat. No. 4,750,137 discloses a method and a computer program for performing the method for optimizing signals being exchanged between a host unit and an addressable-buffer peripheral device. The program optimizes an outgoing signal from the host unit by (1) creating an updated-state map representing the state of the peripheral device buffer expected to exist after processing by the peripheral device of the outgoing signal, (2) performing an exclusive-or (XOR) operation using the updated-state map and a present-state map representing the existing state of the buffer, and (3) constructing and transmitting a substitute outgoing signal which represents only changes to the buffer, and in which all premodified field flags are turned off. Position-dependent characters, such as attribute bytes, are translated into nondata characters prior to incorporation into a map, and are retranslated into their original form for use in the substitute signal.

U.S. Pat. No. 4,858,129 teaches an X-ray CT apparatus in which a plurality of dynamic tomographic images obtained by repeatedly photographing a region of interest of a subject under examination are stored in an image memory for subsequent display on a display device. A processing device extracts data of pixels along a certain the tomographic images and stores the pixel data in the image memory, in the order of photographing time of the tomographic images, thus forming a time sequence image formed of picked-up pixels. The processing device reduces a tomographic image and the time sequence image and rearranges the reduced images in one frame area of the image memory for simultaneous display thereof on the display device.

U.S. Pat. No. 5,021,770 discloses an image display system having a plurality of CRT display screens. The system is of the type in which a number of images of specific portions of a patient having a specific identification code are selected from among a multitude of X-ray

image taken by a plurality of shooting methods, and when the regions or interest are specific, a plurality of appropriate images are further selected using the previously stored aptitude values for the regions and shooting methods and displayed on the plurality of CRT display screens. In order that the segments to be inspected can be pointed to on the screen on which the image of the patient is displayed, a memory is provided which is adapted to previously store codes corresponding to the specific image of the patient and to specify the respective regions of the image in such a manner that they correspond to the pixel positions of the image.

U.S. Pat. No. 4,879,665 teaches a medical picture filing system composed of a picture data memory device, a picture data input-output device for inputting-/outputting the picture data, a retrieving device for storing the picture data into the memory device and extracting it therefrom on the basis of retrieving data, a retrieving data input device for inputting the retrieving data into the retrieving device, a retrieving data storing device for storing the retrieving data, the retrieving data being classified by block of information obtained in one-time examination. When medical pictures are filed, retrieving data collected for each examination is utilized for reducing the amount of retrieving data, while when reproduced, retrieval is carried out for each one-time examination thereby shortening the time required for retrieval.

In light of recent advances in computer data basing, digitization and compression of image data, image enhancement algorithms and cost effective computer technology, the means for improved storage and retrieval of vital patient image data is now possible.

Such system should include the following major features:

1) means to more compactly store and more efficiently retrieve image data and automatically identify the data by patient name, image type, date and the like;

2) means for physicians to quickly and remotely access particular patient image data at the medical facility even if achieved at several different locations;

3) means to prevent loss of vital image data due to ordinary human handling and misplacement errors;

4) means to quickly and affordably access image data from the physician's office;

5) means to enhance the medical images by both contrast enhancement and zooming for improved diagnostics and/or surgical guidance; and

6) means to quickly and conveniently access image data and display on a large screen in the operating room with any desired enhancement or expansion.

As described more fully hereinafter, the present invention provides means to accomplish these goals. The system uses both general purpose system elements well known to those practiced in electronic arts and specific elements having significant novelty.

SUMMARY OF THE INVENTION

The present invention relates to an automated high definition/resolution image storage, retrieval and transmission system capable for storing, transmitting and displaying medical diagnostic quality images for use with medical X-ray films or the like.

The system comprises means to process the image data from patient imaging to physician usage. The major or significant processing stages are described hereinafter. Specifically, the major steps in the image data flow include:

PATIENT RADIOGRAPHY: The patient's body is imaged and a film is exposed as in an X ray room, MRI or CAT scan lab.

FILM PREPARATION: The film(s) is developed to create a visible image with OCR readable patient identification information superimposed thereon.

FILM INTERPRETATION: Commonly, a radiologist drafts an opinion letter for the film(s). This document preferably includes an optical character reader, or OCR, readable patient identification label or standard marking area.

IMAGE SCANNING AND DIGITIZING SUBSYSTEM: A scanner subsystem digitizes each patient image film and/or document on a high resolution scanner. This digitized data is transmitted by a local high speed data link to a separate or remote master storage unit. Patient identification information is read from a standard format on each file by OCR techniques and efficiently stored with the digitized image data. Enhanced scanner resolution and gray scale requirements are provided. Further, to reduce data rate requirements, data compaction or compression is accomplished within the scanner subsystem.

To back-up possible data link down time or scanner down time, the scanner subsystem may include a CD-ROM data storage drive so that image data may continue to be digitized. The CD-ROM disk may then be manually delivered to the file room unit for subsequent use.

In an optional embodiment, the digitized data of one or two images may be written to a compact semi-conductor memory card "RAM Cards". This form of data storage may be used to send selected images for special purposes such as when the image data is needed in another city for second opinion purposes.

At this point in the image data flow, there is a split in which the original film data is stored as a "master" in a file room and the image disk is made available for active "on-line" use in an image storage and retrieval subsystem.

FILM FILING: The patient image films may be placed in the industry standard 14 by 17 inch brown paper folders and placed on conventional filing shelves. However, it is preferred that older films be tagged and stored off-site to reduce the current excessive bulk of films in many hospital file rooms. The system would now make this practical since the original films would seldom need to be accessed.

In the preferred embodiment of the system, the patient may have his entire image data collected and written to one or more of the storage CD-ROM disks for archiving at the hospital.

IMAGE STORAGE AND RETRIEVAL SUBSYSTEM: This subsystem is a remotely controllable, automatically accessable image data subsystem to store and automatically retrieve, on-demand, the compressed digital information contained on the CD-ROM disks.

The image storage and retrieval subsystem has a high-speed data link connection to the scanning and digitizing subsystem and has a write drive recording mechanism which is dedicated to receiving the data from the scanning and digitizing subsystem. This CD-ROM write drive can operate without interrupting remote access operations.

Remote access may be made to the image storage and retrieval subsystem by a variety of telecommunication links. Access will be granted only if a valid user code has been presented. By means of several read-only CD

disk drives and electronic buffering, virtually simultaneous access can be granted to several or more users.

As explained more fully hereinafter, the medical image disk will contain relatively huge quantities of data making it impractical to send over conventional data communication links without very efficient data compression technology. While there are a variety of data compression techniques available, none are well tailored to this application. Thus, novel compression means are in the a remote telecommunication access subsystem.

TELECOMMUNICATION SUBSYSTEM: Occasionally circumstances may warrant manually making an extra copy of the patient's image files to be physically delivered to an authorized requester. However, for the system to fulfill broad service to the health care industry it must be able to efficiently telecommunicate image files to remote locations both cost effectively and within a reasonable time interval.

A novel medical facsimile technology is in the preferred embodiment which works interactively with a remote requester to send only what is needed at acceptable resolutions, and the presented image is progressively updated as the communications connection is maintained until the resolution limits of the user display are reached, after which time, other images are either sent or further enhanced.

The specific technical means for accomplishing this uses the following novel technologies: a) guided image selection and transmission (GIST); b) progressive image enhancement (PIE); c) display compatible resolution (DCR); d) hexagonal pattern classification compression (HexPac) and e) run length coding (RLC), RLC is well known to those skilled in the arts.

It appears practical to send immediately useful patient data in less than one minute over a phone line (9600 baud) whereas it take many hours by conventional coding and transmission means. When wide-band telecommunications as satellite, fiber optic, micro-wave links becomes more generally available at affordable prices, then the more complex data compression techniques described here will be less important, but until that time, these types of techniques are believed essential to overall system success.

This combination of technologies to efficiently compress the image data and transmit remotely comprises the telecommunication access subsystem. In practice, these technologies may be implemented for the most part with available computer modules although several special signal processor boards are needed.

REMOTE DISPLAY TERMINAL: The quality of the image available to the user is limited or determined by the receiving presentation terminal or monitor. Two specific presentation terminal types are envisioned in the preferred embodiment of the system, a modified personal computer terminal for use in a physician's office, hospital nurses' station and the like, and a large screen presentation terminal with remote controlled interaction primarily for operating room use.

Both terminals have means to show the available patient directory of images, and means to select an image, enhancement and zoom on selected areas. Image enhancement has heretofore been impractical for film based images and thus much subtle but important pathological information has been largely lost. This is especially true of X-ray data The ability to enhance subtle contrasted tissues areas is considered to be an important feature and benefit of the system.

An optional high-resolution printer (300 dpi or better) permits the physician to print out selected images. This will be especially valuable when the physician expands and enhances selected critical image areas since a cost effective printer would otherwise not have adequate gray scale or pixel resolution to give diagnostically useful output.

Each terminal consists of a standard high performance personal computer with one or more data source interfaces such as RAM card, CD-ROM disk or data modem, a decompression graphics interface circuit and graphics display. The large screen presentation terminal has a large screen display for easy viewing for a surgeon who may be ten or more feet distant. The large screen presentation terminal also has an optional remote control so that an attending technician or nurse can scroll images, enhance and zoom, at the surgeon's request.

A keynote of each terminal design is a very simple user interface based upon a limited selection menu and obviously pointed-to graphical icons.

The invention accordingly comprises the features of construction, combination of elements, and arrangement of parts which will be exemplified in the construction hereinafter set forth, and the scope of the invention will be indicated in the claims.

BRIEF DESCRIPTION OF THE DRAWINGS

For a fuller understanding of the nature and object of the invention, reference should be had to the following detailed description taken in connection with the accompanying drawings in which:

FIG. **1** is a functional block diagram of the entire system of the present invention.

FIG. **2** is a functional block diagram of the image scanning and digitizing means.

FIG. **3** is a functional block diagram of the image data storage and retrieval means.

FIG. **4** is a perspective view of the image data storage and retrieval means.

FIG. **5** is a functional block diagram of the telecommunication means.

FIG. **6** is a functional block diagram of the remote display terminal means.

FIG. **7** depicts the hexagonal pattern of the hexagonal compression method.

FIG. **8** depicts an actual hexagonal pattern from an X-ray film. FIG. **9** depicts the selected predetermined hexagonal pattern most closely corresponding to the actual hexagonal pattern shown in FIG. **8**.

FIG. **10** graphically represents the predetermined rotational orientations for the predetermined hexagonal patterns.

FIG. **11** graphically depicts a selected gray level slope of the selected predetermined hexagonal pattern of FIG. **9**.

FIG. **12** depicts a single pixel from the predetermined hexagonal pattern.

FIG. **13** depicts a hexagonal pattern reconstructed by a remote display terminal means corresponding to the actual hexagonal pattern shown in FIG. **8**.

FIGS. **14**-A through **14**-H depict the predetermined set of orthogonal gray level patterns.

Similar reference characters refer to similar parts throughout the several views of the drawings.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

As shown in FIG. **1**, the present invention relates to an automated high definition/resolution image storage, retrieval and transmission system generally indicated as **10** for use with medical X-ray film **12** or other documents to provide simultaneous automated access to a common data base by a plurality of remote subscribers upon request from the remote subscribers.

The automated high definition/resolution image storage, retrieval and transmission system **10** comprises an image scanning and digitizing means **14** to transform the visual image from the medical X-ray film **12** or other documents into digital data, an image data storage and retrieval means **16** to store and selectively transfer digital data upon request, a telecommunication means **18** to selectively receive digital data from the image data storage and retrieval means **16** for transmission to one of a plurality of remote visual display terminals each indicated as **20** upon request from the respective remote visual display terminal **20** through a corresponding communications network **21** such as a telephone line, satellite link, cable network or local area network such as Ethernet or an ISDN service for conversion to a visual image for display at the remote requesting site.

To improve automation and tracking, a machine readable indicia or label **22** containing key patient information may be used in association with the medical X-ray film **12**. As shown, the machine readable indicia or label **22** is affixed to the medical X-ray film **12** prior to scanning by the image scanning and digitizing means **14** to provide file access and identification. Furthermore, digital data from alternate digitized image sources collectively indicated as **24** and file identification may be fed to the image data storage and retrieval means **16** for storage and retrieval.

FIG. **2** is a functional block diagram of the image scanning and digitizing means **14** capable of converting the visual image from the medical X-ray film **12** to digitized image data for transmission to the image data storage and retrieval means **16** over a bi-directional high speed data link **25**. Specifically, the image scanning and digitizing means **14** comprises a film loading and scanning section and a data compression and transmission section generally indicated as **26** and **27** respectively and a display and control section generally indicated as **28**. The film loading and scanning section **26** comprises a film input loader **30**, alignment and sizing chamber **32**, optical character reader **34** and film scanner/digitizer **36** capable of at least 500 dots per inch resolution **36**; while, the data compression and transmission section **27** comprises a data buffer memory **38**, low-loss data compression means **40**, local data modem **42** and transmission connector **44** to operatively couple the image scanning and digitizing means **14** to the image data storage and retrieval means **16**. The low-loss data compressor **40** is also operatively coupled to a compact disk data storage drive **46** capable of writing or storing compressed digitized patient image data on a compact disk **48**. The display and control section **28** comprises a keyboard/control console **50**, display terminal **52** and control computer **54** which is operatively coupled to the other components of the image scanning and digitizing means **14** through a plurality of conductors each indicated as **55**. A film collector tray **56** may be disposed adjacent the film scanner/digitizer **36** to receive the medical X-ray film **12** therefrom following processing.

To reduce the approximately 238 Megapixels required to digitize a 14 inch by 17 inch, medical X-ray film **12** with 700 dots or pixels per inch with a two byte level to a manageable size without significant information loss, a linear gray level prediction, modified run-length code generating logic circuitry is embodied within the low-loss data compression means **40** to dynamically compress the digitized data before storage or recording. The image data is compressed with acceptable diagnostic resolution loss. The low-loss data compression means **40** measures the "local" slope of the pixel gray level and continues to compare that estimated gray level for up to an entire scan line until a pixel region is reached which differs from the linear estimate by more than a predetermined amount. The data actually sent for that region consists of the slope of the line, actual level at the origin of the slope line and the number of pixels comprising that region. The circuitry will discard linear gray level slope differences of the original film which can be reliably determined to be noise or image "artifacts". A sudden pixel (if at 1000 dots per inch) dramatic change in gray level could be rejected as dust or film noise for example. The compressed data is a trade-off between complexity, speed and minimum data loss to reduce the total data quantity stored by a factor of approximately three. Thus, about 80 Megapixels of data may still have to be stored per 14 inch by 17 inch film image.

In the preferred embodiment, the bi-directional high speed communications link **25** transmits the low-loss compressed digitized data from the developing lab room to the hospital file room where the image data storage and retrieval means **16** will transfer and store the patient and image data in a new patient file on a compact disk **48**.

Two way communications between the image scanning and digitizing means **14** and the image data storage and retrieval means **16** minimizes data loss by insuring that a compact disk **48** be available to receive and store data. Moreover, the compact disk data storage drive **46** with re-writable ROM technology can record data even if communications with the image data storage and retrieval means **16** is disrupted. Thus the image scanning and digitizing means **14** can automatically start writing data to the compact disk data storage drive **46** as soon as a image data storage and retrieval means **16** fault is sense. The display and control section **28** informs the operator of the system status.

In operation, the film lab technician may stack one or more medical X-ray films **12** onto the input loader **30** as shown in FIG. **2**. A "read" button is depressed on the keyboard/control console **50** and each film **12** is thereafter fed in automatically, digitized and transmitted to the image data storage and retrieval means **16** located in the file room. As the reading of each film **12** is completed, the film **12** is deposited into the film collector tray **56**. System status, number-of-films read logging and so forth are shown on the display terminal **52**.

Initially, the image scanning and digitizing means **14** positions the film **12** in the alignment and sizing chamber **32** on a precision carrying platen for subsequent optical scanning. This platen contains optical sensors to sense the exact film size so only the useful image area is digitized. Once the film **12** is secured onto the movable platen, the film **12** is passed through the optical character reader **34** and then to the film scanner/digitizer **36**.

The patient data and image identification is first recorded onto the remote CD-ROM file directory in the image data storage and retrieval means **16** from the

OCR "pass" and then the compressed scanned image data is sequentially written to a compact disk **48** by a CD write drive for storage with the CD library storage of the image data storage and retrieval means **16** as described more fully hereinafter as the film **12** slowly passes through the film scanner/digitizer **36**.

Specifically, the film scanner/digitizer **36** converts the image to a digital representation of preferably at least a 700 dot per inch resolution. This digital data is temporarily stored in the data buffer memory **38** where the patient data from the optical character reader **34** and corresponding digitized image data from the file scanner/digitizer **36** are properly formatted for subsequent compression and transmission to the image data storage and retrieval means **16**. The stored data is then accessed by and compressed by the data compression means **40** as previously described and transmitted through the local data modem **42** and transmission connector **44** to the image data storage and retrieval means **16** or a compact disk data storage drive **46**. The display and control section **28** permits the X-ray lab staff to monitor system status, report quantity of documents and films processed and allow for scheduling local recording of image data on compact disks **48**.

FIGS. **3** and **4** show the image data storage and retrieval means **16** to receive and store the low-loss compressed digitized patient information and image data from the image scanning and digitizing means **14** and to selectively transmit the stored low-loss compressed digitized patient information and image data to one or more of the remote visual display terminal(s) **20** through corresponding telecommunication means **18** and corresponding communications network(s) **21** upon request from one or more of the remote display terminal(s) **20**.

The image data and retrieval means **16** is essentially a central data storage library for medical subscribers to remotely access and visually display patient data and information.

As described hereinafter, the image data storage and retrieval means **16** is robotically automated to minimize hospital staff requirements. At any given time, it is estimated that a typical hospital may have several hundred active patients with requirements for physician access to corresponding image files. An active patient may require one to three compact disks **48**. Thus, the image data storage and retrieval means **16** should have sufficient means to store and retrieve at least 500 compact disks **48**.

Further, to minimize personnel requirements, the image data storage and retrieval means **16** has a semi-automatic log-in mechanism for updating the compact disk inventory and an automatic mechanism for retrieving and reading the compact disks **48** remotely via communication link interfaces similar to juke box playback mechanisms. Except for the occasional loading of new empty compact disks **48** and removal of inactive compact disks **48**, the operation of the image data storage and retrieval means **16** is fully automatic permitting authorized access at any time.

As described more fully hereinafter, several playback drives with electronic buffering are incorporated so that essentially simultaneous access can be provided to several remote requesting users. An optional duplicating CD write drive and RAM-Card drive permits additional copies to be made locally upon demand for either back-up or other use. The image data storage and retrieval means **16** has an operator's console/desk arrangement for file maintenance and duplicating control by the hospital file room clerk. Control software is a simple menu selection design so that relatively unskilled personnel can maintain the central data storage library or image data bank.

As shown in the functional block diagram of FIG. **3**, the image data storage and retrieval means **16** comprises a local data modem **58** operatively coupled between the image scanning and digitizing means **14** through the transmission connectors **44** and bi-directional high speed communication link **25**, and a selector or multiplexer **60**. A format convertor **62** is operatively coupled between the alternate digitized image source(s) **24** such as CAT **64**, MRI **66** and/or video **68** and control computer **70** which is, in turn, coupled to a control console **72** including a visual display and input means such as a keyboard. The local data mode **58** is also coupled to the hard disk (H/D) of the control computer **70** through a conductor **71**. The other components of the image data storage and retrieval means **16** are coupled to the control computer **70** through a plurality of conductors each indicated as **73**. A CD write drive **74** is operatively coupled between the multiplexer or selector **60** and an auto disk storage/retrieve mechanism **76** which is, in turn, operatively coupled to a CD library storage **78**, a manual load/purge box **80** and a plurality of data retrieval and transmission channels each indicated as **81**. Each data retrieval and transmission channel **81** comprises a CD reader drive **82** operatively coupled through a corresponding data interface **84** to a corresponding transmission connector **86**. In addition, one of the CD reader drives **82** is operatively coupled through a selector switch **88** to an optional CD write/RAM card drive **90** configured to manually receive a compact disk **48** or RAM card.

As shown in FIG. **4**, the CD library storage **78** comprises at least one cabinet **200** to operatively house **800** compact disks **48** arranged or four shelves each indicated as **202** and the auto disk storage/retrieval mechanism **76** which comprises a CD coupler **204** to engage and grasp a selected compact disk **48** and move horizontally on a support member **206** that moves vertically on a pair of end support members each indicated as **208**. An access door **210** permits movement of compact disks **48** to and from the cabinet **200**. However, in normal operation, "old" patient data is removed by writing collected image data to a single compact disk **48** through the CD write/RAM card drive **90** thus freeing internally disposed compact disks **48** for new data. The CD write/-RAM card drive **90** may also be used to collect a patient's image data on a single compact disk **48** for use in the operating room's display terminal. This obviates the need for a high speed internal hospital local area network.

The computer associated with the CD robotic arm and drive mechanism performs ordinary library maintenance functions such as retrieval of outdated files, access statistics, entry of access validation codes, and so forth. This computer subsystem also handles data communication interface functions.

Internal to the environmentally controlled cabinet **200** are a plurality of playback mechanisms (field expandable to six) which are automatically controlled by the accessing physicians via the coupled communications system. Yet another CD-ROM write drive can record new data from the image scanning and digitizing means **14** or perform library functions such as consoli-

dation of a patient's data from several compact disks **48** to a single patient-dedicated compact disk **48**.

The internal computer maintains a file log of which compact disks **48** are empty and where each patient's image data is stored by disk number and track on a disk location. When the image scanning and digitizing means **14** requests to down-load data, the auto disk storage/retrieval mechanism **76**, of the image data storage and retrieval means **16** retrieves the "current" compact disk **48** which is being written with data (if not already loaded), then loads the compact disk **48** into the CD write drive **74**, and signals to the image scanning and digitizing means **14** to transmit. Image data is then recorded with a typical record time of 4 minutes for a full-size, high density image.

Once the robotic arm has delivered the compact disk **48** to the CD write drive **74**, the robotic arm is free to access and place other compact disks **48** onto CD reader drive **82** as commanded by its communications interface. The robotic arm can find and place a disk **48** into the appropriate CD reader **82** in approximately 10 seconds. Thus, there is minimal waiting time for disk access unless all CD readers drives **82** are in use.

As shown in FIG. **3**, data is received through the input transmission connector **44** to the CD write drive **74** through the selector switch **60**. Alternately, other image data from other sources such as CAT scanners **64** or MRI medical equipment **66** may be fed through the format convertor **62** for storage on a compact disk **48**. If the other image sources are written to CD write drive **74**, file identification data must be supplied to the format convertor **62** from the control computer **70**.

The image file data received from the image scanning and digitizing means **14** is directly written to free space on a compact disk **48** in the CD write drive **74**. No other data compression or special formatting is required as the image scanning and digitizing means **14** has performed these functions. As new image data is received from the image scanning and digitizing means **14** or another image source **24**, the image data is sequentially appended to the last file on the compact disk **48** currently being written to. Thus, no attempt is made to organize a single patient's image files onto a single compact disk **48**. However, each file received is logged into the control computer **70** through the conductor **71**. Therefore, the control computer **70** always knows what disk location in the CD library storage **78** contains any specified file. Once a compact disk **48** is filled with image data, the auto disk storage/retrieve mechanism **76** removes the compact disk **48** from the CD write drive **74** and stores the compact disk **48** in an empty location in the CD library storage **78**.

The plurality of data retrieval and transmission storage **78**. channels **81** service the data requests from subscribers. As previously indicated, a single data retrieval and transmission channel **81** includes the select switch **88** to direct image file data to the optional CD write/-RAM card drive **90**. By this means, all image data for an individual patient may be collected on one or more selected compact disks **48** for archiving or other use. However, normally, the control computer **70** will automatically remove old image data by removing the compact disk **48** from the CD library storage **78** and placing the compact disk **48** in the manual load/purge box **80**. The removal age and exceptions information are selected by the system operator from the control console **72**.

The control console **72** is also used to enter and maintain subscriber access identification codes in an "authorization file". This updated user authorization file data is sent through a transmission connector **92** to the telecommunications means **18** internal computer memory accessed by the control computer **70** as needed to accept or reject subscriber data link access requests. The user authorization file normally residing in the telecommunications means **18** may be remotely updated by authorized persons.

The number of data retrieval and transmission channels **81** depends on intended subscriber demand. The image data storage and retrieval means **16** is modular and may be upgraded as demand increases. Each data interface **84** operates cooperatively with the telecommunications means **18** to send only as much information as the telecommunications means **18** can compress and transmit to a remote visual display terminal **20** of a requesting subscriber in a given time interval. Thus, the interface is an asynchronous block-buffered type.

Since the entire system **10** is designed to provide easy and quick access to a patient's medical images, it is vital that these images be transmitted to a variety of locations in a timely and cost effective manner and further data compression is imperative. The telephone network is still the most commonly available network but has a severe data rate limitation of about 1200 bytes per second (9600 baud). While other high speed telecommunication channels such as time-shared cable, satellite link may eventually become commonly available, for the immediately forseeable future, the "phone" network must be used if system **10** is to be practical today.

As noted earlier, a typical medical image may be stored as 119 megabytes of data. At 1200 bytes per second, it could take 27 hours to completely transmit the already compressed medical image data. This is obviously unacceptable. To overcome this obstacle, the telecommunications means **18** as shown in FIG. **5** utilizes five distinct data handling technologies to achieve useful data image transmission in less than one minute:

(1) Guided Image Selection and Transmission or GIST depends upon interactive use by the physician to identify what portions of an image are needed for enhancement or better resolution. Thus the data actually transmitted to the subscriber's visual display terminal **20** is guided by the subscriber observing the image. In particular, once the user has an image displayed on his or her visual display terminal **20**, the user may outline a specific region of interest such as a lesion or tumerous growth for more detailed study. The operator may select this region using a "mouse" or light pen or similar well-known computer display terminal peripheral device Having selected this region, the visual display terminal **20** will display the more detailed pixel data be sent on this region. The telecommunications means **18** will continue to send further precision data until the natural resolution limits of the display are reached or all available data is sent and received. This process of expanding an image region is known as "zooming" in computer-aided design systems. The novel feature here is that the image is further refined in resolution when "zoomed". The means for doing this and knowing when to "stop" further pixel transmission is defined by the PIE and DCR technology described hereinafter,

(2) Progressive Image Enhancement or PIE utilizes the transmission time from the instant a first "crude" image is presented to the subscriber to the present time of observation to progressively enhance the quality of

the presented image. The longer the user observes a selected image, the "better" the image becomes in the sense of pixel resolution and quantity of gray levels. In the preferred embodiment, hexagonal pixel groups are first transmitted using the HexPac pattern compression technology described hereinafter. Once a full terminal screen display has been made composed of these hexagonal patterns, then the telecommunications means **18** transmits more precise pixel detail First all pixels located on the periphery of each hexagonal group are updated with their exact gray level values and thereafter, all inner pixels are similarly updated. If the display terminal's resolution is less than the 1000 dots per inch of the source image data, then pixel groups are sent, such as a square of four pixels, which match the display resolution and "zoom" expansion selected. This display matching technique is further defined hereinafter as DCR,

(3) Display Compatible Resolution or DCR transmits information about the user's terminal **20** back to the telecommunications means **18**. Only data with a resolution compatible with that terminal **20** will be sent. Any excess data-link connect time can be used to send other image data which is likely to be requested or has been pre-specified to be sent.

(4) An image pattern compression method comprising a Hexagonal Pattern Classification or HexPac exploits the two dimensional nature of images. The data received by telecommunications means **18** is first uncompressed and placed into a multi-scanline digital buffer. This image data is then divided up into hexagonal cells and matched against predefined patterns. Many fewer bits of data can be used to represent these predefined patterns thus substantially compressing the image data for phone-line transmission. The pixels of these hexagonal patterns may easily be "refined" by the PIE technology described earlier. If the DCR subsystem determines that the user terminal has a pixel area of, say, 1500 by 1000 dots, then the HexPac technology recreates a new super pixel which is the average gray level of all actual pixels within that super pixel area. This immediately reduces the quantity of pixels to be sent (to only 1500 by 1000 pixels). Without further data compression, this quantity of data would still require about 26 minutes of data transmission time at 9600 baud, the highest available phone network data rate.

(5) Run length coding or RLC permits data to be compressed by specifying how many pixels have the same gray level in a sequence or "run length" of scanning. The image data sent by a CD reader drive **82** to telecommunications means **18** is compressed with run-length coding but is nearly loss-less in the duplication of the original film data. To substantially reduce the quantity of data needed to send an acceptable medical image to a remote user terminal **20** over the data-rate limited phone-line modem, a "lossy" compression is used. Since the PIE and DCR techniques described earlier will eventually provide any degree of diagnostic image integrity desired, it is believed acceptable to initially transmit a "lossy" image provided it gives adequate resolution for the user to begin the analysis and guided image selection. Many fewer bits can describe this "run" of similar gray levels thus compressing the amount of data sent. This technique is well known and often used in facsimile transmission A one dimensional RLC is incorporated in the preferred embodiment but since HexPac elements are being coded, it can be considered more accurately a quasi two dimensional RLC compression.

FIG. **5** is a functional block diagram of the telecommunications means **18** including a control computer **94** operatively coupled to the image data storage and retrieval means **16** through a transmission connector **96**. The various components of the telecommunications means **18** including a status panel **98** with a plurality of system indicators each indicated as **99** and a plurality of data compression channels each generally indicated as **100** coupled to the control computer **94** by a plurality of conductors each indicated as **101**.

Each data compression channel **100** comprises a transmission connector **86**, a communications data interface **102**, a first compression processor or means **104** including logic means to generate the GIST and DCR data compressions and corresponding first data memory **106**, a second compression processor or means **108** including logic means to generate the PIE and HEXPAC data compressions and corresponding second data memory **110** and a third compression processor or means **12** including logic means to generate the RLC data compression and corresponding third data memory **114**, a corresponding modem **116** and a transmission connector **118**.

The control computer **94** coordinates or controls data flow to and from the plurality of data compression channels **100** through the transmission connectors **86** and **118** respectively. Validated subscriber image data requests are transmitted to the image data storage and retrieval means **16** which searches the image library file **78** for availability of the requested compact disk **48**. If available, the image data storage and retrieval means **16** loads the appropriate disk **48** from CD library storage **78** into a CD reader drive **82** and informs telecommunications means from **18** through the transmission connector **96** to the control computer **94** that a specific data interface **84** has data available to be transmitted through the corresponding transmission connectors **86**. Once a subscriber transaction has been turned over to a specific data retrieval and transmission channel **81**, the data compression channel **100** receives the data therefrom unless commanded to stop by a feedback control line. The data interface **102** is used to inform the CD reader drive **82** as to what portion of the image is requested by the first compression means **104**. Generally, the complete image is first requested. Thus the CD reader drive **82** is requested to read the image data from the start.

The data is temporarily stored in the first data memory **106**. Here the pixel data is first expanded from the RLC code into uncompressed pixel data. This is only done on a relatively few number of scan lines—about one tenth of an inch height of original image data. This uncompressed data is then remapped by the first compression means **104** into "larger" pixels whose average intensity is the average of all combined pixels compatible with the display resolution receiving remote visual display terminal **20**. This "super pixel" data is then fed to the second data memory **110**. The super pixel data in memory **110** is then processed by the second compression means **108**. Initially, the lowest resolution image will be transmitted to rapidly form a useful remote image on the requesting remote visual display terminal **20** through a communications network **21**. This will be done by combining super pixels in the second data memory **110** into hexagonal patterns which approximate the group of super pixels. These HexPac data packets are then sent to the third data memory **114**. There the Hex-

Pac data packets are further compressed by the third compression means **112**. These packets of run-length coded HexPac data packets are then transmitted through the corresponding modem **116** and transmission connector **118** over the selected communications network **21**. The modem **116** includes state of the art error control techniques such as block retransmission when a remote error has been detected. Thus, data transmission is essentially error free as needed for compressed data handling.

The control computer **94** includes circuitry means to monitor the activity of each data channel. The identification of each subscriber is logged along with the total connect time for billing purposes. Thus the control computer **94** generally coordinates the plurality of communication links and their connections to the particular data retrieval and transmission channel **81** within the image data storage and retrieval means **16** as well as granting access and performing connection accounting tasks. The status panel **98**, connected to the control computer **94** is used to aide in system debug and indicate operation of the data compression channels **100**. The status panel **98** would not normally be used by hospital personnel but by system service technicians.

The control computer **94** also has a permanent memory such as a hard disk to record subscriber usage data and internally sensed hardware problems. This data may be downloaded on any of the transmission connectors **118** when a correct authorization code has been received. Thus, the servicing company can acquire subscriber usage information remotely for billing purposes and system diagnostic purposes.

The preferred embodiment of the telecommunications means **18** uses modular communication channel hardware. Thus, the module may be customized to function with any type of communication channel such as satellite links, cable networks or a local area network such as Ethernet or ISDN services.

It is important to note that all communications is bidirectional so that if, say, a remote visual display terminal **20** should become temporarily "overloaded" with image data due to decompression processing delays or due to a detected data error, then, the remote visual display terminal **20** may request that data transmission be stopped or a block of data be repeated until it is received correctly.

FIG. **7** graphically shows a hexagonal group of the hexagonal compression method comprising a group of square image pixels partitioned into a hexagonal group. The pixels are numbered for convenience of reference from the inside to the outside in a clock-wise manner. Each hexagonal group or packet comprises 24 super pixels as earlier described but other numbers are possible. It is assumed that each pixel is gray level coded using 2 bytes of data. Thus, the hexagonal group requires (24×2) 48 bytes of data to fully represent the 24 super pixels comprising the image pattern at the user terminal **20**.

FIG. **8** shows a typical pattern as may occur in a region of an X-ray film **12**. The method predefines a group of likely patterns, one of which is represented as a "best" match as in FIG. **9** with the actual pattern in FIG. **8**. As shown in FIGS. **14A** through **14H**, there are 8 possible predetermined representative gray level patterns represented by 3 bits. These patterns are specifically selected to be essentially uncorrelated with each other even rotated relative to each other. As shown in FIG. **10**, these patterns may be rotated through 8 equal angles (another 3 bits of data) to best match the actual pattern. Rotation angle "1" is shown in FIG. **10** as the best match for the given example. Thus far, six bits have been used to approximate the actual pattern of FIG. **8**. As shown in FIGS. **14A** through **14H**, each fictitious pattern includes a dark and light regions and origin. Although FIG. **11** discloses a straight gray level slope corresponding to the pattern shown in FIG. **14A**, the gray level slope will vary with the fictitious pattern. For example, the gray level slope of the fictitious pattern shown in FIG. **14D** would closely approximate a V shape.

FIG. **11** shows how the gray level slope may be discretely selected to best match the slope of the actual pattern. Two bits are used to approximate this slope.

FIG. **12** shows that one particular pixel, such as the darkest pixel, has been selected to be fairly precisely gray level represented by means of 8 bits (256 gray levels).

The total bits required to approximate the actual pattern is 16 or two bytes. FIG. **13** shows how this fictitious or reconstructed pattern may be reproduced at the user terminal **20** when decoded.

In this example, only two bytes were required to represent "adequately" an original 48 bytes of image data. Thus, a 24 to 1 compression ratio has been achieved. Further, run-length encoding (RLC) may be used on these HexPac Groups to further reduce redundant spans of white and black. It is estimated that the combined compression ratio of HexPac and RLC on the super-pixel image is about 36 to 1 for this particular set of parameters. This combined compression technology reduces data transmission time (at 9600 baud) to approximately 43 seconds for an initial useful medical image.

For medical images, further enhancements through the PIE compression should favor the elimination of artificial lining between hexagonal patterns first. As the user continues to view the same image, then the PIE compression will progressively improve the gray level integrity by updating all number 24 pixels to 8 bits of gray level resolution and updating all number 23 pixels to 8 bits of gray level and so forth for all remaining pixels in descending order. This process takes about 10 minutes at 9600 baud to update all peripheral hexagonal pixels and about 20 minutes total for all pixels.

If the user continues to observe or request further image resolution, the telcommunication means **20** causes each pixel gray level to be updated by one additional bit in descending order again until the full 16 bits of gray level is received and stored at the terminal **20** for each super pixel. Each doubling of gray level resolution takes between 1 and 2.6 minutes at 9600 baud depending on the run length statistics of the gray levels.

FIG. **6** is a functional block diagram of a remote visual display terminal **20** to be operatively coupled to one of the data compression channels **100** of the telecommunication means **18** by a communications network **21** and a transmission connector **118**. The visual display terminal **20** comprises a data communications modem **120** operatively coupled to a control computer **122** and RLC decompression means **124**. The RLC decompression means **124** is, in turn, operatively coupled to a memory **126**, a PIE bypass **128** and a pattern select and modifier **130** which is operatively coupled to a Hexpac pattern ROM **132** and the control computer **122**. A memory **134** is operatively coupled between the PIE bypass **128** and pattern selector and modifier **130** and a display drive **136** which is operatively coupled to an

image display **138**. In addition, an image enhancing processor means **139** including circuitry to generate edge contrast enhancement, gray level contrast enhancement by means of gray level region expansion or differential gray level tracking and gray level enhancement or other state of the art image enhancement method well known to those skilled in the art. The control computer **122** is operatively coupled to an interface **140** to a first control or selector means **142** and a second control or selector means including a radio receiver **144** and signal command decoder **146** for use with portable keyboard transmitter **148**. In addition, an optional CD read/write drive **150** may be provided for use with a compact disk **48**.

The modem **120** has built-in compatible error correction technology to communicate with corresponding transmitting data compression channel **100**. After the user has selected the image data storage and retrieval means **16** and validated authority by swiping through an identification magnetic card **152** or otherwise through a magnetic card reader **154** entered an assigned security code, the operator may select a patient and one or more image files presented to him on the display screen **138**. Selection is accomplished by a touch-screen overlay on the first control or selector means **142** or by the keyboard transmitter **148** of the second control or selector means.

Once one or more images have been selected by the user, the modem **120** writes image data to the temporary memory **126** which is actively accessed by the RLC decompression means **124**. This decompressed data describes the HexPac patterns or packets as stripes of the image running, for example sequentially from left to right. These Hexpac pattern specifications, typically 2 data bytes or 16 bits are then routed to a pattern selection processor **130** which accesses the predefined patterns from Read-Only Memory device **132**. Each pattern is then rotated and gray-level modified by processor **130** according to the HexPac 16 bit pattern specification received from the RLC decompression means **124**. Each modified pattern is then written to a graphics display memory circuit **134**. As the graphics display memory circuit **134**, develops the pattern data, the display driver **136** and image display **138** show the image on the screen as it is received. In this manner, the entire "first pass" medical image is painted on the image display **138** screen.

If the user makes no further intervention, then once the image is fully displayed on this "first pass", then the progressive image enhancement technology requests pixel enhancement data. This enhancement data bypasses pattern selector and modifier **130** and is routed through the PIE bypass **128**. In the PIE bypass **128**, the enhanced pixel information is directed to the correct graphic memory locations in the graphics display means circuit **134'**. Thus, the display driver **136** and image display **138** are continually resolution enhanced.

If the image is fully enhanced to the limits set by the DCR in the control computer **122**, the image storage and retrieval means **16** is directed by the control computer **122** to begin sending new image data on the next selected image and begin storing this image data in a second graphics display memory circuit **134''**. This second data memory **134''** can hold one more images and may be selected immediately by the user when he is finished inspecting an earlier image. The user may further direct by touch screen command **142** that these be stored in the computer's hard disk or archived by the optional CD read/write drive **150**.

The user may at any time select a portion of the displayed image for further expansion by enabling or selecting the Guided Image Selection & Transmission (GIST) circuitry in the control computer **122** or image enhancement through the image enhancing processor means **139**. This may be accomplished either by touch screen control means **142** or the second remote control means **144/146/148**. This remote keyboard and transmitter unit **144/146/148** duplicates the on-display simulated push-buttons of the touch screen control means **142**. Coded command signals sent by **148** are received by radio receiver **144** and decoded by **146**. These commands are then accepted by control computer **122** as though they were normal keyboard commands.

The user may terminate a session with the image data storage and retrieval means **16** at any time by selection of stop and escape command. While a printer is not shown in this description, it can be an optional addition to terminal **20**.

In summary, the image data storage and retrieval means **16** selects the first image and writes that data to a temporary memory buffer in the telecommunication means **18**. Information about the subscriber's terminal is uploaded to the telecommunications means **18** so that the Display Compatible Resolution (DCR) logic circuitry knows when to stop sending added data for the requested first image. A special interactive compression computer then compresses this first image data using the HexPac circuitry and sends that data over the data link modem to the subscriber terminal **20**. Error detection and correction methods will generally be used in this communications link protocol.

Once a first "crude" image is sent to the subscriber visual display terminal **20**, then the Progress Image Enhancement (PIE) circuitry begins to send additional data to further refine the resolution of each hexagonal pixel region. If no further guidance is given by the subscriber, the PIE will continue to refine the picture's resolution until its natural limit is sent for the terminal **20**. Thereafter, the PIE will begin sending image data from the second specified film and loading it into yet another memory buffer. Thus, the data link connection is always transmitting useful data even though the subscriber may only be analyzing one image for some time.

However, should the user desire to zoom in on a particular region of an image, he or she may define that region desired on the terminal **20** by the Guided Image Selection & Transmission (GIST) to expand the visual display accordingly. The DCR will recognize the requirement for additional resolution and command the PIE to begin transmitting additional pixel information until such time as the DCR informs it that once again the natural display resolution limit has been reached.

The following image enhancement means present in the instant invention: edge contrast enhancement, gray level contrast enhancement by means of gray level region expansion or differential gray level tracking and gray level enhancement and may be accomplished by the image enhancing processor means **139** in the visual display terminals **20**.

The human eye cannot reliably discern gray level differences less than approximately 2%. Yet, significant tissue density information causes X-ray gray level differences in this range and below. The enhancement technologies above will cause these tissue density differ-

ences to be magnified thus revealing hitherto unseen image data.

To ensure ease of use, the following features are incorporated: touch-screen selection of commands, magnetic card identification of the subscriber or user, icon based menus or selection buttons on the CRT display and split image display screen overlays

It will thus be seen that the objects set forth above, among those made apparent from the preceding description are efficiently attained and since certain changes may be made in the above construction without departing from the scope of the invention, it is intended that all matter contained in the above description or shown in the accompanying drawing shall be interpreted as illustrative and not in a limiting sense.

It is also to be understood that the following claims are intended to cover all of the generic and specific features of the invention herein described, and all statements of the scope of the invention which, as a matter of language, might be said to fall therebetween.

Now that the invention has been described,

What is claimed is:

1. A medical image storage, retrieval and transmission system providing simultaneous automated access to an image data base by a plurality of remote subscribers upon request over a communications network, said system comprising

image digitizing means forming a first digitized representation of said image,

first data compression means generating from said first digitized representation a low-loss second digitized representation of said image,

image data storage and retrieval means comprising

means to receive and store said second digitized representation and to selectively provide said second digitized representation to data channel compression means compressing said second digitized representation to form a third digitized representation, and

telecommunication means including means to selectively transmit said requested third digitized representation to a requesting remote visual display terminal, said telecommunication means initially telecommunicating a first portion of said third digital representation, said remote terminal including means to convert said first portion of said third representation to a visual image having an initial resolution less than a resolution limit of said requesting terminal, said telecommuncation means subsequently telecommunicating a second portion of said third digital representation, said second portion usable with said first portion to form an image having a resolution intermediate said initial resolution and said resolution limit.

2. A system of claim 1 wherein said first data compression means includes logic means to generate a run-length compressed digitized image data signal as said second representation.

3. A system of claim 1 wherein said first data compression means is operatively coupled to an external data storage drive to store said second representation.

4. A system of claim 1 wherein said image data storage and retrieval means comprises a data modem coupled to said image scanning and digitizing means, a write drive operatively coupled to said data modem to receive and to store said second digitized representation, and a plurality of data retrieval and transmission channels, each said channel comprising an image data reader means operatively coupled to said telecommunication means to selectively receive said second digitized representation of said image for transmission to a said requesting remote visual display terminal.

5. A system of claim 4, wherein a said image data reader means is operatively coupled to an external data write drive configured to receive a storage medium and to store compressed digital image data thereon.

6. A system of claim 4 wherein said telecommunication means comprises a control computer operatively coupled to said image data storage and retrieval means to selectively control data flow between said image data storage and retrieval means and said remote visual display terminal and a plurality of data compression channels coupled to said control computer, wherein each said data compression channel comprises a data memory including means to decompress said low-loss second representation of said image data received from said data retrieval and transmission channel and a compression means including logic means to compress and decompress second representation of said image data to form said third digitized representation of said image for transmission over said communication network to a said requesting remote visual display terminal.

7. A system of claim 1 wherein said first portion of said third representation of said image comprises a plurality of super pixels, and said second portion of said third representation comprises data representative of exact gray levels of a first subset of said super pixels and wherein a third portion of said third representation comprises similar data for a third subset of said super pixels.

8. A system of claim 1 wherein said remote terminal further includes means to select a region of a said image and said telecommunication means includes means to transmit a third portion of said third digitized representation, said third portion specific to said selected region, thereby providing an expanded visual display of said selected region, said expanded visual display containing more pixels than were included in said selected region.

9. A system of claim 1 wherein said remote visual display terminal further includes logic means to enhance an edge contrast of a displayed image.

10. A system of claim 1 wherein said remote visual display terminal further includes logic means to enhance gray level contrast by means of gray level region expansion.

11. A system of claim 1 wherein said remote visual display terminal further includes logic means for differential gray level tracking and gray level enhancement.

12. A system of claim 1 wherein individual patient information corresponding to said image is read by an optical character reader for compression and transmission with a said corresponding second digital representation to said image data storage and retrieval means.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 5,321,520

DATED : June 14, 1994

INVENTOR(S) : Jorge J. Inga, et al

It is certified that error appears in the above-indentified patent and that said Letters Patent is hereby corrected as shown below:

Column 20, lines 27 and 28, "and decompress" should read --said decompressed--.

Signed and Sealed this

Twenty-fifth Day of October, 1994

Attest:

Attesting Officer

BRUCE LEHMAN

Commissioner of Patents and Trademarks

US005321520C1

(12) EX PARTE REEXAMINATION CERTIFICATE (8039th)

United States Patent
Inga et al.

(10) **Number: US 5,321,520 C1**
(45) **Certificate Issued: Feb. 22, 2011**

(54) **AUTOMATED HIGH DEFINITION/RESOLUTION IMAGE STORAGE, RETRIEVAL AND TRANSMISSION SYSTEM**

(75) Inventors: **Jorge J. Inga**, Tampa, FL (US); **Thomas V. Saliga**, Tampa, FL (US)

(73) Assignee: **Automated Medical Access Corporation**, Tampa, FL (US)

Reexamination Request:
No. 90/011,260, Sep. 30, 2010

Reexamination Certificate for:

Patent No.:	**5,321,520**
Issued:	**Jun. 14, 1994**
Appl. No.:	**07/915,298**
Filed:	**Jul. 20, 1992**

Certificate of Correction issued Oct. 25, 1994.

(51) **Int. Cl.**
H04N 1/21 (2006.01)
H04N 1/41 (2006.01)

(52) **U.S. Cl.** **358/403**; 358/1.9
(58) **Field of Classification Search** 358/403
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,520,671 A 6/1985 Hardin
4,903,317 A 2/1990 Nishihara et al.
4,941,190 A 7/1990 Joyce

OTHER PUBLICATIONS

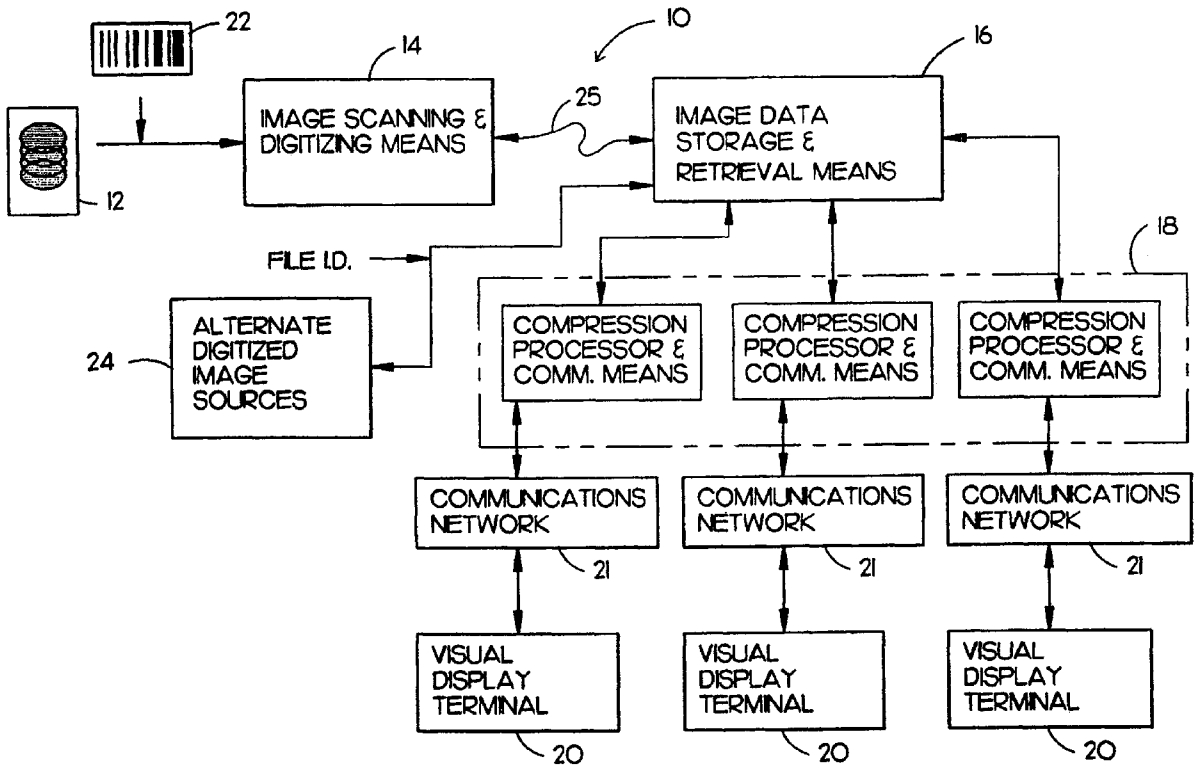
Sharaf E. Elnahas, Progressive Coding and Transmission of Digital Diagnostic Pictures, MI–5 IEEE Transactions on Med. Imaging 73 (Jun. 1986).*

* cited by examiner

Primary Examiner—Henry N Tran

(57) **ABSTRACT**

An automated high definition/resolution image storage, retrieval and transmission system for use with medical X-ray film or other documents to provide simultaneous automated access to a common data base by a plurality of remote subscribers upon request, the automated high definition/resolution image storage, retrieval and transmission system comprising an image scanning and digitizing subsystem to scan and digitize visual image information from an image film or the like; an image data storage and retrieval subsystem to receive and store the digitized information and to selectively provide the digitized information upon request from a remote site, a telecommunication subsystem to selectively transmit the requested digitized information from the image data storage and retrieval subsystem to the requesting remote visual display terminal for conversion to a visual image at the remote site to visually display the requested information from the image data storage and retrieval subsystem.



EX PARTE REEXAMINATION CERTIFICATE ISSUED UNDER 35 U.S.C. 307

NO AMENDMENTS HAVE BEEN MADE TO THE PATENT

AS A RESULT OF REEXAMINATION, IT HAS BEEN DETERMINED THAT:

The patentability of claims **1-12** is confirmed.

* * * * *

US006182114B1

(12) **United States Patent**
Yap et al.

(10) **Patent No.: US 6,182,114 B1**
(45) **Date of Patent: Jan. 30, 2001**

(54) **APPARATUS AND METHOD FOR REALTIME VISUALIZATION USING USER-DEFINED DYNAMIC, MULTI-FOVEATED IMAGES**

(75) Inventors: **Chee K. Yap**; **Ee-Chien Chang**, both of New York, NY (US); **Ting-Jen Yen**, Jersey City, NJ (US)

(73) Assignee: **New York University**, New York, NY (US)

(*) Notice: Under 35 U.S.C. 154(b), the term of this patent shall be extended for 0 days.

(21) Appl. No.: **09/005,174**

(22) Filed: **Jan. 9, 1998**

(51) **Int. Cl.**[7] **G06F 15/16**
(52) **U.S. Cl.** **709/203**; 709/246
(58) **Field of Search** 709/217, 219, 709/246, 247, 203; 707/10; 382/103, 233, 235, 232, 240, 302

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,622,632		11/1986	Tanimoto .	
5,341,466		8/1994	Perlin .	
5,481,622	*	1/1996	Gerhardt et al.	382/103
5,568,598	*	10/1996	Mack et al.	382/302 X
5,710,835	*	1/1998	Bradley	382/233
5,724,070	*	3/1998	Denninghoff et al.	382/235 X
5,861,920	*	1/1999	Mead et al.	382/232 X
5,880,856	*	3/1999	Ferriere	382/240 X
5,920,865	*	7/1999	Ariga	707/10

OTHER PUBLICATIONS

Tams Frajka et al., Progressive Image Coding with Spatially Variable Resolution, IEEE, Proceedings International Conference on Image Processing 1997, Oct. 1997, vol. 1, pp. 53–56.*

E. C. Chang et al., "Realtime Visualization of Large . . . " Mar. 31, 11997,pp. 1–9, Courant Institute of Mathematical Sciences, New York University, N.Y. U.S.A.

E. C. Chang et al., "A Wavelet Approach to Foveating Images", Jan. 10, 1997,pp. 1–11, Courant Institute of Mathematical Sciences, New York University, N.Y. U.S.A.

S.G. Mallat, "A Theory for Multiresolutional Signal Decomposition . . . ", IEEE Transactions on Pattern Analysis and Machine Intelligence,pp. 3–23, Jul. 1989, vol. 11, No. 7, IEEE Computer Society.

News Release, "Wavelet Image Features",Summus'Wavelet Image Compression,Summus 14 pages.

R.L. White et al., "Compression and Progressive Transmission of Astronomical Images", SPIE Technical Conference 2199, 1994.

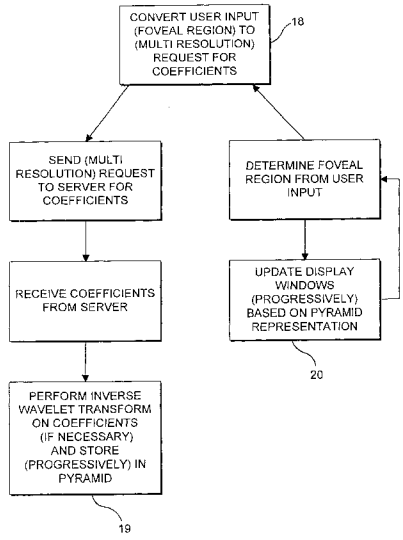
(List continued on next page.)

Primary Examiner—Zarni Maung
Assistant Examiner—Patrice Winder
(74) *Attorney, Agent, or Firm*—Baker Botts, L.L.P.

(57) **ABSTRACT**

A client apparatus which enables a realtime visualization of at least one image. The client apparatus includes a storage device which stores first data corresponding to a multifoveated representation of an original image, and a user input device which providing second data corresponding to at least one visualization command of at least one user. In addition, the client apparatus includes a processing arrangement which generates third data corresponding to a multi-foveated image using the first data, the second data and a foveation operator.

8 Claims, 6 Drawing Sheets



OTHER PUBLICATIONS

E.L. Schwartz, "The Development of Specific Visual . . . " Journal of Theoretical Biology, 69:655–685, 1977.
F.S. Hill Jr. et al.,"Interactive Image Query . . . " Computer Graphics, 17(3), 1983.
T.H. Reeves et al., "Adaptive Foveation of MPEG Video", Proceedings of the 4th ACM International Multimedia Conference, 1996.
R.S. Wallace et al., "Space–variant image processing". Int'l. J. of Computer Vision, 13:1(1994) 71–90.
E.L. Schwartz A quantitative model of the functional architecture: Biological cybernetics, 37(1980) 63–76.
P. Kortum et al., "Implementation of a Foveated Image . . . " Human Vision and Electronic Imagining, SPIE Proceedings vol. 2657, 350–360, 1996.
M.H. Gross et al., "Efficient triangular surface . . . ", IEEE Trans on Visualization and Computer Graphics, 2(2) 1996.

* cited by examiner

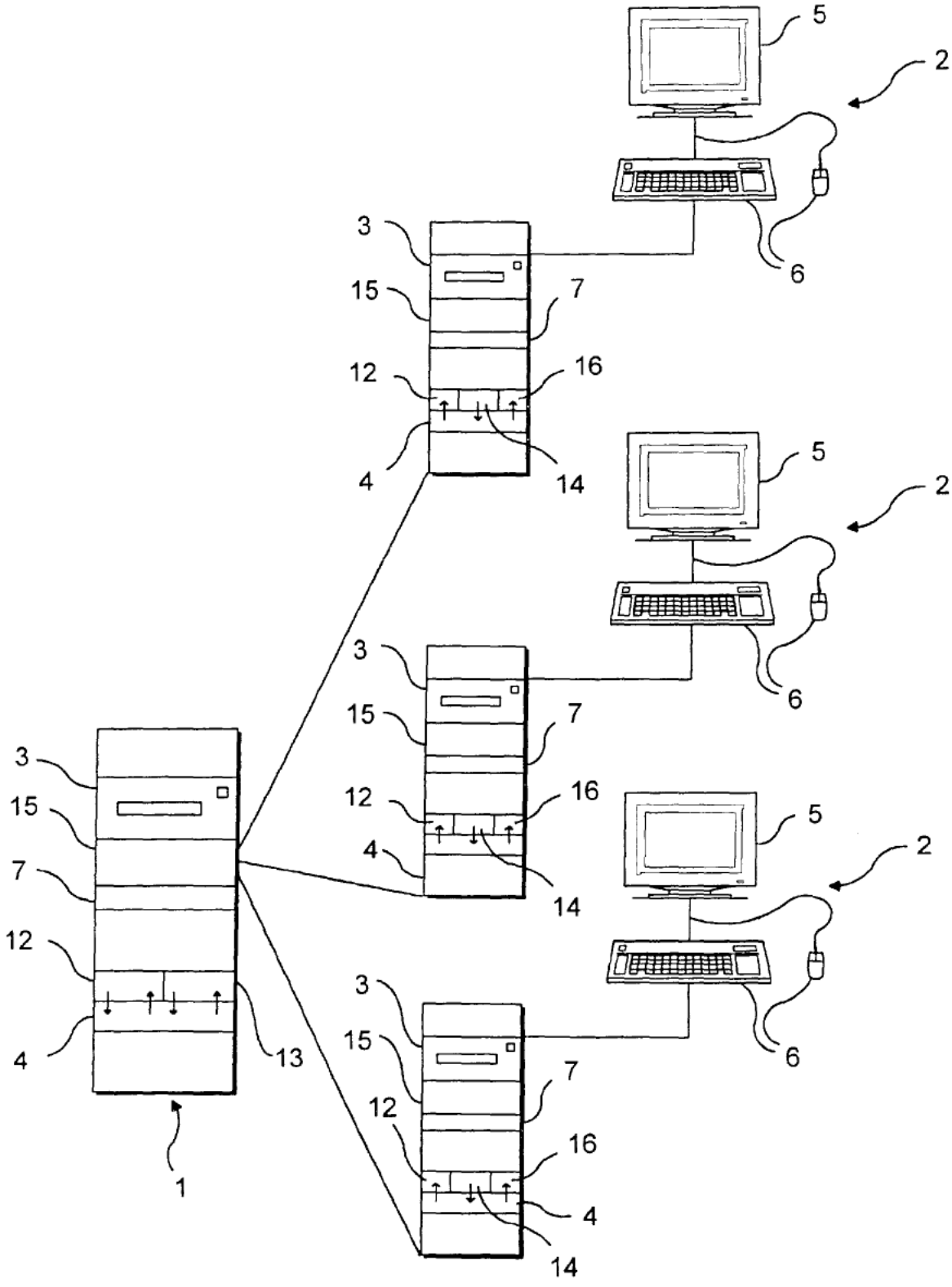


FIG. 1

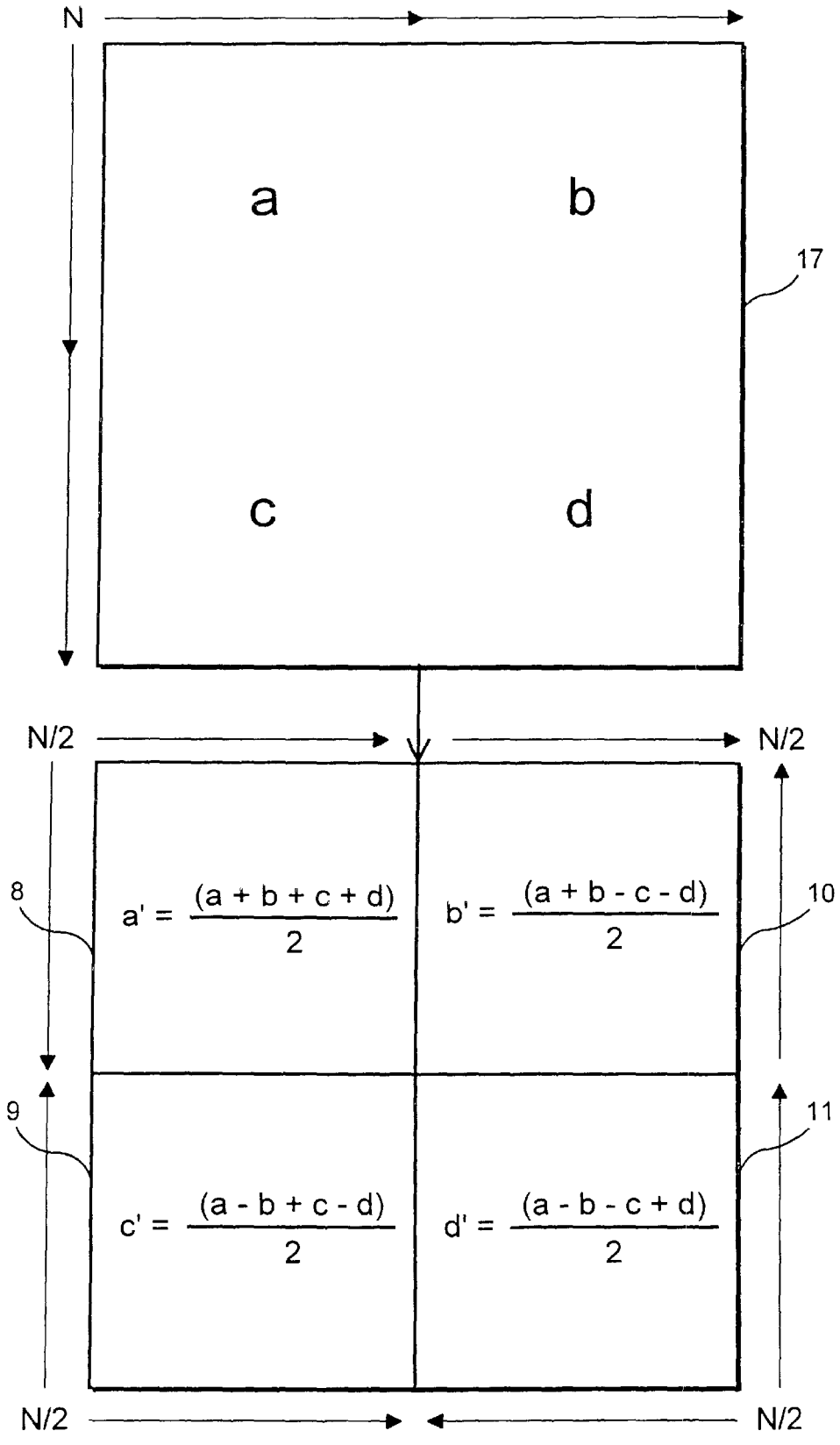

N
a
b
c
d
17
N/2
N/2
8
$a' = \frac{(a + b + c + d)}{2}$
$b' = \frac{(a + b - c - d)}{2}$
10
9
11
$c' = \frac{(a - b + c - d)}{2}$
$d' = \frac{(a - b - c + d)}{2}$
N/2
N/2

FIG. 2A

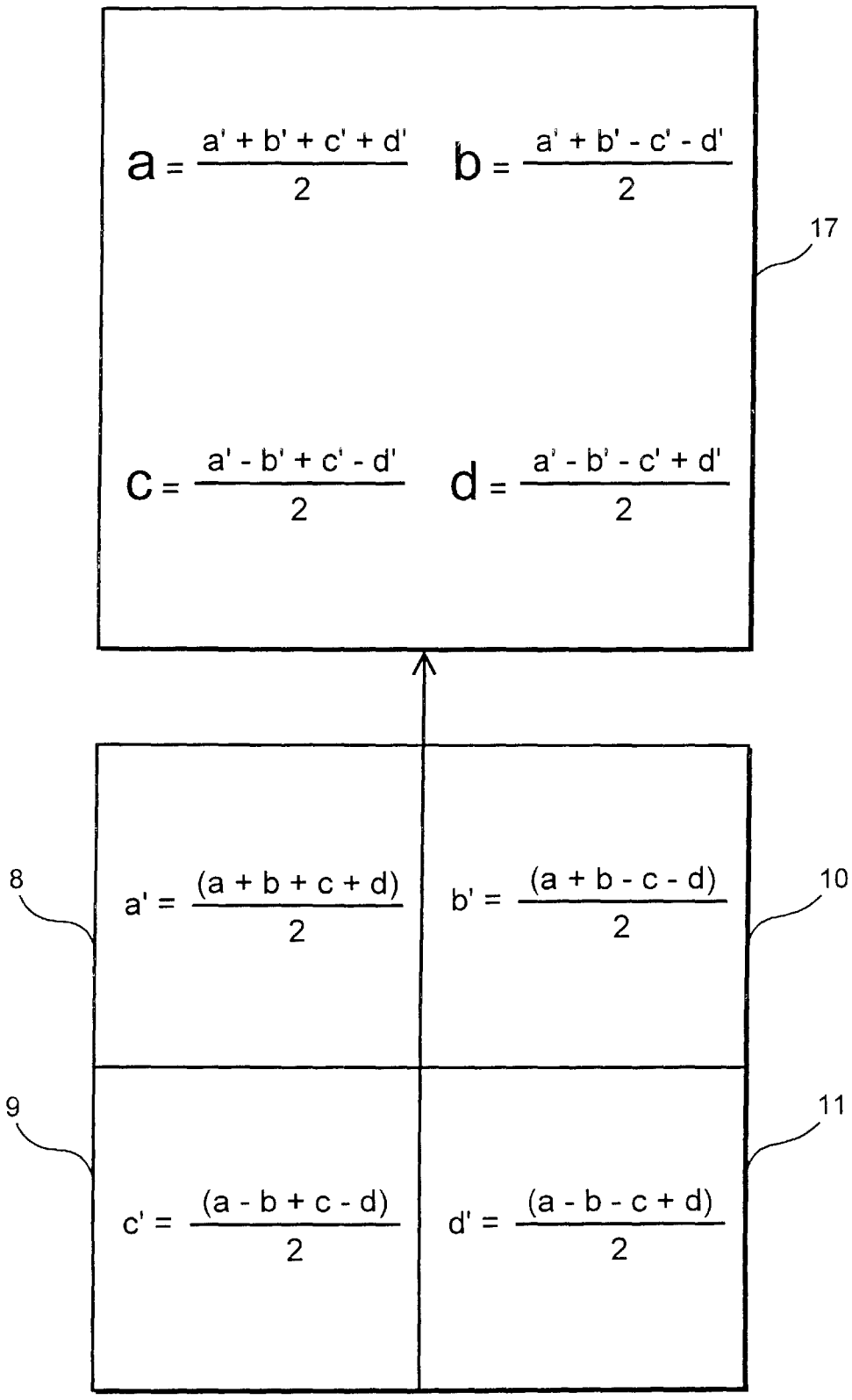

$a = \frac{a' + b' + c' + d'}{2}$
$b = \frac{a' + b' - c' - d'}{2}$
17
$c = \frac{a' - b' + c' - d'}{2}$
$d = \frac{a' - b' - c' + d'}{2}$
8
$a' = \frac{(a + b + c + d)}{2}$
$b' = \frac{(a + b - c - d)}{2}$
10
9
$c' = \frac{(a - b + c - d)}{2}$
$d' = \frac{(a - b - c + d)}{2}$
11

FIG. 2B

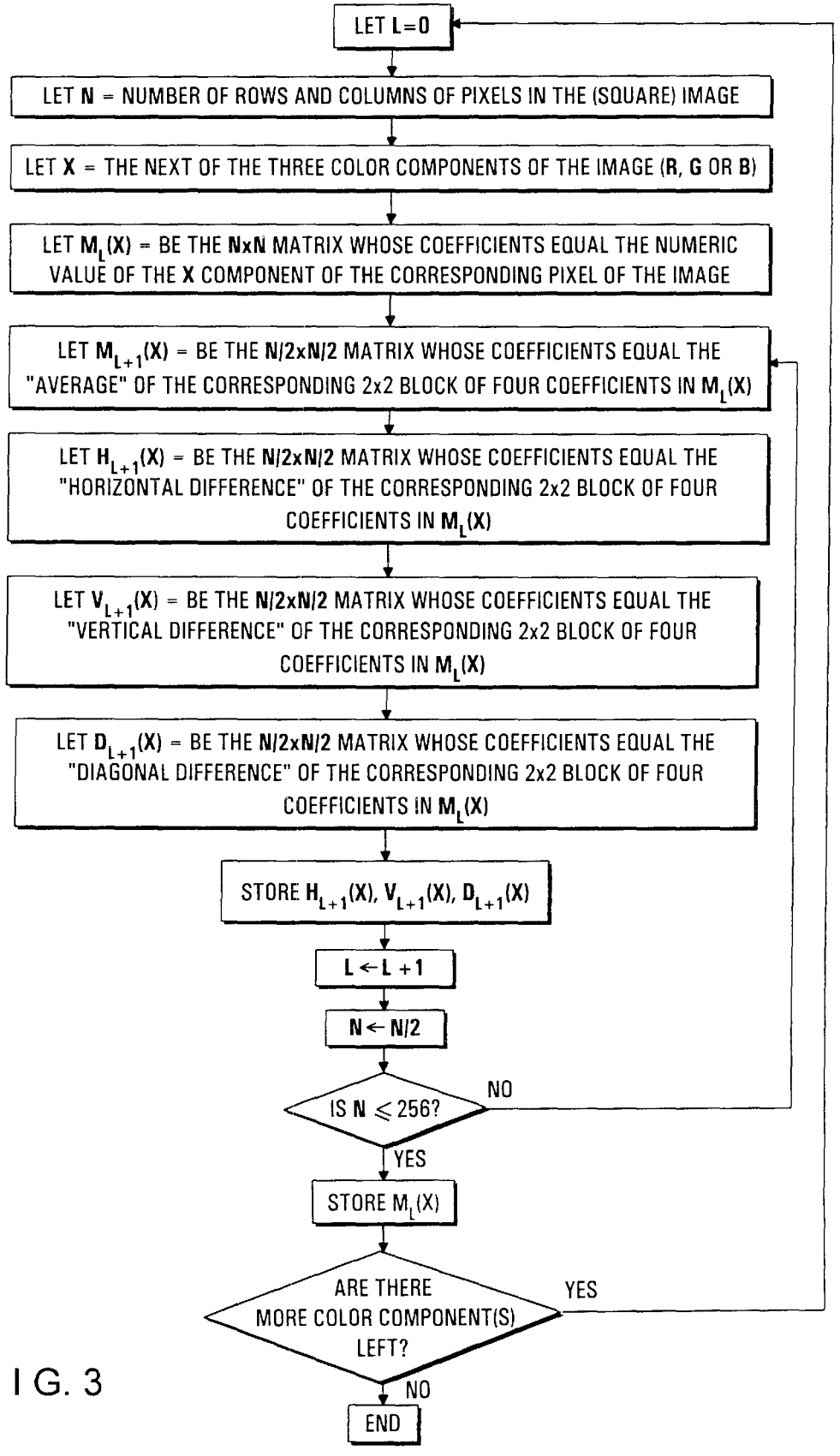

LET L=0
LET N = NUMBER OF ROWS AND COLUMNS OF PIXELS IN THE (SQUARE) IMAGE
LET X = THE NEXT OF THE THREE COLOR COMPONENTS OF THE IMAGE (R, G OR B)
LET $M_L(X)$ = BE THE NxN MATRIX WHOSE COEFFICIENTS EQUAL THE NUMERIC VALUE OF THE X COMPONENT OF THE CORRESPONDING PIXEL OF THE IMAGE
LET $M_{L+1}(X)$ = BE THE N/2xN/2 MATRIX WHOSE COEFFICIENTS EQUAL THE "AVERAGE" OF THE CORRESPONDING 2x2 BLOCK OF FOUR COEFFICIENTS IN $M_L(X)$
LET $H_{L+1}(X)$ = BE THE N/2xN/2 MATRIX WHOSE COEFFICIENTS EQUAL THE "HORIZONTAL DIFFERENCE" OF THE CORRESPONDING 2x2 BLOCK OF FOUR COEFFICIENTS IN $M_L(X)$
LET $V_{L+1}(X)$ = BE THE N/2xN/2 MATRIX WHOSE COEFFICIENTS EQUAL THE "VERTICAL DIFFERENCE" OF THE CORRESPONDING 2x2 BLOCK OF FOUR COEFFICIENTS IN $M_L(X)$
LET $D_{L+1}(X)$ = BE THE N/2xN/2 MATRIX WHOSE COEFFICIENTS EQUAL THE "DIAGONAL DIFFERENCE" OF THE CORRESPONDING 2x2 BLOCK OF FOUR COEFFICIENTS IN $M_L(X)$
STORE $H_{L+1}(X)$, $V_{L+1}(X)$, $D_{L+1}(X)$
L ← L + 1
N ← N/2
IS N ⩽ 256?
NO
YES
STORE $M_L(X)$
ARE THERE MORE COLOR COMPONENT(S) LEFT?
YES
NO
END

FIG. 3

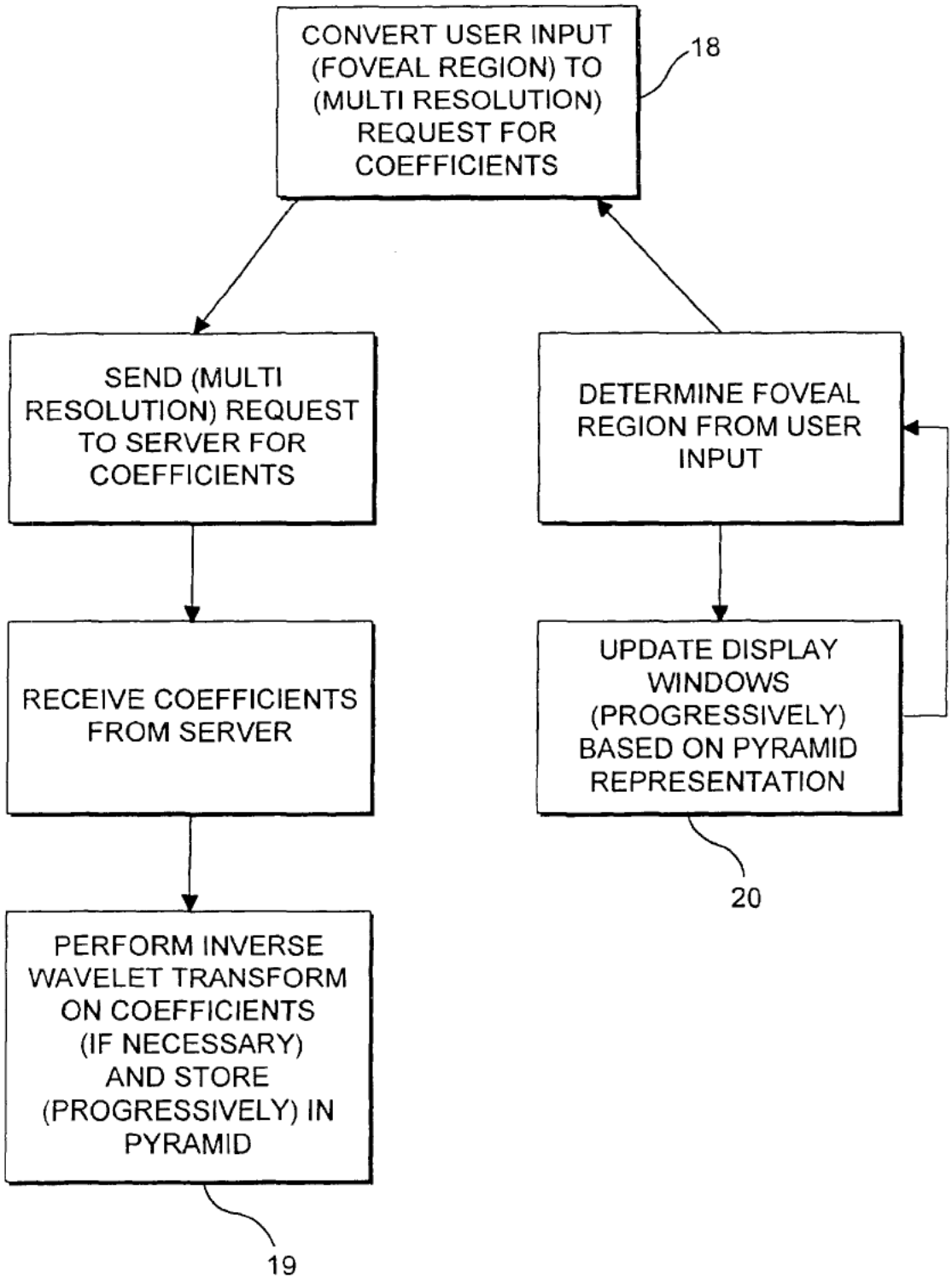
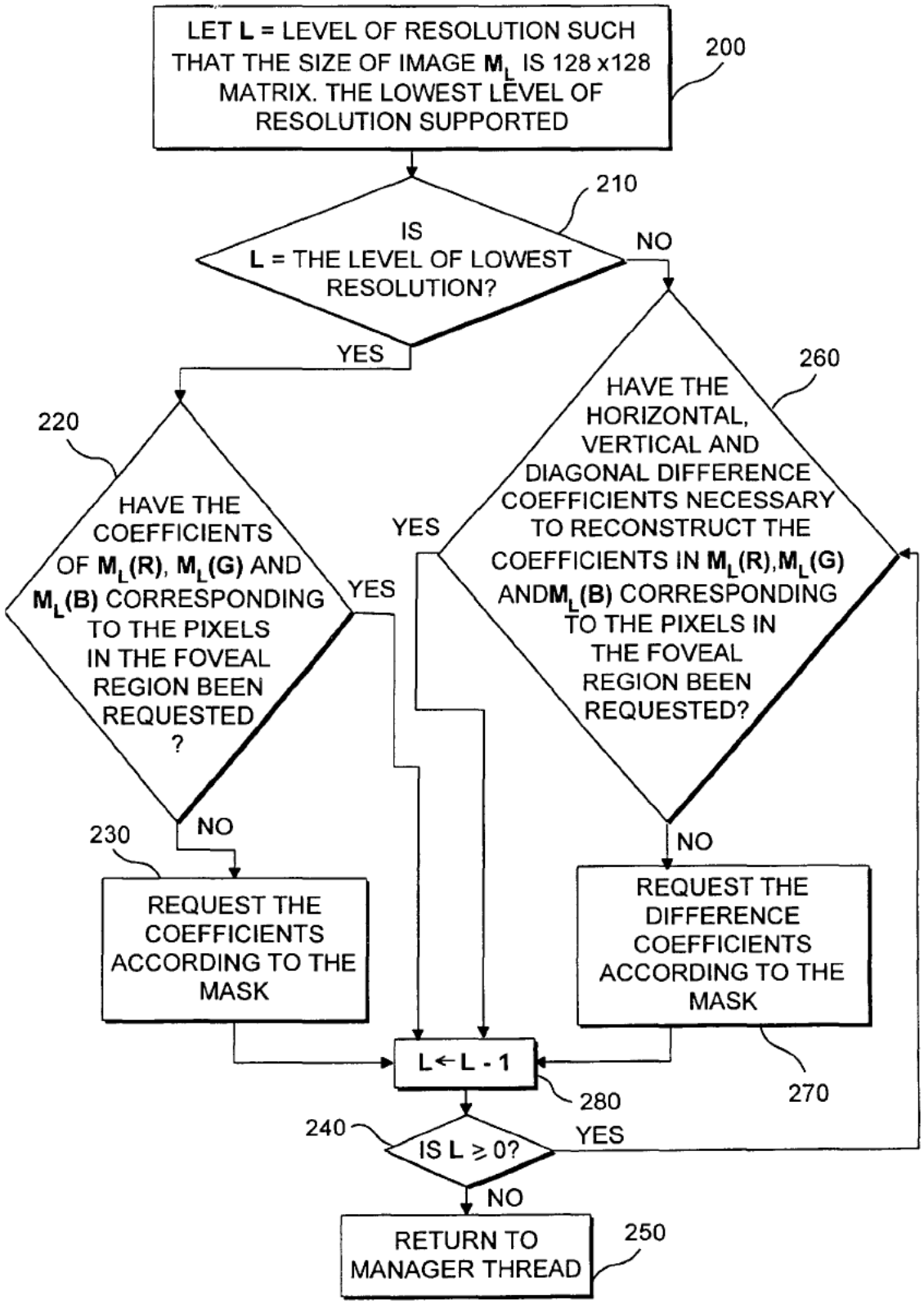

CONVERT USER INPUT (FOVEAL REGION) TO (MULTI RESOLUTION) REQUEST FOR COEFFICIENTS
18
SEND (MULTI RESOLUTION) REQUEST TO SERVER FOR COEFFICIENTS
DETERMINE FOVEAL REGION FROM USER INPUT
RECEIVE COEFFICIENTS FROM SERVER
UPDATE DISPLAY WINDOWS (PROGRESSIVELY) BASED ON PYRAMID REPRESENTATION
20
PERFORM INVERSE WAVELET TRANSFORM ON COEFFICIENTS (IF NECESSARY) AND STORE (PROGRESSIVELY) IN PYRAMID
19

FIG. 4


LET L = LEVEL OF RESOLUTION SUCH THAT THE SIZE OF IMAGE M_L IS 128 x128 MATRIX. THE LOWEST LEVEL OF RESOLUTION SUPPORTED
200
IS L = THE LEVEL OF LOWEST RESOLUTION?
210
NO
YES
HAVE THE COEFFICIENTS OF $M_L(R)$, $M_L(G)$ AND $M_L(B)$ CORRESPONDING TO THE PIXELS IN THE FOVEAL REGION BEEN REQUESTED ?
220
YES
NO
REQUEST THE COEFFICIENTS ACCORDING TO THE MASK
230
HAVE THE HORIZONTAL, VERTICAL AND DIAGONAL DIFFERENCE COEFFICIENTS NECESSARY TO RECONSTRUCT THE COEFFICIENTS IN $M_L(R)$,$M_L(G)$ AND$M_L(B)$ CORRESPONDING TO THE PIXELS IN THE FOVEAL REGION BEEN REQUESTED?
260
YES
NO
REQUEST THE DIFFERENCE COEFFICIENTS ACCORDING TO THE MASK
270
L←L - 1
280
IS L ≥ 0?
240
YES
NO
RETURN TO MANAGER THREAD
250

F I G. 5

APPARATUS AND METHOD FOR REALTIME VISUALIZATION USING USER-DEFINED DYNAMIC, MULTI-FOVEATED IMAGES

FIELD OF THE INVENTION

The present invention relates to a method and apparatus for serving images, even very large images, over a "thinwire" (e.g., over the Internet or any other network or application having bandwidth limitations).

BACKGROUND INFORMATION

The Internet, including the World Wide Web, has gained in popularity in recent years. The Internet enables clients/users to access information in ways never before possible over existing communications lines.

Often, a client/viewer desires to view and have access to relatively large images. For example, a client/viewer may wish to explore a map of a particular geographic location. The whole map, at highest (full) level of resolution will likely require a pixel representation beyond the size of the viewer screen in highest resolution mode.

One response to this restriction is for an Internet server to pre-compute many smaller images of the original image. The smaller images may be lower resolution (zoomed-out) views and/or portions of the original image. Most image archives use this approach. Clearly this is a sub-optimal approach since no preselected set of views can anticipate the needs of all users.

Some map servers (see, e.g., URLs http://www.mapquest.com and http://www.MapOnUs.com) use an improved approach in which the user may zoom and pan over a large image. However, transmission over the Internet involves significant bandwidth limitations (i.e transmission is relatively slow). Accordingly, such map servers suffer from at least three problems:

- Since a brand new image is served up for each zoom or pan request, visual discontinuities in the zooming and panning result. Another reason for this is the discrete nature of the zoom/pan interface controls.
- Significantly less than realtime response.
- The necessarily small fixed size of the viewing window (typically about 3"×4.5"). This does not allow much of a perspective.

To generalize, what is needed is an apparatus and method which allows realtime visualization of large scale images over a "thinwire" model of computation. To put it another way, it is desirable to optimize the model which comprises an image server and a client viewer connected by a low bandwidth line.

One approach to the problem is by means of progressive transmission. Progressive transmission involves sending a relatively low resolution version of an image and then successively transmitting better resolution versions. Because the first, low resolution version of the image requires far less data than the full resolution version, it can be viewed quickly upon transmission. In this way, the viewer is allowed to see lower resolution versions of the image while waiting for the desired resolution version. This gives the transmission the appearance of continuity. In addition, in some instances, the lower resolution version may be sufficient or may in any event exhaust the display capabilities of the viewer display device (e.g., monitor).

Thus, R. L. White and J. W. Percival, "Compression and Progressive Transmission of Astronomical Images," *SPIE Technical Conference* 2199, 1994, describes a progressive transmission technique based on bit planes that is effective for astronomical data.

However, utilizing progressive transmission barely begins to solve the "thinwire" problem. A viewer zooming or panning over a large image (e.g., map) desires realtime response. This of course is not achieved if the viewer must wait for display of the desired resolution of a new quadrant or view of the map each time a zoom and pan is initiated. Progressive transmission does not achieve this realtime response when it is the higher resolution versions of the image which are desired or needed, as these are transmitted later.

The problem could be effectively solved, if, in addition to variable resolution over time (i.e, progressive transmission), resolution is also varied over the physical extent of the image.

Specifically, using foveation techniques, high resolution data is transmitted at the user's gaze point but with lower resolution as one moves away from that point. The very simple rationale underlying these foveation techniques is that the human field of vision (centered at the gaze point) is limited. Most of the pixels rendered at uniform resolution are wasted for visualization purposes. In fact, it has been shown that the spatial resolution of the human eye decreases exponentially away from the center gaze point. E. L. Schwartz, "The Development of Specific Visual Projections in the Monkey and the Goldfish: Outline of a Geometric Theory of Receptotopic Structure," *Journal of Theoretical Biology,* 69:655–685, 1977

The key then is to mimic the movements and spatial resolution of the eye. If the user's gaze point can be tracked in realtime and a truly multi-foveated image transmitted (i.e., a variable resolution image mimicking the spatial resolution of the user's eye from the gaze point), all data necessary or useful to the user would be sent, and nothing more. In this way, the "thinwire" model is optimized, whatever the associated transmission capabilities and bandwidth limitations.

In practice, in part because eye tracking is imperfect, using multi-foveated images is superior to atempting display of an image portion of uniform resolution at the gaze point.

There have in fact been attempts to achieve multifoveated images in a "thinwire" environment.

F. S. Hill Jr., Sheldon Walker Jr. and Fuwen Gao, "Interactive Image Query System Using Progressive Transmission," *Computer Graphics,* 17(3), 1983, describes progressive transmission and a form of foveation for a browser of images in an archive. The realtime requirement does not appear to be a concern.

T. H. Reeves and J. A. Robinson, "Adaptive Foveation of MPEG Video," *Proceedings of the* 4th *ACM International Multimedia Conference,* 1996, gives a method to foveate MPEG-standard video in a thin-wire environment. MPEG-standard could provide a few levels of resolution but they consider only a 2-level foveation. The client/viewer can interactively specify the region of interest to the server/sender.

R. S. Wallace and P. W. Ong and B. B. Bederson and E. L. Schwartz, "Space-variant image processing". Intl. J. Of Computer Vision, 13:1 (1994) 71–90 discusses space-variant images in computer vision. "Space-Variant" may be regarded as synonymous with the term "multifoveated" used above. A biological motivation for such images is the complex logmap model of the transformation from the retina to the visual cortex (E. L. Schwartz, "A quantitative model of the functional architecture of human striate cortex with

application to visual illusion and cortical texture analysis", Biological Cybernetics, 37(1980) 63–76).

Philip Kortum and Wilson S. Geisler, "Implementation of a Foveated Image Coding System For Image Bandwidth Reduction," *Human Vision and Electronic Imaging, SPIE Proceedings Vol.* 2657, 350–360, 1996, implement a real time system for foveation-based visualization. They also noted the possibility of using foveated images to reduce bandwidth of transmission.

M. H. Gross, O. G. Staadt and R. Gatti, "Efficient triangular surface approximations using wavelets and quadtree data structures", IEEE Trans, On Visualization and Computer Graphics, 2(2), 1996, uses wavelets to produce multifoveated images.

Unfortunately, each of the above attempts are essentially based upon fixed super-pixel geometries, which amount to partitioning the visual field into regions of varying (predetermined) sizes called super-pixels, and assigning the average value of the color in the region to the super-pixel. The smaller pixels (higher resolution) are of course intended to be at the gaze point, with progressively larger super-pixels (lower resolution) about the gaze point.

However, effective real-time visulization over a "thin wire" requires precision and flexibility. This cannot be achieved with a geometry of predetermined pixel size. What is needed is a flexible foveation technique which allows one to modify the position and shape of the basic foveal regions, the maximum resolution at the foveal region and the rate at which the resolution falls away. This will allow the "thinwire" model to be optimized.

In addition, none of the above noted references addresses the issue of providing multifoveated images that can be dynamically (incrementally) updated as a function of user input. This property is crucial to the solution of the thinwire problem, since it is essential that information be "streamed" at a rate that optimally matches the bandwidth of the network with the human capacity to absorb the visual information.

SUMMARY OF THE INVENTION

The present invention overcomes the disadvantages of the prior art by utilizing means for tracking or approximating the user's gaze point in realtime and, based on the approximation, transmitting dynamic multifoveated image (s) (i.e., a variable resolution image over its physical extent mimicking the spatial resolution of the user's eye about the approximated gaze point) updated in realtime.

"Dynamic" means that the image resolution is also varying over time. The user interface component of the present invention may provide a variety of means for the user to direct this multifoveation process in real time.

Thus, the invention addresses the model which comprises an image server and a client viewer connected by a low bandwidth line. In effect, the invention reduces the bandwidth from server to client, in exchange for a very modest increase of bandwidth from the client to the server

Another object of the invention is that it allows realtime visualization of large scale images over a "thinwire" model of computation.

An additional advantage is the new degree of user control provided for realtime, active, visualization of images (mainly by way of foveation techniques). The invention allows the user to determine and change in realtime, via input means (for example, without limitation, a mouse pointer or eye tracking technology), the variable resolution over the space of the served up image(s).

An additional advantage is that the invention demonstrates a new standard of performance that can be achieved by large-scale image servers on the World Wide Web at current bandwidth or even in the near future.

Note also, the invention has advantages over the traditional notion of progressive transmission, which has no interactivity. Instead, the progressive transmission of an image has been traditionally predetermined when the image file is prepared. The invention's use of dynamic (constantly changing in realtime based on the user's input) multifoveated images allows the user to determine how the data are progressively transmitted.

Other advantages of the invention include that it allows the creation of the first dynamic and a more general class of multifoveated images. The present invention can use wavelet technology. The flexibility of the foveation approach based on wavelets allows one to easily modify the following parameters of a multifoveated image: the position and shape of the basic foveal region(s), the maximum resolution at the foveal region(s), and the rate at which the resolution falls away. Wavelets can be replaced by any multi resolution pyramid schemes. But it seems that wavelet-based approaches are preferred as they are more flexible and have the best compression properties.

Another advantage is the present invention's use of dynamic data structures and associated algorithms. This helps optimize the "effective real time behavior" of the system. The dynamic data structures allow the use of "partial information" effectively. Here information is partial in the sense that the resolution at each pixel is only partially known. But as additional information is streamed in, the partial information can be augmented. Of course, this principle is a corollary to progressive transmission.

Another advantage is that the dynamic data structures may be well exploited by the special architecture of the client program. For example, the client program may be multi-threaded with one thread (the "manager thread") designed to manage resources (especially bandwidth resources). This manager is able to assess network congestion, and other relevant parameters, and translate any literal user request into the appropriate level of demand for the network. For example, when the user's gaze point is focused on a region of an image, this may be translated into requesting a certain amount, say, X bytes of data. But the manager can reduce this to a request over the network of (say) X/2 bytes of data if the traffic is congested, or if the user is panning very quickly.

Another advantage of the present invention is that the server need send only that information which has not yet been served. This has the advantage of reducing communication traffic.

Further objects and advantages of the invention will become apparent from a consideration of the drawings and ensuing description.

BRIEF DESRIPTION OF DRAWINGS

FIG. **1** shows an embodiment of the present invention including a server, and client(s) as well as their respective components.

FIG. **2***a* illustrates one level of a particular wavelet transform, the Haar wavelet transform, which the server may execute in one embodiment of the present invention.

FIG. **2***b* illustrates one level of the Haar inverse wavelet transform.

FIG. **3** is a flowchart showing an algorithm the server may execute to perform a Haar wavelet transform in one embodiment of the present invention.

FIG. **4** shows Manager, Display and Network threads, which the client(s) may execute in one embodiment of the present invention.

FIG. **5** is a more detailed illustration of a portion of the Manager thread depicted in FIG. **4**.

DETAILED DESCRIPTION OF THE INVENTION

FIG. **1** depicts an overview of the components in an exemplary embodiment of the present invention. A server **1** is comprised of a storage device **3**, a memory device **7** and a computer processing device **4**. The storage device **3** can be implemented as, for example, an internal hard disk, Tape Cartridge, or CD-ROM. The faster access and greater storage capacity the storage device **3** provides, the more preferable the embodiment of the present invention. The memory device **7** can be implemented as, for example, a collection of RAM chips.

The processing device **4** on the server **1** has network protocol processing element **12** and wavelet transform element **13** running off it. The processing device **4** can be implemented with a single microprocessor chip (such as an Intel Pentium chip), printed circuit board, several boards or other device. Again, the faster the speed of the processing device **4**, the more preferable the embodiment. The network protocol processing element **12** can be implemented as a separate "software" (i.e., a program, sub-process) whose instructions are executed by the processing device **4**. Typical examples of such protocols include TCP/IP (the Internet Protocol) or UDP (User Datagram Protocol). The wavelet transform element **13** can also be implemented as separate "software" (i.e., a program, sub-process) whose instructions are executed by the processing device **4**.

In a preferred embodiment of the present invention, the server **1** is a standard workstation or Pentium class system. Also, TCP/IP processing may be used to implement the network protocol processing element **12** because it reduces complexity of implementation. Although a TCP/IP implementation is simplest, it is possible to use the UDP protocol subject to some basic design changes. The relative advantage of using TCP/IP as against UDP is to be determined empirically. An additional advantage of using modern, standard network protocols is that the server **1** can be constructed without knowing anything about the construction of its client(s) **2**.

According to the common design of modern computer systems, the most common embodiments of the present invention will also include an operating system running off the processing means device **4** of the server **1**. Examples of operating systems include, without limitation, Windows 95, Unix and Windows NT. However, there is no reason a processing device **4** could not provide the functions of an "operating system" itself.

The server **1** is connected to a client(s) **2** in a network. Typical examples of such servers **1** include image archive servers and map servers on the World Wide Web.

The client(s) **2** is comprised of a storage device **3**, memory device **7**, display **5**, user input device **6** and processing device **4**. The storage device **3** can be implemented as, for example, an internal hard disks, Tape Cartridge, or CD-ROM. The faster access and greater storage capacity the storage device **3** provides, the more preferable the embodiment of the present invention. The memory device **7** can be implemented as, for example, a collection of RAM chips. The display **5** can be implemented as, for example, any monitor, whether analog or digital. The user input device **6** can be implemented as, for example, a keyboard, mouse, scanner or eye-tracking device.

The client **2** also includes a processing device **4** with network protocol processing element **12** and inverse wavelet transform element means **14** running off it. The processing device **4** can be implemented as, for example, a single microprocessor chip (such as an Intel Pentium chip), printed circuit board, several boards or other device. Again, the faster the run time of the processing device **4**, the more preferable the embodiment. The network protocol processing element **12** again can be implemented as a separate "software" (i.e., a program, sub-process) whose instructions are executed by the processing device **4**. Again, TCP/IP processing may be used to implement the network protocol processing element **12**. The inverse wavelet transform element **14** also may be implemented as separate "software." Also running off the processing device **4** is a user input conversion mechanism **16**, which also can be implemented as "software."

As with the server **1**, according to the common design of modern computer systems, the most common embodiments of the present invention will also include an operating system running off the processing device **4** of the client(s) **2**.

In addition, if the server **1** is connected to the client(s) **2** via a telephone system line or other systems/lines not carrying digital pulses, the server **1** and client(s) **2** both also include a communications converter device **15**. A communications converter device **15** can be implemented as, for example, a modem. The communications converter device **15** converts digital pulses into the frequency/signals carried by the line and also converts the frequency/signals back into digital pulses, allowing digital communication.

In the operation of the present invention, the extent of computational resources (e.g., storage capacity, speed) is a more important consideration for the server **1**, which is generally shared by more than one client **2**, than for the client(s) **2**.

In typical practice of the present invention, the storage device **3** of the server **1** holds an image file, even a very large image file. A number of client **2** users will want to view the image.

Prior to any communication in this regard between the server **1** and client(s) **2**, the wavelet transform element **13** on the server **1** obtains a wavelet transform on the image and stores it in the storage device **3**.

There has been extensive research in the area of wavelet theory. However, briefly, to illustrate, "wavelets" are defined by a group of basis functions which, together with coefficients dependant on an input function, can be used to approximate that function over varying scales, as well as represent the function exactly in the limit. Accordingly, wavelet coefficients can be categorized as "average" or "approximating coefficients" (which approximate the function) and "difference coefficients" (which can be used to reconstruct the original function exactly). The particular approximation used as well as the scale of approximation depend upon the wavelet bases chosen. Once a group of basis functions is chosen, the process of obtaining the relevant wavelet coefficients is called a wavelet transform.

In the preferred embodiment, the Haar wavelet basis functions are used. Accordingly, in the preferred embodiment, the wavelet transform element **13** on the server **1** performs a Haar wavelet transform on a file representation of the image stored in the storage device **3**, and then stores the transform on the storage device **3**. However, it is readily apparent to anyone skilled in the art that any of the wavelet

family of transforms may be chosen to implement the present invention.

Note that once the wavelet transform is stored, the original image file need not be kept, as it can be reconstructed exactly from the transform.

FIG. **2** illustrates one step of the Haar wavelet transform. Start with an n by n matrix of coefficients **17** whose entries correspond to the numeric value of a color component (say, Red, Green or Blue) of a square screen image of n by n pixels. Divide the original matrix **17** into 2 by 2 blocks of four coefficients, and for each 2×2 block, label the coefficient in the first column, first row "a,"; second column, first row "b"; second row, first column "c"; and second row, second column "d."

Then one step of the Haar wavelet transform creates four n/2 by n/2 matrices. The first is an n/2 by n/2 approximation matrix **8** whose entries equal the "average" of the corresponding 2 by 2 block of four coefficients in the original matrix **17**. As is illustrated in FIG. **2**, the coefficient entries in the approximation matrix **8** are not necessarily equal to the average of the corresponding four coefficients a, b, c and d (i.e., a'=(a+b+c+d)/4) in the original matrix **17**. Instead, here, the "average" is defined as (a+b+c+d)/2.

The second is an n/2 by n/2 horizontal difference matrix **10** whose entries equal b'=(a+b−c−d)/2, where a, b, c and d are, respectively, the corresponding 2×2 block of four coefficients in the original matrix **17**. The third is an n/2 by n/2 vertical difference matrix **9** whose entries equal c'=(a−b+c−d)/2, where a, b, c and d are, respectively, the corresponding 2×2 block of four coefficients in the original matrix **17**. The fourth is an n/2 by n/2 diagonal difference matrix **11** whose entries equal d'=(a−b−c+d)/2, where a, b, c and d are, respectively, the corresponding 2×2 block of four coefficients in the original matrix **17**.

A few notes are worthy of consideration. First, the entries a', b', c', d' are the wavelet coefficients. The approximation matrix **8** is an approximation of the original matrix **17** (using the "average" of each 2×2 group of 4 pixels) and is one fourth the size of the original matrix **17**.

Second, each of the 2×2 blocks of four entries in the original matrix **17** has one corresponding entry in each of the four n/2 by n/2 matrices. Accordingly, it can readily be seen from FIG. **2** that each of the 2×2 blocks of four entries in the original matrix **17** can be reconstructed exactly, and the transformation is invertible. Therefore, the original matrix **17** representation of an image can be discarded during processing once the transform is obtained.

Third, the transform can be repeated, each time starting with the last approximation matrix **8** obtained, and then discarding that approximation matrix **8** (which can be reconstructed) once the next wavelet step is obtained. Each step of the transform results in approximation and difference matrices ½ the size of the approximation matrix **8** of the prior step.

Retracing each step to synthesize the original matrix **17** is called the inverse wavelet transform, one step of which is depicted in FIG. **2***b*.

Finally, it can readily be seen that the approximation matrix **8** at varying levels of the wavelet transform can be used as a representation of the relevant color component of the image at varying levels of resolution.

Conceptually then, the wavelet transform is a series of approximation and difference matrices at various levels (or resolutions). The number of coefficients stored in a wavelet transform is equal to the number of pixels in the original matrix **17** image representation. (However, the number of bits in all the coefficients may differ from the number of bits in the pixels. Applying data compression to coefficients turns out to be generally more effective on coefficients.) If we assume the image is very large, the transform matrices must be further decomposed into blocks when stored on the storage means **3**.

FIG. **3** is a flowchart showing one possible implementation of the wavelet transform element **13** which performs a wavelet transform on each color component of the original image. As can be seen from the flowchart, the transform is halted when the size of the approximation matrix is 256× 256, as this may be considered the lowest useful level of resolution.

Once the wavelet transform element **13** stores a transform of the image(s) in the storage means **3** of the server **1**, the server **1** is ready to communicate with client(s) **2**.

In typical practice of the invention the client **2** user initiates a session with an image server **1** and indicates an image the user wishes to view via user input means **6**. The client **2** initiates a request for the 256 by 256 approximation matrix **8** for each color component of the image and sends the request to the server **1** via network protocol processing element **12**. The server **1** receives and processes the request via network protocol processing element **12**. The server **1** sends the 256 by 256 approximation matrices **8** for each color component of the image, which the client **2** receives in similar fashion. The processing device **4** of the client **2** stores the matrices in the storage device **3** and causes a display of the 256 by 256 version of the image on the display **5**. It should be appreciated that the this low level of resolution requires little data and can be displayed quickly. In a map server application, the 256 by 256, coarse resolution version of the image may be useful in a navigation window of the display **5**, as it can provide the user with a position indicator with respect to the overall image.

A more detailed understanding of the operation of the client **2** will become apparent from the discussion of the further, continuous operation of the client **2** below.

Continuous operation of the client(s) **2** is depicted in FIG. **4**. In the preferred embodiment, the client(s) **2** processing device may be constructed using three "threads," the Manager thread **18**, the Network Thread **19** and the Display Thread **20**. Thread programming technology is a common feature of modern computers and is supported by a variety of platforms. Briefly, "threads" are processes that may share a common data space. In this way, the processing means can perform more than one task at a time. Thus, once a session is initiated, the Manager Thread **18**, Network Thread **19** and Display Thread **20** run simultaneously, independently and continually until the session is terminated. However, while "thread technology" is preferred, it is unnecessary to implement the client(s) **2** of the present invention.

The Display Thread **20** can be based on any modern windowing system running off the processing device **4**. One function of the Display Thread **20** is to continuously monitor user input device **6**. In the preferred embodiment, the user input device **6** consists of a mouse or an eye-tracking device, though there are other possible implementations. In a typical embodiment, as the user moves the mouse position, the current position of the mouse pointer on the display **5** determines the foveal region. In other words, it is presumed the user gaze point follows the mouse pointer, since it is the user that is directing the mouse pointer. Accordingly, the display thread **20** continuously monitors the position of the mouse pointer.

In one possible implementation, the Display Thread **20** places user input requests (i.e., foveal regions determined from user input device **6**) as they are obtained in a request queue. Queue's are data structures with first-in-first-out characteristics that are generally known in the art.

The Manager Thread **18** can be thought of as the brain of the client **2**. The Manager Thread **18** converts the user input request in the request queue into requests in the manager request queue, to be processed by the Network Thread **19**. The user input conversion mechanism **16** converts the user determined request into a request for coefficients.

A possible implementation of user input conversion mechanism **16** is depicted in the flow chart in FIG. **5**. Essentially, the user input conversion mechanism **16** requests all the coefficient entries corresponding to the foveal region in the horizontal difference **10** matrices, vertical difference **9** matrices, diagonal difference matrices **11** and approximation matrix **8** of the wavelet transform of the image at each level of resolution. (Recall that only the last level approximation matrix **8** needs to be stored by the server **1**.) That is, wavelet coefficients are requested such that it is possible to reconstruct the coefficients in the original matrix **17** corresponding to the foveal region.

As the coefficients are included in the request, they are masked out. The use of a mask is commonly understood in the art. The mask is maintained to determine which coefficients have been requested so they are not requested again. Each mask can be represented by an array of linked lists (one linked list for each row of the image at each level of resolution).

As shown in FIG. **5**, the input conversion mechanism **16** determines the current level of resolution ("L") of an image ("M_L") such that the image M_L is, e.g., 128×128 pixel matrix (for example, the lowest supported resolution), as shown in Step **200**. Then, the input conversion mechanism **16** determines if the current level L is the lowest resolution level (Step **210**). If so, it is determined if the three color coefficients (i.e., $M_L(R)$, $M_L(G)$, and $M_L(B)$) correspond to the foveal region that has been requested (Step **220**). If that is the case, then the input conversion mechanism **16** confirms that the current region L is indeed the lowest resolution region (Step **240**), and returns the control to the Manager Thread **18** (Step **250**). If, in Step **220**, it is determined that the three color coefficients have not been requested, these coefficients are requested using the mask described above, and the process continues to Step **240**, and the control is returned to the Manager Thread **18** (Step **250**).

If, in Step **210**, it is determined that the current level L is not the lowest resolution level, then the input conversion mechanism **16** determines whether the horizontal, vertical and diagonal difference coefficients (which are necessary to reconstruct the three color coefficients) have been requested (Step **260**). If so, then the input conversion mechanism **16** skips to Step **280** to decrease the current level L by 1. Otherwise a set of difference coefficients may be requested. This set depends on the mask and the foveal parameters (e.g., a shape of the foveal region, a maximum resolution, a rate of decay of the resolution, etc.). The user may select "formal" values for these foveal parameters, but the Manager Thread **18** may, at this point, select the "effective" values for these parameters to ensure a trade-off between (1) achieving a reasonable response time over the estimated current network bandwidth, and (2) achieving a maximum throughput in the transmission of data. The process then continues to Step **280**. Thereafter, the input conversion mechanism **16** determines whether the current level L is greater or equal to zero (Step **240**). If that is the case, the process loops back to step **260**. Otherwise, the control is returned to the Manager Thread **18** (Step **250**).

The Network Thread **19** includes the network protocol processing element **12**. The Network Thread obtains the (next) multi-resolution request for coefficients corresponding to the foveal region from request queue and processes and sends the request to the server **1** via network protocol processing element **12**.

Notice that the data requested is "local" because it represents visual information in the neighborhood of the indicated part of the image. The data is incremental because it represents only the additional information necessary to increase the resolution of the local visual information. (Information already available locally is masked out).

The server **1** receives and processes the request via network protocol processing element **12**, and sends the coefficients requested. When the coefficients are sent, they are masked out. The mask is maintained to determine which coefficients have been sent and for deciding which blocks of data can be released from main memory. Thus, an identical version of the mask is maintained on both the client **2** side and server **1** side.

The Network Thread **19** of the client **2** receives and processes the coefficients. The Network Thread **19** also includes inverse wavelet transform element **14**. The inverse wavelet transform element **14** performs an inverse wavelet transform on the received coefficients and stores the resulting portion of an approximation matrix **8** each time one is obtained (i.e., at each level of resolution) in the storage device **3** of the client **2**. The sub-image is stored at each (progressively higher, larger and less course) level of its resolution.

Note that as the client **2** knows nothing about the image until it is gradually filled in as coefficients are requested. Thus, sparse matrices (sparse, dynamic data structures) and associated algorithms can be used to store parts of the image received from the server **1**. Sparse matrices are known in the art and behave like normal matrices except that the memory space of the matrix are not allocated all at once. Instead the memory is allocated in blocks of sub-matrices. This is reasonable as the whole image may require a considerable amount of space.

Simultaneously, the Display thread **20** (which can be implemented using any modern operating system or windowing system) updates the display **5** based on the pyramid representation stored in the storage device **3**.

Of course, the Display thread **20** continues its monitoring of the user input device **6** and the whole of client **2** processing continues until the session is terminated.

A few points are worthy of mention. Notice that since lower, coarser resolution images will be stored on the client **2** first, they are displayed first Also, the use of foveated images ensures that the incremental data to update the view is small, and the requested data can arrive within the round trip time of a few messages using, for example, the TCP/IP protocol.

Also notice, that a wavelet coefficient at a relatively coarser level of resolution corresponding to the foveal region affects a proportionately larger part of the viewer's screen than a coefficient at a relatively finer level of resolution corresponding to the foveal region (in fact, the resolution on the display **5** exponentially away from the mouse pointer). Also notice the invention takes advantage of progressive transmission, which gives the image perceptual continuity. But unlike the traditional notion of progressive

transmission, it is the client **2** user that is determining transmission ordering, which is not pre-computed because the server **1** doesn't know what the client(s) **2** next request will be. Thus, as noted in the objects and advantages section, the "thinwire" model is optimized.

Note that in the event the thread technology is utilized to implement the present invention, semaphores data structures are useful if the threads share the same data structures (e.g., the request queue). Semaphores are well known in the art and ensure that only one simultaneous process (or "thread") can access and modify a shared data structure at one time. Semaphores are supported by modern operating systems.

CONCLUSION

It is apparent that various useful modifications can be made to the above description while remaining within the scope of the invention.

For example, without limitation, the user can be provided with two modes for display: to always fill the pixels to the highest resolution that is currently available locally or to fill them up to some user specified level. The client **2** display **5** may include a re-sizable viewing window with minimal penalty on the realtime performance of the system. This is not true of previous approaches. There also may be an auxiliary navigation window (which can be re-sized but is best kept fairly small because it displays the entire image at a low resolution). The main purpose of such a navigation window would be to let the viewer know the size and position of the viewing window in relation to the whole image.

It is readily seen that further modifications within the scope of the invention provide further advantages to the user. For example, without limitation, the invention may have the following capabilities: continuous realtime panning, continuous realtime zooming, foveating, varying the foveal resolution and modification of the shape and size of the foveal region. A variable resolution feature may also allow the server **1** to dynamically adjust the amount of transmitted data to match the effective bandwidth of the network.

While the above description contains many specificities, these should not be construed as limitations on the scope of the invention, but rather as an exemplification of one preferred embodiment thereof. Many other variations are possible. Accordingly, the scope of the invention should be determined not by the embodiment(s) illustrated, but by the appended claims and their legal equivalents.

What is claimed is:

1. A client apparatus for enabling a realtime visualization of at least one image, the client apparatus comprising:

a storage device storing first data corresponding to a multifoveated representation of an original image,

a user input device providing second data corresponding to at least one visualization command of at least one user; and

a processing arrangement generating third data corresponding to a multifoveated image using the first data, the second data and a foveation operator.

2. The client apparatus of claim **1**, further comprising a network protocol processing element which provides the third data using a TCP/IP protocol.

3. The client apparatus of claim **1**, wherein the processing element transmits the third data to the at least one client via the Internet.

4. The client apparatus of claim **1**, wherein the user input device includes a mouse device.

5. The client apparatus of claim **1**, wherein the user input device includes at least one of an eye-tracking device and a keyboard.

6. The client apparatus of claim **1**, wherein the foveation operator is specified using parameters that include at least one of:

a set of foveation points,

a shape of a foveated region,

a maximum resolution of the foveated region, and

a rate at which a maximum resolution of the foveal region decays.

7. The client apparatus of claim **1**,

wherein the processing arrangement receives the original image from a server, and

wherein the memory arrangement stores a data structure representing the multifoveated image, the data structure that is optimized for the client apparatus being independent of an image representation provided by a server.

8. The client apparatus of claim **1**, wherein the third data corresponding to the multifoveated image is generated for at least one of

a first arbitrary-shaped foveal region,

a second arbitrarily-fine foveal region, and

an arbitrary union of the first and second foveal regions.

* * * * *

US005179638A

United States Patent [19]

Dawson et al.

[11] Patent Number: 5,179,638

[45] Date of Patent: Jan. 12, 1993

[54] METHOD AND APPARATUS FOR GENERATING A TEXTURE MAPPED PERSPECTIVE VIEW

[75] Inventors: John F. Dawson; Thomas D. Snodgrass; James A. Cousens, all of Albuquerque, N. Mex.

[73] Assignee: Honeywell Inc., Minneapolis, Minn.

[21] Appl. No.: 514,598

[22] Filed: Apr. 26, 1990

[51] Int. Cl.[5] G06F 15/62

[52] U.S. Cl. 395/125; 395/127; 395/130

[58] Field of Search 395/125, 126, 127, 130; 364/443, 723; 340/729

[56] References Cited

U.S. PATENT DOCUMENTS

4,677,576	6/1987	Berlin, Jr. et al.	395/127 X
4,876,651	10/1989	Dawson et al.	395/126 X
4,884,220	11/1989	Dawson et al.	395/125
4,899,293	2/1990	Dawson et al.	395/125 X
4,940,972	7/1990	Mouchot et al.	395/125 X
4,985,854	1/1991	Wittenburg	395/126 X
5,020,014	5/1991	Miller et al.	364/723

Primary Examiner—Gary V. Harkcom
Assistant Examiner—Mark K. Zimmerman
Attorney, Agent, or Firm—Ronald E. Champion; George A. Leone, Sr.

[57] ABSTRACT

A method and apparatus for providing a texture mapped perspective view for digital map systems. The system includes apparatus for storing elevation data, apparatus for storing texture data, apparatus for scanning a projected view volume from the elevation data storing apparatus, apparatus for processing, apparatus for generating a plurality of planar polygons and apparatus for rendering images. The processing apparatus further includes apparatus for receiving the scanned projected view volume from the scanning apparatus, transforming the scanned projected view volume from object space to screen space, and computing surface normals at each vertex of each polygon so as to modulate texture space pixel intensity. The generating apparatus generates the plurality of planar polygons from the transformed vertices and supplies them to the rendering apparatus which then shades each of the planar polygons. In one alternate embodiment of the invention, the polygons are shaded by apparatus of the rendering apparatus assigning one color across the surface of each polygon. In yet another alternate embodiment of the invention, the rendering apparatus interpolates the intensities between the vertices of each polygon in a linear fashion as in Gouraud shading.

8 Claims, 7 Drawing Sheets

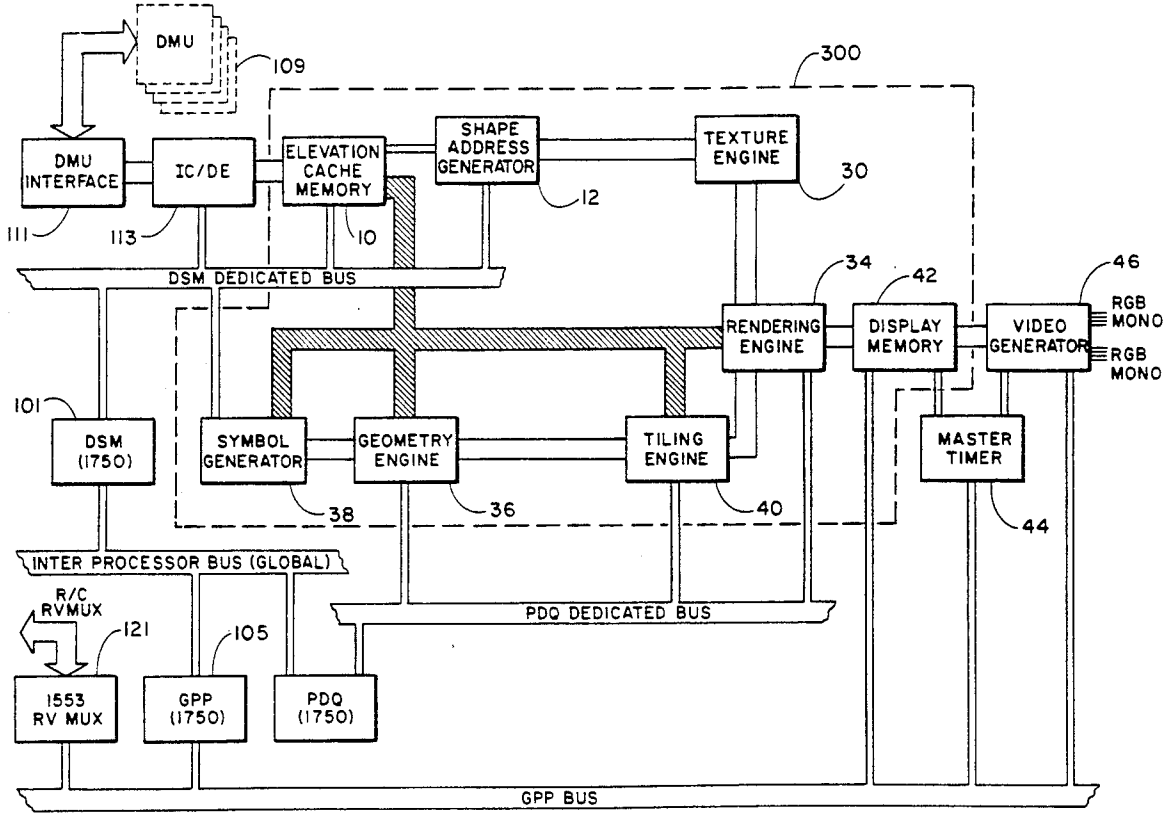


Fig.-1

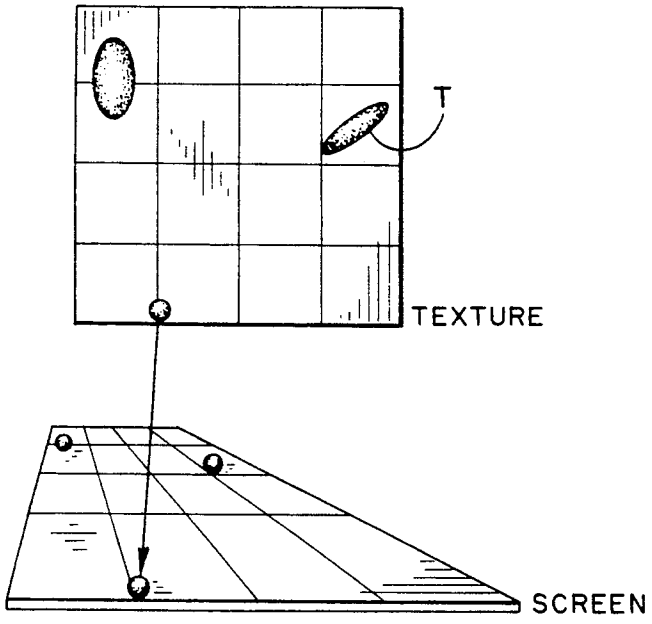

T
TEXTURE
SCREEN

Fig.-2

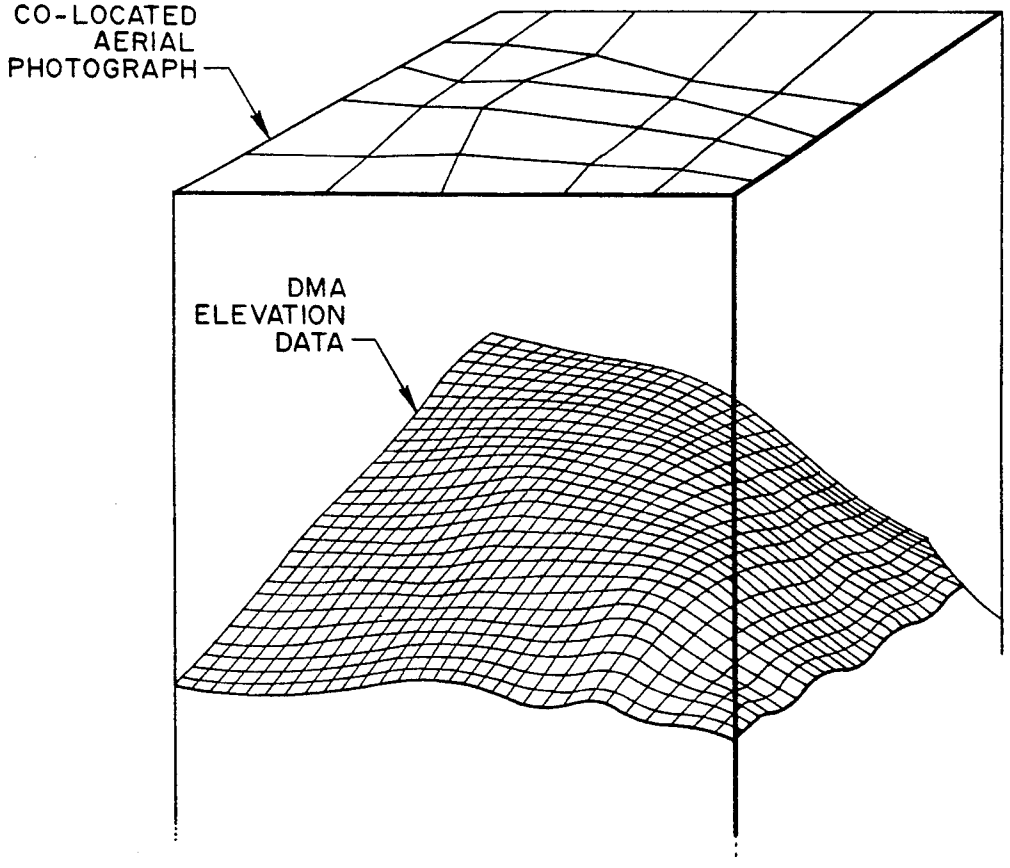

CO-LOCATED
AERIAL
PHOTOGRAPH
DMA
ELEVATION
DATA

Fig.-3

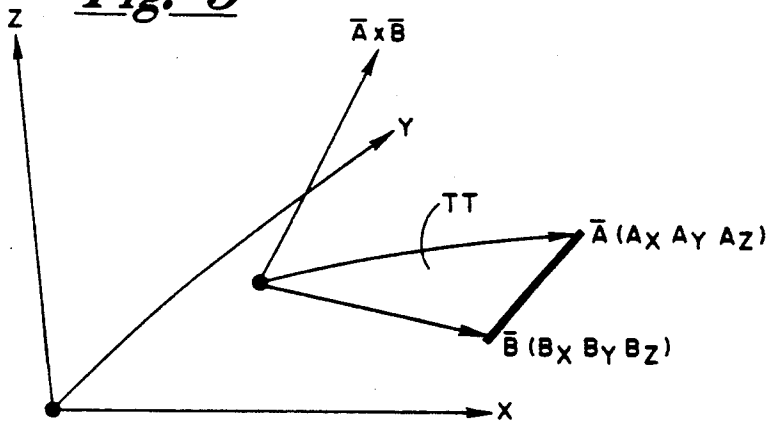
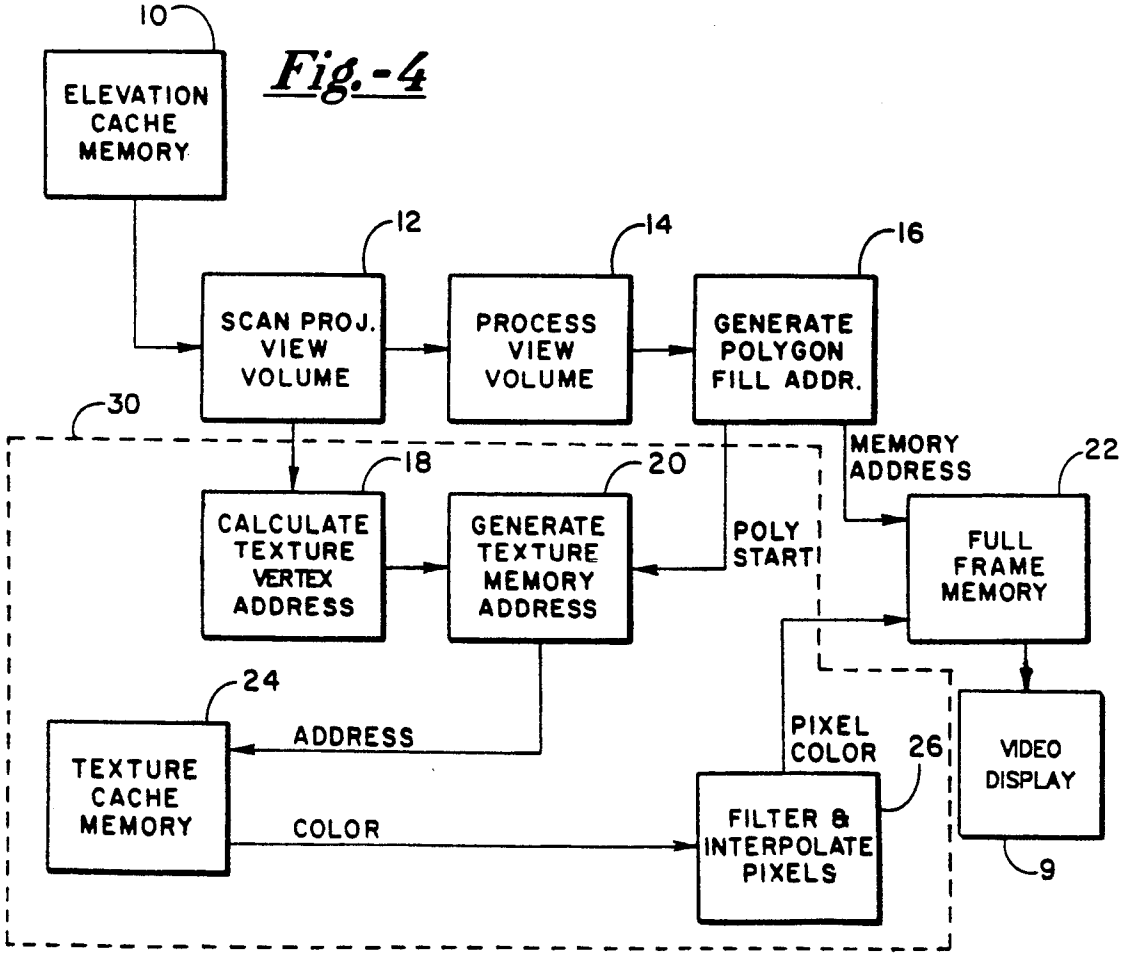

Z
$\bar{A} \times \bar{B}$
Y
TT
$\bar{A}$ (A_X A_Y A_Z)
$\bar{B}$ (B_X B_Y B_Z)
X

Fig.-4


10
ELEVATION CACHE MEMORY
12
SCAN PROJ. VIEW VOLUME
14
PROCESS VIEW VOLUME
16
GENERATE POLYGON FILL ADDR.
30
18
CALCULATE TEXTURE VERTEX ADDRESS
20
GENERATE TEXTURE MEMORY ADDRESS
POLY START
MEMORY ADDRESS
22
FULL FRAME MEMORY
24
TEXTURE CACHE MEMORY
ADDRESS
COLOR
PIXEL COLOR
26
FILTER & INTERPOLATE PIXELS
VIDEO DISPLAY
9

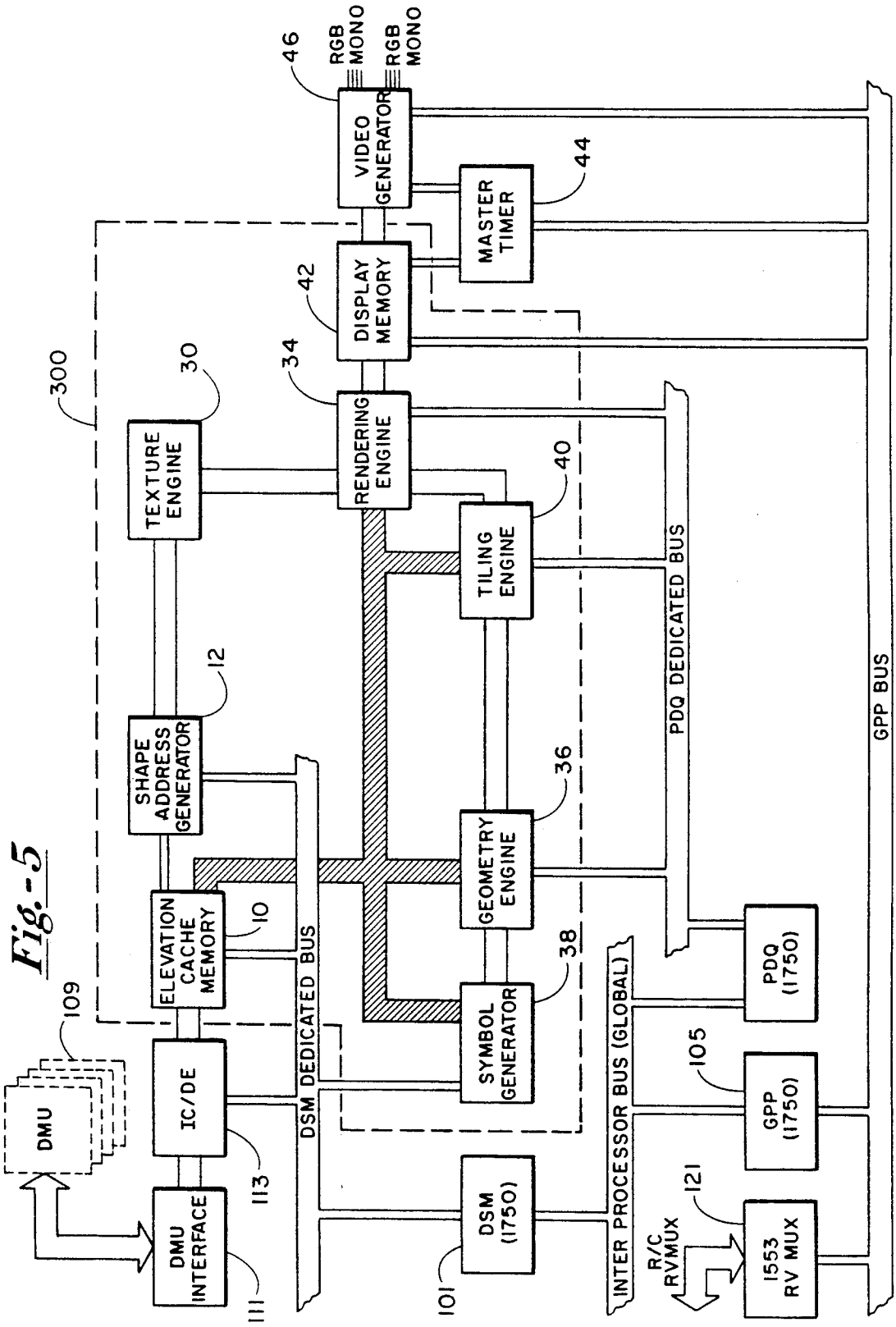

46
RGB MONO
RGB MONO
VIDEO GENERATOR
MASTER TIMER
44
DISPLAY MEMORY
42
34
RENDERING ENGINE
300
30
TEXTURE ENGINE
40
TILING ENGINE
PDQ DEDICATED BUS
GPP BUS
12
SHAPE ADDRESS GENERATOR
36
GEOMETRY ENGINE
ELEVATION CACHE MEMORY
10
38
SYMBOL GENERATOR
PDQ (1750)
109
DMU
IC/DE
DSM DEDICATED BUS
105
GPP (1750)
INTER PROCESSOR BUS (GLOBAL)
113
121
DSM (1750)
R/C RVMUX
1553 RV MUX
DMU INTERFACE
111
101

Fig.-5

Fig.-6

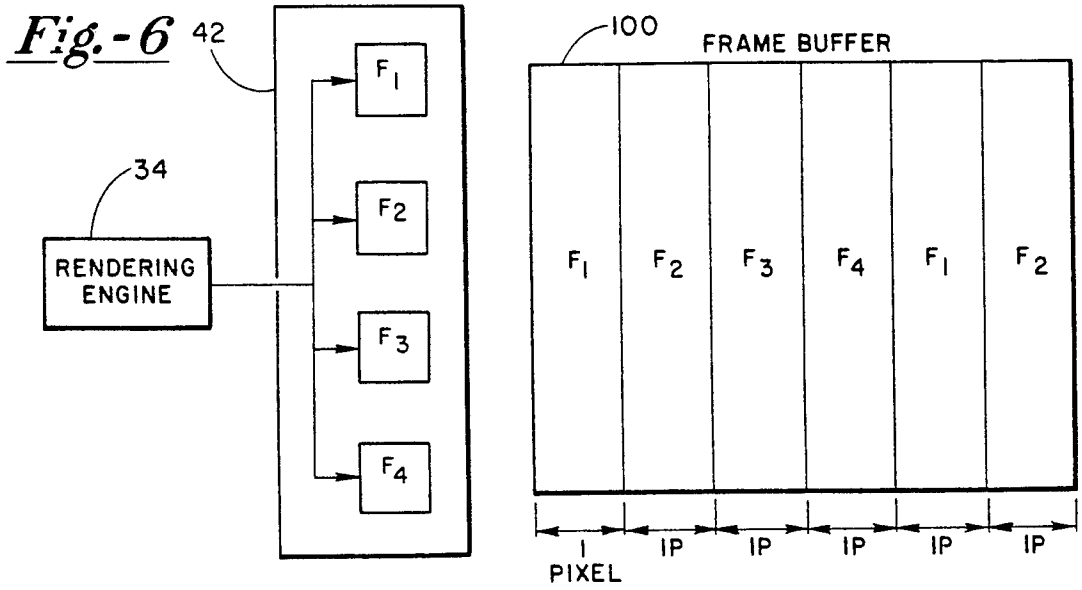


Fig.-7A

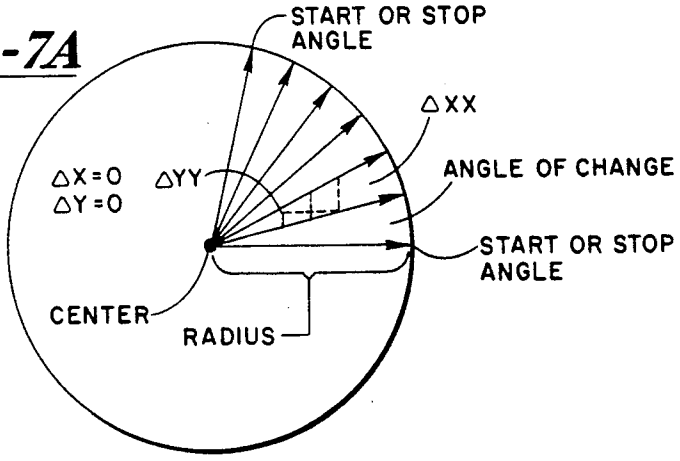


Fig.-7B

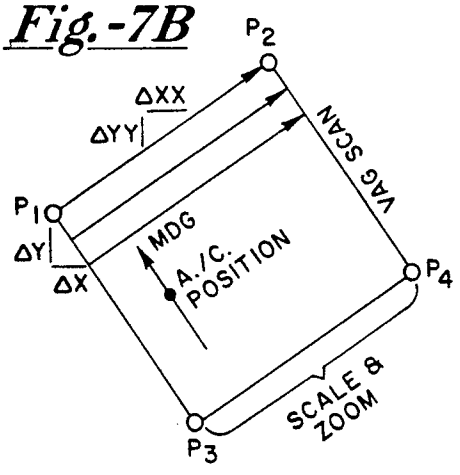


Fig.-7C

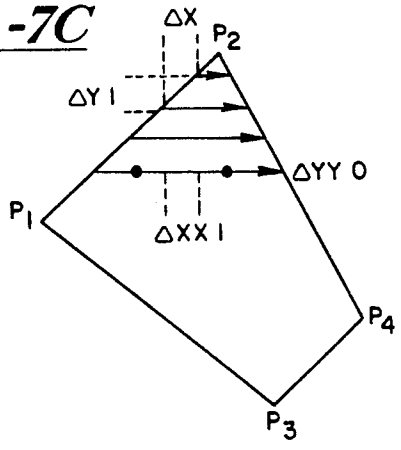
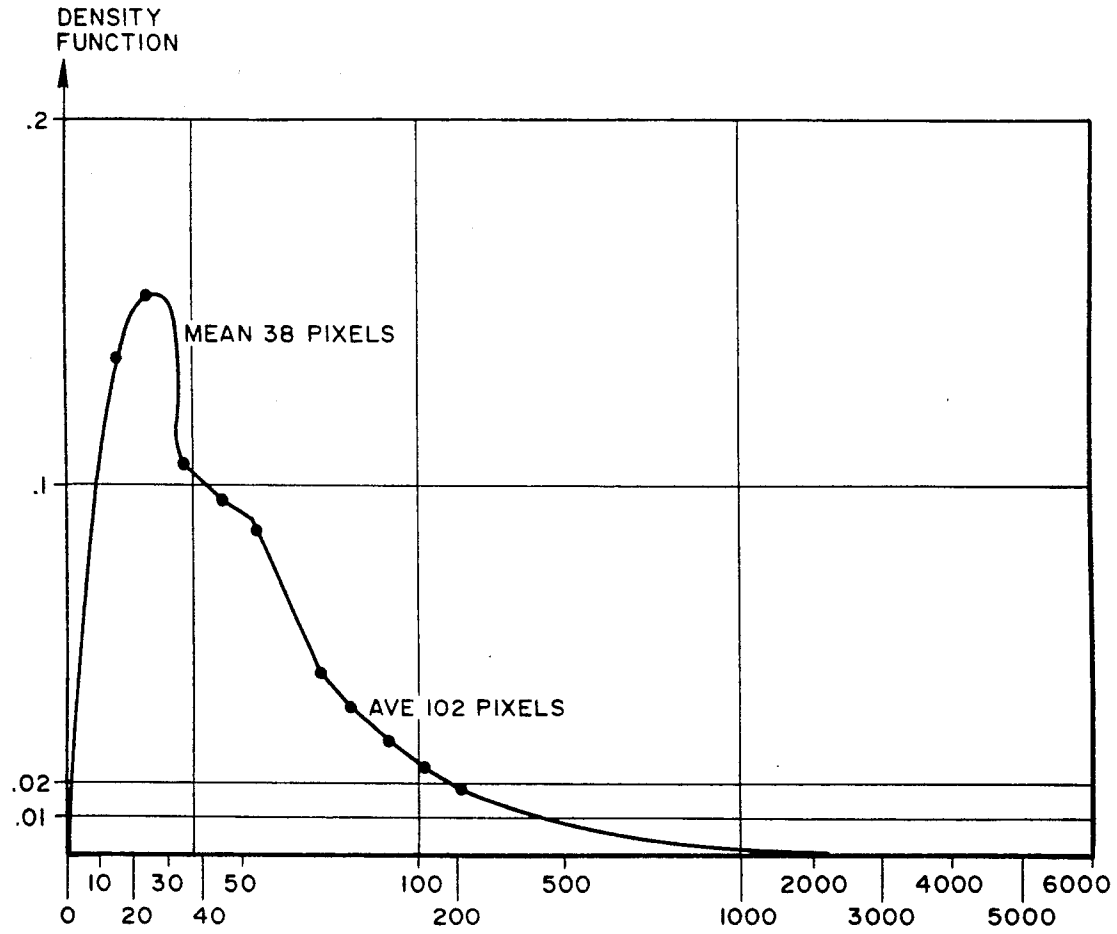


Fig.-8


DENSITY
FUNCTION
.2
.1
.02
.01
MEAN 38 PIXELS
AVE 102 PIXELS
0
10
20
30
40
50
100
200
500
1000
2000
3000
4000
5000
6000

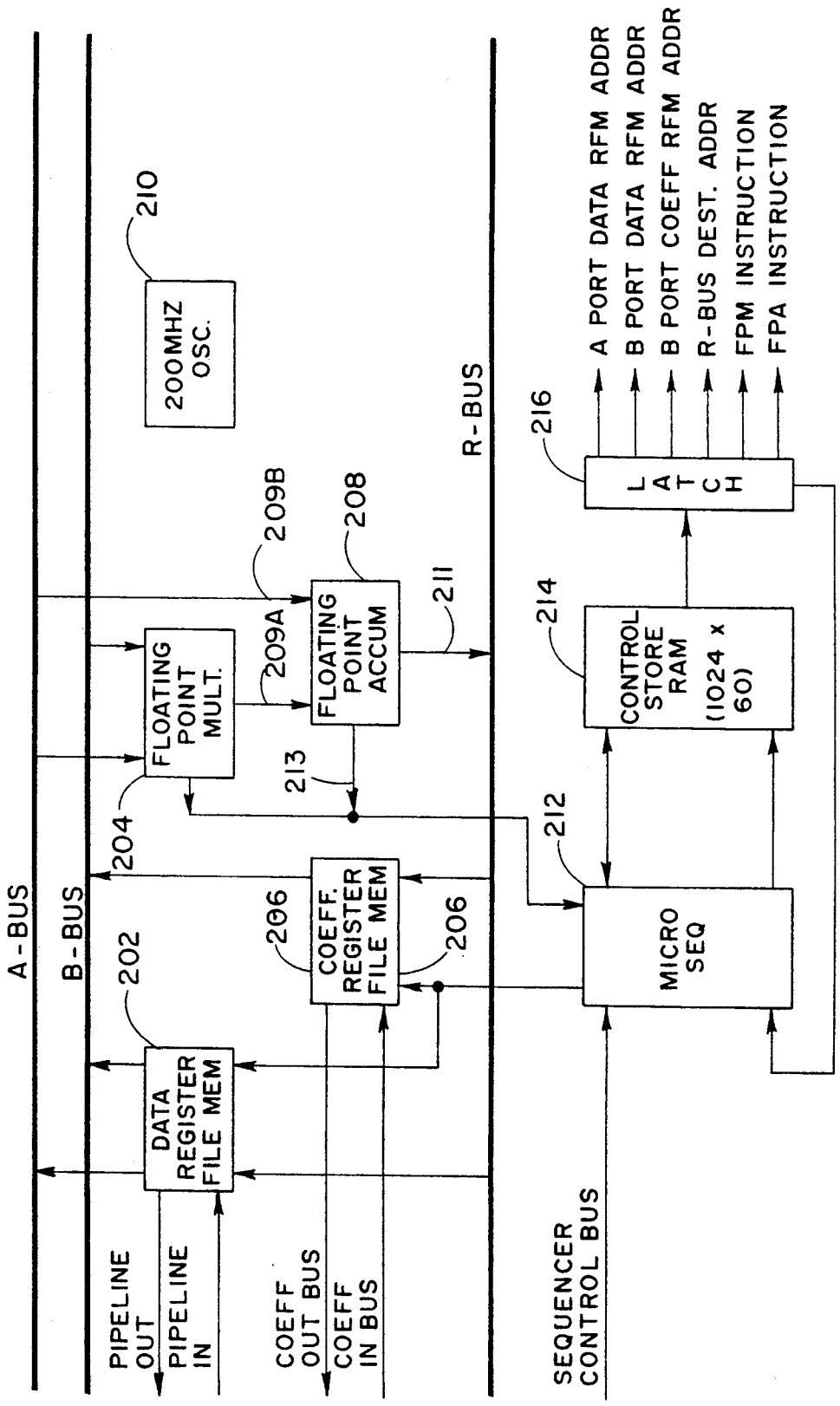

A-BUS
B-BUS
R-BUS
200MHZ OSC.
210
FLOATING POINT MULT.
209A
FLOATING POINT ACCUM
209B
208
211
213
204
206
COEFF. REGISTER FILE MEM
206
202
DATA REGISTER FILE MEM
PIPELINE OUT
PIPELINE IN
COEFF OUT BUS
COEFF IN BUS
SEQUENCER CONTROL BUS
212
MICRO SEQ
214
CONTROL STORE RAM (1024 x 60)
216
LATCH
A PORT DATA RFM ADDR
B PORT DATA RFM ADDR
B PORT COEFF RFM ADDR
R-BUS DEST. ADDR
FPM INSTRUCTION
FPA INSTRUCTION

Fig.-9

Fig.-10A

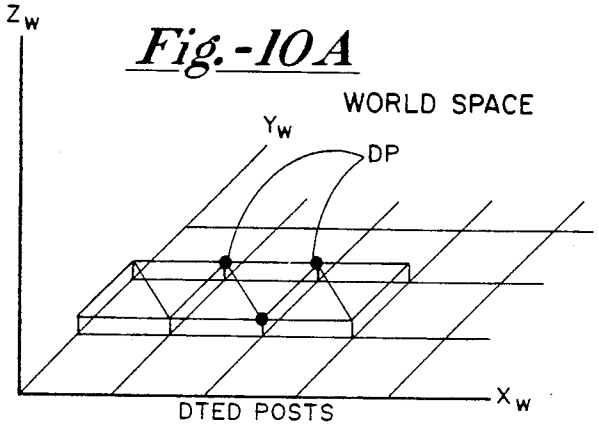


Fig.-10B

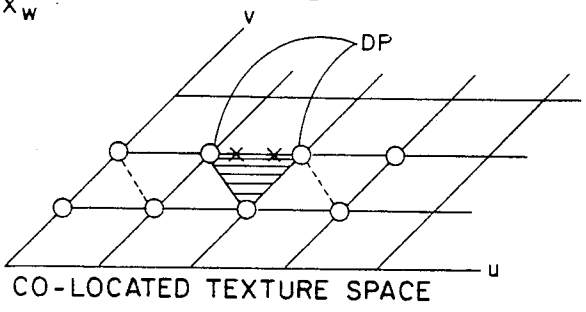


Fig.-10C

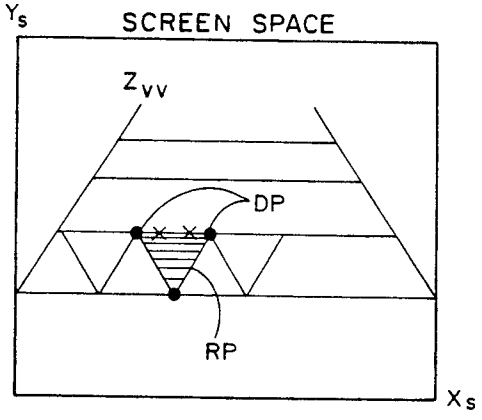
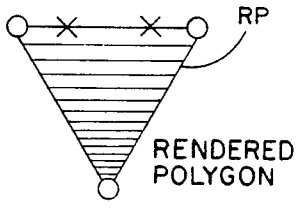


Fig.-10D



METHOD AND APPARATUS FOR GENERATING A TEXTURE MAPPED PERSPECTIVE VIEW

The present invention is directed generally to graphic display systems and, more particularly, to a method and apparatus for generating texture mapped perspective views for a digital map system.

RELATED APPLICATIONS

The following applications are included herein by reference:

(1) U.S. Pat. No. 4,876,651 filed May 11, 1988, issued Oct. 24, 1989 entitled "Digital Map System" which was assigned to the assignee of the present invention;

(2) Assignee copending application Ser. No. 09/514,685 filed Apr. 26, 1990, entitled "High Speed Processor for Digital Signal Processing";

(3) U.S. Pat. No. 4,884,220 entitled "Generator with Variable Scan Patterns" filed Jun. 7. 1988, issued Nov. 28, 1989, which is assigned to the assignee of the present invention;

(4) U.S. Pat. No. 4,899,293 entitled "A method of Storage and Retrieval of Digital Map Data Based Upon a Tessellated Geoid System", filed Dec. 14, 1988, issued Feb. 6, 1990;

(5) U.S. Pat. No. 5,020,014 entitled "Generic Interpolation Pipeline Processor", filed Feb. 7, 1989, issued May 28, 1991, which is assigned to the assignee of the present invention;

(6) Assignee's copending patent application Ser. No. 07/732,725 filed Jul. 18, 1991 entitled "Parallel Polygon/Pixel Rendering Engine Architecture for Computer Graphics" which is a continuation of patent application 07/419,722 filed Oct. 11, 1989 now abandoned;

(7) Assignee's copending patent application Ser. No. 07/514,724 filed Apr. 26, 1990 entitled "Polygon Tiling Engine";

(8) Assignee's copending patent application Ser. No. 07/514,723 filed Apr. 26, 1990 entitled "Polygon Sort Engine"; and

(9) Assignee's copending patent application Ser. No. 07/514,742 filed Apr. 26, 1990 entitled "Three Dimensional Computer Graphic Symbol Generator".

BACKGROUND OF THE INVENTION

Texture mapping is a computer graphics technique which comprises a process of overlaying aerial reconnaissance photographs onto computer generated three dimensional terrain images. It enhances the visual reality of raster scan images substantially while incurring a relatively small increase in computational expense. A frequent criticism of known computer-generated synthesized imagery has been directed to the extreme smoothness of the image. Prior art methods of generating images provide no texture, bumps, outcroppings, or natural abnormalities in the display of digital terrain elevation data (DTED).

In general, texture mapping maps a multidimensional image to a multidimensional space. A texture may be thought of in the usual sense such as sandpaper, a plowed field, a roadbed, a lake, woodgrain and so forth or as the pattern of pixels (picture elements) on a sheet of paper or photographic film. The pixels may be arranged in a regular pattern such as a checkerboard or may exhibit high frequencies as in a detailed photograph of high resolution LandSat imagery. Texture may also be three dimensional in nature as in marble or woodgrain surfaces. For the purposes of the invention, texture mapping is defined to be the mapping of a texture onto a surface in three dimensional object space. As is illustrated schematically in FIG. 1, a texture space object T is mapped to a display screen by means of a perspective transformation.

The implementation of the method of the invention comprises two processes. The first process is geometric warping and the second process is filtering. FIG. 2 illustrates graphically the geometric warping process of the invention for applying texture onto a surface. This process applies the texture onto an object to be mapped analogously to a rubber sheet being stretched over a surface. In a digital map system application, the texture typically comprises an aerial reconnaissance photograph and the object mapped is the surface of the digital terrain data base as shown in FIG. 2. After the geometric warping has been completed, the second process of filtering is performed. In the second process, the image is resampled on the screen grid.

The invention provides a texture mapped perspective view architecture which addresses the need for increased aircraft crew effectiveness, consequently reducing workload, in low altitude flight regimes characterized by the simultaneous requirement to avoid certain terrain and threats. The particular emphasis of the invention is to increase crew situational awareness. Crew situational awareness has been increased to some degree through the addition of a perspective view map display to a plan view capability which already exists in digital map systems. See, for example, assignee's copending application Ser. No. 07/192,798, for a DIGITAL MAP SYSTEM, filed May 11, 1988, issued Oct. 24, 1989 as U.S. Pat. No. 4,876,651 which is incorporated herein by reference in its entirety. The present invention improves the digital map system capability by providing a means for overlaying aerial reconnaissance photographs over the computer generated three dimensional terrain image resulting in a one-to-one correspondence from the digital map image to the real world. In this way the invention provides visually realistic cues which augment the informational display of such a computer generated terrain image. Using these cues an aircraft crew can rapidly make a correlation between the display and the real world.

The architectural challenge presented by texture mapping is that of distributing the processing load to achieve high data throughput using parallel pipelines and then recombining the parallel pixel flow into a single memory module known as a frame buffer. The resulting contention for access to the frame buffer reduces the effective throughput of the pipelines in addition to requiring increased hardware and board space to implement the additional pipelines. The method and apparatus of the invention addresses this challenge by effectively combining the low contention attributes of a single high speed pipeline with the increased processing throughput of parallel pipelines.

SUMMARY OF THE INVENTION

A method and apparatus for providing a texture mapped perspective view for digital map systems is provided. The invention comprises means for storing elevation data, means for storing texture data, means for scanning a projected view volume from the elevation data storing means, means for processing the projected view volume, means for generating a plurality of planar polygons and means for rendering images. The process-

ing means further includes means for receiving the scanned projected view volume from the scanning means, transforming the scanned projected view volume from object space to screen space, and computing surface normals at each vertex of each polygon so as to modulate texture space pixel intensity. The generating means generates the plurality of planar polygons from the transformed vertices and supplies them to the rendering means which then shades each of the planar polygons.

A primary object of the invention is to provide a technology capable of accomplishing a fully integrated digital map display system in an aircraft cockpit.

In one alternate embodiment of the invention, the polygons are shaded by means of the rendering means assigning one color across the surface of each polygon.

In yet another alternate embodiment of the invention, the rendering means interpolates the intensities between the vertices of each polygon in a linear fashion as in Gouraud shading.

It is yet another object of the invention to provide a digital map system including capabilities for perspective view, transparency, texture mapping, hidden line removal, and secondary visual effects such as depth cues and artifact (i.e., anti-aliasing) control.

It is yet another object of the invention to provide the capability for displaying forward looking infrared (FLIR) data and radar return images overlaid onto a plan and perspective view digital map image by fusing images through combining or subtracting other sensor video signals with the digital map terrain display.

It is yet another object of the invention to provide a digital map system with an arbitrary warping capability of one data base onto another data base which is accommodated by the perspective view texture mapping capability of the invention.

Other objects, features and advantages of the invention will become apparent to those skilled in the art through the drawings, description of the preferred embodiment and claims herein. In the drawings, like numerals refer to like elements.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** shows the mapping of a textured object to a display screen by a perspective transformation.

FIG. **2** illustrates graphically the geometric warping process of the invention for applying texture onto a surface.

FIG. **3** illustrates the surface normal calculation as employed by the invention.

FIG. **4** presents a functional block diagram of one embodiment of the invention.

FIG. **5** illustrates a top level block diagram of one embodiment of the texture mapped perspective view architecture of the invention.

FIG. **6** schematically illustrates the frame buffer configuration as employed by one embodiment of the invention.

FIGS. *7a*, *7b* and *7c* illustrate three examples of display format shapes.

FIG. **8** graphs the density function for maximum pixel counts.

FIG. **9** is a block diagram of one embodiment of the geometry array processor as employed by the invention.

FIGS. **10A**, **10B**, **10C** and **10D** illustrated the tagged architectural texture mapping as provided by the invention.

DESCRIPTION OF THE PREFERRED EMBODIMENT

Generally, perspective transformation from texture space having coordinates U, V to screen space having coordinates X, Y requires an intermediate transformation from texture space to object space having coordinates X_0, Y_0, Z_0. Perspective transformation is accomplished through the general perspective transform equation as follows:

$$[X\ Y\ Z\ H] = [X\ Y\ Z\ 1]\ X \begin{bmatrix} A & B & C & P \\ D & E & F & Q \\ G & H & I & R \\ L & M & N & S \end{bmatrix}$$

where a point (X,Y,Z) in 3-space is represented by a four dimensional position vector [X Y Z H] in homogeneous coordinates.

The 3×3 sub-matrix

$$\begin{bmatrix} A & B & C \\ D & E & F \\ G & H & I \end{bmatrix}$$

accomplishes scaling, shearing, and rotation.

The 1×3 row matrix [L M N] produces translation.

The 3×1 column matrix

$$\begin{bmatrix} P \\ Q \\ R \end{bmatrix}$$

produces perspective transformation.

The 1×1 scalar [S] produces overall scaling.

The Cartesian cross-product needed for surface normal requires a square root. As shown in FIG. **3**, the surface normal shown is a vector $A\times B$ perpendicular to the plane formed by edges of a polygon as represented by vectors A and B, where $A\times B$ is the Cartesian cross-product of the two vectors. Normalizing the vector allows calculation for sun angle shading in a perfectly diffusing Lambertian surface. This is accomplished by taking the vector dot product of the surface normal vector with the sun position vector. The resulting angle is inversely proportional to the intensity of the pixel of the surface regardless of the viewing angle. This intensity is used to modulate the texture hue and intensity value.

$$\frac{A\times B}{||A||\,||B||} \text{ where } \begin{matrix} A = Ax^2 + Ay^2 + Az^2 \\ B = Bx^2 + By^2 + Bz^2 \end{matrix}$$

A terrain triangle TT is formed by connecting the endpoints of vectors A and B, from point B_X, B_Y, B_Z to point A_X, A_Y, A_Z.

Having described some of the fundamental basis for the invention, a description of the method of the invention will now be set out in more detail below.

Referring now to FIG. **4**, a functional block diagram of one embodiment of the invention is shown. The invention functionally comprises a means for storing ele-

vation data **10**, a means for storing texture data **24**, a means for scanning a projected view volume from the elevation data storing means **12**, means for processing view volume **14** including means for receiving the scanned projected view volume from the scanning means **12**, means for generating polygon fill addresses **16**, means for calculating texture vertices addresses **18**, means for generating texture memory addresses **20**, means for filtering and interpolating pixels **26**, a full-frame memory **22**, and video display **9**. The processing means **14** further includes means for transforming the scanned projected view volume from object space to screen space and means for computing surface normals at each vertex of each polygon so as to calculate pixel intensity.

The means for storing elevation data **10** may preferably be a cache memory having at least a 50 nsec access time to achieve 20 Hz bi-linear interpolation of a 512×512 pixel resolution screen. The cache memory further may advantageously include a 256×256 bit buffer segment with 2K bytes of shadow RAM used for the display list. The cache memory may arbitrarily be reconfigured from 8 bits deep (data frame) to 64 bits (i.e., comprising the sum of texture map data (24 bits)+DTED (16 bits)+aeronautical chart data (24 bits)). A buffer segment may start at any cache address and may be written horizontally or vertically. Means for storing texture data **24** may advantageously be a texture cache memory which is identical to the elevation cache memory except that it stores pixel information for warping onto the elevation data cache. Referring now to FIG. **5**, a top level block diagram of the texture mapped perspective view architecture is shown. The architecture implements the functions as shown in FIG. **4** and the discussion which follows shall refer to functional blocks in FIG. **4** and corresponding elements in FIG. **5**. In some cases, such as element **14**, there is a one-to-one correspondence between the functional blocks in FIG. **4** and the architectural elements of FIG. **5**. In other cases, as explained hereinbelow, the functions depicted in FIG. **4** are carried out by a plurality of elements shown in FIG. **5**. The elements shown in FIG. **5** comprising the texture mapped perspective view system **300** of the invention include elevation cache memory **10**, shape address generator (SHAG) **12**, texture engine **30**, rendering engine **34**, geometry engine **36**, symbol generator **38**, tiling engine **40**, and display memory **42**. These elements are typically part of a larger digital map system including a digital map unit (DMU) **109**, DMU interface **111**, IC/DE **113**, a display stream manager (DSM) **101**, a general purpose processor (GPP) **105**, RV MUX **121**, PDQ **123**, master time **44**, video generator **46** and a plurality of data bases. The latter elements are described in assignee's Digital Map System U.S. Pat. No. 4,876,651.

GEOMETRY ENGINE

The geometry engine **36** is comprised of one or more geometry array processors (GAPs) which process the 4×4 Euler matrix transformation from object space (sometimes referred to as "world" space) to screen space. The GAPs generate X and Y values in screen coordinates and Zvv values in range depth. The GAPs also compute surface normals at each vertex of a polygon representing an image in object space via Cartesian cross-products for Gouraud shading, or they may assign one surface normal to the entire polygon for flat shading and wire mesh. Intensity calculations are performed using a vector dot product between the surface normal or normals and the illumination source to implement a Lambertian diffusely reflecting surface. Hue and intensity values are then assigned to the polygon. The method and apparatus of the invention also provides a dot rendering scheme wherein the GAPs only transform one vertex of each polygon and the tiling engine **40**, explained in more detail below, is inhibited. In this dot rendering format, hue and intensity are assigned based on the planar polygon containing the vertex and the rendering engine is inhibited. Dot polygons may appear in the same image as multiple vertex polygons or may comprise the entire image itself. The "dots" are passed through the polygon rendering engine **34**. A range to the vertices or polygon (Zvv) is used if a fog or "DaVinci" effect are invoked as explained below. The GAPs also transform three dimensional overlay symbols from world space to screen space.

Referring now to FIG. **9**, a block diagram of one example embodiment of a geometry array processor (GAP) is shown. The GAP comprises a data register file memory **202**, a floating point multiplier **204**, a coefficient register file memory **206**, a floating point accumulator **208**, a 200 MHz oscillator **210**, a microsequencer **212**, a control store RAM **214**, and latch **216**.

The register file memory may advantageously have a capacity of 512 by 32 bits. The floating point accumulator **208** includes two input ports **209A** and **209B** with independent enables, one output port **211**, and a condition code interface **212** responsive to error codes. The floating point accumulator operates on four instructions, namely, multiply, no-op, pass A, and pass B. The microsequencer **212** operates on seven instructions including loop on count, loop on condition, jump, continue, call, return and load counter. The microsequencer includes a debug interface having a read/write (R/W) internal register, R/W control store memory, halt on address, and single step, and further includes a processor interface including a signal interrupt, status register and control register. The GAP is fully explained in the assignee's co-pending application No. 07/514,685 filed Apr. 26, 1990 entitled High Speed Processor for Digital Signal Processing which is incorporated herein by reference in its entirety.

In one alternative embodiment of the invention, it is possible to give the viewer of the display the visual effect of an environment enshrouded in fog. The fog option is implemented by interpolating the color of the triangle vertices toward the fog color. As the triangles get smaller with distance, the fog particles become denser. By using the known relationship between distance and fog density, the fog thickness can be "dialed" or adjusted as needed. The vertex assignment interpolates the vertex color toward the fog color as a function of range toward the horizon. The fog technique may be implemented in the hardware version of the GAP such as may be embodied in a GaAs semiconductor chip. If a linear color space (typically referred to as "RGB" to reflect the primary colors, red, green and blue) is assumed, the amount of fog is added as a function of range to the polygon vertices' color computation by well known techniques. Thus, as the hue is assigned by elevation banding or monochrome default value, the fog color is tacked on. The rendering engine **34**, explained in more detail below, then straight forwardly interpolates the interior points.

In another alternative embodiment of the invention, a DaVinci effect is implemented. The DaVinci effect

causes the terrain to fade into the distance and blend with the horizon. It is implemented as a function of range of the polygon vertices by the GAP. The horizon color is added to the vertices similarly to the fog effect.

SHAPE ADDRESS GENERATOR (SHAG)

The SHAG **12** receives the orthographically projected view volume outline onto cache from the DSM. It calculates the individual line lengths of the scans and the delta x and delta y components. It also scans the elevation posts out of the elevation cache memory and passes them to the GAPs for transformation. In one embodiment of the invention, the SHAG preferably includes two arithmetic logic units (ALUs) to support the **50** nsec cache **10**. In the SHAG, data is generated for the GAPs and control signals are passed to the tiling engine **40**. DFAD data is downloaded into overlay RAM (not shown) and three dimensional symbols are passed to the GAPs from symbol generator **38**. Elevation color banding hue assignment is performed in this function. The SHAG generates shapes for plan view, perspective view, intervisibility, and radar simulation. These are illustrated in FIG. **7**. The SHAG is more fully explained in assignee's copending application, Ser. No. 203,660, Generator With Variable Scan Patterns, filed Jun. 7, 1988 issued as U.S. Pat. No. 4,884,220 on Nov. 28, 1989 which is incorporated herein by reference in its entirety.

A simple Lambertian lighting diffusion model has proved adequate for generating depth cueing in one embodiment of the invention. The sun angle position is completely programmable in azimuth and zenith. It may also be self-positioning based on time of day, time of year, latitude and longitude. A programmable intensity with gray scale instead of color implements the moon angle position algorithm. The display stream manager (DSM) programs the sun angle registers. The illumination intensities of the moon angle position may be varied with the lunar waxing and waning cycles.

TILING ENGINE AND TEXTURE ENGINE

Still referring to FIGS. **4** and **5**, the means for calculating texture vertex address **18** may include the tiling engine **40**. Elevation posts are vertices of planar triangles modeling the surface of the terrain. These posts are "tagged" with the corresponding U,V coordinate address calculated in texture space. This tagging eliminates the need for interpolation by substituting an address lookup. Referring to FIGS. **10A**, **10B**, **10C** and **10D**, with continuing reference to FIGS. **4** and **5**, the tagged architectural texture mapping as employed by the invention is illustrated. FIG. **10A** shows an example of DTED data posts, DP, in world space. FIG. **10B** shows the co-located texture space for the data posts. FIG. **10C** shows the data posts and rendered polygon in screen space. FIG. **10D** illustrates conceptually the interpolation of tagged addresses into a rendered polygon RP. The texture engine **30** performs the tagged data structure management and filtering processes. When the triangles are passed to the rendering engine by the tiling engine for filling with texture, the tagged texture address from the elevation post is used to generate the texture memory address. The texture value is filtered by filtering and interpolation means **26** before being written to full-frame memory **22** prior to display.

The tiling engine generates the planar polygons from the transformed vertices in screen coordinates and passes them to the rendering engine. For terrain polygons, a connectivity offset from one line scan to the next is used to configure the polygons. For overlay symbols, a connectivity list is resident in a buffer memory (not shown) and is utilized for polygon generation. The tiling engine also informs the GAP if it is busy. In one embodiment 512 vertices are resident in a 1K buffer.

All polygons having surface normals more than 90 degrees from LOS are eliminated from rendering. This is known in the art as backface removal. Such polygons do not have to be transformed since they will not be visible on the display screen. Additional connectivity information must be generated if the polygons are non-planar as the transformation process generates implied edges. This requires that the connectivity information be dynamically generated. Thus, only planar polygons with less than 513 vertices are implemented. Non-planar polygons and dynamic connectivity algorithms are not implemented by the tiling engine. The tiling engine is further detailed in assignee's copending applications of even filing date herewith entitled Polygon Tiling Engine, as referenced hereinabove and Polygon Sort Engine, as referenced hereinabove, both of which are incorporated herein by reference.

RENDERING ENGINE

Referring again to FIG. **5**, the rendering engine **34** of the invention provides a means of drawing polygons in a plurality of modes. The rendering engine features may include interpolation algorithms for processing coordinates and color, hidden surface removal, contour lines, aircraft relative color bands, flat shading, Gourand shading, phong shading, mesh format or screen door effects, ridgeline display, transverse slice, backface removal and RECE (aerial reconnaissance) photo modes. With most known methods of image synthesis, the image is generated by breaking the surfaces of the object into polygons, calculating the color and intensity at each vertex of the polygon, and drawing the results into a frame buffer while interpolating the colors across the polygon. The color information at the vertices is calculated from light source data, surface normal, elevation and/or cultural features.

The interpolation of coordinate and color (or intensity) across each polygon must be performed quickly and accurately. This is accomplished by interpolating the coordinate and color at each quantized point or pixel on the edges of the polygon and subsequently interpolating from edge to edge to generate the fill lines. For hidden surface removal, such as is provided by a Z-buffer in a well-known manner, the depth or Z-value for each pixel is also calculated. Furthermore, since color components can vary independently across a surface or set of surfaces, red, green and blue intensities are interpolated independently. Thus, a minimum of six different parameters (X,Y,Z,R,G,B) are independently calculated when rendering polygons with Gouraud shading and interpolated Z-values.

Additional features of the rendering engine include a means of providing contour lines and aircraft relative color bands. For these features the elevation also is interpolated at each pixel. Transparency features dictate that an alpha channel be maintained and similarly interpolated. These requirements imply two additional axes of interpolation bringing the total to eight. The rendering engine is capable of processing polygons of one vertex in its dot mode, two vertices in its line mode, and three to 512 coplanar vertices in its polygon mode.

In the flat shading mode the rendering engine assigns the polygon a single color across its entire surface. An arbitrary vertex is selected to assign both hue and intensity for the entire polygon. This is accomplished by assigning identical RGB values to all vertices. Interpolation is performed normally but results in a constant value. This approach will not speed up the rendering process but will perform the algorithm with no hardware impact.

The Gouraud shading algorithm included in the rendering engine interpolates the intensities between the vertices of each polygon rendered in a linear fashion. This is the default mode. The Phong shading algorithm interpolates the surface normals between the vertices of the polygon between applying the intensity calculations. The rendering engine would thus have to perform an illumination calculation at each pixel after interpolation. This approach would significantly impact the hardware design. This algorithm may be simulated, however, using a weighing function (typically a function of cosine (Θ)) around a narrow band of the intensities. This results in a non-linear interpolation scheme and provides for a simulated specular reflectance. In an alternative embodiment, the GAP may be used to assign the vertices of the polygon this non-linear weighing via the look-up table and the rendering engine would interpolate as in Gouraud shading.

Transparency is implemented in the classical sense using an alpha channel or may be simulated with a screen door effect. The screen door effect simply renders the transparent polygon as normal but then only outputs every other or every third pixel. The mesh format appears as a wire frame overlay with the option of rendering either hidden lines removed or not. In the case of a threat dome symbol, all polygon edges must be displayed as well as the background terrain. In such a case, the fill algorithm of the rendering engine is inhibited and only the polygon edges are rendered. The intensity interpolation is performed on the edges which may have to be two pixels wide to eliminate strobing. In one embodiment, an option for terrain mesh includes the capability for tagging edges for rendering so that the mesh appears as a regular orthogonal grid.

Typical of the heads up display (HUD) format used in aircraft is the ridgeline display and the transverse slice. In the ridgeline format, a line drawing is produced from polygon edges whose slopes change sign relative to the viewpoint. All polygons are transformed, tiled, and then the surface normals are computed and compared to the viewpoint. The tiling engine strips away the vertices of non-ridge contributing edges and passes only the ridge polygons to the rendering engine. In transverse slice mode, fixed range bins relative to the aircraft are defined. A plane orthogonal to the view LOS is then passed through for rendering. The ridges then appear to roll over the terrain as the aircraft flies along. These algorithms are similar to backface removal. They rely upon the polygon surface normal being passed to the tiling engine.

One current implementation of the invention guarantees non-intersecting polygon sides by restricting the polygons rendered to be planar. They may have up to 512 vertices. Polygons may also consist of one or two vertices. The polygon "end" bit is set at the last vertex and processed by the rendering engine. The polygon is tagged with a two bit rendering code to select mesh, transparent, or Gouraud shading. The rendering engine also accomplishes a fine clip to the screen for the polygon and implements a smoothing function for lines.

An optional aerial reconnaissance (RECE) photo mode causes the GAP to texture map an aerial reconnaissance photograph onto the DTED data base. In this mode the hue interpolation of the rendering engine is inhibited as each pixel of the warping is assigned a color from the RECE photo. The intensity component of the color is dithered in a well known manner as a function of the surface normal as well as the Z-depth. These pixels are then processed by the rendering engine for Z-buffer rectification so that other overlays such as threats may be accommodated. The RECE photos used in this mode have been previously warped onto a tessellated geoid data base and thus correspond pixel-for-pixel to the DTED data. See assignee's aforereferenced copending application for A Method of Storage and Retrieval of Digital Map Data Based Upon A Tessellated Geoid System, which is hereby incorporated by reference in its entirety. The photos may be denser than the terrain data. This implies a deeper cache memory to hold the RECE photos. Aeronautical chart warping mode is identical to RECE photos except that aeronautical charts are used in the second cache. DTED warping mode utilizes DTED data to elevation color band aeronautical charts.

The polygon rendering engine may preferably be implemented in a generic interpolation pipeline processor (GIPP) of the type as disclosed in assignee's aforereferenced patent entitled Generic Interpolation Pipeline Processor, which is incorporated herein by reference in its entirety. In one embodiment of the invention, the GIPPs fill in the transformed polygons using a bi-linear interpolation scheme with six axes (X,Y,Z,R,G,B). The primitive will interpolate a 16 bit pair and 8 bit pair of values simultaneously, thus requiring 3 chips for a polygon edge. One embodiment of the system of the invention has been sized to process one million pixels each frame time. This is sufficient to produce a 1K×1K high resolution chart, or a 512×512 DTED frame with an average of four overwrites per pixel during hidden surface removal with GIPPs outputting data at a 60 nsec rate, each FIFO, F1-F4, as shown in FIG. **6**, will receive data on the average of every 240 nsec. An even distribution can be assumed by decoding on the lower 2X address bits. Thus, the memory is divided into one pixel wide columns FIG. **6** is discussed in more detail below.

Referring again to FIGS. **4** and **5**, the "dots" are passed through the GIPPs without further processing. Thus, the end of each polygon's bit is set. A ZB buffer is needed to change the color of a dot at a given pixel for hidden dot removal. Perspective depth cuing is obtained as the dots get closer together as the range from the viewpoint increases.

Bi-linear interpolation mode operates in plan view on either DLMS or aeronautical charts. It achieves 20 Hz interpolation on a 512×512 display. The GIPPs perform the interpolation function.

DATA BASES

A Level I DTED data base is included in one embodiment of the invention and is advantageously sampled on three arc second intervals. Buffer segments are preferably stored at the highest scales (104.24 nm) and the densest data (13.03 nm). With such a scheme, all other scales can be created. A Level II DTED data base is also included and is sampled at one arc second inter-

vals. Buffer segments are preferably stored only at the densest data (5.21 nm).

A DFAD cultural feature data base is stored in a display list of 2K words for each buffer segment. The data structure consists of an icon font call, a location in cache, and transformation coefficients from model space to world space consisting of scaling, rotation, and position (translation). A second data structure comprises a list of polygon vertices in world coordinates and a color or texture. The DFAD data may also be rasterized and overlaid on a terrain similar to aerial reconnaissance photos.

Aeronautical charts at the various scales are warped into the tessellated geoid. This data is 24 bits deep. Pixel data such as LandSat, FLIR, data frames and other scanned in source data may range from one bit up to 24 bits in powers of two (1,2,4,8,16,24).

FRAME BUFFER CONFIGURATION

Referring again to FIG. **6**, the frame buffer configuration of one embodiment of the invention is shown schematically. The frame buffer configuration is implemented by one embodiment of the invention comprises a polygon rendering chip **34** which supplies data to full-frame memory **42**. The full-frame memory **42** advantageously includes first-in, first-out buffers (FIFO) F_1, F_2, F_3 and F_4. As indicated above with respect to the discussion of the rendering engine, the memory is divided up into one pixel wide columns as shown in FIG. **6**. By doing so, however, chip select must changed on every pixel when the master timer **44** shown in FIG. **5** reads the memory. However, by orienting the SHAG scan lines at 90 degrees to the master timer scan lines, the chip select will change on every line. The SHAG starts scanning at the bottom left corner of the display and proceeds to the upper left corner of the display.

With the image broken up in this way, the probability that the GIPP will write to the same FIFO two times in a row, three times, four, and so on can be calculated to determine how deep the FIFO must be. Decoding on the lower order address bits means that the only time the rendering engine will write to the same FIFO twice in a row is when a new scan line is started. At four deep as shown in the frame buffer graph **100**, the chances of the FIFO filling up are approximately one in 6.4K. With an image of 1 million pixels, this will occur an acceptably small number of times for most applications. The perspective view transformations for 10,000 polygons with the power and board area constraints that are imposed by an avionics environment is significant. The data throughput for a given scene complexity can be achieved by adding more pipeline in parallel to the architecture. It is desirable to have as few pipelines as possible, preferably one, so that the image reconstruction at the end of the pipeline does not suffer from an arbitration bottleneck for a Z-buffered display memory.

In one embodiment of the invention, the processing throughput required has been achieved through the use of GaAs VSLI technology for parallel pipelines and a parallel frame buffer design has eliminated contention bottlenecks. A modular architecture allows for additional functions to be added to further the integration of the digital map into the avionics suite. The system architecture of the invention has high flexibility while maintaining speed and data throughput. The polygonal data base structure approach accommodate arbitrary scene complexity and a diversity of data base types.

The data structure of the invention is tagged so that any polygon may be rendered via any of the implemented schemes in a single frame. Thus, a particular image may have Gouraud shaded terrain, transparent threat domes, flat shaded cultural features, lines, and dots. In addition, since each polygon is tagged, a single icon can be comprised of differently shaded polygons. The invention embodies a 24 bit color system, although a production map would be scaled to 12 bits. A 12 bit system provides 4K colors and would require a 32K by 8 RGB RAM look-up table (LUT).

MISCELLANEOUS FEATURES

The display formats in one example of the invention are switchable at less than 600 milliseconds between paper chart, DLMS plan and perspective view. A large cache (1 megabit D-RAMs) is required for texture mapping. Other format displays warp chart data over DTED, or use DTED to pseudo-color the map. For example, change the color palate LUT for transparency. The GAP is used for creating a true orthographic projection of the chart data.

An edit mode for three dimensions is supported by the apparatus of the invention. A three dimensional object such as a "pathway in the sky" may be tagged for editing. This is accomplished by first, moving in two dimensions at a given AGL, secondly, updating the AGL in the three dimensional view, and finally, updating the data base.

The overlay memory from the DMC may be video mixed with the perspective view display memory.

Freeze frame capability is supported by the invention. In this mode, the aircraft position is updated using the cursor. If the aircraft flies off the screen, the display will snap back in at the appropriate place. This capability is implemented in plan view only. There is data frame software included to enable roaming through cache memory. This feature requires a two axis roam joystick or similar control. Resolution of the Z-buffer is 16 bits. This allows 64K meters down range.

The computer generated imagery has an update rate of 20 Hz. The major cycle is programmable and variable with no frame extend invoked. The system will run as fast as it can but will not switch ping-pong display memories until each functional unit issues a "pipeline empty" message to the display memory. The major cycle may also be locked to a fixed frame in multiples of 16.6 milliseconds. In the variable frame mode, the processor clock is used for a smooth frame interpolation for roam or zoom. The frame extend of the DMC is eliminated in perspective view mode. Plan view is implemented in the same pipeline as the perspective view. The GPP **105** loads the countdown register on the master timer to control the update rate.

The slowest update rate is 8.57 Hz. The image must be generated in this time or the memories will switch. This implies a pipeline speed of 40 million pixels per second. In a 512×512 image, it is estimated that there would be 4 million pixels rendered worst case with heavy hidden surface removal. In most cases, only million pixels need be rendered. FIG. **8** illustrates the analysis of pixel over-writes. The minimum requirement for surface normal resolution so that the best image is achieved is 16 bits. Tied to this is the way in which the normal is calculated. Averaging from surrounding tiles gives a smoother image on scale change or zoom. Using one tile is less complex, but results in poorer image

quality. Surface normal is calculated on the fly in accordance with known techniques.

DISPLAY MEMORY

This memory is a combination of scene and overlay with a Z-buffer. It is distributed or partitioned for optimal loading during write, and configured as a frame buffer during read-out. The master time speed required is approximately 50 MHz. The display memory resolution can be configured as $512 \times 512 \times 12$ or as $1024 \times 1024 \times 12$. The Z-buffer is 16 bits deep and $1K \times 1K$ resolution. At the start of each major cycle, the Z-values are set to plus infinity (FF Hex). Infinity (Zmax) is programmable. The back clipping plane is set by the DSM over the control bus.

At the start of each major cycle, the display memory is set to a background color. In certain modes such as mesh or dot, this color will change. A background color register is loaded by the DSM over the configuration bus and used to fill in the memory.

VIDEO GENERATOR/MASTER TIMER

The video generator **46** performs the digital to analog conversion of the image data in the display memory to send to the display head. It combines the data stream from the overlay memory of the DMC with the display memory from the perspective view. The configuration bus loads the color map.

A 30 Hz interlaced refresh rate may be implemented in a system employing the present invention. Color pallets are loadable by the GPP. The invention assumes a linear color space in RGB. All colors at zero intensity go to black.

THREE DIMENSIONAL SYMBOL GENERATOR

The three-dimensional symbol generator **38** performs the following tasks:

1. It places the model to world transformation coefficients in the GAP.

2. It operates in cooperation with the geometry engine to multiply the world to screen transformation matrix by the model to world transformation matrix to form a model to screen transformation matrix. This matrix is stored over the model to world transformation matrix.

3. It operates in cooperation with the model to screen transformation matrix to each point of the symbol from the vertex list to transform the generic icon to the particular symbol.

4. It processes the connectivity list in the tiling engine and forms the screen polygons and passes them to the rendering engine.

One example of a three-dimensional symbol generator is described in detail in the assignee's aforereferenced patent application entitled "Three Dimensional Computer Graphic Symbol Generator".

The symbol generator data base consists of vertex list library and 64K bytes of overlay RAM and a connectivity list. Up to 18K bytes of DFAD (i.e., 2K bytes display list from cache shadow RAM $\times 9$ buffer segments) are loaded into the overlay RAM for cultural feature processing. The rest of the memory holds the threat/intelligence file and the mission planning file for the entire gaming area. The overlay RAM is loaded over the control bus from the DSM processor with the threat and mission planning files. The SHAG loads the DFAD files. The symbol libraries are updated via the configuration bus.

The vertex list contains the relative vertex positions of the generic library icons. In addition, it contains a 16 bit surface normal, a one bit end of polygon flag, and a one bit end of symbol flag. The table is $32K \times 16$ bits. A maximum of 512 vertices may be associated with any given icon. The connectivity list contains the connectivity information of the vertices of the symbol. A 64K by 12 bit table holds this information.

A pathway in the sky format may be implemented in this system. It consists of either a wire frame tunnel or an elevated roadbed for flight path purposes. The wire frame tunnel is a series of connected transparent rectangles generated by the tiling engine of which only the edges are visible (wire mesh). Alternatively, the polygons may be precomputed in world coordinates and stored in a mission planning file. The roadbed is similarly comprised of polygons generated by the tiler along a designated pathway. In either case, the geometry engine must transform these polygons from object space (world coordinate system) to screen space. The transformed vertices are then passed to the rendering engine. The parameters (height, width, frequency) of the tunnel and roadbed polygons are programmable.

Another symbol used in the system is a waypoint flag. Waypoint flags are markers consisting of a transparent or opaque triangle on a vertical staff rendered in perspective. The waypoint flag icon is generated by the symbol generator as a macro from a mission planning file. Alternatively, they may be precomputed as polygons and stored. The geometry engine receives the vertices from the symbol generator and performs the perspective transformation on them. The geometry engine passes the rendering engine the polygons of the flag staff and the scaled font call of the alphanumeric symbol. Plan view format consists of a circle with a number inside and is not passed through the geometry engine.

DFAD data processing consists of a generalized polygon renderer which maps 32K points possible down to 256 polygons or less for a given buffer segment. These polygons are then passed to the rendering engine. This approach may redundantly render terrain and DFAD for the same pixels but easily accommodates declutter of individual features. Another approach is to rasterize the DFAD and use a texture warp function to color the terrain. This would not permit declutter of individual features but only classes (by color). Terrain color show-through in sparse overlay areas would be handled by a transparent color code (screen door effect). No verticality is achieved.

There are 298 categories of aerial, linear, and point features. Linear features must be expanded to a double line to prevent interlace strobing. A point feature contains a length, width, and height which can be used by the symbol generator for expansion. A typical lake contains 900 vertices and produces 10 to 20 active edges for rendering at any given scan line. The number of vertices is limited to 512. The display list is 64K bytes for a 1:250K buffer segment. Any given feature could have 32K vertices.

Up to 2K bytes of display list per buffer segment DTED is accommodated for DFAD. The DSM can tag the classes or individual features for clutter/declutter by toggling bits in the overlay RAM of the SHAG.

The symbol generator processes macros and graphic primitives which are passed to the rendering engine. These primitives include lines, arcs, alphanumerics, and two dimensional symbology. The rendering engine

draws these primitives and outputs pixels which are anti-aliased. The GAP transforms these polygons and passes them to the rendering engine. A complete 4×4 Euler transformation is performed. Typical macros include compass rose and range scale symbols. Given a macro command, the symbol generator produces the primitive graphics calls to the rendering engine. This mode operates in plan view only and implements two dimensional symbols. Those skilled in the art will appreciate that the invention is not limited to specific fonts.

Three dimensional symbology presents the problem of clipping to the view volume. A gross clip is handled by the DSM in the cache memory at scan out time. The base of a threat dome, for example, may lie outside the orthographic projection of the view volume onto cache, yet a part of its dome may end up visible on the screen. The classical implementation performs the functions of tiling, transforming, clipping to the view volume (which generates new polygons), and then rendering. A gross clip boundary is implemented in cache around the view volume projection to guarantee inclusion of the entire symbol. The anomaly under animation to be avoided is that of having symbology sporadically appear and disappear in and out of the frame at the frame boundaries. A fine clip to the screen is performed downstream by the rendering engine. There is a 4K boundary around the screen which is rendered. Outside of this boundary, the symbol will not be rendered. This causes extra rendering which is clipped away.

Threat domes are represented graphically in one embodiment by an inverted conic volume. A threat/intelligence file contains the location and scaling factors for the generic model to be transformed to the specific threats. The tiling engine contains the connectivity information between the vertices and generates the planar polygons. The threat polygons are passed to the rendering engine with various viewing parameters such as mesh, opaque, dot, transparent, and so forth.

Graticles represent latitude and longitude lines, UTM klicks, and so forth which are warped onto the map in perspective. The symbol generator produces these lines.

Freeze frame is implemented in plan view only. The cursor is flown around the screen, and is generated by the symbol generator.

Programmable blink capability is accommodated in the invention. The DSM updates the overlay RAM toggle for display. The processor clock is used during variable frame update rate to control the blink rate.

A generic threat symbol is modeled and stored in the three dimensional symbol generation library. Parameters such as position, threat range, and angular threat view are passed to the symbol generator as a macro call (similar to a compass rose). The symbol generator creates a polygon list for each threat instance by using the parameters to modify the generic model and place it in the world coordinate system of the terrain data base. The polygons are transformed and rendered into screen space by the perspective view pipeline. These polygons form only the outside envelope of the threat cone.

This invention has been described herein in considerable detail in order to comply with the Patent Statues and to provide those skilled in the art with the information needed to apply the novel principles and to construct and use such specialized components as are required. However, it is to be understood that the invention can be carried out by specifically different equipment and devices, and that various modifications, both as to the equipment details and operating procedures, can be accomplished without departing from the scope of the invention itself.

What is claimed is:

1. A system for providing a texture mapped perspective view for a digital map system wherein objects are transformed from texture space having U, V coordinates to screen space having X, Y coordinates comprising:

(a) a cache memory means for storing terrain data including elevation posts, wherein the cache memory means includes an output and an address bus;

(b) a shape address generator means for scanning cache memory having an ADDRESS SIGNAL coupled to the cache memory means address bus wherein the shape address generator means scans the elevation posts out of the cache memory means;

(c) a geometry engine coupled to the cache memory means output to receive the elevation posts scanned from the cache memory by the shape address generator means, the geometry engine including means for

i. transformation of the scanned elevation posts from object space to screen space so as to generate transformed vertices in screen coordinates for each elevation post, and

ii. generating three dimensional coordinates;

(d) a tilling engine coupled to the geometry engine for generating planar polygons from the generated three dimensional coordinates;

(e) a symbol generator to the geometry engine for transmitting a vertex list to the geometry engine wherein the geometry engine operates on the vertex list to transform the vertex list into screen space X, Y coordinates and passes the screen space X, Y coordinates to the tilling engine for generating planar polygons which form icons for display and processing information from the tilling engine into symbols,

(f) a texture engine means coupled to receive the ADDRESS SIGNAL from the shape address generator means including a texture memory and including a means for generating a texture vertex address to texture space correlated to an elevation post address and further including a means for generating a texture memory address for scanning the texture memory wherein the texture memory provides texture data on a texture memory data bus in response to being scanned by the texture memory address;

(g) a rendering engine having an input coupled to the tilling engine and the texture memory data bus for generating image data from the planar polygons; and

(h) a display memory for receiving image data from the rendering engine output wherein the display memory includes at least four first-in, first-out memory buffers.

2. The apparatus of claim **1** wherein each polygon has a surface and the rendering means assigns one color across the surface of each polygon.

3. The apparatus of claim **1** wherein the vertices of each polygon have an intensity and the rendering means interpolates the intensities between the vertices of each polygon in a linear fashion.

4. The apparatus of claim **1** wherein the rendering means further includes means for generating transparent polygons and passing the transparent polygon to the display memory.

5. A method for providing a texture mapped perspective view for a digital map system having a cache memory, a geometry engine coupled to the cache memory, a shape address generator coupled to the cache memory, a tiling engine coupled to the geometry engine, a symbol generator coupled to the geometry engine and the tiling engine, a texture engine coupled to the cache memory, a rendering engine coupled to the tiling engine and the texture engine, and a display memory coupled to the rendering engine, wherein objects are transformed from texture space having U, V coordinates to screen space having X, Y coordinates, the method comprising the steps of:

(a) storing terrain data, including elevation posts, in the cache memory;

(b) scanning the cache memory to retrieve the elevation posts;

(c) transforming the terrain data from elevation posts in object space to transformed vertices in screen space, and

(d) generating planar polygons from the generated three dimensional coordinates;

(e) transmitting a vertex list to the geometry engine, operating the geometry engine to transform the vertex list into screen space X, Y coordinates and passing the screen space X, Y coordinates to the tiling engine for generating planar polygons which form icons for display;

(f) tagging elevation posts with corresponding addresses in texture space;

(g) generating image data in the rendering engine from the planar polygons and the tagged elevation posts; and

(h) storing the generated image data in the display memory wherein the display memory comprises at least four first-in, first-out memory buffers and the step of storing the generated images includes storing the generated image data in the at least four First-in, First-out memory buffers.

6. The method of claim 5 wherein each polygon has a surface and wherein the step of generating image data further includes the steps of assigning one color across the surface of each polygon.

7. The method of claim 5 wherein the vertices of each polygon have an intensity and the step of generating image data further includes the step of interpolating the intensities between the vertices of each polygon in a linear fashion.

8. The method of claim 5 wherein the step of generating image data further includes the step of generating transparent polygons and passing the transparent polygons to the display memory.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 5,179,638
DATED : January 12, 1993
INVENTOR(S) : John F. Dawson, Thomas D. Snodgrass, and James A. Cousens

It is certified that error appears in the above-indentified patent and that said Letters Patent is hereby corrected as shown below:

Column 17, line 21, after "and" insert --generating three dimensional coordinates for the transformed vertices in screen space--.

Signed and Sealed this

Twenty-second Day of March, 1994

Attest:

BRUCE LEHMAN

Attesting Officer

Commissioner of Patents and Trademarks

Pyramidal Parametrics

Lance Williams

Computer Graphics Laboratory
New York Institute of Technology
Old Westbury, New York

Abstract

The mapping of images onto surfaces may substantially increase the realism and information content of computer-generated imagery. The projection of a flat source image onto a curved surface may involve sampling difficulties, however, which are compounded as the view of the surface changes. As the projected scale of the surface increases, interpolation between the original samples of the source image is necessary; as the scale is reduced, approximation of multiple samples in the source is required. Thus a constantly changing sampling window of view-dependent shape must traverse the source image.

To reduce the computation implied by these requirements, a set of prefiltered source images may be created. This approach can be applied to particular advantage in animation, where a large number of frames using the same source image must be generated. This paper advances a "pyramidal parametric" prefiltering and sampling geometry which minimizes aliasing effects and assures continuity within and between target images.

Although the mapping of texture onto surfaces is an excellent example of the process and provided the original motivation for its development, pyramidal parametric data structures admit of wider application. The aliasing of not only surface texture, but also highlights and even the surface representations themselves, may be minimized by pyramidal parametric means.

General Terms: Algorithms.

Keywords and Phrases: Antialiasing, Illumination Models, Modeling, Pyramidal Data Structures, Reflectance Mapping, Texture Mapping, Visible Surface Algorithms.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation--display algorithms; I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling--curve, surface, solid and object representations, geometric algorithms, languages and systems; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism--color, shading, shadowing, and texture.

1. Pyramidal Data Structures

Pyramidal data structures may be based on various subdivisions: binary trees, quad trees, oct trees, or n-dimensional hierarchies [17]. The common feature of these structures is a succession of levels which vary the resolution at which the data is represented.

The decomposition of an image by two-dimensional binary subdivision was a pioneering strategy in computer graphics for visible surface determination [15]. The approach was essentially a synthesis-by-analysis: the image plane was subdivided into quadrants recursively until analysis of a subsection showed that surface ordering was sufficiently simple to permit rendering. Such subdivision and analysis has been subsequently adopted to generate spatial data structures [5], which have been used to represent images [9] both for pattern recognition [13] and for transmission [10], [14]. In the field of computer graphics, such data structures have been adopted for texture mapping [4], [16], and generalized to represent objects in space [11].

The application of pyramidal data to image storage and transmission may permit significant compression of the data to be stored or transmitted. This is so because highly detailed features may be localized within an otherwise low-frequency image, permitting the sampling rate to be reduced for large sections of the image. Besides permitting bandwidth compression, the representation orders data in such a way that the general character of images may be recalled or transmitted before the specific details.

Pattern recognition and classification often require the comparison of a candidate image against a set of canonical patterns. This is an operation the expense of which increases as the square of the resolution at which it is performed. The use of pyramidal data structures in pattern recognition and classification permits the comparison of the gross features of two-dimensional functions preliminary to the minute particulars; a good general reference on this application is [12].

In computer graphics, pyramidal texture maps may be used to perform arbitrary mappings of a function with minimal aliasing artifacts and reduced computation. Once again, images may be represented at different spatial bandwidths. The concern is that inappropriate resolution misrepresents the data; that is, sampling high-resolution data at larger sample intervals invites aliasing.

2. Parametric Interpolation

By a pyramidal parametric data structure, we will mean simply a pyramidal structure with both intra- and inter-level interpolation. Consider the case of an image represented as a two-dimensional array of samples. Interpolation is necessary to produce a continuous function of two parameters, U and V. If, in addition, a third parameter (call it D) moves us up and down a hierarchy of corresponding two-dimensional functions, with interpolation between (or among) the levels of the pyramid providing continuity, the structure is pyramidal parametric.

The practical distinction between such a structure and an ordinary interpolant over an n-dimensional array of samples is that the number of samples representing each level of the pyramid may be different.

3. Mip Mapping

"Mip" mapping is a particular format for two-dimensional parametric functions, which, along with its associated addressing scheme, has been used successfully to bandlimit texture mapping at New York Institute of Technology since 1979. The acronym "mip" is from the Latin phrase "multum in parvo," meaning "many things in a small place." Mip mapping supplements bilinear interpolation of pixel values in the texture map (which may be used to smoothly translate and magnify the texture) with interpolation between prefiltered versions of the map (which may be used to compress many pixels into a small place). In this latter capacity, mip offers much greater speed than texturing algorithms which perform explicit convolution over an area in the texture map for each pixel rendered [1], [6].

Mip owes its speed in compressing texture to two factors. First, a fair amount of filtering of the original texture takes place when the mip map is first created. Second, subsequent filtering is approximated by blending different levels of the mip map. This means that all filters are approximated by linearly interpolating a set of square box filters, the sides of which are powers-of-two pixels in length. Thus, mapping entails a fixed overhead, which is independent of the area filtered to compute a sample.

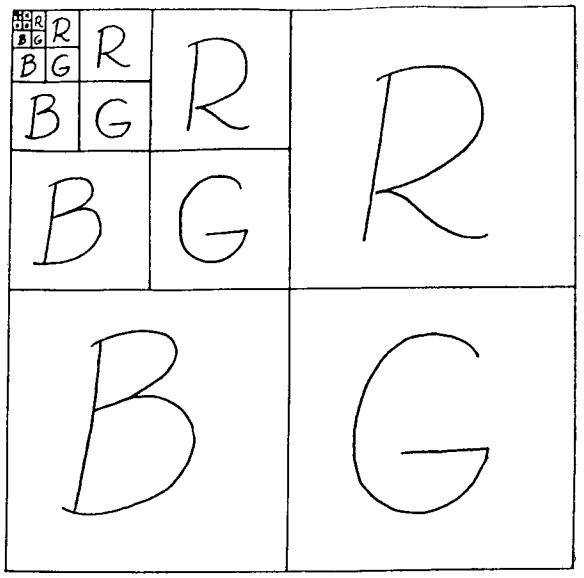


Figure (1)
Structure of a Color Mip Map
Smaller and smaller images diminish into the upper left corner of the map. Each of the images is averaged down from its larger predecessor.

(Below:)
Mip maps are indexed by three coordinates: U, V, and D. U and V are spatial coordinates of the map; D is the variable used to index, and interpolate between, the different levels of the pyramid.

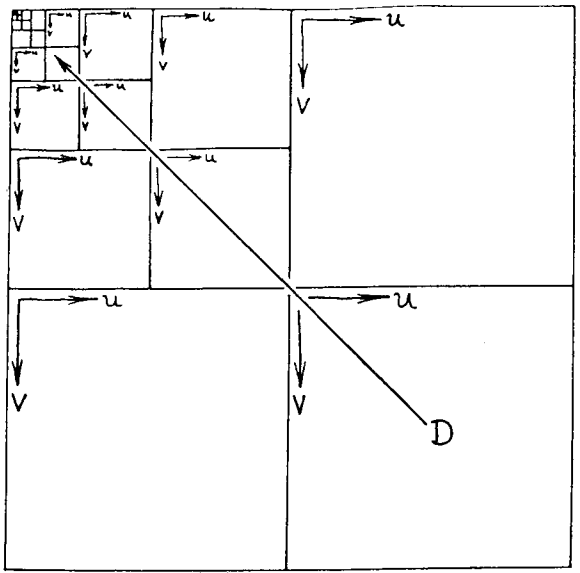


Figure (1) illustrates the memory organization of a color mip map. The image is separated into its red, green, and blue components (R, G, and B in the diagram). Successively filtered and downsampled versions of each component are instanced above and to the left of the originals, in a series of smaller and smaller images, each half the linear dimension (and a quarter the number of

samples) of its parent. Successive divisions by four partition the frame buffer equally among the three components, with a single unused pixel remaining in the upper left-hand corner.

The concept behind this memory organization is that corresponding points in different prefiltered maps can be addressed simply by a binary shift of an input U, V coordinate pair. Since the filtering and sampling are performed at scales which are powers of two, indexing the maps is possible with inexpensive binary scaling. In a hardware implementation, the addresses in all the corresponding maps (now separate memories) would be instantly and simultaneously available from the U, V input.

The routines for creating and accessing mip maps at NYIT are based on simple box (Fourier) window prefiltering, bilinear interpolation of pixels within each map instance, and linear interpolation between two maps for each value of D (the pyramid's vertical coordinate). For each of the three components of a color mip map, this requires 8 pixel reads and 7 multiplications. This choice of filters is strictly for the sake of speed. Note that the bilinear interpolation of pixel values at the extreme edges of each map instance must be performed with pixels from the opposite edge(s) of that map, for texture which is periodic. For non-periodic texture, scaling or clipping of the U, V coordinates prevents the intrusion of an inappropriate map or color component into the interpolation.

The box (Fourier) window used to create the mip maps illustrated here, and the tent (Bartlett) window used to interpolate them, are far from ideal; yet probably the most severe compromise made by mip filtering is that it is symmetrical. Each of the prefiltered levels of the map is filtered equally in X and Y. Choosing a value of D trades off aliasing against blurring, which becomes a tricky proposition as a pixel's projection in the texture map deviates from symmetry. Heckbert [8] suggests:

$$d = \max\left(\sqrt{\left(\frac{\partial u}{\partial x}\right)^2+\left(\frac{\partial v}{\partial x}\right)^2}\ ,\ \sqrt{\left(\frac{\partial u}{\partial y}\right)^2+\left(\frac{\partial v}{\partial y}\right)^2}\right)$$

where D is proportional to the "diameter" of the area in the texture to be filtered, and the partials of U and V (the texture-map coordinates) with respect to X and Y (the screen coordinates) can be calculated from the surface projection.

Illustrations of mapping performed by the mip technique are the subject of Figures (2) through (10). The NYIT Test Frog in Figure (2) is magnified by simple point sampling in (3), and by interpolation in (4). The hapless amphibian is similarly

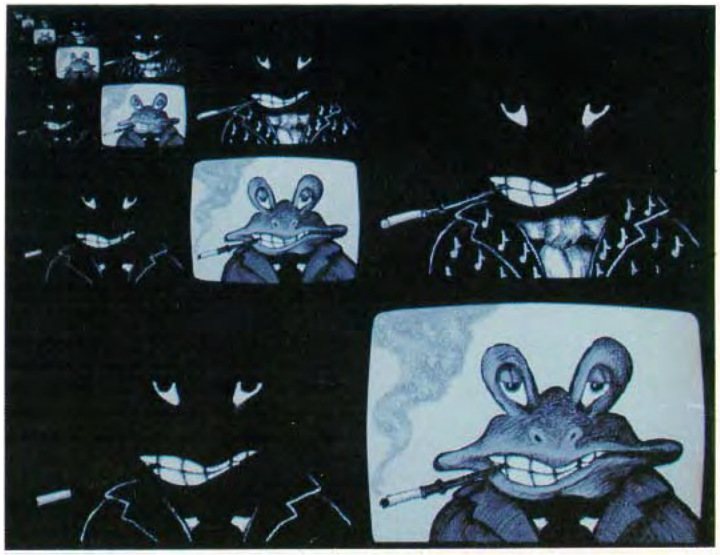


Figure (2)
Mip map of the flexible NYIT Test Frog.

compressed by point sampling in (5) and by mipping in (6).

The more general and interesting case -- continuously variable upsampling and downsampling of the original texture -- is illustrated in (7) on a variety of surfaces. Since the symmetry of mip filtering would be expected to show up badly when texture is compressed in only one dimension, figures (8) through (10) are of especial interest. These pictures, created by Ed Emshwiller at NYIT for his videotape, "Sunstone," were mapped using Alvy Ray Smith's TEXAS animation program, which in turn used MIP to antialias texture. As the panels rotate edge-on, the texture collapses to a line smoothly and without apparent artifacts.

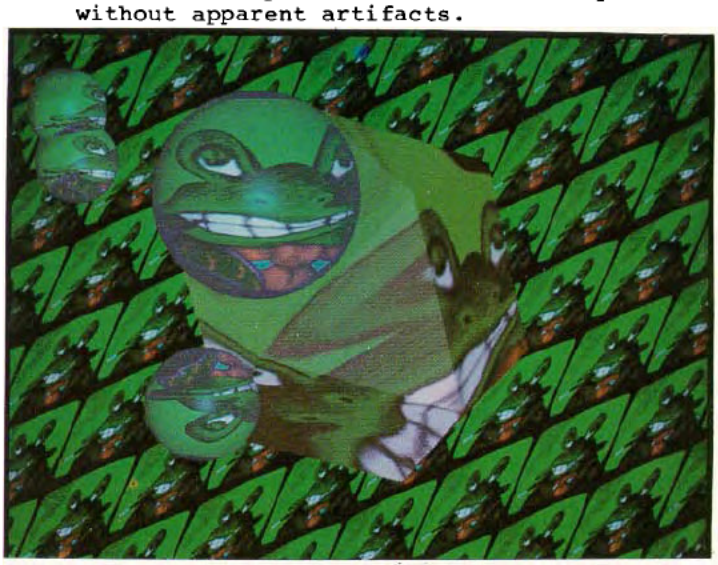
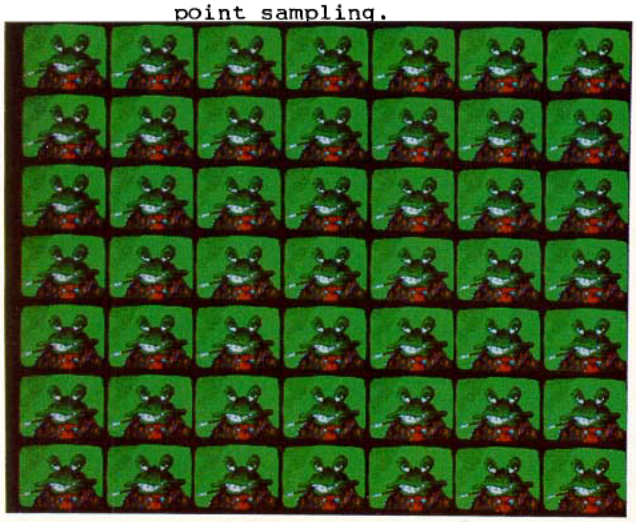


Figure (7)
General mapping: interpolation and pyramidal compression.

Figure (3)
Upsampling the frog: magnification by point sampling.

Figure (4)
Upsampling the frog: magnification by bilinear interpolation.



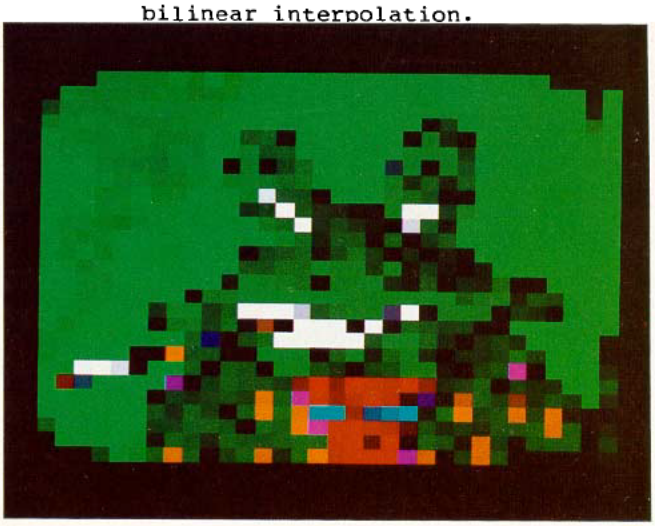
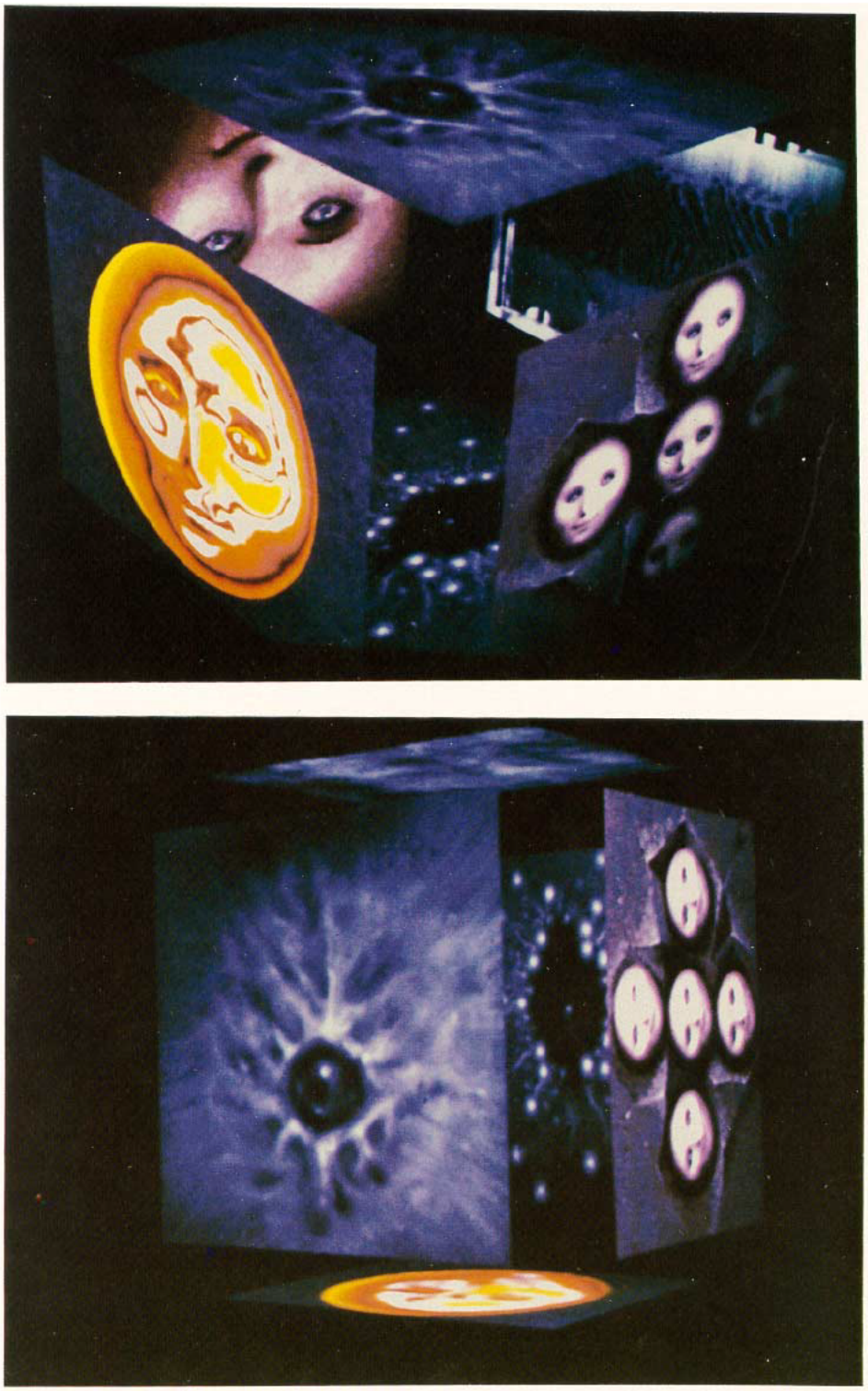
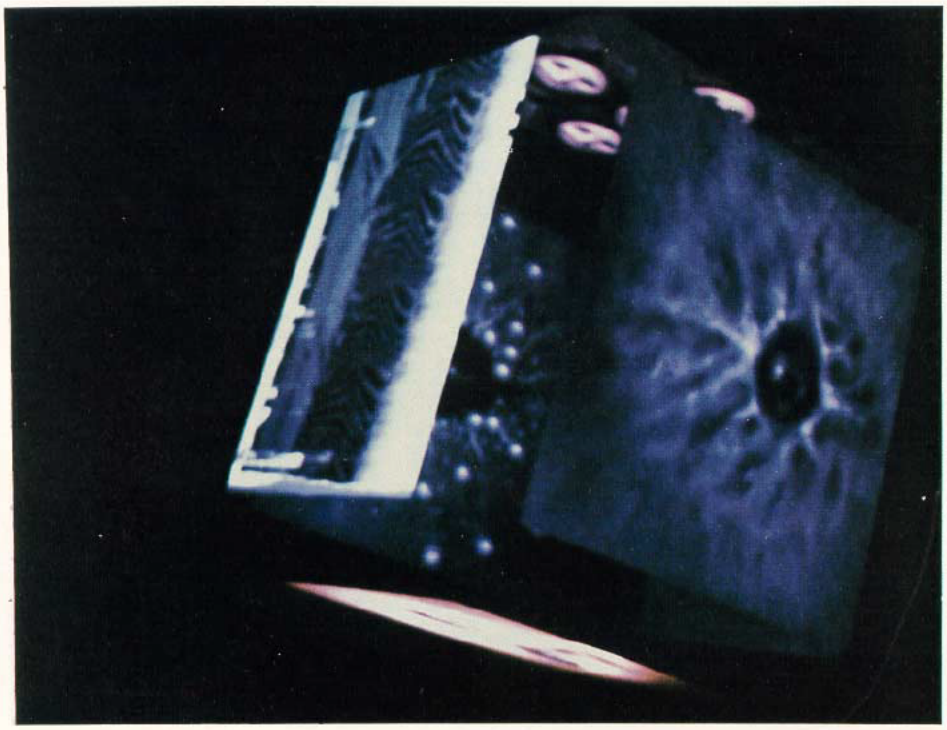


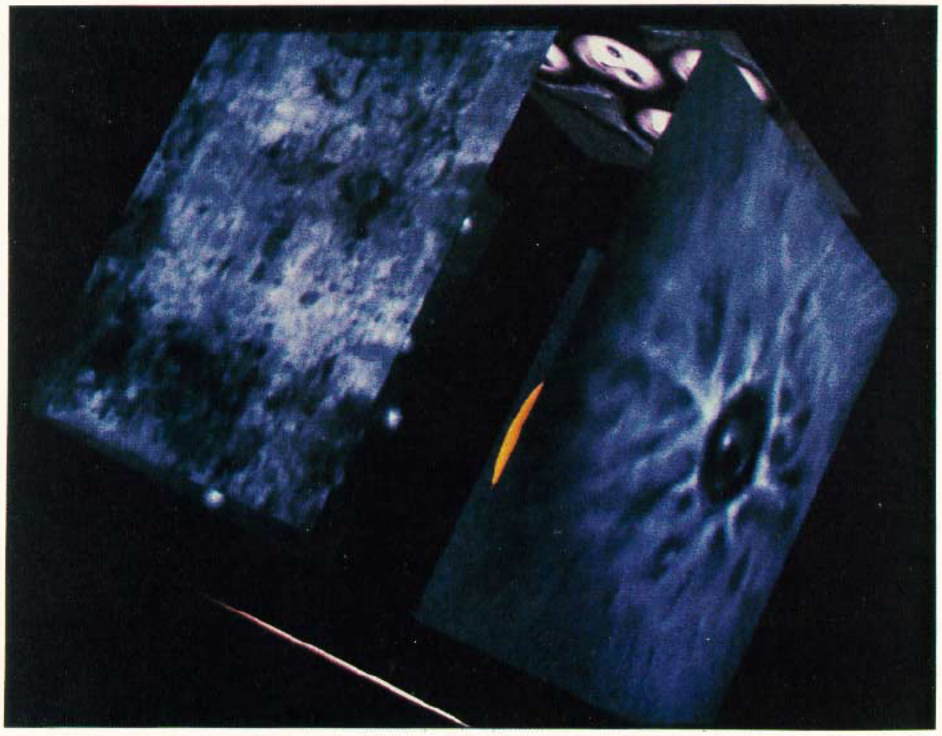
Figure (5)
Downsampling the frog: compression by point sampling (detail, right).

Figure (6)
Downsampling: compression by pyramidal interpolation (detail, right).



Figures (8)-(9)
"Sunstone" by Ed Emshwiller, segment animated by Alvy Ray Smith
Pyramidal parametric texture mapping on polygons.





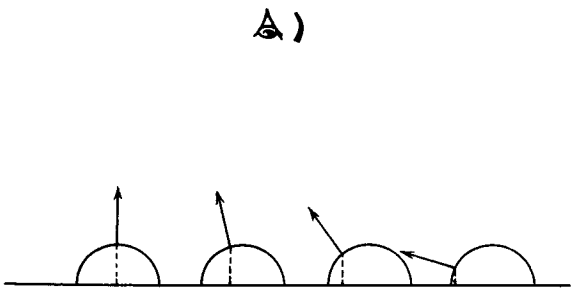
Figures (10)-(11)
"Sunstone" by Ed Emshwiller, segment animated by Alvy Ray Smith
Pyramidal parametric texture mapping on polygons.

4. Highlight Antialiasing

As small or highly curved objects move across a raster, their surface normals may beat erratically with the sampling grid. This causes the shading values to flash annoyingly in motion sequences, a symptom of illumination aliasing. The surface normals essentially point-sample the illumination function.

Figure (12) illustrates samples of the surface normals of a set of parallel cylinders. The cylinders in the diagram are depicted as if from the edge of the image plane; the regularly-spaced vertical line segments are the samples along a single axis. The arrows at the sample points indicate the directions of the surface normals. Depending on the shading formula invoked, there may be very high contrast between samples where the normal is nearly parallel to the sample axis, and samples where the normal points directly at the observer's eye.

Figure (12)



The shading function depends not only on the shape of the surface, but its light reflection properties (characterized by the shading formula), the position of the light source, and the position of the observer's eye. Hanrahan [7] expresses it in honest Greek:

$$\int_x \int_y \varphi(E,N,L) \frac{\partial(u,v)}{\partial(x,y)} \, dxdy$$

where the normal, N, the light sources, L, and the eye, E, are vectors which may each be functions of U and V, and the limits of integration are the X, Y boundaries of the pixel.

Figure (13) illustrates highlight aliasing on a perfectly flat surface. The viewing conventions of the diagram are the same as in Figure (12). "L" is the direction vector of the light source; the surface is a polygon at an angle to the image plane; the dotted bump is a graph of the reflected light, characteristic of a specular surface reflection function. The highlight indicated by the bump falls entirely between the samples. (Note that this is only possible on a flat surface if either the eye or the light is local, a point in space rather than simply a direction vector. Some boring shading formulae exclude the possibility of highlight aliasing on polygons by requiring all flat surfaces to be flat in shading.)

Figure (13)

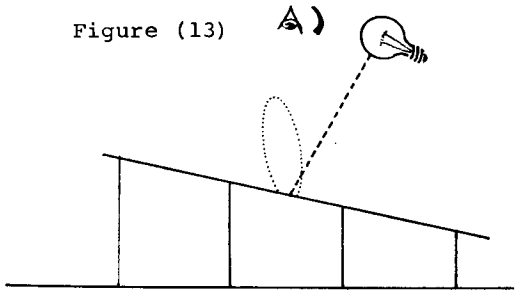
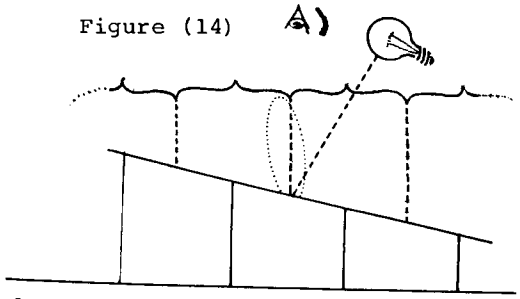


Figure (14)



A first attempt to overcome the limitations of point-sampling the illumination function is to integrate the function over the projected area represented by each sample point. This approach is illustrated in Figure (14). The brackets at each sample represent the area of the surface over which the illumination function is integrated. This procedure is analogous to area-averaging of sampled edges or texture [3].

In order to generalize this approach to curved surfaces, the "sample interval" over which illumination is integrated must be modified according to the local curvature of the surface at a sample. In Figure (15), the area of a surface represented by a pixel has been projected onto a curved surface. The solid angle over which illumination must be integrated is approximated by the volume enclosed by the normals at the pixel corners. The distribution of light within this volume will sum to an estimate of the diffuse reflection over the pixel. If the surface exhibits undulations at the pixel level, however, aliasing will result.

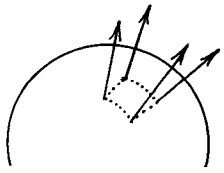


Figure (15)

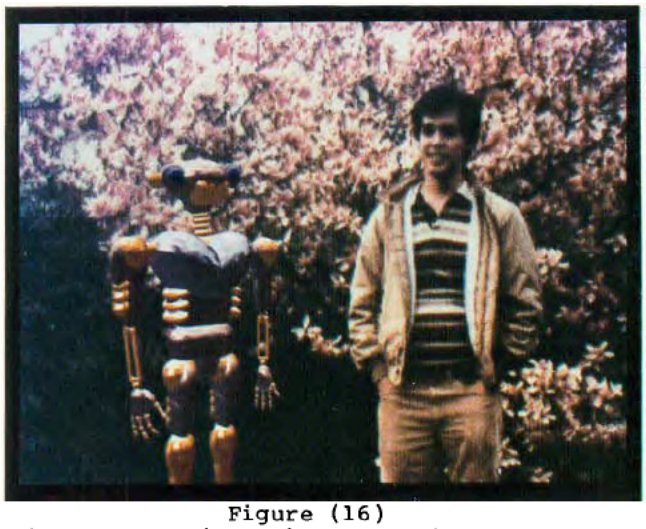


Figure (16)
Michael Chou (right) poses with an imaginary companion. Reflectance maps can enhance the realism of synthetic shading.

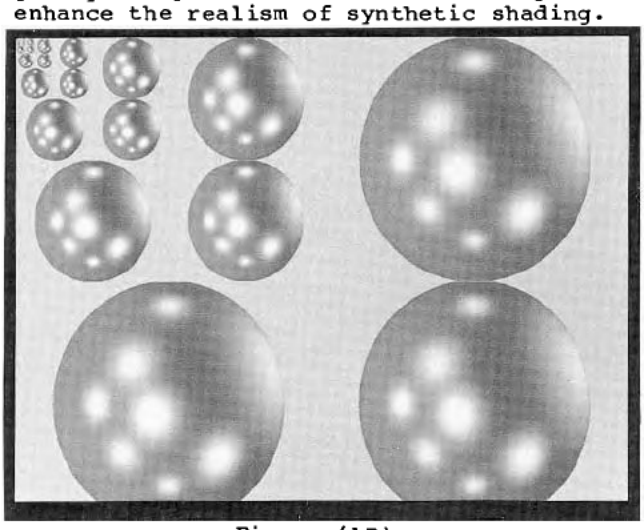


Figure (17)
A pyramidal parametric reflectance map, containing 9 light sources. The region outside the "sphere" is unused

We might divide the surface up into regions of relatively low curvature (as is done in some patch rendering algorithms), and rely on "edge antialiasing" to integrate the different surfaces within a pixel. Alternatively, we may develop some mechanism for limiting the local curvature of surfaces before rendering. This possibility is explored in the next section.

If we represent the illumination of a scene as a two-dimensional map, highlights can be effectively antialiased in much the same way as textures. Blinn and Newell [1] demonstrated specular reflection using an illumination map. The map was an image of the environment (a spherical projection of the scene, indexed by the X and Y components of the surface normals) which could be used to cast reflections onto specular surfaces. The impression of mirrored facets and chrome objects which can be achieved with this method is striking; Figure (16) provides an illustration. Reflectance mapping is not, however, accurate for local reflections. To achieve similar results with three dimensional accuracy requires ray-tracing.

A pyramidal parametric illumination map permits convenient antialiasing of highlights as long as a good measure of local surface curvature is available. The value of "D" used to index the map is proportional to the solid angle subtended by the surface over the pixel being computed; this may be estimated by the same formula used to compute D for ordinary texture mapping. Nine light sources of varying brightness glint raggedly from the test object in Figure (18); the reflectance map in Figure (17) provided the illumination. In Figure (19), convincing highlight antialiasing results from the full pyramidal parametric treatment.

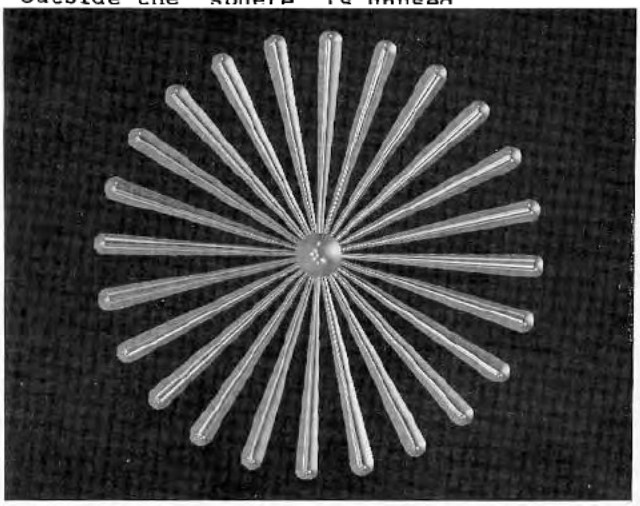


Figure (18) Before

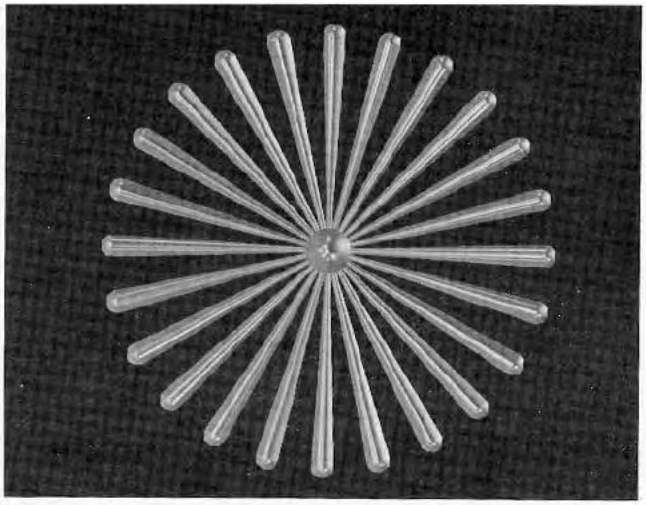
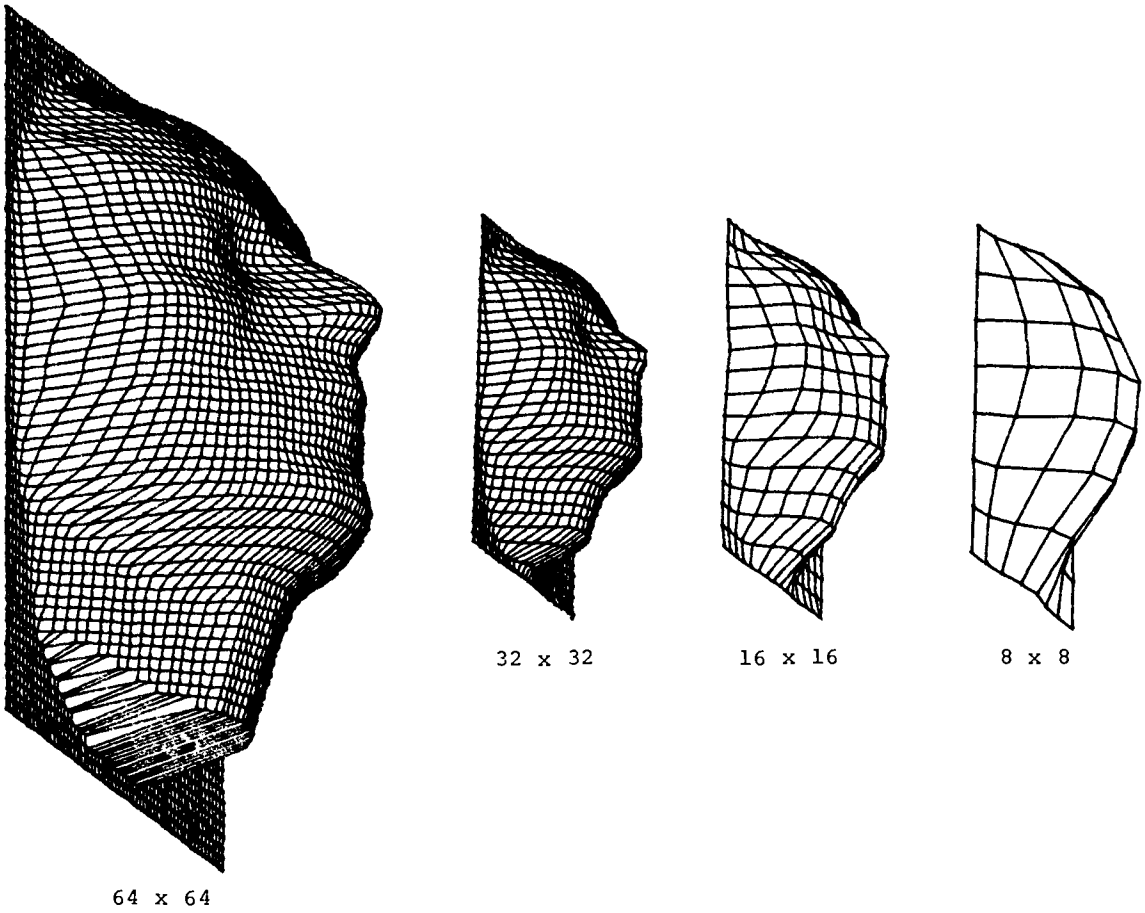


Figure (19) After



Figures (20-23) Different resolution meshes.

5. Levels of Detail in Surface Representation

In addition to bandlimiting texture and illumination functions for mapping onto a surface, pyramidal parametrics may be used to limit the level of detail with which the surface itself is represented. The goal is to represent an object for graphic display as economically as its projection on the image plane permits, without boiling and sparkling aliasing artifacts as the projection changes.

The expense of computing and shading each pixel dominates the cost of many algorithms for rendering higher-order surfaces. For meshes of polygons or patch control points which project onto a small portion of the image, however, the vertex (or control-point) expense dominates. In these situations it is desirable to reduce the number of points used to represent the object.

A pyramidal parametric data structure the components of which are spatial coordinates (the X-Y-Z of the vertices of a rectangular mesh, for example, as opposed to the R-G-B of a texture or illumination map) provides a continuously-variable filtered instance of the surface for sampling at any desired degree of resolution.

Figures (20) through (23) illustrate a simple surface based on a human face model developed by Fred Parke at the University of Utah. As the sampling density varies, so does the filtering of the surface. These faces are filtered and sampled by the same methods previously discussed for texture and reflectance maps. Pyramidal parametric representations such as these appear promising for reducing aliasing effects as well as systematically sampling very large data bases over a wide range of scales and viewing angles.

6. Conclusions

Pyramidal data structures are of proven value in image analysis and have interesting application to image bandwidth compression and transmission. "Pyramidal parametrics," pyramidal data structures with intra- and inter-level interpolation, are here proposed for use in image synthesis. By continuously varying the detail with which data are resolved, pyramidal parametrics provide economical approximate solutions to filtering problems in mapping texture and illumination onto surfaces, and preliminary experiments suggest they may provide flexible surface representations as well.

7. Acknowledgments

I would like to acknowledge Ed Catmull, the first (to my knowledge) to apply multiple prefiltered images to texture mapping: the method was applied to the bicubic patches in his thesis, although it was not described. Credit is also due Tom Duff, who wrote both recursive and scan-order routines for creating mip maps which preserved numerical precision over all map instances; Dick Lundin, who wrote the first assembly-coded mip map accessing routines; Ephraim Cohen, who wrote the second; Rick Ace, who translated Ephraim's PDP-11 versions for the VAX assembler; Paul Heckbert, for refining and speeding up both creation and accessing routines, and investigating various estimates of "D"; Michael Chou, for implementing highlight antialiasing and high-resolution reflectance mapping on quadric surfaces.

I owe special thanks to Jules Bloomenthal, Michael Chou, Pat Hanrahan, and Paul Heckbert for critical reading and numerous helpful suggestions in the course of preparing this text. Photographic support was provided by Michael Lehman.

8. References

[1] Blinn, J., and Newell, M., "Texture and Reflection on Computer Generated Images," CACM, Vol. 19, #10, Oct. 1976, pp. 542-547.

[2] Bui-Tuong Phong, "Illumination for Computer Generated Pictures," PhD. dissertation, Department of Computer Science, University of Utah, December 1978.

[3] Crow, F.C., "The Aliasing Problem in Computer Synthesized Shaded Images," PhD. dissertation, Department of Computer Science, University of Utah, Tech. Report UTEC-CSc-76-015, March 1976.

[4] Dungan, W., Stenger, A., and Sutty, G., "Texture Tile Considerations for Raster Graphics," SIGGRAPH 1978 Proceedings, Vol. 12, #3, August 1978.

[5] Eastman, Charles M., "Representations for Space Planning," CACM, Vol. 13, #4, April 1970.

[6] Feibush, E.A., Levoy, M., and Cook, R.L., "Synthetic Texturing Using Digital Filters," Computer Graphics, Vol. 14, July, 1980.

[7] Hanrahan, Pat, private communication, 1983.

[8] Heckbert, Paul, "Texture Mapping Polygons in Perspective," NYIT Computer Graphics Lab Tech. Memo #13, April, 1983.

[9] Klinger, A., and Dyer, C.R., "Experiments on Picture Representation Using Regular Decomposition," Computer Graphics and Image Processing, #5, March, 1976.

[10] Knowlton, K., "Progressive Transmission of Gray-Scale and Binary Pictures by Simple, Efficient, and Lossless Encoding Schemes," Proceedings of the IEEE, Vol. 68, #7, July 1980, pp. 885-896.

[11] Meagher, D., "Octree Encoding: A New Technique for the Representation, Manipulation, and Display of Arbitrary 3D Objects by Computer," IPL-TR-80-111, Image Processing Lab, Electrical and Systems Engineering Dept., Rensselaer Polytechnic Institute, October 1980.

[12] Tanimoto, S.L., and Klinger, A., Structured Computer Vision, Academic Press, New York, 1980.

[13] Tanimoto, S.L., and Pavlidis, T., "A Hierarchical Data Structure for Picture Processing," Computer Graphics and Image Processing, Vol. 4, #2, June 1975.

[14] Tanimoto, S.L., "Image Processing with Gross Information First," Computer Graphics and Image Processing 9, 1979.

[15] Warnock, J.E., "A Hidden-Line Algorithm for Halftone Picture Representation," Department of Computer Science, University of Utah, TR 4-15, 1969.

[16] Williams, L., "Pyramidal Parametrics," SIGGRAPH tutorial notes, "Advanced Image Synthesis," 1981.

[17] Yau, M.M., and Srihari, S.N., "Recursive Generation of Hierarchical Data Structures for Multidimensional Digital Images," Proceedings of the IEEE Computer Society Conference on Pattern Recognition and Image Processing, August 1981.

Next Up Previous Contents Index
Next: 3.8.2 Texture Magnification **Up:** 3.8.1 Texture Minification **Previous:** 3.8.1 Texture Minification

Mipmapping

`TEXTURE_MIN_FILTER` values `NEAREST_MIPMAP_NEAREST`, `NEAREST_MIPMAP_LINEAR`, `LINEAR_MIPMAP_NEAREST`, and `LINEAR_MIPMAP_LINEAR` each require the use of a *mipmap*. A mipmap is an ordered set of arrays representing the same image; each array has a resolution lower than the previous one. If the texture has dimensions $2^n \times 2^m$, then there are $\max\{n, m\} + 1$ mipmap arrays. The first array is the original texture with dimensions $2^n \times 2^m$. Each subsequent array has dimensions $2^{(k-1)} \times 2^{(l-1)}$ where $2^k \times 2^l$ are the dimensions of the previous array. This is the case as long as both **k>0** and **l>0**. Once either **k=0** or **l=0**, each subsequent array has dimension $1 \times 2^{(l-1)}$ or $2^{(k-1)} \times 1$, respectively, until the last array is reached with dimension 1×1.

Each array in a mipmap is transmitted to the GL using **TexImage2D** or **TexImage1D** ; the array being set is indicated with the *level-of-detail* argument. Level-of-detail numbers proceed from **0** for the original texture array through $p = \max\{n, m\}$ with each unit increase indicating an array of half the dimensions of the previous one as already described. If texturing is enabled (and `TEXTURE_MIN_FILTER` is one that requires a mipmap) at the time a primitive is rasterized and if the set of arrays **0** through **p** is incomplete, based on the dimensions of array **0**, then it is as if texture mapping were disabled. The set of arrays **0** through **p** is incomplete if the internal formats of all the mipmap arrays were not specified with the same symbolic constant, or if the border widths of the mipmap arrays are not the same, or if the dimensions of the mipmap arrays do not follow the sequence described above. Arrays indexed greater than **p** are insignificant.

The mipmap is used in conjunction with the level of detail to approximate the application of an appropriately filtered texture to a fragment. Let $p = \max\{n, m\}$ and let **c** be the value of λ at which the transition from minification to magnification occurs (since this discussion pertains to minification, we are concerned only with values of λ where $\lambda > c$). For `NEAREST_MIPMAP_NEAREST`, if $c < \lambda \leq 0.5$ then the mipmap array with level-of-detail of 0 is selected. Otherwise, the **d**th mipmap array is selected when $d - \frac{1}{2} < \lambda \leq d + \frac{1}{2}$ as long as $1 \leq d \leq p$. If $\lambda > p + \frac{1}{2}$, then the **p**th mipmap array is selected. The rules for `NEAREST` are then applied to the selected array.

The same mipmap array selection rules apply for `LINEAR_MIPMAP_NEAREST` as for `NEAREST_MIPMAP_NEAREST`, but the rules for `LINEAR` are applied to the selected array.

For `NEAREST_MIPMAP_LINEAR`, the level **d-1** and the level **d** mipmap arrays are selected, where $d - 1 \leq \lambda < d$, unless $\lambda \geq p$, in which case the **p**th mipmap array is used for both arrays. The rules

for NEAREST are then applied to each of these arrays, yielding two corresponding texture values τ_{d-1} and τ_d. The final texture value is then found as

$$\tau = [1 - \mathrm{frac}(\lambda)]\tau_{d-1} + \mathrm{frac}(\lambda)\tau_d.$$

LINEAR_MIPMAP_LINEAR has the same effect as NEAREST_MIPMAP_LINEAR except that the rules for LINEAR are applied for each of the two mipmap arrays to generate τ_{d-1} and τ_d.

Next Up Previous Contents Index
Next: 3.8.2 Texture Magnification **Up:** 3.8.1 Texture Minification **Previous:** 3.8.1 Texture Minification

---

David Blythe
Sat Mar 29 02:23:21 PST 1997

Progressive Meshes

Hugues Hoppe
Microsoft Research

ABSTRACT

Highly detailed geometric models are rapidly becoming commonplace in computer graphics. These models, often represented as complex triangle meshes, challenge rendering performance, transmission bandwidth, and storage capacities. This paper introduces the *progressive mesh* (PM) representation, a new scheme for storing and transmitting arbitrary triangle meshes. This efficient, lossless, continuous-resolution representation addresses several practical problems in graphics: smooth geomorphing of level-of-detail approximations, progressive transmission, mesh compression, and selective refinement.

In addition, we present a new mesh simplification procedure for constructing a PM representation from an arbitrary mesh. The goal of this optimization procedure is to preserve not just the geometry of the original mesh, but more importantly its overall appearance as defined by its discrete and scalar appearance attributes such as material identifiers, color values, normals, and texture coordinates. We demonstrate construction of the PM representation and its applications using several practical models.

CR Categories and Subject Descriptors: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling - surfaces and object representations.

Additional Keywords: mesh simplification, level of detail, shape interpolation, progressive transmission, geometry compression.

1 INTRODUCTION

Highly detailed geometric models are necessary to satisfy a growing expectation for realism in computer graphics. Within traditional modeling systems, detailed models are created by applying versatile modeling operations (such as extrusion, constructive solid geometry, and freeform deformations) to a vast array of geometric primitives. For efficient display, these models must usually be tessellated into polygonal approximations—meshes. Detailed meshes are also obtained by scanning physical objects using range scanning systems [5]. In either case, the resulting complex meshes are expensive to store, transmit, and render, thus motivating a number of practical problems:

- *Mesh simplification*: The meshes created by modeling and scanning systems are seldom optimized for rendering efficiency, and can frequently be replaced by nearly indistinguishable approximations with far fewer faces. At present, this process often requires significant user intervention. Mesh simplification tools can hope to automate this painstaking task, and permit the porting of a single model to platforms of varying performance.
- *Level-of-detail (LOD) approximation*: To further improve rendering performance, it is common to define several versions of a model at various levels of detail [3, 8]. A detailed mesh is used when the object is close to the viewer, and coarser approximations are substituted as the object recedes. Since instantaneous switching between LOD meshes may lead to perceptible "popping", one would like to construct smooth visual transitions, *geomorphs*, between meshes at different resolutions.
- *Progressive transmission*: When a mesh is transmitted over a communication line, one would like to show progressively better approximations to the model as data is incrementally received. One approach is to transmit successive LOD approximations, but this requires additional transmission time.
- *Mesh compression*: The problem of minimizing the storage space for a model can be addressed in two orthogonal ways. One is to use mesh simplification to reduce the number of faces. The other is mesh compression: minimizing the space taken to store a particular mesh.
- *Selective refinement*: Each mesh in a LOD representation captures the model at a uniform (view-independent) level of detail. Sometimes it is desirable to adapt the level of refinement in selected regions. For instance, as a user flies over a terrain, the terrain mesh need be fully detailed only near the viewer, and only within the field of view.

In addressing these problems, this paper makes two major contributions. First, it introduces the *progressive mesh* (PM) representation. In PM form, an arbitrary mesh $\hat{M}$ is stored as a much coarser mesh M^0 together with a sequence of n detail records that indicate how to incrementally refine M^0 exactly back into the original mesh $\hat{M} = M^n$. Each of these records stores the information associated with a *vertex split*, an elementary mesh transformation that adds an additional vertex to the mesh. The PM representation of $\hat{M}$ thus defines a continuous sequence of meshes $M^0, M^1, \ldots, M^n$ of increasing accuracy, from which LOD approximations of any desired complexity can be efficiently retrieved. Moreover, geomorphs can be efficiently constructed between any two such meshes. In addition, we show that the PM representation naturally supports progressive transmission, offers a concise encoding of $\hat{M}$ itself, and permits selective refinement. In short, progressive meshes offer an efficient, lossless, continuous-resolution representation.

The other contribution of this paper is a new simplification procedure for constructing a PM representation from a given mesh $\hat{M}$. Unlike previous simplification methods, our procedure seeks to preserve not just the geometry of the mesh surface, but more importantly its overall appearance, as defined by the discrete and scalar attributes associated with its surface.

Email: hhoppe@microsoft.com
Web: http://www.research.microsoft.com/research/graphics/hoppe/

2 MESHES IN COMPUTER GRAPHICS

Models in computer graphics are often represented using triangle meshes.[1] Geometrically, a triangle mesh is a piecewise linear surface consisting of triangular faces pasted together along their edges. As described in [9], the mesh geometry can be denoted by a tuple (K, V), where K is a *simplicial complex* specifying the connectivity of the mesh simplices (the adjacency of the vertices, edges, and faces), and $V = \{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$ is the set of vertex positions defining the shape of the mesh in $\mathbf{R}^3$. More precisely (cf. [9]), we construct a parametric domain $|K| \subset \mathbf{R}^m$ by identifying each vertex of K with a canonical basis vector of $\mathbf{R}^m$, and define the mesh as the image $\phi_V(|K|)$ where $\phi_V : \mathbf{R}^m \to \mathbf{R}^3$ is a linear map.

Often, surface appearance attributes other than geometry are also associated with the mesh. These attributes can be categorized into two types: *discrete* attributes and *scalar* attributes.

Discrete attributes are usually associated with faces of the mesh. A common discrete attribute, the *material identifier*, determines the shader function used in rendering a face of the mesh [18]. For instance, a trivial shader function may involve simple look-up within a specified texture map.

Many scalar attributes are often associated with a mesh, including diffuse color (r, g, b), normal (n_x, n_y, n_z), and texture coordinates (u, v). More generally, these attributes specify the local parameters of shader functions defined on the mesh faces. In simple cases, these scalar attributes are associated with vertices of the mesh. However, to represent discontinuities in the scalar fields, and because adjacent faces may have different shading functions, it is common to associate scalar attributes not with vertices, but with corners of the mesh [1]. A *corner* is defined as a (vertex,face) tuple. Scalar attributes at a corner (v,f) specify the shading parameters for face f at vertex v. For example, along a *crease* (a curve on the surface across which the normal field is not continuous), each vertex has two distinct normals, one associated with the corners on each side of the crease.

We express a mesh as a tuple $M = (K, V, D, S)$ where V specifies its geometry, D is the set of discrete attributes d_f associated with the faces $f = \{j, k, l\} \in K$, and S is the set of scalar attributes $s_{(v,f)}$ associated with the corners (v,f) of K.

The attributes D and S give rise to discontinuities in the visual appearance of the mesh. An edge $\{v_j, v_k\}$ of the mesh is said to be *sharp* if either (1) it is a boundary edge, or (2) its two adjacent faces f_l and f_r have different discrete attributes (i.e. $d_{f_l} \neq d_{f_r}$), or (3) its adjacent corners have different scalar attributes (i.e. $s_{(v_j,f_l)} \neq s_{(v_j,f_r)}$ or $s_{(v_k,f_l)} \neq s_{(v_k,f_r)}$). Together, the set of sharp edges define a set of *discontinuity curves* over the mesh (e.g. the yellow curves in Figure 12).

3 PROGRESSIVE MESH REPRESENTATION

3.1 Overview

Hoppe et al. [9] describe a method, *mesh optimization*, that can be used to approximate an initial mesh $\hat{M}$ by a much simpler one. Their optimization algorithm, reviewed in Section 4.1, traverses the space of possible meshes by successively applying a set of 3 mesh transformations: edge collapse, edge split, and edge swap.

We have discovered that in fact a single one of those transformations, *edge collapse*, is sufficient for effectively simplifying meshes. As shown in Figure 1, an edge collapse transformation $ecol(\{v_s, v_t\})$

[1] We assume in this paper that more general meshes, such as those containing n-sided faces and faces with holes, are first converted into triangle meshes by triangulation. The PM representation could be generalized to handle the more general meshes directly, at the expense of more complex data structures.

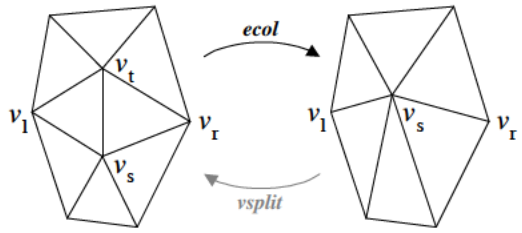


Figure 1: Illustration of the edge collapse transformation.

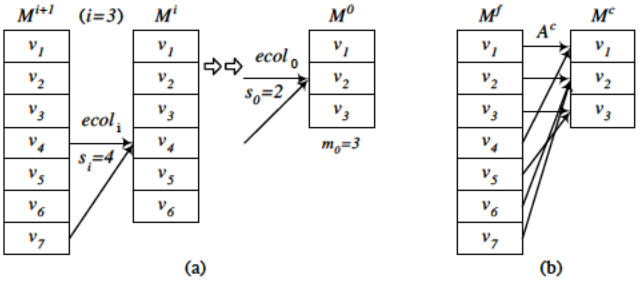


Figure 2: (a) Sequence of edge collapses; (b) Resulting vertex correspondence.

unifies 2 adjacent vertices v_s and v_t into a single vertex v_s. The vertex v_t and the two adjacent faces $\{v_s, v_t, v_l\}$ and $\{v_t, v_s, v_r\}$ vanish in the process. A position $\mathbf{v}_s$ is specified for the new unified vertex.

Thus, an initial mesh $\hat{M} = M^n$ can be simplified into a coarser mesh M^0 by applying a sequence of n successive edge collapse transformations:

$$(\hat{M}=M^n) \xrightarrow{ecol_{n-1}} \ldots \xrightarrow{ecol_1} M^1 \xrightarrow{ecol_0} M^0 .$$

The particular sequence of edge collapse transformations must be chosen carefully, since it determines the quality of the approximating meshes $M^i, i < n$. A scheme for choosing these edge collapses is presented in Section 4.

Let m_0 be the number of vertices in M^0, and let us label the vertices of mesh M^i as $V^i = \{v_1, \ldots, v_{m_0+i}\}$, so that edge $\{v_{s_i}, v_{m_0+i+1}\}$ is collapsed by $ecol_i$ as shown in Figure 2a. As vertices may have different positions in the different meshes, we denote the position of v_j in M^i as $\mathbf{v}_j^i$.

A key observation is that an edge collapse transformation is invertible. Let us call that inverse transformation a *vertex split*, shown as *vsplit* in Figure 1. A vertex split transformation $vsplit(s, l, r, t, A)$ adds near vertex v_s a new vertex v_t and two new faces $\{v_s, v_t, v_l\}$ and $\{v_t, v_s, v_r\}$. (If the edge $\{v_s, v_t\}$ is a boundary edge, we let $v_r = 0$ and only one face is added.) The transformation also updates the attributes of the mesh in the neighborhood of the transformation. This attribute information, denoted by A, includes the positions $\mathbf{v}_s$ and $\mathbf{v}_t$ of the two affected vertices, the discrete attributes $d_{\{v_s,v_t,v_l\}}$ and $d_{\{v_t,v_s,v_r\}}$ of the two new faces, and the scalar attributes of the affected corners $(s_{(v_s,\cdot)}, s_{(v_t,\cdot)}, s_{(v_l,\{v_s,v_t,v_l\})}$, and $s_{(v_r,\{v_t,v_s,v_r\})})$.

Because edge collapse transformations are invertible, we can therefore represent an arbitrary triangle mesh $\hat{M}$ as a simple mesh M^0 together with a sequence of n *vsplit* records:

$$M^0 \xrightarrow{vsplit_0} M^1 \xrightarrow{vsplit_1} \ldots \xrightarrow{vsplit_{n-1}} (M^n=\hat{M})$$

where each record is parametrized as $vsplit_i(s_i, l_i, r_i, A_i)$. We call $(M^0, \{vsplit_0, \ldots, vsplit_{n-1}\})$ a *progressive mesh* (PM) representation of $\hat{M}$.

As an example, the mesh $\hat{M}$ of Figure 5d (13,546 faces) was simplified down to the coarse mesh M^0 of Figure 5a (150 faces) using

6,698 edge collapse transformations. Thus its PM representation consists of M^0 together with a sequence of $n = 6698$ *vsplit* records. From this PM representation, one can extract approximating meshes with any desired number of faces (actually, within ± 1) by applying to M^0 a prefix of the *vsplit* sequence. For example, Figure 5 shows approximating meshes with 150, 500, and 1000 faces.

3.2 Geomorphs

A nice property of the vertex split transformation (and its inverse, edge collapse) is that a smooth visual transition (a *geomorph*) can be created between the two meshes M^i and M^{i+1} in $M^i \xrightarrow{vsplit_i} M^{i+1}$. For the moment let us assume that the meshes contain no attributes other than vertex positions. With this assumption the vertex split record is encoded as $vsplit_i(s_i, l_i, r_i, A_i = (\mathbf{v}_{s_i}^{i+1}, \mathbf{v}_{m_0+i+1}^{i+1}))$. We construct a geomorph $M^G(\alpha)$ with blend parameter $0 \le \alpha \le 1$ such that $M^G(0)$ looks like M^i and $M^G(1)$ looks like M^{i+1}—in fact $M^G(1) = M^{i+1}$—by defining a mesh

$$M^G(\alpha) = (K^{i+1}, V^G(\alpha))$$

whose connectivity is that of M^{i+1} and whose vertex positions linearly interpolate from $v_{s_i} \in M^i$ to the split vertices $v_{s_i}, v_{m_0+i+1} \in M^{i+1}$:

$$\mathbf{v}_j^G(\alpha) = \begin{cases} (\alpha)\mathbf{v}_j^{i+1} + (1-\alpha)\mathbf{v}_{s_i}^i & , \ j \in \{s_i, m_0+i+1\} \\ \mathbf{v}_j^{i+1} = \mathbf{v}_j^i & , \ j \notin \{s_i, m_0+i+1\} \end{cases}$$

Using such geomorphs, an application can smoothly transition from a mesh M^i to meshes M^{i+1} or M^{i-1} without any visible "snapping" of the meshes.

Moreover, since individual *ecol* transformations can be transitioned smoothly, so can the composition of any sequence of them. Geomorphs can therefore be constructed between *any* two meshes of a PM representation. Indeed, given a finer mesh M^f and a coarser mesh M^c, $0 \le c < f \le n$, there exists a natural correspondence between their vertices: each vertex of M^f is related to a unique ancestor vertex of M^c by a surjective map A^c obtained by composing a sequence of *ecol* transformations (Figure 2b). More precisely, each vertex v_j of M^f corresponds with the vertex $v_{A^c(j)}$ in M^c where

$$A^c(j) = \begin{cases} j & , \ j \le m_0 + c \\ A^c(s_{j-m_0-1}) & , \ j > m_0 + c \end{cases} .$$

(In practice, this ancestor information A^c is gathered in a forward fashion as the mesh is refined.) This correspondence allows us to define a geomorph $M^G(\alpha)$ such that $M^G(0)$ looks like M^c and $M^G(1)$ equals M^f. We simply define $M^G(\alpha) = (K^f, V^G(\alpha))$ to have the connectivity of M^f and the vertex positions

$$\mathbf{v}_j^G(\alpha) = (\alpha)\mathbf{v}_j^f + (1-\alpha)\mathbf{v}_{A^c(j)}^c .$$

So far we have outlined the construction of geomorphs between PM meshes containing only position attributes. We can in fact construct geomorphs for meshes containing both discrete and scalar attributes.

Discrete attributes by their nature cannot be smoothly interpolated. Fortunately, these discrete attributes are associated with faces of the mesh, and the "geometric" geomorphs described above smoothly introduce faces. In particular, observe that the faces of M^c are a proper subset of the faces of M^f, and that those faces of M^f missing from M^c are invisible in $M^G(0)$ because they have been collapsed to degenerate (zero area) triangles. Other geomorphing schemes [10, 11, 17] define well-behaved (invertible) parametrizations between meshes at different levels of detail, but these do not permit the construction of geomorphs between meshes with different discrete attributes.

Scalar attributes defined on corners can be smoothly interpolated much like the vertex positions. There is a slight complication in that a corner (v, f) in a mesh M is not naturally associated with any "ancestor corner" in a coarser mesh M^c if f is not a face of M^c. We can still attempt to infer what attribute value (v, f) would have in M^c as follows. We examine the mesh M^{i+1} in which f is first introduced, locate a neighboring corner (v, f') in M^{i+1} whose attribute value is the same, and recursively backtrack from it to a corner in M^c. If there is no neighboring corner in M^{i+1} with an identical attribute value, then the corner (v, f) has no equivalent in M^c and we therefore keep its attribute value constant through the geomorph.

The interpolating function on the scalar attributes need not be linear; for instance, normals are best interpolated over the unit sphere, and colors may be interpolated in a color space other than RGB.

Figure 6 demonstrates a geomorph between two meshes M^{175} (500 faces) and M^{425} (1000 faces) retrieved from the PM representation of the mesh in Figure 5d.

3.3 Progressive transmission

Progressive meshes are a natural representation for progressive transmission. The compact mesh M^0 is transmitted first (using a conventional uni-resolution format), followed by the stream of $vsplit_i$ records. The receiving process incrementally rebuilds $\hat{M}$ as the records arrive, and animates the changing mesh. The changes to the mesh can be geomorphed to avoid visual discontinuities. The original mesh $\hat{M}$ is recovered exactly after all n records are received, since PM is a lossless representation.

The computation of the receiving process should be balanced between the reconstruction of $\hat{M}$ and interactive display. With a slow communication line, a simple strategy is to display the current mesh whenever the input buffer is found to be empty. With a fast communication line, we find that a good strategy is to display meshes whose complexities increase exponentially. (Similar issues arise in the display of images transmitted using progressive JPEG.)

3.4 Mesh compression

Even though the PM representation encodes both $\hat{M}$ and a continuous family of approximations, it is surprisingly space-efficient, for two reasons. First, the locations of the vertex split transformations can be encoded concisely. Instead of storing all three vertex indices (s_i, l_i, r_i) of $vsplit_i$, one need only store s_i and approximately 5 bits to select the remaining two vertices among those adjacent to v_{s_i}.[2] Second, because a vertex split has local effect, one can expect significant coherence in mesh attributes through each transformation. For instance, when vertex $v_{s_i}^i$ is split into $v_{s_i}^{i+1}$ and $v_{m_0+i+1}^{i+1}$, we can predict the positions $\mathbf{v}_{s_i}^{i+1}$ and $\mathbf{v}_{m_0+i+1}^{i+1}$ from $\mathbf{v}_{s_i}^i$, and use delta-encoding to reduce storage. Scalar attributes of corners in M^{i+1} can similarly be predicted from those in M^i. Finally, the material identifiers $d_{\{v_s, v_t, v_l\}}$ and $d_{\{v_t, v_s, v_r\}}$ of the new faces in mesh M^{i+1} can often be predicted from those of adjacent faces in M^i using only a few control bits.

As a result, the size of a carefully designed PM representation should be competitive with that obtained from methods for compressing uni-resolution meshes. Our current prototype implementation was not designed with this goal in mind. However, we analyze the compression of the connectivity K, and report results on the compression of the geometry V. In the following analysis, we assume for simplicity that $m_0 = 0$ since typically $m_0 \ll n$.

A common representation for the mesh connectivity K is to list the three vertex indices for each face. Since the number of vertices is n and the number of faces approximately $2n$, such a list requires $6\lceil \log_2(n) \rceil n$ bits of storage. Using a buffer of 2 vertices, *generalized triangle strip* representations reduce this number to about

[2] On average, v_{s_i} has 6 neighbors, and the number of permutations $P_2^6 = 30$ can be encoded in $\lceil \log_2(P_2^6) \rceil = 5$ bits.

$(\lceil \log_2(n) \rceil + 2k)n$ bits, where vertices are back-referenced once on average and $k \simeq 2$ bits capture the vertex replacement codes [6]. By increasing the vertex buffer size to 16, Deering's *generalized triangle mesh* representation [6] further reduces storage to about $(\frac{1}{8}\lceil \log_2(n) \rceil + 8)n$ bits. Turan [16] shows that planar graphs (and hence the connectivity of closed genus 0 meshes) can be encoded in $12n$ bits. Recent work by Taubin and Rossignac [15] addresses more general meshes. With the PM representation, each $vsplit_i$ requires specification of s_i and its two neighbors, for a total storage of about $(\lceil \log_2(n) \rceil + 5)n$ bits. Although not as concise as [6, 15], this is comparable to generalized triangle strips.

A traditional representation of the mesh geometry V requires storage of $3n$ coordinates, or $96n$ bits with IEEE single-precision floating point. Like Deering [6], we assume that these coordinates can be quantized to 16-bit fixed precision values without significant loss of visual quality, thus reducing storage to $48n$ bits. Deering is able to further compress this storage by delta-encoding the quantized coordinates and Huffman compressing the variable-length deltas. For 16-bit quantization, he reports storage of $35.8n$ bits, which includes both the deltas and the Huffman codes. Using a similar approach with the PM representation, we encode V in $31n$ to $50n$ bits as shown in Table 1. To obtain these results, we exploit a property of our optimization algorithm (Section 4.3): when considering the collapse of an edge $\{v_s, v_t\}$, it considers three starting points for the resulting vertex position $\mathbf{v}_n$: $\{\mathbf{v}_s, \mathbf{v}_t, \frac{\mathbf{v}_s+\mathbf{v}_t}{2}\}$. Depending on the starting point chosen, we delta-encode either $\{\mathbf{v}_s - \mathbf{v}_n, \mathbf{v}_t - \mathbf{v}_n\}$ or $\{\frac{\mathbf{v}_s+\mathbf{v}_t}{2} - \mathbf{v}_n, \frac{\mathbf{v}_t-\mathbf{v}_s}{2}\}$, and use separate Huffman tables for all four quantities.

To further improve compression, we could alter the construction algorithm to forego optimization and let $\mathbf{v}_n \in \{\mathbf{v}_s, \mathbf{v}_t, \frac{\mathbf{v}_s+\mathbf{v}_t}{2}\}$. This would degrade the accuracy of the approximating meshes somewhat, but allows encoding of V in $30n$ to $37n$ bits in our examples. Arithmetic coding [19] of delta lengths does not improve results significantly, reflecting the fact that the Huffman trees are well balanced. Further compression improvements may be achievable by adapting both the quantization level and the delta length models as functions of the *vsplit* record index i, since the magnitude of successive changes tends to decrease.

3.5 Selective refinement

The PM representation also supports selective refinement, whereby detail is added to the model only in desired areas. Let the application supply a callback function REFINE(v) that returns a Boolean value indicating whether the neighborhood of the mesh about v should be further refined. An initial mesh M^c is selectively refined by iterating through the list $\{vsplit_c, \ldots, vsplit_{n-1}\}$ as before, but only performing $vsplit_i(s_i, l_i, r_i, A_i)$ if

(1) all three vertices $\{v_{s_i}, v_{l_i}, v_{r_i}\}$ are present in the mesh, and

(2) REFINE(v_{s_i}) evaluates to TRUE.

(A vertex v_j is absent from the mesh if the prior vertex split that would have introduced it, $vsplit_{j-m_0-1}$, was not performed due to the above conditions.)

As an example, to obtain selective refinement of the model within a view frustum, REFINE(v) is defined to be TRUE if either v or any of its neighbors lies within the frustum. As seen in Figure 7a, condition (1) described above is suboptimal. The problem is that a vertex v_{s_i} within the frustum may fail to be split because its expected neighbor v_{l_i} lies just outside the frustum and was not previously created. The problem is remedied by using a less stringent version of condition (1). Let us define the *closest living ancestor* of a vertex v_j to be the vertex with index

$$A'(j) = \begin{cases} j & , \text{ if } v_j \text{ exists in the mesh} \\ A'(s_{j-m_0-1}) & , \text{ otherwise} \end{cases}$$

The new condition becomes:

(1') v_{s_i} is present in the mesh (i.e. $A'(s_i) = s_i$) and the vertices $v_{A'(l_i)}$ and $v_{A'(r_i)}$ are both adjacent to v_{s_i}.

As when constructing the geomorphs, the ancestor information A' is carried efficiently as the *vsplit* records are parsed. If conditions (1') and (2) are satisfied, $vsplit(s_i, A'(l_i), A'(r_i), A_i)$ is applied to the mesh. A mesh selectively refined with this new strategy is shown in Figure 7b. This same strategy was also used for Figure 10. Note that it is still possible to create geomorphs between M^c and selectively refined meshes thus created.

An interesting application of selective refinement is the transmission of view-dependent models over low-bandwidth communication lines. As the receiver's view changes over time, the sending process need only transmit those *vsplit* records for which REFINE evaluates to TRUE, and of those only the ones not previously transmitted.

4 PROGRESSIVE MESH CONSTRUCTION

The PM representation of an arbitrary mesh $\hat{M}$ requires a sequence of edge collapses transforming $\hat{M} = M^n$ into a base mesh M^0. The quality of the intermediate approximations $M^i, i < n$ depends largely on the algorithm for selecting which edges to collapse and what attributes to assign to the affected neighborhoods, for instance the positions $\mathbf{v}^i_{s_i}$.

There are many possible PM construction algorithms with varying trade-offs of speed and accuracy. At one extreme, a crude and fast scheme for selecting edge collapses is to choose them completely at random. (Some local conditions must be satisfied for an edge collapse to be legal, i.e. manifold preserving [9].) More sophisticated schemes can use heuristics to improve the edge selection strategy, for example the "distance to plane" metric of Schroeder et al. [14]. At the other extreme, one can attempt to find approximating meshes that are optimal with respect to some appearance metric, for instance the E_{dist} geometric metric of Hoppe et al. [9].

Since PM construction is a preprocess that can be performed offline, we chose to design a simplification procedure that invests some time in the selection of edge collapses. Our procedure is similar to the mesh optimization method introduced by Hoppe et al. [9], which is outlined briefly in Section 4.1. Section 4.2 presents an overview of our procedure, and Sections 4.3–4.6 present the details of our optimization scheme for preserving both the shape of the mesh and the scalar and discrete attributes which define its appearance.

4.1 Background: mesh optimization

The goal of mesh optimization [9] is to find a mesh $M = (K, V)$ that both accurately fits a set X of points $\mathbf{x}_i \in \mathbf{R}^3$ and has a small number of vertices. This problem is cast as minimization of an energy function

$$E(M) = E_{dist}(M) + E_{rep}(M) + E_{spring}(M) \ .$$

The first two terms correspond to the two goals of accuracy and conciseness: the *distance energy* term

$$E_{dist}(M) = \sum_i d^2(\mathbf{x}_i, \phi_V(|K|))$$

measures the total squared distance of the points from the mesh, and the *representation energy* term $E_{rep}(M) = c_{rep}m$ penalizes the number m of vertices in M. The third term, the *spring energy* $E_{spring}(M)$ is introduced to regularize the optimization problem. It corresponds to placing on each edge of the mesh a spring of rest length zero and tension κ:

$$E_{spring}(M) = \sum_{\{j,k\} \in K} \kappa \|\mathbf{v}_j - \mathbf{v}_k\|^2 \ .$$

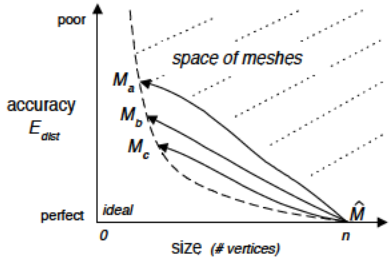


Figure 3: Illustration of the paths taken by mesh optimization using three different settings of c_{rep}.

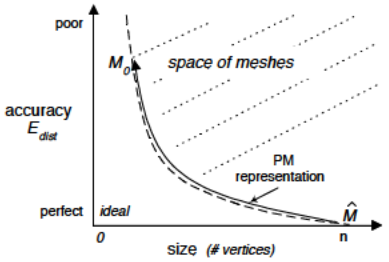


Figure 4: Illustration of the path taken by the new mesh simplification procedure in a graph plotting accuracy vs. mesh size.

The energy function $E(M)$ is minimized using a nested optimization method:

- *Outer loop*: The algorithm optimizes over K, the connectivity of the mesh, by randomly attempting a set of three possible mesh transformations: edge collapse, edge split, and edge swap. This set of transformations is complete, in the sense that any simplicial complex K of the same topological type as $\hat{K}$ can be reached through a sequence of these transformations. For each candidate mesh transformation, $K \rightarrow K'$, the continuous optimization described below computes $E_{K'}$, the minimum of E subject to the new connectivity K'. If $\Delta E = E_{K'} - E_K$ is found to be negative, the mesh transformation is applied (akin to a zero-temperature simulated annealing method).
- *Inner loop*: For each candidate mesh transformation, the algorithm computes $E_{K'} = \min_V E_{dist}(V) + E_{spring}(V)$ by optimizing over the vertex positions V. For the sake of efficiency, the algorithm in fact optimizes only one vertex position $\mathbf{v}_s$, and considers only the subset of points X that project onto the neighborhood affected by $K \rightarrow K'$. To avoid surface self-intersections, the edge collapse is disallowed if the maximum dihedral angle of edges in the resulting neighborhood exceeds some threshold.

Hoppe et al. [9] find that the regularizing spring energy term $E_{spring}(M)$ is most important in the early stages of the optimization, and achieve best results by repeatedly invoking the nested optimization method described above with a schedule of decreasing spring constants κ.

Mesh optimization is demonstrated to be an effective tool for mesh simplification. Given an initial mesh $\hat{M}$ to approximate, a dense set of points X is sampled both at the vertices of $\hat{M}$ and randomly over its faces. The optimization algorithm is then invoked with $\hat{M}$ as the starting mesh. Varying the setting of the representation constant c_{rep} results in optimized meshes with different trade-offs of accuracy and size. The paths taken by these optimizations are shown illustratively in Figure 3.

4.2 Overview of the simplification algorithm

As in mesh optimization [9], we also define an explicit energy metric $E(M)$ to measure the accuracy of simplified meshes $M = (K, V, D, S)$ with respect to the original $\hat{M}$, and we also modify the mesh M starting from $\hat{M}$ while minimizing $E(M)$.

Our energy metric has the following form:

$$E(M) = E_{dist}(M) + E_{spring}(M) + E_{scalar}(M) + E_{disc}(M) .$$

The first two terms, $E_{dist}(M)$ and $E_{spring}(M)$ are identical to those in [9]. The next two terms of $E(M)$ are added to preserve attributes associated with M: $E_{scalar}(M)$ measures the accuracy of its scalar attributes (Section 4.4), and $E_{disc}(M)$ measures the geometric accuracy of its discontinuity curves (Section 4.5). (To achieve scale invariance of the terms, the mesh is uniformly scaled to fit in a unit cube.)

Our scheme for optimizing over the connectivity K of the mesh is rather different from [9]. We have discovered that a mesh can be effectively simplified using edge collapse transformations alone. The edge swap and edge split transformations, useful in the context of surface reconstruction (which motivated [9]), are not essential for simplification. Although in principle our simplification algorithm can no longer traverse the entire space of meshes considered by mesh optimization, we find that the meshes generated by our algorithm are just as good. In fact, because of the priority queue approach described below, our meshes are usually better. Moreover, considering only edge collapses simplifies the implementation, improves performance, and most importantly, gives rise to the PM representation (Section 3).

Rather than randomly attempting mesh transformations as in [9], we place all (legal) candidate edge collapse transformations into a priority queue, where the priority of each transformation is its estimated energy cost ΔE. In each iteration, we perform the transformation at the front of the priority queue (with lowest ΔE), and recompute the priorities of edges in the neighborhood of this transformation. As a consequence, we eliminate the need for the awkward parameter c_{rep} as well as the energy term $E_{rep}(M)$. Instead, we can explicitly specify the number of faces desired in an optimized mesh. Also, a single run of the optimization can generate several such meshes. Indeed, it generates a continuous-resolution family of meshes, namely the PM representation of $\hat{M}$ (Figure 4).

For each edge collapse $K \rightarrow K'$, we compute its cost $\Delta E = E_{K'} - E_K$ by solving a continuous optimization

$$E_{K'} = \min_{V,S} E_{dist}(V) + E_{spring}(V) + E_{scalar}(V, S) + E_{disc}(V)$$

over both the vertex positions V and the scalar attributes S of the mesh with connectivity K'. This minimization is discussed in the next three sections.

4.3 Preserving surface geometry (E_{dist} + E_{spring})

As in [9], we "record" the geometry of the original mesh $\hat{M}$ by sampling from it a set of points X. At a minimum, we sample a point at each vertex of $\hat{M}$. If requested by the user, additional points are sampled randomly over the surface of $\hat{M}$. The energy terms $E_{dist}(M)$ and $E_{spring}(M)$ are defined as in Section 4.1.

For a mesh of fixed connectivity, our method for optimizing the vertex positions to minimize $E_{dist}(V)+E_{spring}(V)$ closely follows that of [9]. Evaluating $E_{dist}(V)$ involves computing the distance of each point $\mathbf{x}_i$ to the mesh. Each of these distances is itself a minimization problem

$$d^2(\mathbf{x}_i, \phi_V(|K|)) = \min_{\mathbf{b}_i \in |K|} \|\mathbf{x}_i - \phi_V(\mathbf{b}_i)\|^2 \qquad (1)$$

where the unknown $\mathbf{b}_i$ is the parametrization of the projection of $\mathbf{x}_i$ on the mesh. The nonlinear minimization of $E_{dist}(V) + E_{spring}(V)$ is performed using an iterative procedure alternating between two steps:

1. For fixed vertex positions V, compute the optimal parametrizations $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_{|X|}\}$ by projecting the points X onto the mesh.
2. For fixed parametrizations B, compute the optimal vertex positions V by solving a sparse linear least-squares problem.

As in [9], when considering $ecol(\{v_s, v_t\})$, we optimize only one vertex position, $\mathbf{v}_s^i$. We perform three different optimizations with different starting points, $\mathbf{v}_s^i = (1-\alpha)\mathbf{v}_s^{i+1} + (\alpha)\mathbf{v}_t^{i+1}$ for $\alpha = \{0, \frac{1}{2}, 1\}$, and accept the best one.

Instead of defining a global spring constant κ for E_{spring} as in [9], we adapt κ each time an edge collapse transformation is considered. Intuitively, the spring energy is most important when few points project onto a neighborhood of faces, since in this case finding the vertex positions minimizing $E_{dist}(V)$ may be an under-constrained problem. Thus, for each edge collapse transformation considered, we set κ as a function of the ratio of the number of points to the number of faces in the neighborhood.[3] With this adaptive scheme, the influence of $E_{spring}(M)$ decreases gradually and adaptively as the mesh is simplified, and we no longer require the expensive schedule of decreasing spring constants.

4.4 Preserving scalar attributes (E_{scalar})

As described in Section 2, we represent piecewise continuous scalar fields by defining scalar attributes S at the mesh corners. We now present our scheme for preserving these scalar fields through the simplification process. For exposition, we find it easier to first present the case of continuous scalar fields, in which the corner attributes at a vertex are identical. The generalization to piecewise continuous fields is discussed shortly.

Optimizing scalar attributes at vertices Let the original mesh $\hat{M}$ have at each vertex v_j not only a position $\mathbf{v}_j \in \mathbf{R}^3$ but also a scalar attribute $\underline{\mathbf{v}}_j \in \mathbf{R}^d$. To capture scalar attributes, we sample at each point $\mathbf{x}_i \in X$ the attribute value $\underline{\mathbf{x}}_i \in \mathbf{R}^d$. We would then like to generalize the distance metric E_{dist} to also measure the deviation of the sampled attribute values $\underline{X}$ from those of M.

One natural way to achieve this is to redefine the distance metric to measure distance in $\mathbf{R}^{3+d}$:

$$d^2((\mathbf{x}_i \;\; \underline{\mathbf{x}}_i), M(K, V, \underline{V})) = \min_{\mathbf{b}_i \in |K|} \left\| (\mathbf{x}_i \;\; \underline{\mathbf{x}}_i) - (\phi_V(\mathbf{b}_i) \;\; \phi_{\underline{V}}(\mathbf{b}_i)) \right\|^2 .$$

This new distance functional could be minimized using the iterative approach of Section 4.3. However, it would be expensive since finding the optimal parametrization $\mathbf{b}_i$ of each point $\mathbf{x}_i$ would require projection in $\mathbf{R}^{3+d}$, and would be non-intuitive since these parametrizations would not be geometrically based.

Instead we opted to determine the parametrizations $\mathbf{b}_i$ using only geometry with equation (1), and to introduce a separate energy term E_{scalar} to measure attribute deviation based on these parametrizations:

$$E_{scalar}(\underline{V}) = (c_{scalar})^2 \sum_i \left\| \underline{\mathbf{x}}_i - \phi_{\underline{V}}(\mathbf{b}_i) \right\|^2$$

where the constant c_{scalar} assigns a relative weight between the scalar attribute errors (E_{scalar}) and the geometric errors (E_{dist}).

Thus, to minimize $E(V, \underline{V}) = E_{dist}(V) + E_{spring}(V) + E_{scalar}(\underline{V})$, our algorithm first finds the vertex position $\mathbf{v}_s$ minimizing $E_{dist}(V) + E_{spring}(V)$ by alternately projecting the points onto the mesh (obtaining the parametrizations $\mathbf{b}_i$) and solving a linear least-squares problem (Section 4.1). Then, using those same parametrizations $\mathbf{b}_i$, it finds the vertex attribute $\underline{\mathbf{v}}_s$ minimizing E_{scalar} by solving a single linear least-squares problem. Hence introducing E_{scalar} into the optimization causes negligible performance overhead.

Since ΔE_{scalar} contributes to the estimated cost ΔE of an edge collapse, we obtain simplified meshes whose faces naturally adapt to the attribute fields, as shown in Figures 8 and 11.

Optimizing scalar attributes at corners Our scheme for optimizing the scalar corner attributes S is a straightforward generalization of the scheme just described. Instead of solving for a single unknown attribute value $\underline{\mathbf{v}}_s$, the algorithm partitions the corners around v_s into continuous sets (based on equivalence of corner attributes) and for each continuous set solves independently for its optimal attribute value.

Range constraints Some scalar attributes have constrained ranges. For instance, the components (r, g, b) of color are typically constrained to lie between 0 and 1. Least-squares optimization may yield color values outside this range. In these cases we clip the optimized values to the given range. For least-squares minimization of a Euclidean norm at a single vertex, this is in fact optimal.

Normals Surface normals (n_x, n_y, n_z) are typically constrained to have unit length, and thus their domain is non-Cartesian. Optimizing over normals would therefore require minimization of a nonlinear functional with nonlinear constraints. We decided to instead simply carry the normals through the simplification process. Specifically, we compute the new normals at vertex $v_{s_i}^i$ by interpolating between the normals at vertices $v_{s_i}^{i+1}$ and $v_{m_0+i+1}^{i+1}$ using the α value that resulted in the best vertex position $\mathbf{v}_{s_i}^i$ in Section 4.3. Fortunately, the absolute directions of normals are less visually important than their discontinuities, and we have a scheme for preserving such discontinuities, as described in the next section.

4.5 Preserving discontinuity curves (E_{disc})

Appearance attributes give rise to a set of discontinuity curves on the mesh, both from differences between discrete face attributes (e.g. material boundaries), and from differences between scalar corner attributes (e.g. creases and shadow boundaries). As these discontinuity curves form noticeable features, we have found it useful to preserve them both topologically and geometrically.

We can detect when a candidate edge collapse would modify the topology of the discontinuity curves using some simple tests on the presence of sharp edges in its neighborhood. Let $sharp(v_j, v_k)$ denote that an edge $\{v_j, v_k\}$ is sharp, and let $\#sharp(v_j)$ be the number of sharp edges adjacent to a vertex v_j. Then, referring to Figure 1, $ecol(\{v_s, v_t\})$ modifies the topology of discontinuity curves if either:

- $sharp(v_s, v_l)$ and $sharp(v_t, v_l)$, or
- $sharp(v_s, v_r)$ and $sharp(v_t, v_r)$, or
- $\#sharp(v_s) \geq 1$ and $\#sharp(v_t) \geq 1$ and not $sharp(v_s, v_t)$, or
- $\#sharp(v_s) \geq 3$ and $\#sharp(v_t) \geq 3$ and $sharp(v_s, v_t)$, or
- $sharp(v_s, v_t)$ and $\#sharp(v_s) = 1$ and $\#sharp(v_t) \neq 2$, or
- $sharp(v_s, v_t)$ and $\#sharp(v_t) = 1$ and $\#sharp(v_s) \neq 2$.

If an edge collapse would modify the topology of discontinuity curves, we either disallow it, or penalize it as discussed in Section 4.6.

To preserve the geometry of the discontinuity curves, we sample an additional set of points X_{disc} from the sharp edges of $\hat{M}$, and define an additional energy term E_{disc} equal to the total squared distances of each of these points to the discontinuity curve from which it was sampled. Thus E_{disc} is defined just like E_{dist}, except that the points X_{disc} are constrained to project onto a set of sharp edges in the mesh. In effect, we are solving a curve fitting problem embedded within the surface fitting problem. Since all boundaries of the surface are defined to be discontinuity curves, our procedure preserves bound-

[3] The neighborhood of an edge collapse transformation is the set of faces shown in Figure 1. Using C notation, we set $\kappa = r < 4 \;?\; 10^{-2} : r < 8 \;?\; 10^{-4} : 10^{-8}$ where r is the ratio of the number of points to faces in the neighborhood.

ary geometry more accurately than [9]. Figure 9 demonstrates the importance of using the E_{disc} energy term in preserving the material boundaries of a mesh with discrete face attributes.

4.6 Permitting changes to topology of discontinuity curves

Some meshes contain numerous discontinuity curves, and these curves may delimit features that are too small to be visible when viewed from a distance. In such cases we have found that strictly preserving the topology of the discontinuity curves unnecessarily curtails simplification. We have therefore adopted a hybrid strategy, which is to permit changes to the topology of the discontinuity curves, but to penalize such changes. When a candidate edge collapse $ecol(\{v_s, v_t\})$ changes the topology of the discontinuity curves, we add to its cost ΔE the value $|X_{disc,\{v_s,v_t\}}| \cdot \|\mathbf{v}_s - \mathbf{v}_t\|^2$ where $|X_{disc,\{v_s,v_t\}}|$ is the number of points of X_{disc} projecting onto $\{v_s, v_t\}$. That simple strategy, although ad hoc, has proven very effective. For example, it allows the dark gray window frames of the "cessna" (visible in Figure 9) to vanish in the simplified meshes (Figures 5a–c).

Table 1: Parameter settings and quantitative results.

Object	Original $\hat{M}$		Base M^0		User param.		$\lvert X_{disc}\rvert$	V	Time
	$m_0 + n$	#faces	m_0	#faces	$\lvert X\rvert-(m_0+n)$	c_{color}		$\frac{\text{bits}}{n}$	mins
cessna	6,795	13,546	97	150	100,000	-	46,811	46	23
terrain	33,847	66,960	3	1	0	-	3,796	46	16
mandrill	40,000	79,202	3	1	0	0.1	4,776	31	19
radiosity	78,923	150,983	1,192	1,191	200,000	0.01	74,316	37	106
fandisk	6,475	12,946	27	50	10,000	-	5,924	50	19

5 RESULTS

Table 1 shows, for the meshes in Figures 5–12, the number of vertices and faces in both $\hat{M}$ and M^0. In general, we let the simplification proceed until no more legal edge collapse transformations are possible. For the "cessna", we stopped at 150 faces to obtain a visually aesthetic base mesh. As indicated, the only user-specified parameters are the number of additional points (besides the $m_0 + n$ vertices of $\hat{M}$) sampled to increase fidelity, and the c_{scalar} constants relating the scalar attribute accuracies to the geometric accuracy. The only scalar attribute we optimized is color, and its c_{scalar} constant is denoted as c_{color}. The number $|X_{disc}|$ of points sampled from sharp edges is set automatically so that the densities of X and X_{disc} are proportional.[4] Execution times were obtained on a 150MHz Indigo2 with 128MB of memory.

Construction of the PM representation proceeds in three steps. First, as the simplification algorithm applies a sequence $ecol_{n-1} \ldots ecol_0$ of transformations to the original mesh, it writes to a file the sequence $vsplit_{n-1} \ldots vsplit_0$ of corresponding inverse transformations. When finished, the algorithm also writes the resulting base mesh M^0. Next, we reverse the order of the *vsplit* records. Finally, we renumber the vertices and faces of $(M^0, vsplit_0 \ldots vsplit_{n-1})$ to match the indexing scheme of Section 3.1 in order to obtain a concise format.

Figure 6 shows a single geomorph between two meshes M^{175} and M^{425} of a PM representation. For interactive LOD, it is useful to select a sequence of meshes from the PM representation, and to construct successive geomorphs between them. We have obtained good results by selecting meshes whose complexities grow exponentially, as in Figure 5. During execution, an application can adjust the granularity of these geomorphs by sampling additional meshes from the PM representation, or freeing some up.

In Figure 10, we selectively refined a terrain (grid of 181×187 vertices) using a new REFINE(v) function that keeps more detail near silhouette edges and near the viewer. More precisely, for the faces F_v adjacent to v, we compute the signed projected screen areas $\{a_f : f \in F_v\}$. We let REFINE(v) return TRUE if

(1) any face $f \in F_v$ lies within the view frustum, and either

(2a) the signs of a_f are not all equal (i.e. v lies near a silhouette edge) or

(2b) $\sum_{f \in F_v} a_f > thresh$ for a screen area threshold $thresh = 0.16^2$ (where total screen area is 1).

6 RELATED WORK

Mesh simplification methods A number of schemes construct a discrete sequence of approximating meshes by repeated application of a simplification procedure. Turk [17] sprinkles a set of points on a mesh, with density weighted by estimates of local curvature, and then retriangulates based on those points. Both Schroeder et al. [14] and Cohen et al. [4] iteratively remove vertices from the mesh and retriangulate the resulting holes. Cohen et al. are able to bound the maximum error of the approximation by restricting it to lie between two offset surfaces. Hoppe et al. [9] find accurate approximations through a general mesh optimization process (Section 4.1). Rossignac and Borrel [12] merge vertices of a model using spatial binning. A unique aspect of their approach is that the topological type of the model may change in the process. Their method is extremely fast, but since it ignores geometric qualities like curvature, the resulting approximations can be far from optimal. Some of the above methods [12, 17] permit the construction of geomorphs between successive simplified meshes.

Multiresolution analysis (MRA) Lounsbery et al. [10, 11] generalize the concept of multiresolution analysis to surfaces of arbitrary topological type. Eck et al. [7] describe how MRA can be applied to the approximation of an arbitrary mesh. Certain et al. [2] extend MRA to capture color, and present a multiresolution Web viewer supporting progressive transmission. MRA has many similarities with the PM scheme, since both store a simple base mesh together with a stream of detail records, and both produce a continuous-resolution representation. It is therefore worthwhile to highlight their differences:

Advantages of PM over MRA:

- MRA requires that the detail terms (wavelets) lie on a domain with subdivision connectivity, and as a result an arbitrary initial mesh $\hat{M}$ can only be recovered to within an ϵ tolerance. In contrast, the PM representation is lossless since $M^n = \hat{M}$.
- Because the approximating meshes $M^i, i < n$ in a PM may have arbitrary connectivity, they can be much better approximations than their MRA counterparts (Figure 12).
- The MRA representation cannot deal effectively with surface creases, unless those creases lie parametrically along edges of the base mesh (Figure 12). PM's can introduce surface creases anywhere and at any level of detail.
- PM's capture continuous, piecewise-continuous, and discrete appearance attributes. MRA schemes can represent discontinuous functions using a piecewise-constant basis (such as the Haar basis as used in [2, 13]), but the resulting approximations have too many discontinuities since none of the basis functions meet continuously. Also, it is not clear how MRA could be extended to capture discrete attributes.

[4] We set $|X_{disc}|$ such that $|X_{disc}|/perim = c(|X|/area)^{\frac{1}{2}}$ where $perim$ is the total length of all sharp edges in $\hat{M}$, $area$ is total area of all faces, and the constant $c = 4.0$ is chosen empirically.

Advantages of MRA over PM:

- The MRA framework provides a parametrization between meshes at various levels of detail, thus making possible multiresolution surface editing. PM's also offer such a parametrization, but it is not smooth, and therefore multiresolution editing may be non-intuitive.
- Eck et al. [7] construct MRA approximations with guaranteed maximum error bounds to $\hat{M}$. Our PM construction algorithm does not provide such bounds, although one could envision using simplification envelopes [4] to achieve this.
- MRA allows geometry and color to be compressed independently [2].

Other related work There has been relatively little work in simplifying arbitrary surfaces with functions defined over them. One special instance is image compression, since an image can be thought of as a set of scalar color functions defined on a quadrilateral surface. Another instance is the framework of Schröder and Sweldens [13] for simplifying functions defined over the sphere. The PM representation, like the MRA representation, is a generalization in that it supports surfaces of arbitrary topological type.

7 SUMMARY AND FUTURE WORK

We have introduced the progressive mesh representation and shown that it naturally supports geomorphs, progressive transmission, compression, and selective refinement. In addition, as a PM construction method, we have presented a new mesh simplification procedure designed to preserve not just the geometry of the original mesh, but also its overall appearance.

There are a number of avenues for future work, including:

- Development of an explicit metric and optimization scheme for preserving surface normals.
- Experimentation with PM editing.
- Representation of articulated or animated models.
- Application of the work to progressive subdivision surfaces.
- Progressive representation of more general simplicial complexes (not just 2-d manifolds).
- Addition of spatial data structures to permit efficient selective refinement.

We envision many practical applications for the PM representation, including streaming of 3D geometry over the Web, efficient storage formats, and continuous LOD in computer graphics applications. The representation may also have applications in finite element methods, as it can be used to generate coarse meshes for multigrid analysis.

ACKNOWLEDGMENTS

I wish to thank Viewpoint Datalabs for providing the "cessna" mesh, Pratt & Whitney for the gas turbine engine component ("fandisk"), Softimage for the "terrain" mesh, and especially Steve Drucker for creating several radiosity models using Lightscape. Thanks also to Michael Cohen, Steven "Shlomo" Gortler, and Jim Kajiya for their enthusiastic support, and to Rick Szeliski for helpful comments on the paper. Mark Kenworthy first coined the term "geomorph" in '92 to distinguish them from image morphs.

REFERENCES

[1] APPLE COMPUTER, INC. *3D graphics programming with QuickDraw 3D.* Addison Wesley, 1995.

[2] CERTAIN, A., POPOVIC, J., DUCHAMP, T., SALESIN, D., STUETZLE, W., AND DEROSE, T. Interactive multiresolution surface viewing. *Computer Graphics (SIGGRAPH '96 Proceedings)* (1996).

[3] CLARK, J. Hierarchical geometric models for visible surface algorithms. *Communications of the ACM 19,* 10 (Oct. 1976), 547–554.

[4] COHEN, J., VARSHNEY, A., MANOCHA, D., TURK, G., WEBER, H., AGARWAL, P., BROOKS, F., AND WRIGHT, W. Simplification envelopes. *Computer Graphics (SIGGRAPH '96 Proceedings)* (1996).

[5] CURLESS, B., AND LEVOY, M. A volumetric method for building complex models from range images. *Computer Graphics (SIGGRAPH '96 Proceedings)* (1996).

[6] DEERING, M. Geometry compression. *Computer Graphics (SIGGRAPH '95 Proceedings)* (1995), 13–20.

[7] ECK, M., DEROSE, T., DUCHAMP, T., HOPPE, H., LOUNSBERY, M., AND STUETZLE, W. Multiresolution analysis of arbitrary meshes. *Computer Graphics (SIGGRAPH '95 Proceedings)* (1995), 173–182.

[8] FUNKHOUSER, T., AND SÉQUIN, C. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. *Computer Graphics (SIGGRAPH '93 Proceedings)* (1995), 247–254.

[9] HOPPE, H., DEROSE, T., DUCHAMP, T., MCDONALD, J., AND STUETZLE, W. Mesh optimization. *Computer Graphics (SIGGRAPH '93 Proceedings)* (1993), 19–26.

[10] LOUNSBERY, J. M. *Multiresolution analysis for surfaces of arbitrary topological type.* PhD thesis, Dept. of Computer Science and Engineering, U. of Washington, 1994.

[11] LOUNSBERY, M., DEROSE, T., AND WARREN, J. Multiresolution analysis for surfaces of arbitrary topological type. Submitted for publication. (TR 93-10-05b, Dept. of Computer Science and Engineering, U. of Washington, January 1994.).

[12] ROSSIGNAC, J., AND BORREL, P. Multi-resolution 3D approximations for rendering complex scenes. In *Modeling in Computer Graphics*, B. Falcidieno and T. L. Kunii, Eds. Springer-Verlag, 1993, pp. 455–465.

[13] SCHRÖDER, P., AND SWELDENS, W. Spherical wavelets: Efficiently representing functions on the sphere. *Computer Graphics (SIGGRAPH '95 Proceedings)* (1995), 161–172.

[14] SCHROEDER, W., ZARGE, J., AND LORENSEN, W. Decimation of triangle meshes. *Computer Graphics (SIGGRAPH '92 Proceedings) 26,* 2 (1992), 65–70.

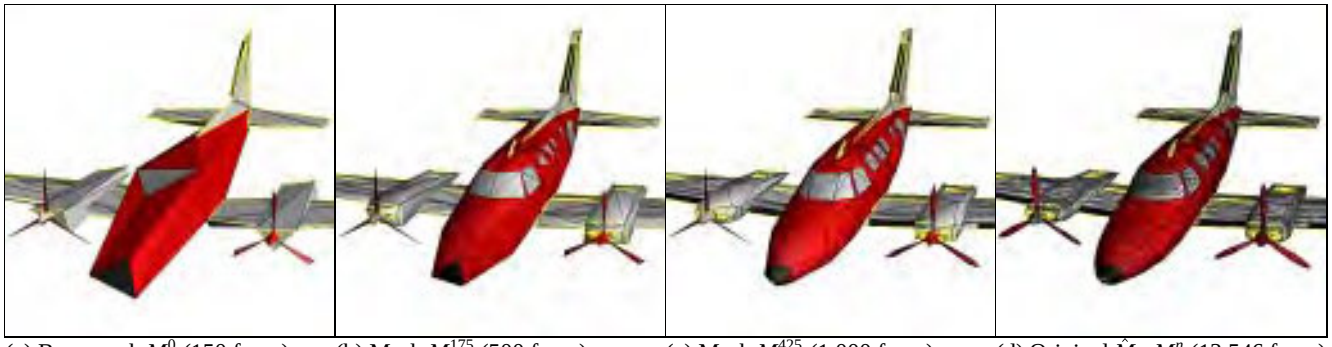
[15] TAUBIN, G., AND ROSSIGNAC, J. Geometry compression through topological surgery. Research Report RC-20340, IBM, January 1996.

[16] TURAN, G. Succinct representations of graphs. *Discrete Applied Mathematics 8* (1984), 289–294.

[17] TURK, G. Re-tiling polygonal surfaces. *Computer Graphics (SIGGRAPH '92 Proceedings) 26,* 2 (1992), 55–64.

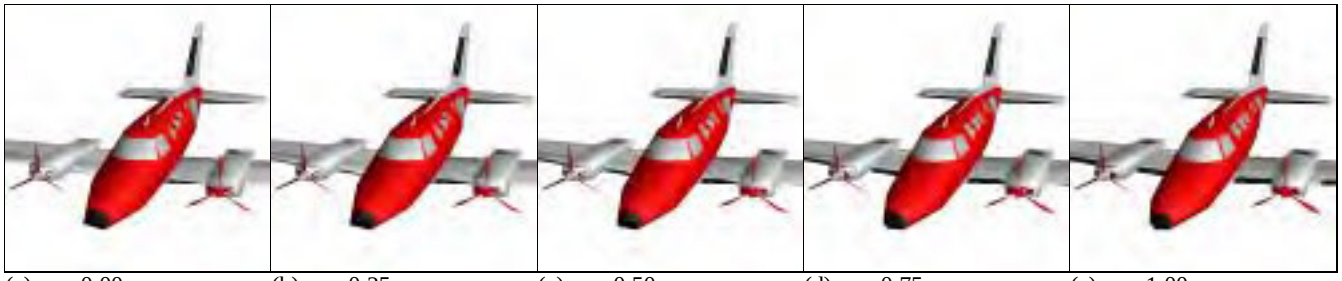
[18] UPSTILL, S. *The RenderMan Companion*. Addison-Wesley, 1990.

[19] WITTEN, I., NEAL, R., AND CLEARY, J. Arithmetic coding for data compression. *Communications of the ACM 30,* 6 (June 1987), 520–540.



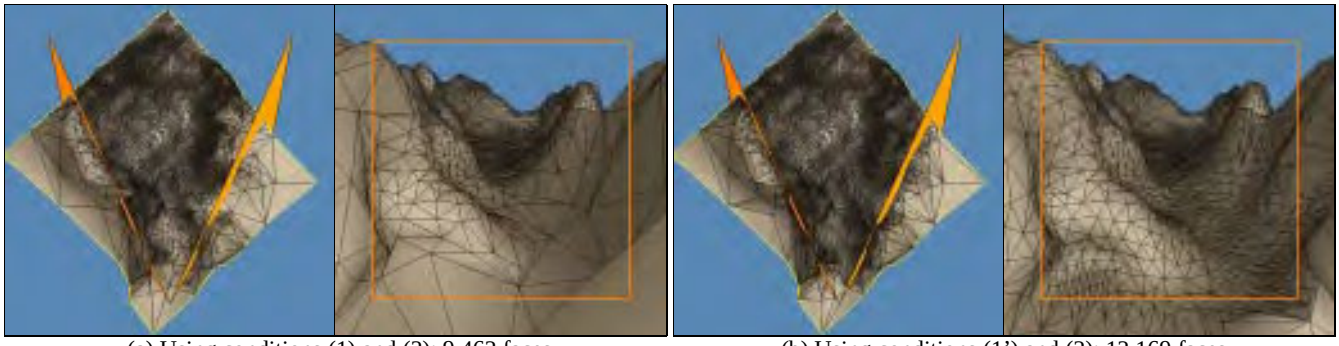
(a) Base mesh M^0 (150 faces) (b) Mesh M^{175} (500 faces) (c) Mesh M^{425} (1,000 faces) (d) Original $\hat{M} = M^n$ (13,546 faces)

Figure 5: The PM representation of an arbitrary mesh $\hat{M}$ captures a continuous-resolution family of approximating meshes $M^0 \ldots M^n = \hat{M}$.



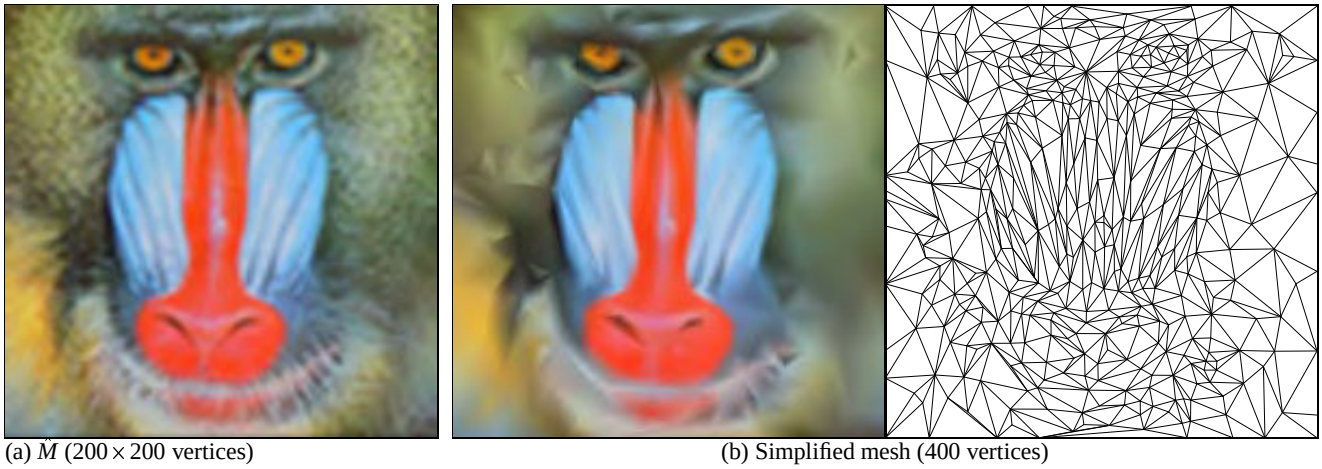
(a) $\alpha = 0.00$ (b) $\alpha = 0.25$ (c) $\alpha = 0.50$ (d) $\alpha = 0.75$ (e) $\alpha = 1.00$

Figure 6: Example of a geomorph $M^G(\alpha)$ defined between $M^G(0) \doteq M^{175}$ (with 500 faces) and $M^G(1) = M^{425}$ (with 1,000 faces).



(a) Using conditions (1) and (2); 9,462 faces (b) Using conditions (1') and (2); 12,169 faces

Figure 7: Example of selective refinement within the view frustum (indicated in orange).



(a) $\hat{M}$ (200×200 vertices) (b) Simplified mesh (400 vertices)

Figure 8: Demonstration of minimizing E_{scalar}: simplification of a mesh with trivial geometry (a square) but complex scalar attribute field. ($\hat{M}$ is a mesh with regular connectivity whose vertex colors correspond to the pixels of an image.)

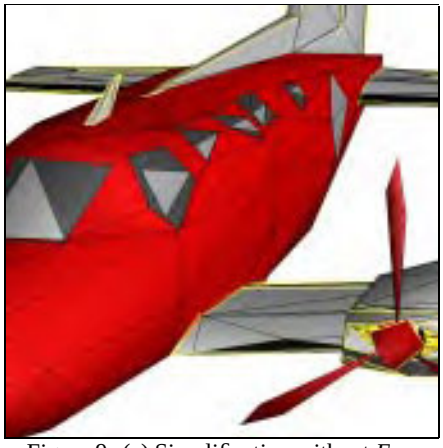


Figure 9: (a) Simplification without E_{disc}

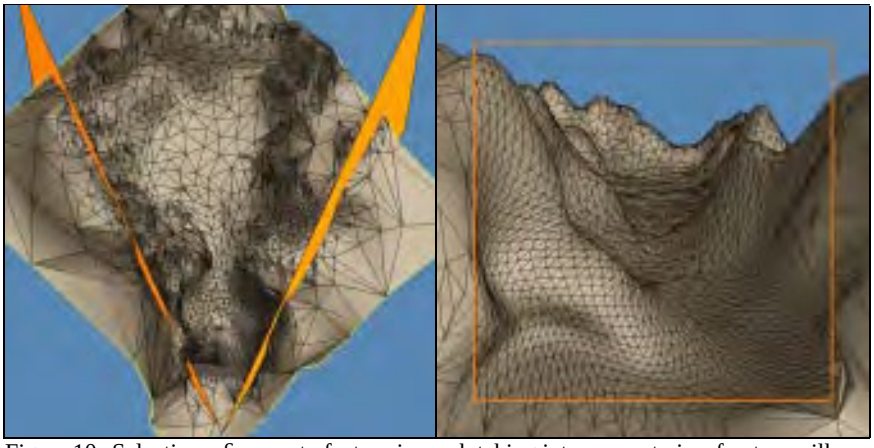
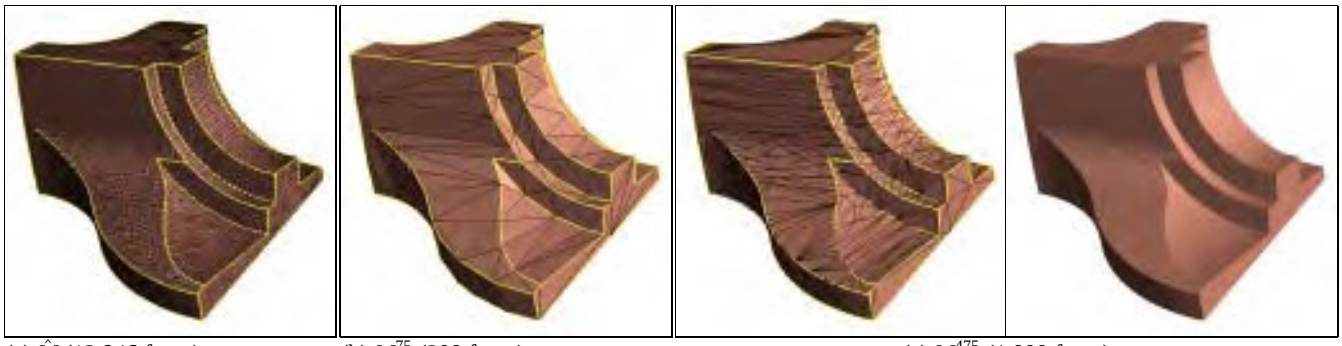
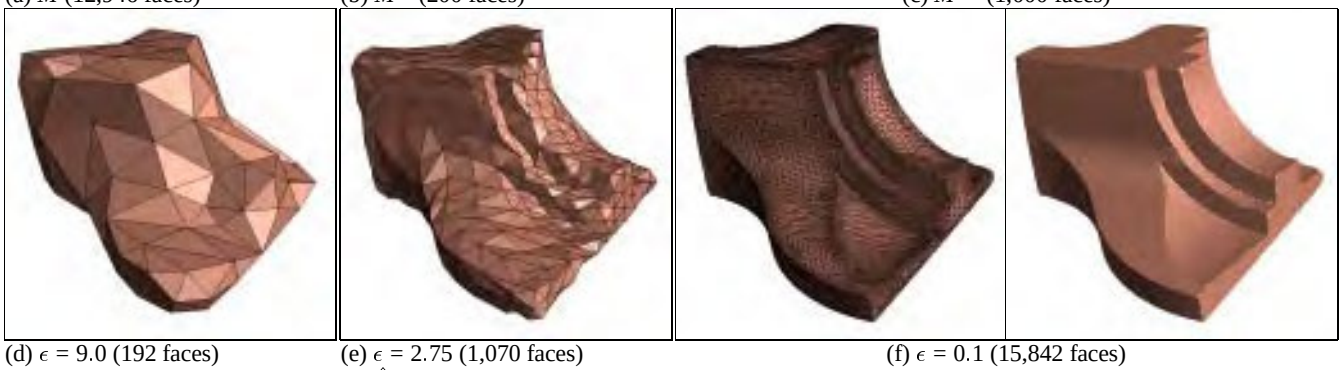


Figure 10: Selective refinement of a terrain mesh taking into account view frustum, silhouette regions, and projected screen size of faces (7,438 faces).

Figure 11: Simplification of a radiosity solution; left: original mesh (150,983 faces); right: simplified mesh (10,000 faces).



(a) $\hat{M}$ (12,946 faces) (b) M^{75} (200 faces) (c) M^{475} (1,000 faces)



(d) ϵ = 9.0 (192 faces) (e) ϵ = 2.75 (1,070 faces) (f) ϵ = 0.1 (15,842 faces)

Figure 12: Approximations of a mesh $\hat{M}$ using (b–c) the PM representation, and (d–f) the MRA scheme of Eck et al. [7]. As demonstrated, MRA cannot recover $\hat{M}$ exactly, cannot deal effectively with surface creases, and produces approximating meshes of inferior quality.