Impact of Rate Control Tools on very fast Non-Embedded Wavele Image Encoders[1]

O. López*[a], M. Martínez-Rach[a], J. Oliver[b], M.P. Malumbres[a]
[a]Miguel Hernández University, Avda. Universidad s/n, Elche, Spain 03202;
[b]Technical University of Valencia, Camino de Vera s/n, Valencia, Spain 46022

ABSTRACT

Recently, there has been an increasing interest in the design of very fast wavelet image encoders focused on applicat (interactive real-time image&video applications, GIS systems, etc) and devices (digital cameras, mobile phones, PI etc) where coding delay and/or available computing resources (working memory and power processing) are critical proper operation. Most of these fast wavelet image encoders are non-embedded in order to reduce complexity, so no control tools are available for scalable coding applications. In this work, we analyze the impact of simple rate cor tools for these encoders in order to determine if the inclusion of rate control functionality is worth enough with respec popular embedded encoders like SPIHT and JPEG2000. We perform the study by adding rate control to the r embedded LTW encoder, showing that the increase in complexity still maintains LTW competitive with respect SPI and JPEG2000 in terms of R/D performance, coding delay and memory consumption.

Keywords: Wavelet image coders, fast image coding, rate control, mobile multimedia communications

1. INTRODUCTION

Great efforts have been made to improve coding efficiency of wavelet-based image encoders, achieving in this wa reduction in the bandwidth or amount of memory needed to transmit or store a compressed image. Unfortunately, ma of these coding optimizations involve higher complexity, requiring faster and more expensive processors. For exam the JPEG 2000 [1] standard uses a large number of contexts and an iterative time-consuming optimization algorit (called PCRD) to improve coding efficiency. Other encoders (like the one proposed in [2]) achieve very good cod efficiency with the introduction of high-order context modeling, being the model formation a really slow process. Ev bit-plane coding employed in many encoders (like [3] and [4]) results in a slow coding process since an image is scan several times, focusing on a different bit-plane in each pass, which in addition causes a high cache miss rate.

The above mentioned encoders are designed to obtain the maximum performance in rate-distortion terms and also broader functionality, but unfortunately other design parameters like complexity or memory resources are not consider as critical as the former ones. Recently, several authors have shown an interest on developing very fast and simp wavelet encoders that are able to get reasonable good performance with reduced requirements of computing resourc The objective of these fast and efficient image encoders is mainly targeted to interactive real-time applications runni under resource constrained devices. In that scenario, the data must be encoded as soon as possible to fit the applicati time restrictions using the scarce available resources in the system (memory and processing power).

Basically, these encoders do not present any type of iterative method, and each coefficient is encoded as soon as it visited. This results in the loss of SNR scalability and precise rate control capabilities. They simply apply a consta quantization to all the wavelet coefficients, encoding the image at a constant and uniform quality, as it happened in t former JPEG standard, where only a quality parameter was available (and no rate control was performed).

In [5], a tree-based wavelet encoder (LTW) is presented. The LTW encoding process avoids bit-plane processing a predictive encoding techniques; instead it uses one-pass coefficient coding and a very reduced number of contexts for t final arithmetic coding stage. The LTW encoder requires a very short memory space for the coding process but it is n SNR scalable in a natural fashion (i.e. it is not SNR-embedded).

[1] This work was funded by the Spanish Ministry of Education and Science under grant TIC2003-00339.
* otoniel@umh.es; phone: +34 96 665 8392

Visual Communications and Image Processing 2007, edited by Chang Wen Chen, Dan Schonfeld, Jiebo Luo,
Proc. of SPIE-IS&T Electronic Imaging, SPIE Vol. 6508, 650829, © 2007 SPIE-IS&T · 0277-786X/07/$18

Another very fast non-embedded encoder has been proposed in [6]. This encoder is called PROGRES. It follows the same ideas of [5], avoiding bit-plane coding and using coefficient trees to encode wavelet coefficients in only one-pass. In this encoder, all the coefficients (and not only the zero coefficients) are arranged in trees. The number of bits needed to encode the highest coefficient in each tree is computed, and all the coefficients at the current subband level are binary encoded with that number of bits. Then, the following subband level is encoded (in decreasing order), simply by computing again the number of bits needed to represent each sub-tree at that level and using that number of bits again.

Recently, the BCWT encoder [7] was proposed. It offers high coding speed, low memory usage and good R/D performance. The key of BCWT encoder is its unique one-pass backward coding, which starts from the lowest level sub-bands and travels backwards. MQD map calculation and coefficient encoding are all carefully integrated inside this pass in such a way that there is as little redundancy as possible for computation and memory usage.

Our purpose is to evaluate and analyze the impact on the performance of explicit rate control tools for non-embedded encoders. We will do it not only in terms of R/D performance but also in terms of coding delay and overall memory usage.

Our first rate control proposal extracts some features from the source image and finds correlations with the quantization parameter for a specific target bit-rate based on a standard set of representative images. Afterwards, we define a simplified model of the encoding engine, to determine an initial estimation that will be adjusted by means of curve fitting techniques based on a standard set of images (KODAK set). And finally, to increase the accuracy of the previous method, we propose a lightweight iterative version for bounding the estimation error. In next section, we show a detailed description of the different rate control proposals.

2. FAST RATE CONTROL ALGORITHMS FOR NON-EMBEDDED ENCODERS

In [8], we proposed three lightweight rate control tools for non-embedded encoders. These tools will predict the proper quantization values that lead to a final bit rate close to the target one. In particular, we propose several bit rate prediction methods with increasing complexity and accuracy. We have chosen LTW [5] for evaluation purposes, although other encoders with scalar quantization could also be used.

2.1 First-order entropy based rate control

The first method is based on the extraction of the first-order entropy. The estimation process is based on the correlation between entropy, target bit rate and quantization parameters.

We employ the KODAK image set as a representative set for our purposes, and the LTW encoder with a finer and a coarser quantizer, called Q and *rplanes* respectively. As expected, there is a correlation between the source image entropy and the quantization parameters. Therefore, we can establish a relationship between the quantization parameters and the entropy for a given target bit rate by using curve and surface fitting techniques. Although this estimation method suffers from severe errors when working at low and high entropy values, the computational complexity is very low. First of all, the corresponding algorithm (see Figure 1) will determine an appropriate value for *rplanes* given a target bit rate. Afterwards, the Q value will be adjusted through surface fitting, taking into account again the target bit rate and the corresponding entropy value.

Inputs: Wavelet Coefficients ($C_{i,j}$), Target bitrate (T_{bpp}) , Surface Fitting Equations for each rplanes (Eq_i)
(E1) Determine *rp (rplanes)* using T_{bpp}
(E2) Calculate Coefficients Entropy, *Se* after substracting *rplanes* less significant bits
(E3) Select surface equation Eq_i using *rplanes*
(E4) Determine quantization parameter (Q)
Q = Solve equation Eq_i
(E5) Encode with *rplanes* and Q

Fig. 1. Entropy-based Algorithm (*Algorithm I*)

2.2 Rate control based on a trivial coding model

It is very difficult to estimate with certain degree of accuracy the bit rate by using only the source image entropy. So, another approach will be focused on the design of a simple model that represents the encoding engine. Given a source image and the target bitrate, the proposed model should supply an accurate estimation of both quantization parameters (Q and *rplanes*). Applying this idea to the LTW encoder, the simplified coding model will lead us to an initial and fast estimation of the resulting bit rate for different values of the coarser quantizer *rplanes* (from 2 to 7). In this model, for each specific value of *rplanes*, the probability distribution of significant and insignificant symbols is calculated. This way, an estimation of the bit-rate produced by the arithmetic encoder and the number of bits required to store the sign and significant bits (which are raw encoded) form the bit rate estimation (E_{bpp}). The resulting estimation gives a biased measure of the real bit rate for all operative bit-rate range (from 0.0625 to 1 bpp). We reduce the error with a correction factor calculated from the Kodak image set.

After that, the target bit-rate, T_{bpp}, will establish the proper value of *rplanes* ($E_{bpp}(rplanes) > T_{bpp} > E_{bpp}(rplanes+1)$). In order to determine the proper value of Q, we noticed that the bit rate progression from the current *rplane* to the next one follows a second order polynomial curve with a common minimum. So, with the estimated values ($E_{bpp}(rplanes)$, $E_{bpp}(rplanes+1)$ and the curve minimum (K_{min}), we can build the corresponding expression that will supply the estimated value of Q for a given target bit rate. The whole algorithm is described in Figure 2.

Inputs: Wavelet Coefficients ($C_{i,j}$), Target bitrate (T_{bpp}) , Q Curve minimun (K_{min})

(E1) for each *rp (rplanes)* **in** [2..7]
  for each $C_{i,j}$ coefficient
    $nbits_{i,j} = \lceil \log_2 (|C_{i,j}|) \rceil$
    if $nbits_{i,j} > rp$
      $Symbol_{(nbits_{i,j} - rp)} += 1$
      $Bits_{total}$ += $nbits_{i,j}$-rp
    else
      $Symbol_{non-significant} += 1$
  Calculate the Symbols Entropy, *Se*
  $E_{bpp} = (Bits_{total} / sizeof(image)) + Se$
(E2) for each *rp* **in** [2..7]
    Apply_Correction_Factor;
(E3) Determine *rp*
    $E_{bpp}(rp) > Tbpp > E_{bpp}(rp+1)$
(E4) Determine quantization parameter (Q)
    Obtain A,B,C using $E_{bpp}(rp)$, $E_{bpp}(rp+1)$ and K_{min} for $T_{Bpp} = A.Q^2 + B.Q + C$ and solve equation
(E5) Encode with *rplanes* and Q

Fig. 2. Model-based algorithm (*Algorithm II*).

2.3 Lightweight Iterative Rate Control

With the rate control method described in the previous subsection, we can define an iterative version to reduce the estimation error with a moderate increase of computational complexity. Thus, depending on the application restrictions, we can get the proper trade-off between both factors: complexity and accuracy in the prediction. Now, we can define the maximum estimation error, and the algorithm will perform iterations until this condition is satisfied or a maximum number of iterations is reached.

In the first iteration, the proposed algorithm will estimate the *rplanes* and Q values for the target bit rate by using the algorithm described in the previous subsection. Then the source image will be coded with the quantization parameters found. If the resulting estimation error is lower than the maximum allowed, then the algorithm finishes, else we make a new Q estimation with the same *rplane* value based on the observed error. If we have reached the maximum number of iterations, the algorithm finishes, else the source image is encoded again with the new Q and a new iteration is performed. In addition, after the third iteration, we recomputed the second order polynomial curve using the real values obtained in previous iterations through an iterative approximation method, in order to increase the accuracy of the estimation process. In Figure 3 we can see the whole algorithm.

Inputs: Wavelet Coefficients ($C_{i,j}$), Target bitrate (T_{bpp}) , Q Curve minimum (K_{min}), Maximum Allowed Error (MAE)

(E1) Obtain *rplanes* and Q parameter using *Algorithm I*

(E2) Iterative stage
```
if error > T_err
  New_T_bpp = T_bpp;
  for i=1 to MAXITERATIONS
    if i in [1..3]
      New_T_bpp += error;
      error = Encode_And_Evaluate_Error
      Points[i] = (Q, Real_bpp)
    else
      Q = Newton (Points (Last three) )
      error = Encode_And_Evaluate_Error
      Points[i] = (Q, Real_bpp)
```

Fig. 3. Iterative algorithm (*Algorithm III)*

3. NUMERICAL RESULTS

In order to analyze the impact of rate control proposals in LTW non-embedded encoder, we have performed several experiments comparing the obtained results with the original LTW encoder. In addition to R/D performance we will also employ other performance metrics like coding delay and memory consumption.

All the evaluated encoders have been tested on an Intel PentiumM 1.4 GHz with 1Gbyte RAM Memory. In order to perform a fair evaluation, we have chosen SPIHT (original version), JPEG2000 (Jasper 1.701.0) and LTW version 1.1, since their source code is available for testing. The correspondent binaries were obtained by means of Visual C++ (version 6.0) compiler with the same project options and under the above mentioned machine.

The test images used in the evaluation are: Lena (512x512), Barbara (512x512), GoldHill (512x512), Café (2560x2048) and Woman (2560x2048).

Table 1 shows the coding delay for all encoders under evaluation. In particular, LTW_RC is the corresponding rate control version of LTW based on a simple coding model (described in subsection 2.2). On the other hand, LTW_RCi is the iterative rate control version of LTW (described in subsection 2.3) with relative (%value) and absolute (ABS suffix) rate control maximum allowed error. We discard the first rate control method (entropy-based) due to its lower accuracy with respect the model-based one.

As expected, JPEG2000 is the slowest encoder and the original LTW is one of the fastest encoders. As shown in Table 1, the LTW_RC version does not introduce a great overhead and it has an acceptable accuracy [8]. If this rate control precision is not enough for the application, LTW_RCi is the candidate at the expense of an increasing complexity. In general, all the rate control versions of LTW are faster than SPIHT, specially the non-iterative version that performs the encoding process two times faster than SPIHT.

In the iterative rate control versions, we have found two ways of defining the maximum allowed error: A relative or an absolute bound. The relative maximum error shows a non lineal behavior, since rate control precision of 1% is not the same at 2 bpp than 0.125 bpp. For very low bit rates, achieving an accuracy of 1% has no effects to R/D performance. The maximum absolute error is fixed independently of the target bit rate, so it produces different relative errors at different bit-rates. Empirically we have chosen an absolute maximum error of 0.04 bpp. It is important to take into account that the intrinsic accuracy of proposed rate control methods is around 5% error at 1 bpp and 9% at 0.125 bpp, as shown in [8].

Table 1. Execution time comparison of the coding process excluding DWT (time in million of CPU cycles)

Codec/ Bitrate	SPIHT	JPEG 2000	LTWOrig	LTW_RC	LTW_ RCi_2%	LTW_ RCi_ABS
CODING Lena (512x512)						
0.125	20.82	158.79	9.75	8.316	20.25	10.03
0.25	29.12	161.92	14.09	11.726	27.08	13.42
0.5	45.82	167.14	22.46	18.356	62.07	40.74
1	79.56	175.64	41.46	36.656	75.75	38.61
DECODING Lena (512x512)						
0.125	11.3	11.46	8.28	6.77	7.2	6.71
0.25	19.38	18.11	13.39	10.8	11.49	10.79
0.5	34.9	30.78	23.48	18.84	20.61	20.12
1	66.8	50.99	46.52	41.11	39.64	40.73

Table 2 shows the R/D evaluation of proposed encoders. The original LTW obtains the bests results (0.2 db approx. for Lena). The rate control versions of LTW have slightly lower PSNR results than SPIHT and JPEG2000, being the LTW_RCi at 2% the one that better R/D behavior shows. The lower performance of rate control versions are mainly due to the bit rate estimation that use to be under the target one (as more accuracy better R/D performance). The maximum relative error for Cafe is 10.8% at 0.25 bpp and the minimum error is 0.32% at 0.125 bpp.

Table 2. PSNR (dB) with different bit-rate and coders

Codec/ Bitrate	SPIHT	JPEG2000	LTWOrig.	LTW_RC	LTW_ RC i_2%	LTW_ RC i_ABS
Lena (512 x 512)						
0.125	31.1	30.81	31.28	30.59	31.06	30.59
0.25	34.15	34.05	34.33	33.65	34.05	33.65
0.5	37.25	37.26	37.39	36.76	37.15	37.08
1	40.46	40.38	40.55	40.34	40.34	40.34
Café (2560 x 2048)						
0.125	20.67	20.74	20.76	20.63	20.63	20.63
0.25	23.03	23.12	23.24	22.6	23.08	22.6
0.5	26.49	26.79	26.85	26.04	26.53	26.53
1	31.74	32.03	32.02	30.89	31.64	31.64

In Table 3 the memory requirements of different encoders under test are shown. The original LTW needs only the amount of memory to store the source image (in-line processing). LTW_RC requires also an extra of 1.2 KB basically used to store the histogram of significant symbols needed to accomplish the rate control algorithm. On the other hand, LTW_RCi version requires twice memory space than LTW and LTW_RC, since at each iteration the original wavelet coefficients must be restored. SPIHT requires near the same memory than LTW_RCi, and JPEG2000 needs three times the memory of LTW.

Table 3. Memory Requirements for evaluated encoders[2] (Kb)

Codec /Image	SPIHT	JPEG2000	LTWOrig.	LTW_RC	LTW_RCi 2%
Lena	3228	4148	2048	2092	3140
Café	46776	65832	21576	21632	42188

Table 4 shows the encoding and decoding delays for Café test image, now including the DWT transform. As it can be shown, the LTW is the fastest one, being the LTW_RC just behind it. However, iterative versions, LTW_RCi, obtain a similar encoding delay than SPIHT, showing a better behavior the maximum absolute error version (LWT_RC_Abs). Also, the decoding delay of LTW and its rate control versions is, in general, shorter than SPIHT and JPEG2000.

It is important to note that in LTW rate control versions, the DWT is performed with a lifting scheme while in SPIHT is carried out as a convolution. Our lifting scheme implementation is faster than SPIHT convolution for small size images; however for large size images like café, SPIHT convolution is faster. Notice that in rate control versions of LTW, an extra time is needed to compute the histogram of significant symbols for *rplanes*=2.

Table 4. Execution time comparison of the coding process including DWT (time in seconds).

Codec/ Bitrate	SPIHT	JPEG2000	LTW Orig.	LTW_RC	LTW_ RCi 2%	LTW_ RCi_Abs
CODING Cafe (2560x2048)						
0.125	0.670	3.406	0.540	0.714	0.764	0.764
0.25	0.816	3.393	0.600	0.757	1.035	0.805
0.5	1.112	3.436	0.707	0.849	1.222	1.223
1	1.709	3.534	0.891	1.009	1.555	1.554
DECODING cafe (2560x2048)						
0.125	0.516	0.517	0.529	0.519	0.514	0.514
0.25	0.627	0.604	0.604	0.575	0.583	0.568
0.5	0.861	0.759	0.736	0.696	0.703	0.703
1	1.336	1.058	0.964	0.902	0.919	0.919

[2] Results obtained with the Windows XP task manager, peak memory usage column.

(a) (b) (c)

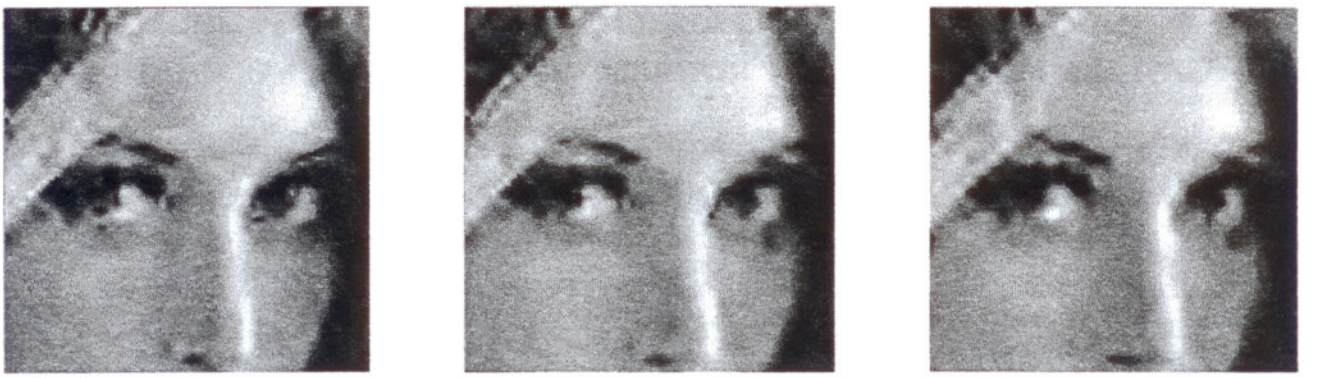
Fig. 4. Lena compressed at 0.125 bpp (a) LTW_RCi_2%, (b) SPIHT, (c) JPEG2000

Fig. 5. Original Lena test image (512x512)

(a) (b) (c)

Fig. 6. Lena compressed at 0.0625 bpp (a) LTW_RCi_2%, (b) SPIHT, (c) JPEG2000

Both Figures 4 and 6 show Lena test image (512x512) compressed at 0.125 bpp and 0.0625 bpp with (a) LTW_RCi, (b) SPIHT and (c) JPEG2000. Although SPIHT encoder is in terms of PSNR slightly better than LTW_RCi and JPEG2000, subjective test does not show perceptible differences between reconstructed versions of Lena image. At 0.0625 bpp the difference in PSNR between LTW_RCi or SPIHT and JASPER is near 0.5 dB. This difference is only visible if we carry out a zoom over the eyes zone as it can be seeing in Figure 7. Both SPIHT and LTW_RCi have similar behavior.



(a) (b) (c)

Fig. 7. Zoom over eyes zone in reconstructed Lena at 0.0625 bpp – (a) LTW_RCi_2%, (b) SPIHT, (c) JPEG2000

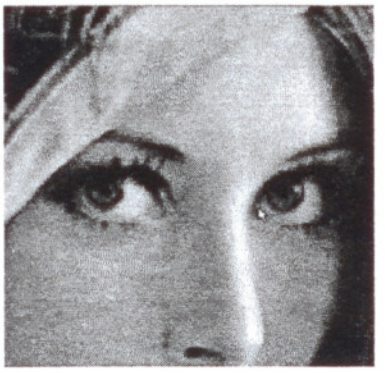


Fig. 8. Zoom over eyes zone in original Lena

4. CONCLUSIONS

In this paper we present three different implementations of rate control methods over the LTW encoder [5], and we compare their performance with SPIHT and JPEG2000 encoders in R/D, execution time and memory consumption. Here we present the first preliminary results that show how we can add rate control functionality to non-embedded wavelet encoders without a significant increase of complexity and little performance looses. We offer several simple rate control tools, being the LTW_RC proposal the one that exhibits the best trade-off between R/D performance, coding delay (2 times faster than SPIHT and 8.8 times faster than JPEG2000) and overall memory usage (similar than original LTW).

We are developing a very fast LTW version (including rate control tool) that will be able to encode in real time an SDTV video signal (INTRA coding only) with very low memory demands and good R/D performance. For that purpose, we will use Huffman entropy coding instead arithmetic coding (in [9] we show that we can reduce 4 times on average the LTW coding delay), a integer-to-integer DWT with lifting, and finally the line-based coding approach of LTW (the last two improvements will largely reduce the required memory).

REFERENCES

[1] ISO/IEC 15444-1, JPEG2000 image coding system, 2000.

[2] X. Wu, "Compression of Wavelet Transform Coefficients," The Transform and Data Compression Handbook, pp. 347-378, CRC Press, 2001.

[3] A. Said, 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, nº 3, 1996.

[4] C. Chrysafis, A. Said, A. Drukarev, A. Islam, W. A. Pearlman, "SBHP- A low complexity wavelet coder," in Proc. IEEE Int. Conference on Acoustics, Speech, and Signal Processing, 2000.

[5] J. Oliver, M. P. Malumbres, "Fast and efficient spatial scalable image compression using wavelet lower trees," in Proc. IEEE Data Compression Conference, Snowbird, UT, March 2003.

[6] Yushin Cho, W. A. Pearlman, A. Said, "Low complexity resolution progressive image coding algorithm: PROGRES (Progressive Resolution Decompression)", in Proc. IEEE International Conference on Image Processing, September 2005.

[7] Jiangling Guo, Sunanda Mitra, Brian Nutter, Tanja Karp, " A Fast and Low Complexity Image Codec based on Backward Coding of Wavelet Trees", Proc. Of the Data Compression Conference, 2006.

[8] O. López, M. Martinez-Rach, J. Oliver and M.P. Malumbres and "A heuristic bit rate control for non-embedded wavelet image encoders", 48th International Symposium ELMAR 2006, Jun 2006.

[9] J. Oliver and M.P. Malumbres, "Huffman Coding Of Wavelet Lower Trees For Very Fast Image Compression", in proc. Of International Conference on Acoustics, Speech, and Signal (ICASSP 2006) (ISBN: 1-4244-0469-X), 2006.