ISSN 1392 – 124X INFORMATION TECHNOLOGY AND CONTROL, 2011, Vol. 40, No. 1

SPEEDING-UP IMAGE ENCODING TIMES IN THE SPIHT ALGORITHM

Jonas Valantinas, Deividas Kančelkis

Department of Applied Mathematics, Kaunas University of Technology
Studentų St. 50-325c, LT-51368 Kaunas, Lithuania
e-mail: jonas.valantinas@ktu.lt

crossref http://dx.doi.org/10.5755/j01.itc.40.1.187

Abstract. In this paper, a new approach (scheme) to the analysis of quad-trees in the discrete wavelet spectrum of a digital image is proposed. During the pre-scanning phase, the proposed scheme generates problem-oriented binary codes for the whole set of quad-tree roots (wavelet coefficients) and thereby accumulates information on the significance of respective descendants (wavelet coefficients comprising quad-trees on the view). The developed scheme can be efficiently applied to any zero-tree based image coder, such as the embedded zero-tree wavelet (EZW) algorithm of Shapiro and set partitioning in hierarchical trees (SPIHT) by Said and Pearlman. Fairly impressive performance of the proposed quad-tree analysis scheme, in the sense of image encoding times, is demonstrated using the SPIHT algorithm and the discrete Le Gall wavelet transform.

Keywords: discrete wavelet transforms, quad-trees, zero-tree based image coders, Le Gall wavelets, SPIHT.

1. Introduction

Over the last few decades, the discrete wavelet transform (DWT), as well as wavelets themselves, has gained widespread acceptance in signal processing in general and in image compression in particular [1-4]. In many applications wavelet-based schemes (also known as sub-band coding) outperform other coding schemes like the one (JPEG) based on DCT [5]. Since there is no need to block the input image and its basis functions have variable length, wavelet coding schemes at higher compression rates avoid blocking artefacts (so peculiar to JPEG). Wavelet-based coding provides substantial improvements in image quality at higher compression ratios, is more robust under transmission of images and also facilitates progressive image reconstruction.

Highly useful are image coders that allow progressive encoding with an embedded bit stream, such as the embedded zero-tree wavelet (EZW) image coder, suggested by Shapiro [6]. With embedded bit streams, the wavelet coefficients are encoded in bit planes, with the most significant bit planes being transmitted first. In that way, the decoder can cease decoding at any point in the bit stream, and it will reconstruct an image with required level of accuracy. Different variants of zero-tree based progressive image coders have been developed since Shapiro introduced his algorithm in 1993. The SPIHT (Set Partitioning in Hierarchical Trees) algorithm, proposed by Said and Pearlman, shows excellent results in this class of coders [7]. Some other interesting ideas and innovative proposals in the area are presented in [8-12].

Embedded bit plane encoding is more efficient if one reorders the wavelet coefficient data in such a way that coefficients with small absolute values tend to get clustered together, increasing the lengths of the zero run in the bit planes. Data structures such as insignificant quad-trees (zero-trees) are very efficient in achieving such clustering of zeros. They are used in EZW, SPIHT, and other wavelet-based image coders.

Though there is a number of wavelet-based image coding schemes available, the need for improved performance and wide commercial usage demand newer and better techniques to be developed.

Modest attempts to improve image encoding times in zero-tree based image coding procedures were made by Kunal Mukherjee et al. [13]. Unfortunately, their RMF (Recursive Merge Filter) based EZW algorithm is bound up with Haar wavelets, and is absolutely inapplicable to higher order wavelets (Le Gall, Daubechies, etc.).

In this paper, we propose a novel idea (scheme) for the improved analysis of quad-trees in the discrete wavelet spectrum of the image under processing. The proposed scheme generates finite task-oriented binary codes for all roots (wavelet coefficients) of available quad-trees, estimates the current threshold value and makes a decision over the significance of wavelet coefficients (descendants) comprising quad-trees on the view. The developed scheme, being applied to SPIHT encoder, noticeably improves image encoding times ($(3-11)\%$, for lossless compression, and (5 – 90)%, for lossy compression; Section 4) and, naturally, the overall performance of the encoder. In parallels, impact of the image smoothness level on the efficiency of SPIHT image encoders is touched on.

2. Set partitioning in hierarchical trees (SPIHT) coding

The SPIHT coder is a highly refined version of the EZW algorithm and is a powerful image compression algorithm that produces an embedded bit stream from which the best reconstructed images (in the mean square error sense) can be extracted at various bit rates. Some of the best results, for a wide class of images, have been obtained with SPIHT. Hence, it has become the state-of-the-art algorithm for image compression.

For better understanding of the proposed ideas, we here briefly present the encoding procedure of the conventional SPIHT algorithm [7].

Let $[Y(k_1,k_2)]$ stand for the discrete wavelet spectrum of the digital image under processing $[X(m_1,m_2)]$; $k_1,k_2,m_1,m_2 \in \{0,1,\ldots,N-1\}$, $N=2^n$, $n \in \mathrm{N}$. Also, let $O(k_1,k_2)$ denote the set of indices of all offspring (children) of the wavelet coefficient (node) $Y(k_1,k_2)$, $D(k_1,k_2)$ – the set of indices of all descendants (children, grandchildren, etc.) of $Y(k_1,k_2)$, $L(k_1,k_2)$ – the set of all descendants except the offspring (Figure 1).

The SPIHT encoder explores three control lists, namely: *LIP* – the list of insignificant (with respect to a given threshold *T*) wavelet coefficients (points, nodes), *LSP* – the list of significant (with respect to *T*) wavelet coefficients and *LIS* – the list of insignificant (with respect to *T*) sets (zero-trees). The contents of *LIS* are classified in types *D* and *L* which represent the $D(k_1,k_2)$ and $L(k_1,k_2)$ cases, respectively. By the way, a wavelet coefficient $Y(k_1,k_2)$ is said to be insignificant with respect to a given threshold $T=T_r=2^r$ $(r \in \{0,1,2,\ldots\})$ if $|Y(k_1,k_2)|<T_r$; otherwise ($|Y(k_1,k_2)| \geq T_r$), it is said to be significant.

Now, the image encoding phase is as follows.

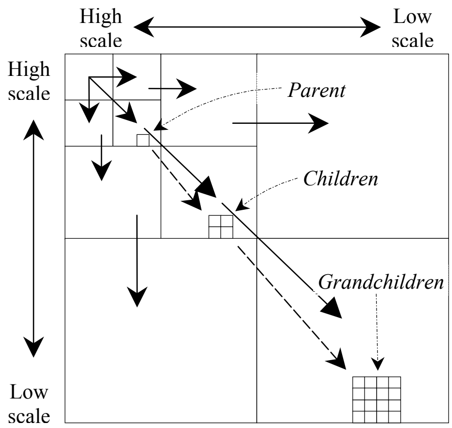
1. Compute $r=r_{\max}=\lfloor \log_2 \max\{|Y(k_1,k_2)|\} \rfloor$; initialize control sets: $LIP=\{(0,0),(0,1),(1,0),(1,1)\}$, $LIS=\{(0,1),(1,0),(1,1)\}$ with contents of type D, and $LSP=\varnothing$.

2. For each $(k_1,k_2) \in LIP$, output $S_r(k_1,k_2)$; here $S_r(k_1,k_2)=1$, if $|Y(k_1,k_2)| \geq T_r$, and $S_r(k_1,k_2)=0$, otherwise. If $S_r(k_1,k_2)=1$, then output the sign of the coefficient $Y(k_1,k_2)$ (0 – negative, 1 - positive).

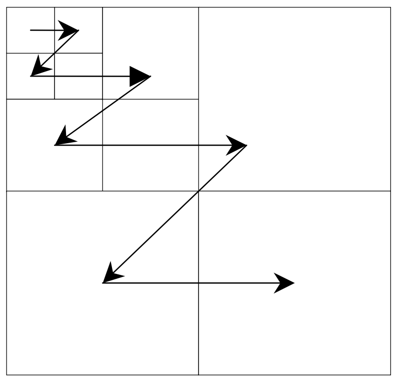
3. For each set, specified by $(k_1,k_2) \in LIS$, do: if type *D*, compute $S_r(D(k_1,k_2))$ ($S_r(D(k_1,k_2))=1$, if $\max\{|Y(k_1^*,k_2^*)| \,|\, (k_1^*,k_2^*) \in D(k_1,k_2)\} \geq 2^r$, otherwise, $S_r(D(k_1,k_2))=0$) and if $S_r(D(k_1,k_2))=1$, output $S_r(k_1^*,k_2^*)$, for each $(k_1^*,k_2^*) \in O(k_1,k_2)$, and if $S_r(k_1^*,k_2^*)=1$, add (k_1^*,k_2^*) to the *LSP* and output sign of the coefficient $Y(k_1^*,k_2^*)$, otherwise, add (k_1^*,k_2^*) to the end of the *LIP* (note: if $S_r(k_1^*,k_2^*)=0$, for all $(k_1^*,k_2^*) \in O(k_1,k_2)$, add (k_1,k_2) to the end of the *LIS* as an entry of type *L*) else if type *L*, compute $S_r(L(k_1,k_2))$ and if $S_r(L(k_1,k_2))=1$, add each $(k_1^*,k_2^*) \in O(k_1,k_2)$ to the end of the *LIS* as an entry of type *D* and remove (k_1,k_2) from the *LIS*.

4. For each (k_1,k_2) in *LSP* (except those just added above), output the *r*-th most significant bit of the coefficient $Y(k_1,k_2)$.

5. If $r>0$, then decrease the value of r by one and go to the step 2.



(a)



(b)

Figure 1. Wavelet coefficients are scanned in a left-to-right order: (a) parent- offspring dependencies in spatial orientation tree (quad-tree structure); (b) the scanning trajectory is such that ancestors are always traversed before descendants

For more detailed description of the algorithm, we refer the reader to [7].

3. Improved quad-tree analysis scheme in the SPIHT algorithm

The most time-consuming operation of the SPIHT algorithm is bound up with finding numerical values of the parameters $S_r(D(k_1,k_2))$ and $S_r(L(k_1,k_2))$, for $(k_1,k_2) \in LIS$ (Step 3; Section 2), because at that time all quad-trees in the discrete wavelet spectrum $[Y(k_1,k_2)]$ of the image $[X(m_1,m_2)]$ are analysed repeatedly for significant nodes (wavelet coefficients) with respect to decreasing threshold values.

To avoid repeated scanning and verification of wavelet coefficients, comprising volumes of quad-trees ($D(k_1,k_2)$ and $L(k_1,k_2)$) in the DWT spectrum $[Y(k_1,k_2)]$, for significance with respect to changing threshold values, we have developed an original quad-tree analysis scheme, which guarantees better performance of the encoding phase in SPIHT.

Let $[Y(k_1,k_2)]$, as before, be the discrete wavelet (Haar, Le Gall, Daubechies, etc.) spectrum of the image $[X(m_1,m_2)]$ ($k_1,k_2,m_1,m_2 \in \{0,1,\ldots,N-1\}$) and $r_{\max} = \left\lfloor \log_2 \max\{\,|Y(k_1,k_2)|\,|\, k_1,k_2 \in \{0,1,\ldots,N-1\}\} \right\rfloor$.

Consider a coefficient $Y(k_1,k_2)$ ($(k_1,k_2) \in LIS$) which is the root (parent) of the quad-tree comprising a particular set of wavelet coefficients (descendants).

Let us associate $Y(k_1,k_2)$ with two binary codes (one for the offspring of $Y(k_1,k_2)$, another for the descendants of $Y(k_1,k_2)$, except offspring) of length ($r_{\max}+1$) each, namely:

$$U(k_1,k_2) = \langle u_{r_{\max}}(k_1,k_2) \ldots u_1(k_1,k_2)\, u_0(k_1,k_2) \rangle ,$$

$$V(k_1,k_2) = \langle v_{r_{\max}}(k_1,k_2) \ldots v_1(k_1,k_2)\, v_0(k_1,k_2) \rangle .$$

The above codes are generated by the one-pass scanning of the discrete wavelet spectrum $[Y(k_1,k_2)]$ as shown below.

1. $u_r(k_1,k_2)=1$, if at least one of coefficients (taken by absolute value) $Y(2k_1, 2k_2)$, $Y(2k_1+1, 2k_2)$, $Y(2k_1, 2k_2+1)$ or $Y(2k_1+1, 2k_2+1)$ falls into the half-open interval $[2^r, 2^{r+1})$, $r \in \{0,1,\ldots,r_{\max}\}$, and $u_r(k_1,k_2)=0$, otherwise;

2. For all $r \in \{0,1,\ldots,r_{\max}\}$, $v_r(k_1,k_2)$ is equal to the logical sum of $u_r(2k_1,2k_2)$, $u_r(2k_1,2k_2+1)$, $u_r(2k_1+1,2k_2)$ and $u_r(2k_1+1,2k_2+1)$, provided $N/8 \le \max\{k_1,k_2\} \le N/4-1$, and $v_r(k_1,k_2)$ is equal to the logical sum of $u_r(2k_1,2k_2)$, $v_r(2k_1,2k_2)$, $u_r(2k_1,2k_2+1)$, $v_r(2k_1,2k_2+1)$, $u_r(2k_1+1,2k_2)$, $v_r(2k_1+1,2k_2)$, $u_r(2k_1+1,2k_2+1)$, $v_r(2k_1+1,2k_2+1)$, provided $1 \le \max\{k_1,k_2\} \le N/8-1$.

We here observe that in the first instance it is absolutely necessary to generate binary codes $V(k_1,k_2)$ with index pairs (k_1,k_2) satisfying the inequality $N/8 \le \max\{k_1,k_2\} \le N/4-1$, then codes $V(k_1,k_2)$ with index pairs (k_1,k_2) satisfying the inequality $N/16 \le \max\{k_1,k_2\} \le N/8-1$, and, finally, codes $V(k_1,k_2)$ with index pairs (k_1,k_2) satisfying the equality $\max\{k_1,k_2\}=1$ (Figure 2).

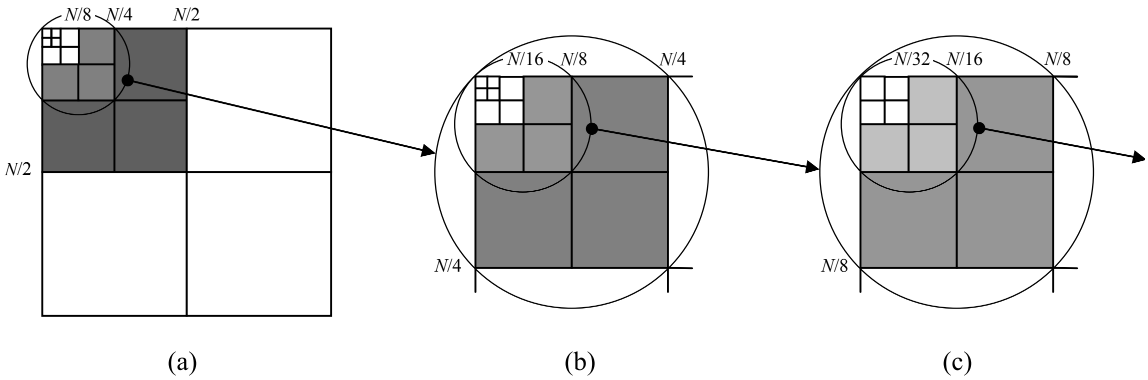


Figure 2. Generation of binary codes $V(k_1,k_2)$ for the descendants of $Y(k_1,k_2)$ ($k_1,k_2 \in \{0,1,\ldots,N/4-1\}$): (a) Binary codes $U(k_1,k_2)$, associated with wavelet coefficients in the dark-grey region, are used to generate binary codes $V(k_1,k_2)$, associated with wavelet coefficients in the lighter-grey region; (b) – (c) Binary codes $U(k_1,k_2)$ and $V(k_1,k_2)$ in the dark-grey regions are used to generate binary codes $V(k_1,k_2)$ in the lighter-grey regions

Thus, to state that the quad-tree, specified by the wavelet coefficient (root, parent) $Y(k_1,k_2)$ ($k_1,k_2 \in \{0,1,\ldots,N/2-1\}$), has no significant wavelet coefficients (descendants) with respect to the threshold $T=T_r=2^r$ ($r \in \{0,1,\ldots,r_{\max}\}$), it suffices to ascertain that $u_r(k_1,k_2)=0$ and $v_r(k_1,k_2)=0$, instead of analysing the whole tree for significance (the key moment of the proposed idea).

It goes without saying that the described quad-tree analysis scheme can be used with other coding algorithms similar to the SPIHT algorithm and for other data as images.

4. Experimental results

To implement both the conventional and the new (supplemented with the proposed quad-tree analysis scheme) versions of the SPIHT algorithm, the discrete Le Gall (wavelet) transform (DLGT) was employed. The latter transform possesses a tolerable “energy compaction” property, has a fast performing technique and facilitates lossless compression of digital images. Incidentally, the default reversible transform in JPEG 2000 is implemented exactly by means of DLGT [3].

On purpose to estimate efficiency of the developed quad-tree analysis scheme, a number of digital images of size 256×256, characterized by different smoothness level, were processed, namely (Figure 3): *Acura.bmp*, *Cameraman.bmp*, *Forest.bmp*. Computer simulation was performed on a PC with CPU: Intel® Core(TM)2 Quad CPU Q8200@ 2.33 GHz, RAM 3 GB, OS System: 32-bit Windows Vista; Programming language: Java.

As it can be seen from Table 1, application of the developed quad-tree analysis scheme to lossless encoding (the threshold value $T = T_0 = 1$), as well as to lossy encoding ($T = T_r > 1$), of test images leads to noticeable image encoding time gains $\tau_{SPIHT} - \tau^*_{SPIHT}$ (τ_{SPIHT} denotes the time needed to encode a particular test image using the conventional SPIHT algorithm, whereas τ^*_{SPIHT} - the time needed to encode the same image using the new proposed version of the SPIHT algorithm).

For instance, in the case of lossless image encoding ($T = T_0 = 1$), the new version of the SPIHT algorithm performs (1.06–1.12) times better than the conventional SPIHT algorithm. Furthermore, with increasing values of the threshold T, the obtainable image encoding speed gains $\omega_\tau = (\tau_{SPIHT} - \tau^*_{SPIHT}) / \tau_{SPIHT} \cdot 100$ (%) have tendency to increase (Figure 4).

Also, we here observe that the overall performance of both the conventional SPIHT algorithm and the modified SPIHT algorithm depends on the smoothness level (class) of the image under processing [14]. The lower smoothness of the image, the longer image encoding times (for both algorithms), and slightly less visible (in terms of ω_τ; lossy encoding) advantage of the modified SPIHT algorithm over the conventional one (Figure 5).

Finally, we notice that the smoothness level α of test images was determined by computing the rate of “decay” of respective DCT coefficients. In particular, $\alpha = 1.12$, for the test image *Acura.bmp*, $\alpha = 0.77$, for *Cameraman.bmp*, and $\alpha = 0.40$, for *Forest.bmp*.

(a)

(b)

(c)

Figure 3. Test images 256x256: (a) *Acura.bmp*; (b) *Cameraman.bmp*; (c) *Forest.bmp*

5. Conclusion

In the paper, a novel scheme for the accelerated analysis of quad-trees in the discrete wavelet spectrum of a digital image is proposed. The proposed scheme generates problem-oriented binary codes for the entire set of quad-tree roots (wavelet coefficients) and accumulates information on the significance of respective descendants (wavelet coefficients comprising quad-trees on the view).

The developed scheme can be successfully applied to any zero-tree based image encoder, such as EZW algorithm, SPIHT algorithm, EBCOT (Embedded

Block Coding with Optimal Truncation [3]) and others.

Table 1. Image encoding times τ_{SPIHT} and τ^*_{SPIHT} (sec).

Test image / Threshold	*Acura*		*Cameraman*		*Forest*	
	τ_{SPIHT}	τ^*_{SPIHT}	τ_{SPIHT}	τ^*_{SPIHT}	τ_{SPIHT}	τ^*_{SPIHT}
$T = T_0 = 1$	9.513	8.481	10.904	10.626	44.370	41.732
$T = T_1 = 2$	2.632	2.372	7.580	7.209	35.009	33.028
$T = T_2 = 4$	1.017	0.597	3.121	3.070	25.420	24.206
$T = T_3 = 8$	0.698	0.294	1.429	1.269	19.820	17.937
$T = T_4 = 16$	0.503	0.121	0.691	0.544	15.297	13.830
$T = T_5 = 32$	0.391	0.034	0.283	0.155	10.061	9.377

Numerous experimental results show that implementation of the proposed quad-tree analysis scheme in the SPIHT algorithm noticeably improves image encoding times ($(3-11)\%$, for lossless image compression, and $(5-90)\%$, for lossy compression) and, naturally, the overall performance of the encoder.

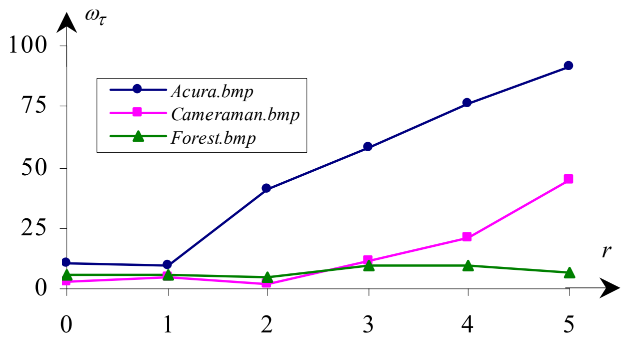


Figure 4. Image encoding speed gains (lossless and lossy image compression)

Also, the image encoding speed gains directly depend on the smoothness class of the image under processing. The lower smoothness of the image, the longer image encoding times (for both versions of the SPIHT algorithm) and merits of the new SPIHT algorithm stop manifesting so visibly in comparison with the conventional SPIHT algorithm.

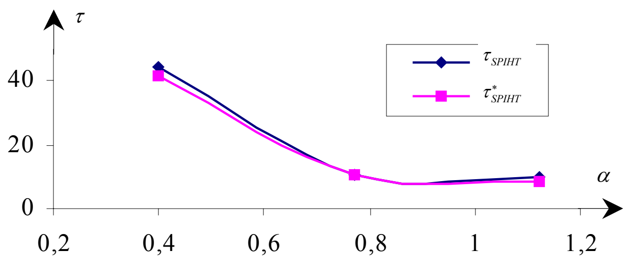


Figure 5. Impact of image smoothness on the efficiency of SPIHT encoders

References

[1] **I. Daubechies.** Ten lectures on wavelets. *SIAM, Philadelphia*, 1992.

[2] **M. Antonini, M. Barlaud, P. Mathieu, I. Daubechies.** Image coding using wavelet transform. *IEEE Trans. on Image Processing*, *Vol.*1, 1992, 205-220.

[3] **C. Christopoulos, A. Skodras, T. Ebrahimi.** The JPEG2000 Still Image Coding System, An Overview. *IEEE Trans. on Consumer Electronics*, *Vol.* 46, *No.* 4, 2000, 1103-1127.

[4] **M. Vettterli.** Wavelets, approximation and compression. *IEEE Signal Processing Mag.*, *Vol.*18, 2001, 59-73.

[5] **G.K. Wallace.** The JPEG Still Picture Compression Standard. *Communications of ACM*, *Vol.*34, *No.*4, 1991, 30-44.

[6] **J.M. Shapiro.** Embedded image coding using zerotrees of wavelet coefficients*. IEEE Trans. Signal Processing*, *Vol.*41, 1993, 3445-3462.

[7] **A. Said, W.A. Pearlman.** A new, fast and efficient image codec based on set partitioning in hierarchical trees. *IEEE Trans. Circuits Syst. Video Technology*, *Vol.*6, 1996, 243-250.

[8] **R.S.H. Istepanian, A.A. Petrosian.** Optimal Zonal Wavelet-Based ECG Data Compression for Mobile Telecardiology System. *IEEE Trans. Inform. Technol. Biomed.*, *Vol.* 4, 2000, 200-211.

[9] **R.M. Rao, A.S. Bopadikar.** Wavelet Transforms: Introduction to Theory and Applications. *Pearson Education, Asia*, 2002.

[10] **H.F. Ates, M.T. Orchard.** An Adaptive Edge Model in the Wavelet Domain for Wavelet Image Coding. *Signal Processing: Image Communications*, *Vol.*20, *Issue* 2, 2005, 169-185.

[11] **L. Kaur, R.C. Chauhan, S.C. Saxena.** Wavelet based compression of medical ultrasound images using vector quantization. *Journal of Medical Engineering &Technology*, *Vol.* 30, *Issue* 3, 2006, 128-133.

[12] **J. Valantinas, D. Kančelkis.** A new wavelet-based approach to progressive encoding of regions of interest in a digital signal. *Information Technology and Control*, *Vol.*38, *No.*3, 2009, 195-202.

[13] **P.S. Jamkhandi, A. Mukherjee, K. Mukherjee, R. Franceschini.** Parallel hardware-software architecture for computation of DWT using the recursive merge filtering algorithm. *Proc. of* 14^{th} *Intern. Parallel and Distributed Processing Symp.* (IPDPS 2000), *Cancun, Mexico, May* 1-5, 2000.

[14] **T. Žumbakis, J. Valantinas.** The use of image smoothness estimates in speeding up fractal image compression. *Proc. of* 14*th Scandinavian Conf. on Image Analysis* (SCIA 2005), *Joensuu, Finland, June* 19-22, 2005, LNCS 3540, 1167-1176.

Received June 2010.