A Method for Improving the Progressive Image Coding Algorithms

Ovidiu COSMA
Department of Mathematics and Computer Science
North University Center
Baia Mare, Roumania
e-mail: ovidiu.cosma@yahoo.com

***Abstract*—This article presents a method for increasing the performance of the progressive coding algorithms for the subbands of images, by representing the coefficients with a code that reduces the truncation error.**

***Keywords*—image compression; subband coding**

I. Introduction

The progressive image coding algorithms have the advantage that they produce an embedded code that has the property that all the representations of the same image at lower bit rates are embedded in the beginning of the bit stream corresponding to the target bit rate. Thus the coding and the decoding processes can be interrupted after producing respectively processing a certain amount of data.

This type of algorithms usually process the transformed image in several passes in which a bit from each of the coefficients that exceed a certain threshold is inserted in the output stream. All the bits processed at the same pass form a bit plane. The best-known algorithms of this kind are the progressive coding variant of JPEG [7], EZW (Embedded image coding using Zerotrees of Wavelet coefficients) [5] and SPIHT (Set Partitioning In Hierarchical Trees) [4]. The last two were developed for coding the subbands of images, computed with the DWT (Discrete Wavelet Transform) [3], [1], and perform the cuantization and codification of the coefficients in an integrated manner.

The SPIHT algorithm [4] codes an image in several passes, each of which having an associated threshold. At each of the passes, the coefficients whose absolute value exceeds the threshold are moved in a List of Significant Coefficients (LSC), the threshold is halved and the next bit from all the coefficients in LSP is inserted in the output stream of the encoder. The process is ended when the bit rate reaches a certain value. For an efficient speculation of the correlations between different subbands, a special data structure called zerotree is used. A zerotree is a quadtree with all the nodes smaller or equal with the root. A whole zerotree that contains insignificant coefficients is represented with a single symbol.

The existing algorithms are focused on the efficient specification of the bit planes. The decoder completes the bits in the planes that it has no information about with zeroes. Thus the coefficients are truncated at the bit that corresponds to the last of the bit planes, and the approximation improves with each new pass.

This article presents a new numeric code that reduces the truncation error of the transform coefficients and implicitly improves the quality of the reconstructed images.

II. The HIA Code

HIA (Half Interval Approximation) is a new numeric code, which allows a more advantageous representation of the coefficients in image subbands. Even if there are certain values for which the arithmetic representation is more accurate, in the majority of the situations, the HIA code reduces the truncation error.

Let's consider a positive value $x \in [p, \quad a \cdot p)$, where $a > 1$ and $p = a^n$. The first non – zero bit in the HIA code of *x*, is placed at position *n*, and indicates that *x* is situated in the above interval. Each of the following bits of the HIA code, halves this interval. The longer the code, the shorter the interval and the value is usually decoded with better precision.

The algorithm presented in Fig.1 can be used for generating the HIA code of the positive value *x*, with the precision determined by ε. The result will be placed in variable *c*.

The decoding algorithm is presented in Fig. 2. The HIA code is taken from variable *c* and the result is placed in $\tilde{x}^{HIA}$ which represents the reconstructed value of *x*.

III. The Truncation Error

Let's consider a value $x \in [p, \quad 2 \cdot p)$, where $p = 2^n$. If a = 2, the HIA code is similar to the binary arithmetic representation. The differences appear only at decoding. The first non – zero bit of *x* is placed at position *n*.

$n = \lfloor log_a x \rfloor$
$l = a^n$, $r = a^{n+1}$, $c = 0$
set the bit at position n of c
while ($r - l > \varepsilon$ and $n >$ the position of the least significant bit of c)
  $n = n - 1$, $m = (l + r)/2$
  if ($x \geq m$) **then**
    $l = m$
    set the bit at position n of c
  else
    $r = m$

Fig. 1. HIA coding algorithm

n = the position of the first non-zero bit of c.
$l = a^n$, $r = a^{n+1}$
while ($n >$ the position of the least significant bit of c)
  $n = n - 1$, $m = (l + r)/2$
  if (the bit at position n of c = 1) then
    $l = m$
  else
    $r = m$
$\widetilde{x}^{HIA} = (l + r)/2$

Fig. 2. HIA decoding algorithm

In the case of the arithmetic representation, the truncation error at bit n, (e^n) is:

$$e^n = x - \widetilde{x}^n = x - p \in [0, \quad p) \tag{1}$$

where $\widetilde{x}^n$ is the value of x, truncated at bit n.

In the case that the HIA code is truncated at the same bit, the decoder will place the reconstructed value at the middle of the interval: $\widetilde{x}^{HIA,n} = 3p/2$. The truncation error $e^{HIA,n}$ is in this case:

$$e^{HIA,n} = x - \widetilde{x}^{HIA,n} = x - 3p/2 \in [-p/2, \quad p/2)$$

$$| e^{HIA,n} | \in [0, \quad p/2] \tag{2}$$

Relations (1) and (2) indicate that the HIA coding halves the upper limit of the truncation error for the coefficients that became significant at the pass corresponding to the threshold p. It is obvious that in the case of truncation at bit n, the arithmetic representation is more accurate for the values in the interval $[p, \quad 5p/4)$, and the HIA coding for the ones in the interval $(5p/4, \quad 2p)$. This aspect is illustrated in Fig.3.

Because the interval of the coefficients for which the HIA coding is advantageous is three times longer, the probability that the coefficients that become significant at the pass corresponding to the threshold p are more precise represented is 3/4 in the case of the HIA code, and 1/4 in the case of the arithmetic representation.

There are a few drawbacks related to the HIA code. In the first place, there are situations in which the precision decreases by adding an extra bit to the code. One of this situations is presented in Fig.4, where $\widetilde{x}^{HIA,n}$ and $\widetilde{x}^{HIA,n-1}$ represent the reconstructed values of x when the HIA code is truncated at the bits n, respectively n-1. The ends of the interval through which $\widetilde{x}^{HIA,n}$ is represented are indicated by l and r.

This unwanted effect appears only if x is close enough to the middle of the interval, more precisely if $| x - m | < (r - l)/8$, where m is the middle of the $[l, \quad r]$ interval. If the coefficients are uniformly distributed, this situation appears in 25% of the cases.

The second drawback refers to the fact that most of the arithmetic values cannot be exactly represented on a finite number of bits. The only exceptions are the values situated at the middle of the intervals $[a^n, \quad a^{n+1})$. If a certain value is exactly represented on n bits, than it will never again be exactly represented on $n + k$ bits, where $k > 0$.

For the field of image compression there is no need to exactly represent the transform coefficients, and the HIA code has the advantage that on average, for the entire set of coefficients that represent a transformed image, a lower truncation error is obtained.

IV. THE EVALUATION OF THE HIA CODE

The application whose block scheme is presented in Fig.5 was built for the evaluation of the HIA code. The first module extracts the luminance component Y from the test images that are represented in the RGB colour space [2]. Then the Discrete Wavelet Transform (DWT) of Y is computed using the Villasenor 18/10 digital filter [9]. The resulting coefficients are coded at different bit rates with the SPIHT algorithm. Constant $a = 2$ was chosen for the evaluation of the HIA code. The remaining blocks perform the inverse transforms, and the comparator module outputs the PSNR [6] computed for the reconstructed luminance component $\widetilde{\mathrm{Y}}$.

PSNR (Peak Signal-to-Noise Ratio) and RMSE (Root Mean Square Error) are two of the most common quality measures used in image compression [6]. These are defined in relations (3) and (4), where N is the total number of pixels, x_i are the pixels in the original image and $\widetilde{x}_i$ are those in the reconstructed one.

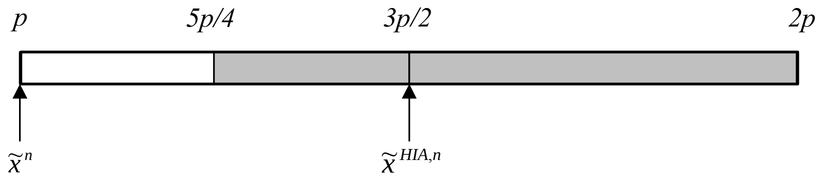

Fig. 3. The interval of values for which the HIA code is advantageous.

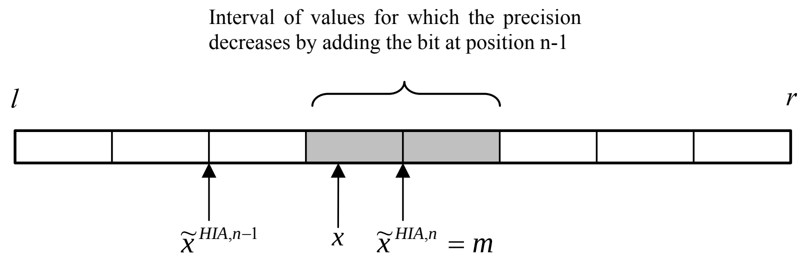

Fig. 4. Values for which the precision will decrease at the next pass.

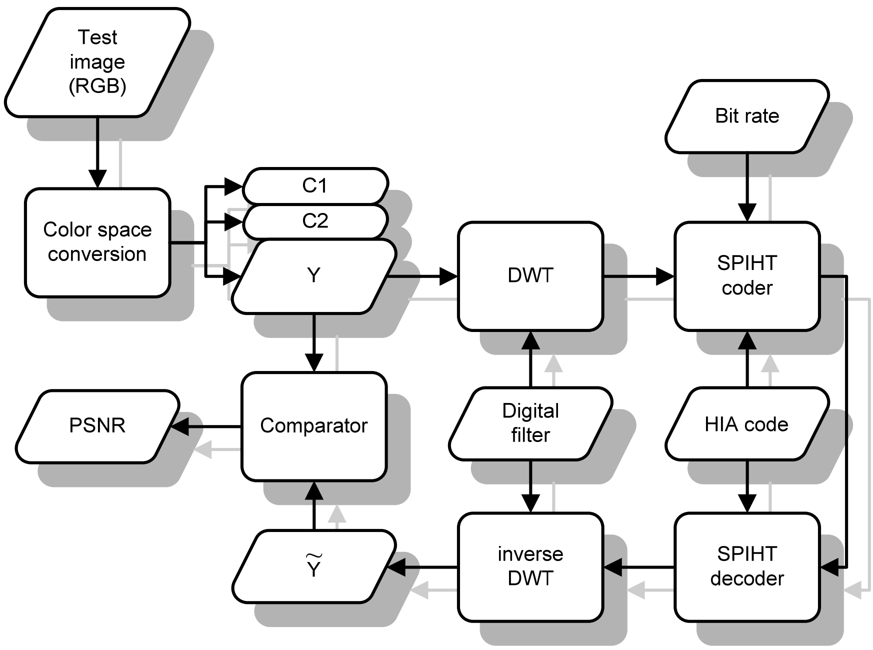

Fig. 5. The performance evaluating application's block scheme.

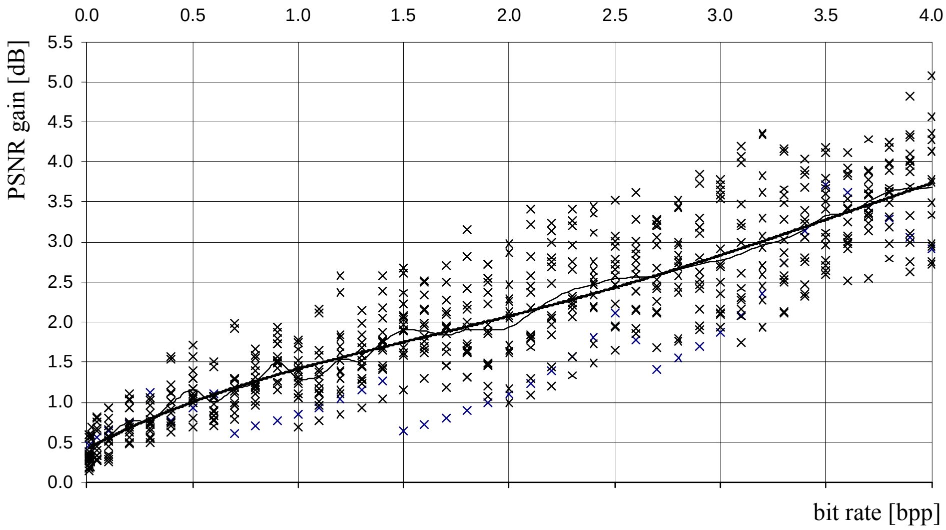

Fig. 6. PSNR gain with the HIA code.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \tilde{x}_i)^2} \tag{3}$$

$$PSNR = 20\log_{10}\frac{\max_i |x_i|}{RMSE} \tag{4}$$

V. Experimental Results

A set of 20 test images with various amounts of details and smooth regions was used in the evaluation process. The results expressed by the increase of the PSNR are presented in Fig. 6. The thin line represents the average computed for the entire set of images, and the thick one represents a polynomial regression curve.

VI. Conclusions

When used in combination with the SPIHT algorithm, the HIA code increases the PSNR computed for the reconstructed images on average with 0.5 – 4.7 dB, depending on the bit rate. The best results were obtained for images reach in details, and at low compression ratios, because in those cases there are more coefficients to benefit from the HIA code.

For obtaining images of similar qualities, the original SPIHT algorithm generates an image representation that is longer on average with 20 – 25% than in the case of using the HIA code. The computing effort required for performing the HIA coding and decoding is negligible. In the particular case that was evaluated, the HIA code is similar to the binary arithmetic representation of the coefficients, and the decoding process requires a few simple operations.

References

[1] O. Cosma, "Wavelet Transform application in image compression", Buletinul Ştiinţific al Universităţii din Baia Mare seria B, 2000

[2] C. Poynton, "A Guided Tour of Color Space", SMPTE Advanced Television and Electronic Imaging Conf. 1995

[3] L. M. Reissell, "Multiresolution and Wavelets", SIGGRAPH'95 Course Notes

[4] A. Said, W. A. Pearlman, "A New Fast and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees", IEEE Transactions on Circuits and Systems for Video Technology, vol.6, 1996

[5] J. M. Shapiro, "Embedded Image Coding Using Zerotrees of Wavelet Coefficients", IEEE Transactions on Signal Processing, vol. 41 no. 12, 12.1993

[6] W. Sweldens, Wavelets, "Signal Compression and Image Processing", SIGGRAPH'95 Course Notes

[7] G. K. Wallace, "The JPEG Still Picture Compression Standard", Communications of the ACM, vol. 34, issue 4, 04.1991

[8] Signal and Image Processing Group, University of Bath, http://dmsun4.bath.ac.uk