Shape Adaptive Discrete Wavelet Transforms for Arbitrarily Shaped Visual Object Coding

Shipeng Li

Weiping Li

Lehigh University, Department of EECS
19 Memorial Drive, Bethlehem, PA 18015

Abstract

This paper presents a shape adaptive wavelet coding technique for coding arbitrarily shaped still texture. This technique includes shape adaptive discrete wavelet transforms (SA-DWT) and extentions of zerotree entropy (ZTE) coding and embedded zerotree wavelet (EZW) coding. Shape adaptive wavelet coding is needed for efficiently coding arbitrarily shaped visual objects, which is essential for object-oriented multimedia applications. The challenge is to achieve high coding efficiency while satisfying the functionality of representing arbitrarily shaped visual texture. One of the features of the SA-DWT is that the number of coefficients after SA-DWT is identical to the number of pixels in the original arbitrarily shaped visual object. Another feature of the SA-DWT is that the spatial correlation, locality properties of wavelet transforms, and self-similarity across subbands are well preserved in the SA-DWT. Also, for a rectangular region, the SA-DWT becomes identical to the conventional wavelet transforms. For the same reason, the extentions of ZTE and EZW to coding arbitrarily shaped visual objects carefully treat "don't-care" nodes in the wavelet trees. Comparison of shape adaptive wavelet coding with other coding schemes for arbitrarily shaped visual objects shows that shape adaptive wavelet coding always achieves better coding efficiency than other schemes. One implementation of the shape adaptive wavelet coding technique has been included into the new multimedia coding standard MPEG-4 for coding arbitrarily shaped still texture. Software implementation is also available.

KEYWORDS: object-oriented image coding, arbitrarily shaped object coding, shape adaptive discrete wavelet transform (SA-DWT), extension of zerotree entropy coding (ZTE), extension of embedded zerotree wavelet coding (EZW), MPEG-4

1 INTRODUCTION

The functionality of making visual objects available in the compressed form has become a very important feature in the next generation visual coding standards such as MPEG-4 [1, 2], since it provides great flexibility for manipulating visual objects in multimedia applications and could potentially improve visual quality in very low bit rate coding. There are two parts in coding an arbitrarily shaped visual object. The first part is to code the shape of the visual object and the second part is to code the texture of the visual object (pixels inside the object region). This paper is on the texture coding part. More specifically, this paper is on texture coding of still objects (not video objects).

Since there have been a considerable amount of research efforts on coding rectangular-shaped images and video, such as discrete cosine transform (DCT) coding and wavelet transform coding, it would be straightforward to first find the bounding box of the arbitrarily shaped visual object, then pad values into the pixel positions outside object, and code pixels inside the object and the padded pixels in the rectangular bounding box together using the conventional methods. However, this approach would not be efficient.

Therefore, there have been continuous efforts in developing new signal processing and coding techniques for arbitrarily shaped regions [7, 8, 9, 10, 11, 14, 15]. Among them, the shape-adaptive DCT (SA-DCT) is the most popular one. The SA-DCT generates the same number of DCT coefficients as the number of pixels in an arbitrarily shaped 8×8 image block. The SA-DCT algorithm achieves a transform efficiency similar to the shape-adaptive Gilge DCT [8, 10], but is implemented with a lower complexity. For standard rectangular image blocks containing 8×8 pixels, the SA-DCT becomes identical to the standard 8×8 DCT. Since the SA-DCT always flushes samples in an arbitrarily shaped block to a certain edge of a rectangular bounding block before performing row or column DCT transforms, some spatial correlation may be lost. It is not efficient to perform column DCT transforms on a set of coefficients that are from different frequency bands after the row DCT transforms [10, 12, 13].

There have been some proposals for coding arbitrarily shaped image objects using the wavelet transforms. One of them is based on padding techniques [16, 17]. The conventional wavelet transform is performed in the padded rectangular region, and coding coefficient selection (CCS) and coding coefficient insertion (CCI) techniques are used to improve coding efficiency. Another one is macroblock-region based wavelet coding [18]. This technique first pads undefined region with zeros and then apply the wavelet transforms on the padded rectangular region. In order to use ZeroTree Coding (ZTC), an arbitrary shape is quantized into wavelet block boundary. In this scheme, the wavelet transform inevitably blurs the edges of the arbitrarily shaped objects and results in more coefficients to be coded than the number of pixels contained in the object. These two methods try to make improvement on the straightforward padding methods and do not solve the fundamental problem of how to efficiently perform wavelet transform directly to an arbitrarily shaped region and efficiently code just enough wavelet coefficients. Consequently, the performance of these techniques is not competitive.

In [22, 23], Egger et al. proposed a wavelet transform scheme that adapts to an arbitrary shape similar to the idea of the SA-DCT. In this scheme, before a 1-D wavelet transform, all the pixels in a row (or column) are first flushed to the right boundary of a bounding box, and a wavelet transform is then applied on all the valid pixels in that row (or column). After the wavelet transform, the wavelet coefficients are then redistributed back to the proper spatial locations to prepare for the wavelet transform in the other direction. By taking advantage of the locality property of wavelet transform, this scheme distinguishes itself from the concept of SA-DCT by redistribution of wavelet coefficients, which preserves the spatial relations of the wavelet coefficients and thus makes the wavelet decompositions in the second direction more efficient than the SA-DCT. However, this scheme has its problems too. First, the flushing operation would merge unrelated image segments in the same row (or column) into one segment, which could cause significant high frequency components at the merged boundaries. Secondly, redistribution of the wavelet coefficients obtained from such a merged segment can not guarantee that the distributed coefficients have exactly the same phase with each other. This degrades the efficiency of the wavelet decompositions in the second direction. Thirdly, the scheme handles an odd-length segment without taking advantage of the properties of wavelet filters. The wavelet transforms are only applied to the even-length segments and the last pixel of an odd-length segment is always independently scaled and put into the lowpass bands, which could cause some discontinuities in the wavelet decompositions in the second direction.

A novel shape adaptive discrete wavelet transform (SA-DWT) for arbitrarily shaped object coding was proposed by Li, *et al* [19, 20, 21, 24]. This SA-DWT scheme can be directly applied to the arbitrarily shaped region. The SA-DWT transforms the samples in the arbitrarily shaped region into the same number of coefficients in the subband domain while keeping the spatial correlation, locality, and self-similarity across subbands. Approaches applying the SA-DWT scheme to Embedded Zerotree Wavelet (EZW) Coding and Vector Wavelet Coding (VWC) schemes were also proposed [6].

This paper provides a comprehensive description of the shape adaptive wavelet coding technique. In Section 2, wavelet transforms specially designed for arbitrarily shaped regions are described. Methods of using different wavelet filters, such as orthogonal filters, even symmetric biorthogonal filters, and odd symmetric biorthogonal filters in the SA-DWT are presented. The number of coefficients after SA-DWT is identical to the number of pixels contained in the arbitrarily shaped image region. The spatial correlation and other wavelet transform properties, such as locality and self-similarity across subbands, are well preserved in the SA-DWT. For a rectangular region, the SA-DWT becomes identical to a conventional wavelet transform.

Keeping the number of wavelet coefficients the same as the number of pixels is a necessary condition for an efficient coding method. On the other hand, lossy shape coding may reduce the number of pixels to be coded at the expense of shape quality. As mentioned previously, there are two aspects of coding an arbitrarily shaped visual object. First is to code the shape. This may be a lossless or a lossy coding method.

If a lossy shape coding method is used, the number of pixels within the reconstructed shape is usually less than that within the original shape. Because shape coding is always performed before texture coding, the reconstructed shape is always used for the SA-DWT. Therefore, as far as the SA-DWT is concerned, the number of pixels in the image domain is referred to as that within the reconstructed shape. In other words, the tradeoff between the number of pixels to be coded and the quality of shape is made at the time of shape coding. The SA-DWT just maintains the number of wavelet coefficients to be the same as the number of pixels to be coded.

In addition to the transform part, another challenge is how to extend zerotree coding for arbitrarily shaped regions. Section 3 addresses this issue and presents an efficient method for applying zerotree coding to partial wavelet trees that are formed by the SA-DWT of an arbitrarily shaped object. Results of shape adaptive wavelet coding and comparisons of different options and with other coding schemes are reported in Section 4. It is shown that shape adaptive wavelet coding always achieves better coding efficiency than other schemes. Section 5 concludes this paper.

2 SHAPE ADAPTIVE DISCRETE WAVELET TRANSFORMS

There are two components in the shape adaptive discrete wavelet transform (SA-DWT). One is a way to handle wavelet transforms for arbitrary length image segments. The other one is a subsampling method for arbitrary length image segments at arbitrary locations. The SA-DWT allows odd length or small length image segments to be decomposed into the transform domain in a similar manner to the even and long length segments, while maintaining the number of coefficients in the transform domain identical to the number of pixels in the image domain. The scale of the transform domain coefficients within each subband is the same to avoid sharp changes in subbands.

A proper subsampling method is important for the SA-DWT too. One consideration is that it should preserve the spatial correlation and self-similarity property of wavelet transforms so that 2-D (horizontal and vertical directions) separable wavelet decompositions and pyramid wavelet decompositions can still be applied to the arbitrarily shaped image region without loss of spatial correlation. Another consideration is the effect of the subsampling strategy on the efficiency of zerotree coding.

2.1 Arbitrary Length Wavelet Decomposition

Depending on the wavelet filters used, the methods to handle the wavelet decomposition of an arbitrary length segment may differ. In this sub-section, we discuss three categories of wavelet filters, namely, orthogonal wavelets, odd symmetric biorthogonal wavelets, and even symmetric biorthogonal wavelets. The orthogonal wavelets have orthogonal lowpass and highpass filters and an even number of taps for each filter. The odd symmetric biorthogonal wavelets have biorthogonal and symmetric lowpass and highpass filters and an odd number of taps for each filter. The even symmetric biorthogonal wavelets have biorthogonal symmetric lowpass and anti-symmetric highpass filters and an even number of taps for each filter. For image and video coding, all these wavelet filters are FIR filters with real taps.

2.1.1 *Boundary extensions for wavelet decomposition*

One of the issues in applying a wavelet decomposition to a finite-length signal segment is how to deal with the boundaries (leading and trailing) of the signal segment. In order to maintain the perfect reconstruction property of the wavelet transform, the undefined pixel locations outside the finite-length signal segment are filled with values related to the pixels inside the finite-length signal segment. This is called a boundary extension. The extensions for our discussion include periodic extension and symmetric extensions.

Fig. 1 illustrates the periodic extension in which the finite-length signal segment is considered to be one period of a periodic signal. To describe the symmetric extension, we adopt the same terminology used in MPEG-4[3]. There are several symmetric extensions and they are classified into two categories for leading boundary and trailing boundary respectively. There are three types of symmetric extensions in each category (Type A, Type B, and Type C). Fig. 2 illustrates the different types of symmetric extensions. In each figure, the arrow points to the symmetric point of the corresponding extension type. If the signal segment is much shorter than the filter length, the symmetric extension is performed several times by alternately extending

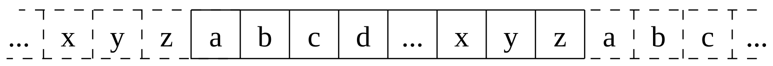


Figure 1: Periodic extensions at leading and trailing boundaries of a finite length signal segment.

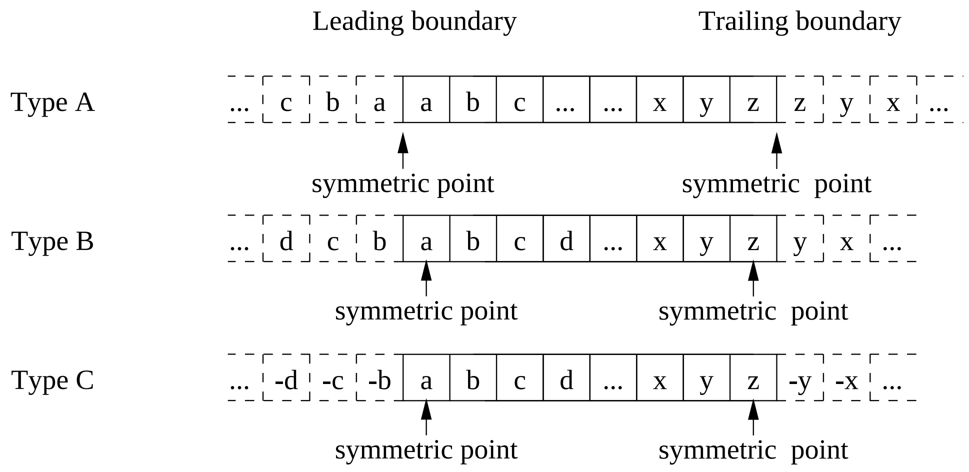


Figure 2: Symmetric extension types at the leading and trailing boundaries of a signal segment.

the values at both leading and trailing boundaries. Fig. 3 illustrates how to alternately extend the leading and trailing boundaries for a short signal segment.

Once extensions are applied on a finite-length, say N points, signal segment, the signal can be considered to have infinite length in any mathematical equations. The signal samples with indexes less than 0 or greater than $N-1$ are derived from the extensions.

2.1.2 *Arbitrary-length wavelet decomposition for orthogonal wavelets*

Let $g(i)$, $h(i)$, $e(i)$, and $f(i)$ be the impulse responses of the lowpass analysis, highpass analysis, lowpass synthesis, and highpass synthesis orthogonal wavelet filters with L taps and $i = 0, \ldots, L-1$. For orthogonal wavelet filters, the filter length, L is an even number. Let $x(i)$ be the input signal with a finite and even length N and periodic extensions. The relations between these filters and wavelet analysis and synthesis processes are given in Appendix A.1.

From Appendix A.1, we learn that the subsampling positions for wavelet coefficients can be either at odd positions or at even positions. If subsampling occurs at even positions, including its extensions, i.e., $\ldots, -2, 0, 2, \ldots, 2i, \ldots$, we refer this kind of subsampling to *even* subsampling. If the subsampling occurs at odd positions, including its extensions, i.e., $\ldots, -1, 1, \ldots, 2i+1, \ldots$, we refer this kind of subsampling to *odd* subsampling. We'll use these terms for the rest of this paper.

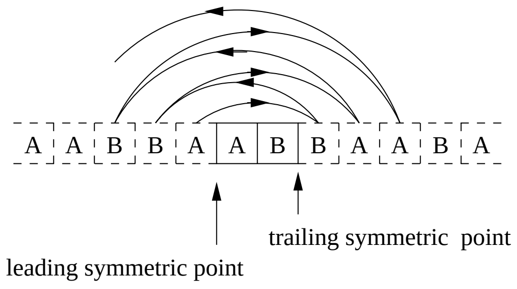


Figure 3: Symmetric extensions (type A) at leading and trailing boundaries for a short signal segment.

Fig. 4 illustrates the analysis and synthesis processes where the subsampling takes place at the even positions.

Let $\{x(j),\ j = 0, \ldots, N-1\}$ be the input signal segment with a finite length N. If N is odd, the straightforward periodic extension approach cannot be used, since one cannot recover the coefficients needed for periodic extension in the synthesis process. Li *et al* [19, 21] proposed an approach to handle the odd length signal segment. A complete description of this approach together with the even length case is given as follows. (s is defined in Appendix A.1 to indicate subsampling positions.)

1. *If $N = 1$, this isolated sample is repeatedly extended and the lowpass wavelet analysis filter is applied to obtain a single lowpass wavelet coefficient. (Note: this is equivalent to scaling this isolated sample by a factor $\sum_{i=0}^{L-1} g(i)$ and this happens to be $\sqrt{2}$ for orthogonal wavelets.) The synthesis process just scales the single wavelet coefficient by $1/\sqrt{2}$ and puts it in the correct position in the original signal domain.*

2. *If N is greater than 1 and N is even, the L-tap wavelet analysis is performed on the N samples with periodic extension (Fig. 4). $N/2$ lowpass wavelet coefficients $C(i)$, $i = 0, \ldots, N/2-1$, are generated by Eq. (A-4) and (A-6), and $N/2$ highpass wavelet coefficients $D(i)$, $i = 0, \ldots, N/2-1$, are generated by Eq. (A-5) and (A-7).*

   The synthesis process begins with periodic extension of the lowpass and highpass wavelet coefficients $C(i)$ and $D(i)$ (Fig. 4). The extended lowpass and highpass wavelet coefficients are then upsampled according to Eq. (A-8) and (A-9), respectively. Eq. (A-10), (A-11), and (A-12) are then applied to reconstruct the signal segment $r(j)$, $j = 0, \ldots, N-1$.

3. *If N is greater than 1 and N is odd, the L-tap wavelet transform is performed on the first $N-1$ (an even number) samples with periodic extension. $(N-1)/2$ wavelet coefficients are generated in the same way as in 2 (segment length $N-1$) for the lowpass and highpass wavelet coefficients, respectively. The left-over sample is treated in the same way as in 1 (scaled by $\sqrt{2}$) and is then appended to the end of lowpass wavelet coefficients forming a total of $(N+1)/2$ coefficients.*

From the above description, we observe that,

1. the number of coefficients in the transform domain is identical to that of the pixels in the image domain;

2. there is a unique inverse wavelet transform that can perfectly reconstruct the original image segment as long as the L-tap wavelet filters used have perfect reconstruction property and the subsampling positions are known;

3. this length-adaptive wavelet transform preserves the locality of an image segment without generating out-of-boundary coefficients;

4. this length-adaptive wavelet transform keeps the same scale for coefficients in lowpass object for odd length segment by scaling the left-over sample. This avoids a sharp change that may occur at the segment boundary if this sample is not scaled.

2.1.3 *Arbitrary length wavelet decomposition for odd symmetric biorthogonal wavelets*

Periodic extensions used in orthogonal wavelets work for any wavelet transforms. However, if the signal segment is long, the correlation between the end of the signal and the start of the signal is small. There could be a good chance of a sharp change at the transition from the end of previous signal period to the start of the next signal period if the periodic extension method is used. Another commonly used extension type is symmetric extensions. However, to ensure perfect reconstruction, symmetric extensions are only possible for symmetric biorthogonal wavelet transforms. Another reason for using biorthogonal wavelets is because there is no non-trivial orthogonal wavelet that is linear-phase (thus symmetric or antisymmetric), perfect reconstruction, and FIR with real filter taps.

In the symmetric extension scheme, the signal is extended symmetrically at the leading and trailing boundaries of a signal segment. The neighboring samples with such symmetric extensions have the same close

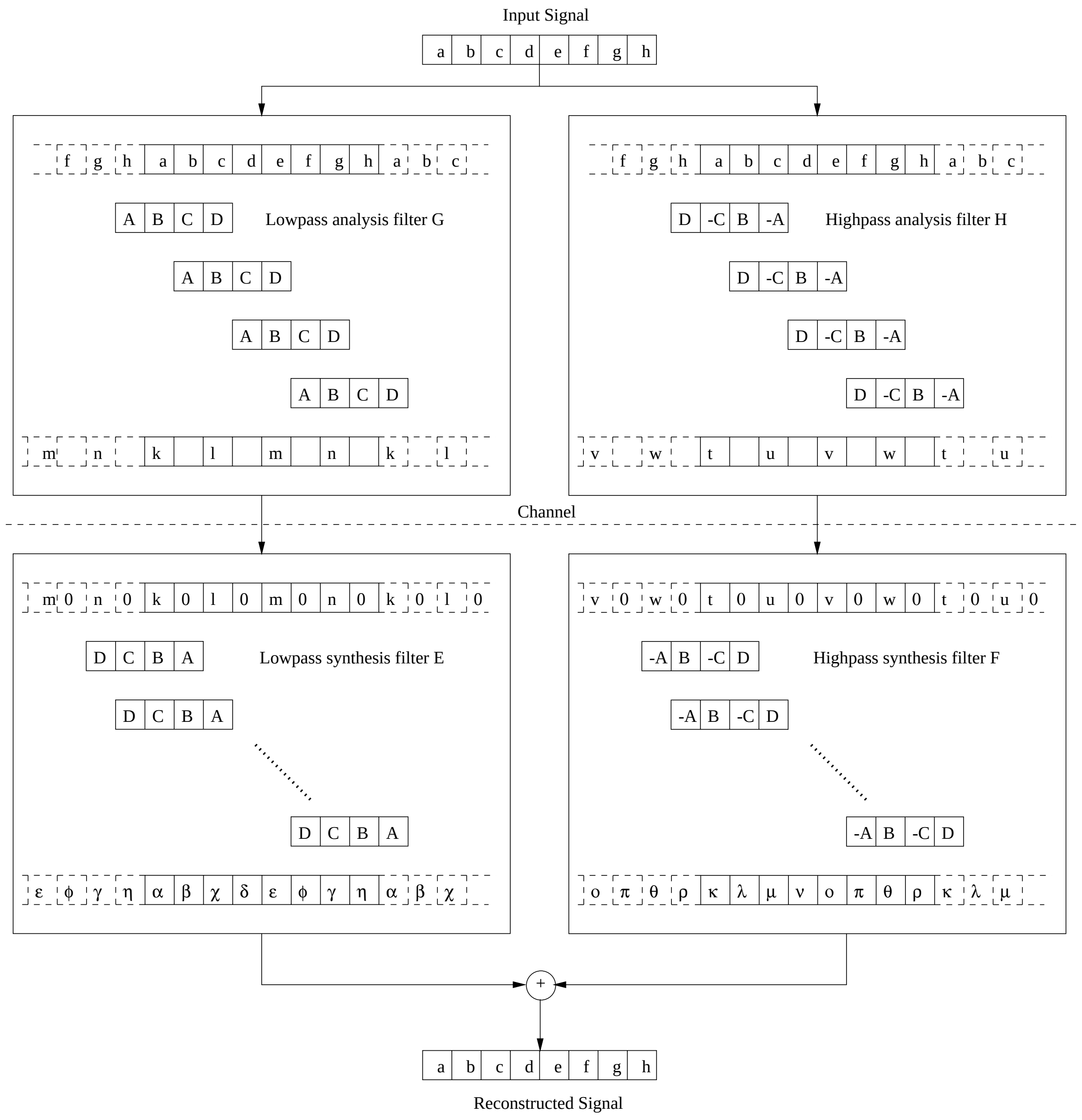

Input Signal
a b c d e f g h
f g h a b c d e f g h a b c
A B C D
Lowpass analysis filter G
D -C B -A
Highpass analysis filter H
m n k l m n k l
v w t u v w t u
Channel
m 0 n 0 k 0 l 0 m 0 n 0 k 0 l 0
v 0 w 0 t 0 u 0 v 0 w 0 t 0 u 0
D C B A
Lowpass synthesis filter E
-A B -C D
Highpass synthesis filter F
ε φ γ η α β χ δ ε φ γ η α β χ
ο π θ ρ κ λ μ ν ο π θ ρ κ λ μ
+
a b c d e f g h
Reconstructed Signal

Figure 4: The analysis and synthesis processes of an even length signal segment using orthogonal wavelet transforms.

correlation as in the original signal segment. Therefore, sharp transitions are avoided. Another advantage of biorthogonal wavelets is that they can provide linear phase filters, therefore, eliminating the phase distortion caused by magnitude distortion of transformed coefficients. This is very important when they are applied to signal compression where magnitudes of the transformed coefficients is most likely to be quantized.

Let $\{g(i), i = 0, \ldots, L_g - 1\}$, $\{h(i), i = 0, \ldots, L_h - 1\}$, $\{e(i), i = 0, \ldots, L_h - 1\}$, and $\{f(i), i = 0, \ldots, L_g - 1\}$ be the impulse responses of the lowpass analysis filter, highpass analysis filter, lowpass synthesis filter, and highpass synthesis filter, respectively. The filter lengths, both L_g and L_h, are odd numbers. Let $x(i)$ be the input signal with a finite length with appropriate extensions at the leading and trailing boundaries. The relations between these filters, and wavelet analysis and synthesis processes are given in Appendix A.2.

Assuming a signal segment $\{x(j), j = 0, \ldots, N-1\}$, with length of N, and combining symmetric extensions, filtering, and subsampling together, the arbitrary length wavelet decomposition using odd symmetric wavelet transforms can be described as follows. (s is defined in Appendix A.2 to indicate subsampling positions.)

1. *If $N = 1$, this isolated sample is repeatedly extended and the lowpass wavelet analysis filter is applied to obtain a single lowpass wavelet coefficient. (Note: this is equivalent to scaling this sample by a factor $K = \sum_{i=0}^{L_g-1} g(i)$ and it happens to be $\sqrt{2}$ for some normalized biorthogonal wavelets.)*

   The synthesis process simply scales this single lowpass wavelet coefficient by a factor of $1/K$ and puts it in the correct position in the original signal domain.

2. *If N is greater than 1, and $p = 0$ if N is even, $p = 1$ if N is odd, the signal segment is extended using the type B extensions (Fig. 2). The $(N/2 + p(1-s))$ lowpass wavelet coefficients $C(i)$, $i = s, \ldots, (N/2 - (1-p)(1-s))$, are generated by Eq. (A-17) and (A-19). The $(N/2 + ps)$ highpass wavelet coefficients $D(i)$, $i = 0, \ldots, (N/2 - 1 + ps)$, are generated by Eq. (A-18) and (A-20).*

   The synthesis process begins with upsampling the lowpass and highpass wavelet coefficients using Eq. (A-21) and (A-22), respectively. As the results, an upsampled lowpass segment $P(j)$ and an upsampled highpass segment $Q(j)$ are obtained, where $j = 0, \ldots, N-1$. The upsampled lowpass and highpass segments $P(j)$ and $Q(j)$ are then extended using the type B extension (Fig. 2). The extended lowpass and highpass signal $P(j)$ and $Q(j)$ are then synthesized using Eq. (A-23), (A-24), and (A-25) to reconstruct the signal segment $r(j)$, $j = 0, \ldots, N-1$.

Fig. 5 illustrates the analysis and synthesis processes with even-subsampling for the lowpass wavelet coefficients and odd-sampling for the highpass wavelet coefficients for an odd length signal segment.

The default wavelet in MPEG-4 [2] is an odd symmetric biorthogonal wavelet. The above description is completely equivalent to that in [2]. The difference is in the description of upsampling and symmetric extensions for synthesis. In [2], symmetric extensions are described before upsampling. The above description puts upsampling before symmetric extensions, which is more concise. The MPEG-4 reference software actually follows the above concise description.

2.1.4 *Arbitrary-length wavelet decomposition for even symmetric biorthogonal wavelets*

Let $\{g(i), i = 0, \ldots, L_g - 1\}$, $\{h(i), i = 0, \ldots, L_h - 1\}$, $\{e(i), i = 0, \ldots, L_h - 1\}$, and $\{f(i), i = 0, \ldots, L_g - 1\}$ be the impulse responses of the lowpass analysis filter, highpass analysis filter, lowpass synthesis filter, and highpass synthesis filter, respectively. The filter lengths, both L_g and L_h, are even numbers. Let $x(i)$ be the input signal with a finite length and appropriate extensions at the leading and trailing boundaries. The relations between these filters, and wavelet analysis and synthesis processes are given in Appendix A.3.

Assuming a signal segment $\{x(j), j = 0, \ldots, N-1\}$, with length of N, and combining symmetric extensions, filtering, and subsampling together, the arbitrary length wavelet decomposition using even symmetric wavelet transform can be described as follows. (s is defined in Appendix A.3 to indicate subsampling positions.)

1. *If $N = 1$, this isolated sample is repeatedly extended, and the lowpass wavelet filter is applied to obtain a single lowpass wavelet coefficient. (Note: this is equivalent to scaling this sample by a factor $K = \sum_{i=0}^{L_g-1} g(i)$.)*

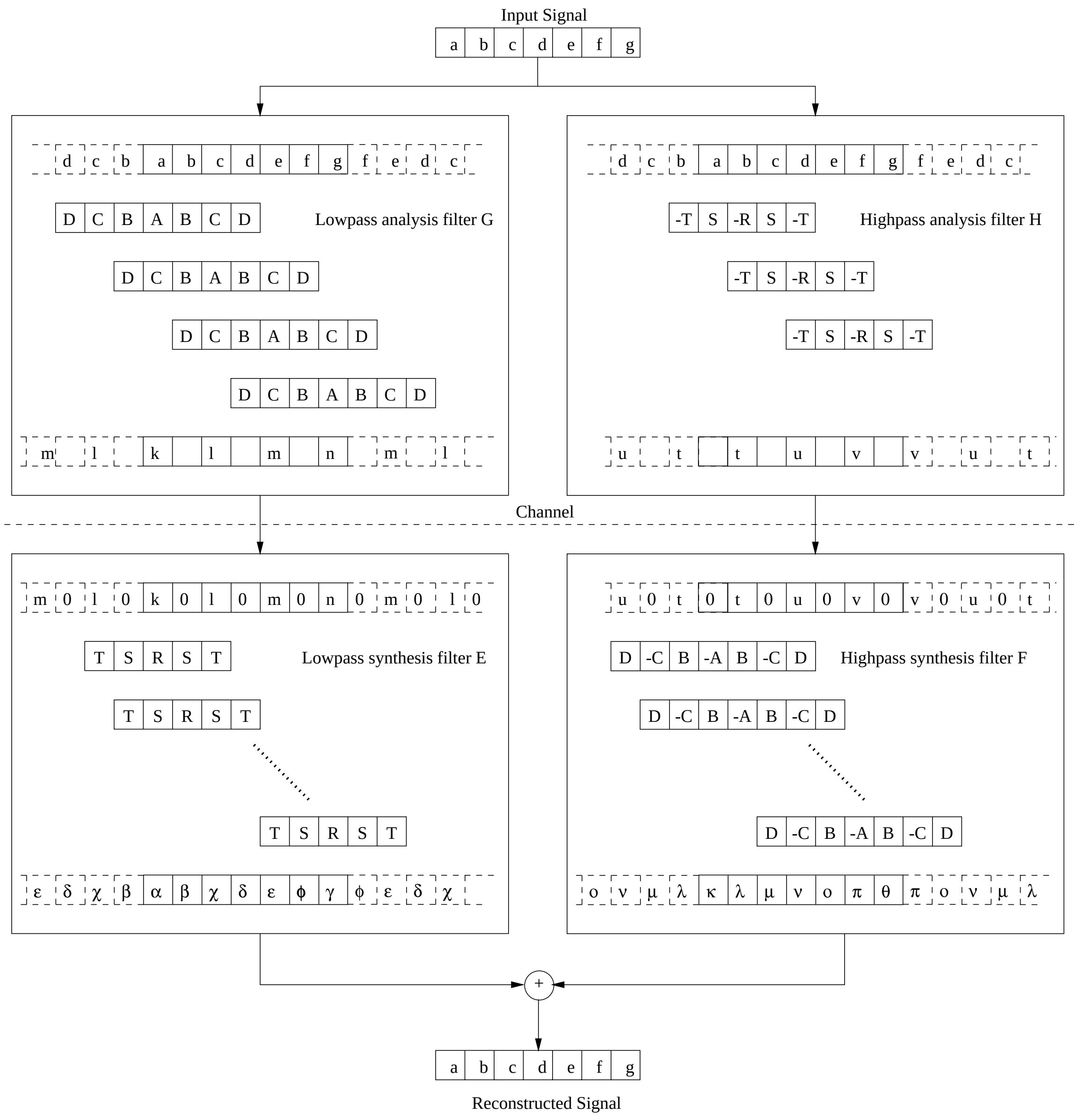

Input Signal
a b c d e f g
d c b a b c d e f g f e d c
D C B A B C D
Lowpass analysis filter G
m l k l m n m l
-T S -R S -T
Highpass analysis filter H
u t t u v v u t
Channel
m 0 l 0 k 0 l 0 m 0 n 0 m 0 l 0
T S R S T
Lowpass synthesis filter E
ε δ χ β α β χ δ ε ϕ γ ϕ ε δ χ
u 0 t 0 t 0 u 0 v 0 v 0 u 0 t
D -C B -A B -C D
Highpass synthesis filter F
ο ν μ λ κ λ μ ν ο π θ π ο ν μ λ
+
a b c d e f g
Reconstructed Signal

Figure 5: The analysis and synthesis processes of an odd length signal segment using odd symmetric biorthogonal wavelet transforms.

The synthesis process simply scales this single lowpass wavelet coefficient by a factor of $1/K$ and puts it in the correct position in the original signal domain.

2. *If N is greater than 1, and $p = 0$ if N is even, $p = 1$ if N is odd, the signal segment is extended using the type A extension (Fig. 2). The $(N/2 + 1 - (1 - p)(1 - s))$ lowpass wavelet coefficients $C(i)$, $i = 0, \ldots, (N/2 - (1-p)(1-s))$, are generated by Eq. (A-30) and (A-32). The $(N/2+1-(1-p)(1-s))$ highpass wavelet coefficients $D(i)$, $i = 0, \ldots, (N/2 - (1 - p)(1 - s))$, are generated by Eq. (A-31) and (A-33).*

   For the odd-subsampling case ($s = 1$) and N is even ($p = 0$), the wavelet decomposition generates $N/2 + 1$ wavelet coefficients for both lowpass and highpass bands. However, the first and last highpass wavelet coefficients are always zeros. In this case, only $N/2 - 1$ highpass coefficients, $\{D(i),\ i = 1, \ldots, N/2 - 1\}$, need to be transmitted to the synthesis end. Similarly, when $p = 1$ and $s = 1$, the first highpass wavelet coefficient, $D(0)$, is always zero; and when $p = 1$ and $s = 0$ the last highpass wavelet coefficient, $D((N + 1)/2 - 1)$, is always zero. Therefore when $p = 1$, only $(N - 1)/2$ highpass wavelet coefficients $D(i)$, $i = s, \ldots, ((N - 1)/2 - 1 + s)$ need to be transmitted.

   The synthesis process begins by inserting the known and not-transmitted zeros, if there are any, into the highpass wavelet coefficients. Then the lowpass wavelet coefficients $C(i)$ and the highpass wavelet coefficients $D(i)$ are upsampled using Eq. (A-34) and (A-35), respectively. The upsampled lowpass signal segment $P(j)$, $j = -1, \ldots, N-1$ is symmetrically extended using the type B extensions (Fig. 2). The upsampled highpass segment $Q(j)$, $j = -1, \ldots, N-1$, is extended using the type C anti-symmetric extensions (Fig. 2). Eq. (A-36), (A-37), and (A-38) are then applied to the extended lowpass and highpass wavelet coefficients to reconstruct the signal segment $r(j)$, $j = 0, \ldots, N - 1$.

Fig. 6 gives an example of the analysis and synthesis processes with even-subsampling for an odd-length signal segment.

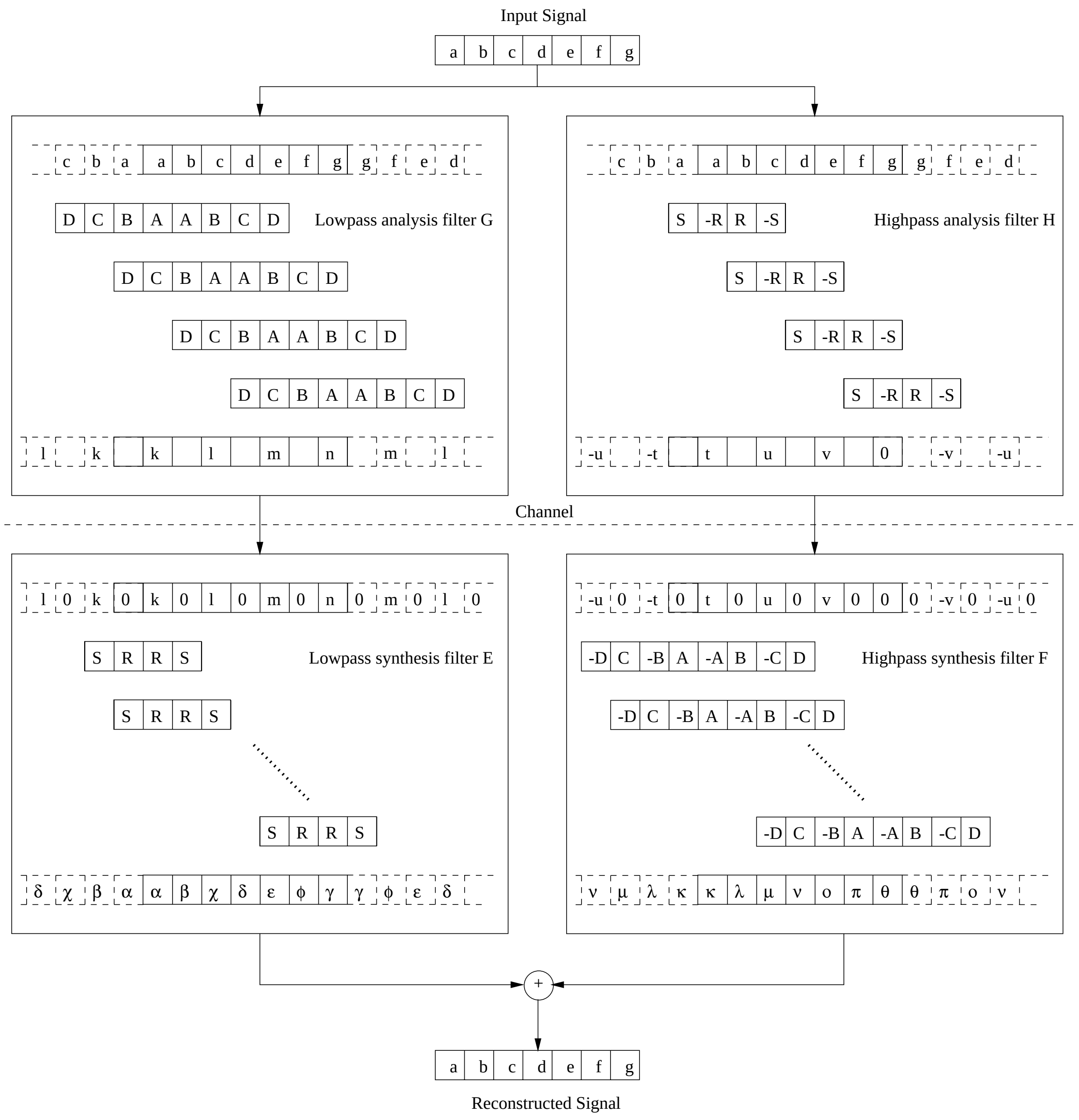
2.2 Subsampling Strategies in Shape Adaptive Discrete Wavelet Transforms

The length-adaptive wavelet transforms discussed in the previous subsections solve the problem of wavelet decomposition on an arbitrary length signal segment (long or short, even or odd). In the discussions, we purposely leave the subsampling issue open. For each case of the arbitrary length wavelet transforms, we discussed two options in subsampling the lowpass and highpass wavelet coefficients, i.e., *even subsampling* and *odd subsampling*. Different subsampling strategies have different advantages and disadvantages in terms of coding efficiency. In this sub-section, we discuss two possible subsampling strategies.

2.2.1 *Subsampling strategy favoring zerotree coding efficiency*

Since Zerotree Coding (ZTC) scheme is used for entropy coding, the properties of ZTC should be considered to choose a proper subsampling strategy. One of the features of ZTC is that, when all the descendants of a tree node are insignificant or zeros or *don't-cares*, the coding process does not need to continue for that tree branch from that node on. Therefore, the obvious subsampling strategy to take advantage of this ZTC property is to always allocate more valid wavelet coefficients in the lower subbands (close to roots of the wavelet tree) and more *don't-care* nodes in the higher subbands. Li, *et al* [21] observed that by always choosing *even* subsampling *locally* across all signal segments for orthogonal wavelets, we can guarantee that the number of the lowpass band wavelet coefficients is not less than that of the highpass band wavelet coefficients. Here, *local* sampling positions refer to positions relative to the beginning of a signal segment, and *global* subsampling positions refer to positions relative to the bounding box of the visual object. The same observation is true for the even symmetric biorthogonal filters. For odd symmetric filters, this is achieved if the lowpass subsampling is locally fixed to *even subsampling* and the highpass subsampling is locally fixed to *odd subsampling*.

Since the signal segments in an arbitrarily shaped visual object are neither all starting from odd positions nor all starting from even positions, the phases of some of the lowpass and highpass wavelet coefficients may be skewed by one sample when subsampling is locally fixed for all signal segments. This is not desired for wavelet decomposition in the second direction. However, since the phases of the subsampled wavelet coefficients differ at most by one sample, the spatial relations across subbands can still be preserved to a



Input Signal
a b c d e f g
c b a a b c d e f g g f e d
D C B A A B C D
Lowpass analysis filter G
l k k l m n m l
S -R R -S
Highpass analysis filter H
-u -t t u v 0 -v -u
Channel
l 0 k 0 k 0 l 0 m 0 n 0 m 0 l 0
S R R S
Lowpass synthesis filter E
δ χ β α α β χ δ ε φ γ γ φ ε δ
-u 0 -t 0 t 0 u 0 v 0 0 0 -v 0 -u 0
-D C -B A -A B -C D
Highpass synthesis filter F
ν μ λ κ κ λ μ ν ο π θ θ π ο ν
+
a b c d e f g
Reconstructed Signal

Figure 6: The analysis and synthesis processes of an odd length signal segment using even symmetric biorthogonal wavelet transforms.

certain extent. For very low bit rate coding, zerotree coding efficiency is more important and this subsampling strategy achieves better overall coding efficiency.

2.2.2 *Subsampling strategy favoring signal processing gain*

In contrast, another subsampling strategy does not fix even sampling or odd sampling locally for all the signal segments. It strictly maintains the spatial relations across the subbands by using either even sampling or odd sampling according to the position of a signal segment relative to the bounding box. Instead of fixing subsampling positions *locally* for each signal segment, this strategy fixes subsampling positions of the highpass and lowpass wavelet coefficients at *global* even or odd positions relative to the bounding box. Since the starting position of each segment in a visual object may not be always at an even or odd position, the local subsampling positions in each segment have to be adjusted to achieve global *even* or *odd* subsampling. For example, with odd symmetric biorthogonal filters, to achieve global *even* subsampling in lowpass bands and global *odd* subsampling in highpass bands, we need to choose local *even* subsampling in lowpass bands and local *odd* subsampling in highpass bands for all segments starting from even positions, and choose local *odd* subsampling in lowpass bands and local *even* subsampling in highpass bands for all segments starting from odd positions.

This subsampling strategy preserves the spatial correlation across subbands, therefore, it can achieve more signal processing gain. Its drawback is that it may introduce more highpass band coefficients than lowpass band coefficients and could potentially degrade zerotree coding efficiency.

For a rectangular region, the above two subsampling strategies converge into a normal wavelet subsampling scheme.

2.3 *Two-dimensional shape adaptive discrete wavelet transforms*

Based on the length-adaptive wavelet transform algorithms and the subsampling strategies discussed above, the 2-D SA-DWT (pyramid decomposition) for an arbitrarily shaped visual object can be described as follows:

1. *Within the bounding box of the arbitrarily shaped object, use shape information to identify the first row of pixels belonging to the object to be transformed;*
2. *Within each row, identify the first segment of consecutive pixels;*
3. *Apply the length-adaptive 1-D wavelet transform to this segment with a proper subsampling strategy;*
4. *The lowpass wavelet coefficients are placed into the corresponding row in the lowpass band. The highpass wavelet coefficients are placed into the corresponding row in the highpass band; (Fig. 7(a)(b) illustrates this step for the subsampling strategy favoring the ZTC gain.)*
5. *Perform the above operations for the next segment of consecutive pixels in the row;*
6. *Perform the above operations for the next row of pixels;*
7. *Perform the above operations for each column of the lowpass and highpass objects;*
8. *Perform the above operations to the lowpass-lowpass band object until the level of wavelet decomposition is reached. (Fig. 7(c) illustrates the result of a 2-level wavelet decomposition of an arbitrarily shaped object.)*

The above 2-D SA-DWT algorithm provides a way to efficiently decompose an arbitrarily shaped object into a multi-resolution object pyramid. The spatial correlation, locality and object shape are well preserved throughout the SA-DWT. Thus it enables multi-resolution coding of arbitrarily shaped objects. This method ensures that the number of coefficients to be coded in the transform domain is exactly the same as that in the image domain. The treatment of odd number of pixels in a segment ensures that there is not too much energy leaked into highpass bands in a pyramid wavelet decomposition. Note that if the object is a rectangular image, the 2-D SA-DWT is identical to a standard 2-D wavelet transform.

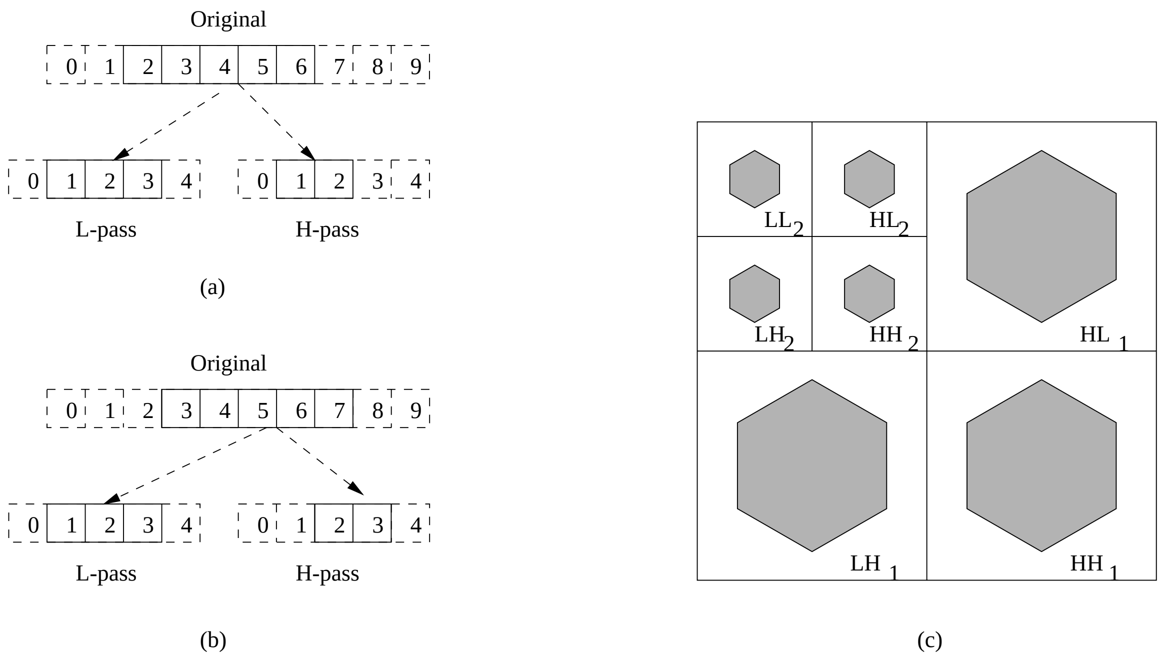


Figure 7: 2-D SA-DWT. (a) and (b): Examples of locations of the lowpass and highpass objects in the transform domain; (c) Multi-resolution decomposition of an arbitrarily shaped object.

The discussion of optimized implementation of the SA-DWT is out of the scope of this paper. However, based on the discussion in this section, we can provide an estimation on the memory requirement and the computational complexity. The SA-DWT can be implemented as line based wavelet transform as described above. Thus the working buffer needed is no larger than the larger of width and height of the bounding box. In order to store the orginal image and the transformed coefficients, there needs a main memory of $W \times H$ words, where W and H are width and height of the bounding box of the visual object, respectively. Memory access is another aspect of memory requirement, sometimes especially in hardware implementation, it is more important than memory size. The current implementation of SA-DWT is performed using separable 1D wavelet transforms, that is, each level of decomposition needs to read and write the memory for that subband twice. In contrast, the SA-DCT needs a working buffer of $8 \times 8 = 64$ words and a main memory of $W \times H$ words and each frame needs only to read and write memory once.

The computational complexity of the SA-DWT depends on the number of pixels in the object. If the bounding box of the visual object has a width W and a height H, the computational complexity of the SA-DWT is no more than that of a conventional wavelet transform on a $W \times H$ image. The overhead of the SA-DWT comes from searching segments in a row or column. Since this is a simple logic operation, it doesn't require too much in computational complexity. It increases the hardware design complexity since new logic for handling the segment search is required. Similar to the conventional wavelet transform, the complexity of the SA-DWT depends on the number of filter taps. It could be simpler than the SA-DCT if the number of filter taps is few and it could also be more complex than the SA-DCT if the wavelet filter has a large number of taps. Table 1 gives the estimation of the memory requirement and computational complexity of the worst-case SA-DWT and SA-DCT. For the SA-DWT, we use MPEG-4 default (9,3) wavelet filters in Table 4 and take advantage of the symmetric property of the wavelet filters. The computation of SA-DWT can be much simpler if implemented using lifting scheme, further discussion is beyond the scope of this paper. For the SA-DCT, we use the fast algorithm for 8-point DCT (i.e. 11 multiplications and 29 additions)[25] for the estimation. Again, we assume the bounding box has size $W \times H$ and there are L levels of the SA-DWT decomposition.

Table 1: Comparisons of Memory Requirements and Computation Complexity between SA-DWT and SA-DCT schemes

	working buffer (words)	main memory (words)	memory access (words)	computational complexity (additions (+) and multiplications (×))
SA-DWT	$max(W,H)$	WH	$\frac{8}{3}(1-4^{-L})WH$ (reads) $\frac{8}{3}(1-4^{-L})WH$ (writes)	$\frac{40}{3}(1-4^{-L})WH$ (+) $\frac{28}{3}(1-4^{-L})WH$ (×)
SA-DCT	64	WH	WH (reads) WH (writes)	$\frac{29}{4}WH$ (+) $\frac{11}{4}WH$ (×)

3 EXTENSIONS OF ZEROTREE CODING

In the SA-DWT decomposition, the shape mask is decomposed into a pyramid of subbands in the same way as the SA-DWT so that we know which wavelet tree nodes have valid wavelet coefficients and which ones have *don't-care* values. We have to pay attention to the way of coding the multi-resolution arbitrarily shaped objects with these *don't-care* values (corresponding to the out-of-boundary pixels or out-nodes). In this section, we discuss how to extend the conventional zerotree coding methods to the shape adaptive case.

In the MPEG-4 standard, both *Embedded Zerotree Wavelet (EZW)* coding and *ZeroTree Entropy (ZTE)* coding are included [2] for still texture coding to support a wide range of functionalities with spatial and SNR scalabilities. EZW coding [4] provides an embedded bit stream with very fine SNR scalability and precise bit rate control. An extension of the EZW coding technique into shape adaptive wavelet coding is described in [21]. ZTE coding is an improvement over EZW coding in terms of coding efficiency [5]. One prominent feature of ZTE coding is to use explicit quantization and therefore it can be optimized and dynamically adapted to scene content. Another improvement made in MPEG-4 is to separately code the low-low subband from the higher subbands. The wavelet coefficients in the low-low subband are coded with an adaptive prediction method. For shape adaptive wavelet coding, there are some *don't-care* values in the low-low subband and they are not coded. For prediction of other wavelet coefficients in the low-low subband, the *don't-care* values are considered to be zeros.

As discussed previously, the SA-DWT decomposes the arbitrarily shaped objects in the image domain to a hierarchical structure with a set of subbands of varying resolutions. Each subband has a corresponding shape mask associated with it to specify the locations of the valid coefficients in that subband. Fig. 8 shows the parent-child relation of SA-DWT trees descending from a coefficient in the subband LL_3. As shown in the figure, there are three types of nodes in a tree: zeros, non-zeros, and out-nodes (with *don't-care* values). The major task is to extend the conventional zerotree coding (EZW or ZTE) methods to the case with out-nodes. A simple way is to set those *don't-care* values to zeros and then apply the conventional EZW or ZTE coding method. However, this requires bits to code the out-notes such as a *don't-care* tree (the parent and all of its children have *don't-care* values). This is a waste of bits because out-nodes do not need to be coded as the shape mask already indicates their status. Therefore, we should treat out-nodes differently from zeros. Although we don't want to use bits to code an out-node, we have to decide what to do with its children nodes. One way is not to code any information about the status of the children nodes of the *don't-care* node. In this way, we always assume that it has four children to be examined further. When the decoder scans to this node, it will be informed by the shape information that this node is a *don't-care* node and it will continue to scan its four children nodes. By doing so, all the *don't-care* nodes in a tree structure need not be coded. This approach performs well when there are only sparse valid nodes in a tree structure. One disadvantage of this approach is that, even if a *don't-care* node has four zerotree root children, it still needs to code four zerotree root symbols instead of one (if the *don't-care* value is treated as a zero). Another way is to selectively treat an out-node as a zero. But this is equivalent to creating another symbol for coding some *don't-care* values. Through extensive experiments, we decide to use the method of not coding out-nodes [2, 24]. The extensions of ZTC coding to handle the SA-DWT coefficients are then given as follows.

Encoding and decoding SA-DWT coefficients are the same as encoding and decoding regular wavelet coefficients except keeping track of the locations of the wavelet coefficients according to the shape informa-

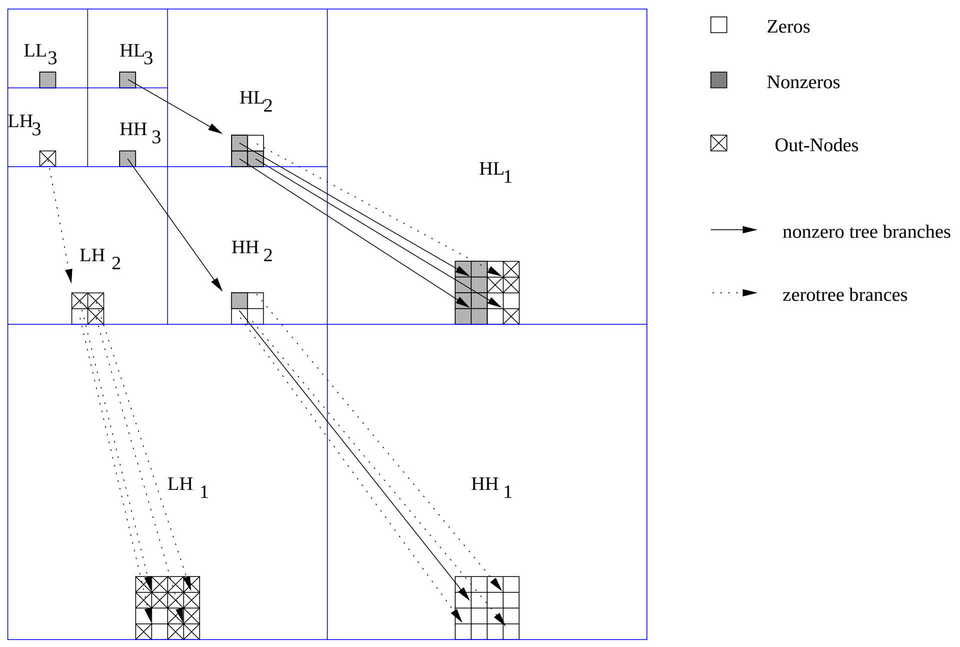


Figure 8: Parent-child relation of wavelet trees in SA-DWT subbands.

tion. Same as encoding and decoding regular wavelet coefficients, a zerotree symbol is used to determine whether encoding and decoding are needed for its children nodes. The difference is that some zerotree nodes correspond to the pixel locations outside the shape boundary and no bits should be used for these out-nodes. The following description applies to both EZW and ZTE coding schemes for each coding pass (i.e. for each bit-plane coding in EZW and each quantization step in ZTE).

At the root layer of the wavelet tree (the top 3 AC bands), the shape information is examined to determine whether a node is an out-node.

- *If it is an out-node, no bits are used for this node and the four children of this node are marked "to-be-encoded" (TBE) in encoding or "to-be-decoded" (TBD) in decoding;*
- *otherwise, a symbol is encoded/decoded for this node using an adaptive arithmetic encoder/decoder.*
  - *If the symbol is either isolated-zero (IZ) or value (VAL), the four children of this node are marked TBE/TBD; otherwise, the symbol is either zerotree-root (ZTR) or valued-zerotree-root (VZTR) and the four children of this node are marked "no-code" (NC).*
  - *If the symbol is VAL or VZTR, a non-zero wavelet coefficient is encoded/decoded for this node; otherwise, the symbol is either IZ or ZTR and the wavelet coefficient is set to zero for this node.*

At any layer between the root layer and the leaf layer, the shape information is examined to determine whether a node is an out-node.

- *If it is an out-node, no bits are used for this node and the four children of this node are marked as either TBE/TBD or NC depending on whether this node itself is marked TBE/TBD or NC respectively;*
- *otherwise,*
  - *if it is marked NC, no bits are used for this node and the wavelet coefficient is zero for this node and the four children nodes are marked NC;*
  - *otherwise, a symbol is encoded/decoded for this node using an adaptive arithmetic encoder/decoder.*
    - *If the symbol is either isolated-zero (IZ) or value (VAL), the four children of this node are marked TBE/TBD; otherwise, the symbol is either zerotree-root (ZTR) or valued-zerotree-root (VZTR) and the four children of this node are marked "no-code" (NC).*

Table 2: Number of decomposition levels

picture size	luminance	chrominance
QCIF	4 levels	3 levels
CIF	5 levels	4 levels

  * *If the symbol is VAL or VZTR, a non-zero wavelet coefficient is encoded/decoded for this node using an arithmetic encoder/decoder; otherwise, the symbol is either IZ or ZTR and the wavelet coefficient is set to zero for this node.*

At the leaf layer, the shape information is examined again to determine whether a node is an out-node.

- *If it is an out-node, no bits are used for this node;*
- *otherwise,*
  - *if it is marked NC, no bits are used for this node and the wavelet coefficient is set to zero for this node;*
  - *otherwise, a wavelet coefficient is encoded/decoded for this node using an adaptive arithmetic encoder/decoder.*

For rectangular visual objects, the above extensions are exactly the same as regualr ZTC schemes.

4 EXPERIMENTAL RESULTS

Extensive experiments have been conducted to study the shape adaptive wavelet coding schemes described above. The results have been compared with that of other coding schemes, such as, SA-DCT coding [3, 10] and Egger's wavelet coding scheme [22, 23]. Comparison results have also been obtained for different wavelet filters (orthogonal, odd-symmetric biorthogonal, even-symmetric biorthogonal), for different subsampling strategies, and for different entropy coding schemes (EZW and ZTE).

The object shape information in the simulation are coded using the MPGE-4 shape coding tool. The objective test results are represented in the form of PSNR-bitrate curves and the shape bits are excluded from the bitrate since they are independent of the texture coding scheme. Only the texture bitrate (including both luminance and chrominance components) are used for comparison. Note that the bitrate (in bits per pixel) is calculated based on the number of pixels in an object with the reconstructed shape and the PSNR value is also calculated over the pixels in the object with the reconstructed shape.

4.1 Simulation Setup

For SA-DCT coding, the simulation program is the MPEG-4 verification model (VM) reference software (version 7) [3]. The software used for the wavelet schemes are modified based on the MPEG-4 still texture coding tools in which we contributed the shape adaptive wavelet coding implementation. The source material used in the simulations is from MPEG-4 test sequences. Single frames that have arbitrarily shaped regions are used in this experiment. The test conditions are given in Table 2 and Table 3.

The simulation results show a consistent similarity across these test sequences and across the luminance and chrominance color components. Therefore, only the Y component of Akiyo CIF sequence is presented in the PNSR-bitrate curves hereafter.

4.2 Comparison of Different Wavelet Filters for SA-DWT

We have discussed three different types of wavelet filters that can be used in SA-DWT, namely, orthogonal wavelet filters, odd-symmetric biorthogonal wavelet filters, even-symmetric biorthogonal wavelet filters. The comparison results presented are by no means complete and conclusive due to the vast variety of different

Table 3: Test sequences from MPEG-4

Sequence	Object	Frame number	Format	Shape Bitrate(bpp)
Akiyo	Woman	0	CIF	0.03657
Akiyo	Woman	0	QCIF	0.06059
Sean	Man	0	CIF	0.04380
Sean	Man	0	QCIF	0.07856
Coast guard	Big boat	100	CIF	0.15267
Coast guard	Big boat	100	QCIF	0.29966
Weather	Woman	0	CIF	0.03167
Weather	Woman	0	QCIF	0.05581

Table 4: Wavelet filters used in the simulation

Filter Type	Lowpass Taps $g(n)$	Highpass Taps $h(n)$
Orthogonal (4,4) (Daubechies)	{ 0.48296291314453, 0.83651630373781, 0.22414386804201, -0.12940952255126 }	{-0.12940952255126, -0.22414386804201 0.83651630373781, -0.48296291314453 }
Odd-Symmetric (9,3) (MPEG-4 default)	$\frac{\sqrt{2}}{128}\cdot$ {3, -6, -16, 38, 90, 38, -16, -6, 3 }	$\frac{\sqrt{2}}{4}\cdot$ {-1,2,-1}
Even-Symmetric (12,4)	{-0.01381067932005, 0.04143203796015, 0.05248058141619, -0.26792717880897, -0.07181553246426, 0.96674755240348, 0.96674755240348, -0.07181553246426, -0.26792717880897, 0.05248058141619, 0.04143203796015, -0.01381067932005}	{ 0.17677669529664, -0.53033008588991, 0.53033008588991, -0.17677669529664}

wavelet filters. They are presented here to give the readers some ideas on how much a difference the different wavelet filters could make in the SA-DWT.

Fig. 9 gives a typical PSNR-bitrate curve we have obtained in our simulation. The orthogonal wavelet filters are the Daubechies 4-tap orthogonal filters; the odd symmetric biorthogonal filters are the default 9-3 filters for MPEG-4 visual texture coding; the even-symmetric filters are 12-4 filters. The filter taps of all the filters are given in Table 4.

In this test, the SA-DWT subsampling strategy was chosen to be favoring the signal processing gain. For orthogonal filters and even-symmetric filters, this is *global* even-subsampling. For odd-symmetric filters this is *global* even-subsampling for lowpass wavelet coefficients and *global* odd-subsampling for highpass wavelet coefficients. Both EZW and ZTE coding schemes are simulated, since they give the similar results, only the results for ZTE (single quantizer mode) coding is presented. The simulation results show that the odd-symmetric filters usually achieve about 1 dB PSNR gain over orthogonal filters and even-symmetric filters. The orthogonal filters achieve almost the same performance as even-symmetric filters.

4.3 Comparison of EZW and ZTE Coding Schemes with SA-DWT

Fig. 10 gives the comparison of EZW and ZTE coding schemes. In the ZTE coding scheme, single quantizer (SQ) mode is used. The wavelet filters are the default wavelet filters used in MPEG-4 still texture coding given in Table 4. Subsampling strategy is chosen to favor the signal processing gain as in the last test. It is clearly shown in Fig. 10 that using either EZW or ZTE coding scheme with the SA-DWT produces more or less the same coding efficiency. While EZW has the embedded bitstream feature, ZTE has the flexible quantization feature. That's why they both have been adopted in the MPEG-4 standard to support different applications[2].

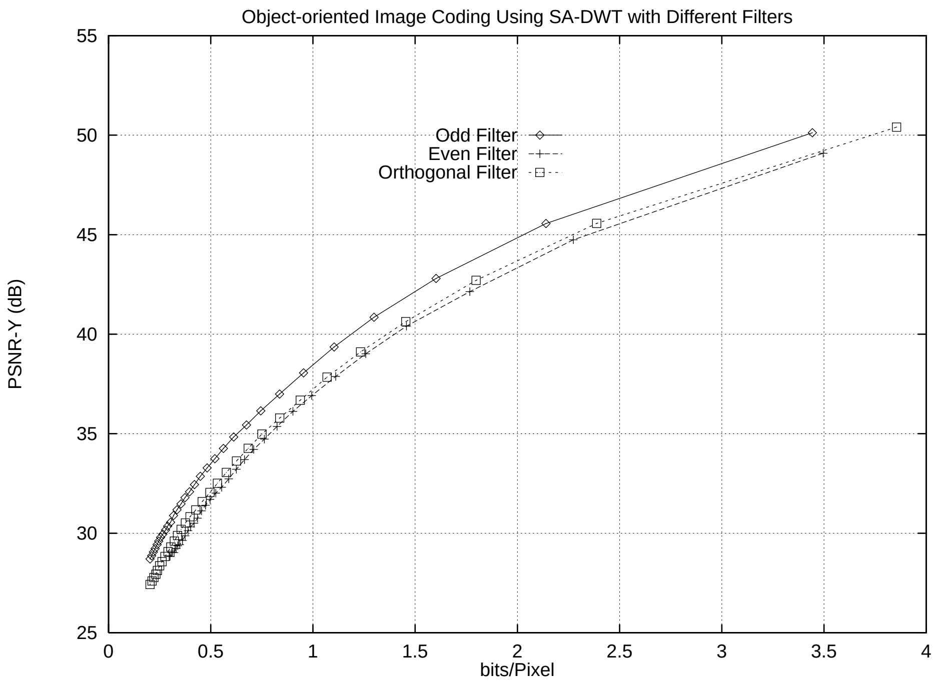


Figure 9: Comparison of different wavelet filters in shape adaptive wavelet coding (Akiyo-CIF).

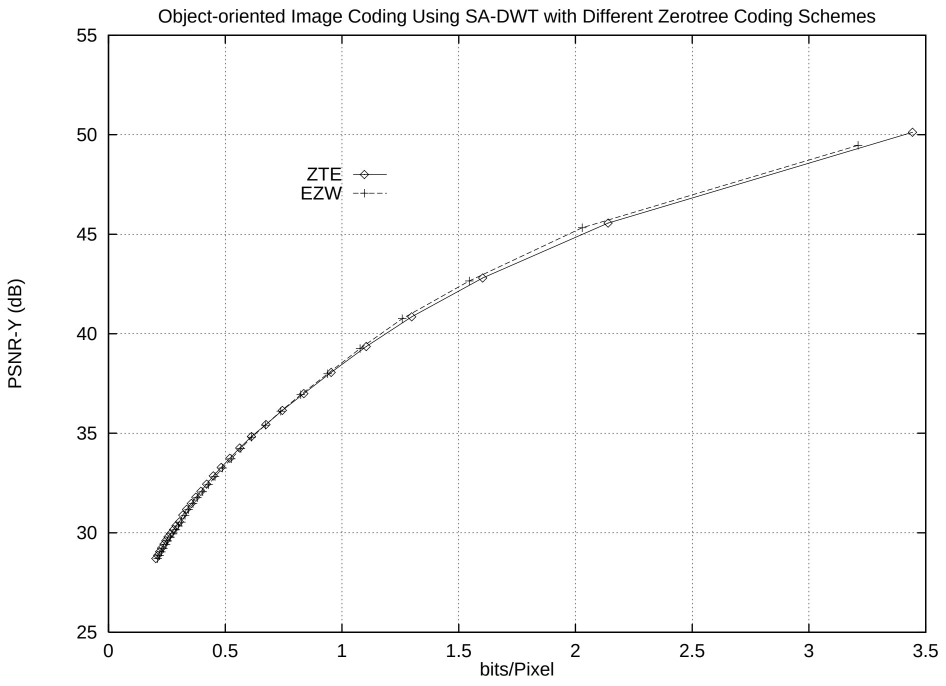


Figure 10: Comparison of different entropy coding schemes in shape adaptive wavelet coding (Akiyo-CIF).

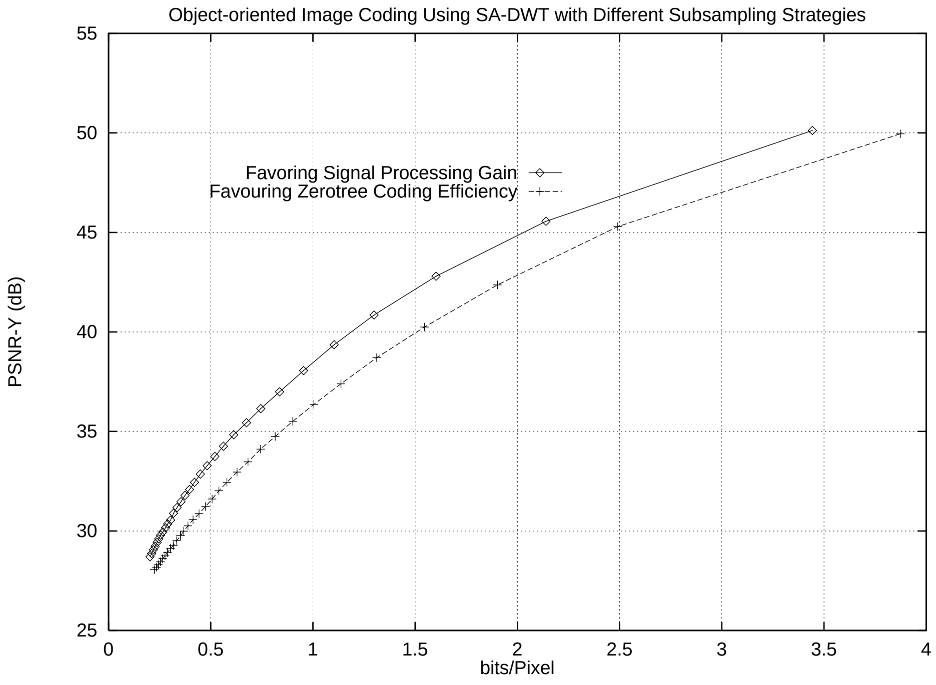


Figure 11: Comparison of different subsampling strategies in shape adaptive wavelet coding (Akiyo-CIF).

4.4 Comparison of Different Subsampling Strategies

As discussed in section 2.2, there are two subsampling strategies that may be useful in practice. One favors the zero-tree coding efficiency, the other one favors the signal processing gain. Fig. 11 presents the comparison of the simulation results of these two subsampling strategies under the same test conditions. In these simulations, ZTE (SQ) coding scheme and the default MPEG-4 wavelet filters are used. The simulation results show that under most bitrates, the subsampling strategy favoring signal processing gain have about 2-2.5 dB performance improvement over the subsampling strategy favoring the zerotree coding efficiency. However, at the lower bitrate end, the difference between these two subsampling strategies becomes smaller as we expected.

4.5 Comparison with Other Coding Schemes

In this set of tests, the SA-DWT results are used to compare with the results of SA-DCT coding [10] and Egger's wavelet coding [22, 23], macroblock region-based wavelet coding [18]. The SA-DCT results are obtained using the MPEG-4 VM 7.0 software and Table 5 shows the conditions for the simulation, where the meanings of fields in Table 5 are as follows [3].

video_object_layer_shape indicates whether object shape is rectangular (00), gray scale shape (10) or binary shape (01); video_layer_quant_type indicates the type of quantization method selected: H. 263 quantization method (0) or MPEG-1/2 quantization method (1); intra_acdc_pred_disable indicates whether the AC/DC prediction of intra-coded blocks is disabled (0 means enabled); if separate_motion_shape_texture is 1, all the coding data (e.g. motion, shape, texture, etc.) are grouped together, if this field is 0, the coding data are grouped macroblock by macroblock; disable_sadct indicates whether to turn on SA-DCT (0) or not (1).

For the wavelet based coding schemes, the default MPEG-4 wavelet filters and the ZTE (SQ) coding scheme are used. The subsampling strategy for SA-DWT is chosen to be favoring signal processing gain.

Fig. 12 presents the PSNR-bitrate curves for these different coding schemes. Clearly, the shape adaptive wavelet coding scheme in MPEG-4 achieves the best coding efficiency. The SA-DCT achieves the second best coding efficiency which is about 1.5-2dB lower than SA-DWT. The wavelet method proposed by Egger *et al* [22, 23] is about 2-2.5dB lower than SA-DWT. The macroblock based wavelet coding method has the

Table 5: Test conditions for SA-DCT in MPEG-4 VM 7.0

video_object_layer_shape	01(binary shape)
video_object_layer_quant_type	0
intra_acdc_pred_disable	0
separate_motion_shape_texture	0
disable_sadct	0

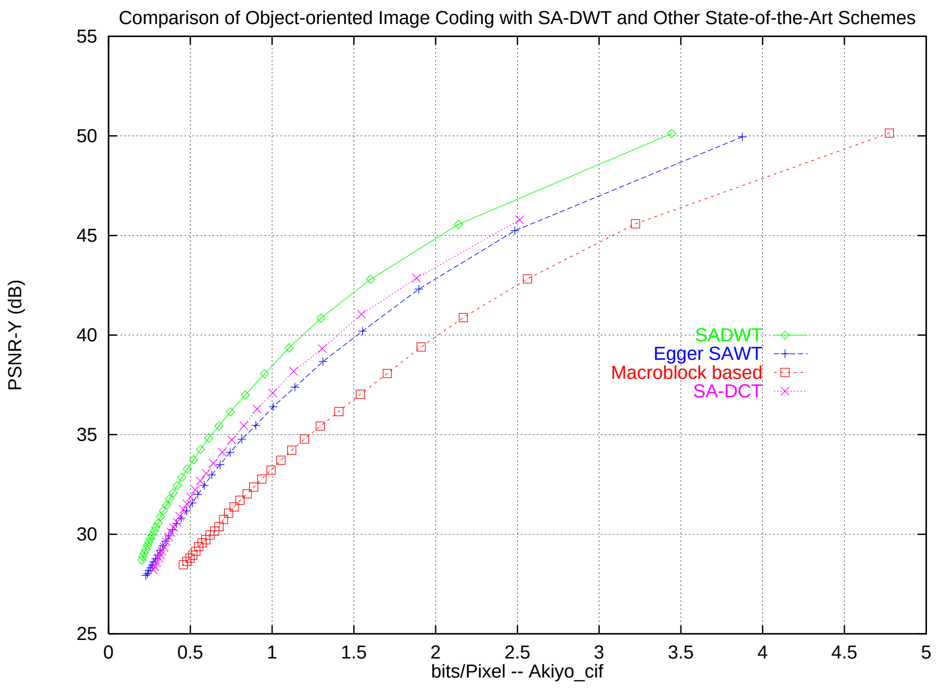


Figure 12: Comparison of SA-DWT coding with other coding schemes (Akiyo-CIF).

worst performance.

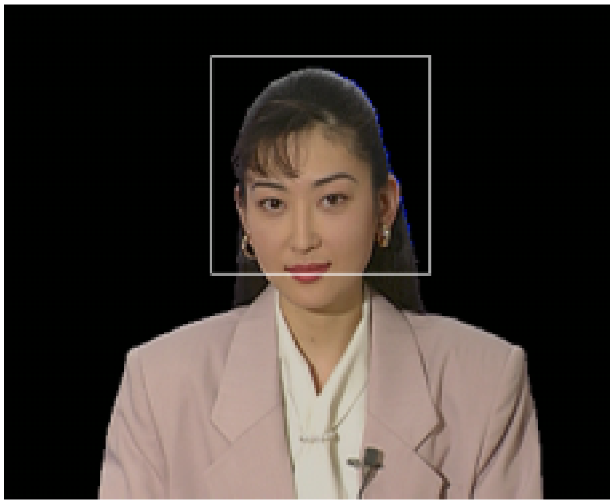
The reconstructed visual objects (Akiyo Cif) using these different schemes are given in Fig. 13 (a) through (e). Fig. 13(a) gives the original visual object to be encoded. Fig. 13(b) through Fig. 13(e) give the the enlarged portion (the white box in Fig.13(a)) of the results of the original, SA-DWT coding in MPEG-4, SA-DCT coding, and Egger's wavelet method, respectively.

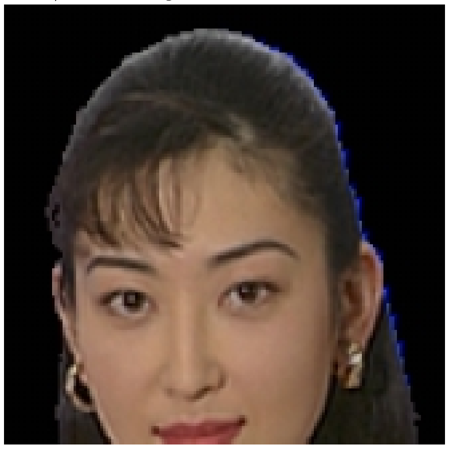
A more demanding visual object is the foreground buidlings in building image (1024x1024 size). Fig. 14 (a) through (e) give the enlargments of portion of the visual object.

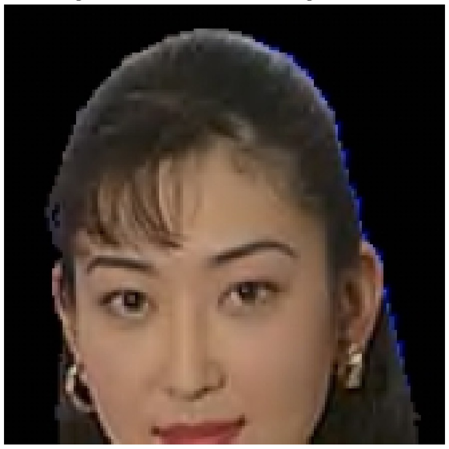
It is clearly shown in Fig. 13 and 14, that the SA-DWT scheme produces much superior visual quality than SA-DCT and Egger's wavelet schemes. SA-DCT scheme suffers the blocking artifacts and the Egger's wavelet scheme suffers more noise.

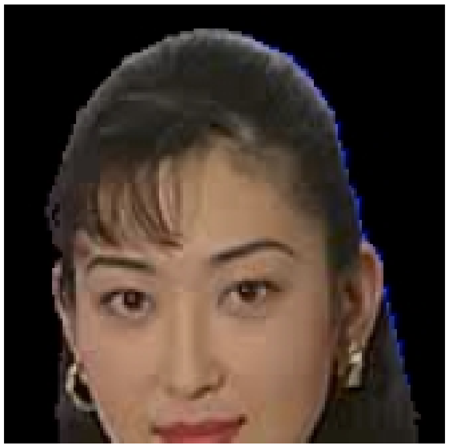
4.6 Summary

SA-DWT coding is a very efficient technique for coding arbitrarily shaped visual objects. In our simulation, odd-symmetric wavelet filters provide better coding performance than orthogonal wavelet filters and even-symmetric filters. The EZW coding and ZTE coding schemes provide more or less the same coding efficiency but different flexibilities and functionalities. In most cases, the subsampling strategy that favors the signal processing gain achieves better coding performance. However, in very low bit rate case, the subsampling strategy that favors zerotree coding gain tends to be more efficient. Compared with SA-DCT coding and Egger's wavelet coding, SA-DWT coding provides higher PSNR values and visibly better quality for all test


(a) Original visual object (Akiyo_cif) where region in the white box is to be enlarged in (b) through (e)


(b) 2x enlarged portion of the original visual object


(c) 2x enlarged portion of the reconstructed visual object at 0.9538bpp using SA-DWT coding in MPEG-4 (PSNR-Y=38.06dB; PSNR-U=43.43dB; PSNR-V=43.25dB).


(d) 2x enlarged portion of the reconstructed visual object at 1.0042bpp using SA-DCT coding (PSNR-Y=37.09dB; PSNR-U=42.14dB; PSNR-V=42.36dB).

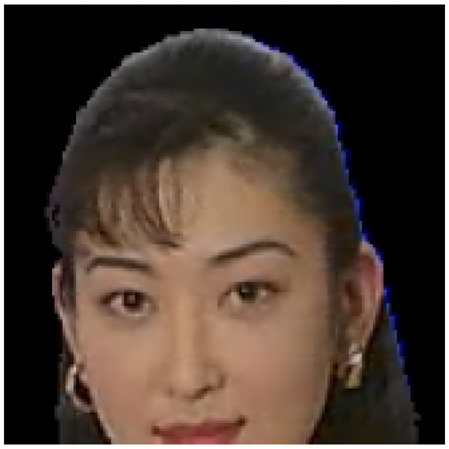
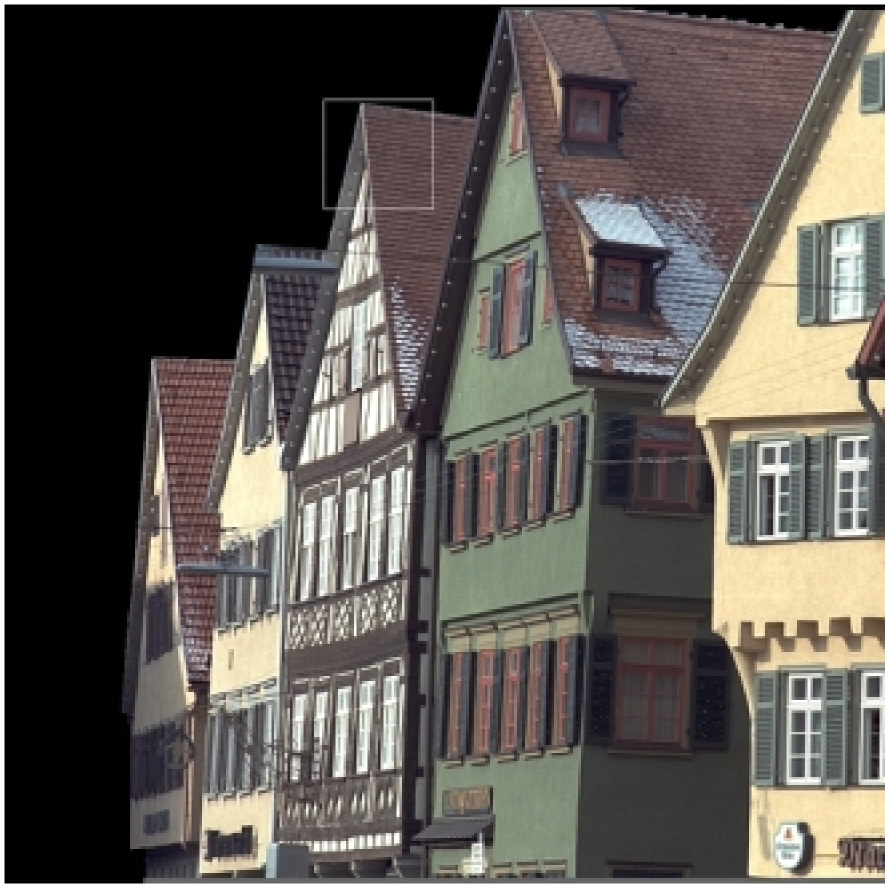
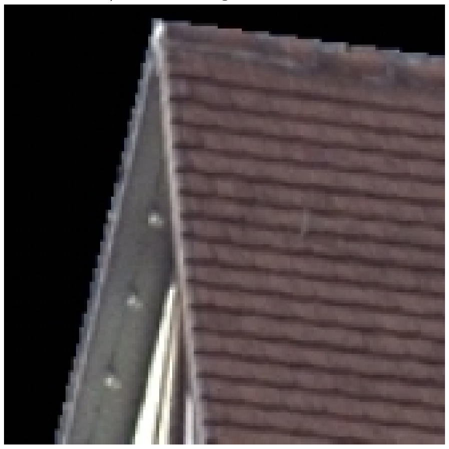
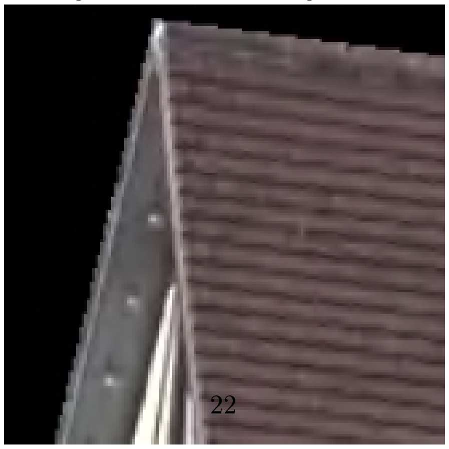

(e) 2x enlarged portion of the reconstructed visual object at 1.0065bpp using Egger's wavelet method (PSNR-Y=36.40dB; PSNR-U=42.53dB; PSNR-V=42.40dB).

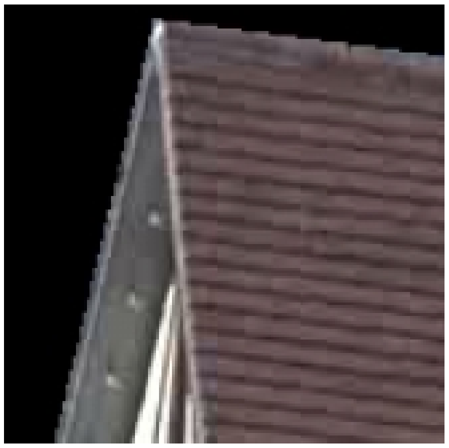
Figure 13: Original and reconstructed visual objects in the Akiyo-CIF sequence.


(a) Original visual object (Building 1024x1024) where region in the white box is to be enlarged in (b) through (e)


(b) 2x enlarged portion of the original visual object



(c) 2x enlarged portion of the reconstructed visual object at 0.8889bpp using SA-DWT coding in MPEG-4 (PSNR-Y=32.73dB; PSNR-U=42.26dB; PSNR-V=42.43dB).


(d) 2x enlarged portion of the reconstructed visual object at 0.9542bpp using SA-DCT coding (PSNR-Y=32.64dB; PSNR-U=41.89dB; PSNR-V=42.20dB).

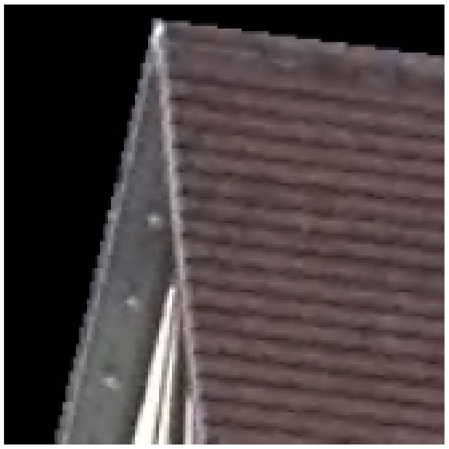

(e) 2x enlarged portion of the reconstructed visual object at 0.9227bpp using Egger's wavelet method (PSNR-Y=31.80dB; PSNR-U=42.81dB; PSNR-V=42.12dB).

Figure 14: Original and reconstructed visual objects in the building image.

sequences at all bitrate levels (in most cases, with a smaller number of total bits too).

5 CONCLUSION

This paper presents a comprehensive description of shape adaptive wavelet coding schemes for coding arbitrarily shaped visual objects. The number of wavelet coefficients after the SA-DWT is identical to the number of pixels in the arbitrarily shaped visual object. The spatial correlation and wavelet transform properties, such as locality property and self-similarity across subbands, are well preserved in the SA-DWT. For a rectangular region, the SA-DWT becomes identical to a conventional wavelet transform. Detailed descriptions of the SA-DWT techniques for three different types of wavelet filters, namely, orthogonal filters, odd symmetric biorthogonal filters, and even symmetric biorthogonal filters, are presented. The subsampling strategies for the SA-DWT coefficients are discussed. An efficient method of extending the Zerotree Coding technique to coding the SA-DWT coefficients with *don't-care* values is presented. Extensive experiment results have shown that the shape adaptive wavelet coding technique consistently performs better than SA-DCT coding and other wavelet based schemes for coding arbitrarily-shaped visual objects.

ACKNOWLEDGEMENT

The authors would like to thank Dr. Zhixiong Wu of OKI Electric Industry Co. and Dr. Hongqiao Sun of Vector Vision, Inc. for their contributions to the implementation and experiments of shape adaptive wavelet coding during the MPEG-4 process. The authors would also like to thank the reviewers for their comments that improved the presentation of the paper.

Appendix

The properties of different wavelet filters and wavelet analysis and synthesis processes are summarized as follows.

A.1 Orthogonal Filters

Assuming orthogonal wavelet filters with L taps, let the lowpass analysis filter taps be $g(i)$, $i = 0, \ldots, L-1$ and highpass analysis filter taps be $h(i)$, $i = 0, \ldots, L-1$. They have the following relationship,

$$h(i) = (-1)^{L-1-i} g(L-1-i), \text{ for } i = 0, \ldots, L-1. \tag{A-1}$$

Let the lowpass synthesis filter be $e(i)$, then

$$e(i) = g(L-1-i), \text{ for } i = 0, \ldots, L-1. \tag{A-2}$$

Let the highpass synthesis filter be $f(i)$, then

$$f(i) = h(L-1-i), \text{ for } i = 0, \ldots, L-1. \tag{A-3}$$

Note that, for orthogonal filters, the filter length L is an even number.

The analysis filtering process is given by

$$T(i) = \sum_{j=0}^{L-1} x(i+j-L/2+1) g(L-1-j) \text{ (lowpass);} \tag{A-4}$$

$$S(i) = \sum_{j=0}^{L-1} x(i+j-L/2+1) h(L-1-j) \text{ (highpass);} \tag{A-5}$$

where $T(i)$ and $S(i)$ are the lowpass band and highpass band filter outputs before subsampling, respectively. The wavelet coefficients from the analysis are obtained by subsampling the above filtering results by a factor of 2. Subsampling can be either at even positions or at odd positions and can be described as follows,

$$C(i) = T(2i - s); \tag{A-6}$$

$$D(i) = S(2i - s); \tag{A-7}$$

where $C(i)$ and $D(i)$ are the lowpass and highpass wavelet coefficients, respectively, and $s = 0$ if subsampling at even positions and $s = 1$ if subsampling at odd positions.

To perform wavelet synthesis, these coefficients are first upsampled by a factor of 2. The upsampling process is given as follows,

$$P(2i - s) = C(i); P(2i + 1 - s) = 0; \tag{A-8}$$

$$Q(2i - s) = D(i); Q(2i + 1 - s) = 0; \tag{A-9}$$

where $P(k)$ and $Q(k)$ are upsampled lowpass and highpass coefficients, respectively. Then the synthesis filtering process is given as follows:

$$u(i) = \sum_{j=0}^{L-1} P(i - L/2 + j)e(L - 1 - j) \text{ (lowpass)}; \tag{A-10}$$

$$v(i) = \sum_{j=0}^{L-1} Q(i - L/2 + j)f(L - 1 - j) \text{ (highpass)}; \tag{A-11}$$

$$r(i) = u(i) + v(i); \tag{A-12}$$

where $r(i)$ is the reconstructed signal.

A.2 Odd Symmetric Biorthogonal Filters

Assuming odd symmetric biorthogonal wavelet filters with L_g (odd) taps for the lowpass filter and L_h (odd) taps for the highpass filter, let the lowpass analysis filter taps be $g(i)$, $i = 0, \ldots, L_g - 1$ and the highpass analysis filter taps be $h(i)$, $i = 0, \ldots, L_h - 1$. They have the following properties,

$$g(i) = g(L_g - 1 - i), \text{ for } i = 0, \ldots, (L_g - 1)/2, \tag{A-13}$$

$$h(i) = h(L_h - 1 - i), \text{ for } i = 0, \ldots, (L_h - 1)/2. \tag{A-14}$$

Let the lowpass synthesis filter be $e(i)$, then

$$e(i) = (-1)^{i+1}h(i), \text{ for } i = 0, \ldots, L_h - 1; \tag{A-15}$$

Let the highpass synthesis filter be $f(i)$, then

$$f(i) = (-1)^i g(i), \text{ for } i = 0, \ldots, L_g - 1. \tag{A-16}$$

The analysis filtering process is given by

$$T(i) = \sum_{j=0}^{L_g-1} x(i + j - (L_g - 1)/2)g(L_g - 1 - j) \text{ (lowpass)}, \tag{A-17}$$

$$S(i) = \sum_{j=0}^{L_h-1} x(i + j - (L_h - 1)/2)h(L_h - 1 - j) \text{ (highpass)}, \tag{A-18}$$

where $T(i)$ and $S(i)$ are the lowpass band and highpass band filter outputs before subsampling, respectively. The wavelet coefficients from the analysis are obtained by subsampling the above filtering results by a factor of 2. Subsampling can be either at even positions or at odd positions. However, in order to use the symmetric extensions, the subsampling of lowpass coefficients and that of highpass coefficients always have one sample shift. If the subsampling positions of lowpass coefficients are even, then the sub-sampling positions of highpass coefficients should be odd, or vice versa. The subsampling process is described as follows,

$$C(i) = T(2i - s); \tag{A-19}$$

$$D(i) = S(2i + 1 - s); \tag{A-20}$$

where $C(i)$ and $D(i)$ are the lowpass and highpass wavelet coefficients, respectively. $s = 0$ if the lowpass subsampling positions are even and $s = 1$ if they are odd. Note that subsampling of highpass coefficients always has one sample advance. To perform synthesis, these coefficients are first upsampled by a factor of 2. The upsampling process is given as follows,

$$P(2i - s) = C(i); P(2i + 1 - s) = 0; \tag{A-21}$$

$$Q(2i + 1 - s) = D(i); Q(2i + s) = 0; \tag{A-22}$$

where $P(k)$ and $Q(k)$ are upsampled lowpass and highpass coefficients, respectively. Then the synthesis filtering process is given as follows:

$$u(i) = \sum_{j=0}^{L_h-1} P(i - (L_h - 1)/2 + j)e(L_h - 1 - j) \text{ (lowpass)}, \tag{A-23}$$

$$v(i) = \sum_{j=0}^{L_g-1} Q(i - (L_g - 1)/2 + j)f(L_g - 1 - j) \text{ (highpass)}, \tag{A-24}$$

$$r(i) = u(i) + v(i), \tag{A-25}$$

where $r(i)$ is the reconstructed signal.

A.3 Even Symmetric Biorthogonal Filters

Assuming even symmetric biorthogonal wavelet filters with L_g (even) taps for the lowpass filter and L_h (even) taps for the highpass filter, let the lowpass analysis filter taps be $g(i)$, $i = 0, \ldots, L_g - 1$ and the highpass analysis filter taps be $h(i)$, $i = 0, \ldots, L_h - 1$. They have the following properties,

$$g(i) = g(L_g - 1 - i), \text{ for } i = 0, \ldots, L_g/2 - 1, \tag{A-26}$$

$$h(i) = h(L_h - 1 - i), \text{ for } i = 0, \ldots, L_h/2 - 1. \tag{A-27}$$

Let the lowpass synthesis filter be $e(i)$, then

$$e(i) = (-1)^{i+1}h(i), \text{ for } i = 0, \ldots, L_h - 1. \tag{A-28}$$

Let the highpass synthesis filter be $f(i)$, then

$$f(i) = (-1)^{i}g(i), \text{ for } i = 0, \ldots, L_g - 1. \tag{A-29}$$

The analysis filtering process is given by

$$T(i) = \sum_{j=0}^{L_g-1} x(i + j - L_g/2 + 1)g(L_g - 1 - j) \text{ (lowpass)}, \tag{A-30}$$

$$S(i) = \sum_{j=0}^{L_h-1} x(i + j - L_h/2 + 1)h(L_h - 1 - j) \text{ (highpass)}, \tag{A-31}$$

where $T(i)$ and $S(i)$ are the lowpass band and highpass band filter outputs before subsampling, respectively. The wavelet coefficients from the analysis are obtained by subsampling the above filtering results by a factor of 2. Subsampling can be either at even positions or at odd positions. The subsampling process is described as follows,

$$C(i) = T(2i - s); \tag{A-32}$$

$$D(i) = S(2i - s); \tag{A-33}$$

where $C(i)$ and $D(i)$ are the lowpass and highpass wavelet coefficients, respectively. $s = 0$ if the subsampling positions for lowpass and highpass coefficients are even and $s = 1$ otherwise. To perform synthesis, these wavelet coefficients are first upsampled by a factor of 2. The upsampling process is given as follows,

$$P(2i - s) = C(i); P(2i + 1 - s) = 0; \tag{A-34}$$

$$Q(2i - s) = D(i); Q(2i + 1 - s) = 0; \tag{A-35}$$

where $P(k)$ and $Q(k)$ are upsampled lowpass and highpass wavelet coefficients, respectively. Then the synthesis filtering process is given as follows,

$$u(i) = \sum_{j=0}^{L_h-1} P(i - L_h/2 + j)e(L_h - 1 - j) \text{ (lowpass)}, \tag{A-36}$$

$$v(i) = \sum_{j=0}^{L_g-1} Q(i - L_g/2 + j)f(L_g - 1 - j) \text{ (highpass)}, \tag{A-37}$$

$$r(i) = u(i) + v(i) \tag{A-38}$$

where $r(i)$ is the reconstructed signal.

References

[1] R. Koenen (Editor), "Overview of the MPEG-4 Standard," ISO/IEC JTC/SC29/WG21, MPEG-99-N2925, March 1999, Seoul, South Korea.

[2] ISO/IEC JTC1-SC29-WG11, "INFORMATION TECHNOLOGY - GENERIC CODING OF AUDIO-VISUAL OBJECTS, Part 2: Visual," ISO/IEC 14496-2, Final Draft of International Standard, Dec. 18, 1998.

[3] ISO/IEC JTC1-SC29-WG11, "MPEG-4 Video Verification Model Version 7.0," MPEG97-N1642, April 1997.

[4] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Transactions on Signal Processing*, Vol. 41, No. 12, Dec., 1993.

[5] S. A. Martucci, I. Sodagar, T. Chiang, Y.-Q. Zhang, "A Zerotree Wavelet Video Coder," *IEEE Transactions on Circuit and Systems for Video Technology: special issue on MPEG-4*, Vol. 7, No. 1, pp.109-118, Feb. 1997.

[6] W. Li, H. Q. Cao, S. Li, F. Ling, S. A. Segan, H. Sun, J. P. Wus, Y.-Q. Zhang, "A video coding algorithm using vector-based techniques," *IEEE Transactions on Circuit and Systems for Video Technology: special issue on MPEG-4*, Vol. 7, No. 1, pp.146-157, Feb. 1997.

[7] S.-F. Chang and D. G. Messerschmitt, "Transform coding of arbitrarily-shaped image segments," in *Proc. 1st ACM Int. Conf. Multimedia*, Anaheim, CA, 1993, pp. 83-90.

[8] M. Gilge, T. Engelhardt, R. Mehlan, "Coding of arbitrarily shaped image segments based on a generalized orthogonal transform," *Signal Process: Image Comm.*, Vol. 1, pp. 153-180, Oct. 1989.

[9] T. Sikora and B. Makai, "Low complex shape-adaptive DCT for generic and functional coding of segmented video," in *Proc. Workshop Image Anal. Image Coding*, Berlin, FRG, Nov. 1993.

[10] T. Sikora and B. Makai, “Shape-adaptive DCT for generic coding of video,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 1, pp. 59-62, Feb. 1995.

[11] T. Sikora, S. Bauer and B. Makai, “Efficiency of shape-adaptive 2-D transforms for coding of arbitrarily shaped image segments,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 3, June 1995.

[12] M. Bi, S. H. Ong and Y. H. Ang, “Comment on ‘Shape-adaptive DCT for generic coding of video’, ” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 6, pp. 686-688, Dec. 1996.

[13] P. Kauff and K. Schuur, “Shape-adaptive DCT with block-based DC separation and Delta DC correction,” *IEEE Trans. Circuits and Systems for Video Technology*, vol. 8, no. 3, pp. 237-242, June 1998.

[14] C.-S. Park, D.-D. Hwang, J.-T. Lim, H.-S. Kim, K.-H. Chang, and S.-D. Kim, “Daewoo proposal for region texture coding,” *ISO/IEC JTC/SC29/WG11, MPEG-95-m0555*, Jan. 1996.

[15] E. Jensen, K. Rijk, I. Lagendijk, P. van Beek, “Coding of arbitrarily shaped image segments,” *Proceedings of workshop on image analysis and synthesis in image coding*, Oct. 1994, Berlin, Germany.

[16] T. Aono, N. Ito, S. Watanabe and H. Kusao, “Object-based wavelet transform (OWT) tool for video,” *ISO/IEC JTC/SC29/WG11, MPEG-95-m0378*, Oct. 1995.

[17] T. Aono, N. Ito, H. Katata and H. Kusao, “Wavelet based video coding algorithm for object scalability,” *ISO/IEC JTC/SC29/WG11, MPEG-96-m0583*, Jan. 1996.

[18] J. Muller (editor), “Core-Experiments on MPEG-4 Video - Efficient Coding,” *ISO/IEC JTC/SC29/WG11, MPEG-96-N1250*, March 1996.

[19] S. Li, W. Li, F. Ling, H. Sun, J. P. Wus, “Shape adaptive vector wavelet coding of arbitrarily shaped texture,” *ISO/IEC JTC/SC29/WG11, MPEG-96-m1027*, June 1996.

[20] W. Li, F. Ling, H. Sun, “Report on Core Experiment O3 (Shape adaptive wavelet coding of arbitrarily shaped texture),” *ISO/IEC JTC/SC29/WG11, MPEG-97-m2385*, July 1997.

[21] S. Li and W. Li, ”Shape Adaptive Discrete Wavelet Transform for Coding Arbitrarily Shaped Texture,” *Proceedings of SPIE – VCIP’97*, Vol. 3024, San Jose, Feb 12-14, 1997.

[22] O. Egger, P. Fleury and T. Ebrahimi, “Shape-Adaptive Wavelet Transform for Zerotree Coding,” *Proceedings of the European Workshop on Image Analysis and Coding for TV, HDTV and Multimedia Application*, pp. 201-208, Rennes, France, Feb. 1996.

[23] O. Egger, “Region Representation Using Nonlinear Techniques with applications to image and video coding,” Ph. D. Dissertation, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, 1997.

[24] S. Li, W. Li, H. Sun and Z. Wu, “Shape Adaptive Wavelet Coding,” *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. 5, pp. 281-284, ISCAS’98, Monterey, California, May 1998.

[25] Z. Wang, “Fast Algorithms for Discrete W Transform and for the Discrete Fourier Transform,” *IEEE Trans. on Acoustics, Speech and Signal Processing*, Vol. ASSP-32, No. 4, Aug. 1984.