Low Bit-Rate, Scalable Video Coding with 3D Set Partitioning in Hierarchical Trees (3D SPIHT)[†]

Beong-Jo Kim
Samsung Electronics Company
Samsung Plaza Building
Telecommunications R & D Center, 9th Floor
Seohyeon-Dong, Pundang-Gu, Sungnam-Shi, Kyounggi-Do
Seoul 463-050, KOREA
Tel: +82-342-779-6723 Fax: +82-342-779-6709
E-mail: bjkim@telecom.samsung.co.kr

Zixiang Xiong
Department of Electrical Engineering
Texas A & M University
College Station, TX 77843
Tel: (409) 862-8683 Fax: (409) 862-4630
E-mail: zx@ee.tamu.edu

William A. Pearlman
Electrical, Computer and Systems Engineering Department
Rensselaer Polytechnic Institute
110 Eighth Street
Troy, NY 12180-3590
Tel: (518) 276-6082 Fax: (518) 276-6261
E-mail: pearlman@rpi.edu

[†]Work performed in the Center for Image Processing Research at Rensselaer Polytechnic Institute.

Low Bit-Rate, Scalable Video Coding with 3D Set Partitioning in Hierarchical Trees (3D SPIHT)

Beong-Jo Kim, Zixiang Xiong, and William A. Pearlman

Abstract

In this paper, we propose a low bit-rate embedded video coding scheme that utilizes a three-dimensional (3D) extension of the set partitioning in hierarchical trees (SPIHT) algorithm which has proved so successful in still image coding. Three-dimensional spatio-temporal orientation trees coupled with powerful SPIHT sorting and refinement renders 3D SPIHT video coder so efficient that it provides comparable performance to H.263 objectively and subjectively when operated at the bit-rates of 30 to 60 kilobits per second with minimal system complexity. Extension to color-embedded video coding is accomplished without explicit bit allocation, and can be used for any color plane representation. In addition to being rate scalable, the proposed video coder allows multiresolutional scalability in encoding and decoding in both time and space from one bit-stream. This added functionality along with many desirable attributes, such as full embeddedness for progressive transmission, precise rate control for constant bit-rate traffic, and low-complexity for possible software-only video applications, makes the proposed video coder an attractive candidate for multimedia applications.

Keywords: video compression, SPIHT, scalability, progressive transmission, embeddedness, multimedia

1 Introduction

The demand for video transmission and delivery over both high and low bandwidth channels has accelerated. The high bandwidth applications include digital video by satellite (DVS) and the future high-definition television (HDTV), both based on MPEG-2 compression technology. The low bandwidth applications are dominated by transmission over the Internet, where most modems transmit at speeds below 64 kilobits per second (Kbps). Under these stringent conditions, delivering compressed video at acceptable quality becomes a challenging task, since the required compression ratios are quite high. Nonetheless, the current test model standard of H.263 does a creditable job in providing video of acceptable quality for certain applications at low bit rates, but better schemes and enhancements with increased functionality are actively being sought by the MPEG-4 and MPEG-7 standards committees.

Functionality now regarded as essential for emerging video and multimedia applications are resolution and fidelity (rate) scalability, the capability of progressive transmission by increasing resolution and increasing fidelity. Moreover, if a system is truly progressive by rate or fidelity, then it can presumably handle both the high-rate and low-rate regimes of digital satellite and Internet video, respectively. Both H.263 and MPEG-2 are based on block discrete cosine transform (DCT) coding of displaced frame differences, where displacements or motion vectors are determined through block-matching estimation methods. The coding algorithms in H.263 and MPEG-2 are not inherently scalable in rate or resolution, but there exists a limited progressive capability within these standards.

In this paper, we present a three-dimensional subband-based video coder that is fast and efficient, and possesses inherently the multimedia functionality of resolution and fine-grain rate scalability in addition to other desirable attributes. Subband coding has been known to be a very effective coding technique. It can be extended naturally to video sequences due to its simplicity and non-recursive structure that limits error propagation within a certain group of frames (GOF). Three-dimensional (3D) subband coding schemes have been designed and applied for mainly high or medium bit-rate video coding. Karlsson and Vetterli [1] took the first step toward 3D subband coding (SBC) using a simple 2-tap Haar filter for temporal filtering. Podilchuk, Jayant, and Farvardin [2] used the same 3D subband framework without motion compensation. It employed adaptive differential pulse code modulation (DPCM), and vector quantization to overcome the lack of motion compensation.

Kronander [3] first presented motion compensated temporal filtering within the 3D SBC framework. Ohm [4, 5] and later Choi and Woods [6, 7] refined the method and utilized different quantization techniques in application to subbands produced by perfect reconstruction filter banks.

Due to the multiresolutional nature of SBC schemes, several scalable 3D SBC schemes have

appeared. Bove and Lippman [8] proposed multiresolutional video coding with a 3D subband structure. Taubman and Zakhor [9, 10] introduced a multi-rate video coding system using global motion compensation for camera panning, in which the video sequence was pre-distorted by translating consecutive frames before temporal filtering with 2-tap Haar filters. The algorithm generates a scalable bit-stream in terms of bit-rate, spatial resolution, and frame rate.

Meanwhile, there have been several research activities on embedded video coding systems based on significance tree quantization, which was introduced by Shapiro for still image coding as the embedded zerotree wavelet (EZW) coder [11]. It was later improved through a more efficient state description in [12] and called improved EZW or IEZW. This two-dimensional (2D) embedded zerotree (IEZW) method has been extended to 3D IEZW for video coding by Chen and Pearlman [13], and showed promise of an effective and computationally simple video coding system without motion compensation, and obtained excellent numerical and visual results. A 3D zero-tree coding through modified EZW has also been used with good results in compression of volumetric images [14]. Recently, a highly scalable embedded 3D SBC system with *tri-zerotrees* [15] for low bit-rate environment was reported with coding results visually comparable, but numerically slightly inferior to H.263. Our current work is very much related to previous 3D subband embedded video coding systems in [13, 14, 15]. Our main contribution is to design an even more efficient and computationally simple video coding system with many desirable attributes.

In this paper, we employ a three dimensional extension of the highly successful set partitioning in hierarchical trees (SPIHT) still image codec [16] and propose a 3D wavelet coding system with features such as complete embeddedness for progressive fidelity transmission, precise rate control for constant bit-rate (CBR) traffic, low-complexity for possible software-only real time applications, and multiresolution scalability. A predecessor of this system operating at higher rates showed superior reconstruction fidelity with lower encoding and decoding time and complexity than MPEG-2 on 30 frames per second, monochrome SIF sequences [17]. The proposed video coding scheme will be tested at low bit rates and will be benchmarked against H.263 to assess its suitability for Internet applications.

The organization of this paper follows. Section 2 summarizes the overall system structure of our proposed video coder. Basic principles of SPIHT will be explained in Section 3, followed in Section 4 by explanations of 3D-SPIHT's attributes of full monochrome and color embeddedness, and multiresolution encoding/decoding scalability. Motion compensation in our proposed video coder is addressed briefly in Section 5. Sections 6 and 7 provide implementation details and simulation results. Section 8 concludes the paper.

2 System Overview

2.1 System Configuration

The proposed video coding system, as shown in Figure 1 consists primarily of a 3D analysis part either with or without motion compensation, and a coding part with the 3D SPIHT kernel that executes the coding algorithm. As we can see, the decoder has the structure symmetric to that of encoder. A group of contiguous frames, hereafter called GOF, will be first temporally transformed with/without motion compensation. Then, each resulting frame will again be separately transformed in the spatial domain. When motion compensated filtering is performed, the motion vectors are separately lossless-coded, and transmitted over the transmission channel with high priority. With our coding system, there is no complication of a rate allocation, nor is there a feedback loop of prediction error signal, which may slow down the efficiency of the system. With the 3D SPIHT kernel, the preset rate will be allocated over each frame of the GOF automatically according to the distribution of actual magnitudes. However, it is possible to introduce a scheme for bit re-alignment by simply scaling one or more subbands to emphasize or deemphasize the bands so as to artificially control the visual quality of the video. This scheme is also applicable to color planes of video, since it is well known fact that chrominance components are less sensitive than the luminance component to the human observer.

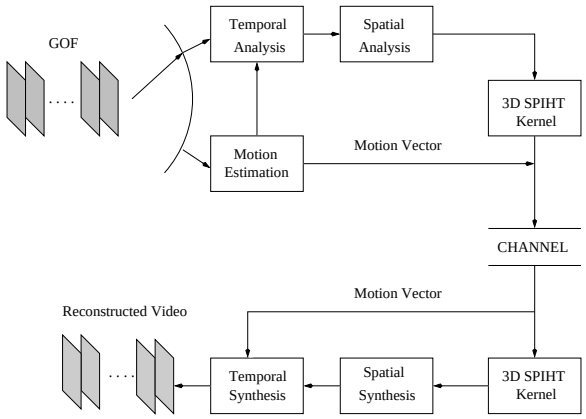


Figure 1: System configuration.

2.2 3D Subband Structure

A GOF is first decomposed temporally and spatially into three-dimensional subbands when fed to a bank of separable (unitary) filters and subsampled. Figure 2 illustrates how a GOF is decomposed into four temporal frequency bands and ten spatial frequency bands by recursive decomposition of the lowest temporal subband before or after recursive splitting of the lowest spatial frequency subband. Since the temporal high frequency usually does not contain much energy, previous works

[2, 4, 5, 6] apply only one level of decomposition to the temporal high frequency band. However, we found that with 3D SPIHT, further dyadic decompositions in the temporal high frequency band give advantages over the traditional way in terms of peak signal-to-noise ratio (PSNR) and visual quality. The similar idea of so-called wavelet packet decomposition [15] also reported better visual quality. In addition, there has been some research on optimum wavelet packet image decomposition to optimize jointly the transform or decomposition tree and the quantization [18, 19]. Nonetheless, the subsequent spatial analysis in this work is fixed for the sake of simplicity and fast execution. The total number of samples in the GOF remains the same at each step in temporal or spatial analysis through the critical subsampling process.

An important issue associated with 3D SBC is the choice of filters. Different filters in general show quite different signal characteristics in the transform domain in terms of energy compaction, and error signal in the high frequency bands [20]. The recent introduction of wavelet theory offers promise for designing better filters for image/video coding. However, the investigation of optimum filter design is beyond of the scope of this paper. We shall employ known filters which have shown good performance in other subband or wavelet coding systems.

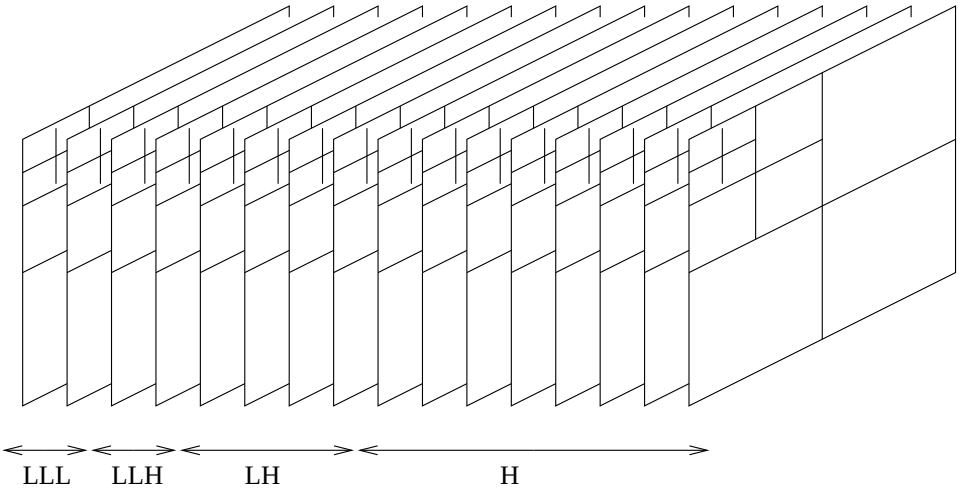


Figure 2: 2D spatial wavelet decomposition followed by 1D temporal decomposition of a GOF (group of frames).

Figure 3 shows two templates, the lowest temporal subband, and the highest temporal subband, of typical 3D wavelet transformed frames with the "Foreman" sequence of QCIF format (176×144). We chose 2 levels of decomposition in the spatial domain just for illustration of the different 3D subband spatial characteristics in the temporal high frequency band. Hence, the lowest spatial band of each frame has dimensions of 44×36. Each spatial band of the frames is appropriately scaled before it is displayed. Although most of the energy is concentrated in the temporal low frequency, there exists much spatial residual redundancy in the high temporal frequency band due to the object or camera motion. This is the main motivation of further spatial decomposition even

in the temporal high subband.

Besides, we can obviously observe not only spatial similarity inside each frame across the different scale, but also temporal similarity between two frames, which will be efficiently exploited by the 3D SPIHT algorithm. When there is fast motion or a scene change, temporal linkages of pixels through trees do not provide any advantage in predicting insignificance (with respect to a given magnitude threshold). However, linkages in trees contained within a frame will still be effective for prediction of insignificance spatially.

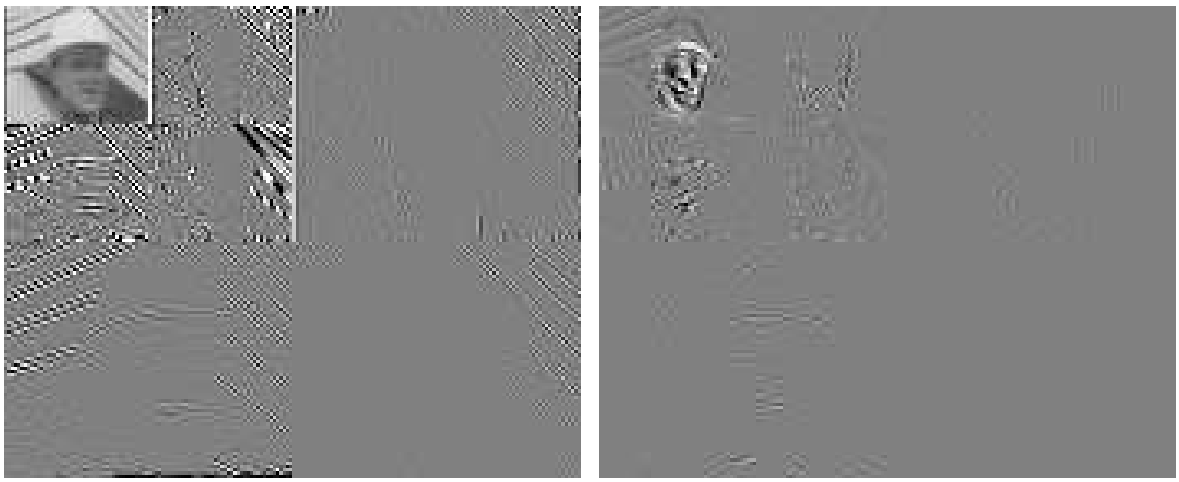


Figure 3: Lowest and the highest temporal subband frames with 2 levels of dyadic spatial decomposition.

3 SPIHT

Since the proposed video coder is based on the SPIHT image coding algorithm [16], the basic principles of SPIHT will be described briefly in this section. The SPIHT algorithm utilizes three basic concepts: (1) searching for sets in *spatial-orientation trees* in a wavelet transform; (2) partitioning the wavelet transform coefficients in these trees into sets defined by the level of the highest significant bit in a bit-plane representation of their magnitudes; and (3) coding and transmitting bits associated with the highest remaining bit planes first.

Spatial orientation trees are groups of wavelet transform coefficients organized into trees rooted in the lowest frequency or coarsest scale subband with offspring in several generations along the same spatial orientation in the higher frequency (resolution) subbands. Figure 4(a) depicts the key for parent-offspring relationship of coefficients to tree nodes for a two-dimensional (2D) wavelet transform with two levels of decomposition. In the spatial orientation trees, each node consists of 2×2 adjacent pixels, and each pixel in the node has 4 offspring except at the highest level of the pyramid, where one pixel in a node indicated by '*' in this figure does not have any offspring. Spatial

orientation trees were introduced to exploit self-similarity and magnitude localization properties in a 2D wavelet transformed image. In particular, if a coefficient magnitude in a certain node of a spatial orientation tree does not exceed a given threshold, it is very likely that none of its descendants will exceed that threshold.

In the case of a wavelet packet transform, frequency bands other than the lowest may be recursively split. A node in the tree at a given level then becomes associated with one pixel at the same corresponding spatial location in each of the four subbands generated from the split. An example of the parent-offspring relationships in such a tree is shown in Figure 4(b), where one of the high frequency bands is further split.

SPIHT consists of two main stages, sorting and refinement. In the sorting stage, SPIHT sets a magnitude threshold 2^n, where n is called the level of significance, and seeks to identify three entities in the spatial-orientation trees: isolated coefficients significant at level n (magnitude no less than 2^n); isolated coefficients insignificant at level n (magnitude less than 2^n); and sets of coefficients insignificant at level n (all their magnitudes less than 2^n). For a given n, the algorithm searches each tree, partitioning the tree into sets of the three entities above and moves their co-ordinates respectively to one of three lists: the list of isolated significant pixels (LSP); the list of isolated insignificant pixels (LIP); and the list of insignificant sets (LIS). The last set can be identified by a single co-ordinate, due to the partitioning rule in the search, where the set of descendants having a significant member is split into its (four) offspring and a subset of all descendants of offspring. When a coefficient is tested and found insignificant, a "0" bit is emitted to the output bit stream and its co-ordinate is moved to the LIP for subsequent testing at lower n. When a coefficient is found significant, a "1" bit and a sign bit are emitted and its co-ordinate is moved to the LSP. When an LIS set is tested for significance at level n, a "0" bit is emitted if insignificant. But when found significant, a "1" bit is emitted and the set is partitioned into offspring and descendants of offspring. The offspring are moved to the end of the LIP and subsequently tested for significance at the same n and also to the LIS as roots of their descendant sets and subsequently tested for significance at the same n.

The bit significance number n is successively lowered in unit increments from the maximum n_{max} of the largest magnitude coefficient. At a given n, the n^{th} of every member of the LSP found significant at a higher n is emitted to the codestream, adding to the "1"'s in the n^{th} bit of the coefficients just found significant for the same n. This is called the refinement stage of the algorithm. When n is decremented, the LIP is tested for significance, and the test result is emitted as a "0" or "1" bit for insignificant or significant, respectively. If significant, its co-ordinates are moved to the LSP and a sign bit is emitted. Then the LIS is visited and its tree sets are partitioned according to the results of the significance tests. The process terminates when the desired bit rate or quality level is reached.

The decoder of the code bitstream receives the outputs of the significance tests and can therefore build the same lists, the LIP, LIS, and LSP, as in the encoder. Therefore, as input bits are read from the codestream, it reconstructs the magnitude and sign bits of LSP members seen by the encoder. The coefficients of the final LIP and LIS sets are set to zero. In the wavelet transform of an image, large sets of zero values exist which are identified efficiently by SPIHT with a single bit. Moreover, significant coefficients are never represented by more bits than needed in their natural binary representation, since the highest "1" bit is always known. One can refer to [16] for more details.

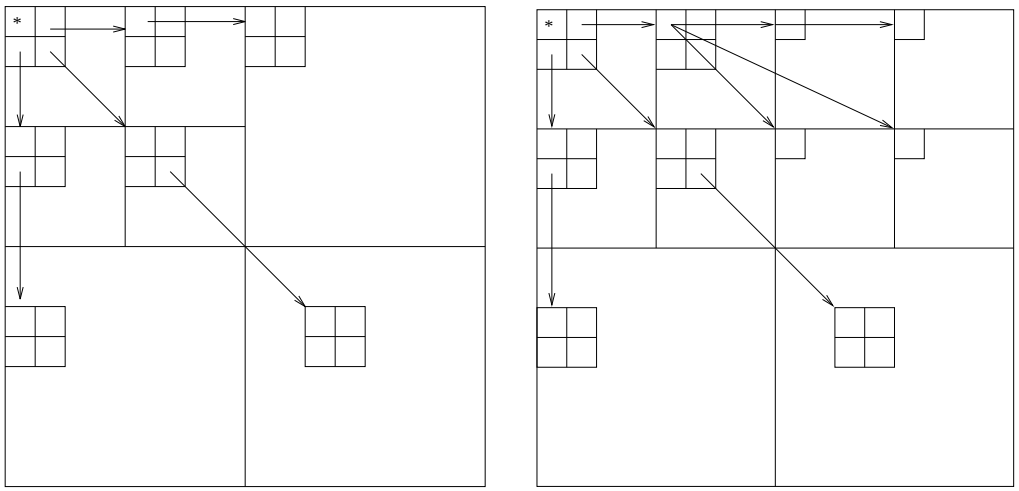


Figure 4: Spatial orientation tree for (a) the dyadic wavelet decomposition case and (b) a wavelet packet decomposition case.

4 3D SPIHT and Some Attributes

This section introduces the extension of the concept of SPIHT still image coding to 3D video coding. Our main concern is to keep the same simplicity of 2D SPIHT, while still providing high performance, full embeddedness, and precise rate control.

4.1 Spatio-temporal Orientation Trees

Here, we provide the 3D SPIHT scheme extended from the 2D SPIHT, having the following three similar characteristics: (1) partial ordering by magnitude of the *3D wavelet* transformed video with a *3D set partitioning* algorithm, (2) ordered bit plane transmission of refinement bits, and (3) exploitation of self-similarity across *spatio-temporal* orientation trees. In this way, the compressed bit stream will be completely embedded, so that a single file for a video sequence can provide progressive video quality, that is, the algorithm can be stopped at any compressed file size or let

run until nearly lossless reconstruction is obtained, which is desirable in many applications including HDTV.

In the previous section, we have studied the basic concepts of 2D SPIHT. We have seen that there is no constraint to dimensionality in the algorithm itself, as pixels are sorted regardless of dimensionality. If all pixels are lined up in magnitude decreasing order, then what matters is how to transmit significance information with respect to a given threshold. In 3D SPIHT, sorting of pixels proceeds just as it would with 2D SPIHT, the only difference being 3D rather than 2D tree sets. Once the sorting is done, the refinement stage of 3D SPIHT will be exactly the same.

A natural question arises as to how to sort the pixels of a three dimensional video sequence. Recall that for an efficient sorting algorithm, 2D SPIHT utilizes a 2D subband/wavelet transform to compact most of the energy to a small number of pixels, and generates a large number of pixels with small or even zero value. Extending this idea, one can easily consider a 3D wavelet transform operating on a 3D video sequence, which will naturally lead to a 3D video coding scheme.

On the 3D subband structure, we define a new 3D *spatio-temporal* orientation tree, and its parent-offspring relationships. When the spatial and temporal filtering alternate, so that the decomposition is purely dyadic, a straightforward extension from the 2D case is to form a node in 3D SPIHT as a block eight adjacent pixels with two extending to each of the three dimension, hence forming a node of $2 \times 2 \times 2$ pixels. The root nodes (at the highest level of the pyramid) have one pixel with no descendants and the other seven pointing to 8 offspring in a $2 \times 2 \times 2$ cube at corresponding locations at the same level. For non-root and non-leaf nodes, a pixel has 8 offspring in a $2 \times 2 \times 2$ cube one level below in the pyramid. Figure 5(a) depicts these parent-offspring relationships in the case of a two-level dyadic 3D decomposition with 15 subbands, produced by a once repeated spatial-horizontal, spatial-vertical, and temporal splitting in that order. A wavelet packet transform, on the other hand, may produce a split of a given subband at any level into a number of smaller subbands, so that the $2 \times 2 \times 2$ offspring nodes are split into pixels in these smaller subbands at the corresponding orientations in the nodes at the original level. Figure 5(b) illustrates the parent-offspring relationships in 21 subbands produced by two levels of spatial horizontal and vertical dyadic splitting followed by a two level temporal dyadic splitting.

The packet transform has been chosen in this work, because it allows a different number of decompositions between the spatial and temporal dimensions and thereby achieves better compression results than the purely dyadic decompositions. Therefore, we can decompose into more spatial levels than temporal ones. It is advantageous to limit the number of frames to be buffered to form a coding unit in order to save memory and coding delay. We do not want to limit the spatial decompositions to the same small number, since more spatial decompositions usually produce noticeable coding gains. Indeed, in previous articles [13, 17], we have reported video coding results using the same number of decompositions spatially and temporally. In the trees for the packet transform,

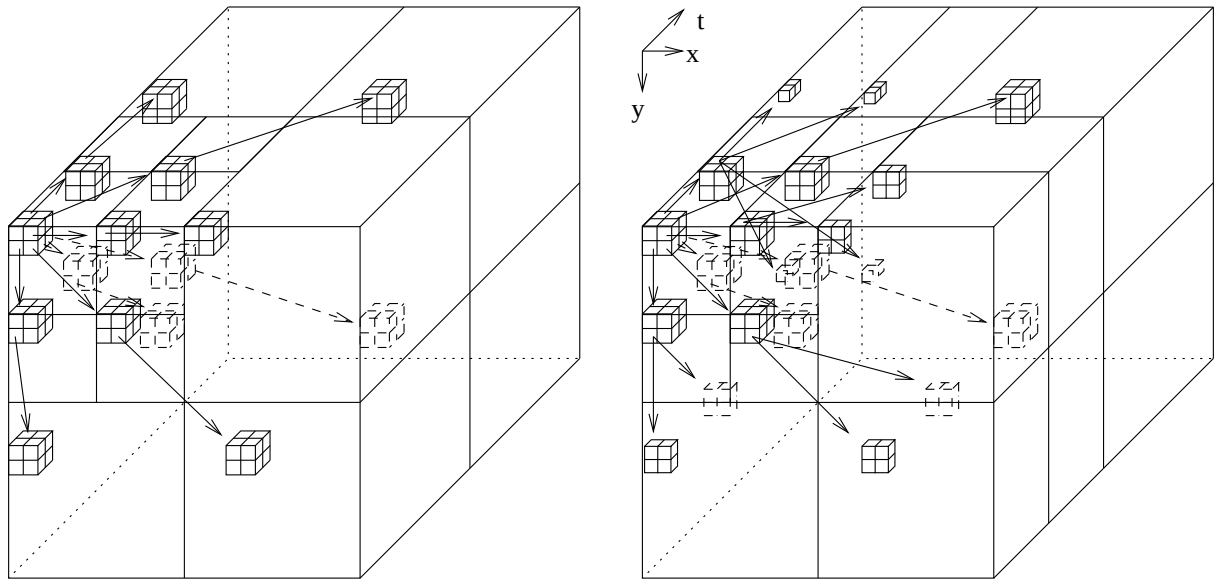


Figure 5: *Spatio-temporal* orientation tree for (a) the 2-level dyadic wavelet decomposition case (15 subbands) and (b) a 2-level wavelet packet decompostion case (2D spatial + 1D temporal, 21 subbands).

the offspring pixels are not always contiguous and the maximum depths may be different, but they can be tracked internally within the algorithm without extra overhead in bit rate.

With a smaller group of frames in a coding unit and allowance of different depth trees, there are more possibilities in the choice of filter implementations. For example, one can now use a shorter filter with a GOF of 4 or 8, such as the Haar or S+P [21] filters, which use only integer operations, with the latter being the more efficient. Only two or three decompositions are possible with a GOF of four or eight, respectively. However, with a 352×288 CIF frame, for example, five spatial (dyadic) decompositions can be achieved with the high performance 9/7 biorthogonal filter [22].

Since we have already 3D wavelet-transformed the video sequence to set up 3D spatio-temporal trees, the next step is compression of the coefficients into a bit-stream. Essentially, it can be done by feeding the 3D data structure to the 3D SPIHT coding kernel. The 3D SPIHT kernel will sort the data according to the magnitude along the *spatio-temporal orientation trees* (sorting pass), and refine the bit plane by adding necessary bits (refinement pass). At the destination, the decoder will follow the same execution path conveyed by the received significance decision bits to recover the data.

4.2 Color Video Coding

So far, we have considered only one color plane, namely luminance. In this section, we will consider a simple application of the 3D SPIHT to any color video coding, while still retaining full

embeddedness, and precise rate control.

A simple application of the SPIHT to color video would be to code each color plane separately as does a conventional color video coder. Then, the generated bit-stream of each plane would be serially concatenated. However, this simple method would require allocation of bits among color components, losing precise rate control, and would fail to meet the requirement of the full embeddedness of the video codec since the decoder needs to wait until the full bit-stream arrives to reconstruct and display. Instead, one can treat all color planes as one unit at the coding stage, and generate one *mixed* bit-stream so that we can stop at any point of the bit-stream and reconstruct the color video of the best quality at the given bit-rate. In addition, we want the algorithm to automatically allocate bits optimally among the color planes. By doing so, we will still keep the claimed full embeddedness and precise rate control of 3D SPIHT. The bit-streams generated by both methods are depicted in the Figure 6, where the first one shows a conventional color bit-stream, while the second shows how the color embedded bit-stream is generated, from which it is clear that we can stop at any point of the bit-stream, and can still reconstruct a color video at that bit-rate as opposed to the first case.

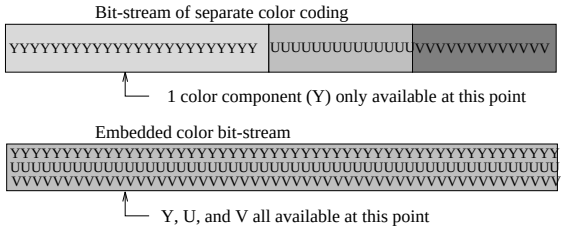


Figure 6: Bit-streams of two different methods:separate color coding, embedded co lor coding.

Let us consider a tri-stimulus color space with luminance Y plane such as YUV, YCrCb, etc. for simplicity. Each such color plane will be separately wavelet transformed, having its own pyramid structure. Now, to code all color planes together, the 3D SPIHT algorithm will initialize the LIP and LIS with the appropriate coordinates of the top level in all three planes. Figure 7 shows the initial internal structure of the LIP and LIS, where Y,U, and V stand for the coordinates of each root pixel in each color plane. Since each color plane has its own spatial orientation trees, which are mutually exclusive and exhaustive among the color planes, it automatically assigns the bits among the planes according to the significance of the magnitudes of their own coordinates. The effect of the order in which the root pixels of each color plane are initialized will be negligible except when coding at extremely low bit-rate. Note also that the wavelet transforms and sizes may be different among the three planes without affecting the method. For example, for our video sequences in YUV 4:2:0 (or 4:1:1) format, the U and V chrominance planes are one-fourth the size of the luminance Y plane, as depicted in Figure 7.

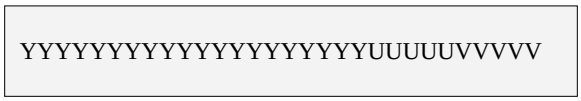


Figure 7: Initial internal structure of LIP and LIS, assuming the U and V planes are one-fourth the size of the Y plane.

4.3 Scalability of SPIHT image/video Coder

4.3.1 Overview

In this section, we address multiresolution encoding and decoding in the 3D SPIHT video coder. Although the proposed video coder naturally gives scalability in rate, it is highly desirable also to have temporal and/or spatial scalabilities for today's many multimedia applications such as video database browsing and multicast network distributions. Multiresolution decoding allows us to decode video sequences at different rate and spatial/temporal resolutions from one bit-stream. Furthermore, a layered bit-stream can be generated with multiresolution encoding, from which the higher resolution layers can be used to increase the spatial/temporal resolution of the video sequence obtained from the low resolution layer. In other words, we achieve full scalability in rate and partial scalability in space and time with multiresolution encoding and decoding.

Since the 3D SPIHT video coder is based on the multiresolution wavelet decomposition, it is relatively easy to add multiresolutional encoding and decoding as functionalities in partial spatial/temporal scalability. In the following subsections, we first concentrate on the simpler case of multiresolutional decoding, in which an encoded bit-stream is assumed to be available at the decoder. This approach is quite attractive since we do not need to change the encoder structure. The idea of multiresolutional decoding is very simple: we partition the embedded bit-stream into portions according to their corresponding spatio-temporal frequency locations, and only decode the ones that contribute to the resolution we want.

We then turn to multiresolutional encoding, where we describe the idea of generating a layered bit-stream by modifying the encoder. Depending on bandwidth availability, different combinations of the layers can be transmitted to the decoder to reconstruct video sequences with different spatial/temporal resolutions. Since the 3D SPIHT video coder is symmetric, the decoder as well as the encoder knows exactly which information bits contribute to which temporal/spatial locations. This makes multiresolutional encoding possible as we can order the original bit-stream into layers, with each layer corresponding to a different resolution (or portion). Although the layered bit-stream is not fully embedded, the first layer is still rate scalable.

4.3.2 Multiresolutional Decoding

As we have seen previously, the 3D SPIHT algorithm uses significance map coding and spatial orientation trees to efficiently predict the insignificance of descendant pixels with respect to a current threshold. It refines each wavelet coefficient successively by adding residual bits in the refinement stage. The algorithm stops when the size of the encoded bit-stream reaches the exact target bit-rate. The final bit-stream consists of significance test bits, sign bits, and refinement bits.

In order to achieve multiresolution decoding, we have to partition the received bit-stream into portions according to their corresponding temporal/spatial location. This is done by putting two flags (one spatial and one temporal) in the bit-stream during the process of *decoding*, when we scan through the bit-stream and mark the portion that corresponds to the temporal/spatial locations defined by the input resolution parameters. As the received bit-stream from the decoder is embedded, this partitioning process can terminate at any point of the bit-stream that is specified by the decoding bit-rate. Figure 8 shows such a bit-stream partitioning. The dark-gray portion of the bit-stream contributes to low-resolution video sequence, while the light-gray portion corresponds to coefficients in the high resolution. We only decode the dark-gray portion of the bit-stream for a low-resolution sequence and scale down the 3D wavelet coefficients appropriately before the inverse 3D wavelet transformation. We can further partition the dark-gray portion of the bit-stream in Figure 8 for decoding in even lower resolutions.

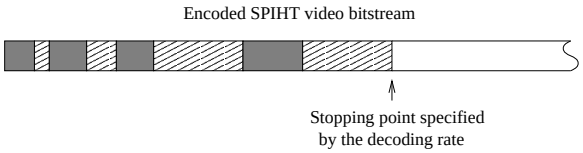


Figure 8: Partitioning of the SPIHT encoded bit-stream into portions according to their corresponding temporal/spatial locations.

By varying the temporal and spatial flags in decoding, we can obtain different combinations of spatial/temporal resolutions in the encoder. For example, if we encode a QCIF sequence at 24 f/s using a 3-level spatial-temporal decomposition, we can have in the decoder three possible spatial resolutions (176×144, 88×72, 44×36), three possible temporal resolutions (24, 12, 6), and any bit rate that is upper-bounded by the encoding bit-rate. Any combination of the three sets of parameters is an admissible decoding format.

Obvious advantages of scalable video decoding are savings in memory and decoding time. In addition, as illustrated in Figure 8, information bits corresponding to a specific spatial/temporal resolution are not distributed uniformly over the compressed bit-stream in general. Most of the lower resolution information is crowded at the beginning part of the bit-stream, and after a certain point, most of the bit rate is spent in coding the highest frequency bands which contain the detail

of video which are not usually visible at reduced spatial/temporal resolution. What this means is that we can set a very small bit-rate for even faster decoding and browsing applications, saving decoding time and channel bandwidth with negligible degradation in the decoded video sequence.

4.4 Multiresolutional Encoding

The aim of multiresolutional encoding is to generate a layered bit-stream. But, information bits corresponding to different resolutions in the original bit-stream are interleaved. Fortunately, the SPIHT algorithm allows us to keep track of the temporal/spatial resolutions associated with these information bits. Thus, we can change the original encoder so that the new encoded bit-stream is layered in temporal/spatial resolutions. Specifically, multiresolutional encoding amounts to putting into the first (low resolution) layer all the bits needed to decode a low resolution video sequence, in the second (higher resolution) layer those to be added to the first layer for decoding a higher resolution video sequence and so on. This process is illustrated in Figure 9 for the two-layer case, where scattered segments of the dark-gray (and light-gray) portion in the original bit-stream are put together in the first (and second) layer of the new bit-stream. A low resolution video sequence can be decoded from the first layer (dark-gray portion) alone, and a full resolution video sequence from both the first and the second layers.

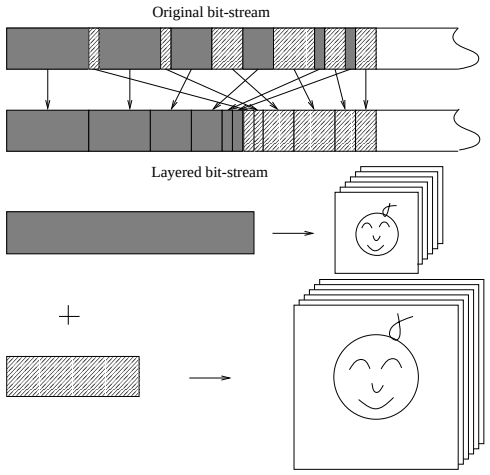


Figure 9: A multiresolutional encoder generates a layered bit-stream, from which the higher resolution layers can be use to increase the spatial resolution of the frame obtained from the low resolution layer.

As the layered bit-stream is a reordered version of the original one, we lose overall scalability in rate after multiresolutional encoding. But the first layer (i.e., the dark gray layer in Figure 9) is still embedded, and it can be used for lower resolution decoding.

Unlike multiresolutional decoding in which the full resolution encoded bit-stream has to be transmitted and stored in the decoder, multiresolutional encoding has the advantage of wasting no

bits in transmission and decoding at lower resolution. The disadvantages are that it requires both the encoder and the decoder agree on the resolution parameters and the loss of embeddedness at higher resolution, as mentioned previously.

5 Motion Compensation

When there is a considerable amount of motion either in the form of global camera motion or object motion within a GOF, the PSNR of reconstructed video will fluctuate noticeably due to pixels with high magnitude in the temporal high frequency. Typical applications with low bitrate deal with rather simple video sequences with slow object and camera motion. However, the usual frame sampling rate of 10 frames per second in low bitrate applications may give rise to sufficient pixel displacement between frames to benefit from a scheme that estimates these displacements and suitably compensates for them. The degree of any improvement in performance must be assessed versus the the cost in extra complexity. It is in this spirit that we offer a particular scheme of motion compensation as an option for our coder, which we call motion-compensated (MC) 3D SPIHT.

5.1 Hierarchical Motion Estimation

Herein we utilize a typical hierarchical block matching scheme (see [23] or [24]) to estimate pixel displacements (motion vectors) between successive frames in the sequence. In a hierarchical scheme, starting from the highest (coarsest) level of the pyramid, motion estimates at a given level are successively used for the initial estimates of the motion vectors at the next lower (finer) level until the final estimates for the full frame are found. We utilize the idea of motion compensated filtering first proposed by Ohm [4, 5] in this framework and developed further by Choi and Woods [6, 7]. For every given pixel in the initial frame, motion vectors are used to trace a motion trajectory through the frames of the sequence. Problems arise when pixels in a frame are not connected to a trajectory or to more than one trajectory. We adopt the method of [6, 7] to treat this so-called *connected/unconnected* pixel problem, so that every pixel associated with one and only one trajectory. Filtering is applied to each trajectory to achieve the required temporal decomposition.

One of the main problems in incorporating motion compensation into video coding, especially at low bit-rates, is the overhead associated with the motion vector. A smaller block size generally increases the amount of overhead information. A block size that is too large fails to reasonably estimate diverse motions in the frames. In [7], several different block sizes were used with a hierarchical, variable size block matching method. In that case, additional overhead for the image map for variable sizes of blocks and the motion vectors are needed to be transmitted as side information. Since we can not know the characteristics of the video beforehand, it is difficult to choose optimum block size and search window size. However, empirically we provide Table 1 for the parameters for our hierarchical block matching method for motion estimation, where there are

Level	3	2	1
Search Window	± 2	± 2	± 2
Option 1	4	8	16
Option 2	8	16	32

Table 1: Set of parameters for 3-Level hierarchical block matching.

two options in choosing block size. Motion vectors obtained at a certain level will be scaled up by 2 to be used for initial motion vectors at the next stage.

6 Implementation Details

Performance of video coding systems with the same basic algorithm can be quite different according to the actual implementation. Thus, it is necessary to specify the main features of the implementation in detail. In this section, we will describe some issues such as filter choice, image extension, and modeling of arithmetic coding, that are important in practical implementation.

The S+P [21] and Haar (2 tap) [7, 2] filters are used only for the temporal direction, with the Haar used only when motion-compensated filtering is utilized. The 9/7-tap biorthogonal wavelet filters [22] have the best energy compaction properties, but having more taps, are used for spatial and temporal decomposition for a size 16 GOF. For GOF sizes of 4 and 8, the S+P filter is used to produce one and two levels of temporal decomposition, respectively. In all cases, three levels of decomposition are performed with the 9/7 biorthogonal wavelet filters.

With these filters, filtering is performed with a convolution operation recursively. Since we need to preserve an even number for dimensions of the highest level of pyramid image of each frame, given some desired number of spatial levels, it is often necessary to extend the frame to a larger image before 2D spatial transformation. For example, we want to have at least 3 levels of spatial filtering for QCIF (176×144) video sequence with 4:2:0 or 4:1:1 subsampled format. For the luminance component, the highest level of the image is 22×18. However, for chrominance components, it will be 11×9 which is not appropriate for the coding stage. To allow 3 levels of decomposition, a simple extrapolation scheme is used to make dimension of chrominance component 96×80 which results in 12×10 of root image after 3 levels of decomposition. Generally, when extending the image by artificially augmenting its boundary can cause some loss of performance. However, when extending the image in a smooth fashion to avoid generation of artificial high frequency coefficients, the performance loss is expected to be minimal.

After the 3D subband/wavelet (wavelet packet) transformation is completed, the 3D SPIHT algorithm is applied to the resulting multiresolution pyramid. Then, the output bit-stream is further compressed with an arithmetic encoder [25]. To increase the coding efficiency, groups of $2 \times 2 \times 2$ coordinates were kept together in the list. Since the amount of information to be coded

depends on the number of insignificant pixels m in that group, we use several different adaptive models, each with 2^m symbols, where $m \in \{1, 2, 3, 4, 5, 6, 7, 8\}$, to code the information in a group of 8 pixels. By using different models for the different number of insignificant pixels, each adaptive model contains better estimates of the probabilities conditioned to the fact that a certain number of adjacent pixels are significant or insignificant.

Lastly, when motion compensated temporal filtering is applied, we will need to code motion vectors. We call a group of motion vectors belonging to a given level of temporal decomposition a motion vector field (MVF). When GOF = 16, we have 8 first level MVFs, 4 second level MVFs, and 2 third level MVF, resulting in total 14 MVFs to code. In the experiment, we have found that MVFs from different levels have quite different statistical characteristics. In general, more unconnected pixels are generated at higher levels of temporal decomposition. Furthermore, horizontal and vertical motion are assumed to be independent of each other. Thus, each motion vector component is separately coded conditioned to the level of decomposition. For the chrominance components, the motion vector obtained from the luminance component will be used with an appropriate scale factor.

7 Coding Results

7.1 Coding of QCIF Sequences

We now turn to the testing of the 3D SPIHT codec with color video QCIF sequences with a frame size of 176×144 and frame rate of 10 f/s (frames per second). A forerunner of this codec, having no option for motion compensation, proved superior to MPEG-2 in tests with SIF (352x240) monochrome 30 f/s sequences [17]. In those tests GOF sizes of 4,8, and 16 frames were used, but, because filtering and coding are so much faster per frame than with the larger SIF frames, a GOF size of 16 was selected for all these experiments. In this section, we shall provide simulation results and compare the proposed video codec with H.263 in various aspects, such as objective and subjective performance, system complexity, and performance at different camera and object motion. The latest test model number of H.263, tmn2.0 (test model number 2.0), was downloaded from the public domain (ftp://bonde.nta.no/pub/tmn/). As in H.263, the 3D SPIHT video codec is a fully implemented software video codec.

We tested three different color video sequences: "Carphone", "Mother and Daughter", and "Hall Monitor". These video sequences cover a variety of object and camera motions. "Carphone" is a representative sequence for video-telephone application. This has more complicated object motion than the other two, and camera is assumed to be stationary. However, the background outside the car window changes very rapidly. "Mother and Daughter" is a typical head-and-shoulder sequence with relatively small object motion and camera is also stationary. The last sequence "Hall Monitor"

is suitable for a monitoring application. The background is always fixed and some objects (persons) appear and then disappear.

All the tests were performed at the frame-rate of 10 f/s by coding every third frame. The successive frames of this downsampled sequence are now much less correlated than in the original sequence. Therefore, the action of temporal filtering is less effective for further decorrelation and energy compaction. For a parallel comparison, we first run with H.263 at a target bit-rates (30k and 60k) with all the advanced options (-D -F -E) enabled. In general, our H.263 software (by Telenor) did not provide simultaneous exact bit rate and frame rate due to the buffer control. For example, H.263 produced actual bit-rate and frame-rate of 30080 bps and 9.17 f/s for the target bit-rate and frame-rate of 30000 bps and 10 f/s. With our proposed video codec, the exact target bit rates can be obtained. However, since 3D SPIHT is fully embedded video codec, we only needed to decode at the target bit-rates with one bit-stream compressed at the larger bit-rate.

Figures 10, 11, and 12 show frame-by-frame PSNR comparison among the luminance frames coded by MC 3D SPIHT, 3D SPIHT, and H.263. From the figures, 3D SPIHT is 0.89 – 1.39 dB worse than H.263 at the bit-rate of 30k. At the bit-rate of 60k, less PSNR advantage of H.263 over 3D SPIHT can be observed except the "Hall Monitor" sequence for which 3D SPIHT has a small advantage over H.263. In comparing MC 3D SPIHT with 3D SPIHT, MC 3D SPIHT generally gives more stable PSNR fluctuation than 3D SPIHT (without MC), since MC reduces large magnitude pixels in temporal high subbands resulting in more uniform rate allocations over the GOF. In addition, a small advantage of MC 3D SPIHT over 3D SPIHT was obtained for the sequences with translational object motion. However, for the "Hall Monitor" sequence, where 3D SPIHT outperforms MC 3D SPIHT in terms of average PSNR, side information associated with motion vectors seemed to exceed the gain provided by motion compensation. In terms of visual quality, H.263, 3D SPIHT, and MC 3D SPIHT showed very competitive performance as shown in Figures 13, 14, and 15 although our proposed video coders have lower PSNR values at the bit rate of 30 kbps. In general, H.263 preserves edge information of objects better than 3D SPIHT, while 3D SPIHT exhibits blurs in some local regions. However, as we can observe in Figure 13, H.263 suffers from blocking artifacts around the mouth of the person and background outside the car window. In overall, 3D SPIHT and MC 3D SPIHT showed comparable results in terms of average PSNR and visual quality of reconstructed frames. As in most 3D subband video coders, one can observe that the PSNR dips at the GOF boundaries, partly due to object motion and partly due to the boundary extension for temporal filtering. However, they are not manifested visibly in the reconstructed video. Smaller GOF sizes of 4 and 8 have been used with shorter filters, such as Haar and S+P, for temporal filtering. Consistent with the SIF sequences, the results are not significantly different, but slightly inferior in terms of average PSNR. Again, the fluctuation in PSNR from frame to frame is smaller with the shorter GOF's.

Bit-rate	30 Kbps (Y U V) dB	60 Kbps (Y U V) dB
3-D SPIHT	32.95 38.15 40.68	37.95 40.41 42.38
H.263	33.84 37.79 40.03	37.50 39.81 41.65

Table 2: Coding Results of "Hall monitor" sequence (frames 0 - 285) at bit-rates of 30 Kbps and 60 Kbps, and frame-rate of 10 f/s.

As we discussed earlier section, scalability in video coding is very important for many multimedia applications. Figure 16 shows first frames of the decoded "Carphone" sequence at different spatial/temporal resolutions from the same encoded bit-stream. Note the improved quality of the lower spatial/temporal resolutions. It is due to the fact that the SPIHT algorithm extracts first the bits of largest significance in the largest magnitude coefficients, which reside in the lower spatial/temporal resolutions. At low bit rates, the low resolutions will receive most of the bits and be reconstructed fairly accurately, while the higher resolutions will receive relatively few bits and be reconstructed rather coarsely.

7.2 Embedded Color Coding

Next, we provide simulation results of color-embedded coding of QCIF video sequences. Recall that QCIF has the 4:2:0 (often called 4:1:1) format, where both chrominance planes are subsampled horizontally and vertically by a factor of two. Note that not many video coding schemes perform reasonably well in both high bit-rate and low bit-rate. To demonstrate embeddedness of 3-D color video coder, we reconstructed video at bit-rates of 30 Kbps and 60 Kbps, and frame-rate of 10 f/s, as before, by coding every third frame, with the same color-embedded bit-stream compressed at higher bit-rate 100 Kbps. In this simulation, we remind the reader that 16 frames were chosen for a GOF, since required memory and computational time are reasonable with QCIF sized video. The average PSNRs coding frame $0 - 285$ (96 frames coded) are shown in the Table 2, in which we can observe that 3-D SPIHT gives average PSNRs of Y component slightly inferior to those of H.263 and average PSNRs of U and V components slightly better than those of H.263. However, visually 3-D SPIHT and H.263 show competitive performances as shown in Figure 15. Overall, color 3-D SPIHT still preserves precise rate control, self-adjusting rate allocation according to the magnitude distribution of the pixels in all three color planes, and full embeddedness.

7.3 Computation Times

Now, we assess the computational complexity in terms of the run times of the stages of transformation, encoding, decoding, and search for maximum magnitudes. First, relative times per frame are shown in Table 3 for 3D SPIHT, motion-compensated (MC) 3D SPIHT and H.263 for encoding the Carphone sequence at 10 f/s, every third frame from frame 0 to 285, at 30 kbps. As seen in the table, 3D SPIHT is 2.53 times faster than H.263, but with motion compensation is 1.2 times

slower. In motion-compensated 3D SPIHT, most of the execution time is spent in hierarchical motion estimation.

	3D SPIHT	MC 3D SPIHT	H.263
Relative complexity	1	3.09	2.53

Table 3: Comparison of 3D SPIHT, MC 3D SPIHT, and H.263 relative computational complexity for encoding 0 - 285 "Carphone" sequence at 10 f/s, and 30 kbps. Complexity is computed on SUN SPARC 20 machine.

A more detailed breakdown of computation times was undertaken for the various stages in the encoding and decoding of QCIF 4:2:0 YUV color sequences. Table 4 shows run times of these stages on a SUN SPARC 20 for 96 frames of the "Carphone" sequence, specifically every third frame of frames 0–285 (10 f/s), encoded at 30 Kbps. Note that 57 % of the encoding time and 78 % of the decoding time are spent on wavelet transformation and its inverse, respectively. Considering that the 3D SPIHT coder is not fully optimized and there have been recent advances in fast filter design, 3D SPIHT shows promise of becoming a real-time software-only video codec, as seen from the actual coding and decoding times in Table 4.

Functions	time in sec	relative time (%)
3D wavelet transform	16.22	56.95
3D SPIHT compression	1.27	4.46
Maximum magnitude computation	6.16	21.63
I/O	4.83	16.96
Total time	28.48	100.00

Functions	time in sec	relative time (%)
3D inverse-wavelet transform	14.60	78.37
3D SPIHT decompression	0.86	4.62
I/O	3.17	17.01
Total time	18.63	100.00

Table 4: System complexity of 3D SPIHT encoder/decoder in terms of time. Coding time is based on Sun SPARC 20. The video "Carphone" (0 - 285) was coded at 10 f/s, and 30 kbps. Total time is for encoding/decoding 96 frames.

In order to quantitatively assess the time saving in decoding video sequences progressively by increasing resolution, we give the total decoder running time on a SUN SPARC 20, including inverse wavelet transform, and I/O in Table 5 on a SUN SPARC 20. From Table 5, we observe that significant computational time saving can be obtained with the multiresolutional decoding

scheme. In particular, when we decoded at half spatial/temporal resolution, we obtained more than 5 times faster decoding speed than full resolution decoding. A significant amount of that time saving results from fewer levels of inverse wavelet transforms on smaller images.

	Spatial full (%)	Spatial half (%)
Temporal full	100	31.52
Temporal half	62.76	19.39

Table 5: Relative decoding time in % for the "Carphone" sequence at different temporal and spatial resolutions. Full resolution time is 18.63 second for decoding frame 0 – 285 at 10 f/s and 30 kbps. (Hence, total number of frames decoded is 96.)

8 Conclusions

We have presented an embedded subband-based video coder, which employs the SPIHT coding algorithm in 3D spatio-temporal orientation trees in video coding, analogous to the 2D spatial orientation trees in image coding. The video coder is fully embedded, so that different degrees of monochrome or color video quality can thus be obtained with a single compressed bit stream. The cost for this embeddedness is the coding delay (latency) to accept 16 frames into a buffer and a memory size of the order of the size of the coding unit to execute the 3D SPIHT algorithm. However, there are implementations under investigation which substantially reduce the latency and dynamic memory usage. The simulations with 16-frame units provided the desired performance in quality and speed. However, very little loss in quality is encountered for smaller coding units of 8 or even 4 frames, which would give consequent reductions in latency and memory usage with the present implementation.

The algorithm approximates the original sequence successively by bit plane quantization according to magnitude comparisons, so that precise rate control and self-adjusting rate allocations are automatically achieved. In addition, spatial and temporal scalability can be easily incorporated into the system to meet various types of display parameter requirements.

The proposed video coding is so efficient that, even without motion compensation, it showed comparable results, visually and measurably, to that of H.263 for the sequences tested. Without motion compensation, the temporal decorrelation obtained from subbanding along the frames is more effective for the higher 30 f/s frame rate sequences of MPEG-2 than for the 10 f/s sequences of QCIF. The local motion-compensated filtering used with QCIF provided more uniformity in rate and PSNR over a group of frames, which was hard to discern visually. For video sequences in which there is considerable camera pan and zoom, a global motion compensation scheme, such as that proposed in [9], will probably be required to maintain coding efficiency. However, attractive

features of 3D SPIHT are speed and simplicity, so one must be careful not to sacrifice these features for minimal gains in coding efficiency.

References

[1] G. Karlsson and M. Vetterli, "Three dimensional subband coding of video," *Proc. Int. Conf. on Acoust., Speech, and Signal Processing (ICASSP)*, pp. 1100–1103, April 1988.

[2] C. I. Podilchuk, N. S. Jayant, and N. Farvardin, "Three-dimensional subband coding of video," *IEEE Transactions on Image Processing*, vol. 4, no. 2, pp. 125–139, Feb. 1995.

[3] T. Kronander, "New results on 3-dimensional motion compensated subband coding," *Proc. PCS-90*, Mar 1990.

[4] J. R. Ohm, "Advanced packet video coding based on layered VQ and SBC techniques," *IEEE Transactions on Circuit and System for Video Technology*, vol. 3, no. 3, pp. 208–221, June 1993.

[5] J. R. Ohm, "Three-dimensional subband coding with motion compensation," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 559–571, Sep. 1994.

[6] S. J. Choi and J. W. Woods, "Motion-compensated 3-D subband coding of video," *IEEE Transactions on Image Processing*, vol. 8, pp. 155–167, Feb. 1999.

[7] S. J. Choi, *Three-Dimensional Subband/Wavelet Coding of Video with Motion Compensation*, Ph.D. thesis, Rensslaer Polytechnic Institute, 1996.

[8] V. M. Bove and A. B. Lippman, "Scalable open-architecture television," *SMPTE J.*, pp. 2–5, Jan. 1992.

[9] D. Taubman and A. Zakhor, "Multirate 3-d subband coding of video," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 572–588, Sep. 1994.

[10] D. Taubman, *Directionality and scalability im Image and Video Compression*, Ph.D. thesis, University of California, Bekerley, 1994.

[11] J. M. Shapiro, "An embedded wavelet hierarchical image coder," *Proc. IEEE Int. Conf. on Acoust., Speech, and Signal Processing (ICASSP), San Francisco*, pp. IV 657–660, March 1992.

[12] A. Said and W. A. Pearlman, "Image compression using the spatial-orientation tree," *Proc. IEEE Int. Symp. Circuits and Systems*, pp. 279–282, May 1993.

[13] Y. Chen and W. A. Pearlman, "Three-dimensional subband coding of video using the zero-tree method," *Visual Communications and Image Processing '96, Proc. SPIE 2727*, pp. 1302–1309, March 1996.

[14] J. Luo, X. Wang, C. W. Chen, and K. J. Parker, "Volumetric medical image compression with three-dimensional wavelet transform and octave zerotree coding," *Visual Communications and Image Processing'96, Proc. SPIE 2727*, pp. 579–590, March 1996.

[15] J. Y. Tham, S. Ranganath, and A. A. Kassim, "Highly scalable wavelet-based video codec for very low bit-rate environment," *IEEE Journal on Selected Area in Communications*, vol. 16, pp. 12–27, Jan. 1998.

[16] A. Said and W. A. Pearlman, "A new fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, pp. 243–250, June 1996.

[17] B.-J. Kim and W. A. Pearlman, "An embedded wavelet video coder using three-dimensional set partitioning in hierarchical trees (SPIHT)," in *Proc. IEEE Data Compression Conference*, Snowbird, UT, Mar. 1997, IEEE Computer Society, pp. 251–260.

[18] Z. Xiong, K. Ramchandran, and M. T. Orchard, "Wavelet packet image coding using space-frequency quantization," *IEEE Transactions on Image Processing*, vol. 7, pp. 892–898, June 1998.

[19] Z. Xiong, K. Ramchandran, and M. T. Orchard, "Space-frequency quantization for wavelet image coding," *IEEE Transactions on Image Processing*, vol. 6, pp. 677–693, May 1997.

[20] K. Sayood, *Introduction to Data Compression*, Morgan Kaufman Publisher, Inc, 1996, pp. 285–354.

[21] A. Said and W. A. Pearlman, "A new fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE Transactions on Image Processing*, vol. 5, no. 9, pp. 1303–1310, Sep. 1996.

[22] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *IEEE Transactions on Image Processing*, vol. 1, pp. 205–220, April 1992.

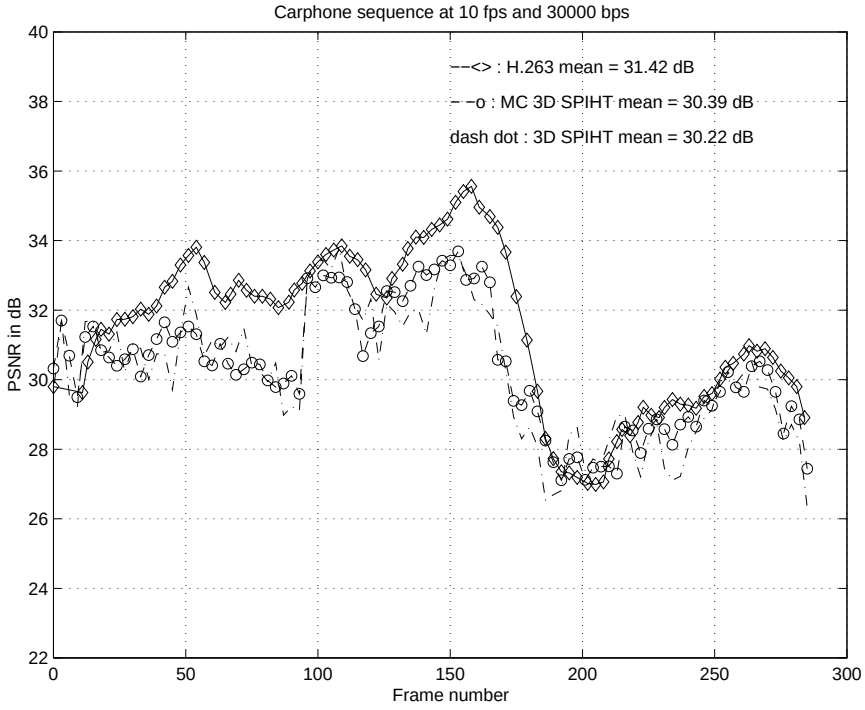
[23] F. Dufaux and F. Moscheni, "Motion estimation techniques for digital tv: a review and a new contribution," *Proceedings of the IEEE*, vol. 83, pp. 858–876, June 1995.

[24] A. M. Tekalp, *Digital Video Processing*, Prentice Hall, Inc, 1995.

[25] I. H. Witten, R. M. Neal, and J. G. Cleary, "Arithmetic coding for data compression," *Communications of the ACM*, vol. 30, no. 6, pp. 520–540, June 1987.

List of Tables

List of Figures



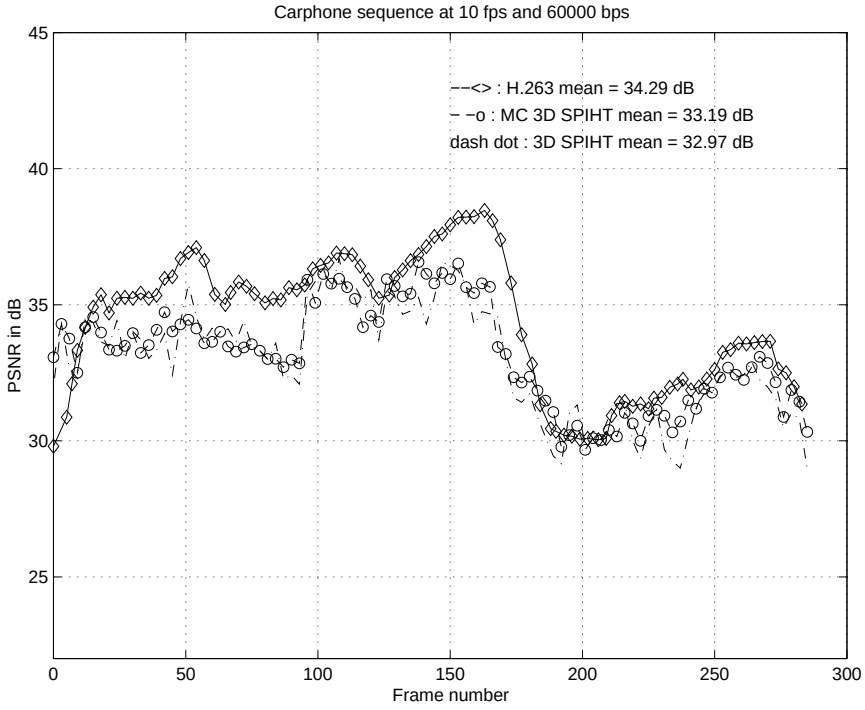


Figure 10: Frame-by-frame luminance PSNR comparison of 3D SPIHT, MC 3D SPIHT, and H.263 at 30 and 60 kbps and 10 f/s with "Carphone" sequence.

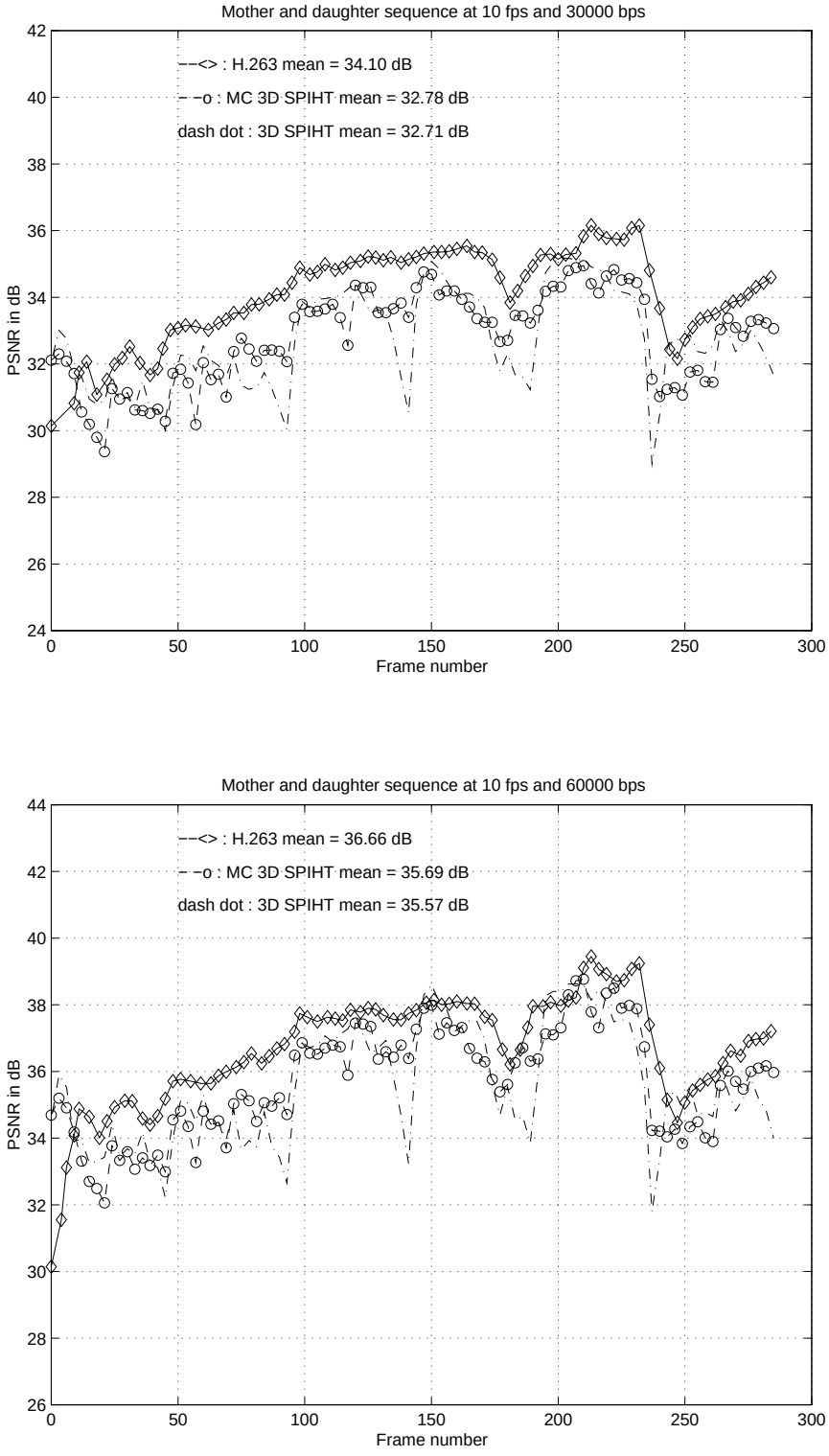


Figure 11: Frame-by-frame luminance PSNR comparison of 3D SPIHT, MC 3D SPIHT, and H.263 at 30 and 60 kbps and 10 f/s with "Mother and Daughter" sequence.

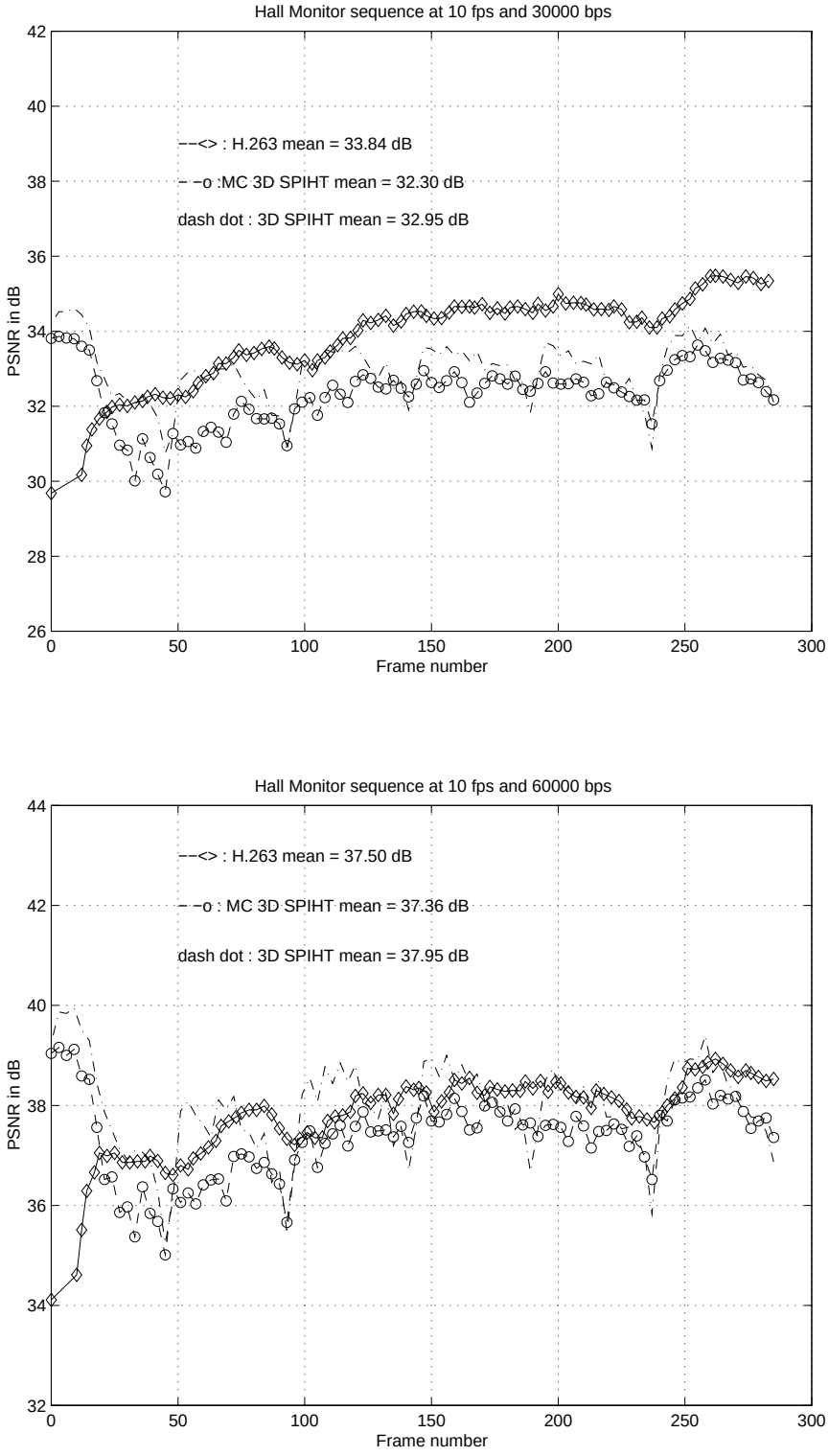


Figure 12: Frame-by-frame luminance PSNR comparison of 3D SPIHT, MC 3D SPIHT, and H.263 at 30 and 60 kbps and 10 f/s with "Hall Monitor" sequence.

Figure 13: The same reconstructed frames at 30 kbps and 10 f/s (a)Top-left: original "Carphone" frame 198 (b)Top-right: H.263 (c)Bottom-left: 3D SPIHT (d)Bottom-right: MC 3D SPIHT.

Figure 14: The same reconstructed frames at 30 kbps and 10 f/s (a)Top-left: original "Mother and Daughter" frame 102 (b)Top-right: H.263 (c)Bottom-left: 3D SPIHT (d)Bottom-right: MC 3D SPIHT.

Figure 15: The same reconstructed frames at 30 kbps and 10 f/s (a)Top-left: original "Hall Monitor" frame 123 (b)Top-right: H.263 (c)Bottom-left: 3D SPIHT (d)Bottom-right: MC 3D SPIHT.

Figure 16: Multiresolutional decoded frame 0 of "Carphone" sequence with the embedded 3D SPIHT video coder (a)Top-left: spatial half resolution (88 x 72 and 10 f/s) (b)Top-right: spatial and temporal half resolution (88×72 and 5 f/s) (c)Bottom-left: temporal half resolution (176×144 and 5 f/s) (d)Bottom-right: full resolution (176×144 and 10 f/s).