SNR Scalability Based on Bitplane Coding of Matching Pursuit Atoms at Low Bit Rates: Fine-Grained and Two-Layer

Jian-Liang Lin, Wen-Liang Hwang, and Soo-Chang Pei, *Fellow, IEEE*

***Abstract*—Because channel capacity varies depending on network traffic and the capacity of each reception, fine granularity scalability (FGS) of video coding has emerged as an important area in multimedia streaming applications. We propose an FGS video codec and a two-layer signal-to-noise ratio (SNR) scalable video codec, based on matching pursuits and bitplane coding. The temporal and spatial redundancy of atom positions between adjacent bitplanes are explored using the quadtree representation for a bitplane and the quadtree prediction algorithm. The efficiency of encoding atom positions is evaluated. Our FGS combines successive bitplane quantization of atom modula and quadtree prediction of atom positions. The performance of our FGS is compared with that of the discrete-cosine-transform-based FGS codec. The quality of a base layer or enhancement layer video in a two-layer scalable video codec can be adjusted without changing the bit rates. We propose using a combined frame obtained from the combination of the reconstructed base layer and enhancement layer images to estimate motion vectors. The performance of our two-layer codec is illustrated.**

***Index Terms*—Fine granularity scalability (FGS), matching pursuits (MPs), two-layer, streaming.**

I. Introduction

As THE development of multimedia applications grow, video techniques have been gradually changing from one-to-one (simulcast) to one-to-many (multicast) communications. Due to channel capacity variation and disparate requirements for different receivers, it is necessary to develop video coding and transmission techniques that are efficient and scalable to Internet heterogeneity. Although representing a video with multiple redundancy in different bit rates is a simple solution for multicasting in most commercial systems, this approach cannot efficiently cope with channel capacity variation [11], [8]. In contrast, video scalability is a better solution as it generates a single bitstream for all intended recipients, and each decoder can reconstruct a varied quality video within a specific bit rate range. Depending on the specification of receivers, a scalable system can support scalability either in frame rate (temporal scalability), frame resolution (spatial scalability), frame quality (SNR scalability) or a hybrid of these (hybrid scalability). Despite the fact that many scalable coding methods have been developed in recent years, they are still inefficient, especially at low bit rates [4]. Most of the existing systems use hybrid motion-compensated discrete cosine transform (DCT) for video coding. Although the hybrid motion-compensation algorithm may not be the best solution for video scalability, the hybrid scheme is simple, efficient, and has a small delay in performing frame prediction. In our study, we focus on developing SNR scalability algorithms at low bit rates in a hybrid motion-compensation video coding system in which the frame residuals are encoded using matching pursuits (MPs).

Vetterli and Kalker have translated motion compensation and DCT hybrid video coding into MPs [20]. They encode frames by the MP algorithm and a dictionary composed of motion blocks and DCT bases. Neff and Zakhor use MPs to represent the motion residual image [13]. According to their results, coding residuals using MPs attains a better performance than that of DCT in terms of PSNR and perceptual quality at low bit rates. MPs have less decoder complexity at low bit rates because a DCT decoder requires post processing to remove blocky and ringing artifacts to achieve reasonable quality at low bit rates, whereas the MP decoder achieves comparable quality without post processing [14]. The encoder of an MP-based video codec is extremely intensive in computation. Several methods have been proposed to reduce the computational load by representing the bases in a dictionary as a combination of simpler bases whose inner products can be obtained with less computational complexity [12], [22], [3]. It was shown in [15], [23] that the performance of a nonscalable MP codec can be improved by *in-the-loop* quantization where quantization noise is included in the MP algorithm. Certain MP SNR-scalable schemes have been proposed in [1], [21]. A fine granularity scalability (FGS) produces a continuous bitstream with increasing PSNR for a wide range of bit rates. An MP FGS coding algorithm is presented in [1] in which enhancement layer scalability is achieved by successively encoding groups of atoms in which the number of atoms in a group is the primary parameter controlling scalability. A better coding efficiency than FGS can be obtained at the expense of coarser scalability. A two-layer system is coarse scalable because the bitstream does not provide a continuous quality improvement over a wide range of bit rates. The lower layer delivers minimal bit rates while the upper layer delivers the highest possible quality. An

Manuscript received September 27, 2002; revised January 6, 2003. This work was supported in part by the National Science Council, R.O.C., under Contracts NSC 93-2213-E-001-009, NSC 91-2219-E-002-044, and NSC 93-2752-E-002-006-PAE. This paper was recommended by Associate Editor F. Pereira.

J.-L. Lin is with the Graduate Institute of Communication Engineering, National Taiwan University, Taipei 10617, Taiwan, R.O.C. and also with the Institute of Information Science, Academia Sinica,Taipei 115, Taiwan, R.O.C. (e-mail: jian@iis.sinica.edu.tw).

W.-L. Hwang is with the Institute of Information Science, Academia Sinica, NanKang 115, Taipei, Taiwan R.O.C. (e-mail: whwang@iis.sinica.edu.tw).

S.-C. Pei is with the Department of Electrical Engineering, National Taiwan University, Taipei 10617, Taiwan, R.O.C. (e-mail: pei@cc.ee.ntu.edu.tw).

Digital Object Identifier 10.1109/TCSVT.2004.839997

estimation theoretic approach to improve the performance of a two-layer system is proposed in [16] in which prediction was made from the previous base layer and enhancement layer.

Both our SNR FGS video codec and our two-layer SNR scalable video codec are based on successive bitplane quantization coding of the atoms selected from motion residuals using MPs. The proposed FGS algorithm uses bitplane coding and uses the spatial and temporal dependence between bitplanes to exploit the redundancy in the bitplanes. The efficiency of our algorithm in encoding atom positions lies in using quadtree to represent a bitplane and to perform bitplane prediction. Our experiments indicate that our algorithm can achieve a 1–2 bit reduction in encoding atom positions to that of the theoretical lower bound given in [1]. This bound is derived from the assumption that atom positions are uniformly and identically distributed random variables in each frame. The bound can be out-performed if the redundancy of atom positions is exploited. We then combine position coding and progressive refinement of atom modula in our MP FGS structure.

A two-layer scalable system is able to decide which layer to emphasize without changing bit rates. If available bandwidth is full most of the time, a bitstream is generated by using more information from the residuals of the high quality (enhancement) layer. Likewise, if it is limited, the residuals of the low quality (base) layer are emphasized. Using a linear combination of base layer and enhancement layer residuals in a two-layer scalable system has attracted much attention since it was published in [1]. Following this combination approach, we propose the use of a combined frame obtained from a linear combination of the reconstructed base layer and enhancement layer images to estimate motion vectors. Since both base and enhancement layer reconstructed images are used in our motion vector estimation, a better performance is attained in the enhancement layer.

In Section II, we review the MP algorithm and introduce the bitplane-based multidimensional algorithm for encoding atom positions in residuals. Because there is spatial and temporal overlap between bitplanes, the algorithm that removes their redundancy is described in Section II-B1. The final framework that combines successive bitplane encoding of atom modula and atom positions redundancy reduction is given in Section II-B2. In Section III, we propose our FGS algorithm and compare PSNR performance with that of an FGS DCT-based codec. Section IV presents a two-layer scalable MP codec and demonstrates its performance. Section V gives the conclusion.

II. Progressive Atom Coding

A. Image MP

We use MP algorithm proposed by Mallat and Zhang [10]. Let $\mathcal{D}$ be a dictionary of over-complete image bases $\{g_\gamma(\mathbf{x})\}$, where γ is the index. The algorithm decomposes an image into a linear expansion of the bases in the dictionary by a succession of greedy steps as follows.

The image $f(\mathbf{x})$ is first decomposed into

$$f(\mathbf{x}) = \langle f(\mathbf{x}), g_{\gamma_0}(\mathbf{x})\rangle g_{\gamma_0}(\mathbf{x}) + Rf(\mathbf{x}) \tag{1}$$

where $g_{\gamma_0}(\mathbf{x}) = \arg_{g_\gamma(\mathbf{x})\in D}\max\{|\langle f(\mathbf{x}), g_\gamma(\mathbf{x})\rangle|\}$ and $Rf(\mathbf{x})$ is the residual image after approximating $f(\mathbf{x})$ in the direction of $g_{\gamma_0}(\mathbf{x})$. The dictionary element $g_{\gamma_0}(\mathbf{x})$ together with the inner product value $< f(\mathbf{x}), g_{\gamma_0}(\mathbf{x}) >$ is called an atom. The MP algorithm then decomposes the residual image $Rf(\mathbf{x})$ by projecting it on a basis function of $\mathcal{D}$, as was done for $f(\mathbf{x})$. After M iterations, an approximation of the image $f(\mathbf{x})$ can be obtained from the M atoms by

$$\tilde{f}_M(\mathbf{x}) = \sum_{k=0}^{M-1} \langle R^k f(\mathbf{x}), g_{\gamma_k}(\mathbf{x})\rangle g_{\gamma_k}(\mathbf{x}) \tag{2}$$

and $\tilde{f}_M(\mathbf{x})$ converges strongly to $f(\mathbf{x})$ as $M \to \infty$.

B. Set Partitioning Coding of Atoms

When motion residual images are encoded using MP, the coding efficiency lies in economically encoding atoms. An atom includes an inner product value as well as a basis function containing location and index. Following, we will illustrate an algorithm which can efficiently encode atoms progressively.

Following the well-known set partitioning strategy adopted in the EZW [18] and SPIHT [17] algorithms, we apply the bitplane based successive-approximation quantization (SAQ) on the inner product values to encode the atoms selected from a motion residual using MPs. This algorithm will successively improve the resolution of a residual frame from many scans of the bitplanes to find new significant atoms and refine the values of existing significant atoms. Initially, all atoms are assumed insignificant. Let the N_{BP}th bitplane be the most significant bitplane of all the atoms. At the ith step, a threshold whose value is set to $2^{N_{\mathrm{BP}}-i}$ is compared to the inner products of insignificant atoms. An insignificant atom becomes significant when its absolute inner product value is larger than the current threshold. The positions and the index of the basis function and the sign of the inner product of the atom are encoded, and then the atom is added to the significant set (also called *AtomList*). The atom will remain in the significant set and the atom's inner product value will be successively refined by the following refinement steps.

When a MP is used together with a bitplane based SAQ, the positions of atoms must be carefully handled to ensure coding efficiency. The energy of a transformed-based coding method is usually concentrated in some bases. This occurs in low frequency bands for DCT, and in coarser scales for a wavelet transform. There are efficient bitplane scanning orders for both DCT and wavelet transform. The left subfigure of Fig. 1 shows one DCT method using a zig-zag order. The middle subfigure shows one wavelet transform method using a tree order. Unlike the transformed-based coding methods, the atoms of a MP can be randomly positioned in a residual frame, as shown in the rightmost subfigure. The energies of atoms are scattered over the residual frame according to the contents in the frame. Thus, neither DCT zig-zag ordering nor wavelet tree ordering can encode the atom positions efficiently. In consequence, we should develop an efficient scanning order for the atom positions in a bitplane to attain better coding efficiency.

1) Quadtree and Quadtree Prediction of Atom Position: Position coding efficiency is dependent on the number of atoms in a bitplane. To specify this dependency, a theoretical lower bound for atom positions is proposed in [1] as a comparison reference for atom position encoding algorithms. The

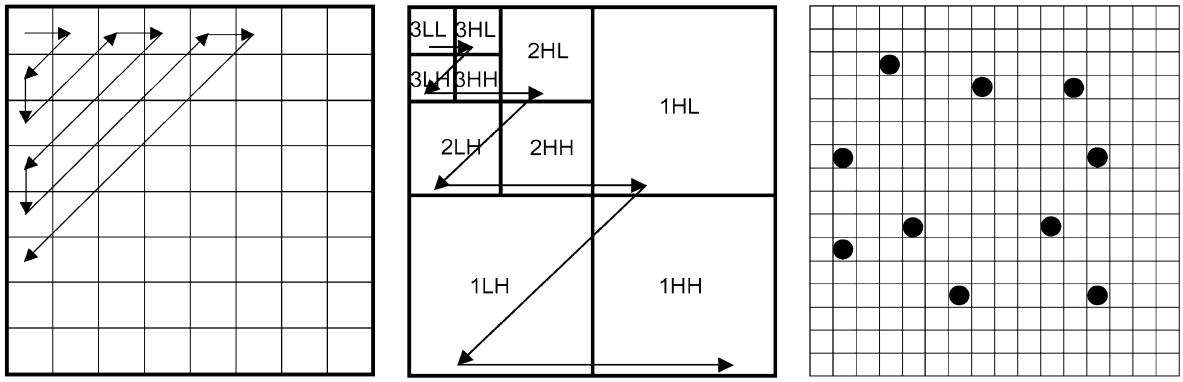

Fig. 1. Left: DCT scan-order. Middle: EZW scan-order. Right: atom positions.

lower bound is derived by assuming that atoms are uniformly and independently distributed on an $N_1 \times N_2$ image and that no pixel of the image has more than one atom. The latter assumption is valid at low bit rates since only a few atoms are selected in a residual, and the probability that more than one atom exists at each pixel location is low. Our simulation on sequences Akiyo and Container at low bit rates shows that the probability that more than one atom are selected at the same location is less than 1%. If there are n atoms on the image, the entropy for encoding the positions of an atom will be

$$\log_2 \binom{N_1 \times N_2}{n} / n.$$

Note that atoms usually distribute nonuniformly in residuals. This bound can be out-performed if an atom position encoding algorithm takes advantage of nonuniformity in atom distribution and removes the redundancy among them.

In [1], the *NumberSplit* algorithm is presented to encode atom positions built on a multidimensional searching algorithm. In this algorithm, the total number of atoms of a residual frame is decided and given to a decoder. A frame is then separated into two halves, and the number of atoms in the first half is given to the decoder. The number is entropy-coded by an adaptive Huffman table which is built according to the number of atoms to be coded at each frame. The decoder can use the total number of atoms and the number of atoms in the first half to obtain the number of atoms in the other half. Each half is further divided recursively until it reaches either one pixel or a region that contains no atom. In [1], atoms tend to cluster around regions of high residual error. By taking advantage of nonuniform atom clustering in a residual, the *NumberSplit* method spends an average of 0.25 b/atom position less than the theoretical lower bound. Nevertheless, the *NumberSplit* algorithm does not explore temporal dependencies between residual frames, and encoding the number of atoms yields a relatively complex entropy coder which requires more computation to achieve efficiency. In contrast to this algorithm, we propose an algorithm based on a quadtree representation of a bitplane. This algorithm predicts the quadtree for the adjacent bitplane using the quadtree for the current bitplane to remove spatial and temporal redundancy. Simulations show that the efficiency of our approach in encoding atom positions is improved to 1–2 b below that of the theoretical low bound for uniform independent atom distribution.

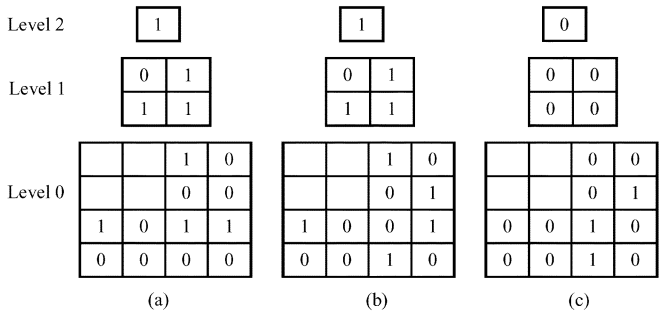

Fig. 2. Quadtree and quadtree prediction. (a) Quadtree to be predicted. (b) Quadtree from which to predict. (c) EXCLUSIVE-OR of (a) and (b). Note that one can obtain (a) by (b) EXCLUSIVE-OR (c).

We will explain how we implement quadtree, then give the details of simulation results. Quadtree is a simple image decomposition algorithm and has been used successfully in representing binary images at different resolution levels [19]. A bitplane can be represented by a quadtree. If the entire bitplane has at least one atom, we label the root node "1." Four children, representing four quadrants of the bitplane, are added to the root node. If the bitplane has no atom, we label it "0." This process can be applied recursively to each of the four children until each child node represents a single pixel. Thus, if the image size is $2^{l_{\max}} \times 2^{l_{\max}}$, the quadtree has at most $l_{\max} + 1$ levels. Quadtree-based multiresolution representation is resistant to small variations of bit patterns between bitplanes. An example is given in Fig. 2 where the bit patterns at level 0 (the finest resolution of the quadtree) are different in Fig. 2(a) and (b). However, at upper levels of quadtrees of Fig. 2(a) and (b) (which correspond to coarser resolutions of the quadtrees) the same patterns occur. In other words, the small variations of 0 and 1 between the bitplanes do not propagate to upper levels. Hence, if two bitplanes have a lot of bit pattern overlap, the quadtree of one bitplane can be used to efficiently predict the quadtree of the other bitplane.

In video encoding at low bit rates, corresponding bitplanes in two consecutive frames and adjacent bitplanes within a frame tend to have many redundant structures. The temporal and spatial dependences of these bitplanes are exploited in the following recursive *Quadtree_Prediction* algorithm. The bitplane b of the current frame is first represented as a quadtree Q_b^t. The quadtree is then predicted either from the quadtree of the corresponding bitplane in the previous frame or from the

quadtree of the union of all the previous bitplanes at the same frame. We use $Q_b^t(k,i,j)$ to denote the node at the (i,j)th position in level k of the quadtree corresponding to the bth bitplane in the tth residual frame. For any P-frame in a GOP, starting from the most significant bitplane (which is the $N_{\rm BP}$th bitplane) toward the least significant bitplane, our *encoder* enters this algorithm from the root node of Q_b^t. The nodes in the tree are then traversed from the root in the depth-first order. Note that other multidimensional tree search orderings can be used to traverse a quadtree [9]. We give our encoder prediction algorithm as follows. We use notation $Q_b^{'t}$ to denote the differential quadtree which was obtained from predicting Q_b^t from that of its previous bitplanes. Note that our decoder uses the same algorithm described below, except that Q_b^t and $Q_b^{'t}$ in the algorithm are switched.

Quadtree_Prediction(Q_b^t,k,i,j)
{
IF Q_b^{t-1} is used to predict Q_b^t
1: Output the bit $(Q_b^{'t}(k,i,j)=Q_b^t(k,i,j)\oplus Q_b^{t-1}(k,i,j))$;
OTHERWISE
2: Output the bit $(Q_b^{'t}(k,i,j) = Q_b^t(k,i,j)\oplus \bigcup_{p=b+1}^{N_{\rm BP}} Q_p^t(k,i,j))$;
IF $Q_b^t(k,i,j)=0$
The node is a leaf node;
ELSE IF ($Q_b^t(k,i,j)=1$ and $k=0$)
3: The node is a leaf node and associates with one or more atoms;
4: The index of the basis and sign of the inner product of each atom and a symbol indicating the last atom at the position are entropy encoded;

ELSE encode its four children
5: *Quadtree_Prediction*$(Q_b^t,k-1,2i,2j)$;
6: *Quadtree_Prediction*$(Q_b^t,k-1,2i,2j+1)$;
7: *Quadtree_Prediction*$(Q_b^t,k-1,2i+1,2j)$;
8: *Quadtree_Prediction*$(Q_b^t,k-1,2i+1,2j+1)$;
}

Step 1 in the above algorithm uses temporal prediction of the current bitplane. Differences in quadtrees are obtained from using EXCLUSIVE-OR $(\oplus)$ on corresponding nodes in Q_b^t and Q_b^{t-1}. Step 2 uses spatial prediction in which the quadtree corresponding to the bitplane obtained from the union of the bitplanes from $b+1$ to $N_{\rm BP}$ in the current frame is used to predict, again by EXCLUSIVE-OR, the quadtree of the current bitplane. In Step 3 and 4, a "1" at a terminal node indicates new significant atoms whose basis indexes and signs of their inner products are then entropy-encoded. Since more than one atom can become significant at a location of a bitplane, we introduce one symbol *last*, with a value of either 1 or 0, which indicates whether an atom is the last atom at a given position of a bitplane. In our implementation, each atom is associated with a triplet (*index*, *sign*, *last*). If two atoms becomes significant at the same location of a bitplane, then the *last* in the first atom is 0 and the *last* of the second atom is 1. Two atoms may become significant at the same location but in different bitplanes. In this case, the *last* of both atoms is 1.

If a node is not a leaf, then its four children are visited in a depth-first search order. Fig. 2 is a simple example illustrating the quadtree for the current bitplane (to be predicted) and the previous bitplane (from which to predict). The differential quadtree is the result of applying EXCLUSIVE-OR on the corresponding nodes in (a) and (b). The blanks in the top left corner of all subfigures indicate that nodes located there are not encoded since their parent nodes have value 0. One can recover from using EXCLUSIVE-OR on Fig. 2(b) and (c). The differential quadtree in Fig. 2(c) is traversed in depth-first order and yields the sequence 0 0 0 0000 0 0001 0 1100. Since there are many zeroes, the entropy of the differential quadtree is relatively low. Moreover, the symbols used in the sequence are only zeroes and ones, thus they can be encoded efficiently via the adaptive arithmetic code. When applying our algorithm to encode atoms, the index of the basis function, the sign of inner product value, and the last symbol of each atom are each entropy-encoded by adaptive arithmetic codes.

The performance of the proposed algorithm depends on the correlation of atoms between bitplanes. A node at a higher level of a quadtree represents a larger block in the bitplane. It is likely that corresponding higher level nodes have the same value. This is particularly ture when coding slow motion sequences in which larger motion residual errors most likely occur at the boundaries of moving objects. These errors tend to be located in almost the same region of two adjacent frames. As for coding efficiency of our algorithm in Fig. 3 we demonstrate and compare the average bits used to encode each atom position over Sean, Akiyo and Container sequences using 10 frames/s in a QCIF format and a 10-s testing time. The first three bitplanes for each frame in Sean and Akiyo are encoded while the first four bitplanes are encoded in each frame of the Container sequence. These bitplanes are encoded using either temporal bitplane prediction, spatial bitplane prediction, or no prediction. The X-axis in the figure is the average number of atoms for each bitplane and the Y-axis is the average number of bits that encodes each atom position for each bitplane. From left to right, the first mark in a curve of either quadtree or quadtree prediction corresponds to the first significant bitplane, the second mark to the second significant bitplane, and so on. Squares (X,Y) in the *NumberSplit* curve indicate that an average of Y bits is used to encode one atom position. All atoms are partitioned into groups of X atoms. The clustering parameter f in *NumberSplit* is set to 0.5.

The coding efficiency of temporal bitplane prediction is evidently superior to all the others including that derived from theoretical lower bound. This bound, obtained by assuming that atoms are uniformly and identically distributed, can be out-performed if an algorithm can effectively take advantage of nonuniformity in atom distributions. The performances of quadtree without prediction and the *NumberSplit* algorithm have similar results. Quadtree results using spatial bitplane prediction are slightly better than those without.

2) Progressive Atom Coding Algorithm: This section provides the final framework that combines the set partitioning scheme for atom modula and atom positions encoding. We alternatively apply the following two phases on each bitplane in a residual frame after initially setting the required parameter which would be the most significant bitplane in a video. In the

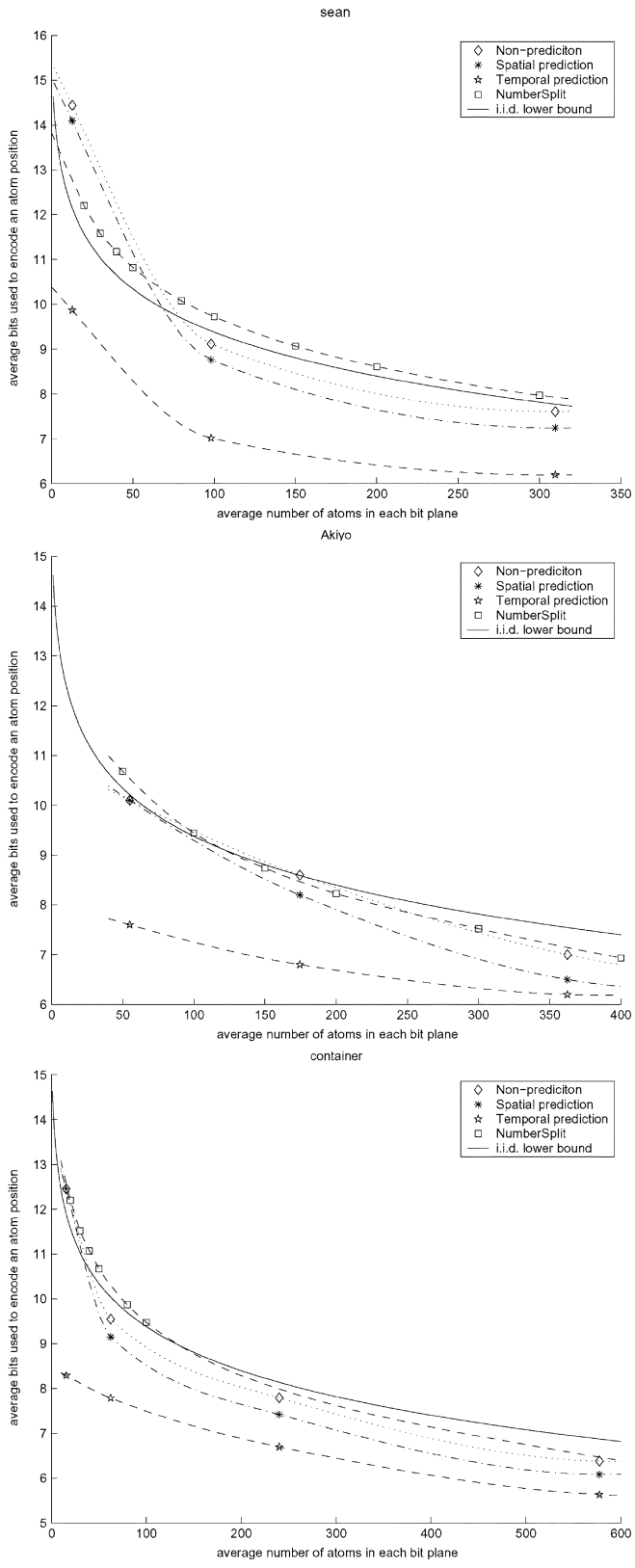

Fig. 3. Comparison of atom position coeffiency of quadtree, quadtree prediction and *NumberSplit* sequences from top to bottom are Sean, Akiyo, and Container, respectively.

refinement phase, atom modula in the *AtomList* are refined by adding one extra bit of information. In the sorting phase, new significant atoms are found in the current bitplane and are then appended to the *AtomList*. The following gives the algorithmic description of the framework.

Step 1) **Initialization**: Set the *AtomList* as empty. Let n be the most significant bitplane for motion residual frames of our video.

Step 2) **Refinement Phase**: Produce a bitstream corresponding to one extra bit from the modulus of each atom in the *AtomList*.

Step 3) **Sorting Phase**: Use the *Quadtree_Prediction* algorithm to generate a set of new atoms whose absolute values of inner products are within $[2^n, 2^{n+1})$ and then include the new atoms to the end of the *AtomList*.

Step 3.1) If the bitplane-shift parameter $b > 0$, then encode the values in bitplanes $n-1, \ldots, n-b$ of the new significant atoms.

Step 4) **Next bitplane**: Decrease n by 1 and go to Step 2.

Note that the above algorithm is not the same as that proposed in [18], [17]. Compared to theirs, our algorithm allows more than one bitplane atom modulus to be encoded at the sorting pass for each new significant atom. Parameter b in Step 3.1, which gives the number of extra bitplanes from the current bitplane, is used in encoding the modulus of a new significant atom. The position of each atom is encoded only in the pass in which the atom becomes significant and then the atom's modulus is refined by successive passes through the refinement phase. Encoding the position of a new significant atom requires more bits than in refining the atom's modulus. The purpose of encoding more than one bitplane for a new significant atom is to increase the distortion-reduction δD of atom modulus to compensate for the relative large rate R in encoding the new significant atom.

Let the largest atom modulus be normalized to 1 and let m atoms be newly significant at the kth significant bitplane. The total bits spent is at most $R_m + mb$, where R_m is the bits for the atoms and mb is the bits for coding extra modula of the atoms. In MPs, $(R_m)/(m) \gg b$ is satisfied for new significant atoms in a bitplane and reasonable b. Using b extra bitplanes when coding a new significant atom reduces a fraction of at most 2^b distortion over that without using an extra bitplane (with approximately the same number of bits). This yields an increase of PSNR for encoding new significant atoms. An atom newly significant at the kth significant bitplane has a normalized modulus error of at most 2^{-k} and, with bitplane-shift parameter b, the distortion becomes $2^{-(k+b)}$. Encoding the modulus of a new significant atom with more than one bitplane is a feature in our low bit rate video coding. The efficacy of this feature in FGS at low bit rates is illustrated in the next section. Fig. 4 gives the order in which bitplanes are included in the sorting and refinement phases of our algorithm. The process of sorting and refining bitplanes of the left subfigure is $b = 0$ which is the same as that in [18], [17]. In the right subfigure, three bitplanes of atom modula are encoded at the sorting phase, while one bitplane is encoded at the enhancement phase.

III. FGS Matching Pursuit Video Codec

A video streaming encoder compresses and stores video streams and simultaneously transmits them on demand through

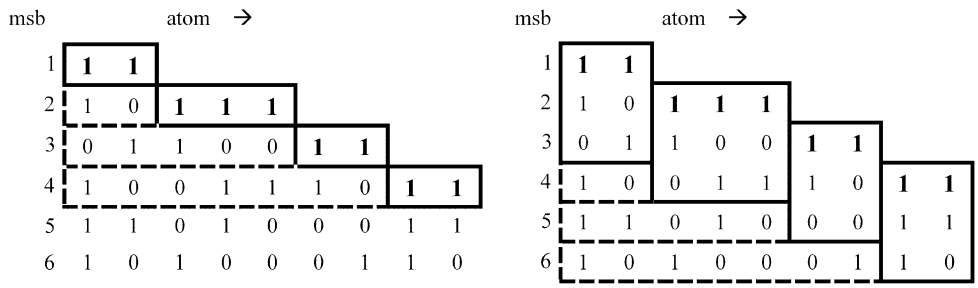

Fig. 4. Modulus sorting and refinement comparison using different b after four passes. Left: $b = 0$. Right: $b = 2$. bitplane values included in sorting and in refinement phases are, respectively, indicated by solid boxes and dashed boxes.

a scalability-aware transport mechanism to a large amount of users [2]. There are many video scalable encoder applications. Among them, the MPEG-4 video approach, which is known as FGS, has attracted the most attention as it has achieved a fine balance between coding efficiency and coding complexity for producing scalable bitstreams. A basic FGS framework requires two layers: the base layer and the enhancement layer. The base layer includes a motion prediction and has an encoder with highly efficient coding in low bit rates. Any decoder must be able to decode the base-layer bitstream. In principle, any FGS method can be used to produce streaming bitstreams for the enhancement layer. The bitplane-based DCT FGS encoding method is still the most widely used.

A. Proposed FGS

Our proposed two-layer FGS MP video codec is shown in Fig. 5. Our base layer encoder, shown at the top subfigure, performs motion compensation and encodes motion residual frames using a MP. Although an MP-based video encoder is more complex, its decoder is comparably less complex than other methods. Both the base layer and the enhancement layer of our MP FGS coders use the progressive atom encoding algorithm proposed in Section II-B2 in which the atom position is encoded by the *Quadtree_Prediction* algorithm. The position encoding algorithm must store both a quadtree for each base layer bitplane in the previous frame (in order to perform temporal prediction), and a quadtree for the union of previous bitplanes at the current frame (to perform spatial prediction). Representing the quadtree of an N pixels bitplane takes at most $(4N/3)$ bits. If we operate at a bit rate with two bitplanes as base layer, we would need at most $4N$ bits for storage. Although slightly more storage is required, the entropy-coder is a two symbol adaptive arithmetic code which is easily implemented at a decoder site. The bottom subfigure of Fig. 5 illustrates an example of our bitplane-based FGS to encode atom position. Temporal prediction is carried out only in base layer bitplanes while spatial prediction is carried out in enhancement layer bitplanes. A bitplane in the enhancement layer is predicted from all the previous bitplanes from the same frame. The spatial or temporal bitplane predictions are based on operations on quadtrees and the details of these operations are given in Section II-B1. The PSNR can be *lifted* by finely quantizing atom modula of new significant atoms by setting parameter b, see Section II-B2. Fig. 6 shows experimental results of PSNR performance at low bit rates with different b.

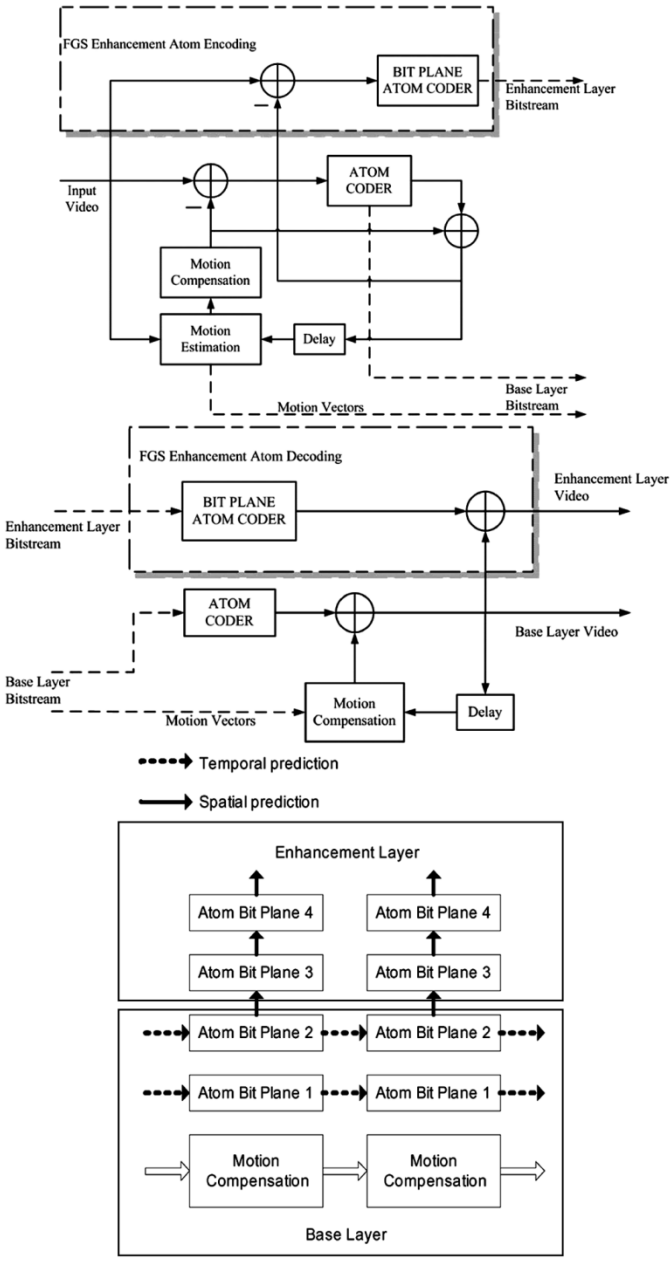

Fig. 5. Our FGS MP codec. Top: encoder. Middle: decoder. Bottom: atom positions encoded by bitplane-based predictions.

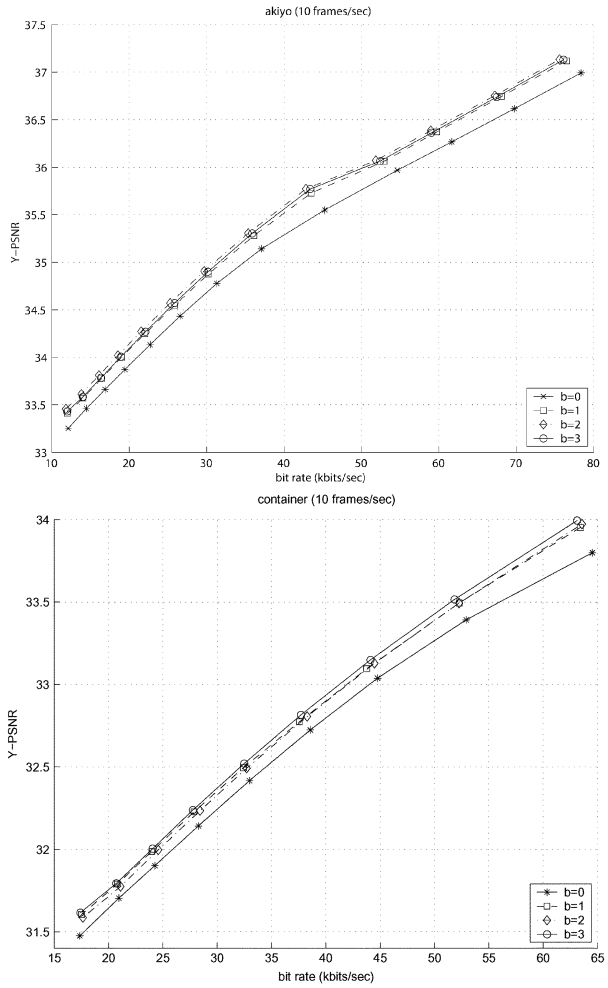

Fig. 6. PSNR at low bit rates by setting different b. Top: Akiyo. Bottom: Container. There is an average 0.2-dB improvement if b is set to either 2 or 3.

B. Performance Evaluations and Comparisons

The performance of our MP FGS at various bit rates in terms of luminance PSNR (Y-PSNR) are compared with those of DCT FGS. The motion vectors of both codecs are obtained from standard H.263 [5]. Our FGS DCT-based codec follows Amendment 2 of the MPEG-4 FGS DCT-based video codec in encoding the enhancement layer of a residual frame [7]. In all the following experiments, we divide a frame into blocks and select an atom from only one block at each iteration [13]. We use the weighted energy block search algorithm, with weights provided in [1], to find atoms. The first two parts of this section compare performances of our bitplane-based MP FGS to bitplane-based DCT FGS at different low bit rates. The third part provides the evaluation of our FGS to a bitplane-based nonscalable MP.

1) Comparison of MPs and DCT FGS Using the Same Residual Image: For a fair comparison of coding efficiency of residual images between bitplane-based MP FGS and DCT FGS, different codecs should encode the same residual images using the same number of bits. We set up the experiments so that the base layer of the current frame is obtained from the base layer of the previous frame using only motion compensation. In other words, a frame is estimated from the previous base layer using motion compensation and the residual is the enhancement layer of the frame. Accordingly, the differences in encoding residual images by the codecs will not be involved in predicting the following frames, and the codecs will encode the same residual images.

Fig. 7 shows Y-PSNR comparisons of our MP FGS codec with that of the DCT FGS codec using various sequences in QCIF at 10 frames/s. Because both codecs encode the same residuals in the enhancement layer, as indicated in the Fig. 7, the performance of bitplane-based MP FGS is better than that of DCT FGS in encoding residuals at low bit rates. The slope corresponding to the MP bitplane encoder is higher than that of the DCT bitplane at bit rates close to the base layer bit rate in each sequence, and it yields that the curve of MP is above that of DCT in each sequence. This may be because MP first picked up a few locally energy-concentrated patterns, causing a high reduction in distortions, so that the MP slope is initially higher than that of the DCT slope. The slopes of the two curves become approximately the same as the bit rate increases. For the Akiyo sequence, at 100 kb/s, the largest PSNR gain of the MP bitplane encoder over the DCT bitplane encoder is about 0.8 dB.

2) Comparisons at the Base Layer and the Enhancement Layer: We evaluated the coding efficiency of both layers in MP FGS and DCT FGS by comparing their Y-PSNR at various low bit rates. Both codecs have the same intraframe (I-frame) encoded by the DCT method. The other frames are all inter-frame (P-frame). Unrestricted motion vector mode and advanced prediction mode in standard H.263 are used in both motion vector codecs. For all comparisons, the base layer of the FGS MP-based codec includes either one or two most significant bitplanes. The base layer bit rates of the FGS DCT-based codec is determined by using a fixed quantization step (QP) for all frames in the sequence. This assures that the base layer's bit rates of the two codecs are as close as possible. The frame rate of all sequences is 10 frames/s and the format for all sequences is QCIF.

The average Y-PSNR versus the bit rates is plotted in Fig. 8. Base layer performance corresponds to the beginning points whose X-axes are at the lowest bit rates in the curves. In all experiments, the Y-PSNR of our MP-based codec is better than that of the DCT-based codec, both in the base layer and enhancement layer. The average base layer improvement is about 0.7 dB. Also, the Y-PSNR of the MP FGS increases faster than that of the DCT FGS as bit rates increase.

3) Evaluating Bitplane Representation and Prediction of FGS: Here we use different bitplane numbers as our base layer and compare performances with a bitplane-based nonscalable codec at low bit rates. Note that our bitplane-based nonscalable codec does not optimize at a particular bit rate and it is not the best nonscalable MP codec. Nevertheless, it provides a reference we can use to evaluate important aspects of our FGS.

Fig. 9 gives the PSNR performance versus bit rates of the Akiyo and Container sequences. The curve corresponding to the top envelope of each subfigure is the performance of our nonscalable codec in which all the bitplanes are temporally predicted. The previous reconstructed frame from all the bits is used to perform motion compensation for the current frame. The rest of the curves, from the bottom to the top, correspond to perfor-

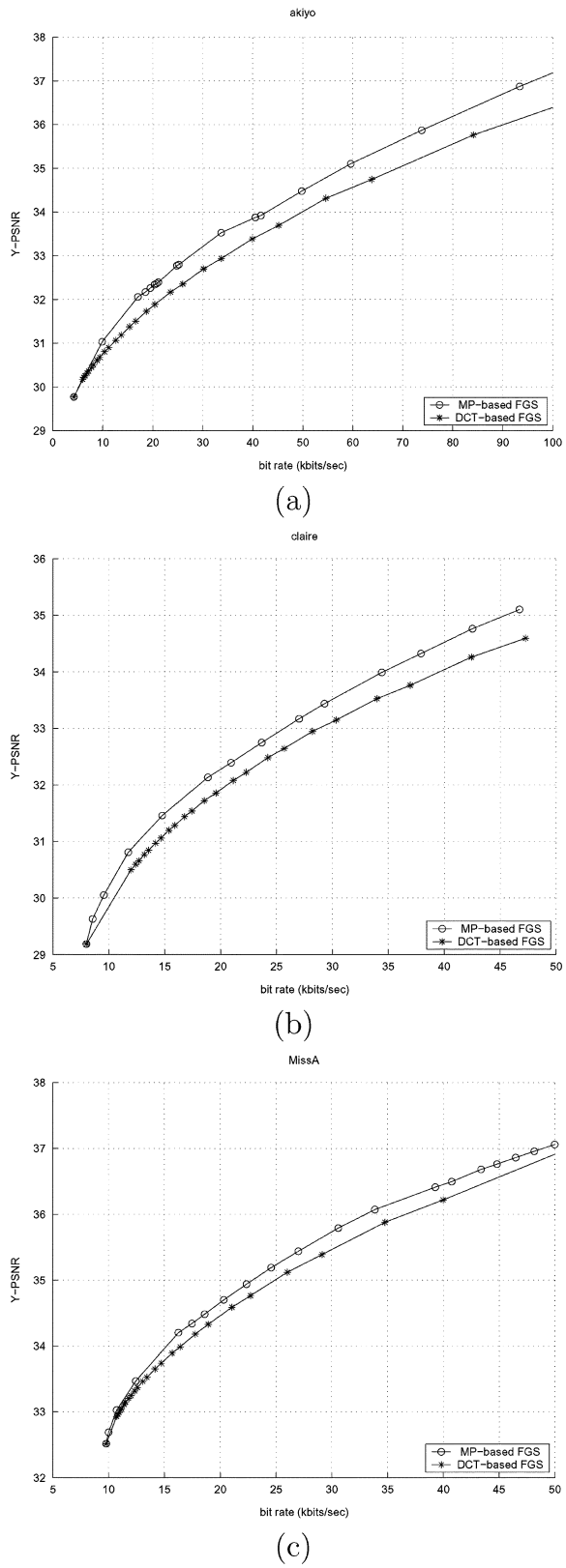

Fig. 7. Comparison of the same residual images of bitplane-based MP and that of DCT methods for (a) 10-s Akiyo, (b) 5-s Claire, and (c) 3-s Miss America sequences. The frame rate is 10 frame/s and bitplane-shift parameter $b = 3$.

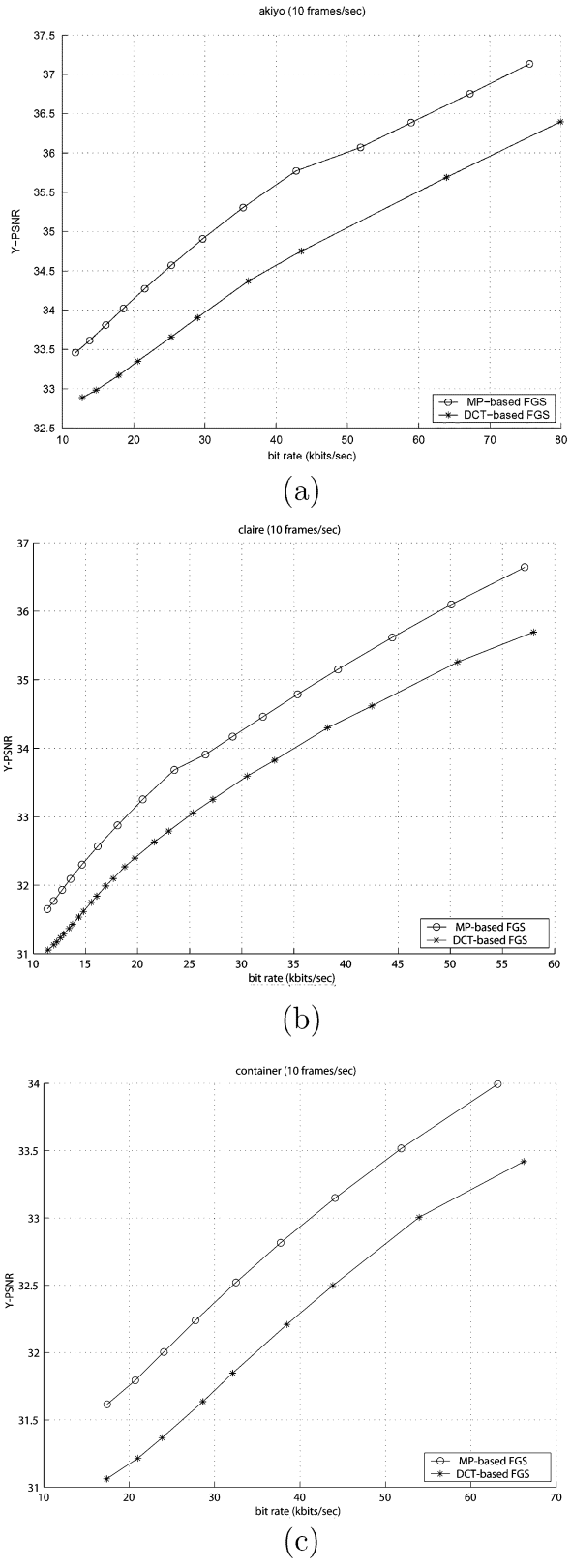

Fig. 8. Comparison of MP-based FGS with DCT-based FGS for (a) Akiyo, (b) Claire, and (c) Container. All sequences are 10 frames/s, in QCIF, for 3.3 s. The bitplane-shift parameter $b = 3$.

mance using one bitplane, two bitplanes and three bitplanes as our base layer. The efficacy of using bitplane prediction and motion compensation manifests at relatively lower bit rates when the PSNR gap between consecutive curves is comparably large. The curve using one bitplane as base layer shows the greatest PSNR increase when bit rates are close to the base layer. This is

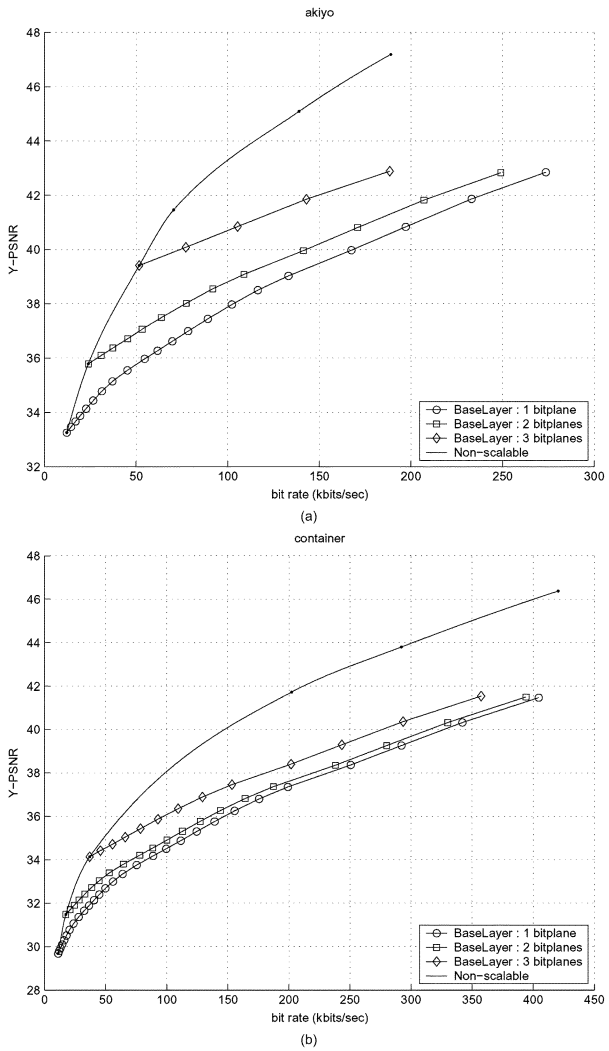

Fig. 9. Comparison of our FGS with various bitplane numbers as base layers to a bitplaned-based nonscalable codec. Sequences (a) Akiyo and (b) Container are 10 frames/s in QCIF for 3.3 s.

because MP selected a few atoms with energy concentrated in the first two significant bitplanes and, therefore, caused a higher slope increase at lower bit rates. The gap between curves continues to decrease until higher bit rates are reached at which point the curves are almost parallel.

IV. Two-Layer SNR Scalable System

Enhancement layer information of the proposed FGS MP codec is not used in predicting the following frame. However, enhancement layer PSNR performance can be further improved at the expense of coarser scalability if the enhancement layer is used to predict the next frame. This can occur in a two-layer system where two residual images are obtained and a single motion vector set is used to compensate both layers [1], [21]. The two-layer scalable approach allows an encoder to control the relative quality of layers by adjusting atom allocations between the base and enhancement layers while retaining a fixed total bit rate. If the available bandwidth is full most of the time, then it is worth operating on the best quality in the enhancement layer at the expense at a lower quality in the base

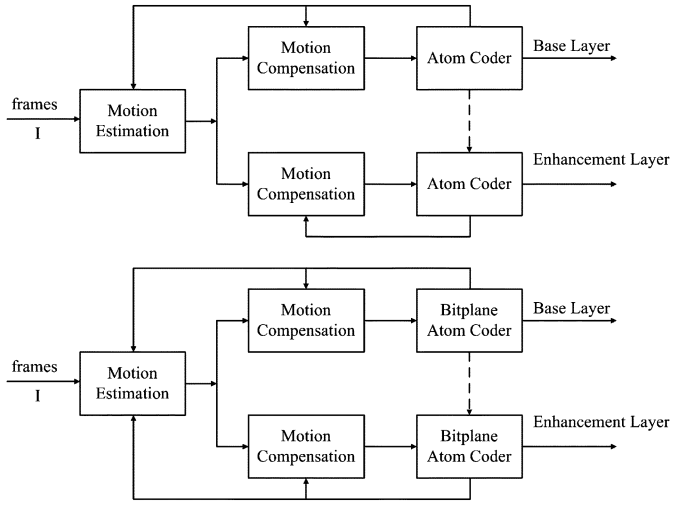

Fig. 10. Two-layer SNR scalable codec. Top: from [1]. Bottom: ours.

layer. On the other hand, if the channel bandwidth is low, more atoms are allocated to maintain the highest quality in the base layer while the enhancement layer operates of a lower quality.

In [1], motion vectors are computed from the base layer. Base layer atoms are chosen from the frame that is obtained by a proper linear combination of base and enhancement residuals, while the enhancement layer atoms are found only from the enhancement residual. In [21], motion vectors are chosen from the layer we want to emphasize. Let us suppose that the motion vectors are computed from the base layer and that enhancement layer atoms are chosen either from searching among the atoms defining the base layer residual, or from searching new atoms in the enhancement layer residual that are not defined in the base layer residual. The encoder decides which atom to choose by selecting the maximum rate/distortion value. The enhancement layer performance of a multilayer scalable codec can be improved by an estimation theoretic approach. This is done by using all the available information a layer has in order to predict the enhancement layer. An approach is proposed in [16] in which the information of the base-layer value, base layer quantizer, the previous enhancement layer value and enhancement layer quantizer are all included in estimating the current enhancement layer value.

In this section, we propose a two-layer scalable MP algorithm. The our schematic diagram and that in [1] are given in Fig. 10. The main difference between them is motion compensation. Our algorithm uses previously reconstructed images from both layers to perform motion estimation but [1] does so using reconstructed image from *only* the base layer.

A. Motion Estimation From Two Reconstructed Images

In [1], MP is applied to the residual image obtained from a linear combination of base layer and enhancement layer residuals. According to the derivation in [1, eq. (5)–(7)], this process is equivalent to finding the atom to minimize the objective

$$J(g,t) = \min_{\langle t,g\rangle} \alpha\|R_b - tg\|^2 + (1-\alpha)\|R_e - tg\|^2 \quad (3)$$

where R_b and R_e are, respectively, the base layer and enhancement layer residual. The solutions of the minimization process

$$g_{\max} = \max_g |\langle \alpha R_b + (1-\alpha)R_e, g\rangle|$$
$$t = \langle \alpha R_b + (1-\alpha)R_e, g_{\max}\rangle$$

indicate that the atom that minimizes the objective in (3) is found by applying MP on the combined residual $\alpha R_b + (1-\alpha)R_e$. Note that R_b and R_e are motion vector dependent and we use $R_b(\vec{v})$ and $R_e(\vec{v})$ to manifest the dependence. The approach in [1] focuses on the selection of atoms from the given R_b, R_e, and α, but it does not calculate the best motion vector $\vec{v}^*$ that minimizes the residual $\alpha R_b(\vec{v}) + (1-\alpha)R_e(\vec{v})$ for a given α. Our two-layer approach, however, does do so. Let $\hat{I}(\vec{v})$ be the motion prediction of the current frame I with the vector $\vec{v}$ from $\alpha\tilde{I}_b^p + (1-\alpha)\tilde{I}_e^p$, where $\tilde{I}_b^p$ and $\tilde{I}_e^p$ are, respectively, previously reconstructed frames from the base layer and the enhancement layer. Then

$$\begin{aligned}\hat{I}(\vec{v}) &= \Gamma\left(\alpha\tilde{I}_b^p + (1-\alpha)\tilde{I}_e^p, \vec{v}\right)\\ &= \alpha\Gamma\left(\tilde{I}_b^p, \vec{v}\right) + (1-\alpha)\Gamma\left(\tilde{I}_e^p, \vec{v}\right)\\ &= \alpha\hat{I}_b(\vec{v}) + (1-\alpha)\hat{I}_e(\vec{v}) \qquad (4)\end{aligned}$$

where Γ performs motion prediction, $\hat{I}_b(\vec{v})$ and $\hat{I}_e(\vec{v})$ are, respectively, predicted base layer and enhancement layer frames using motion compensation. Using (4), the motion vector $\vec{v}^*$ giving with the least amount of prediction error for the current frame I is

$$\begin{aligned}&\arg\min_{\vec{v}} \|I - \hat{I}(\vec{v})\|\\ &= \arg\min_{\vec{v}} \|\alpha(I - \hat{I}_b(\vec{v})) + (1-\alpha)(I - \hat{I}_e(\vec{v}))\|\\ &= \arg\min_{\vec{v}} \|\alpha R_b(\vec{v}) + (1-\alpha)R_e(\vec{v})\|. \qquad (5)\end{aligned}$$

Our two-layer approach finds the best motion vector $\vec{v}^*$ according to the criterion in (5) and obtains base layer residual $R_b = R_b(\vec{v}^*)$ and enhancement layer residuals $R_e = R_e(\vec{v}^*)$. The criterion in (3) is then used on the residuals to find atoms.

Let us compare the efficacy of our approach to that in [1], where motion vectors used to compensate the base layer and enhancement layer are estimated purely from the previous reconstructed image of base layer $\tilde{I}_b^p$. When α is small, the base layer has the worst performance. Thus, the approach in [1] effects motion vector estimation and causes the performance of the enhancement layer to worsen. In our approach, we also use one set of motion vectors to compensate for both layers. However, the set of motion vectors is estimated from a combination of the previous base layer reconstructed images with weight α and the previous reconstructed image in the enhancement layer with weight $1-\alpha$. Motion vectors are estimated mainly from the previous reconstructed frame of the enhancement layer when α is small.

B. bitplane-Based Residuals Encoding

Similar to that proposed in [1], the atoms in our base layer are chosen from a combined residual image by

$$\alpha R_b + (1-\alpha)R_e \qquad (6)$$

where α is an adjustable parameter indicating the percentage of R_b and R_e in the combined base layer residual. Then, the base layer residual is represented using the n most significant bitplanes from the MP algorithm which takes atoms from $\alpha R_b + (1-\alpha)R_e$. Our enhancement layer residual is represented from the selected bases in encoding the combined base layer residual, and also from bases in the enhancement layer dictionary. In coding the enhancement layer residual R_e, the bitplane atom coder not only refines the inner products of the atoms of the base layer to represent the enhancement layer residual, but also encodes new atoms for it. The refined inner product values and the new atoms in enhancement layer residual are then encoded by our bitplane based successive quantization algorithm, as presented in Section II-B2.

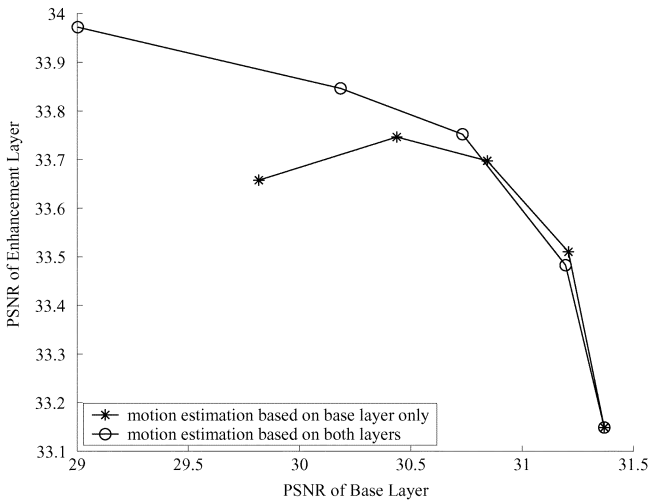

Fig. 11. PSNR of base layer and enhancement layer with α varying from 0 to 1 with a step of 0.25 where the point on the right of the horizontal axis in each curve is $\alpha = 1$, and the point on the left of the horizontal axis is $\alpha = 0$. The base layer bit rate is 17 kb/s and the total bit rate is 40 kb/s.

C. Comparisons

We compared the Y-PSNR performance of our two-layer MP system to that obtained from a modification of the implementation in [1]. Both methods use bitplane coding of the residuals while their only difference is in motion vector estimation. The bitplane-shift parameter b in all comparisons hereafter is set to 0 to simplify our implementation. The testing sequence is Sean encoded at 10 frames/s in QCIF format and the testing time is 10 s. The first frame of both codecs is I-frame encoded by DCT, while the other 99 frames are all P-frames. We used three bitplanes for the base layer and one bitplane for the enhancement layer in encoding all the P-frames. In Fig. 11, the average Y-PSNRs attained from different methods are shown with α varying from 0 to 1 with a step of 0.25. As indicated in the figure, the proposed two-layer method has a better enhancement layer performance. The Y-PSNR gain in the enhancement layer is obvious when α is small. Furthermore, the enhancement layer performance of our two-layer method can be easily adjusted by changing the value of α since our enhancement layer performance decreases monotonically as α increases. In contrast, the enhancement layer performance in [1] initially increases but eventually begins to decrease as α increases.

We also compared the performance of our two-layer codec with that of the DCT-based H.263+ coder [6]. There are struc-

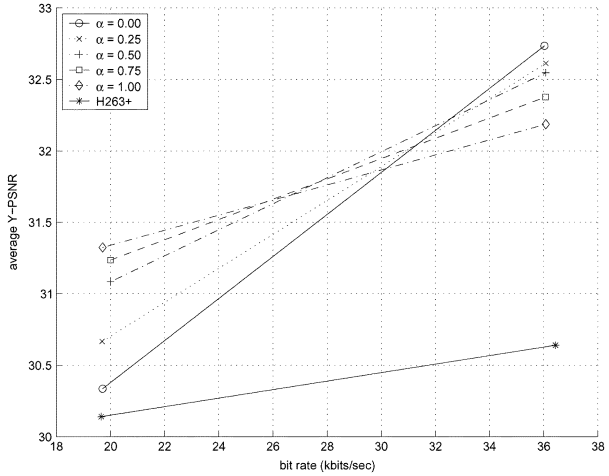

Fig. 12. PSNR of base layer and enhancement layer for our two-layer and H.263+. The point on the right of the horizontal axis in each curve is the enhancement layer while the point on the left is the base layer. The PSNRs of the base layer and the enhancement layer are the averages of all P-frames.

tural differences between H.263+ and ours. In H.263+, the enhancement layer is bi-directionally predicted by using both the previous image from the enhancement layer and the current base layer image. The motion compensations of the base layer and the enhancement layer use different sets of motion vectors. The experimental H.263+ codec is TMN3.2, implemented by the University of British Columbia. Fig. 12 illustrates the Y-PSNRs of the base layer and the enhancement layer for our two-layer codec with various α and those for H.263+. Sean is our testing sequence which is encoded at 10 frames/s for a 3-s testing time. Compared to the PSNRs of H.263+, the improvement of our codec in the base layer is between 0.2 and 1.2 dB and, in the enhancement layer, between 1.5 and 2 dB.

V. Conclusion

We propose an FGS video codec and a two-layer scalable video codec. Both use bitplane coding with MP atoms. The proposed FGS algorithm uses the spatial and temporal dependence between bitplanes to exploit the redundancy in the bitplanes. The efficiency of our algorithm in encoding atom positions lies in using a quadtree to represent a bitplane and to perform bitplane prediction. The PSNR of the proposed algorithm is compared to and out performs that of the DCT-based FGS algorithm. A two-layer scalable MP codec is introduced in which the motion vectors are estimated from the weighted sum of the reconstructed base layer and enhancement layer images. With this approach, a better performance is attained in enhancement layer.

References

[1] O. Al-Shaykh, E. Miloslavsky, T. Nomura, R. Neff, and A. Zakhor, "Video compression using matching pursuits," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 1, pp. 123–143, Feb. 1999.

[2] P. A. Chou and Z. Miao, "Rate-distortion optimizedsender-driven streaming over best-effort networks," in *Proc. IEEE 4th Workshop Multimedia Signal Process.*, 2001, pp. 587–592.

[3] P. Czerepinski, C. Davies, N. Canagarajah, and D. Bull, "Matching pursuits video coding: Dictionaries and fast implementation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 7, pp. 1103–1115, Jul. 2000.

[4] B. Girod, M. Kalman, Y. J. Liang, and R. Zhang, "Advances in channel-adaptive video streaming," *J. Wirelesss Commun. Mobile Comput.*, to be published.

[5] *Video Coding for Audio Visual Services at Px64 kbit/s*, 1995. ITU-T Recommendation H.263.

[6] *Video Coding for Low bit rate Communication*, Sept. 1997. ITU-T Recommendation H.263.

[7] *Streaming Video Profile*, 2001. ISO/IEC 14 496-2:2001.

[8] W. Li, "Overview of fine granularity scalability in MPEG-4 video standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 3, pp. 301–317, Mar. 2001.

[9] J. Le and S. Lei, "An embedded still image coder with rate-distortion optimization," *IEEE Trans. Image Process.*, vol. 8, no. 7, pp. 913–924, Jul. 1999.

[10] G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.

[11] S. McCanne, M. Vetterli, and V. Jacobson, "Low-complexity video coding for receiver driven layered multicast," *IEEE J. Sel. Areas Commun.*, vol. 15, no. 8, pp. 983–1001, Aug. 1997.

[12] D. W. Redmill, D. R. Bull, and P. Czerepinki, "Video coding using a fast nonseparable matching pursuits algorithm," in *Proc. IEEE Int. Conf. Image Process.*, 1998, pp. 769–773.

[13] R. Neff and A. Zakhor, "Very low bit rate video coding based on matching pursuits," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 2, pp. 158–171, Feb. 1997.

[14] R. Neff, T. Nomura, and A. Zakhor, "Decoder complexity and performance comparison of matching pursuit and DCT-based MEPG-4 video codecs," in *Proc. IEEE Int. Conf. Image Process.*, 1998, pp. 783–787.

[15] R. Neff and A. Zakhor, "Modulus quantization for matching pursuit video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 6, pp. 895–912, Jun. 2000.

[16] K. Rose and S. L. Regunathan, "Toward optimality in scalable predictive coding," *IEEE Trans. Image Process.*, vol. 10, no. 7, pp. 965–976, Jul. 2001.

[17] A. Said and W. A. Pearlman, "An image multiresolution representation for lossless and lossy compression," *IEEE Trans. Image Process.*, vol. 5, no. 9, pp. 1303–1310, Sep. 1996.

[18] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3445–3462, Dec. 1993.

[19] E. Shusterman and M. Feder, "Image compression via improved quadtree decomposition algorithms," *IEEE Trans. Image Process.*, vol. 3, no. 2, pp. 207–215, Mar. 1994.

[20] M. Vetterli and T. Kalker, "Matching pursuit for compression and application to motion compensated video coding," in *Proc. IEEE Int. Conf. Image Process.*, 1994, pp. 725–729.

[21] C. De Vleeschouwer and B. Macq, "SNR scalability based on matching pursuits," *IEEE Trans. Multimedia*, vol. 2, no. 4, pp. 198–208, 2000.

[22] ——, "Subband dictionaries for low cost matching pursuits of video residues," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 7, pp. 984–993, Oct. 1999.

[23] C. De Vleeschouwer and A. Zakhor, "Atom modulus quantization for matching pursuit video coding," in *Proc. IEEE Int. Conf. Image Process.*, 2002, pp. 681–684.

Jian-Liang Lin was born in I-Lan, Taiwan, R.O.C., in 1975. He received the B.S. degree in electronic engineering from Fu Jen Catholic University, Taipei, Taiwan, in 1997, and the M.S. degree in electrical engineering from National Tsing Hua University, Hsinchu, Taiwan, in 1999. He is currently working toward the Ph.D. degree in communication engineering at National Taiwan University, Taipei.

He is a Research Assistant at the Institute of Information Science, Academia Sinica, Taipei. His research interests include image and video compression, matching pursuits and scalable video coding.

Wen-Liang Hwang received the B.S. degree in nuclear engineering from the National Tsing Hua University, Hsinchu, Taiwan, R.O.C., the M.S. degree in electrical engineering from the Polytechnic Institute of New York, New York, and the Ph.D. degree in computer science from New York University, New York, in 1993.

He was a Postdoctoral Researcher with the Department of Mathematics, University of California, Irvine, in 1994. He became a member of the Institute of Information Science, Academia Sinica, Taipei, Taiwan, in January 1995. He is currently an Associate Research Fellow. His research includes wavelet analysis, signal and image processing, multimedia communication, and computer vision.

Dr. Hwang was awarded the Academia Sinica Research Award for Junior Research in 2001.

Soo-Chang Pei (S'71–M'86–SM'89–F'00) was born in Soo-Auo, Taiwan. R.O.C., in 1949. He received the B.S.E.E. degree from National Taiwan University (NTU), Taipei, Taiwan, R.O.C., in 1970 and the M.S.E.E. and Ph.D. degrees from the University of California, Santa Barbara (UCSB), in 1972 and 1975, respectively.

He was an Engineering Officer in the Chinese Navy Shipyard from 1970 to 1971. From 1971 to 1975, he was a Research Assistant at UCSB. He was the Professor and Chairman in the Electrical Engineering Department, Tatung Institute of Technology from 1981 to 1983 and at NTU from 1995 to 1998. Presently, he is a Professor in the Electrical Engineering Department at NTU. His research interests include digital signal processing, image processing, optical information processing, and laser holography.

Dr. Pei received the National Sun Yet-Sen Academic Achievement Award in Engineering in 1984, the Distinguished Research Award from the National Science Council, R.O.C., from 1990 to 1998, the Outstanding Electrical Engineering Professor Award from the Chinese Institute of Electrical Engineering in 1998, the Academic Achievement Award in Engineering from the Ministry of Education in 1998, the IEEE Fellow in 2000 for contributions to the development of digital eigenfilter design, color image coding, and signal compression, and the Pan Wen-Yuan Distinguished Research Award in 2002 for electrical engineering education in Taiwan, and the National Chair Professor Award from Ministry of Education in 2002. He has been President of the Chinese Image Processing and Pattern Recognition Society in Taiwan from 1996 to 1998, and is a member of Eta Kappa Nu and the Optical Society of America.