Noname manuscript No.
(will be inserted by the editor)

Enhancing LTW image encoder with perceptual coding and GPU-optimized 2D-DWT transform

Miguel O. Martínez-Rach · Otoniel López-Granado · Vicente Galiano · Hector Migallón · Jesús Llor · Manuel P. Malumbres

Received: date / Accepted: date

Abstract When optimizing a wavelet image coder two main targets are 1) improving its Rate Distortion (R/D) performance and 2) reduce the coding times. In general the encoding engine is the main responsible of achieving R/D performance and also use to be more complex than the decoding part. A large number of works about R/D or complexity optimizations can be found, but only a few tackle the problem of increasing R/D performance while reducing the computational cost at the same time, like Kakadu, an optimized version of JPEG2000. In this work we propose an optimization of the E_LTW encoder with the aim to increase its R/D performance through perceptual encoding techniques and reduce the encoding time by means of a GPU-optimized version of the 2D-DWT transform. The results show that in both performance dimensions our enhanced encoder achieves good results compared with Kakadu and SPIHT encoders and achieving speed-ups of 6x with respect to the original E_LTW encoder.

Keywords Wavelet Image Coding · Perceptual Coding · Contrast Sensitivity Function · GPU-optimization

1 Introduction

Wavelet transforms have reported good performance for image compression, therefore many state-of-the-art image codecs, including the JPEG2000 image coding standard, use the Discrete Wavelet Transform (DWT) [9,12].The use of

This research was supported by the Spanish Ministry of Education and Science under grant TIN2011-27543-C03-03

Miguel O. Martínez-Rach, López-Granado, V.Galiano, H.Migallón, J.Llor, P. Malumbres
Physics and Computer Architecture Department
Miguel Hernández University. Elche, Spain 03202
Tel.: +34-966658364
E-mail: {**mmrach**,otoniel,vgaliano,hmigallon,jllor,mels}@umh.es

wavelet coefficient-trees and successive approximations was introduced by the EZW [13] with a bit-plane coding approximation. SPIHT [12], an advanced version of EZW, process the wavelet coefficient trees in a more efficient way by partitioning the coefficients depending on their significance. Both EZW and SPIHT need the coefficient-tree construction to search for significant coefficients through a multiple iterative process at each bit-plane, which involves high computational complexity.

Bit-plane coding is implemented by the JPEG2000 encoding codeblocks with three passes per plane, so the most important information, from a R/D point of view, is first encoded. It also uses an optional and low complexity post compression optimization algorithm, based on the Lagrange multiplier method. Besides, it uses a large number of contexts for the arithmetic encoder. This post-compression rate-distortion optimization algorithm selects the most important coefficients by weighting them, based on the MSE distortion measurement.

Wavelet-based image processing systems are typically implemented with memory intensive algorithms and with higher execution time than other encoders based on other transforms. In usual 2D-DWT implementations [6], the image decomposition is computed by means of a convolution filtering process and so, its complexity rises as the filter length increases. The image is transformed at every decomposition level, first column by column and then row by row.

In [4], authors propose the E_LTW codec with sign coding, precise rate-control and some optimizations avoids bit-plane processing, at the cost of not being embedded, but with very low memory requirements and similar R/D performance than the one obtained by embedded encoders like JPEG2000 and SPIHT.

Part II of the JPEG2000 standard includes visual progressive weighting [17] and visual masking by setting the weights based on the Human Visual System (HVS) Contrast Sensitivity Function (CSF). Many other image encoders have included much of the knowledge of our human visual system in order to obtain a better perceptual quality of the compressed images. The most widely used characteristic is the contrast adaptability of the HVS, because HVS is more sensitive to contrast than to absolute luminance [15]. The CSF relates the spatial frequency with the contrast sensitivity.

This perceptual coding will improve the perceptual quality of the reconstructed images, so that for a desired rate range, a better perceptual R/D behavior is achieved. Although most studies employ PSNR metric to measure image quality performance, it is well known that this metric not allways capture the distortion perceived by the human being. Therefore, we decided to use objective quality assessment metrics which desing is inspired in the HVS, since our proposal includes perceptual-based encoding techniques that may not be properly evaluated by the PSNR metric.

In this work, we propose the PE_LTW (Perceptually Enhanced LTW) as an enhanced version of the E_LTW encoder by including perceptual coding based in the CSF and the use of GPU-optimized 2D-DWT algorithms based on the methods described in [6,16].

After improving the perceptual R/D behavior of our proposal, we proceed to optimize the 2D-DWT transform module by using GPU processing to reduce the overall encoding time. From previous work, we have defined a CUDA implementation of 2D-DWT transform that is able to considerably reduce the 2D-DWT computation time.

To test the behavior of our proposal we have compared the performance of our PE_LTW encoder in terms of perceptual quality and encoding delays with the Kakadu implementation of the JPEG2000 standard, with and withoug enabling its perceptual weighting mode, and with the SPIHT image encoder.

2 Encoding System

2.1 Encoder

The basic idea of this encoder is very simple: after computing the 2D-DWT transform of an image, the perceptually weighted wavelet coefficients are uniformly quantized and then encoded with arithmetic coding.

As mentioned, the 2D-DWT computation stage runs on a GPU and includes the perceptual weighting based on the CSF and implemented as an Invariant Scaling Factor Weighting (ISFW) [10] that weights the obtained coefficient depending on the importance that the frequency subband has for the HVS contrast sensitivity. We detail the CSF and the ISFW later in next sections.

The uniform quantization of the perceptually weighted coefficients is performed by means of two strategies: one coarser and another finer. The finer one consists in applying a scalar uniform quantization (Q) to the coefficients. The coarser one is based on removing the least significant bit planes (rplanes) from coefficients.

For the coding stage, if the absolute value of a coefficient and all its descendants (considering the classic quad-tree structure from [12]) is lower than a threshold value ($2^{rplanes}$),the entire tree is encoded with a single symbol, which we call LOWER symbol (indicating that all the coefficients in the tree are lower than $2^{rplanes}$ and so they form a lower-tree). But if a coefficient is lower than the threshold and not all its descendants are lower than it, that coefficient is encoded with an ISOLATED LOWER symbol. On the other hand, for each wavelet coefficient higher than $2^{rplanes}$, we encode a symbol indicating the number of bits needed to represent that coefficient, along with a binary coded representation of its bits and sign (note that the *rplanes* less significant bits are not encoded).

The encoder exploits the sign neighborhood correlation of wavelet subband type (HL,LH,HH) as Deever assesses in [2] by encoding the prediction of the sign (success of failure).

The proposed encoder also includes the rate control algorithm presented in [5] but taking into account the sign coding and the intrinsic error model of the rate control. As the rate control underestimates the target rate, the required bits to match the target bit-rate are added to the bitstream. The selected bits correspond to the bit-planes (lower or equal to the *rplanes* quantization parameter) of significant coefficients added to the output bitstream following a particular order, from low frequency subbands to the highest one.

More details about the coding and decoding algorithms, along with a formal description and an example of use can be found in [4,11].

2.2 The Contrast Sensitivity Function

In [10] authors explain how the sensitivity to contrast of the HVS can be exploited by means of the CSF curve to enhance the perceptual or subjective quality of the DWT encoded images. A comprehensive review of HVS-models for quality assessment/image compression is found in [15]. Most of these models take into account the varying sensitivity over spatial frequency, color, and the inhibiting effects of strong local contrasts or activity, called masking.

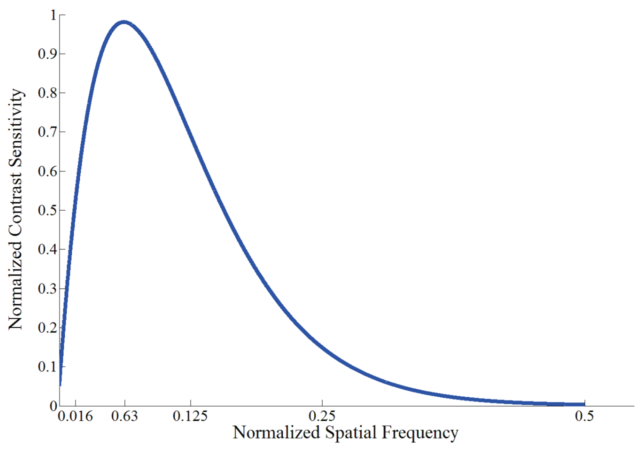


Fig. 1: Contrast Sensitivity Funcion

Complex HVS-models implement each of these low level visual effects as a separate stage. Then the overall model consists of the successive processing of each stage. One of the initial HVS stages is the visual sensitivity as a function of spatial frequency that is described by the CSF. A closed form model of the CSF for luminance images [7] is given by:

$$H(f) = 2.6(0.0192 + 0.114f)e^{-(0.114f)^{1.1}} \quad (1)$$

where spatial frequency is $f = (f_x^2 + f_y^2)^{1/2}$ with units of cycles/degree (f_x and f_y, are the horizontal and vertical spatial frequencies). The frequency is usually measured in cycles per optical degree (cpd), which makes the CSF independent of the viewing distance.

Figure 1 depicts the CSF curve obtained with Equation 1, it characterizes luminance sensitivity as a function of normalized spatial frequency (CSF=1/Contrast threshold). As shown, CSF is a bandpass filter, which is most sensitive to normalized spatial frequencies between 0.025 and 0.125 and less sensitive to very low and very high frequencies. The reason why we can not distinguish patterns with high frequencies is the limited number of photoreceptors in our eye. CSF curves exist for chrominance as well. However, unlike luminance stimuli, human sensitivity to chrominance stimuli is relatively uniform across spatial frequency.

One of the first works that demostrate that the MSE cannot reliably predict the difference of the perceived quality of two images can be found in [7]. They propose, by the way of psychovisual experiments, the aforementioned model of the CSF, that is well suited and widely used ([17][18][19][20]) for wavelet based codecs, therefore we adopt this model.

2.3 Using the CSF

In [10] authors explain how the CSF can be implemented in wavelet based codecs. Some codecs, like the JPEG2000 standard Part II, introduce the CSF as a Visual Progressive Single Factor Weighting, by replacing the MSE by the CSF-Weighted MSE (WMSE) and optimizing system parameters to minimize WMSE for a given bit-rate. This is done in the PCRD-OPT (Post-Compression Rate Distortion Optimization) algorithm where the WMSE replaces the MSE as the cost function which drives the formation of quality layers [17].

CSF weights can be obtained also by applying to each frequency subband the appropriate contrast detection threshold. In [19], subjective experiments were performed to obtain a model that expresses the threshold DWT noise as a function of spatial frequency. Using this model, authors obtain a perceptually lossless quantization matrix for the linear phase 9/7 DWT. By the use of this quantization matrix each subband is quantized by a value that weights the overall resulting quantized image at the threshold of artifacts visibility. For supra-threshold quantization a uniform quantization stage is afterwards performed.

However, we introduce the CSF in the encoder using the ISFW strategy proposed also in [10]. So, from the CSF curve we obtain the weights for scaling the wavelet coefficients. This weighting can be introduced after wavelet filtering stage and before a uniform quantization stage is applied. The weighting is a simple multiplication of the wavelet coefficients in each frequency subband by the corresponding weight. In the decoder, the inverse of this weight is applied. The CSF weights do not need to be explicitly transmitted to de decoder.

This stage is independent to the other encoder modules (wavelet filtering, quantization, etc).

The granularity of the correspondence between frequency and weighting value is a key issue. As wavelet based codecs obtain a multiresolution signal decomposition the easiest association is to find a unique weighting value (or contrast detection threshold) for each wavelet frequency subband. If further decompositions of the frequency domain are done, for example a finer association could be done between frequency and weights by the use of packet wavelets [3].

	LL	LH	HH	HL
L1	1.0	1.1795	1.0000	1.7873
L2	1.0	3.4678	2.4457	4.8524
L3	1.0	6.2038	5.5841	6.4957
L4	1.0	6.4177	6.4964	6.1187
L5	1.0	5.1014	5.5254	4.5678
L6	1.0	3.5546	3.9300	3.1580

Table 1: Proposed CSF Weighting matrix

We perform the ISFW implementation based on [1] but increasing the granularity at the subband level. This is done in the transform stage of the PE-LTW encoder by multiplying each coefficient in a wavelet subband for its corresponding weighting factor. In spite of the fact that the CSF (equation 1) is independent of the viewing distance, in order to introduce it as a scaling factor, the resolution and the viewing distance must be fixed. Although an observer can look at the images from any distance, as stated in [10] the assumption of "worst case viewing conditions" can produce CSF weighting factors that works properly for all diferent viewing distances and media resolutions. So after fixing viewing conditions, we obtain the weighting matrix, see Table 1. For each wavelet level decomposition and orientation the weights are directly obtained from the CSF curve, by normalizing the corresponding values so that the most perceptually important frequencies are scaled with higher values, while the less important are preserved. This scaling process augment the magnitude of all wavelet coefficients, (except for those in LL subband) that are neither scaled nor quantized in our coding algorithm. Our tests reveal that thanks to the weighting process, the uniform quantization stage preserves a very good balance between bit-rate and perceptual quality in all the quantization range, from under-threshold (perceptually lossless) to suprathreshold quantization (lossy).

2.4 GPU 2D-DWT Optimization

To develop the 2D-DWT optimized version we will use the NVIDIA GTX 280 GPU that contains 30 multiprocessors with 8 cores in each multiprocessor, 1 GB of global memory, and 16 KB of shared memory by block (or SM).

Firstly, we will define our GPU-based 2D-DWT algorithm, named as CUDA Conv 9/7, as the reference algorithm. It will only use the GPU shared memory space to store the buffer that will contain a copy of the working row/column data. The constant memory space is used to store the filter taps. We call each CUDA kernel with a one-dimensional number of thread blocks, NBLOCKS, and a one-dimensional number of threads by block, NTHREADS.

In the horizontal DWT filtering process, each image row is stored in the threads shared memory. After that, in the vertical filtering, each column is processed in the same way. The row or column size determines the NBLOCKS parameter, which must be greater or equal to the image width in the horizontal step or the image height in the vertical step. One of the goals in the proposed CUDA-based methods, is not to increase memory requirements, so we will store the resulting wavelet coefficients in the original image memory space.

For computing the DWT, the threads use the shared memory space, where latency access is extremely low. The CUDA-Sep 9/7 algorithm stores the original image in the GPU global memory but computes the filtering steps from the shared memory.

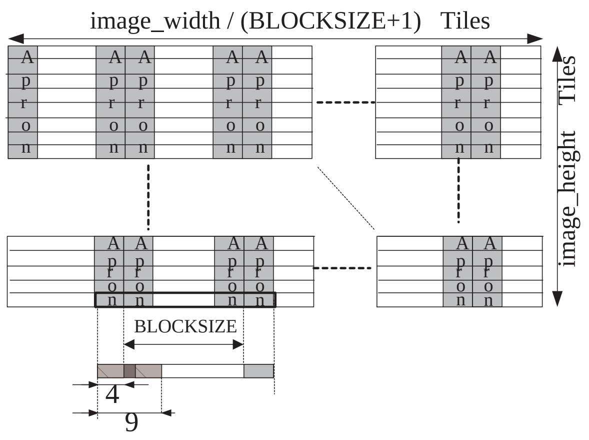
As it can be seen in Figure 2, execution in the GPU is composed by threads grouped in a number of 32 threads called warp. Each warp must load a block of the image from the global memory into a shared memory array with BLOCKSIZE pixels. The number of thread blocks, NBLOCKS, or tiles depends on BLOCKSIZE and image dimensions. Moreover, pixels located in the border of the block, also need neighbor pixels from other blocks to compute the convolution. These regions are called apron and are showed in Figure 2(a) and Figure 2(b). The size of the apron region depends on the filter radius (being the filter radius the half of the filter length minus 1). In both subfigures, the values of the filter radius and the filter length corresponding to the Daubechies 9/7 filter are represented.

We can reduce the number of idle threads by reducing the total number of threads per block and also using each thread to load multiple pixels into shared memory. This ensures that all threads of each warp are active during the computation stage. Note that the number of threads in a block must be a multiple of the warp size (32 threads on GTX 280) for optimal efficiency.

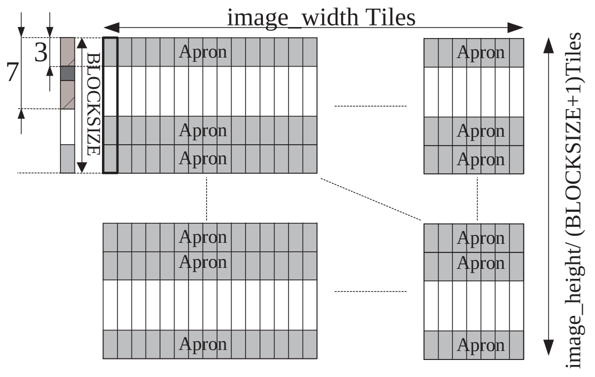
To achieve higher efficiency and higher memory throughput, the GPU attempts to coalesce accesses from multiple threads into a single memory transaction. If all threads within a warp (32 threads) simultaneously read consecutive words then single large read of the 32 values can be performed at optimum speed. In the CUDA-Sep 9/7 algorithm, the convolution process is separated in two stages:

1. The row filtering stage
2. The column filtering stage

Each row/column filtering stage is separated into two sub-stages: (a) the threads load a block of pixels of one row/column from the global memory into the shared memory, and (b) each thread computes the filter over the data stored in the shared memory and stores the result in the global memory. For the



(a) Shared memory for row filter.



(b) Shared memory for column filter.

Fig. 2: Shared Memory for Daubechies 9/7 filter.

column filtering, the resulting coefficient is stored in the global memory after performing the perceptual weighting, i.e. multiplying the final coefficient by the perceptual weight corresponding to the wavelet subband of the coefficient.

In the row or column filtering, the pixels located in the image block borders also needs adjacent pixels from other thread block to compute the DWT. The apron region must also be loaded in the shared memory, but only for reading purposes, because the filtered value of pixels located there are computed by other threads block.

The speed-up achieved by the DWT GPU-based algorithm is up to 20x relative to the sequential implementation in one core. Note that wavelet transform is only a single first step in an image/video encoder and the wavelet coefficients obtained must be processed according to the final application, i.e. arithmetic encoded.

3 Performance Evaluation

All evaluated encoders have been tested on an Intel Pentium Core 2 CPU at 1.8 GHz with 6GB of RAM memory. We use the NVIDIA GTX 280 GPU that contains 30 multiprocessors with 8 cores in each multiprocessor, 1 GB of global memory, and 16 KB of shared memory by block (or SM).

The proposed encoder is compared width Kakadu 5.2.5 and SPIHT (Sphit 8.01) encoders with two set of test images: a) 512x512 image resolution set including Lena, Barbara, Balloon, Horse, Goldhill, Boat, Mandrill and Zelda, and b) 2048x2560 image resolution set including Cafe, Bike and Woman. When comparing with Kadadu, we perform two comparisons, one labeled as Kakadu_csf, which has enabled its perceptual weighting mode (with the perceptual weights presented in [17]), and the other one, labeled as Kakadu, without perceptual weights.

First we analyze the speed-up of the GPU-based 2D-DWT algorithm described in previous section with respect to the traditional convolution algorithm running in a single core processor.

In Table 2 we show for each test image, at different bit-rates, the encoding times for SPIHT, Kakadu and our proposal in milliseconds. The first six columns are related to our proposal: The *SEQ-DWT* column shows the time required by the DWT running on a single core. The *GPU-DWT* column shows the time of the CUDA-Sep 9/7 DWT version when running on GPU. The *Rate&Code* column shows the time required by the rate control and the encoding stage, being this time common for both, the sequential and the GPU 2D-DWT versions. The *T.SEQ* column shows the total time for sequential version and the *T.GPU* the total time for GPU version. Finally the *Speed-up* column shows the speed-up of GPU version compared to the sequential version. The last two columns are the total execution time, also in milliseconds, for the other encoders, SPIHT and Kakadu.

When the target bit-rate is low, i.e. high compression rate, the uniform quantization of the wavelet coefficients produces a great number of non significant coefficients in low decomposition levels, being the root of the zero tree located at higher decomposition levels. This fact reduces the computation cost because only the root of a zero tree needs to be encoded. As a consequence, the overall number of operations is reduced and the gain of GPU optimized version is reduced too.

Table 3 shows the comparison of the average execution times (milliseconds) of each image in the test set at different compression rates. The PE_LTW is faster than SPIHT regardless of the target rate for any image size. However the Kakadu encoder is still faster than the PE_LTW. Although the PE_LTW runs its DWT stage over the GPU, it is the only optimized stage in the whole encoder. By contrast all encoding stages in the Kakadu 5.2.5 are fully opti-

			PE_LTW				SPIHT	Kakadu
bpp	SEQ DWT	GPU DWT	Rate & Coder	T.SEQ	T.GPU	Speed-up	Total	Total
Lena								
1.00	17.08	0.85	31.80	48.88	32.65	1.50	93.04	13.00
0.5	17.23	0.86	16.15	33.38	17.01	1.96	185.74	9.00
0.25	17.17	0.86	10.39	27.56	11.25	2.45	198.64	8.00
0.125	17.57	0.88	7.73	25.30	8.61	2.94	220.15	7.00
Barbara								
1.00	17.89	0.89	27.26	45.15	28.16	1.60	77.80	15.00
0.5	17.42	0.87	17.04	34.46	17.91	1.92	72.37	9.00
0.25	17.45	0.87	11.53	28.98	12.40	2.34	42.59	8.00
0.125	17.49	0.87	8.38	25.87	9.25	2.79	35.04	7.00
Goldhill								
1.00	17.61	0.88	30.62	48.23	31.50	1.53	99.46	12.00
0.5	18.13	0.91	18.21	36.34	19.12	1.90	52.72	24.00
0.25	17.30	0.86	11.51	28.81	12.38	2.33	45.51	8.00
0.125	17.42	0.87	7.97	25.39	8.84	2.87	28.86	7.00
Boat								
1.00	17.02	0.85	27.44	44.46	28.29	1.57	79.05	11.00
0.5	17.35	0.87	17.13	34.49	18.00	1.92	51.22	9.00
0.25	17.03	0.85	11.35	28.37	12.20	2.33	41.98	7.00
0.125	17.13	0.86	7.95	25.07	8.80	2.85	59.12	8.00
Mandrill								
1.00	17.99	0.90	32.85	50.84	33.75	1.51	94.06	19.00
0.5	17.89	0.89	19.98	37.87	20.87	1.81	51.86	11.00
0.25	17.59	0.88	13.11	30.69	13.99	2.19	40.83	8.00
0.125	17.87	0.89	8.59	26.46	9.48	2.79	47.26	8.00
Balloon								
1.00	16.89	0.84	26.86	43.75	27.71	1.58	104.25	12.00
0.5	17.27	0.86	16.39	33.67	17.26	1.95	45.25	9.00
0.25	16.89	0.84	10.92	27.81	11.77	2.36	36.91	8.00
0.125	16.89	0.84	8.06	24.95	8.90	2.80	29.03	7.00
Horse								
1.00	17.60	0.88	31.81	49.42	32.69	1.51	86.45	13.00
0.5	17.34	0.87	18.49	35.83	19.36	1.85	56.35	9.00
0.25	17.33	0.87	11.38	28.71	12.25	2.34	36.74	9.00
0.125	17.55	0.88	8.25	25.80	9.12	2.83	43.10	8.00
Zelda								
1.00	17.11	0.86	35.36	52.48	36.22	1.45	57.56	11.00
0.5	17.08	0.85	16.58	33.65	17.43	1.93	34.68	9.00
0.25	17.39	0.87	10.48	27.87	11.35	2.46	25.36	8.00
0.125	17.25	0.86	7.40	24.65	8.26	2.98	26.44	7.00
Cafe								
1.00	419.10	20.95	521.75	940.85	542.71	1.73	719.54	197.00
0.5	418.50	20.92	325.41	743.91	346.34	2.15	1854.99	129.00
0.25	418.97	20.95	217.20	636.17	238.15	2.67	1104.76	105.00
0.125	418.73	20.94	150.93	569.66	171.86	3.31	733.09	90.00
Bike								
1.00	412.87	20.64	508.61	921.48	529.26	1.74	1265.46	171.00
0.5	413.13	20.66	296.34	709.47	317.00	2.24	1867.98	121.00
0.25	415.15	20.76	191.44	606.59	212.20	2.86	943.82	101.00
0.125	414.18	20.71	134.58	548.76	155.29	3.53	762.22	88.00
Woman								
1.00	414.49	20.72	527.83	942.31	548.55	1.72	819.65	169.00
0.5	414.12	20.71	321.25	735.36	341.95	2.15	1528.94	137.00
0.25	418.81	20.94	215.76	634.57	236.70	2.68	913.84	95.00
0.125	417.78	20.89	151.65	569.43	172.54	3.30	699.80	89.00

Table 2: GPU vs SEQ PE_LTW speed-up and total encoding time comparison with SPIHT and Kakadu

mized. Besides of the use of multi-thread and multi core hardware capabilities, Kakadu uses processor intrinsics capabilities like MMX/SSE/SSE2/SIMD and uses a very fast multicomponent transform, i.e. block transform which is well suited for parallelization.

	PE_LTW Mean times	SPIHT	Kakadu	Speed-up Comparisson	
Rates(bpp)	T.GPU	Total	Total	vs. SPIHT	vs. Kakadu
		512x512			
1	31.4	86.5	13.3	2.76	0.42
0.5	18.4	68.8	11.1	3.74	0.61
0.25	12.2	58.6	8.0	4.80	0.66
0.125	8.9	61.1	7.4	6.86	0.83
		2048x2560			
1	540.2	934.9	179.0	1.73	0.33
0.5	335.1	1750.6	129.0	5.22	0.38
0.25	229.0	987.5	100.3	4.31	0.44
0.125	166.6	731.7	89.0	4.39	0.53

Table 3: Speedup comparison by target bitrate

4 R/D Evaluation

For evaluating image encoders the most common performance metric is the well known R/D, the trade-off between encoder bit-rate (bpp) and the reconstructed quality typically measured in dBs through the PSNR of luminance color plane. However, it is also well-known that the PSNR quality measurement is not close to the human perception of quality and sometimes it gives wrong quality scores, leading to erroneous conclusions when evaluating different encoding strategies.

Figure 3 show the R/D comparison of the Woman (2048x2560) image compressed with the PE_LTW encoder, SPIHT, Kakadu and Kadadu_csf, using PSNR as quality metric. A misleading conclusion after looking at R/D curves for the PE_LTW and Kakadu_csf, is that the encoding strategy of those proposals are inappropriate, since their quality results are always lower than the other encoders, specially at high bit-rates.

There are several studies about the convenience of using other image quality assessment metrics than PSNR that better fit to human perceptual quality assessment (i.e. subjective tests results) [3,8,14,18]. One of the best behaving objective quality assessment metric is VIF [15], which has been proven [3,8] to have a better correlation with subjective perception than other metrics that are usually used for encoders comparisons [14,18]. The VIF metric uses statistic models of natural scenes in conjunction with distortion models in order to quantify the statistical information shared between the test and reference images.

As an example of how measuring perceptual qualitty of images with PSNR is misleading, we show in Figure 4 a subjective comparison of the three en-

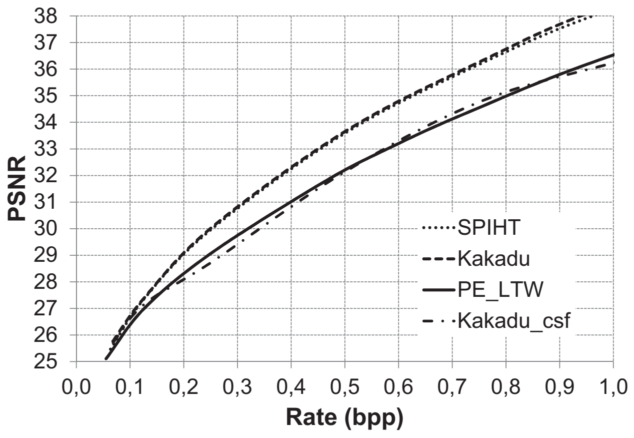


Fig. 3: PSNR R/D comparison of Woman image encoded with PE_LTW, SPIHT and Kakadu. Rates are in bpp.

coders with a cropped region of Woman test image compressed at 0.25 bpp. In this case the third image, encoded with PE_LTW seems to have better subjective quality than the other two. This observation contradicts the conclusion obtained from Figure 3 that suggest that at this rate is the PE_LTW is worse than SPIHT and Kakadu. The same behavior can be observed as well with the other test images. So it is better not to trust in how PNSR ranks quality and use instead a perceptual inspired qualitty assessment metric like VIF that, as stated in [3,8], has much better correlation with humann perception of quality.

So, we will use the VIF metric in our R/D comparisons of the three encoders. Figure 5 shows some of the R/D results for some of the test images. As shown in this plots, the PE_LTW encoder can achieve higher compression rates while maintaining the same perceptual quality than the other encoders, i.e. a bit-rate saving is obtained while using the PE_LTW instead Kakadu or SPIHT for a desired quality.

Table 4 shows the comparison of the rate gain obtained with PE_LTW versus Kakadu, SPIHT and Kakadu_csf, when encoding the image set. The VIF interval varies from 0.1 to 0.95 VIF quality units. This table groups the results by image resolution. Results are expressed as percentage of saved rate in the mentioned VIF interval.

5 Conclusions

We have presented a perceptual image wavelet encoder whose 2D-DWT stage is implemented in CUDA running on a GPU. Our proposed perceptual encoder reveals the importance of exploiting the Contrast Sensitivity Function (CSF) behavior of the HVS by means of an accurate perceptual weighting of the wavelet coefficients. PE_LTW is very competitive in terms of perceptual

P_ELTW	vs. Kakadu	vs. SPIHT	vs. Kakadu_csf
Images	% Rate Saved	% Rate Saved	% Rate Saved
512x512	Mean	Mean	Mean
Lena	13.87%	16.83%	5.23%
Barbara	11.39%	17.44%	-2.61%
Goldhill	7.76%	13.07%	0.09%
Boat	8.58%	12.02%	0.47%
Mandrill	19.13%	22.01%	3.08%
Balloon	10.45%	10.75%	2.16%
Horse	14.96%	14.91%	3.74%
Zelda	17.22%	20.43%	8.46%
Mean 512x512	**12.92%**	**15.93%**	**2.58%**
2048x2560			
Cafe	9.63%	12.34%	1.43%
Bike	9.24%	15.57%	-0.80%
Woman	5.21%	11.46%	3.75%
Mean 2048x2560	**8.03%**	**13.12%**	**1.46%**

Table 4: Rate savings of PE_LTW versus Kakadu, SPIHT and Kakadu with perceptual weights $Kakadu\_csf$.

quality being able to obtain important bit-rate savings regardless the image resolution and at any bit-rate when compared with SPIHT and Kakadu with and widthout its perceptual weighting mode enabled. The PE_LTW is able to produce a quality equivalent image with respect to the other two encoders with a reduced rate.

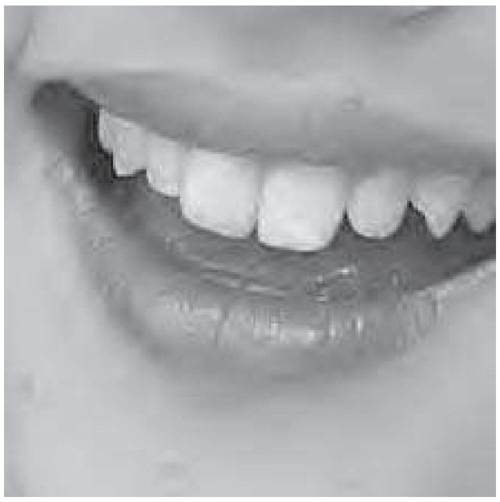
As the 2D-DWT transform runs on a GPU, the overall encoding time is highly reduced compared with the sequential version of the same encoder, obtaining maximum speed-ups of 6.86 for 512x512 images and 4.39 for 2048x2560 images.

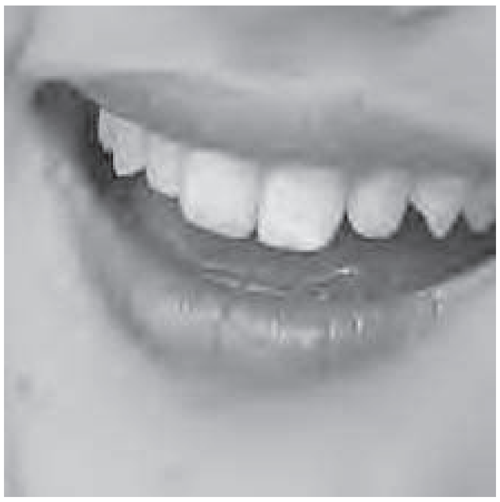
Comparing with SPIHT and Kakadu, our proposal is clearly faster than SPIHT but needs additional optimizations to improve Kakadu times.

References

1. A.P. Beegan, L.R. Iyer, A.E. Bell, V.R. Maher, and M.A. Ross. Design and evaluation of perceptual masks for wavelet image compression. In *Digital Signal Processing Workshop, 2002 and the 2nd Signal Processing Education Workshop. Proceedings of 2002 IEEE 10th*, pages 88 – 93, Oct. 2002.
2. A. Deever and S.S. Hemami. What's your sign?: efficient sign coding for embedded wavelet image coding. In *Data Compression Conference, 2000. Proceedings. DCC 2000*, pages 273 –282, 2000.
3. Xinbo Gao, Wen Lu, Dacheng Tao, and Xuelong Li. Image quality assessment based on multiscale geometric analysis. *IEEE TRANSACTIONS ON IMAGE PROCESSING*, 18(7):1409–1423, 2009.
4. O. Lopez, M. Martinez, P. Pinol, M.P. Malumbres, and J. Oliver. E-ltw: An enhanced ltw encoder with sign coding and precise rate control. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 2821 –2824, nov. 2009.
5. O. López, M. Martinez-Rach, J. Oliver, and M.P. Malumbres. Impact of rate control tools on very fast non-embedded wavelet image encoders. In *Visual Communications and Image Processing 2007*, January 2007.

6. S. G. Mallat. A theory for multi-resolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, July 1989.
7. J. Mannos and D. Sakrison. The effects of a visual fidelity criterion of the encoding of images. *Information Theory, IEEE Transactions on*, 20(4):525 – 536, July 1974.
8. M. Martinez-Rach, O. Lopez, P. Piñol, J. Oliver, and M.P. Malumbres. A study of objective quality assessment metrics for video codec design and evaluation. In *Eight IEEE International Symposium on Multimedia*, volume 1, ISBN 0-7695-2746-9, pages 517–524, San Diego, California, Dec 2006. IEEE Computer Society.
9. ISO/IEC JTC 1/SC 29/WG 1 N1890. Jpeg 2000 image coding system. part 1:core coding system,, September 2000.
10. Marcus J. Nadenau, Julien Reichel, and Murat Kunt. Wavelet-based color image compression: Exploiting the contrast sensitivity function. *IEEE TRANSACTIONS ON IMAGE PROCESSING*, 12(1), 2003.
11. J. Oliver and M.P. Malumbres. Low-complexity multiresolution image compression using wavelet lower trees. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(11):1437–1444, Nov 2006.
12. A. Said and A. Pearlman. A new, fast and efficient image codec based on set partitioning in hierarchicaltrees. *IEEE Transactions on Circuits, Systems and Video Technology*, 6(3):243–250, 1996.
13. J.M. Shapiro. A fast technique for identifying zerotrees in the EZW algorithm. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, 3:1455–1458, May 1996.
14. H. R. Sheikh, M. F. Sabir, and A. C. Bovik. A statistical evaluation of recent full reference image quality assessment algorithms. *IEEE Transactions on Image Processing*, 15(11):3440– 3451, 2006.
15. Hamid Rahim Sheikh, Alan Conrad Bovik, and Gustavo de Veciana. An information fidelity criterion for image quality assessment using natural scene statistics. *IEEE Transactions on Image Processing*, 14(12), 2005.
16. W. Sweldens. The lifting scheme: a custom-design construction of biorthogonal wavelets. *Applied and Computational Harmonic Analysis*, 3(2):186–200, April 1996.
17. David S Taubman and Michael W. Marcellin. *JPEG2000 Image Compression Fundamentals, Standars and Practice*. Number ISBN: 0-7923-7519-X. Springer Science+Business Media, Inc., 2002.
18. Z. Wang, A. Bovik, H. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4), 2004.
19. Andrew B. Watson, Gloria Y. Yang, Joshua A. Solomon, and John Villasenor. Visibility of wavelet quantization noise. *IEEE TRANSACTIONS ON IMAGE PROCESSING*, 6(8):1164–1175, 1997.


(a) SPIHT PSNR=29.95 dB


(b) Kakadu PSNR=30.01 dB

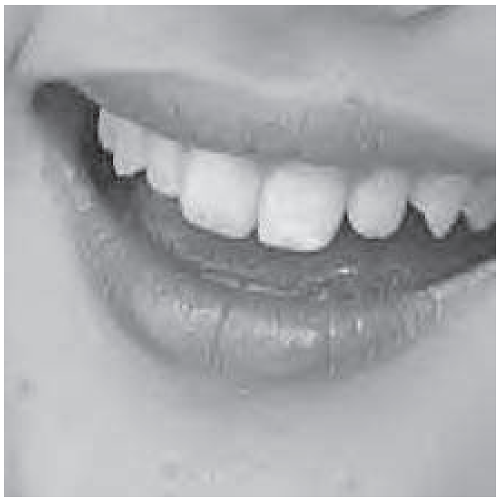
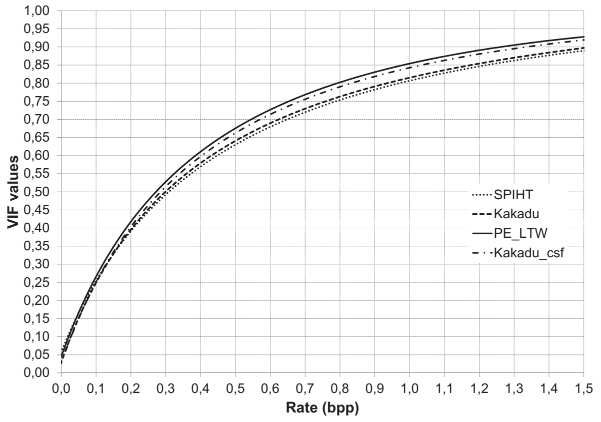
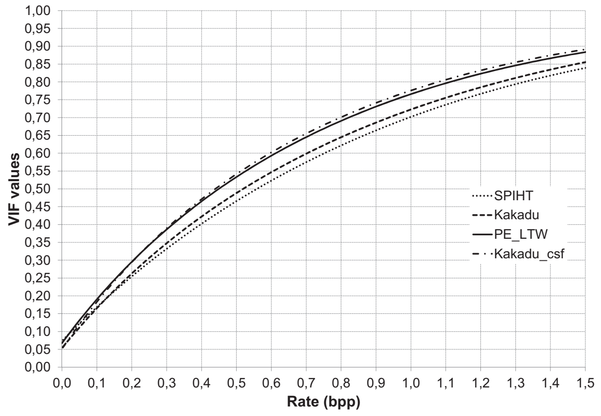

(c) PE_LTW PSNR=29.11 dB

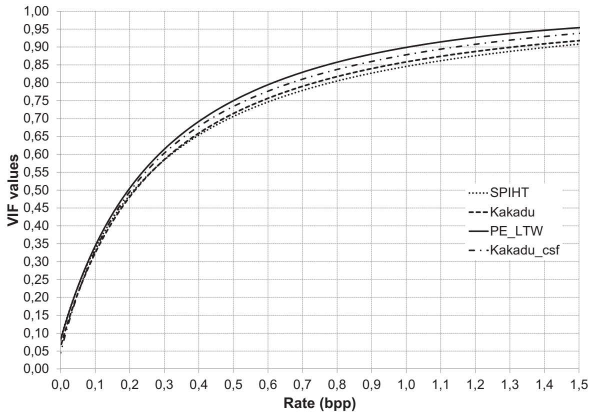
Fig. 4: Subjective comparison of the Woman image encoded at 0.25 bpp with a) SPIHT b) Kakadu and c) PE_LTW



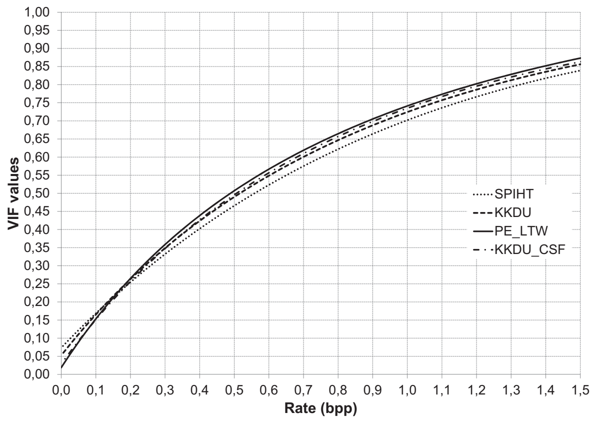
(a) Lena



(b) Barbara



(c) Zelda



(d) Woman

Fig. 5: VIF R/D comparisons for different images from the test set.