Flexible, highly scalable, object-based wavelet image compression algorithm for network applications

H. Danyali and A. Mertins

Abstract: The authors propose a highly scalable image compression scheme based on the set partitioning in hierarchical trees (SPIHT) algorithm. The proposed algorithm, called highly scalable SPIHT (HS-SPIHT), adds the spatial scalability feature to the SPIHT algorithm through the introduction of multiple resolution-dependent lists and a resolution-dependent sorting pass. It keeps the important features of the original SPIHT algorithm such as compression efficiency, full SNR scalability and low complexity. The flexible bitstream of the HS-SPIHT encoder can easily be adapted to various resolution requirements at any bit rate. The parsing process can be carried out on-the-fly without decoding the bitstream by a simple parser (transcoder) that forms a part of a smart network. The HS-SPIHT algorithm is further developed for fully scalable coding of arbitrarily shaped visual objects. The proposed highly scalable algorithm finds applications in progressive web browsing, visual databases and especially in image transmission over heterogeneous networks.

1 Introduction

Traditional image coding systems have only focused on efficient compression of image data The main objective of such systems is optimising image quality at a given bit rate. Owing to the explosive growth of the Internet and networking technology, nowadays a huge number of users with different processing capabilities and network access bandwidth can easily communicate. For transmission of visual data on such a heterogenous network, efficient compression alone is not sufficient. The main challenge here is how a single flexible bitstream can be provided to optimally service various users according to their bandwidth and computing capabilities. Scalable image coding is a response to this challenge. A scalable image coder generates a bitstream that consists of a set of embedded parts that offer increasingly better signal-to-noise ratio (SNR) or/and higher spatial resolution. Different parts of this bitstream can be selected and decoded by a scalable decoder to meet certain quality or/and resolution requirements. In the case of an entirely scalable bitstream, different types of decoders with different complexity and access bandwidth can coexist. While low performance decoders only decode a small portion of the bitstream and reconstruct a low quality and/or low resolution version of the encoded image, higher performance decoders have an opportunity to receive and decode further portions of the bitstream in order to achieve higher quality and/or higher resolution.

The two types of scalability in the case of image coding are SNR scalability and spatial scalability. SNR scalability is a feature in the encoded bitstream that allows decoders to decode the image in the same spatial resolution but with different fidelity. A fully SNR scalable bitstream, also known as an embedded bitstream, can be truncated at any point to achieve the best possible reconstruction for the number of bits received. Spatial or resolution scalability, on the other hand, is a feature in the encoded bitstream that allows decoders to decode the image with different spatial resolutions.

Over the past decade wavelet-based image/video compression schemes have become increasingly important and gained widespread acceptance. An example is the new JPEG2000 still image compression standard [1]. Because of their inherent multiresolution signal representation, wavelet-based coding schemes have the potential to support both SNR and spatial scalability. Among the wavelet-based image coding methods the class of so-called embedded wavelet image coding provides complete SNR scalability. In this class, Shapiro [2] pioneered the embedded zerotree wavelet (EZW) coding scheme based on the idea of grouping spatially coefficients at different scales to trees and efficiently predicting zero coefficients across scales. Many researchers have since worked on variations of the original zerotree method to achieve further improvements [3–9]. Almost all of these improvements are about compression efficiency. An improved version of the EZW algorithm that uses an improved symbol set for zerotree encoding, and proper syntax and markers for the compressed bitstream to allow extraction of various qualities and resolutions was reported in [7]. However, the decoder needs some additional side information to decode the bitstream. Tham *et al.* [10] introduced a new zerotree structure called tri-zerotree and used a layered coding strategy with the concept of embedded resolution block coding to achieve a high degree of scalability for video coding.

An important development of the EZW algorithm, called set partitioning in hierarchical trees (SPIHT) has been introduced by Said and Pearlman [6]. (SPIHT will be briefly explained in Section 2.) The success of the algorithm in compression efficiency and simplicity makes it well known

IEE Proceedings online no. 20040734

doi: 10.1049/ip-vis:20040734

Paper first received 2nd October 2003 and in revised form 16th May 2004

H. Danyali is with the Department of Electrical Engineering, University of Kurdistan, Sanandaj, Iran

A. Mertins is with the Signal Processing Group, Institute of Physics, University of Oldenburg, 26111 Oldenburg, Germany

as a benchmark for embedded wavelet image coding. Further improvements of SPIHT have been reported [11–16]. Although the SPIHT coder is fully SNR scalable with excellent compression properties, it does not support spatial scalability and does not provide a bitstream that can be parsed easily according to the resolution and rate desired by a decoder. A multiresolution encoding and decoding scheme based on the SPIHT algorithm is reported in [13, 17]. In order to achieve multiresolution decoding from a single bitstream, this method partitions the encoded bitstream into portions according to their dependency on the wavelet subbands. This is done by putting flags in the bitstream during the process of decoding, when the whole bitstream is scanned and only the portions that correspond to the spatial location defined by the decoder's desired resolution are marked and decoded. Although this multiresolution decoding scheme saves decoding time for lower resolutions, it needs to receive the whole bitstream at the full resolution even for decoding lower resolutions. For multiresolution encoding the coder in [13, 17] offers a layered bitstream. The major drawback of this multiresolution encoding is that it imposes a high number of unnecessary bits on the lowest resolution layer, which are actually not required for decoding at that resolution, and therefore causes a very high rate (bit per pixel) for full coding of the lowest resolution. Moreover, only the lowest resolution layer of the bitstream is rate embedded, and the bitstream is not parsable to obtain rate embedded bitstreams for any desired resolution.

In this paper, a fully scalable image coding scheme based on the SPIHT algorithm is presented. We modify the SPIHT algorithm to support both spatial and SNR scalability features, while keeping its compression efficiency and low complexity. The encoder creates a bitstream that can easily be parsed to achieve different levels of resolution or/and quality requested by the decoder.

2 Set partitioning in hierarchical trees (SPIHT)

The SPIHT algorithm defines and partitions sets in the wavelet decomposed image using a special data structure called a 'spatial orientation tree'. A spatial orientation tree is a group of wavelet coefficients organised into a tree rooted in the lowest frequency (coarsest scale) subband with offspring in several generations along the same spatial

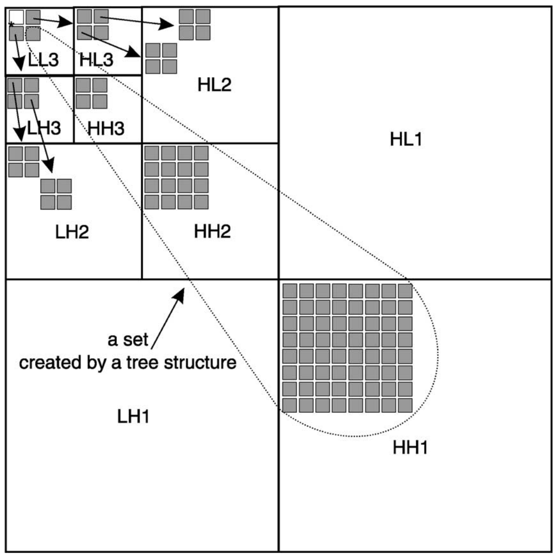


Fig. 1 *Parent–children dependency and spatial orientation trees across wavelet subbands in SPIHT*

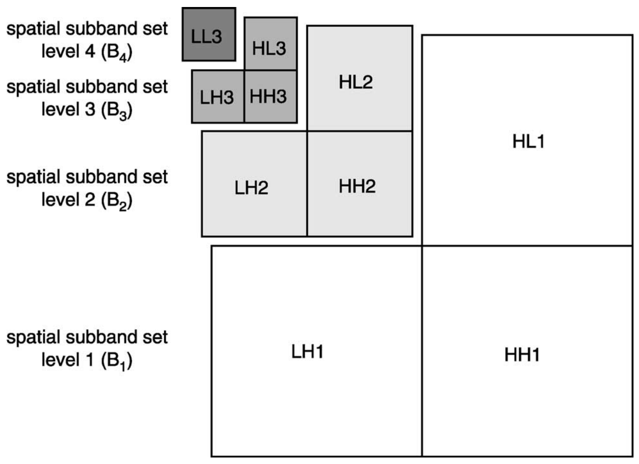
orientation in the higher frequency subbands. Figure 1 shows a spatial orientation tree and the parent–children dependency defined by the SPIHT algorithm across subbands in the wavelet image. The tree is defined in such a way that each node has either no offspring (the leaves) or four offspring at the same spatial location in the next finer subband level. The pixels in the lowest frequency subband are grouped into blocks of 2×2 adjacent pixels, and in each block one of them, which is marked by $*$ in Fig. 1, has no descendants.

The SPIHT algorithm consists of three stages: initialisation, sorting and refinement. It sorts the wavelet coefficients into three ordered lists: the list of insignificant sets (LIS), the list of insignificant pixels (LIP), and the list of significant pixels (LSP). At the initialisation stage the SPIHT algorithm first defines a start threshold based on the maximum value in the wavelet pyramid, then sets the LSP as an empty list and puts the coordinates of all coefficients in the coarsest level of the wavelet pyramid (i.e. the lowest frequency band; LL band) into the LIP and those which have descendants also into the LIS. In the sorting pass, the algorithm first sorts the elements of the LIP and then the sets with roots in the LIS. For each pixel in the LIP it performs a significance test against the current threshold and outputs the test result to the output bitstream. All test results are encoded as either 0 or 1, depending on the test outcome, so that the SPIHT algorithm directly produces a binary bitstream. If a coefficient is significant, its sign is coded and then its coordinate is moved to the LSP. During the sorting pass of LIS, the SPIHT encoder carries out the significance test for each set in the LIS and outputs the significance information. If a set is significant, it is partitioned into its offspring and leaves. Sorting and partitioning are carried out until all significant coefficients have been found and stored in the LSP. After the sorting pass for all elements in the LIP and LIS, SPIHT does a refinement pass with the current threshold for all entries in the LSP, except those which have been moved to the LSP during the last sorting pass. Then the current threshold is divided by two and the sorting and refinement stages are continued until a predefined bit-budget is exhausted. Details of SPIHT algorithms are presented in [6].

3 Highly scalable SPIHT (HS-SPIHT)

3.1 HS-SPIHT concepts

In general, applying N levels of wavelet decomposition to an image allows at most $N+1$ levels of spatial resolution. To distinguish between different resolution levels, we denote the lowest spatial resolution level as level $N+1$. The full image then becomes resolution level 1. Thus the actual spatial resolution related to level k is $1/2^{k-1}$ of the resolution of the original image. The three subbands (HL_k, LH_k, HH_k) that need to be added to increase the spatial resolution from level $k+1$ to level k are called 'spatial subband set' level k and referred to as B_k (see Fig. 2). An algorithm that provides full spatial scalability would encode the different resolution subbands levels separately, allowing a transcoder or decoder to directly access the data needed to reconstruct with a desired spatial resolution. The original SPIHT algorithm, however, processes the entire wavelet tree in each bitplane coding level and produces a bitstream that contains the information about the different spatial resolutions in no particular order.



spatial subband set level 4 (B_4)
spatial subband set level 3 (B_3)
spatial subband set level 2 (B_2)
spatial subband set level 1 (B_1)
LL3
HL3
LH3
HH3
HL2
LH2
HH2
HL1
LH1
HH1

Fig. 2 *Different spatial resolution subband levels that need to be coded separately to provide spatial scalability*

For adding spatial scalability to the SPIHT algorithm, we define the following concepts:

(i) all coding information related to a given level of spatial subband set (see Fig. 2), in a given bitplane level must be distinguishable in the output bitstream.
(ii) the tree structure distributed among different resolution subbands in the wavelet image, must be kept in the same way as for SPIHT to keep the compression efficiency of SPIHT.
(iii) the proposed algorithm must be reversible, such that there will be no need to send extra information to the decoder to recognise each coding part.

The first concept is to have a flexible bitstream which allows reordering for various quality and spatial resolutions, while the second and third concepts keep the compression efficiency of the original SPIHT algorithm.

3.2 HS-SPIHT description

In [18] the SPIHT algorithm was modified to support spatial scalability by adding a new list called the list of delayed insignificant sets (LDIS) to the SPIHT lists and modifying the SPIHT sorting pass. The highly scalable SPIHT (HS-SPIHT) algorithm proposed in this paper solves the spatial scalability problem through the introduction of multiple resolution-dependent lists and a resolution-dependent sorting pass. For each spatial subband set, B_k (see Fig. 2), a set of LIP, LSP and LIS lists is defined, therefore there are LIP$_k$, LSP$_k$, and LIS$_k$ for $k = k_{\max}\ k_{\max} - 1, \ldots, 1$ where $k_{\max}$ is the maximum number of spatial resolution levels supported by the encoder. In each bitplane, the HS-SPIHT coder starts encoding from the maximum resolution level ($k_{\max}$) and proceeds to the lowest level (level 1). For the resolution-dependent sorting pass of the lists that belong to level k, the algorithm first does the sorting pass for the coefficients in the LIP$_k$ in the same way as SPIHT and then processes the LIS$_k$ list. During processing the LIS$_k$, sets that lie outside the resolution level k are moved to the LIS$_{k-1}$. After the algorithm has finished the sorting and refinement passes for level k it will do the same procedure for the lists related to level $k-1$. According to the magnitude of the coefficients in the wavelet pyramid, coding of higher resolution bands usually starts from lower bitplanes. The total number of bits belonging to a particular bitplane is the same for SPIHT and HS-SPIHT, but HS-SPIHT arranges them according to their spatial resolution dependency.

In the following, the sets and symbols required for HS-SPIHT are defined and the entire coding algorithm is listed.

3.3 HS-SPIHT coding algorithm

Definitions:
- $c(i,j)$: wavelet transformed coefficient at coordinate (i,j).
- $O(i,j)$: set of coordinates of all offspring of node (i,j) (the same as for SPIHT, see Fig. 1).
- $D(i,j)$: set of coordinates of all descendants of node (i,j).
- $L(i,j)$: set of coordinates of all leaves of node (i,j). $L(i,j) = D(i,j) - O(i,j)$.
- H: set of coordinates of all nodes in the coarsest level of wavelet coefficients pyramid.
- $S_n(i,j)$: significance test of a set of coordinates $\{(i,j)\}$ at bitplane n

$$Sn(i,j) = \begin{cases} 1 & \text{if } \max_{\{(i,j)\}}\{|c(i,j)|\} \geq 2^n \\ 0 & \text{otherwise} \end{cases}$$

- type A sets: for sets of type A the significance tests are to be applied to all descendants $D(i,j)$.
- type B sets: for sets of type B the significance tests are to be applied only to the leaves $L(i,j)$.
- $n_{\max}$: maximum bitplane level required for coding $n_{\max} = \lfloor \log_2(\max_{\{(i,j)\}}\{|c(i,j)|\}) \rfloor$.
- $k_{\max}$: maximum level of spatial scalability to be supported by the bitstream $(1 \leq k_{\max} \leq N+1)$, where N is the number of wavelet decomposition levels applied to the image.
- B_k: spatial subband set level k

$$B_k = \begin{cases} \{LL_{k-1}\} & \text{if } k = N+1 \\ \{HL_k,\ LH_k,\ HH_k\} & 1 \leq k \leq N \end{cases}$$

- R_k: a set of subbands in the decomposition image which belongs to spatial resolution level k $(1 \leq k \leq k_{\max})$ of the image.

$$R_k = \{B_{N+1}, B_N, \ldots B_{k+1}, B_k\}$$

- LIP$_k$: list of insignificant pixels belong to B_k.
- LSP$_k$: list of significant pixels belong to B_k.
- LIS$_k$: list of insignificant sets need to be processed during coding of B_k in each bitplane.

HS-SPIHT coding steps

Step 1. Initialisation

- $n = n_{\max}$; and output it;
- LSP$_k = \emptyset,\ \forall k,\ 1 \leq k \leq k_{\max}$;
- $$\text{LIP}_k = \begin{cases} \emptyset & \text{for } 1 \leq k < k_{\max} \\ \{(i,j)\},\ \forall(i,j) \in H & k = k_{\max} \end{cases}$$
- LIS$_k = \emptyset,\ \forall k,\ 1 \leq k < k_{\max}$;
- LIS$_{k_{\max}} = \{(i,j)\}$ as type A, $\forall(i,j) \in H$ which have descendants;
- $k = k_{\max}$;

Step 2. Resolution-dependent sorting pass
- SortLIP-Enc(n, k);
- SortLIS-Enc(n, k);

Step 3. Refinement pass
- RefineLSP-Enc(n, k);

Step 4. Resolution scale update
- if $(k > 1)$
  - $k = k - 1$;
  - go to step 2;
- else, $k = k_{\max}$;

Step 5. Quantisation-step update
- if $(n > 0)$

```
    – $n = n - 1$;
    – go to step 2;
  • else, end of coding.
```

Pseudo-code

```
SortLIP-Enc(k, n){
  • for each entry $(i, j)$ in the $LIP_k$ do:
    – output $S_n(i,j)$;
    – if $(S_n(i,j) = 1)$, then move $(i, j)$ to the $LSP_k$, output
      the sign of $c(i, j)$;
}

SortLIS-Enc(n, k){
  for each entry $(i, j)$ in the $LIS_k$
    • if the entry is of type A
      – if $(\forall(x,y) \in D(i,j) : (x,y) \notin R_k)$, then move $(i, j)$
        to $LIS_{k-1}$ as type A;
      – else
        * output $S_n(D(i,j))$;
        * if $(S_n(D(i,j)) = 1)$ then for each $(p,q) \in O(i,j)$
          · output $S_n(p,q)$;
          · if $(S_n(p,q) = 1)$, add $(p, q)$ to the $LSP_k$,
            output the sign of $c(p, q)$;
          · else, add $(p, q)$ to the end of the $LIP_k$;
        * if $(L(i,j) \neq \emptyset)$, move $(i,j)$ to the end of the $LIS_k$
          as an entry of type B;
        * else, remove entry $(i, j)$ from the $LIS_k$;
    • if the entry is of type B
      – if $(\forall(x,y) \in L(i,j) : (x,y) \notin R_k)$, then move $(i, j)$
        to $LIS_k - 1$ as type B;
      – else
        * output $S_n(L(i,j))$;
        * if $(S_n(L(i,j)) = 1)$
          · add each $(p,q) \in O(i,j)$ to the end of the
            $LIS_k$ as an entry of type A;
          · remove $(i, j)$ from the $LIS_k$.
}

RefineLSP-Enc(n, k){
  • for each entry $(i, j)$ in the $LIS_k$, except those included
    in the last sorting pass (i.e. the ones with the same $n$),
    output the $n$th most significant bit of $|c(i,j)|$.
}
```

4 Bitstream formation, parsing and decoding

The structure of the bitstream generated by the HS-SPIHT encoder is shown in Fig. 3. The bitstream is constructed of different parts according to the different bitplane levels (P^n). Inside each bitplane code part, the bits that belong to the different spatial subband sets P_k^n are separable. A header at the beginning of the bitstream identifies the number of spatial resolution levels supported by the encoder, as well as information such as the image dimension, number of wavelet decomposition levels, and the maximum bitplane level required for coding. At the beginning of each bitplane there is an additional header that provides the information required for identifying the individual resolution code parts (P_k^n).

The HS-SPIHT encoder bitstream can easily be reordered for multiresolution decoding at any desirable bit rate. This feature is necessary for visual information transmission specially over a heterogeneous network where visual information needs to be multicasted to a variety of different users with different capabilities and network bandwidth access based on the scenario of encoding-once, decoding multiple-times as is illustrated in Fig. 4. In such cases, the original image is first encoded by the scalable encoders at a high bit rate. The bitstream is then stored on an image server. Different users with different resolution and bit rate requirements send their request to the server, and the server or a parser (transcoder) within the network, provides them with a properly tailored bitstream that is easily obtained by selecting the related parts of the original bitstream and ordering them in such a way that the user request is fulfilled.

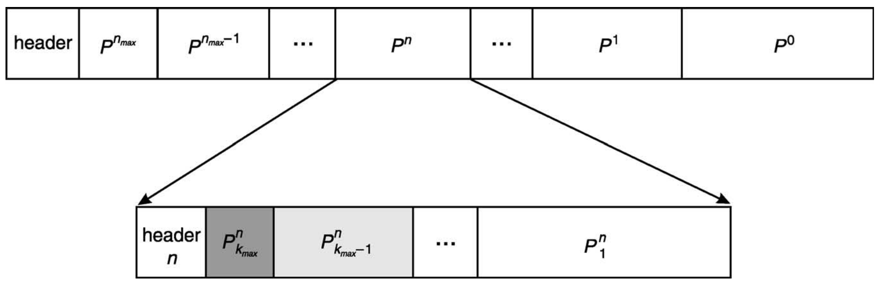


Fig. 3 *Structure of HS-SPIHT encoder bitstream which is progressive by quality*
P_k^n is related to code part of spatial subband set level k (B_k) at bitplane level n

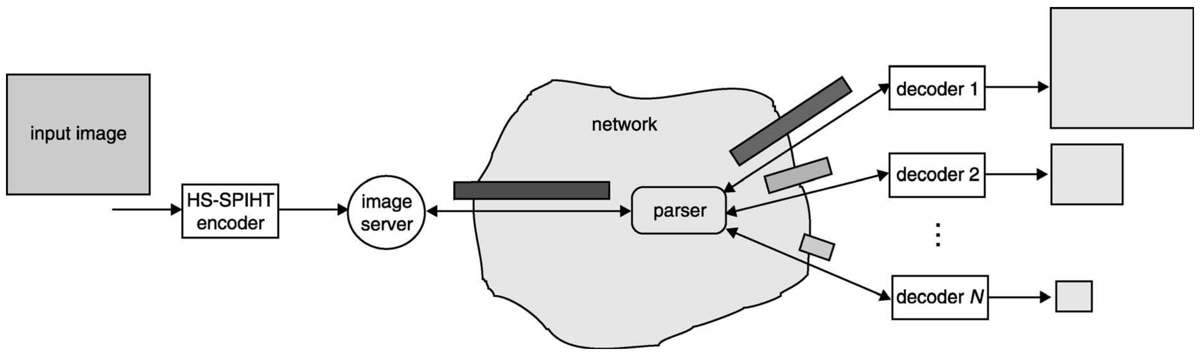


Fig. 4 *Example of image server and parser in network providing various bitstreams for different resolutions or/and quality levels requested by different users*

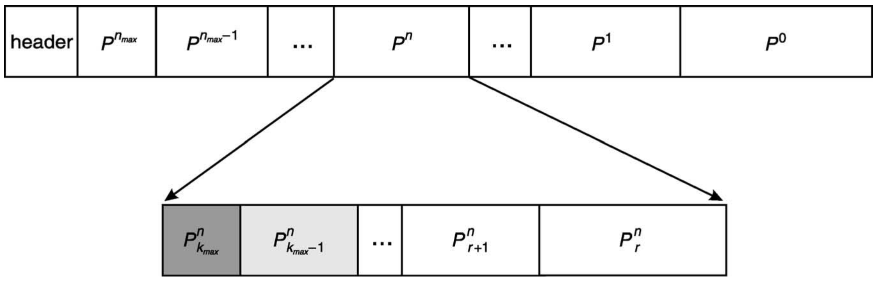


Fig. 5 *Reordered HS-SPIHT bitstream for spatial resolution level r*

Figure 5 shows an example of a reordered bitstream for spatial resolution level r. In each bitplane only the parts that belong to the spatial resolution levels greater or equal to the requested level are kept and the other parts are removed. All header information at the beginning of each bitplane code part (P^n) are used solely by the image parser and do not need to be sent to the decoder.

For decoding the parsed bitstream, the decoder directly follows the encoder, with the output command replaced by an input command, similar to the original SPIHT algorithm, but it needs to keep track of the various resolution-dependent lists LIP_k, LSP_k, and LIS_k for $k = k_{\max}, k_{\max} - 1, \ldots, r$ where r is the required resolution level. Note that the total storage requirement for all lists LIP_k, LSP_k, and LIS_k for $k = k_{\max}, k_{\max} - 1, \ldots, r$ is less (for $r > 1$) or the same as (for $k = 1$) for the LIS, LIP, and LSP used by the SPIHT encoder.

5 Object-based HS-SPIHT (OBHS-SPIHT) algorithm

In this Section, the proposed HS-SPIHT coding algorithm is further developed for efficient highly scalable texture coding of visual objects. The developed algorithm, called object-based HS-SPIHT (OBHS-SPIHT), only encodes wavelet coefficients that belong to the wavelet decomposed object. The object is assumed to have any arbitrary shape. It is also assumed that the shape information of the object is available at both the encoder and the decoder. In the following, the modifications that need to be done to turn HS-SPIHT into OBHS-SPIHT are listed.

(i) Modification of the initialisation stage

- Only the coordinates of the coefficients that are located inside the decomposed mask (see Fig. 6) in the lowest frequency band are put into the $\mathrm{LIP}_{k_{\max}}$ (the LIP related to the maximum level of spatial subband set).
- Only the root of those sets that are not entirely outside the decomposed mask (e.g. S_2 and S_3 sets in Fig. 6) are put into the $\mathrm{LIS}_{k_{\max}}$ (the LIS related to the maximum level of spatial subband set).

(ii) Modification of the sorting pass

- In the sorting pass of LIS_k, the significant test is only done for the sets that at least partly belong to the decomposed mask.
- If the related set to an entry of type A in the LIS_k is partly within the decomposed mask (e.g. set S_2in Fig. 6), and it is known to be significant, those offspring of the set root which are located outside the decomposed mask will be discarded.
- If the related set to an entry of type B in the LIS_k belongs to the decomposed mask, and is known to be significant,

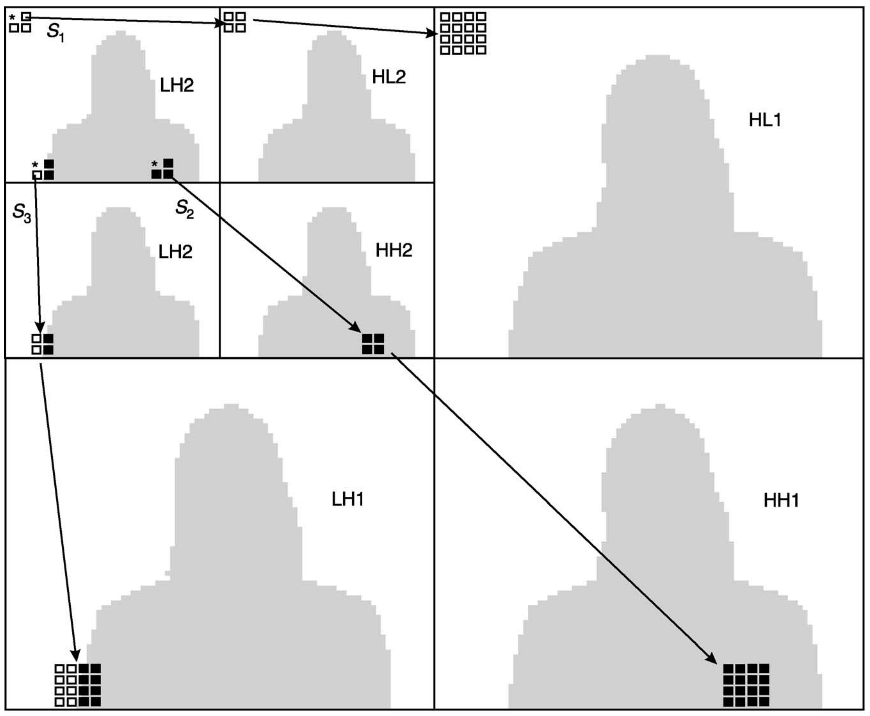


Fig. 6 *Orientation of wavelet decomposed mask of arbitrarily shaped object, and three different sets across wavelet subbands*

Set S_1 is completely outside the decomposed mask, while set S_2 is entirely within the object, and set S_3 is partly inside the region. All set roots are located in lowest frequency band only those offspring of the set root which are located inside the decomposed mask are considered as new set roots, and added to the end of LIS_k.

The aforementioned modifications guarantee that all entries in the LIP_k and LSP_k for all k $(1 \leq k \leq k_{\max})$ are located inside the decomposed mask, therefore no modifications of the LIP sorting pass and LSP refinement pass are required.

Having the shape information and the resulting decomposed shape mask, the decoder is able to take the same steps as the encoder without requiring any kind of overhead information.

6 Experimental results

6.1 HS-SPIHT results

The HS-SPIHT coding algorithm and the original SPIHT algorithm were fully implemented in software. The popular 8 bits per pixel, greyscale test images, ‘Lena’, ‘Barbara’ and ‘Goldhill’ were utilised in the simulation. The original resolution of these images is 512×512 pixels. We adopted Daubechies 9/7 filter banks [19] with symmetric extension at the image boundaries for wavelet analysis/synthesis of the images. Five levels of wavelet decomposition were first applied to each test image, then the HS-SPIHT encoder was set to encode the entire wavelet pyramid from bitplane maximum to bitplane 0 with the maximum number of spatial scalability levels (with 5 levels of wavelet decomposition, the maximum spatial scalability levels is 6).

To provide numerical results for multiresolution decoding, the HS-SPIHT bitstream, generated by the HS-SPIHT encoder, was fed into the parser to produce progressive (by quality) bitstreams for different spatial resolutions. The bitstream for each spatial resolution was decoded by the HS-SPIHT decoder at different rates and the fidelity was measured by the peak signal-to-noise ratio defined as

$$PSNR = 10 \log_{10} \frac{PEAK^2}{MSE} \text{dB}$$

where MSE is the mean squared error between the original and the reconstructed image, and $PEAK$ is the maximum possible magnitude for a pixel inside the image. The $PEAK$ value is 255 for an 8 bits/pixel original image (level 1) and $255 \times 2^{k-1}$ for resolution level k. This is due to the fact that resolution level k is obtained from the original image after applying $k-1$ levels of 2-D wavelet decomposition with filters having a DC amplification of $\sqrt{2}$. The bit rates for all levels were calculated according to the number of pixels in the original full size image. This enables us not only to compare the results obtained for a given resolution at different bit rates but also to compare the results related to different spatial resolutions at a given coding budget. Also all the results for SPIHT and HS-SPIHT were obtained by decoding the binary bitstreams. As shown in [6], an improved coding performance (about 0.3–0.6 dB) for SPIHT and consequently for HS-SPIHT can be achieved by further compressing the binary bitstreams with an arithmetic coder. However, it should not be forgotten that the main objective of the HS-SPIHT is to provide a high degree of combined spatial and SNR scalability, therefore there is no intent to focus on arithmetic coding at this stage.

Figures 7, 8 and 9 compare rate-distortion results obtained by HS-SPIHT and SPIHT decoders for the ‘Lena’, ‘Goldhill’ and ‘Barbara’ test images, respectively. Each Figure shows the results for three spatial resolution levels, levels 1 to 3. The results for spatial resolution level 1

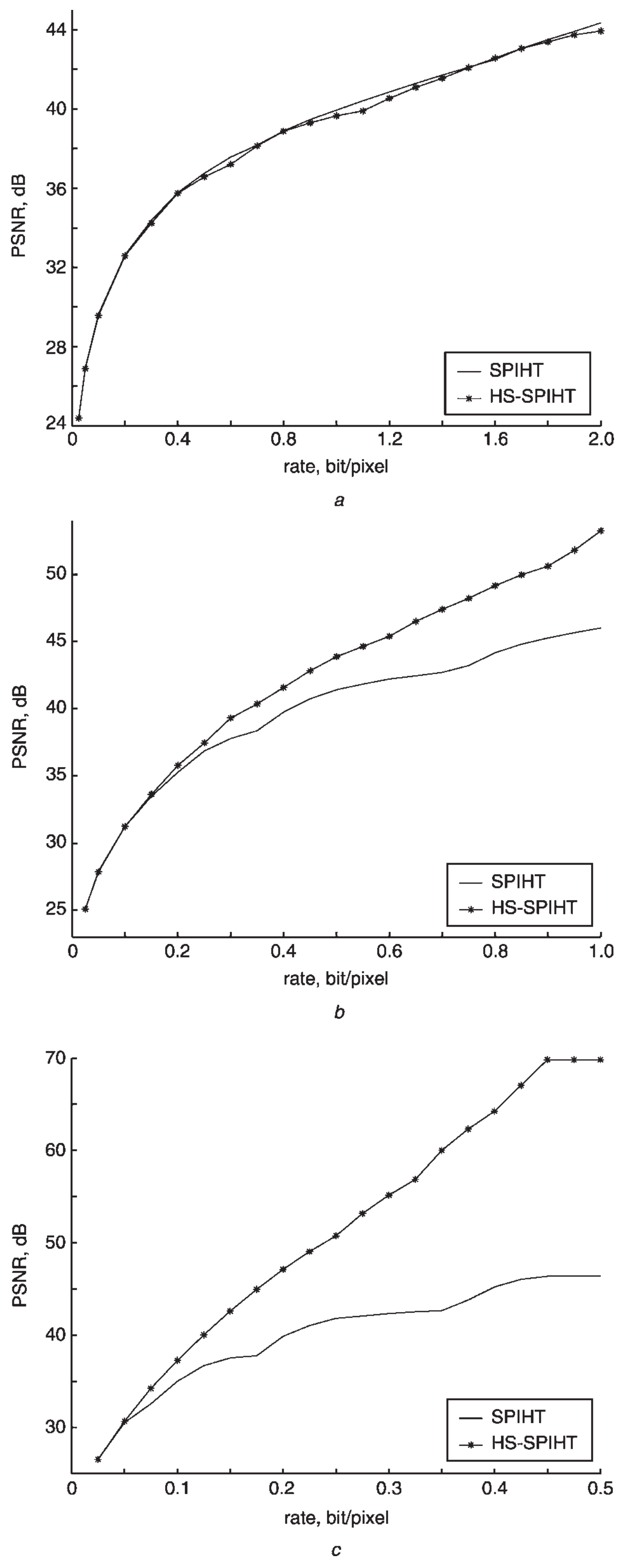


Fig. 7 *Comparison of rate-distortion results obtained by SPIHT and HS-SPIHT decoders for ‘Lena’ test image at different spatial resolution levels*

a Level 1 (original image size 512×152)
b Level 2 (256×256)
c Level 3 (128×128)

(original image) clearly show that the HS-SPIHT algorithm keeps the compression efficiency and the rate embeddedness property of the SPIHT algorithm. The small deviations in compression efficiency between HS-SPIHT and SPIHT at some bit rates in this level are due to their different scanning order of the wavelet coefficients during bitplane coding. For resolution levels 2 and 3, the HS-SPIHT decoder obtained

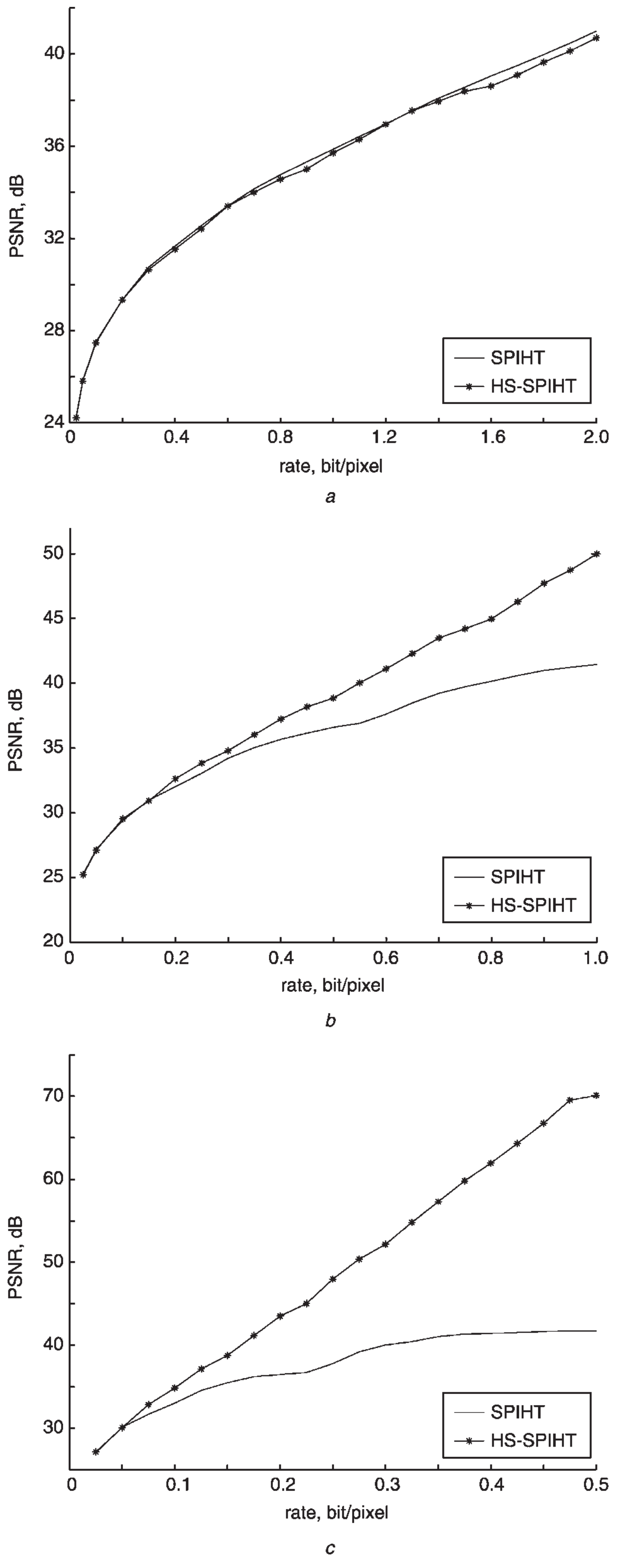


Fig. 8 *Comparison of rate-distortion results obtained by SPIHT and HS-SPIHT decoders for 'Goldhill' test image at different spatial resolution levels*

a Level 1 (original image size 512 × 512)
b Level 2 (256 × 256)
c Level 3 (128 × 128)

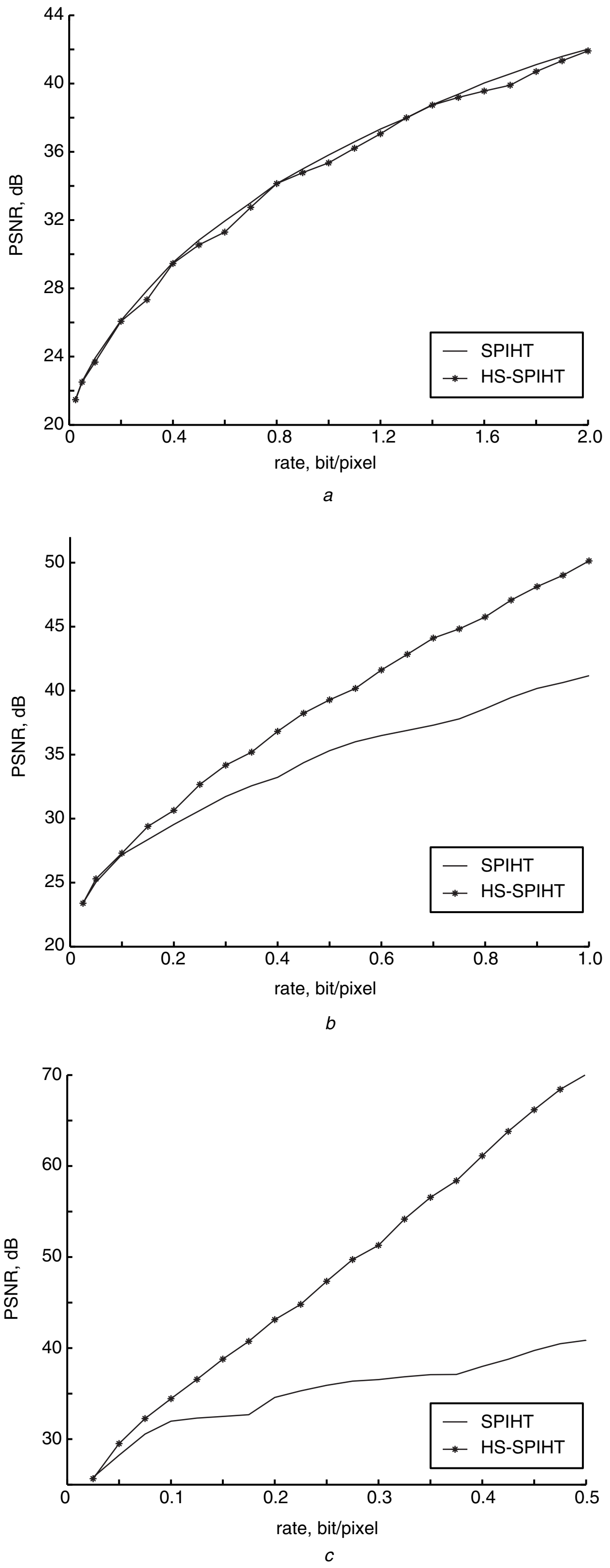


Fig. 9 *Comparison of rate-distortion results obtained by SPIHT and HS-SPIHT decoders for 'Barbara' test image at different spatial resolution levels*

a Level 1 (original image size 512 × 512)
b Level 2 (256 × 256)
c Level 3 (128 × 128)

the proper bitstreams tailored by the parser for each resolution level, while for SPIHT the whole image was first decoded at each bit rate, and then the requested spatial resolutions of the reconstructed and the original images were compared. All bits in the reordered HS-SPIHT bitstream for a particular resolution belong only to that resolution, while in the SPIHT bitstream bits that belong to different resolution levels are interwoven. Therefore, as expected, the performance of HS-SPIHT is much better than SPIHT for resolution levels greater than one. As the resolution level increases, the difference between

Table 1: PSNR results obtained by the HS-SPIHT decoder for reconstructing test images at different spatial resolutions and bit rates

	HS-SPIHT PSNR, dB			
Bit rate, bit/pixel (full)	Resolution: 1/8	Resolution: 1/4	Resolution: 1/2	Resolution: full
Lena				
0.0625	45.29	32.38	28.74	27.60
0.125	68.18	40.03	32.35	30.38
0.25	76.36 (0.143 bit/pixel)	50.75	37.45	33.34
0.5	–	69.70 (0.448 bit/pixel)	43.88	36.57
1	–	–	53.26	39.94
Barbara				
0.0625	42.90	30.85	25.94	22.98
0.125	66.41	36.56	28.42	24.13
0.25	76.48 (0.150 bit/pixel)	47.34	32.67	26.70
0.5	–	70.06 (0.486 bit/pixel)	39.28	30.55
1	–	–	50.13	35.35
Goldhill				
0.0625	44.5	31.48	27.80	26.28
0.125	68.10	37.14	30.82	28.03
0.25	76.41 (0.145 bit/pixel)	47.97	33.85	30.12
0.5	–	70.11 (0.478 bit/pixel)	38.86	32.42
1	–	–	49.98	35.71

HS-SPIHT and SPIHT becomes more and more significant. Table 1 summarises parts of the HS-SPIHT PSNR results given in Figs. 7–9 for different spatial resolutions of the test images. As expected, and the results confirm, for a given bit rate, the PSNR is increased by decreasing the spatial resolution. Having a limitation on bandwidth and consequently bit budget in many image transmission applications, this feature enables the decoder to decode a lower resolution version of the original image at high quality, instead of decoding the high-resolution image at low quality. Moreover HS-SPIHT, in combination with a simple parser, not only provides any desirable resolution and bit rate for the scalable decoder, but also introduces another kind of scalability which we call complexity scalability. To realise complexity scalability, Table 2 compares the times taken by the HS-SPIHT decoder to decode different resolution levels and bit rates of 'Lena'. This Table shows the time ratio based on the time needed to decode the original resolution (level 1) at a bit rate of 1 bit per pixel.

Table 3 provides the number of bytes in the HS-SPIHT bitstream spent coding different resolutions versions of Lena image at bit rates 0.1, 0.5 and 1 respectively.

Table 2: Times spent by HS-SPIHT decoder to decode the 'Lena' (512 × 512) image at different spatial resolutions and bit rates

	Decoding time ratio			
Bit rate, bit/pixel (full)	Resolution: 1/8	Resolution: 1/4	Resolution: 1/2	Resolution: full
0.0625	0.0241	0.0326	0.0456	0.1155
0.125	0.0299	0.0482	0.0663	0.1343
0.25	–	0.0784	0.1098	0.1867
0.5	–	–	0.2710	0.3199
1	–	–	0.9417	1

Values are calculated in ratio to the time spent decoding the full resolution image at 1 bit/pixel

Table 3: Number of bytes spent coding different resolutions of 'Lena' (512 × 512) by scalable encoders at different bit rates

		Spatial resolution					
Encoder	Bit rate, bit/pixel (full)	1/32	1/16	1/8	1/4	1/2	full
HS-SPIHT	0.1	288	827	1679	2615	3211	3276
	0.5	352	1103	3043	7704	13563	16384
	1	384	1234	3605	10160	23345	32768
SPIHT [17]	0.1	2478	2625	2865	3136	3275	3276
	1	22893	23272	24198	26137	29460	32768

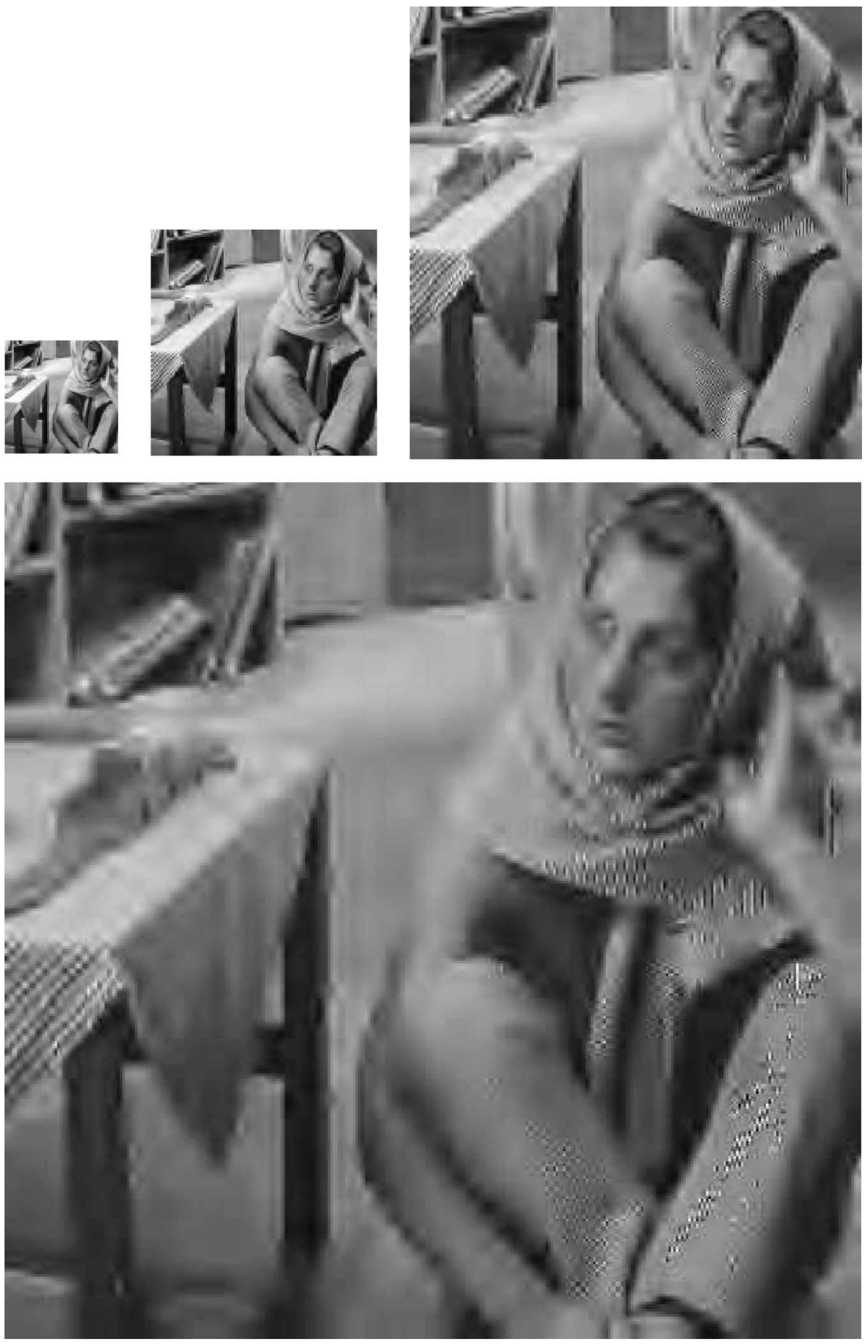


Fig. 10 *Reconstructed spatial resolution levels 1 to 4 of 'Barbara' at 0.0625 bit/pixel by HS-SPIHT decoder*

For comparison, the results for multiresolution SPIHT [17] by Xiong *et al.* are also included. For spatial resolution lower than the original resolution, their codec is obviously inefficient, in comparison with our HS-SPIHT coder. This is because in this multiresolution encoding method, during the low resolution coding, the sorting information associated with (unused) higher resolution levels has not been removed. On the other hand, using the resolution-dependent lists in the HS-SPIHT algorithm enables HS-SPIHT to only encode the necessary information for each resolution level.

Figures 10 and 11 give some visual results for scalable decoding. Four different spatial resolutions (levels 1 to 4) of the 'Barbara' image, reconstructed by the HS-SPIHT decoder at 0.0625 and 0.125 bit/pixel, respectively, are shown in these figures.

6.2 OBHS-SPIHT results

This Section provides some results for object-based coding by the OBHS-SPIHT coding system. The OBHS-SPIHT encoder and decoder algorithms were fully implemented in software. An efficient, non-expansive shape-adaptive DWT (SA-DWT) approach based on the method introduced in [20] was employed and implemented to decompose arbitrarily shaped objects. For the filtering process in the SA-DWT, 9/7-tap filters [19] were employed and symmetric extension at the boundary of the objects was applied. The first frames of two MPEG-4 CIF colour (in YUV format) test sequences, 'Akiyo' and 'Foreman', were used for the test. Only the foreground objects of the test images were considered for coding. The shape/segmentation masks for these test sequences are supplied by MPEG. Four levels of decomposition by the SA-DWT were first applied to the input object, then the OBHS-SPIHT encoder was set to encode the decomposed object with five levels of spatial scalability support.

Tables 4 and 5 compare PSNR results of OBHS-SPIHT and OB-SPIHT obtained for all colour components (i.e. Y, U and V) of the test objects at various spatial resolutions and

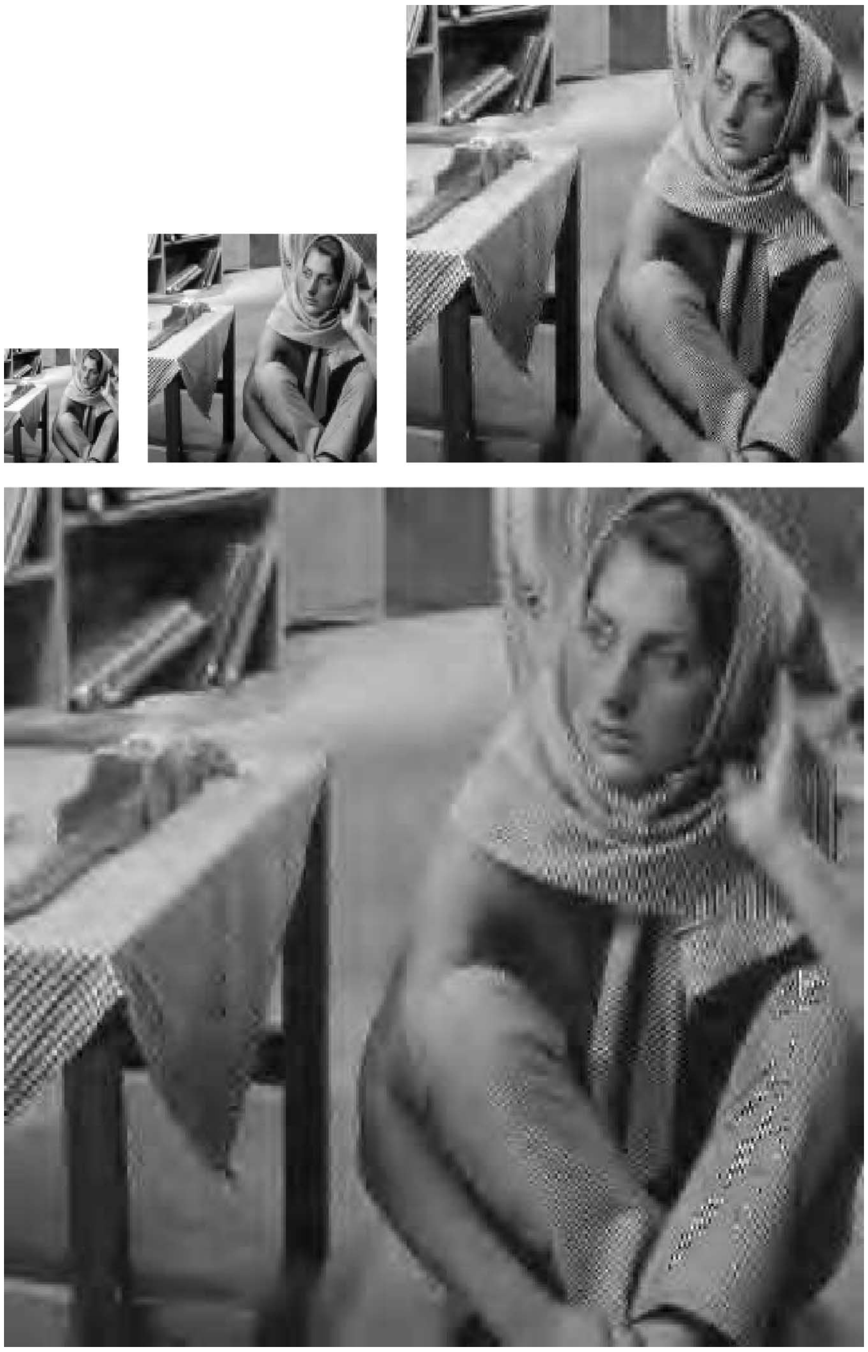


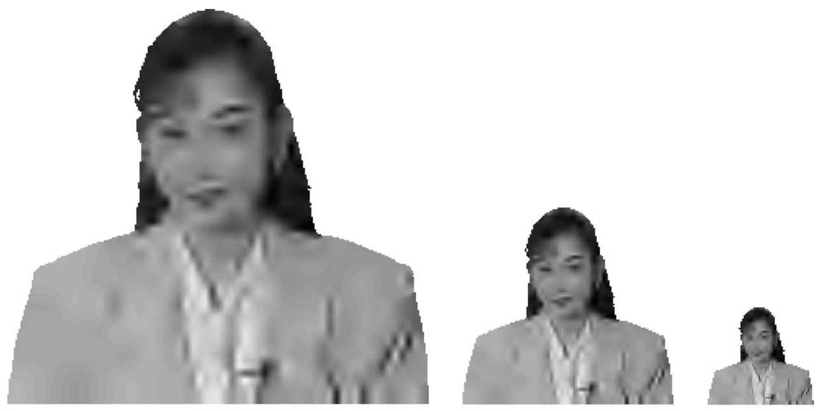
Fig. 11 *Reconstructed spatial resolution levels 1 to 4 of 'Barbara' at 0.125 bit/pixel by HS-SPIHT decoder*

Table 4: PSNR results for foreground object of 'Akiyo' still image (first frame of MPEG-4 CIF 'Akiyo' sequence) at different spatial resolutions and bit rates

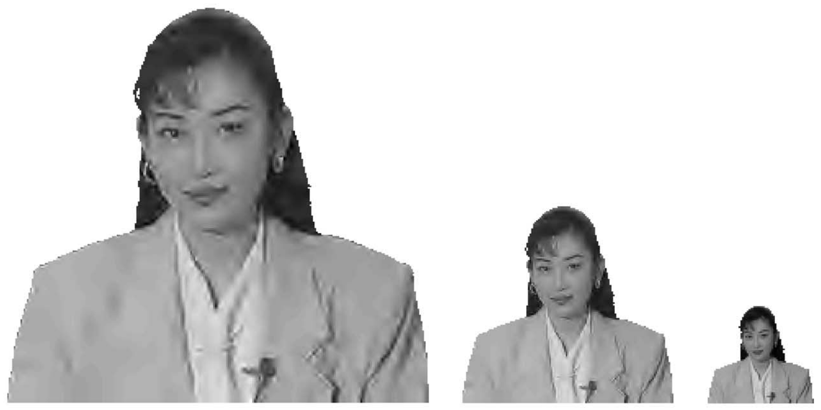
	Rate		OB-SPIHT			OBHS-SPIHT		
Spatial resolution	bit/pixel	bits	Y	U	V	Y	U	V
Full	0.25	9446	27.59	38.26	37.44	27.51	38.26	37.44
	0.50	18892	32.16	40.39	39.54	31.98	40.39	39.48
	1	37783	37.77	42.99	42.88	37.71	43.13	42.94
Half	0.125	4723	24.79	34.57	32.48	24.97	35.40	36.14
	0.25	9446	29.70	39.04	38.25	30.06	39.04	38.25
	0.5	18892	35.41	42.57	42.01	37.08	42.76	42.32
	1	37783	41.42	48.48	48.50	47.11	48.83	48.84
Quarter	0.0625	2361	21.78	20.91	23.81	22.52	30.53	31.31
	0.125	4723	27.83	35.15	32.92	28.13	36.00	37.04
	0.25	9446	32.30	41.11	40.39	38.01	42.40	41.47
	0.5	18892	38.25	48.94	49.10	54.18	53.54	53.87

Table 5: PSNR results for foreground object of 'Foreman' still image (first frame of MPEG-4 CIF 'Foreman' sequence) at different spatial resolutions and bit rates

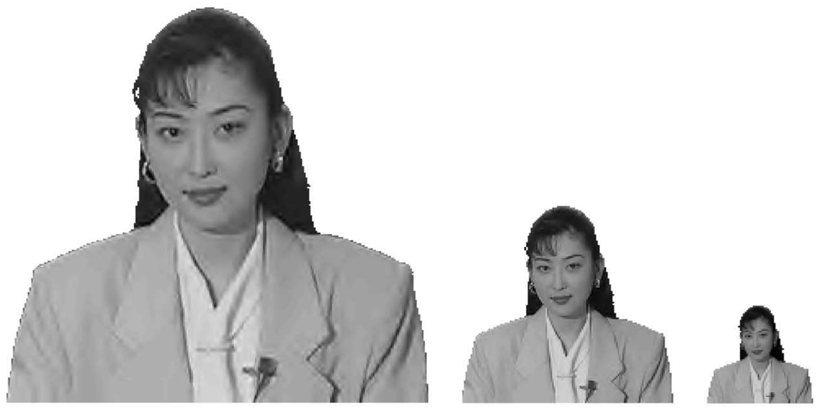
	Rate		OB-SPIHT			OBHS-SPIHT		
Spatial resolution	bit/pixel	bits	Y	U	V	Y	U	V
Full	0.25	8930	32.16	37.98	36.58	32.21	38.46	36.91
	0.50	17860	37.72	41.73	41.12	37.62	41.73	41.09
	1	35720	43.23	45.65	45.75	42.94	45.61	45.70
Half	0.125	4465	27.59	31.70	30.38	27.78	35.35	33.41
	0.25	8930	33.36	39.32	38.05	34.10	39.79	38.52
	0.5	17860	40.01	44.35	43.59	41.57	44.34	43.68
	1	35720	46.94	50.24	50.04	50.90	51.82	52.14
Quarter	0.0625	2232	21.31	20.31	21.17	22.95	23.05	23.12
	0.125	4465	28.92	32.03	30.82	29.34	35.96	34.21
	0.25	8930	36.43	41.79	41.11	39.34	41.90	41.90
	0.5	17860	42.50	51.06	48.69	55.63	55.35	55.49



a



b



c

Fig. 12 *Luminance of decoded 'Akiyo' still object by OBHS-SPIHT at full (left), half (middle) and quarter (right) spatial resolution*
a 0.125 bit/pixel; *b* 0.25 bit/pixel; *c* 0.5 bit/pixel

Table 6: PSNR comparison for foreground object of the first frame of 'Akiyo' CIF sequences at full spatial resolution

Coding algorithm	Rate, bit/pixel	PSNR, dB		
		Y	U	V
OBHS-SPIHT	1	37.71	43.16	42.94
OB-SPIHT	1	37.77	42.99	42.88
OB-SPECK [23]	1	37.55	42.55	42.25
MPEG-4 SA-ZTE [20]	0.9538	38.06	43.43	43.25
SA-DCT [20]	1.0042	37.09	42.14	42.36
Egger's SAWT [26]	1.0065	36.40	42.53	42.40
OWT [27]	0.875	34.13	–	–

bit rates. The OB-SPIHT results refer to our implementation of the original SPIHT algorithm for object-based coding, similar to [21, 22]. Similar to the results provided by the HS-SPIHT for frame-based coding in the previous Section, for spatial resolution level 1, the results clearly show that the OBHS-SPIHT algorithm does not sacrifice the compression efficiency of the OB-SPIHT and for resolution levels 2 and 3, the performance of OBHS-SPIHT is much better than for OB-SPIHT because of its spatial scalability support.

For the Y component, which is the most important component and consumes most of the coding budget, the results show more improvement than for the U and V components. The reason is that the U and V components are more correlated than Y and most of their energy is located in the lowest frequency band in the decomposed image, therefore only a few coefficients are considered at the required resolutions (level 2 and level 3) during the resolution-dependent sorting pass of the OBHS-SPIHT. However, the coding performance for these components (i.e. U and V) is high enough, even for low bit rates for both OB-SPIHT and OBHS-SPIHT. As the resolution level increases, the difference between OBHS-SPIHT and OB-SPIHT results becomes more and more significant. Moreover, for spatial resolutions lower than the full resolution, by increasing the bit rate, OBHS-SPIHT shows more improvement. This is due to the fact that at higher bit rates, where lower bitplanes are also included in the bitplane coding, more coefficients are known to be significant, therefore more information in the OB-SPIHT bitstream can be found which is not related to the required resolution, while the parsed OBHS-SPIHT bitstream includes only the information that belongs to the required resolution. Figure 12 shows some subjective results of multiresolution decoding of the Akiyo object at three bit rates (0.125, 0.25 and 0.5 bit/pixel) obtained by the OBHS-SPIHT decoder.

Table 6 compares the compression efficiency of the OBHS-SPIHT algorithm with some state-of-the-art object-based coding algorithms for coding the foreground of the first frame of the 'Akiyo' CIF sequence. The OB-SPECK coder [23] is an extension of a binary version of SPECK [24]. The SA-ZTE and SA-DCT, respectively based upon SA-DWT and SA-DCT, are implemented in the MPEG-4 verification model reference software [20]. Egger's codec [25] uses a shape-adaptive wavelet transform and employs EZW [2]. The results for OB-SPIHT are obtained from our implementation of an extended version of the original SPIHT coder for object-based processing. Both the OB-SPIHT and OBHS-SPIHT results in this Table are obtained from decoding the binary bitstreams without applying extra arithmetic coding. In addition to providing efficient compression performance, the OBHS-SPIHT fully supports resolution scalability while the other coders (except OB-SPECK) reported in this Table are not resolution scalable.

7 Conclusions

We have proposed a highly scalable SPIHT (HS-SPIHT) algorithm that supports combined spatial and SNR scalability. The flexible, fully scalable binary bitstream of the HS-SPIHT algorithm can easily be parsed to provide rate embedded sub-bitstreams for all lower spatial resolution decoding. The parsing process is done without the need to decode the main bitstream. The HS-SPIHT does not sacrifice the compression efficiency and low complexity of the original SPIHT algorithm when adding the spatial scalability feature. A modification of the proposed algorithm for highly scalable texture coding of arbitrarily shaped visual objects has also been presented. The experimental results for both frame-based and object-based coding show that for the highest resolution level the HS-SPIHT is mostly equivalent to the original SPIHT algorithm while offering the opportunity to decode lower resolution versions with improved quality. The proposed highly scalable object-based image coding algorithm can be utilised in many multimedia applications such as image storage and retrieval systems, progressive web browsing and multimedia information transmission, especially over heterogenous networks.

8 References

1 ISO/IEC, 'ISO/IEC FCD 15444-1: Information technology - JPEG image coding system: core coding system', ISO/IEC JTC1/SC 29/WG 1 N1646, Mar. 2000

2 Shapiro, J.M.: 'Embedded image coding using zerotree of wavelet coefficients', *IEEE Trans. Signal Process.*, 1993, **41**, pp. 3445–3462

3 Zandi, A., Allen, J.D., Schwartz, E.L., and Boliek, M.: 'CREW: compression with reversible embedded wavelet'. Proc. IEEE Data Compression Conf., March 1995, pp. 212–221

4 Chen, Y., and Pearlman, W.A.: 'Three-dimensional subband coding of video using the zero-tree method', *Proc. SPIE–Int. Soc. Opt. Eng.*, 1996, **2727**, pp. 1302–1309

5 Martucci, S.A., and Sodagar, I.: 'Entropy coding of wavelet coefficients for very low bit rate video'. Proc. IEEE Int. Conf. on Image Processing (ICIP'1996), September 1996, vol. 2, pp. 533–536

6 Said, A., and Pearlman, W.A.: 'A new, fast and efficient image codec based on set partitioning in hierarchical trees', *IEEE Trans. Circuits Syst. Video Technol.*, 1996, **6**, pp. 243–250

7 Liang, J.: 'Highly scalable image coding for multimedia applications'. Proc. ACM Multimedia 97, November 1997, pp. 11–19

8 Wang, Q., and Ghanbari, M.: 'Scalable coding of very high resolution video using the virtual zerotree', *IEEE Trans. Circuits Syst. Video Technol.*, 1997, **7**, (5), pp. 719–727

9 Martucci, S.A., Sodagar, I., Chiang, T., and Zhang, Y.-Q.: 'A zerotree wavelet video coder', *IEEE Trans. Circuits Syst. Video Technol.*, 1997, **7**, (1), pp. 109–118

10 Tham, J.Y., Ranganath, S., and Kassim, A.A.: 'Highly scalable wavelet-based video codec for very low bit-rate environment', *IEEE J. Sel. Areas Commun.*, 1998, **16**, (1), pp. 12–27

11 Kim, B.-J., and Pearlman, W.A.: 'An embedded video coder using three-dimensional set partitioning in hierarchical trees (SPIHT)'. Proc. IEEE Data Compression Conf., March 1997, pp. 251–260

12 Karlenkar, J., and Desai, U.B.: 'SPIHT video coder'. Proc. IEEE Region Ten Int. Conf. on Global Connectivity in Energy, Computer, Communication and Control (TENCON'1998), 1998, vol. 1, pp. 45–48

13 Kim, B.-J., Xiong, Z., and Pearlman, W.A.: 'Low bit-rate scalable video coding with 3-D set partitioning in hierarchical trees (3-D SPIHT)', *IEEE Trans. Circuits Syst. Video Technol.*, 2000, **10**, (8), pp. 1374–1387

14 Zho, J., and Lawson, S.: 'Improvements of the SPIHT for image coding by wavelet transform'. Proc. IEE Seminar on Time-scale and Time-Frequency Analysis and Applications (Ref. No. 2000/019), 2000, pp. 24/1–24/5

15 Khan, E., and Ghanbari, M.: 'Very low bit rate video coding using virtual SPIHT', *Electron. Lett.*, 2001, **37**, (1), pp. 40–42

16 Cai, H., and Zeng, B.: 'A new SPIHT algorithm based on variable sorting thresholds'. Proc. IEEE Int. Symp. on Circuits and Systems (ISCAS'2001), May 2001, vol. 5, pp. 231–234

17 Xiong, Z., Kim, B.-J., and Pearlman, W.A.: 'Multiresolutional encoding and decoding in embedded image and video coders'. Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'1998), Seattle, WA, USA, May 1998, vol. 6, pp. 3709–3712

18 Danyali, H., and Mertins, A.: ‘Highly scalable image compression based on SPIHT for network applications’. Proc. IEEE Int. Conf. on Image Processing (ICIP’2002), Rochester, NY, USA, September 2002, vol. 1, pp. 217–220
19 Antonini, M., Barlaud, M., Mathieu, P., and Daubechies, I.: ‘Image coding using wavelet transform’, *IEEE Trans. Image Process.*, 1992, **1**, pp. 205–220
20 Li, S., and Li, W.: ‘Shape-adaptive discrete wavelet transforms for arbitrarily shaped visual object coding’, *IEEE Trans. Circuits Syst. Video Technol.*, 2000, **10**, (5), pp. 725–743
21 Minami, G., Xiong, Z., Wang, A., and Mehrotra, S.: ‘3-D wavelet coding of video with arbitrary regions of support’, *IEEE Trans. Circuits Syst. Video Technol.*, 2001, **11**, (9), pp. 1063–1068
22 Yuan, Y., and Chan, C.W.: ‘Coding of arbitrarily shaped video objects based on SPIHT’, *Electron. Lett.*, 2000, **36**, (13), pp. 1105–1106
23 Lu, Z., and Pearlman, W.A.: ‘Wavelet coding of video objects by object-based SPECK algorithm’. Proc. Picture Coding Symp. (PCS’2001), Seoul, Korea, Apr. 2001, pp. 413–416
24 Islam, A., and Pearlman, W.A.: ‘An embedded and efficient low-complexity hierarchical image coder’, *Proc. SPIE–Int. Soc. Opt. Eng.*, 1999, **3653**, pp. 294–305
25 Egger, O., Ebrahimi, T., and Kunt, M.: ‘Arbitrarily-shaped wavelet packets for zerotree coding’. Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP’1996), May 1996, vol. 4, pp. 2335–2338
26 Egger, O., Fleury, P., and Ebrahimi, T.: ‘Shape adaptive wavelet transform for zerotree coding’. Proc. European Workshop Image Analysis and Coding for TV, HDTV and Multimedia Applications, Rennes, France, February 1996, vol. 1, pp. 201–208
27 Katata, H., Ito, N., Aono, T., and Kusao, H.: ‘Object wavelet transform for coding of arbitrarily shaped image segments’, *IEEE Trans. Circuits Syst. Video Technol.*, 1997, **7**, (1), pp. 234–237