# Techniques for Speeding up the SPIHT Progressive Image Compressor

V.G. Ruiz, M.F. López, J.J. Fernández, I. García*

Computer Architecture & Electronics Dpt.

University of Almería, 04120 Almería. Spain.

**Abstract**

This work analyses a codec for progressive image transmission which is based on a wavelet transform (U(S+P)T, Unitary Sequential Transform with a spatial predictor) followed by a lossless compressor (SPIHT, Set Partitioning In Hierarchical Trees). As a lossless compressor U(S+P)T+SPIHT performs as good as CALIC or LOCO-I and better than FELICS and the lossless version of JPEG. Additionally, it is able to rebuild high quality images using only a very few amount of the data at the code stream. All these properties makes U(S+P)T+SPIHT very useful for progressive image transmission. In this work, a description of our U(S+P)T+SPIHT version is made and numerical results obtained from extensive performance evaluations are shown. Comparisons of our experimental results to those obtained by the implementation provided by Said and Pearlman show that our solution obtains almost the same compression rates but is twice faster.

**Keywords:** SPIHT, image compression, image transmission, S+P wavelet transform.

## 1 Introduction

There exist many practical applications, such as telemedicine, teleastronomy or database retrieval among others, where collections of images generated in one place must be available

*This work was supported by the Ministry of Education of Spain (CICYT TIC99-0361)

for remote users. In these environments, images are exchanged in a compressed format which must allow remote users to receive the real image or an good approximation of the real image as fast as possible. Progressive image transmission is the technique which allows to obtain a high quality version of the original image from a minimal amount of data and is also able to refine the image at the reception of new data until an exact replica of the original image is obtained at the end of the transmission procedure.

The requirements for a progressive image transmission environment are the following, although they depend on the specific application:

1.- The information visually most relevant must be sent first.

2.- At any time, the size of the reconstructed image at the receiver is the same that the original image. This is important in applications where the distance between several objects must be measured; e.g. in teleastronomy.

3.- In order to minimize the time for transmission, the code-stream must be compressed as much as possible.

4.- For many applications (telemedicine, teleastronomy) at the end of the transmission the reconstructed image must be equal to the original one, so the compressor must be a lossless compressor.

5.- The receiver should be able to refine the reconstructed image when a new bit or group of bits arrives.

With these constraints in mind, a progressive image transmission system consists of: (1) a reversible transform capable of representing the most relevant information with the minimal amount of data or bits, (2) a mechanism for determining the most relevant information and (3) a lossless compressor which permits to reduce the size of the code-stream. In Section 2, the Discrete Wavelet Transform (Unitary Sequential + Prediction Transform, U(S+P)T) as well as the scheme to select the ordering of data for transmission used in our work are described. Sections 3 and 4 are devoted to specify characteristics of our implementation for the Set Partitioning In Hierarchical Trees (SPIHT) coder. Finally, in Section 5 an evaluation of U(S+P)T + SPIHT as lossless compressor and as progressive image transmission mechanism is carried out. It will be shown that our software implementation is twice faster than the version of Said and Pearlman.

## 2 U(S+P)T in progressive transmission

Most of the image compressors are based on a spectral representation of an image because it allows to concentrate most of the energy of the image on a small number of the transform coefficients. In this work a Unitary Discrete Wavelet Transform (Unitary Sequential + Prediction Transform, U(S+P)T) [1, 2] is used as the first processing stage of the compression process.

U(S+P)T is appropriate for progressive transmission because the relevant information can be described with a small amount of data, and moreover, U(S+P)T is $\log_2 N$ faster than other transforms such as DCT [2].

U(S+P)T is a unitary transform based on the Sequential Transform (ST) and a prediction stage. The Sequential Transform is similar to the Haar Transform but it uses integer instead of real arithmetic. Basically, the Sequential Transform (direct and inverse) of a sampled signal $f[x_i]$ $(0 \leq i < 2^n)$ can be defined by the following filters [1]:

$$l[x_i] = \lfloor \tfrac{f[2x_i]+f[2x_i+1]}{2} \rfloor, \quad h[x_i] = f[2x_i] - f[2x_i+1]$$

$$f[2x_i] = l[x_i] + \lfloor \tfrac{h[x_i]+1}{2} \rfloor, \quad f[2x_i+1] = f[2x_i] - h[x_i]$$

The prediction stage of the (S+P) Transform [2] is applied on the high frequency coefficients of the Sequential Transform; i.e. the high frequency coefficients of (S+P) Transform ($h_d[x_i]$) are obtained as:

$$h_d[x_i] = h[x_i] - \lfloor \hat{h}[x_i] + \tfrac{1}{2} \rfloor,$$

$$\hat{h}[x_i] = \tfrac{1}{4}(\Delta l[x_i] - h[x_i+1]) + \tfrac{3}{8}\Delta l[x_i+1],$$

$$\Delta l[x_i] = l[x_i-1] - l[x_i]$$

Both direct and inverse transforms are separable, so for two-dimensional data structures (images) they are obtained by applying the one-dimensional transform to the rows of the image and then to the columns of the transformed data. In Figure 1 the wavelet transform of the image of *lena* is shown.

For a progressive image transmission, when (S+P)T has been computed, it is necessary to determine which data from the transformed image contain the most relevant information

Figure 1: Wavelet decomposition of *lena*.

because this should be the ordering for data transmission. Previous experiments have shown that a magnitude ordering transmission applied to U(S+P)T provides the minimum error for the reconstructed image [3]. This is more evident at the beginning of the transmission, when only a very small percentage of the data has been received. U(S+P)T is obtained by weighting the transform coefficients at every quadrant with the factors specify in Figure 2.

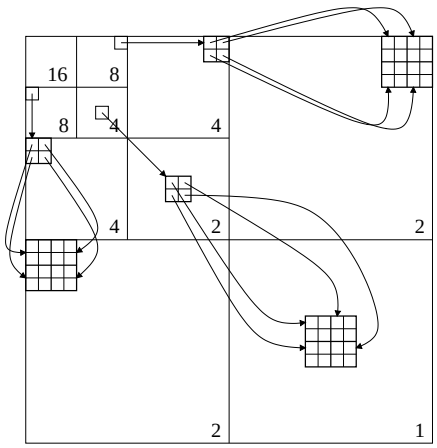


Figure 2: SOTs built by SPIHT and weighting factors of U(S+P)T.

# 3 SPIHT: A Bit-Plane Compressor

Transmitting wavelet coefficients according to a decreasing magnitude ordering strategy is characterized by the fact that the reconstruction MSE decreases quadratically with the

value of the received coefficient [4]. On its hand, the use of a bit-plane ordering transmission strategy has this behavior, in terms of the reconstruction distortion, and moreover presents two additional interesting properties:

1.- Coefficients need not be ordered according to their magnitude before being transmitted, which allows to reduce the time to start the transmission.

2.- The reconstruction error is further minimized since the most significant bits are transmitted first.

The wavelet multiresolution spectrum has the property that there exists a spatial self-similarity relationship among the coefficients at different levels and frequency subbands [2]. This spatial relationship is defined in a hierarchical pyramid constructed with recursive four-subband splitting, as shown in Figure 2: A coefficient $(i, j)$ in the wavelet representation has four direct descendants (offsprings) at the locations $\mathcal{O}(i, j) = \{(2i, 2j), (2i, 2j+1), (2i+1, 2j), (2i+1, 2j+1)\}$, and each of them recursively maintains a spatial similarity to its corresponding four offsprings. That kind of pyramid structure is commonly known as *spatial orientation tree* (SOT).

SPIHT is an efficient compression algorithm that takes advantage of the spatial similarity contained in the wavelet space to efficiently compress the coefficients in a bit-plane ordering. As an example, Figure 1 shows the similarity among subbands within a level or between bands at adjacent levels in the wavelet space. More specifically, if a given coefficient $(i, j)$ is significant at the bit-plane $p$ (i.e., its absolute value is greater than or equal to $2^p$), then some of its descendants $\mathcal{O}(i, j)$ will probably be significant, and viceversa. SPIHT takes into account this fact to optimally find the location of the wavelet coefficient that are significant in every bit-plane.

One of the main features of SPIHT is that the ordering data are not explicitly transmitted. Instead, the encoder (transmitter) and the decoder (receiver) synchronously execute the same search algorithm based on the results of comparisons on the branching points. The encoder transmits the results of each comparison with only one bit whereas the decoder receives them from the code-stream and can then duplicate the path followed by the encoder. The code-bits that are transmitted allow the decoder to find the exact location of the coefficients that become significant in every bit-plane. The SPIHT algorithm generates an efficient

code-stream since the result of every binary comparison is equally likely [4].

# 4 SPIHT Implementation Details

Several new features have been added to our own version of the SPIHT compressor that, in general, makes it even more efficient in terms of computation time compared to the original algorithm from A. Said and W.A. Pearlman [4]. Some of these features are briefly presented in the following.

**Searching for coefficient bits**. The information that the encoder must transmit first is the index of the most significant bit-plane, which may be determined by sweeping across the whole wavelet transform of the image in search of the highest coefficient. On the other hand, the encoder continuously has to search for significant bits of the coefficients, and transmit them.

A new procedure has been devised which does not need to do the initial thorough searching and that, furthermore, is able to speed up the process of determining the significant bits of the coefficients. The procedure is based on a lookup table which, for every node in the SOT of the wavelet transform and for every bit-plane, holds only one bit indicating whether the tree hanging from each of its offsprings is a *zerotree* (i.e., every node in the tree is zero) or not.

**Changing the encoding method**. As described above, the code-bits transmitted by SPIHT allow to locate the significant coefficients in every bit-plane. As SPIHT further proceeds with new planes, the amount of significant bits and the size of the code-stream progressively increase. At the moment in which the code-stream for a given bit-plane generated by SPIHT is longer than the own unencoded bit-plane, SPIHT stops and the remaining bit-planes are not encoded at all. This behavior allows to increase the efficiency, mostly in terms of speed and memory usage, at a not significant payoff in compression rate.

**Coding the refinement bits**. Our implementation of SPIHT does not code those bits that did become significant in previous bit-planes (those bits are commonly known as *refinement*

bits at the present bit-plane). The fact that the spatial correlation among the refinement bits within a bit-plane does not justify that encoding is the reasoning behind that behavior.

**Encoding independently of the image size**. In order to be widely applicable, a progressive transmission algorithm capable of working at top with independency of the image size is highly desirable. The SPIHT algorithm described so far is slowed down for non-square images since several alternatives arise in search of significant bits due to the uneven recursive four-subband splitting of the spatial orientation tree.

Our implementation of SPIHT forces an uniform subband splitting by adding null (zero) coefficients wherever needed. As a consequence, the peculiarities of the algorithm are removed from the most general case, allowing the algorithm to always work at full speed.

# 5 Results and Conclusions

SPIHT has been evaluated in terms of compression rate and speed. Experimental results from the executions of the author's algorithm [5] (SPIHT+Arith) and of our own implementation (SPIHT-F) have been obtained using a set of test images. Both implementations differ in: (1) the code-stream generated by SPIHT+Arith is entropy compressed using Arithmetic Coding [6] and (2) SPIHT-F does not compress the least significant bit-planes, while SPIHT+Arith uses a lossless non-progressive image compressor [2] based on Arithmetic Coding.

The test image corpus is made up of 9 images with different characteristics (see Table 1). *lena, barb, boats* and *goldhill* are 8 bpp standard images. *tvg, B0100, 29jul24, 29jul25* and *29jul26* are 16 bpp astronomical images[1].

The compression rate metric used is the number of bits per pixel computed as:

$$\frac{\text{Compressed Image Size}}{\text{Original Image Size}} \times \text{bpp}$$

Table 2 shows that compression rates for both implementations are quite similar, and even for one of the astronomical images SPIHT-F provides better rate. Table 3 shows

[1]Kindly donated by The German-Spanish Astronomical Center at Calar Alto, which is operated by the Max-Planck-Institut für Astronomie.

Table 1: Image Corpus.

| image | resolution | bpp | entropy |
|---|---|---|---|
| lena | $512 \times 512$ | 8 | 7.45 |
| barb | $512 \times 512$ | 8 | 7.47 |
| boats | $512 \times 512$ | 8 | 7.12 |
| goldhill | $512 \times 512$ | 8 | 7.48 |
| tvg | $574 \times 764$ | 16 | 9.26 |
| B0100 | $1024 \times 1048$ | 16 | 6.38 |
| 29jul0024 | $1024 \times 1148$ | 16 | 7.18 |
| 29jul0025 | $1024 \times 1148$ | 16 | 9.04 |
| 29jul0026 | $1024 \times 1148$ | 16 | 8.89 |

Table 2: Lossless Compression Rates.

| image | SPIHT-F | SPIHT+Arith | $\frac{\text{SPIHT+Arith}}{\text{SPIHT-F}}$ |
|---|---|---|---|
| lena | 4.45 | 4.19 | 0.94 |
| barb | 4.80 | 4.58 | 0.95 |
| boats | 4.58 | 4.34 | 0.95 |
| goldhill | 5.03 | 4.78 | 0.95 |
| tvg | 8.13 | 8.14 | 1.00 |
| B0100 | 4.90 | 4.71 | 0.96 |
| 29jul0024 | 7.08 | 6.71 | 0.95 |
| 29jul0025 | 8.58 | 8.00 | 0.93 |
| 29jul0026 | 8.24 | 7.86 | 0.95 |

Table 3: Lossless Compression/Decompression Times.

| image | SPIHT-F | SPIHT+Arith | speed-up |
|---|---|---|---|
| lena | 0.77/0.72 | 1.29/1.38 | 1.66/1.92 |
| barb | 0.92/0.82 | 1.45/1.54 | 1.58/1.88 |
| boats | 0.91/0.84 | 1.31/1.44 | 1.44/1.71 |
| goldhill | 0.81/0.75 | 1.52/1.62 | 1.88/2.16 |
| tvg | 1.71/1.54 | 4.16/4.75 | 2.43/3.08 |
| B0100 | 3.34/2.87 | 6.17/6.49 | 1.85/2.26 |
| 29jul0024 | 4.59/4.00 | 10.21/11.33 | 2.22/2.83 |
| 29jul0025 | 5.76/5.07 | 11.87/13.29 | 2.06/2.62 |
| 29jul0026 | 5.32/4.70 | 11.70/13.03 | 2.20/2.77 |

compression/decompression times. The last column shows the improvement of SPIHT-F over SPIHT+Arith. For the encoder, SPIHT-F proves to be between 1.4 and 2.4 times faster than SPIHT+Arith, while for the decoder, the speed-up ranks between 1.7 and 3.1. For the 16 bpp images, the gains that have been obtained are, in general, higher than those for 8 bpp images.

Due to the fact that the quality of the reconstructed image depends on the amount of received data, in Figure 3, average values of PSNR (in dB) for the 8 bpp and 16 bpp images have been represented as a function of bpp for both implementations. For 8 bpp images, SPIHT+Arith outperforms SPIHT-F approximately 1 dB, and for 16 bpp images, 4 dB. However, from a visual point of view, these gains are negligible.

In view of these results we can conclude that progressive image transmission by means of SPIHT can be speeded up significantly at not considerable payoff in compression rate, as long as the implementation is provided with the features described in Section 4. The most distinctive feature of our implementation from the original algorithm is that neither Arithmetic Coding nor other encoders are used at all to compress the least significant bit-planes.

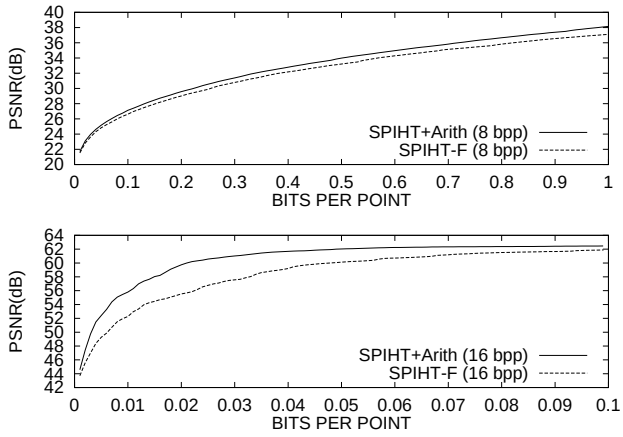


Figure 3: Average values of PSNR obtained from the progressive reconstruction of the test images.

# References

[1] H. Blume and A. Fand, "Medical Imaging III: Image Capture and Display," *Proc. SPIE of Medical Imaging*, vol. 1091, pp. 2–2, 1989.

[2] A. Said and W.A. Pearlman, "An Image Multiresolution Representation for Lossless and Lossy Compression," *IEEE Tran. on Image Proc.*, vol. 5, no. 9, pp. 1303–1310, 1996.

[3] V.G. Ruiz, J.J. Fernández, and I. García, "Image Compression for Progressive Transmission," in *IASTED International Conference on Applied Informatics*, 2001, pp. 519–524.

[4] A. Said and W.A. Pearlman, "A New Fast and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees," *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 6, pp. 243–250, 1996.

[5] A. Said and W.A. Pearlman, "SPIHT Image Compression Programs," http://www.cipr.rpi.edu/research/SPIHT/spiht3.html.

[6] I.H. Witten, R.M. Neal, and J.G. Cleary, "Arithmetic Coding for Data Compression," *Communications of the ACM*, vol. 30, no. 6, pp. 520–540, 1987.