Multidimensional Quantizers for Scalable Video Compression

Abha Singh and V. Michael Bove, Jr.

Abstract—**Scalable video compression requires the creation of an encoded bit stream that may be decoded in part if channel bandwidth drops, decoder resources are limited, or a smaller image than the source is desired. A separable spatiotemporal subband decomposition is combined with vector and lattice quantizers modified so that the individual subbands may be decoded scalably. This results in finer bandwidth control and more flexibility than simply discarding entire subbands.**

I. Introduction

A. Background

DATA compression has been an active area of research for many years; its study becomes even more compelling in the area of digital image representation and transmission due to the large amount of data associated with even a single image. Now that a variety of compression algorithms have been established and are starting to be proposed for consumer applications, it becomes increasingly important to examine (and increase) the degrees of freedom inherent in these methods. Relatively little research has been done in the area of scalable digital coding, which may be broadly defined as the generation of bit streams that produce useful images even when only a subset is decoded. This partial decoding may be done because the recipient wants an image of smaller size than the source, because the decoder lacks processing resources to decode the full bit stream or insufficient channel bandwidth exists (perhaps momentarily) to carry the full signal. Effectively, scalability decouples the parameters of decoding from the encoding.

A great deal of work is currently being focused on the development of high definition television systems (HDTV), with most proposed standards being tied to a single set of parameters (such as bandwidth, resolution, aspect ratio, and frame rate). During the past two years, this research has moved toward fully digital representations of the signal in order to achieve greater coding efficiency and immunity to channel impairments. The emphasis of the majority of HDTV research (a legacy of the initial investigations done at NHK in the 1970's) seems to be that the television consumer thirsts only for more scan lines or more picture elements per screen. Yet, with current advances in entertainment technology it is apparent that there is a great deal of appeal in features that place more control in the hands of the viewer; for example, picture-in-picture displays that allow the viewing of two or more programs concurrently, video recorders that allow viewers to record programs and view them later (perhaps with the ability to trade off image quality for storage capacity), and remote controls that allow viewers to create their own program menus from moment to moment, have all proven to have great mass appeal in the marketplace. Such flexible use of the broadcast material should be only the beginning of what will be possible when television moves to the digital domain, provided a correspondingly flexible image representation is employed.

We suggest that resolution increase is of but secondary importance among the coming changes in television, and that the more significant development will be the recognition that because digital representations are amenable to computation, they have fundamental advantages beyond bandwidth compression and noise resistance. Scalability, if designed into a compression scheme, is one of the chief advantages: with a scalable representation, a small countertop receiver need not contain the amount of signal processing hardware and frame memory that a large projection receiver would require. In a switched service, the smaller receiver would also receive less bandwidth and probably be billed less. A broadcast or recorded signal could be decoded at various resolutions, allowing consumers to determine the receiver quality for which they are willing to pay. A digital VCR would, without decoding the bit stream, be able to allow a broad range of quality versus capacity tradeoff. Adding picture-in-picture would not double the amount of digital decoding electronics in a receiver but might add only a tenth or a quarter more.

In any event, the requirements of any compressed digital video system are such that it is certain that the next generation of television receivers will contain greater computational complexity than the current generation of computer workstations. At the same time, computer workstations are beginning to develop the ability to display motion video, and many now operate their screens at higher refresh rates and pixel rates than any of the proposed HDTV systems. In interactive computing applications, it is often necessary to support resizable windows, varying transport bandwidths and decoder capabilities, and other requirements not necessarily present in traditional broadcast circumstances. Whether (as some have argued) or not the computing and television worlds are about to merge

Manuscript received June 1992; revised August 1992. This work was supoorted by DARPA/ISO under contract DAAD 05-90-C-0333, with additional support from the Movies of the Future and Television of Tomorrow research consortia.

A. Singh is with Future Systems Technology, International Business Machines Corporation, Austin, TX 78758.

V. M. Bove, Jr. is with the Media Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139.

IEEE Log Number 9204416.

into one, it is likely (and desirable) that at some level there will be a great deal of hardware and algorithmic commonality. Again, development of digital representations for advanced TV should take into account applications beyond transmission at a fixed rate to a single type of decoder and single resolution.

B. Scalable Open Architecture Television

In order to address the above concerns as well as others (such as the frame-rate mismatch between European television, American television, and workstation screens), the MIT Media Laboratory is researching video representations which decouple the frame rate, resolution, and bandwidth of the encoding from those of the decoding. This research is titled *Scalable Open Architecture Television.* A number of such representations are under investigation; the one used in this article is based on separable spatiotemporal subband filtering [3]. Quadrature mirror filters (QMF's) are used to form a pyramid of octave-wide oriented spatial frequency bands [26], while M-band filter sets divide image sequences into temporal bands of 10 or 12 Hz in width, as proposed by Schreiber [22]. Zeros are filled into the higher subbands not contained in the original image source [2], or nonlinear methods are used to synthesize higher frequencies [18].

This particular Open Architecture Television format allows a single input image sequence to be displayed on a variety of display systems by simply dropping or retaining the appropriate spatial and temporal frequency bands. Multiple sources originating at different frame rates can also be simultaneously displayed in separate windows on a single monitor. By converting all incoming signals internally to a common format, the Open Architecture Television system accepts input video produced at various resolutions, frame rates, and aspect ratios. Scalability according to bandwidth and processing power available at the receiver is another characteristic of the system, resulting directly from the fact that the system is decoupled from the source parameters. Since the representation is hierarchical, new levels may be added to extend it as higher-quality image sources and displays become available, without obsolescing lower-resolution equipment. Further, this representation of the signal has been shown to be a useful framework for applying multidimensional quantizers for compression [21].

C. Methodology

The issue of transmitter and decoder scalability is the focus of this investigation. Most image compression schemes require the encoder and decoders to operate at the same bit rate; however, this imposes an undesirable constraint on the receiver. We would instead like to produce a bit stream that might be compressed and stored at a high-quality level, and transmitted in part to match the current channel capacity. The decoder then has the option to decode all or part of the transmitted portion. Scaling the image size or reducing the load on the decoder by simply discarding a whole octave of spatial subbands (either by the receiver in a broadcast application or by the transmitter in a narrowcast application) is straightforward, but the bandwidth granularity afforded to a transmitter by this technique is fairly coarse. If we wish to support finer bandwidth control and maximize quality for the capacity available, we need to be able to reshape the quantization per component so that rather than dropping one subband entirely (effectively allocating zero bits to it) we send fewer bits worth of several bands [24].

Two quantizer schemes will be implemented and evaluated in terms of scalability. These quantizers will be applied to individual subbands, and thus have the added feature of allowing the information in the image to be weighted according to psychophysical methods. Thus, information may be concentrated in the bands to which the human visual system is most sensitive. One of the quantizers that will be evaluated in this work is one that selects codewords based on a mean square error criterion, using a kd-tree data structure [21]. In addition, a lattice quantizer of dimension eight will be studied. Based on the E_8 lattice, noted for its dense packing, this quantizer maps vectors to points in the lattice structure. Both of these quantizing schemes will be further modified to introduce the scalability aspect, restricting the codebooks of lesser codeword populations to be subsets of the larger ones [24]. Thus, at transmission, the codewords can be transmitted beginning with the smallest subset and continually adding additional vectors.

II. Quantization of Signals

A. Background

Quantization [15],[20],[14],[19] is a method for coding signals such that an approximate signal is constructed from a finite set of possible values. A given sample of the signal is specified by an index k into the finite set if it falls into the corresponding interval.

$$x_k < x < x_{k+1} \qquad \text{where } k \in \{1, 2, \cdots, L\}. \tag{1}$$

k is transmitted to the receiver to identify the particular value in the finite set that should be used to represent the signal's amplitude at that point. The value that k maps to is called the representative level, or reconstruction value, and the amplitudes x_k are decision levels or thresholds. L denotes the number of elements in the finite set of reconstruction values. The mapping $y = Q(x)$, the quantizer characteristic, is a staircase-shaped function. A quantizer may be classified as either uniform or nonuniform. Uniform quantizers simply divide the domain into equally spaced regions, and usually use the mean of each region as the representative level; nonuniform quantizers, however, have variable-length representative level intervals, which are functions of k. It should be noted that since the coded data is formed from a reduced set of values, irrecoverable information loss will occur.

B. Vector Quantization

Vector quantization [14], [19], [15], [20] expands the concepts employed in scalar quantization to the multidimensional case. A consequence of the rate distortion theory is that a coder that better utilizes the redundancy in a set of data will yield a lower bit rate for a given distortion rate. As a trivial example,

consider a block of data consisting of all ones. Transmission of the data could be accomplished by sending each element separately, achieving a bit rate of 1 bit/sample. However, if the whole block is taken together and indexed to a like block in a codebook, only the index need be transmitted; this would yield a bit rate of $1/xy$ bits/sample, where x is the horizontal size and y is the vertical size of the data. In general, a vector quantizer is a memoryless encoder since the mapping of input data to the output vectors is independent for each vector. The rate of an L element vector quantizer may be expressed as follows:

$$R = \frac{\log_2 L}{N} \text{ bits/sample} \tag{2}$$

where N is the number of dimensions of the vector. An important feature of vector quantization is that fractional bit rates are achievable, unlike those achievable with scalar quantization.

The basis for quantization is grounded in the area of information theory. Data compression, of which vector quantization is one type, attempts to reduce bit rate by performing redundancy reduction. As such, vector quantization performs this reduction by focusing on four interrelated properties of vectors and signals:

- correlation
- nonlinear dependence
- probability density function (pdf) shape
- vector dimensionality.

Vector quantization utilizes these properties by using them to determine appropriate placement of the vectors in the N-dimensional shape [14],[19].

1) Tree-Based Algorithms: Although vector quantization produces gains in coding rates over scalar quantization, in practice the gain in reduced bit rate proved to be outweighed by the increased computational requirements of the early multidimensional quantization schemes such as the algorithm developed by Linde, Buzo, and Gray (LBG) [17]. This algorithm to partition the vector space is based on an iterative method to converge on a minimum distortion codebook. Such algorithms normally require on the order of N^2 computations (N being the dimension).

Since the LBG algorithm proves to be so computationally demanding, more efficient algorithms have been devised based on a tree structure. Such tree-based methods yield implementations that require $N \log N$ computations. Bentley [1] introduced kd-trees (k dimensional trees) which make it possible to characterize multidimensional data as a binary search tree, with the expense being in the form of constraints placed on the splitting process. Kd-trees split the data by selecting a dimension along which the split will occur at each node. The extension that makes the kd-tree unique is that the dimension selected at each node for the split may be different from one node to the next. In general, the operation of splitting the tree at each node results in a cut of the k-dimensional region of interest with a $(k-1)$ dimensional hyperplane. By organizing data into branches along the tree, this mechanism produces a structure that is very well suited to partitioning a space and grouping vectors that are close together in the Euclidean sense. For a more detailed discussion of the actual implementation of the kd-tree structure for vector quantization, see [21],[5].

An advantage of this method over the general LBG algorithm is that the number of steps to completion is predetermined in the kd-tree system for a given codebook size. However, aggregate distortion is no longer the criterion for splitting the space; instead, divisions are based on the single dimension that has the highest overall distortion. Still, loss in image quality is outweighed by the gains in computational efficiency.

2) Scalable Vector Quantization: In applying the kd-tree implementation of vector quantization to scalable video, one very important change must be introduced in the algorithm. In the method described in the previous section, the codebook is built from the centroids of each leaf. However, in order to create codebooks that will be subsets of one another, the code vectors must be chosen from the set for the most populated codebook. Such a criterion for constructing the codebook will no longer yield an optimal code. Therefore, it should be noted that scalability comes at a price.

Optimal codebooks are generated by fulfilling the following two conditions.

- Average distortion is minimized:

$$\min D = \min \Big(\lim_{M \to \infty} (1/M) \sum_{n=1}^{M} d[x(n), y(n)] \Big).$$

- Each code vector y_i is chosen to minimize the average distortion in C_i:

$$\min D_i = \min E[d(x,y) | x \in C_i]$$

where $x(n)$ is the input and $y(n)$ is the output. If $x(n)$ is stationary and ergodic, then the sample average above tends to

$$D = E[d(x,y)]. \tag{3}$$

To evaluate the distortion for a typical vector quantizer, consider the case of a mean square error selection criterion. Generally, the distortion in a given cell is given by the following relation:

$$D_i = \frac{1}{n_i} \sum_{x \in C_i} d(x, y_i). \tag{4}$$

Previous analysis [17] yields the following result:

$$y = \frac{1}{n} \sum_{x_i \in C_i} x_i \tag{5}$$

which is the average value of the x_i's in the cluster. Thus, the optimal values for the codebook vectors are the average of the vectors within a cell. However, if the code vectors are chosen to be the closest vector to the centroid, the codebook can no longer be optimal. This follows from the following argument, where x_k is the vector selected to be the code vector:

$$d(x, x_k) = \frac{1}{n} \sum_{x_i \in C_i} (x_i - x_k)^2. \tag{6}$$

In addition, x_k may be expressed in terms of the centroid y

$$x_k - y = \epsilon. \tag{7}$$

Substituting (7) into (6) and applying some algebraic manipulations yields the following equation:

$$d(x, x_k) = \frac{1}{n} \sum_{x_i \in C_i} [(x_i - y)^2 - 2\epsilon(x_i - y) + \epsilon^2]. \tag{8}$$

Written another way, this becomes

$$d(x, x_k) = \frac{1}{n} \sum_{x_i \in C_i} (x_i - y)^2 - \frac{2\epsilon}{n} \sum_{x_i \in C_i} (x_i - y) + \epsilon^2. \tag{9}$$

Note that the last two terms in (9) correspond to the error associated with selecting the closest vector in a leaf to the centroid as the representative vector. Relative to the actual centroid value [the first term in (9)], the error is small and, therefore, should not cause serious degradation in the image.

Another aspect of this approach for scalable video is that as the tree is traversed, additional vectors correspond to additional detail. In fact, the detail in the image will be added in histogram order since the kd-tree partitioning employed is based on selecting the median coordinate value, which tends to add vectors according to frequency in the histogram. Hence, regions of higher vector incidence will be allocated more codewords while regions of lesser activity in the histogram are allocated fewer codewords. This histogram-order resolution enhancement is a highly desirable consequence of the vector quantizer since it allows the additional bandwidth to be allocated to regions of greater importance.

Some closely related work is reported by Mahesh and Pearlman [27]. They developed a multiple-rate tree-structured vector quantizer for subband coded still images as a method for reducing codebook transmission overhead while permitting rate allocation to be tuned for the properties of specific images.

C. Lattice Quantizer

1) Background: Shannon's rate distortion theory results showed that a lower bound on coding performance may be determined for a given rate (or distortion) [15]. In the years since the proposal of the theory, much work has been focused on determining a general set of characteristics for an optimal code. Buda shows that some lattice codes achieve the performance of the optimal codes proposed by Shannon [4]. Furthermore, Gersho [13] has conjectured that an optimal quantizer in any dimension has cells congruent to some prototype (for large M, the number of codewords). Thus, study of lattice quantizers seems well motivated. Some of the gains associated with a lattice structure have already been touched upon; for example, the shape of the cells used for vector quantization has been shown to have an impact on the achievable rate of the coder. A lattice quantizer is an extension of the vector quantizer system described earlier; however, the representative vectors are selected from points at the center of the sets of polytope cells which make up a lattice structure.

Before proceeding further with coding characteristics and implementation of lattice quantizers, some basic definitions will be presented. First, a lattice is formed by the set of all vectors spanning an M-dimensional space. That is, if a set of N linearly independent vectors in an M-dimensional real Euclidean space $\boldsymbol{a}_1, \cdots, \boldsymbol{a}_N, M \geq N$, the set of all linear combinations of these vectors

$$\boldsymbol{x} = u_1\boldsymbol{a}_1 + \cdots + u_N\boldsymbol{a}_N \tag{10}$$

is called an N-dimensional lattice [8]. Thus, the uniform structure of the lattice is apparent from its definition; the points in the lattice are uniformly spaced in the N-dimensional space. A more general structure is *sphere packing*, which is defined as the set of points in $\boldsymbol{R}^n$ (these are the centers of the spheres) such that

$$\text{dist}\,(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)})^2 = \|\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(j)}\| \geq 4\rho^2 \qquad \text{for all } i \neq j \tag{11}$$

where $\boldsymbol{x}^{(k)}$ are the points in the space, and ρ is the radius of the spheres.[1] This condition guarantees that the spheres do not overlap [25]. While lattice packing is sphere packing, sphere packing is not necessarily lattice packing. The distinction between the two is that for lattice packing, the points $\boldsymbol{x}^{(i)}$ must form a group under componentwise addition. Associated with both sphere and lattice packings is a region around each point known as the *Voronoi region*. This region defines the area around each point for which any other point in that region is closest to that particular lattice or sphere point.[2] From the earlier constraint for nonoverlapping spheres, it can be shown that the Voronoi region around each point in the sphere packing is the sphere with that point $\boldsymbol{x}^{(i)}$ at the center and radius ρ if mean squared error is to be minimized [13]. In general, this is one of the properties that makes lattice quantizers attractive; when the lattice is used as a quantizer, all points that lie within the Voronoi region of one of the lattice points are represented by that lattice point.

Construction of spherical or lattice codes is generally accomplished by choosing a finite set of points in the packing. For the image coding undertaken in this work, the E_8 lattice, which is the densest packing in eight dimensions, was employed. In addition, an efficient coding algorithm has been developed by Conway and Sloane [7],[8].

2) Scalable Video Using a Lattice Quantizer: A rather efficient algorithm has been developed for finding the closest point in the E_8 lattice to a data point in the space $\boldsymbol{R}^8$ [8]. Thus, the algorithm finds the Voronoi region to which the data point should be mapped. The algorithm, as applied to image coding, begins by dividing the image into blocks of eight pixels. Orientation of these blocks is not constrained by the quantizer, but psychophysical evidence suggests that the blocks should be as square as possible since the human visual system's response to horizontal and vertical stimuli is nearly isotropic. Therefore, the blocks are chosen to be 2×4 pixels. Once the blocks have been defined, they are treated as points in eight-dimensional space.

In order to limit the number of representative vectors, either the vectors may be scaled at input (this effectively expands

[1] It should be noted that the term *sphere* is used to describe some polytope region around each point in the set, and does not necessarily denote the geometry of the surrounding region.

[2] Voronoi regions are also referred to as Dirichlet regions, Brillouin zones, Wigner–Seitz cells, and nearest neighbor regions [28].

TABLE I
NINE- BIT SUBBAND RECONSTRUCTION: BIT RATE CALCULATIONS

Sequence #	Band #	# Channels	Size	B/p	Mb/s
Fball1	0	1	60 × 88	9	0.71
Fball2	0, 1, 2, 4	4	60 × 88	9	2.85
Fball3	0–6	7	60 × 88	9	4.99
Fball4	0–6	7	60 × 88	9	4.99
	8, 10	2	120 × 176	9	5.70
Fball5	0–7	8	60 × 88	9	5.70
	8, 10	2	120 × 176	9	5.70
Fball6	0–7	8	60 × 88	9	5.70
	8–11	4	120 × 176	9	11.40
Fball7	0–7	8	60 × 88	9	5.70
	8–11	4	120 × 176	9	11.40
	14, 16	2	240 × 352	9	22.81
Fball8	0–7	8	60 × 88	9	5.70
	8–12	5	120 × 176	9	14.25
	14, 16	2	240 × 352	9	22.81
Fball9	0–7	8	60 × 88	9	5.70
	8–11	4	120 × 176	9	11.40
	14–17	4	240 × 352	9	45.62
Fball10	0–7	8	60 × 88	9	5.70
	8–12	5	120 × 176	9	14.25
	14–17	4	240 × 352	9	45.62
Fball11	0–7	8	60 × 88	9	5.70
	8–13	6	120 × 176	9	17.10
	14–19	6	240 × 352	9	68.42

the Voronoi regions to include more than one lattice point) or the structure inherent in kd tree algorithms may be used to select the best n representative vectors based on a mean squared error (MSE) distortion measure. The results of the second method are better because the structure of the kd tree allows the vectors to be chosen such that their distribution may more closely match the distribution of the input data. This is particularly important in image data, whereas binary data over a modem line is better suited to limiting codewords by scaling or employing coset codes [6],[9],[12],[16].

As before, certain tradeoffs must be made in order to introduce scalability aspects to the coder. In the case of the lattice quantizer, the set of vectors that make up any particular codebook are all chosen from the same initial codebook set.

III. ANALYSIS AND COMPARATIVE RESULTS

A. Subband Coding

This section will focus on the experimental results from subband analysis of an image sequence. For this experiment, the first second on the "football" sequence was studied; some of the sample reconstructions, with their corresponding bit rates, are shown in Fig. 1 and Table II. The bands may be ranked in order of importance in order to add information in a manner that makes efficient use of the additional bandwidth. The reconstructions in this experiment were done according to a pattern motivated by psychovisual properties of the human visual system. First, the eye's response to temporal

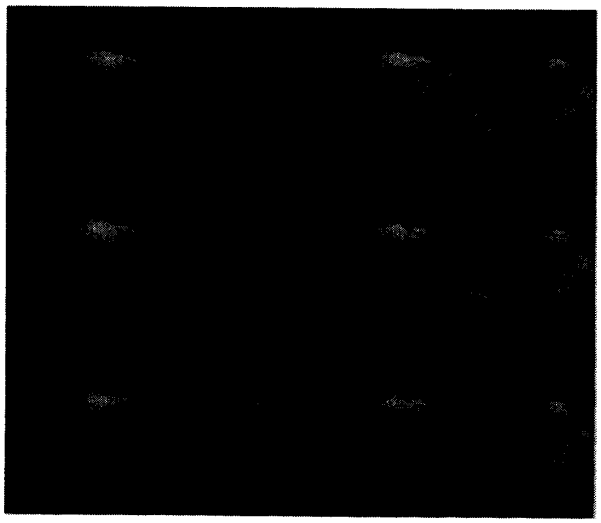

Fig. 1. Nine-bit subband reconstructions (left-to-right, top-to-bottom): "Fball2," "Fball3," "Fball5," "Fball6," "Fball10," and "Perfect Reconstruction at 9 b/p."

stimuli is bimodal; it acts as an integrator at low frequencies and as a differentiator at high frequencies. Essentially, the transfer function for the human visual system is lowpass in both the temporal and spatial dimensions [20]. For this reason, dividing the signal temporally increases efficiency, and for these experiments the signal was divided into two

TABLE II
NINE- BIT SUBBAND RECONSTRUCTION: SUBJECTIVE RESULTS, SNR, AND MSE

Sequence #	Ranking	Rating	SNR (dB)	MSE	Bit Rate (Mb/s)
Fball1	11	1.10	22.013	365.39	0.71
Fball2	10	1.40	24.448	208.56	2.85
Fball3	8	2.30	25.246	183.51	4.99
Fball4	9	1.95	25.246	173.55	10.69
Fball5	7	3.20	25.480	164.45	11.40
Fball6	4	3.90	25.774	153.69	17.10
Fball7	6	3.30	27.754	97.416	39.91
Fball8	3	4.20	28.221	87.49	42.76
Fball9	5	3.20	36.462	13.12	62.72
Fball10	2	4.20	42.603	3.19	65.57
Fball11	1	4.70	infinite	0	91.22

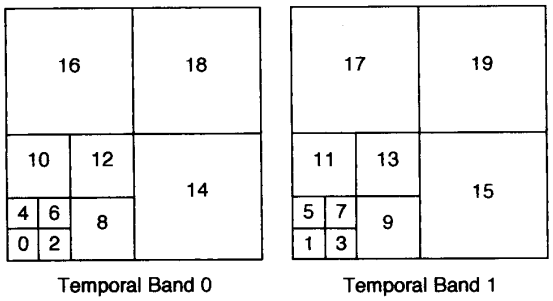

Fig. 2. Subband decomposition model.

temporal bands. High temporal subbands received more quantization error than the corresponding low temporal components.

Furthermore, the human visual system has reduced acuity for detail at oblique orientations while its response to vertical and horizontal edge detail (high-frequency information) is approximately equal. In addition, sensitivity to low-frequency spatial information, where most of the energy in the signal is concentrated, is greater. In order to separate the energy in the image so that bit allocation may take advantage of the human visual system model, the image is divided into frequency subbands. By splitting the image into octave width separations in the frequency domain, energy content is successively separated into low and high frequency bands. Our experiments used the three octave splits spatially to concentrate most of the energy into a small number of coefficients (the lowest spatiotemporal band). The two-band temporal split coupled with this spatial decomposition results in twenty bands (Fig. 2).

As a first experiment, we will simply quantize each band to nine bits per sample, an amount which has proven in earlier experiments to produce acceptable results, Then, we will reconstruct a subset of these subbands—effectively, we are restricting the bit allocation to either nine or zero bits per component. It should be noted that as each additional band is added to a bit stream, the bandwidth occupied by the sequence increases in large steps. In addition, as the separate bands are added, the increased detail is readily apparent. This will not be the case in some of the late experiments, a consequence that seems quite reasonable when increases in bandwidth are made more incremental. Table II compares calculated SNR and mean square error for each reconstruction. In addition, subjective results are also included in the table, where averaged perceptual values are tabulated. In these tests, subjects were asked to rank the images on a scale from one to five, five being perceptually transparent (artifact-free) and one being completely unacceptable. If the mid-range (three) is taken to be the minimum acceptable image quality, these results indicate that by reconstructing all of the lowest levels of the pyramid and all but the diagonal bands of level 2 (bands 12 and 13 in Fig. 2), an acceptable image results. Another result of the subband decomposition experiments is that the order in which the bands are added is important. Thus, a reconstruction may have a larger bit rate but, if the bands are not reconstructed appropriately, the perceptual quality will be lower than another sequence at a lower bit rate. This is evident in comparing the reconstruction used in the fourth sequence (38.49 Mb/s) to that of the ninth sequence (17.11 Mb/s), which had a much better perceived quality. Note that the figures for SNR and mean square error do not indicate this discrepancy. This comparison indicates that analysis of scalable coding should couple quantitative results with subjective results, since the mathematical qualities do not always indicate how well the image will be perceived. Quantitative studies are still important, however, since they measure the integrity of the data.

B. Variable Codebook Size Coding

1) Vector Quantization: The perceptual results of our scalable kd-tree vector quantizer proved better than the lattice quantizer. Perceptual results showed improvements with increased bit rates, though often incremental changes were barely perceptible (Fig. 3). The perceived quality roughly followed the SNR and mean square error measures. The bit rates are shown in Table III, and results of the quantitative and qualitative tests are given in Table IV.

The coder employed in these experiments is relatively efficient, certainly much more efficient than the regular LBG algorithm. The number of calculations decreases as the tree is built. The tree is built according to an "optimal tree" building algorithm developed by Bentley [1]. He proposes this method for cases which would require both a large number of searches of the tree and maintaining a static size tree (not addition or deletion of nodes) or for the cases in which the nodes (vectors) would arrive in a nonrandom order [1]. Our use of the kd tree for vector quantization creates both of these cases, since the quantizer will transverse the completed (and static) tree many times and the vectors are provided to the kd tree algorithm in a decidedly nonrandom order. The complexity of the algorithm will be analyzed according to worst-case behavior, in order to determine a lower bound on performance. Such a measure will also provide some indication of run time. This sort of analysis attempts to divorce the performance measure from machine, input, and implementation [23].

The basic algorithm used in this quantizer may be summarized as follows.

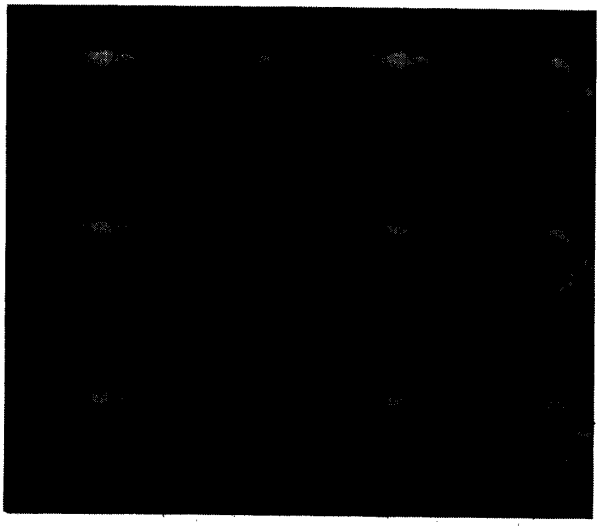

Fig. 3. Vector quantizer reconstructions (left-to-right, top-to-bottom): "Fball0," "Fball1," "Fball3," "Fball4," "Fball5," and "Original Fball."

- Calculate the selection value for each node, and along which dimension the split should occur.
- Compute the split point (based on median) within the selected node, and divide the node into two leaves.
- Compute the representative vectors for the newly formed leaves.

If the input is N vectors of k dimension, the first step requires Nk calculations on the first iteration, since selection requires one calculation per vector per dimension. The "optimized tree" building algorithm creates a balanced tree by mandating that the number of nodes in each subtree (leaf) of each node is no more than a difference of one from the other [1]. Since the space is divided evenly into two leaves, each with $N/2$ vectors then at the next iteration $k(N/2)+k(N/2) = Nk$ calculations are needed. At the next pass, however, only one of the leaves will be split, so only $k(N/4) + k(N/4)$ calculations will be required. In this manner, the number of iterations may be expressed in the following manner.

$$\text{Total} = Nk + 2\left(\frac{N}{2}k\right) + 2\left(\frac{N}{4}k\right) + 2\left(\frac{N}{4}k\right) + 2\left(\frac{N}{8}k\right) + 2\left(\frac{N}{8}k\right) + 2\left(\frac{N}{8}k\right) + 2\left(\frac{N}{8}k\right) + \cdots. \tag{12}$$

It is readily apparent that this will sum to some multiple of Nk and, with closer examination of the sum, it is clear that the total will be:

$$\text{Total} = Nk \log_2 N \text{calculations}. \tag{13}$$

The second step in the algorithm will require one calculation per vector in the selected leaf; this, too, will decrease as the tree is traversed. In fact, from the analysis of the first step, the calculations will be $N \log_2 N$. Finally, the last step requires two calculations per iteration, yielding a total of $2N$ calculations. It should be noted that these calculations are for building a completely ordered tree; that is, a tree with one leaf for each distinct input vector. Summing the contribution of these three steps, the complexity is determined to be $O(Nk \log_2 N)$.

TABLE III
VECTOR QUANTIZER RECONSTRUCTION: BIT RATE CALCULATIONS

Sequence #Z	Band #	# Channels	Size	B/p	Mb/s
Fball1	0–7	8	60 × 88	9	5.70
	8, 10	2	120 × 176	0.75	0.4752
	9, 11	2	120 × 176	0.625	0.396
Fball2	0–7	8	60 × 88	9	5.70
	8, 10	2	120 × 176	0.875	0.5544
	9, 11, 12	3	120 × 176	0.625	0.594
Fball3	0–7	8	60 × 88	9	5.70
	8, 10	2	120 × 176	1.0	0.6336
	9, 11	2	120 × 176	0.625	0.396
	12	1	120 × 176	0.75	0.2376
	13	1	120 × 176	0.5	0.1584
Fball4	0–7	8	60 × 88	9	5.70
	8, 10	2	120 × 176	1.0	0.6336
	9, 11, 12	3	120 × 176	0.75	0.7128
	13	1	120 × 176	0.625	0.198
	14, 16	2	240 × 352	0.625	1.584
Fball5	0–7	8	60 × 88	9	5.70
	8, 10	2	120 × 176	1.0	0.6336
	9, 11	2	120 × 176	0.875	0.5544
	12, 13	2	120 × 176	0.75	0.4752
	14, 16	2	240 × 352	0.75	1.9008
	15, 17	2	240 × 352	0.625	1.584

TABLE IV
VECTOR QUANTIZER RECONSTRUCTION: SUBJECTIVE RESULTS, SNR, AND MSE

Sequence #	Ranking	Rating	SNR (dB)	MSE	MSE
Fball1	5	3.20	23.504	259.22	6.57
Fball2	1	3.80	23.510	258.82	6.85
Fball3	4	3.50	23.513	258.65	7.13
Fball4	3	3.50	23.593	253.92	8.83
Fball5	2	3.70	23.643	251.02	10.85

2) Lattice Quantization: In general, the perceptual results for the scalable lattice quantizer matched the SNR and mean square error evaluations, with only a few exceptions. Results of the bit rate calculations and subjective tests are given in Tables V and VI, respectively. Sample reconstructions are shown in Fig. 4.

TABLE V
LATTICE QUANTIZER RECONSTRUCTION: BIT RATE CALCULATIONS

Sequence #	Band #	# Channels	Size	B/p	Mb/s
Fball1	0, 1, 2, 4	4	60 × 88	9	2.85
	3, 5	2	60 × 88	0.75	0.1188
	6, 7	2	60 × 88	0.625	0.099
Fball2	0, 1, 2, 4	4	60 × 88	9	2.85
	3, 5	2	60 × 88	1.0	0.1584
	6, 7	2	60 × 88	0.75	0.1188
	8, 10	2	120 × 176	0.75	0.4752
	9, 11	2	120 × 176	0.625	0.396
Fball3	0, 1, 2, 4	4	60 × 88	9	2.85
	3, 5	2	60 × 88	1.0	0.1584
	6, 7	2	60 × 88	0.875	0.1386
	8–11	4	120 × 176	0.875	1.1088
	12	1	120 × 176	0.625	0.198
Fball4	0–7	8	60 × 88	9	5.70
	8, 10	2	120 × 176	0.875	0.5544
	9, 11, 12	3	120 × 176	0.625	0.594
Fball5	0–7	8	60 × 88	9	5.70
	8, 10	2	120 × 176	1.0	0.6336
	12	1	120 × 176	0.75	0.2376
	9, 11, 13	3	120 × 176	0.625	0.594
Fball6	0–7	8	60 × 88	9	5.70
	8, 10	2	120 × 176	1.0	0.6336
	9, 11	2	120 × 176	0.875	0.5544
	12	1	120 × 176	0.75	0.2376
	13	1	120 × 176	0.625	0.198
Fball7	0–7	8	60 × 88	9	5.70
	8, 10	2	120 × 176	1.0	0.6336
	9, 11, 12	3	120 × 176	0.75	0.7128
	14, 16	2	240 × 352	0.625	1.584
Fball8	0–7	8	60 × 88	9	5.70
	8,10	2	120 × 176	1.0	0.6336
	9, 11	2	120 × 176	0.875	0.5544
	12, 13	2	120 × 176	0.75	0.4752
	14, 16	2	240 × 352	0.75	1.90
	15, 17	2	240 × 352	0.625	1.58

TABLE VI
LATTICE QUANTIZER RECONSTRUCTION: SUBJECTIVE RESULTS, SNR, AND MSE

Sequence #	Ranking	Rating	SNR (dB)	MSE	Bit Rate (Mb/s)
Fball1	8	2.20	24.75	194.34	3.07
Fball2	7	2.90	25.18	176.30	4.00
Fball3	6	3.30	25.33	170.05	4.45
Fball4	3	3.80	25.35	169.32	6.85
Fball5	1	4.30	25.43	166.39	7.17
Fball6	5	3.60	25.41	167.09	7.32
Fball7	2	4.00	26.04	144.48	8.63
Fball8	4	3.70	27.43	105.00	10.84

Fig. 4. Lattice quantizer reconstructions (left-to-right, top-to-bottom): "Fball1," "Fball4," "Fball6," and "Fball8."

The complexity of the lattice coder is tractable for these applications. Recall from the earlier discussions that the vectors are of dimension eight (2×4). Assuming N vectors as input and a final codebook size of M, the closest lattice points are computed according to the algorithm described in Section II-C. Instead of recursively converging on the representative vectors for the codebook, this method does a single pass through the data with the following calculations.

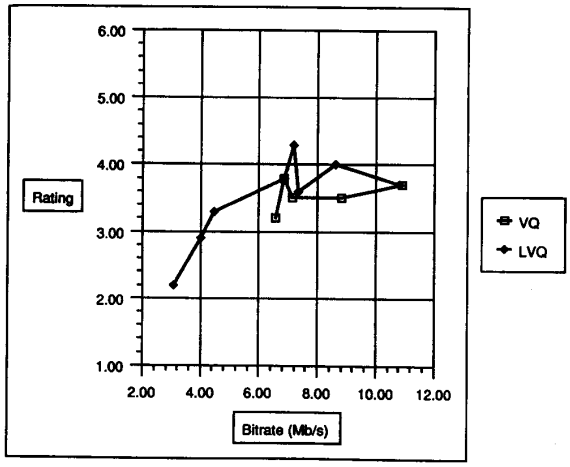

Fig. 5. Subjective results as a function of decoded bit rates.

- Find the vector with integer components closest to the input vector components.
- Compute the sum of the components of this vector.
- Determine the vector with integer components closest to the input vector, except the one furthest from an integer is rounded the wrong way.
- Find the sum of the components of this vector.
- Select, as a first guess, whichever yields an even sum.
- Repeat the above steps for the vector formed by subtracting 1/2 from each component of the input vector.
- Select, as the second guess, whichever of the two resulting vectors has an even sum of components, and add 1/2 to each component.
- Compute the mean square error between each guess and the input vector, and select the closest of the two guesses as the representative vector.

Each step requires Nk calculations; since k is eight, the total number of calculations per step is $8N$. This results in a total of approximately $80N$ calculations; thus, the complexity of the coder is $O(N)$. In order to limit the size of the codebook, the vectors resulting from the lattice quantizer are arranged using a kd tree algorithm, and the vector closest to the centroid in each leaf is selected as a representative vector. Hence, the algorithm requires an additional amount of calculation $O(Nk \log_2 N)$.[3] The complexity of this coder is comparable to that of the kd tree-coupled vector quantizer, and the structure of the lattice affords some greater efficiency in developing the codebook. A comparison of the kd tree-coupled vector quantizer and the lattice quantizer is shown graphically in Fig. 5. The graph plots the subjective results versus bit rates for each quantizer.

IV. Conclusion

In an effort to develop a coding method that would allow a single data stream to be encoded at various rates, two different multidimensional quantization schemes were extended to include scalability. By decoupling the input source from the parameters of the output device, this scheme allows the channel and decoding resources to be allocated most efficiently. In addition, the methods studied here may easily be incorporated into the structure of the separable subband Open Architecture representation mentioned earlier. Such flexibility in transmission and decoding will benefit both entertainment and professional applications of digital video.

The quantizers studied both achieve a fair degree of compression at acceptable image quality and further introduce a fine granularity in bandwidth control.[4] From the results of these experiments, it appears that a good balance between granularity in scalability, coder complexity, and image quality is achieable by employing the variable codebook size coding scheme coupled with either the quantizer discussed by Romano [21] or a lattice quantizer. Computational complexity of these two quantizers appears manageable, although perhaps not realizable in real time with present-day general-purpose microprocessor technology. Still, the basic algorithm seems to be a promising one.

[3]The codebook may also be limited in size by scaling the input vectors so that they map to fewer distinct vectors in the lattice structure. Coding results from this method, however, do not lend themselves as readily to scalable applications.

Acknowledgment

The authors wish to thank A. Netravali and A. Lippman for their insights and assistance with this research.

References

[1] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM.*, vol. 18, no. 9, pp. 509–517, Sept. 1975.
[2] V. M. Bove, Jr. and A. B. Lippman, "Open architecture television receivers and extensible/intercompatible digital video representations," in *Proc. IEEE ISCAS*, 1990, pp. 1294–1297.
[3] V. M. Bove, Jr. and A. B. Lippman, "Scalable open architecture television," *SMPTE J.*, vol. 101, no. 1, pp. 2–5, Jan. 1992.
[4] R. De Buda, "Some optimal codes have structure," *IEEE J. Select. Areas Commun.*, vol. 7, no. 6, pp. 893–899, Aug. 1989.
[5] W. J. Butera, "Multiscale coding of images," Master's thesis, Mass. Inst. Technol., Sept. 1988.
[6] A. R. Calderbank and N. J. A. Sloane, "New trellis codes based on lattices and cosets," *IEEE Trans. Inform. Theory*, vol. IT-33, no. 2, pp. 177–195, Mar. 1987.
[7] J. H. Conway and N. J. A. Sloane, "A fast encoding method for lattice codes and quantizers," *IEEE Trans. Inform. Theory.*, vol. IT-29, no. 6, pp. 820–824, Mar. 1982.
[8] J. H. Conway and N. J. A. Sloane, "Fast quantizing and decoding algorithms for lattice quantizers and codes," *IEEE Trans. Inform. Theory*, vol. IT-28, no. 2, pp. 227–232, Mar. 1982.
[9] G. D. Forney, Jr., "Coset codes—Part I: Introduction and geometrical classifications," *IEEE Trans. Inform. Theory*, vol. 34, no. 5, pp. 1123–1151, Sept. 1988.
[10] G. D. Forney, Jr., "Coset codes—Part II: Binary lattices and related codes," *IEEE Trans. Inform. Theory.*, vol. 34, no. 5, pp. 1152–1187, Sept. 1988.
[11] G. D. Forney, Jr., "Multidimensional constellations—Part I: Introduction, figures of merit, and generalized cross constellations," *IEEE J. Select. Areas Commun.*, vol. 7, no. 6, pp. 877–892, Aug. 1989.
[12] G. D. Forney, Jr., "Multidimensional constellations—Part II: Voronoi constellations," *IEEE J. Select. Areas Commun.*, vol. 7, no. 6, pp. 941–958, Aug. 1989.
[13] A. Gersho, "Asymptotically optimal block quantization," *IEEE Trans. Inform. Theory.*, vol. IT-28, no. 4, pp. 373–380, July 1979.
[14] R. M. Gray, "Vector quantization," *IEEE ASSP Mag.*, vol. 4, Apr. 1984.
[15] N. S. Jayant and P. Noll, *Digital Coding of Waveforms.* Englewood Cliffs, NJ: Prentice-Hall, 1984.
[16] G. R. Lang and F. M. Longstaff, "A leech lattice modem," *IEEE J. Select. Areas Commun.*, vol. 7, no. 6, pp. 968–973, Aug. 1989.

[4]Additional analysis and discussion of this methodology may be found in [24].

[17] 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.
[18] L. L. Liu, "Digital intermediate format for video frame rate conversion," Master's thesis, Mass. Inst. Technol., Sept. 1990.
[19] J. Makhoul *et al.*, "Vector quantization in speech coding," *Proc. IEEE*, vol. 73, no. 11, pp. 1551–1588, Nov. 1985.
[20] A. N. Netravali and B. G. Haskell, *Digital Pictures: Applications of Communications Theory*. New York: Plenum, 1988.
[21] P. Romano, Jr., "Vector quantization for spatiotemporal sub-band coding," Master's thesis, Mass. Inst. Technol., Sept. 1989.
[22] W. F. Schreiber *et al.*, "Channel-compatible 6-Mhz HDTV distribution systems," *SMPTE J.*, vol. 98, no. 1, pp. 5–13, Jan. 1989.
[23] R. Sedgewick, *Algorithms in C*. Reading, MA: Addison Wesley, 1990.
[24] A. Singh, "Variable resolution video," Master's thesis, Mass. Inst. Technol., June 1991.
[25] N. J. A. Sloane, "Tables of sphere packings and spherical codes," *IEEE Trans. Inform. Theory*, vol. IT-27, no. 3, pp. 327–338, May 1981.
[26] M. Vetterli, "Multi-dimensional sub-band coding: Some theory and applications," *Signal Process.*, vol. 6, no. 2, pp. 97–112, Feb. 1984.
[27] B. Mahesh and W. A. Pearlman, "Multiple-rate structured vector quantization of image pyramids," *J. Vis. Commun. Image Represent.*, vol. 2, no. 2, pp. 103–113, June 1991.
[28] J. H. Conway and N. J. A. Sloane, "Voronoi regions of lattices, second moments of polytopes, and quantization," *IEEE Trans. Inform. Theory*, vol. IT-28, no. 2, pp. 211–226, Mar. 1982.

Abha Singh received the B.S.E.E. and B.S. degrees in applied mathematics from Southern Methodist University, Dallas, TX, in 1989, and the M.S.E.E. degree from the Massachusetts Institute of Technology, Cambridge, in 1991.

As a member of the M.I.T. Media Laboratory Entertainment and Information Systems Group, her research focused on video coding algorithms, particularly scalable video. Currently, she is a staff member in IBM's Advanced Workstation Division, Future Systems Technology Group. Her research interests include image coding, audio and video synchronization, and multimedia. She is a member of Tau Beta Pi, Eta Kappa Nu, and Phi Beta Kappa.

V. Michael Bove, Jr. received the B.S.E.E. degree, the M.S. degree in visual studies, and the Ph.D. degree in media technology from the Massachusetts Institute of Technology, Cambridge.

Since 1989, he has been Assistant Professor of Media Technology, working in the M.I.T. Media Laboratory. His research includes advanced digital television systems, design of hardware and software architectures for video processing, and the application of machine vision and computer graphics techniques for producing structural models of real scenes.