Zerotree and baseline approach for image segment compression

Radoslav Vargic, Jan Procháska
Faculty of Electrical Engineering and Information Technology
Slovak Technical University in Bratislava
Ilkovičova 3, 812 19 Bratislava, Slovakia
vargic@ktl.elf.stuba.sk

Abstract

In this article we compare the efficiency of zero tree and basic baseline coding approach for wavelet based compression of segmented images. We present modification to SPIHT algorithm to improve it's coding efficiency for image segments. To achieve it, we introduce a new, energy based wavelet transformation. In addition a general simple enhancement for SPIHT is proposed.

1. Introduction

For region and object oriented methods there are needed algorithms that can deal with arbitrarily shaped image regions, called segments. In this article we approximate the texture in segments using discrete wavelet transform (DWT) [4] defined on the rectangle circumscribing the given image segment. To partition the image into segments (see Fig. 1) we used an unsupervised segmentation method for colour – texture regions used in [1]. Each segment is coded separately and with bit budget proportional to segment size. In the first approach we use modified SPIHT [3] algorithm as example of zero tree method. We compare the results with baseline approach [4]. Whole coding process is performed in well known three steps transform coding schema depicted in Fig.2.

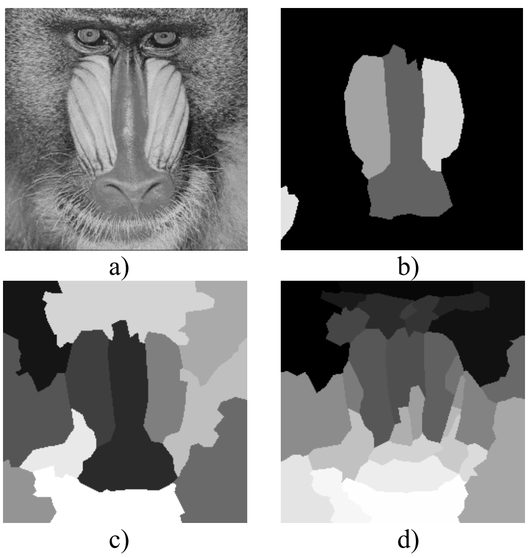

Fig. 1: Original Baboon image a) and masks for b) 5 c) 12 d) 30 segments

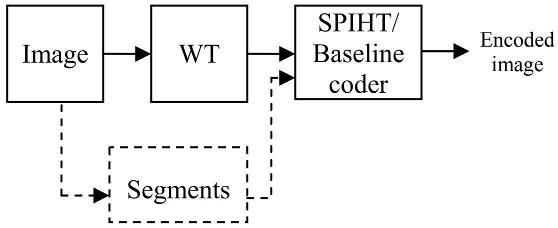

Fig.2. Used transform coding scheme

2. SPIHT algorithm

The SPIHT algorithm [3] is zerotree-based method, that partially orders the spectral coefficients and is sending the most important information first. It uses the spatial dependencies in the DWT spectrum; see Fig.3, where insignificant coefficients are often clustered together at different scales. The spectrum is partitioned into two types of sets and single spectral coefficients $c_{i,j}$. The first type is a *D(i,j)* set which contains all descendants of coefficient *c(i,j)*. The second type is a *L(i,j)* set which is defined as D(i,j)-*O(i,j)* where *O(i,j)* represents offspring of the coefficient *(i,j)*. The set meanings are on depicted in Fig.4.

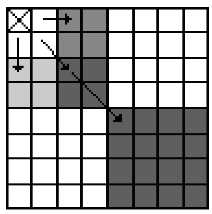

Fig.3: Spatial dependencies between spectrum coefficients in DWT

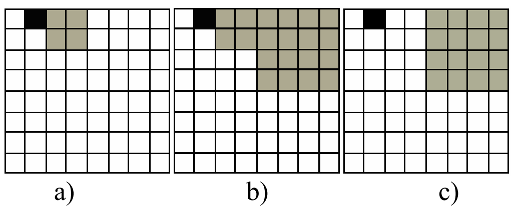

Fig.3: Example of sets (gray) for [i=0,j=1] (black) : a) O(i,j) , b) D(i,j) , c) L(i,j)

In the coding progress the algorithm tests these sets for significance (i.e. if they contain a significant coefficient at a certain magnitude level). The comparison results are coded as single bits (0 – coefficient/set insignificant otherwise 1).

Three lists are required to store all necessary information. *LIP* – list of insignificant pixel, contains all pixels (spectral coefficients) that are not significant yet, *LIS* – list of insignificant sets contains insignificant D-sets and L-sets. *LSP* list of significant pixels contains pixels that became significant in the magnitude (significance) test. In the initialization is *LSP* empty, *LIP* contains coefficients (0,0); (0,1); (1,0); (1,1) and *LIS* contains D(0,1); D(1,0); D(1,1). The coding is simple. In every iteration are insignificant coefficients and sets tested for significance. In the first iteration are magnitudes compared against *threshold* 2^n where $n = \lfloor \log_2(\max_{i,j}\{|c_{i,j}|\}) \rfloor$. This threshold is divided by two after every iteration (i.e. n is decremented). The algorithm's iteration has two main steps: the *Sorting pass* and the *Refinement pass*.

In the *Sorting pass* are tested coefficients in LIP and sets in LIS. If a LIP coefficient becomes significant its sign is coded and moved to LSP. If a set becomes significant it is partitioned depending on its type. If it is a *D(i,j)*-set than *O(i,j)* are tested for significance (if the test is positive their signs are coded and the coefficients are moved to *LSP* (otherwise they are moved to *LIP*). The *D(i,j)* is then marked as *L(i,j)* (if nonzero) and is appended on the end of *LIS* (it will be tested during the same iteration, the same *sorting pass*). If a *L(i,j)*-set is significant then its four *O(i,j)* are appended to *LIS* as *D*-sets (they will be tested during the same iteration).

In the *Refinement pass* is for each entry of *LSP* (except those included in the last sorting pass i.e. with the same threshold) coded its *n*-th most significant bit. After *Refinement* pass decrements the coder *n* and skips to the *Sorting pass*.

Note that after every iteration are all coefficients in the spectrum greater or equal than threshold "identified" and stored in *LSP*. The decoder functions similarly. It inputs the coded significance bits and in fact it duplicates coder's path. The incoming sign bits and most significance bits uses to update the spectrum (where $c_{i,j}$=0) and "builds it from the scratch". The detailed description of SPIHT can be found in [3]

2.1. Modification in SPIHT for image segments

To use SPIHT for segmented images one must modify the algorithm to use it with segmented region. In the case we have a rectangular area that encloses the image segment we must ensure that the coefficient outside the segment will be not evaluated (in significance comparisons) so redundant information will be not coded.

To ensure this, each time a coefficient is being tested it is also tested if it lies within the segment. The second modification affects also the testing of *L* and *D*-sets. Some created sets could be empty (all their descendants lie outside the segment) or some may not contain all descendants (some of them lie within the segment and some outside it).

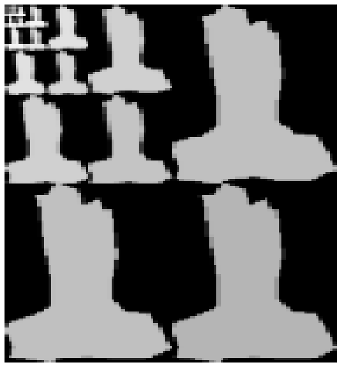

Fig.4: Spectral Segment mask for Baboon's nose segment used in modified spiht for coefficient validity test

To enable the fast validity test, we have to produce spectral segment validity mask in the form shown in Fig.4. Simple, but not accurate approach for produce this mask was used in [2]. To achieve more accurate representation we propose simple modificatioin to DWT to produce validity image itself. We take image of segment where the pixel value in segment is equal one and outside of segment is zero. We do forward DWT but instead of convolving with impulse responses of analysis filters we convolve with their quadratic walues. This results in non-zero walue in wavelet spectrum, where would be valid spectral coefficient in normal case. This approach is universal, fast and simple.

2.2. Generall enhancements in SPIHT

In the classical implementation if a D(i,j) is at a certain level significant, it is partitioned into O(i,j) and L(i,j). The four offspring are tested for significance. The significant offspring are appended to LSP the insignificant ones to LIP. The created set L(i,j) is tested for significance later. In one case can be the efficiency of the algorithm improved.

If all the offspring are insignificant then we already know that the set L(i,j) is significant and we don't need to perform a significance test later. If we use this property, than we can spare one bit, that is coded as the as the comparison result, every time in this case. The significance of L(i,j) does not depend in order in which we test O(i,j) for significance. Using this enhancement we were able to achieve MSE around 1-3 lower than in normal SPIHT implementation.

3. Baseline approach

Another approach to code the wavelet spectrum is to quantize it and use entropic and run-length coders to code the resulting symbol set. To exploit the shape of the wavelet spectra one can read out the coefficients according to their orientation in subband hierarchy as shown in Fig. 5. Such coders are called „baseline coder“ (BC) [4] because of their similaity to baseline JPEG coders. We adapted the implementation available from http://www.geoffdavis.net/dartmouth/ /wavelet/wavelet.html to work with segmented images.

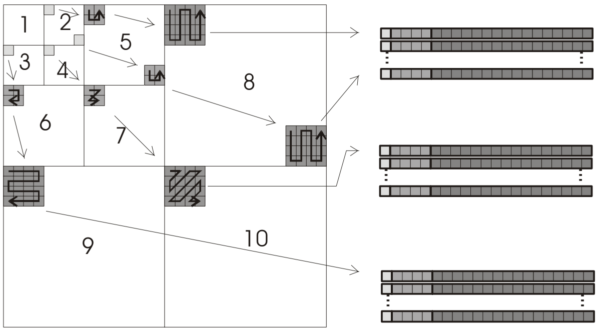

Fig. 5. Wavelet baseline coder - treatment with wavelet coefficients in different subbands

4. Comparison and Results

We compared the performance of the modified SPIHT algorithm and original wavelet baseline coder. Both were adapted to appropriate work with image segments. There was no interest to compare the absolute coding efficiency of both algorithms. We wanted mainly to test the behavior of these two approaches as the number of segments grows and/or bit rate decreases.

Segments	Bpp/ Method	0.15	0.25	0.5	1
5	SPIHT	412.2	347.4	249.8	141.5
	BC	434.7	3566	262.2	148.8
12	SPIHT	429.6	371.1	278.6	171.6
	BC	532.9	418.1	301.3	183.4
30	SPIHT	-	-	294.1	183.8
	BC	-	-	351.1	211.5

Tab. 1. Resulting MSE values for both methods when coding Baboon image using various segment numbers and bit rates. Used DWT with FBI 9/7 filters

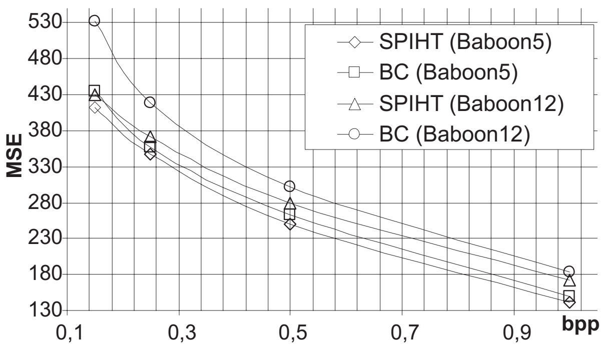

Fig. 6. Coding of segmented baboon Image, both SPITH and BC have approximately the same behavior when number of segments is increased or bit rate is changed

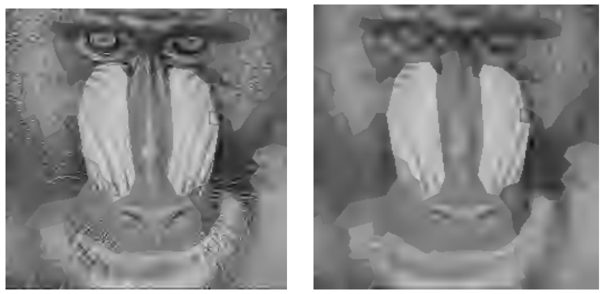

Fig.7: Baboon (12 segments) encoded at rate 0.15 bpp with SPIHT (left) and Baseline coder (right).

Both approaches have bit allocation problems, when the size of come segments is too small. The test showed (see Tab. 1 and Fig. 6), that both approaches have approximately the same behavior when number of segments is increased or bit rate is changed. This could be avoided by more sophisticated bit allocation algorithm.

5. Conclusion

In this article we modified the SPIHT algorithm to code more effectively the image segments. Effect of general enhancement (part 2.2) was 1-3 MSE units. The effect of SPIHT masking (part 2.2) was about 1% of MSE and mostly in higher bit rates (0.5-1). Further, we tested both main approaches, baseline and zerotree, to code the segmented images. The results showed that the efficiency of both approaches with regards of segment count and bit rate remains similar, i.e. there is no clear favorite for compression of segmented images.

6. References

[1] T. Karlubíková et al., New Scheme for Region Approximation and Coding with Shape Independent Transform, Proc. Photogrammetric Computer Vision (PCV’02). Graz, Austria, 2002, ISPRS Commision III, Vol. 34, Part 3B, ISSN 1682-1750, pp. 214-217.

[2] Y. Liu, S. Ranganath, X. Zhou, “Wavelet-based Image Segment Representation”, Electronics Letters, Vol. 38, No. 19, pp. 1091-1092, 2002.

[3] A. Said, Pearlman, W., A. 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, pp. 243-250, June 1996

[4] Polec,J.,Karlubíková,T., Vargic, R Oravec,M, Pavlovičová,J.,: Vybrané metódy kompresie dát-Kódovanie obrazov, UK v Bratislave, 2000