LOW BIT RATE SIGMA FILTERED PERCEPTUAL IMAGE CODING

Tse-Hua Lan, Ahmed H. Tewfik, and Chung-Hui Kuo

Dept. of Electrical and Computer Engineering.
University of Minnesota, tsehua,tewfik,chkuo@ee.umn.edu

ABSTRACT

In this work we address the common problem of low bit rate image coding, i.e. perceptual quality loss. We propose a novel scheme to tackle this problem. The salient part of this work is to first encode the most important visual features of images extracted from Sigma filtering preprocessing. The preliminary results show that the image decoded by our scheme contains much less artifacts comparing to many other low-bit-rate decoded images. Our decoded image is very pleasant to human eyes. Unlike other low bit rate progressive coding methods, in which the low-bit-rate decoded image quality is good only at low resolution, our scheme provides good perceptual quality regardless of resolution size. With the advance of the Internet and multimedia integration, our method provides a promising way to make image browsing faster, and at the same time the image perceptual quality is maintained.

1. INTRODUCTION

Low bit rate image coding is a challenging task. With the advance of the Internet technology, more and more image data have been downloaded every day from the Net. The low bit rate image coding provides an advantage to users before they decided to download a whole image. However, current state of the art low-bit-rate coded images are perceptually pleasant at low resolution only. At normal resolution size, those images suffer from denoising effect 1 (e.g., ringing at edges). In this work we devise a novel scheme to encode images at low bit rate with good perceptual quality regardless of resolution size. The main idea behind this novel method is based on Sigma filter smoothing technique. Sigma filtered images present an interesting characteristic, i.e., small variance regions are smoothed out, but strong edges are enhanced. Perceptual importance coding has been around for many years. In the literature, morphologic [1, 2, 3, 4, 5] based image coding algorithms have been proposed in many ways. The main idea behind those methods is to combine morphological operations in the transformation stage or data encoding stage. They are not targeted for low bit rate coding. How to select perceptual important features of an image automatically is a big research issue. Here we present a prefiltering perceptual importance enhancing method to encode images at low bit rates.

Due to the failure of encoding perceptual important features of many state of the art image coders at low bit rates and the aforementioned Sigma filtering technique, We are motivated to think about using Sigma filter to preprocess an image to enhance perceptual important features. It would transform a natural-scene image to a cartoon-like image. Ideally the cartoon-like image is easier to be encoded at low bit rate and it preserves all important information about how the scene looks like, but smoothes out some low energy detail. Our human eyes are less sensitive to the cartoon-like image than the image with some loss of information.

This paper is organized as follows: 1. An introduction to Sigma filter concept. Since Sigma filtering technique is not well known in the image coding circle, we will present it in detail. 2. Perceptual importance coding. in this section we are to analyze an image processed by Sigma filtering technique, some statistics are presented. Perceptual importance concept and our coding scheme will be presented here as well. 3. Experimental results and conclusion.

2. SIGMA FILTERING

Under our prefiltering perceptual coding concept, it is necessary to process the underlying image into piecewise smooth regions while preserving even enhancing important edges. The traditional low pass filters are not able to accomplish this goal. Fortunately, in image segmentation literature there are several techniques being developed to achieve such goal. Anisotropic diffusion [6], total variation minimization [7, 8], and Sigma filter [9, 10] are among the most efficient approaches. The anisotropic diffusion assigns different diffusion constant to each pixel and start the diffusion process. The total variation minimization method treats this problem as an image restoration problem and imposes the following regularization functional:

$$TV(u) = \int_\Omega |\nabla u|\, dxdy = \int_\Omega \sqrt{u_x^2 + u_y^2}\, dxdy$$

The anisotropic diffusion has the advantage of being suitable for parallel implementation but requires a lot of iterations to converge to a piecewise smooth surface function. The total variation minimization, on the other hand, can achieve a comparable result within a much less iterations, but the computational complexity is very high in each iteration.

To circumvent the potential obstacles with the above two methods, Sigma filter is considered as a good candidate. Next, We will demonstrate that this technique is easy to implement and thus with less computational complexity. For each operation is confined within a predefined window, it is possible to take advantage of the parallel computation technique to further increase its speed as well.

2.1. Original Sigma filter

An image can be treated as a 2-dimensional surface function. To process that surface into piecewise smooth regions while preserving important edges, first one has to identify pixels within the processing window that belong to center pixel class, then apply a low pass filter only within that region. By assuming the pixel distribution within a small window is additive Gaussian noise process, the pixels value in the window can be characterized as a mean μ plus a Gaussian noise term n ($u = \mu + n$). If one applies a threshold Σ to the window, with Σ equal to 2Σ, then the selected pixel value region contains 95% of the distribution of the center pixel class. The original Sigma filter was based on this assumption [11].

2.2. Threshold Estimation and Edge Preserving

We should point out that, if we follow the model $u = \mu + n$ near edges, with maximum likelihood (ML) estimation of the local sample variance within that small window. The estimate variance will usually be much larger than the theoretical variance. Hence, if we choose the Σ in the Sigma filter as 2 times the ML estimate of the local sample standard deviation within the sliding window, the effective range $[-2\sigma, 2\sigma]$ will cover two or more different regions across the edge. As a result, the filter becomes basically a space variant low pass filter, and the edge information will be lost during the process. Therefore, a new thresholding method needs to be developed. C. Kuo and A. Tewfik [10] have devised an efficient threshold estimation algorithm. They estimate the local sample variance only at edge points via the ML estimation. By smoothing and normalizing the histogram of the estimated local sample standard deviation, we can treat the obtained curve as a probability density function $p(x)$. Hence, the calculated value Σ such that $P(\Sigma) = 0.1$, where $P(\Sigma) = \int_0^{\Sigma} p(x)\,dx$ can be chosen as the threshold used in the Sigma filter.

2.3. Filter Design

The design of the Sigma filter is described as follows:

1. Apply the estimated Σ from the above subsection as the threshold.
2. $Iteration = 1 \cdots I$.
3. Slide a window with its size $(2n+1) \times (2m+1)$ through an image with center pixel u_{ij}, n and m could be any integer.
4. Apply one of the following two threshold schemes:
   - Fixed threshold: $\Delta = \Sigma$, which is derived from previous threshold selection procedure.
   - Adaptive threshold: To preserve less significant edges, we can adopt the following equation:

     $$\Delta = \Sigma \times \max[0.2, \frac{1}{1 + \frac{\sigma}{\Sigma}}]$$

     Assuming Σ is the estimated local standard deviation.
5. Apply the following equations to determine class membership:

   $$\delta_{kl} = \begin{cases} w(x) & : \quad (u_{ij} - \Delta) \le u_{kl} \le (u_{ij} + \Delta) \\ 0 & : \quad else \end{cases} \tag{1}$$

   $w(x)$ is a weighting function centered at u_{ij}. One obvious choice is to set $w(x) = 1$.
6. The output pixel value $\tilde{u}_{ij}$

   $$\tilde{u}_{ij} = \frac{\sum_{k=-n}^{k=n} \sum_{l=-m}^{l=m} \delta_{kl} u_{k+i,l+j}}{\sum_{k=-n}^{k=n} \sum_{l=-m}^{l=m} \delta_{kl}} \tag{2}$$

7. Let

   $$\begin{aligned} M &= \sum_{kl} \delta_{kl} \\ K &= min[n+1, m+1] \end{aligned}$$

   If $M < K$, recalculate the mean of four immediate neighboring pixels as the center pixel and go back to step 5.
8. Until the entire image is processed.
9. Relax the threshold by a small number ϵ, i.e., $\Sigma - \epsilon$ and go back to step 2.

Note that the above computational complexity is very similar to the median filter. Fig. 2 presents the Lena image and the Sigma filtered one. One can see the low pass and edge-preserving nature of the Sigma filtered image.

3. PERCEPTUAL IMPORTANCE CODING SCHEME

As mentioned in the introduction that the idea behind prefiltering approach in low bit rate image coding is that the processed image would carry the most perceptually important information for human eyes. Lots of less important details of the image should be removed from the filter. Our Sigma filter has proved the above statement visually after we see fig. 2. However, as far as the image compression is concerned, we still need to find out if the filtered image carries less information. Figure 3 shows histograms of non-processed image and processed one. One can see that the histogram of the filtered image tends to cluster into certain pixel values. The wavelet coefficients of both images are shown in figure 4. One can see clearly that the filtered one contains less coefficients. Figure 5 illustrates more clearly about above observation. For any embedding coding scheme (e.g., SPIHT [12]) or many advanced coding algorithms, the threshold of the absolute wavelet coefficients that corresponds to low bit rate range is from 16 to 32. At this range, the bottom figure of figure 5 shows a reduction of 22% of significant coefficients of the filtered image over the non-filtered one (look at threshold 2^i, where $i = 4, 5$). In SPIHT those reductions means more details (lower thresholded coefficients) could be sent out, so the better reconstruction quality is for the low bit rate imaging.

3.1. Coding scheme

Our proposed idea is simple, let us first Sigma-filter a natural image to produce an edge-enhanced one. Then we use wavelet transform to get a low-low band sub-image. This low-low band sub-image preserves more strong edges than the one without pre-Sigma-filtering. Because of vast area of a Sigma filtered image is smoothed out, and "certain high frequency information is kept at low-low band", now we just need to use wavelet image coding techniques (here we use SPIHT) to encode the low-low band sub-image at a moderate bit rate. It in turn yields low bit rate bit stream. After decoding the encoded image, we might enhance the image again by using Sigma filter, due to the loss of high frequency bands (see fig. 6). Another important point in our coding scheme is that we use a centering mask that weighs heavily at the center of an image, so that the center area gets more refinement. In SPIHT, it means that the root begins at the center of the lowest frequency "low-low" band.

4. EXPERIMENTAL RESULTS AND CONCLUSION

Here we use our proposed method to encode Lena image (512x512) to 0.10 bpp. In fig. 7 we can see that the decoded image before post-filtering presents some aliasing effect at the edges. After Sigma filtering the image (fig. 9), it looks a little bit better for our eyes. Fig. 8 is the SPIHT [12] decoded image at 0.10 bpp. We can clearly see our method looks much better than the SPIHT one. We have conducted a small scale subjective testing, in which 4 experts in image processing area are requested to give their opinion. All participants chose our method for lesser visual artifacts. 3 out of 4 chose the proposed image for better visual presentation. Figure 10 shows the zoom-in version of Lena's right eye. The SPIHT decoded image at

0.1 bpp shows many visual artifacts. Even though, the SPIHT encoded image is better than ours at PSNR comparison (30.0 dB and 28.1 dB at 0.1 bpp). At higher bit rate (0.5 bpp, after decoding the difference image (fig. 11), and add to the Sigma-filtered compressed one), both methods look very close to the original one (fig. 12). The PSNR performance is similar for both of them (37.2 dB and 36.5 dB). We conclude that an image coded by our proposed method presents better perceptual quality at low bit rates, especially, around edges. This gain is due to the fact that Sigma filter can remove many less important details of an image (let us call this phenomenon as "pre-compression"). However, there is a price to pay: additional processing time of Sigma filtering. For many applications (e.g., non real-time transmission, Internet browsing, etc) it is not an issue. Besides, our proposed Sigma filtering algorithm is only 2 to 3 times less faster than a regular Median filter.

Figure 1: Denoising effect, wavelet coefficients greater than 64. Ringing effect is very noticeable

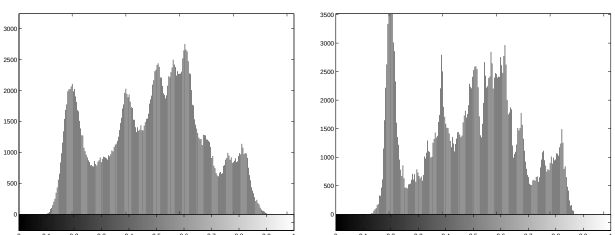

Figure 3: Histogram: left: original Lena, right: Sigma filtered Lena

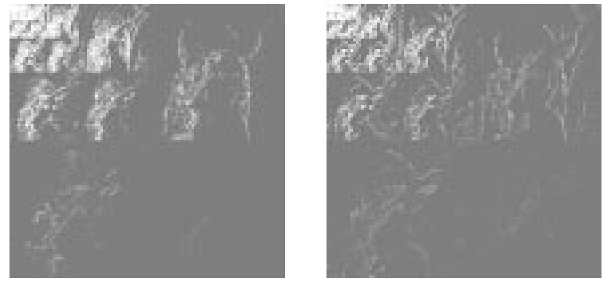

Figure 4: The absolute value of wavelet coefficients greater than 16 (white): left: original Lena, right: Sigma filtered Lena

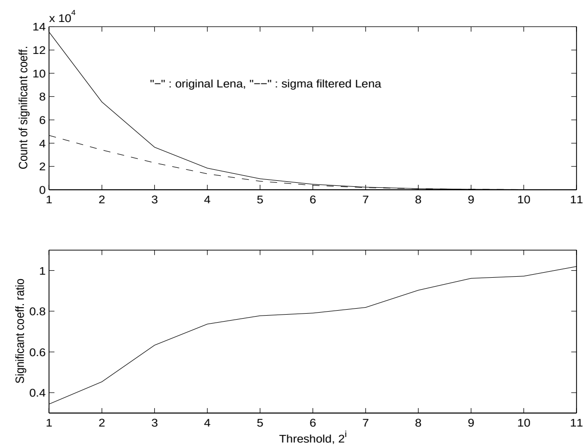

Figure 5: Count of the absolute value of wavelet coefficients and the ratio between two images coeff. versus threshold. Top: Count number. Bottom: Ratio of the number of Sigma filtered coeff. over the original coeff.

5. REFERENCES

[1] X. Ran and Nariman Farvardin, "A Perceptually Motivated Three-Component Image Model - Part II: Applications to Image Compression," *IEEE Transaction on Image Processing*, vol. 4, pp. 430–447, 1995.

[2] L. Overturf, M. Comer, and E. Delp, "Color Image Coding Using Morphological Pyramid Decomposition," *IEEE Transaction on Image Processing*, vol. 4, pp. 177–185, 1999.

[3] P.Salembier, P. Brigger, J. Casas, and M. Pardas, "Morphological Operators for Image and Compression," *IEEE Transaction on Image Processing*, vol. 5, pp. 881–897, 1996.

[4] S. Servetto, K. Ramchandran, and M. Orchard, "Image Coding Based on a Morphological Representation of Wavelet Data," *To appear in the IEEE Transaction on Image Processing*, 1999.

[5] O. Egger, W. Li, and M. Kunt, "High Compression Image Coding Using an Adaptive Morphological Subband Decomposition," *Proceedings of IEEE*, vol. 83, pp. 272–287, 1995.

[6] P. Perona and J. Malik, "Scale-Space and Edge Detection Using Anisotropic Diffusion," *IEEE PAMI*, vol. 12, pp. 629–639, 1990.

[7] P. Blomgren and T. F. Chan and P. Mulet and C. K. Wong, "Total Variation Image Restoration: Numercial methods and Extension," *IEEE Transaction on Image Processing*, vol. 3, pp. 384–387, 1997.

[8] L. I. Rudin and S. Osher and E. Fatemi, "Nonlinear Total Variation based Noise Removal Algorithm," *Physica D*, vol. 60, pp. 259–268, 1992.

[9] J. Lee, "Digital Image Enhancement and the Noise Filtering by Use of Local Statistics," *IEEE PAMI*, vol. 2, pp. 165–168, 1980.

[10] C. Kuo and A. Tewfik, "Multiscale Sigma Filter and Active Contour for Image Segmentation," *Submitted to ICIP99*, 1999.

[11] J. Lee, "Digital Image Smoothing and the Sigma Filter," *Computer Vision, Graphics and Image Processing*, no. 24, pp. 255–269, 1983.

[12] A. Said and W. 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, pp. 243–250, June 1996.

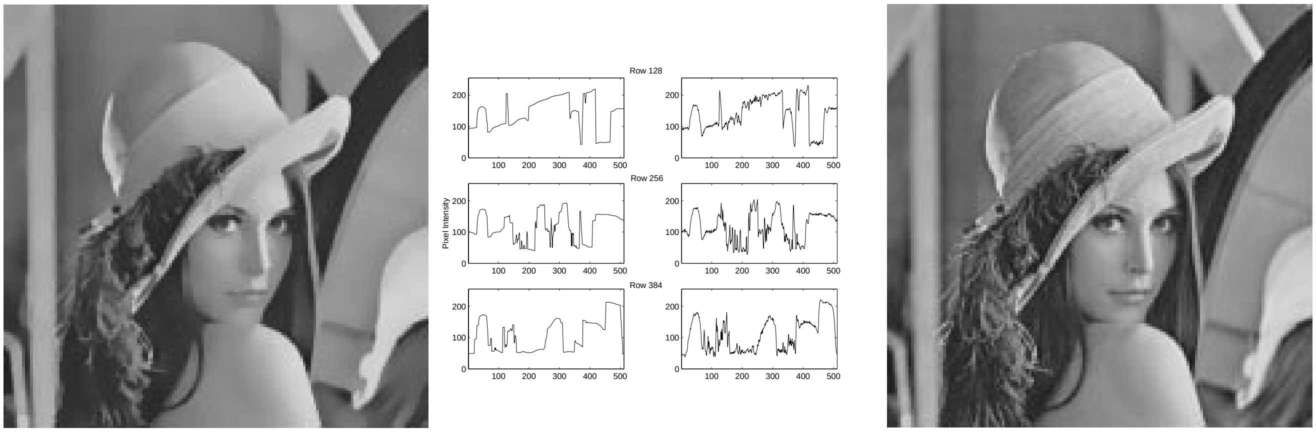

Figure 2: Left: Sigma filtered Lena, middle: Pixel intensity of different rows of Sigma-filtered lena (left) and original lena (right), right: Original Lena 512x512

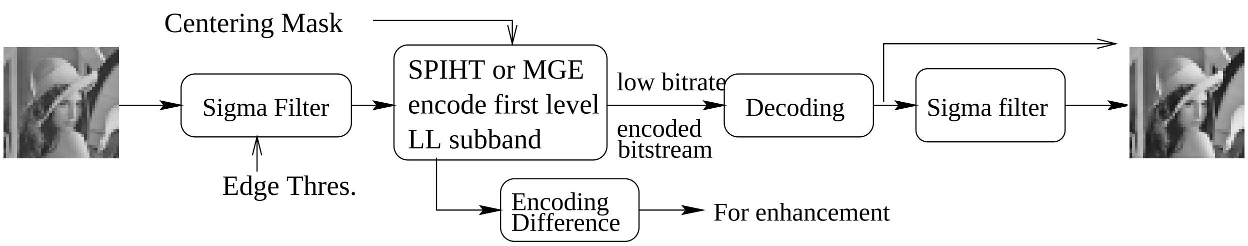

Figure 6: Proposed method

Figure 7: Reconstructed Lena using our proposed method at 0.10 bpp

Figure 8: SPIHT decoded Lena at 0.10 bpp

Figure 11: Difference between original Lena and proposed 0.10 bpp Lena

Figure 12: Lena at 0.5 bpp, left: Proposed method, right: SPIHT

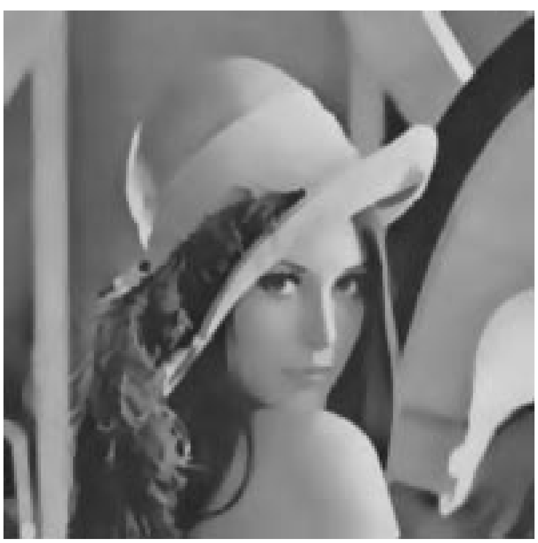

Figure 9: Post filtered image of figure 7

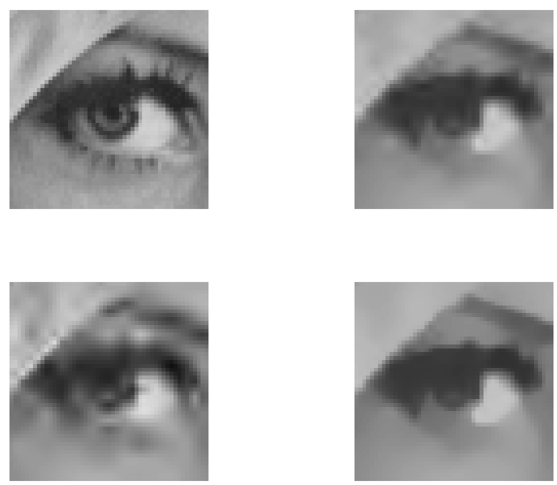

Figure 10: Comparison of eyes at 0.1 bpp. Top left: Original Lena, top right: proposed method, bottom left: SPIHT, bottom right: post-filtering