Wavelets: Applications to Image Compression-II

Sachin P Nanavati and Prasanta K Panigrahi

We explain here, the wavelet based thresholding procedure, one of the key factors behind the successful application of wavelets in image compression. We then elaborate on quantization and go on to outline the basic ideas underlying Huffman coding, the other important tool for data compression.

Sachin P Nanavati has joined the Scientific and Engineering Computing Group of C-DAC, Pune. He enjoys trekking and bird watching.

Prasanta K Panigrahi is currently with the Quantum Information and Quantum Optics Division at PRL. His current research interests are in the areas of quantum information, solitons and wavelets.

Introduction

In the first part of this article [1], we have described in detail the origin of various kinds of redundancies in still images, which makes them amenable for compression [2,3]. We have also pointed out the advantages of wavelets over the earlier used discrete cosine transform. In this article, we describe the various steps in image compression like thresholding, quantization and entropy encoding. We describe the run-length coding, differential pulse code modulation and the most popular Huffman coding. These procedures take advantage of the different types of redundancies, for achieving compression of images. We start with the operation of thresholding.

Thresholder

Once DWT is performed, the next task is thresholding, which is neglecting certain wavelet coefficients. For doing this one has to decide the *value of a threshold* and *how to apply the same.*

Value of the Threshold

This is an important step which affects the quality of the compressed image. The basic idea is to truncate the insignificant coefficients, since the amount of information contained in them is negligible.

[1] Wavelets: Applications to Image Compression I, *Resonance*, Vol.10, No.2, pp.52-61.

Keywords
JPEG-2000, image compression, discrete wavelet transform (DWT), Donoho thresholding, Huffman coding.

Ideally one should have a uniform recipe for thresholding so that the procedure can be automated.

The question of deciding the value of threshold is a problem in itself. Ideally, one should have a uniform recipe, which would work satisfactorily for a given set of problems, so that the procedure is automated. One such method by Donoho and co-authors [4] gives an asymptotically optimal formula called the *universal threshold* t:

$$t = \sigma\sqrt{(2 \ln N)}. \tag{1}$$

Here, σ = standard deviation of the N wavelet coefficients.

The value of t should be calculated for each level of decomposition and only for the high-pass coefficients. The low-pass coefficients are usually kept untouched so as to facilitate further decomposition.

How to Apply a Threshold

There are two ways in which threshold can be applied.

a) **Hard threshold:** If x is the set of wavelet coefficients, then threshold value t is given by,

$$T(t,x) = \begin{cases} 0 & \text{if } |x| < t \\ x & \text{otherwise,} \end{cases} \tag{2}$$

i.e., all the values of x which are less than threshold t are equated to zero. This condition is shown in *Figure* 1(b).

b) **Soft threshold:** In this case, all the coefficients x lesser than threshold t are mapped to zero. Then t is subtracted from all $x \geq t$. This condition is depicted by the following equation:

Soft threshold gives a better *peak signal to noise ratio* as compared to hard threshold.

$$T(t,x) = \begin{cases} 0 & \text{if } |x| < t \\ sign(x)(|x| - t) & \text{otherwise.} \end{cases} \tag{3}$$

This condition is shown in *Figure* 1(c). Usually, soft threshold gives a better *peak signal to noise ratio* (PSNR) as compared to hard threshold.

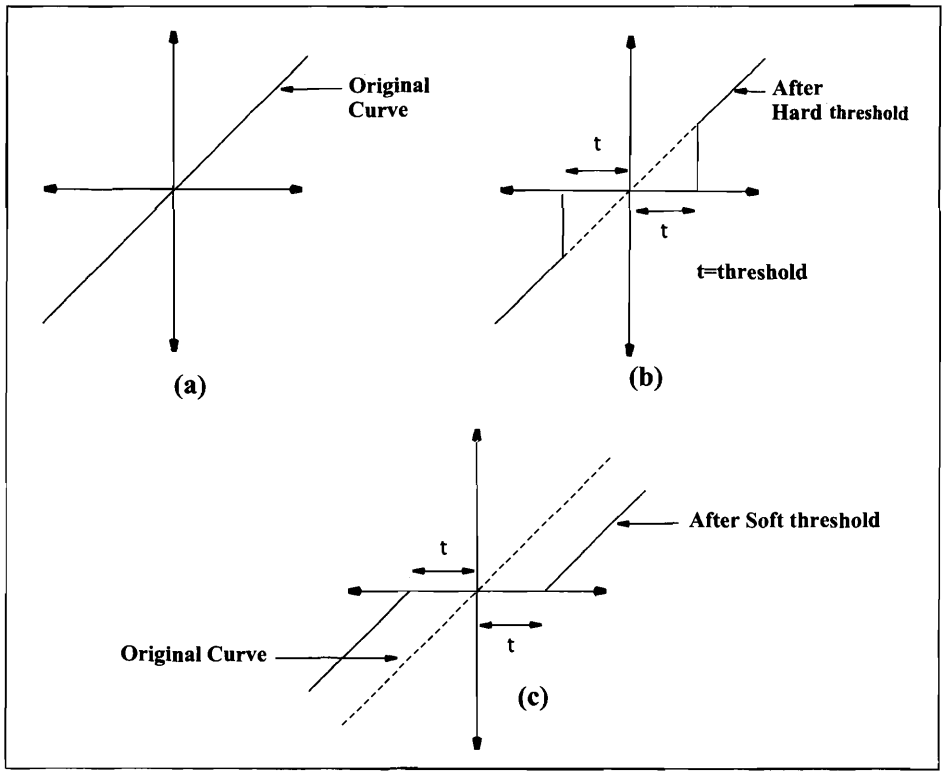

Figure 1. Different thresholds: (a) original (b) hard and (c) soft.

Quantizer

Higher compression ratios can be obtained by *quantizing* the non-zero wavelet coefficients, before they are encoded. A quantizer is a many-to-one function $Q(x)$ that maps many input values into a (usually much) smaller set of output values. Quantizers are staircase functions characterized by a set of numbers $\{d_i,\ i = 0, ..., N\}$ called *decision points* and a set of numbers $\{r_i,\ i = 0, ..., N-1\}$ called *reconstruction levels*. An input value x is mapped to a reconstruction level r_i, if x lies in the interval $(d_i, d_{(i+1)}]$.

To achieve best results, a separate quantizer should be designed for each scale, taking into account statistical properties of the scale's coefficients and, for images, properties of the human visual system. The coefficient statistics guide the quantizer design for each scale, while the

We have devised models for an alternate representation of the image, in which its *interpixel* redundancies were reduced. This last model, which is a *lossless technique*, then aims at eliminating the *coding* redundancies.

human visual system guides the allocation of bits among the different scales. For our present purpose, a simple *uniform* quantizer (i.e., constant step size) is used. The wavelet coefficients (*Figure* 4 on p.26), after thresholding were uniformly quantized into 256 different bins. Thus the size of each bin was $(x_{\max} - x_{\min})/256$, where $x_{\min}$ and $x_{\max}$ are the wavelet coefficients with minimum and maximum values, respectively. To minimize the maximum error (*minimax* condition), centroid of each bin is assigned to all the coefficients falling in that bin. For discussions on non-uniform quantizers, interested readers can refer [3].

Entropy Encoder

This is the last component in the compression model. Till now, we have devised models for an alternate representation of the image, in which its *interpixel* redundancies were reduced. This last model, which is a *lossless technique*, then aims at eliminating the *coding* redundancies, whose notion will be clear by considering an example. Suppose, we have a domain in an image, where pixel values are uniform or the variation in them is uniform. Now one requires 8 bpp (bits per pixel) for representing each pixel since the values range from 0 to 255. Thus representing each pixel with the same (or constant difference) value will introduce coding redundancy. This can be eliminated, if we transform the real values into some *symbolic* form, usually a binary system, where each symbol corresponds to a particular value. We will discuss a few coding techniques and analyse their performances.

Run Length Encoding

Run-length encoding (RLE) makes use of the fact that nearby pixels in an image will probably have the same brightness value. This redundancy can then be coded as follows,

Original image data (8-bit)

127 127 127 127 129 129 129

Run-length encoded image data

127 4 129 2

This technique will be useful for encoding an *online* signal. But data explosion problems can occur and even a single data error will obstruct full decompression.

Predictive image compression techniques assume that a pixel's brightness can be predicted given the value of the preceding pixel.

Differential Pulse Code Modulation

Predictive image compression techniques assume that a pixel's brightness can be predicted given the value of the preceding pixel. Differential pulse code modulation (DPCM) codes the differences between two adjacent pixels. DPCM starts coding at the top left-hand corner of an image and works left to right, until all the image is encoded as shown:

Original Image Data

86 86 86 86 88 89 89 89 89 90 90

86–0–0–0–2–1–0–0–0–1–0

DPCM Code

This technique will be useful for images that have larger runs of equal-value pixels.

Huffman Coding

This is the most popular statistical data compression technique for removing coding redundancy. It assigns the smallest possible number of code symbols per source symbol and hence reduces the average code length used to represent the set of given values. The general idea is to assign least number of bits to most probable (or frequent) values occurring in an image. The Huffman code is an example of a code which is optimal when all symbols have possibilities of occurrence which are

The Huffman code is an example of a code which is optimal when all symbols have possibilities of occurrence which are integral powers of 1/2.

For a given frequency distribution, there are many possible Huffman codes, but the total compressed length will be the same.

integral powers of $\frac{1}{2}$. A Huffman code can be built in the following manner:

- Rank all symbols in decreasing order of probability of occurrence.
- Successively combine the two symbols of the lowest probability to form a new composite symbol (source reduction); eventually we will build a binary tree, where each node is the probability of all nodes beneath it.
- Trace the path to each leaf, noticing the direction at each node.

For a given frequency distribution, there are many possible Huffman codes, but the total compressed length will be the same. It is possible to define a *canonical* Huffman tree, that is, pick one out of many alternative trees. Such a canonical tree can then be represented very compactly, by transmitting only the bit length of each code. For more details, interested readers can refer to [5,6].

Results

We now briefly discuss the results obtained from our analysis. *Figure* 2 is the original Lena image [7]. Lena,

Figure 2. Original Lena image [7].

Figure 3. Image reconstruction after soft(left) and hard (right) threshold.

an *academic model* in the image processing community, has emerged as a benchmark image for testing the efficacy of various algorithms. This gives a chance to researchers across the globe, to compare and contrast their respective methods. The image was subjected to one level DWT, using *Daubechies'-6* wavelet. The coefficients thus obtained, were thresholded using Donoho's formula (equation(1)), for both *hard* and *soft* (equations (2) and (3)) cases. The results are as shown in *Figure* 3. It is worth remembering that in *soft* thresholding, the wavelet coefficients less than or equal to the threshold are equated to zero and the remaining ones are reduced by the same amount, whereas in the *hard* case, the coefficients above the threshold value are not altered. The *PSNR* when calculated using equations (2) and (3) turns out to be approximately 42 dB for *soft* case and 45 dB for *hard* case.

To get an idea about the usefulness of DWT for image compression, let us consider the histograms of the original image and its DWT coefficients (*Figure* 4). The histogram showing the gray scale values (ranging between 0 to 255) of the original image, shows a wide distribution. The histogram after 1-level DWT, reveals that almost 68% of the coefficients are mapped to zero. Thus the

Lena, an *academic model* in the image processing community, has emerged as a benchmark image for testing the efficacy of various algorithms.

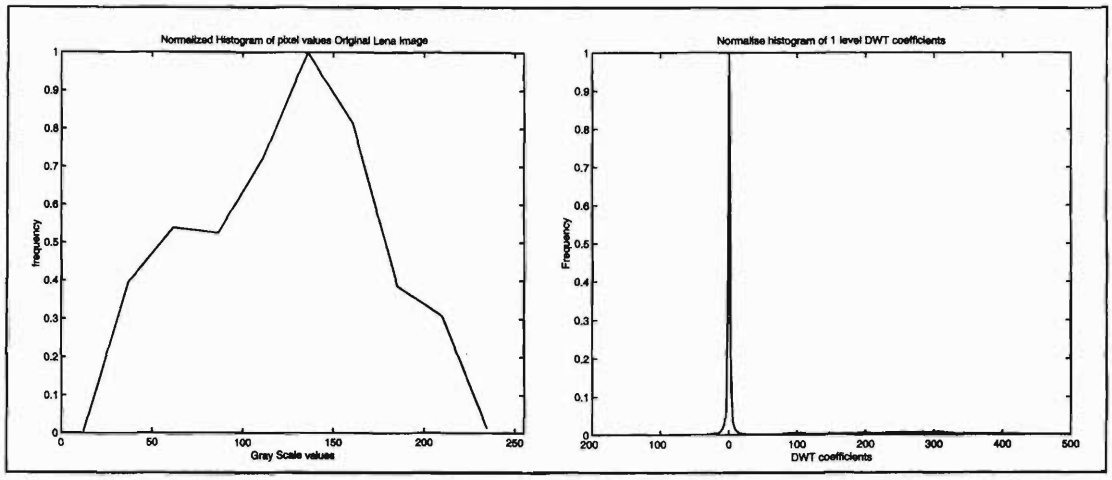

Figure 4. Peak normalized histogram of the grayscale values in the original image (left) and its histogram after 1-level discrete wavelet transform (right).

Figure 5. Reconstructed images after soft threshold + quantization (top) and after hard threshod + quantization (bottom). Note that the quality of the image is better in the one which has used soft thresholding and quantization.

only job now is to code *efficiently*, the remaining 32% of coefficients. These *non-zero* coefficients keep track of the differences or the changes in the image. One can further proceed by quantizing the thresholded coefficients. As explained earlier, wavelet coefficients after thresholding were uniformly quantized into 256 different bins. The value of the centroid of a bin was assigned to all the coefficients falling in that bin. *Figure* 5 shows the image reconstructed, after quantization of soft and hard thresholded coefficients. Here the value of $PSNR$ further degrades to about 31 dB, but the compression ratio increases manyfold.

In conclusion, we have described the application of wavelet transform to image compression, a subject of intense current interest. The properties of images which makes them amenable for compression were pointed out as also the advantage of wavelets over DCT of JPEG-93. The various steps like thresholding, quantization and their usefulness for compression were also pointed out. We dealt with the concept of entropy encoders to bring out the idea of lossless coding transparently.

Suggested Reading

[1] S Nanavati and P Panigrahi, Wavelets: Applications to image compression-I, *Resonance*, Vol.10, No.2, pp.52-61, 2005.

[2] P N Topiwala (Editor), *Wavelet Image and Video Compression*, Kluwer Academic, Norwell, USA, 1998.

[3] M L Hilton, B D Jawerth and A Sengupta, Compressing Still and Moving Images with Wavelets, *Multimedia Systems*, Vol.2, No.3, April 1994.

[4] D L Donoho, De-noising by soft thresholding, *IEEE Trans. Inform. Theory*, Vol. 41, pp. 613-627, 1995.

[5] R C Gonzalez and R E Woods, *Digital Image Processing*, Pearson Education Inc, Delhi, 2003.

[6] D S Taubman and M W Marcellin, *JPEG2000: Image compression fundamentals, standards and practice*, Kluwer Academic, Norwell, USA, 2002.

[7] S Saha, Image compression – from DCT to Wavelets: A Review , *ACM Crossroads Students Magazine* available at http://www.acm.org/crossroads/xrds6-3/sahaimgcoding.html

[8] http://www.jpeg.org} Official site of the Joint Photographic Experts Group (JPEG).

Address for Correspondence
Sachin P Nanavati
National PARAM Supercomputing Facility, Centre for Development of Advanced Computing (C-DAC), Pune University Campus, Ganesh Khind, Pune 411 007, India.
Email: sachinn@cdacindia.com

Prasanta K Panigrahi
Physical Research Laboratory
Navrangpura
Ahmedabad 380 009, India.
Email: prasanta@prl.ernet.in
URL:www.prl.ernet.in/~prasanta