MULTIRESOLUTIONAL ENCODING AND DECODING IN EMBEDDED IMAGE AND VIDEO CODERS

Zixiang Xiong[†], *Beong-Jo Kim*[‡], *and William A. Pearlman*[‡]
[†]Department of Electrical Engineering, University of Hawaii, Honolulu, HI 96822
[‡]Center for Image Processing Research, Rensselaer Polytechnic Institute, Troy, NY 12180

ABSTRACT

We address multiresolutional encoding and decoding within the embedded zerotree wavelet (EZW) framework for both images and video. By varying a resolution parameter, one can obtain decoded images at different resolutions from one single encoded bitstream, which is already rate scalable for EZW coders. Similarly one can decode video sequences at different rates and different spatial and temporal resolutions from one bitstream. Furthermore, a layered bitstream can be generated with multiresolutional encoding, from which the higher resolution layers can be used to increase the spatial/temporal resolution of the images/video obtained from the low resolution layer. In other words, we have achieved full scalability in rate and partial scalability in space and time. This added spatial/temporal scalability is significant for emerging multimedia applications such as fast decoding, image/video database browsing, telemedicine, multipoint video conferencing, and distance learning.

1. INTRODUCTION

With the fast evolution of multimedia systems, image and video compression is becoming the key enabling technology for delivering various image/video services over heterogeneous networks. Shapiro successfully developed a good practical EZW image coder [1], which was later improved by Said and Pearlman in their work of set partitioning in hierarchical trees (SPIHT) [2]. The SPIHT scheme was recently extended from 2-D to 3-D for video coding [3]. Although the coders in [1, 2, 3] offer scalability in rate, it is highly desirable to have temporal and/or spatial scalabilities in these coders for many applications such as video browsing and multicast network distributions.

We address both encoder and decoder spatial/temporal scalabilities within the EZW framework. Our presentation is based on the SPIHT image coder [2] as it is an improved version of Shapiro's EZW coder [1]. Unless otherwise specified, we henceforth assume that we are dealing with images instead of video. Like Shapiro's EZW coder, the SPIHT image coder uses zerotree quantization to efficiently predict the children nodes based on the significance/insignificance of their parent. It refines each wavelet coefficient on a bitmap base by successive set partitioning, and it stops when the size of the encoded bitstream reaches the exact target bitrate. The final encoded bitstream consists of a small header, sorting bits, sign bits, and refinement bits.

Since the SPIHT image coder in [2] is based on the multiresolutional wavelet decomposition, it is relatively easy to add multiresolutional encoding and decoding as functionalities in partial spatial/temporal scalability. We first concentrate on the simpler case of multiresolutional decoding, in which an encoded bitstream is assumed to be available at the decoder, and no modification to the encoder is needed. This approach is quite attractive since we do not need to change the encoder structure, and the decoder will work independently of the encoder. The idea of multiresolutional decoding is very simple: we partition the embedded bitstream into portions according to subbands, and only decode those that contribute to the resolution we want.

We then turn to multiresolutional encoding, where we modify the encoder such that the resulting multiresolutional encoder generates a layered bitstream, from which the higher resolution layers can be used to increase the spatial resolution of the image obtained from the low resolution layer. But modifying an embedded bitstream is not trivial, especially when the bitstream is generated using arithmetic coding (AC) whose efficiency depends on the context of the input source. Rearranging the SPIHT bitstream into layers before AC usually changes the final bitrate. Fortunately different bits in SPIHT are entropy coded using different modes. For example, the sorting bits are coded using several adaptive models in AC, whereas the sign bits and refinement bits are coded without modeling (i.e., by assuming uniform distribution). Thus we can keep the order of the sorting bits and rearrange the sign bits and refinement bits without changing the final bitrate after AC. This makes multiresolutional encoding possible as we can order the original bitstream into layers, with each layer corresponding to a different resolution (or portion). Although the layered bitstream is not fully embedded, the first layer is still rate scalable.

2. MULTIRESOLUTIONAL DECODING

In order to achieve multiresolutional decoding, we have to partition the encoded bitstream into portions according to subbands. This is done by putting flags in the bitstream during the process of decoding, when we scan through the bitstream and mark the portion that corresponds to the spatial locations defined by the decoder's desired resolution. As the encoded bitstream is embedded, this partitioning process can terminate at any point that is specified by the decoding bitrate. Fig. 1 shows such a bitstream partitioning for half resolution decoding. The gray portion of the bitstream contributes to the half resolution image, while the shaded portion corresponds to coefficients in the three highest frequency bands. We only decode the gray portion of the bitstream for the half resolution image. We also scale down the decoded wavelet coefficients by a power of two before applying the inverse wavelet transform. For decoding in lower resolutions, the gray portion of the bitstream in Fig. 1 is further partitioned in a similar manner.

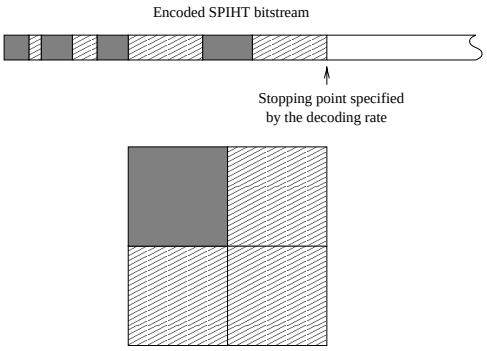

Figure 1: Partitioning of the encoded bitstream into portions according to subbands.

For the video case, we work on the encoded bitstream generated by the 3D SPIHT video coder [3], which is a direct generalization of the image coder [2]. There are two versions of the 3D SPIHT video coder: one with motion compensation (MC), one without. We use the version without MC as it is computationally much faster than the one with MC, and it gives comparable PSNR results to the H.263 video coder [4]. The basic bitstream partitioning method is the same as in the image case, except that we now have two types of flags: one for spatial scalability, another one for temporal scalability. We also scale down the 3D wavelet coefficients properly before applying the inverse wavelet transform.

3. MULTIRESOLUTIONAL ENCODING

The aim of multiresolutional encoding is to generate a layered bitstream. Although information bits (e.g., signs bits and refinement bits) corresponding to different resolutions in the original bitstream are interleaved, the SPIHT algorithm allows us to keep track of the spatial resolutions associated with these bits. Thus we can change the original encoder so that the new encoded bitstream is layered in spatial resolutions. Specifically, multiresolutional encoding amounts to putting in the first (low resolution) layer all the bits needed to decode a low resolution image, in the second (higher resolution) layer those to be added to the first layer for decoding a higher resolution image, and so on. This process is illustrated in Fig. 2 for the two-layer case, where scattered segments of the gray (and shaded) portion in the original bitstream are put together in the first (and second) layer of the new bitstream. A half resolution image can be decoded from the first layer (gray portion) alone, and a full resolution image from both the first and the second layers.

As the layered bitstream is a reordered version of the original embedded SPIHT bitstream, we lose overall scalability in rate after multiresolutional encoding. But the first layer (i.e., the gray layer in Fig. 2) of the layered bitstream is still embedded.

This multiresolutional encoding scheme for images can be readily extended to video in the 3D SPIHT coder. We also note that it is possible to combine multiresolutional encoding and decoding. For example, the first layer (gray portion) of the bitstream generated by the multiresolutional encoder can be used for decoding in lower resolutions.

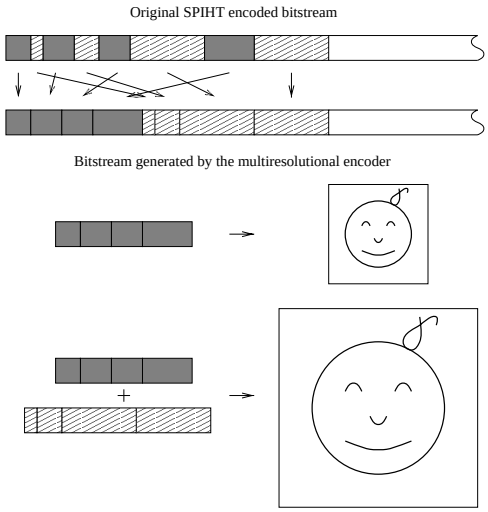

Figure 2: A multiresolutional encoder generates a layered bitstream, from which the higher resolution layers can be use to increase the spatial resolution of the image obtained from the low resolution layer.

4. RESULTS

We have implemented multiresolutional encoding and decoding for both the SPIHT image and video coders. For the image coder, versions with and without AC are both considered. Software for the image case, which is copyrighted, is available via anonymous ftp to `lena.eng.hawaii.edu` with the path `pub/scalability` in `encode_decode.tar.gz`.

4.1. MULTIRESOLUTIONAL DECODING

For results reported in this subsection, we assume a bitstream generated by the original encoder is available at the decoder. Fig. 5 shows the decoded Lena images at different resolutions and different rates from the same encoded bitstream generated by the SPIHT coder with AC. The full resolution 512×512 Lena image is decoded using six levels of inverse wavelet transform. We see that although the decoded full resolution images at relatively high (1 b/p) and very low (0.1 b/p) bitrates look quite different, this difference is much less pronounced between their corresponding half (and quarter) resolution decoded images. What this means is that we can set a very small (0.08 to 0.1 b/p) bitrate for fast decoding and browsing applications, saving decoding time.

Fig. 6 shows the first frames of decoded Foreman sequences at different spatial/temporal resolutions and different bitrates from the same 3D SPIHT encoded video bitstream. The full resolution QCIF Foreman sequence is decoded using three levels of inverse wavelet transform. The half resolution sequences are downsampled in both time and space.

In order to quantitatively assess the time savings in decoding multiresolutional images and video sequences, we give in Tables 1-2 the decoder running time (inverse wavelet transform and I/O included) on a SUN SPARC 20 for the

experiments reported in Figs. 5-6. Decoder running time without AC is also included for the image case. Results in Table 1 indicate that, without AC, half resolution image decoding is 3-4 times faster, quarter resolution image decoding can be 4-12 times faster than full resolution decoding. The time savings is smaller when AC is used, but quarter resolution image decoding can still be 4 times faster than full resolution decoding at 0.1 b/p. From Table 2, we see that, for video sequences, half resolution decoding can be as many as 4-5 times faster than full resolution decoding.

	Without AC		With AC	
	1 b/p	0.1 b/p	1 b/p	0.1 b/p
Full	2.95 sec	2.23 sec	4.90 sec	2.33 sec
Half	1.18 sec	0.58 sec	3.38 sec	0.78 sec
Quarter	0.75 sec	0.18 sec	2.98 sec	0.43 sec

Table 1: Decoding time for the Lena image at different rates and resolutions.

	27.59 kbits/s	20.97 kbits/s
Full (96 frames)	30.37sec	29.04 sec
Half (48 frames)	8.36 sec	6.61 sec

Table 2: Decoding time for the Foreman sequence at different rates and spatial resolutions.

endtable

4.2. MULTIRESOLUTIONAL ENCODING

Fig. 3 (a) shows the two-layer bitstream after multiresolutional encoding for Lena at 1 b/p with AC. Layer boundaries are marked in number of bytes. The decoded half and full resolution images are the same as those in Fig. 5 (a). That is: the first layer will be decoded to the half resolution image, and the first and the second layers together to the full resolution image in Fig. 5 (a), respectively. The difference between decoding from a layered bitstream and decoding from the original bitstream is that in the first case the decoder does not need to search for the whole bitstream for decoding lower resolution images. Fig. 3 (b) shows the two-layer bitstream after multiresolutional encoding for Lena at 0.1 b/p. The decoded half and full resolution images are the same as those in Fig. 5 (b). Table 3 gives the layer boundaries of the six-layer bitstreams for Lena encoded at 1 b/p and 0.1 b/p with AC.

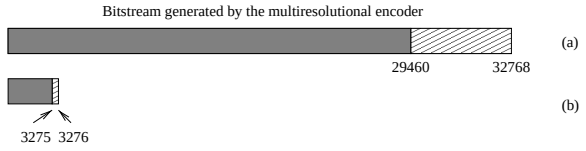

Figure 3: Two-layer bitstreams generated by multiresolutional encoding for Lena (layer boundaries marked in number of bytes). (a) Encoding at 1 b/p. (b) Encoding at 0.1 b/p.

Fig. 4 shows the layered bitstreams after multiresolutional 3D video encoding of the QCIF Foreman sequence. The decoded video sequences at different resolutions are the same as those in Fig. 6. The first layer will be decoded to the half resolution video, the first and the second layers to the full resolution video in Fig. 6, respectively.

Results in Figs. 3-4 indicate that, at relatively low bitrates, the second (or the last) layer of the bitstream is very short because most of the bits are used in coding the low resolution images/video. Thus multiresolutional encoding is more efficient at relatively high bitrate scenarios when a short first layer (or portion) can be used to decode the low resolution image/video. Also, as a large percentage of the bitstream is usually in the first layer, the time savings in low resolution decoding seems to result from the fewer levels of inverse wavelet transforms.

	1/64	1/32	1/16	1/8	1/4	Half	Full
1 b/p	22758	22893	23272	24198	26137	29460	32768
.1 b/p	2411	2478	2625	2865	3136	3275	3276

Table 3: Layer boundaries (in number of bytes) at different resolutions.

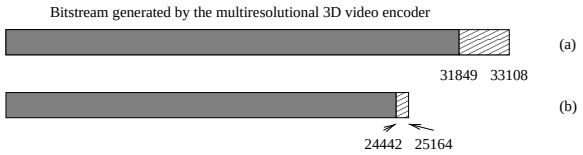

Figure 4: Layered bitstreams generated by multiresolutional 3D video encoding (layer boundaries marked in number of bytes). 96 frames of the QCIF Foreman sequence are encoded. (a) Encoding at a frame rate of 10 frames/sec and a bitrate of 27.59 kbits/sec. (b) Encoding at a frame rate of 10 frames/sec and a bitrate of 20.97 kbits/sec.

5. CONCLUSIONS

In this paper, we presented a simple way of decoding multiresolutional images and video sequences from a single embedded bitstream. An obvious advantage of it is the savings in decoding time. We also addressed multiresolutional encoding which has many applications in network communications. A good example will be scalable multicast video transmission in heterogeneous networks (ISDN, Internet, and Ethernets, etc.). Given the popularity of the SPIHT coder, we believe that our current work is a worthy addition to the original algorithm.

6. REFERENCES

[1] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. on Signal Processing*, vol. 41, 12, pp. 3445-3463, December, 1993.

[2] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. Circuits and Systems for Video Technology*, Vol. 6, 3, pp. 243-250, June, 1996.

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

[4] ITU-T Recommendation H.263, "Video coding for low bitrate communication," Dec. 1995.

(a)

(b)

Figure 5: Multiresolutional decoding in the embedded SPIHT image coder [2] with arithmetic coding. (a) Decoded quarter, half, and full (512×512) resolution Lena images from the same encoded bitstream with the bitrate being set at 1 b/p. The full resolution image has a PSNR of 40.40 dB. (b) Decoded quarter, half, and full (512×512) resolution Lena images from the same encoded bitstream with the bitrate being set at 0.1 b/p. The full resolution image has a PSNR of 30.22 dB.

(a)

(b)

Figure 6: Multiresolutional decoding in the embedded 3D SPIHT video coder [3]. (a) The first frame of the half (both in space and time) and full resolution decoded QCIF (176×144) Foreman sequence. For the full resolution, encoding/decoding frame rate is 10 frames/sec, decoding bitrate is 27.59 kbits/sec. The average PSNR_Y over 96 frames is 26.63 dB, (b) The first frame of the half and full (both in space and time) resolution decoded QCIF Foreman sequence from the same embedded bitstream but with a decoding bitrate of 20.97 kbits/sec. The average PSNR_Y over 96 frames is 25.85 dB.