Video Compression Using Matching Pursuits

Osama K. Al-Shaykh, Eugene Miloslavsky, Toshio Nomura, Ralph Neff, and Avideh Zakhor

***Abstract*— The use of matching pursuit (MP) to code video using overcomplete Gabor basis functions has recently been introduced. In this paper, we propose new functionalities such as SNR scalability and arbitrary shape coding for video coding based on matching pursuit. We improve the performance of the baseline algorithm presented earlier by proposing a new search and a new position coding technique. The resulting algorithm is compared to the earlier one and to DCT-based coding.**

***Index Terms*—Arbitrary shape coding, matching pursuit, scalability, video compression.**

I. Introduction

ALL existing video compression standards are hybrid systems in that the compression is achieved in two main stages: the first stage, motion estimation and compensation, predicts each frame from its neighboring frames, compresses the prediction parameters, and produces the prediction error frame; the second stage codes the prediction error. All existing video compression standards use block-based discrete cosine transform (DCT) to code the residual error [1], [2], [4]. Although DCT video coding is efficient, it introduces undesirable blocking artifacts, especially at low bit rates. Moreover, due to bit-rate restrictions, some blocks are only represented by one or a small number of coarsely quantized transform coefficients, resulting in artifacts commonly known as ringing and mosquito noise. Other approaches such as wavelets [7] introduce ringing or rippling artifacts, which become most bothersome in the vicinity of image edges.

Neff and Zakhor have recently applied the matching pursuit (MP) technique of Mallat and Zhang [6] to code the motion prediction error signal [8]. The MP coder divides each motion residual into blocks, and measures the energy of each block. The center of the block with the largest energy value is adopted as an initial estimate for an inner product search. A dictionary of Gabor basis vectors is then exhaustively matched to an $S \times S$ window around the initial estimate. The location, basis vector index, and value of the largest quantized inner product are then coded together. This procedure is applied recursively until either the bit budget is exhausted or the distortion goes below a prespecified threshold.

Video sequences coded using matching pursuit do not suffer from either blocking or ringing artifacts since the basis vectors are only coded when they are well matched to the residual signal. As the bit rate decreases, the distortion introduced by matching pursuit coding takes the form of a gradually increasing blurriness or loss of detail.

In this paper, we propose two new functionalities, i.e., SNR scalability and coding of arbitrary shaped video objects, based on matching pursuit. We also propose two ways to improve the performance of the baseline algorithm in [8]. These include new search strategies and new position coding techniques.

This paper is organized as follows. Section II reviews video coding using matching pursuit, Section III provides a pessimistic bound on coding efficiency for positions of a group of atoms, Section IV discusses SNR scalability, Section V introduces new position coding and search strategies to improve coding efficiency, and compares the performance of matching pursuit with that of DCT-based coders (MPEG-4 [5]), Section VI extends the MP coder to support arbitrary shape video sequences, and finally, Section VII concludes the paper.

II. Matching Pursuit Video Coder

Representing a signal using an overcomplete basis set implies that there is more than one representation for the signal. For coding purposes, we are interested in representing the signal with the fewest basis vectors. This is an NP-complete problem [6]. Different approaches have been investigated to find or approximate the solution. Matching pursuit is a multistage algorithm, which in each stage finds the basis vector that minimizes the mean-squared error [6].

Suppose we want to represent a signal $f[i]$ using basis vectors from an overcomplete dictionary or basis set $\mathcal{G}$. Individual dictionary vectors can be denoted as

$$g_\gamma[i] \in \mathcal{G}. \tag{1}$$

Here, γ is an indexing parameter associated with a particular dictionary element. The decomposition begins by choosing γ to maximize the absolute value of the following inner product:

$$t = \langle f[i], g_\gamma[i] \rangle \tag{2}$$

where t is the transform or expansion coefficient. A residual signal is computed as

$$R[i] = f[i] - t\, g_\gamma[i]. \tag{3}$$

This residual signal is then expanded in the same way as the original signal. The procedure continues iteratively until either a set number of expansion coefficients is generated or some energy threshold for the residual is reached. Each stage k yields a dictionary structure specified by γ_k, an expansion coefficient $t[k]$, and a residual R_k, which is passed on to the next stage. After a total of M stages, the signal can be

Manuscript received October 23, 1997; revised April 24, 1998. This work was supported by the Office of Naval Research under Grant N00014-92-J-1732, AA SERT Supplement N00014-95-1-1163, the Air Force Office of Scientific Research under Grant F49620-96-1-0199, California State Program MICRO, Sun Microsystems, Philips, Tektronix, LG Electronics, and Sharp Corporation. This paper was recommended by Associate Editor M.-T. Sun.

The authors are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720 USA.

Publisher Item Identifier S 1051-8215(99)01188-X.

1051–8215/99$10.00 © 1999 IEEE

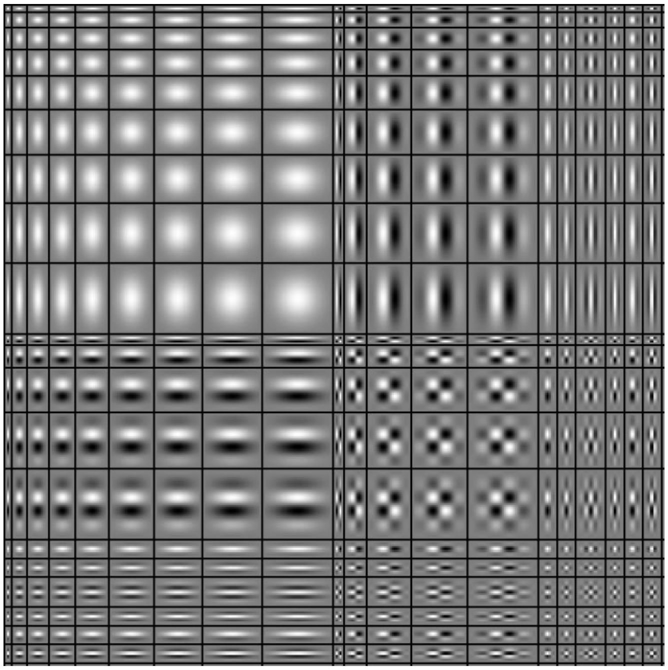


Fig. 1. Separable two-dimensional 20 × 20 Gabor dictionary.

approximated by a linear function of the dictionary elements:

$$\hat{f}[i] = \sum_{k=1}^{M} t[k]\ g_{\gamma_k}[i]. \tag{4}$$

Direct application of matching pursuit to represent motion compensation residual error is computationly intensive to the extent that it makes the algorithm nonpractical. This is because an $N_1 \times N_2$ residual image with one $N_1 N_2$ luma pixels and two $(N_1/2)(N_2/2)$ chroma components, and a dictionary set of P basis functions would require the computation of $(3N_1 N_2 P/2)$ inner products. For $N_1 = 176, N_2 = 144$ (QCIF image), and $P = 400$, we need to compute 15.2 million inner products. If the average support of the basis functions is 15×15, we need 3.4 billion multiplications and additions each time one function is computed. Clearly, such a level of computation would make the algorithm too prohibitive from an implementation point of view.

To overcome this computational complexity, the matching pursuit video coder in [8] first divides each motion residual into blocks, and measures the energy of each block. The center of the block with the largest energy value is adopted as an initial estimate for the inner-product search. A dictionary of Gabor basis vectors, shown in Fig. 1, is then exhaustively matched to an $S \times S$ window around the initial estimate. The exhaustive search can be thought of as follows. Each $N \times N$ dictionary structure is centered at each location in the search window, and the inner product between the structure and the corresponding $N \times N$ region of image data is computed. The largest inner product is then quantized. The location, basis vector index, and quantized inner product are then coded together.

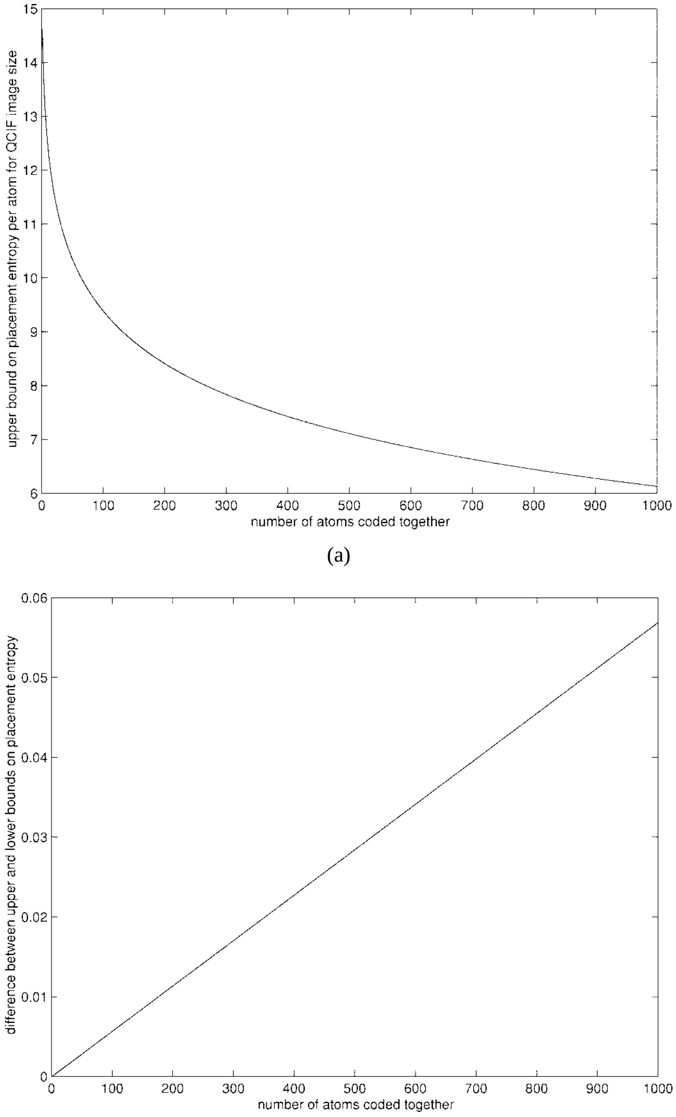
The decoder needs to know the basis function used to represent the residual error, its locations, and the value of the quantized inner product. For a more efficient coder, the basis index and the inner product are coded using variable-length codes (VLC). To code atom positions, the atoms are sorted in position order from left to right and top to bottom within the residual image. A differential coding strategy employs three basic codeword tables. The first table $P1$ is used at the beginning of a screen line to indicate the horizontal distance from the left side of the image to the location of the first atom on the line. For additional atoms on the same line, the second table $P2$ is used to transmit the interatom distances. The $P2$ table also contains an escape code indicating that no additional atoms exist on the current line. The escape code, when used, is always followed by a $P3$ code, indicating how many lines in the image may be skipped before the next line containing coded atoms. The $P3$ code is then followed by a $P1$ code since the next atom will be the first on a particular line. No special codeword is needed to indicate the end of the atom field since the number of coded atoms is transmitted as header information.

III. Theoretical Bounds on Position Coding Efficiency

In this section, we derive a theoretical bound on the number of bits used for atom position coding. As we will see, the theoretical bound shows that the efficiency of atom position coding improves as the number of atoms that are to be coded together is increased. This bound is relevant in understanding coding efficiency losses involved in achieving SNR scalability.

To characterize the dependence of coding efficiency on the number of atoms coded together in a single group, consider a situation where atoms are assumed to be uniformly and independently distributed on an $N_1 \times N_2$ image. Our goal is to derive an expression for entropy of various placements of a atoms on an $N_1 \times N_2$ image, without taking the order of atoms into consideration. This problem is equivalent to finding the entropy of the various ways in which a indistinguishable balls can be put into n baskets where, in our case, n is $N_1 N_2$ and a is the size of the group of atoms whose positions are to be coded. For example, in the case where $a = 2, n = 2$, there are three possible placements: both balls in the first basket, both balls in the second basket, and one ball in each basket. The probabilities of the first and second placements are 0.25 and of the third 0.5 since balls are placed into baskets independently and uniformly. In general, there are $\binom{a+n-1}{a}$ different placements of a balls in n baskets. Of course, some of these placements have different probabilities, so if we assume that they all have equal probability, we will get an upper bound on the entropy which is $\log_2(\binom{a+n-1}{a})$. Also, note that there are $\binom{n}{a}$ placements in which no basket has more than one ball, and that all such placements are equally probable. Therefore, $\log_2(\binom{n}{a})$ is the lower bound on the entropy of placement distribution. Thus, the entropy per atom is between $(\log_2(\binom{n}{a}))/a)$ and $(\log_2(\binom{a+n-1}{a}))/a)$.

Fig. 2(a) shows the upper bound on the number of position bits per atom needed for 176 × 144 QCIF images as a function of the number of atoms coded together. As expected, the coding efficiency improves as the number of atoms increases. This is in agreement with the experimental results in Section IV-A2. Fig. 2(b) shows the difference between the theoretical lower and upper bounds as a function of the number of atoms coded together, again for QCIF images.



(a)

(b)

Fig. 2. Lower and upper bound on the entropy of placement of a uniformly iid distributed atoms normalized by a. (a) Upper bound. (b) Difference between upper and lower bounds.

As seen, the two bounds are fairly tight, with the maximum difference being 0.0569 bits/atom at $a = 1000$. In Section IV-A1, we present a nonadaptive algorithm that not only achieves, but also outperforms these bounds by taking advantage of nonuniformity in atom distribution.

IV. SNR-Scalable Matching Pursuit Coder

Developing scalable video compression algorithms has attracted considerable attention in recent years. SNR-scalable compression refers to encoding a sequence in such a way that different quality video can be reconstructed by decoding a subset of the encoded bit stream. Scalable compression is useful in today's heterogeneous networking environment in which different users have different rates, resolution, display, and computational capabilities.

Scalable video compression schemes can be broadly classified into two categories: 1) coarse grain scalability with few widely different available bit rates, and 2) fine grain scalability with a continuum of available bit rates. The two-layer codecs used in ATM applications are an example of the first class [9], and the video codec developed by Taubman and Zakhor [11] is an example of the second class. In most codecs in the second class, fine grain scalability is achieved via multirate quantization of the DCT or wavelet coefficients [10], [11]. However, as will be seen shortly, for an MP-based codec, a natural way of achieving both fine and coarse scalability is through the number of atoms.

In this section, we will investigate a number of scalable video coding schemes based on matching pursuits. In Section IV-A, we will propose a one-residual image system offering fine grain scalability, and in Section IV-B, we will examine a two-residual image system offering better coding efficiency at the expense of coarser scalability.

A. Using One Residual Image

Fig. 3 illustrates our basic approach to SNR scalability in the "one residual image" scheme. As seen, the motion compensation residual image is formed from the previously reconstructed base layer frame in order to avoid the drifting problem. Furthermore, the encoder only keeps track of only one residual image, namely, the one corresponding to the base layer. Once the residual image is found, a certain number of atoms is used to code the base layer, and additional atoms are used to code the enhancement layer. This way, the decoder can stop at any time after decoding the base layer information without losing track of the encoder. Moreover, if the enhancement layer atoms are coded a few at a time as they are found, we can have a scalable coder with resolution of a few atoms, e.g., 100 bits/frame. As we have seen in Section III, there is a tradeoff between coding efficiency and the fineness of scalability. In the next section, we will describe a practical atom position coding method that greatly improves coding of small groups of atoms, and will present performance results of a fine grain SNR-scalable MP-based coder with one residual image.

1) NumberSplit—A Method for Coding the Position of Atoms: The NumberSplit algorithm, used for coding atom positions in the results of Section IV-A2, is based on the divide-and-conquer idea. First, the total number of atoms T coded on a given residual image is transmitted in the header. Then the image is divided into two halves along a larger dimension, and the number of atoms in the left or top half (depending on how the image was split) is coded. Note that if we assume that each atom falls uniformly and independently of other atoms onto either half, then the number of atoms in the first half is binomially distributed on $\{0, \cdots, T\}$ with $p = 0.5$. Since we know the total number of atoms on an image and the distribution of the number of atoms in the first half, we can construct a Huffman table to encode the number of atoms in the first half. The total number of atoms and the number of atoms in the first half allows the decoder to calculate the number of atoms in the second half. This algorithm is then applied recursively to the halves of the image until there are no more atoms in a given half image or until the size of the half image is one pixel. The Huffman tables used in encoding

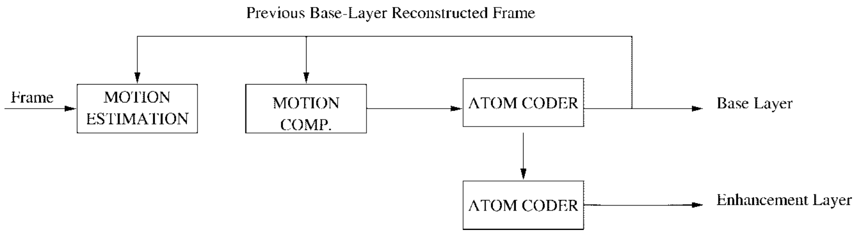


Fig. 3. Block diagram illustrating the one-residual approach for two-layer SNR scalability.

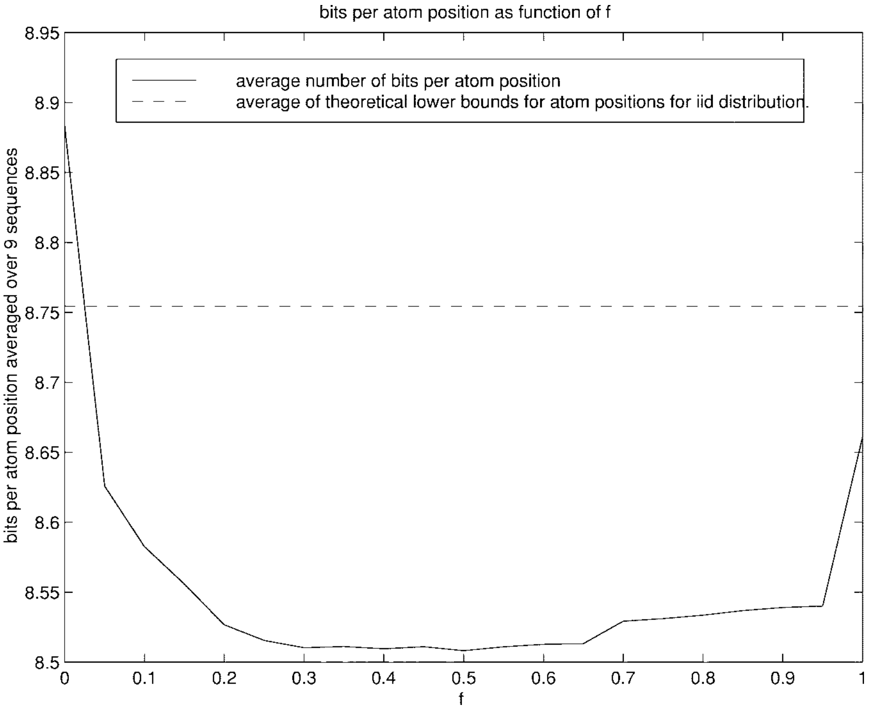


Fig. 4. Average number of bits per atoms position as a function of f for following sequences: Container Ship QCIF at 10 kbit/s, Mother-Daughter QCIF at 10 kbit/s, Hall Monitor QCIF at 10 kbit/s, Container Ship QCIF at 24 kbit/s, Mother-Daughter QCIF at 24 kbit/s, Silent Voice QCIF at 24 kbit/s, Foreman QCIF at 48 kbit/s, Coast Guard QCIF at 48 kbit/s, and News CIF at 48 kbit/s.

are built dynamically, depending on the number of atoms to be coded, which allows the NumberSplit method to avoid inefficiencies of fixed tables at a cost of more computation.

In real residual images, atoms are placed at the locations where motion estimation is ineffective. For this reason, atoms are not distributed uniformly and independently on an image—they tend to "cluster" around the regions of high residual error. One heuristic way to tune the NumberSplit algorithm to the real-life images is to modify the binomial distribution to account for clustering. It is easy to see that if atoms tend to cluster, the probability of many atoms being in the same half of the image is higher than if atoms are independent and identically distributed. To account for this, we emphasize the tails of binomial distribution in the following way: all probabilities of splits that are smaller than some fraction f of the maximum probability in the distribution are set to $f\times$ maximum probability, followed by renormalization of the distribution. In Fig. 4, we see that as f is varied from 0 to 1, the average number of bits per position of an atom for various video sequences and bit rates is between 8.52 and 8.87 bits/position. It is interesting to note that, by taking advantage of the dependence and nonuniformity in the atom distribution, the NumberSplit method spends fewer bits per atom position than the theoretical lower bound for the uniform independent atom distribution described in Section III. The value of $f = 0.2$ was used in the experiments described in Section IV-A2 to explore the tradeoff between rate granularity and compression efficiency for scalable video.

By coding all atoms together, NumberSplit achieves good compression efficiency, but also suffers from poor error resilience since a single transmission error may affect all atoms. However, the effects of such an error would not bring long-lasting drift and quality degradation problems since we are only going to use NumberSplit to code atoms in the enhancement layer for the scheme described in Fig. 3. In the single residual scheme in Fig. 3, atoms in the enhancement layer are not in the motion compensation loop, so that errors in atom positions for the enhancement layer will not propagate to the future frames.

2) Fine Scalability—Granularity Versus Coding Efficiency: We have developed a finely scalable codec based on the approach shown in Fig. 3 in which the enhancement layer atoms are coded in groups of N atoms at a time, where N can range from 5 to 100. We use the NumberSplit algorithm

TABLE I
BIT RATES USING NUMBERSPLIT AND FIXED HUFFMAN TABLES FOR SENDING 50 ATOMS IN BASE LAYER AND 100 ATOMS IN ENHANCEMENT LAYER IN GROUPS OF VARIOUS SIZES FOR CONTAINER SEQUENCE AT 7.5 FRAMES/S

Size of the groups (in atoms)	Bitrates (in Kbits/s) using NumberSplit	Bitrates (in Kbits/s) using fixed Huffman tables	NumberSplit gain (in %)
5	30.16	33.39	10.70
10	29.56	31.83	7.67
20	28.90	30.34	4.99
50	27.96	28.80	3.02
100	27.19	27.78	2.15

for efficient position coding of each group of N atoms in the enhancement layer. The advantages of using this position coding technique over the one originally proposed in [8] and described in Section II are twofold. First, as will be seen later, simulation results in Table I show that for small values of N, this position coding scheme is superior to the one originally developed in [8]; second, the NumberSplit algorithm does not need trained Huffman tables for each value of N, and as such, requires no statistics to be gathered for each value of N.

Fig. 5 shows the average PSNR versus bit-rate characteristics of the above scalable codec based on the NumberSplit position coding for 10 s of the *Container* sequence coded at 7.5 frames/s. Three different allocations of atoms between layers have been used: (50,100), (75,75), and (100,50) atoms coded in base and enhancement layers, respectively, on each frame. In all three cases, the total number of atoms is 150. In this scheme, the atoms in the base layer are sent together using the method described in Section II, while the atoms in the enhancement layer are sent in groups using NumberSplit. The nonscalable coder is defined in [8]. It uses the method of Section II to code atom positions, and is identical to the scalable coder in all other respects.

Table I compares the total bit rates required to send the scalable bit stream, with 50 atoms in the base layer and 100 atoms in the enhancement layer for the groups of various sizes using NumberSplit and using the fixed Huffman-tables-based method described in Section II. The table was generated using 10 s of the *Container* sequence at 7.5 frames/s. The gains, ranging from 10.7 to 2.1%, are especially pronounced for small groups of atoms whose statistics are described poorly by fixed Huffman tables.

Several observations can also be made from the results in Fig. 5: First, as the group size is increased, the bit rate required to achieve the same PSNR value drops, in agreement with the results of Section III. In fact, for the case where there are 50 atoms in the base layer, using a group size of 5 atoms instead of 100 atoms produces a 10% increase in bit rate for the full enhancement layer, but allows for 900 bit/s levels of granularity.

The second observation to be made from Fig. 5 is that the PSNR of the enhancement layer improves as the relative number of the atoms in the enhancement layer to base layer decreases. This happens because more bits are being allocated to the base layer, so that the images in the prediction loop are of better quality, and enhancement layer frames encode much less important structural information. These enhancement layer frames can be coded better with fewer bits than the enhancement layer frames corresponding to a smaller base layer and coded with more bits. The relationship between the quality of a base layer and the corresponding enhancement layer is nonlinear.

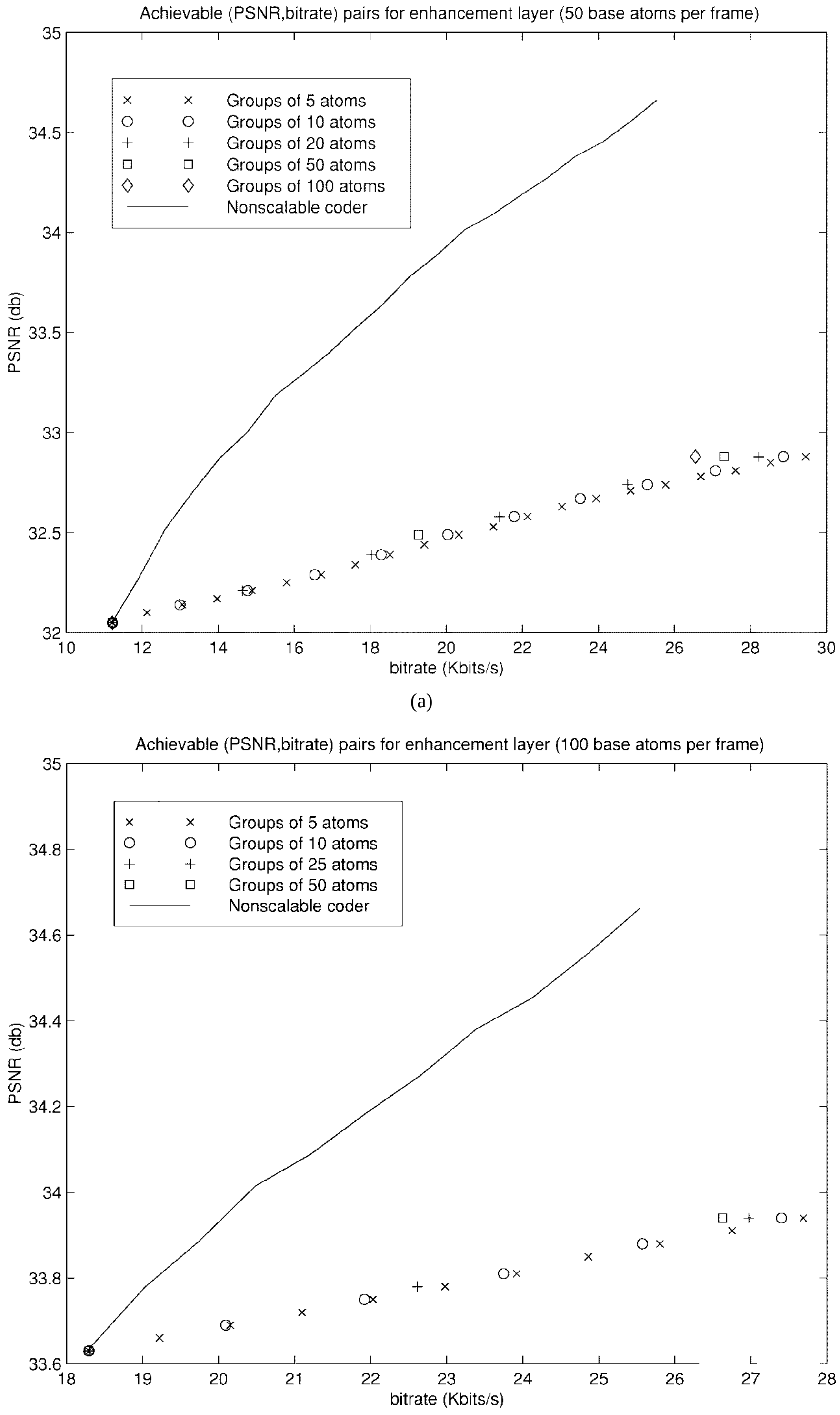
Finally, comparing the PSNR after the enhancement layer with that of a nonscalable coder, the loss is between 0.72 and 1.78, depending on the atom allocation. This is mainly due to the fact that refinements produced by atoms in the enhancement layer are not propagated to the next frame via motion compensation. In the next section, we discuss a method that comes closer to the performance of nonscalable codec.

B. Using Two Residuals

Fig. 5 shows that using a single layer for motion compensation is not efficient. To improve the coding efficiency, we will investigate schemes based on two residual images (Fig. 6). This method of coding multiple residuals is similar to predictive coding of EP frames in H.263+ Annex O [3]. The first residual is the base layer residual image, i.e., the image reconstructed using only the atoms of the base layer, and the second residual image is the enhancement layer residual image. While constructing both residual images, we use the same motion vectors that were computed using the base layer image. Atom positions are coded using the method described in [8].

Since the enhancement layer will use the atoms of the base layer, the choice of the base layer atoms will affect the quality of both the base and enhancement layers. One way to control the quality of both layers is by using different atom allocations. That is, if we want to improve the quality of the base layer, we allocate more atoms to the base layer. However, this implies increasing the bit rate of the base layer, which is usually predetermined by the application at hand. In some of these applications, one is more concerned with having good enhancement layer images, while in others, better base layer images are desired. In the remainder of this section, we propose a way of adjusting the quality between base and enhancement layers while keeping their relative bit rates fixed.

Our approach is to: 1) consider both the base and enhancement layer residuals when finding the atoms that belong to both layers, and 2) only consider the enhancement residual when finding the remaining atoms that only belong to the enhancement layer. One way to accomplish 1) is to minimize a weighted sum of the error of the base layer and the



(a)

(b)

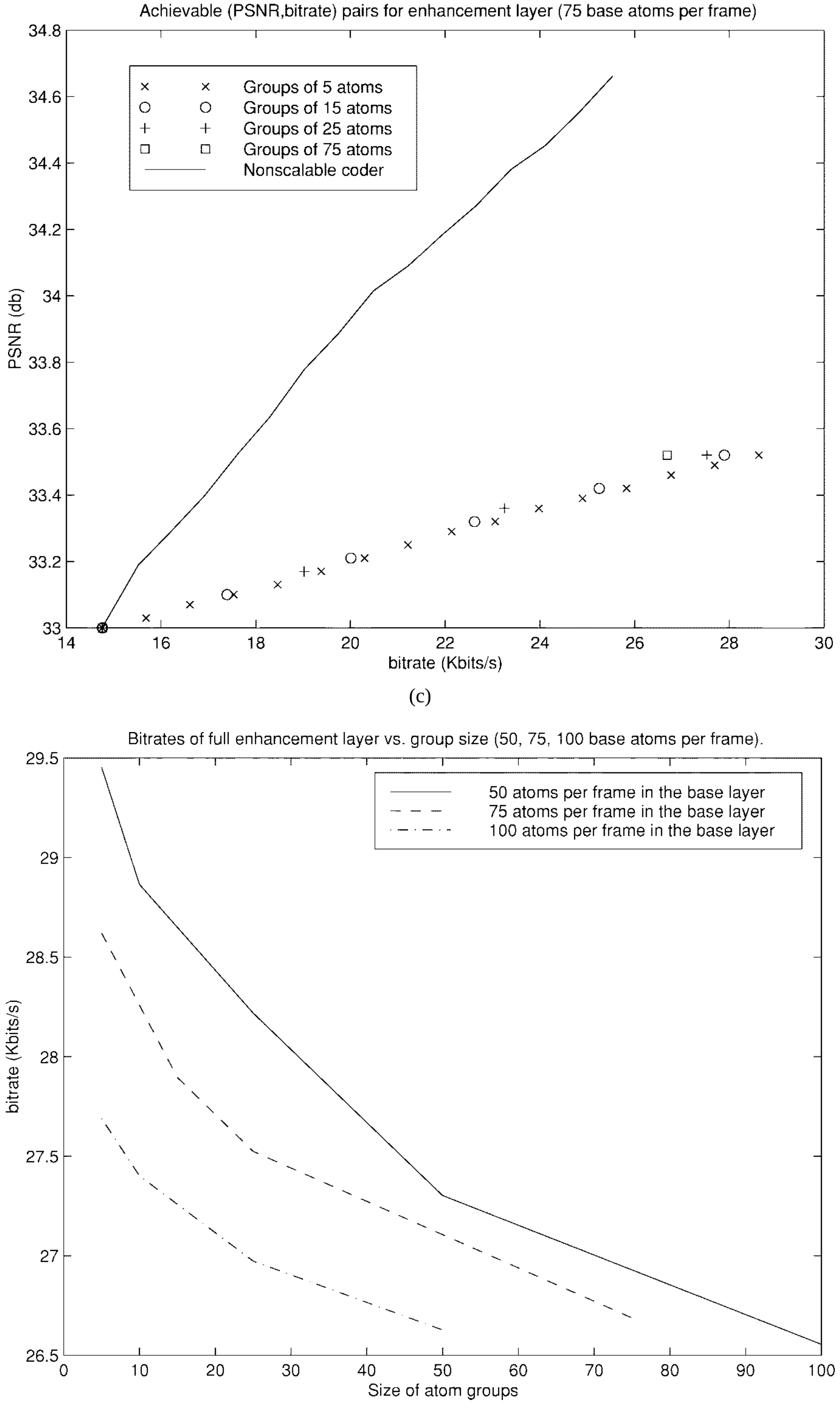
Fig. 5. Achievable (PSNR, bit rate) points for enhancement layer as a function of size of groups in which atoms are coded for (a) 50 atoms coded at the base layer and (b) 75 atoms coded at the base layer as a function of the size of the coding groups.

enhancement layer for finding each atom. That is, if the residual of the base layer is R_b and the residual of the enhancement layer is R_e, we want to minimize

$$J(g,t) = \min_{t,g} \alpha_1 \|R_b - t \cdot g\|^2 + \alpha_2 \|R_e - t \cdot g\|^2 \quad (5)$$

where t is the expansion coefficient and $g \in \mathcal{G}$ is a basis function in the dictionary $\mathcal{G}, \alpha_1$, and α_2 are positive weights that reflect the importance of each layer. The solution to this is to find the basis function that would give the highest inner product, i.e.,

$$g_{\max} = \max_{g \in \mathcal{G}} \frac{|\langle \alpha_1 R_b + \alpha_2 R_e, g \rangle|}{\alpha_1 + \alpha_2} \quad (6)$$



(c)

(d)

Fig. 5 *(Continued.)* Achievable (PSNR, bit rate) points for enhancement layer as a function of size of groups in which atoms are coded for (c) 100 atoms coded at the base layer and (d) bit rates for the full enhancement layer as a function of the size of the coding groups.

and the corresponding transform coefficient is

$$t = \frac{\langle \alpha_1 R_b + \alpha_2 R_e, g_{\max} \rangle}{\alpha_1 + \alpha_2} \tag{7}$$

where $\langle x, y \rangle$ is the inner product of x and y, and $g_{\max}$ is the basis function with the largest absolute inner product with $\alpha_1 R_b + \alpha_2 Re$. Without loss of generality, we assume $\alpha_1 + \alpha_2 = 1.0$, i.e., α_1 and $\alpha_2 \in [0, 1]$.

Fig. 7 shows the effect of α_1 and α_2 on five different atom allocations between the base and enhancement layers. As the value of α_2 increases from 0 to 1, we find that: 1) the PSNR of the base layer decreases by 2–3 dB, depending on atom allocation, and 2) the PSNR of the enhancement layer increases

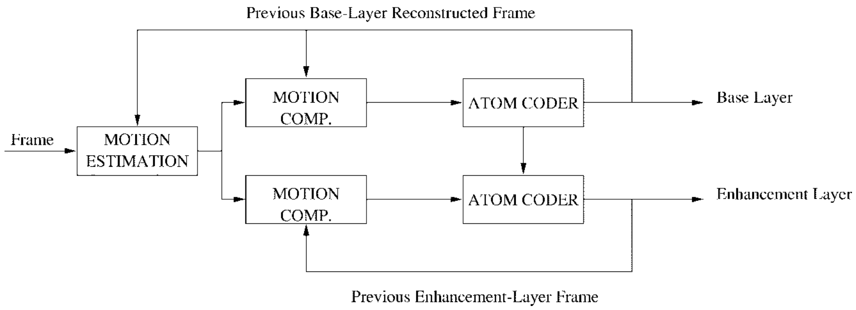
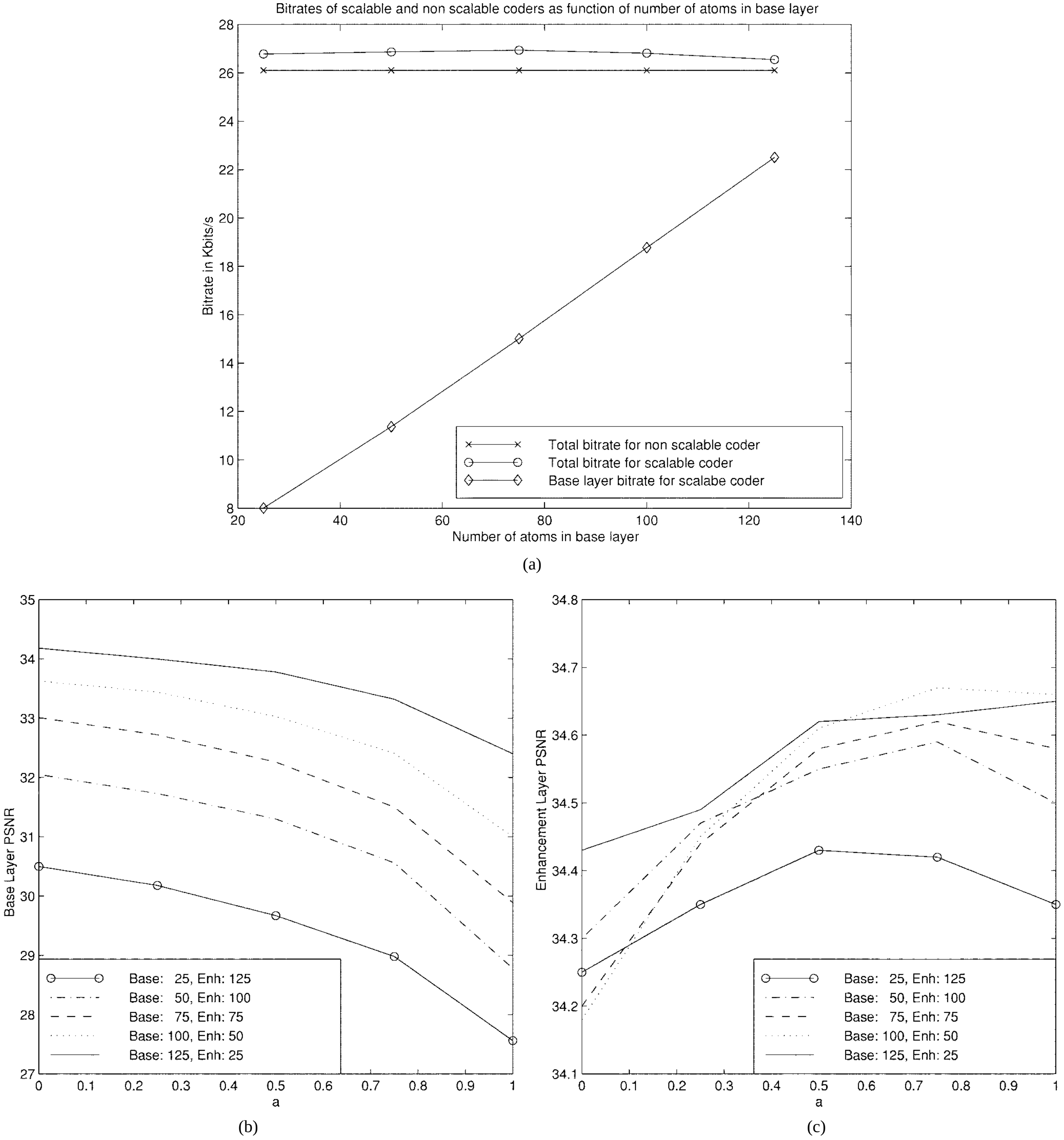


Fig. 6. Block diagram illustrating the two-residual approach for two-layer SNR scalability.

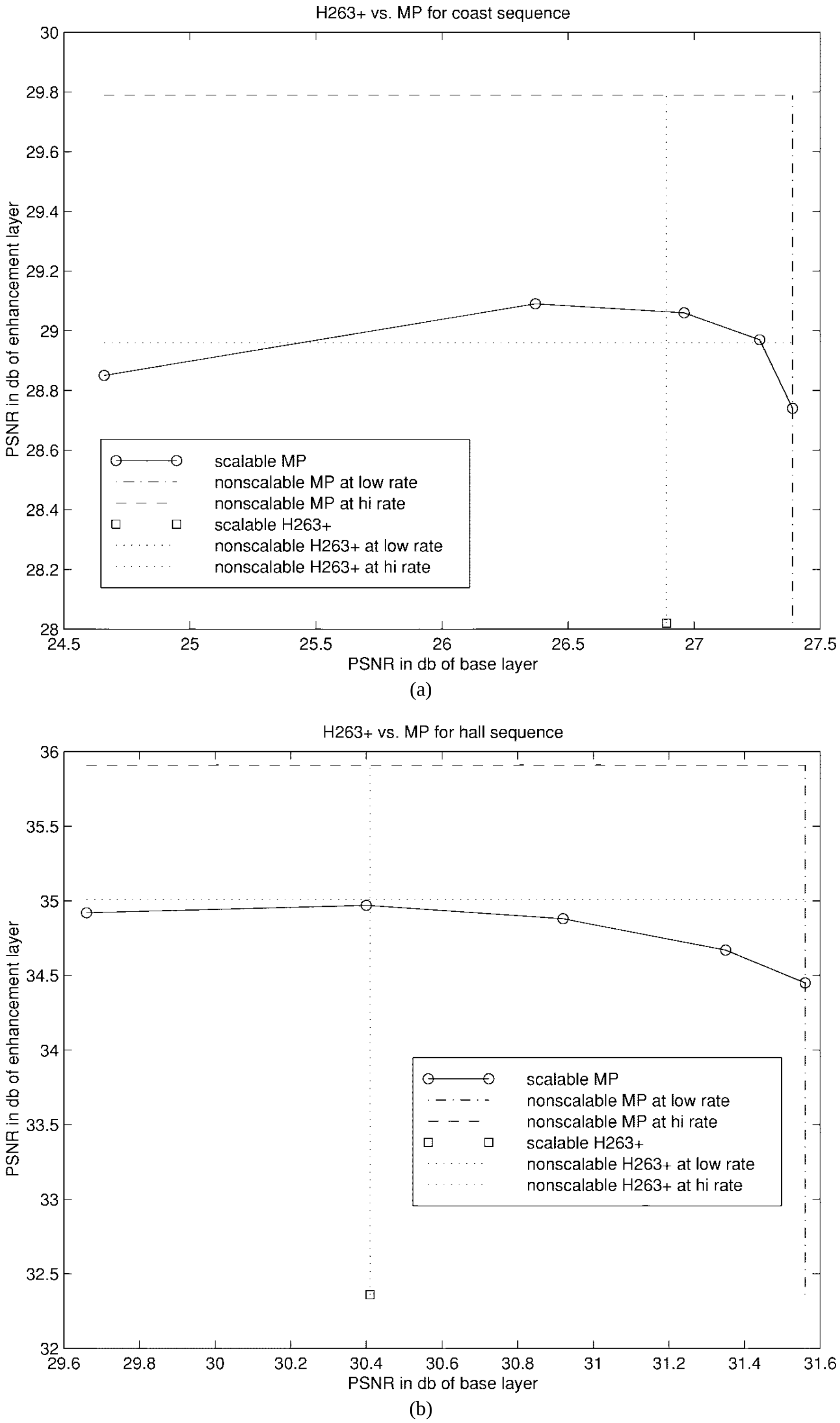


(a)

(b)

(c)

Fig. 7. Approach II for rate scalability where $a = \alpha_2 = 1 - \alpha_1$. (a) Bit rate of the base and enhancement layer. (b) PSNR of the base layer. (c) PSNR of the enhancement layer. The legend describes the atom distribution between both layers.



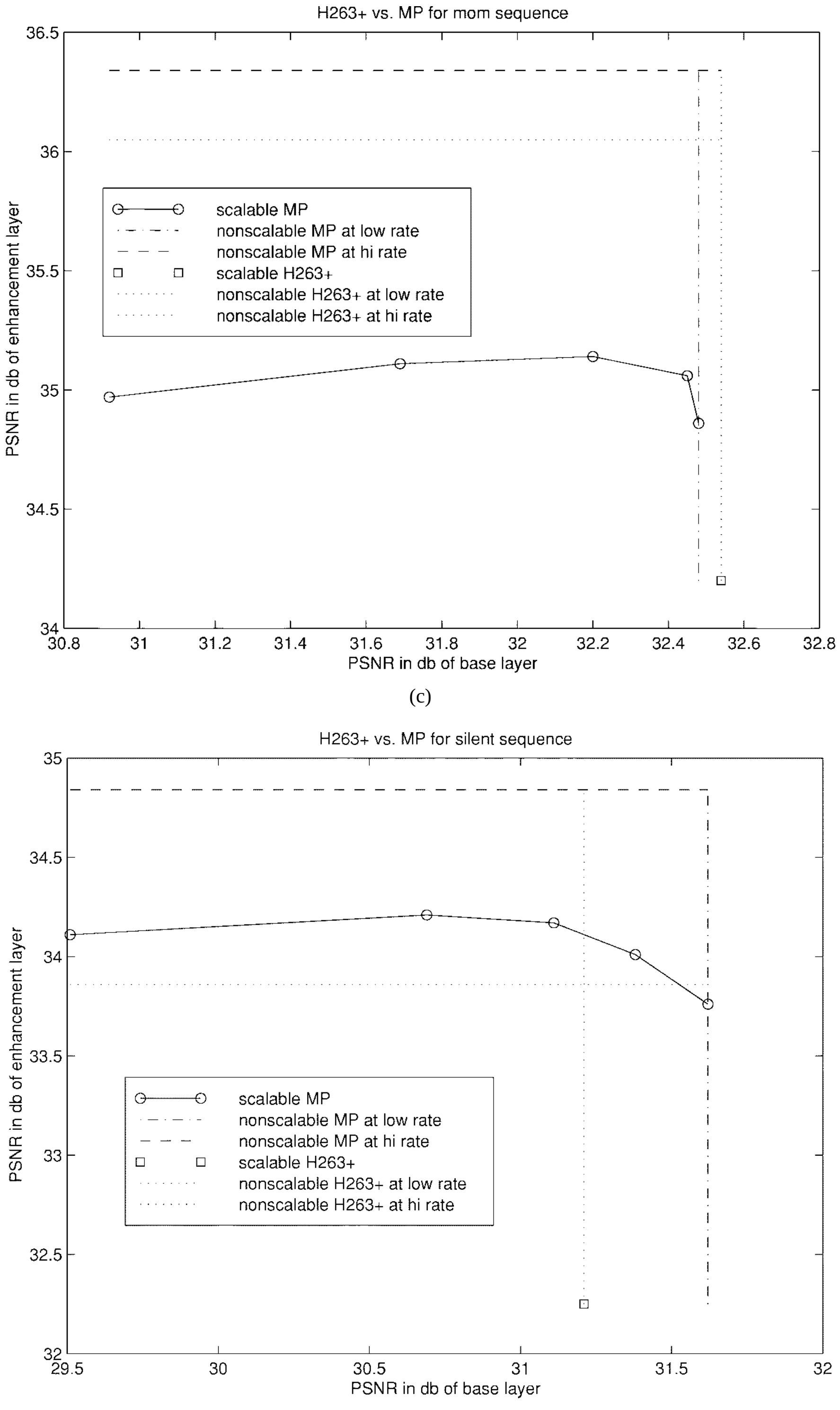
(a)

(b)

Fig. 8. Comparison of H.263+ and MP coders in scalable and nonscalable modes. (a) Coast Guard. (b) Hall Monitor.

by about 0.1–0.5 dB, depending on the relative number of atoms between the base and enhancement layers. Note that the PSNR change in the enhancement layer is considerably smaller than that of the base layer as α_2 changes from 0 to 1. As seen, for a given rate for the enhancement and base layers, one can trade off the relative PSNR performance of the two layers by choosing the appropriate values of α_1 and α_2.

Another interesting conclusion to be drawn from Fig. 7 is that, if the quality of the enhancement layer is much more important than that of the base layer, then $\alpha_2 = 1$ is a better choice than $\alpha_2 = 0$. An example of a situation like this is in applications where scalable video is used over a time-varying channel such as the Internet. In this case, if the available bandwidth is at full capacity most of the time, it is worthwhile to keep the enhancement layer at as high a quality as possible at the expense of the base layer. On the other hand in applications where the base and enhancement layers are equally important, it is more reasonable to operate



(c)

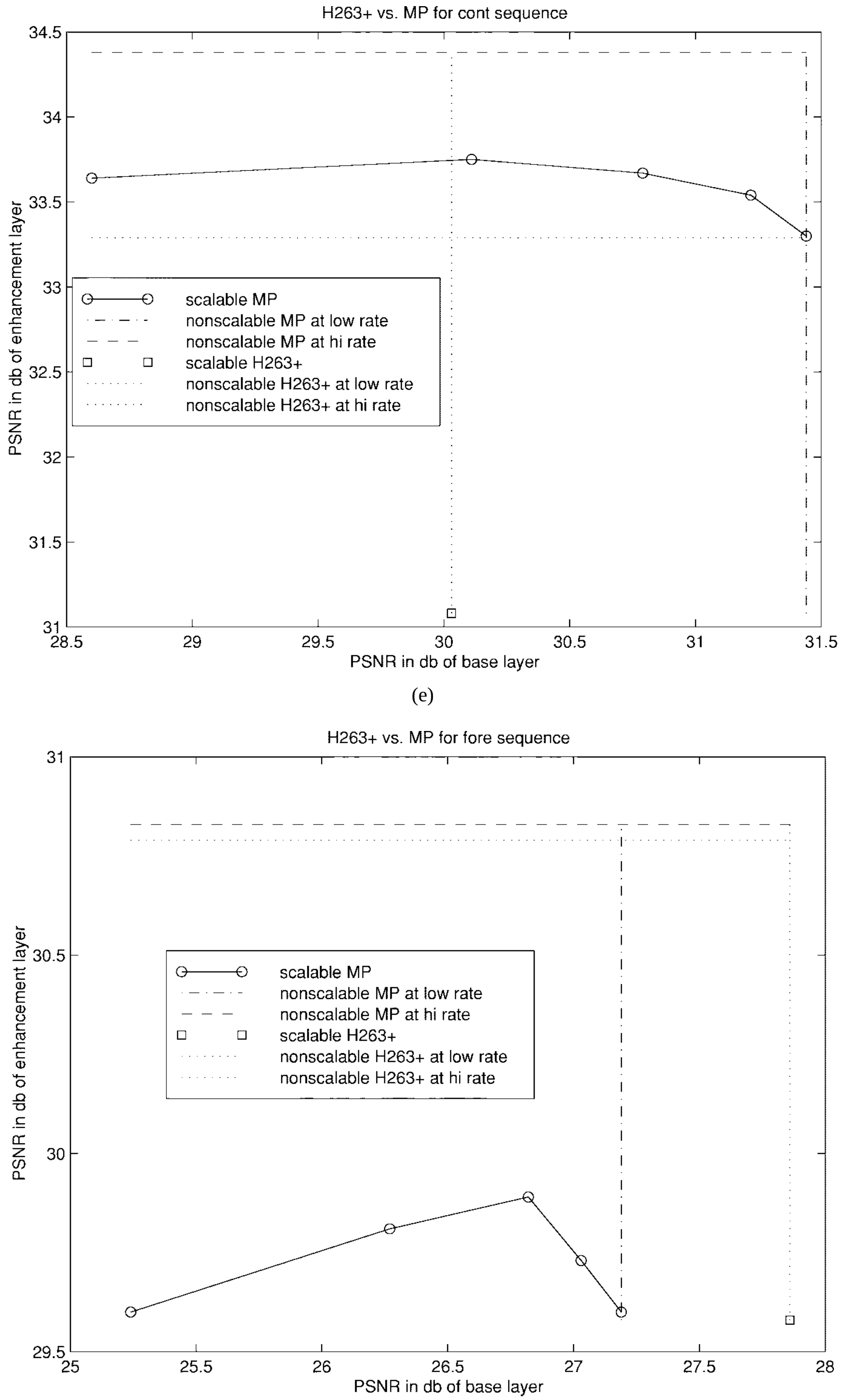
(d)

Fig. 8. *(Continued.)* Comparison of H.263+ and MP coders in scalable and nonscalable modes. (c) Mother-Daughter. (d) Silent Voice.

at $\alpha_2 = 0$ rather than $\alpha_2 = 1$. An example of such situation is transmission over time-varying channels such as the Internet where one expects the channel to spend 50% of the time at higher bandwidth and the other 50% at lower bandwidth.

It is also interesting to compare the PSNR and bit-rate performance of the enhancement layer to the case where the bit stream is not scalable. The nonscalable codec with 150 atoms achieves a PSNR of 34.65 dB and a bit rate of 26.1 kbit/s. Comparing this with the results in Fig. 7, the enhancement layer of the scalable coder has 0.5 dB lower PSNR performance at a 0.9 kbit/s higher bit rate. This is in contrast with the results obtained in Section IV-A where there is a much larger gap between the performances of the scalable and nonscalable codecs.

Another interesting observation to be made from Fig. 7 is that the PSNR of the enhancement layer increases up to



(e)

(f)

Fig. 8. *(Continued.)* Comparison of H.263+ and MP coders in scalable and nonscalable modes. (e) Container. (f) Foreman.

some α_2, and then decreases.[1] This is because a higher α_2 results in degrading the quality of the base layer, which is used for motion estimation. This results in less precise motion estimation, reducing, in turn, the quality of the base layer reconstructed images and enhancement layer reconstructed images.

To summarize, the values of α_1 and α_2 can be thought of as a knob that controls the quality of the resulting images without affecting the bit rate. We should also mention that the decoder does not need to know the values of α_1 and α_2 since they are specified at the encoder.

1) Comparison with H.263+ Coder: In this section, we compare the performances of the MP coder described in Section IV-B and the DCT-based H.263+ coder [3]. There are

[1] This is more noticeable for the case when the number of atoms of the enhancement layer is much larger than the number of atoms of the base layer.

TABLE II
VARIOUS PARAMETERS USED IN COMPARISON OF MP AND H263+; ALL SEQUENCES ARE QCIF

Sequence	Container	Hall	Mom	Silent	Coastguard	Foreman
Base layer QP	16.00	17.00	14.00	13.00	22.00	24.00
Enh. layer QP	12.00	10.00	9.00	9.00	17.00	16.00
Non scalable coder QP	9.00	8.00	7.00	8.00	14.00	13.00
Framerate (frames/s)	7.50	7.50	7.50	10.00	10.00	10.00
Base layer and MP non scalable bitrates (Kbits/s)	10.63	10.14	9.29	22.96	24.46	24.81
Enh. layer total bitrates and MP non scalable bitrates (Kbits/s)	22.76	23.14	23.50	44.82	47.77	46.68
Bitrates (Kbits/s) for H263+ non scalable coder	22.25	24.32	25.21	41.42	46.62	48.60

a number of important differences between the H.263+ scalable codec and the scalable codec described in Section IV-B. First, in our proposed MP codec, the same set of motion vectors is used for the base and enhancement layers, whereas in H.263+, two sets of motion vectors are used. Second, unlike H.263+, where enhancement layer frames are predicted bidirectionally from the previous enhancement layer frame and current base layer reference frame, the MP enhancement layer frames are only predicted from the previous enhancement layer frame. Finally, as expected, the MP codec in Section IV-B uses the matching pursuit algorithm for coding the residuals, while H.263+ uses DCT. In our comparisons, we have used the publicly available version 3.1.2 implementation of the H.263+ standard from the University of British Columbia [13].

Fig. 8 shows the plot of the base and enhancement layers' PSNR's for MP and H.263+ scalable and nonscalable codecs for six different sequences. The circles correspond to PSNR's of scalable MP with values of α_1 varying from 0 to 1 in increments of 0.25. In all comparisons, rate control is determined by running the H.263+ coder with a fixed quantization step size for all frames in the sequence. The MP coder uses the same intraframes as the H.263+ coder, and the same number of bits for both the base and enhancement layers of each frame up to a precision of about 50 bits. Since the quantization step size in the H.263+ coder only takes integer values from 1 to 31, the total bit rates of the scalable and non scalable H.263+ runs are slightly different. Nonscalable MP runs are based on the first frames and bit rates generated by nonscalable H.263+ runs, with the bits spent on all but the first frame prorated in such a way that they add up to the total bit rate of the enhancement layer for the scalable run of H.263+. This way, scalable MP, scalable H.263+, and nonscalable MP runs use the same number of bits, with nonscalable H.263+ runs producing slightly different bit rates. Various parameters used to generate the results in Fig. 8 are summarized in Table II.

From Fig. 8, we see that for most sequences, if we keep the base layer PSNR for MP and H.263+ identical by exploiting the alpha factor, MP outperforms H.263+ by 0.5–2.5 dB at the enhancement layer. This is true for all sequences except for *Foreman* and *Mother-Daughter* sequences where MP does worse at the base layer. We also see that the difference between performances of MP scalable and nonscalable coders for the enhancement layer is between 0.63 and 1.46 dB, depending on the sequence and values of α_1 and α_2, while for the H.263+ coder, the gap ranges from 0.84 to 2.65 dB. Also, with the exception of the *Foreman* and *Mother-Daughter* sequences, the MP nonscalable coder outperforms the H.263+ nonscalable coder at both bit rates. The MP coder also allows us to flexibly trade off the PSNR of the base and enhancement layers, as discussed in Section IV-B, providing a continuum of operating points. As α_1 changes from 0 to 1, the change in PSNR is much larger in the base layer than in the enhancement layer. So, from a practical point of view, $\alpha_1 = 0.75$ represents a good compromise between the qualities of the base and enhancement layers.

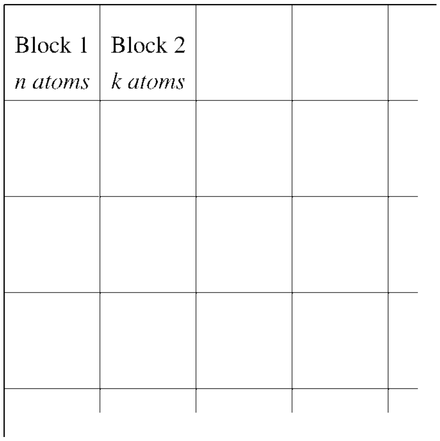


Fig. 9. Example illustrating weighted search. There are n and k atoms found in blocks 1 and 2 of the residual frame, respectively.

V. CODING EFFICIENCY IMPROVEMENTS

In this section, we propose two ways of improving the baseline algorithm proposed in [8]. In Section V-A, we modify the basic search strategy in [8] to find atoms. In Section V-B, we modify the position coding scheme in [8] in order to make it more error resilient.

A. Weighted Energy Search

In this section, we propose a new search strategy for the basic video coding algorithm in [8]. The search strategy in [8] first determines the block with the highest energy, and then uses the center of that block as the center of exhaustive search

TABLE III
WEIGHTS FOR ENERGY SEARCH THAT REFLECT THE ENERGY DECREASE AFTER EACH ATOM

Number of visits	1	2	3	4	5	6	7	8	9	10
Weight	1.00	.590	.470	.416	.374	.354	.333	.326	.301	.290
Number of visits	11	12	13	14	15	16	17	18	19	20
Weight	.279	.258	.235	.235	.223	.217	.201	.182	.180	.180

TABLE IV
WEIGHTS FOR ENERGY SEARCH AND OPTIMIZED TO INCREASE THE AVERAGE PSNR

Number of visits	1	2	3	4	5	6	7	8	9	10
Weight	1.00	.590	.440	.350	.307	.268	.240	.225	.210	.200
Number of visits	11	12	13	14	15	16	17	18	19	20
Weight	.200	.200	.200	.200	.200	.200	.200	.200	.200	.200

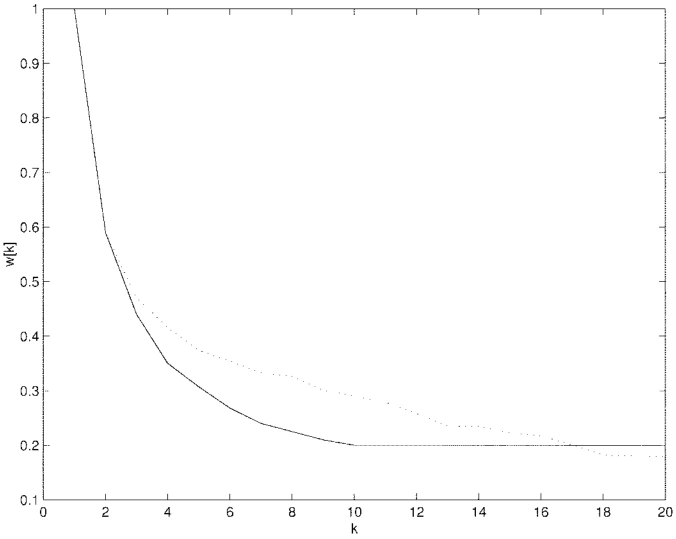


Fig. 10. Heuristic weights (solid line) and weights computed as the normalized energy decrease (dotted line).

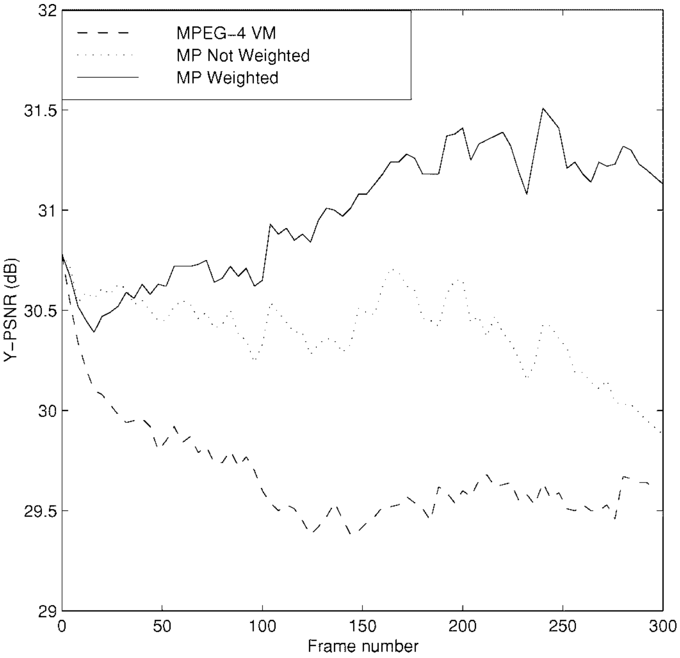


Fig. 11. Luminance PSNR versus frame number for MPEG-4 verification model (dash–dotted line), unweighted search matching pursuit (dotted line), and weighted search matching pursuit (solid line). The sequence used is *Container Ship* coded at 7.5 frames/s and 10 kbit/s. All frames except the first are P-frames, and the number of bits per frame matches in all approaches.

over an $S \times S$ region. In [12], Banham and Brailean modify the basic search strategy in such a way that the blocks closer to the center of the image are more likely to be chosen for exhaustive search. This strategy is based on an assumption that the most important information is located in the center of a frame. The motivation behind our modified search strategy is that the basic search strategy in [8] does not necessarily result in the most rapid energy decrease of the residual signal. From the coding efficiency point of view, it is highly desirable to reduce the residual energy with as few atoms as possible. In the remainder of this section, we will propose a new strategy for finding the best block whose center is subsequently used for exhaustive search.

Recall that Mallat and Zhang [6] have shown that if we are representing a signal f using an overcomplete set and matching pursuits, then the relationship between the energy of the residual after $n+1$ atoms, $||R^{n+1}f||$, and the energy of the residual after n atoms $||R^n f||$ is

$$||R^{n+1}f||^2 \leq ||R^n f||^2 (1 - w[n]) \tag{8}$$

where $w[n] = \lambda^2(R^n f)$ is the rate of decrease, which depends on the correlation between $R^n f$ and the basis functions, and is defined as

$$\lambda(R^n f) = \sup_{\gamma \in \mathcal{G}} \frac{|\langle R^n, g_\gamma \rangle|}{||R^n f||}. \tag{9}$$

This means that

$$\Delta R^{n+1} = ||R^n f||^2 - ||R^{n+1} f||^2 \geq w[n] \cdot ||R^n f||^2 \tag{10}$$

where ΔR^{n+1} is the energy decrease after coding atom $n+1$. That is, the energy decrease after $n+1$ atoms is bounded by the energy after n atoms weighted by a factor that depends on the correlation between the residual after n atoms and the basis set. Using (10), the weights are related to the energies by

$$w[n] \leq \Delta[n] = \frac{||R^{n+1}f||^2 - ||R^n f||^2}{||R^n f||^2} \tag{11}$$

where $\Delta[n]$ is the normalized energy decrease.

TABLE V
Comparison Among Unweighted Search (UW), Weighted Search with Weights Shown in Table I (WO), and Weighted Search with Weights Shown in Table II (WH)

Sequence	Format	Rate		UW	WO	WH	WH - UW
		frame	Bit				
Container-ship	QCIF	10 K	7.5	30.41	30.94	30.99	+0.58
Hall-Monitor	QCIF	10 K	7.5	30.55	31.12	31.17	+0.62
Mother-Daughter	QCIF	10 K	7.5	32.73	32.66	32.74	+0.01
Container-ship	QCIF	24 K	10.0	34.05	34.23	34.21	+0.16
Silent-Voice	QCIF	24 K	10.0	31.17	31.72	31.71	+0.54
Mother-Daughter	QCIF	24 K	10.0	35.46	35.54	35.56	+0.20
Coast-Guard	QCIF	48 K	10.0	29.95	29.95	29.84	-0.11
Foreman	QCIF	48 K	10.0	30.64	30.79	30.78	+0.14

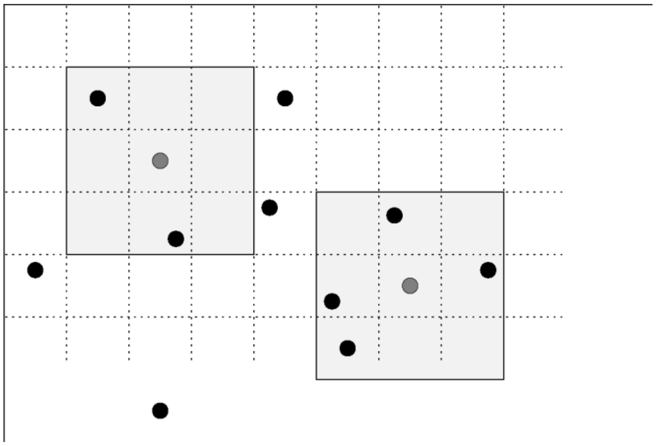


Fig. 12. The effect of error if the gray atom was contaminated for the mode where atoms in each macroblock are coded separately. The dotted lines are the boundaries of the macroblocks. The black atoms are received correctly, while gray ones are contaminated. The light gray area is the area that may be affected by the error.

The main idea behind our proposed search strategy is that the expected energy decrease for each block diminishes as more atoms are coded in that block. As such, blocks with few coded atoms may be better search candidates for reducing the energy than those with higher energy, but with more atoms.

This concept can be used in predicting the block that will decrease the energy the most. For example, while searching for the best block in Fig. 9, the following is true:

$$\Delta R_1^{n+1} \geq w_1[n] \| R_1^n f \|^2 \tag{12}$$

$$\Delta R_2^{k+1} \geq w_2[k] \| R_2^k f \|^2 \tag{13}$$

where R_1 and R_2 are the residuals of blocks 1 and 2, respectively, n and k are the number of atoms found in blocks 1 and 2, respectively, and

$$w_1[n] = \lambda^2(R_1^n) \tag{14}$$

$$w_2[n] = \lambda^2(R_2^n). \tag{15}$$

Thus, if we can estimate the weights w_1 and w_2, we obtain a lower bound on the energy decrease using (12) and (13).

Computing the weights using (9) involves an exhaustive search to find the highest inner product from each block, and use them to compute the weights or pick the one that reduces the energy the most. This is very computationally intensive.

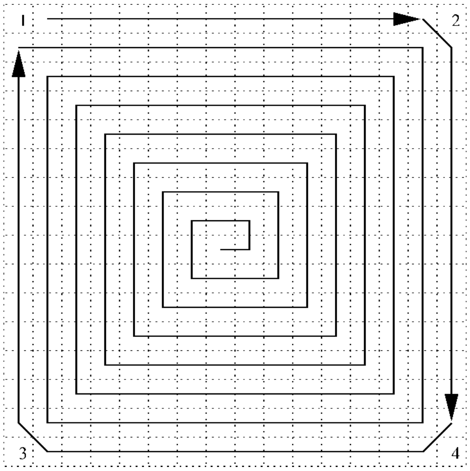


Fig. 13. Scan used to code the atoms in a macroblock. First pixels 1, 2, 3, and 4 are coded.

TABLE VI
Macroblock Types

Mode	Atoms	Code	Mode	Atoms	Code
INTRA	Yes	101110	INTRA	No	101111
INTER	Yes	110	INTER	No	111
INTER4V	Yes	1010	INTER4V	No	10110
INTER0	Yes	100	INTER0	No	0

To reduce the complexity, we assume that the weights of all blocks in the frame are the same, i.e., $w_i[n] = w[n]$, for all i. This implies that the weights are only a function of the number of atoms already in the block, and that they are independent of the location of the block. This reduces our problem to estimating a set of weights to be used for the whole frame or sequence. Our approach is to estimate the weights from training sequences. We use the unweighted search-matching pursuits to choose the search area as described in [8]. We compute the normalized energy decrease $\Delta[n]$ after finding each atom per block, where n is the number of times the current block has been visited so far. That is, we find the decrease in the energy of the block after finding the atom, and normalize it to the energy of the block before finding the atom. We then average $\Delta[n]$ for all three training sequences. Then the weights $w[n] = \Delta[n]$ are normalized to have a maximum of 1.0. The last step is unnecessary, however, since it indicates

TABLE VII
AVERAGE PSNR OF DIFFERENT SEQUENCES AT DIFFERENT BIT RATES FOR THE FRAME-BASED POSITION MP CODER [8] AND THE MACROBLOCK-BASED POSITION MP CODER

Sequence	Format	Rate		PSNR (dB)			
		Bit	Frame		Frame-based	Macro-block-based	Difference
CONTAINER-SHIP	QCIF	10 K	7.5	Y	31.29	31.40	+0.11
				Cb	37.10	37.22	+0.12
				Cr	37.41	37.53	+0.12
HALL-MONITOR	QCIF	10 K	7.5	Y	31.58	31.77	+0.19
				Cb	36.39	36.52	+0.13
				Cr	39.50	39.74	+0.24
MOTHER-DAUGHTER	QCIF	10 K	7.5	Y	33.06	33.21	+0.15
				Cb	38.58	38.86	+0.28
				Cr	39.97	39.97	0.00
CONTAINER-SHIP	QCIF	24 K	10.0	Y	34.27	34.38	+0.11
				Cb	40.33	40.50	+0.17
				Cr	40.45	40.40	-0.05
SILENT-VOICE	QCIF	24 K	10.0	Y	32.04	32.22	+0.18
				Cb	36.35	36.71	+0.36
				Cr	37.53	37.74	+0.21
MOTHER-DAUGHTER	QCIF	24 K	10.0	Y	35.68	35.80	+0.12
				Cb	40.72	40.93	+0.21
				Cr	41.65	41.77	+0.12
COAST-GUARD	QCIF	48 K	10.0	Y	30.02	30.05	+0.03
				Cb	40.27	40.32	+0.05
				Cr	42.82	42.93	+0.11
FOREMAN	QCIF	48 K	10.0	Y	31.11	31.11	0.00
				Cb	37.85	37.81	-0.04
				Cr	38.27	38.36	+0.09
NEWS	CIF	48 K	7.5	Y	32.18	32.44	+0.26
				Cb	37.28	37.74	+0.46
				Cr	38.13	38.34	+0.21
COAST-GUARD	CIF	112 K	15.0	Y	27.50	27.64	+0.14
				Cb	38.07	38.44	+0.37
				Cr	41.22	41.46	+0.24
FOREMAN	CIF	112 K	15.0	Y	30.05	30.11	+0.06
				Cb	36.40	36.45	+0.05
				Cr	37.42	37.72	+0.30
NEWS	CIF	112 K	15.0	Y	35.40	35.52	+0.12
				Cb	39.97	40.19	+0.22
				Cr	40.44	40.63	+0.19
MOBILE-CALENDAR	SIF	1 M	30.0	Y	26.90	26.92	+0.02
				Cb	33.67	33.68	+0.01
				Cr	33.34	33.37	+0.03
STEFAN	SIF	1 M	30.0	Y	29.51	29.58	+0.07
				Cb	36.45	36.56	+0.11
				Cr	36.36	36.43	+0.07

the relative energy decrease expected between a block visited once and another visited n times. Table III shows the resulting weights using this approach.

Another approach to find the weights by training is by optimizing each weight individually by maximizing the PSNR of a set of training sequences. That is, we fix $w[1] = 1.0$, then we vary $w[2]$ in order to maximize the average PSNR of the training sequences. After that, we fix $w[2]$ and start tweaking $w[3]$, and so forth. Table IV shows the resulting weights using this technique. Fig. 10 shows that these weights are very similar to the weights that reflect the energy decrease, and they have the same trend.

Fig. 11 compares the luminance PSNR of each frame for weighted and unweighted search. It is interesting to note that the PSNR increases as we code more frames for weighted search. Table V compares the average luminance PSNR for the weighted and unweighted searches, showing that weighted search always outperforms unweighted search.

B. Macroblock-Based Position Coding

In video transmission over noisy channels, it is important for bit streams to be robust to transmission errors. It is also important, in case of errors, for the error to be limited to a small region, and not to propagate to other areas. However, in the position coding scheme introduced in [8], the atoms can appear in any position in the frame, and the error cannot be limited to an area if it occurs. This is because the atom parameters are coded using VLC tables.

We address this problem by developing a new position coding mechanism that limits the effect of an error to a macroblock (16×16 pixels) and its immediate neighbors. The new position scheme codes atoms that are in the same macroblock together. Thus, if an error occurs, it would affect a known area in the image with the maximum area shown in Fig. 12. If the maximum size of the basis function is 32, the maximum number of blocks affected is nine, i.e.,

TABLE VIII
AVERAGE PSNR OF DIFFERENT SEQUENCES AT DIFFERENT BIT RATES FOR A DCT CODER (MPEG-4 VM) AND MATCHING PURSUIT CODER

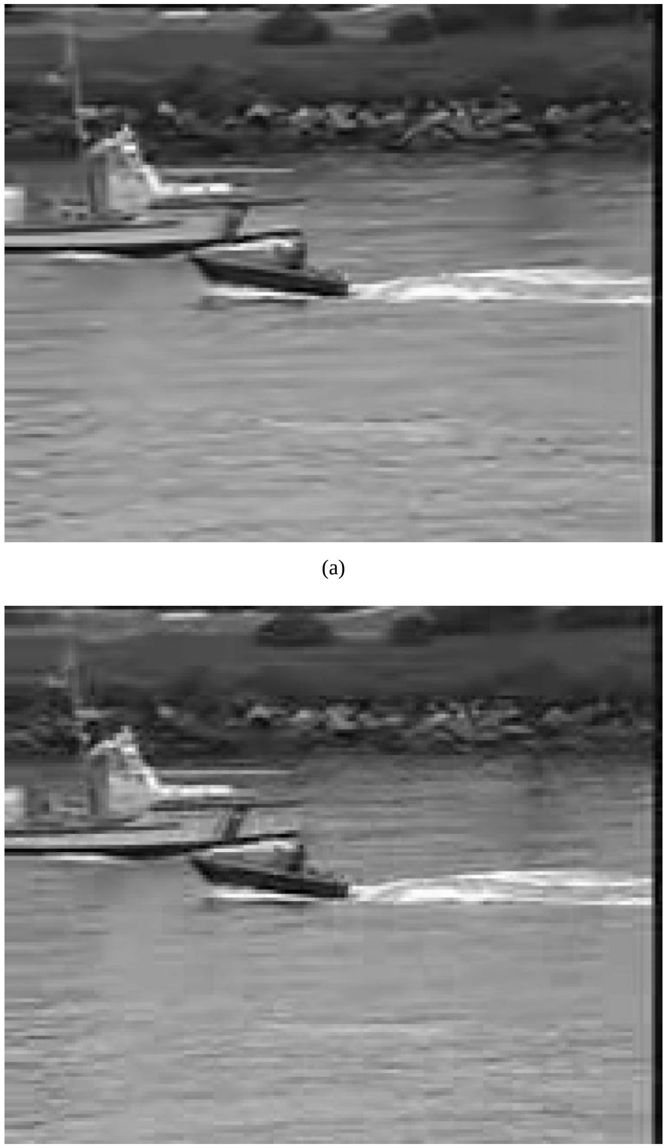
Sequence	Format	Rate		PSNR (dB)			
		Bit	Frame		DCT	MP	MP - DCT
CONTAINER-SHIP	QCIF	10 K	7.5	Y	29.88	31.40	+1.52
				Cb	37.00	37.22	+0.22
				Cr	36.53	37.53	+1.00
HALL-MONITOR	QCIF	10 K	7.5	Y	30.30	31.77	+1.47
				Cb	36.52	36.52	0.00
				Cr	39.63	39.74	+0.11
MOTHER-DAUGHTER	QCIF	10 K	7.5	Y	32.64	33.21	+0.57
				Cb	38.73	38.86	+0.13
				Cr	39.65	39.97	+0.32
CONTAINER-SHIP	QCIF	24 K	10.0	Y	33.38	34.38	+1.00
				Cb	39.45	40.50	+1.05
				Cr	38.63	40.40	+1.77
SILENT-VOICE	QCIF	24 K	10.0	Y	31.04	32.22	+1.18
				Cb	35.26	36.71	+1.45
				Cr	36.93	37.74	+0.81
MOTHER-DAUGHTER	QCIF	24 K	10.0	Y	35.27	35.80	+0.53
				Cb	40.13	40.93	+0.80
				Cr	40.91	41.77	+0.86
COAST-GUARD	QCIF	48 K	10.0	Y	29.42	30.05	+0.63
				Cb	40.00	40.32	+0.32
				Cr	41.90	42.93	+1.03
FOREMAN	QCIF	48 K	10.0	Y	31.14	31.11	-0.03
				Cb	37.22	37.81	+0.59
				Cr	37.39	38.36	+0.97
NEWS	CIF	48 K	7.5	Y	31.14	32.44	+1.30
				Cb	35.93	37.74	+1.81
				Cr	37.39	38.34	+0.95
COAST-GUARD	CIF	112 K	15.0	Y	26.07	27.64	+1.57
				Cb	38.05	38.44	+0.39
				Cr	40.27	41.46	+1.19
FOREMAN	CIF	112 K	15.0	Y	28.66	30.11	+1.45
				Cb	35.41	36.45	+1.04
				Cr	35.90	37.72	+1.82
NEWS	CIF	112 K	15.0	Y	34.23	35.52	+1.29
				Cb	38.27	40.19	+1.92
				Cr	39.08	40.63	+1.55
MOBILE-CALENDAR	SIF	1 M	30.0	Y	26.45	26.92	+0.47
				Cb	30.82	33.68	+2.86
				Cr	30.35	33.37	+3.02
STEFAN	SIF	1 M	30.0	Y	29.49	29.58	+0.09
				Cb	34.85	36.56	+1.71
				Cr	34.48	36.43	+1.95

3 × 3 macroblocks around the macroblock where the error occurred. The idea of coding atoms on a macroblock by macroblock basis to improve error resilience was introduced by Banham and Brailean in [12]. Our method improves the macroblock-based position coding technique described in [12] by increasing the coding efficiency without sacrificing error resilience.

The atoms of each macroblock are reordered according to the scan shown in Fig. 13. Afterwards, the atoms are coded differentially. Four different VLC tables are used to code the atoms depending on the number of atoms in it. With a small loss of coding efficiency, we can code the position of each atom absolutely within a macroblock. This will limit the effect of the loss of an atom only to the support of that atom.

Since the atoms are coded on a macroblock level, we can multiplex them with the motion information. An efficient way to combine them is by defining eight different macroblock types. The eight types are defined because we are using an MPEG-4 (or H.263) motion model that allows either INTRA or INTER macroblocks. INTRA macroblocks are coded independently of the other frames. INTER macro blocks depend on other frames because motion compensation is used to code them. In MPEG-4 and H.263, there are two INTER macroblocks: INTER, which uses one motion vector, and INTER4V, which uses four motion vectors per macroblock. These eight types are: INTRA macroblock without atoms, INTRA macroblock with atoms, INTER macroblock without atoms, INTER macroblock with atoms, INTER4V macroblock without atoms, INTER4V macroblock with atoms, INTER macroblock with zero motion vector (INTER0) without atoms, and INTER0 macroblock with atoms. The INTER0 types were added because they are very common, especially at low bit rates. These types and their codes are summarized in Table VI.

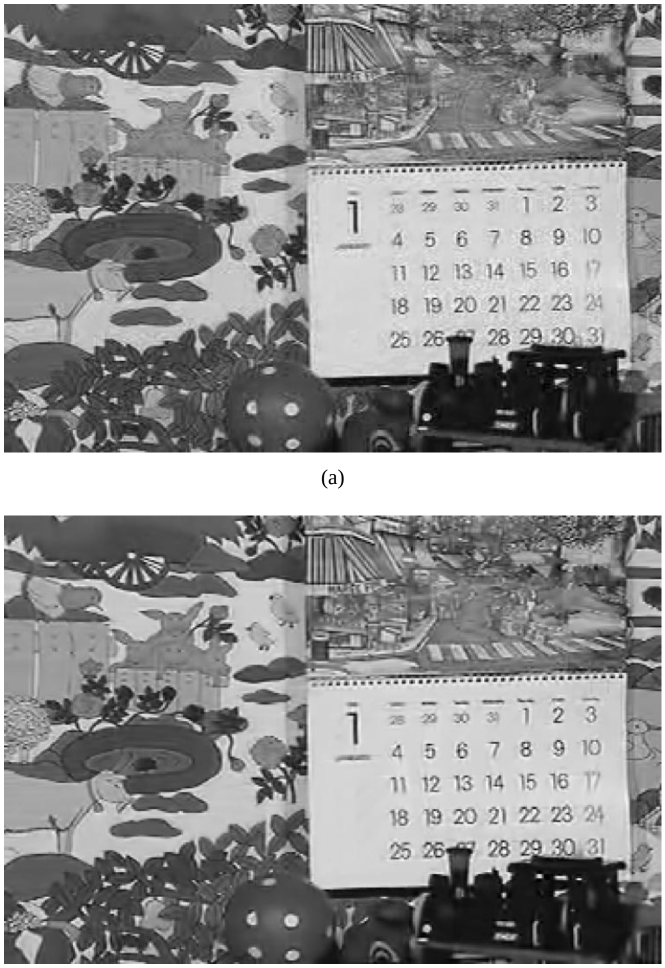
Table VII compares the coding efficiency of the frame-based [8] and macroblock-based position coding schemes. One would expect an error-resilient scheme to be less efficient than a scheme without such properties. However, the average luminance and choma PSNR's of the macroblock-based mode se-



(a)

(b)

Fig. 14. Frame 50 of 10 frame/s *Coast Guard* sequence coded at 48 kbit/s using: (a) DCT-based coder (MPEG-4 VM) and (b) matching pursuit coder. Blocking artifacts can be noticed on the DCT coded frame.



(a)

(b)

Fig. 15. Frame 20 of 30 frame/s *Mobile Calendar* sequence coded at 1 Mbit/s using: (a) DCT-based coder (MPEG-4 VM) and (b) matching pursuit coder. Blocking artifacts can be noticed on the DCT coded frame.

quences is higher than that of the frame-based mode in all but one sequence. The gain ranges between −0.01 and 0.21 dB.

Macroblock-based mode achieves these gains by taking into account the properties of atom distribution and the correlation between motion vectors and atom locations. The scanning order in Fig. 13, combined with differential coding, utilizes the fact that atoms are more likely to lie on the corners and edges of macroblocks to improve coding efficiency. By multiplexing atom presence information with motion vectors, we utilize the fact that atoms are unlikely to appear in a macroblock with no motion. Thus, the macroblock-based mode offers better coding efficiency performance and a potential for better error resilience.

C. Comparison with DCT Approaches

All video compression standards are DCT-based coders [1], [2], [4], [5], so it is of interest to compare the performance of the MP video coder to that of a DCT-based coder. In this section, we compare the performance of the MP coder with that of MoMuSys version VM8-971 006 of the MPEG-4 DCT-based coder. The comparison is done using the sequences tabulated in Table VIII coded at bit rates that range between 10 kbit/s and 1 Mbit/s. The first frame for both approaches is coded using the MPEG-4 DCT INTRA mode. Both first frames are identical, and both coders code each frame with the same number of bits up to the resolution of the MP coder, i.e., 30 bits.

Table VIII shows the average luminance and chroma PSNR's for these different sequences. In all but one example of Table VIII, the matching pursuit coder has a higher average PSNR than the DCT coder. Fig. 14 shows frame 50 of the 10 frame/s QCIF *Coast Guard* sequence coded at 48 kbit/s using the MPEG-4 DCT coder [5] and the MP coder. The DCT coded frame suffers from blocking artifacts. Fig. 16(a) compares the luminance PSNR for each frame of the sequence for the MPEG-4 VM DCT coder and the MP coder. Fig. 15 shows frame 20 of the 30 frame/s SIF *Mobile Calandar* sequence coded at 1 Mbit/s using the MPEG-4 DCT coder [5] and the MP coder. Fig. 15 compares the luminance PSNR per

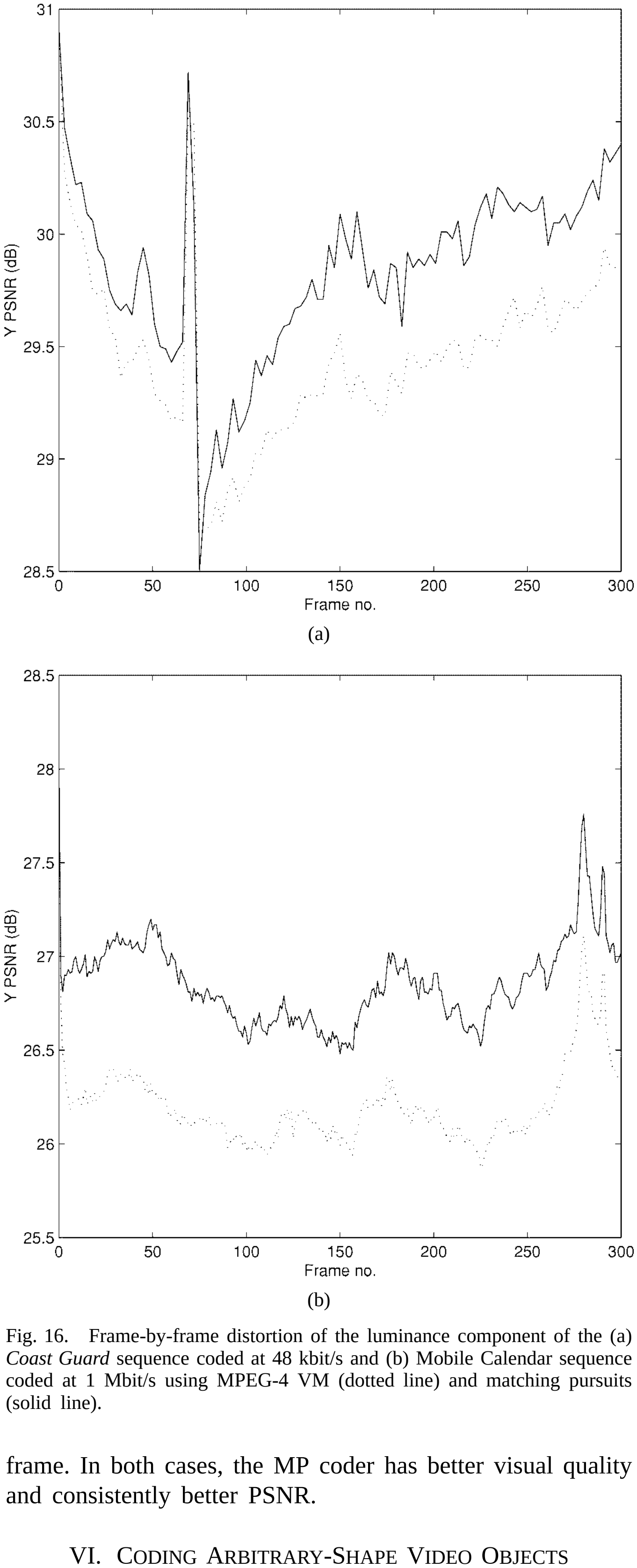


Fig. 16. Frame-by-frame distortion of the luminance component of the (a) *Coast Guard* sequence coded at 48 kbit/s and (b) Mobile Calendar sequence coded at 1 Mbit/s using MPEG-4 VM (dotted line) and matching pursuits (solid line).

frame. In both cases, the MP coder has better visual quality and consistently better PSNR.

VI. Coding Arbitrary-Shape Video Objects

One of the main differences between MPEG-4 and other video coding algorithms is the ability to code and randomly access arbitrary shape objects. This is an important functionality for many applications, e.g., multimedia databases, video games, etc. In recent years, several approaches have been proposed to extend DCT-based techniques in order to handle arbitrary-shaped objects [14]–[17]. In this section, we extend the MP coder to support coding texture of arbitrary-shaped objects. It should be clear, however, that we do not intend here to develop a shape coder based on matching pursuits, but to extend the MP coder to support the coding texture of arbitrary shaped objects. Thus, we will use the MPEG-4 arbitrary-shape coder to code the shape information [5] for all of our experimental results.

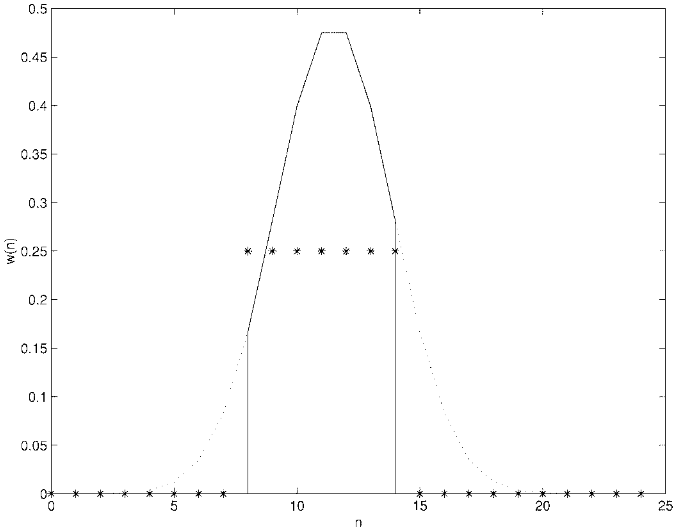


Fig. 17. Padding technique used in computing the inner products. The stars of value 0.25 correspond to pixels that are part of the signal. The solid part of the basis function is used to compute the inner product, and both the inner product and the basis function are renormalized by the norm of this part. The dotted part of the basis function is discarded when computing the inner product.

Two problems should be addressed when extending the MP coder to support arbitrary-shaped objects. The first problem is concerned with computing the inner products and comparing them, especially at the boundaries of the object. The second problem is how to code the atoms within the shape. The second problem mainly deals with position coding.

One solution to the first problem is to pad the object with zero values. Thus, when computing the inner product, we only consider the object pixels. This means that our basis functions have changed on the boundaries to fit the object better. Fig. 17 gives an illustration of how padding is done. The norm of each basis function will change according to the pixels it covers. We compensate for this by renormalizing the basis function and the inner product while searching and reconstructing. Moreover, for better performance, the energy of each block when doing the energy search is weighted by the ratio of the pixels of the block are in the object.

We will now describe two approaches to code the atoms when coding video objects with arbitrary shapes.

1) Bounding Box Approach: The first approach is the simplest extension of the rectangular case. We simply fit the shape inside a rectangular frame as shown in Fig. 18, and use the original matching pursuits coder. It should be noted that since MPEG-4 codes shape in 16 × 16 blocks, we extend the frame to include these blocks, even though their region of support might be outside the object. Fig. 19 shows two frames of

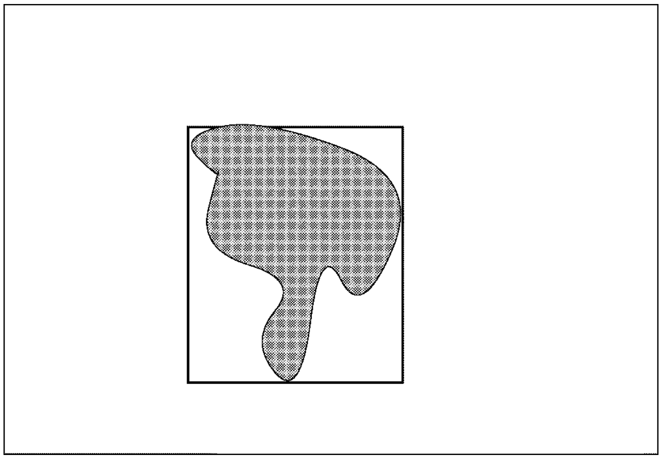


Fig. 18. Approach I to code the atoms by fitting a rectangular frame around the shaded object.

the *Coast Guard* sequence coded using SA-DCT and SA-MP coders. The SA-DCT frame suffers from blocking artifacts, loss of detail, and changes the color of the front of the boat from red to black. The SA-MP frame is cleaner, has more details, and does not have color problems.

2) Coding Only Objects: The second approach only scans the atoms inside the object without fitting the shape with a rectangular frame.

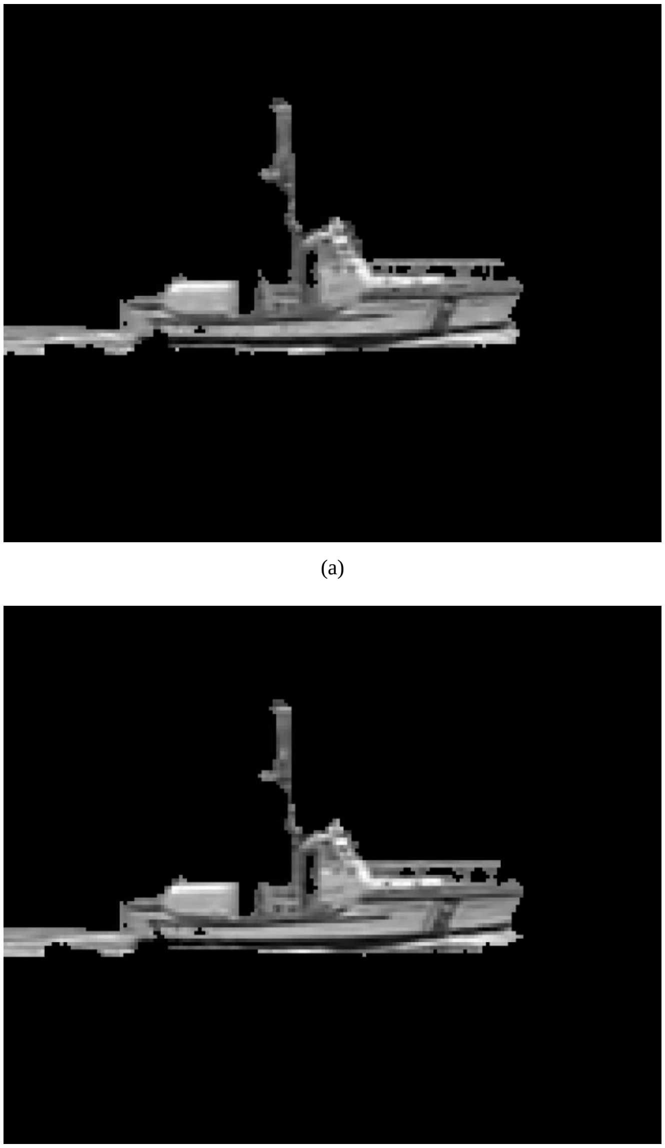
Table IX shows the PSNR performance of this approach, as compared to the bounding box approach and the latest SA-DCT results from the MPEG-4 Verification Model (VM). As seen, there are three sets of results corresponding to the matching pursuit approach.

- MP1 stands for the case where the bounding box approach is used, and the positions of the atoms are coded in the "separate mode" as described in our original codec in [8]; in the separate mode, all of the motion vectors are coded separately from the atoms and their positions. This is in contrast to the "combined" macro block position coding technique described in Section V-B where the residual image is divided into macroblocks and the positions of motion vectors and atoms in each macroblock are coded together.
- MP2 stands for "coding only objects" with separate mode.
- MP3 stands for "coding only objects" with combined mode.

As seen, The MP3 performs better than MP2, which in turn performs better than MP1. The last column in the table compares the performance of the MP3 and SA-DCT technique. As seen, the luminance performance of MP3 is better than that of VM in 11 cases, and worse than VM in two cases. As for chroma, in ten cases, MP3 outperforms VM in both U and V components and in one case, VM outperforms MP3 in both U and V.

VII. Conclusions

In this paper, we extended the basic residual video coding scheme described in [8] to address issues regarding scalability, arbitrary shape coding, and improved coding efficiency. In the area of scalability, we found that the most natural way of generating scalable video using MP is to use the number of coded atoms. We proposed two basic schemes for both fine and coarse level scalability in Sections IV-A and IV-B, respectively, and found that it is possible to achieve a very fine level of rate scalability at the expense of loss in PSNR. For the fine grain scalable codec in Section IV-A, we proposed a new position coding algorithm, NumberSplit, that does not require any Huffman tables and outperforms the original position coding scheme in [8]. For the coarse grain scalable codec in Section IV-B, we developed a way of trading off the quality of the enhancement layer with that of the base layer without changing the bit rate of each layer. This is desirable in applications where the rates of both layers are fixed, but the quality of the reconstructed frames for one layer is more important than the other one.



(a)

(b)

Fig. 19. Frame 30 of the 10 frame/s *Coast Guard* sequence *(big boat)* coded at 16 kbit/s using: (a) SA-DCT-based coder (MPEG-4 VM) and (b) SA-MP coder. Details are lost by the deblocking filter on the DCT coded frame.

TABLE IX
COMPARISON BETWEEN SA-MP AND SA-DCT AS USED IN MPEG-4; VM: SA-DCT, COMBINED MODE; MP1: SA-MP, BOUNDING BOX APPROACH, SEPARATE MODE; MP2: SA-MP, CODING ONLY OBJECTS, SEPARATE MODE; MP3: SA-MP, CODING ONLY OBJECTS, COMBINED MODE

Sequence		VM	MP1	MP2	MP3	MP3-VM
COAST-GUARD	Bit-Rate(Kbits/s)	21.81	21.60	21.66	21.94	
object0	PSNR_Y(dB)	28.94	29.70	29.73	29.82	0.88
(water)	PSNR_U(dB)	45.42	48.84	49.38	49.54	4.42
10frames/s	PSNR_V(dB)	46.14	48.54	49.26	49.53	3.39
COAST-GUARD	Bit-Rate(Kbits/s)	15.71	15.78	15.80	15.79	
object1	PSNR_Y(dB)	27.17	27.45	27.57	27.53	0.36
(big boat)	PSNR_U(dB)	39.66	39.87	39.89	40.24	0.58
10frames/s	PSNR_V(dB)	36.54	39.31	38.87	38.36	1.82
COAST-GUARD	Bit-Rate(Kbits/s)	9.499	9.506	9.489	9.357	
object2	PSNR_Y(dB)	25.72	26.30	26.27	26.33	0.61
(small boat)	PSNR_U(dB)	38.25	38.80	38.97	38.65	0.40
10frames/s	PSNR_V(dB)	37.83	39.70	39.52	38.91	1.08
COAST-GUARD	Bit-Rate(Kbits/s)	10.73	10.86	10.86	10.76	
object3	PSNR_Y(dB)	26.48	27.20	27.15	27.25	0.77
(land)	PSNR_U(dB)	36.47	36.34	36.22	36.34	-0.13
10frames/s	PSNR_V(dB)	42.49	42.39	42.46	42.43	-0.06
CHILDREN	Bit-Rate(Kbits/s)	23.25	23.49	23.43	23.08	
object0	PSNR_Y(dB)	29.09	29.07	29.14	29.16	0.07
(background)	PSNR_U(dB)	28.56	29.26	29.33	29.26	0.70
10frames/s	PSNR_V(dB)	29.84	30.46	30.54	30.63	0.79
CHILDREN	Bit-Rate(Kbits/s)	46.80	47.00	46.95	46.38	
object1	PSNR_Y(dB)	26.60	25.85	26.15	25.88	-0.72
(kids and ball)	PSNR_U(dB)	29.93	30.98	31.08	30.91	0.98
10frames/s	PSNR_V(dB)	29.57	30.86	31.04	30.84	1.27
HALL-MONITOR	Bit-Rate(Kbits/s)	11.31	11.46	11.48	11.22	
object0	PSNR_Y(dB)	31.02	31.44	31.51	31.49	0.47
(background)	PSNR_U(dB)	36.82	36.76	36.77	36.78	-0.04
10frames/s	PSNR_V(dB)	40.34	40.35	40.32	40.35	0.01
HALL-MONITOR	Bit-Rate(Kbits/s)	5.325	5.450	5.441	5.275	
object1	PSNR_Y(dB)	25.39	25.18	25.28	25.53	0.14
(person1)	PSNR_U(dB)	33.61	33.90	34.40	34.16	0.55
7.5frame/s	PSNR_V(dB)	38.04	36.91	37.06	37.10	-0.94
HALL-MONITOR	Bit-Rate(Kbits/s)	3.833	3.950	3.953	3.811	
object2	PSNR_Y(dB)	24.75	24.69	24.84	25.12	0.37
(person2)	PSNR_U(dB)	34.41	34.46	34.44	34.50	0.09
7.5frames/s	PSNR_V(dB)	37.52	37.79	37.94	38.04	0.52
WEATHER	Bit-Rate(Kbits/s)	27.66	27.82	27.81	27.54	
object0	PSNR_Y(dB)	27.52	27.19	27.32	27.33	-0.19
(background)	PSNR_U(dB)	25.96	27.23	27.39	27.38	1.42
7.5frames/s	PSNR_V(dB)	27.91	28.93	29.09	29.01	1.10
WEATHER	Bit-Rate(Kbits/s)	22.43	22.59	22.56	22.17	
object1	PSNR_Y(dB)	28.75	28.70	28.87	28.88	0.13
(Akiyo)	PSNR_U(dB)	30.46	31.46	31.83	31.69	1.23
7.5frames/s	PSNR_V(dB)	35.12	35.78	35.80	35.74	0.62
CONTAINER-SHIP	Bit-Rate(Kbits/s)	5.726	5.870	5.849	5.745	
object0	PSNR_Y(dB)	34.83	35.49	35.63	35.73	0.90
(water)	PSNR_U(dB)	41.79	42.41	42.44	42.47	0.68
7.5frames/s	PSNR_V(dB)	40.86	41.08	41.27	41.38	0.52
CONTAINER-SHIP	Bit-Rate(Kbits/s)	8.923	9.009	9.038	8.945	
object1	PSNR_Y(dB)	26.06	26.94	26.96	27.05	0.99
(ship)	PSNR_U(dB)	33.48	33.78	34.08	34.04	0.56
7.5frames/s	PSNR_V(dB)	31.71	32.85	32.94	32.99	1.28

In the area of arbitrary-shape coding, we found that conventional frame-based MP can be easily extended to MP texture coding of arbitrary shapes. In spite of its simplicity, the shape-adaptive MP outperforms shape-adaptive DCT by as much as 1 dB.

Finally, we demonstrated ways to improve the performance of the matching pursuits video coder as described in [8] by introducing a new weighted energy search and a new "combined" position coding technique. The combined position coding has the added advantage that it makes the MP codec more robust to errors. With these improvements, the performance of the MP codec exceeds that of the DCT-based VM in MPEG-4 by as much as 1.4 dB. In addition, from a visual point of view, the MP coder did not suffer from blocking and ringing artifacts, color bleeding, and loss of detail that are typical for low bit-rate DCT coders.

ACKNOWLEDGMENT

The authors wish to thank G. Côté, B. Erol, M. Gallant, and Prof. F. Kossentini from the University of British Columbia for providing them with the H263+ coder.

References

[1] CCITT Recommendation H.261, *Video Codec for Audio Visual Services at $p \times 64$ kbit/s*, 1990.

[2] CCITT Recommendation H.263, *Video Codec for Audio Visual Services at $p \times 64$ kbit/s*, 1995.

[3] ITU-T Recommendation H.263, *Video Coding for Low Bit Rate Communication*, pp. 107–110, Sept. 1997

[4] Committee Draft of Standard ISO11172, *Coding of Moving Pictures and Associated Audio*, ISO/MPEG 90/176, Dec. 1990.

[5] *MPEG-4 Video Verification Model 8.0*, ISO/IEC JTC1/SC29/WG11 MPEG-97 Doc. W1796, July 1997.

[6] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Processing*, vol. 41, pp. 3397–3415, Dec. 1993.

[7] S. A. Martucci, I. Sodagar, T. Chiang, and Y. -Q. Zhang, "A zerotree wavelet coder," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 109–118, Feb. 1997.

[8] R. Neff and A. Zakhor, "Very low bit rate video coding based on matching pursuits," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 158–171, Feb. 1997.

[9] M. Ghanbari and V. Seferidis, "Efficient H.261-based two-layer video codecs for ATM networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, pp. 171–175, Apr. 1995.

[10] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Processing*, vol. 41, pp. 3445–3462, Dec. 1993.

[11] D. Taubman and A. Zakhor, "Multi-rate 3-D subband coding of video," *IEEE Trans. Image Processing*, vol. 3, pp. 572–588, Sept. 1994.

[12] M. Banham and J. Brailean, "A selective update approach to matching pursuits video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 119–129, Feb. 1997.

[13] G. Côté, B. Erol, M. Gallant, and F. Kossentini, "H.263+: Video coding at low bit rates," submitted to *IEEE Trans. Circuit Syst. Video Technol.* Available WWW: http://www.ee.ubc.ca/image/h263plus.

[14] T. Sikora, S. Bauer, and B. Makai, "Efficiency of shape-adaptive 2-D transforms for coding of arbitrarily shaped image segments," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, pp. 254–258, June 1995.

[15] P. Kauff, B. Makai, S. Rauthenberg, U. Golz, and T. Sikora, "Functional coding of video using a shape-adaptive DCT algorithm and an object-based motion prediction toolbox," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 181–196, Feb. 1997.

[16] T. Sikora, "Low complexity shape-adaptive DCT for coding of arbitrarily shaped image segments," *Signal Processing: Image Commun.*, vol. 7, pp. 381–395, Nov. 1995.

[17] T. Sikora and B. Makai, "Shape-adaptive DCT for generic coding of video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, pp. 59–62, Feb. 1995.

Osama K. Al-Shaykh received the B.Sc. degree with Excellence in 1990 from the University of Jordan, the M.Sc. degree in 1992 from Iowa State University, and the Ph.D. degree in 1996 from Georgia Institute of Technology (all in electrical engineering).

After graduation, he was a Visiting Postdoctoral Researcher at the Video and Image Processing Laboratory, University of California at Berkeley. Since September, 1997, he has been with the multimedia group in Rockwell Science Center. His research interests include image and video processing, medical imaging, neural networks, and pattern recognition.

In 1991, Dr. Al-Shaykh was awarded the Fulbright scholarship and ISU president's scholarship. He is a member of Σ X.

Eugene Miloslavsky received the B.S. degree from the University of California at Berkeley in 1992. He is currentlly working on the Ph.D. degree in the Video and Image Processing lab at the University of California at Berkeley.

His research interests include scalable coding of video, rate control algorithms, video compression, image processing and wavelets.

Toshio Nomura received the B.E. and M.E. degrees in electrical engineering from Kyoto University, Japan, in 1989 and 1991, respectively.

In 1991, he joined Sharp Corporation, Chiba, Japan. From 1997 to 1998, he was a Visiting Industrial Fellow at the University of California at Berkeley. His current research interests include video and image processing.

Ralph Neff is currently a Ph.D. student in the Video and Image Processing (VIP) lab at the University of California at Berkeley. He received the B.S. degree in electrical engineering at University of Illinois at Urbana-Champaign in 1992, and the M.S. degree from Berkeley in 1994 for work on matching pursuit based video compression.

He joined the VIP lab in 1993, and has been a participant in MPEG-4 since 1995.

Avideh Zakhor received the B.S. degree from California Institute of Technology, Pasadena, and the S.M. and Ph.D. degrees from Massachusetts Institute of Technology, Cambridge, all in electrical engineering, in 1983, 1985, and 1987, respectively.

In 1988, she joined the Faculty at the University of California at Berkeley where she is currently Professor in the Department of Electrical Engineering and Computer Sciences. Her research interests are in the general area of signal processing and its applications to images and video, and biomedical data. She has been a consultant to a number of industrial organizations and in 1996 cofounded OPC technology in San Jose, CA. She holds five U.S. patents and is the co-author of *Oversampled A/D Converters* with S. Hein.

In 1997, Dr. Zakhor received the best paper award for her 1994 paper on oversampled A/D converters with S. Hein from IEEE Signal Processing Society and the best paper award for her 1996 paper on scalable video with D. Taubman from IEEE Circuits and Systems Society. She was a General Motors scholar from 1982 to 1983, received the Henry Ford Engineering Award and Caltech Prize in 1983, was a Hertz fellow from 1984 to 1988, received the Presidential Young Investigators (PYI) award, IBM junior faculty development award, and Analog Devices junior faculty development award in 1990, and Office of Naval Research (ONR) young investigator award in 1992. She is currently a member of the technical committee for image and multidimensional digital signal processing.