The Laplacian Pyramid as a Compact Image Code

PETER J. BURT, MEMBER, IEEE, AND EDWARD H. ADELSON

Abstract—**We describe a technique for image encoding in which local operators of many scales but identical shape serve as the basis functions. The representation differs from established techniques in that the code elements are localized in spatial frequency as well as in space.**

Pixel-to-pixel correlations are first removed by subtracting a low-pass filtered copy of the image from the image itself. The result is a net data compression since the difference, or error, image has low variance and entropy, and the low-pass filtered image may represented at reduced sample density. Further data compression is achieved by quantizing the difference image. These steps are then repeated to compress the low-pass image. Iteration of the process at appropriately expanded scales generates a pyramid data structure.

The encoding process is equivalent to sampling the image with Laplacian operators of many scales. Thus, the code tends to enhance salient image features. A further advantage of the present code is that it is well suited for many image analysis tasks as well as for image compression. Fast algorithms are described for coding and decoding.

INTRODUCTION

A COMMON characteristic of images is that neighboring pixels are highly correlated. To represent the image directly in terms of the pixel values is therefore inefficient: most of the encoded information is redundant. The first task in designing an efficient, compressed code is to find a representation which, in effect, decorrelates the image pixels. This has been achieved through predictive and through transform techniques (cf. [9], [10] for recent reviews).

In predictive coding, pixels are encoded sequentially in a raster format. However, prior to encoding each pixel, its value is predicted from previously coded pixels in the same and preceding raster lines. The predicted pixel value, which represents redundant information, is subtracted from the actual pixel value, and only the difference, or prediction error, is encoded. Since only previously encoded pixels are used in predicting each pixel's value, this process is said to be causal. Restriction to causal prediction facilitates decoding: to decode a given pixel, its predicted value is recomputed from already decoded neighboring pixels, and added to the stored prediction error.

Noncausal prediction, based on a symmetric neighborhood centered at each pixel, should yield more accurate prediction and, hence, greater data compression. However, this approach does not permit simple sequential coding. Noncausal approaches to image coding typically involve image transforms, or the solution to large sets of simultaneous equations. Rather than encoding pixels sequentially, such techniques encode them all at once, or by blocks.

Both predictive and transform techniques have advantages. The former is relatively simple to implement and is readily adapted to local image characteristics. The latter generally provides greater data compression, but at the expense of considerably greater computation.

Here we shall describe a new technique for removing image correlation which combines features of predictive and transform methods. The technique is noncausal, yet computations are relatively simple and local.

The predicted value for each pixel is computed as a local weighted average, using a unimodal Gaussian-like (or related trimodal) weighting function centered on the pixel itself. The predicted values for all pixels are first obtained by convolving this weighting function with the image. The result is a low-pass filtered image which is then subtracted from the original.

Let $g_0(ij)$ be the original image, and $g_1(ij)$ be the result of applying an appropriate low-pass filter to g_0. The prediction error $L_0(ij)$ is then given by

$$L_0(ij) = g_0(ij) - g_1(ij).$$

Rather than encode g_0, we encode L_0 and g_1. This results in a net data compression because a) L_0 is largely decorrelated, and so may be represented pixel by pixel with many fewer bits than g_0, and b) g_1 is low-pass filtered, and so may be encoded at a reduced sample rate.

Further data compression is achieved by iterating this process. The reduced image g_1 is itself low-pass filtered to yield g_2 and a second error image is obtained: $L_2(ij) = g_1(ij) - g_2(ij)$. By repeating these steps several times we obtain a sequence of two-dimensional arrays $L_0, L_1, L_2, \cdots, L_n$. In our implementation each is smaller than its predecessor by a scale factor of 1/2 due to reduced sample density. If we now imagine these arrays stacked one above another, the result is a tapering pyramid data structure. The value at each node in the pyramid represents the difference between two Gaussian-like or related functions convolved with the original image. The difference between these two functions is similar to the "Laplacian" operators commonly used in image enhancement [13]. Thus, we refer to the proposed compressed image representation as the Laplacian-pyramid code.

The coding scheme outlined above will be practical only if required filtering computations can be performed with an efficient algorithm. A suitable fast algorithm has recently been developed [2] and will be described in the next section.

Paper approved by the Editor for Signal Processing and Communication Electronics of the IEEE Communications Society for publication after presentation in part at the Conference on Pattern Recognition and Image Processing, Dallas, TX, 1981. Manuscript received April 12, 1982; revised July 21, 1982. This work was supported in part by the National Science Foundation under Grant MCS-79-23422 and by the National Institutes of Health under Postdoctoral Training Grant EY07003.

P. J. Burt is with the Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12181.

E. H. Adelson is with the RCA David Sarnoff Research Center, Princeton, NJ 08540.

0090-6778/83/0400-0532$01.00 © 1983 IEEE

THE GAUSSIAN PYRAMID

The first step in Laplacian pyramid coding is to low-pass filter the original image g_0 to obtain image g_1. We say that g_1 is a "reduced" version of g_0 in that both resolution and sample density are decreased. In a similar way we form g_2 as a reduced version of g_1, and so on. Filtering is performed by a procedure equivalent to convolution with one of a family of local, symmetric weighting functions. An important member of this family resembles the Gaussian probability distribution, so the sequence of images $g_0, g_1, \cdots, g_n$ is called the Gaussian pyramid.[1]

A fast algorithm for generating the Gaussian pyramid is given in the next subsection. In the following subsection we show how the same algorithm can be used to "expand" an image array by interpolating values between sample points. This device is used here to help visualize the contents of levels in the Gaussian pyramid, and in the next section to define the Laplacian pyramid.

Gaussian Pyramid Generation

Suppose the image is represented initially by the array g_0 which contains C columns and R rows of pixels. Each pixel represents the light intensity at the corresponding image point by an integer I between 0 and $K-1$. This image becomes the bottom or zero level of the Gaussian pyramid. Pyramid level 1 contains image g_1, which is a reduced or low-pass filtered version of g_0. Each value within level 1 is computed as a weighted average of values in level 0 within a 5-by-5 window. Each value within level 2, representing g_2, is then obtained from values within level 1 by applying the same pattern of weights. A graphical representation of this process in one dimension is given in Fig. 1. The size of the weighting function is not critical [2]. We have selected the 5-by-5 pattern because it provides adequate filtering at low computational cost.

The level-to-level averaging process is performed by the function REDUCE.

$$g_k = \text{REDUCE}\,(g_{k-1}) \tag{1}$$

which means, for levels $0 < l < N$ and nodes i, j, $0 \leqslant i < C_l$, $0 \leqslant j < R_l$,

$$g_l(i,j) = \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m,n) g_{l-1}(2i+m, 2j+n).$$

Here N refers to the number of levels in the pyramid, while C_l and R_l are the dimensions of the lth level. Note in Fig. 1 that the density of nodes is reduced by half in one dimension, or by a fourth in two dimensions from level to level. The dimensions of the original image are appropriate for pyramid construction if integers M_C, M_R, and N exist such that $C = M_C 2^N + 1$ and $R = M_R 2^N + 1$. (For example, if M_C and M_R are both 3 and N is 5, then images measure 97 by 97 pixels.) The dimensions of g_l are $C_l = M_C 2^{N-l} + 1$ and $R_l = M_R 2^{N-l} + 1$.

[1] We will refer to this set of low-pass filtered images as the Gaussian pyramid, even though in some cases it will be generated with a trimodal rather than unimodal weighting function.

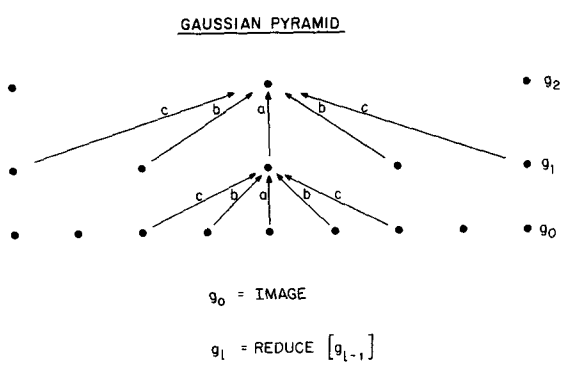


Fig. 1. A one-dimensional graphic representation of the process which generates a Gaussian pyramid. Each row of dots represents nodes within a level of the pyramid. The value of each node in the zero level is just the gray level of a corresponding image pixel. The value of each node in a high level is the weighted average of node values in the next lower level. Note that node spacing doubles from level to level, while the same weighting pattern or "generating kernel" is used to generate all levels.

The Generating Kernel

Note that the same 5-by-5 pattern of weights w is used to generate each pyramid array from its predecessor. This weighting pattern, called the generating kernel, is chosen subject to certain constraints [2]. For simplicity we make w separable:

$$w(m,n) = \hat{w}(m)\hat{w}(n).$$

The one-dimensional, length 5, function $\hat{w}$ is normalized

$$\sum_{m=-2}^{2} \hat{w}(m) = 1$$

and symmetric

$$\hat{w}(i) = \hat{w}(-i) \qquad \text{for } i = 0,\ 1, 2.$$

An additional constraint is called equal contribution. This stipulates that all nodes at a given level must contribute the same total weight (=1/4) to nodes at the next higher level. Let $\hat{w}(0) = a$, $\hat{w}(-1) = \hat{w}(1) = b$, and $\hat{w}(-2) = \hat{w}(2) = c$. In this case equal contribution requires that $a + 2c = 2b$. These three constraints are satisfied when

$$\hat{w}(0) = a$$

$$\hat{w}(-1) = \hat{w}(1) = 1/4$$

$$\hat{w}(-2) = \hat{w}(2) = 1/4 - a/2.$$

Equivalent Weighting Functions

Iterative pyramid generation is equivalent to convolving the image g_0 with a set of "equivalent weighting functions" h_l:

$$g_l = h_l \oplus g_0$$

or

$$g_l(i,j) = \sum_{m=-M_l}^{M_l} \sum_{n=-M_l}^{M_l} h_l(m,n) g_0(i2^l + m \cdot j2^l + n).$$

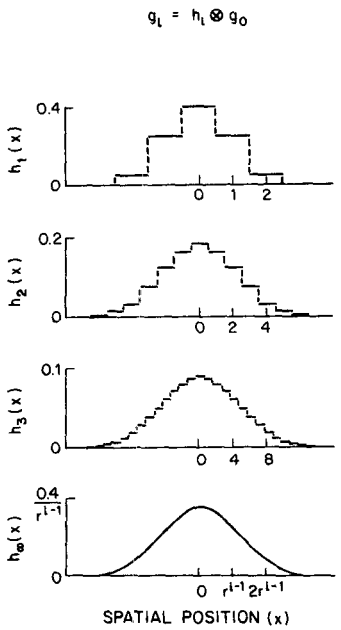


Fig. 2. The equivalent weighting functions $h_l(x)$ for nodes in levels 1, 2, 3, and infinity of the Gaussian pyramid. Note that axis scales have been adjusted by factors of 2 to aid comparison. Here the parameter a of the generating kernel is 0.4, and the resulting equivalent weighting functions closely resemble the Gaussian probability density functions.

The size M_l of the equivalent weighting function doubles from one level to the next, as does the distance between samples.

Equivalent weighting functions for Gaussian-pyramid levels 1, 2, and 3 are shown in Fig. 2. In this case $a = 0.4$. The shape of the equivalent function converges rapidly to a characteristic form with successively higher levels of the pyramid, so that only its scale changes. However, this shape does depend on the choice of a in the generating kernel. Characteristic shapes for four choices of a are shown in Fig. 3. Note that the equivalent weighting functions are particularly Gaussian-like when $a = 0.4$. When $a = 0.5$ the shape is triangular; when $a = 0.3$ it is flatter and broader than a Gaussian. With $a = 0.6$ the central positive mode is sharply peaked, and is flanked by small negative lobes.

Fast Filter

The effect of convolving an image with one of the equivalent weighting functions h_l is to blur, or low-pass filter, the image. The pyramid algorithm reduces the filter band limit by an octave from level to level, and reduces the sample interval by the same factor. This is a very fast algorithm, requiring fewer computational steps to compute a set of filtered images than are required by the fast Fourier transform to compute a single filtered image [2].

Example: Fig. 4 illustrates the contents of a Gaussian pyramid generated with $a = 0.4$. The original image, on the far left, measures 257 by 257. This becomes level 0 on the pyramid. Each higher level array is roughly half as large in each dimension as its predecessor, due to reduced sample density.

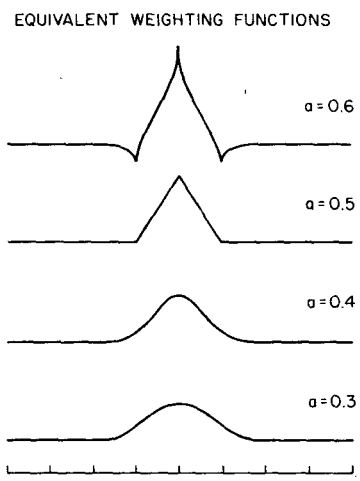


Fig. 3. The shape of the equivalent weighting function depends on the choice of parameter a. For $a = 0.5$, the function is triangular; for $a = 0.4$ it is Gaussian-like, and for $a = 0.3$ it is broader than Gaussian. For $a = 0.6$ the function is trimodal.

Gaussian Pyramid Interpolation

We now define a function EXPAND as the reverse of REDUCE. Its effect is to expand an $(M + 1)$-by-$(N + 1)$ array into a $(2M + 1)$-by-$(2N + 1)$ array by interpolating new node values between the given values. Thus, EXPAND applied to array g_l of the Gaussian pyramid would yield an array $g_{l,1}$ which is the same size as g_{l-1}.

Let $g_{l,n}$ be the result of expanding g_l n times. Then

$$g_{l,0} = g_l$$

and

$$g_{l,n} = \text{EXPAND}\,(g_{l,n-1}).$$

By EXPAND we mean, for levels $0 < l \leq N$ and $0 \leq n$ and nodes i, j, $0 \leq i < C_{l-n}$, $0 \leq j < R_{l-n}$,

$$g_{l,n}(ij) = 4 \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m,n) \cdot g_{l,n-1}\left(\frac{i-m}{2}, \frac{j-n}{2}\right). \tag{2}$$

Only terms for which $(i - m)/2$ and $(j - n)/2$ are integers are included in this sum.

If we apply EXPAND l times to image g_l, we obtain $g_{l,l}$, which is the same size as the original image g_0. Although full expansion will not be used in image coding, we will use it to help visualize the contents of various arrays within pyramid structures. The top row of Fig. 5 shows image $g_{0,0}$, $g_{1,1}$, $g_{2,2}$, ··· obtained by expanding levels of the pyramid in Fig. 4. The low-pass filter effect of the Gaussian pyramid is now shown clearly.

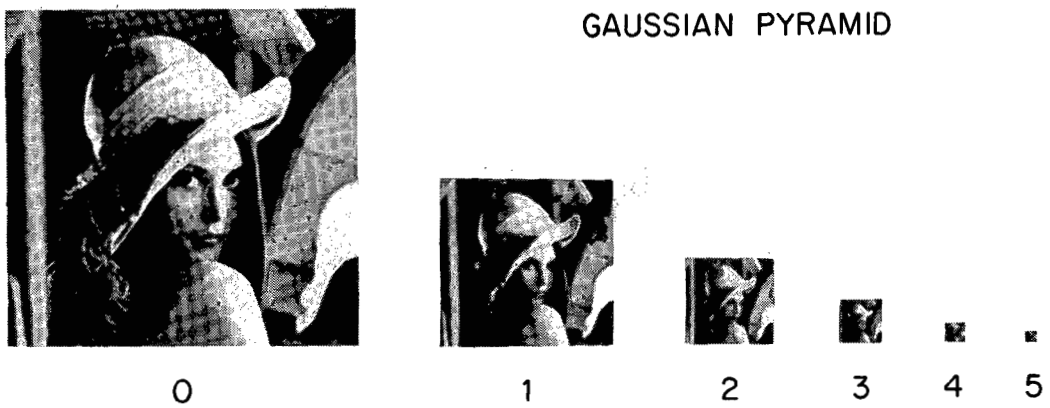


Fig. 4. First six levels of the Gaussian pyramid for the "Lady" image. The original image, level 0, measures 257 by 257 pixels, and each higher level array is roughly half the dimensions of its predecessor. Thus, level 5 measures just 9 by 9 pixels.

THE LAPLACIAN PYRAMID

Recall that our purpose for constructing the reduced image g_1 is that it may serve as a prediction for pixel values in the original image g_0. To obtain a compressed representation, we encode the error image which remains when an expanded g_1 is subtracted from g_0. This image becomes the bottom level of the Laplacian pyramid. The next level is generated by encoding g_1 in the same way. We now give a formal definition for the Laplacian pyramid, and examine its properties.

Laplacian Pyramid Generation

The Laplacian pyramid is a sequence of error images L_0, L_1, $\cdots$, L_N. Each is the difference between two levels of the Gaussian pyramid. Thus, for $0 \leqslant l < N$,

$$L_l = g_l - \text{EXPAND}\,(g_{l+1})$$
$$= g_l - g_{l+1,1}. \tag{3}$$

Since there is no image g_{N+1} to serve as the prediction image for g_N, we say $L_N = g_N$.

Equivalent Weighting Functions

The value at each node in the Laplacian pyramid is the difference between the convolutions of two equivalent weighting functions h_l, h_{l+1} with the original image. Again, this is similar to convolving an appropriately scaled Laplacian weighting function with the image. The node value could have been obtained directly by applying this operator, although at considerably greater computational cost.

Just as we may view the Gaussian pyramid as a set of lowpass filtered copies of the original image, we may view the Laplacian pyramid as a set of bandpass filtered copies of the image. The scale of the Laplacian operator doubles from level to level of the pyramid, while the center frequency of the passband is reduced by an octave.

In order to illustrate the contents of the Laplacian pyramid, it is helpful to interpolate between sample points. This may be done within the pyramid structure by Gaussian interpolation. Let $L_{l,n}$ be the result of expanding L_l n times using (2). Then, $L_{l,l}$ is the size of the original image.

The expanded Laplacian pyramid levels for the "Lady" image of Fig. 4 are shown in the bottom row of Fig. 5. Note that image features such as edges and bars appear enhanced in the Laplacian pyramid. Enhanced features are segregated by size: fine details are prominent in $L_{0,0}$, while progressively coarser features are prominent in the higher level images.

Decoding

It can be shown that the original image can be recovered exactly by expanding, then summing all the levels of the Laplacian pyramid:

$$g_0 = \sum_{l=0}^{N} L_{l,l}.$$

A more efficient procedure is to expand L_N once and add it to L_{N-1}, then expand this image once and add it to L_{N-2}, and so on until level 0 is reached and g_0 is recovered. This procedure simply reverses the steps in Laplacian pyramid generation. From (3) we see that

$$g_N = L_N \tag{4}$$

and for $l = N-1, N-2, \cdots, 0$,

$$g_l = L_l + \text{EXPAND}\,(g_{l+1}).$$

Entropy

If we assume that the pixel values of an image representation are statistically independent, then the minimum number of bits per pixel required to exactly encode the image is given by the entropy of the pixel value distribution. This optimum may be approached in practice through techniques such as variable length coding.

The histogram of pixel values for the "Lady" image is shown in Fig. 6(a). If we let the observed frequency of occurrence $f(i)$ of each gray level i be an estimate of its probability of occurrence in this and other similar images, then the entropy

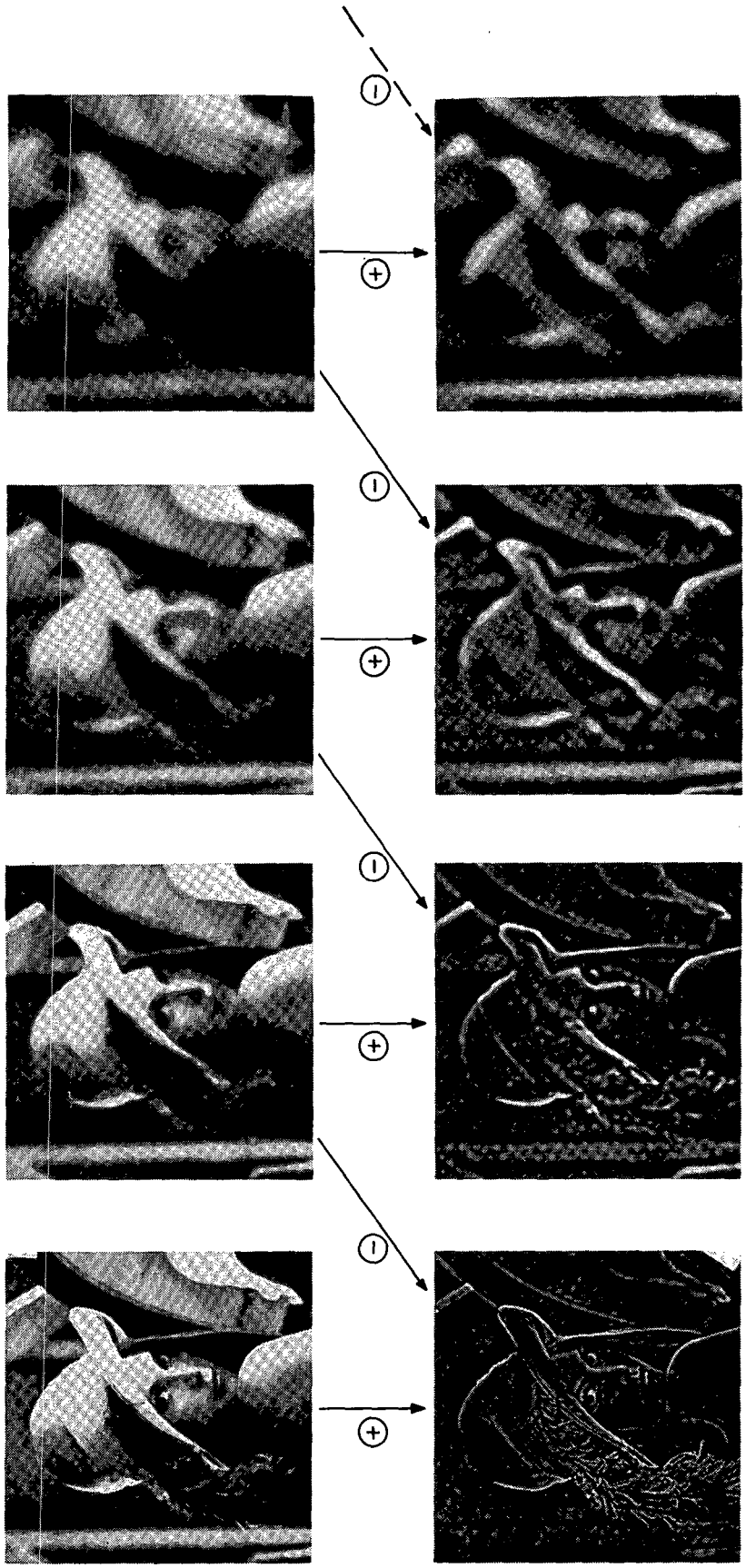


Fig. 5. First four levels of the Gaussian and Laplacian pyramids. Gaussian images, upper row, were obtained by expanding pyramid arrays (Fig. 4) through Gaussian interpolation. Each level of the Laplacian pyramid is the difference between the corresponding and next higher levels of the Gaussian pyramid.

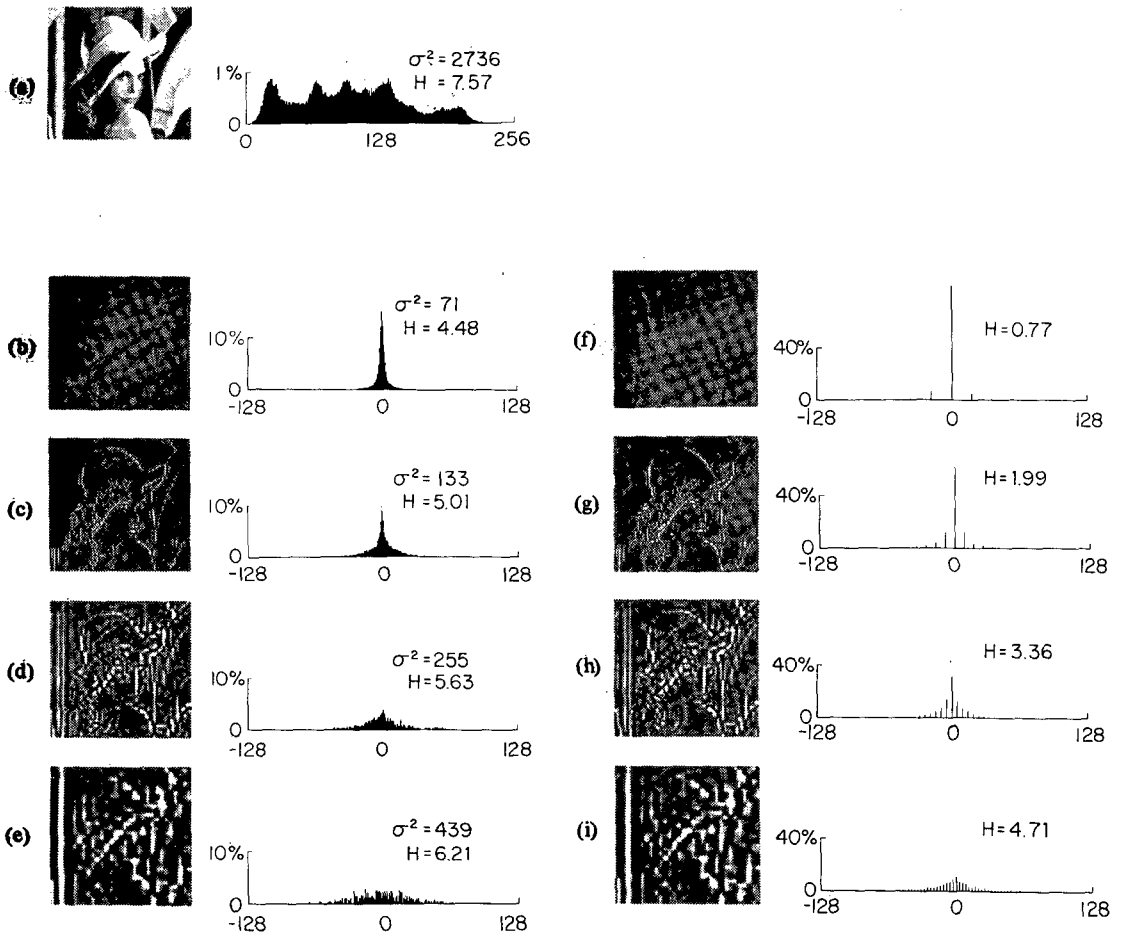


Fig. 6. The distribution of pixel gray level values at various stages of the encoding process. The histogram of the original image is given in (a). (b)-(e) give histograms for levels 0-3 of the Laplacian pyramid with generating parameter $a = 0.6$. Histograms following quantization at each level are shown in (f)-(i). Note that pixel values in the Laplacian pyramid are concentrated near zero, permitting data compression through shortened and variable length code words. Substantial further reduction is realized through quantization (particularly at low pyramid levels) and reduced sample density (particularly at high pyramid levels).

is given by

$$H = -\sum_{i=0}^{255} f(i) \log_2 f(i).$$

The maximum entropy would be 8 in this case since the image is initially represented at 256 gray levels, and would be obtained when all gray levels were equally likely. The actual entropy estimate for "Lady" is slightly less than this, at 7.57.

The technique of subtracting a predicted value from each image pixel, as in the Laplacian pyramid, removes much of the pixel-to-pixel correlation. Decorrelation also results in a concentration of pixel values around zero, and, therefore, in reduced variance and entropy. The degree to which these measures are reduced depends on the value of the parameter "a" used in pyramid generation (see Fig. 7). We found that the greatest reduction was obtained for $a = 0.6$ in our examples. Levels of the Gaussian pyramid appeared "crisper" when generated with this value of a than when generated with a smaller value such as 0.4, which yields more Guassian-like equivalent weighting functions. Thus, the selection $a = 0.6$ had perceptual as well as computational advantages. The first four levels of the corresponding Laplacian pyramid and their histograms are shown in Fig. 6(b)-(e). Variance (σ^2) and entropy (H) are also shown for each level. These quantities generally are found to increase from level to level, as in this example.

QUANTIZATION

Entropy can be substantially reduced by quantizing the pixel values in each level of the Laplacian pyramid. This introduces quantization errors, but through the proper choice of the number and distribution of quantization levels, the degradation may be made almost imperceptible to human observers. We illustrate this procedure with uniform quantization. The range of pixel values is divided into bins of size n, and the quantized value $C_l(i, j)$ for pixel $L_l(i, j)$ is just the middle

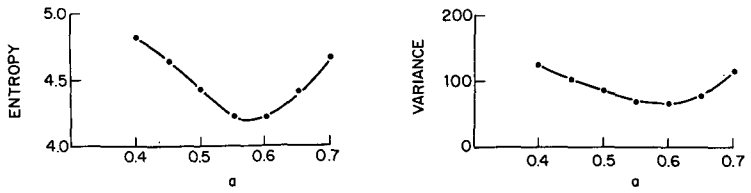


Fig. 7. Entropy and variance of pixel values in Laplacian pyramid level 0 as a function of the parameter "a" for the "Lady" image. Greatest reduction is obtained for $a \cong 0.6$. This estimate of the optimal "a" was also obtained at other pyramid levels and for other images.

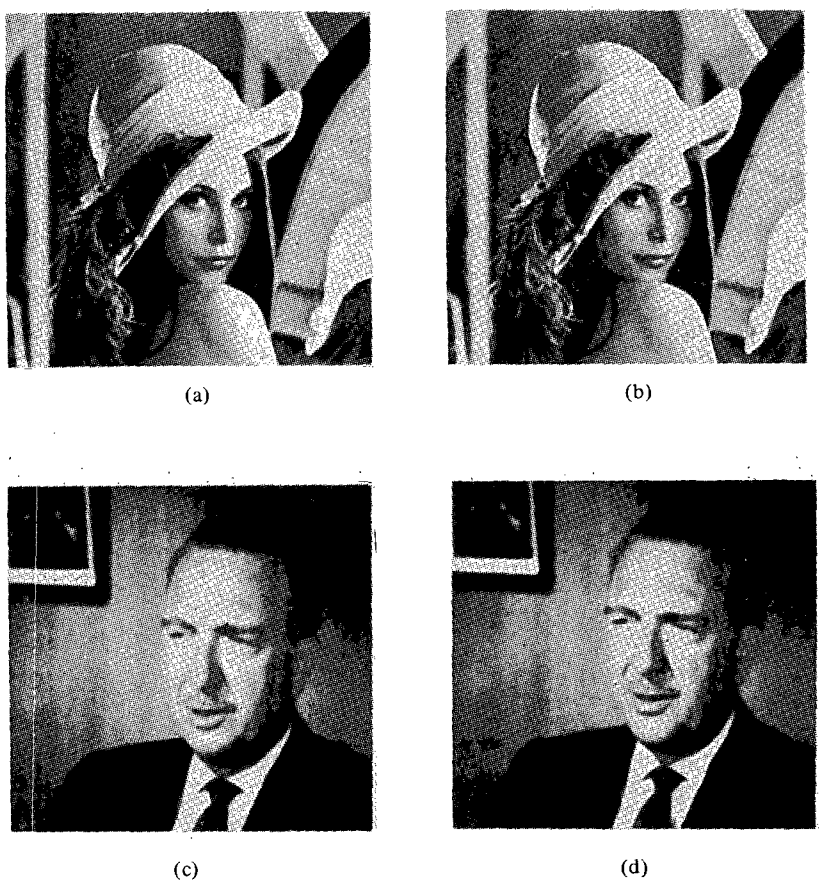


Fig. 8. Examples of image data compression using the Laplacian pyramid code. (a) and (c) give the original "Lady" and "Walter" images, while (b) and (d) give their encoded versions. The data rates are 1.58 and 0.73 bits/pixel for "Lady" and "Walter," respectively. The corresponding mean square errors were 0.88 percent and 0.43 percent, respectively.

value of the bin which contains $L_l(i,j)$:

$$C_l(i,j) = mn \quad \text{if } (m - 1/2)n < L_l(i,j) \leqslant (m + 1/2)n. \quad (5)$$

The quantized image is reconstructed through the expand and sum procedure (4) using C values in the place of L values.

Results of quantizing the "Lady" image are shown in Fig. 6(f)-(i). The bin size for each level was chosen by increasing n until degradation was just perceptible when viewed from a distance of approximately five times the image width (pixel-pixel separation $\cong$ 3 min arc). Note that bin size becomes smaller at higher levels (lower spatial frequencies). Bin size at a given pyramid level reflects the sensitivity of the human observer to contrast errors within the spatial frequency bands represented at that level. Humans are fairly sensitive to contrast perturbations at low and medium spatial frequencies, but relatively insensitive to such perturbations at high spatial frequencies [3], [4], [7].

This increased observer sensitivity along with the increased data variance noted above means that more quantization levels must be used at high pyramid levels than at low levels. Fortunately, these pixels contribute little to the overall bit rate for the image, due to their low sample density. The low-level (high-frequency) pixels, which are densely sampled, can be coarsely quantized (cf. [6], [11], [12]).

RESULTS

The final result of encoding, quantization, and reconstruction are shown in Fig. 8. The original "Lady" image is shown in Fig. 8(a); the encoded version, at 1.58 bits/pixel, is shown in Fig. 8(b). We assume that variable-length code words are used to take advantage of the nonuniform distribution of

node values, so the bit rate for a given pyramid level is its estimated entropy times its sample density, and the bit rate for the image is the sum of that for all levels. The same procedure was performed on the "Walter" image; the original is shown in Fig. 8(c). while the version encoded at 0.73 bits/pixel is shown in Fig. 8(d). In both cases, the encoded images are almost indistinguishable from the originals under viewing conditions as stated above.

PROGRESSIVE TRANSMISSION

It should also be observed that the Laplacian pyramid code is particularly well suited for progressive image transmission. In this type of transmission a coarse rendition of the image is sent first to give the receiver an early impression of image content, then subsequent transmission provides image detail of progressively finer resolution [5]. The observer may terminate transmission of an image as soon as its contents are recognized, or as soon as it becomes evident that the image will not be of interest. To achieve progressive transmission, the topmost level of the pyramid code is sent first, and expanded in the receiving pyramid to form an initial, very coarse image. The next lower level is then transmitted, expanded, and added to the first, and so on. At the receiving end, the initial image appears very blurry, but then comes steadily into "focus." This progression is illustrated in Fig. 9, from left to right. Note that while 1.58 bits are required for each pixel of the full transmission (rightmost image), about half of these, or 0.81 bits, are needed for each pixel for the previous image (second from right, Fig. 9), and 0.31 for the image previous to that (third from right).

SUMMARY AND CONCLUSION

The Laplacian pyramid is a versatile data structure with many attractive features for image processing. It represents an image as a series of quasi-bandpassed images, each sampled at successively sparser densities. The resulting code elements, which form a self-similar structure, are localized in both space and spatial frequency. By appropriately choosing the parameters of the encoding and quantizing scheme, one can substantially reduce the entropy in the representation, and simultaneously stay within the distortion limits imposed by the sensitivity of the human visual system.

Fig. 10 summarizes the steps in Laplacian pyramid coding. The first step, shown on the far left, is bottom-up construction of the Gaussian pyramid images $g_0, g_1, \cdots, g_N$ [see (1)]. The Laplacian pyramid images $L_0, L_1, \cdots, L_N$ are then obtained as the difference between successive Gaussian levels [see (3)]. These are quantized to yield the compressed code represented by the pyramid of values $C_l(ij)$ [see (5)]. Finally, image reconstruction follows an expand-and-sum procedure [see (4)] using C values in the place of L values. Here we designate the reconstructed image by r_0.

It should also be observed that the Laplacian pyramid encoding scheme requires relatively simple computations. The computations are local and may be performed in parallel, and the same computations are iterated to build each pyramid level from its predecessors. We may envision performing Lapla-

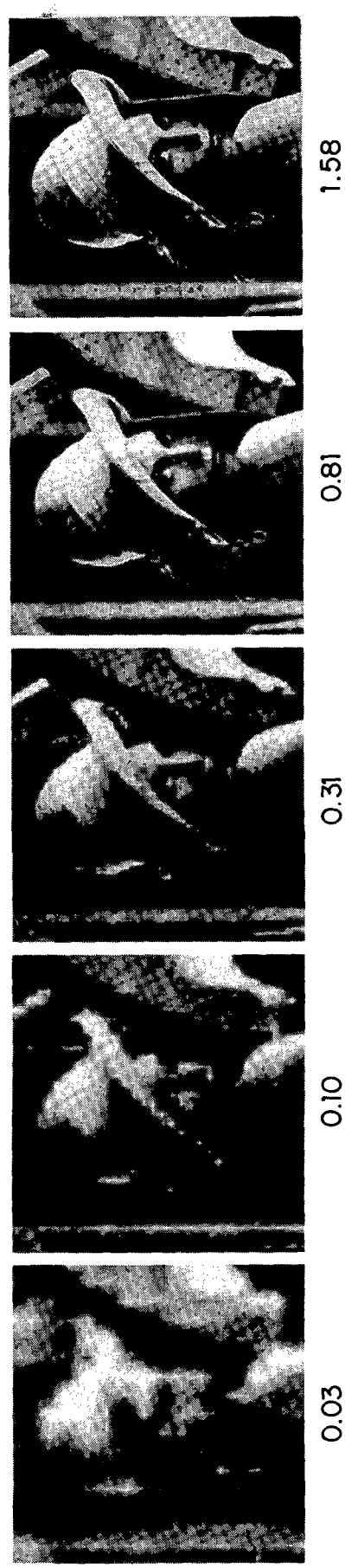


Fig. 9. Laplacian pyramid code applied to progressive image transmission. High levels of the pyramid are transmitted first to give the receiver a quick but very coarse rendition of the image. The receiver's image is then progressively refined by adding successively lower pyramid levels as these are transmitted. In the example shown here, the leftmost figure shows reconstruction using pyramid levels 4–8, or just 0.03 bits/pixel. The following four figures show the reconstruction after pyramid levels 3, 2, 1, and 0 have been added. The cumulative data rates are shown under each figures in bits per pixel.

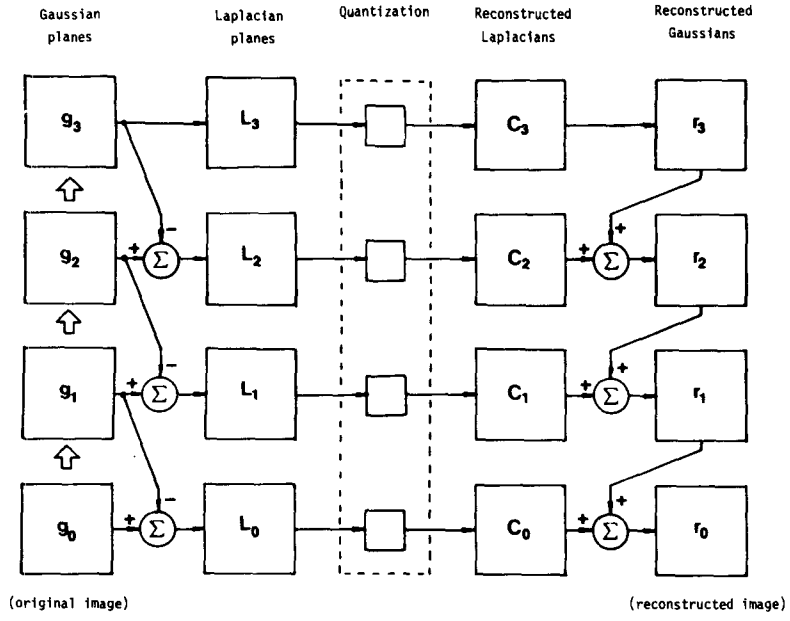


Fig. 10. A summary of the steps in Laplacian pyramid coding and decoding. First, the original image g_0 (lower left) is used to generate Gaussian pyramid levels $g_1, g_2, \cdots$ through repeated local averaging. Levels of the Laplacian pyramid $L_0, L_1, \cdots$ are then computed as the differences between adjacent Gaussian levels. Laplacian pyramid elements are quantized to yield the Laplacian pyramid code $C_0, C_1, C_2, \cdots$. Finally, a reconstructed image r_0 is generated by summing levels of the code pyramid.

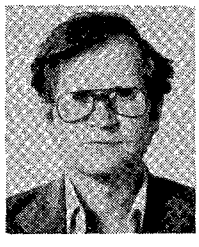
cian coding and decoding in real time using array processors and a pipeline architecture.

An additional benefit, previously noted, is that in computing the Laplacian pyramid, one automatically has access to quasi-bandpass copies of the image. In this representation, image features of various sizes are enhanced and are directly available for various image processing (e.g., [1]) and pattern recognition tasks.

REFERENCES

[1] K. D. Baker and G. D. Sullivan, "Multiple bandpass filters in image processing," *Proc. IEE*, vol. 127, pp. 173–184, 1980.
[2] P. J. Burt, "Fast filter transforms for image processing," *Comput. Graphics, Image Processing*, vol. 16, pp. 20–51, 1981.
[3] C. R. Carlson and R. W. Cohen, "Visibility of displayed information," Off. Naval Res., Tech. Rep., Contr. N00014-74-C-0184, 1978.
[4] ——, "A simple psychophysical model for predicting the visibility of displayed information," *Proc. Soc. Inform. Display*, pp. 229–246, 1980.
[5] K. Knowlton, "Progressive transmission of grayscale and binary pictures by simple, efficient, and lossless encoding schemes," *Proc. IEEE*, vol. 68, pp. 885–896, 1980.
[6] E. R. Kretzmer, "Reduced-alphabet representation of television signals," in *IRE Nat. Conv. Rec.*, 1956, pp. 140–147.
[7] J. J. Kulikowski and A. Gorea, "Complete adaptation to patterned stimuli: A necessary and sufficient condition for Weber's law for contrast," *Vision Res.*, vol. 18, pp. 1223–1227, 1978.
[8] A. N. Netravali and B. Prasada, "Adaptive quantization of picture signals using spatial masking," *Proc. IEEE*, vol. 65, pp. 536–548, 1977.
[9] A. N. Netravali and J. O. Limb, "Picture coding: A review," *Proc. IEEE*, vol. 68, pp. 336–406, 1980.
[10] W. K. Pratt, Ed., *Image Transmission Tecniques*. New York: Academic, 1979.
[11] W. F. Schreiber, C. F. Knapp, and N. D. Key, "Synthetic highs, an experimental TV bandwidth reduction system," *J. Soc. Motion Pict. Telev. Eng.*, vol. 68, pp. 525–537, 1959.
[12] W. F. Schreiber and D. E. Troxel, U.S. Patent 4 268 861, 1981.
[13] A. Rosenfeld and A. Kak, *Digital Picture Processing*. New York: Academic, 1976.

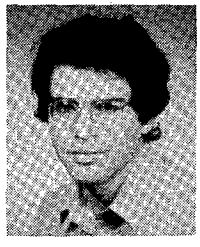
★



Peter J. Burt (M'80) received the B.A. degree in physics from Harvard University, Cambridge, MA, in 1968, and the M.S. and Ph.D. degrees in computer science from the University of Massachusetts, Amherst, in 1974 and 1976, respectively.

From 1968 to 1972 he conducted research in sonar, particularly in acoustic imaging devices, at the USN Underwater Sound Laboratory, New London, CT, and in London, England. As a Postdoctoral Fellow, he has studied both natural vision and computer image understanding at New York University, New York, NY (1976–1978), Bell Laboratories (1978–1979), and the University of Maryland, College Park (1979–1980). He has been a member of the faculty at Rensselaer Polytechnic Institute, Troy, NY, since 1980.

★



Edward H. Adelson received the B.A. degree in physics and philosophy from Yale University, New Haven, CT, in 1974, and the Ph.D. degree in experimental psychology from the University of Michigan, Ann Arbor, in 1979.

From 1979 to 1981 he was Postdoctoral Fellow at New York University, New York, NY. Since 1981, he has been at RCA David Sarnoff Research Center, Princeton, NJ, as a member of the Technical Staff in the Image Quality and Human Perception Research Group. His research interests center on visual processes in both human and machine visual systems, and include psychophysics, image processing, and artificial intelligence.

Dr. Adelson is a member of the Optical Society of America, the Association for Research in Vision and Ophthalmology, and Phi Beta Kappa.