Dead Zone Quantization in Wavelet Image Compression — Mini Project in ECE 253a —

Jacob Ström

May 12, 1996

Abstract

A common quantizer implementation of wavelet image compression systems is scalar quantization followed by run-length and Huffman coding. The low complexity of such a quantizer system makes it suitable for hardware implementations. By replacing the scalar quantizer with a dead zone quantizer, the performance of the system in terms of PSNR/bpp increases. The purpose of this mini project is to investigate how big the dead zone should be for optimal performance and how much quality can be gained. It is found that for the test image used the optimal size is about 1.9 steps and that the quality increases about 0.5 dB.

1 Introduction

Recently, wavelet encoding of images has emerged as a promising technique for achieving high compression ratios in combination with low distortion. Compression schemes like [1] that are based on zerotree encoding introduced by Shapiro, are among the best known algorithms in terms of PSNR/bpp[1] performance. However, these schemes have a complexity that makes it hard to implement them in hardware for real time performance. A common way to encode the wavelet coefficients with reasonable low complexity is shown i figure 1. After

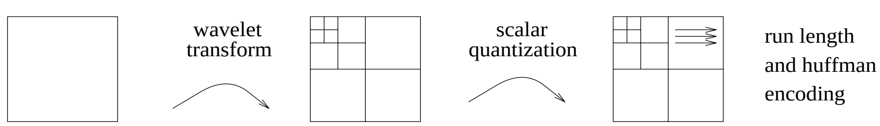

Figure 1: A wavelet encoding system of low complexity: Scalar quantising followed by run length and Huffman encoding.

[1]PSNR=Peak Signal to Noise Ratio, bpp = bits per pixel

the wavelet transform, the subbands are scalar quantized, run length and then Huffman encoded. In a system like this, a huge penalty in bitrate is paid every time a run of zeros is broken. All nonzero coefficients, independent of size, have the same negative impact in the run length encoder. Small coefficients increases the quality of the image the least, which means that you will get very little 'bang for the buck' when encoding them, and they still mess up the run length encoding as much as the big coefficients. Using a deadzone quantizer (depicted in figure 2) instead of a scalar one makes is possible to block the smallest coefficients. Longer runs of zeros are now achieved, and this enhances the performance of the run-length encoder. The performance of the whole system in terms of psnr/bpp goes up with just a slight penalty in complexity. The purpose of this paper is to find the optimal size of the dead zone and then investigate how much is gained in PNSR for a constant bitrate.

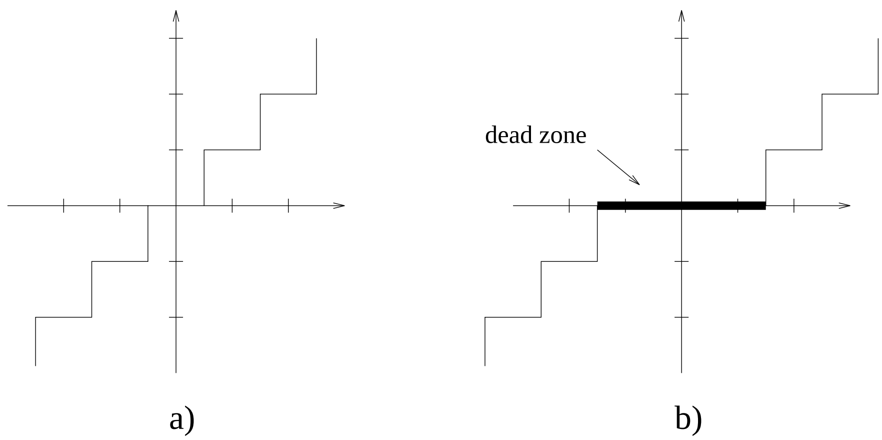

Figure 2: A scalar quantizer a) and a dead zone quantizer with a dead zone of 3 steps b). Note that the scalar quantizer can be thought of as having a 1 step big dead zone.

2 Experiment

An experiment was done with a 512x256 cut from the Lena image. It was compressed with a dead zone quantizer with different sizes δ of the dead zone. The image was decompressed and a PSNR value was calculated by comparing with the original image. The target bitrate was 0.2 bpp (3277 bytes) but since it was impossible to hit that bitrat exactly the PSNR value was an interpolated value of the PSNRs calculated from two images of nearby bitrates, for example 3250 bytes and 3287 bytes. The result is presented in figure 3. In the diagram you can see that the optimal value of δ is 1.9 and that the gain in PSNR is about

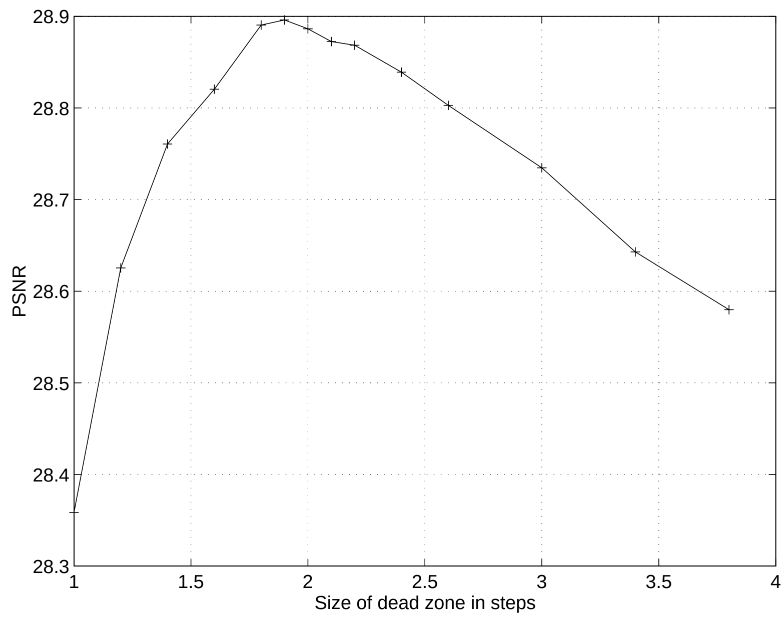

Figure 3: The PSNR as a function of the dead zone size δ. Note that the optimal value of δ is 1.9.

0.5dB for this bitrate. After having found the optimal δ the test image was compressed at various bitrate with $\delta = 1.9$ and $\delta = 1$ (normal scalar quantizer). The PSNR was measured and the results are shown in figure 4.

3 Side Effects

In figure 5, the three images (from top to bottom) show the result of the Lena image being compressed (and decompreesed) with a dead zone of 1, 1.9 and 3 steps respectively. (Remember that a dead zone of 1 step is equivalent with a normal scalar quantizer). The middle image is about 0.5 dB higher in PSNR than the top one, and that can be seen in the image. The mouth region for instance is better preserved in the middle image. When it comes to contrast, however, the topmost image seems to the sharpest. The dead zone quantizer has a low pass effect on the image. This is because the dead zone quantizer gives more attention to big coefficients. Big coefficients are (due to the unitary wavelet transform) in the low pass subbands. This lowpass effect is most easily seen if comparing the top- and bottommost images. This low pass effect need not be a disadvantage. If compressing video, for instance, small artifacts at

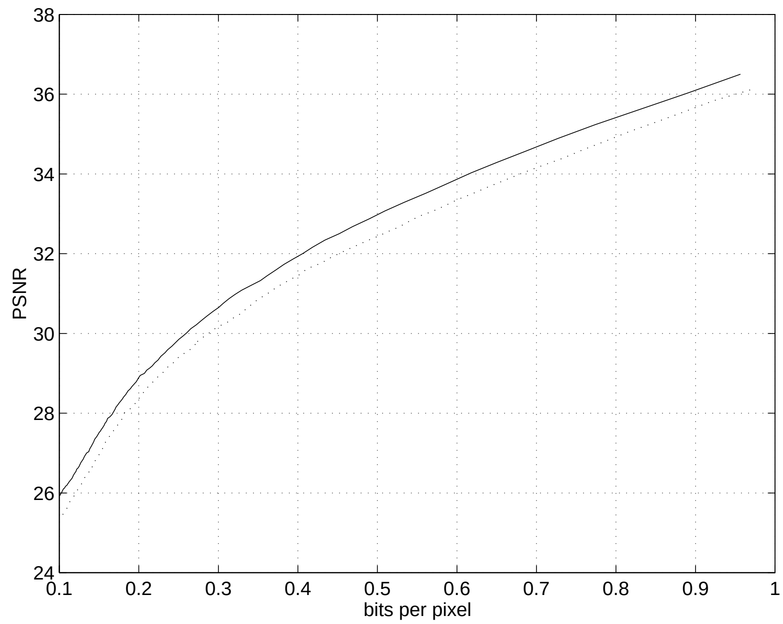

Figure 4: The PSNR as a function of bitrate for the normal scalar quantizer (dotted line) and the dead zone quantizer (solid line) with $\delta = 1.9$. Note that the dead zone quantizer is about 0.5dB better all the way.

ever changing location is very irritating. The low pass effect combined with the higher PSNR ratio might be just perfect in that application. The bottommost image in fiugre 5 is interesting also from a complexity point of view, since it is the easiest one to implement in hardware. In the encoding phase, it is equivalent of doing a normal quantizing and just erase all the -1's and +1's. On the decoder nothing needs to be done, the image format will be binary compatible with the one used for a normal scalar quantizer. Thus there is no complexity penalty for the gain in PSNR.

4 Conclusion

Two experiments were carried out. In the first an optimal size for the dead zone in a compression system with run length coding was found to be 1.9 steps, when using the Lena image and a bitrate of 0.2 bpp. A side effect of the quantizing is a low pass effect on the decompressed image. This need not be disadvantageous for all applications (e.g. video compression). A dead zone of 3 steps is binary compatible with a normal scalar quantizer and requires no extra complexity for decompression.

References

[1] A. Said and W.A. Pearlman, *A New Fast and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees*, IEEE Transactions on Circuits and Systems for Video Technology.

Figure 5: Top:$\delta = 1$. (Normal scalar quantizer). PSNR = 28.40, 3294 bytes. Middle: $\delta = 1.9$, PSNR = 28.88, 3266 bytes. Bottom: $\delta = 3$, PSNR = 28.71, 3260 bytes. Note that although the top one has the lowest PSNR, it has the highest contrast. Also note that the bottommost is the easiest to implement in hardware.