Three-dimensional shape-adaptive discrete wavelet transforms for efficient object-based video coding

Ji-Zheng Xu[1], Shipeng Li[2*], and Ya-Qin Zhang[2]

[1] Univ. of Science and Technology of China, Heifei, Anhui, China, 230026
[2] Microsoft Research China, 5F Sigma Center, No. 49 Zhichun Rd., Beijing, China, 100080

ABSTRACT

In this paper, we present an object-based coding scheme using three-dimensional shape-adaptive discrete wavelet transforms (SA-DWT). Rather than straightforward extension of 2D SA-DWT, a novel way to handle the temporal wavelet transform using a motion model is proposed to achieve higher coding efficiency. Corresponding to this transform scheme, we use a 3D entropy coding algorithm called Motion-based Embedded Subband Coding with Optimized Truncation (ESCOT) to code the wavelet coefficients. Results show that ESCOT can achieve comparable coding performance with the state-of-the-art MPEG-4 verification model (VM) 13.0 while having the scalability and flexibility of the bitstream in low bit-rate object-based video coding. And in relative higher bit-rate, our coding approach outperforms MPEG-4 VM 13.0 by about 2.5dB.

Keywords: SA-DWT, object-based video coding, three-dimensional wavelet transforms

1. INTRODUCTION

In recent years, with the explosive growth of the Internet and the great advances in hardware technologies and software developments, a lot of new multimedia applications are emerging rapidly. Although the storage capability of the digital devices and the bandwidth of the networks are increasing constantly and rapidly, video compression still plays an essential role in these applications due to the explosive growth of the multimedia contents both for leisure and at work. Now these applications require not only high compression efficiency, but also more functionalities and flexibility provided by the coding techniques. For example, in order to facilitate contend-based media processing, retrieval and indexing, and to support user interaction, object-based video coding is desired; to achieve the objective of video delivery over heterogeneous networks (e.g., the Internet) and wireless channels, error resilience and bit-rate scalability are required; to make a coded video bitstream to be used by all kinds of digital devices, regardless their computational, display and memory capabilities, the resolution scalability and temporal scalability are needed.

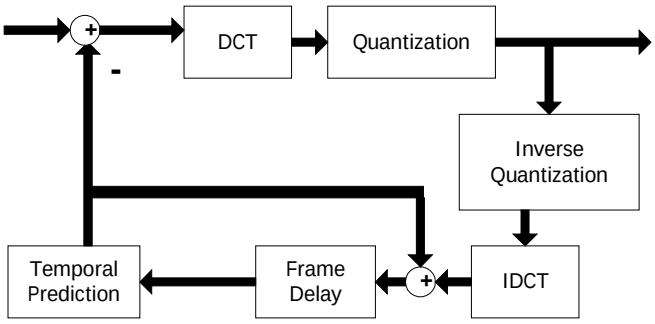

Figure 1: Motion compensated predictive video coding scheme.

The conventional coding system such as H.261/H.263 and MPEG-1/2 do not satisfy the above requirements because of lack of functionalities. The recent MPEG-4 standard adopts object-based video coding scheme so that they can support more applications. However, the scalabilities in MPEG-4 are very limited. Experiments with MPEG-2, MPEG-4, and H.263 using

*Correspondence: spli@microsoft.com; Tel: +86-10-62617711; Fax: +86-10-62555337

scalability modes show that generally the coding efficiency would lose 0.5-1.5dB with every layer, compared with a monolithic (non-layered) coding scheme [1, 2]. These standard coders are all based on a predictive structure showed in Figure 1. It is difficult for these coding schemes to achieve scalability efficiently since there is always a potential drifting problem associate with predictive coding. Currently, there are proposals for MPEG-4 streaming video profile on fine granularity scalable video coding. However, these proposals are limited to provide flexible rate scalability only and the coding efficiency is still much lower than non-layered coding scheme [11]. As an alternative to the predictive approaches in various video coding standards, 3-D wavelet video coding has been investigated recently by several researchers [3, 4, 5]. It turns out that 3-D wavelet transform video coding is competitive with standard motion compensated predictive coding. In [4], Kim and Pearlman reported that their 3D SPIHT algorithm outperformed MPEG-2 by an average of 0.8dB on the standard 30fps SIF sequence *table tennis* and *football* at 760-2530kbps. Moreover, it is easy to achieve bit-rate scalability in 3-D wavelet video coding. For example, the 3D SPIHT coder [4] generates a fully embedded bitstream that can be truncated at any point and still decodable to the best quality available. On the other hand, the straightforward 3D wavelet coding doesn't use motion information that is proven to be very effective in predictive coders in terms of removing temporal redundancy. Although the computationally intensive motion estimation is avoided, this makes the performance of 3D wavelet video coding very sensitive to the motion. And as we will show later, in low bit-rate coding, most high-pass coefficients of the video are discarded so that the 3-D wavelet video coder actually plays a low-pass filter role. Without motion information, motion blur will occur because of the temporal averaging effect of several frames. Moreover, most 3-D wavelet video coders currently do not support object-based functionality, which is a very desirable feature in the next generation multimedia applications.

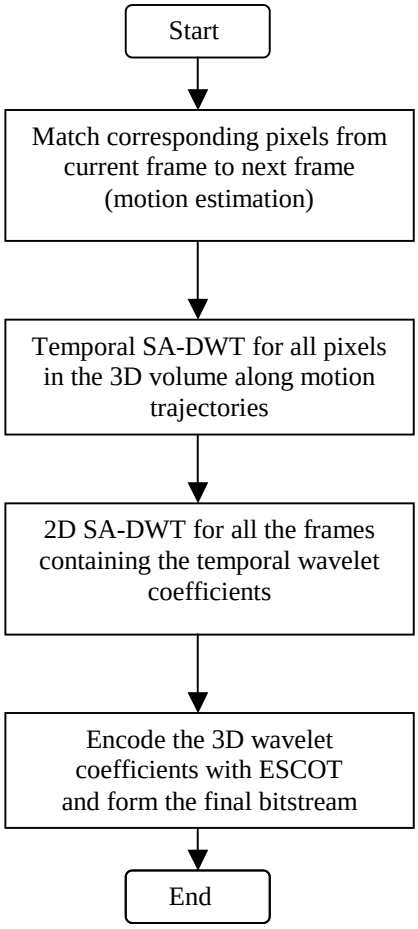

Figure 2: Flowchart of a 3D SA-DWT based video coding scheme.

In this paper, we propose a new 3-D transform and coding scheme that is suitable for object-based video coding. Our scheme is based on shape-adaptive discrete wavelet transform (SA-DWT) technique proposed by Li, et al [6,7]. SA-DWT is an efficient wavelet transform algorithm for arbitrarily shaped visual objects. The number of coefficients after SA-DWT is identical to the number of pixels in the original arbitrarily shaped visual object. The spatial correlation, locality properties of wavelet transforms, and self-similarity across sub-bands are well preserved in the SA-DWT. Moreover, for a rectangular region, the SA-DWT becomes identical to the conventional wavelet transforms. And the coding efficiency of SA-DWT is much better than that of SA-DCT for arbitrarily shaped image objects. All these features make SA-DWT very suitable for object-based video coding. Instead of using a straightforward method similar to [4] to extend the original 2D SA-DWT to 3D SA-DWT, we use motion information for the SA-DWT in the temporal direction. First, the motion trajectory of each pixel inside the video object is traced using some motion estimation algorithm. Then a 1-D SA-DWT in the temporal direction is performed along each of the motion trajectory to get temporally decomposed frames. Finally, a 2-D SA-DWT is applied within each of the temporally decomposed frame. After 3D SA-DWT transforms, we code the coefficients with an entropy coder called *Motion-based Embedded Sub-band Coding with Optimized Truncation Using (ESCOT)*, which takes advantage of the motion information and encodes each sub-band independently. At the end, a bitstream is formed in the way that meets the scalability requirements. The new 3D SA-DWT solves two problems:

1. It can handle arbitrarily shaped video objects while having flexible bit-rate, spatial and temporal scalabilities as in most wavelet-based coding schemes.
2. It tracks the video object motion and always performs the wavelet transform among corresponding pixels for that object while keeping the spatial correlation within a frame. So it will efficiently decompose the video-object sequence and more efficient compression is feasible.

Figure 2 gives the flowchart of the proposed 3D SA-DWT with ESCOT coder.

The paper is organized as follows. Section 2 presents the new 3D shape-adaptive wavelet transforms using motion trajectories. Motion information is used in the temporal direction to obtain more efficient wavelet decomposition and to reduce the motion blurring artifacts for low bit-rate coding. A JPEG-2000 like entropy coder is proposed in section 3. The proposed entropy coder generates an independent embedded bitstream for each sub-band and the rate-distortion curve is used to optimize the bit-allocation among sub-bands. Motion information again is used to form the contexts for arithmetic coding. In section 3.2, experimental results and comparisons with MPEG-4 object-based video coding scheme are given. Results show that our coder can achieve comparable coding performance with the state-of the art MPEG-4 standard while having the scalability and flexibility of the bitstream in low bit-rate object-based video coding. And in relative higher bit-rate, our coding approach yields much better results than MPEG-4 VM by about 2.5dB. Section 5 concludes this paper.

2. THREE-DIMENSIONAL SHAPE-ADAPTIVE DISCRETE WAVELET TRANSFORMS USING MOTION TRAJECTORIES

The basic idea of the new 3D-DWT scheme is that the temporal decomposition is always performed along the motion trajectories. Figure 3 illustrates the concept of such a temporal transform. The corresponding pixels (1D array) obtained from motion estimation or other matching schemes are first aligned in the temporal direction. Then a 1D SA-DWT will be applied to this 1D array to obtain a 1D coefficient array. The coefficients in this 1D array will then be redistributed to their corresponding spatial position in each frame. Since a video object normally is not limited to 2D translation movement, and it may move in/out or zoom in/out of a video scene any time, we have to consider the following four cases: (Figure 4)

a. Continuing pixels: This is the normal case. These pixels can find one-to-one corresponding pixels between two frames. In this case, the temporal 1D array will be extended to include the corresponding pixel from the next frame. (Figure 4(a))
b. Terminating pixels: These pixels cannot find any corresponding pixels in the next frame. Thus the 1D temporal array will be ended at this terminating pixel. (Figure 4(b))
c. Emerging pixels: These pixels are not corresponding pixels for any pixels in the previous frame. In this case, the emerging pixel will start a new 1D-array. (Figure 4(c))
d. Colliding pixels: These pixels are the corresponding pixels of more than one pixel in the previous frame. In this case, the colliding pixel will be assigned to only one of the corresponding pixels in the previous frame, and all the other corresponding pixels are marked as terminating pixels. (Figure 4(d))

Now let's formulate the procedure of the 3D SA-DWT transform for a video sequence. Given a group of pictures P_i, for i=0, …, N-1, assume the motion of each pixel with reference to the next picture has been obtained using a motion estimation

algorithm (for example, block matching algorithm). It should be noted that after motion estimation, some of the pixels in the current frame may not find any correspondence in the next frame (terminating pixels) and it is also possible some pixels in the current frame may not have any correspondence in the previous frame (emerging pixels). It is also true that more than one pixels in the current frame may have the same corresponding pixel in the next frame (colliding pixel). Please note that all pixels in the last frame P_{N-1} are terminating pixels. We also assume odd-symmetric bi-orthogonal wavelet filters are used. Other types of wavelet filters can also be used for the 3D SA-DWT according to the arbitrary-length wavelet transform described in [6].

The 3D SA-DWT can be described as follows,

1. Initialize:
   Set i=0, and mark all pixels within object boundary in all these N frames as UNSCANNED;
2. Form threads for temporal SA-DWT:
   2.1. For every pixel $p_i(x_i,y_i)$ within object boundary in frame P_i ,
      2.1.1. If it is marked as UNSCANNED, then it becomes the first pixel of a new temporal thread. Let j=i;
      2.1.2. If $p_j(x_j,y_j)$ is a terminating pixel then $p_j(x_j,y_j)$ is the last pixel of this temporal thread, go to 2.14. If $p_j(x_j,y_j)$ is not a terminating pixel and its corresponding pixel $p_{j+1}(x_{j+1}, y_{j+1})$ in frame P_{j+1} is marked as UNSCANNED, where $(x_{j+1}, y_{j+1})= (x+mv_x, y+mv_y)$ and (mv_x, mv_y) is the motion vector from pixel $p_j(x_j,y_j)$ in frame P_j to its corresponding pixel $p_{j+1}(x_{j+1}, y_{j+1})$ in frame P_{j+1} , then $p_{j+1}(x_{j+1}, y_{j+1})$ is added as the next pixel in this temporal thread and is marked as SCANNED. If the corresponding pixel in the next frame is marked as SCANNED, then this temporal thread terminates at this pixel.
      2.1.3. j=j+1, if j<N go to 2.1.2
      2.1.4. Perform 1D arbitrary length wavelet filtering (refer to [6]) for this temporal thread:
      $p_k(x_k,y_k)$, k=i,...,j-1
      and obtain a transformed low-pass thread:
      $L_k(x_k,y_k)$, k=i,...,j-1;
      and high-pass thread
      $H_k(x_k,y_k)$, k=i,...,j-1.
      These low-pass coefficients $L_k(x_k,y_k)$ are placed into the low-pass frame k at position (x_k,y_k). And these high-pass coefficients $H_k(x_k,y_k)$ are placed into the high-pass frame k at position (x_k,y_k).
      Isolated pixels can be scaled by srqt(2) and put back into their corresponding positions in both low-pass and high-pass frames.
3. i=i+1, if i<N goto 2.1
4. Subsample low-pass frames at even frames to obtain temporal low-pass frames and subsample high-pass frames at odd frames to obtain temporal high-pass frames.
5. If more temporal decomposition levels are needed, repeat step 1-4 for the low-pass frames. Note that the motion vectors from frame P_k to P_{k+2} can be obtained by adding the motion vectors from P_k to P_{k+1} and P_{k+1} to P_{k+2} .
6. Perform spatial 2D SA-DWT transforms according to their spatial shapes for every temporally transformed frame.

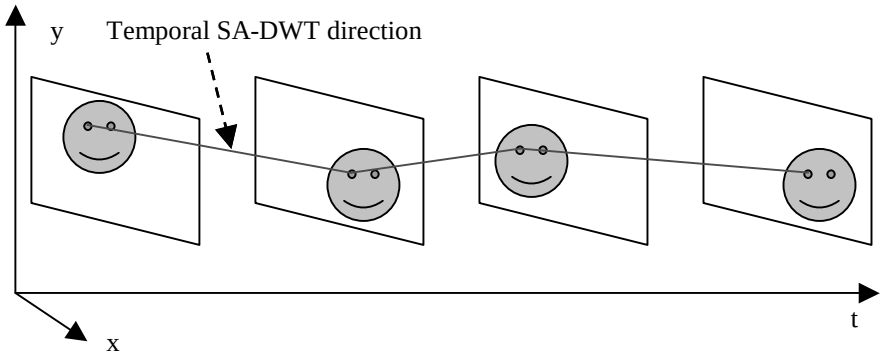

Figure 3: An illustration of a temporal motion trajectory for the 3D SA-DWT in a video object sequence.

To obtain the motion vector of each pixel in the object, we just use a block-based motion estimation algorithm. For each block in frame i that contains pixels from the video object, we search the best-matched block in frame i+1 and estimate the motion vector for that block. Note that for 3D SA-DWT purpose, the motion vector of every pixel within a block is set to the same as that of the block. We can search and represent the motion information using any technologies developed for the standard motion compensated predictive coding system, like MPEG-4, which are very mature now.

3. MOTION-BASED EMBEDDED SUBBAND CODING WITH OPTIMIZED TRUNCATION

3.1 Entropy Coding

After transformation, the wavelet coefficients with compact energy can be coded with any 3-D wavelet coding algorithms, e. g., 3-D SPIHT[8]. In this paper, we propose to use a powerful and flexible entropy coding algorithm for the 3D wavelet coefficients. We called it *Motion-based Embedded Sub-band Coding with Optimized Truncation (ESCOT)*. The entropy coding technique used in ESCOT is very similar to the EBCOT algorithm adopted in JPEG-2000 [9], which demonstrates high compression efficiency and other functionalities. However, in ESCOT coding scheme, we design a different coding structure and use a new set of coding contexts that make it very suitable for scalable video object compression and the proposed 3D SA-DWT algorithm.

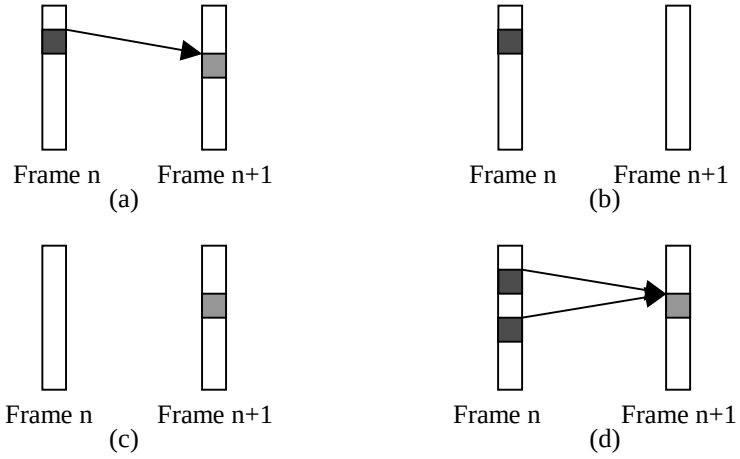

Figure 4: Scenarios in Temporal SA-DWT: (a) Normal continuing pixels; (b) Terminating pixels; (c) Emerging pixels; (d) Colliding pixels

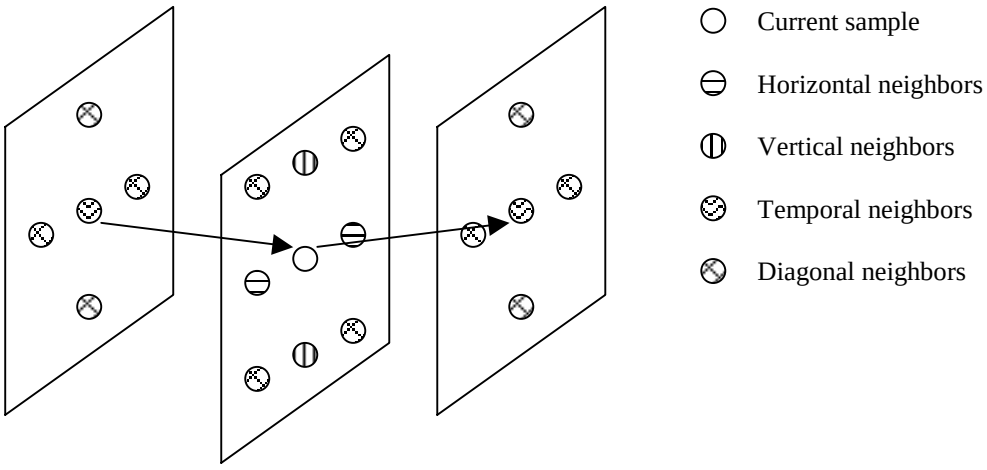

Figure 5: Immediate neighbors of a sample

In ESCOT algorithm, each sub-band is coded independently. The advantage of doing so is that each sub-band can be decoded independently to achieve flexible spatial and temporal scalabilities. The user can mix arbitrary number of spatio-temporal sub-bands in any order to obtain the desired spatial and temporal resolution. Another advantage is that rate-distortion optimization can be done among sub-bands, which may improve compression efficiency.

In order to reduce the number of contexts used in the arithmetic coding, we take advantage of the symmetric property of wavelet sub-bands. For example, the LLH sub-band, HLL sub-band, LHL sub-band can share the same contexts and coding scheme if we transpose the HLL and LHL sub-bands to have the same orientation as LLH sub-band before encoding.

After sub-band transposition, there are only four classes of sub-bands, LLL, LLH, LHH and HHH sub-bands. And for each sub-band, the quantized coefficients are coded bit-plane by bit-plane. In a given bit-plane, different primitives are used to code a sample's information of this bit-plane. There are three coding primitives: *Zero Coding (ZC)*, *Sign Coding (SC)* and *Magnitude Refinement (MR)*. The ZC and SC primitives are used to code new information for a single sample that is not yet significant in the current bit-plane. And MR is used to code new information of a sample that is already significant. Let $\sigma[i,j,k]$ be a binary-valued state variable, which denotes the significance of the sample at position [i,j,k] , i.e. the position in the transposed sub-band. $\sigma[i,j,k]$ is initialized to 0 and toggled to 1 when the corresponding sample's first non-zero bit-plane value is encoded. We also define $\chi[i,j,k]$ as the sign of that sample, which is 0 when the sample is positive and 1 when the sample is negative.

Zero Coding: When a sample is not yet significant in the previous bit-plane, i.e. $\sigma[i,j,k]=0$, this primitive operation is used to code the new information about the sample. It tells whether the sample become significant or not in the current bit-plane. The ZC operation uses the information of the current sample's neighbors as the context to code the current sample's significance information. In detail, we consider three categories of a sample's neighbors (see Figure 5):

- Immediate horizontal neighbors.
  We denote the number of these neighbors that are significant by h, $0<h<2$.
- Immediate vertical neighbors.
  We denote the number of these neighbors that are significant by v, $0<v<2$.
- Immediate temporal neighbors.
  We denote the number of these neighbors that are significant by a, $0<a<2$.
- Immediate diagonal neighbors.
  We denote the number of these neighbors that are significant by d, $0<d<12$.

Note that the temporal neighbors of a sample are not defined as the samples that have the same spatial positions in the previous and next frames. Here, sample x and sample y are temporal neighbor means that x and y are in the same motion trajectory and in the previous and next frames, respectively. In other words, the temporal neighbors are linked by the motion vectors. Since there is more correlation along the motion direction, we may improve the coding efficiency. The motion vector for a sample in a high level sub-band can be derived by the motion vectors in the low level sub-bands. In detail, in spatial decomposition, motion vectors are down-sampled when we down-sample the wavelet coefficients. And because the range and resolution of the sub-bands are half of the original sub-bands, the magnitude of the motion vectors should be divided by 2 to represent the motion of the samples in that sub-band. If a sample have no correspondent motion vector, we just assign a zero motion vector to it.
The context assignment map for Zero Coding is listed in Table 1. If the conditions of two or more rows are satisfied in the same time, the lowest-numbered context is selected. An adaptive context-based arithmetic coder is used to code the significance symbols of Zero Coding.

Sign Coding: Once a sample becomes significant in the current bit-plane, Sign Coding operation is called to code the sign of the sample. Sign Coding also utilizes an adaptive context-based arithmetic coder to compress the sign symbols. Similarly, we define three quantities, h, v and a, by

$$h=\min\{1, \max\{-1, \sigma[i-1,j,k].(1-2\chi[i-1,j,k])+ \sigma[i+1,j,k].(1-2\chi[i+1,j,k])\}\}$$
$$v=\min\{1, \max\{-1, \sigma[i,j-1,k].(1-2\chi[i,j-1,k])+ \sigma[i,j+1,k].(1-2\chi[i,j+1,k])\}\}$$
$$a=\min\{1, \max\{-1, \sigma[i,j,k-1].(1-2\chi[i,j,k-1])+ \sigma[i,j,k+1].(1-2\chi[i,j,k+1])\}\}$$

The context assignments are showed in Table 2. $\hat{\chi}$ means the sign symbol prediction in a given context. And the symbol sent to the arithmetic coder is $\hat{\chi}$ XOR χ.

Magnitude Refinement: Magnitude Refinement is used to code the new information of a sample that has already become significant in the previous bit-plane. This operation has three contexts. In detail, if MR operation is not yet used in this sample, the context is 0. If MR has been used in the sample and the sample has at least one significant neighbor by now, then the context is 1, otherwise the context is 2.

Using these three primitive operations, a sub-band coefficient can be coded without loss. The coding process is as follows. For each bit-plane, the coding procedure consists of three consecutive passes. Each pass processes a "fractional bit-plane". The reason for introducing multiple coding passes is to ensure that each sub-band has a finely embedded bitstream. And by separating zero coding and magnitude refinement operation into different passes, it is convenient to design efficient and meaningful context assignment. In each pass, the scanning order is along i-direction firstly, then j-direction and k-direction lastly. The three passes are described below.

Significant propagation pass: In this pass, the samples that are not yet significant but have *preferred neighborhood* are processed. We call that a sample has *preferred neighborhood* if and only if that the sample has at least a significant immediate diagonal neighbor, for HHH sub-band; or at least a significant horizontal, or vertical, or temporal neighbor, for other sub-bands. For the sample that satisfies these conditions, we apply the ZC primitive to code the symbol of current bit-plane of this sample. If the sample becomes significant in the current bit-plane, then SC primitive is used to code the sign.

Magnitude Refinement pass: We code the samples that are already significant in this pass. The symbols of these samples in the current bit-plane are coded by MR primitive.

Normalization pass: During this pass, those samples that are not yet coded in the previous two passes are coded. These samples are insignificant, so ZC and SC primitives are applied in this pass.

Table 1: Context assignment map for Zero Coding

LLL and LLH sub-band				
h	v	a	d	Context
2	x	x	x	0
1	≥1	x	x	0
1	0	≥1	x	1
1	0	0	x	2
0	2	0	x	3
0	1	0	x	4
0	0	≥1	x	5
0	0	0	3	6
0	0	0	2	7
0	0	0	1	8
0	0	0	0	9

LHH sub-band			
h	v+a	d	Context
2	x	x	0
1	≥3	x	0
1	≥1	≥4	1
1	≥1	x	2
1	0	≥4	3
1	0	x	4
0	≥3	x	5
0	≥1	≥4	6
0	≥1	x	7
0	0	≥4	8
0	0	x	9

HHH sub-band		
d	h+v+a	context
≥6	x	0
≥4	≥3	1
≥4	x	2
≥2	≥4	3
≥2	≥2	4
≥2	x	5
≥0	≥4	6
≥0	≥2	7
≥0	1	8
≥0	0	9

Table 2: Context assignment and sign prediction map for Sign Coding

h=-1			
v	a	$\hat{\chi}$	Context
-1	-1	0	0
-1	0	0	1
-1	1	0	2
0	-1	0	3
0	0	0	4
0	1	0	5
1	-1	0	6
1	0	0	7
1	1	0	8

H=0			
v	a	$\hat{\chi}$	Context
-1	-1	0	9
-1	0	0	10
-1	1	0	11
0	-1	0	12
0	0	0	13
0	1	1	12
1	-1	1	11
1	0	1	10
1	1	1	9

h=1			
v	a	$\hat{\chi}$	Context
-1	-1	1	8
-1	0	1	7
-1	1	1	6
0	-1	1	5
0	0	1	4
0	1	1	3
1	-1	1	2
1	0	1	1
1	1	1	0

3.2 Bitstream Construction and Scalability

In the previous stage of sub-band entropy coding, a bitstream is formed for each sub-band. In the current stage, the bitstream of each sub-band will be truncated and multiplexed to construct a final bitstream. The question is how to determine where a bitstream should be truncated and how to multiplex these bitstreams in order to achieve more functionalities, for example, PSNR scalability and resolution scalability. The following describes the proposed optimal bitstream truncation and construction procedure.

1. *Bitstream truncation with rate distortion optimization.*

   Given a specific bit-rate $R_{\max}$, our objective is to construct a bitstream that satisfies the bit-rate constraint and with minimal distortion. As in EBCOT algorithm, the end of each entropy coding pass is a candidate truncation point. The value in the R-D curve at each candidate truncation point can be obtained since we calculate the bit length and the distortion reduction at the end of each pass. Therefore, we can get an approximate R-D curve and find the convex hull of the R-D curve, and truncation can only take place at the candidate truncation points that are at the convex hull of R-D curve. The reason to do so is to guarantee that at every truncation point the bitstream is rate-distortion optimized. Given a rate-distortion slope threshold λ, one can find those truncation points of a sub-band where the rate-distortion slope is greater than λ. To satisfy the bit-rate constraint and to make the distortion minimal, the smallest value of λ such that $R_\lambda \le R_{\max}$ is chosen. The algorithm to find such a threshold can be found in [9].

2. *Multi-layer bitstream construction.*

   To achieve quality scalability, a multi-layer bitstream is formed and each layer includes a quality level's data. To make a N-layer bitstream, we first select $\lambda_1 > \lambda_2 > \mathrm{L} > \lambda_N$ which satisfy $R_{\lambda_N} \le R_{\max}$. So with every threshold, we can find a truncation point and obtain a layer of bitstream from each sub-band. The corresponding layers from all the sub-bands constitute the layers of the final bitstream. According to the available bandwidth and the computation capability, the decoder can select first few layers to be decoded. The fractional bit-plane coding ensures that the bitstream is embedded with fine granularity.

Since we code each sub-band independently, the bitstream of each sub-band is separable. The decoder can easily extract only a few sub-bands and decode only these sub-bands, so the implementation of resolution scalability and temporal scalability is natural and easy. According to the requirement of applications, the final bitstream can be constructed in an order to meet the requirement. The preceding multi-layer bitstream construction method enables the bitstream with quality scalability. To obtain resolution or temporal (frame rate) scalability, the bitstream can be assembled sub-band by sub-band, with the lower resolution or lower temporal sub-band in the beginning. Moreover, the final bitstream can be rearranged to achieve other scalability easily because the offset and the length of each layer of bitstream from each sub-band are coded in the header of the bitstream. This property makes the final bitstream very flexible to be re-used for all sorts of applications without re-encoding again.

4. EXPERIMENTAL RESULTS

The proposed 3-D SA-DWT with ESCOT video object coder has been test using MPEG-4 test sequences. The results of two pre-segmented test sequences: Akiyo and Coast_guard, (QCIF format) are presented. For Akiyo sequence, the foreground lady is the object to be coded. For Coast_guard sequence, the boat (Object 1) is the object to be coded. The coding results of our approach are compared with that of MPEG-4 Verification Model version 13.0[10]. The shape and motion information coding schemes used are the same as in MPEG-4 VM. In 3-D SADWT, sequences are first split into GOPs each with 32 frames. Then we apply a 4-level temporal SA-DWT decomposition followed by 3-level spatial Mallat SA-DWT decomposition. The coefficients are coded using the proposed coding scheme described in this paper. For MPEG-4 VM, object-based coding option is chosen; all frames are coded with P-mode except the first I-frame. Since it is difficult to specify the exact texture bit-rate in MPEG-4 VM, MPEG-4 VM is first run to get the bit-rates at different quality levels. Then the proposed coder is used to get the exactly matching bit-rate. The results are shown in Table 3. 160 frames are coded for both sequences. To illustrate the effect of motion information, the results of 3D SA-DWT coder without motion information is also presented.

In Table 3, for Akiyo sequence, due to the slow motion nature of the sequence, the coding performance of the 3D SA-DWT coders with motion information and without motion information are almost the same. But for object 1 in Coast_guard sequence, the gain using motion information is obvious.

From this table, we can also see that the performance of the proposed coding algorithm is comparable with MPEG-4 VM at low bit-rates. However, at relatively higher bit-rate, the proposed coder shows significant improvement in coding efficiency over MPEG-4 VM. It is important to note that we achieve the performance while making the bitstream scalable (rate, temporal, and spatial) and flexible. Different rates of bitstream can be extracted from a single one. These functionalites are sometimes more important than the coding efficiency.

Table 3: Comparison of 3D SA-DWT coder with MPEG-4 VM in terms of average PSNR (dB) at different bit rates.

	Akiyo(foreground)			Coast_guard(Object 1)		
Texture rate (bpp)	0.0233	0.0648	0.2	0.1522	0.4431	1.219
MPEG-4 VM	28.01	31.36	35.80	24.23	28.18	33.36
3-D SADWT coder	27.09	31.92	38.23	23.57	28.32	35.06
3-D SADWT without motion information	27.45	31.85	38.12	21.79	25.78	31.78

5. CONCLUSIONS

In this paper, we propose a novel transform scheme for object-based video coding -- 3-D SA-DWT and an efficient entropy coding algorithm based on this transform — ESCOT. Unlike other 3D wavelet coding schemes, motion information is used for both 3D shape adaptive wavelet transforms and the entropy coding. Experimental results show that the proposed coding system has comparable coding efficiency with MPEG4 VM (at high bit-rate much better than MPEG-4) while having more functionalities and flexibilities, such as, flexible rate scalability, spatial scalability, and temporal scalability. These functionalities and object-based coding scheme make the proposed coding system very suitable for numerous applications like video streaming, interactive multimedia applications.

References

1. B. G. Haskell, A. Puri and A. N. Netravali, *Digital Video: An Introduction to MPEG-2.* Chapman & Hall, New York, 1997.
2. L. Yang, F. C. M. Martins, and T. R. Gardos, "Improving H.263+ scalability performance for very low bit rate applications," In *Proc. Visual Communications and Image Processing*, San Jose, CA, January 1999. SPIE.
3. D. Taubman and A. Zakhor, "Multirate 3-D subband coding of video," *IEEE Trans. Image Processing*, 3(5): 572-589, September 1994.
4. B.-J. Kim and W. A. Pearlman, "An embedded wavelet video coder using three-dimensional set partitioning hierarchical trees(SPIHT)," In *Proc. Data Compression Conference*, pages 251-260, Snowbird, UT, March 1997. IEEE Computer Society.
5. A. Wang, Z. Xiong, P. A. Chou, and S. Mehrotra, "Three Dimensional Wavelet Coding of Video with Global Motion Compensation," In *Proc. Data Compression Conference*, 1999. SPIE.
6. S. Li and W. Li, "Shape Adpative Discrete Wavelet Transforms for Arbitrarily-shaped Visual Object Coding", to be appeared in *IEEE Trans. Circuits and Systems for Video Technology*.
7. S. Li, W. Li, H. Sun, and Z. Wu, " Shape adaptive wavelet coding," In *Proceedings of the IEEE International Symposium on Circuits and Systems.* Vol. 5, pp. 281-284, ISCAS'98, Monterey, California, May 1998.
8. W. A. Pearlman, B.-J. Kim, and Z. Xiong, "Embedded video subband coding with 3D SPIHT," in *Wavelet Image and Video Compression*, P. Topiwala, Ed. Kluwer, 1998.
9. D. Taubman (editor), "JPEG2000 Verification Model: Version VM4.1," ISO/IEC JTC 1/SC 29/WG1 N1286.
10. "MPEG-4 Video Verification Model version 13.0", ISO/IEC JTC 1/SC29/WG11 N2687, March, 1999.
11. S. Li, F. Wu and Y.-Q. Zhang, "Study of a new approach to improve FGS video coding efficiency," MPEG-99/m5583, Maui, Dec., 1999.